Министерства цифрового развития, связи и массовых коммуникаций
Российской Федерации
Ордена Трудового Красного Знамени федеральное государственное
бюджетное образовательное учреждение высшего образования «Московский
технический университет связи и информатики»
(МТУСИ)

На правах рукописи

Городничев Михаил Геннадьевич

Методология создания информационной системы управления транспортными потоками

Специальность 2.3.8. Информатика и информационные процессы
(технические науки)

Диссертация на соискание учёной степени
доктора технических наук

Научный консультант:
доктор технических наук, доцент
Яшина Марина Викторовна

Москва — 2026

Оглавление

Стр.

Введение 5

Глава 1. Современное состояние интеллектуальных транспортных систем 20

1.1 Архитектурные основы ИТС 23

1.2 Иерархия и уровни управления в ИТС 45

1.3 Подсистемы ИТС 57

1.3.1 Подсистема мониторинга и семантического восприятия . . 57

1.3.2 Подсистема оценки состояния транспортного потока . . . 60

1.3.3 Подсистема управления и оптимизации 62

1.3.4 Подсистема информирования и координации участников . 64

1.3.5 Подсистема безопасности, управления инцидентами и устойчивого функционирования 66

1.4 Методология оценки эффективности внедрения ИТС 69

1.5 Выводы по главе 76

Глава 2. Разработка метода прогнозирования макроскопических характеристик транспортного потока 79

2.1 Анализ современного состояния прогнозирования макроскопических характеристик транспортного потока 82

2.2 Разработка метода PTSTG: PatchTST-Graph for Spatio-Temporal Forecasting 88

2.3 Выводы по главе 106

Глава 3. Разработка метода и средств прогнозирования микроскопических характеристик транспортного потока 108

3.1 Анализ существующих методов прогнозирования микроскопических характеристик транспортного потока 111

3.2 Разработка метода сбора микроскопических данных транспортного потока 130

Стр.

3.3 Разработка информационной системы функционирования модели следования за лидером . . . 144
3.4 Разработка метода динамической коррекции коэффициентов модели следования за лидером . . . 153
3.5 Разработка гибридного метода прогнозирования микроскопических характеристик транспортного потока . . . 164
3.6 Выводы по главе . . . 188

Глава 4. Разработка методологии конфигурирования архитектуры исполнения моделей ИИ в ИТС . . . 192
4.1 Архитектура исполнения моделей ИИ . . . 194
4.1.1 Исполнение моделей с использованием реляционных БД в ИТС . . . 201
4.1.2 Предлагаемая реализация исполнения моделей . . . 205
4.1.3 Исполнение модели с использованием GPU в ИТС . . . 212
4.1.4 Исполнение модели с использованием Hadoop в ИТС . . . 219
4.1.5 Исполнение модели с использованием S3 в ИТС . . . 227
4.1.6 Исполнение модели с использованием In-Memory БД в ИТС 232
4.1.7 Обращение к модели по HTTPS/gRPC в ИТС . . . 238
4.1.8 Обращение к модели через Kafka в ИТС . . . 243
4.2 Разработка методологии построения архитектуры исполнения моделей искусственного интеллекта в ИТС . . . 249
4.3 Выводы по главе . . . 265

Глава 5. Разработка методологии построения инфраструктуры интеллектуальных транспортных систем . . . 269
5.1 Разработка метода децентрализованного взаимодействия подсистем ИТС . . . 269
5.2 Разработка метода кластеризации серверов в ИТС . . . 272
5.3 Разработка метода динамического распределения нагрузки для повышения отказоустойчивости ИТС . . . 298
5.4 Разработка метода генерации синтетических данных транспортного потока для проведения тестирования ИТС . . . 312
5.5 Выводы по главе . . . 327

Стр.

Заключение 329

Список сокращений 331

Список литературы 335

Приложение А. Профили полей и ограничения 365

Приложение Б. Примеры запросов 368

Приложение В. Декларативные утверждения для unit-тестов конфигурации БД 370

Приложение Г. Декларативные утверждения для unit-тестов конфигурации GPU 371

Приложение Д. Декларативные утверждения для автоматических проверок 373

Приложение Е. Декларативные утверждения для unit-тестов конфигурации S3 375

Приложение Ж. Декларативные утверждения для unit-тестов конфигурации In-Memory БД 376

Приложение З. Декларативные утверждения для unit-тестов контура HTTPS/gRPC 377

Приложение И. Декларативные утверждения для unit-тестов конфигурации Kafka 378

Приложение К. Свидетельства о государственной регистрации программ для ЭВМ 379

Приложение Л. Акты о внедрении 394

Введение

Актуальность темы исследования. Развитие городских агломераций, рост уровня автомобилизации, повышение интенсивности перевозок и усложнение структуры транспортных сетей приводят к существенному увеличению требований к эффективности управления дорожным движением. Транспортная инфраструктура функционирует в условиях высокой динамичности, пространственной неоднородности и воздействия множества внешних факторов, включая изменения интенсивности движения, погодные условия, инциденты и особенности поведения участников дорожного движения. В этих условиях традиционные подходы к управлению транспортными потоками, основанные преимущественно на статических схемах регулирования и локальной автоматизации, оказываются недостаточными для обеспечения требуемого уровня пропускной способности, безопасности и устойчивости транспортной системы. Современный этап развития интеллектуальных транспортных систем характеризуется переходом от отдельных автоматизированных решений к интегрированным информационным системам, использующим технологии больших данных, машинного обучения, распределённых вычислений и искусственного интеллекта. Интеллектуальные транспортные системы рассматриваются как сложные социотехнические комплексы, объединяющие средства мониторинга, коммуникационные сети, платформы обработки данных и механизмы автоматизированного управления.

Методы прогнозирования транспортного потока играют важную роль в современных интеллектуальных транспортных системах. На макроуровне они используются для оценки параметров транспортного потока, включая интенсивность движения, скорость и загруженность дорожной сети, тем самым обеспечивая основу для адаптивного управления, навигационных услуг и систем информирования пользователей. На уровне микромоделирования основная задача состоит в описании поведения отдельных транспортных средств и оценке характера их взаимодействия в изменяющейся дорожной обстановке. Ключевой характеристикой транспортных процессов является их нестационарный характер, который формируется динамикой дорожной ситуации, человеческим фактором и изменчивостью статистических свойств наблюдаемых данных.

Несмотря на значительный объем исследований в области интеллектуальных транспортных систем, существующие методы и архитектурные решения имеют ряд ограничений. Пространственно-временные модели прогнозирования часто демонстрируют снижение эффективности при изменении структуры транспортной сети или статистических характеристик входных данных. Традиционные микроскопические модели требуют трудоемкой калибровки и обладают ограниченной адаптивностью к изменяющимся условиям движения, тогда как модели, основанные исключительно на данных, могут демонстрировать накопление ошибок и аномальное поведение при многократном применении. В то же время практическая реализация интеллектуальных методов осложняется отсутствием формализации в архитектурах выполнения моделей искусственного интеллекта, а также проблемами, связанными с масштабируемостью, отказоустойчивостью и соблюдением требований к качеству обслуживания в условиях нестационарной нагрузки.

Дополнительную актуальность приобретают задачи построения распределенной инфраструктуры интеллектуальных транспортных систем, способной обеспечивать надежный сбор, хранение и обработку больших объемов разнородной информации, а также воспроизводимую верификацию алгоритмов и устойчивое функционирование вычислительных контуров.

Таким образом, актуальность темы исследования определяется необходимостью разработки комплексной методологии создания информационной системы управления транспортными потоками, объединяющей методы прогнозирования макроскопических и микроскопических характеристик транспортного потока, формализованные подходы к построению архитектуры исполнения моделей искусственного интеллекта и методологию построения инфраструктуры интеллектуальных транспортных систем. Решение данных задач имеет существенное научное и практическое значение для повышения эффективности, устойчивости и масштабируемости современных интеллектуальных транспортных систем.

Степень разработанности темы исследования. Степень разработанности темы определяется широким кругом исследований, соответствующих направлениям, в части разработки компьютерных методов и моделей описания, оценки и оптимизации информационных процессов и ресурсов, технологий извлечения и анализа информации в больших базах данных, а также архитектур программно-аппаратных комплексов и методов обеспечения надёжной обработ-

ки информации в распределённых интеллектуальных транспортных системах. Существенный массив работ посвящён пространственно-временному анализу и прогнозированию транспортных характеристик на основе данных сенсорных сетей и моделей машинного обучения, включая графовые и нейросетевые подходы, развиваемые в исследованиях Zheyi Wu, Junbo Li, Bin Yu, Chaoqun Chen, Jie Zhou, Jun Zhang, Thomas N. Kipf, William L. Hamilton, Petar Veličković, Max Welling, Yoshua Bengio, Geoffrey Hinton, Yann LeCun.

Значительная часть публикаций относится к микроскопическому моделированию транспортной динамики и моделям следования за лидером, где сформированы классические параметризованные модели и методы их калибровки, а также рассматривается влияние нестационарности условий движения и вариативности поведения водителя. В данном направлении развиваются исследования Denos C. Gazis, Robert Herman, Richard W. Rothery, Peter G. Gipps, Martin Treiber, Arne Kesting, Dirk Helbing, Masao Bando, Rui Jiang, Carlos Laval, Vincenzo Punzo.

Отдельное направление составляют методы, ориентированные на устойчивость моделей, в том числе при последовательном применении, и на формализуемый контроль качества результатов прогнозирования и управления, включая подходы обучения с подкреплением и методы регуляризации поведения политик. Данные вопросы отражены в работах Timothy P. Lillicrap, Scott Fujimoto, Tuomas Haarnoja.

В части построения и эксплуатации информационных систем обработки транспортных данных развиваются исследования распределённых информационных систем, алгоритмов согласования, обеспечения отказоустойчивости, масштабируемости и соблюдения требований к качеству сервиса. Существенный вклад в соответствующие области внесён Leslie Lamport, Diego Ongaro, John Ousterhout, Jeffrey Dean, Sanjay Ghemawat, Armando Fox.

В отечественной научной школе вопросы интеллектуальных транспортных систем, транспортной телематики и архитектурной организации ИТС рассматриваются в работах В. В. Комарова, С. А. Гарагана, И. А. Евстигнеева, С. В. Жанказиева, Р. В. Душкина и М. В. Яшиной, посвященных вопросам стандартизации, проектирования, внедрения и развития интеллектуальных транспортных систем, цифровой транспортной инфраструктуры и информационно-управляющих комплексов транспортной отрасли.

Отдельное направление отечественных исследований связано с моделированием транспортных потоков, транспортной динамики и разработкой методов управления движением. Существенный вклад в развитие данного направления внесли А. П. Буслаев, А. Г. Таташев и А. В. Гасников, чьи работы посвящены математическому моделированию транспортных процессов, анализу транспортных сетей, вопросам транспортного равновесия, оптимизации потоков и развитию методов управления транспортными системами в условиях высокой динамичности и неопределенности среды.

Методологические и вычислительные основы моделирования сложных динамических процессов, обработки данных и построения высокопроизводительных распределенных вычислительных систем получили развитие в работах А. С. Бугаева, А. М. Валуева и Б. Н. Четверушкина. Их исследования в области прикладной математики, вычислительной физики, численных методов и высокопроизводительных вычислений сформировали научную основу для разработки и анализа сложных информационно-вычислительных систем, включая системы обработки и моделирования транспортных процессов.

Анализ и обобщение работ по теме показывают, что при высокой степени проработанности отдельных компонентов, включая модели пространственно-временного прогнозирования, классические и модели микродинамики основанные на данных, а также методы построения распределённых интеллектуальных транспортных систем, сохраняется методологический разрыв между задачами извлечения закономерностей из транспортных данных, устойчивым моделированием поведения в условиях нестационарности и формализуемым обеспечением требований к надёжности и качеству сервиса при внедрении и эксплуатации соответствующих информационных процессов и ресурсов.

Научная проблема исследования заключается в отсутствии комплексных методов и методологий оценки и оптимизации информационных процессов и ресурсов интеллектуальных транспортных систем, обеспечивающих извлечение закономерностей и прогнозирование по пространственно-временным данным при нестационарности. Существенными проявлениями проблемы являются недостаточная адаптивность параметризованных моделей к изменению условий, возникновение аномальных предсказаний у моделей, основанных на данных, при последовательном применении, ограниченная пригодность доступных информационных ресурсов в виде наборов данных для воспроизводимой оценки и обучения, а также недостаточная формализация архитектур и про-

цедур верификации, обеспечивающих устойчивость обработки информации в условиях нестационарной нагрузки.

Целью диссертационной работы является повышение эффективности функционирования интеллектуальных транспортных систем за счёт разработки методов прогнозирования макроскопических и микроскопических характеристик транспортного потока, формирования методологии построения и верификации архитектуры информационной системы сбора, хранения и обработки данных о состоянии среды.

Для достижения поставленной цели необходимо было решить следующие **задачи**:

1. Провести анализ современного состояния интеллектуальных транспортных систем.
2. Разработать метод прогнозирования макроскопических характеристик транспортного потока. Экспериментально оценить предложенный метод на общепринятых наборах данных пространственно-временного прогнозирования транспортных потоков и сравнить его с базовыми методами.
3. Разработать метод прогнозирования микроскопических характеристик транспортного потока, позволяющий учитывать динамически изменяемый контекст сцены. Провести экспериментальное исследование предложенного метода на общепринятых наборах данных, оценив устойчивость при последовательном применении и чувствительность к накоплению ошибок.
4. Разработать и формализовать методологию конфигурирования архитектуры исполнения методов искусственного интеллекта в интеллектуальных транспортных системах, ориентированную на соблюдение требований уровня сервиса по задержке, пропускной способности и доступности в условиях нестационарной нагрузки.
5. Разработать архитектуру системы обработки больших данных для интеллектуальных транспортных систем структурированную по функциональным зонам.
6. Разработать и обосновать методологию построения инфраструктуры ИТС, ориентированную на отказоустойчивость, масштабируемость и адаптируемость сбора, хранения и обработки больших объёмов разнородных данных.

Объектом исследования процессы функционирования интеллектуальных транспортных систем, включающие сбор и обработку разнородных данных о дорожной обстановке, а также методы моделирования и прогнозирования характеристик транспортного потока.

Предметом исследования научно-методический комплекс методов прогнозирования макроскопических и микроскопических характеристик транспортного потока, методология построения, верификации и выбора архитектур исполнения моделей искусственного интеллекта и инфраструктурных решений интеллектуальных транспортных систем при ограничениях по качеству сервиса, масштабируемости и отказоустойчивости.

Научная новизна результатов диссертации. В процессе выполнения диссертационной работы получены следующие оригинальные научные результаты:

1. Разработан метод пространственно-временного прогнозирования макроскопических характеристик транспортного потока PTSTG, *отличающийся от существующих* синтезом патч-ориентированного временного кодирования с адаптивной стохастической по строкам матрицей смежности, обучаемой из данных без использования фиксированной дорожной топологии, и вычислительно простым графовым уточнением узловых представлений.
2. Разработан метод динамической коррекции коэффициентов классической модели следования за лидером, *отличающийся от существующих*, динамической адаптацией параметров во времени на основе текущего состояния движения при сохранении физической интерпретируемости параметров модели.
3. Разработан гибридный метод прогнозирования микроскопических характеристик транспортного потока, основанный на данных, *отличающийся от существующих* реализацией функционального разделения модулей, при котором блок, основанный на данных, обеспечивает точность воспроизведения динамики (короткосрочное и долгосроное окно), а классический блок задаёт формализуемые ограничения и подавляет аномальные воздействия в замкнутом контуре.
4. Разработана комплексная методология построения архитектуры исполнения моделей искусственного интеллекта в интеллектуальных транспортных системах, *отличающаяся от существующих* формали-

зацией архитектуры в виде ориентированного мультиграфа исполнения с параметризованными вершинами и рёбрами, введением вероятностных ограничений по задержке и устойчивости по интенсивности входа, и обеспечением воспроизводимого синтеза и выбора конфигурации исполнения на основе измеримых метрик.

5. Разработана архитектура системы сбора, хранения и обработки больших разнородных данных для интеллектуальных транспортных систем, *отличающаяся от существующих* структурированием по функциональным зонам, охватывающая весь процесс функционирования системы начиная со сбора данных до аналитики, хранения и визуализации.
6. Разработана методология построения инфраструктуры интеллектуальных транспортных систем, *отличающаяся от существующих* тем, что объединяет архитектуру обработки больших данных с механизмом децентрализованной кластеризации серверов на основе метрики сетевого расстояния (RTT), консенсуса Raft и автоматического выбора лидера сервисом `Agent`, а также специализированным алгоритмом динамического распределения приоритетных задач при ограничении по времени выполнения и методом генерации статистически обоснованных синтетических данных транспортного потока.

Теоретическая и практическая значимость. Теоретическая значимость диссертационного исследования заключается в развитии методологических основ создания системы управления транспортными потоками на базе интеллектуальной обработки пространственно-временных данных и адаптивного моделирования транспортных процессов. В работе:

- обоснована реализация замкнутого контура управления транспортными потоками с систематизацией сценариев взаимодействия подсистем мониторинга, прогнозирования и формирования управляющих воздействий в составе интеллектуальной транспортной системы;
- установлена связь между выбором алгоритмических и технологических решений (прогнозирование макро- и микроскопических характеристик, адаптивная коррекция параметров моделей, архитектура исполнения вычислений) и показателями качества функционирования системы управления, включая устойчивость моделирования и точность прогнозирования в условиях нестационарности транспортной нагрузки;

- введено новое представление об архитектуре комплекса обработки транспортных данных как распределённой системы информационного обмена, обоснована структурно-функциональная модель, включающая компоненты сбора, хранения и обработки больших разнородных данных, а также средства гарантированного исполнения моделей искусственного интеллекта по требованиям к задержке, пропускной способности и доступности;
- показана концептуальная возможность интеграции пространственно-временного прогнозирования и гибридного моделирования транспортных процессов с распределённым исполнением моделей искусственного интеллекта в едином вычислительном контуре, обеспечивающем масштабируемость на крупные сенсорные сети и устойчивость к изменениям статистических свойств входных данных.

Практическая значимость работы определяется возможностью применения результатов диссертационного исследования при проектировании, внедрении и эксплуатации интеллектуальных транспортных систем и центров управления дорожным движением. Разработанные методы пространственно-временного прогнозирования и гибридного моделирования, а также предложенные решения по архитектуре сбора, хранения и обработки больших разнородных данных и по организации отказоустойчивого распределённого исполнения моделей искусственного интеллекта могут быть использованы:

- в системах мониторинга и прогнозирования состояния улично-дорожной сети для поддержки оперативного управления дорожным движением (адаптивное регулирование, маршрутизация, информирование участников движения);
- в программно-аппаратных комплексах обработки транспортных данных, обеспечивающих протокольную совместимость, интеграцию с внешними информационными системами и соблюдение требований к качеству обслуживания прикладных сервисов;
- при разработке и испытаниях инфраструктуры георасределённых ИТС, включая воспроизводимую валидацию моделей при ограниченности натурных данных.

Методы исследования. Методологическую основу работы составляет системный подход, включающий системный анализ предметной области, декомпозицию задач и формализацию требований к качеству функционирования

интеллектуальных транспортных систем. В ходе исследования использовались методы математического моделирования и идентификации параметров для описания микроскопической динамики движения и адаптации параметров классических моделей, методы интеллектуального анализа данных и машинного обучения для построения прогнозных моделей по пространственно-временным данным, а также методы теории графов и анализа временных рядов для учёта пространственных взаимодействий и временных зависимостей. Для обеспечения обоснованности выводов применялись методы теории вероятностей и математической статистики, включая процедуры сравнительного анализа и оценку качества моделей по формализованным метрикам. Исследование включало вычислительный и имитационный эксперимент, в том числе оценку поведения моделей при последовательном применении, а также экспериментальную верификацию на натурных и общепринятых наборах данных. При решении задач проектирования архитектуры исполнения и инфраструктуры применялись методы многокритериальной оптимизации и верификации ограничений по качеству сервиса.

Основные положения, выносимые на защиту:

1. Метод пространственно-временного прогнозирования макроскопических характеристик транспортного потока PTSTG, *позволяющий* выполнять однопроходное многогоризонтное прогнозирование на горизонтах 15/30/60 минут при сопоставимой точности и обеспечивать масштабируемость на крупные сенсорные сети за счёт разрежения межузловых взаимодействий.
2. Метод динамической коррекции коэффициентов классической модели следования за лидером, *позволяющий* обеспечивать учёт нестационарности дорожной среды и повышение устойчивости моделирования в условиях вариативности внешних факторов и режимов движения.
3. Гибридный метод прогнозирования микроскопических характеристик транспортного потока, основанный на данных, *позволяющий* снижать накопление ошибки при последовательном применении модели и повышает устойчивость при сдвиге распределения при сохранении точности в типичных режимах движения.
4. Комплексная методология построения архитектуры исполнения моделей искусственного интеллекта в интеллектуальных транспортных системах, *позволяющая* гарантировать заданные уровни задержки,

пропускной способности и доступности в условиях нестационарной нагрузки ИТС.

5. Архитектура системы сбора, хранения и обработки больших разнородных данных для интеллектуальных транспортных систем, *позволяющая* снизить сложность интеграции с внешними системами за счет учета протокольной совместимости, адаптированной к типу источника и сценарию взаимодействия.
6. Методология построения инфраструктуры интеллектуальных транспортных систем, *позволяющая* обеспечивать самонастройку и отказоустойчивость георасределённой ИТС и выполнять воспроизводимое тестирование при отсутствии реальных данных.

Степень достоверности и апробации результатов Основные результаты диссертационного исследования были представлены на 44 международных конференциях, форумах и научных семинарах:

Международная научно-практическая конференция «Quality Management, Transport and Information Security, Information Technologies» (ITQMIS) (Санкт-Петербург, 2018; Сочи, 2019). Международная научная конференция «Wave electronics and its application in information and telecommunication systems» (WECONF) (Санкт-Петербург, 2018-2024). Международная конференция «Технологии и предпринимательство в цифровом обществе» (TEDS) (Москва, 2019). Международная конференция «Systems of signals generating and processing in the field of on board communications» (Москва, 2019-2021, 2023-2025). Международная конференция «Systems of Signal Synchronization, Generating and Processing in Telecommunications» (SYNCHROINFO) (Ярославль, 2019; Светлогорск, 2020-2021; Архангельск, 2022; Псков, 2023; Выборг, 2024; Тюмень, 2025). International Conference on Engineering Management of Communication and Technology (EMCTECH) (Вена, 2020). Международная конференция «Intelligent technologies and electronic devices in vehicle and road transport complex» (TIRVED) (Москва, 2021, 2023, 2025). Международная отраслевая научно-техническая конференция «Технологии информационного общества» (Москва, 2017-2019). Международная конференция «Транспортные потоки на сетях» (Сириус, 2023, 2025). Международная научно-техническая конференция «Новые информационные технологии в телекоммуникациях и почтовой связи» (Минск, 2021-2022). International Conference Computational and Mathematical Methods in Science and Engineering (Кадис, 2025). Международный промышлен-

ный форум «Интеллект машин и механизмов» (Севастополь, 2021). Научный семинар «Фундаментальные и прикладные проблемы развития автомобильно-дорожного комплекса России» академика РАН, д.ф.-м.н., профессора Валерия Васильевича Козлова (Москва, 2018-2025).

Результаты также обсуждались на заседании кафедры, а также на научных семинарах в МТУСИ.

Апробация результатов подтверждается свидетельствами о регистрации программ для ЭВМ, представленных в приложении К.

Личный вклад.

Все основные научные результаты и положения, выносимые на защиту, получены лично автором или при его определяющем участии. Автору принадлежит постановка научной проблемы, формирование цели и задач исследования, разработка методологических, математических, алгоритмических и архитектурных решений, проведение вычислительных и имитационных экспериментов, анализ и интерпретация полученных результатов, а также подготовка публикаций по теме диссертационного исследования. В работах, опубликованных в соавторстве, соискателю принадлежит определяющая роль в постановке исследовательских задач, формализации предлагаемых методов, разработке архитектурных и алгоритмических решений, выборе экспериментальных сценариев, анализе результатов и обобщении выводов применительно к методологии создания системы управления транспортными потоками.

В работах [1; 2] соискателю принадлежит разработка общего подхода к пространственно-временному прогнозированию макроскопических характеристик транспортного потока, включая обоснование применения патч-ориентированного временного кодирования, адаптивной обучаемой матрицы смежности и графового уточнения узловых представлений.

В работах [3—14], связанных со сбором, определением и моделированием характеристик транспортного потока, соискателю принадлежит постановка и решение задач прогнозирования микроскопических характеристик транспортного потока.

В работах [4; 6; 11; 13; 15—30] соискателем разработаны и исследованы интеллектуальные и гибридные методы обработки транспортных данных, распознавания объектов и характеристик транспортной сцены, анализа устойчивости нейросетевых решений и прогнозирования динамики движения.

В работах [31—48] соискателю принадлежит разработка и обоснование архитектурных решений для сбора, хранения, обработки и защищенного информационного обмена в интеллектуальных транспортных системах.

В работах [31—36; 41—45; 47; 48], связанных с поддержкой функционирования ИТС, обработкой и хранением транспортных данных, соискателем разработана архитектура системы сбора, хранения и обработки больших разнородных данных для интеллектуальных транспортных систем.

В работах [5; 10; 29; 30; 37—40; 46—59], связанных с генерацией синтетических данных, динамическим распределением ресурсов, поддержкой функционирования ИТС, передачей информации между транспортными средствами и адаптивным управлением светофорами, соискателем разработана методология построения инфраструктуры интеллектуальных транспортных систем.

В публикациях [5; 7; 52; 54—60] соискателем дополнительно обоснованы подходы к применению обучения с подкреплением, имитационного моделирования и цифрового представления дорожной среды для задач управления транспортными потоками, выбора траекторий движения и испытания алгоритмов ИТС.

В публикациях [11; 14; 18; 19; 21; 24—29; 61—63] автором исследованы методы сбора и прогнозирования микро и макро-характеристик транспортного потока поступающих из различных источников информации о дорожной ситуации. Эти результаты использованы при формировании подсистем мониторинга и семантического восприятия в составе интеллектуальной транспортной системы.

В публикациях [41—48] соискателем проработаны вопросы разработки информационных систем, обработки данных и применения распределённых технологий в задачах транспортной и цифровой инфраструктуры. Данные результаты использованы при обосновании архитектурных решений, обеспечивающих интероперабельность, надежность и воспроизводимость информационных процессов ИТС.

Автором лично выполнены систематизация и обобщение результатов исследования, сформулированы выводы по главам и заключение диссертации, определены теоретическая и практическая значимость работы, подготовлены материалы апробации и публикации по теме диссертационного исследования. Совокупность публикаций [1—63] подтверждает полноту апробации основных

научных результатов диссертации и их соответствие положениям, выносимым на защиту.

Благодарности. Особую благодарность автор выражает доктору физико-математических наук, профессору Буслаеву Александру Павловичу, научному руководителю кандидатской диссертации, чьи научные идеи, профессиональное наставничество и личный пример оказали значительное влияние на формирование научных взглядов автора и во многом определили направление настоящего исследования.

Реализация и внедрение. Алгоритмы, методы и методология, разработанные в настоящем исследовании, внедрены в следующих организациях:

- «Минимакс-94» (АО) (г. Москва): внедрение указанных результатов позволило повысить точность прогнозирования транспортной ситуации, улучшить качество управления транспортными потоками и повысить оперативность реагирования на изменения дорожной обстановки;
- «Эр Си Технолоджис» (ООО) (г. Москва): внедрение указанных результатов позволило повысить надёжность функционирования информационных систем, обеспечить масштабируемость обработки данных и соблюдение требований к качеству сервиса;
- «Транснависофт» (ООО) (г. Москва): внедрение указанных результатов позволило повысить устойчивость моделирования транспортных процессов, обеспечить воспроизводимость оценки качества работы интеллектуальных транспортных систем и повысить эффективность анализа транспортных данных;
- учебный процесс кафедры «Математическая кибернетика и информационные технологии» ордена Трудового Красного Знамени федерального государственного бюджетного образовательного учреждения высшего образования «Московский технический университет связи и информатики» (МТУСИ);
- учебный процесс кафедры «Инженерия и математика прикладных систем искусственного интеллекта» федерального государственного бюджетного образовательного учреждения высшего образования «Московский автомобильно-дорожный государственный технический университет (МАДИ)».

Подтверждается соответствующими актами внедрения результатов диссертационной работы (5 шт.), представленных в приложении Л.

Соответствие специальности. Тематика и результаты диссертации соответствуют следующим направлениям специальности: 2.3.8. Информатика и информационные процессы.

1. п. 8 - разработка систем принятия решений на основе баз данных и знаний, реализующих имитационные модели прогнозирования изменения материальных процессов и событий, поскольку разработаны подходы к прогнозированию и устойчивому моделированию транспортного движения, включая адаптивную коррекцию параметров микроскопических моделей следования и оценку поведения потока в меняющихся режимах;
2. п. 9 - разработка архитектур программно-аппаратных комплексов поддержки цифровых технологий сбора, хранения и передачи информации в инфокоммуникационных системах, в том числе с использованием «облачных» интернет-технологий, и оценка их эффективности, поскольку обоснована архитектура исполнения моделей искусственного интеллекта в составе ИТС с требованиями к задержке, пропускной способности и доступности, а также предложены принципы организации вычислительной инфраструктуры;
3. п. 13 - разработка и применение методов распознавания образов, кластерного анализа, нейросетевых и нечетких технологий, решающих правил, мягких вычислений при анализе разнородной информации в базах данных, поскольку предложены и экспериментально исследованы интеллектуальные (в том числе гибридные) методы прогнозирования макро- и микроскопических характеристик транспортного потока, ориентированные на устойчивость к нестационарности и дрейфу данных;
4. п. 14 - разработка и исследование принципов организации и функционирования распределенных информационных систем и баз данных, прикладных протоколов информационных сетей, форматов представления данных и языков информационного поиска в распределенных информационных ресурсах, поскольку разработана архитектура системы сбора, хранения и обработки больших разнородных транспортных данных с учетом интероперабельности компонентов и интеграции с внешними информационными системами;

Публикации. По материалам диссертации опубликовано 63 научные работы (в том числе 16 без соавторов), из которых: публикации в рецензируемых

периодических изданиях, входящих в перечень ВАК – 10; в научных изданиях, индексируемых в международных базах Scopus/WoS – 38 (из которых: Q1 – 6, Q2 – 2, которые в соответствии с Рекомендацией ВАК от 21.12.2023, приравниваются к публикациям в рецензируемых изданиях категории К1); свидетельства о государственной регистрации программ для ЭВМ – 15 (которые приравниваются к публикациям в рецензируемых изданиях, рекомендованных ВАК).

Объем и структура работы. Диссертация состоит из введения, 5 глав, заключения и 11 приложений. Полный объём диссертации составляет 399 страниц, включая 66 рисунков и 50 таблиц. Список литературы содержит 277 наименований.

Глава 1. Современное состояние интеллектуальных транспортных систем

Согласно ГОСТ Р 56829-2015 под понятием интеллектуальная транспортная система (ИТС) понимается система управления, интегрирующая современные информационные и телематические технологии и предназначенная для автоматизированного поиска и принятия к реализации максимально эффективных сценариев управления транспортно-дорожным комплексом региона, конкретным транспортным средством или группой транспортных средств с целью обеспечения заданной мобильности населения, максимизации показателей использования дорожной сети, повышения безопасности и эффективности транспортного процесса, комфортности для водителей и пользователей транспорта.

Современные интеллектуальные транспортные системы рассматриваются в литературе как комплексные социотехнические системы, объединяющие дорожную инфраструктуру, средства связи, сенсорные подсистемы, вычислительные платформы, методы анализа данных и механизмы автоматизированного управления. В ранних и обзорных работах ИТС трактуются преимущественно как совокупность технологий повышения безопасности, пропускной способности и эффективности транспортной сети [64—67]. Последующее развитие исследований сместило акцент подходы основанные на данных, в которых ключевую роль играют сбор, интеграция и аналитическая обработка транспортных данных [68].

В отечественной литературе ИТС также описываются как совокупность телематических, информационных и управляющих решений, предназначенных для повышения безопасности дорожного движения, эффективности эксплуатации транспортной инфраструктуры и качества транспортного обслуживания [69—72]. При этом особое значение имеет согласование архитектурных решений с нормативными требованиями, эксплуатационными ограничениями и условиями внедрения на конкретной территории.

Обзорные исследования фиксируют устойчивое расширение предметной области ИТС: от локальных систем управления дорожным движением и информирования водителей к интегрированным платформам, включающим сенсорные сети, интеллектуальную обработку данных, кооперативные коммуникации, цифровые сервисы мобильности и автоматизированное принятие

решений [73—79]. В прикладных исследованиях также подчёркивается, что архитектура и функциональный состав ИТС существенно зависят от локальных институциональных, инфраструктурных и экономических условий внедрения [80—86].

В настоящем исследовании ИТС рассматривается как система городского и регионального уровней, представляющая собой социотехническую архитектуру, внутри которой в согласованном режиме функционируют домены управления дорожным движением (*ATMS*), общественного транспорта и приоритизации подвижного состава (*APTS/TSP*), информирования участников дорожного движения (*ATIS*), кооперативных транспортных коммуникаций (*C-ITS/V2X*), дорожной метеорологии (*RWIS*), а также обеспечения безопасности и управления инцидентами.

Домен интеллектуальной транспортной системы - специфическая область применения, которая включает в себя одну или более сервисных групп.

Сервисная группа интеллектуальной транспортной системы - один или более схожих или сопряженных сервисов, предназначенных для пользователей ИТС.

Архитектурная модель ИТС трактуется как многослойная конструкция включающая в себя связь, данные, приложения, представление, дополненная сквозными функциями синхронизации времени, обеспечения безопасности, мониторинга и управления жизненным циклом данных и моделей.

Ключевым принципом выступает интеллектуальный контур мониторинг — прогноз — управление (МПУ), трактуемый как минимальная самодостаточная замкнутая петля управления, обеспечивающая не только реактивное реагирование, но и проактивное воздействие на динамику транспортной сети при явном учёте неопределённостей и ограничений безопасности. Ниже представлено описание контура МПУ.

– **Мониторинг.** Преобразование неоднородных сенсорных потоков (видеоданные, радиолокация, индукционные петли, телеметрия подвижного состава и др.) в структурированные, топологически согласованные семантические представления транспортного состояния (интенсивности, скорости, длины и динамика очередей, траектории, события) с обязательной оценкой качества, надёжности и неопределённости оценок.

– **Прогноз и оценка состояния.** Вероятностная мультигоризонтная оценка и прогноз на графе улично-дорожной сети с ассимиляцией текущих наблюдений, учётом физических ограничений и последующей статистической калибровкой распределений для обеспечения корректности доверительных характеристик.
– **Управление.** Иерархический синтез управляющих воздействий на базе моделей предиктивного управления (MPC), реализуемый в конфигурации контура безопасного исполнения, обеспечивающего соблюдение инвариантов нормативных требований, межфазных интервалов, пешеходных фаз, а также ограничений скорости изменений управляющих параметров.

Указанный контур реализует трассируемое сопряжение архитектурных решений с измеримыми показателями эффективности и процедурами приёмочных испытаний. В отличие от типовых автоматизированных систем, он обеспечивает операционно значимый, воспроизводимый и экономически интерпретируемый результат.

В данном контексте последовательно развивается архитектура интеллектуальных транспортных систем, их иерархическая структура и уровни управления, а также функциональные подсистемы, которые будут описаны далее в данной главе. Особое внимание уделяется методологии анализа, практикам внедрения и количественной оценке достигаемых эффектов. Ключевым элементом, обеспечивающим интеграцию инженерных решений, измеримость показателей функционирования и управляемость эксплуатационных процессов, выступает интеллектуальный контур МПУ, который выполняет роль несущей структурной основы всей системы. Простая автоматизация, предполагающая реализацию функций уровня контроллера, сбор телеметрических данных и передачу команд, образует необходимый компонент инфраструктуры, однако критериям интеллектуальной транспортной системы сама по себе не отвечает. Достижение целевых показателей эффективности требует сквозного мониторинга, прогнозирования динамики транспортных потоков и механизмов управления, опирающихся на осознанную оценку рисков.

Проектирование и разработка интеллектуальных транспортных систем предполагают соответствие требованиям международных стандартов (ISO, ETSI, IEEE, SAE, 3GPP) и нормативно-правовой базе Российской Федерации, в состав которой входят Федеральные законы №196-ФЗ, №257-ФЗ, №152-ФЗ,

№187-ФЗ, Правила дорожного движения (утв. Постановлением Правительства РФ №1090), Постановление Правительства РФ №1119, Технический регламент Таможенного союза ТР ТС 018/2011, а также решения Государственной комиссии по радиочастотам (ГКРЧ) в диапазоне 5,9 ГГц. Без учета этих нормативных документов невозможно провести обоснование архитектурных решений, дать количественную оценку эффективности и рисков, а также сформировать обоснованный план внедрения ИТС.

1.1 Архитектурные основы ИТС

Целесообразно рассматривать ИТС как композицию доменов. Эти домены различаются по функциям, источникам данных, временным горизонтам принимаемых решений и стандартам взаимодействия. При этом они объединены едиными архитектурными принципами.

Такой подход устраняет искусственное противопоставление полевых комплексов и центральных платформ. Реальная ценность формируется на их стыке. Например, фазовый план светофора и алгоритм координации неразрывно связаны с надёжной доставкой телеметрии и управляемым качеством данных. Кооперативные сервисы V2X требуют не только радиоинтерфейсов, но и доверенной инфраструктуры, точной синхронизации времени и соблюдения регламентов информационной безопасности.

Архитектурное представление ИТС целесообразно связывать не только с перечнем отдельных сервисов, но и с формальными моделями взаимодействия компонентов. В работе [87] предложен онтологический подход к построению архитектуры ИТС на основе сети транспортных датчиков, что подчёркивает важность семантической согласованности данных, сервисов и управляющих воздействий. В исследованиях, посвящённых инфраструктуре ИТС, также отмечается, что устойчивое функционирование таких систем требует согласования физического уровня, уровня связи, информационных сервисов и управляющих приложений [88; 89].

Переход к архитектуре, ориентированной на большие данные и сетевое взаимодействие транспортных средств, усиливает требования к масштабируемости, интероперабельности и отказоустойчивости. Модели ИТС, основанные

на больших данных, рассматривают транспортную сеть как источник непрерывных потоков разнородной информации, требующих потоковой обработки, хранения, агрегации и последующего использования в прогнозных и управляющих задачах [90; 91]. В рамках IoT-ориентированных подходов дополнительно подчёркивается необходимость распределения вычислений между периферийными устройствами, облачной инфраструктурой и прикладными сервисами управления трафиком [92; 93]. Отдельным направлением является эффективная передача данных в ИТС, поскольку частота публикации сообщений, пропускная способность каналов и задержки доставки непосредственно влияют на применимость управляющих контуров в реальном времени [94].

Типология доменов интеллектуальных транспортных систем не сводится к перечню сервисов. Каждый домен опирается на собственные технические и операционные ограничения.

Управление дорожным движением требует реактивности на уровне светофорного контроллера и согласованных координационных планов на коридорном и городском уровнях. Информационные сервисы зависят от агрегированных, репрезентативных и стабильных данных о состоянии трафика. Диспетчеризация общественного транспорта предъявляет требования к предсказуемости маршрутов, алгоритмам предоставления приоритета и консистентности сообщений о времени прибытия. Кооперативная безопасность накладывает ограничения на латентность передачи, и точность временной синхронизации. Дорожная метеорология, в свою очередь, ориентирована на устойчивое качество измерений и корректные модели влияния погодных условий на эксплуатационные решения.

Обобщённая карта соответствий между доменами, их функциями, источниками данных, временными горизонтами и применяемыми стандартами представлена в таблице 1. Таблица служит основой для согласования интерфейсов и профилей обмена данными между ведомствами и поставщиками.

Таблица 1 — Типология доменов ИТС: функции, данные, временные горизонты, стандарты

Домен	Ключевые функции	Источники данных	Горизонт	Профили стандартов
ATMS/ АСУДД	Адаптивное управление СФ, координация коридоров, VSL, инциденты	Датчики/видео, контроллеры СФ, FCD/Probe	с–мин	ISO 19091; NTCIP; TMDD; ГОСТ Р 52282/52289
ATIS	Маршрутизация, предупреждения, прогнозы времени	DATEX II, TPEG, FCD, соцданные	мин–ч	ISO/TS 21219; DATEX II
APTS	AVL/APS, TSP, расписания, пассажирская информация	AVL, SIRI/NeTEx, валидаторы	с–ч	SIRI/NeTEx; GTFS/GTFS-RT
ETC/EFC	Взимание платы, клиринг, интероперабельность	DSRC 5.8, бортовые трандеры, биллинг	с–мин	EN 15509; ISO 12855
C-ITS/V2X	CAM/DENM, SPaT/MAP, BSM, предупреждения	RSU/OBU, PKI, GNSS/PTP	мс–с	ETSI EN 302 637; SAE J2735/J2945; ETSI TS 103 097
RWIS	Мониторинг погоды/состояния дороги	Метеостанции, датчики покрытия	мин–ч	NTCIP 1204; DATEX II
Freight/ CVO	Мониторинг регламентируемых ТС, дор. сборы	TARV, тахографы, RFID	ч–д	ISO 15638
Safety/ ADAS/ eCall	Критические уведомления, предупреждения, HMI	OBU, ЭРА-ГЛОНАСС, CV data	мс–с	ISO 26262; SOTIF; ГОСТ Р 55534
MaaS	Билеты, интеграция сервисов, тарифы	API операторов, платежные шлюзы	мин–ч	SIRI/NeTEx; открытые API

Методологически существенным свойством доменной типологии выступает её согласованность с уровнями управления и соответствующими временными масштабами. На уровне устройства и узла определяющими оказываются атомарность и предсказуемость действий, тогда как коридорный уровень требует совместимости локальных реакций с глобальными ограничениями, а городской и региональный уровни ориентированы на согласованность управляющих политик, ресурсных бюджетов и процедур обмена данными.

Такое разграничение уровней упрощает проверку реализации. Если цикл приоритетного движения общественного транспорта не обеспечивает согласованную реакцию диспетчера светофора на запрос приоритета, то аналитические методы на более высоких уровнях не смогут устранить систематические задержки на местном уровне. Именно поэтому уделяется особое внимание к семантической совместимости сообщений SPaT и MAP при рассмотрении взаимодействия между системами управления дорожным движением, диспетчеризацией общественного транспорта и ИТС. В том числе необходимо отслеживать корректность запросов приоритета. Эти элементы должны строго

соответствовать фактическому состоянию групп светофоров, а также требованиям правил дорожного движения и национальным стандартам. Зачастую формируют карты ответственности, которые позволяет единообразно описывать интерфейсы между компонентами. В такую схему включаются также меры обеспечения информационной безопасности и защиты приватности, а также явное указание на то, какая организация или роль несёт ответственность за соблюдение каждого из этих требований.

Для решения данных проблем при проектировании и эксплуатации ИТС используют многослойную архитектуру.

В рамках настоящей работы доменная модель используется для:

1. согласования информационных процессов между подсистемами мониторинга, прогнозирования и управления;
2. проектирования сквозных функций (безопасность, синхронизация времени, наблюдаемость, управление конфигурациями, управление жизненным циклом данных и моделей);
3. трассируемой привязки метрик и SLA к архитектурным слоям и доменам;
4. упрощения верификации интероперабельности и процедуры приёмочных испытаний.

С точки зрения инженерной дисциплины архитектура ИТС должна одновременно обеспечивать: интероперабельность и заменяемость компонентов, измеримость качества сервиса и воспроизводимость результатов, безопасную деградацию и восстановление, юридическую и нормативную корректность эксплуатации в условиях Российской Федерации.

Под эталонной архитектурой понимается набор согласованных архитектурных представлений, интерфейсов и типовых компонент, служащих основой для проектирования и внедрения систем в конкретной предметной области. Эталонные архитектуры в ИТС применяются для унификации терминов, границ подсистем, протоколов и профилей обмена, а также для снижения рисков интеграции.

Для сопоставления эталонных архитектур в контексте городской/региональной ИТС используются следующие критерии:

– **Область охвата**: дорожное движение, общественный транспорт, кооперативные сервисы, информационные сервисы, инфраструктурные функции;

- **Уровни и горизонты управления**: узел-коридор-сеть, требования к задержке и устойчивости;
- **Интероперабельность**: наличие формальных интерфейсов, профилей данных, механизмов совместимости;
- **Безопасность и доверие**: наличие модели доверия и требований к криптографическим процедурам и управлению ключами;
- **Эксплуатационная пригодность**: наблюдаемость, управляемость конфигураций, деградационные режимы, воспроизводимость;
- **Поддержка данных и ИИ**: управление качеством данных, происхождением, версиями моделей, контроль дрейфа.

На практике одновременно используются несколько слоёв эталонных архитектур:

1. **Системно-архитектурные рамки**, задающие принципиальную декомпозицию и представления;
2. **Коммуникационные и станционные модели**, описывающие протокольные уровни и роли узлов;
3. **Домен-ориентированные интерфейсы** (управление движением, информирование и т.д.), фиксирующие форматы данных, семантику событий и критерии качества.

В таблице 2 приведено обобщённое сравнение архитектурных подходов, используемых в современных ИТС, и их применимость к задачам настоящей работы.

Таблица 2 — Сравнение архитектур

Подход	**Сильные стороны**	**Ограничения/риски**	**Значимость для диссертации**
Архитектурные принципы взаимосвязных систем	Унифицируют декомпозицию по подсистемам и контрактам; улучшают трассируемость требований	Сами по себе не задают конкретных протоколов/профилей; требуют доменной конкретизации	Обосновывают рассмотрение ИТС как композиции доменов и необходимость сквозных функций
Стек C-ITS (кооперативные сервисы)	Чётко фиксируют роли дорожных, бортовых и центральных узлов; позволяют формализовать задержки, синхронизацию времени и безопасность сообщений	Высокая чувствительность к синхронизации времени и корректности семантики; риск «спецификационного перегруза» при переносе в обзорный раздел	Используются для формализации профилей сообщений, требований SLA, доверия и верификации интероперабельности
Доменно-ориентированные интерфейсы (управление движением/информирование)	Поддерживают межведомственную интеграцию; обеспечивают переносимость метрик и процедур приёмки	Различия в локальных словарях и профилях приводят к несовместимости при внедрении без профилирования	Необходимы для карты ответственности, SLA и методологии оценки эффектов внедрения
Data/AI платформа как референсная архитектура (стриминг, lake/DWH, MLOps)	Обеспечивает воспроизводимость, наблюдаемость, контроль дрейфа и регламентированное обновление моделей	Без формализации SLA и устойчивости возможны каскадные деградации при нестационарной нагрузке	Является основой методологии исполнения моделей ИИ при SLA по задержке/пропускной способности/доступности

Сравнение показывает, что ни одна архитектура не закрывает одновременно требования доменной интероперабельности, ограничений по задержке на нижних уровнях управления, эксплуатационной воспроизводимости и верифицируемости инвариантов безопасности при исполнении моделей ИИ.

В рамках настоящей работы принимается следующее:

1. Доменная модель (ATMS/ATIS/APTS/C-ITS/RWIS и др.) используется для разделения ответственности и формализации контрактов;
2. Многослойная архитектура используется для проектирования масштабируемости и независимой эволюции слоёв;
3. Сквозные функции (безопасность, наблюдаемость, конфигурации, управление данными и моделями) используются для обеспечения воспроизводимости и управляемой деградации;
4. Контур МПУ используется как минимальная замкнутая петля, связывающая архитектуру с измеримыми эффектами и приёмочными испытаниями.

Таким образом достигается согласовать разнородных стандартов и интерфейсов через профилирование. Обеспечивается контролируемая эксплуатация ИИ-моделей и потоковой обработки данных при нестационарной нагрузке. Позволяет достигать целевых значений SLA на каждом уровне управления.

Многослойная архитектура, изображённая на рисунке 1.1, обеспечивает чёткое разделение ответственности между полевым уровнем, сетью связи, платформой данных, прикладным слоем и уровнем представления. Это разделение позволяет развивать отдельные компоненты системы независимо, сохраняя сквозную совместимость.Изменение технологии доступа не должно влиять на модели данных или информационные процессы. Аналогично, обновление алгоритмов управления не должно прерывать работу конвейеров телеметрии. Каждый уровень функционирует в рамках собственных соглашений об уровне обслуживания и целевых показателей качества обслуживания.

Согласованность между слоями обеспечивается поперечными функциями. Такими как управление конфигурациями, единая система безопасности, сквозная наблюдаемость и синхронизация времени. Эти функции поддерживают целостность системы, обеспечивают обнаружение отказов и контролируют внесение изменений без нарушения устойчивости эксплуатации.

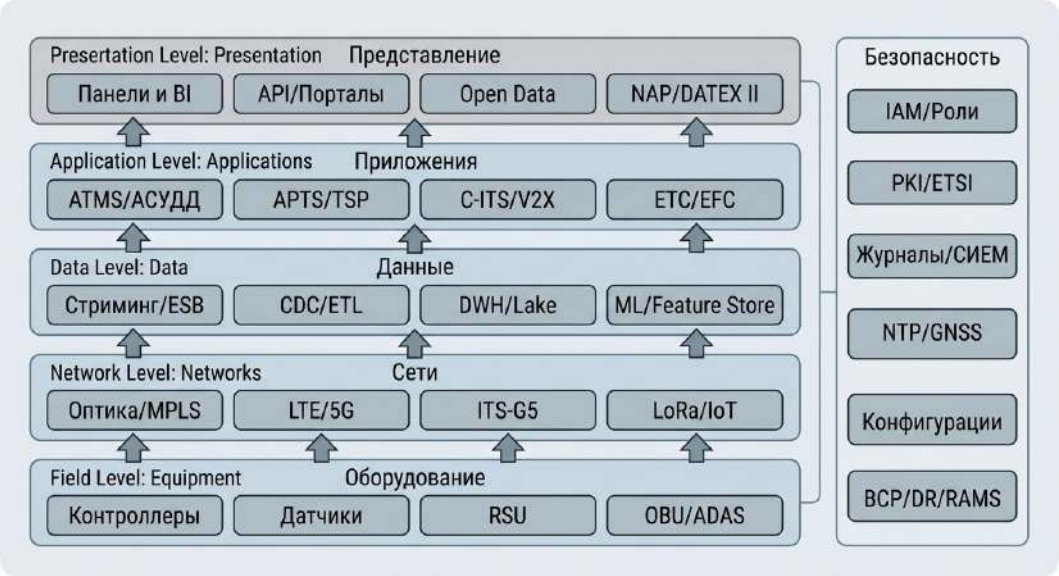

Рисунок 1.1 — Многослойная архитектура ИТС

Практическая ценность декомпозиции становится очевидной в сценариях сбоев и восстановления. Если центр управления недоступен, диспетчеры светофоров должны поддерживать планы действий в чрезвычайных ситуациях. Придорожные службы (RSU) переключаются в пассивный режим и перестают генерировать оперативные запросы, которые не подтверждаются фактическими фазами цикла работы светофора.

Если сигнал GNSS пропадает, в городских транспортных сетях активируется протокол точного времени PTP. Если он также недоступен или недостаточно точен, используются локальные таймеры, учитывающие оценку дрейфа и строгие временные ограничения на срок действия кэшированных сертификатов. Мониторинг позволяет вести полный журнал событий. Такой журнал необходим как для объективной проверки последствий, так и для анализа после инцидента.

Ключевое преимущество многослойной архитектуры это независимая масштабируемость. Полевой уровень масштабируется числом узлов и RSU. Сетевой слой масштабируется пропускной способностью и количеством точек доступа. Платформа данных масштабируется горизонтальным расширением потоковой обработки и хранилищ. Прикладной слой масштабируется числом доменных сервисов и вычислительных очередей. Уровень представления масштабируется в зависимости от количества потребителей API и информационных панелей.

Модель станции интеллектуальной транспортной системы по ISO 21217 служит универсальным интерфейсом для описания бортовых устройств, до-

рожных станций и центральной инфраструктуры. На схеме 1.2 показано, как прикладные сервисы взаимодействуют с функциями уровня объектов и протоколом GeoNetworking, а затем с транспортным и канальными уровнями стека связи.

Функции безопасности и управления жизненным циклом проходят через всю архитектуру. Данные функции обеспечивают строгую последовательность процедур выпуска, ротации и отзыва криптографических ключей. Обновление программного обеспечения должно удовлетворять требованиям UN R156 В РФ станции должны функционировать в рамках решений Государственной комиссии по радиочастотам для полосы 5,9 ГГц. Определяют планирование зон радиопокрытия, допустимую плотность и мощность дорожных станций. Возможен выбор технологического стека ITS-G5/802.11-OCB, C-V2X/3GPP PC5 или гибридной конфигурации. На рисунке 1.2 представлена иерархия станции ИТС согласно ISO 21217.

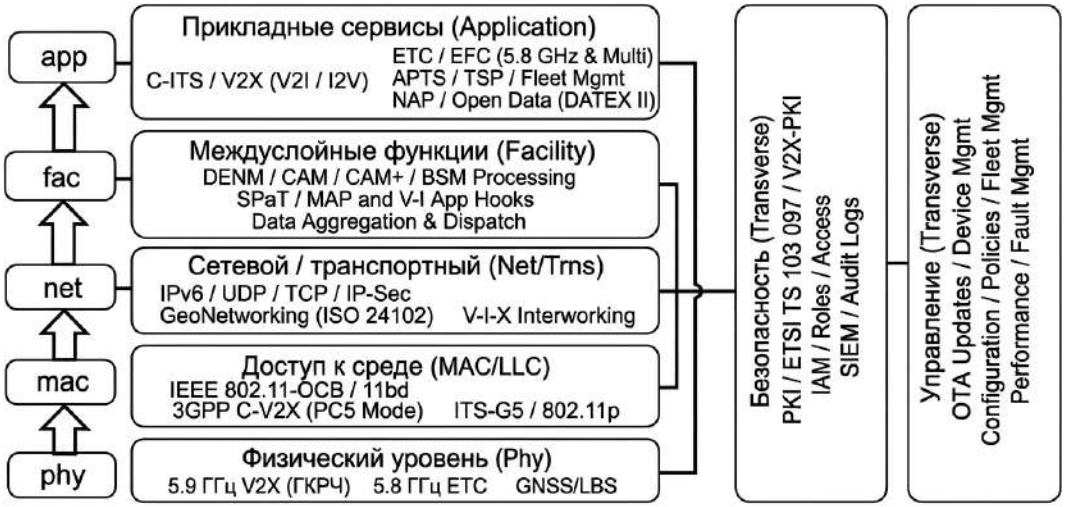

Рисунок 1.2 — ISO 21217. Логические уровни и функции безопасности/управления

Кооперативные сервисы обеспечивают восприятие окружающей обстановки за счет передачи сообщений форматов CAM, DENM и BSM. Сообщения содержат информацию о динамических событиях и состоянии транспортных средств. Информация о топологии и состоянии перекрестков, содержится в сообщениях SPaT и MAP.

Для состоятельности системы профиль кооперативных сообщений должен быть строго определён. Требуется согласованная частота публикации, единые

для всех участников правила синхронизации времени, ограничения на максимальный размер полезной нагрузки. Наиболее важно обеспечить однозначную семантику полей и допуски на отклонение между передаваемыми данными и фактическими состояниями светофорных фаз. Для согласованности в приложении А приведены полевые профили V2X, SPaT (SAE J2735), MAP (SAE J2735), DENM (ETSI EN 302 637-3). На рисунках 1.3 и 1.4 представлены общие сервисы данных и декомпозиция подсистем ИТС.

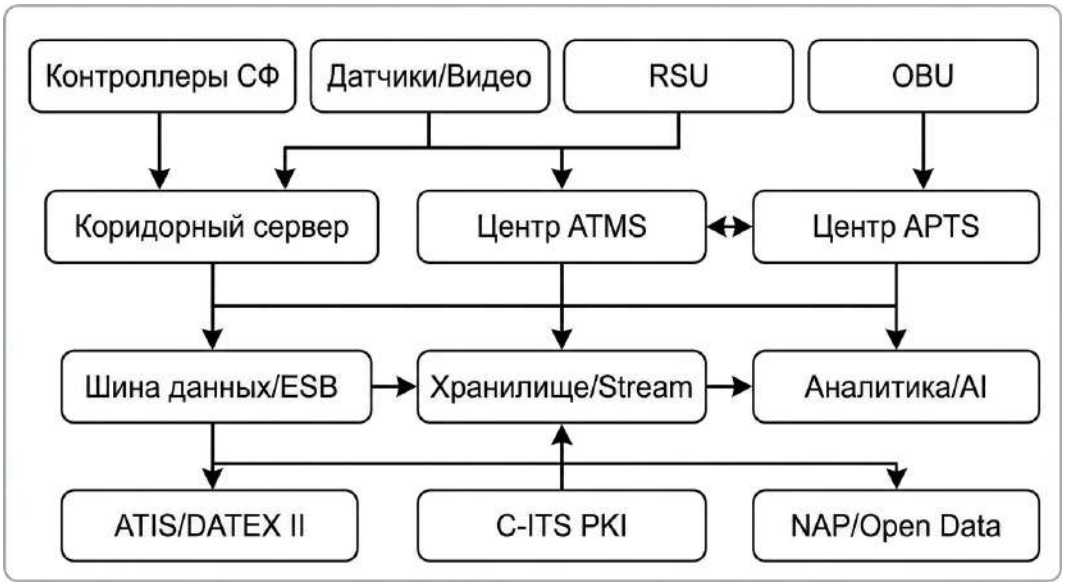

Рисунок 1.3 — Функциональная декомпозиция ATMS/APTS/C-ITS/ATIS и общие сервисы данных

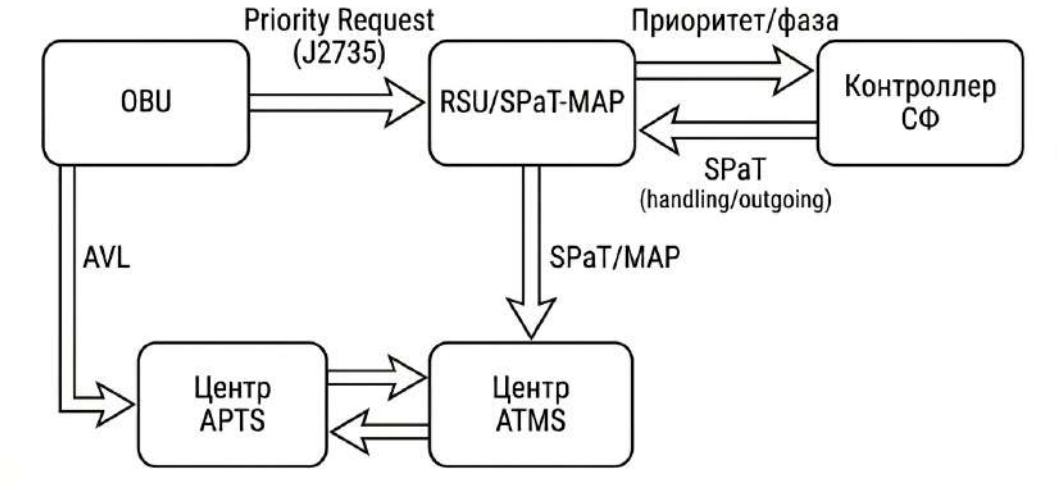

Рисунок 1.4 — Контур приоритета общественного транспорта

Информационный слой ИТС является основой для реализации подходов основанных на данных, к мониторингу, прогнозированию и управлению

транспортными потоками. В обзорных работах по большим данным в ИТС подчёркивается, что транспортные системы генерируют высокочастотные, пространственно распределённые и неоднородные данные, включающие показания датчиков, GPS-треки, видеопотоки, данные мобильных устройств, журналы светофорных контроллеров и сообщения кооперативных систем [95—97]. Для обработки таких данных применяются методы вычислительного интеллекта, потоковой аналитики, машинного обучения и оптимизации [98; 99].

В контексте умных городов ИТС рассматриваются как один из ключевых доменов применения больших данных и Интернета вещей. Исследования показывают, что интеграция IoT и аналитики данных позволяет формировать цифровое представление городской среды, выявлять аномалии, оценивать состояние инфраструктуры и поддерживать принятие решений в реальном времени [100—105]. В логистических и транспортно-операционных сценариях такие подходы также используются для повышения прозрачности процессов, сокращения задержек и улучшения управляемости цепочек поставок [106].

Практическая реализация информационного слоя требует специализированных платформ хранения и обработки транспортных данных. В работах [107; 108] рассматриваются облачные и мультикластерные аналитические платформы для обработки дорожных данных. Системы управления транспортными большими данными и фреймворки обработки потоков ориентированы на обеспечение масштабируемости, низкой задержки и воспроизводимости вычислений [109—111]. При этом для задач мониторинга и повышения безопасности на городских магистралях значимы не только агрегированные показатели движения, но и оперативное выявление опасных состояний транспортного потока [112; 113].

Отдельным вопросом является объединение данных из различных источников. Методы объединения данных позволяют интегрировать данные с видеокамер, радаров, индуктивных датчиков, GPS-треков и отчетов транспортных средств в единую оценку состояния транспортной сети [114]. Один источник данных не обеспечивает полного понимания ситуации на дорогах, особенно в условиях ограниченной видимости, отказов датчиков, потери информации и колебаний интенсивности движения.

Информационный слой, схематически изображенный на рисунке 1.5, обеспечивает непрерывность и воспроизводимость вычислений.Потоковые конвейеры преобразуют телеметрию и события в детерминированный поток данных, который доступен прикладным службам с минимальной задержкой.

Пакетная обработка данных объединяет справочные наборы данных и архивные данные. Системы хранения разделяют оперативные и исторические данные, позволяя сохранять долгосрочную историю без ущерба для оперативной обработки.

На более высоком уровне функционируют службы управления качеством данных, каталог метаданных и система отслеживания происхождения. Это обеспечивает контролируемую эволюцию данных, обратную совместимость интерфейсов и отслеживаемость версий при расчете ключевых показателей эффективности и подготовке отчетов о результатах внедрения.

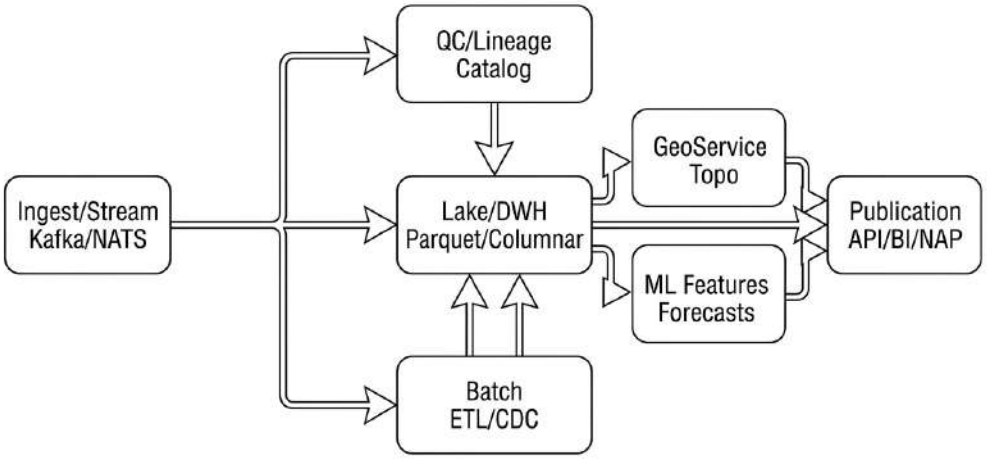

Рисунок 1.5 — Информационная архитектура

Семантическая совместимость обеспечивается строгим профилем его применения. DATEX II в дорожном центре должен быть согласован с локальными словарями и кодировками дорожных событий. SIRI и NeTEx должны быть согласованы с реальной структурой маршрутов, расписаний и точек остановок. GTFS согласован с политиками публикации, принятыми для внешних потребителей данных.

Сообщения CAM, DENM, SPaT и MAP должны быть согласованы по временной метке и геометрическому описанию перекрёстка и не содержать семантически противоречивых полей. Интерфейс между кооперативными системами V2X и автоматизированными системами управления дорожным движением должен быть устойчив к пропускам сообщений, дубликатам и сетевым джиттерам.

Сквозная синхронизация времени является краеугольным камнем всей архитектуры. Она начинается с источников GNSS на полевых устройствах, продолжается через протокол PTP в магистральной транспортной сети и завершается использованием NTP для сервисов, не требующих высокой временной точности.

Качество обслуживания в интеллектуальных транспортных системах является многомерным. Оно включает латентность, доступность, устойчивость к сбоям, предсказуемость, семантическую согласованность и обеспечение безопасности.

Для кооперативной связи инженерной целью является доставка защищённого кадра от дорожной станции к бортовому устройству за миллисекунды. Для локального управления требования ещё строже — обработка и реакция внутри светофорного контроллера должны укладываться в субсекундные интервалы. Коридорная координация должна сохранять устойчивость при нерегулярных задержках и пропусках данных. Публикации информационных сервисов требуют чётко задокументированных уровней доступности API и окон обслуживания.

Обобщённые ориентиры качества обслуживания и соответствующие им соглашения об уровне обслуживания приведены в таблице 3. В ней также учтены деградационные режимы. Переход на автономные безопасные планы при потере связи, кэширование списков отозванных сертификатов, резервирование каналов и другие процедуры поддержания базовой функциональности. Пример построения гибридной связи в ИТС представлен на рисунке 1.6.

Таблица 3 — Ключевые атрибуты качества и ориентиры SLA по доменам

Домен	Латентность (цель)	Доступность	Примечания по устойчивости
C-ITS/V2X	10–100 мс	Локальная, 99,9% на узле	Edge-обработка; кэширование CRL; fallback без V2X
ATMS локальный	10–50 мс	99,95% контроллера	Атомарность фаз; план безопасной деградации
ATMS коридор	0.5–2 с	99,5% центра	Буферы данных; асинхронные алгоритмы
APTS/TSP	0.5–2 с для TSP	99,5%	Дублирование каналов; альтернативные правила TSP
ATIS	10–60 с	99,5% API	Репликация; кэширование ответов

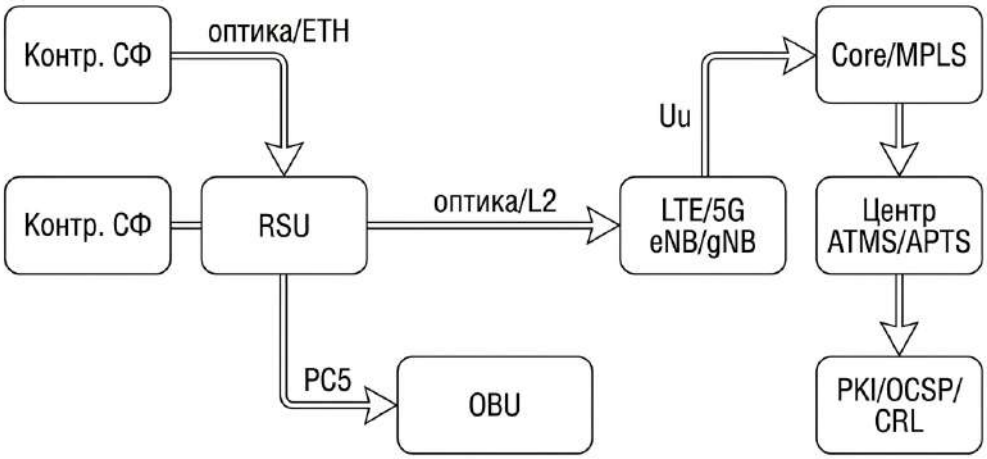

Рисунок 1.6 — Построение гибридной связи в ИТС

Доверие и безопасность в кооперативном домене строятся на стандарте ETSI TS 103 097 и использовании аппаратной защиты криптографических ключей. Над этим уровнем реализуются процессы по ISO/SAE 21434 и требования UN R155 и UN R156, касающиеся управления уязвимостями и жизненным циклом программного обеспечения.

Практически это предполагает развёртывание иерархической инфраструктуры открытых ключей: корневых и промежуточных центров сертификации, отзыва сертификатов, а также строгих политик выпуска, ротации и отзыва ключей. Конечные узлы должны быть защищёнными. Дорожные станции (RSU) оснащаются аппаратными модулями безопасности (HSM), а бортовые устройства (OBU) — доверенными платформенными модулями (TPM). Внутренний периметр центра управления обеспечивается системой управления доступом (IAM), сетевой сегментацией и сквозной наблюдаемостью всех операций. Структура доверия представлена на рисунке 1.7.

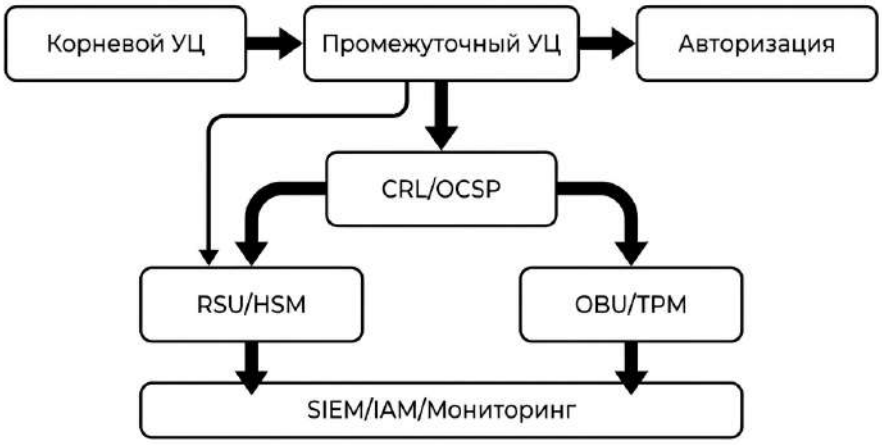

Рисунок 1.7 — Доверенная модель C-ITS

Масштабирование интеллектуальной транспортной системы от коридора к городу и региону требует чёткого распределения функций и предсказуемого поведения транспортной сети связи. Критические задачи, такие как кооперативная связь V2X и локальная координация светофоров, остаются на периферии. Функции оптимизации, аналитики, управления политиками доступа (NAP) и инфраструктуры открытых ключей (PKI) централизуются.

Согласование доменов с уровнями управления представлено в таблице 4. Оно служит основой для распределения эксплуатационной ответственности и установления допустимых сроков реакции.

Таблица 4 — Соответствие доменов, уровней управления, решений и интерфейсов

Домен	Уровень	Решения	Интерфейсы/данные
ATMS	Узел/коридор/город	Локальные планы, координация, спецрежимы	Контроллер API; ISO 19091; TMDD
APTS	Узел/город	TSP, диспетчеризация, прогноз прибытия	SIRI/NeTEx; AVL; SPaT/MAP
C-ITS	Устройство/узел	CAM/DENM, SPaT/MAP, BSM	ETSI EN 302 637; SAE J2735; ETSI 103097
ATIS	Город/регион	Публикации, предупреждения, прогнозы	DATEX II; ISO/TS 21219
ETC	Устройство/коридор	Транзакции	EN 15509; ISO 12855

Для обеспечения интероперабельности и предотвращения привязки к конкретным поставщикам применяется профилирование международных стан-

дартов. В таблице 5 показано, как фиксировать технические и семантические ограничения в независимой от вендора форме. На рисунке 1.8 представлена схема развертывания централизованной платформы сбора, обработки и хранения данных с оконечными устройствами.

Таблица 5 — Пример профилей интероперабельности по доменам

Домен	Стек/форматы	Ограничения профиля	Проверка соответствия
V2X/C-ITS	ETSI CAM/DENM; SAE SPaT/MAP; ETSI 103097	Ограниченный набор событий; кодировка PER/UPER; валидные координаты/проекции	PCAP/валидатор; журналы подписей; CRL/OCSP логи
ATMS	TMDD/NTCIP; ISO 19091	Фикс. словари состояний, синхронизация времени, интервалы публикации	Контроль расхождения фаз; симуляции
ATIS	DATEX II; TPEG	Локальный словарь, уровни детализации, SLA API	Валидация JSON; мониторинг SLA
APTS	SIRI/NeTEx; GTFS	Консистентность с AVL, правила опозданий	Сверка расписаний/факта

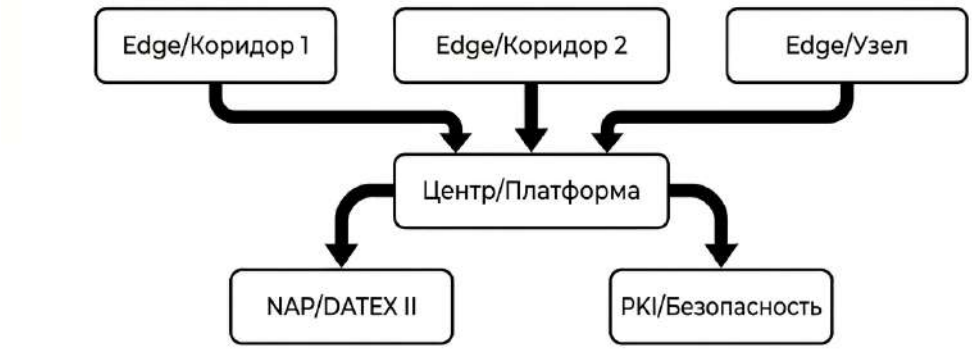

Рисунок 1.8 — Шаблон развертывания с распределенными узлами и центральной платформой

Алгоритм работы дорожной станции является важным компонентом всей системы. Он включает в себя считывание информации в реальном времени о состоянии участка дорожной сети, генерацию сообщений SPaT и MAP с

временными метками, привязанными к общему эталонному времени, криптографическую подпись сегмента данных и его передачу по радиоканалу. Одновременно телеметрические данные передаются в центр управления по различным каналам связи. Конвейер обработки данных, показанный на рисунке 1.9, охватывает все обязательные этапы обработки: синхронизацию времени, проверку данных и информационную безопасность. При этом периферийное устройство не требует избыточной логики. Все операции строго ограничены минимальной функциональностью, необходимой для корректной и безопасной работы.

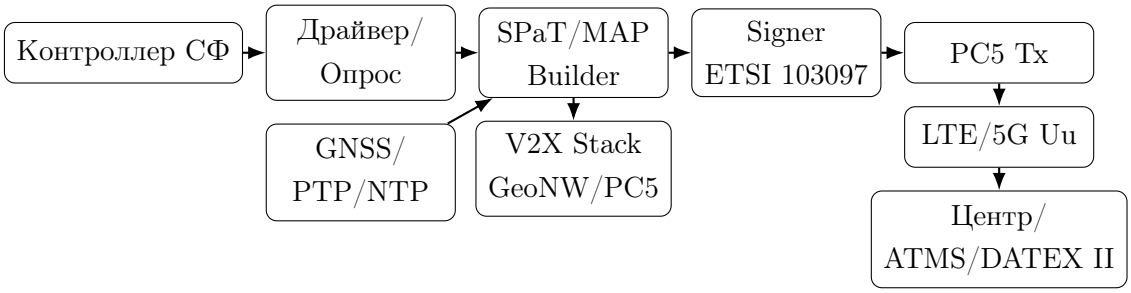

Рисунок 1.9 — Устройство RSU

Комплексная архитектура интеллектуальной транспортной системы — это не совокупность локальных оптимизаций, а строго профильная система, включающая в себя взаимосвязные подсистемы.

Иерархическая структура с минимальными, но строго соблюдаемыми соглашениями между уровнями позволяет разрабатывать технологии и модернизировать оборудование без ущерба для стабильности прикладных служб или целостности данных. Совместимые профили обмена сообщениями и надежная инфраструктура обеспечивают операционную совместимость и кибербезопасность. Интеграция с системами моделирования и цифровыми двойниками ускоряет проверку управленческих гипотез и снижает риски при масштабировании решений.

Учёт решений Государственной комиссии по радиочастотам в части использования полосы 5,9 ГГц, требований правил дорожного движения и ГОСТ к визуальной организации движения, ограничений федерального закона № 152-ФЗ «О персональных данных» и требований закона № 187-ФЗ «О безопасности критической информационной инфраструктуры Российской Федерации» делает такую архитектуру не только инженерно состоятельной, но и юридически корректной.

Международный уровень устанавливает архитектурные и коммуникационные основы интеллектуальных транспортных систем. Стандарты ISO формируют базовую структуру совместимости. К ним относятся ISO 14813 (идентичный российский национальный стандарт ГОСТ Р ИСО 14813-1-2011), определяющий референсные модели ИТС, ISO 21217 (на его основе разработан национальный стандарт Российской Федерации ПНСТ 461–2020) и ISO 24102 (и другие версии стандарта ISO 24102-1:2013, ISO 24102-2:2015, ISO 24102-3:2013, ISO 24102-4:2013, ISO 24102-5:2013), задающие архитектуру и управление станциями ИТС. Стандарт ISO 19091 (на момент написания стандарт ISO/AWI TS 19091 находится в разработке), регулирующий взаимодействие с объектами регулирования и форматы сообщений SPaT и MAP. Стандарт ISO 17572, описывающий схемы ссылочного местоположения, не определяет физический формат для реализации методов указания местоположения, но устанавливает требования к физическим форматам. Эти документы обеспечивают согласованность между полевыми устройствами, центрами управления и транспортными средствами.

Стандарты IEEE и SAE охватывают стек кооперативных сообщений и радиодоступа. IEEE 1609.x и IEEE 802.11-OCB определяют протоколы WAVE и ITS-G5. SAE J2735 и SAE J2945 задают семантику и требования к производительности для сообщений BSM, SPaT, MAP и других типов данных. Кооперация через сотовые сети регулируется спецификациями 3GPP TS 23.285 и TS 23.287, а также стандартами бокового интерфейса PC5.

Европейский набор стандартов ETSI и CEN дополняет архитектуру на транспортном и прикладном уровнях. ETSI EN 302 571 и EN 302 663 определяют физический и канальный уровни ITS-G5. EN 302 637-2 и EN 302 637-3 регламентируют форматы сообщений CAM и DENM. TS 102 636 задаёт правила геосетевого взаимодействия. Стандарт TS 103 097 устанавливает требования к инфраструктуре управления ключами и сертификатам для C-ITS. Обмен дорожной и пассажирской информацией обеспечивается стандартами CEN DATEX II, EN 15531 (SIRI) и CEN/TS 16614 (NeTEx). Электронное взимание платы регулируется стандартами ISO 12855 и EN 15509.

Американская модель опирается на архитектуру ARC-IT и семейство протоколов NTCIP. Эти стандарты обеспечивают совместимость полевых устройств — контроллеров светофоров, дорожных метеостанций, табло переменной информации с центрами управления. Они также определяют словари

данных, такие как TMDD, и правила визуального оформления средств организации дорожного движения, закреплённые в руководстве MUTCD.

Российская нормативно-правовая база сосредоточена на безопасности дорожного движения, техническом регулировании транспортных средств, обработке данных, защите информации, а также на инфраструктуре связи и радиочастотном обеспечении.

Основу правового регулирования составляют федеральные законы. К ним относятся закон № 196-ФЗ о безопасности дорожного движения, № 257-ФЗ об автомобильных дорогах и дорожной деятельности, № 16-ФЗ о транспортной безопасности, № 149-ФЗ об информации и защите информации, № 152-ФЗ о персональных данных, № 187-ФЗ о безопасности критической информационной инфраструктуры, № 126-ФЗ о связи, № 184-ФЗ о техническом регулировании и № 162-ФЗ о стандартизации.

Обязательные требования к средствам организации дорожного движения закреплены в Правилах дорожного движения, утверждённых постановлением Правительства РФ № 1090, и в гармонизированном своде национальных стандартов. Среди них — ГОСТ Р 52289 и ГОСТ Р 52290 по применению дорожных знаков, ГОСТ Р 52282 по светофорам, ГОСТ Р 52286 по горизонтальной разметке и ГОСТ Р 50597-2017 по эксплуатационному состоянию дорог и улиц.

На стыке с автомобильной промышленностью применяется технический регламент ТР ТС 018/2011 ЕАЭС «О безопасности колёсных транспортных средств». Он включает требования к системе экстренного реагирования ЭРА-ГЛОНАСС, детализированные в ГОСТ Р 55534-2013. Эти положения влияют на профили бортовых устройств и интерфейсы связи.

В сфере защиты информации и персональных данных действуют постановление Правительства РФ № 1119 и приказы ФСТЭК России № 17 и № 21. Для объектов критической информационной инфраструктуры применяются требования закона № 187-ФЗ и соответствующие методики категорирования и защиты.

Содержательная интеграция международных стандартов и российских нормативно-правовых актов требует явного сопоставления по сферам регулирования. В сводной форме в таблице 6, где каждая строка соотносит домен требований с источниками на международном, региональном и национальном уровнях и поясняет способ связки и условия применимости.

Таблица 6 — Сопоставление ключевых сфер регулирования ИТС: международный, европейский, американский и российский уровни

Сфера	**Международный уровень**	**Европейский контекст**	**США**	**Российская база и интеграция**
Безопасность дорожного движения и средства ОДД	ISO 19091 (V2I), ISO 14813 (референсные модели)	Директива 2010/40/EC (рамка ИТС)	MUTCD (внешний вид/применение средств ОДД)	ПП РФ № 1090; ГОСТ Р 52289-52290; интеграция V2I через локализацию SPaT/MAP
Техническое регулирование ТС и eCall	WP.29 UN R157/ALKS; ISO 26262; SOTIF	Регламент (EU) 2015/758 (eCall)	FMVSS/NCAP; OEM-процессы безопасности	ТР ТС 018/2011; ГОСТ Р 55534-2013 (ЭРА-ГЛОНАСС); адаптация V2X
Кибербезопасность и обновления ПО	UN R155 и UN R156; ISO/SAE 21434	C-ITS PKI профили ETSI TS 103 097	NHTSA/SAE; OEM CSMS/SUMS	187-ФЗ (КИИ), 149-ФЗ; локальная политика ИБ; PKI C-ITS
Персональные данные и приватность	ISO/IEC 27001/27002/27005 (ИСМБ)	GDPR и DPIA-практики	Штатные правила, NIST Privacy Framework	152-ФЗ; ПП РФ № 1119; приказы ФСТЭК № 17/21; анонимизация данных
Архитектура/ интероперабельность ИТС	ISO 21217/24102; ISO 14813	ETSI C-ITS Cooperative Stack	ARC-IT; NTCIP; TMDD	Применение международных моделей; мэппинг интерфейсов к ГОСТ; совместимость через шлюзы
Данные о дорогах и ОТ	DATEX II; TPEG	SIRI/NeTEx; национальные точки доступа	GTFS; GTFS-realtime	Рекомендательное применение DATEX II, SIRI/NeTEx, GTFS; локальные API
Электронное взимание платы	ISO 12855; интероперабельность клиринга	EN 15509; EETS-профили	5.8 DSRC и OEM-решения	257-ФЗ; эксплуатационные профили 5.8 ГГц; совместимость с ETC

Продолжение на следующей странице

Сфера	**Международный уровень**	**Европейский контекст**	**США**	**Российская база и интеграция**
Радиочастоты и спектр для V2X	ITU-R рекомендации; 3GPP PC5	EN 302 571; 5.9 ГГц ITS-G5	5.9 ГГц для C-V2X	Решения ГКРЧ по 5.9 ГГц; требования ЭМС и регистрации РЭС
Эксплуатация и мониторинг	ISO 14813; RAMS-подходы	Руководства C-ITS Deployment	Руководства FHWA/ITE	Методические рекомендации Минтранса и Росавтодора по ИТС/АСУДД

Сопоставление показывает, что обязательные требования в российской практике сосредоточены в нескольких ключевых областях. К ним относятся безопасность дорожного движения и визуальное оформление средств организации дорожного движения, техническое регулирование транспортных средств, защита персональных данных и объектов критической информационной инфраструктуры, а также радиочастотное регулирование кооперативных систем связи.

Международные стандарты, касающиеся архитектуры, форматов сообщений и моделей данных, применяются в качестве рекомендательных ориентиров. Они служат основой для обеспечения интероперабельности и повышения качества реализации систем. Их адаптация на местном уровне осуществляется через проектные профили и соглашения о форматах обмена данными. Такие профили разрабатываются с учётом требований российских нормативных документов и не противоречат ГОСТ и Правилам дорожного движения.

Практическая совместимость интеллектуальных транспортных систем основана на согласованных моделях данных. Поэтому целесообразно увязывать домены сообщений с их нормативным статусом и операционной применимостью.

Для кооперативных сообщений безопасности выделяются две параллельные ветви: ETSI CAM/DENM и SAE/IEEE BSM/SPaT/MAP. В центрах управления и на шлюзах необходимо обеспечить корректное сопоставление полей и единообразное толкование событий, источников времени и геопривязки.

Ниже представлена сводная таблица 7 ключевых групп данных, соответствующих стандартов, областей применения и замечаний по их статусу в Российской Федерации.

Таблица 7 — Модели данных и сообщения в ИТС

Область	Стандарт/профиль	Тип применения	Статус и примечания в РФ
Кооперативная безопасность V2X	ETSI CAM/DENM; ETSI TS 103 097 (PKI)	Обмен сообщениями безопасности и событий	Рекомендательно при развертывании C-ITS; требует соответствия решениям ГКРЧ и политики PKI оператора
Фазы и геометрия перекрёстков	SAE J2735 SPaT/MAP; ISO 19091 (V2I)	Приоритет ОТ, информирование ТС, координация фаз	Актуально для АСУДД; мэппинг в центр с учётом ПДД и ГОСТ по светофорам
Базовые сообщения ТС	SAE J2735 BSM	Кооперативная телеметрия	Возможна поддержка через шлюзы; вопросы ПДн решаются обезличиванием
Дорожные события и состояния	CEN DATEX II	Межведомственный обмен	Применимо для региональных/федеральных систем; адаптация под локальные справочники
Пассажирская информация	CEN EN 15531 (SIRI); CEN/TS 16614 (NeTEx); GTFS	Диспетчеризация и публикация расписаний	Используется де-факто; нормативно не закреплено, но совместимо с отечественными практиками
Вещание дорожной информации	ISO/TS 21219 (TPEG)	Широковещательные сервисы	Используется операторами сервисов; прямых национальных ограничений нет
Электронное взимание платы	ISO 12855; EN 15509; CEN DSRC 5.8	Клиринг, интерфейсы ETC	Соответствует правовому режиму платных дорог; спектральный профиль 5.8 ГГц сложившийся

Ключевым архитектурным ограничением для кооперативных интеллектуальных транспортных систем является радиочастотная доступность. Европейский подход основан на стандарте ETSI EN 302 571 и использует полосу 5,9 ГГц для технологий ITS-G5. Американская регуляторная практика после перераспределения частотного ресурса преимущественно ориентирована на C-V2X в верхней части того же диапазона.

С учётом изложенного, практическая стратегия согласования нормативной базы при реализации проекта интеллектуальной транспортной системы строится по принципу профилирования сверху вниз.

На первом этапе фиксируются юридически обязательные рамки. К ним относятся Правила дорожного движения и национальные стандарты по средствам

организации дорожного движения, требования законодательства в области обработки персональных данных и защиты информации, правила использования радиочастот и регистрации радиоэлектронных средств согласно решениям Государственной комиссии по радиочастотам, а также применимость технических регламентов Таможенного союза и требований к системе ЭРА-ГЛОНАСС.

На втором этапе выбирается технологический профиль кооперативной связи — ITS-G5, C-V2X или гибридный вариант, совместимый с выделенными поддиапазонами, ограничениями по мощности и условиями электромагнитной совместимости.

Далее определяются модели интеграции данных. Для передачи информации о дорожных событиях и состоянии инфраструктуры применяется DATEX II. Для сведений об общественном транспорте используются SIRI и NeTEx, а при внешней публикации — GTFS. На кооперативном уровне задействуются сообщения SPaT, MAP, CAM, DENM и BSM. Для всех форматов устанавливаются правила сопоставления полей, источники синхронизации времени и политики управления качеством данных.

На завершающем этапе закрепляются методики оценки эффективности и эксплуатационные регламенты. Они согласуются с методическими рекомендациями Министерства транспорта и Росавтодора, что обеспечивает воспроизводимость результатов и их сопоставимость между различными территориями.

В совокупности нормативно-правовая база интеллектуальных транспортных систем, оформленная в виде связного профиля требований, обеспечивает юридическую корректность, интероперабельность и системное управление рисками на всех этапах жизненного цикла.

1.2 Иерархия и уровни управления в ИТС

Методы управления в ИТС эволюционируют от классических алгоритмов адаптивного регулирования к моделям, использующим машинное обучение, глубокое обучение и обучение с подкреплением. Обзор литературы посвященной применению обучения с подкреплением в ИТС показывает, что методы глубокого обучения с подкреплением применяются для управления светофорными объектами, маршрутизации, координации транспортных потоков,

управления спросом и оптимизации работы транспортной инфраструктуры [115]. Их преимущество состоит в способности учитывать нелинейные зависимости и долгосрочные последствия управляющих действий, однако практическое применение требует безопасного контура исполнения, ограничений на допустимые действия и верификации поведения в симуляционной среде.

В задачах управления движением также применяются нейросетевые и гибридные методы обработки сигналов, включая нейронные сети для управления трафиком в условиях высокоскоростных транспортных систем и беспроводной связи [116]. Для автоматизированных транспортных средств и систем автономного движения важным направлением является генерация тестовых сценариев, поскольку качество проверки управляющих алгоритмов определяется полнотой, разнообразием и реалистичностью ситуаций, используемых при испытаниях [117]. Это позволяет связать задачи управления ИТС с задачами верификации, стресс-тестирования и оценки безопасности автоматизированных систем.

Иерархия управления транспортными потоками представляет собой структурированный способ организации причинных связей между мониторингом, прогнозированием и управляющим воздействием в пространственно-временном континууме.

На нижнем уровне иерархии осуществляется непосредственное восприятие состояния УДС и первичная интерпретация потоковых данных. Фиксируются локальные очереди, скорости на элементарных участках сети, события на границах зон (въезды, выезды, локальные затруднения, инциденты), а также микроскопические траектории отдельных транспортных средств.

В рамках отдельного перекрёстка формируются локальные управляющие решения. К таким решениям относятся карта фаз светофорного цикла, информация о приоритетах, критерии перенастройки фаз. В то время как коридорный уровень настраивается исходя из агрегированной информации нескольких перекрёстков. В том числе решается задача синхронизации управляющих сигналов на УДС в целом. Основная цель функционирования данного уровня минимизация очередей и сохранение устойчивости потока. Сетевой уровень позволяет управлять распределением спроса по всей рассматриваемой территории на основе макроскопической фундаментальной диаграммы. Благодаря этому возможно управлять скоростными режимами и ограничением входных потоков. Рассматриваемая декомпозиция функций по уровням позво-

ляет объединять локальные и глобальные правила управления транспортными потоками. Локальные правила являются атомарными и предсказуемыми на коротких временных горизонтах, что позволяет осуществлять оперативную реакцию на состояние сети. Глобальные правила позволяют сохранять устойчивость системы в целом, и при этом не нарушать локальной динамики.

В рамках описанной иерархии мониторинг представляет собой активный процесс фильтрации, калибровки и приведения разнородных источников к единому операционному базису. Компьютерное зрение обеспечивает плотные поля детекций и траекторий транспортных средств с их атрибутами, магнитометрические и индукционные датчики дают устойчивые оценки интенсивности и занятости проезжей части, радарные и лидарные системы измеряют скорости и интервалы с высокой пространственной разрешающей способностью, а бортовая телеметрия дополняет картину микроскопическими траекториями и показателями динамики. Ни один из перечисленных источников не обладает исчерпывающей надёжностью, поэтому практическая архитектура мониторинга строится на принципах слабонаблюдаемой многомодальности, где каждое измерение сопровождается калибровочной функцией и явной моделью неопределённости.

Конвейер мониторинга начинается с семантического извлечения признаков на периферийных устройствах, после чего включаются механизмы контроля качества, охватывающие доверительные интервалы для оценок скорости, вероятности корректной ассоциации трека с физической целью и метрики согласованности показаний счётчиков. На заключительном этапе формируется пространственно-временной граф, в котором вершины соответствуют узлам и сегментам улично-дорожной сети, рёбра отражают физические и логические связи между ними, а состояния вершин представлены векторами признаков, агрегированными на нескольких временных масштабах. Такая структура служит единым информационным фундаментом для всех последующих уровней прогнозирования и управления.

В задаче мониторинга требуется извлечение и нормализация признаков из различных сенсоров. В условиях городской УДС мониторинг осложняется такими факторами, как частичные перекрытия объектов, резкие изменения освещённости, блики на поверхности проезжей части и высокая, неравномерная плотность транспортного потока. Для обеспечения устойчивости применяется комбинированный подход, объединяющий детекцию объектов, многократное

сопоставление траекторий и реидентификацию транспортных средств при переходе между зонами видимости смежных сенсоров.

Современные детекторы основаны на свёрточных и трансформерных архитектурах, однако их выходные данные должны быть привязаны к координатам дорожной топологии. Для этого используются методы самокалибровки камер по внешним ориентирующим точкам. Дополнительно применяется обучение без учителя на длительных видеопоследовательностях, что позволяет корректировать проекционные матрицы до уровня погрешностей, не влияющих на корректность определения маршрутной принадлежности траекторий.

Прямая связь между визуальными характеристиками и топологией сети устанавливается путем сведения полных траекторий к ключевым поворотным точкам и последующего сравнения полученных последовательностей с шаблонами допустимых маневров на определенном перекрестке. Это свойство позволяет реконструировать схемы движения при неполных данных, что происходит в периоды ограниченной видимости. Данное свойство приводит к повышению надежности прогнозов относительно макроскопических характеристик и очередей в отдельных узлах.

Для реализации описанного подхода осуществляется непрерывный мониторинг показателей достоверности и метрик. К ним относятся доля правильных сопоставлений траекторий между кадрами и стабильность показаний счетчика на УДС при различных погодных условиях и переменных нагрузках. На данном этапе решается задача согласования оценок, полученных с помощью видеодатчиков и радиолокационных систем. Согласование оценок между датчиками различной модальности критично в зонах с перекрытиями. При активации триггера система переключается в деградационный режим работы. В этом режиме частота обновлений состояния снижается, семантика упрощается, окна агрегации информации расширяются, а механизмы управления, чувствительные к шуму, отключаются.

Таким образом ИИ компоненты нижнего уровня не должны подменять физические ограничения объекта управления, нормативно-правовые регламенты и требования функциональной безопасности. Их назначение состоит в повышении наблюдаемости и информационной насыщенности системы за счёт извлечения дополнительных признаков и оценок состояния. Однако исключительно в пределах строго определённых доверительных зон, границы которых обосновы-

ваются текущими показателями качества исходных данных и подтверждаются устойчивостью используемых моделей.

Важным следствием описанной организации мониторинга является возможность формирования единых, структурированных репрезентаций, пригодных для последующих задач прогнозирования и управления. Если каждый узел и каждый сегмент графа УДС представлен компактным вектором признаков, включающим текущие оценки интенсивности потоков, средних скоростей, параметров светофорных циклов, остаточной пропускной способности, а также метки неопределённости и контекстуальную информацию (погодные условия, массовые мероприятия, текущие режимы работы), то вся система становится совместимой с классом пространственно-временых моделей. Такие модели способны масштабироваться от локальных участков в пределах квартала до крупных городских агломераций без потери семантической целостности.

В данном контексте иерархия выполняет двойную функцию. Она не только разделяет уровни принятия управляющих решений, но и структурирует уровни абстракции данных. На нижних слоях репрезентации сохраняют высокое временное и пространственное разрешение, отражая микроскопическую динамику сети. По мере продвижения вверх признаки становятся всё более агрегированными и статистически устойчивыми. Временные окна расширяются, пространственные единицы обобщаются, а шум подавляется за счёт усреднения и фильтрации. Вместе с тем, на верхних уровнях возрастает требование к строгости гарантий согласованности агрегирования. Это означает необходимость сохранения физического смысла величин, и с позиции математической корректности перехода от микроскопического описания к макроскопическому. Схема иерархии управления представлена на рисунке 1.10.

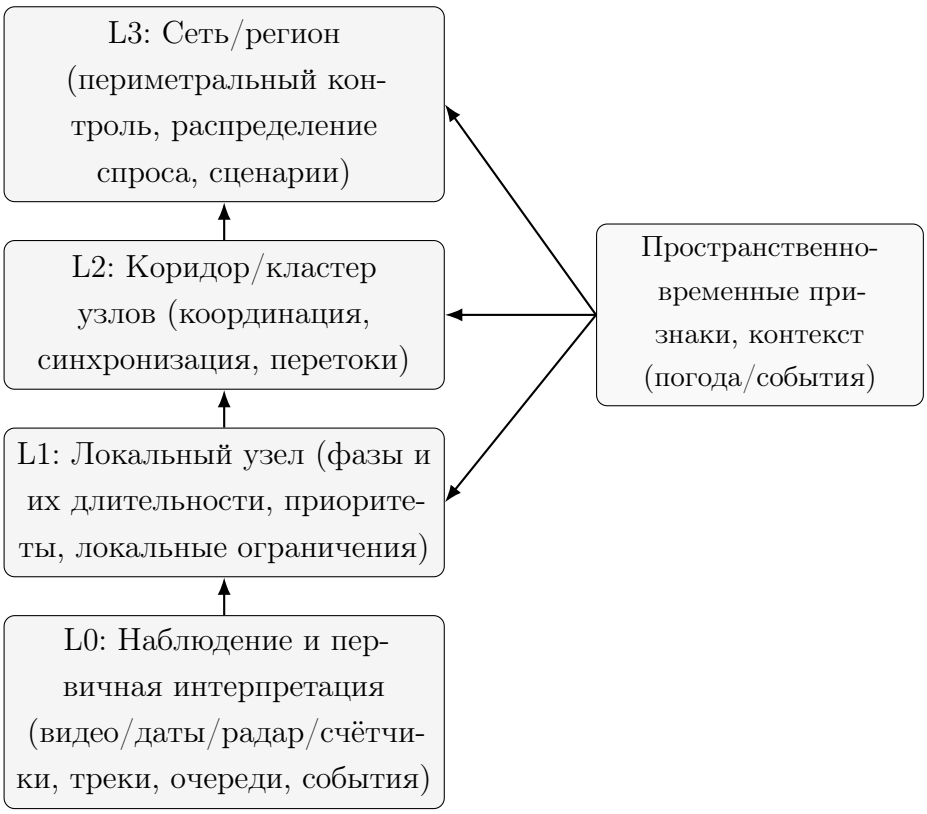

Рисунок 1.10 — Иерархиция управления

С практической точки зрения ключевой задачей нижнего уровня мониторинга является формирование чистого временного ряда измерений и событий, пригодной для обучения, верификации и адаптации моделей на более высоких уровнях иерархии. Примерами служат генерация псевдо-меток плотности потока вдоль коридора на основе согласованности показаний индукционных счётчиков и видеотрекинга, реконструкция скоростей по парам временных меток проезда одного и того же транспортного средства в смежных полях зрения камер, а также сопоставление оценок длины очередей с наблюдаемыми задержками для калибровки локальных моделей накопления и рассеяния очередей.

Вся информация передаётся вверх по иерархии в согласованной форме, обеспечивая тем самым, что последующие этапы прогнозирования и управления оперируют вероятностными представлениями о состоянии сети, а не точечными оценками. Вероятностное представление принципиально для принятия надёжных решений в условиях неопределённости.

Прогноз служит мостом между мониторингом и управляющим воздействием, причём его задача состоит в явном представлении неопределённости и возможных сценариев на горизонтах от минут до часов, что позволяет регуляторам принимать решения с учётом рисков. Мировая практика использует

модели, объединяющие топологию и динамику сети в единой пространственно-временной структуре: сеть задаётся графом, каждой вершине и ребру которого сопоставлен временной ряд признаков, а динамика моделируется свёртками по графу и по времени либо механизмами внимания с маскированием по топологии. Среди таких моделей выделяются диффузионные с рекуррентным или свёрточным временем, спектральные и пространственные графовые свёртки, а также трансформеры с иерархией масштабов, позволяющие выявлять нелинейные связи между элементами сети и учитывать внешний контекст.

Для практического применения прогнозов необходимы моделирование распределений ошибок и переносимость моделей между участками сети. Универсальных методов мониторинга и прогнозирования не существует, поэтому надёжность обеспечивается систематической калибровкой, ансамблями архитектур и регуляризацией параметров, снижающей избыточную уверенность при малом объёме данных. Физическая согласованность прогнозов достигается мягким внедрением законов сохранения плотности и потока в качестве ограничений при обучении нейросетей, что снижает риск аномальных предсказаний. Альтернативный путь предлагает ассимиляция данных через фильтры Калмана и их нелинейные расширения, удерживающие модель близко к реальности даже при резких изменениях спроса.

Раннее обнаружение значимых событий, таких как дорожно-транспортные происшествия или всплески спроса, требует гибридных методов, поскольку редкие явления плохо моделируются классическими подходами из-за дисбаланса и разнообразия форм. Практические системы сочетают нелинейные детекторы новизны с генеративными моделями, формирующими распределение возможных сценариев, предоставляя операторам дополнительное время для принятия решений. Перенос моделей между узлами и городами решается через метаобучение и адаптацию к домену, при которых модель обучается на данных множества коридоров с регуляризацией на инварианты, включающие топологические признаки и нормированное время проезда. Данный подход позволяет проводить незначительное дообучение модели на целевом участке по малой выборке и продолжать дообучение по мере накопления данных.

На коридорном и сетевом уровнях целесообразно использовать многоуровневые предсказатели. Нижние слои работают на горизонтах секунд–минут, средние на 15–30 минут, верхние на час и более. Такая структура позволяет согласовать прогноз с управлением. Локальные регуляторы опираются на

ближайшие горизонты, коридорные на средние, а сетевые политики на долгосрочные сценарии. Пример данной методологии представлен на рисунке 1.11.

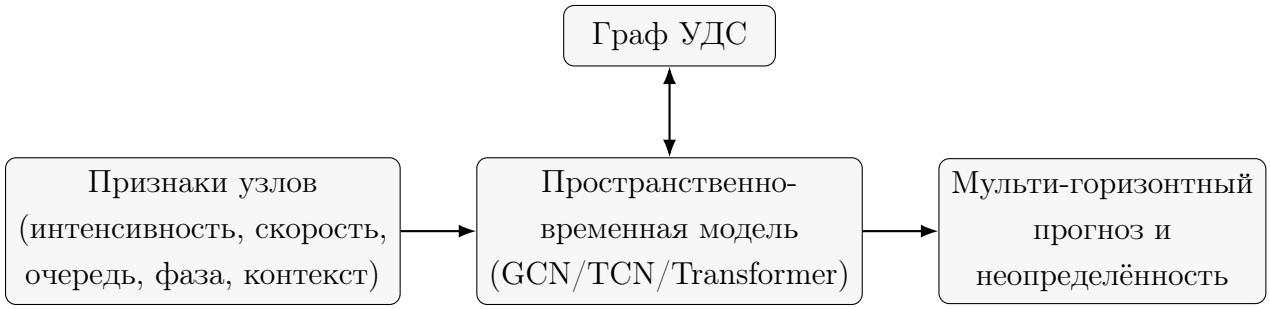

Рисунок 1.11 — Методология прогноза от признаков к пространственно-временным моделям и мульти-горизонтным предсказаниям с неопределённостью

Вопрос калибровки и верификации в конечном счёте сводится к качеству данных. Оценка времени поездки по телеметрическим данным требует точной привязки к дороге и устранения систематических смещений. Истинные значения длины очередей формируются путём совмещения визуальных оценок, данных локальных датчиков и журналов светофорных контроллеров.

В хорошей практике такие данные собираются как эталонные подвыборки с чёткой методологией отбора и обработки. Их используют для регулярной проверки и донастройки прогнозных моделей. Прогноз без калиброванной шкалы ведёт к избыточной уверенности регулятора, что в транспортных системах неприемлемо.

Иерархия мониторинга и прогноза отвечает на вопрос о текущем состоянии и вероятном будущем. В то время как иерархия управления определяет допустимые и предпочтительные управляющие воздействия. Локальный уровень обеспечивает оперативную устойчивость системы.

Управление светофорными циклами формально сводится к дискретно-непрерывной оптимизации. Выбираются последовательность фаз и их длительности при соблюдении ограничений на безопасность, минимальные пешеходные интервалы и локальные регламенты. В стационарных условиях эффективны детекторно-управляемые актюаторы, псевдостохастические алгоритмы и локальный прогнозирующий контроллер. При нестабильном спросе и высокой вариативности потоков предпочтение отдают адаптивным политикам. К ним относятся стратегия максимального давления, минимизация взвешенных очередей и их градиентов, а также обучение с подкреплением для оптимизации совокупной пропускной способности и надёжности времени проезда.

На коридорном уровне управление становится координацией. Синхронизируются сдвиги фаз, ширина окон зелёной волны и распределяются буферы на узлах для поглощения колебаний притоков.

Существуют два основных подхода. Среди подходов к управлению транспортными потоками предлагаем рассматривать два направления, различающихся по природе оптимизационного механизма. Первое опирается на прогнозирующее управление с явным учётом неопределённости и вероятностных ограничений, т.е. система заранее просчитывает сценарии и удерживает решение в допустимой области. Второе строится на иерархическом обучении с подкреплением, при котором агент верхнего уровня настраивает параметры локальных политик, не выходя за пределы заданных безопасных границ.

Гибкость второго направления объясняется способностью учитывать нелинейные эффекты, которые жёсткая оптимизация либо игнорирует, либо аппроксимирует с потерей точности. Правда, гибкость требует дополнительного контура безопасности, включающего фильтры недопустимых комбинаций фаз и штрафы за приближение к перегрузкам. Штрафы формулируются через чувствительные критерии, в частности через условное значение риска по времени проезда, что позволяет системе реагировать на угрозу перегрузки до её наступления.

Оба направления преследуют одну цель: удерживать сеть вблизи точки оптимальной загрузки, где пропускная способность достигает максимума, а вариативность времени проезда остаётся управляемой. В операционной практике выбор между прогнозирующим управлением со сценарным прогнозом и иерархической многоагентной системой определяется требованиями к скорости реакции и доступностью данных. В многоагентной архитектуре верхние уровни задают целевые пропускные способности и рабочие точки, нижние обеспечивают локальную реакцию, и плотная связь между агентами при этом не требуется.

Достаточным оказывается обмен агрегированными показателями: очередями и плотностями на границах зон, укрупнёнными прогнозами притоков. Ограничение объёма передаваемой информации снижает чувствительность системы к неточностям прогнозных данных, т.е. разреженная коммуникация здесь работает как встроенный механизм робастности.

Отдельного внимания заслуживает контур безопасности, без которого любой адаптивный регулятор остаётся инженерно незрелым. Контур проверяет инварианты: исключает конфликтующие сигналы, соблюдает минимальные пе-

шеходные интервалы, ограничивает производные длительностей фаз и сдвигов между циклами заданными порогами. Дополнительно реализуются механизмы подавления избыточной цикличности, возникающей из-за шума в потоках, а также функции аварийного отключения и ручного управления.

Интерпретируемость системы обеспечивается через локальные линейные аппроксимации в окрестности текущего режима и через механизмы внимания с визуализацией вклада признаков. Постфактум-методы оценки важности, среди которых наиболее распространены SHAP и интегральные градиенты, позволяют операторам верифицировать логику принятых решений без доступа к внутренним параметрам модели. Для операторов особенно полезны простые макроиндикаторы. Описанную методологи можно представить в виде схемы на рисунке 1.12.

Рисунок 1.12 — Методология выбора и обучения политики в цифровом двойнике и безопасное исполнение с обратной связью из реальности

На практике управление ограничено вычислительными ресурсами и временными задержками. Локальные решения принимаются за доли секунды. Координация требует нескольких секунд или десятков секунд. Сетевые решения формируются в течение минут.

Вычисления распределяются между георасределенными серверами. Локальные агенты используют лёгкие модели. К ним относятся градиенты очередей, линейные предикторы и компактные управляющие политики. На коридорном уровне применяются агрегированные прогностические блоки. На сетевом уровне работают сценарные оптимизаторы.

В эту структуру встроен приоритет общественного транспорта. На коротком горизонте фазы корректируются на основе прогнозов прибытия, полученных из треков или моделей движения. Эти корректировки ограничены безопасными диапазонами и допустимым уровнем нарушения координации на коридоре. В среднесрочной перспективе маршрутные планы общественного

транспорта согласуются так, чтобы приоритеты не конфликтовали и не нарушали баланс потоков на пересечениях.

Отдельно выделяются методы оффлайн-обучения и инициализации. Поведение операторов и существующих систем управления содержит ценную информацию о допустимых манёврах и устойчивых паттернах. Эту информацию можно извлечь и использовать для старта через обучение на демонстрациях. Далее модель дообучают на симуляциях и в реальном времени по схемам, с введением новых действий.

Оффлайн-обучение с подкреплением по журналам контроля позволяет строить политики без вмешательства в работающие процессы. Это возможно при условии корректной оценки поведенческой политики и учёта рисков.

В сочетании с формальной верификацией фаз такие подходы дают то, что требуется от интеллектуального управления в городских условиях. Целью таких методов является устойчивое улучшение работы системы при соблюдении всех физических, операционных и нормативных ограничений.

Мониторинг включает не только потоки телеметрии, но и согласованную с архитектурой методологию проверки гипотез и приёмки решений.

На нижнем уровне осуществляется мониторинг качества и стабильности извлеченных признаков. Система оценивает долю обнаружений и стабильных траекторий, согласованность скоростей и очередей в зонах перекрытия датчиков, а также частоту пропущенных обнаружений и ложных срабатываний при обнаружении инцидентов.

На уровне управления отслеживаются задержки в принятии решений. Контролируется уровень соответствия пределам безопасности. Здесь регистрируют частоту срабатывания защитных цепей, и измерение устойчивости системы к шуму во входных данных. В конечном итоге оцениваются результаты. К ним относятся медианное и процентильное время в пути, изменчивость, надежность времени в пути (TTTR), пропускная способность, показатели безопасности, такие как TTC (время до столкновения) и PET (время после нарушения границ), а также экологические показатели, включая интенсивность выбросов, рассчитанную на основе профилей скорости и ускорения.

Верификация требует корректного дизайна экспериментов. Сравнение до и после оправдано на локальных участках при стабильных внешних условиях и наличии надёжных контрольных точек. На коридорах применяют метод разностей, чтобы отделить эффект от общих трендов и сезонности. На городском

уровне используют синтетический контроль, формируя виртуального двойника из непричастных участков.

Дополнительные элементы надёжности — MLOps и управление политиками. Репозиторий данных с каталогом и историей происхождения обеспечивает воспроизводимость расчётов и экспериментов. Мониторинг срабатываний и дрейфов инициирует дообучение или переключение на резервные простые политики. Контроль версий моделей и политик исключает незаметные развёртывания.

Журналы решений регуляторов являются полноправными артефактами, подлежащими аудиту. По ним можно восстановить, какие признаки и прогнозы привели к конкретной реакции, с какими уровнями неопределённости и ограничений.

Для оператора и регулятора необходимы как информационные панели, так и функциональный чёрный ящик для разбора инцидентов. На сетевом уровне полезны деревья решений, не как классификаторы, а как объяснительные схемы. Они показывают, при каких уровнях загрузки и в каких коридорах активируются те или иные политики и какие компромиссы при этом принимаются.

На этом фоне ценность иерархии становится очевидной. Разделение ответственности и временных горизонтов упрощает доказательство корректности работы системы. Локальные аспекты безопасности допускают формальную верификацию. Коридорные решения проверяются симуляцией и полевыми экспериментами. Сетевые эффекты оцениваются через эксперименты с синтетическим контролем и методы обучения с подкреплением.

Переходы между уровнями служат контрактными точками. Предикторы обязаны явно указывать свою неопределённость. Регуляторы должны соблюдать ограничения и чётко фиксировать допущения, лежащие в основе решений. Операционная команда должна обладать инструментами для мониторинга качества и гибкого перехода к упрощённым режимам при деградации данных или моделей.

В такой рамке искусственный интеллект представляет собой семейство инструментов, органично встроенных в инженерную дисциплину управления транспортом.

1.3 Подсистемы ИТС

1.3.1 Подсистема мониторинга и семантического восприятия

Подсистема мониторинга в современных ИТС всё чаще опирается на методы компьютерного зрения, визуальной аналитики и интеллектуальной обработки сенсорных данных. Обзор визуализации транспортных данных показывает, что представление потоков, скоростей, заторов, инцидентов и пространственно-временных паттернов движения является самостоятельной задачей, необходимой как для операторов, так и для последующей аналитической обработки [118]. Прикладные исследования визуализации на основе GPS-данных транспортных средств демонстрируют возможность восстановления структуры потоков и выявления закономерностей движения на городском уровне [119].

Компьютерное зрение в ИТС применяется для детекции транспортных средств, пешеходов и объектов дорожной сцены, классификации участников движения, трекинга, оценки плотности потока, обнаружения нарушений и инцидентов [120]. При этом эксплуатационные условия существенно влияют на надёжность моделей: ночное время, осадки, блики, перекрытия объектов и изменение ракурса приводят к деградации качества детекции, что требует специальных методов адаптации и контроля достоверности [121]. Дополнительным источником данных являются мобильные и смартфон-ориентированные сенсорные системы, позволяющие получать оценки ускорений, вибраций, скорости и дорожных событий непосредственно из транспортного потока [122].

Подсистема мониторинга и семантического восприятия выполняет функцию обработчика первичных данных в ИТС [61]. В ней сырые данные преобразуются в устойчивые семантические представления: потоки, очереди, скорости, траектории, события и контекст.

Состоятельностью системы является строгая архитектура извлечения признаков и управления неопределённостью. Сенсоры различаются по физическому принципу действия. К ним относятся видеокамеры, миллиметровые радары, лидары, магнитометры, индукционные петли, пьезодатчики и ме-

теостанции. Они также отличаются по охвату — точечному, линейному или площадному. Также по частоте измерений и устойчивости к внешним условиям.

Видеоподсистема обеспечивает максимальную семантическую плотность данных. Передовые архитектуры используют Edge-вычисления, которые позволяют проводить несложные операции на камере или периферийном устройстве. К таким операциям относятся детекция и трекинг ТС и классификация атрибутов объектов.

Зачастую для решения такого класса задач используют конвейеры обработки данных, которые сочетают в себе вычисление на устройстве и центральном сервере. При проектирование конвейера необходимо закладывать требования устойчивости к реальным условиям дорожной обстановки. Особенности предметной области включают в себя неполную видимость, перекрытия, изменение освещенности, различная погода, изменение ракурсов и недостаток разметки дорожного полотна.

Многомодальность повышает устойчивость восприятия. Радар обеспечивает надёжные оценки скорости и интервалов при любой освещённости. Лидар даёт точную геометрию и сегментацию даже на сложных фонах. Магнитометры и индукционные контуры предоставляют стабильные данные о проездах и уровне занятости проезжей части. На уровне слияния данных система решает две задачи, различающихся по природе погрешности. Временное выравнивание устраняет рассогласование, возникающее из-за различной задержки сенсоров и неравномерной частоты измерений. Пространственное выравнивание обеспечивает калибровку и привязку всех измерений к топологии УДС, т.е. переводит разнородные сигналы в единую систему координат, пригодную для последующего анализа.

Третьим обязательным уровнем выступает контроль качества данных. Формируются доверительные интервалы для скоростей, вероятности корректной принадлежности треков и индикаторы аномальности входных потоков. Публикуемые в общую шину данных, эти метрики используются последующими подсистемами для принятия решений с учётом рисков, т.е. неопределённость измерения становится явным операндом управляющей логики.

Особого внимания требует привязка сенсоров к реальному пространству. Традиционные методы ручной калибровки камер уязвимы к дрейфу во времени, и в городских условиях практически неизбежен переход к самокалибровке в процессе эксплуатации. Магнитометры и индукционные контуры при этом со-

храняют преимущество стабильности: предоставляемые ими данные о проездах и уровне занятости проезжей части не требуют геометрической перекалибровки, что делает их надёжным опорным слоем при верификации показаний остальных сенсоров.

Такой подход использует устойчивые паттерны движения, амплитуду и фазу потоков на опорных участках, известные ограничения скорости, а также видимые элементы инфраструктуры. Нейросетевые методы, оценивающие параметры проекций, предварительно обученные на синтетических данных и донастроенные на реальных, позволяют удерживать ошибки в пределах, не влияющих на корректную маршрутизацию треков между полосами.

Дополнительную надёжность обеспечивают явные инварианты. К ним относятся запрет на пересечение физических барьеров и движение против разрешённого направления, а также штрафы за траектории, несовместимые с топологией данного участка УДС.

Смена парадигмы с «видео в центре» на «события и признаки в центре» снижает нагрузку на сеть и хранилища. Подсистема мониторинга публикует в потоковую шину не сырые данные, а структурированные сообщения. К ним относятся детекции, треки, срезы очередей, агрегированные показатели по временным окнам и метаданные о качестве. Сырые кадры передаются только по запросу для обучения или верификации. Только такой дисциплинированный цикл позволяет своевременно выявлять и предотвращать незаметное ухудшение качества, вызванное сезонными изменениями, обновлениями оборудования или сдвигами в поведении транспортных потоков.

Зрелость рассматриваемой подсистемы ИТС проявляется в управляемых сценариях деградации. При ухудшении погоды или освещённости система переходит в режим грубых признаков. Треки заменяются на счётчики проездов, детализация атрибутов снижается, но сохраняется наблюдаемость макропараметров.

При перегрузке вычислительных ресурсов применяется динамический даунсемплинг с фокусом на зоны интереса. При снижении точности детекции повышаются пороги принятия решений и расширяются доверительные интервалы. Все такие изменения сопровождаются явной сигнализацией для вышестоящих уровней.

Логика адаптивного качества должна быть прозрачной для всех потребителей данных. Источник обязан передавать не только оценки состояния, но и

метрики их надёжности. Это позволяет операторам и автоматическим регуляторам корректно интерпретировать информацию и принимать обоснованные решения.

В общем виде подсистема мониторинга и семантического восприятия представлена на рисунке 1.13.

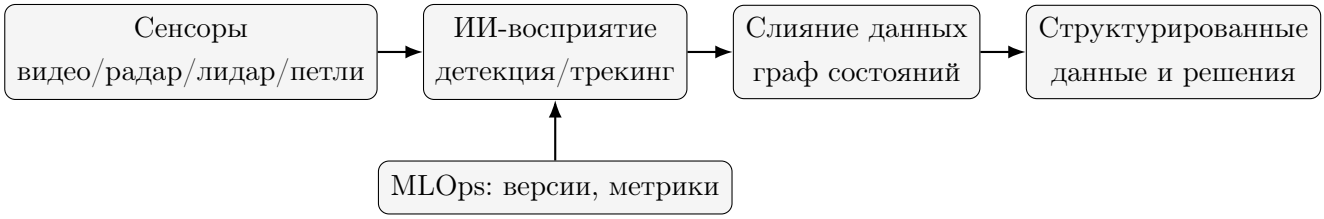

Рисунок 1.13 — Подсистема мониторинга

При обучении моделей применяют минимизацию объёма данных, агрегирование и процедурные оценки воздействия на приватность. Все наборы сопровождаются описаниями в формате структурировано записанных данных. Такой подход не замедляет технологическое развитие. Он предотвращает регуляторные и репутационные риски, что напрямую повышает устойчивость и жизнеспособность проекта.

1.3.2 Подсистема оценки состояния транспортного потока

Целью подсистемы оценки состояния является восстановление переменных, недоступных прямому измерению. К ним относятся плотность на участках, распределения транспортных средств по полосам, матрицы корреспонденций и вероятности блокировок. Эта оценка служит основой для кратко- и среднесрочного прогноза.

Искусственный интеллект в этой задаче сочетается с физическими принципами. Традиционные статистические модели обладают высокой гибкостью, но без учёта инвариантов, быстро теряют достоверность при редких событиях. Традиционные физические модели устойчивы, но недостаточно точны в условиях сложной городской сети.

Практика показывает преимущество гибридных подходов. Используют графовые нейросети с внедрением физических ограничений, трансформеры с

маскированием по топологии сети, ансамблевые фильтры для коррекции состояния в реальном времени и вероятностные выходные слои для получения калиброванных доверительных интервалов.

Схема данных для оценки состояния строится на топологии УДС. Граф состоит из вершин и рёбер. Вершины представляют узлы, детекторы или суперузлы. Рёбра соответствуют участкам, направлениям движения или полосам.

На этой структуре функционируют два контура. Первый выполняет реконструкцию текущего состояния. С учётом степени доверия к разным источникам и пространственно-временных корреляций он оценивает плотности и очереди даже при неполных данных.

Второй контур формирует прогноз. Он выдаёт предсказания на несколько горизонтов - 5, 15, 30 и 60 минут, с учётом неопределённости. Ключевое требование это калибровка прогнозов. Заявленный квантиль должен соответствовать фактической доле покрытия. Без этого регулятор либо излишне консервативен, либо принимает необоснованные риски.

В современных архитектурах используется иерархический подход. Локальные модели прогнозируют характеристики в рамках отдельных узлов на основе высокочастотных данных. Мезомасштабные модели функционируют на коридорах. На макроуровне происходит агрегация информации от локальных и мезомасштабных моделей. Иерахическая структура позволяет повысить точность управления и интерпретацию принятых решений. Функционирование данного подхода включает в себя оценку матриц корреспонденций и динамическое перераспределение потоков. По наблюдаемым характеристикам транспортного потока, восстанавливаются вероятностные матрицы источников и стоков, что позволяет распределять маршруты с учётом ограничений.

Инцидентный прогноз относится к задачам редких событий. Прямое обучение бинарного классификатора страдает от сильного дисбаланса и высокой неоднородности причин инцидентов. Более эффективный подход это обнаружение новизны и раннее предупреждение. Сначала моделируется нормальное состояние с учётом времени суток, дня недели и погодных условий. Затем выявляются значимые отклонения от этой нормы.

В том числе для повышения точности определения инцидентов используются многомодальные признаки. К таким признакам относятся резкие изменения дисперсии скоростей, аномалии в динамике очередей, визуально зафиксированные остановки транспортных средств. Результат сопровождается

доверительной оценкой вероятности инцидента. Для расширения возможностей применяют генеративные сценарные модели, включая вариационные и диффузионные. Они формируют набор возможных траекторий развития события и оценивают риск его влияния на соседние элементы УДС.

Для устойчивой работы в производственной среде необходимы предохранительные механизмы. При выходе индикаторов дрейфа за допустимые пределы подсистема автоматически снижает сложность модели, как правило переключаясь на более грубую, расширяет доверительные интервалы и уведомляет регуляторы о снижении надёжности прогнозов.

Важный организационный аспект это связка прогнозной подсистемы с цифровым двойником. Интеграция предполагает согласованную топологию сети, единые профили генерации транспортного спроса, воспроизводимый сценарный и механизмы реальных наблюдений в симуляциях. Настройка модели достигается калибровкой по архивным данным, включая телеметрию и видеозаписи. Доверие к системе формируется на основе статистики расхождений между прогнозами и фактическими исходами. Без данной процедуры операторы будут сомневаться в достоверности значений, и как следствие встает вопрос о целесообразности развёртывания решений в реальной инфраструктуре.

1.3.3 Подсистема управления и оптимизации

Подсистема управления преобразует оценку состояния и прогноз в управляющие воздействия. При этом она соблюдает ограничения безопасности, юридические нормы и эксплуатационные соглашения об уровне обслуживания.

Алгоритмы и архитектуры различаются по уровням, но объединены единым принципом иерархической организации и наличием контура безопасного исполнения.

На уровне узла, то есть светофорного объекта, решаются ключевые задачи. К ним относятся выбор последовательности фаз и их длительностей, соблюдение интервалов, защита уязвимых участников движения и обслуживание приоритетов для общественного и специального транспорта.

Классическое адаптивное управление, например на основе детекторов подъезда, обеспечивает базовую реактивность. В условиях нестабильного го-

родского трафика предпочтение получают стратегии максимального давления и локальный прогнозирующий контроллер с использованием краткосрочных прогнозов.

Обучение с подкреплением приобретает практическую применимость лишь при наличии строгой иерархии ограничений, внутри которой управляющая политика формирует предварительные предложения, контур безопасного исполнения отфильтровывает недопустимые варианты, а контроллер реализует оставшиеся решения с учётом ограничений на скорость изменений. Интерпретируемость принимаемых решений при этом выступает самостоятельным требованием: операторы должны понимать логику системы, для чего применяются визуализация напряжённости потоков, оценка вклада признаков через механизмы внимания и локально-линейные аппроксимации в окрестности текущего режима работы.

На коридорном уровне задачи координации и формирования зелёных волн решаются сочетанием прогнозирующего управления с вероятностными ограничениями и графовых предикторов, принимающих на вход прогнозы притоков, плотностей и оценки неопределённости, а на выходе формирующих сдвиги фаз и временные окна зелёной волны. Альтернативой служит иерархическое обучение с подкреплением, где верхний агент задаёт параметры локальных политик, а нижние агенты реализуют эти указания на отдельных узлах. Разрешение конфликтов приоритетов строится на допустимых нарушениях координации, ранжировании запросов и эвристиках. Обязательным условием устойчивости координации остаётся сохранение буферов для сглаживания колебаний очередей. Без учета буферов случайные колебания спроса быстро разрушают согласованность системы.

Задачей сетевого уровня является удержание транспортной системы в окрестности оптимальной загрузки. Методы ИИ оценивает текущее состояние сети и моделирует траектории её развития при различных управляющих воздействиях. Прогнозирующее управление и сценарные оптимизаторы формируют конкретные действия. Учёт неопределённости при этом является обязательным. Каждое управляющее решение сопровождается оценкой риска.

Модуль приоритета общественного транспорта хорошо иллюстрирует принцип интеграции подсистем. Мониторинг предоставляет точные оценки времени прибытия. Прогнозная подсистема формирует вероятностные окна прибытия, а управляющая подсистема корректирует фазы светофоров при допустимых

ограничениях. Управляющая подсистема минимизирует ущерб для пересекающихся потоков. Работоспособность в реальных условиях требует согласования профилей сообщений и интерфейсов на границах подсистем, а также устойчивой привязки к геометрии участка УДС и точной синхронизации времени. На подсистему управления накладываются строгие эксплуатационные требования. В них входят ограничения на задержку принятия решений, устойчивость к пропускам данных, отказоустойчивость с возможностью перехода в деградационный режим, а также полное журналирование и возможность аудита. Методы ИИ функционирует исключительно в рамках установленных правил. Модель не может иметь прямого управления устройством без контура безопасности.

1.3.4 Подсистема информирования и координации участников

Подсистема информирования и координации выполняет функцию голоса интеллектуальной транспортной системы. Она влияет на поведение участников движения до и во время поездки. В современных городах её ценность определяется не объёмом информации, а её качеством и своевременностью, релевантностью и предсказуемостью эффекта. Искусственный интеллект решает в этой подсистеме две задачи. Первая это персонализация и прогнозирование реакции пользователей. Определяется, какой контент, в какой форме, через какой канал и в какой момент вызовет полезное изменение поведения, например, смену маршрута, времени выезда или транспортного режима.

Вторая задача это координация массового поведения без нарушения устойчивости системы. Это необходимо для того, чтобы избежать саморазрушающего руководства, при котором рекомендации теряют эффективность из-за массового следования им. Вместо этого система стремится к квазистационарной стабилизации спроса.

Информационные сервисы (ATIS) предоставляют три типа сигналов. Первый сигнал это текущая ситуация (дорожно-транспортные происшествия, дорожные работы, погодные аномалии, перекрытия и небезопасные условия). Второй сигнал это формирование рекомендаций, таких как объездные маршруты, альтернативные узлы, целевые скоростные ограничения. Третий сигнал предназначен для информировании о прогнозах. Искусственный интеллект ин-

тегрируется в ядро формирования рекомендаций. Персонализаторы обучаются на архивных данных и контексте, чтобы учитывать предпочтения и ограничения пользователей. При этом они обязаны соблюдать принципы приватности и минимизации объёма обрабатываемых данных.

На макроуровне поведение транспортного потока моделируется средствами теории игр, где массовое следование единой рекомендации уничтожает её собственную выгоду и порождает новые зоны перегрузки. Для предотвращения подобной неустойчивости применяются стохастические рекомендации, распределяющие пользователей по альтернативным маршрутам, а также тарифные и регуляторные сигналы, которые корректируют поведение участников, сохраняя свободу их выбора.

Системы интегрированной мобильности, т.е. платформы класса MaaS, представляют собой отдельный контекст применения искусственного интеллекта, где прогнозирование охватывает доступность и загрузку разнородных транспортных сервисов, включая общественный транспорт, микромобильность и каршеринг. Построение кросс-модальных маршрутов ведётся с учётом надёжности, пользовательских предпочтений и полных обобщённых затрат, как временных, так и денежных. Рекомендации при этом сопровождаются доверительными интервалами по времени и резервным планом на случай отклонений, что повышает как доверие пользователей, так и реальную полезность сервиса.

Внешние интерфейсы включают табло переменной информации (VMS), мобильные приложения, API для сторонних сервисов и открытые порталы. Критически важны соглашения об уровне обслуживания. К ним относятся задержки публикаций, согласованность информации между каналами и устойчивость работы в пиковые периоды. Модули искусственного интеллекта обязаны функционировать в рамках дисциплины A/B-тестирования. Эффект оценивается не по субъективным метрикам, а по объективным показателям трафика (времени проезда, надёжности маршрутов и загрузке альтернативных направлений). Дополнительно применяются причинно-следственные модели, такие как uplift-модели. Они позволяют выделить влияние информационного сигнала, отделив его от других внешних и внутренних факторов.

Общая схема подсистемы информирования и координации участников представлена на рисунке 1.14.

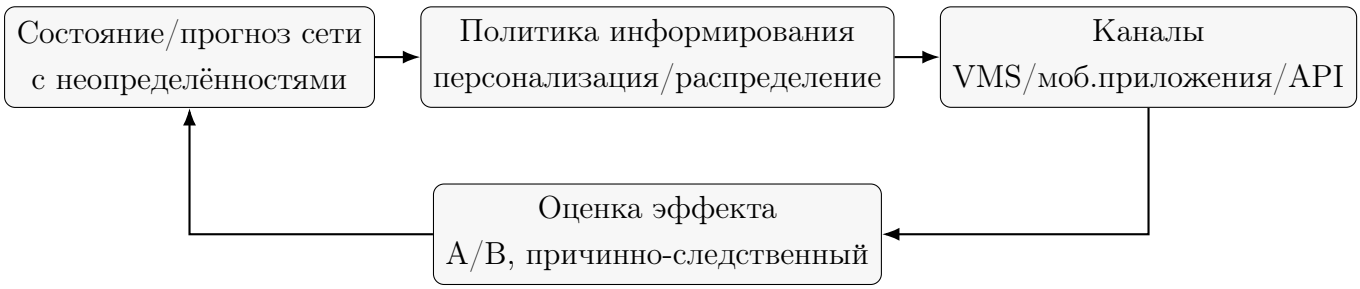

Рисунок 1.14 — Подсистема информирования

Эта подсистема несёт ответственность за прозрачность и доверие. Пользователи должны понимать, почему им предлагаются те или иные маршруты и время выезда, как учитывается неопределённость прогнозов и что произойдёт при отклонении от рекомендаций.

Интерпретируемость и объяснимость касаются не только внутренней логики искусственного интеллекта, но и всего сервиса в целом. Сообщения должны быть понятными, поведение системы — стабильным, информация — непротиворечивой между всеми каналами и согласованной с реальной дорожной обстановкой. Только при соблюдении этих условий информационная подсистема становится регулятором транспортного спроса, а не источником раздражения и недоверия.

1.3.5 Подсистема безопасности, управления инцидентами и устойчивого функционирования

Безопасность и устойчивость охватывают четыре функции: предотвращение, обнаружение, реагирование и восстановление. Искусственный интеллект повышает скорость и точность на каждом из этих этапов, но требует качественно подготовленного набора данных. Ложноположительные срабатывания недопустимы в условиях перегруженной диспетчерской. Пропуск реальных угроз ведёт к человеческим и материальным потерям. [62]

Правильная архитектура строится как каскад. На каждом этапе последовательно повышается уверенность в событии и возрастает стоимость вмешательства. Такой подход минимизирует как ложные срабатывания, так и пропуски критических инцидентов.

Предотвращение начинается с оценки риска. Подсистема мониторинга и прогноза вычисляет суррогатные показатели безопасности, такие как время до столкновения (TTC), время присутствия в зоне потенциальных конфликтов (PET). Модули искусственного интеллекта ранжируют локации по вероятности возникновения опасных ситуаций с учётом поведения участников движения, геометрии перекрёстков, текущих фаз светофоров и контекстных факторов.

На основе этой оценки подсистема управления может проактивно корректировать параметры. Например, сдвигать окна координации, увеличивать длительность красного сигнала для пешеходов в неблагоприятных погодных условиях или снижать скоростные ограничения перед сложными участками. Одновременно подсистема информирования выдаёт предупреждения и способствует растяжению потоков, снижая плотность и вероятность конфликтов.

Обнаружение состоит из двух этапах. Первый основан на детекторах, анализирующих макроскопические характеристики транспортного потока, а именно скорость, интенсивность, плотность. В том числе сюда относятся визуальные признаки остановки транспортных средств, аномалии в траекториях и резкие изменения длины очередей.

Второй слой выполняет подтверждающую функцию. Он включает агрегацию данных с видео и радаров, категоризацию инцидентов, неисправность транспортного средства, посторонний объект или опасное дорожное покрытие. В том числе он осуществляет оценку влияния какие полосы и направления затронуты, какова ожидаемая динамика очереди и время разрешения ситуации. Искусственный интеллект должен выдавать не только класс события, но и меру уверенности в нём, а также набор возможных управляющих действий, соответствующих текущей обстановке.

Реагирование на инциденты выстраивается как взаимодействие автоматизированных систем и человека, где каждый уровень включается сообразно сложности ситуации. Автоматизированные системы обеспечивают мгновенные действия: вывод сообщений на табло переменной информации, локальные корректировки фаз светофоров и скоростных ограничений, отправку уведомлений аварийным и диспетчерским службам.

При сложных инцидентах, где цена ошибки высока или требуется межведомственная координация, в принятие решений включается человек, а искусственный интеллект принимает роль ассистента. Ассистент предлагает конкретный план действий с оценкой сценарных исходов, объясняет причины

рекомендаций и обновляет прогноз по мере поступления новых данных. Эффективность реагирования при этом измеряется сокращением времени разрешения инцидента и минимизацией числа вторичных происшествий, а не формальным закрытием заявки.

Восстановление охватывает возврат системы к исходному состоянию и извлечение новых инцидентов и сценариев их решения в качестве обучающего материала. Постфактум-анализ, проводимый подсистемой, позволяет определить корректно сработавшие элементы, выявить возможности для дальнейшей автоматизации и сформировать направления улучшения детекторов и методов слияния данных, замыкая тем самым цикл непрерывного совершенствования.

Дополнительно реализуются планы обеспечения непрерывности функционирования. При частичном отказе узлов связи, вычислений или сенсоров, система переходит в деградационные режимы. Компоненты на основе искусственного интеллекта обязаны иметь иерархию упрощения. От сложных моделей система может переключаться к простым эвристикам, а при необходимости — к статическим сценариям. Каждый переход сопровождается чёткой сигнализацией для вышестоящих уровней управления.

Обобщенная схема подсистемы безопасности и управления инцидентами представлена на рисунке 1.15.

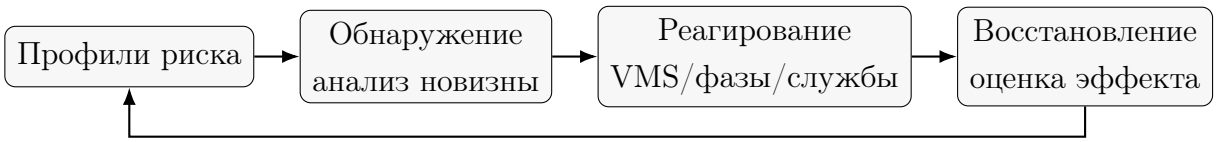

Рисунок 1.15 — Подсистема безопасности и управления инцидентами

Под устойчивостью понимается способность системы сохранять функции и восстанавливаться после отказов. Для интеллектуальных транспортных систем это означает отсутствие каскадных отказов. Локальные сбои не должны вызывать системные нарушения. Регуляторы не должны зацикливаться при шуме в данных. Информационные сервисы обязаны сохранять базовую работоспособность даже в деградировавшем режиме.

Искусственный интеллект повышает устойчивость тремя способами. Первый способ это оценка уязвимостей. Определяются критические узлы и участки, опасные сценарии и зоны недостаточной наблюдаемости. Второй способ это проведение стресс-тестирования. Проводятся симуляции с широким набором

экстремальных и маловероятных событий. Третий способ заключается во внедрении адаптивных управляющих политик. Они плавно снижают сложность при ухудшении качества входных данных и автоматически возвращаются к штатному режиму после восстановления условий.

Вместе описанные пять подсистем ИТС — мониторинг и семантическое восприятие, оценка и прогноз, управление и оптимизация, информирование и координация, безопасность и устойчивость — образуют основу зрелой интеллектуальной транспортной системы. Искусственный интеллект присутствует в каждой из них, но лишь как компонент дисциплинированной архитектуры. Эта архитектура опирается на чётких интерфейсах взаимодействия на стыках подсистем, явное управление неопределённостью, воспроизводимые метрики и регламентированные процедуры эксплуатации.

Такой подход направляет усилия на создание системы, способной функционировать в реальных городских условиях — с шумом в данных, сезонными изменениями, редкими событиями и жёсткими политико-правовыми ограничениями.

1.4 Методология оценки эффективности внедрения ИТС

Оценка эффективности внедрения ИТС должна учитывать не только улучшение отдельных транспортных показателей, но и качество функционирования всей архитектуры: доступность сервисов, задержки передачи данных, устойчивость к отказам, корректность сообщений, воспроизводимость расчётов и экономическую интерпретируемость результата. В исследованиях эффективности ИТС подчёркивается, что технологическое внедрение должно сопровождаться количественной оценкой влияния на безопасность, пропускную способность, время поездки, эксплуатационные затраты и качество транспортного обслуживания [64].

Для задач безопасности значимы как прямые показатели аварийности, так и суррогатные метрики, вычисляемые по данным мониторинга, видеонаблюдения, траекторий и кооперативных сообщений. Большие данные позволяют выявлять опасные состояния транспортного потока, оценивать риск инцидентов и формировать ранние предупреждения [112; 113]. При этом корректность вы-

водов зависит от качества экспериментального дизайна, наличия контрольных участков, учёта сезонности, внешних возмущений и статистической значимости наблюдаемого эффекта.

Следовательно, внедрение ИТС должно рассматриваться как полный жизненный цикл: проектирование архитектуры, развёртывание сенсорной и коммуникационной инфраструктуры, накопление и контроль качества данных, обучение и сопровождение моделей, эксплуатационная верификация, оценка эффектов и последующее обновление решений. Такой подход позволяет перейти от описания отдельных интеллектуальных компонентов к построению воспроизводимой инженерной методологии создания и оценки интеллектуальных транспортных систем.

Для оценки эффективности внедрения ИТС вводится методология оценки эффектов, опирающееся на архитектурные решения, с той же строгостью, с которой архитектура опирается на нормативные требования. Без этого экспериментальные выводы рискуют приписывать наблюдаемые изменения неверным причинам.

Центральный принцип заключается в том, что каждая гипотеза об эффекте формулируется в терминах наблюдаемых величин, имеющих устойчивую привязку к конкретному архитектурному слою. Проверка проводится по процедурам, согласованным с эксплуатационными соглашениями об уровне обслуживания и профилями сообщений.

Операциональные определения ключевых показателей эффективности выводятся от задач домена к измеримым признакам.

В домене мобильности базовыми являются медианное время поездки и надёжность маршрута, оцениваемая через индексированный буфер и показатель PTI. Показатель PTI сколько времени необходимо запланировать на конкретную поездку. PTI определяется как отношение 95-го перцентиля времени поездки к времени проезда в условиях свободного потока на том же участке.

В области безопасности, при недостатке статистики дорожно-транспортных происшествий, применяются суррогатные показатели. Их источниками служат данные компьютерного зрения и кооперативные сообщения. К ним относятся TTC, PET, частота конфликтов по калиброванным порогам, а также частота и пространственный охват предупреждений, передаваемых в сообщениях CAM и DENM.

Для оценки эксплуатационной устойчивости инфраструктуры используются метрики доступности сервисов и конечных узлов, сквозная задержка доставки данных и показатели надёжности. Показатель надежности оценивается как среднее время между отказами и среднее время восстановления, которое сопоставляется с целевыми значениями из соглашений об уровне обслуживания.

Экологические эффекты оцениваются по профилям скорости и ускорения с последующей аппроксимацией выбросов. Пользовательский опыт измеряется точностью и своевременностью информирования, долей корректных прогнозов времени прибытия и отсутствием противоречивых или дублирующих рекомендаций.

В домене кооперативных интеллектуальных транспортных систем выделяется отдельная группа архитектурно-функциональных метрик эффективности. К ним относятся доля валидных подписанных кадров, задержка между изменением фазы и публикацией SPaT, расхождение прогнозируемого поля maxEndTime с фактическим моментом переключения, доля корректных соответствий между физической группой светофора и полем eventState, частота генерации CAM по сценарным триггерам без избыточности, а также корректность применения словаря причин и подпричин в сообщениях DENM. Эти индикаторы связывают гипотезы об эффективности напрямую с реализацией стека V2X и логикой работы контроллеров.

Привязка ключевых показателей эффективности к архитектуре начинается с построения карты данных. Определяется, какие источники доступны на каждом уровне системы.

В поле используются журналы светофорных контроллеров, PCAP-записи радиоэфира на RSU, логи криптографических операций и проверок подписей, телеметрия бортовых устройств и систем AVL, а также данные метеостанций. На сетевом уровне фиксируются метрики потерь, джиттера и задержек по отдельным сегментам транспортной сети. На платформе доступны потоки стрим-шин, витрины обработанных данных и агрегированные наборы. На уровне приложений регистрируются выходы оптимизаторов, статусы обработчиков инцидентов и публикации в форматах DATEX II, SIRI, GTFS и TPEG.

Критически важны две оси согласования. Первая — сквозная синхронизация времени между слоями через единый источник UTC, основанный на GNSS, с резервированием по PTP и NTP. Вторая — регистрация происхождения данных, обеспечивающая воспроизводимость всех расчётов и анализов.

Именно поэтому в профиле V2X строго нормируются временные допуски. Эти ограничения обеспечивают контролируемость измеряемых метрик и позволяют объективно сопоставлять результаты испытаний между разными поставщиками оборудования и программного обеспечения.

Дизайн экспериментов строится в соответствии с иерархией уровней управления. На уровне отдельного контроллера или перекрёстка естественным подходом является сравнение до и после внедрения функции с использованием соседних узлов в качестве контрольных. Такой дизайн дополняется событийным анализом в окрестности момента активации функции, например при включении приоритета общественного транспорта или публикации сообщений SPaT/MAP.

В городском и региональном масштабе применяются методы синтетического контроля и панельные регрессии с фиксированными эффектами. Они позволяют одновременно учитывать пространственную неоднородность по узлам.

Во всех случаях явно задаются окна наблюдения до и после вмешательства, период выравнивания сезонных эффектов, а также процедуры контроля внешних возмущений — погоды, дорожных работ, массовых мероприятий. Обязательной является проверка предварительных трендов на параллельность в контрольной и экспериментальной группах.

Приёмочные испытания должны охватывать как доменные ключевые показатели эффективности, так и архитектурно-функциональные свойства. Без этого существует риск принять систему, демонстрирующую хорошие цифры в отчётах, но нестабильную в реальной эксплуатации.

В домене автоматизированных систем управления дорожным движением проверяется детерминированность реакций локального контроллера и точное соответствие сообщений SPaT фактическим фазам светофора. Требуется, чтобы расхождение между прогнозируемым полем `maxEndTime` и моментом реального переключения не превышало 250 мс в штатных условиях, отсутствовали ошибки в поле `eventState`, а средняя задержка публикации не выходила за пределы, установленные в профиле.

Для систем диспетчеризации общественного транспорта и приоритета проверяется корректность полного цикла приоритетного обслуживания. Доля корректно обработанных запросов, улучшение времени проезда и надёжности по маршрутам, а также отсутствие систематического ухудшения условий на пересекающихся потоках.

Кооперативные интеллектуальные транспортные системы расширяют классическую архитектуру ИТС за счёт прямого обмена данными между транспортными средствами, дорожной инфраструктурой и центрами управления. В работах по соединенным транспортным средствам подчёркивается, что эффективность таких систем определяется не только радиоканалом, но и согласованием сетевых протоколов, аналитики транспортных данных и приложений управления трафиком [123]. Аналитика данных в C-ITS используется для оценки состояния окружающей обстановки, предупреждения об опасных событиях, поддержки кооперативного управления и повышения устойчивости транспортной сети [124].

Перспективным направлением развития C-ITS является использование федеративного обучения, позволяющего обучать модели на распределённых данных транспортных средств и инфраструктуры без централизованной передачи исходных данных. Такой подход потенциально снижает риски нарушения приватности, однако требует решения задач неоднородности данных, устойчивости к дрейфу распределений, безопасности обновлений и согласования моделей между участниками кооперативной сети [125]. В совокупности это усиливает требование к архитектуре ИТС как к системе, в которой коммуникационный стек, управление данными, доверенная инфраструктура и жизненный цикл моделей должны рассматриваться совместно.

В кооперативном домене C-ITS/V2X верифицируются следующие аспекты: не менее 99,9% кадров имеют проверяемые и валидные цифровые подписи. Частота и триггеры генерации CAM соответствуют реальным условиям движения без избыточного шума. Поля `causeCode` и `subCauseCode` в сообщениях DENM используются непротиворечиво и в соответствии со стандартом. Границы геозоны ретрансляции на RSU заданы корректно и согласованы с топологией перекрёстка.

Для информационных сервисов ATIS оцениваются доля корректно доставленных предупреждений, их своевременность и согласованность публикаций между каналами — DATEX II, внешние API, табло переменной информации.

Часть критериев неизбежно носит сквозной характер. К ним относятся доступность компонентов, сквозная задержка доставки данных, устойчивость к деградациям через активацию резервных режимов, полнота и точность журналирования, а также соблюдение процедур информационной безопасности и защиты персональных данных.

В событийном анализе вместо одного индикатора воздействия вводится набор коэффициентов. Это позволяет явно проверить параллельность предварительных трендов и проследить динамику эффекта до и после вмешательства.

Мощность теста и минимально различимое изменение (MDE) рассчитываются до начала эксперимента на основе исторических дисперсий и ожидаемого охвата. Без этого существует риск интерпретировать статистически недостоверный результат как отсутствие эффекта, тогда как на самом деле причина заключается в недостаточной информативности данных.

Валидация включает анализ чувствительности к выбору временного окна, набору ковариат, методу нормализации и стратегии обработки выбросов. Рекомендуется проводить тесты на предынтервенционных интервалах и использовать ложные контрольные показатели.

Экономическая интерпретация возвращает инженерные результаты в плоскость управленческих решений. Выигрыш времени количественно переводится в экономическую ценность как для пользователей, так и для оператора. Снижение числа конфликтных ситуаций интерпретируется как ожидаемое сокращение аварийности и связанных с ней затрат. Уменьшение дисперсии времени проезда рассматривается как повышение надёжности транспортных сервисов и источник косвенных экономических эффектов.

В рамках жизненного цикла проекта эти эффекты агрегируются в анализе затрат и выгод (CBA) и анализе жизненного цикла стоимости (LCCA). Учёт включает совокупные капитальные и операционные расходы (TOTEX), сценарную неопределённость и рисковые надбавки. На ранних этапах допустимы ориентировочные диапазонные оценки. В финальной приёмке целесообразны интервальные значения с явным сценарным разложением. Данные показатели позволяют объективно оценить применимость архитектуры ИТС. В данной работе для оценки предлагается использовать матрицу трассируемости, представленная в таблице 8. Матрица включает в себя архитектурный компонент, эффект, измеритель, приёмочный тест, порог. Матрица интегрируется с реестром требований и набором тест-кейсов, объединяя техническую и эмпирическую стороны проекта. Она обеспечивает сквозную прослеживаемость от архитектурной реализации до эмпирической проверки. Это упрощает процедуры приёмки, а также последующие аудиты и повторные проверки в ходе жизненного цикла системы.

Таблица 8 — Матрица трассируемости

Слой / компонент	**Эффект**	**Измеритель**	**Приёмочный тест**	**Порог**
Поле (RSU, контроллеры)	Задержка публикации SPaT, консистентность `eventState`	Журналы контроллера	Сопоставление поля `maxEndTime` и момента фактического переключения фазы	$\leqslant 250$ мс
Сеть	Джиттер, потери, сквозная задержка на сегментах	Телеметрия маршрутизаторов	Нагрузочное моделирование с контролем	SLA
Платформа	Доля поздних и пропущенных сообщений в стриме	Метрики стрим-обработки, логи консьюмеров	Повторяемость эталонных трасс с проверкой полноты и синхронности	$\geqslant$ 99,5% доставки в окно $\leqslant 1$ с
Прикладной слой (ATMS, C-ITS, APTS)	Время поездки, надёжность, суррогаты безопасности	Телеметрия, транзакционные логи, V2X- события	Событийный анализ с фиксированными окнами и ковариатами	Статистически значимое улучшение

Воспроизводимость результатов является важнейшим условием доверия. Для повышения доверия все алгоритмы должны быть версионными. Параметры анализа должны быть четко задокументированы. Исходные наборы данных должны включать метаданные, указывающие источники временных меток и цепочки преобразований. Данные, содержащие персональную информацию, должны пройти процедуру псевдонимизации. Для публичных публикаций предоставляются автоматически сгенерированные отчеты с контрольными суммами, чтобы гарантировать, что результаты останутся неизменными при повторном запуске процесса.

В таком виде методология завершает полный цикл. Архитектура обеспечивает измеримую основу для мониторинга. Ключевые показатели эффективности определяют цели и методы их мониторинга. Экспериментальный дизайн позволяет отличать причинно-следственные связи от простых корреляций. Приемочные испытания подтверждают инженерное качество реализации. Экономическая интерпретация переводит полученные результаты обратно в логику принятия управленческих и инвестиционных решений.

Обзор литературы показывает, что современная ИТС - это не изолированная автоматизированная система управления дорожным движением. Это распределенная интеллектуальная информационная система, в которой архитектура, сенсорная инфраструктура, большие данные, совместные коммуникации, методы ИИ и контуры управления безопасностью образуют единую систему. Существующие исследования подробно раскрывают отдельные аспекты ИТС: общую концепцию и классификацию систем [66; 67; 76], архитектуру и стандартизацию [69; 87], обработку больших транспортных данных [95; 110; 111], компьютерное зрение и мониторинг [120], кооперативные транспортные коммуникации [123; 124], а также интеллектуальное управление и обучение с подкреплением [115].

Однако существует методологический разрыв между моделями прогнозирования, безопасной реализацией управляющих решений и формализованной оценкой эффективности распределенных ИТС в условиях нестационарных транспортных потоков, ограничений по задержкам и требований к эксплуатационной надежности.

1.5 Выводы по главе

В главе выполнен систематический анализ современного состояния интеллектуальных транспортных систем как социотехнических распределённых систем, в которых домены управления дорожным движением, информирования участников, приоритизации общественного транспорта, кооперативных сервисов V2X, дорожной метеорологии и безопасности функционируют в рамках единой многослойной архитектуры и сквозных функций. Показано, что инженерно состоятельная ИТС не сводится к автоматизации отдельных устройств и каналов связи, а должна быть организована вокруг интеллектуального замкнутого контура МПУ, обеспечивающего трассируемую привязку архитектурных решений к измеримым показателям качества и процедурам приёмочных испытаний.

Показано, что практическая реализуемость ИТС определяется согласованием доменной модели и ответственности, многослойной архитектуры, и иерархии управления с различными временными горизонтами решений и

ограничениями SLA. Выявлено, что ключевые эксплуатационные требования формируются не только метриками точности моделей, но и ограничениями по задержке, пропускной способности, доступности, устойчивости к нестационарной нагрузке и деградационным режимам.

Проведённое сопоставление подсистем ИТС и применяемых методов позволяет выделить ряд системных ограничений, которые формируют научную проблему и являются основой дальнейшего исследования.

Несмотря на достижения в области нейросетевых моделей, сохраняются ограничения, касающиеся одновременного учета долгосрочных временных зависимостей и пространственных взаимодействий при масштабировании до больших сетей, а также реализации однопроходного многогоризонтного прогнозирования без авторегрессионного расширения.

Выявлено, что на практике практике часто отсутствует единая, формализованная структура, позволяющая синтезировать конфигурации выполнения моделей ИИ с вероятностными ограничениями на задержку и стабильность в зависимости от интенсивности входа. Это увеличивает риск каскадных отказов при нестационарных нагрузках.

Показано, что эффект внедрения ИТС должен оцениваться через трассируемые метрики и корректный дизайн экспериментов. Вместе с тем, на практике ограниченная пригодность доступных наборов данных и недостаточная формализация происхождения/качества данных затрудняют воспроизводимую верификацию прогнозных и управляющих компонент, а также стресс-тестирование.

Многослойная архитектура и доменная типология задают рамки, однако остаётся потребность в методологии, обеспечивающей отказоустойчивость, масштабируемость и самонастройку распределённой платформы обработки больших данных и исполнения ИИ-моделей, включая децентрализованную кластеризацию, динамическую реконфигурацию и воспроизводимое нагрузочное тестирование.

Таким образом, проведённый в главе анализ требований международных стандартов, нормативно-правовой базы Российской Федерации, а также системный анализ современного состояния интеллектуальных транспортных систем позволили выделить ряд системных ограничений, которые формируют научную проблему и являются основой дальнейшего исследования.

В главе показано, что при высокой степени проработанности отдельных компонент ИТС сохраняется методологический разрыв между извлечением закономерностей и прогнозированием по пространственно-временным данным в условиях нестационарности, устойчивым микроскопическим моделированием поведения в замкнутом контуре, и формализуемым обеспечением требований к надёжности и качеству сервиса распределённой системы исполнения моделей при ограничениях по задержке, пропускной способности и доступности. Указанные результаты и выявленные ограничения обосновывают необходимость дальнейших глав диссертационной работы.

Глава 2. Разработка метода прогнозирования макроскопических характеристик транспортного потока

Прогнозирование транспортных потоков в ИТС используется как расчётная основа для управления перегруженными участками, оценки времени в пути и поддержки навигационных сервисов. [48; 60] В практических условиях эта задача осложняется тем, что скорость и интенсивность движения меняются не только во времени, но и по сети связанных дорожных участков. Поэтому при обработке данных транспортных сенсоров всё чаще применяются модели машинного обучения, включая глубокие нейронные сети, способные восстанавливать нелинейные пространственно-временные зависимости. [3; 7; 25]

В ранних нейросетевых подходах основное внимание уделялось временной зависимости. Для этого применялись рекуррентные архитектуры LSTM [126] и GRU [127]. Их скрытое состояние передаётся от одного шага к следующему и тем самым хранит часть предшествующей динамики ряда. На коротких горизонтах данные подходы обычно работают устойчиво. При увеличении длины последовательности появляются известные ограничения: обучение становится чувствительным к затуханию градиента, а дальние события хуже влияют на текущий прогноз.

Следующий шаг был связан уже не только с временной, но и с сетевой структурой данных. В моделях STGCN [128] и DCRNN [129] дорожная сеть задаётся графом, а пространственное взаимодействие описывается через графовые свёртки или диффузионные операторы. Это позволяет учитывать физическую связность сенсоров и направление распространения возмущений по сети. Ограничение такого подхода состоит в том, что граф обычно задаётся заранее. Между тем фактические корреляции между участками меняются в зависимости от времени суток, режима регулирования и перераспределения спроса.

Трансформерные архитектуры [130] решают задачу иначе. Они не задают связи между позициями заранее, а вычисляют веса внимания по данным. Для временных рядов эта идея была развита в моделях Informer [131], Autoformer [132] и PatchTST [133]. В них используются разные способы снижения вычислительной стоимости: разреженное внимание, декомпозиция ряда на компоненты и разбиение последовательности на патчи. В транспортной задаче это важно,

поскольку входное окно может быть длинным, а прогноз требуется строить сразу на несколько горизонтов.

Патчинг меняет единицу обработки временного ряда: вместо отдельного временного шага модель работает с фрагментом ряда, внутри которого уже закодирована некоторая локальная динамика. Длина последовательности внимания при этом сокращается пропорционально размеру патча, что делает обучение устойчивее на длинных горизонтах. Однако и здесь сохраняется та же слабость, что преследовала трансформерные решения с самого начала: пространственные связи между сенсорами либо игнорируются полностью, либо учитываются косвенно через совместную обработку многомерного входа, без явного механизма, кодирующего топологическую или функциональную близость узлов сети.

Сравнение этих подходов показывает, что ни один из них сам по себе не охватывает всю сферу транспортного прогнозирования. Рекуррентные модели в основном ориентированы на временную динамику. Графовые модели включают топологию, но часто используют фиксированную матрицу смежности. Трансформаторы лучше работают в долгосрочном контексте, но ограниченно используют структуру дорожной сети в чисто временном контексте. Поэтому далее рассматривается подход, в котором временное кодирование совмещается с отдельным пространственным блоком, а графовая структура уточняется по данным.

Выделяются две практические проблемы. Во-первых, статические графы не отражают изменчивых и контекстно-зависимых связей между участками: корреляции варьируются в зависимости от времени суток, инцидентов и перераспределения маршрутов. [45] Во-вторых, чисто временные трансформеры, несмотря на способность улавливать периодичности и долгосрочные тренды, не обеспечивают соблюдения релевантных транспортных ограничений, естественных для сетевой структуры.

Для решения этих проблем предлагается компактный гибридный метод **PTSTG** (PatchTST-Graph). [1] Он объединяет патч-ориентированный трансформер с адаптивной матрицей смежности, обучаемой из данных, и лёгкими графовыми блоками уточнения. Каждый временной ряд сенсора разбивается на перекрывающиеся патчи, которые кодируются трансформером. Усреднение по патч-токенам даёт устойчивое вложение узла, отражающее как краткосрочные колебания, так и долгосрочную периодичность.

На основе этих вложений строится матрица смежности: каждая строка нормируется, а связи определяются не по географической близости, а по обучаемым латентным признакам узлов. [4] Затем лёгкие графовые блоки с остаточными соединениями и полносвязными слоями распространяют информацию между узлами. Прогноз на все горизонты формируется за один проход без итеративного развёртывания.

Архитектура PTSTG проектировалась с учётом операционных ограничений реальных интеллектуальных транспортных систем, и каждое из принятых решений отвечает конкретному классу ограничений. Патч-ориентированное кодирование сокращает длину последовательности внимания и снижает потребление памяти, тогда как прогнозирование на несколько шагов вперёд выполняется за единственный прямой проход, что исключает накопление ошибок, характерное для авторегрессионных подходов. Обучаемая матрица смежности допускает разрежение на этапе вывода по принципу top-k соседей, обеспечивая субквадратичную сложность по числу сенсоров. [8]

Привязанность к статической топологии при этом снимается: модель способна выявлять функциональные связи между географически удалёнными участками, что особенно существенно в режимах перераспределения потоков, когда физическая близость перестаёт быть надёжным предиктором корреляции. Полученная матрица смежности допускает качественную интерпретацию, в частности выделение коридоров, склонных к совместным заторам.

В этой главе фиксируются несколько результатов. Во-первых, предлагается метод PTSTG, в котором патч-ориентированное временное кодирование дополнено графовым уточнением на основе обучаемой матрицы смежности. Во-вторых, прогноз на все выбранные горизонты формируется за один прямой проход, что важно для последующего использования модели в контуре ИТС. В-третьих, матрица смежности не задаётся только физической близостью сенсоров, а обучается по данным и может отражать функциональные связи между участками. В-четвёртых, экспериментальная часть позволяет сопоставить точность метода с базовыми моделями и отдельно оценить вычислительную пригодность предложенной архитектуры.

Метод PTSTG не позиционируется как универсально превосходящий существующие решения по точности на всех наборах данных и горизонтах прогнозирования. Его основная ценность заключается в обеспечении рационального сочетания вычислительной эффективности, масштабируемости и

практической применимости. Ключевыми свойствами метода являются однопроходное прогнозирование, патч-уровневое кодирование времени и лёгкое пространственное уточнение на основе данных. Эксперименты подтверждают сопоставимую точность при значительно более благоприятных характеристиках для развёртывания.

Далее метод PTSTG рассматривается как практическая попытка совместить два требования: сохранить эффективность патч-ориентированного временного кодировщика и добавить явное пространственное взаимодействие между узлами сети. При увеличении числа сенсоров такой метод может масштабироваться за счёт разрежения межузловых связей. [24]

2.1 Анализ современного состояния прогнозирования макроскопических характеристик транспортного потока

Рекуррентные модели LSTM и GRU стали одним из первых нейросетевых инструментов для работы с транспортными временными рядами [126; 127]. За счёт механизмов затворов они удерживают часть предыдущего состояния и поэтому подходят для краткосрочного прогноза скорости или интенсивности. При увеличении горизонта проявляются известные ограничения: последовательная обработка затрудняет обучение, а дальний контекст постепенно теряет влияние на текущий прогноз. Это стало одной из причин перехода к архитектурам с механизмом внимания.

Трансформерные модели стали отдельной линией развития методов прогнозирования временных рядов. В этих моделях основной вопрос состоит не только в наличии внимания, но и в том, как уменьшить его вычислительную стоимость и какие предположения о структуре ряда заложить в архитектуру. Для транспортных данных это особенно важно, поскольку входной ряд имеет одновременно временную и пространственную природу.

Одной из ранних архитектур, ориентированных на длинные горизонты, стала модель Informer [131]. В ней ProbSparse-внимание уменьшает стоимость обработки последовательности по сравнению с классическим полным вниманием сложности $O(L^2)$. Дополнительно используется дистилляция представлений, уменьшающая длину последовательности в глубине сети. Генеративный декодер

строит прогнозный горизонт за один проход, что важно для задач, где задержка вывода модели имеет прикладное значение.

Другое направление работ связано с включением в архитектуру предположений о структуре временного ряда. Autoformer [132] использует декомпозицию на сезонную и трендовую компоненты и применяет механизм автокорреляции для поиска повторяющихся фрагментов. В работе показано снижение ошибки на долгосрочных прогнозах на данных энергетики и транспорта. FEDformer [134] развивает эту идею в частотной области. Разреженное представление Фурье совместно с сезонно-трендовой декомпозицией позволяет учитывать глобальные компоненты ряда без полного квадратичного внимания.

Модель PatchTST [133] меняет единицу обработки ряда. Вместо отдельных временных отсчётов используются перекрывающиеся патчи, внутри которых сохраняется локальная динамика. За счёт этого уменьшается длина последовательности внимания и снижается расход памяти при работе с длинным входным окном. Для транспортной задачи это полезно, но не снимает пространственное ограничение: такие модели в базовой форме в основном работают с временным измерением и не задают явную связь между сенсорами дорожной сети.

Crossformer [135] пытается учесть межрядовые взаимодействия более явно. Многомерный ряд представляется как матрица «время × признаки», после чего применяется двухэтапное внимание по временным и межсенсорным связям. Non-stationary Transformer [136] отдельно работает с проблемой сдвига распределений через обучаемую нормализацию. ETSformer [137] использует идеи экспоненциального сглаживания и выделяет уровень, рост и сезонность, что делает модель более интерпретируемой в задачах временного прогноза.

Zeng et al. [138] показали, что простая двухслойная линейная модель DLinear может превосходить более сложные трансформерные архитектуры на ряде стандартных бенчмарков долгосрочного прогнозирования. Этот результат важен методически. Он показывает, что качество прогноза определяется не только сложностью модели, но и соответствием архитектурных предпосылок природе ряда. Если данные имеют выраженную трендовую или сезонную структуру, линейная декомпозиционная модель может оказаться достаточно сильной. Если же требуется учитывать нелинейные зависимости, пространственные взаимодействия и изменение корреляций между узлами, возможностей такой модели уже недостаточно. За пределами задач скорости и интенсивности

трансформерные архитектуры также применяются в пользовательско-ориентированных сценариях ИТС, например в модели TripChain2RecDeepSurv для предсказания изменений поведенческого статуса пользователей общественного транспорта [139].

Графовые пространственно-временные методы.

Транспортный поток естественно описывается на графе. Вершинами могут быть сенсоры, перекрёстки или участки дороги, а рёбрами — физические или функциональные связи между ними. Такое представление позволяет перейти от независимого прогноза по каждому ряду к совместному учёту временной динамики и пространственного распространения возмущений.

В модели STGCN [128] пространственная часть описывается графовыми свёртками на основе полиномов Чебышёва, а временная часть — свёртками с затворами. Такой вариант позволил уйти от чисто рекуррентной обработки и учитывать локальное распространение транспортных возмущений по дорожной сети. DCRNN [129] рассматривает движение как диффузионный процесс на ориентированном графе и совмещает диффузионные свёртки с рекуррентными блоками GRU. STSGCN [140] строит локальные пространственно-временные графы внутри скользящих окон и тем самым лучше описывает синхронные взаимодействия на коротких горизонтах.

Ранние графовые модели имели два ограничения. Во-первых, рекурсивная часть могла накапливать ошибки с увеличением временного интервала. Во-вторых, матрица смежности часто строилась на основе расстояния и оставалась неизменной. Graph WaveNet [141] заменил рекуррентную обработку причинными свёртками и ввёл адаптивную матрицу смежности, вычисляемую через обучаемые вложения узлов. Такая матрица $\mathbf{W}_{\text{learned}}$ может отражать скрытые корреляции между сенсорами, которые не следуют напрямую из географической близости.

MTGNN [142] и AGCRN [143] развивают идею латентной топологии транспортной сети. MTGNN рассматривает многомерный временной ряд как граф с поддающимися обучению связями между узлами и использует эти связи для учета глобальных корреляций. AGCRN дополнительно адаптирует параметры модели к отдельным узлам и строит граф непосредственно на основе временного ряда. Оба подхода повышают производительность по сравнению с моделями

с фиксированной смежностью, но сохраняют повторяющийся компонент, который ограничивает их применимость в долгосрочной перспективе.

Модель GMAN [144] использует пространственные и временные блоки внимания в архитектуре кодировщика-декодировщика. Таким образом промежуточный трансформерный слой напрямую сопоставляет прошлые и будущие шаги, что приводит к снижению накопления ошибок. Механизм внимания над дорожным графом эффективно улавливает динамические пространственные влияния. На момент 2020 года модель GMAN показала лучшие результаты среди сопоставимых архитектур. В спектральном направлении StemGNN [145] применяет графовое преобразование Фурье для совместного моделирования спектральных компонент временных рядов и межсенсорных зависимостей. Данная реализация позволяет превзойти Graph WaveNet на нескольких бенчмарках.

Каждое поколение моделей от STGCN к StemGNN было направлено на решение предыдущих поколений, что порождало при этом новые. Вместо статического графа стали использовать адаптивный. Вместо рекуррентных блоков стали использовать свёрточные и механизмы внимания. Спектральные методы позволили использовать дополнительное измерение для захвата зависимостей. Накопленный опыт указывает на то, что наиболее перспективным направлением остаётся совместное обучение топологии и динамики. Граф порождается из данных. Временные зависимости моделируются без рекуррентных ограничений.

Графовые пространственно-временные модели позволяют включить структуру УДС в процедуру прогноза и тем самым улучшают описание пространственного взаимодействия между участками сети. Однако остаются два ограничения. Первое связано с обработкой длинных временных зависимостей, особенно в моделях с рекуррентными блоками. Второе связано с нестационарностью спроса, при которой статичный граф уже не отражает фактические связи между участками. Поэтому дальнейшее развитие логично связывать с гибридными архитектурами, где графовое пространственное моделирование сочетается с более сильными временными механизмами.

Гибридные пространственно-временные методы с обучением графа.

Гибридные методы пытаются совместить два ранее разделённых направления: графовое описание дорожной сети и трансформерное кодирование

временных зависимостей. Единой методики такого совмещения пока не сложилось, поэтому в литературе используются разные архитектурные решения.

В работе [146] предложена модель Spacetimeformer. В ней измерения всех сенсоров преобразуются в одну последовательность, а стандартный трансформер обрабатывает временные и межсенсорные связи совместно. Такой вариант не требует заранее заданного графа, но длина последовательности быстро растёт при увеличении числа узлов. Для крупных транспортных сетей это приводит к росту вычислительных затрат. Кроме того, отказ от явной топологии усложняет интерпретацию найденных связей.

Другой вариант состоит в раздельной обработке пространственного и временного измерений. К этому классу относится STTN [147]. Модель использует пространственный блок для учёта влияния соседних узлов и отдельный временной блок для динамики каждого узла. Такой подход даёт выигрыш относительно рекуррентных и свёрточных базовых моделей на средних горизонтах. Его ограничение состоит в том, что строгое разделение доменов не всегда описывает ситуации, где пространственное влияние и временная задержка проявляются одновременно.

В моделях ASTTN [148] и ASTGNN [149] используется адаптивное описание пространственно-временных связей. Эти архитектуры вводят обучаемые матрицы смежности и признаки временного контекста, что позволяет учитывать периодичность трафика и изменение связей между узлами. При этом не все успешные решения используют явный графовый оператор. STAEformer [150] показывает, что трансформер с обучаемыми пространственными и временными вложениями может превосходить ряд гибридных моделей на транспортных наборах данных. В такой реализации часть роли графа фактически переносится на систему вложений.

STGormer [151] также использует обучаемое разделение режимов, но делает это через механизм смеси экспертов. Разные подсети специализируются на различных типах транспортного поведения, а распределение нагрузки между экспертами можно использовать для анализа того, какие паттерны модель различает. Такое решение повышает выразительность архитектуры, но одновременно усложняет обучение и последующее развёртывание.

PatchTST формирует патч-вложения для временных рядов, после чего эти представления уточняются через обучаемую стохастическую матрицу смежности A. Затем прогноз на все горизонты строится за один прямой проход [150].

При необходимости матрица смежности может быть разрежена через top-k соседей, что снижает стоимость межузлового смешивания при большом числе сенсоров.

Сравнение этих моделей показывает, что способы объединения графа и трансформера различаются по цене и по интерпретируемости. Модели без явного графового оператора проще включаются в стандартный трансформерный стек, но хуже показывают, какие именно связи между участками были использованы. Модели с явной графовой компонентой дают более понятную пространственную структуру, но требуют дополнительных операций над узлами. PTSTG занимает промежуточную позицию: пространственная информация задаётся обучаемой матрицей смежности, а вычислительная стоимость может контролироваться разрежением.

Адаптивное обучение графа и построение матрицы смежности из данных.

В современных моделях транспортного прогноза граф всё чаще не задаётся полностью заранее, а уточняется по данным. В ранних подходах матрица смежности обычно строилась по расстоянию между сенсорами или по физической связности дорог. Такая топология удобна для интерпретации, но она не отражает все функциональные корреляции, возникающие из-за маршрутизации, светофорного регулирования и внешних событий.

Graph WaveNet [141] обучает дополнительную матрицу смежности через латентные вложения узлов. В этом случае граф становится параметром модели, а не только внешним описанием дорожной сети. AGCRN [143] формирует граф из временных рядов и тем самым уменьшает зависимость от заранее заданной топологии. ASTGNN [149] строит динамические связи с помощью внимания, а DGCRN [152] использует разные графы для разных временных интервалов. Эти подходы позволяют находить связи между участками, которые могут быть удалены географически, но связаны общим режимом движения.

У обучаемого графа есть важное ограничение. Его связи не обязаны совпадать с реальными дорогами. Поэтому такую матрицу нельзя автоматически трактовать как физическую топологию УДС. В инженерной постановке её нужно проверять отдельно: сопоставлять с коридорами движения, маршрутными альтернативами и режимами регулирования. По этой причине ряд моделей,

включая Graph WaveNet и DGCRN, совмещают фиксированный и обучаемый графы, сохраняя физическую топологию как опорную структуру.

Отдельное направление связано с регуляризованным обучением структуры. В GTS [153] граф оценивается как разрежённая дискретная матрица с L_0-регуляризацией и возможным использованием априорной топологии. Такой подход позволяет получить структуру, близкую к физической сети, но скорректированную по данным. STFGNN [154] обрабатывает географический и обучаемый графы параллельно и объединяет их на каждом временном шаге. За счёт этого модель учитывает как локальные связи, так и более дальние функциональные зависимости.

Таким образом, развитие моделей идёт от чисто временной обработки к совместному описанию временной динамики и адаптивной пространственной структуры. Для задачи настоящей главы это означает, что модель должна сохранять эффективность временного кодирования и при этом учитывать связи между узлами сети. PTSTG строится именно в этой логике. PatchTST используется как временной кодировщик, а обучаемая стохастическая матрица смежности добавляет пространственное взаимодействие между сенсорами.

2.2 Разработка метода PTSTG: PatchTST-Graph for Spatio-Temporal Forecasting

Модель **PTSTG (PatchTST-Graph)** используется для пространственно-временного прогноза по данным сенсорной сети. В ней временная часть строится на патч-ориентированном кодировании, а пространственная часть задаётся обучаемым межузловым взаимодействием. На вход подаётся фиксированное историческое окно для каждого узла. На выходе за один прямой проход формируется прогноз на заданный горизонт. Разделение временного и пространственного блоков позволяет отдельно контролировать сложность кодировщика и графового уточнения.

Постановка задачи

Заданы исторические наблюдения

$$X \in \mathbb{R}^{B \times T_{\text{in}} \times N \times C_{\text{in}}}, \tag{2.1}$$

и целевые значения

$$Y \in \mathbb{R}^{B \times H \times N \times C_{\text{out}}}, \tag{2.2}$$

где B — размер батча, T_{in} — длина входного окна, N — число узлов, $C_{\text{in}}, C_{\text{out}}$ — размерности входных и выходных признаков, H — прогнозный горизонт. Требуется построить параметризованное отображение Φ_θ, такое что

$$\hat{Y} = \Phi_\theta(X) \in \mathbb{R}^{B \times H \times N \times C_{\text{out}}}. \tag{2.3}$$

Параметры θ подбираются по обучающей выборке так, чтобы прогноз $\hat{Y} = \Phi_\theta(X)$ был близок к целевому тензору Y. В этой работе задача записывается как минимизация эмпирического риска:

$$\theta^\star = \arg\min_\theta \; \mathbb{E}_{(X,Y)\sim\mathcal{D}}\big[\mathcal{L}(\Phi_\theta(X),Y)\big] \;\approx\; \arg\min_\theta \; \frac{1}{|\mathcal{D}|} \sum_{(X,Y)\in\mathcal{D}} \mathcal{L}(\Phi_\theta(X),Y), \tag{2.4}$$

где $\mathcal{D}$ - обучающая выборка, а функция потерь $\mathcal{L}(\hat{Y},Y)$ задаётся далее и агрегирует ошибку по всем шагам прогнозного горизонта.

Здесь H задаёт число дискретных шагов прогноза. При шаге дискретизации Δt длительность прогнозного интервала равна $H\Delta t$. В экспериментах используется $\Delta t = 5$ минут, поэтому значения $H \in \{3,6,12\}$ соответствуют горизонтам 15, 30 и 60 минут.

Перед обучением выполняется стандартизация отдельно по узлам и каналам с использованием статистик обучающей выборки. Далее используются следующие обозначения: d — размерность вложений, P — длина патча, s — шаг сдвига, $L = \lfloor (T_{\text{in}} - P)/s \rfloor + 1$ — число патчей, r — ранг графовой матрицы, L_g — число графовых блоков.

Обработка экзогенных переменных

Экзогенные признаки могут быть добавлены в PTSTG тремя способами.

1. *Поузловые временные ковариаты* $Z \in \mathbb{R}^{B \times T_{\text{in}} \times N \times C_{\text{exo}}}$ (например, погода или инциденты по каждому узлу) конкатенируются по признаковому измерению с входными данными:

$$X' = \text{Concat}_{\text{channels}}(X, Z) \in \mathbb{R}^{B \times T_{\text{in}} \times N \times (C_{\text{in}} + C_{\text{exo}})},$$

и обрабатываются стандартным патчингом и трансформерным кодировщиком.

2. *Глобальные временные ковариаты.* Пусть $G \in \mathbb{R}^{B\times C_g}$ обозначает признаки, общие для всех узлов в пределах рассматриваемого окна, например агрегированную погоду или календарные индикаторы. Эти признаки проецируются малой полносвязной сетью $\varphi : \mathbb{R}^{C_g} \to \mathbb{R}^d$ и добавляются ко всем патч-токенам:

$$\tilde{E}_i = E_i + \varphi(G), \qquad \tilde{E}_i, E_i \in \mathbb{R}^{B\times L\times d}, \tag{2.5}$$

где прибавление выполняется с трансляцией по измерению токенов на длину L. Здесь $E_i \in \mathbb{R}^{B\times L\times d}$ обозначает последовательность патч-токенов для узла i, полученную после патчинга временного ряда и линейной проекции каждого патча в d-мерное пространство.

3. *Статические узловые признаки* $S \in \mathbb{R}^{N\times C_s}$, например тип дороги или число полос, преобразуются через вложение $\psi : \mathbb{R}^{C_s} \to \mathbb{R}^d$ и добавляются к узловым представлениям после пулинга:

$$h_i \leftarrow h_i + \psi(S_i).$$

Экзогенные переменные также могут влиять на обучаемую матрицу смежности. Пусть E_1, E_2 — матрицы узловых вложений, используемые при построении аффинностей. Для их модуляции можно использовать FiLM-подобное преобразование или гейтинг:

$$\hat{E}_1 = \gamma_1 \odot E_1 + \beta_1, \quad \hat{E}_2 = \gamma_2 \odot E_2 + \beta_2,$$

где $(\gamma_\cdot, \beta_\cdot) = f_{\text{cond}}(G, S)$.

После этого матрица смежности A формируется через softmax по строкам уже на модулированных вложениях $\hat{E}_1, \hat{E}_2$. Это даёт возможность учитывать погоду, календарь или события без изменения основного вычислительного графа модели. Календарные признаки, включая время суток, день недели и праздники, могут входить в G или Z.

Патч-ориентированный временной кодировщик

Временной модуль следует принципу PatchTST [133]. Для узла i с последовательностью $x_i \in \mathbb{R}^{T_{\text{in}}\times C_{\text{in}}}$ извлекаются L перекрывающихся фрагментов длины P и шага s:

$$U_i = \mathcal{P}(x_i; P, s) \in \mathbb{R}^{L\times(PC_{\text{in}})}. \tag{2.6}$$

Каждый фрагмент линейно проецируется в пространство размерности d, к нему добавляются позиционные кодировки, после чего применяется стек пренормированных слоёв трансформерного кодировщика:

$$E_i = U_i W_p + b_p \in \mathbb{R}^{L\times d}, \qquad Z_i = \mathrm{TransformerEncoder}\big(\mathrm{LN}(E_i + \mathrm{PE})\big) \in \mathbb{R}^{L\times d}. \tag{2.7}$$

Здесь $W_p \in \mathbb{R}^{(PC_{in})\times d}$ и $b_p \in \mathbb{R}^d$ — обучаемые параметры линейной проекции, отображающей векторизованный патч размерности PC_{in} в d-мерное пространство эмбеддингов. Иными словами, для каждого патча $u_{i,\ell} \in \mathbb{R}^{PC_{in}}$ соответствующий патч-токен вычисляется как $e_{i,\ell} = u_{i,\ell} W_p + b_p \in \mathbb{R}^d$. Для краткости в обозначениях выше опущено измерение батча. В реализации $U_i \in \mathbb{R}^{B\times L\times (PC_{in})}$ и $E_i \in \mathbb{R}^{B\times L\times d}$, а операция умножения на W_p выполняется независимо для каждого элемента батча.

Представление узла формируется усреднением по всем патч-токенам:

$$h_i = \frac{1}{L}\sum_{\ell=1}^{L} Z_{i\ell} \in \mathbb{R}^d. \tag{2.8}$$

Усреднение по патч-токенам даёт компактное представление узла. Локальная динамика сохраняется внутри патчей, а итоговый вектор используется как вход для пространственного блока. Объединение вложений всех узлов даёт начальное пространственное представление:

$$H^{(0)} = \mathrm{Stack}(h_1, \ldots, h_N) \in \mathbb{R}^{B\times N\times d}. \tag{2.9}$$

Адаптивное обучение графа

Пространственные зависимости в PTSTG задаются через обучаемую стохастическую по строкам матрицу смежности. Для её построения используются две матрицы узловых вложений $E_1, E_2 \in \mathbb{R}^{N\times r}$, из которых вычисляется неотрицательная матрица аффинностей:

$$S = \mathrm{ReLU}(E_1 E_2^\top) \in \mathbb{R}^{N\times N}, \qquad \tilde{S} = \tfrac{1}{2}(S + S^\top). \tag{2.10}$$

$E_2^\top \in \mathbb{R}^{r\times N}$ - транспонированная матрица $E_2 \in \mathbb{R}^{N\times r}$, а $S^\top$ - транспонированная матрица S. Симметризация $\tilde{S} = \frac{1}{2}(S + S^\top)$ применяется для получения симметричной матрицы аффинностей. На следующем шаге матрица смежности A, стохастическая по строкам, строится с помощью softmax. Softmax применяется по строкам с возможной температурой $\tau > 0$. Из этого получаем матрицу

смежности:

$$A = \mathrm{softmax}_{\mathrm{row}}\left(\frac{\tilde{S}}{\tau}\right) \in \mathbb{R}^{N\times N}, \qquad A\mathbf{1} = \mathbf{1}. \tag{2.11}$$

Здесь $\mathbf{1} \in \mathbb{R}^N$ обозначает вектор, все компоненты которого равны единице. Равенство $A\mathbf{1} = \mathbf{1}$ означает, что A является стохастической по строкам матрицей:

$$\sum_{j=1}^{N} A_{ij} = 1, \qquad i = 1,\ldots,N.$$

Каждая строка A задаёт распределение весов по узлам, участвующим в смешивании признаков. Операция AH поэтому интерпретируется как взвешенное усреднение узловых представлений.

Такой способ построения графа близок к адаптивным графам в пространственно-временных GNN [141; 143]. Для крупных сетей матрица A может быть разрежена: в каждой строке сохраняются только k наибольших значений, после чего строка нормируется повторно. Это уменьшает стоимость межузлового смешивания и не требует изменения остальных блоков модели.

Блок узлового графа

Пусть $H^{(\ell)} \in \mathbb{R}^{B\times N\times d}$ — вложения узлов на слое ℓ. Графовый блок выполняет межузловое смешивание через матрицу A, после чего применяется нормализация и позиционный полносвязный блок:

$$\widehat{H} = H^{(\ell)} + (AH^{(\ell)})W_\ell, \quad W_\ell \in \mathbb{R}^{d\times d}, \tag{2.12}$$

$$\widetilde{H} = \mathrm{LN}(\widehat{H}), \tag{2.13}$$

$$H^{(\ell+1)} = \widetilde{H} + \mathrm{MLP}(\widetilde{H}), \tag{2.14}$$

где $\mathrm{MLP}(x) = \mathrm{Drop} \circ W_2 \circ \mathrm{GELU} \circ \mathrm{Drop} \circ W_1(x)$, $W_1 \in \mathbb{R}^{d\times 4d}$, $W_2 \in \mathbb{R}^{4d\times d}$. Здесь Drop обозначает dropout, LN — нормализацию представлений, а MLP — позиционное полносвязное нелинейное преобразование признаков. Умножение $(AH^{(\ell)})$ выполняется с трансляцией по батчу: для каждого b вычисляется $(AH^{(\ell)})_b = A\,H_b^{(\ell)}$. Межузловое взаимодействие остаётся линейным, поэтому пространственное уточнение не становится тяжёлым вычислительным блоком.

Прогнозирующий блок

После применения L_g графовых блоков вложения узлов нормализуются и линейно проецируются на полный прогнозный горизонт:

$$\bar{H} = \mathrm{LN}\big(H^{(L_g)}\big), \qquad Y^{\star} = \mathrm{MLP}_{\mathrm{head}}(\bar{H}) \in \mathbb{R}^{B\times N\times (HC_{\mathrm{out}})}. \tag{2.15}$$

Результирующий тензор преобразуется к стандартному виду прогноза:

$$\hat{Y} = \mathrm{Reshape}(Y^{\star}) \in \mathbb{R}^{B\times H\times N\times C_{\mathrm{out}}}. \tag{2.16}$$

Такой способ декодирования не использует авторегрессионное развёртывание. Все шаги горизонта формируются сразу, что уменьшает риск переноса ошибки с одного прогнозного шага на следующий. Метод представлена на рисунке 2.1.

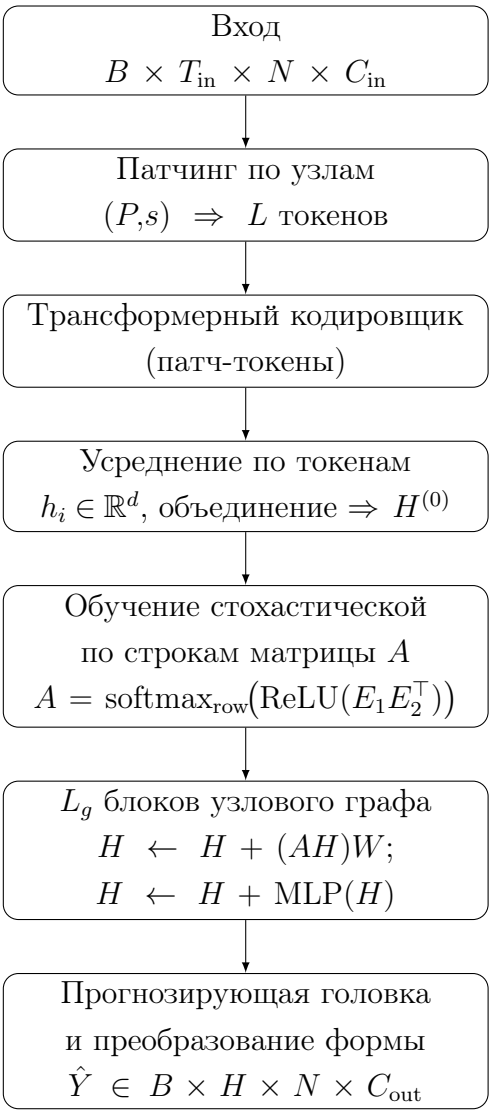

Рисунок 2.1 — PTSTG: патч-уровневое временное кодирование, обучаемая матрица смежности A, лёгкое графовое уточнение и однопроходная многопрогнозная головка.

Алгоритмическое описание

Последовательность операций PTSTG приведена в алгоритме 1. В нём отдельно показаны патч-ориентированное кодирование временных рядов, построение адаптивной матрицы смежности и графовое уточнение узловых представлений. Таблица 9 фиксирует отличие PTSTG от базового PatchTST: временной кодировщик сохраняется, но после него добавляется пространственный блок.

В алгоритме используются обозначения, введённые выше: P — длина патча, s — шаг сдвига, L — число патчей, d — размерность вложений, r — ранг графовой матрицы, L_g — глубина графового стека. Операция softmax применяется по строкам, если не указано иное.

Algorithm 1 Прямой проход и цикл обучения PTSTG

Require: Обучающая выборка $\mathcal{D} = \{(X,Y)\}$; длина патча P; шаг сдвига s; размерность вложений d; число графовых слоёв L_g; оптимизатор Adam; функция потерь $\mathcal{L}$.

Ensure: Прогноз на полный горизонт $\hat{Y}$.

Инициализировать трансформерный кодировщик, проекционные слои, вложения графа E_1, E_2, веса графовых блоков $\{W_\ell\}$ и прогнозирующую головку.

for each минибатч (X,Y) **do**

  Временное кодирование: для каждого узла i вычислить $U_i \leftarrow \mathcal{P}(X[:, :, i, :]; P, s)$;

  $Z_i \leftarrow \mathrm{TransformerEncoder}(\mathrm{LN}(U_i W_p + b_p + \mathrm{PE}))$;

  $h_i \leftarrow \frac{1}{L}\sum_\ell Z_{i\ell}$.

  $H^{(0)} \leftarrow \mathrm{Stack}(h_1, \ldots, h_N)$.

  Адаптивная матрица смежности: $S \leftarrow \mathrm{ReLU}(E_1 E_2^\top)$; $\tilde{S} \leftarrow \frac{1}{2}(S + S^\top)$; $A \leftarrow \mathrm{softmax}_{\mathrm{row}}(\tilde{S})$.

  Графовое уточнение: для $\ell = 0, \ldots, L_g - 1$:

  $\widehat{H} \leftarrow H^{(\ell)} + (A H^{(\ell)}) W_\ell$;

  $\widetilde{H} \leftarrow \mathrm{LN}(\widehat{H})$;

  $H^{(\ell+1)} \leftarrow \widetilde{H} + \mathrm{MLP}(\widetilde{H})$.

  Прогноз: $\hat{Y} \leftarrow \mathrm{Reshape}\big(\mathrm{MLP}_{\mathrm{head}}(\mathrm{LN}(H^{(L_g)}))\big)$.

  Вычислить $\mathcal{J} = \mathcal{L}(\hat{Y}, Y)$ и обновить параметры.

end for

Таблица 9 — Сравнение PatchTST с предлагаемым методом PTSTG

Аспект	**PatchTST (базовая модель)**	**PTSTG (предлагаемая)**
Обработка входа	Независимый по каналам временной кодировщик для каждого узла/ряда	Тот же временной кодировщик по узлам, затем пространственное уточнение вложений узлов
Временные токены	Перекрывающиеся патчи фиксированной длины, позиционные кодировки, трансформерный кодировщик	Идентичный патчинг и кодировщик; выходы усредняются в узловые вложения
Пространственное моделирование	Отсутствует (чисто временная модель)	Обучаемая стохастическая по строкам матрица смежности и лёгкое графовое уточнение
Источник смежности	Не используется	Извлекается из данных, обучается, при выводе может быть разрежена
Пространственная операция	Неприменима	Линейное агрегирование соседей и позиционный MLP с остаточными связями
Стратегия декодирования	Однопроходная многопрогнозная головка	Та же головка (идентичное использование)
Основная цель	Эффективное извлечение долгосрочных временных паттернов	Временная эффективность *плюс* обучаемое пространственное взаимодействие
Вычислительная сложность	Определяется вниманием по патчам во времени	Добавляется дешёвое межузловое смешивание; возможно разрежение top-k
Поведение при обучении	Стабильная оптимизация за счёт патчинга	Аналогичная стабильность; обучение смежности сходится быстро на практике
Режим вывода	Один прямой проход	Один прямой проход; масштабируемость обеспечивается разреженной смежностью
Интерпретируемость	Только временные паттерны внимания	Временное внимание u интерпретируемая обученная смежность между узлами

Вычислительная сложность

Пусть $L = \lfloor (T_{\text{in}} - P)/s \rfloor + 1$ — число патч-токенов. Для одного временного слоя сложность самовнимания составляет $\mathcal{O}(BNL^2d)$, а сложность полносвязного подслоя — $\mathcal{O}(BNLd^2)$. Память для активаций в блоке внимания оценивается как $\mathcal{O}(BNL^2d)$. Построение адаптивного графа требует вычисления произведе-

ния $E_1 E_2^\top$ и имеет сложность $\mathcal{O}(N^2 r)$, где r задаёт размерность латентных узловых вложений.

Каждый блок узлового графа добавляет $\mathcal{O}(BN^2 d)$ операций за счёт умножения (AH) и $\mathcal{O}(BNd^2)$ за счёт MLP. Если применяется top-k разрежение по строкам матрицы A, пространственный член уменьшается до $\mathcal{O}(BNkd)$, а объём памяти для хранения смежности — до $\mathcal{O}(Nk)$. Поэтому для больших сенсорных сетей именно разрежение матрицы смежности является основным средством контроля вычислительной стоимости.

Целевая функция обучения

Обучение выполняется в прямой многогоризонтной постановке. Функция потерь задаётся как:

$$\mathcal{L}(\hat{Y}, Y) = \sum_{t=1}^{H} w_t \, \|\hat{Y}_{(:,t,:,:)} - Y_{(:,t,:,:)}\|_p, \qquad p \in \{1, 2\}, \tag{2.17}$$

где $w_t \geqslant 0$ — веса горизонтов, которые по умолчанию выбираются равномерными. Оптимизатор, расписание скорости обучения и регуляризация относятся к процедуре обучения, а не к структуре отображения $\hat{Y} = \Phi_\theta(X)$. Поэтому их значения фиксируются в экспериментальном разделе.

В экспериментах в качестве критерия оптимизации используется среднеквадратичная ошибка (MSE), что соответствует выбору $p = 2$ при равномерных весах $w_t \equiv 1$. Ранняя остановка также выполняется по значению MSE на валидационной выборке.

Свойства обучаемой матрицы смежности

Матрица A является стохастической по строкам, поскольку $A\mathbf{1} = \mathbf{1}$ по построению (2.11). При этом она не обязана быть симметричной: softmax применяется построчно и может давать разные веса для пар (i,j) и (j,i). Если требуется симметричная разреженность, можно использовать симметричную нормализацию $D^{-1/2} A D^{-1/2}$, проекции Синьорна или взаимный отбор top-k. С точки зрения численной устойчивости важно, что смешивание через A задаёт выпуклые комбинации узловых вложений.

Подробности реализации

В реализации используются трансформерные блоки с преднормировкой и GELU-активацией. Линейные слои инициализируются усечённым нормальным распределением. Перед временным кодировщиком добавляются синусоидальные позиционные кодировки. В базовой версии применяется усреднение по патч-токенам. При необходимости оно может быть заменено на attention-пулинг без изменения остальных блоков модели. Для больших значений N применяется top-k разрежение матрицы A с последующей перенормировкой строк.

Учет дополнительных признаков

Дополнительные признаки включаются в модель методом описанным выше. Поузловые временные ковариаты конкатенируются с входным тензором по канальному измерению. Глобальные признаки проецируются в пространство размерности d и добавляются к патч-токенам. Статические признаки узлов добавляются к $H^{(0)}$. Если требуется учитывать влияние контекста на пространственные связи, эти признаки используются для модуляции вложений, из которых строится матрица A.

Ограничения

У базовой версии PTSTG есть три ограничения. Без разрежения операция (AH) имеет квадратичную сложность по числу узлов N. Усреднение по патчам может сглаживать отдельные короткие изменения в нестационарных рядах. Кроме того, матрица A в текущей формулировке не зависит от конкретного входного окна. Динамические графы и многоуровневое патчирование совместимы с предложенным методом, но в данной работе рассматриваются как расширения базовой модели.

Динамическая смежность

В базовой реализации матрица A является статической относительно входного окна. Это упрощает обучение и вывод, но ограничивает реакцию модели на смену режима движения. Для пиковых периодов, инцидентов или временных ограничений пропускной способности возможна модификация, при которой матрица A_t формируется по текущему окну. Один из вариантов состоит в использовании гиперсети f_φ, которая модулирует вложения E_1, E_2 на основе

узловых представлений $\{h_i\}$:

$$A_t = \mathrm{softmax}_{\mathrm{row}}\left(\frac{\mathrm{ReLU}(\tilde{E}_1\tilde{E}_2^{\top})}{\tau}\right). \tag{2.18}$$

Эксперименты

Экспериментальная оценка PTSTG проводится на наборах данных со скоростями транспортного потока. Сравнение выполняется с базовыми моделями трёх групп: чисто временными, графовыми и гибридными пространственно-временными архитектурами. Основные критерии сравнения — точность на горизонтах 15, 30 и 60 минут и вычислительная пригодность модели для последующего развёртывания.

Наборы данных

Эксперименты выполнены в соответствии с устоявшимся протоколом [129; 141].

METR–LA. Данные о скорости движения, собранные с интервалом 5 минут 207 индуктивными петлями в Лос-Анджелесе в период с марта по июнь 2012 года. Временные ряды выровнены по времени. Пропуски в данных восполняются линейной интерполяцией при коротких разрывах; разрывы, превышающие заданный порог, маскируются.

PEMS–BAY. Скорости, измеренные 325 сенсорами в районе Сан-Франциско с 5-минутным интервалом в период с января по май 2017 года. Предобработка идентична METR–LA: выравнивание, интерполяция коротких пропусков, маскирование длинных.

LargeST. Масштабный бенчмарк, объединяющий сети городских дорог нескольких мегаполисов. Каждый город представлен десятками тысяч сенсоров с 5-минутной дискретизацией. Набор предоставляет стандартизированные разбиения на обучающую, валидационную и тестовую выборки, а также единый метод построения графа (например, географический k-ближайших соседей), что позволяет проводить сопоставимые эксперименты на крупных сетях. Использованы официальные разбиения и протокол оценки.

Все временные ряды стандартизованы (z-оценка) по каждому узлу и каналу отдельно с использованием только статистик обучающей выборки. Для METR–LA и PEMS–BAY применено хронологическое разбиение в пропорции 70%/10%/20% на обучающую, валидационную и тестовую части. Для LargeST

использованы официальные разбиения. Если не указано иное, приводятся результаты на горизонтах {3,6,12} шагов, что соответствует 15, 30 и 60 минутам.

Метрики оценки

В качестве основных метрик используются средняя абсолютная ошибка (MAE) и корень из среднеквадратичной ошибки (RMSE). Пусть $\{\hat{y}_{t,n}, y_{t,n}\}$ — прогноз и истинное значение для момента времени t и узла n. Агрегация проводится по всем моментам тестовой выборки $\mathcal{T}$ и всем узлам $\mathcal{N}$, где $M = |\mathcal{T}||\mathcal{N}|$:

$$\mathrm{MAE} = \frac{1}{M}\sum_{t\in\mathcal{T}}\sum_{n\in\mathcal{N}}\left|\hat{y}_{t,n} - y_{t,n}\right|, \quad \mathrm{RMSE} = \sqrt{\frac{1}{M}\sum_{t\in\mathcal{T}}\sum_{n\in\mathcal{N}}\left(\hat{y}_{t,n} - y_{t,n}\right)^2}. \tag{2.19}$$

Для средней абсолютной процентной ошибки (MAPE) знаменатель ограничен снизу, чтобы избежать деления на значения, близкие к нулю. В экспериментах использовано $\varepsilon = 1{,}0$ (в единицах скорости):

$$\mathrm{MAPE} = 100 \times \frac{1}{M}\sum_{t\in\mathcal{T}}\sum_{n\in\mathcal{N}}\left|\frac{\hat{y}_{t,n} - y_{t,n}}{\max(|y_{t,n}|, \varepsilon)}\right|. \tag{2.20}$$

Поскольку разные работы применяют различные значения ε и стратегии обработки пропусков, для сопоставления с литературой приоритет отдаётся MAE и RMSE. MAPE приводится исключительно для справки.

Сравнение проведено с представительными методами из трёх классов: только временных, графовых и гибридных пространственно-временных моделей. При наличии официального кода использовались рекомендованные настройки. В противном случае методы были воспроизведены по описаниям в статьях с настройкой гиперпараметров на валидационной выборке.

Классические и чисто временные модели. Среднее по истории (HA), ARIMA (с фильтром Калмана), векторная авторегрессия (VAR), метод опорных векторов (SVR), полносвязная нейронная сеть (FNN) и FC-LSTM.

Графовые пространственно-временные модели. STGCN [128], DCRNN [129], Graph WaveNet (GWNET) [141], AGCRN [143], GMAN [144], DGCRN [152].

Гибридные и с блоком внимания. STTN [147], ASTGNN [149], D^{2}STGNN [155], а также вариант PatchTST [133], обрабатывающий каждый канал независимо, — как сильная временная базовая модель без явного учёта пространственных связей.

Этот набор охватывает как трансформерные подходы, так и модели с фиксированными, адаптивными и динамическими графами. На METR–LA и

PEMS–BAY приведены результаты для устойчивых и широко используемых базовых моделей, чтобы избежать искажений из-за различий в реализациях. На крупномасштабном бенчмарке LargeST включён расширенный набор современных гибридных архитектур. В данной работе PTSTG рассматривается как компактная альтернатива существующим пространственно-временным моделям. Основной акцент сделан на сочетании точности, умеренной вычислительной сложности и возможности дальнейшего масштабирования.

Настройка обучения PTSTG

Оптимизация и регуляризация. Используется оптимизатор Adam с косинусным затуханием скорости обучения и линейным разогревом. Обучение останавливается по ранней остановке на основе MSE на валидационной выборке. Применяются обрезка градиентов по значению, весовая регуляризация и дропаут в полносвязных подслоях. Если не указано иное, результаты получены в одном запуске с фиксированным случайным зерном. Статистическая проверка значимости в данной версии не проводится.

Входные и выходные данные. Каждому узлу подаётся фиксированное окно истории длины T_{in}. Модель формирует прогноз на полный горизонт H за один проход. Стандартизация (z-оценка) выполняется отдельно по каждому узлу и каналу на основе статистик обучающей выборки и обращается при оценке.

Конфигурация модели. PTSTG включает независимый по каналам патч-ориентированный трансформерный кодировщик, обучаемую низкоранговую адаптивную матрицу смежности со строковой температурой $\tau = 1{,}0$ и лёгкий стек блоков узлового графа. По умолчанию разрежение матрицы A не применяется на этапе обучения; при выводе возможно использование top-k разрежения по строкам с последующей перенормировкой (значение k указывается при использовании). В функции потерь (2.17) используются равномерные веса $w_t = 1$.

Основные результаты

Результаты экспериментов приведены для трёх наборов данных и трёх горизонтов прогноза. Основными метриками являются MAE и RMSE. MAPE используется как дополнительный показатель, поскольку его значение зависит от способа обработки малых скоростей и выбора нижнего ограничения знаменателя.

METR–LA

На наборе METR–LA метод PTSTG показывает качество, близкое к сильным графовым моделям на коротком и среднем горизонтах. По таблице 10 видно, что на горизонте 15 минут модель даёт низкое значение RMSE, но по MAE и MAPE уступает DCRNN. На горизонтах 30 и 60 минут разрыв относительно лучшей модели увеличивается. Этот результат указывает, что базовая конфигурация PTSTG лучше проявляет себя на коротком и среднем прогнозном интервале, тогда как дальний горизонт требует более ёмкого пространственного блока или дополнительной настройки графовой части.

Таблица 10 — Результаты на METR-LA на горизонтах 15/30/60 мин (меньше — лучше). Значения базовых моделей взяты дословно из литературы (например,[129; 141]). Повторный запуск не проводился. Результаты PTSTG получены в одном эксперименте с фиксированным зерном (`seed = 2023`). Незначительные расхождения между публикациями могут быть вызваны различиями в предобработке и соглашениях по MAPE (ε). Лучшие значения выделены **жирным**.

Горизонт (T)	**Метрика**	**HA**	**$ARIMA_{Kal}$**	**VAR**	**SVR**	**FNN**	**FC-LSTM**	**DCRNN**	**PTSTG**
15 мин	MAE	4.16	3.99	4.42	3.99	3.99	3.44	**2.77**	2.85
	RMSE	7.80	8.21	7.89	8.45	7.94	6.30	5.38	**5.33**
	MAPE	13.0%	9.60%	10.2%	9.3%	9.9%	9.6%	**7.3%**	7.7%
30 мин	MAE	4.16	5.15	5.41	5.05	4.23	3.77	**3.15**	3.39
	RMSE	7.80	10.45	9.13	10.87	8.17	7.23	6.45	**6.41**
	MAPE	13.0%	12.7%	12.7%	12.1%	12.9%	10.9%	**8.8%**	9.7%
1 ч	MAE	4.16	6.90	6.52	6.72	4.49	4.37	**3.60**	4.19
	RMSE	7.80	13.23	10.11	13.76	8.69	8.69	**7.59**	7.69
	MAPE	13.0%	17.4%	15.8%	16.7%	14.0%	13.2%	**10.5%**	12.67%

PEMS–BAY

На наборе PEMS–BAY картина отличается. PTSTG сохраняет сопоставимый порядок ошибки на коротком горизонте, но при увеличении горизонта уступает DCRNN заметнее, чем на METR–LA. Вероятная причина состоит в том, что в базовой версии используется статическая обучаемая матрица смежности. Она отражает усреднённые функциональные связи между узлами, но не перестраивается для каждого входного окна. Поэтому при изменении режима движения, особенно на дальнем горизонте, ошибка может расти быстрее.

Таблица 11 — Результаты на PEMS-BAY на горизонтах 15/30/60 мин (меньше — лучше). Значения базовых моделей взяты дословно из литературы (например, [129; 141]); повторный запуск не проводился. Результаты PTSTG получены в одном эксперименте с фиксированным зерном (`seed = 2023`). Ввиду несогласованности определений MAPE в предшествующих работах основное сравнение следует проводить по MAE и RMSE. Лучшие значения выделены **жирным**.

Горизонт (T)	**Метрика**	**HA**	**ARIMA$_{Kal}$**	**VAR**	**SVR**	**FNN**	**FC-LSTM**	**DCRNN**	**PTSTG**
15 мин	MAE	2.88	1.62	1.74	1.85	2.20	2.05	**1.38**	1.55
	RMSE	5.59	3.30	3.16	3.59	4.42	4.19	**2.95**	3.12
	MAPE	6.8%	3.5%	3.6%	3.8%	5.19%	4.8%	**2.9%**	3.43%
30 мин	MAE	2.88	2.33	2.32	2.48	2.30	2.20	**1.74**	2.08
	RMSE	5.59	4.76	4.25	5.18	4.63	4.55	**3.97**	4.28
	MAPE	6.8%	5.4%	5.0%	5.5%	5.43%	5.2%	**3.90%**	4.84%
1 ч	MAE	2.88	3.38	2.93	3.28	2.46	2.37	**2.07**	2.81
	RMSE	5.59	6.50	5.44	7.08	4.98	4.96	**4.74**	5.54
	MAPE	6.8%	8.3%	6.50%	8.0%	5.89%	5.7%	**4.90%**	6.81%

LargeST

На LargeST метод PTSTG показывает более сильный результат относительно рассматриваемых базовых моделей. По таблице 12 видно, что модель не обязательно является лучшей на каждом отдельном горизонте, однако даёт хорошие средние значения по MAE, RMSE и MAPE. Для данной работы это важно, поскольку LargeST лучше отражает задачу масштабирования на крупную сенсорную сеть. Здесь преимущество PTSTG связано с тем, что временной кодировщик работает по патчам, а пространственное смешивание может быть ограничено разреженной матрицей смежности.

Таблица 12 — Результаты на бенчмарке LargeST (стандартное отклонение) на горизонтах {3,6,12} (меньше — лучше). Значения базовых моделей взяты из официальных отчётов LargeST и последующих публикаций (например, [156] для D^{2}STGNN); повторный запуск не проводился. Результаты PTSTG получены в одном эксперименте с фиксированным зерном (`seed = 2023`). Численные различия могут возникать из-за вариаций в охвате городов, построении графа и способе ограничения знаменателя в MAPE. Лучшие значения выделены **жирным**.

Метод	Парам.	Гор. 3			Гор. 6			Гор. 12			Среднее		
		MAE	RMSE	MAPE	MAE	RMSE	MAPE	MAE	RMSE	MAPE	MAE	RMSE	MAPE
HL	–	33.61	50.97	20.77	57.80	84.92	37.73	101.74	140.14	76.84	60.79	87.40	41.88
LSTM	98K	19.03	30.53	11.81	25.84	40.87	16.44	37.63	59.07	25.45	26.44	41.73	17.20
DCRNN	373K	17.14	27.47	11.12	20.99	33.29	13.95	26.99	42.86	18.67	21.03	33.37	14.13
AGCRN	761K	15.71	27.85	11.48	18.06	31.51	13.06	21.86	39.44	16.52	18.09	32.01	13.28
STGCN	508K	17.45	29.99	12.42	19.55	33.69	13.68	23.21	41.23	16.32	19.67	34.14	13.86
GWNET	311K	15.24	25.13	9.86	17.74	29.51	11.70	21.56	36.82	15.13	17.74	29.62	11.88
ASTGCN	2.2M	19.56	31.33	12.18	24.13	37.95	15.38	30.96	49.17	21.98	23.70	37.63	15.65
STTN	114K	16.22	26.22	10.63	18.76	30.98	12.80	22.62	39.09	16.14	18.69	31.11	12.82
STGODE	729K	16.75	28.04	11.00	19.71	33.56	13.16	23.67	42.12	16.58	19.55	33.57	13.22
DSTAGNN	3.9M	18.13	28.96	11.38	21.71	34.44	13.93	27.51	43.95	19.34	21.82	34.68	14.40
DGCRN	243K	15.34	25.35	10.01	18.05	30.06	11.90	22.06	37.51	15.27	18.02	30.09	12.07
D^{2}STGNN	406K	**14.92**	**24.95**	**9.56**	**17.52**	**29.24**	**11.36**	22.62	37.14	14.86	17.85	29.51	11.54
PTSTG	880K	16.42	26.42	11.04	18.86	30.37	12.93	22.69	**36.42**	16.59	**17.50**	**28.18**	**11.07**

Вычислительные характеристики

В таблице 13 приведены вычислительные характеристики PTSTG. Эти данные нужны для оценки не только точности, но и практической применимости модели. В рассматриваемой конфигурации модель содержит менее одного миллиона параметров и формирует прогноз на полный горизонт за один прямой проход. Это делает её пригодной для экспериментов на одном GPU и снижает требования к вычислительной инфраструктуре по сравнению с более тяжёлыми пространственно-временными архитектурами.

Таблица 13 — Вычислительные характеристики PTSTG на одном потребительском GPU. Аппаратное обеспечение: NVIDIA GeForce RTX 4060 Ti (8 ГБ), FP32; версия PyTorch: 2.8.0. Входные данные: батч $B = 32$, длина последовательности Seq_len $= 36$, горизонт Horizon $= 12$. Операции (FLOPs) указаны на один прямой проход при $B = 1$; задержка и пропускная способность — на один минибатч ($B = 32$).

Модель	**Парам. (М)**	**FLOPs (Г)**	**Обучение /эпоха (мин)**	**Средн. задержка (мс)**	**p95 задержка (мс)**	**Пропускн. способность (посл./с)**	**Пик VRAM (ГБ)**
PTSTG (наша)	0.88	119.163	3.101	56.44	61.54	564.04	3.492

Абляции и эффективность

Для оценки вклада основных проектных решений рассматривались три компонента PTSTG: патч-ориентированное временное кодирование, обучаемая матрица смежности и глубина графового уточнения. Патчинг уменьшает длину последовательности внимания и делает обработку длинного входного окна менее затратной. Обучаемая матрица смежности позволяет учитывать функциональные связи между сенсорами, которые не всегда совпадают с географической близостью. Графовый стек уточняет узловые представления, но при увеличении глубины может повышать вычислительную стоимость и усиливать сглаживание признаков.

В данной версии экспериментов основной акцент сделан на сравнении PTSTG с базовыми моделями и на проверке вычислительной пригодности архитектуры. Полный абляционный анализ по нескольким случайным зернам и вариантам top-k разрежения требует отдельного вычислительного эксперимента. Поэтому приведённые результаты следует рассматривать как проверку базовой конфигурации метода, а не как исчерпывающее исследование всех гиперпараметров.

PTSTG реализована на PyTorch. Данные стандартизуются по узлам и каналам на основе статистик обучающей выборки. Обучение выполняется с оптимизатором Adam, косинусным затуханием скорости обучения, ранней остановкой по MSE на валидационной выборке, обрезкой градиентов, dropout и весовой регуляризацией. Точные параметры обучения и архитектуры приведены в таблицах 14 и 15.

Таблица 14 — Гиперпараметры обучения и оптимизации для PTSTG. Seq_len — длина входного окна в 5-минутных шагах; Horizon — длина прогноза в шагах.

Набор данных	Seq_len	Horizon	Шаг	Батч	Макс. эпох	Терп	Скор. обуч.	Рег. весов	Обр. град.	Дроп	Потери	Зерно
METR-LA	36	12	5 мин	32	50	5	1×10^{-4}	1×10^{-3}	5.0	0.30	MSE	2023
PEMS-BAY	36	12	5 мин	64	50	5	1×10^{-4}	1×10^{-3}	5.0	0.30	MSE	2023
LargeST (SD)	36	12	5 мин	32	50	5	1×10^{-4}	1×10^{-3}	5.0	0.30	MSE	2023

Таблица 15 — Архитектурные гиперпараметры PTSTG. d_{model} — размерность вложений; `n_heads`/`n_layers` относятся к трансформерному кодировщику PatchTST; `patch_len`/`stride` задают параметры временного патчинга; `graph_rank` определяет ранг низкоранговой адаптивной матрицы смежности; `graph_layers` — число блоков узлового графа.

Набор данных	d_{model}	n_heads	n_layers	patch_len	stride	graph_rank	graph_layers	In_dim	Out_dim	Drop
METR-LA	128	4	2	8	4	12	3	1	1	0.30
PEMS-BAY	128	4	2	12	6	16	3	1	1	0.30
LargeST (SD)	128	4	2	8	4	12	3	1	1	0.30

Все результаты PTSTG получены при фиксированном случайном зерне 2023. Такой протокол позволяет сопоставить базовую конфигурацию модели с опубликованными значениями, но не заменяет статистическую проверку по нескольким независимым запускам. Поэтому результаты следует трактовать как экспериментальную оценку предложенной архитектуры, а не как окончательное утверждение о превосходстве над всеми альтернативами.

При практическом использовании модели необходимо учитывать неоднородность ошибок по участкам сети. Качество прогноза может различаться для магистралей, городских артериальных дорог, участков с высокой плотностью сенсоров и зон с нестабильными измерениями. Поэтому при внедрении PTSTG целесообразно дополнительно оценивать MAE и RMSE по типам дорог, районам и временным периодам.

Отдельное ограничение связано с пропусками данных и отказами сенсоров. В экспериментальном протоколе короткие пропуски интерполируются, а длинные разрывы маскируются. В реальной ИТС возможны более сложные

ситуации: дрейф показаний, асинхронные временные метки, длительная потеря отдельных узлов и изменение состава сенсорной сети. Для таких условий требуется отдельная проверка устойчивости модели.

Матрица смежности A в базовой версии PTSTG обучается как статический параметр и не зависит от конкретного входного окна. Это упрощает обучение и вывод, но ограничивает реакцию модели на резкие изменения дорожного режима. Поэтому дальнейшее развитие метода связано с динамической смежностью, включением экзогенных признаков, разрежением графа и расширенным абляционным анализом.

2.3 Выводы по главе

В главе рассмотрена задача прогнозирования макроскопических характеристик транспортного потока по данным сенсорной сети. Анализ существующих подходов показал, что рекуррентные модели хорошо описывают локальную временную динамику, но ограничены последовательной обработкой. Графовые пространственно-временные модели учитывают структуру УДС, однако часто используют заранее заданную матрицу смежности. Трансформерные модели и патч-ориентированные архитектуры лучше работают с длинным временным контекстом, но в базовой форме не задают явного пространственного взаимодействия между сенсорами.

Для учёта этих ограничений разработан метод PTSTG. Его временная часть основана на патч-ориентированном кодировании рядов, а пространственная часть использует обучаемую стохастическую по строкам матрицу смежности. Такая матрица формируется из латентных вложений узлов и не требует полного задания дорожной топологии. После временного кодировщика выполняется лёгкое графовое уточнение узловых представлений, а прогноз на все горизонты формируется за один прямой проход.

Отдельное значение имеет способ построения графа. В PTSTG связи между узлами определяются не только физической близостью сенсоров, но и статистикой данных. Поэтому модель может учитывать функциональные зависимости между участками сети, которые возникают из-за маршрутизации, координации светофорного управления и перераспределения спроса. При этом

вычислительная стоимость пространственного блока может быть снижена за счёт top-k разрежения матрицы смежности.

Эксперименты проводились на наборах METR–LA, PEMS–BAY и LargeST для горизонтов 15, 30 и 60 минут. На METR–LA метод показывает близкое качество к сильным графовым моделям на коротком и среднем горизонтах. На PEMS–BAY ошибка быстрее растёт при увеличении горизонта, что указывает на ограничение статической обучаемой смежности. На LargeST результаты наиболее устойчивы относительно рассматриваемых базовых моделей, что важно для проверки применимости метода на крупной сенсорной сети.

Полученные результаты не следует трактовать как универсальное превосходство PTSTG над всеми существующими архитектурами. Более корректный вывод состоит в том, что предложенный метод реализует компромисс между точностью, числом параметров и вычислительной простотой. Метод сохраняет преимущества патч-ориентированного временного кодировщика и добавляет явный механизм межузлового взаимодействия без перехода к тяжёлому полному пространственно-временному вниманию.

Основные ограничения базовой версии связаны с тремя факторами. Во-первых, без разрежения операция межузлового смешивания имеет квадратичную сложность по числу сенсоров. Во-вторых, усреднение по патчам может сглаживать короткие нестационарные изменения. В-третьих, матрица смежности A не зависит от конкретного входного окна. Дальнейшее развитие метода связано с динамической смежностью, включением экзогенных признаков, расширенной абляционной проверкой и стратифицированной оценкой ошибок по типам участков сети.

Глава 3. Разработка метода и средств прогнозирования микроскопических характеристик транспортного потока

Исследование и разработка методов управления УДС требует качественного моделирования, которое могло бы дать картину динамики транспортных потоков в зависимости от различных условий. [63] Для этого существуют определенные модели, среди которых в рамках данной главы будут рассмотрены модели следования за лидером. Суть данных моделей заключается в изучении влияния лидирующего автомобиля на следующие за ним, где влияние представляется в виде зависимости оптимальной скорости от дистанции до впереди идущего. [55] Их использование поможет исследовать динамику движения и прогнозировать поведение транспортного потока в различных ситуациях.

Для моделирования часто используются данные, которые были получены в условиях, не соответствующих реальным процессам дорожного движения. Это могут быть испытательные полигоны, небольшие участки дороги в условиях хорошего освещения и с благоприятной погодой, данные из одного конкретного региона. Кроме того, в реальном движении могут происходить различные случайные события, обгоны, задержки, также встречаются разные скоростные режимы.

Таким образом, существует реальная потребность в дополнительном изучении вопроса сбора данных для построения процесса эффективного моделирования транспортных потоков. Необходимо подобрать такие методики и типы данных, чтобы модели смогли адаптироваться к существующим условиям и работать адекватно. [37]

За последние годы было представлено большое количество научных публикаций в области исследования дорожного движения, основанных на заранее собранных данных или включающих сбор данных по новой методике [157; 158]. В данной главе будет рассматриваться модель следования за лидером, занимающая важное место в исследованиях в области управления транспортными потоками. Модель следования за лидером позволяет изучать поведение объектов (транспортных средств (ТС), управляемых водителем или искусственным интеллектом) при следовании друг за другом. Такое изучение может помочь лучше понять характеристики транспортного потока решить основные проблемы, присутствующие в обозначенной области: образование заторов и повышение

безопасности вождения. А также подобные исследования создают базу для распространения и улучшения автопилотируемых автомобилей.

В литературе модели следования за лидером классифицируются по различным основаниям: по используемой логике управления, физической интерпретируемости, способу калибровки, типу входных данных и принципу построения прогноза [159; 160]. В рамках настоящей работы используется укрупнённая классификация, разделяющая модели следования за лидером на две группы: математические (теоретико-ориентированные) и модели, основанные на данных. Такое деление соответствует различию между моделями, задаваемыми аналитическими зависимостями и физически интерпретируемыми параметрами, и моделями, в которых зависимость между состоянием лидера, состоянием ведомого ТС и управляющей реакцией восстанавливается по траекторным данным. Следует отметить, что данная классификация не исключает более детальных таксономий, в которых отдельно выделяются модели типа стимул-реакция, модели безопасной дистанции, психофизиологические модели, модели оптимальной скорости, модели желаемой цели, а также современные обучаемые и гибридные подходы [161; 162]. Математические модели также называют классическими или традиционными. Они базируются на принципах кинематики, что позволяет четко понимать физические процессы и описывать поведение автомобилей в различных дорожных условиях. К числу известных математических моделей относятся: модель Пайпса [163], модель General Motors (GM) [164], модель Хелли [165], модель Ньюэлла [166], модель полной разницы скоростей (FVD) [167], модель интеллектуального водителя (IDM) [168], модель Гиппса [169].

С развитием технологий и нарастающим количеством данных всё большую популярность приобретают модели следования за лидером, основанные на данных. Эти модели используют данные о траекториях движения для построения моделей, что позволяет им выявлять сложные зависимости и с хорошей точностью предсказывать поведение автомобилей на дороге. В этой категории моделей можно выделить: искусственные нейронные сети, рекуррентные нейронные сети, свёрточные нейронные сети, сети с долговременной кратковременной памятью и гибридные архитектуры (CNN-LSTM) [159; 160; 170].

В настоящее время таксономия моделей следования за лидером все еще находится в стадии разработки. В этой главе будет использована классификация, основанная на двух основных группах, упомянутых выше. Сильная сторона

математических моделей заключается в их интерпретируемости. Параметры модели имеют физический смысл — время реакции или желаемое безопасное расстояние. Данное свойство значительно расширяет их применимость для теоретических исследований. Модели такого типа не требуют значительных вычислительных мощностей и относительно просты в реализации, что способствует их широкому использованию в ПО для моделирования дорожного движения и системах управления в реальном времени.

С другой стороны, многие традиционные модели основаны на идеализированных предположениях: однородность транспортных потоков, статичность поведение водителей и беспрепятственность движения по одной полосе. В реальных условиях дорожного движения поведение водителей сильно зависит от контекста. На паттерн поведения водителя влияют психологические, погодные и ситуационные факторы, которые не всегда можно учесть с помощью фиксированного набора уравнений. В результате традиционные модели часто не могут обобщить различные дорожные условия или географический контекст. Кроме того, результаты таких моделей могут зависеть от калибровки.

Главное преимущество моделей, основанных на данных, заключается в их способности выявлять и обобщать закономерности в поведении водителя. К ним относятся задержки реакции, непоследовательные ответы и адаптация к окружающей среде. Это делает их подходящими для более гибкого, масштабируемого и обобщаемого моделирования. Однако эти модели имеют ограничения. Главной проблемой является зависимость от количества и качества данных, с помощью которых можно обеспечить адекватное обучение моделей. Данный класс моделей часто функционируют как черные ящики. Они имеют ограниченную интерпретируемость, что приводит к затруднению понимания причинно-следственных механизмов. Это может вызвать проблемы при разработке приложений, требующих гарантий безопасности.

Выбор между традиционными и основанными на данных моделями следования за лидером зависит от конкретных требований. Традиционные модели характеризуются надежностью, простотой и аналитической ясностью, что делает их подходящими для фундаментальных исследований и проектирования транспортных систем. В свою очередь, модели, основанные на данных, могут обеспечить гибкость и высокую точность прогнозирования, учитывая более широкий спектр характеристик. Однако, поскольку они требуют достаточного объема данных, они лучше всего подходят для приложений, требующих

детального моделирования поведения водителя. Кроме того, продолжаются исследования по объединению этих двух классов моделей в гибридные схему [171; 172].

При работе как с классическими моделями следования за лидером, так и с моделями, основанными на данных, исследователям необходимы наборы данных.

3.1 Анализ существующих методов прогнозирования микроскопических характеристик транспортного потока

NGSIM (Next Generation Simulation) относится к наиболее часто используемым источникам траекторных данных для микроскопического моделирования транспортного потока. Набор был сформирован Федеральным управлением автомобильных дорог США и включает участки I-80, US-101 и отдельные городские перекрёстки. Данные получены по видеозаписям с высоким временным разрешением и содержат координаты ТС, скорость, ускорение, полосу движения и дистанции до соседних автомобилей. [14]

Практическая ценность NGSIM состоит в том, что он даёт воспроизводимую основу для проверки моделей следования за лидером, перестроения и микроскопической симуляции. При этом использовать его как универсальный источник данных нельзя. В литературе отмечаются шумы траекторий, физически некорректные значения ускорений и расстояний, а также ограниченный пространственный и временной охват наблюдений. Эти недостатки особенно существенны для моделей следования за лидером, поскольку ошибка в дистанции или ускорении напрямую влияет на оценку реакции ведомого автомобиля.

Для задач настоящей работы NGSIM имеет дополнительное ограничение. Он отражает американские дорожные условия и не включает значимую для российских дорог сезонную вариативность. В нём фактически не представлены движение при снеге, гололёде, низких температурах и ухудшенной видимости. Поэтому модели, обученные только на NGSIM, требуют отдельной проверки и адаптации перед переносом на другие дорожные условия.

pNEUMA содержит траектории ТС, полученные с помощью беспилотных летательных аппаратов в городской среде Афин. Высотная съёмка позволяет

одновременно наблюдать несколько связанных участков дороги и фиксировать движение на перекрёстках, в зонах высокой плотности и в местах взаимодействия транспортных и пешеходных потоков. Данные включают координаты, скорости и ускорения с частотой 10 Гц. По сравнению с NGSIM этот набор лучше описывает сложную городскую сцену, однако его также нельзя считать универсальным. pNEUMA связан с конкретной городской инфраструктурой, локальной культурой вождения и условиями движения Южной Европы. Поэтому при использовании такого источника для обучения моделей требуется проверять переносимость результатов на другие страны, климатические условия и типы улично-дорожной сети.

HighD включает траектории тТС, полученные с помощью дронов на немецких автомагистралях. Аэрофотосъёмка обеспечивает высокую точность позиционирования и позволяет анализировать скорость, ускорение, тип ТС, положение по полосам и межполосные взаимодействия. Набор полезен для исследования перестроений, обгонов и поведения автомобилей на скоростных дорогах. Основное ограничение HighD связано с областью наблюдения. Данные собраны преимущественно на немецких автобанах и слабо описывают городские режимы, светофорное регулирование, низкоскоростные участки, зимние условия и локальные особенности движения в других странах. Поэтому HighD целесообразно использовать как качественный источник траекторий для автомагистралей, но не как единственную основу для калибровки модели в целевой транспортной среде.

Сопоставление NGSIM, pNEUMA и HighD показывает, что открытые траекторные наборы полезны для сравнения моделей, но не закрывают задачу проверки в целевых дорожных, сезонных и погодных условиях. Поэтому далее рассматривается сбор собственного корпуса микроскопических данных.

Методы сбора данных

Сбор микроскопических данных должен обеспечивать восстановление траекторий и парных взаимодействий ТС. Для моделей следования за лидером важны координаты, скорость, ускорение, направление движения, расстояние до лидера и относительная скорость. Эти признаки позволяют перейти от

отдельных измерений к временным рядам, описывающим реакцию ведомого автомобиля на изменение состояния лидера.

Географические координаты используются для восстановления маршрута и расчёта расстояния между участниками движения. При синхронной регистрации нескольких автомобилей такие данные позволяют выделять пары лидер–ведомый и проверять, движутся ли они в одном направлении. Скорость применяется как основной наблюдаемый динамический показатель. Ускорение может вычисляться по скорости, но требует фильтрации, поскольку чувствительно к шуму и ошибкам дискретизации.

Видео и фотоданные расширяют описание дорожной сцены. Их можно получать со стационарных камер, дронов, видеорегистраторов и смартфонов. По таким данным можно выявлять моменты следования, оценивать плотность потока, фиксировать перестроения и проверять расстояния между автомобилями. Ограничения связаны с ракурсом, погодой, освещённостью, перекрытиями объектов и калибровкой камеры.

Погодные условия рассматриваются как отдельный контекстный фактор. Дождь, снег, туман, гололёд и низкая освещённость изменяют сцепление с покрытием, видимость и выбор безопасной дистанции. Поэтому при сборе данных необходимо фиксировать не только кинематику ТС, но и внешние условия поездки, включая время суток, осадки, температуру и состояние дорожного покрытия.

Дополнительно могут учитываться сведения о водителе и автомобиле. Возраст, стаж вождения, привычный стиль управления и технические характеристики автомобиля влияют на выбор скорости и дистанции. Такие признаки не заменяют траекторные измерения, но помогают объяснять различия в реакции водителей при сходной дорожной обстановке.

В работе [157] подробно рассмотрены ограничения набора NGSIM. Для участков US-101 и I-80 длительность наблюдений составляет 45 минут, а длина контролируемых зон равна 640 и 500 м соответственно. Для городских магистралей временной интервал составляет приблизительно 30 минут. Этого времени достаточно для анализа отдельных инцидентов, но оно не всегда отражает развитие пробок или волн ускорения/замедления на более длинных участках сети [173].

К существенным ограничениям NGSIM относятся потеря видимости ТС из-за ракурсов камеры и перекрытий, некорректные значения расстояния

между автомобилями и некоторые нереалистичные значения ускорения. Эти ошибки особенно заметны в задачах типа «ведущий-ведомый», где расстояние и ускорение являются частью базового описания реакции ведомого. Поэтому перед использованием таких данных необходимы фильтрация, реконструкция траектории и проверка физической правдоподобности наблюдений.

Тем не менее, этот набор данных остается ценным ресурсом с открытым исходным кодом для проверки как микроскопических, так и макроскопических моделей транспортных потоков. Он используется для сравнения результатов моделирования, реконструкции траекторий и оценки поведения модели в воспроизводимых условиях [173; 174].

Примером является работа [175], где фрагмент US-101 применяется для обучения модели следования за лидером на основе Random Forest. Авторы исключают координатные признаки Local X, Local Y, Global X и Global Y как избыточные для выбранной постановки и оставляют скорость, идентификатор полосы, данные о предшествующем автомобиле и дистанцию. Такой отбор признаков снижает размерность входа и делает модель ближе к задаче продольного взаимодействия.

В статье [176] исследуются методы сбора и анализа данных о динамике движения автомобилей в условиях вне полосы движения. Основное внимание уделяется разработке и калибровке системы сбора данных на основе изображений и данных, полученных с помощью GPS.

Данные для этого исследования были собраны в марте 2018 года на прямых участках трассы NH-27 в Гувахати, Индия. Эксперименты проводились с участием двух автомобилей, двигающихся в колонне на участке длиной 37.33 км в дневное время. Автомобили были оборудованы GPS-приемниками Racelogic Video V-box, записывающими положение каждого автомобиля с интервалом 0.1 с, и видеосистемой, прикрепленной к лобовому стеклу. Позиционные данные GPS использовались для вычисления скоростей автомобилей и продольных/поперечных ускорений, а продольные и поперечные расстояния между автомобилями оценивались из синхронизированной видеозаписи с помощью полуавтоматической системы извлечения траекторий.

Продольные расстояния представляют собой расстояние между ведущим и лидирующим транспортными средствами в продольном направлении. Термин "поперечные расстояния используется для описания поперечных расстояний между транспортными средствами вне полосы движения. Термин "centerline

separation (CS)"используется для описания поперечного расстояния между центральными линиями двух автомобилей, движущихся в колонне.

Ошибки в измерениях учитывались, и максимальная ошибка оценивалась как нормальная распределенная величина, увеличивающаяся с увеличением расстояния от точки калибровки. Точность измерений скоростей составляла 0.16 км/ч, учитывая, что исследователи не удалялись от точки калибровки дальше 8 метров. Данный метод сбора данных, адаптированн к изучению поведения водителей при движении за впереди идущим транспортным средством. Водители постепенно изменяли скорость и самостоятельно определяли безопасную дистанцию до впереди идущего транспортного средства, основываясь на собственном суждении и опыте вождения.

В статье [176] рассматривается сбор данных о движении транспортных средств в условиях движения вне полосы. Эксперимент проводился на прямых участках автомагистрали NH-27 в Гувахати. В нем участвовали два автомобиля, двигавшиеся колонной по участку длиной 37,33 км в дневное время. Автомобили были оборудованы GPS-приемниками Racelogic Video V-box, которые записывали местоположение с интервалом 0,1 секунды, и видеосистемой, установленной на лобовом стекле.

Данные GPS использовались для расчета скорости и ускорения, а синхронизированные видеозаписи - для оценки продольного и поперечного расстояния между ТС. Продольное расстояние описывает расстояние между ведущим и следующим автомобилем в направлении движения. Боковое расстояние характеризует смещение в сторону, что особенно важно при движении вне полосы движения, когда взаимодействие транспортных средств не ограничивается одномерным сценарием.

Авторы учитывали погрешности измерений и связывали максимальную ошибку с расстоянием до точки калибровки. Точность измерения скорости составляла 0,16 км/ч при ограничении удаления от точки калибровки. Водители самостоятельно регулировали скорость и дистанцию до впереди идущего автомобиля. Такая схема позволяет наблюдать за естественной реакцией следующего ТС, но в то же время ограничивает обобщаемость результатов.

Ограничения эксперимента обусловлены небольшим количеством ТС, использованием одного прямого участка дороги, проведением исследования в дневное время и отсутствием значительных погодных изменений. Кроме того, видеосистема чувствительна к углу установки камеры, а некоторые этапы обра-

ботки зависят от ручной разметки. Поэтому результаты данного исследования полезны для анализа продольных и поперечных расстояний, но требуют проверки на при различных условиях.

В публикациях [177; 178] предложен метод сбора данных о движении автомобилей в колонне с использованием бинокулярного стереозрения и GNSS RTK. Основная задача метода состоит в измерении дистанции между лидером и ведомым автомобилем, относительной скорости и траектории движения. В экспериментах применялись бинокулярные камеры с разрешением 1024×768 и частотой 3 кадра в секунду, а также GNSS RTK-система iRTK2, поддерживающая BDS, GPS, GLONASS и GALILEO.

Обработка изображений выполнялась с использованием OpenCV. Расстояние до объекта вычислялось по разности координат одного и того же объекта на левом и правом изображениях. Перед измерениями выполнялась калибровка камер, синхронизация изображений и выбор отслеживаемого объекта. Для сопровождения объекта применялся алгоритм CSRT. Такая схема позволяет перейти от видеокадров к численной оценке дистанции, пригодной для анализа движения в колонне. Комбинация бинокулярных камер и GNSS RTK повышает точность измерений по сравнению с одиночной видеосистемой. Для камер с фокусным расстоянием 25 мм в рассматриваемых работах получено значение MAPE 3,2%. Для очистки данных от выбросов применялась робастная локально взвешенная регрессия, а в близкой постановке также использовался расширенный фильтр Калмана [177; 178].

Вместе с тем у подхода остаются эксплуатационные ограничения. Камеры с фокусным расстоянием 25 мм имеют узкое поле зрения, что затрудняет отслеживание объектов на малых дистанциях. Плохая погода, туман, дождь и снег могут ухудшать качество изображения. Калибровка камер требует ручного выбора опорных точек, поэтому процедура остаётся трудоёмкой и может вносить дополнительные ошибки. Для задач настоящей работы этот подход важен как пример точного измерительного комплекса, но он менее удобен для массового сбора данных в обычном дорожном движении.

В работе [179] исследуется распознавание стиля вождения по траекториям ТС, извлечённым из видеозаписей. В качестве источника используется реконструированная часть NGSIM, из которой предварительно удалён шум. Для классификации применяются SVM, Random Forest, KNN и MLP. Признаки извлекаются с помощью дискретного преобразования Фурье, дискретного

вейвлет-преобразования и статистических характеристик траекторий. Наилучший результат в этой работе показал SVM при использовании признаков DWT. Такой результат показывает, что предварительная обработка траекторий и выбор информативных признаков существенно влияют на качество распознавания поведения водителя. Для микроскопического моделирования это важно, поскольку стиль управления может менять выбор дистанции, интенсивность ускорения и реакцию на торможение лидера.

Ограничение исследования связано с источником данных. Фрагмент I-80 из NGSIM относится к конкретному участку автомагистрали и был собран в ограниченный период времени. Поэтому результат нельзя напрямую переносить на другие страны, типы дорог, погодные условия и режимы движения. Кроме того, в работе используются показатели риска ITTC, THW и MMTC, но не рассматривается более широкая комбинация признаков безопасности. Для данного исследования этот результат полезен как подтверждение значимости поведенческих признаков, но не заменяет сбор собственных данных.

В публикации [180] анализируется метод движущихся автомобилей как способ сбора данных о дорожном движении. Исследование выполнено на шести участках городских многополосных автомагистралей. Сравнивались два подхода: стационарные измерения с помощью видеокамер и радаров, а также метод движущегося автомобиля. При стационарном подходе интенсивность движения фиксировалась видеокамерами, скорость измерялась радарами, а время в пути определялось по двухточечным видеозаписям. Во втором подходе испытуемый автомобиль выполнял многократные проезды по ходу и против направления потока. Наблюдатели фиксировали обгоняемые, встречные и обгоняющие автомобили, время движения и длину участка. Расчёты интенсивности и скорости выполнялись по классическим уравнениям MCM.

Сопоставление показало высокую корреляцию между MCM и стационарными измерениями по интенсивности движения, особенно во внепиковые периоды. Для скорости и времени в пути средние арифметические и гармонические значения также показали близость результатов. При этом средние и медианные оценки имели большую дисперсию, а качество метода снижалось при свободном потоке и при высокой интенсивности движения. Для задач следования за лидером MCM имеет ограниченную применимость. Метод хорошо подходит для получения агрегированных характеристик потока, но хуже описывает локальные переходные процессы. Между тем именно такие процессы

важны для микроскопического моделирования, поскольку реакция ведомого автомобиля проявляется в изменении дистанции, скорости и относительной скорости на коротких интервалах времени.

Сопоставление рассмотренных подходов показывает, что каждый источник микроскопических данных закрывает только часть задачи. Стационарные видеосистемы и наборы типа NGSIM дают плотные траекторные наблюдения, но ограничены зоной съёмки, качеством распознавания и условиями конкретного региона. GPS, GNSS RTK, видеорегистраторы и бинокулярное стереозрение повышают точность отдельных измерений, но требуют специализированного оборудования, калибровки и контролируемой организации эксперимента. Метод движущихся автомобилей удобен для оценки агрегированных характеристик, но не даёт достаточной детализации локальной реакции лидера и ведомого автомобиля.

Для данного исследования необходим иной метод сбора данных. Он должен позволять увеличивать число участников, регистрировать рутинные поездки, учитывать погодные условия и сезонность, а также создавать синхронизированные временные ряды для пар «лидер-ведомый». Эти требования оправдывают разработку запатентованного метода сбора микроскопических данных на основе мобильного клиент-серверного приложения. Такое приложение должно записывать координаты, скорость, направление движения, а также информацию о водителе, транспортном средстве и окружающей среде без необходимости установки специализированной измерительной системы.

Модели микроскопического поведения участников транспортного потока

Микроскопическое моделирование описывает движение отдельных участников потока и их взаимодействие с ближайшим окружением. В этой группе задач рассматриваются не только модели следования за лидером. Поведение водителя может задаваться через продольное движение, перестроение, выбор допустимого интервала, выбор скорости, реакцию на светофорное регулирование и выбор маршрута.

Для данной главы основное значение имеют модели продольного движения. Они описывают изменение скорости, ускорения и дистанции между автомобилями, движущимися в одном направлении. Наиболее важный частный случай — модель следования за лидером. В ней состояние ведомого автомобиля связывается с состоянием впереди идущего автомобиля, межмашинной дистанцией и относительной скоростью. Такая постановка нужна для оценки реакции водителя на торможение или ускорение лидера, а также для анализа устойчивости продольных возмущений.

Модели поперечного маневрирования решают другую задачу. Они описывают перестроение, обгон, выбор полосы и завершение манёвра. Для таких моделей требуется учитывать боковое положение автомобиля, доступность соседних полос, правила начала манёвра и конфликтные зоны. Модели выбора допустимого интервала применяются при въезде в поток, перестроении, поворотах и проезде нерегулируемых пересечений. Их объектом является решение о приемлемости временного или пространственного зазора, а не непрерывная динамика пары лидер-ведомый.

Модели выбора скорости задают желаемую или допустимую скорость с учётом ограничений, плотности потока, дорожной обстановки и индивидуального стиля водителя. Они важны в режиме свободного движения, когда впереди идущий автомобиль не определяет поведение водителя напрямую. Для задачи следования за лидером таких моделей недостаточно, поскольку они не задают явную связь между скоростью ведомого, скоростью лидера и дистанцией. Модели реакции на сигнальное регулирование описывают остановки, стартовые задержки и движение очередей, а модели выбора маршрута относятся уже к стратегическому уровню поведения.

Поэтому далее рассматриваются прежде всего модели следования за лидером и их расширения. Такая фокусировка связана с экспериментальной частью главы: собственный набор данных формируется как последовательность наблюдений пары лидер-ведомый и включает скорость лидера, скорость ведомого, дистанцию, погодные признаки, сведения о водителе и автомобиле. На рисунке 3.1 приведена классификация моделей прогнозирования микроскопических характеристик транспортного потока.

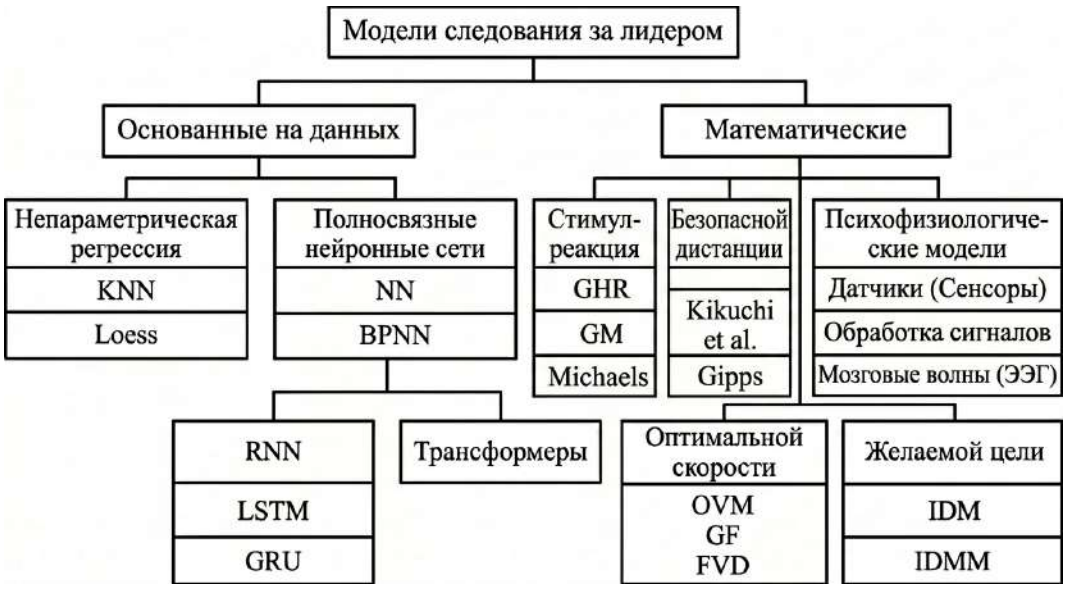

Рисунок 3.1 — Классификация моделей прогнозирования микроскопических характеристик

В работе [171] предложен гибридный подход к моделированию следования за лидером, объединяющий IDM и LSTM. IDM используется как физически интерпретируемая модель зависимости между скоростью, ускорением и дистанцией. LSTM отвечает за восстановление временной зависимости по траекторным данным. Смысл такой схемы состоит в том, чтобы не отказываться от классической модели, но дополнить её компонентой, способной учитывать историю движения.

Для формирования выборки авторы организовали собственный эксперимент. Расстояние между ведомым автомобилем и лидером измерялось лазерным дальномером. Скорость, ускорение и положение дроссельной заслонки фиксировались через OBD-CAN. GPS использовался для регистрации положения и дорожного контекста. Измерения синхронизировались и сохранялись на вычислительном модуле.

Эксперимент проводился на городских дорогах Нанкина при ограничении скорости 60 км/ч и наличии светофорного регулирования. Маршрут имел длину 7,3 км, а испытуемый автомобиль управлялся одним водителем. Поездки выполнялись при благоприятной погоде в интервале с 16 до 17 часов. Режим следования определялся через 5-секундный интервал: если расстояние до впереди идущего автомобиля было меньше пути, который автомобиль проходит за 5 секунд при текущей скорости, фрагмент относился к следованию за лиде-

ром. Всего было собрано 16 800 записей, из которых 15 000 использовались для обучения.

Анализ показал, что LSTM дает меньшую ошибку MSE, но требует больше вычислительного времени. Это связано с использованием исторических данных о движении, в то время как некоторые альтернативные модели работают исключительно с текущим состоянием. Для объединения IDM и LSTM авторы использовали адаптивный фильтр Калмана, который корректирует прогноз на основе новых наблюдений. Полученная модель LSTM-IDM показала более низкие значения MSE и MAE по сравнению с отдельными моделями IDM и LSTM.

Для настоящего исследования эта работа важна как пример гибридизации классической модели и нейросетевого блока. Однако её экспериментальная база ограничена одним водителем, одним маршрутом и благоприятными погодными условиями. Поэтому по таким данным трудно оценить влияние сезонности, влажного покрытия, ухудшенной видимости и различий в стиле управления. Кроме того, схема IDM-LSTM-фильтр Калмана повышает точность, но делает модель более сложной для воспроизведения и эксплуатации.

В публикации [181] рассматривается включение долгосрочных и краткосрочных характеристик водителя в модели следования за лидером. Исходным источником служит NGSIM. Перед анализом скорость и ускорение ТС фильтруются, поскольку исходные траектории содержат шум и отдельные физически неправдоподобные значения.

Долгосрочный стиль водителя описывается через среднюю скорость, ускорение, временной интервал до столкновения THW, замедление и частоту изменений ускорения или торможения. Для группировки применяются спектральная кластеризация и метод k-средних. В результате водители разделяются на группы с разным профилем поведения. Агрессивные водители имеют меньшие значения THW и более высокие ускорения и замедления. Спокойные водители, напротив, сохраняют больший временной интервал и менее резко меняют скорость.

Краткосрочное поведение анализируется после внешних стимулов, например после перестроения впереди идущего автомобиля. Водитель относится к стабильной или нестабильной группе в зависимости от того, сохраняются ли его долгосрочные характеристики после такого воздействия. Затем долгосрочные и краткосрочные признаки включаются в классические модели следования за лидером. В модифицированной IDM используются параметры, связанные с

желаемой скоростью и THW. В модифицированной модели Гиппса учитываются желаемые ускорение и замедление.

По результатам экспериментов LSTD-IDM снизила ошибку на 36,67% относительно исходной IDM, а LSTD-Gipps снизила ошибку на 23,78% относительно исходной модели Гиппса. Этот результат подтверждает, что поведенческая неоднородность водителей влияет на точность микроскопического моделирования. Для настоящей работы вывод состоит в том, что признаки водителя целесообразно использовать как контекст модели. При этом зависимость от NGSIM сохраняет прежние ограничения, связанные с регионом, временем наблюдения и качеством исходных траекторий.

В работе [182] исследуется калибровка репрезентативных параметров моделей следования за лидером по реальным траекториям. Авторы исходят из того, что параметры модели не являются независимыми. Если оценивать их по отдельности, часть поведенческой структуры теряется. Для проверки рассматриваются модель Ньюэлла, модель Гиппса, IDM и Wiedemann 1999.

Источником данных является исследование SHRP2 Naturalistic Driving Study. Этот набор данных включает информацию о большом количестве поездок на автомобиле, совершенных участниками в течение длительного периода времени. Транспортные средства были оснащены регистраторами данных по шине CAN, внешними датчиками и видеокамерами. Кроме того, были собраны данные анкет водителей. В отличие от коротких дорожных тестов, этот источник данных лучше отражает поведение в нормальных условиях вождения.

Авторы сравнили восемь процедур калибровки. Некоторые из методов сохраняли корреляции между параметрами, включая подходы максимального правдоподобия и байесовские процедуры. Другие предполагали независимость параметров, что упрощает вычисления, но может снизить реалистичность модели. Оценка проводилась с использованием 10-кратной перекрестной проверки. Наилучшие результаты были получены с использованием методов, сохраняющих корреляции между параметрами.

Для настоящей работы этот результат важен с методической точки зрения. Он показывает, что параметры модели следования за лидером следует рассматривать как связанную систему, а не как набор независимых коэффициентов. При этом сложные процедуры статистического вывода могут быть трудны для применения в реальном времени. Поэтому адаптация параметров

должна сохранять взаимосвязь коэффициентов, но оставаться вычислительно простой.

В публикации [183] модель IDM рассматривается с учётом V2V- коммуникации. Прямая связь между ТС позволяет передавать сведения о скорости, направлении движения, состоянии дороги и предупреждениях без промежуточной инфраструктуры [184]. Для подключённых и автономных автомобилей это важно, поскольку водитель или автоматизированный модуль получает информацию о действиях лидера раньше, чем она проявится только через изменение дистанции [185].

Задача исследования состояла в том, чтобы оценить, как предупреждающие V2V-сообщения меняют поведение ведомого автомобиля, и модифицировать IDM с учётом этих сообщений. Калибровка выполнялась по собранным данным через минимизацию среднеквадратичной ошибки между наблюдаемыми и смоделированными скоростями. Для сравнения использовались исходная IDM и модифицированная модель MIDM.

Эксперимент проводился с двумя автомобилями, оснащёнными высокоточным GPS и MK5 V2V-устройствами. GPS фиксировал положение, скорость и ускорение с частотой 20 Гц. V2V-устройства передавали предупреждающие сообщения между автомобилями. Рассматривались два сценария: торможение и остановка ведомого автомобиля после торможения лидера, а также последовательность торможений и ускорений.

Для анализа изменения поведения применялся серый корреляционный анализ. Основным поведенческим фактором было выбрано ускорение ведомого автомобиля. Без предупреждающих сообщений наиболее близким фактором оказалась пространственная дистанция. После получения V2V-сообщений более значимой стала разность скоростей. Это указывает на изменение информационной основы реакции водителя. Достоинство работы состоит в экспериментальном включении V2V-сигнала в модель следования за лидером. Ограничение связано с контролируемой постановкой и малым числом автомобилей. Полученные данные хорошо подходят для проверки гипотезы о влиянии предупреждающих сообщений, но недостаточны для обобщения на широкий набор дорожных сценариев.

В статье [161] предложен бенчмарк FollowNet для сопоставимой оценки моделей следования за лидером. Его основная задача состоит в том, чтобы привести разнородные траекторные наборы к единому формату событий

следования. Для этого вводятся правила отбора фрагментов, в которых на протяжении всего события сохраняется один и тот же лидер, длительность события составляет не менее 15 секунд, а поперечное расстояние между ТС не превышает 2 м. Для отдельных наборов применяются дополнительные фильтры. Например, в HighD исключаются события со средней скоростью ниже 2 м/с, а также фрагменты, где скорость остаётся ниже 0,2 м/с более 5 секунд.

Такой бенчмарк полезен тем, что объединяет события из нескольких дорожных сред и снижает зависимость оценки от одного источника данных. Авторы анализируют шесть ключевых параметров по пяти наборам данных и показывают, что каждый источник имеет собственную структуру, связанную с местом сбора, типом дороги и сценарием движения.

В экспериментальной части сравниваются две классические модели и три модели, основанные на данных. Классические модели представлены IDM и GHR. IDM используется как распространённая интерпретируемая модель продольной динамики, а GHR — как более простая модель типа стимул–реакция. Среди моделей, основанных на данных, рассматриваются полносвязная нейронная сеть, LSTM и DDPG.

Полносвязная сеть реализована как трёхслойная модель с гиперболическим тангенсом, оптимизатором Adam и функцией потерь MSE. Для LSTM подбираются размер скрытого слоя, число рекуррентных слоёв и dropout. В DDPG используется модифицированная функция вознаграждения со штрафом за столкновение.

Качество оценивается по двум показателям: среднеквадратичной ошибке расстояния и уровню столкновений. Такое сочетание метрик принципиально для моделей следования за лидером. Низкая ошибка дистанции не гарантирует безопасного поведения при последовательном применении. Классические модели обычно дают более высокую ошибку по расстоянию, но при корректной калибровке сохраняют нулевой уровень столкновений. Модели, основанные на данных, лучше аппроксимируют дистанцию, но не всегда сохраняют безопасную траекторию. DDPG показывает лучший компромисс в своём классе, хотя на HighD также фиксируются случаи столкновений.

Для настоящей работы FollowNet важен как пример протокола, где точность и безопасность оцениваются совместно. Он также показывает, что объединение нескольких известных наборов данных не устраняет полностью проблему переносимости. В реальных условиях могут возникать агрессивные

реакции водителей, экстренные торможения, редкие погодные ситуации и региональные особенности движения, которые слабо представлены в открытых корпусах.

Более широкая таксономия моделей следования за лидером дана в [160]. Авторы рассматривают такие модели в междисциплинарном контексте: транспортное моделирование, автономное движение, адаптивный круиз-контроль и кооперативный адаптивный круиз-контроль. В обзоре выделяются три группы моделей.

- Модели, основанные на теории, включая Gipps, IDM, OVM и другие параметрические модели.
- Модели, основанные на данных, включая нейронные сети, методы машинного обучения и обучение с подкреплением.
- Модели, основанные на знаниях, включая большие языковые, зрительно-языковые и мультимодальные модели.

Выделение моделей, основанных на знаниях, отражает современный переход к использованию более широкого контекста дорожной сцены. Такие модели могут обрабатывать текстовые описания, изображения, LiDAR-данные, видеопотоки и другие сенсорные признаки. Их потенциальное преимущество состоит в учёте информации, которая не сводится к скорости, ускорению и дистанции. При этом для задач следования за лидером данный класс моделей пока не заменяет динамические модели. Остаются вопросы проверяемости, вычислительной стоимости, интерпретируемости и гарантии безопасного поведения.

Связь между классическими моделями и моделями, основанными на данных, рассматривается в [171]. В работе предложен физически информированный подход PIDL-CF. Его идея состоит в том, чтобы использовать классическую модель как источник предметного ограничения, а нейронную сеть — как средство аппроксимации нелинейных зависимостей, которые трудно описать простой параметрической формой.

PIDL-CF включает две части: физически неинформированную нейронную сеть и вычислительный граф, построенный на основе физической модели. Функция потерь задаётся как взвешенная сумма двух среднеквадратичных ошибок. Первая ошибка измеряет расхождение между наблюдаемым и предсказанным ускорением. Вторая ограничивает расхождение между ускорением, предсказанным нейронной сетью, и ускорением, полученным из физической модели.

Такое построение снижает риск переобучения и повышает устойчивость к шуму и пропускам данных. Для настоящего исследования этот результат важен тем, что физические ограничения могут использоваться не только после прогноза, но и непосредственно при обучении модели. В дальнейшем эта идея используется в более прикладной форме: классическая модель задаёт ограничение безопасной дистанции, а модель, основанная на данных, формирует прогноз динамики.

Классические модели следования за лидером продолжают использоваться и в задачах смешанного движения. В работе [186] анализируется устойчивость транспортного потока в условиях, характерных для развивающихся стран. Рассматриваются слабая дисциплина движения по полосам, неоднородность типов ТС и частые обгоны. Для сравнения используются OVM, FVDM, IDM и IDMM. В набор данных входят мотоциклы, легковые автомобили, автобусы и грузовые автомобили. Калибровка параметров выполняется генетическим алгоритмом в MATLAB.

Авторы получили условия устойчивости для каждой модели на основе откалиброванных параметров и оценили их для различных типов ТС. Кроме того, они предлагают простое расширение модели IDM на двумерное пространство, что позволяет учитывать более сложные сценарии взаимодействия.

Ограничение такого подхода связано с необходимостью отдельной калибровки для разных типов ТС и условий движения. Это повышает трудоёмкость применения модели и затрудняет перенос параметров. Для настоящей работы данный результат показывает, что статическая параметризация классической модели недостаточна при выраженной неоднородности потока. Поэтому далее требуется механизм адаптации параметров к контексту движения.

Публикация [175] посвящена модели следования за лидером на основе случайного леса. В качестве исходного источника используется NGSIM US-101. Перед обучением авторы выполняют отбор признаков и исключают поля, которые дублируют уже имеющуюся информацию или не связаны напрямую с продольным взаимодействием. В итоговый набор входят скорость, идентификатор полосы, признаки лидирующего ТС и дистанция.

Random Forest сравнивается с искусственной нейронной сетью, моделью GM и моделью полного различия скоростей FVD. Качество оценивается по ME, MAE, MARE и RMSE. По результатам экспериментов случайный лес показывает более высокую точность относительно выбранных базовых моделей.

Преимущество метода связано с ансамблевым построением, устойчивостью к шуму и сравнительно простой настройкой.

Для настоящего исследования этот результат полезен как пример модели, которая хорошо работает с табличными признаками и не требует сложной временной архитектуры. Однако Random Forest не задаёт явной физической структуры продольного взаимодействия. Без дополнительных ограничений такая модель не гарантирует безопасное поведение при последовательном развёртывании. Поэтому её можно рассматривать как сильную базовую модель по точности, но не как самостоятельный механизм контроля безопасной дистанции.

В статье [170] предложена модель следования за лидером, объединяющая CNN и LSTM. Цель работы состоит не только в повышении точности прогноза, но и в проверке переносимости модели между данными разной природы. Для обучения и тестирования используются база естественного вождения и данные экспериментов с коммерческими системами адаптивного круиз-контроля.

Авторы отдельно отмечают, что значительная часть исследований в этой области опирается на NGSIM. Такая концентрация на одном источнике создаёт риск методической зависимости от конкретных дорог, региона, способа съёмки и качества траекторий. Поэтому проверка на нескольких наборах данных является необходимой частью оценки модели следования за лидером.

Архитектура CNN-LSTM использует свёрточные слои для извлечения локальных признаков и LSTM для описания временной зависимости. В свёрточных слоях применяется ReLU, а функция потерь задаётся как MSE. Модель сравнивается с IDM и LSTM и показывает более низкие значения MSE. Это подтверждает, что предварительное извлечение локальных паттернов перед рекуррентной обработкой может улучшать прогноз. При этом архитектура остаётся зависимой от качества обучающей выборки и требует проверки на сценариях, отличающихся от исходных данных.

Похожие архитектуры рассматриваются в работе [187], где сравниваются традиционные и нейросетевые модели для регулируемого перекрёстка. Среди математических моделей анализируются DM, Gipps и FVD. Среди моделей, основанных на данных, используются CNN-LSTM и Conv-LSTM. Различие между ними состоит в способе совмещения пространственной и временной обработки. В CNN-LSTM признаки сначала извлекаются свёрточными слоями, а затем передаются в LSTM. В Conv-LSTM свёрточные операции встроены непосредственно в рекуррентные ячейки.

Для экспериментов используются наборы Lankershim и Peach Tree. Исходные данные предварительно фильтруются. В выборку включаются только легковые автомобили, исключаются смена полосы, ДТП и режим свободного движения, а длительность события следования задаётся равной 15 секундам. После фильтрации применяется скользящее среднее для сглаживания скорости и ускорения. В работе рассматриваются два режима движения. Первый соответствует непрерывному движению через перекрёсток. Второй описывает прерывистое движение, когда водитель вынужден останавливаться из-за сигнального регулирования. Качество оценивается по RMSE для ускорения, скорости и траектории. CNN-LSTM и Conv-LSTM показывают высокую точность в обоих режимах.

Ограничение исследования связано с тем, что вычислительные затраты моделей обсуждаются недостаточно подробно. Для эксплуатационной ИТС это существенно. Модель должна не только точно воспроизводить траекторию, но и укладываться в допустимую задержку обработки, особенно если она используется в контуре прогнозирования или управления.

Среди недостатков стоит отметить отсутствие обсуждения вычислительных затрат моделей CNN-LSTM и Conv-LSTM, хотя это важно для оценки применимости в реальных системах управления движением.

Ограничения моделей, основанных на данных, подробно рассматриваются в [188]. Авторы выделяют слабую обобщающую способность при обучении на ограниченных выборках, зависимость от качества исходных траекторий и недостаточный учёт факторов безопасности. В качестве решения предложена модель Attention-Based Fused Car-Following (ABF-CF), объединяющая реальные и смоделированные данные.

ABF-CF использует архитектуру seq2seq с механизмом внимания. Будущие ускорения ТС прогнозируются на основе GRU с вниманием. Для расширения обучающей выборки авторы объединяют траектории NGSIM с синтетическими данными, полученными с помощью IDM. Такая схема позволяет частично компенсировать ограниченность реальных наблюдений и добавить физически осмысленные сценарии движения.

Для повышения безопасности в качестве входных переменных включены TTC и DRAC. Первый показатель характеризует время до столкновения, а второй представляет собой замедление, необходимое для предотвращения аварии. Их использование делает модель более чувствительной к опасным

условиям близости. Однако такой подход требует дополнительной проверки в реальном времени в более широком диапазоне дорожных сценариев. Для диссертационного исследования этот результат важен, поскольку он подтверждает необходимость сочетания прогностической модели с явными показателями безопасности.

Проведенный анализ показывает, что модели следования за лидером остаются основным инструментом для микроскопического прогнозирования продольной динамики транспортного потока. Классические модели, включая IDM, Gipps, GHR и их модификации, сохраняют физическую интерпретируемость и позволяют явно задавать ограничения безопасности. Их слабость заключается в калибровке параметров и переносе между дорожными, погодными и поведенческими условиями.

Модели, основанные на данных, лучше выявляют нелинейные зависимости и индивидуальные характеристики реакции водителя. К ним относятся Random Forest, LSTM, CNN-LSTM, Conv-LSTM, модели GRU с механизмом внимания и другие архитектуры нейронных сетей. Однако высокая точность на тестовом наборе не гарантирует безопасного поведения при непрерывной работе. Это особенно важно для задачи «лидер-последователь», где ошибки прогнозирования могут накапливаться и приводить к физически небезопасному расстоянию.

Гибридные подходы наиболее актуальны для задачи, рассматриваемой в этой главе. В этих подходах классическая модель определяет структуру продольного взаимодействия или ограничение безопасности, в то время как обучаемый компонент уточняет прогноз на основе данных. Этот принцип используется в LSTM-IDM, PIDL-CF, ABF-CF и других схемах. Для дальнейших исследований этот вывод дает основную логику: необходимо не только обучить модель прогнозирования, но и связать ее с проверенным ограничением безопасного расстояния.

Отдельный вывод относится к данным. Открытые наборы NGSIM, HighD, pNEUMA и другие источники полезны для воспроизводимого сравнения моделей, но они не полностью отражают целевые условия эксплуатации. В них ограниченно представлены сезонность, снег, гололёд, низкая видимость, локальные особенности дорожной инфраструктуры и региональные различия в поведении водителей. Поэтому дальнейшая часть главы посвящена разработке собственного метода сбора микроскопических данных, пригодных для обуче-

ния и проверки моделей следования за лидером в более близких к целевой среде условиях.

3.2 Разработка метода сбора микроскопических данных транспортного потока

Для моделирования следования за лидером необходимы данные, собранные не только в одном типовом режиме движения, но и при изменении внешних условий, состава участников и характеристик автомобилей. [17; 23; 42; 57] В противном случае модель хорошо воспроизводит отдельный сценарий, но хуже переносится на другие дороги, погоду и стили управления. [58]

Открытые наборы NGSIM, highD, Waymo Open Dataset и Woven Planet удобны для воспроизводимого сравнения моделей, однако они не закрывают задачу натурной проверки в целевой среде. Такие данные привязаны к конкретной стране, типу дороги, способу съёмки и периоду наблюдения. В них ограниченно представлены зимние условия, локальная инфраструктура, российская организация движения и поведенческая неоднородность водителей.

Поэтому в работе дополнительно выполнен собственный сбор микроскопических данных. Для этой цели разработано мобильное клиент-серверное приложение, через которое параметры движения регистрируются непосредственно со смартфонов водителей. [11; 40; 59] Такой вариант выбран не как замена высокоточному измерительному комплексу, а как практичный способ увеличить число поездок, участников, маршрутов и погодных ситуаций без установки специального оборудования в каждый автомобиль.

Смартфон в данной постановке используется как доступный регистратор движения и канал передачи данных. GPS-модуль фиксирует координаты, скорость и направление движения. Эти величины необходимы для восстановления маршрута, синхронизации наблюдений и расчёта расстояния между ТС. MEMS-сенсоры, включая акселерометр и гироскоп, могут дополнительно фиксировать резкие ускорения, торможения, повороты и перестроения. В базовой версии метода основная роль отводится GPS-координатам, скорости и направлению движения, а инерциальные данные рассматриваются как возможное расширение признакового состава.

Практическое преимущество такой схемы состоит в низкой организационной стоимости эксперимента. Водителю не требуется подключаться к CAN-шине, устанавливать LiDAR, радар или отдельный GPS-регистратор. Достаточно установить приложение, выбрать автомобиль и разрешить передачу геолокационных данных. Записи передаются на сервер через мобильный интернет, где выполняются первичная синхронизация, сохранение измерений и дальнейшая обработка. Такой способ сбора данных согласуется с практикой применения смартфонов в задачах транспортной телематики [158].

Координаты автомобиля фиксируются по GPS и используются для восстановления траектории поездки. По ним проверяется, находятся ли два автомобиля на одном участке дороги и движутся ли они в согласованном направлении. На серверной стороне координаты также применяются для вычисления расстояния между автомобилями по формуле Хаверсина. Это расстояние не имеет точности LiDAR или радарного измерения, но после синхронизации и фильтрации пригодно для анализа продольной динамики в рассматриваемой постановке.

Скорость используется как основная динамическая переменная и как целевая величина в задаче прогнозирования. По изменению скорости можно оценивать реакцию ведомого автомобиля на ускорение или торможение лидера. В отличие от ускорения, скорость, получаемая по GPS или API смартфона, обычно менее чувствительна к единичным шумовым выбросам. Поэтому ускорение в данной работе рассматривается как производная характеристика, требующая дополнительной фильтрации.

Межмашинная дистанция является ключевым признаком для модели следования за лидером. Она определяет, находится ли ведомый автомобиль в режиме свободного движения или в режиме продольного взаимодействия с лидером. В профессиональных экспериментах такая дистанция измеряется LiDAR, радаром или бинокулярной видеосистемой. В разработанном методе расстояние оценивается сервером по геолокационным данным пары автомобилей. Такой выбор снижает точность отдельного измерения, но позволяет собирать данные в большем числе обычных поездок.

Информация о водителе используется для учёта поведенческой неоднородности. Возраст, пол и стаж вождения не объясняют поведение сами по себе, но могут быть связаны с выбором скорости, дистанции и интенсивности реакции

на изменение обстановки. [189] В модели эти величины рассматриваются как контекстные признаки, дополняющие кинематические ряды.

Сведения об автомобиле нужны для приближённого учёта технической неоднородности. Модель и год выпуска могут косвенно отражать различия в массе, тормозной системе, динамике разгона и общем эксплуатационном состоянии. Эти признаки не заменяют подробные технические параметры, но позволяют отделить часть различий, связанных не с водителем, а с ТС. Погодные данные фиксируют внешний контекст поездки. Осадки, температура, туман и потенциальное обледенение влияют на сцепление с покрытием, видимость и субъективную оценку безопасной дистанции. Поэтому погодные признаки используются при последующей проверке переносимости модели между режимами движения и при динамической коррекции коэффициентов классической модели.

MEMS-датчики смартфона могут дополнять GPS-наблюдения. Акселерометр и гироскоп полезны для фиксации локальных событий, которые плохо видны по секундным GPS-координатам: резкого торможения, ускорения, поворота или смены полосы. В базовой версии датасета акцент сделан на координатах, скорости, направлении движения и серверном расчёте дистанции, однако архитектура приложения допускает расширение за счёт инерциальных измерений.

В последующих экспериментах скорость ведомого автомобиля используется как основная целевая переменная. Такой выбор связан с особенностями смартфонной схемы сбора данных. Скорость регистрируется устойчивее, чем ускорение, а ускорение обычно получается численным дифференцированием и сильнее зависит от шума. При этом скорость сохраняет прямую связь с задачей следования за лидером, поскольку изменение скорости ведомого автомобиля отражает его реакцию на движение лидера и текущую дистанцию. С точки зрения сбора данных, скорость можно измерить с хорошей точностью посредством датчиков GPS, API смартфонов и телеметрии автомобиля. В отличие от ускорения, измерения скорости обычно менее шумные и более стабильные. Более того, как подчеркивается в [158], скорость, определяемая по GPS, достаточно точна для использования в транспортном моделировании, особенно в сочетании с методами фильтрации, такими как сглаживание Калмана.

Оценка безопасного расстояния между транспортными средствами

При рассмотрении продольной динамики в микроскопическом транспортном потоке ключевым является формализуемое условие *динамического габарита* (недопущения наложения траекторий по координате движения), которое связывает взаимное расположение ТС с их скоростями и допустимыми режимами ускорения/торможения. В настоящей работе в качестве теоретической основы для последующей калибровки и сопоставления с моделями, основанными на данных, принимаются классические постановки, развиваемые в рамках моделей SSSR-Micro: модель *«следования за лидером»* и обратная ей модель *«заднего привода»*.

Рассматривается движение ТС в одном направлении друг за другом

$$x_0(t) < x_1(t) < ... < x_{n+1}(t), \forall\ t > 0,$$

где $x_n(t)$ - траектория движения ТС с номером n по прямой.

Основным понятием является *динамический габарит*. Динамический габарит рассматривается как расстояние безопасности в зависимости от скорости с коэффициентами. Предполагается однотипный динамический габарит, т.е. не рассматривается разный характер управления.

Система дифференциальных неравенств эквивалентна условиям безопасного движения

$$x_{n+1}(t) - x_n(t) \geqslant C_0 + C_1\dot{x}_n(t) + C_2\dot{x}_n^2(t). \tag{3.1}$$

Предполагается, что скорость $\dot{x}_n$ - абсолютно непрерывна и ограничена

$$|\dot{x}_n(t)| \leqslant M_1, \forall\ n = 0,...,N,\ \forall\ t > 0, \tag{3.2}$$

и, учитывая однонаправленность движения (в положительном направлении),

$$0 \leqslant \dot{x}_n(t) \leqslant M_1, \forall\ n = 0,...,N,\ \forall\ t > 0, \tag{3.3}$$

ускорение $\ddot{x}_n$ существует почти всюду и ограничена

$$M_2 \leqslant \ddot{x}_n(t) \leqslant M_3, \forall\ n = 0,...,N,\ \forall\ t > 0 \tag{3.4}$$

где M_2 и M_3, вообще говоря, несимметричны и связаны с торможением (сцепление) и ускорением.

Модель "Следования за лидером"

Имеется цепочка ТС x_1,...,x_{N+1}. $\forall\ n = 0,...,N$ справедливо

$$\begin{cases} x_{n+1}(t) - x_n(t) = C_0 + C_1\dot{x}_n(t) + C_2\dot{x}_n^2(t) \\ 0 \leqslant \dot{x}_n(t) \leqslant M_1, \forall\ n = 0,...,N,\ \forall\ t > 0 \\ \left|\ddot{x}_n(t)\right| \leqslant M_2, \forall\ n = 0,...,N,\ \forall\ t > 0 \end{cases} \tag{3.5}$$

начальные условия

$$x_1(0) = x_1,...,x_n(0) = x_n. \tag{3.6}$$

Граничное условие (3.7) - модель "Следования за лидером"

$$x_{N+1}(t) = f(t), \tag{3.7}$$

где задаётся функция $f(t)$ движения x_{N+1} ТС (лидера). Необходимо восстановить функцию движения последующих ТС. Модель (3.5)-(3.7) называется моделью "Следования за лидером".

Модель "Заднего привода"

$$\begin{cases} x_{n+1}(t) - x_n(t) = C_0 + C_1\dot{x}_n(t) + C_2\dot{x}_n^2(t) \\ 0 \leqslant \dot{x}_n \leqslant M_1, \forall\ n = 0,...,N,\ \forall\ t > 0 \\ \left|\ddot{x}_n\right| \leqslant M_2, \forall\ n = 0,...,N,\ \forall\ t > 0 \end{cases} \tag{3.8}$$

начальные условия

$$x_1(0) = x_1,...,x_n(0) = x_N. \tag{3.9}$$

Граничное условие (3.10) - модель "Заднего привода"

$$x_1(t) = f(t) \tag{3.10}$$

где задаётся функция f(t) движения последнего ТС x_1. Модель (3.8)-(3.10) называется моделью "Заднего привода".

В рамках настоящего исследования данные постановки используются как базовые классические модели. С одной стороны, они обеспечивают строгую интерпретацию параметров безопасности через динамический габарит, а с другой - задают удобную основу для последующей интеграции с методами, основанными на данных, и для анализа эффектов управления в замкнутом контуре. [56]

Синтез моделей «следования за лидером» и «заднего привода» естественным образом приводит к возможности *двунаправленного управления* движением колонны ТС. Модель «следования за лидером» реализует управление с начала потока через задание траектории ведущего ТС, тогда как модель «заднего привода» задаёт управление с конца через фиксацию движения замыкающего ТС. Совместное применение этих постановок позволяет формировать согласованные управляющие воздействия на обоих концах цепочки, обеспечивая подавление продольных возмущений и стабилизацию межмашинных интервалов при выполнении условий динамического габарита и ограничений на скорость и ускорение. В прикладном плане такая двусторонняя схема создаёт теоретическую основу для построения кооперативных стратегий управления (в подключённых и автоматизированных колоннах), в которых корректирующие воздействия могут генерироваться как на основании информации о поведении лидера, так и на основании состояния замыкающего ТС, повышая устойчивость и безопасность движения.

Для организации адекватного моделирования посредством модели следования за лидером необходимо работать с данными, которые отражают сложность и изменчивость реальной дорожной обстановки. Данные часто собираются в контролируемых условиях или ограничиваются определенными географическими регионами и типами трафика. Данные, используемые в этой работе, были собраны непосредственно в реальных дорожных сценариях с помощью специально разработанного клиент-серверного приложения.

Описание экспериментов по сбору данных

Натурные эксперименты по сбору микроскопических данных транспортного потока проводились в течение трёх лет и были ориентированы на получение репрезентативной выборки режимов продольного взаимодействия ТС (включая

устойчивое следование, переходные процессы разгона/торможения и режимы с выраженной нестационарностью потока). В отличие от постановок, ограниченных малым числом экипажей или единичным маршрутом, в настоящем исследовании экспериментальная программа была масштабирована как по числу участников, так и по пространственно-временной вариативности условий движения.

Состав участников и парк транспортных средств. В эксперименте участвовали 37 человек (водителей с разным стажем и полом), сбор данных осуществлялся с использованием 21 автомобиля. Привлечение расширенного пула водителей и ТС было необходимо для обеспечения поведенческой и технической неоднородности, характерной для реального транспортного потока. В частности, при одинаковой дорожной обстановке различия в предпочтениях по дистанции, интенсивности торможения и характеру набора скорости формируют существенную вариативность траекторных данных. Различия в динамических свойствах ТС (в том числе обусловленные конструктивными особенностями и годом выпуска) отражаются в наблюдаемой продольной динамике.

Сбор данных выполнялся на совокупности маршрутов, отражающих принципиально различные типы дорожной инфраструктуры и режимы движения. В рамках экспериментальной программы были выделены следующие классы участков:

1. **Магистральный сценарий (М-11).** Участки автомагистрали М-11 использовались для фиксации режимов движения с повышенными скоростями и длительными интервалами квазистационарного следования, а также для наблюдения редких, но значимых переходных процессов, возникающих при изменении плотности потока.
2. **Коридорные маршруты.** Под коридорами в работе понимаются протяжённые направления движения, включающие последовательность взаимосвязанных участков и обеспечивающие устойчивое накопление данных в условиях переменной нагрузки.
3. **Маршруты в пределах одной городской дороги.** Данный сценарий отражает внутригородское движение с влиянием локальных факторов (светофорное регулирование, неоднородность потока, ло-

кальные сужения, остановки/старты), что приводит к высокой доле разгонно-тормозных режимов и выраженной вариативности дистанций.

4. **Сценарий высокой интенсивности (Третье транспортное кольцо).** Участки ТТК использовались для получения данных при повышенной плотности и частых возмущениях потока, включая режимы с регулярными коррекциями скорости, а также с характерными кратковременными сокращениями интервалов.
5. **Квартальные маршруты.** Маршруты внутри квартала обеспечивали низкоскоростные режимы, частые изменения условий движения и существенную долю переходных процессов, обусловленных локальными конфликтами и манёврами.

Временное покрытие, сезоны и погодные классы. Эксперименты проводились в разные периоды года, что позволило включить в данные сезонную и метеорологическую нестационарность. В явном виде в рамках экспериментальной программы были представлены следующие сезоны:

- **весна**, включая периоды с переменной температурой и эпизодическими осадками;
- **лето**, характеризующееся широким диапазоном интенсивностей движения и устойчивыми условиями сцепления на сухом покрытии;
- **осень**, включая периоды с устойчивыми дождями и ухудшением видимости;
- **зима**, включая условия низких температур, снегопада и потенциального обледенения.

С учётом сезонности фиксировались и погодные классы, представленные в данных: ясная/сухая погода, дождь, снег, туман, пониженная температура/-гололёд.

Отнесение конкретной поездки к тому или иному классу осуществлялось по совокупности метеорологических признаков, сопоставленных с временными метками поездки (температура воздуха, наличие и интенсивность осадков, а при необходимости - дополнительные индикаторы ухудшения видимости).

Сбор осуществлялся в рамках реального дорожного движения. Перед каждой сессией участникам предоставлялись единые инструкции. Решения о скорости и дистанции до впереди идущего ТС должны приниматься исходя

из текущей дорожной обстановки и собственного привычного стиля вождения. Фиксированные профили скорости не навязывались. Запуск и завершение каждой поездки выполнялись пользователем в мобильном клиенте, после чего сессии присваивался уникальный идентификатор (`trip id`) для последующей сегментации и воспроизводимой обработки данных.

Суммарная дистанция, пройденная в рамках всех экспериментальных сессий за три года, превысила 1800 км. Такой объём наблюдений в сочетании с многообразием дорожных сценариев и внешних условий обеспечивает необходимую вариативность временных рядов для последующей калибровки и обучения моделей следования за лидером, включая анализ устойчивости и переносимости результатов между различными режимами движения.

Каждой поездке присваивался уникальный идентификатор, и данные маркировались соответствующим образом, чтобы обеспечить сегментацию и последующую обработку. Поскольку мобильное приложение может быть установлено на большинстве устройств под управлением операционной системы Android с минимальными настройками и не требует специализированного оборудования, дополнительные данные могут быть без труда собраны во время будущих поездок в различных погодных условиях. Водители могут использовать приложение во время обычных поездок, чтобы расширить набор данных. Со временем такое итеративное расширение позволит модели обучаться на основе более широкого спектра данных, и стать более надежной и адаптируемой к изменчивости реальных условий дорожного движения.

Для дополнительной характеристики сформированного корпуса данных были оценены не только суммарная протяжённость поездок, но и состав участников, разнородность маршрутов, сезонно-погодное покрытие и пригодность временных рядов для обучения моделей, основанных на данных. Обобщённые характеристики собранного набора данных приведены в таблице 16.

Таблица 16 — Обобщённые характеристики собственного набора микроскопических данных транспортного потока

Характеристика	**Значение / описание**
Период сбора данных	Три года натурных экспериментов в условиях реального дорожного движения.
Число водителей	37 участников с различным стажем вождения, полом и индивидуальными предпочтениями выбора скорости и дистанции.
Число автомобилей	21 автомобиль, что обеспечивает техническую неоднородность наблюдений, связанную с различиями в динамических свойствах ТС.
Суммарная протяжённость поездок	Более 1800 км по совокупности экспериментальных сессий.
Типы дорожных сценариев	Автомагистральный сценарий, коридорные маршруты, маршруты в пределах городской дороги, участки высокой интенсивности движения, квартальные маршруты.
Примеры маршрутов	Участки автомагистрали М-11, Третье транспортное кольцо, городские и внутриквартальные маршруты.
Сезонное покрытие	Весна, лето, осень, зима.
Погодные классы	Ясная/сухая погода, дождь, снег, туман, пониженная температура и условия потенциального гололёда.
Частота дискретизации	1 с, что позволяет формировать последовательные временные окна для обучения моделей прогнозирования.
Основные динамические признаки	Скорость ведомого ТС, скорость лидера, межмашинная дистанция, направление движения, координаты, временная метка.
Контекстные признаки	Погодные параметры, сезон, характеристики водителя, характеристики автомобиля.
Тип формируемых наблюдений	Синхронизированные парные наблюдения лидер–ведомый, пригодные для задач прогнозирования скорости и дистанции.
Общее число исходных записей	32 632
Число записей после фильтрации и синхронизации	23 175

Сопоставление сформированного корпуса данных с распространёнными наборами NGSIM, highD и pNEUMA показывает, что собственный набор данных

не заменяет указанные эталонные источники, но является более релевантным для задач настоящего исследования. Международные наборы данных обладают высокой ценностью для воспроизводимого сравнения моделей, однако они отражают дорожную инфраструктуру, культуру вождения, климатические условия и правила движения тех регионов, в которых были собраны. В частности, NGSIM характеризуется ограниченным пространственно-временным охватом и относится к дорожным условиям США. highD собран на немецких автомагистралях и ориентирован преимущественно на высокоскоростные сценарии. pNEUMA отражает специфику городского движения в Афинах.

В отличие от указанных наборов, данные, собранные в настоящей работе, характеризуют реальные условия эксплуатации, соответствующие рассматриваемой транспортной среде. Движение по российским городским дорогам и магистралям, наличие сезонной изменчивости, зимних условий, дождя, снега, тумана, локальных городских заторов, участков с высокой интенсивностью и низкоскоростных квартальных маршрутов. Поэтому данный корпус данных является предпочтительным для калибровки и обучения моделей, ориентированных на применение в рассматриваемых условиях, поскольку включает те факторы, которые отсутствуют или представлены ограниченно в зарубежных наборах данных.

С точки зрения обучения моделей, основанных на данных, важным является то, что исходные записи формируются в виде временных рядов с шагом дискретизации 1 с. При использовании окна длины $seq_len = 10$ каждая непрерывная сессия длиной T_j отсчётов формирует

$$N_j^{seq} = \max(0, T_j - seq_len)$$

обучающих последовательностей, а общий объём обучающих примеров определяется выражением

$$N^{seq} = \sum_{j=1}^{J} \max(0, T_j - seq_len),$$

где J - число непрерывных сегментов после предварительной обработки. Таким образом, даже при умеренной длительности отдельных поездок суммарная протяжённость более 1800 км и секундная дискретизация обеспечивают формирование большого числа перекрывающихся последовательностей, достаточного для обучения и валидации нейросетевых моделей LSTM, CNN-LSTM и

полносвязных сетей. При этом данные сохраняют не только объём, но и содержательную разноплановость, необходимую для проверки устойчивости моделей при изменении маршрута, сезона и погодного класса.

Обработка собранных данных

Предварительная обработка собранных микроскопических данных выполнялась с целью обеспечения физической правдоподобности временных рядов, синхронизации данных между участниками (лидер/ведомый) и внешними источниками, устранения пропусков и измерительных артефактов, характерных для GPS/мобильных сенсоров, и подготовки согласованного набора признаков для обучения и оценки моделей следования за лидером в дискретном времени.

Исходные записи формировались мобильным клиентом и поступали на сервер с дискретизацией 1 с. Каждая запись включала: временную метку, географические координаты (широта, долгота), скорость, направление движения (угол), идентификатор пользователя, идентификатор выбранного автомобиля, а также связанный набор метеорологических признаков (температура, осадки и др.), сопоставляемых по времени и координатам. Временные метки исходно фиксировались с миллисекундной точностью и далее приводились к единому формату, обеспечивающему корректную сортировку и повторную выборку временных рядов. На этом же шаге выполнялась проверка монотонности временной оси внутри каждой поездки и устранение возможных дубликатов, возникающих при сетевых повторах отправки.

Для последующей воспроизводимой обработки данные сегментировались на отдельные сессии (поездки) с назначением уникального идентификатора `trip_id`. Сегментация выполнялась по комбинации признаков: пользователь, автомобиль и непрерывный временной интервал, отделённый от соседних интервалов паузой/разрывом. Начальные и конечные фрагменты, соответствующие парковочным манёврам, стоянке, запуску/остановке приложения и иным нехарактерным для режима следования состояниям, исключались из рассмотрения, поскольку они приводят к смещению распределений скорости и дистанции и увеличивают долю аномальных переходов.

Данные, получаемые со смартфона, содержат шум и единичные аномалии, обусловленные особенностями GPS, а также нестабильным сетевым соединением. В связи с этим применялся каскад процедур контроля качества:

1. **Удаление грубых выбросов скорости.** Редкие значения скорости, противоречащие динамическим ограничениям и характеру дорожного сценария, маркировались как выбросы и заменялись локальными оценками (средним ближайших корректных соседних значений) либо исключались в зависимости от длительности аномального фрагмента.
2. **Исправление одиночных нулей скорости.** Если фиксировалась последовательность вида $v_{t-1} > 0$, $v_t = 0$, $v_{t+1} > 0$ при отсутствии иных признаков остановки (кратковременность и согласованность координат), значение v_t считалось измерительным артефактом и заменялось интерполяцией по соседним отсчётам.
3. **Разрешение противоречий при параллельных наблюдениях.** В случаях, когда при близких временных метках в пределах одного контекста наблюдались противоречивые значения (часть записей 0 и часть - реалистичные скорости), применялось правило согласования, при котором предпочтение отдавалось физически непротиворечивому варианту (ненулевому) при подтверждении большинством наблюдений в окне.

Отдельно подчёркивается, что сглаживание применялось таким образом, чтобы не подавлять реальные переходные процессы (разгон/торможение), а устранять только одиночные/редкие артефакты измерения.

Ключевым этапом подготовки микроскопических данных является формирование согласованных парных наблюдений “лидер-ведомый”. На серверной стороне для каждого момента времени использовалась логика определения совместного движения (находятся ли два ТС на одном направлении движения и в пространственной близости) и логика определения положения “впереди/сзади” по проекции относительного вектора положения на вектор направления движения. В результате каждой записи присваивался статус `isLeader`, после чего выполнялось выравнивание наблюдений лидера и ведомого по временным меткам.

На уровне выборки формировались два синхронизированных временных ряда скорости:

$$v_t^f \equiv \texttt{speed_follower}, \qquad v_t^\ell \equiv \texttt{speed_leader},$$

что необходимо для постановки задачи прогнозирования в терминах продольного взаимодействия двух ТС.

Динамический габарит является базовым признаком для моделей следования за лидером и одновременно одним из основных показателей безопасности. В рамках предложенного метода она вычислялась на сервере по GPS-координатам парных наблюдений с использованием геодезической формулы. Практика показала, что при дискретизации порядка 1 секунды возможны пропуски вычисленной дистанции, возникающие из-за несинхронности поступления записей или временной потери измерений у одного из участников пары.

Для обеспечения непрерывности временного ряда дистанции применялась линейная интерполяция на коротких разрывах. Длинные разрывы рассматривались как границы сегментов и приводили к разбиению поездки на под-сегменты, чтобы исключить неинформативные вставки. Такой подход сохраняет корректную динамику дистанции в режимах следования и позволяет использовать последовательностные модели без искусственных скачков.

Метеорологические характеристики были сопоставлены с поездками на основе времени и координат. Они использовались в качестве внешних факторов, влияющих на расстояние поездки и динамику движения по долготе. Для дальнейшего анализа и для возможности разделения выборки поездки были дополнительно отнесены к классам погоды, характерным для всего набора данных. Сезон (весна/лето/осень/зима) определялся на основе календарной даты поездки и использовался в качестве дополнительной переменной для анализа применимости моделей к разным сезонам.

Для учёта поведенческой и технической неоднородности в выборку включались атрибуты водителя и автомобиля. Пол кодировался бинарно (0 - мужчина, 1 - женщина). Возраст и стаж вождения использовались как числовые признаки. Год выпуска автомобиля также оставлялся числовым. Категориальный признак модели автомобиля кодировался методом **label encoding** (каждой модели соответствует целочисленный идентификатор), что обеспечивает масштабируемость при увеличении числа уникальных моделей без разрастания размерности признакового пространства, характерного для one-hot кодирования.

Итоговый датасет строился на уровне синхронизированных парных наблюдений лидер-ведомый и включал динамические и статические признаки:

$$\big[\texttt{speed_follower}, \texttt{speed_leader}, \texttt{distance},$$
$$\texttt{temp}, \texttt{precip}, \texttt{age},$$
$$\texttt{driving_experience}, \texttt{gender},$$
$$\texttt{car_year}, \texttt{car_model}\big].$$

Для обучения последовательностных моделей формировались окна фиксированной длины seq_len (в работе использовалось $seq_len = 10$), что соответствует использованию последних десяти отсчётов состояния для предсказания следующего значения целевой переменной (скорости ведомого). При построении окон обеспечивалась временная непрерывность внутри каждого `trip_id` и исключалось перешивание окон через границы разрывов и смены контекста.

Описанный метод обработки обеспечивает переход от сырого потока мобильной телеметрии к структурированным и синхронизированным временным рядам в терминах задачи следования за лидером. Это создаёт основу для корректного обучения и последующей замкнутоконтурной оценки моделей следования за лидером с учётом многообразия дорожных сценариев, сезонов и погодных условий, представленных в собранном корпусе данных.

3.3 Разработка информационной системы функционирования модели следования за лидером

В предыдущих разделах были рассмотрены актуальные публикации, теоретические аспекты моделей следования за лидером и организация сбора данных в реальных условиях вождения. Основываясь на этой базе, в данном разделе представлены детали реализации клиент-серверного мобильного приложения, разработанного для организации моделирования поведения автомобиля при следовании за лидером на основе данных. В разделе описываются выбранные технологии, архитектура приложения и его модульные компоненты, включая контроллеры на стороне сервера и схему базы данных.

Была разработана основа будущей системы, архитектура которой представлена на рисунке 3.2. Эта версия приложения предполагала упрощенный сценарий использования, в рамках которого пользователь отправляет со своего смартфона на борту автомобиля данные, которые серверная сторона обрабатывает и записывает в базу данных. Это нужно для организации сбора первоначального набора данных, чтобы протестировать в целом возможность такого подхода и обучить модель с выбранной архитектурой на этих данных.

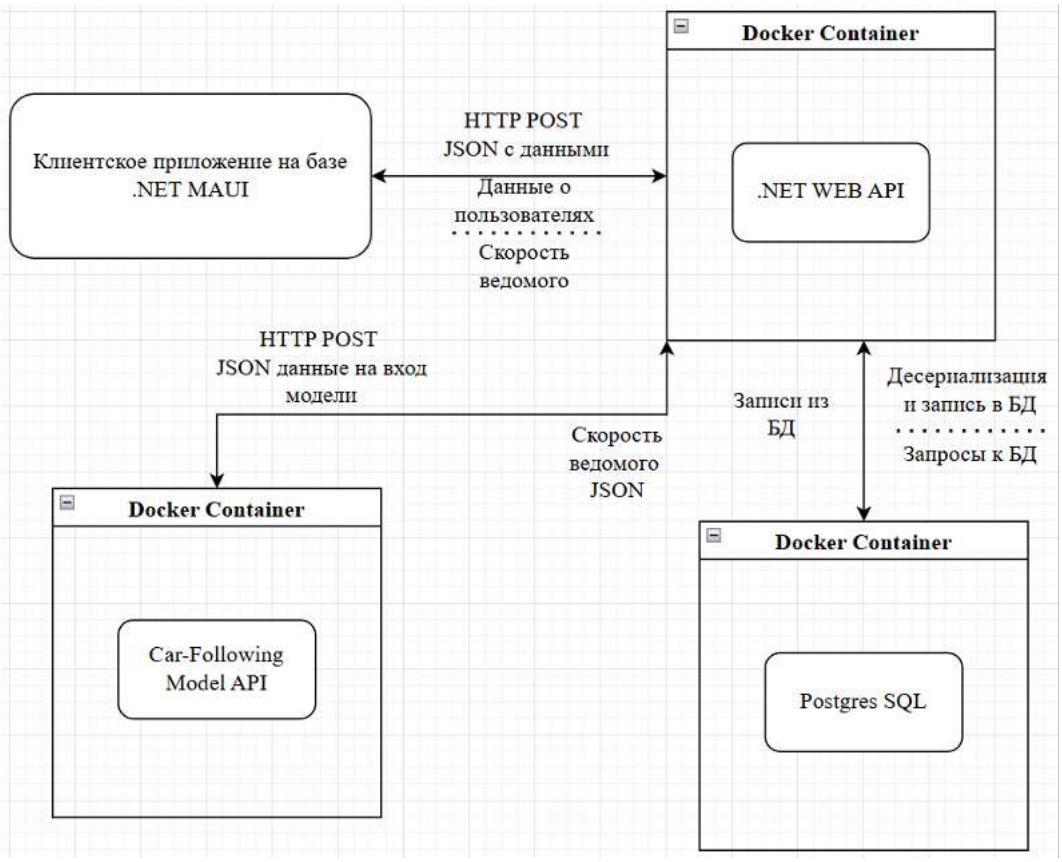

Рисунок 3.2 — Архитектура приложения

Пользовательский интерфейс клиентского мобильного приложения представляет из себя 4 страницы: страница входа/регистрации, страница регистрации аккаунта, основная страница с функционалом передачи данных и получения рекомендаций, страница с информацией о пользователе и возможностью добавить автомобили, которые будут привязаны к конкретному пользователю.

По нажатию кнопки «Начать передачу данных» запускается метод Start_SendingData, который инициирует GPS-слежение с помощью дополнительно установленного плагина CrossGeolocator. Последний реагирует на изменения в положении смартфона в пространстве и позволяет посредством

обработчика событий OnPositionChanged отображать на экране текущие координаты и скорость автомобиля, а также направление в градусах. При этом система спросит у пользователя разрешение на использование GPS при работе данного приложения. Данные о широте, долготе, скорости хранятся в переменных уровня класса, чтобы их можно было упаковать в HTTP-запрос. Затем вызывается метод SendDataToServerAsync(). Здесь происходит фактическое взаимодействие с сервером. Метод создает анонимный объект, содержащий поля данных: широта, долгота, скорость, имя пользователя, current_car и угол направления. Этот объект сериализуется в тип JSON с помощью System.Text.Json.JsonSerializer.Serialize. Для отправки данных приложение использует статический экземпляр HttpClient. Полезная нагрузка в формате JSON оборачивается в объект StringContent с кодировкой UTF-8 и типом MIME application/json. Затем выполняется POST-запрос к конечной точке сервера /api/MainData с помощью HttpClient.PostAsync. Эта операция является асинхронной и включает в себя обработку исключений, чтобы сообщить о любых сетевых ошибках в пользовательский интерфейс.

На клиентской стороне также работает метод StartWeatherUpdates, который вызывает вспомогательный метод SendWeatherAsync и аналогичным вышеописанному образом отправляет на отдельную конечную точку на сервере Post запрос с координатами и именем пользователя. Эти действия выполняются раз в 60 минут, так как погода не меняется часто, нет смысла создавать лишнюю нагрузку на сеть передачи данных и сервер. Обработка сервером данного запроса рассматривается далее.

В случае успешного выполнения запроса клиент получает от сервера ответ в формате JSON. В ответе содержится необходимая для эксперимента информация: прогнозируемая скорость и текущий статус лидера/ведомого. Приложение десериализует этот ответ в DTO-класс MainDataResponse. Элементы пользовательского интерфейса - значение предсказанной скорости и метка статуса лидера, обновляются в главном потоке посредством объектов класса MainDataResponse, чтобы отразить данные, полученные от сервера.

При работе со страницей, посвященной данным пользователя - Profile1, конструктор извлекает логин текущего пользователя из локального хранилища устройства с помощью метода Preferences.Get. Эта строка логина затем используется для асинхронного запроса полного профиля пользователя из Web API посредством вызова метода GetUserDataFromServer. Он выполняет HTTP GET-

запрос к конечной точке /api/User/userName. Сервер отвечает JSON-объектом с информацией о пользователе, который включает в себя не только имя и ID, но и список ТС, связанных с этим пользователем. Ответ десериализуется в DTO-класс UserC с помощью JsonConvert.DeserializeObject. Необходимо также рассмотреть, как реализована серверная логика разработанного приложения. В основе функциональности лежит класс MainDataController.cs. Этот контроллер отвечает за получение данных о вождении от клиента. Когда на конечную точку /api/MainData поступает POST-запрос, контроллер извлекает полезную нагрузку в формате JSON в DTO-класс MainData. К данным добавляется временная метка для согласования записей, применяется фильтр Калмана для сглаживания значения угла направления движения автомобиля. Затем информация сохраняется в таблице main_data посредством класса AppDbContext, который наследует DbContext. Если пользователь не является лидером, контроллер пытается получить прогноз скорости, агрегируя последние записи этого пользователя. То есть информация о последних 10 записанных скоростях запрашивается в базе данных по имени пользователя, который на данный момент имеет статус ведомого. Создается отдельный объект с необходимой для модели информацией и отправляется в виде файла JSON в сервис FastAPI. Ответ, включающий прогнозируемую скорость, возвращается клиенту вместе с другой необходимой информацией.

Помимо логики, отвечающей за создание новых записей в базе данных и реализации POST запросов в API модели, разработано несколько необходимых для поставленной перед приложением задачи. Эти методы - IsLikelyOnSameRoad, DetermineLeader и UpdateIsLeaderStatus - используются для определения, двигаются в данный момент времени ТС по одной и той же дороге, для выявления лидера в колонне автомобилей, а также для сохранения отображения этих состояний в базе данных и на графическом интерфейсе пользователя.

Метод IsLikelyOnSameRoad предназначен для оценки того, находятся ли два ТС на одной дороге и движутся ли они в одном направлении. Он принимает в качестве аргументов два объекта MainData и сначала вычисляет расстояние с помощью класса DistanceCalculator по формуле Хаверсина, используя координаты. Если это расстояние не превышает 500 метров, метод считает, что автомобили находятся на одной дороге. Если же дистанция превышает заданный порог, сравниваются значения углов движения, которые представля-

ют собой направление движения автомобиля в градусах. Если угловая разница между двумя значениями составляет менее 60 градусов с учетом кругового обхода, метод возвращает True, указывая на то, что автомобили, скорее всего, движутся в одном направлении по одному и тому же участку дороги.

Метод DetermineLeader предоставляет логику для вывода информации о том, какой из двух автомобилей является лидером в ситуации следования. Для вычисляется вектор относительного положения между двумя ТС и проецируется на вектор движения текущего. В частности, вычисляется дельта широты и долготы, преобразуется в метры, а затем вычисляется проекция относительного вектора на нормализованный вектор направления, полученный из угла направления ТС. Если проекция положительная, это означает, что другой автомобиль впереди. В противном случае лидирует текущий автомобиль. На основании этого обновляется статус лидера.

Метод UpdateIsLeaderStatus содержит логику обновления базы данных для вышеописанного процесса. Он извлекает последнюю запись для данного пользователя из таблицы main_data и проверяет, соответствует ли текущий статус лидера потенциальному обновлению. Если нет, метод изменяет флаг isleader и сохраняет изменения. Это гарантирует, что роли лидера и ведомого последовательно отслеживаются и точно отражаются в базе данных. На стороне сервера работают и другие контроллеры для поддержки функционала, связанного с обработкой данных о погоде, данных о пользователях и связанных с ними автомобилях.

Класс UserController.cs организацию логики работы с пользователями. Когда клиент отправляет POST-запрос на регистрацию нового пользователя, контроллер проверяет, существует ли уже пользователь с таким именем в базе данных. Если нет, он добавляет пользователя в соответствующую таблицу и возвращает идентификатор вновь созданного пользователя. Для получения данных о пользователях используется конечная точка по адресу /api/User/userName. Посредством последней есть возможность извлекать информацию о пользователе по его имени. Также в ответе возвращаются данные о связанных с пользователем автомобилях. Полученный объект сериализуется и возвращается на клиента в формате JSON, позволяя клиенту заполнить необходимые для пользовательского интерфейса поля, профиль пользователя или выбор автомобиля.

CarDataController.cs предназначен для организации функциональности регистрации автомобилей. Когда клиент отправляет данные о новом автомобиле через POST, контроллер сначала проверяет, существует ли идентификатор такой пользователя в базе данных. В случае успешной проверки автомобиль добавляется в соответствующую таблицу. Эта функциональность позволяет в рамках эксперимента одному водителю использовать несколько автомобилей.

Данные о погоде собираются посредством WeatherController.cs, который получает данные о местоположении от клиента и извлекает соответствующую информацию о погоде с помощью внешнего API. Он делает GET-запрос к сервису WeatherAPI, используя полученные с клиента широту и долготу, из полученного в ответе JSON файла извлекает необходимую информацию: температуру, скорость ветра и количество осадков, и сохраняет в таблице базы данных weather_data.

API для взаимодействия модели с клиентским приложением содержит один метод POST с конечной точкой /predict-speed,. Эта конечная точка реализуется с помощью декоратора @app.post(«/predict-speed») и привязывается к функции predict_speed. При получении POST-запроса FastAPI автоматически разбирает полезную нагрузку JSON на список объектов Record благодаря валидации данных Pydantic. Каждая запись содержит поля speed_follower, speed_leader, distance, temp, precip_mm, age, driving_experience, gender_encoded, car_year и car_model_encoded. Эти признаки представляют собой с одной стороны динамическое состояние автомобиля в моменты следования за лидером, а с другой статические атрибуты водителя и автомобиля, формируя входное пространство для модели.

Сначала функция проверяет, что список содержит не менее 10 записей -это важное требование, поскольку модель ожидает последовательность входов seq_len = 10. Если записей оказывается недостаточно, метод выдает ошибку HTTP 400 с информационным сообщением, возвращаемым клиенту в формате JSON. Далее входящие данные преобразуются в Pandas DataFrame для облегчения предварительной обработки. После проверки данные упорядочиваются в соответствии с порядком признаков модели и масштабируются. К нормализованным данным применяется скользящее окно, чтобы сгенерировать последовательные входные данные для модели. Эти последовательности преобразуются в тензор PyTorch. После вычисления прогнозируемые результаты

преобразуются из нормализованной шкалы в денормализованные значения скорости.

Для поддержки основной функциональности мобильного клиент-серверного приложения необходима структурированная реляционная база данных. Она применяется для хранения всех входящих данных, профилей пользователей, метаданных автомобиля, погодных условий.

База данных состоит из нескольких сущностей: User, Car, MainData, WeatherData и ClientDistance. Схема базы данных представлена на рисунке 3.3.

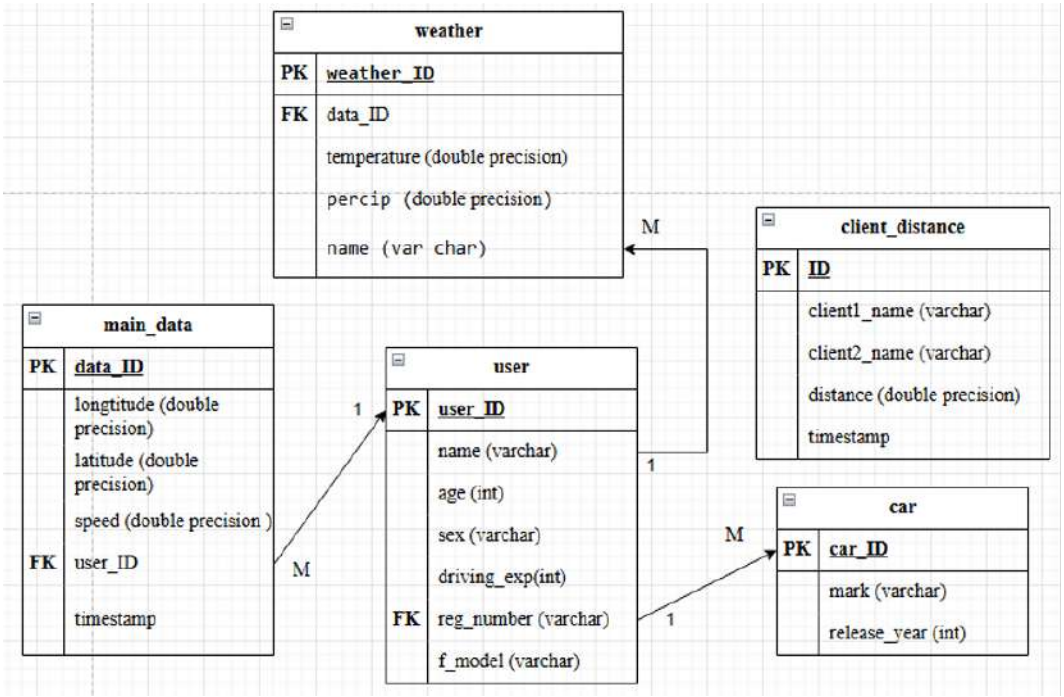

Рисунок 3.3 — Схема базы данных

Отношение между таблицами User и Car - один-ко-многим. Каждый пользователь может зарегистрировать несколько автомобилей, но каждый автомобиль принадлежит только одному пользователю. Эта связь определяется внешним ключом в таблице Car, указывающим на первичный ключ User. Такая структура поддерживает требование к пользователям управлять несколькими автомобилями, что отражено в пользовательском интерфейсе клиента, где реализована возможность из списка своих зарегистрированных автомобилей.

Каждая запись в таблице MainData связана с конкретным пользователем, идентифицируемым полем username. Устанавливается логическая связь «один ко многим» - один пользователь может создавать множество записей с данными. Как и в случае с MainData, таблица WeatherData содержит поле имени пользователя, образуя связь «один-ко-многим» между User и WeatherData. Данные о

погоде извлекаются на основе GPS-координат, предоставленных клиентом, поэтому в базе данных необходимо отобразить соответствие между пользователем и местоположением. Также стоит отметить, что сервер извлекает последнюю запись о погоде именно для данного пользователя при построении входного набора признаков. Сущность MainData выступает основной в процессе сбора данных. В ней хранятся GPS-координаты, скорость, направление движения автомобилей и атрибуты состояния, которые показывают является ли пользователь лидером или ведомым. Каждая запись MainData связана с пользователем и неявно с конкретным автомобилем. Данные записываются каждую секунду во время поездки и выступают необходимой основой для вычисления расстояния между автомобилями и запуска процесса рекомендаций через модель посредством FastAPI.

Данные WeatherData привязаны к пользователю, а также косвенно к конкретному времени и местоположению.

В таблице ClientDistance регистрируются вычисленные расстояния между парами пользователей, которые фиксируются, когда их местоположение и направление движения удовлетворяют условиям, обозначенным выше.

Таблицы, зависящие от времени (MainData, и ClientDistance), включают поле временной метки. Это важная деталь, поскольку используемая модель зависит от последовательностей данных, а не только от отдельно взятых значений.

Практическая апробация разработанного клиент-серверного приложения показала возможность его использования не только как демонстрационного программного средства, но и как инструмента натурного сбора микроскопических данных транспортного потока. В ходе экспериментальных поездок приложение использовалось для регистрации координат, скорости, направления движения, идентификатора пользователя, выбранного автомобиля и связанных погодных параметров. Система обеспечивала формирование последовательных временных рядов с дискретизацией порядка 1 с, что соответствует требованиям последующего построения окон длиной $seq_len = 10$ для обучения и применения моделей следования за лидером.

К числу практически значимых характеристик разработанного приложения относятся: возможность одновременного сопровождения нескольких мобильных клиентов, соответствующих различным ТС; автоматическое определение пары лидер-ведомый на основе координат, направления движения и

взаимного расположения автомобилей; вычисление межмашинной дистанции на серверной стороне; обновление динамических параметров движения с секундной дискретизацией; периодическое обновление погодных данных с интервалом 60 мин; сохранение всех наблюдений в реляционной базе данных с временными метками и последующая передача последних 10 записей ведомого ТС в сервис прогнозирования скорости.

В практическом режиме функционирования один цикл обработки включает получение телеметрии от мобильного клиента, запись данных в таблицу `main_data`, проверку совместного движения ТС, определение статуса лидера или ведомого, расчёт дистанции между участниками, формирование входного окна признаков и обращение к сервису модели через API. Такая организация позволяет использовать приложение в замкнутом контуре. Данные, полученные от пользователя, не только сохраняются для последующего анализа, но и используются для оперативного получения прогноза скорости ведомого.

Разработанное приложение было использовано при формировании собственного корпуса микроскопических данных транспортного потока. Тем самым подтверждена практическая реализуемость предложенной схемы сбора данных. Приложение может применяться в обычных дорожных поездках без установки специализированного оборудования в автомобиль, а расширение числа участников эксперимента достигается за счёт установки мобильного клиента на дополнительные смартфоны. Это обеспечивает масштабируемость сбора данных и возможность постепенного расширения корпуса наблюдений за счёт новых маршрутов, погодных условий, водителей.

Правовая охрана разработанного программного решения подтверждается зарегистрированным результатом интеллектуальной деятельности: свидетельство о государственной регистрации программы для ЭВМ № 2020612021, дата регистрации 13.02.2020; зарегистрированным результатом интеллектуальной деятельности: свидетельство о государственной регистрации программы для ЭВМ № 2020612483, дата регистрации 25.02.2020. Свидетельства представлены в приложении К.

Наличие зарегистрированных РИД подтверждает самостоятельность разработки и возможность её дальнейшего использования как программного компонента информационной системы сбора, хранения и обработки микроскопических данных транспортного потока.

3.4 Разработка метода динамической коррекции коэффициентов модели следования за лидером

В этом разделе рассматривается проблема динамической корректировки коэффициентов C_0, C_1, C_2 на основе контекстной информации о движении и внешних условиях. Цель исследования — определить, какие методы аппроксимации обеспечивают наилучший компромисс между точностью оценки требуемого расстояния и скорости, стабильностью оценок при изменении дорожной обстановки и сохранением физически правдоподобного поведения в замкнутой системе. Важным требованием является возможность переноса результатов на различные типы маршрутов. Влияние сезонности и погодных условий рассматривается отдельно, поскольку они влияют на схемы выбора расстояния и характер продольных маневров.

Исследование построено как сравнение методов, различающихся степенью нелинейности и способностью учитывать взаимодействие между факторами. В качестве базового метода используется линейная регрессия, поскольку она позволяет оценить направление влияния контекстных особенностей на коэффициенты и служит эталоном интерпретируемости.

Полиномиальная регрессия рассматривается как контролируемое расширение базовой модели, поскольку она вводит нелинейные зависимости и взаимодействия между признаками без использования неинтерпретируемых представлений. Для более гибкой аппроксимации используется градиентный бустинг, способный учитывать неоднородность наблюдений и нелинейные эффекты, возникающие из-за изменений в условиях дорожного движения и погодных условиях.

Кроме того, рассматриваются подходы на основе нейронных сетей, включая NN, LSTM и CNN-LSTM, поскольку в реальных условиях вождения коэффициенты могут зависеть от динамического контекста, а не только от состояния в конкретный момент времени. При таком подходе коррекция может основываться на короткой истории движения, отражая инерцию реакции водителя и задержку в адаптации к изменяющимся условиям.

Методы сравниваются в двух взаимодополняющих режимах. В первом режиме качество оценивается как точность оценки коэффициентов на основе данных, что позволяет проанализировать, насколько хорошо метод адаптирует-

ся к контексту и воспроизводит наблюдаемое поведение в выборке. Во втором режиме коррекция проверяется в замкнутой системе, где оценивается влияние прогнозируемых коэффициентов на динамику следующего транспортного средства и на безопасность вождения. Этот подход имеет фундаментальное значение, поскольку даже небольшие ошибки в коэффициентах могут привести к накоплению отклонений с течением времени и вызвать нестабильное поведение. Верификация с обратной связью позволяет нам различать методы, которые хорошо соответствуют наблюдениям, и те, которые остаются стабильными и правдоподобными в ходе моделирования траекторий.

Особое внимание уделяется устойчивости коррекции при переносе результатов между различными сценариями и сезонами. С этой целью процедура оценки разработана таким образом, что тестовые сегменты отличаются от обучающих сегментов по типу маршрута или сочетанию сезонных и погодных условий. Такая экспериментальная схема позволяет проверить, является ли коррекция действительно динамической и контекстно-зависимой, или же она просто воспроизводит статистические данные определенного подмножества данных. В дополнение контролируется диапазон допустимых значений коэффициентов, чтобы исключить режимы, в которых коррекция приводит к аномальным или небезопасным требованиям по дистанции. Это обеспечивает сопоставимость результатов между методами и снижает риск переобучения, которое проявляется в виде экстремальных значений C_0, C_1, C_2 на редких комбинациях признаков.

Таким образом, исследование направлено на построение и проверку методов динамической коррекции C_0, C_1, C_2, которые учитывают многообразие дорожных сценариев и внешних условий, представленных в корпусе данных, и обеспечивают устойчивое применение в задачах моделирования следования за лидером в реальном движении.

Постановка задачи динамической коррекции коэффициентов модели следования за лидером Динамическая коррекция рассматривается как задача восстановления отображения

$$g:\ \mathbf{z}_t \mapsto \widehat{\mathbf{C}}_t, \qquad \widehat{\mathbf{C}}_t = (\widehat{C}_{0,t},\widehat{C}_{1,t},\widehat{C}_{2,t}),$$

которое задаёт адаптированные параметры динамического габарита и тем самым определяет требуемую дистанцию $\widehat{s}_{\mathrm{req}}(t)$. Обучение (или калибровка)

функции $g(\cdot)$ может выполняться по собранным траекториям, минимизируя расхождение между наблюдаемой динамикой следования и динамикой, индуцируемой условием безопасности вида

$$s(t) \geqslant \widehat{s}_{\text{req}}(t),$$

а также удовлетворяя кинематическим ограничениям на скорость и ускорение.

Обучение нейросетевых моделей выполнялось с использованием PyTorch. Для ускорения вычислений применялась CUDA на графическом процессоре NVIDIA GeForce RTX 4070 Ti. Входные признаки включали динамические параметры пары «лидер-ведомый» и контекстные признаки. Данные разделялись на обучающую, тестовую и валидационную выборки в соотношении 70/20/10. Для последовательных моделей формировались окна фиксированной длины $seq_len = 10$, что позволяло учитывать краткосрочную историю движения.

В задаче прогнозирования скорости ведомого ТС были обучены три архитектуры. LSTM была реализована одним LSTM-слоем с размерностью скрытого состояния 256 и полносвязным выходным слоем. Использовался оптимизатор Adam с шагом обучения 0.01 и функция потерь MSE. Обучение выполнялось батчами размера 64 в течение 20 эпох с ранней остановкой при отсутствии улучшений в течение 5 эпох. Для выбора ключевых гиперпараметров применялся метод grid search. Результаты представлены на рисунке 3.4.

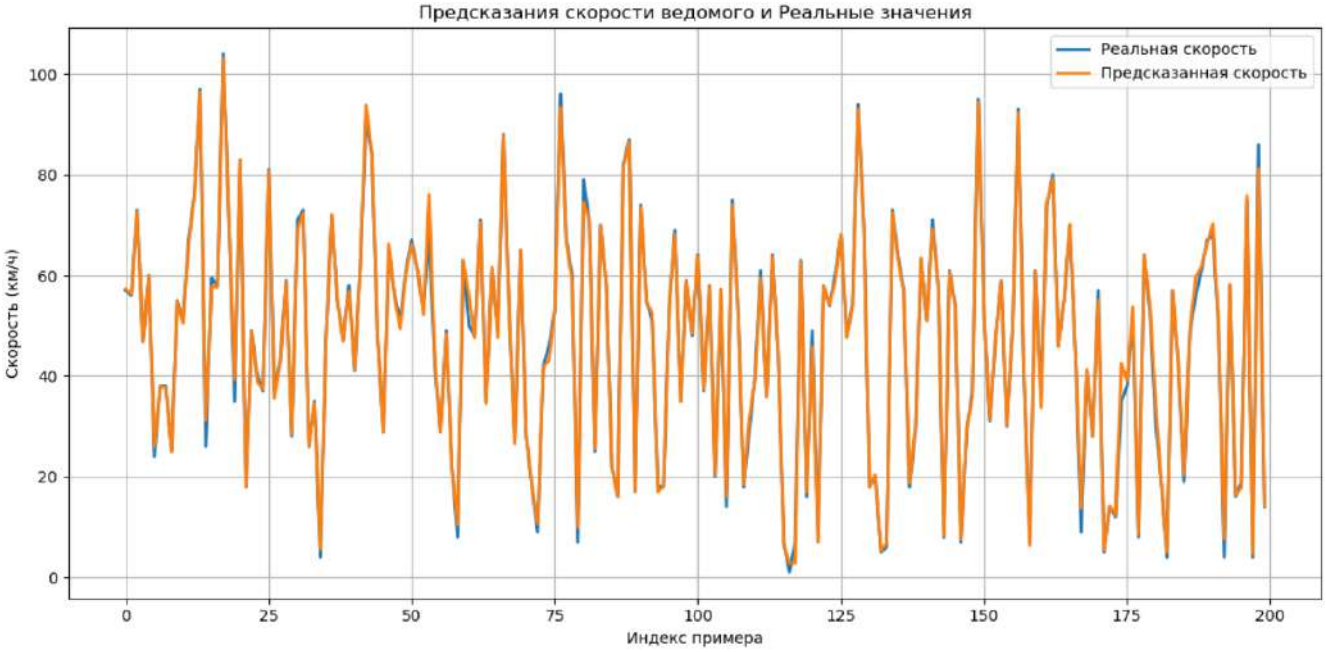

Рисунок 3.4 — Результаты предсказаний LSTM

Полносвязная нейронная сеть использовала два скрытых слоя по 256 нейронов с активацией ReLU и линейный выход. Поскольку данная архитектура

не моделирует временную структуру напрямую, последовательность длины 10 преобразовывалась в единый входной вектор посредством time-delay embedding. Результаты представлены на рисунке 3.5.

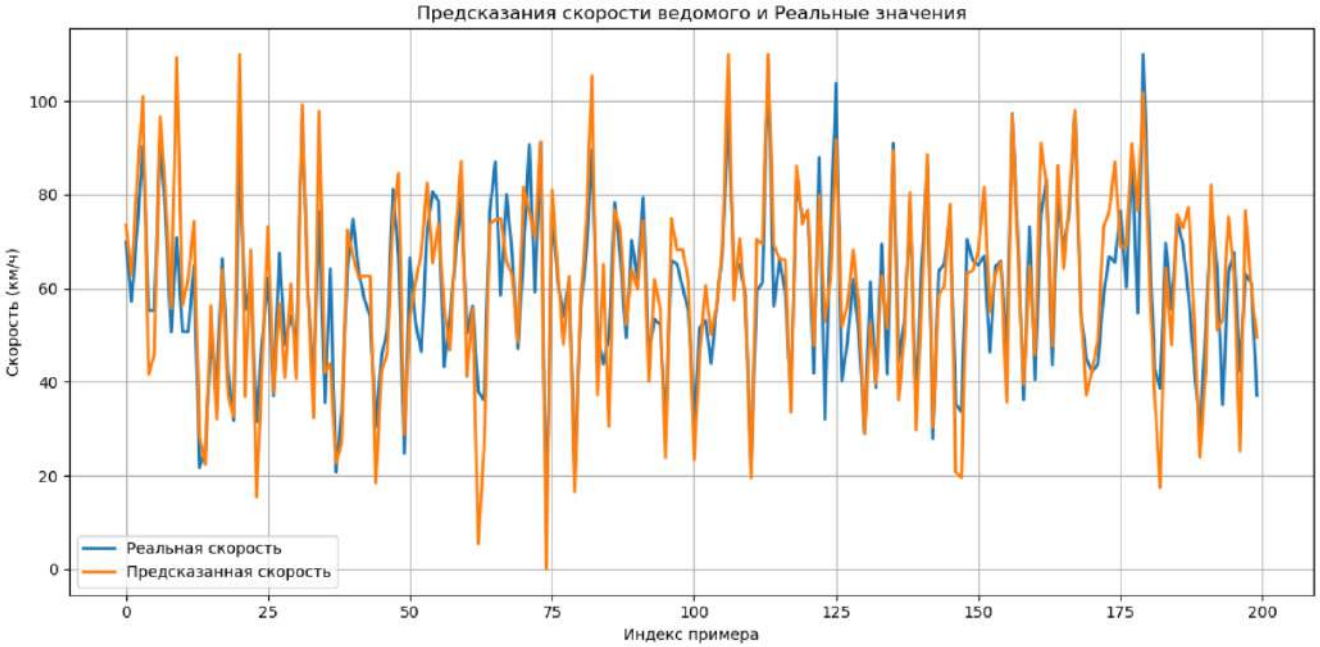

Рисунок 3.5 — График предсказаний полносвязной нейронной сети

Модель CNN-LSTM включала свёрточный слой с 64 фильтрами, затем применялись batch normalization и ReLU. Извлечённые признаки передавались в LSTM-блок с одним слоем и 64 нейронами. Обучение проводилось оптимизатором Adam батчами размера 128 в течение 60 эпох при функции потерь MSE. Результаты представлены на рисунке 3.6.

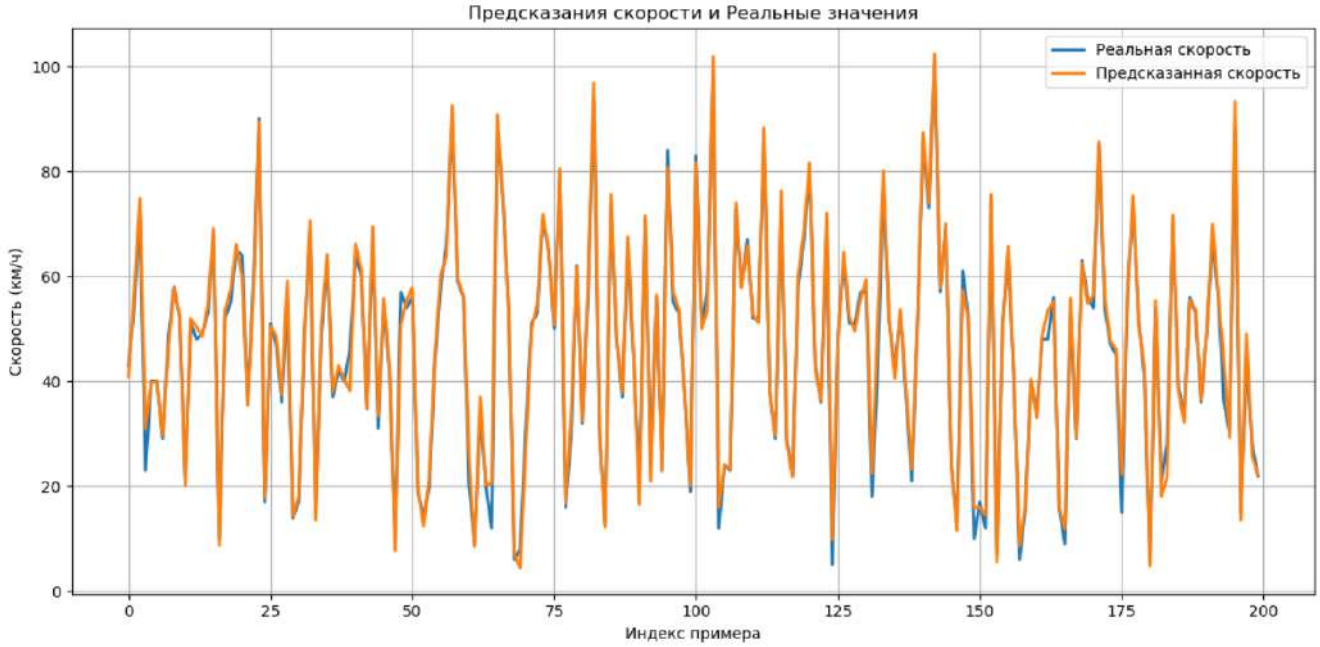

Рисунок 3.6 — График предсказаний гибридной модели CNN-LSTM

Качество оценивалось по RMSE. По результатам обучения LSTM продемонстрировала наилучшее качество с RMSE порядка 1.2 км/ч. CNN-LSTM показала RMSE порядка 3 км/ч. Полносвязная сеть показала RMSE порядка 9.8 км/ч. Полученный результат интерпретируется как подтверждение важности учёта временной зависимости в данных. При рассматриваемой конфигурации CNN-LSTM не превзошла LSTM, что указывает на отсутствие выигрыша от усложнения архитектуры при данных и выбранном наборе гиперпараметров.

Для прогнозирования динамического габарита были рассмотрены линейная регрессия, полиномиальная регрессия и градиентный бустинг. Данные модели выбирались как воспроизводимые базовые методы, позволяющие оценить вклад признаков и качество аппроксимации на разнородных условиях движения. Качество оценивалось метриками MSE, RMSE, MAE и R^2. Линейная регрессия дала MSE = 615.80, RMSE = 24.82 , MAE = 15.65 , R^2 = 0.20. Полиномиальная регрессия дала MSE = 596.01, RMSE = 24.41 , MAE = 15.27 , R^2 = 0.22. Градиентный бустинг показал более высокое качество с MSE = 351.06, RMSE = 18.74 , MAE = 11.87 , R^2 = 0.54. Результаты представлены на рисунках 3.7 и 3.8.

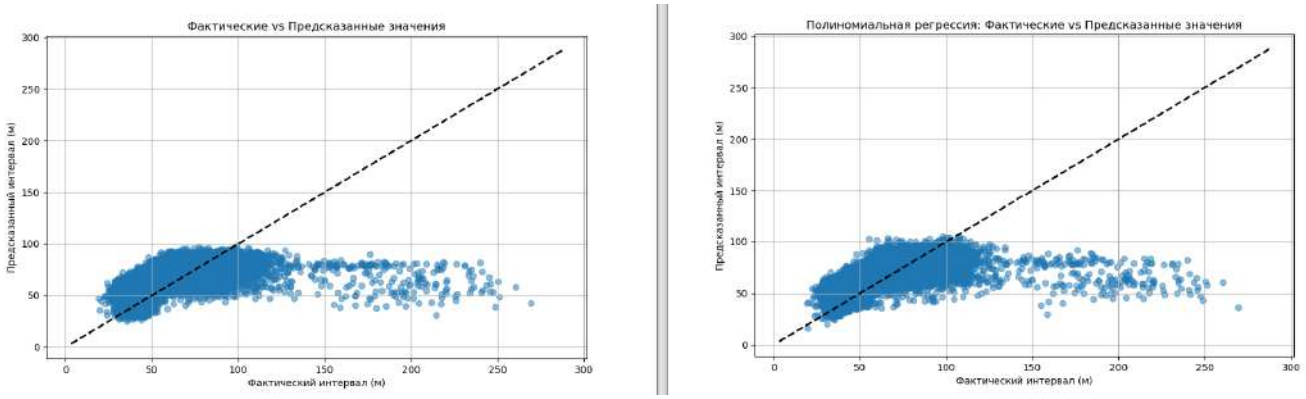

Рисунок 3.7 — Сопоставление фактической и прогнозируемой дистанции для линейной и полиномиальной регрессии на тестовой выборке

В расширенной постановке в набор признаков был добавлен категориальный признак *weather* со значениями от 0 до 6, где 0 соответствует ясной погоде, 1-3 соответствуют дождю по интенсивности, а 4-6 соответствуют снегу по интенсивности. После добавления *weather* качество градиентного бустинга улучшилось до MSE = 289.42, RMSE = 17.01 , MAE = 10.37 , R^2 = 0.63. Для LSTM, обученной в задаче прогнозирования дистанции на обновлённом наборе признаков, получено MSE = 267.78, RMSE = 16.36 , MAE = 10.61 ,

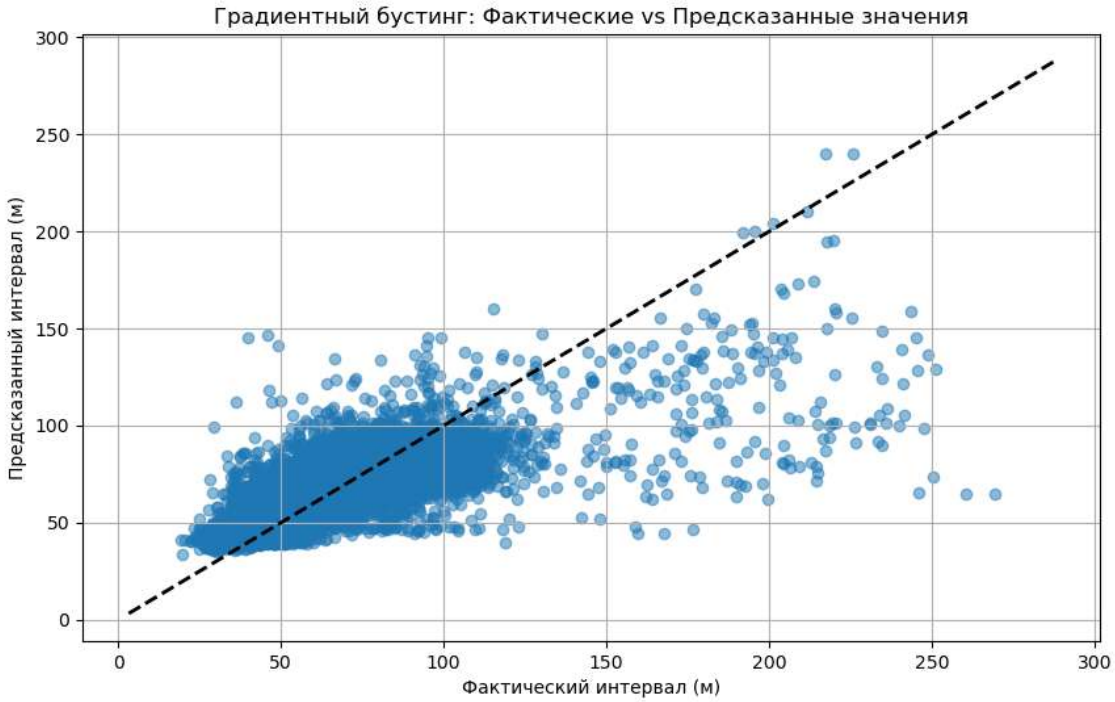

Рисунок 3.8 — Сопоставление фактической и прогнозируемой дистанции для градиентного бустинга на тестовой выборке

$R^2 = 0.67$. Это подтверждает, что внешние условия являются значимым фактором вариативности дистанции и их явное представление улучшает качество модели. Результаты представлены на рисунке 3.9.

Для перехода от предсказаний к интерпретируемым коэффициентам C_0,C_1,C_2 применяется калибровка по множеству наблюдений $\{(v_i,s_i)\}_{i=1}^{n}$. В качестве базовой процедуры предлагается регуляризованная оценка параметров по методу наименьших квадратов

$$\min_{C_0,C_1,C_2} \sum_{i=1}^{n} w_i \Big(s_i - (C_0 + C_1 v_i + C_2 v_i^2) \Big)^2 + \lambda_0 C_0^2 + \lambda_1 C_1^2 + \lambda_2 C_2^2, \qquad (3.11)$$

где веса w_i вводятся для учёта неоднородности качества измерений и режимов движения, а регуляризаторы λ_0,λ_1,λ_2 снижают чувствительность оценок к шуму и редким режимам. Для сохранения физической правдоподобности вводятся ограничения на знаки коэффициентов

$$C_0 \geqslant 0, \qquad C_1 \geqslant 0, \qquad C_2 \geqslant 0. \qquad (3.12)$$

Постановка (3.11)-(3.12) является выпуклой, что обеспечивает устойчивое численное решение и интерпретируемость результата. Калибровка выполняется

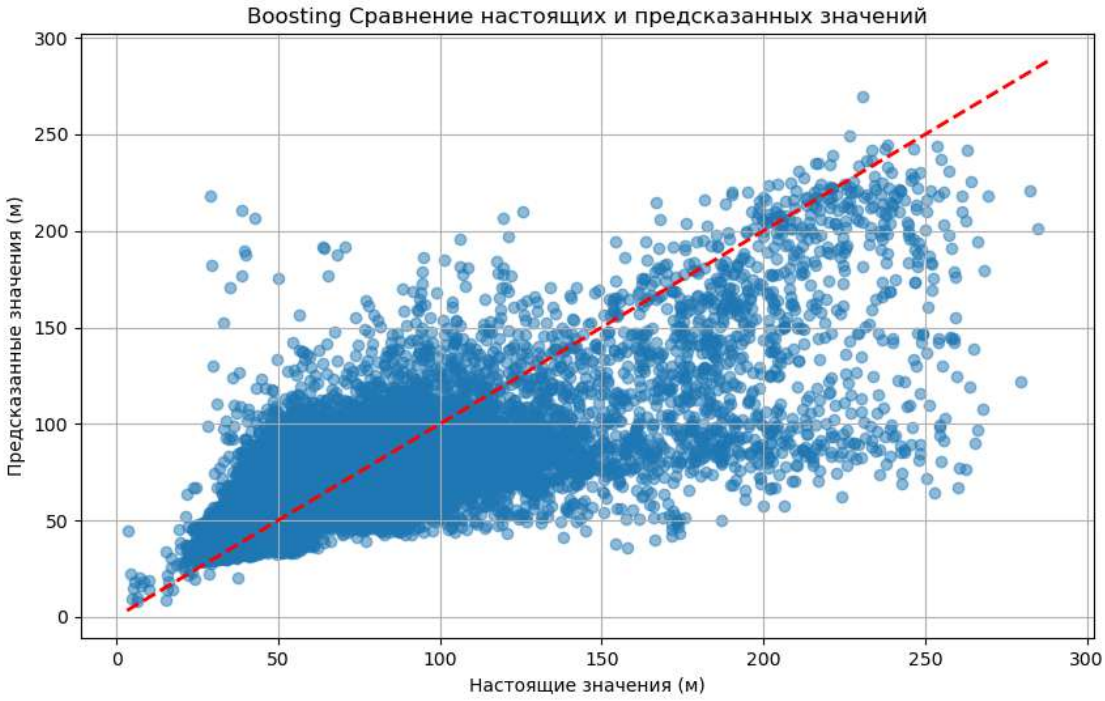

Рисунок 3.9 — Прогноз дистанции градиентным бустингом на обновлённом наборе признаков с *weather*

на отдельных подмножествах данных, классифицированных по сценарию вождения, сезону и погодным условиям. Такой подход позволяет получить набор коэффициентов, адаптированных к конкретным условиям вождения, которые затем можно использовать для динамической коррекции.

Сравнение моделей нейронных сетей для прогнозирования скорости показало, что LSTM превосходит CNN-LSTM и полносвязные сети по показателю RMSE. Это делает LSTM предпочтительной моделью для компонента краткосрочного прогнозирования скорости в системе, где требуется стабильное воспроизведение динамики на протяжении последовательности наблюдений.

В задаче прогнозирования динамического профиля линейная и полиномиальная регрессия показали ограниченную эффективность, о чем свидетельствуют низкие значения R^2 и высокие значения RMSE. Градиентный бустинг продемонстрировал значительные улучшения по всем показателям и остается предпочтительным интерпретируемым базовым методом для прогнозирования расстояний при ограничениях вычислительной сложности. Добавление признака *weather* улучшает производительность как модели бустинга, так и LSTM в задаче определения расстояния. Следовательно, явное кодирование погодных условий следует рассматривать как важный компонент представления признаков.

Для динамической корректировки коэффициентов C_0, C_1, C_2 целесообразно отделить задачу прогнозирования от задачи интерпретируемой параметризации. Нейронные сети и ансамблевые модели используются как в качестве высокоточных предикторов скорости и расстояния, так и в качестве источника контекстно-зависимых эталонных соотношений. Интерпретируемые коэффициенты C_0,C_1,C_2 должны получаться через регуляризованную калибровку по множеству точек (3.11) с ограничениями (3.12) на разделенных поднаборах. Эта двухэтапная структура позволяет нам сочетать точность методов, основанных на данных, с требованием явной интерпретируемости и физической правдоподобности коэффициентов в модели следования за лидером.

Формальное описание метода динамической коррекции коэффициентов модели следования за лидером

Классические модели следования за лидером в практических приложениях, как правило, предполагают фиксированные параметры безопасного разъезда и тем самым слабо отражают изменчивость реальной среды. В натурных условиях параметры безопасного следования зависят от типа дороги, сезонно-погодного режима, видимости и сцепления, а также от поведенческой неоднородности водителей и технических характеристик автомобилей. При переносе фиксированных коэффициентов на новые условия возникает систематическое смещение оценки безопасной дистанции, что приводит либо к чрезмерному консерватизму, либо к риску нарушения безопасности. Для устранения указанного ограничения вводится метод динамической коррекции коэффициентов C_0,C_1,C_2, которая позволяет адаптировать классическую модель к контексту. Одновременно классическая модель с динамически скорректированными коэффициентами используется далее как формальный ограничитель аномальных событий при применении моделей следования за лидером, основанных на данных. [56] Такой ограничитель необходим, поскольку модели основанные на данных демонстрируют высокую точность на распределении обучения, но могут порождать аномальные или небезопасные прогнозы на редких сочетаниях признаков, при сдвиге распределения и при накоплении ошибок в замкнутом контуре.

Обозначения и входные данные

Пусть в дискретные моменты времени t доступны наблюдения пары «лидер-ведомый»

$$\mathcal{S}_t = \left(s_t, v_t^f, v_t^\ell, \Delta v_t\right), \qquad \Delta v_t = v_t^f - v_t^\ell,$$

а также контекстный вектор $\mathbf{z}_t$, включающий по крайней мере тип дорожного сценария, сезон и погодный класс, а при наличии также температурные и осадочные показатели и атрибуты водителя и автомобиля. Предполагается, что Δt является шагом дискретизации, согласованным с частотой сбора данных.

Шаг 1. Контекстное разделение данных. Корпус наблюдений разбивается на набор контекстных сегментов $\{\mathcal{D}_k\}_{k=1}^{K}$, где сегмент задаётся сочетанием факторов $(road_type, season, weather)$ с возможным укрупнением классов при недостатке статистики. Разделение необходимо для устойчивой оценки коэффициентов в условиях неоднородности данных.

Шаг 2. Получение опорной оценки дистанции. Для построения устойчивой опорной зависимости «скорость-дистанция» в каждом сегменте фиксируется опорная модель $\widehat{s}_t$. Выбор опорной модели задаётся результатами проведённого исследования. Линейная и полиномиальная регрессии используются как базовые ориентиры, но из-за ограниченного качества по R^2 и RMSE не рассматриваются как предпочтительные. Для получения более точной опорной оценки применяются либо градиентный бустинг, либо LSTM. Градиентный бустинг рассматривается как предпочтительный метод для опорного прогнозирования дистанции при ограничениях вычислительной сложности и необходимости устойчивой нелинейной аппроксимации. LSTM рассматривается как предпочтительный метод при необходимости учёта последовательной структуры и при наличии выраженной инерционности поведения в данных. Выбор между бустингом и LSTM фиксируется для каждого сегмента по метрикам на валидации. В практическом контуре допускается использование бустинга для дистанции и LSTM для скорости, что согласуется с тем, что LSTM показала наилучшее качество по RMSE в задаче прогнозирования скорости.

Шаг 3. Формирование множества калибровочных точек. В каждом сегменте $\mathcal{D}_k$ формируется множество точек

$$\mathcal{P}_k = \{(v_i, \tilde{s}_i, w_i)\}_{i=1}^{n_k},$$

где v_i берётся как $v_{t_i}^f$, а $\tilde{s}_i$ является целевой дистанцией для калибровки. В качестве $\tilde{s}_i$ допускается использовать либо наблюдаемую дистанцию s_{t_i}, либо опорную оценку $\widehat{s}_{t_i}$, либо их сглаженную комбинацию. Использование $\widehat{s}_t$ целесообразно в сегментах с повышенным уровнем шумов измерения дистанции. Веса w_i задают доверие к точке и понижают вклад наблюдений с артефактами синхронизации, пропусками и короткими разрывами.

Шаг 4. Калибровка $\widehat{C}_{0,k}$,$\widehat{C}_{1,k}$,$\widehat{C}_{2,k}$ по множеству точек. Для каждого сегмента коэффициенты оцениваются решением выпуклой задачи регуляризованной калибровки с ограничениями

$$\widehat{\mathbf{C}}_k = \arg\min_{C_0,C_1,C_2} \sum_{i=1}^{n_k} w_i \Big(\tilde{s}_i - (C_0 + C_1 v_i + C_2 v_i^2)\Big)^2 + \lambda_0 C_0^2 + \lambda_1 C_1^2 + \lambda_2 C_2^2, \quad (3.13)$$

$$C_0 \geqslant 0, \qquad C_1 \geqslant 0, \qquad C_2 \geqslant 0. \quad (3.14)$$

Параметры регуляризации λ_0,λ_1,λ_2 выбираются так, чтобы обеспечить устойчивость оценок при переносе между поездками внутри сегмента. Результатом является таблица сегментных коэффициентов $\mathcal{T} = \{(k, \widehat{\mathbf{C}}_k)\}$.

Шаг 5. Правило обновления коэффициентов в режиме реального времени. В режиме для каждого момента времени t определяется сегмент $k(\mathbf{z}_t)$ и выбираются коэффициенты

$$\widehat{\mathbf{C}}_t = \widehat{\mathbf{C}}_{k(\mathbf{z}_t)}.$$

При смене сегмента допускается сглаживание перехода коэффициентов по времени, чтобы избежать скачков ограничения.

Шаг 6. Вычисление динамически корректируемого требования безопасности. На каждом шаге вычисляется требуемая дистанция

$$s_t^{req} = \widehat{C}_{0,t} + \widehat{C}_{1,t} v_t^f + \widehat{C}_{2,t} (v_t^f)^2, \quad (3.15)$$

после чего применяется нижнее отсечение $s_t^{req} \leftarrow \max(s_t^{req}, s_{\min})$, согласованное с порогом фиксации столкновения ε_c.

Шаг 7. Прогноз модели следования за лидером, основанной на данных. Пусть модель, основанная на данных, формирует прогноз $\widehat{v}_{t+1}^{ML}$ или $\widehat{a}_t^{ML}$ на основе окна длины seq_len. Для совместимости допускается постановка через прогноз скорости. Тогда прогнозируемая дистанция на следующий шаг вычисляется как

$$\widehat{s}_{t+1}^{ML} = \max\bigl(0, s_t + (v_t^{\ell} - \widehat{v}_{t+1}^{ML})\Delta t\bigr).$$

Шаг 8. Детекция аномального события относительно классического ограничения. Прогноз маркируется как аномальный, если

$$\widehat{s}_{t+1}^{ML} < s_{t+1}^{req} \quad \text{или} \quad \widehat{s}_{t+1}^{ML} \leqslant \varepsilon_c, \tag{3.16}$$

и дополнительно, если требуется, контролируются ограничения на скорость и резкость изменения действия в терминах выбранных эксплуатационных пределов.

Шаг 9. Проекция прогноза на допустимое множество. Если аномалия отсутствует, принимается $\widehat{v}_{t+1}^{ML}$. Если аномалия выявлена, строится безопасный прогноз $\widehat{v}_{t+1}^{safe}$, который минимально отклоняется от $\widehat{v}_{t+1}^{ML}$ и обеспечивает выполнение ограничения. [18] В простейшем варианте используется одномерная проекция по скорости

$$\widehat{v}_{t+1}^{safe} = \arg\min_{v\in[0,v_{\max}]} (v - \widehat{v}_{t+1}^{ML})^2 \quad \text{при условии} \quad \max\bigl(0, s_t + (v_t^{\ell} - v)\Delta t\bigr) \geqslant s_{t+1}^{req}. \tag{3.17}$$

Итоговый прогноз определяется правилом переключения

$$\widehat{v}_{t+1} = \begin{cases} \widehat{v}_{t+1}^{ML}, & \text{если условие (3.16) не выполняется,} \\ \widehat{v}_{t+1}^{safe}, & \text{если условие (3.16) выполняется.} \end{cases}$$

Шаг 10. Логирование срабатываний и итеративное уточнение. Фиксируются контекст $\mathbf{z}_t$, выбранные коэффициенты $\widehat{\mathbf{C}}_t$, значения s_t^{req}, исходный прогноз $\widehat{v}_{t+1}^{ML}$ и скорректированный прогноз $\widehat{v}_{t+1}^{safe}$. Накопленные события используются для уточнения сегментной калибровки и для дообучения модели, основанной на данных, на режимах, в которых она чаще нарушает ограничение.

Результатом является классическая модель с динамически корректируемыми коэффициентами C_0,C_1,C_2, которая используется как ограничение поверх

модели следования за лидером, основанной на данных. Метод обеспечивает динамическое обновление контекста классической модели следования за лидером. Это позволяет подавлять аномальные прогнозы и снижать риск появления аномальных событий в замкнутом контуре, при этом сохраняется точность модели, основанной на данных, в режимах, где её прогнозы не нарушают ограничение. [27]

3.5 Разработка гибридного метода прогнозирования микроскопических характеристик транспортного потока

В этом разделе изучаются политики продольного следования за лидером, обучаемые по данным и оцениваемые в замкнутом контуре. Рассматриваются только нейросетевые модели, гибридная схема на базе IDM и трансформероподобные архитектуры с разреженным вниманием. Анализ выполнен на шести наборах траекторий движения и использует композитную функцию качества, объединяющую точность следования, плавность и показатели безопасности. Для семейств моделей, представленных несколькими конфигурациями, метрики сведены как на уровне семейств, так и на уровне отдельных прогонов внутри каждого семейства по всем наборам данных.

Подходы к моделированию динамики следования за лидером основанных на данных сформировались как развитие микроскопических моделей, ориентированных на воспроизведение продольной реакции ведомого ТС на динамику лидера. [16] Практический интерес к таким моделям обусловлен двумя обстоятельствами. Во-первых, наблюдаемое поведение водителей в реальном потоке проявляет выраженную контекстную зависимость и неоднородность, которая трудно описывается фиксированным набором параметров. Во-вторых, в прикладных задачах, связанных с интеллектуальными транспортными системами и автоматизированным вождением, требуется не только воспроизводить среднюю реакцию, но и устойчиво работать в замкнутом контуре при изменении условий, редких режимах и возмущениях. [20] В этих условиях модели, обучаемые на данных, выступают как аппроксиматоры политики продольного управления.

В современной литературе по моделям следования за лидером обычно формулируются как отображения, которые получают на вход текущие и истори-

ческие наблюдения межмашинной дистанции, скорости ведомого и лидера и их разности, а на выходе формируют ускорение, скорость или дистанцию на следующий шаг. В зависимости от постановки, модель может быть ориентирована на одношаговый прогноз или на многократное развёртывание в замкнутом контуре, что предъявляет требования к устойчивости и предотвращению накопления ошибок. На уровне используемых семейств методов в прикладных исследованиях доминируют три направления. Первое направление включает полносвязные нейронные сети и другие статические аппроксиматоры, которые применяются как базовые ориентиры качества. Второе направление связано с рекуррентными архитектурами, которые учитывают инерционность и задержку реакции водителя и более естественно описывают динамику следования [126]. Третье направление включает методы, которые эксплуатируют длинный контекст и сложные зависимости в последовательности, в том числе через каузальные свёртки или механизмы внимания, что мотивируется стремлением улучшить переносимость и устойчивость при длительных переходных процессах [190].

Сопоставимость результатов в данной области существенно усложняется из-за различий в датасетах, критериях выделения эпизодов следования и используемых метриках. Для решения этой проблемы предложены унифицированные бенчмарки, приводящие разнородные траекторные данные к единому формату, выделяющие эпизоды следования и сопоставляющие классические модели и модели основанных на данных в едином протоколе оценивания. В частности, FollowNet задаёт согласованную процедуру формирования событий следования, сопоставляет базовые нейросетевые модели, LSTM и модели обучения с подкреплением с классическими моделями и демонстрирует, что повышение точности по ошибке дистанции не эквивалентно повышению безопасности в замкнутом контуре [161]. Наличие подобных сравнительных работ является важным аргументом в пользу рассмотрения в данной диссертации нескольких семейств моделей, поскольку в следовании за лидером наблюдается устойчивый компромисс между точностью аппроксимации, плавностью и безопасностью.

Полносвязные нейронные сети используются в задачах "лидер-последователь"в качестве минимально сложного нелинейного вычислителя, который прогнозирует следующую скорость или ускорение на основе характеристик входного состояния. Их преимущество заключается в простоте обучения и воспроизводимости. Ограничение заключается в том, что без явного механизма

запоминания они вынуждены получать информацию о динамике путем расширения функций, объединяя несколько предыдущих шагов в единый вектор.

В этом случае временная структура кодируется неявно, и качество зависит от выбора длины окна и нормализации. Тем не менее, многослойный персептрон (MLP) остается полезным эталоном, поскольку он позволяет нам отличить вклад самих функций от вклада специализированных механизмов моделирования последовательности.

Рекуррентные модели, прежде всего базирующиеся на LSTM, относятся к числу наиболее распространённых классов моделей следования за лидером, основанных на данных. Их применимость объясняется тем, что реакция водителя не является мгновенной и зависит от краткосрочной истории, включая накопленные возмущения и динамику сближения. Базовые свойства LSTM как механизма памяти закреплены в классической работе [126], а их использование в транспортном моделировании широко отражено в обзорах по применению нейросетей для моделей следования за лидером. В прикладных исследованиях LSTM применяется как самостоятельная политика или как компонент гибридного решения, где нейросетевой прогноз дополняется классическим безопасностным механизмом. В задаче планирования траектории следования за лидером предложен подход, в котором LSTM используется для генерации траектории, а безопасность усиливается посредством дополнительного механизма управления и ограничений [191]. Важным выводом таких работ является то, что рекуррентная модель способна воспроизводить временные паттерны реакции, однако без внешнего ограничения она может генерировать нежелательные режимы при развёртывании, особенно при переносе на новые условия.

Дальнейшее развитие рекуррентных подходов в моделях следования за лидером, основанных на данных, включает гибридные методы, в которых классическая модель обеспечивает физически интерпретируемую основу, а модуль нейронной сети компенсирует систематические ошибки и учитывает индивидуальные характеристики. Примером является сочетание LSTM с IDM посредством фильтрации, где классический компонент отвечает за физическую структуру, а модуль нейронной сети адаптируется к поведению водителя. Связанная с этим идея заключается в том, чтобы включить характеристики факторов, влияющих на поведение в долгосрочной и краткосрочной перспективе, и интегрировать их в классические модели. В частности, данные NGSIM показывают, что кластеризация факторов, влияющих на поведение,

на основе показателей риска и динамических индикаторов повышает точность модифицированных классических моделей, поскольку параметры отражают неоднородность поведения [192]. Эти результаты важны для данной диссертации, поскольку они демонстрируют, что улучшение качества достигается за счет явного учета контекста и неоднородности, а не просто за счет увеличения сложности аппроксиматора.

В качестве альтернативы рекуррентным механизмам памяти используются каузальные сверточные архитектуры, которые кодируют исторические зависимости с помощью временных сверток. Такие подходы называют TCN и они привлекли к себе внимание благодаря надежной реализации причинно-следственной связи и способности расширять поле восприятия. При прогнозировании микроскопических траекторий и связанных с ними режимов движения используются подходы, основанные на свертке, в сочетании с рекуррентными нейронными сетями. Это подчеркивает ценность извлечения локальных закономерностей и последующей интеграции их во временную динамику.

Гибридные модели CNN-LSTM следования за лидером были представлены в ряде исследований и демонстрируют повышенную точность по сравнению с базовыми моделями LSTM и классическими моделями, при условии, что они должным образом настроены и данные правильно предварительно обработаны. В то же время такие архитектуры могут быть чувствительны к выбору гиперпараметров и характеристик набора данных, что требует оценки по нескольким наборам данных и в условиях замкнутого цикла.

Отдельный класс исследований посвящён переносу и обобщаемости моделей. В следовании за лидером это связано с тем, что значительная доля работ обучается на NGSIM, который обладает известными проблемами качества траекторий и ограниченной репрезентативностью по географии и условиям [193]. Следствием является рост интереса к унифицированным сравнениям на нескольких датасетах и к методам, которые улучшают устойчивость к шуму данных и к сдвигу распределения.

В этом контексте развиваются гибридные физически информированные подходы, где нейросеть обучается с учётом физической структуры и штрафов, связанных с динамикой, что повышает устойчивость к неполным данным и шумам. Для моделей следования за лидером, основанных на данных, предложены физически информированные архитектуры, сочетающие нейросетевой модуль

и физический вычислительный граф, где функция потерь включает согласование ускорений и структурные ограничения [171].

Данные работы показывают целесообразность использования в данной диссертации механизма ограничения аномалий на базе классической модели с динамической коррекцией параметров, представленной выше.

На горизонтах, где требуется учитывать длинный контекст, в следовании за лидером начинают применяться модули внимания и трансформероподобные архитектуры. Мотивация состоит в том, что реакция ведомого может зависеть от длительной предыстории и от структурных изменений режима движения, включая медленные волны торможений и разгонов, особенно на протяжённых участках и при нестационарной плотности потока. В рамках следования за лидером предложены трансформерные архитектуры для длительного прогнозирования траектории ведомого, где показано преимущество по точности относительно базовых рекуррентных и классических моделей при адекватном обучении [190]. Данный класс работ является прямым обоснованием включения в сравнение семейства Transformer.

Для задач, где входное окно существенно увеличивается, важным становится вопрос вычислительной эффективности. Архитектура Informer предложена как модификация трансформера для длинных последовательностей на основе разреженного внимания и процедур уменьшения длины последовательности [131]. Несмотря на то, что Informer представлен как метод прогнозирования временных рядов, архитектурный принцип снижения вычислительных затрат и ориентированность на длительные зависимости делают его применимым в постановке следования за лидером при соответствующей адаптации входных признаков и целевой переменной.

В данной диссертации Informer включается в сравнение не как универсальная модель временных рядов, а как представитель семейства моделей с механизмами внимания, потенциально более устойчивый на длинных окнах при тех же требованиях каузальности. Таким образом, выбор семейств моделей MLP, LSTM, TCN и Informer в данной диссертации имеет содержательное обоснование внутри предметной области следования за лидером.

MLP задаёт воспроизводимую нелинейную базовую линию без явной памяти. LSTM представляет основной класс моделей, которые используют временную структуру реакции ведомого. TCN представляет альтернативный каузальный механизм учёта истории с иными свойствами устойчивости. Informer

представляет вычислительно ориентированную архитектуру с модифицированными блоками внимания, предназначенную для длинных окон.

Сопоставление указанных семейств в единой постановке и на нескольких наборах траекторий создаёт эмпирическую основу для выбора базовой модели, которая далее используется для разработки нового метода прогнозирования микроскопических характеристик транспортного потока.

Динамическая постановка и метрики

Для исключения неоднозначности введём используемые обозначения. Пусть t - номер дискретного временного шага, Δt - шаг дискретизации, s_t - фактическая межмашинная дистанция между лидером и ведомым ТС, м; v_t^f - скорость ведомого ТС, м/с; v_t^ℓ - скорость лидирующего ТС, м/с; $\Delta v_t = v_t^f - v_t^\ell$ - относительная скорость, м/с; $\hat{s}_t$ - прогнозируемая межмашинная дистанция, м; $\hat{a}_t$ - ускорение, применяемое моделью в замкнутом контуре, м/с^2; a_t^{gt} - оценка фактического ускорения по данным, м/с^2; $\widehat{TTC}_t$ - прогнозируемое время до столкновения, с; τ - минимально допустимый порог времени до столкновения, с; $(x)_+ = \max(x,0)$.

Фактическое ускорение, используемое при обучении как эталонное значение, оценивается по измеренной скорости ведомого :

$$a_t^{gt} = \frac{v_{t+1}^{f,obs} - v_t^{f,obs}}{\Delta t},$$

где $v_t^{f,obs}$ - измеренная скорость ведомого в момент времени t.

Модельный замкнутый контур записывается как

$$v_{t+1}^f = \max\bigl(0, v_t^f + a_t \Delta t\bigr), \qquad s_{t+1} = \max\bigl(0, s_t + (v_t^\ell - v_t^f)\Delta t\bigr). \tag{3.18}$$

Входной вектор признаков строится по наблюдаемому состоянию

$$x_t = \bigl[s_t,\, v_t^f,\, v_t^\ell,\, \Delta v_t,\, h_t,\, \mathrm{TTC}_t\bigr], \quad \Delta v_t = v_t^f - v_t^\ell, \quad h_t = \frac{s_t}{\max(v_t^f, \varepsilon)}, \tag{3.19}$$

где $\varepsilon > 0$ предотвращает деление на нуль. Время до столкновения определяется как

$$\mathrm{TTC}_t = \begin{cases} \frac{s_t}{\Delta v_t}, & \Delta v_t > 0, \\ +\infty, & \Delta v_t \leqslant 0, \end{cases} \tag{3.20}$$

и далее клипируется сверху для численной устойчивости визуализаций и агрегатов.

Гибридная политика Residual IDM использует суперпозицию физической компоненты и нейросетевой поправки. Пусть $p = (a_{\max}, b_{\text{comf}}, v_0, T, s_0, \delta)$ параметры IDM. Желаемая дистанция имеет вид

$$s^\star = s_0 + v^f T + \max\left(0, \frac{v^f \Delta v}{2\sqrt{a_{\max} b_{\text{comf}}}}\right), \tag{3.21}$$

ускорение по IDM задаётся выражением

$$a^{\text{IDM}} = a_{\max}\left(1 - \left(\frac{v^f}{v_0}\right)^{\delta} - \left(\frac{s^\star}{\max(s_t, \varepsilon)}\right)^2\right), \tag{3.22}$$

а итоговое действие определяется формулой $a_t = a^{\text{IDM}}(x_t, p) + f_\theta(x_t)$, где f_θ малогабаритная нейросеть.

Для обеспечения размерностной корректности композитной функции потерь каждое слагаемое нормируется на соответствующий масштаб:

$$\begin{aligned}\mathcal{L} = {} & w_s \mathbb{E}\left[\left(\frac{\hat{s}_t - s_t}{s_{\text{scale}}}\right)^2\right] \\ & + w_j \mathbb{E}\left[\frac{1}{j_{\text{scale}}}\left|\frac{\hat{a}_t - \hat{a}_{t-1}}{\Delta t}\right|\right] \\ & + w_\tau \mathbb{E}\left[\left(\frac{(\tau - \widehat{TTC}_t)_+}{\tau}\right)^2\right] \\ & + w_a \mathbb{E}\left[\left(\frac{\hat{a}_t - a_t^{gt}}{a_{\text{scale}}}\right)^2\right].\end{aligned} \tag{3.23}$$

где s_{scale} - характерный масштаб дистанции, м; a_{scale} - характерный масштаб ускорения, м/с^2; j_{scale} - характерный масштаб рывка, м/с^3; w_s,w_j,w_τ,w_a - безразмерные весовые коэффициенты.

В такой записи все слагаемые функции потерь являются безразмерными: первое слагаемое штрафует ошибку прогноза дистанции, второе - резкие изменения ускорения, третье - нарушение безопасного порога по времени до столкновения, четвёртое - отклонение применённого ускорения от оценки, полученной по измеренной скорости.

Оценка качества моделей выполняется на траекториях, получаемых при развёртывании политики в замкнутом контуре. Пусть для каждой траектории рассматриваются дискретные моменты времени $t = 1, \ldots, T$ с шагом Δt.

Обозначим истинную (наблюдаемую) межмашинную дистанцию как s_t, смоделированную дистанцию как $\hat{s}_t$, скорость ведомого как v_t^f, скорость лидера как v_t^ℓ, относительную скорость $\Delta v_t = v_t^f - v_t^\ell$, ускорение ведомого как $\hat{a}_t$. Для агрегирования по множеству траекторий используется усреднение по всем временным индексам и всем эпизодам тестового набора. Ниже приведены метрики точности следования, безопасности и плавности.

Метрики точности по дистанции

Среднеквадратичная ошибка по дистанции определяется как

$$\text{MSE_spacing} = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{T_i}\sum_{t=1}^{T_i}\left(\hat{s}_t^{(i)} - s_t^{(i)}\right)^2, \tag{3.24}$$

где i индексы эпизодов, T_i длина i-го эпизода, N число эпизодов.

Средняя абсолютная ошибка по дистанции определяется как

$$\text{MAE_spacing} = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{T_i}\sum_{t=1}^{T_i}\left|\hat{s}_t^{(i)} - s_t^{(i)}\right|. \tag{3.25}$$

Корень из среднеквадратичной ошибки по дистанции определяется как

$$\text{RMSE_spacing} = \sqrt{\text{MSE_spacing}}. \tag{3.26}$$

Метрика столкновений

Доля столкновений определяется по порогу $\varepsilon_c > 0$, который задаёт событие столкновения или наложения траекторий по дистанции. Для каждого шага вводится индикатор

$$\mathbb{I}_t^{(i)} = \mathbb{I}\left[\hat{s}_t^{(i)} \leqslant \varepsilon_c\right].$$

Тогда метрика частоты столкновений в промилле задаётся как

$$\text{Collision_rate_per_mille} = 1000 \cdot \frac{\sum_{i=1}^{N}\sum_{t=1}^{T_i}\mathbb{I}_t^{(i)}}{\sum_{i=1}^{N} T_i}. \tag{3.27}$$

Метрика плавности управления

Плавность оценивается средним модулем рывка, где рывок аппроксимируется конечной разностью ускорения:

$$\text{Jerk_mean_abs} = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{T_i-1}\sum_{t=2}^{T_i}\left|\frac{\hat{a}_t^{(i)}-\hat{a}_{t-1}^{(i)}}{\Delta t}\right|. \tag{3.28}$$

При необходимости рывок клипируется по верхней границе для подавления влияния единичных численных выбросов, что отдельно фиксируется в протоколе расчёта.

Метрики TTC (time-to-collision)

Время до столкновения определяется как

$$\text{TTC}_t = \begin{cases} \frac{\hat{s}_t}{\Delta v_t}, & \Delta v_t > 0, \\ +\infty, & \Delta v_t \leqslant 0. \end{cases} \tag{3.29}$$

Для устойчивого агрегирования вводится клипирование сверху по порогу 30 секунд:

$$\text{TTC}_t^{cap30} = \min(\text{TTC}_t, 30).$$

Минимальный TTC с клипированием и последующим усреднением по эпизодам определяется как

$$\text{Min_TTC_mean_cap30} = \frac{1}{N}\sum_{i=1}^{N}\min_{t=1,\dots,T_i}\text{TTC}_t^{cap30,(i)}. \tag{3.30}$$

Среднее TTC с клипированием определяется как

$$\text{TTC_mean_cap30} = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{T_i}\sum_{t=1}^{T_i}\text{TTC}_t^{cap30,(i)}. \tag{3.31}$$

Квантильная метрика определяется через пятый процентиль значений TTC_t^{cap30} внутри каждого эпизода с последующим усреднением по эпизодам:

$$\text{TTC_p05_mean} = \frac{1}{N}\sum_{i=1}^{N}Q_{0.05}\Big(\{\text{TTC}_t^{cap30,(i)}\}_{t=1}^{T_i}\Big), \tag{3.32}$$

где $Q_{0.05}(\cdot)$ обозначает 5-й процентиль.

Метрика по TTC при пороге 1.5 секунды определяется как доля временных шагов, на которых TTC_t^{cap30} меньше порога:

$$\mathrm{TTC_risk_1.5} = \frac{\sum_{i=1}^{N}\sum_{t=1}^{T_i}\mathbb{I}\big[\mathrm{TTC}_t^{cap30,(i)} < 1.5\big]}{\sum_{i=1}^{N} T_i}. \tag{3.33}$$

Метрики PET (post-encroachment time)

Метрика PET используется как дополнительный показатель риска конфликтного сближения в продольной постановке. В рамках настоящего протокола PET вычисляется по заданной функции $\mathrm{PET}_t = \mathrm{PET}(s_t, v_t^f, v_t^\ell)$, фиксированной в программной реализации расчёта метрик. Для устойчивого агрегирования вводится клипирование сверху по порогу 30 секунд:

$$\mathrm{PET}_t^{cap30} = \min(\mathrm{PET}_t, 30).$$

Далее рассчитываются агрегаты, аналогичные TTC.

Минимальный PET с клипированием и последующим усреднением по эпизодам определяется как

$$\mathrm{PET_min_cap30} = \frac{1}{N}\sum_{i=1}^{N}\min_{t=1,\dots,T_i}\mathrm{PET}_t^{cap30,(i)}. \tag{3.34}$$

Пятый процентиль PET с клипированием и последующим усреднением по эпизодам определяется как

$$\mathrm{PET_p05} = \frac{1}{N}\sum_{i=1}^{N} Q_{0.05}\Big(\{\mathrm{PET}_t^{cap30,(i)}\}_{t=1}^{T_i}\Big). \tag{3.35}$$

Метрика PET при пороге 1.5 секунды определяется как доля временных шагов, на которых PET_t^{cap30} меньше порога:

$$\mathrm{PET_risk_1.5} = \frac{\sum_{i=1}^{N}\sum_{t=1}^{T_i}\mathbb{I}\big[\mathrm{PET}_t^{cap30,(i)} < 1.5\big]}{\sum_{i=1}^{N} T_i}. \tag{3.36}$$

Средний PET с клипированием определяется как

$$\mathrm{PET_mean_cap30} = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{T_i}\sum_{t=1}^{T_i}\mathrm{PET}_t^{cap30,(i)}. \tag{3.37}$$

Метрики PSD (preferred safety distance)

Метрика PSD применяется как относительная оценка выполнения предпочтительного безопасного интервала. В рамках настоящего протокола PSD вычисляется по заданной функции $\mathrm{PSD}_t = \mathrm{PSD}(s_t, v_t^f, \mathbf{z}_t)$, фиксированной в программной реализации расчёта метрик. Для устойчивости агрегирования вводится клипирование сверху по порогу 3:

$$\mathrm{PSD}_t^{cap3} = \min(\mathrm{PSD}_t, 3).$$

Далее рассчитываются агрегаты.

Минимальное значение PSD с клипированием определяется как

$$\mathrm{PSD_min_cap3} = \frac{1}{N}\sum_{i=1}^{N} \min_{t=1,\dots,T_i} \mathrm{PSD}_t^{cap3,(i)}. \tag{3.38}$$

Пятый процентиль PSD с клипированием определяется как

$$\mathrm{PSD_p05} = \frac{1}{N}\sum_{i=1}^{N} Q_{0.05}\Big(\{\mathrm{PSD}_t^{cap3,(i)}\}_{t=1}^{T_i}\Big). \tag{3.39}$$

Риск-метрика PSD определяется как доля временных шагов, на которых PSD_t^{cap3} меньше 1:

$$\mathrm{PSD_risk_lt1} = \frac{\sum_{i=1}^{N}\sum_{t=1}^{T_i} \mathbb{I}\big[\mathrm{PSD}_t^{cap3,(i)} < 1\big]}{\sum_{i=1}^{N} T_i}. \tag{3.40}$$

Среднее значение PSD с клипированием определяется как

$$\mathrm{PSD_mean_cap3} = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{T_i}\sum_{t=1}^{T_i} \mathrm{PSD}_t^{cap3,(i)}. \tag{3.41}$$

Сравнительная оценка моделей следования за лидером, основанных на данных, в условиях замкнутого контура

Для каждого семейства моделей фиксируются используемая архитектура, набор конфигураций, и приводится интерпретация наблюдаемых эффектов.

MLP и Residual IDM

В качестве базовой аппроксимации без явного механизма памяти рассматривается многослойный перцептрон (MLP), который реализует отображение из текущего наблюдаемого состояния и контекстных признаков в управляющее воздействие либо в прогнозируемую величину на следующий шаг.

Конфигурации MLP задаются числом скрытых слоёв и размерностью скрытого представления, а также настройками регуляризации и функции потерь в соответствии с общей постановкой. Для корректного сопоставления с последовательностными моделями входное окно фиксированной длины подаётся в модель в виде вектора после операции time-delay embedding, чтобы MLP получала одинаковый информационный объём по истории. Результаты по всем датасетам и ключевым метрикам приведены ниже в агрегированной таблице.

Интерпретация результатов показывает, что MLP существенно уступает остальным нейросетевым моделям по точности следования и демонстрирует повышенную частоту небезопасных режимов на части данных. Данный результат согласуется с тем, что временная динамика реакции ведомого в режимах следования не может быть устойчиво восстановлена из статического отображения без специализированного механизма памяти. В рамках дальнейшего исследования MLP используется как нижняя опорная граница качества.

Residual IDM рассматривается как гибридная модель, в которой действие формируется суммой физического компонента и нейросетевой поправки. В рамках общего сравнения Residual IDM используется как интерпретируемая физически информированная базовая линия и как кандидат на устойчивое поведение в замкнутом контуре. Конфигурации Residual IDM в данном исследовании фиксируются на уровне семейства, поскольку основная вариативность задаётся параметрами физической части и способом обучения поправки. Результаты Residual IDM по всем датасетам приведены ниже в агрегированной таблице. Интерпретация показывает, что гибридный подход обеспечивает устойчивое поведение и, как правило, избегает столкновений, однако уступает нейросетевым моделям по точности на датасетах с выраженной неоднородностью режима движения. Данный результат соответствует ожидаемой роли физического базиса как стабилизатора и ограничителя.

Семейство LSTM

Семейство LSTM рассматривается как основной класс моделей, которые реализуют параметризацию зависимости реакции ведомого от истории движения. Во всех конфигурациях используется фиксированное окно длины seq_len, а предсказание выполняется по состоянию последовательности. Внутрисемейственные различия в данном исследовании задаются не столько изменением архитектуры, сколько изменением весов штрафов в композитной функции потерь, что позволяет контролировать компромисс между точностью, плавностью и показателями безопасности.

Внутрисемейственные прогоны обозначены как LSTM-B, LSTM-00, LSTM-02, LSTM-20, LSTM-22 и LSTM-S. Эти варианты различаются весами штрафов, используемых в обучении, что отражено в тексте постановки и обеспечивает сопоставимость по данным и по протоколу оценки. Результаты по MSE_s и по рывку для прогонов приведены в таблицах 17 и 18, а столкновений приведены в таблице 19.

Таблица 17 — LSTM: значения MSE_s по датасетам и прогонам

Dataset	LSTM-B	LSTM-00	LSTM-02	LSTM-20	LSTM-22	LSTM-S
highd	25.69	36.96	31.48	22.23	28.19	23.65
ngsim	65.12	22.04	81.41	21.08	43.18	95.72
das1	283.16	284.67	428.96	114.74	336.87	455.14
das2	100.39	123.06	135.25	56.31	123.12	140.57
waymo	32.90	18.44	35.67	15.22	19.89	30.18
lyft	100.51	92.30	121.56	84.03	114.27	109.00
Собранный датасет	20.25	16.65	21.06	14.32	18.78	20.51

Таблица 18 — LSTM: средний модуль рывка по датасетам и прогонам

Dataset	LSTM-B	LSTM-00	LSTM-02	LSTM-20	LSTM-22	LSTM-S
highd	1.40	2.88	0.51	1.08	0.59	4.78
ngsim	0.90	12.28	0.67	3.84	0.63	2.32
das1	0.95	1.04	0.56	4.34	0.76	1.82
das2	0.75	1.06	0.36	3.65	0.48	1.70
waymo	1.60	5.81	0.99	3.76	0.38	5.46
lyft	1.31	1.56	1.12	3.72	1.16	1.98
Собранный датасет	0.68	1.04	0.43	3.64	0.25	1.32

Таблица 19 — LSTM: доля столкновений (в промилле) по датасетам и прогонам

Dataset	LSTM-B	LSTM-00	LSTM-02	LSTM-20	LSTM-22	LSTM-S
highd	14.65	28.51	6.32	11.66	7.07	46.29
ngsim	9.97	10.49	7.82	37.49	7.45	23.27
das1	10.44	11.28	6.79	42.17	8.66	18.59
das2	8.57	11.47	4.92	35.71	6.04	17.46
waymo	16.53	55.93	10.82	36.74	5.11	52.66
lyft	13.81	16.15	12.03	36.37	12.41	20.08
Собранный датасет	7.91	11.28	5.57	35.62	3.89	13.91

Представленные здесь результаты указывают на устойчивый компромисс внутри семейства. Конфигурации с более высоким коэффициентом штрафа за безопасность улучшают показатели, ориентированные на TTC, но могут увеличить ошибку опережения или расстояния в зависимости от комбинации весов. Конфигурации с более сильным штрафом за опережение приводят к более плавной динамике, но могут демонстрировать повышенную ошибку во время резких переходов. При сравнении всех семейств LSTM в совокупности демонстрирует наиболее стабильный профиль производительности и становится основным кандидатом на дальнейшую модификацию.

Семейство TCN

Семейство TCN рассматривается как причинно-следственная сверточная альтернатива рекуррентным моделям. Конфигурации внутри семейства различаются по глубине сети и параметрам сверточного ядра, которые изменяют эффективное рецептивное поле и динамику реакции замкнутой системы. Эти конфигурации обозначаются как TCN-4k3, TCN-3k3 и TCN-5k3. Результаты по MSE_s, рывку, TTC и столкновениям приведены в таблицах 20-21.

Таблица 20 — TCN: значения MSE_s по датасетам и прогонам

Dataset	TCN-4k3	TCN-3k3	TCN-5k3
highd	43.19	73.10	67.35
ngsim	254.19	304.86	257.56
das1	1164.75	891.41	982.79
das2	284.34	292.93	213.94
waymo	195.42	229.25	248.58
lyft	405.06	427.36	382.33
Собранный датасет	33.81	40.93	36.01

Таблица 21 — TCN: доля столкновений (в промилле) по датасетам и прогонам

Dataset	TCN-4k3	TCN-3k3	TCN-5k3
highd	14.80	70.63	8.69
ngsim	9.13	47.95	7.31
das1	92.85	20.96	25.03
das2	8.03	39.01	3.89
waymo	8.47	75.22	24.61
lyft	12.83	13.44	15.45
Собранный датасет	5.42	18.07	4.11

Представленные результаты показывают, что производительность TCN в значительной степени зависит от выбора глубины и ядра. На некоторых наборах данных TCN достигает конкурентоспособной точности. Однако определенные конфигурации сопровождаются увеличением дрожания или ухудшением метрик, ориентированных на TTC, что указывает на необходимость тщательной регуляризации и выбора конфигурации для достижения желаемого профиля «точность-гладкость-безопасность».

Семейство Informer

Семейство Informer рассматривается как класс методов с механизмом внимания, разработанным для последовательностей с большими окнами и сниженными вычислительными затратами на механизм внимания. В исследовании

используются две конфигурации, обозначенные как INF-50 и INF-100D. Разница заключается в размере окна и наличии дистилляции. Результаты по MSE_s, рывку, TTC и столкновениям приведены в таблицах 22-24.

Таблица 22 — Informer: значения MSE_s по датасетам и прогонам

Dataset	INF-50	INF-100D
highd	25.77	23.23
ngsim	20.59	11.19
das1	29.10	28.26
das2	22.06	18.40
waymo	12.17	10.62
lyft	33.23	31.08
Собранный датасет	14.73	12.27

В таблице показано влияние длины окна и степени дистилляции на точность отслеживания расстояния в замкнутой системе.

Таблица 23 — Informer: средний модуль рывка по датасетам и прогонам

Dataset	INF-50	INF-100D
highd	1.21	0.95
ngsim	3.90	2.37
das1	3.17	1.77
das2	2.36	1.70
waymo	4.86	3.73
lyft	3.42	1.20
Собранный датасет	1.03	0.90

Таблица 24 — Informer: доля столкновений по датасетам и прогонам

Dataset	INF-50	INF-100D
highd	12.70	5.31
ngsim	9.15	7.67
das1	68.40	28.61
das2	45.38	26.62
waymo	15.41	10.32
lyft	35.51	11.41
Собранный датасет	15.62	5.38

Интерпретация результатов показывает, что конфигурация с длинным окном и дистилляцией в большинстве случаев демонстрирует преимущество с точки зрения точности и плавности, тогда как конфигурация с коротким окном

может давать более консервативный профиль, основанный на метриках, ориентированных на TTC. Это согласуется с тем фактом, что увеличение контекста улучшает восстановление режима движения, но может изменить реакцию на локальные возмущения.

Агрегирование по семействам и комплексный вывод

После анализа семейств моделей и внутрисемейных запусков представлены сводные таблицы ключевых метрик, используемых в окончательном сравнении семейств моделей. В таблицах выделены наилучшие значения в соответствии с направлением оптимизации метрик. Минимизируются MSE $_s$, средний рывок и частота столкновений. Максимизируются метрики, характеризующие временной запас до столкновения, при условии корректности единиц измерения и ограничения.

Таблица 25 — Агрегированные значения MSE_s по датасетам и семействам моделей

Dataset	Residual IDM	MLP	LSTM agg	TCN agg	Informer agg
highd	113.29	328.14	28.03	61.21	24.50
ngsim	252.99	1355.69	54.76	272.20	15.89
das1	519.02	853.14	317.26	1012.98	28.68
das2	363.16	445.42	113.12	263.74	20.23
waymo	182.85	465.69	25.38	224.42	11.40
lyft	428.49	580.99	103.61	404.92	32.16
Собранный датасет	105.21	308.72	18.60	36.92	13.50

Таблица является основным сравнением точности следования по дистанции между семействами моделей на каждом датасете и в среднем.

Таблица 26 — Агрегированные значения среднего модуля рывка по датасетам и семействам моделей

Dataset	LSTM agg	TCN agg	Informer agg
highd	1.87	7.86	1.08
ngsim	3.44	17.24	3.14
das1	1.58	4.50	2.47
das2	1.33	3.91	2.03
waymo	3.00	13.29	4.30
lyft	1.81	9.69	2.31
Собранный датасет	1.23	3.54	0.97

Сопоставление конфигураций внутри семейств позволяет выделить не только лучшее семейство в среднем, но и конкретные реализации, которые целесообразно рассматривать как базовые кандидаты для последующей модификацию. В рамках проведённых экспериментов наибольший интерес представляют конфигурации `LSTM-20` и `INF-100D`, поскольку они демонстрируют устойчиво высокий профиль качества при развёртывании в замкнутом контуре и при этом допускают конструктивное расширение архитектуры без потери воспроизводимости постановки.

Конфигурация `LSTM-20` является перспективным кандидатом на роль базовой модели управления. Очевидно, что `LSTM-20` обеспечивает устойчивую среднюю ошибку расстояния среди рассмотренных запусков LSTM, сохраняя при этом стабильную производительность на разных наборах данных — то есть без ухудшения на отдельных корпусах. `LSTM-20` не является минимизатором рывков, но обеспечивает контролируемую плавность со значительно более высокой точностью измерения расстояния. Данная конфигурация подходит для применений, где необходимо обеспечить точное отслеживание цели при сохранении приемлемого уровня точности управления. Нулевые значения столкновений во всех прогонах LSTM подтверждают, что конфигурация не индуцирует аварийные режимы в рамках принятого протокола роллаута и порога ε_c. Различия внутри семейства проявляются прежде всего в компромиссе «точность-плавность-безопасность», задаваемом весами в функции потерь.

Конфигурация `INF-100D` выделяется как кандидат на базовую модель длинного контекста. `INF-100D` демонстрирует существенно более низкие значения MSE_s на большинстве датасетов по сравнению с `INF-50`. Это также обеспечивает более благоприятный профиль переходной характеристики, что

указывает на способность модели создавать более плавную реакцию при использовании расширенного окна и механизма дистилляции. В такой виде `INF-100D` может использоваться в качестве точного предиктора долгосрочной динамики, в то время как безопасность и консервативность обеспечиваются отдельным контуром ограничений, позволяющим распределять функции точности и безопасности между различными компонентами.

Решение использовать архитектуру LSTM-Informer обусловлено тем, что экспериментальные результаты показывают взаимодополняемость этих семейств. LSTM в конфигурации `LSTM-20` обеспечивает стабильную работу в коротких контекстах и формирует динамическое внутреннее состояние, которое точно соответствует характеру реакции водителя и позволяет интерпретировать данные с точки зрения памяти и задержки реакции. В конфигурации `INF-100D` модуль Informer обеспечивает более эффективную обработку длинных окон и демонстрирует высокую точность отслеживания расстояний при сохранении приемлемой плавности, что особенно важно в участках с длительными переходными процессами. Таким образом, сочетание LSTM и Informer позволяет одновременно учитывать инерционные свойства отклика и использовать длинный контекст без чрезмерного увеличения вычислительных затрат, что критически важно для замкнутых контуров управления.

Дополнительным аргументом в пользу Informer является его интерпретируемость на уровне механизма внимания. В отличие от полностью моделей черного ящика, архитектуры с блоком внимания позволяют анализировать распределение внимания по временным позициям во входной последовательности и, таким образом, определять, какие сегменты истории движения оказывают наибольшее влияние на предсказание. В контексте задачи следования за лидером это позволяет проверить, соответствует ли поведение модели физической реальности, определить значимые интервалы ускорения/замедления, а также диагностировать причины снижения производительности в конкретных сценариях.

Хотя интерпретация блока внимания не является прямой физической параметризацией и не заменяет интерпретируемые коэффициенты классической модели, она обеспечивает важную прозрачность при анализе долгосрочных прогнозов и при построении гибридных схем.

Проведённое сравнение семейств моделей и анализ лучших конфигураций показали, что для задачи следования за лидером в замкнутом контуре требуется

одновременно удовлетворять трём требованиям: высокая точность воспроизведения дистанции, устойчивость к накоплению ошибок при развёртывании, контроль аномальных и небезопасных режимов (аномальные ускорения, разрыв дистанции, столкновения). Результаты по семействам и прогонным конфигурациям указывают, что ни одна из чисто нейросетевых архитектур не гарантирует выполнение всех требований без дополнительных механизмов стабилизации.

На основании изложенного далее предлагается метод, в котором предобученная модель формирует признаки и представления для дальнейшего предсказания в архитектуре LSTM-Informer, а аномальные события в замкнутом контуре подавляются внешним ограничителем на базе классической модели следования за лидером с динамической коррекцией коэффициентов C_0, C_1, C_2 в постановке SSSR-Micro. Такая конструкция обеспечивает разделение задач, где модуль основанный на данных отвечает за точность воспроизведения динамики, а классический модуль обеспечивает формализуемые ограничения безопасности и подавление аномальных режимов при сдвиге распределения и накоплении ошибок в роллаутах.

На основании полученных результатов в качестве основы выбираются следующие блоки.

1. **Предобученный Bi-LSTM энкодер (каузальные признаки).** Bi-LSTM обеспечивает более информативное представление временного окна за счёт обучения на последовательности как в прямом, так и в обратном направлении. Однако в онлайн-прогнозе требуется каузальность. Поэтому в методе используется извлечение forward-ветви из предобученной Bi-LSTM. Так же возможно использовать замороженный экстрактор признаков с последующей каузализацией на уровне окна. Данный блок нужен для повышения переносимости и устойчивости представления состояния без существенного роста вычислительной сложности на инференсе. BiLSTM отвечает за определение зависимостей как в прямом, так и в обратном направлении. Каждое значение на временной шкале обрабатываются с двух сторон, из прошлого в будущее и наоборот, а результаты объединяются. В задачах моделирования может быть важно знать не только, что происходило до текущего момента времени, но и что будет в будущем для более точного понимания контекста.

2. **Гибридная архитектура LSTM-Informer.** LSTM-компонента стабильно моделирует инерционность и задержку реакции (локальная динамика), а Informer-компонента эффективно обрабатывает длинные зависимости, возникающие на протяжённых переходных процессах. Informer выбран вместо полного Transformer как более подходящий для длинного окна по вычислительной сложности за счёт разреженного внимания (ProbSparse) и механизма дистилляции.
3. **Механизм интерпретации и диагностики устойчивости.** В реализации включён режим интерпритации, который позволяет анализировать Informer-внимание (какие временные позиции входа оказываются наиболее значимыми) и диагностировать поведение модели. Дополнительно предлагается производить расчёт оценки Липшицевой нормы (как индикатора чувствительности к возмущениям входа), что важно для понимания причин резких управляющих воздействий и роста рывка.
4. **Коррекция аномальных выходов на базе классической модели.** Даже при высокой точности по MSE возможны редкие аномальные выходы нейросети (слишком сильное ускорение/торможение и т.д.), которые при замкнутом развёртывании приводят к накоплению ошибки и ухудшению безопасности. [19] Для предотвращения таких событий метод дополняется внешним корректирующим контуром, основанным на модели следования за лидером SSSR-Micro с динамической коррекцией коэффициентов C_0, C_1, C_2, разработанной ранее в диссертации.

Таким образом, предлагаемый метод строится как гибридная композиция. Нейросетевой модуль отвечает за точность и извлечение зависимостей в данных, а классический модуль с динамической коррекцией коэффициентов обеспечивает подавление аномальных режимов и формализуемые ограничения в замкнутом контуре.

Формальное описание предлагаемого метода

Используется дискретизация с шагом Δt. Обозначим наблюдаемое состояние:

s_t - дистанция, v_t^f - скорость ведомого, v_t^ℓ - скорость лидера, $\Delta v_t = v_t^f - v_t^\ell$.

Вектор признаков на шаге t определяется как

$$x_t = \left[s_t,\, v_t^f,\, v_t^\ell,\, \Delta v_t,\, h_t,\, \mathrm{TTC}_t^{clip}\right], \qquad h_t = \frac{s_t}{\max(v_t^f, \varepsilon)}, \tag{3.42}$$

где $\varepsilon > 0$ предотвращает деление на ноль, а TTC^{clip} соответствует реализации (клипирование до 10 секунд во входных каналах).

Шаг 1. Формирование окна наблюдений. Для каждого шага формируется каузальное окно длины L:

$$X_t = [x_{t-L+1}, \ldots, x_t] \in \mathbb{R}^{L \times F}, \qquad F = 6.$$

Шаг 2. Извлечение каузальных признаков предобученной Bi-LSTM. Предобученная Bi-LSTM обучается офлайн на последовательностях (в двунаправленном режиме). Для использования в онлайн-контуре выполняется каузализация одним из двух способов:

- **Вариант 1:** извлекается forward-ветвь Bi-LSTM, образуя однонаправленный энкодер $\Phi_{\rightarrow}$, и вычисляются признаки

$$H_t^{\mathrm{causal}} = \Phi_{\rightarrow}(X_t) \in \mathbb{R}^{L \times D_s}. \tag{3.43}$$

- **Вариант 2 (для диагностики):** используется замороженный `BiLSTMFeatureExtractor` с выходом $\mathbb{R}^{L \times 2D_s}$. В этом случае требуется дополнительная каузализация, поскольку двунаправленная обработка использует будущие элементы внутри окна.

Шаг 3. Прогноз нейросетевого действия гибридной моделью LSTM-Informer. Основная модель π_θ реализует два параллельных представления окна X_t:

1. **LSTM-ветвь** (или внешняя $\Phi_{\rightarrow}$), формирующая $H_t^{\text{seq}} \in \mathbb{R}^{L\times D_s}$;
2. **Informer-ветвь**, формирующая $H_t^{\text{inf}} \in \mathbb{R}^{L\times D}$ за счёт каузального разреженного внимания ProbSparse и дистилляции по времени.

Слияние ветвей выполняется через `fusionconcat` или `fusionadd`:

$$H_t^{\text{fus}} = \begin{cases} \big[H_t^{\text{seq}};\ H_t^{\text{inf}}\big], & \texttt{concat}, \\ A(H_t^{\text{seq}}) + H_t^{\text{inf}}, & \texttt{add}, \end{cases} \tag{3.44}$$

где $A(\cdot)$ - линейное выравнивание размерностей.

Результатом является прогноз ускорения:

$$\hat{a}_t^{\text{nn}} = \pi_\theta(X_t, H_t^{\text{causal}}), \tag{3.45}$$

где в реализации прогнозируется последовательность $\hat{a}_{t-L+1:t}$, но для замкнутого контура используется текущее значение $\hat{a}_t$.

Шаг 4. Детекция аномального выхода нейросети. Вводится предикат аномальности $\mathcal{A}(\cdot)$, который активируется при нарушении инвариантов безопасного движения или при выходе за допустимые диапазоны. В практической реализации это включает:

$$\mathcal{A}_t = \mathbb{I}\big[s_t < s_{\min}\big] \ \vee\ \mathbb{I}\big[\hat{a}_t^{\text{nn}} \notin [a_{\min}, a_{\max}]\big] \ \vee\ \mathbb{I}\big[\widehat{\text{TTC}}_t < \tau\big],$$

где $\widehat{\text{TTC}}_t$ вычисляется по прогнозной динамике/текущему состоянию (в зависимости от выбранного протокола). Конкретный набор условий фиксируется в программной реализации эксперимента.

Шаг 5. Коррекция аномального выхода на базе модели SSSR-Micro с динамической коррекцией C_0,C_1,C_2. Действие нейросети корректируется с использованием классической модели безопасной дистанции SSSR-Micro, в которой вводится динамический габарит:

$$s_{\text{req}}^{\text{BG}}(v_t^f) = C_0 + C_1 v_t^f + C_2 (v_t^f)^2, \tag{3.46}$$

а коэффициенты C_0,C_1,C_2 *адаптируются во времени* в соответствии с разработанным ранее методом (раздел с динамической коррекцией коэффициентов).

Коррекция реализуется как проекция нейросетевого действия на допустимое множество $\mathcal{U}_t$, заданное условиями безопасности (требованием $s_{t+1} \geqslant$

$s_{\text{req}}^{\text{BG}}(v_{t+1}^f)$ и ограничениями по ускорению):

$$\hat{a}_t = \begin{cases} \hat{a}_t^{\text{nn}}, & \mathcal{A}_t = 0, \\ \Pi_{\mathcal{U}_t}\left(\hat{a}_t^{\text{nn}}\right), & \mathcal{A}_t = 1, \end{cases} \tag{3.47}$$

где $\Pi_{\mathcal{U}_t}$ - оператор проекции, вычисляемый с использованием ограничения и динамически скорректированных коэффициентов C_0,C_1,C_2. Этот шаг является ключевым отличием предлагаемого метода от подходов, основанных на нейронных сетях. Он гарантирует подавление редких аномальных режимов без необходимости переобучения нейронной сети специально для редких событий.

Шаг 6. Обновление состояния в замкнутом контуре. Скорость и дистанция обновляются по дискретной кинематике:

$$\hat{v}_{t+1}^f = \max\left(0,\, \hat{v}_t^f + \hat{a}_t \Delta t\right), \tag{3.48}$$

$$\hat{s}_{t+1} = \max\left(0,\, \hat{s}_t + \left(v_t^\ell - \hat{v}_t^f\right)\Delta t\right). \tag{3.49}$$

Шаг 7. Диагностика устойчивости и интерпретация Informer. В ходе обучения/валидации дополнительно вычисляется оценка Липшицевой нормы (верхняя оценка чувствительности выхода к возмущениям входа), используемая как диагностический индикатор потенциальной резкости управления и склонности к усилению шумов.

Informer интерпретируется посредством анализа механизма ProbSparse. Выбирается подмножество временных индексов, которые способствуют обновлению контекста. Эти индексы интерпретируются как ключевые фрагменты истории, влияющие на решение модели (фрагменты, отмечающие начало замедления лидера, периоды устойчивой конвергенции и т. д.). Этот анализ повышает прозрачность работы компонента, выполняющегося в течение длительного времени, и предоставляет инструмент для диагностики снижения производительности на конкретных наборах данных.

Bi-LSTM генерирует представление последовательности, которое более устойчиво к шуму и частичным пропускам, чем обучение причинно-следственной модели с нуля. Извлечение прямой ветви сохраняет причинно-следственную связь во время вывода и позволяет использовать преимущества предварительного обучения в качестве инициализации признаков.

Компонент LSTM обеспечивает стабильное моделирование инерции реакции и краткосрочных закономерностей, что имеет решающее значение для следования за лидером. Компонент Informer дополняет это механизмом долговременного контекста с вычислительно эффективным механизмом внимания, что важно в долгосрочных переходных процессах.

Результаты работы разработанного алгоритма представлены в таблице 27. Как видно из результатов, предлагаемый метод показывает наилучшие результаты по метрикам.

Таблица 27 — Результаты исследования предлагаемого метода

Dataset	MSE	MAE	RMSE	Collisions	Jerk
highd	23.90	3.91	4.89	0.00	2.89
ngsim	15.06	3.10	3.88	0.00	1.89
das1	26.34	4.10	5.13	0.00	2.55
das2	17.77	3.38	4.22	0.00	1.90
waymo	10.30	2.57	3.21	0.00	2.82
lyft	31.32	4.48	5.60	0.00	1.55
Собранный датасет	11.28	2.69	3.36	0.00	1.00
Среднее	19.42	3.46	4.33	0.00	2.09

Предложенный гибридный метод сочетает в себе точность моделей, основанных на данных, в выявлении сложных закономерностей, способность эффективно учитывать долгосрочный контекст и проводить интерпретируемый анализ ключевых временных сегментов с использованием Informer, а также формализованную коррекцию выбросов на основе классической модели с динамически адаптируемыми коэффициентами.

3.6 Выводы по главе

В главе проведено исследование методов прогнозирования микроскопических характеристик транспортного потока, и выполнено их сопоставление с точки зрения применимости в практических сценариях. Показано, что классические модели опираются на фиксированные параметры и при изменении

дорожной обстановки, стиля вождения и характеристик транспортного потока не обеспечивают необходимой адаптации без дополнительной настройки. Установлено также, что модели, основанные на данных, способны точнее воспроизводить наблюдаемую динамику в пределах обучающего распределения, но при замкнутом развёртывании и при сдвиге распределения могут давать аномальные предсказания, которые редки по метрикам типа MSE, но приводят к накоплению ошибки и ухудшению безопасности. [15] Отдельной проблемой выявлена ограниченная пригодность распространённых наборов данных для целевой постановки, что обусловило необходимость разработки собственного метода сбора данных, ориентированного на воспроизводимость, полноту и контролируемое качество измерений.

Разработан метод сбора данных для задач следования за лидером, ориентированный на получение репрезентативных траекторных наблюдений в широком диапазоне условий эксплуатации. В отличие от подходов, в которых акцент делается преимущественно на техническом канале измерения, предложенный метод изначально предусматривает учёт факторов, влияющих на поведение водителя и динамику потока, включая погодные условия и связанные с ними метеорологические признаки, а также освещённость, и индивидуальные характеристики водителя, в том числе стаж, пол и возраст. [12; 43; 53] Отнесение конкретной поездки к тому или иному классу осуществлялось по совокупности метеорологических признаков, сопоставленных с временными характеристиками поездки, что обеспечило сопоставимость сценариев при последующем анализе и обучении моделей. Предложенный метод отличается ориентацией на последующую валидацию в замкнутом контуре и на контроль качества данных на этапах фильтрации, синхронизации и разметки, что снижает влияние шумов и разрывов и повышает переносимость результатов между моделями и сценариями.

Сбор данных выполнен в естественных дорожных условиях с покрытием суммарного пробега более 1800 км, что позволило включить в выборку разнообразные режимы движения и состояния внешней среды. Кампания по сбору данных проводилась в течение 3 лет и включала 37 участника (21 автомобиль), и сформированы временные ряды параметров движения, пригодные для последующей фильтрации, синхронизации и разметки под задачи моделирования следования за лидером. В результате сформирован набор данных, который поддерживает как калибровку и проверку классических моделей, так и обучение

моделей, основанных на данных, при контроле репрезентативности по внешним условиям и по характеристикам водителей.

Предложен метод динамической коррекции коэффициентов классической модели следования за лидером SSSR-Micro. Динамическая коррекция сформулирована как задача восстановления отображения, которое по текущему состоянию движения позволяет адаптировать коэффициенты C_0, C_1, C_2 во времени, сохраняя физическую интерпретируемость модели и обеспечивая подстройку к изменяющимся условиям. По сравнению с традиционной статической калибровкой параметров полученный подход позволяет учитывать нестационарность среды и различия между режимами движения без переопределения структуры базовой модели.

Выполнено исследование моделей следования за лидером, основанных на данных, на популярных наборах, а также проведено сравнение в условиях замкнутого контура, что позволило оценить не только точность одношагового предсказания, но и устойчивость поведения при последовательном применении модели. По результатам исследования определены классы моделей и принципы отбора архитектур, которые согласуются с выводами о сильных и слабых сторонах подходов, выявленных при анализе классических и методов основанных на данных, и учитывают риск накопления ошибки и появления редких аномальных воздействий.

Разработан метод следования за лидером, основанный на данных, в котором предобученная модель формирует признаки и представления для дальнейшего предсказания в гибридной архитектуре LSTM-Informer, а безопасность в замкнутом контуре обеспечивается внешним ограничителем на базе классической модели SSSR-Micro с динамической коррекцией коэффициентов C_0, C_1, C_2. [54] Отличие предложенного решения от существующих подходов состоит в разделении функций модулей, поскольку модуль, основанный на данных, отвечает за точность воспроизведения динамики, а классический модуль обеспечивает формализуемые ограничения безопасности и подавление аномальных режимов. Новизна заключается в построении гибридной схемы для замкнутого развёртывания, где редкие аномальные выходы нейросети корректируются внешним контуром, что повышает устойчивость при сдвиге распределения и снижает деградацию качества из-за накопления ошибки, при сохранении высокой точности в типичных режимах движения.

Разработан метод следования за лидером, основанный на данных, в котором предобученная Bi-LSTM используется для извлечения дополнительных каузальных признаков состояния, повышающих переносимость и устойчивость представления без существенного роста вычислительной сложности на инференсе. В качестве предсказательной части применяется гибридная архитектура LSTM-Informer, где LSTM-компонента устойчиво работает на краткосрочных окнах и моделирует инерционность и задержку реакции водителя в локальной динамике, а Informer-компонента предназначена для обработки длинных последовательностей и протяжённых переходных процессов при приемлемой вычислительной сложности за счёт разреженного внимания и дистилляции. Использование Informer также повышает интерпретируемость модели, поскольку веса механизма внимания позволяют анализировать вклад временных фрагментов длинного контекста в формирование прогноза. Безопасность в замкнутом контуре обеспечивается внешним ограничителем на базе классической модели SSSR-Micro с динамической коррекцией коэффициентов C_0, C_1, C_2, что позволяет подавлять редкие аномальные выходы нейросетевого блока и повышать устойчивость при сдвиге распределения при сохранении высокой точности в типичных режимах движения. Отличие предложенного решения от существующих подходов состоит в разделении функций, поскольку модуль, основанный на данных, отвечает за точность воспроизведения динамики, а классический модуль обеспечивает формализуемые ограничения безопасности и подавление аномальных режимов.

Глава 4. Разработка методологии конфигурирования архитектуры исполнения моделей ИИ в ИТС

В ИТС модели искусственного интеллекта могут исполняться в разных режимах. Для части задач достаточно пакетной обработки, когда данные заранее накоплены и передаются модели по расписанию или по событию запуска задания. Такой режим далее обозначается как пакетное исполнение моделей ИИ (ПЭМ). Он применяется для расчёта отчётных показателей, переобучения моделей, ночной обработки архивов, построения витрин данных и выполнения аналитических процедур, не требующих немедленного отклика. [46; 52]

В ПЭМ основное значение имеют пропускная способность, воспроизводимость расчёта и контроль ресурсов. Требования к миллисекундной задержке здесь обычно не являются определяющими. Важнее, чтобы одно и то же задание при одинаковых входных данных, версии модели и конфигурации среды давало воспроизводимый результат, а ход выполнения можно было проверить по журналам и артефактам запуска.

Формально ПЭМ соответствует режиму обработки, при котором входные данные задаются как конечный набор $D = \{x_i\}_{i=1}^{N}$, а результат вычисления определяется отображением $\hat{Y} = F_\theta(D)$. Запуск задания выполняется дискретно, то есть по расписанию или по внешнему триггеру. В отличие от онлайн-сервиса, здесь допустима задержка, связанная с подготовкой окружения, чтением данных и распределением вычислений по кластеру.

Типовой контур пакетного исполнения включает несколько операций.

1. **Сборка модели.** Код модели, зависимости, параметры запуска и конфигурационные файлы объединяются в воспроизводимый дистрибутив. Чаще всего модель реализуется на Python, но в инфраструктуре могут также использоваться Java, Scala или R. Сборка автоматизируется средствами CI/CD, например Jenkins, чтобы исключить ручную подготовку окружения перед каждым запуском.
2. **Развёртывание модели.** После сборки дистрибутив передаётся в среду пакетного исполнения. В качестве вычислительной базы может использоваться Hadoop-кластер, GreenPlum или иная платформа обработки данных. [31] Для распределённых вычислений применяется Apache Spark. Его базовая структура RDD (Resilient Distributed

Dataset) представляет собой неизменяемую коллекцию данных, распределённую между узлами кластера и поддерживающую операции `map`, `filter` и `reduce`. Для структурированных данных дополнительно используются Spark DataFrame и Dataset, включая SQL-подобные операции. [34]

3. **Управление ресурсами и запуск.** Планировщик YARN (Yet Another Resource Negotiator) распределяет CPU, память и при наличии соответствующей конфигурации GPU-ресурсы между заданиями. Это позволяет запускать несколько вычислительных процессов на одном кластере и контролировать конкуренцию за ресурсы.
4. **Мониторинг и отказоустойчивость.** Для каждого задания фиксируются статус выполнения, журналы, параметры запуска и сведения об ошибках. При сбое возможен повторный запуск задания или его отдельных этапов. Такой механизм особенно важен при обработке больших массивов транспортных данных, где полный пересчёт может занимать значительное время.
5. **Запуск и интеграция.** Пакетные задания могут запускаться по расписанию, по событию поступления новых данных или через REST API. В последнем случае API выполняет роль интерфейса управления заданием, а не интерфейса низколатентного инференса.
6. **Масштабирование и хранение данных.** Для хранения больших массивов данных используется HDFS или совместимое распределённое хранилище. Горизонтальное масштабирование достигается добавлением вычислительных и дисковых узлов, что важно при росте объёма архивов телеметрии, журналов, траекторий и агрегированных транспортных показателей.

Другой режим связан с потоковым исполнением моделей. Он применяется тогда, когда модель должна работать внутри прикладного контура и отвечать на входные события без длительного ожидания. В ИТС такой режим нужен для онлайн-прогноза, обработки телеметрии, выдачи оценок состояния, реакции на инциденты и интеграции с сервисами управления. В этом случае модель обычно разворачивается как контейнеризованный сервис в Kubernetes, использует Docker-образ и предоставляет интерфейс через HTTP, gRPC или брокер сообщений Kafka. При необходимости к сервису подключаются GPU-ресурсы.

Для потокового режима MLOps-цепочка строится иначе, чем для ПЭМ. Здесь важны не только воспроизводимость сборки, но и контроль версии модели, безопасное развёртывание, проверка доступности сервиса, мониторинг задержки и масштабирование под нагрузкой.

1. **Сборка модели.** Код модели, зависимости и конфигурация упаковываются в Docker-образ. Jenkins запускает сборку, проверяет состав артефактов и формирует версию образа, пригодную для развёртывания.
2. **Проверка качества.** Перед публикацией образа выполняется автоматическая проверка кода и тестов. Для статического анализа может использоваться SonarQube. На этом этапе выявляются ошибки реализации, проблемы покрытия тестами и нарушения правил разработки.
3. **Тестирование и валидация.** Выполняются модульные, интеграционные и при необходимости нагрузочные тесты. Проверяется не только корректность выхода модели, но и работоспособность сервиса в целевом окружении.
4. **Упаковка.** После успешного прохождения проверок Docker-образ публикуется в репозитории артефактов. Там сохраняются версия модели, версия образа и метаданные сборки.
5. **Развёртывание в Kubernetes.** Модель разворачивается в кластере через MLOps-пайплайн. Конфигурация задаёт число реплик, лимиты ресурсов, переменные окружения, сетевые интерфейсы и правила обновления.
6. **Мониторинг и масштабирование.** После развёртывания контролируются доступность сервиса, задержка ответа, ошибки, загрузка CPU/GPU и потребление памяти. Для мониторинга может использоваться Prometheus. Автоматическое масштабирование настраивается по нагрузке и эксплуатационным метрикам. [194; 195]

4.1 Архитектура исполнения моделей ИИ

После сбора и предварительной обработки данных модель должна быть размещена в вычислительной среде, где она выполняет инференс над входными

потоками. Для ИТС этот этап не является только вопросом выбора сервера или библиотеки. Здесь одновременно учитываются задержка ответа, пропускная способность, доступность сервиса и устойчивость при изменении интенсивности событий. Эти требования особенно заметны в задачах управления движением, прогноза заторов, оценки времени поездки и обнаружения инцидентов. Поэтому архитектура исполнения выбирается с учётом целевой задачи, доступной инфраструктуры, периферийных узлов, каналов связи и ограничений по SLA. [196; 197]

Поток данных задаётся последовательностью $\{x_t\}_{t\geqslant 0} \subset \mathbb{R}^d$, где x_t описывает наблюдения сенсоров и внешний контекст в дискретный момент времени. Индекс t соответствует поступлению очередного события или сообщения в систему [198; 199]. Сквозная задержка обозначается как L_t. Её компоненты включают задержку получения и первичной загрузки данных L_t^{ing}, задержку предобработки и извлечения признаков L_t^{feat}, задержку инференса L_t^{inf} и задержку постобработки L_t^{post}.

Эти величины рассматриваются как случайные характеристики события с индексом t. Для моделей, зависящих от истории, используется оконная агрегация $W_t = \{x_{t-h+1}, \ldots, x_t\}$ и отображение признаков $\xi_t = \varphi(W_t, c_t)$, где $c_t \in \mathcal{C}$ задаёт геопространственный и операционный контекст дорожной сети. Предсказание формируется моделью $y_t = f_\theta(\xi_t, s_{t-1})$, а скрытое состояние обновляется как $s_t = u(s_{t-1}, \xi_t)$. Такая запись подходит для рекуррентных, графовых, свёрточных и трансформерных моделей, применяемых к пространственно-временным данным ИТС. Для больших языковых моделей и близких по вычислительной структуре архитектур дополнительно важны управление памятью и планирование инференса [200; 201].

Контур исполнения удобно представить вычислительным графом $\mathcal{G} = (V, E)$. Вершины соответствуют этапам обработки от приёма входного сообщения до публикации результата, а рёбра задают передачу данных между этапами. Если задержка в узле $v \in V$ равна τ_v, то сквозная задержка события оценивается суммой основных компонент:

$$L_t = L_t^{\mathrm{ing}} + L_t^{\mathrm{feat}} + L_t^{\mathrm{inf}} + L_t^{\mathrm{post}},$$

где учитывается транспортная задержка, извлечение признаков, вычисление отклика модели и операторы публикации результатов. Для систем ИТС задается

соглашение по уровню сервиса

$$\mathbb{P}\{L_t \leqslant \Delta\} \geqslant 1 - \varepsilon\,,$$

где Δ задаёт допустимую задержку, а ε задаёт допустимую долю нарушений. В ИТС это ограничение важно не только для среднего режима, но и для пиковых нагрузок, деградаций связи и частичных отказов инфраструктуры. [196; 197]

В потоковом режиме входная нагрузка описывается интенсивностью $\lambda(t)$. Производительность узла инференса зависит не только от модели, но и от параметров её исполнения. Поэтому отдельно задаются размер микропакета, формат вычислений и класс аппаратной платформы.

Пусть $b \in \mathbb{N}$ обозначает размер микропакета, то есть число событий или запросов, обрабатываемых одним вызовом модели. Пусть $q \in \mathcal{Q}$ задаёт формат представления весов и активаций, например FP32, FP16 или INT8. Пусть $h \in \mathcal{H}$ обозначает класс аппаратной платформы: CPU-узел, GPU определённого поколения или специализированный ускоритель. Тогда $\tau_{\text{inf}}(b,q,h)$ трактуется как время выполнения инференса для одного микропакета размера b при выбранных параметрах q и h. Эффективная скорость обслуживания записывается как

$$\mu(b, q, h) = \frac{b}{\tau_{\text{inf}}(b, q, h)} \tag{4.1}$$

Из 4.1 следует, что пропускная способность стадии инференса управляется тремя группами параметров. Увеличение b обычно повышает μ, поскольку накладные расходы вызова модели распределяются на большее число запросов. Однако при этом возрастает задержка микропакета, что может ухудшить латентность отдельного события. Формат q задаёт компромисс между скоростью и точностью. Более грубое квантование может уменьшить время вычислений и обменов с памятью, но для него требуется контроль условия $\text{Acc}(q) \geqslant \text{Acc}_{\min}$. Платформа h определяет базовый профиль производительности: вычислительную мощность, пропускную способность памяти и ограничения ввода-вывода.

Для обеспечения устойчивости очередей выбирается коэффициент загрузки $\rho \in (0,1)$, после чего требуемое число параллельных реплик оценивается функцией автомасштабирования [202; 203]

$$n^{\star}(t) = \left\lceil \frac{\lambda(t)}{\rho\, \mu\big(b(t), q(t), h(t)\big)} \right\rceil \tag{4.2}$$

Поскольку $\mu(b,q,h)$ является функцией параметров эксплуатации, условие устойчивости очередей

$$\lambda(t) \;<\; \rho\, n\, \mu\big(b(t),q(t),h(t)\big), \qquad 0<\rho<1, \tag{4.3}$$

следует трактовать как совместное ограничение на интенсивность входящего потока $\lambda(t)$, число параллельных реплик n, а также тактические параметры эксплуатации (b,q,h). Следовательно, размер микропакета b, формат вычислений q и класс узла h выступают как параметры эксплуатации. Их изменение позволяет поддерживать запас по загрузке ρ при нестационарном входном потоке.

Для целей анализа удобно явно выделить источники зависимости $\tau_{\text{inf}}(b,q,h)$. В простейшем приближении можно записать

$$\tau_{\text{inf}}(b,q,h) \;=\; \tau_{\text{oh}}(h) \;+\; \tau_{\text{comp}}(b,q,h) \;+\; \tau_{\text{mem}}(b,q,h), \tag{4.4}$$

где $\tau_{\text{oh}}(h)$ описывает постоянные накладные расходы на вызов/планирование на платформе h, τ_{comp} - вычислительную составляющую (зависящую от формата q и степени параллелизма платформы h), а τ_{mem} - память и ввод-вывод-зависимую составляющую (чтение весов, активаций, обмены по шине). Такая декомпозиция показывает, какая часть инференса становится узким местом. При малых микропакетах заметнее постоянные накладные расходы. При больших микропакетах возрастает роль вычислений и обменов с памятью. Поэтому выбор b, q и h должен выполняться не изолированно, а совместно с требованиями к задержке и точности.

Настройка пакетизации и квантования используется как оперативный механизм реакции на всплески нагрузки. Для ускорителей задержка инференса часто растёт медленнее, чем размер микропакета. В упрощённой форме это можно записать как

$$L_t^{\text{inf}}(b) \approx \tau_0 + \tau_1\, b^{\gamma} \quad \text{при} \quad 0<\gamma<1$$

Поэтому значение b выбирается как компромисс: крупный микропакет лучше загружает ускоритель, но может нарушить ограничение по задержке отдельного события.

Специфика ИТС влияет и на выбор места исполнения модели. Если инференс выполняется в удалённом центре обработки данных, то в задержку входит доставка входных сообщений по транспортной сети. В первом приближении сетевая составляющая может быть описана зависимостью

$$L_t^{\text{net}}(d) \approx \alpha + \beta d\,,$$

где d отражает эффективную сетевую дистанцию с учётом топологии и промежуточных узлов.

Перенос инференса на периферийный узел сокращает транспортное плечо между источником данных и моделью. В результате уменьшается компонента L_t^{ing}, связанная с доставкой, буферизацией и ожиданием входных данных. Помимо снижения средней задержки, этот подход также уменьшает вариативность задержки, поскольку часть трафика не проходит через магистральные каналы, где могут возникать конкурирующие потоки и сетевые очереди. Этот эффект особенно важен для задач обнаружения угроз, где реакция должна быть практически в режиме реального времени.

Для уточнения интерпретации введем разложение компонента задержки на входе:

$$L_t^{\text{ing}} \;=\; L_t^{\text{acq}} \;+\; L_t^{\text{tx}} \;+\; L_t^{\text{queue}}, \tag{4.5}$$

где L_t^{acq} - задержка сбора/считывания данных с источника, L_t^{tx} - транспортная задержка доставки входных данных до вычислительного узла, выполняющего инференс, L_t^{queue} - ожидание в сетевых/входных буферах (включая эффекты очередей при конкурирующем трафике). При переносе инференса на периферию в первую очередь уменьшается L_t^{tx}. Дополнительно может снижаться L_t^{queue}, если входные данные больше не конкурируют с магистральным фоновым трафиком.

В работающей ИТС отказ отдельного экземпляра сервиса не должен приводить к остановке процесса вывода. Поэтому сервис модели размещается в нескольких репликах, изолированных друг от друга на уровне контейнеров и узлов кластера. Пусть компонент имеет $r \in \mathbb{N}$ реплик, а результат считается доступным, если хотя бы одна реплика отвечает на запрос. Через $A_1 \in [0,1]$ обозначим доступность одной реплики на выбранном интервале наблюдения. При предположении статистической независимости отказов реплик доступность ансамбля равна

$$A_r \;=\; 1 - (1 - A_1)^r \,. \tag{4.6}$$

Условие $A_r \geqslant A_{\min}$ далее используется как ограничение при выборе числа реплик и схемы размещения. Само увеличение r не решает всю задачу, поскольку реплики могут конкурировать за общий сетевой канал, память или ускоритель. Поэтому вместе с доступностью контролируется вероятность

просрочки $P\{L_t > \Delta\}$. В этой записи надёжность вычислительного контура связывается с устойчивостью задержки, а не только с фактом работоспособности сервиса.

В потоковом режиме модель постепенно сталкивается с данными, которые отличаются от обучающей выборки. Причиной могут быть сезонность, ремонтные работы, аварии, изменение маршрутов общественного транспорта или перераспределение спроса. Для контроля такого изменения вводится статистика $D(P_t,P_{\text{ref}})$. Здесь P_t описывает распределение признаков или ответов модели в текущем окне, а P_{ref} соответствует выбранному эталонному периоду. Если выполняется условие

$$D\big(\mathcal{P}_t, \mathcal{P}_{\text{ref}}\big) > \tau$$

запускается процедура адаптации. В простом варианте она сводится к пересчёту признаков φ, переключению на более устойчивую версию модели или постановке задания на дообучение параметров θ. Остановка сервиса при этом не требуется. Дополнительно вводится фильтр уверенности ответа:

$$\eta_t = \max_k p_\theta(y_t = k \mid \xi_t)\,, \quad \eta_t < \eta_{\min} \Rightarrow \text{эскалация для дополнительной проверки}\,.$$

В этом случае низкая уверенность не трактуется как штатный прогноз. Ответ передаётся на дополнительную проверку, либо заменяется более консервативной оценкой в зависимости от прикладного сценария.

На уровне реализации модель разворачивается как изолированный сервис. Контейнеризация упрощает перенос между средами, а оркестрация позволяет задавать реплики, лимиты ресурсов, правила обновления и проверки готовности [196; 204]. Если входной поток поступает через брокер сообщений, для стабилизации нагрузки используется обратное давление. Потребитель не должен получать больше сообщений, чем способен обработать без роста очереди.

Для ИТС обычно требуется совместить периферийную обработку и центральную аналитику. На периферии остаются операции, чувствительные к задержке: первичная предобработка, фильтрация, быстрый инференс и локальная реакция. В центральной части выполняются агрегирование, переобучение, отчётная аналитика и сценарные расчёты.

Выбор конкретной реализации определяется целевыми метриками. Для сценариев оперативного управления движением накладывается ограничение $L_t \leqslant \Delta$ с малым ε. Для пакетной отчетности допускается увеличение b, что

повышает μ при умеренном росте $L_t^{\inf}$. Для оценки времени поездки применяется составная модель маршрута на графе дорог. Пусть $e \in E_{\text{road}}$ обозначает ребро дорожной сети, тогда прогноз на горизонте записывается как

$$\widehat{T}(r) = \sum_{e \in r} \widehat{t}_e(\xi_t)$$

где r есть последовательность ребер, а $\widehat{t_e}$ предсказывается локальной моделью на основе текущих признаков и состояния. Для прогноза скорости используется отображение

$$\widehat{v}_{t+\delta} = g_\theta(\xi_t)\,.$$

Для анализа инцидентов важна высокочувствительная детекция редких событий, что требует балансировки порога срабатывания и пропускной способности при ограничении на долю ложных тревог.

Методологически потоковая эксплуатация в ИТС описывается как решение задачи оптимизации, где минимизируется сквозная задержка при ограничениях на точность, стоимость и надежность [196; 197]. Пусть $C(n, h)$ отражает стоимость ресурсов, тогда

$$\begin{aligned} &\min_{n,b,q,h} \ \mathbb{E}[L_t] \\ &\text{при условиях} \quad \mathbb{P}\{L_t \leqslant \Delta\} \geqslant 1 - \varepsilon\,, \\ &\qquad\qquad\ \ \mathrm{Acc}(q) \geqslant \mathrm{Acc}_{\min}\,, \\ &\qquad\qquad\ \ A_n \geqslant A_{\min}\,, \\ &\qquad\qquad\ \ C(n, h) \leqslant C_{\max}\,. \end{aligned} \tag{4.7}$$

Выбор конфигурации далее рассматривается как инженерная задача с несколькими ограничениями. Необходимо подобрать размещение модели, число реплик, режим пакетизации, формат вычислений и класс аппаратной платформы так, чтобы выполнить SLA по задержке и доступности, не превысить ресурсный бюджет и сохранить минимально допустимое качество предсказаний.

4.1.1 Исполнение моделей с использованием реляционных БД в ИТС

Реляционная БД в данном варианте используется как внешний источник и приёмник данных для сервиса модели. Такой вариант нужен в тех случаях, когда ИТС уже имеет действующие реестры дорожной инфраструктуры, таблицы телеметрии, журналы событий и витрины отчётности. Переход на специализированное хранилище в такой ситуации не всегда возможен, поэтому доступ к данным организуется через JDBC и SQL-диалект используемой СУБД [205]. В этом режиме модель читает признаки из реляционной схемы, формирует результат инференса и записывает обратно прогнозы, служебные признаки или метрики качества.

В схеме выделяются проксирующий слой, сервис модели и контроллер базы данных. Контроллер выполняет запросы к БД через JDBC, передаёт данные сервису модели по HTTP или gRPC и фиксирует технические события в журнале наблюдаемости. Разделение полезно так как вычислительный сервис модели можно масштабировать отдельно от слоя доступа к БД, а подключение к существующей реляционной инфраструктуре заказчика не требует изменения кода модели. Архитектура взаимодействия компонентов представлена на рисунке 4.1.

Информационный поток в этом варианте задаётся через схему $\Sigma = \{R_1, \ldots, R_m\}$. Каждое отношение R_k содержит временные и пространственные атрибуты, связанные с улично-дорожной сетью. Пусть $\mathcal{F}(e,t)$ обозначает вектор признаков для дорожного сегмента e в момент t. Параметризованный запрос $Q(\pi,t)$ возвращает упорядоченное множество строк $\mathcal{R}_t = \{r_1, \ldots, r_N\}$, а векторизатор φ преобразует эти строки в признаки $\mathcal{F}(e,t) = \varphi(\mathcal{R}_t)$. После этого модель вычисляет прогноз $y_t = f_\theta(\mathcal{F}(e,t))$. Такая схема применима для прогнозирования скоростей, времени поездки и других показателей, если нужные признаки доступны в реляционной БД.

Режим HTTP с потоковой выдачей операторов SELECT В HTTP-варианте контроллер запускает $Q(\pi,t)$ и передаёт строки сервису модели постепенно, не загружая весь результат запроса в память [196; 198]. Поток $\mathcal{R}_t$ разбивается на фрагменты размера w. Если средний размер строки равен σ, то

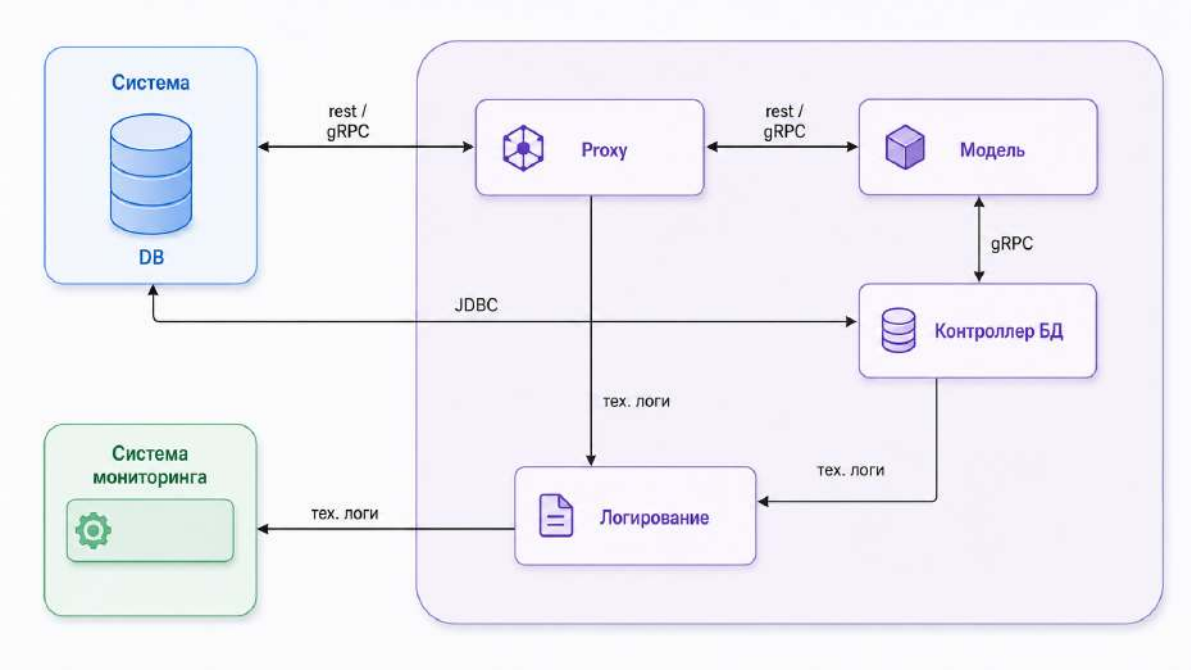

Рисунок 4.1 — Архитектура исполнения моделей с использованием реляционной БД в контуре ИТС

пиковое потребление памяти на стороне контроллера приближённо оценивается как σw. Сервис модели обрабатывает такие фрагменты аналогично. Поэтому конвейер может работать с большим результатом запроса без накопления полного набора строк.

Сквозная задержка представляется суммой компонент

$$L_{\mathrm{e2e}} \approx T_{\mathrm{fb}} + \sum_{j=1}^{N} \left(\frac{\sigma}{B} + \alpha_{\mathrm{ctrl}} + \alpha_{\mathrm{mdl}} \right),$$

где T_{fb} задаёт время до поступления первой строки, B обозначает эффективную пропускную способность сети, а α_{ctrl} и α_{mdl} описывают удельные затраты сериализации и обработки в элементах конвейера. Размер фрагмента ограничен условиями $\sigma w \leqslant M_{\mathrm{ctrl}}$ и $\sigma w \leqslant M_{\mathrm{mdl}}$. Потоковая схема предпочтительна при большом N, когда полный результат запроса трудно удерживать в памяти контроллера или сервиса модели.

Режим gRPC с простым вызовом RPC. В gRPC-варианте контроллер сначала полностью выполняет $Q(\pi,t)$, затем формирует одно protocol-сообщение и передаёт его сервису модели. Объём такого сообщения равен $S = \sigma N + \delta$, где δ учитывает служебные заголовки. Для корректной работы должны выполняться ограничения $S \leqslant S_{\max}$, $S \leqslant M_{\mathrm{ctrl}}$ и $S \leqslant M_{\mathrm{mdl}}$. Этот режим полезен для

выборок среднего размера и в ситуациях, когда важно получить полный результат за один вызов. Для операций изменения данных используется пакетное выполнение запросов, что снижает накладные расходы, связанные с операциями фиксации транзакций.

Отдельная проблема возникает при передаче параметров gRPC в виде строк. Пусть $\mathbf{p}_{\text{str}} = (p_1, \ldots, p_k)$ задаёт набор текстовых значений. Перед выполнением SQL-запроса эти значения необходимо преобразовать в ожидаемые типы данных. Это можно сделать с помощью преобразования ψ или через явные приведения типов в запросе. Для отдельного параметра используется правило

$$\psi_j(p_j) = \begin{cases} \text{CAST}(p_j \text{ AS INTEGER}) & \text{если требуется целочисленный тип} \\ \text{CAST}(p_j \text{ AS TIMESTAMP}) & \text{если требуется временной тип} \\ p_j & \text{в остальных случаях} \end{cases}$$

Это правило устраняет любую неоднозначность в сопоставлении параметров со схемой базы данных. При реализации параметры следует связывать с помощью подготовленных JDBC-операторов и белых списков разрешенных преобразований. Это снижает риск SQL-инъекций, ошибок и некорректных сравнений строковых и числовых значений. Сводная информация о методах интерфейса приведена в таблице 28.

Таблица 28 — Описание интерфейса контроллера БД

№	Метод	Протокол	Назначение	Исходные данные	Возвращаемое значение
1	`db/v1/select`	HTTP	Выполнение операторов выборки в потоковом режиме без полной загрузки результата в память контроллера и сервиса модели	Структура тела запроса указана в листинге 1	Поток JSON записей из БД, структура строки приведена в листинге 2
2	`db/v1/update`	HTTP	Выполнение запросов изменения данных, включая `UPDATE`, `INSERT`, `DELETE`	Без тела запроса либо параметры по соглашению, зависящему от шаблона запроса	Количество затронутых записей

продолжение на следующей странице

№	Метод	Протокол	Назначение	Исходные данные	Возвращаемое значение
3	`db/v1/batch-update`	HTTP	Выполнение `INSERT` в пакетном режиме с уменьшением транзакционных накладных расходов	Структура тела запроса приведена в листинге 3	Суммарное количество вставленных записей
4	`Execute`	gRPC	Метод общего назначения для выполнения запросов изменения данных, а также для операторов `SELECT` в непотоковом режиме	Определение запросов `Execution Request` и `Batch Execution Request` приведено в листинге 4	Ответ `Query Execution Response` с `Query Result` описан в листинге 5
5	`BatchExecute`	gRPC	Выполнение `INSERT` в пакетном режиме через `Batch Execution Request`	Определение сообщения совпадает с листингом 4 для пакетного варианта	`Query Execution Response` с полем `records_affected`

Чтение и запись пакетами. Пусть пакет содержит B операций обновления, а удельная стоимость обработки одной строки равна τ_{row}. Тогда время выполнения одной транзакции оценивается как $T_{\text{batch}} \approx \tau_{\text{begin}} + B\tau_{\text{row}} + \tau_{\text{commit}}$. При росте B средняя стоимость одной записи обычно снижается, пока не возникает насыщение подсистемы журналирования или блокировок. Поэтому B выбирается по минимуму T_{batch}/B с учётом ограничения на задержку сервиса и допустимого размера журнала. Для ИТС это важно при сохранении признаков, прогнозов и метрик, которые затем используются в отчётности и контроле SLA.

Модель производительности и критерии выбора режима. Производительность режима доступа к БД можно описать через $\mu = 1/\tau$, где τ включает планирование запроса, чтение страниц, сериализацию и передачу по сети. Для потокового HTTP важным параметром является T_{fb}, то есть время до первой

строки. Для gRPC результат сначала полностью формируется, а затем передаётся одним сообщением. Это уменьшает число сетевых обменов, но повышает требования к памяти. Поэтому выбор режима сводится к проверке условий

$$\sigma N \leqslant \min\left(S_{\max}, M_{\text{ctrl}}, M_{\text{mdl}}\right) \Rightarrow gRPC \text{ предпочтителен}$$

и

$$\sigma N > \min\left(S_{\max}, M_{\text{ctrl}}, M_{\text{mdl}}\right) \Rightarrow \text{потоковый HTTP обеспечивает устойчивость.}$$

В ИТС эти условия проверяются вместе с ограничением по сквозной задержке $\mathbb{P}\{L_{\text{e2e}} \leqslant \Delta\} \geqslant 1 - \varepsilon$. Параметр Δ задаёт допустимую латентность для выбранного сценария, а ε фиксирует допустимую долю нарушений при пиковых нагрузках.

Транзакционность и согласованность чтения. Для аналитических запросов `SELECT` в потоковом режиме обычно достаточно уровня изоляции `Read Committed`. Он снижает накладные расходы и подходит для задач, где допустимо читать актуальное состояние без фиксации полного снимка. Если расчёт чувствителен к согласованности, например при сверке метрик качества маршрутов, предпочтителен уровень `Repeatable Read`. Запись признаков и метрик следует выполнять короткими транзакциями с идемпотентными ключами. Пусть таким ключом служит пара (e,t). Тогда повторный вызов может обрабатываться через `upsert`, а дублирование записи не меняет итоговое состояние.

4.1.2 Предлагаемая реализация исполнения моделей

Реализация выполнена для кластера Kubernetes. В контуре выделены маршрутизатор внешних вызовов, проксирующий контроллер, сервис модели, контроллер доступа к реляционной БД и подсистема журналирования [206; 207]. Такая декомпозиция нужна для того, чтобы отделить код модели от доступа к БД, маршрутизации и технического мониторинга. Информационные потоки приведены в таблице 29. Защищённое взаимодействие между компонентами задаётся через mTLS на границах приложений [208]. На рисунке 4.2

показана схема компонентов и маркировка маршрутов, используемая далее при описании потоков.

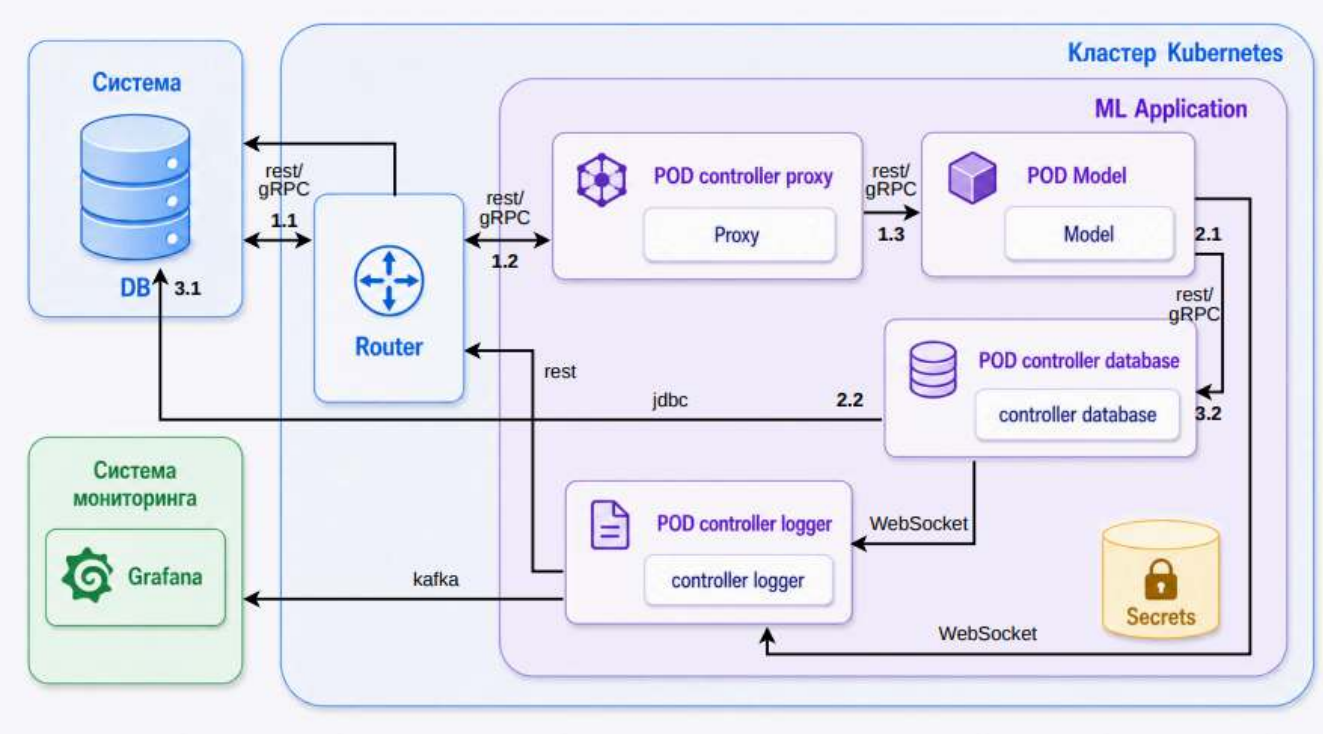

Рисунок 4.2 — Предлагаемая архитектура исполнения моделей с использованием реляционной БД в кластере Kubernetes

Множество вершин задаётся как V = {System DB, Router, Proxy, Model, Controller DB, Controller Logger, Grafana, Secrets}. Информационные потоки $E = \{1.1, 1.2, 1.3, 2.1, 2.2, 3.1, 3.2\}$ рассматриваются как ориентированные рёбра графа исполнения. Для основного пути инференса используется композиция $\pi_1 = (1.1 \circ 1.2 \circ 1.3)$. Для обращения модели к реляционной БД используется путь $\pi_2 = (2.1 \circ 2.2 \circ 3.1 \circ 3.2)$. Первый путь передаёт входные данные в сервис модели. Второй путь описывает обращение к контроллеру БД, выполнение запроса и возврат результата. Обозначения вида $i.j$ используются как идентификаторы потоков. Они не задают иерархию компонентов. Символ $\circ$ далее обозначает последовательное соединение рёбер ориентированного графа, то есть сквозной маршрут, в котором конец одного ребра совпадает с началом следующего.

Условия синхронности для взаимодействий модели с контроллером базы формулируются как $\forall e \in \{2.1, 3.2\} \Rightarrow \text{mode}(e) = \text{sync}$.

Таблица 29 — Информационные потоки взаимодействия в предлагаемой архитектуре

Код	Описание взаимодействия	Инициатор	Получатели	Сущности в потоке	Механизм реализации
1	Отправка данных для эксплуатации модели				
1.1	Передача данных в ML-приложение через маршрутизатор	Система	Router	вектор данных	REST / gRPC (mTLS)
1.2	Проксирование входного потока на граничный контроллер	Router	Controller –Proxy	вектор данных	REST / gRPC (mTLS)
1.3	Доставка входного потока к сервису модели и завершение TLS на границе	Controller –Proxy	Модель	вектор данных	REST / gRPC (mTLS)
2	Обращение к реляционной БД через контроллер базы				
2.1	Инициация запроса к БД из кода модели	Модель	Controller –Database	вектор данных	REST / gRPC (mTLS)
2.2	Формирование SQL и обращение к целевой БД	Controller –Database	БД системы	вектор данных	JDBC (mTLS)
3	Возврат результата из БД в модель через контроллер базы				
3.1	Передача результата выполнения запроса от БД контроллеру	БД системы	Controller –Database	вектор данных	JDBC (mTLS)
3.2	Возврат результата модели после обработки на контроллере	Controller –Database	Модель	вектор данных	REST / gRPC (mTLS)

Контроллер базы данных включается только для тех моделей, которым требуется доступ к реляционной БД. Если модель работает без внешней БД, данный компонент из конфигурации исключается. Обращение модели к контроллеру выполняется синхронно. Это упрощает трассировку запроса и поз-

воляет явно связывать входной вызов модели с конкретным SQL-обращением. В состав контроллера входит JDBC-драйвер нужного семейства СУБД и карта запросов, передаваемая при развёртывании. Контроллер выполняет операции класса DML и не выходит за пределы заданного набора запросов. Журналы и метрики передаются в отдельный сервис, где фиксируются технические события, ошибки и показатели качества выполнения. В одном ML-приложении может быть развёрнуто несколько экземпляров сервиса модели. Для каждого экземпляра задаётся собственный контроллер БД, связанный с одной родительской базой данных. Доступ к контроллеру выполняется через внутрикластерный адрес типа ClusterIP. Параметры подключения к БД хранятся в Kubernetes Secrets. К ним относятся имя пользователя, пароль и материалы для mTLS [208]. Подготовка секретов относится к зоне ответственности администратора.

Контроллер не зависит от языка реализации модели. Для модели он выглядит как обычный внутренний сервис, вызываемый по REST или gRPC. При более жёстких требованиях безопасности могут добавляться дополнительные факторы аутентификации и обязательное шифрование всех транспортных соединений. Такой вариант поставки позволяет не включать в артефакт модели отдельные драйверы доступа к БД и не смешивать логику исполнения с логикой интеграции.

Формальная спецификация требований безопасности описывается ограничениями

$$\forall e \in E \Rightarrow \mathrm{TLS}(e) = 1\,, \quad \mathrm{Creds}(\text{Controller DB}) \in \mathrm{Secrets}\,,$$

где функция TLS указывает на обязательность защищённого канала, а множество Secrets хранит идентификационные данные и ключевые материалы. Свойство многомодельности формулируется как соответствие

$$\forall\ \mathrm{Model}_i\ \exists\ \text{Controller DB}_i \text{ такое, что}$$
$$\mathrm{mode}(2.1_i) = \mathrm{sync}\ \wedge\ \mathrm{bind}(\text{Controller DB}_i) = \mathrm{DB}_{\mathrm{parent}}\,.$$

Это условие фиксирует простое правило эксплуатации, а именно один экземпляр контроллера обслуживает одну родительскую БД. За счёт этого легче проверять права доступа, трассировать SQL-запросы и разбирать ошибки синхронного вызова.

Верификация доступности путей в конвейере CI

Верификация проводится до выпуска конфигурации в эксплуатационную среду. Проверяются конфигурация Kubernetes и фактическая связность компонентов в плоскости данных. Для этого используется ориентированный граф исполнения $\mathcal{G} = (V,E)$, где вершины соответствуют сервисам $V = \{\mathrm{Router}, \mathrm{ControllerProxy}, \mathrm{Model}, \mathrm{ControllerDB}, \mathrm{SystemDB}\}$. Маршруты определяются композициями $\pi_1 = (1.1 \circ 1.2 \circ 1.3)$ и $\pi_2 = (2.1 \circ 2.2 \circ 3.1 \circ 3.2)$. Для ребра e задаются функции источника и приёмника $\mathrm{src}(e), \mathrm{dst}(e) \in V$. В терминах кластера вводятся предикаты $\mathrm{Ready}(v)$ готовность подов сервиса, $\mathrm{TLS}(e)$ наличие взаимной аутентификации по сертификатам, $\mathrm{RBAC}(e)$ право вызова по учётной записи модели, $\mathrm{HP}(v)$ пройденная проверка на работоспособность, $\mathrm{EP}(v)$ наличие эндпоинтов у сервисов Kubernetes [206], $\mathrm{MNT}(v,\Sigma)$ отображение множества Σ конфиденциальных артефактов конфигурации, хранимых в подсистеме Kubernetes Secrets, на поды сервиса. Эти артефакты необходимы для выполнения требований безопасности и связности контура, $\mathrm{OK}(e)$ успех контрольного запроса по протоколу соединения ребра, $L(e)$ наблюдаемая задержка передачи по ребру, Δ_{inf} порог задержки для пути π_1, Δ_{db} порог задержки для пути π_2. Сквозная задержка пути вычисляется как $L(\pi) = \sum_{e \in \pi} L(e)$.

Инвариант доступности для произвольного пути π определяется выражением

$$\begin{aligned}\mathcal{I}(\pi) = &\Big(\bigwedge_{e \in \pi} \mathrm{Ready}(\mathrm{src}(e)) \wedge \mathrm{Ready}(\mathrm{dst}(e)) \Big) \\ &\wedge \Big(\bigwedge_{e \in \pi} \mathrm{TLS}(e) \wedge \mathrm{RBAC}(e) \wedge \mathrm{OK}(e) \Big) \\ &\wedge \Big(\bigwedge_{v \in V(\pi)} \mathrm{HP}(v) \wedge \mathrm{EP}(v) \Big) .\end{aligned}$$

где $V(\pi)$ есть множество вершин, лежащих на пути π. Инвариант корректного управления секретами задаётся предикатом

$$\mathcal{S}(\pi) = \bigwedge_{v \in V(\pi)} \mathrm{MNT}(v,\Sigma).$$

Критерий прохождения этапа конвейера [209; 210] формулируется как

$$\mathcal{I}(\pi_1) \wedge \mathcal{S}(\pi_1) \wedge L(\pi_1) \leqslant \Delta_{\mathrm{inf}} \quad \wedge \quad \mathcal{I}(\pi_2) \wedge \mathcal{S}(\pi_2) \wedge L(\pi_2) \leqslant \Delta_{\mathrm{db}}.$$

Если хотя бы один из предикатов не выполняется, конфигурация не проходит этап CI. В этом случае причина отказа фиксируется по конкретному ребру или вершине графа, а не описывается как общая ошибка развёртывания.

Оценивание $L(e)$ выполняется активной проверкой. Для HTTP-ребра отправляется контрольный запрос с минимальным телом. Для gRPC используется метод проверки состояния сервиса. Для JDBC устанавливается соединение и выполняется выражение вида SELECT 1. Учётные данные берутся из пространства секретов Σ, связанного с сервисом модели и контроллером БД. В приложении В приведены декларативные утверждения для unit-тестов конфигурации.

Для воспроизводимой проверки задаются контрольные значения параметров. Пусть $\Delta_{\text{inf}} = 200\,\text{ms}$ для пути π_1 и $\Delta_{\text{db}} = 500\,\text{ms}$ для пути π_2. Эти значения выбираются по характеристикам канала и требованиям к времени реакции ИТС. Результаты измерений регистрируются в системе отслеживания, привязываясь к номеру ребра и идентификатору сборки. Для каждого ребра e сохраняется пара $(\overline{L}(e), \operatorname{Var}[L(e)])$. Она позволяет отслеживать деградации и оценивать вероятность выхода задержки за порог.

Для проверки прав доступа выполняется проверка равенства множеств

$$\text{Scopes}(\text{model-sa}) = \{\text{call ControllerDB}\}$$

и

$$\text{Scopes}(\text{controller-db-sa}) = \{\text{call SystemDB}\}.$$

Такой тест исключает появление лишних полномочий у сервисных учётных записей. Для mTLS проверяется цепочка сертификатов из пространства Σ [208]. Предикат $\text{TLS}(e)$ считается истинным только при отсутствии ошибок проверки даты, имени узла и подписи, а также при соответствии набора шифров политике безопасности предприятия.

Маршрутизация внутри кластера проверяется отдельно. Для службы должен существовать ClusterIP, а конечные точки должны быть выбраны на основе меток соответствующего развертывания. Формально

$$\text{EP}(\text{model}) \iff \exists\, p \in \text{Pods таких, что } \text{labels}(p) \supset \text{selector}(\text{Service}_{\text{model}}).$$

Это условие предотвращает ситуацию, когда служба Kubernetes существует, но не связана ни с одним из подов. Аналогичные проверки применяются к контроллеру базы данных и пограничному прокси.

Сквозная верификация трактуется как доказательство существования допустимого пути с ограничением задержки. Для π_1 и π_2 утверждение имеет вид

$$\begin{gathered}\big(\exists\,\pi_1:\ \mathcal{I}(\pi_1)\ \wedge\ L(\pi_1)\leqslant\Delta_{\mathrm{inf}}\big)\ \wedge\\ \big(\exists\,\pi_2:\ \mathcal{I}(\pi_2)\ \wedge\ L(\pi_2)\leqslant\Delta_{\mathrm{db}}\big)\end{gathered}$$

Если критерий выполняется, конфигурация допускается к следующему этапу поставки. Если критерий не выполняется, выпуск блокируется до исправления связности, прав доступа, TLS-конфигурации или превышения задержки на одном из путей.

Интеграция с моделями ИТС

В ИТС реляционная БД чаще всего хранит события датчиков, наблюдения по участкам дорог и справочники инфраструктуры. Пусть $\mathrm{EVT}(e,t,s,v)$ содержит агрегаты скорости и интенсивности, а $\mathrm{CFG}(e,a)$ задаёт атрибуты дорожного сегмента. Вектор признаков рассчитывается как

$$\mathcal{F}(e,t)=\Phi\big(\mathrm{EVT}[t-h+1,t],\mathrm{CFG}(e,\cdot)\big)$$

с оконной агрегацией по интервалу длины h. Результат инференса записывается в таблицу $\mathrm{PRED}(e,t,\hat{y})$ с указанием версии модели или хэша параметров θ. Метрики качества, например средняя ошибка на маршруте, сохраняются в $\mathrm{MET}(r,\tau,\mathrm{mae})$ и используются при ретроспективной оценке работы сервиса.

Доступность повышается репликацией контроллера и граничного прокси. При предположениях, принятых в (4.6), доступность ансамбля из r реплик вычисляется по формуле (4.6), а r выбирается из условия $A_r\geqslant A_{\min}$. В журналах сохраняются параметры запросов, коды ошибок, объёмы переданных данных и оценка μ для каждого этапа. Эти данные передаются в систему мониторинга. По ним выявляются деградации, строятся профили пропускной способности и настраивается обратное давление между контроллером и сервисом модели.

Таким образом, вариант с внешней реляционной БД целесообразен там, где у заказчика уже есть рабочие реестры, журналы и витрины данных. Потоковый HTTP подходит для больших выборок и ограниченной памяти. gRPC удобен при умеренном объёме данных и требовании передать результат одним

вызовом. Оба режима совместимы с пакетной записью через JDBC, что снижает стоимость транзакций и делает задержку более предсказуемой.

4.1.3 Исполнение модели с использованием GPU в ИТС

GPU используется в тех сценариях ИТС, где инференс модели содержит тяжёлые матричные операции, свёртки или обработку батчей пространственно-временных данных. Это относится к видеоаналитике, графовым моделям прогноза движения и классификации событий по плотным потокам признаков. В таком контуре графический процессор выполняет наиболее затратную часть вычислений, а CPU сохраняет функции подготовки признаков, маршрутизации, сериализации и обмена с внешними сервисами [209; 210]. Поэтому GPU рассматривается не как отдельная архитектура ИТС, а как ускоряющий ресурс внутри общего контура исполнения модели.

Топология взаимодействия компонентов представлена на рисунке 4.3. Внешняя система передаёт входной вектор в проксирующий компонент. Далее запрос поступает в сервис модели, где вычислительная часть выполняется на GPU. Технические журналы и метрики исполнения передаются в подсистему мониторинга. Межсервисные вызовы выполняются по защищённым каналам с взаимной аутентификацией.

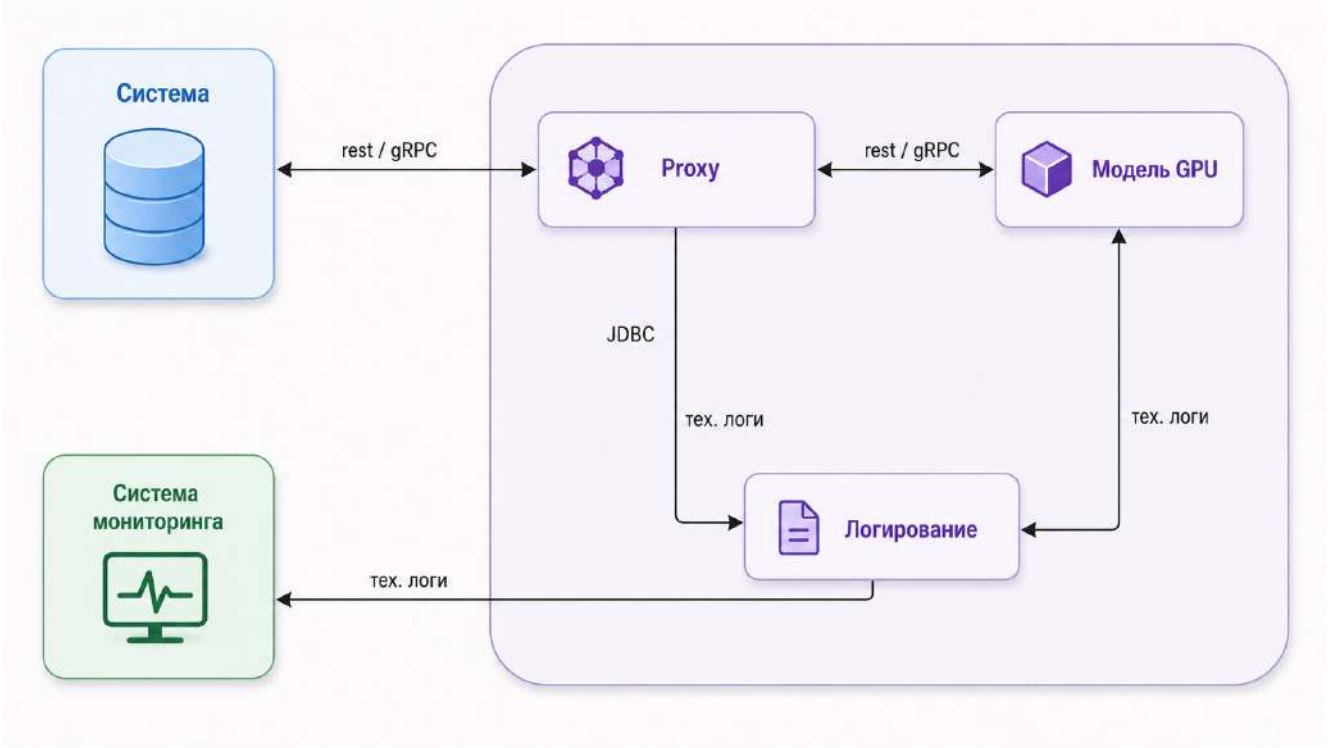

Рисунок 4.3 — Архитектура исполнения модели на GPU в контуре ИТС

Для оценки задержки используется потоковая схема исполнения. Пусть входные наблюдения образуют последовательность $\{x_t\}_{t\geqslant 0} \subset \mathbb{R}^d$ [202; 203]. На маршрутизаторе несколько событий могут объединяться в микропакет размера b. Среднее время его накопления обозначается W_b. После этого выполняются три операции: передача данных из оперативной памяти узла в память устройства $T_{\mathrm{h2d}}(b)$, вычисление ядра модели $T_{\mathrm{kern}}(b)$ и возврат результата $T_{\mathrm{d2h}}(b)$. Если эти стадии выполняются последовательно, задержка микропакета равна

$$T_{\mathrm{naive}}(b) = W_b + T_{\mathrm{h2d}}(b) + T_{\mathrm{kern}}(b) + T_{\mathrm{d2h}}(b)\,.$$

Если используются независимые CUDA-потоки и двойная буферизация, часть стадий можно перекрыть по времени. Пока один микропакет обрабатывается ядром модели, следующий может передаваться в память устройства, а предыдущий — возвращаться на хост. В стационарном режиме средняя стоимость обработки одного микропакета тогда аппроксимируется величиной

$$T_*(b) \approx W_b + \max\{T_{\mathrm{h2d}}(b),\, T_{\mathrm{kern}}(b),\, T_{\mathrm{d2h}}(b)\}\,,$$

что ведёт к эффективной скорости обслуживания

$$\mu_{\mathrm{gpu}}(b) = \frac{b}{T_*(b)}\,.$$

Размер микропакета нельзя увеличивать произвольно. Он ограничен доступной видеопамятью устройства [211]. Пусть M_V обозначает доступный объём видеопамяти, m_{mdl} — постоянную часть размещения модели, а $m_{\mathrm{act}}(b)$ — объём активаций и промежуточных буферов, зависящий от b. Условие допустимости записывается как

$$m_{\mathrm{mdl}} + m_{\mathrm{act}}(b) \leqslant M_V \,.$$

Время вычисления ядра может уменьшаться при переходе к пониженной или смешанной точности [212; 213]. Если модель допускает режим FP16, то

$$T_{\mathrm{kern}}^{\mathrm{fp16}}(b) \approx \kappa_{\mathrm{fp16}}\, T_{\mathrm{kern}}^{\mathrm{fp32}}(b)\,, \quad 0 < \kappa_{\mathrm{fp16}} < 1\,.$$

В ИТС такой эффект особенно важен для свёрточных моделей видеоаналитики и пространственно-временных графовых моделей движения. При этом переход к пониженной точности должен сопровождаться проверкой качества прогноза или классификации.

В рассматриваемой реализации модель разворачивается в Kubernetes. Сервис модели запрашивает GPU через ресурсный тип `nvidia.com/gpu`. Узлы выбираются по меткам, а параметры защищённого транспорта задаются политиками предприятия. Топология реализации показана на рисунке 4.4. В ней выделены граничный прокси, сервис модели с доступом к GPU и микросервис журналирования. Журналы и метрики могут передаваться по WebSocket, если требуется постоянный канал публикации событий мониторинга.

На рисунке 4.4 показано соответствие между логическим маршрутом обработки и его инфраструктурной реализацией. Идентификаторы вида $1.i$ относятся к функциональным этапам обработки данных. Идентификаторы вида $2.i$ описывают инфраструктурные операции, за счёт которых эти этапы выполняются в кластере. Поэтому одно ребро результирующего графа может иметь пару идентификаторов $(1.i, 2.i)$. Такая маркировка нужна для сопоставления прикладного действия, например вызова модели, с конкретными сетевыми, контейнерными и аппаратными условиями его выполнения.

Информационные потоки приведены в таблице 30. Входной путь включает передачу данных от внешней системы к прокси и последующую доставку запроса в сервис модели. Обратный путь возвращает результат через прокси инициатору. Для внешнего и внутреннего вызова допускаются REST или gRPC. Требования к защищённому транспорту сохраняются независимо от выбранного протокола.

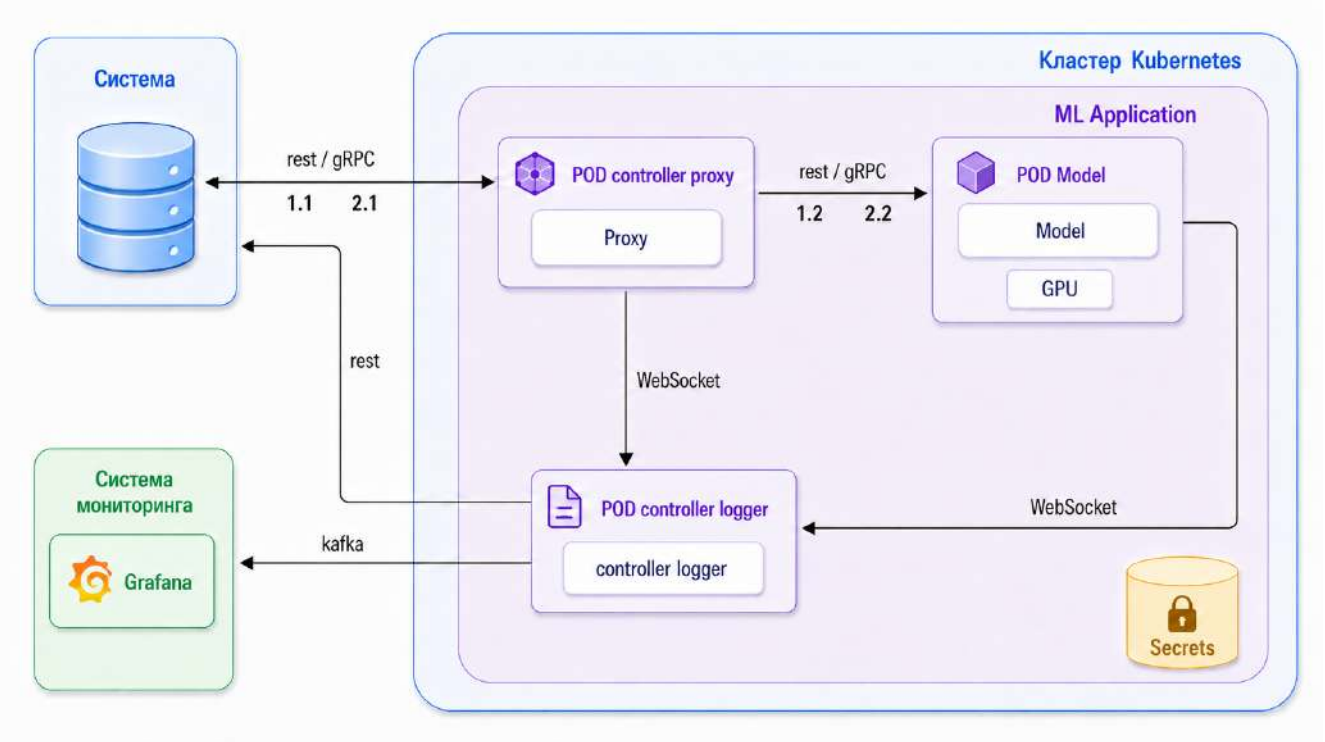

Рисунок 4.4 — Реализация исполнения модели на GPU в кластере Kubernetes

Таблица 30 — Информационные потоки взаимодействия при исполнении модели на GPU

Код	Описание взаимодействия	Инициатор	Получатели	Сущности в потоке	Механизм реализации
1	Передача данных для исполнения на GPU				
1.1	Входной вызов в ML-приложение	Система	Controller –Proxy	вектор данных	REST / gRPC (mTLS)
1.2	Доставка запроса в сервис модели	Controller –Proxy	Модель	вектор данных	REST / gRPC (mTLS)
2	Исполнение и возврат результата				
2.1	Публикация результата в прокси с терминацией TLS на границе	Модель	Controller –Proxy	вектор данных	REST / gRPC (mTLS)
2.2	Возврат результата инициатору	Controller –Proxy	Система	вектор данных	REST / gRPC (mTLS)

Производительность GPU-контура далее описывается через модель обслуживания входного потока. Пусть интенсивность входящих событий равна $\lambda(t)$,

а для исполнения используется s независимых CUDA-потоков. Если стадии передачи данных, вычисления и возврата результата хорошо перекрываются, то в стационарном режиме пропускная способность оценивается выражением

$$\mu_{\text{gpu}}^{(s)}(b) \approx \frac{s\,b}{T_{\text{fill}} + K\,\max\{T_{\text{h2d}}(b), T_{\text{kern}}(b), T_{\text{d2h}}(b)\}}\,,$$

где K — число микропакетов в серии, а T_{fill} — стоимость заполнения конвейера. Условие устойчивости очереди записывается как $\lambda(t) < \mu_{\text{gpu}}^{(s)}(b)$. Задержка ответа складывается из времени накопления микропакета и времени его прохождения через вычислительный конвейер:

$$L_{\text{gpu}}(b,s) \approx W_b + T_*(b)\,,$$

где W_b зависит от интенсивности входа и стратегии пакетизации. Для Пуассоновского входа используем приближение $W_b \approx \frac{b-1}{\lambda}$. Ограничение уровня сервиса формулируется как

$$\mathbb{P}\{L_{\text{gpu}}(b,s) \leqslant \Delta\} \geqslant 1 - \varepsilon\,.$$

Параметры эксплуатации задаются тройкой (b, s, p), где b — размер микропакета, s — число CUDA-потоков, а p — режим точности вычислений [202]. При выборе этих параметров минимизируется ожидаемая задержка. Одновременно учитываются ограничения по видеопамяти, стоимости и допустимому тепловому режиму устройства:

$$\min_{b,s,p}\ \mathbb{E}[L_{\text{gpu}}(b,s)] \quad \text{при условиях} \quad m_{\text{mdl}} + m_{\text{act}}(b) \leqslant M_V\,, \quad \mathbb{P}\{L_{\text{gpu}} \leqslant \Delta\} \geqslant 1-\varepsilon\,.$$

В ИТС GPU целесообразен прежде всего для пространственно-временных моделей скорости движения, видеоаналитики и классификации событий безопасности. Для видеоаналитики вход обычно представляется тензором $X \in \mathbb{R}^{b\times c\times h\times w}$. Для дорожных графов используется батч графов $\mathcal{B} = \{G_1, \ldots, G_b\}$, где списки смежности и признаки узлов передаются в память устройства. Для уменьшения T_{h2d} целесообразно использовать закреплённую память хоста и асинхронные копирования с явной синхронизацией барьеров. Функционально такая архитектура требует наличия GPU-узлов в кластере, стандартного конвейера развёртывания сервиса модели и отдельного канала журналирования. Технические и бизнесовые события поступают в сервис журналирования, где

они агрегируются и фильтруются по уровню важности. Для подсистемы безопасности сохраняются защищённый канал и взаимная аутентификация как при входящем вызове, так и при возврате результата во внешнюю систему.

Нефункциональные условия задают минимальные ресурсы для CPU, оперативной памяти и GPU. CPU и память нужны не только для запуска контейнера, но и для предобработки признаков, сериализации и сжатия результатов. Kubernetes выбирает узлы по меткам, квотам и заявленным ресурсам. Если платформа поддерживает изоляцию GPU, допускается выделение логических срезов устройства. Нагрузочное тестирование выполняется для нескольких значений b и p, после чего выбирается конфигурация, удовлетворяющая ограничениям по $L_{\mathrm{gpu}}(b,s)$ и $\mu^{(s)}_{\mathrm{gpu}}(b)$.

Контроль деградаций выполняется по телеметрии устройства. Пусть Θ_t обозначает вектор, включающий температуру, тактовые частоты, утилизацию вычислительных блоков и занятую память. Если $\Theta_t \notin \mathcal{A}$, нагрузка снижается или часть запросов переводится на другие экземпляры модели. Отказоустойчивость обеспечивается репликацией сервиса и наличием нескольких узлов с GPU. Для маршрутизации может использоваться детерминированное распределение по ключам потока. Это уменьшает миграцию состояния и делает поведение кэшей более предсказуемым.

GPU-контур может использовать ранее описанный механизм доступа к реляционной БД. Если модели нужны признаки из внешних источников, запросы к ним выполняются до передачи данных на устройство или в параллельном потоке, не блокирующем вычисление ядра модели. Параметры аутентификации и материалы шифрования хранятся в подсистеме секретов. Они не должны включаться в образ контейнера модели.

Верификация GPU-контура в конвейере CI

В CI проверяется не только наличие GPU, но и пригодность полного маршрута инференса. Верификация выполняется для ориентированного пути $\pi_{\mathrm{gpu}} = (1.1 \circ 1.2 \circ 2.1 \circ 2.2)$, который связывает внешнюю систему, прокси и сервис модели на устройстве. Рассматривается граф кластера $\mathcal{G} = (V,E)$ с выделенным подмножеством узлов $V_{\mathrm{gpu}} \subseteq V$ на которых доступен ресурс nvidia.com/gpu.

Для $v \in V_{\text{gpu}}$ вводятся предикаты готовности $\text{Ready}(v)$ и совместимости программного стека $\text{Compat}(v,c)$ для контейнера c. Эта проверка включает версии драйвера, CUDA, cuDNN и рантайма контейнера. Для ребра $e \in E$ задаются предикаты защищённого транспорта $\text{TLS}(e)$ и корректности вызова $\text{OK}(e)$. Свойство $\text{GPUVis}(p)$ означает, что устройство видно внутри пода p, доступно через CUDA, а переменная окружения CUDA_VISIBLE_DEVICES задана корректно.

Инвариант доступности пути в CI определяется выражением

$$\begin{aligned}\mathcal{I}_{\text{gpu}}(\pi_{\text{gpu}}) = &\Big(\bigwedge_{e \in \pi_{\text{gpu}}} \text{TLS}(e) \wedge \text{OK}(e) \Big) \\ &\wedge \Big(\bigwedge_{v \in V(\pi_{\text{gpu}})} \text{Ready}(v) \Big) \\ &\wedge \text{GPUVis}(\text{pod}_{\text{model}}) \\ &\wedge \text{Compat}(v_{\text{gpu}}, c_{\text{model}}) \,.\end{aligned}$$

Совместимость раскладывается на проверки

$$\begin{aligned}\text{Compat}(v,c) \iff &\text{Driver}(v) \geqslant v_{\min} \wedge \\ &\text{CUDA}(c) \in \mathcal{C}(\text{Driver}(v)) \wedge \\ &\text{cuDNN}(c) \in \mathcal{D}(\text{CUDA}(c)) \,.\end{aligned}$$

Если используется технология разделения ресурсов устройства, вводится предикат $\text{MIG}(v,k)$ о наличии не менее k срезов ресурса и корректном закреплении их за подом модели.

Производительность проверяется микро-бенчмарком. Он отдельно измеряет передачу данных на устройство, выполнение ядра модели и возврат результата. Для микропакета размера b фиксируются $T_{\text{h2d}}(b)$, $T_{\text{kern}}(b)$, $T_{\text{d2h}}(b)$. Полная задержка одного шага оценивается как

$$L_{\text{step}}(b) = W_b + \max\{T_{\text{h2d}}(b), T_{\text{kern}}(b), T_{\text{d2h}}(b)\},$$

где W_b обозначает время накопления микропакета. Пусть по результатам K прогонов получены наблюдения $\{L_i\}_{i=1}^{K}$. Опорное условие уровня сервиса для заданного порога Δ_{gpu} и уровня доверия α формулируется как

$$\overline{L} + z_{1-\alpha}\,\frac{s_L}{\sqrt{K}} \leqslant \Delta_{\text{gpu}} \,,$$

где $\overline{L}$ и s_L есть выборочные среднее и стандартное отклонение, а $z_{1-\alpha}$ есть квантиль стандартного нормального распределения. Условие гарантирует $\mathbb{P}\{L \leqslant \Delta_{\mathrm{gpu}}\} \geqslant 1-\alpha$ при нормальном приближении.

Для интерпретации эффективности ускорителя используется оценка пропускной способности

$$\widehat{\mu}_{\mathrm{gpu}}(b) = \frac{b}{\overline{L}},$$

которая сравнивается с целевым профилем нагрузки λ. Критерий устойчивости очереди имеет вид $\lambda < \widehat{\mu}_{\mathrm{gpu}}(b)$. При необходимости дополнительно контролируется тепловой режим устройства. Пусть Θ_t обозначает вектор телеметрии, включающий температуру, тактовые частоты и утилизацию памяти. Применяется ограничение $\Theta_t \in \mathcal{A}$, где $\mathcal{A}$ задаёт допустимое множество значений. Это ограничение снижает риск термального троттлинга под нагрузкой. В приложении Г приведены декларативные утверждения для unit-тестов конфигурации GPU.

Для обеспечения воспроизводимости фиксируются параметры испытаний. Пусть $\Delta_{\mathrm{gpu}} = 200\,\mathrm{ms}$ для микропакета размера b, а также пороговые значения $\mathrm{TH}_{\mathrm{h2d}} = 20\,\mathrm{ms}$, $\mathrm{TH}_{\mathrm{kern}} = 150\,\mathrm{ms}$, $\mathrm{TH}_{\mathrm{d2h}} = 20\,\mathrm{ms}$. Если выполняются условия $\mathcal{I}_{\mathrm{gpu}}(\pi_{\mathrm{gpu}})$ и $L_{\mathrm{step}}(b) \leqslant \Delta_{\mathrm{gpu}}$, конфигурация допускается к выпуску.

Финальный критерий этапа CI формулируется как

$$\left(\mathcal{I}_{\mathrm{gpu}}(\pi_{\mathrm{gpu}}) \wedge \overline{L} + z_{1-\alpha}\,\frac{s_L}{\sqrt{K}} \leqslant \Delta_{\mathrm{gpu}}\right) \Rightarrow$$

контур исполнения на GPU соответствует требованиям ИТС.

4.1.4 Исполнение модели с использованием Hadoop в ИТС

Hadoop-ориентированный вариант исполнения применяется для задач, где модель должна обработать крупный накопленный массив данных, а не один входной запрос. В ИТС к таким задачам относятся пересчёт архивных траекторий, построение витрин признаков, пакетный инференс по истории движения, расчёт отчётных показателей и подготовка данных для последующего обучения.

В этих сценариях важнее предсказуемое время завершения задания и устойчивость к большому объёму входа, чем миллисекундная задержка отдельного события [214—217].

В рассматриваемой схеме хранение и чтение данных выполняются через HDFS, а вычисления запускаются в Spark. Доступ к Hadoop выносится в отдельный контроллер. Он отделяет модель от деталей HDFS, WebHDFS, Hive и доменной аутентификации. Результат задания возвращается инициатору через защищённый маршрут.

Структурная схема взаимодействия сервисов представлена на рисунке 4.5. В ней выделены внешняя система, прокси, сервис модели, подсистема журналирования и контроллер доступа к Hadoop. Развёртывание в Kubernetes с участием Spark показано на рисунке 4.6. На этой схеме отдельно показаны роли Spark Master и Spark Worker, а также входной и выходной сетевые маршруты.

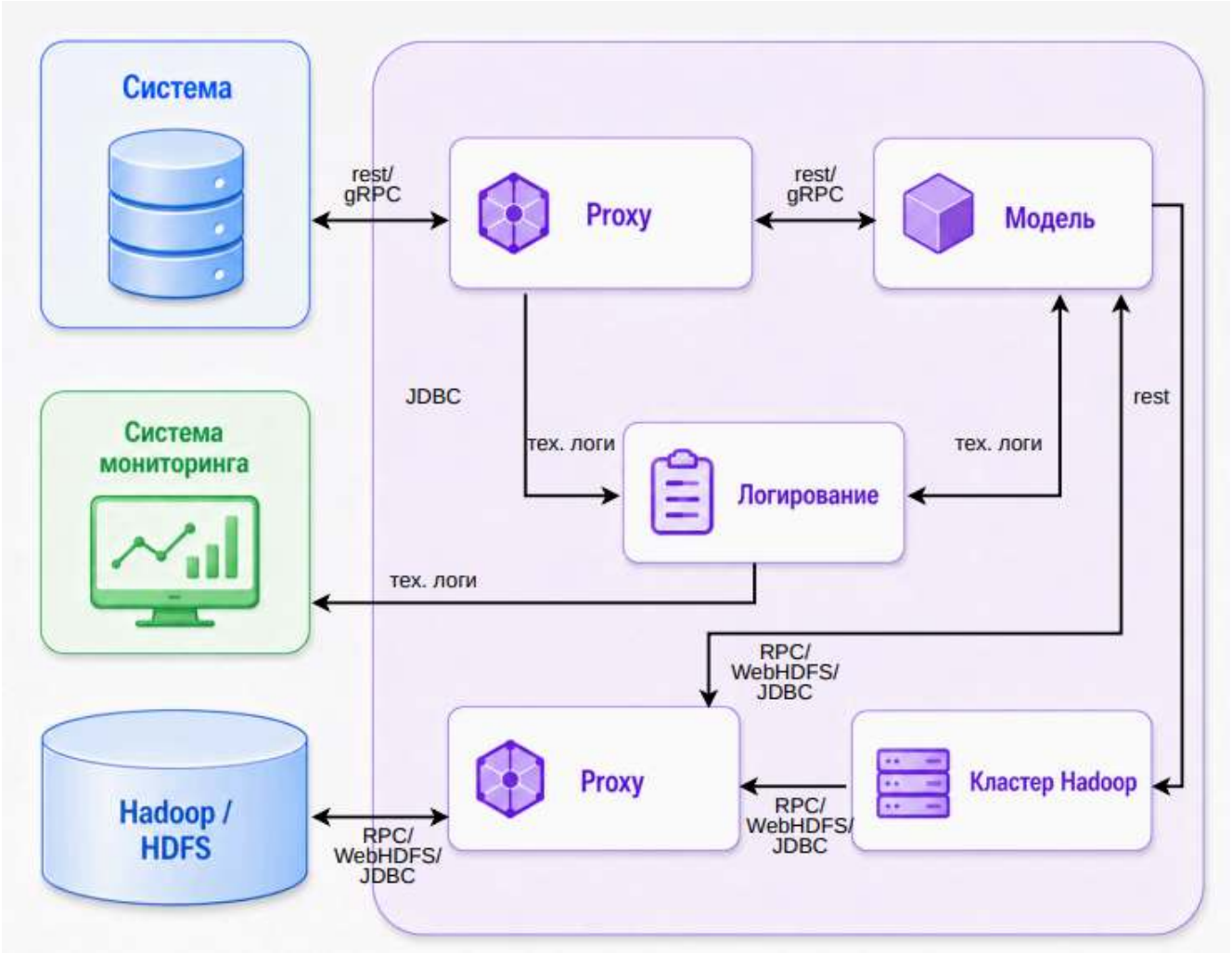

Рисунок 4.5 — Архитектура исполнения модели с интеграцией Hadoop в контуре ИТС

Задание Spark может запускаться двумя способами: по внешнему вызову или по расписанию [194]. Первый вариант удобен, когда пересчёт инициируется прикладной системой. Второй вариант используется для ночных или регламентных запусков. В Kubernetes входной запрос проходит через ingress и поступает в сервис модели. Сервис формирует параметры `spark-submit` и передаёт их в Spark Master. Чтение и запись данных выполняются через контроллер Hadoop, который использует HDFS API или WebHDFS и проходит доменную аутентификацию. Расположение логических ролей и сетевых связей в Kubernetes представлено на рисунке 4.6.

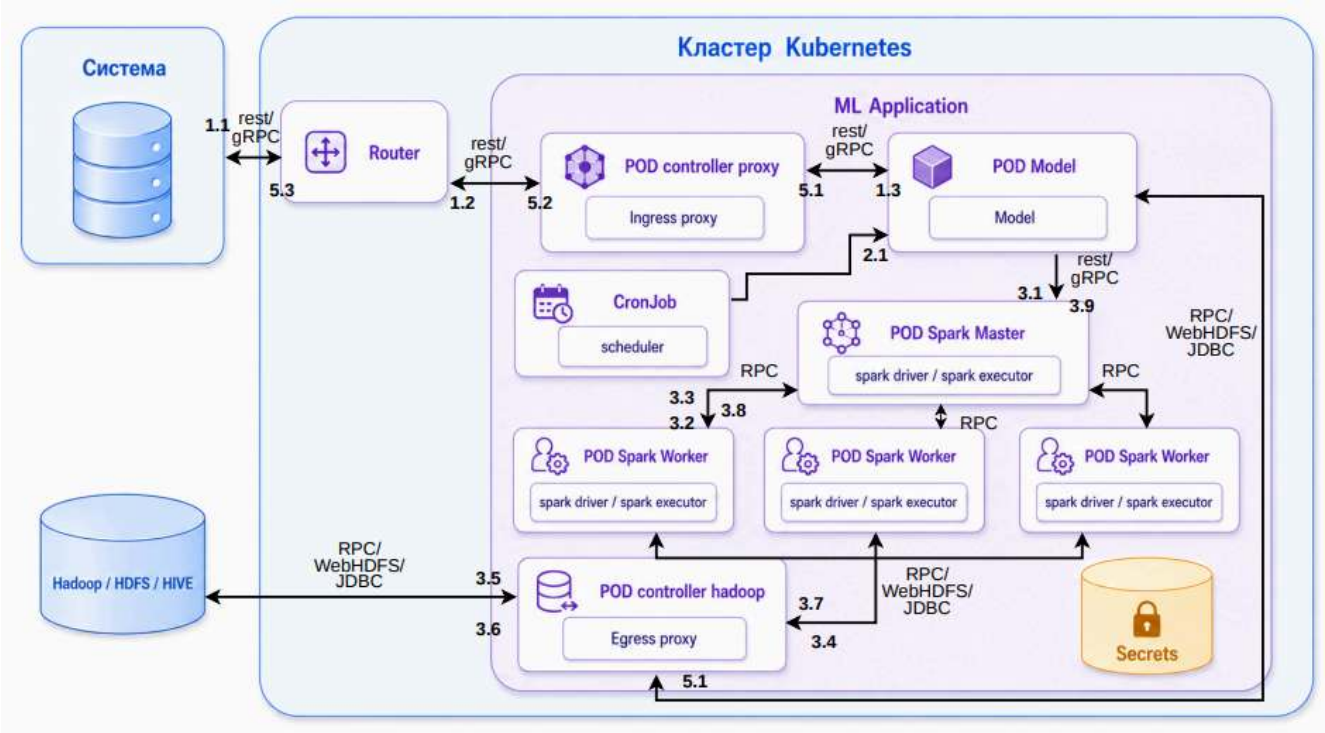

Рисунок 4.6 — Реализация исполнения модели с использованием Hadoop и Spark в кластере Kubernetes

Информационные потоки приведены в таблице 31. Их нумерация согласована с метками на рисунке 4.6. В таблицы приведены фазы передачи параметров запуска, выполнения задания Spark, обмена с HDFS или Hive и возврата результата инициатору.

Таблица 31 — Информационные потоки взаимодействия при исполнении модели с использованием Hadoop и Spark

Код	Описание взаимодействия	Инициатор	Получатели	Сущности в потоке	Механизм реализации
1	Передача данных для запуска модели				
1.1	Входной вызов с параметрами инференса	Система	Router	входной вектор	REST / gRPC (mTLS)
1.2	Доставка в контроллер прокси и терминация сертификата	Router	Controller –Proxy	входной вектор	REST / gRPC (mTLS)
1.3	Проксирование в сервис модели	Controller –Proxy	Модель	входной вектор	REST / gRPC (mTLS)
2	Запуск по расписанию				
2.1	Инициирование запуска по расписанию	Scheduler	Модель	запрос на исполнение	REST / gRPC (mTLS)
3	Исполнение Spark приложения				
3.1	Отправка команды `spark- submit`	Модель	Spark Master	параметры запуска Spark	RPC
3.2	Инициализация процессов driver и executors на узлах кластера	Spark Master	Spark Master / Spark Worker	метаинформация запуска	RPC
3.3	Доставка задач от driver к executors	Spark Driver	Spark Executor	метаинформация	RPC
3.4	Обращение к данным в Hadoop через контроллер	Spark Executor	Controller –Hadoop	входной / выходной вектор и промежуточные данные	RPC / WebHDFS / JDBC
3.5	Получение метаданных и чтение данных из HDFS	Controller –Hadoop	HDFS	входной вектор и промежуточные данные	RPC / WebHDFS / JDBC и Kerberos

продолжение на следующей странице

Код	Описание взаимодействия	Инициатор	Получатели	Сущности в потоке	Механизм реализации
3.6	Запись данных в HDFS	Controller –Hadoop	HDFS	выходной вектор и промежуточные данные	RPC / WebHDFS / JDBC и Kerberos
3.7	Возврат данных исполнителю	Controller –Hadoop	Spark Executor	входной вектор и промежуточные данные	RPC / WebHDFS / JDBC и mTLS
3.8	Возврат результатов задач исполнителю driver	Spark Executor	Spark Driver	выходной вектор и промежуточные данные	RPC
3.9	Возврат результата работы модели	Spark Driver	Модель	выходной вектор	REST / gRPC (mTLS)
4	Публикация результата инициатору				
4.1	Передача результата в прокси	Модель	Controller –Proxy	выходной вектор	REST / gRPC (mTLS)
4.2	Доставка результата в маршрутизатор	Controller –Proxy	Router	выходной вектор	REST / gRPC (mTLS)
4.3	Возврат результата внешней системе	Router	Система	выходной вектор	REST / gRPC (mTLS)
5	Прямые операции чтения и записи в HDFS и Hive				
5.1	Выполнение операций чтения и записи через контроллер Hadoop	Модель	Controller –Hadoop	входной / выходной вектор и промежуточные данные	REST / gRPC (mTLS)

Оценка производительности начинается с числа разделов входного набора. Пусть объём данных равен S байт, а размер блока HDFS равен B. Тогда число партиций определяется как

$$P = \left\lceil \frac{S}{B} \right\rceil .$$

Пусть в кластере запущено E исполнителей (*executors*) Spark. Каждый исполнитель имеет $C \in \mathbb{N}$ вычислительных слотов, то есть может одновременно выполнять не более C задач. Совокупная верхняя граница параллелизма

равна EC. Если партиций меньше, чем доступных слотов, часть ресурсов останется незагруженной. Поэтому эффективная степень параллелизма задаётся как $Q = \min(P,EC)$.

Для отдельной стадии s время выполнения можно оценить через число задач и доступный параллелизм:

$$T_s \approx \left\lceil \frac{P_s}{Q} \right\rceil \tau_s \,,$$

где P_s — число задач в стадии, а τ_s — средняя длительность одной задачи. Для стадии чтения из HDFS применяется оценка

$$\tau_{\text{read}} \approx \alpha \frac{B}{\beta_{\text{io}}} + \gamma \,,$$

где β_{io} обозначает эффективную пропускную способность ввода, α учитывает долю читаемых данных, а γ задаёт накладные расходы планирования. Стоимость шифрования и аутентификации обозначается δ_{sec}. Для стадий перемешивания используется отдельная оценка:

$$\tau_{\text{shf}} \approx \frac{\sigma B}{\beta_{\text{net}}} + \delta_{\text{sec}} \,,$$

где β_{net} обозначает пропускную способность сети, а σ отражает объём данных, передаваемых между исполнителями при перемешивании.

Общее время выполнения задания определяется критическим путём графа стадий Spark:

$$T_{\text{job}} \approx T_{\text{sub}} + \sum_{s \in \mathcal{C}} T_s \,,$$

где T_{sub} описывает накладные расходы запуска через `spark-submit`, а $\mathcal{C}$ обозначает последовательность стадий на критическом пути [217; 218]. Число исполнителей и рабочих узлов выбирается так, чтобы выполнялось ограничение $T_{\text{job}} \leqslant \Delta_{\text{hdp}}$.

Кластер Spark в данной архитектуре включается конфигурационно. Если пакетное исполнение не требуется, соответствующий контур может быть исключён из развёртывания. В рассматриваемом варианте Spark работает в режиме Standalone, а число рабочих узлов задаётся параметрами развёртывания.

Приложение запускается из сервиса модели, создаёт Spark-контекст и подключается к роли Master. Master управляет процессами исполнителей и

распределяет задачи. Driver координирует выполнение приложения и собирает результаты стадий. Доступ к Hadoop выполняется через контроллер, использующий HDFS API или WebHDFS. Аутентификация выполняется по доменной схеме, а ключевые материалы хранятся в подсистеме секретов. После завершения задания процессы executor останавливаются, а рабочие узлы остаются доступными для следующих запусков. Межузловой обмен внутри Spark выполняется через RPC.

Мониторинг строится на телеметрии Spark и метриках Dropwizard. Экспорт выполняется через коннекторы уровня кластера. Метрики разделяются по ролям Master, Worker, Executor и Driver. Такое разделение помогает понять, где возникла деградация: при планировании, чтении данных, выполнении задач или сборе результата. Технические и бизнесовые журналы передаются в выделенный сервис журналирования, а затем публикуются в систему мониторинга.

Нефункциональные условия задают способ развёртывания в промышленной среде. Если инфраструктура предприятия использует `DeploymentConfigs`, конфигурация должна быть совместима с этим объектом. Параметры безопасности передаются через секреты и не включаются в контейнеры модели, Spark-компонентов или контроллера Hadoop.

Верификация контура Hadoop в конвейере CI

В CI для Hadoop-контура проверяются две группы условий. Первая относится к инфраструктуре: готовность Spark и HDFS, корректность сервисных эндпоинтов, TLS и Kerberos. Вторая относится к производительности: контрольное задание должно завершаться за допустимое время. Для проверки используется путь $\pi_{\mathrm{hdp}} = (1.1 \circ 1.2 \circ 1.3 \circ 3.1 \circ 3.2 \circ 3.3 \circ 3.4 \circ 3.5 \circ 3.7 \circ 3.8 \circ 3.9 \circ 4.1 \circ 4.2 \circ 4.3)$.

Для этого задаются предикаты готовности ролей $\mathrm{Ready}(\cdot)$, доступности сервисов $\mathrm{EP}(\cdot)$, защищённости каналов $\mathrm{TLS}(e)$, валидности Kerberos $\mathrm{KRB}(\cdot)$ и корректной отправки задания $\mathrm{OK}_{\mathrm{spark}}(\cdot)$. Инвариант доступности пути имеет

вид

$$\begin{aligned}\mathcal{I}_{\mathrm{hdp}}(\pi_{\mathrm{hdp}}) = &\Big(\bigwedge_{e \in \pi_{\mathrm{hdp}}} \mathrm{TLS}(e) \Big) \\ &\wedge \mathrm{Ready}(\mathrm{Master}) \ \wedge \bigwedge_{w \in \mathrm{Workers}} \mathrm{Ready}(w) \\ &\wedge \mathrm{EP}(\mathrm{HDFS}) \ \wedge \ \mathrm{KRB}(\mathrm{Controller\text{-}Hadoop}) \\ &\wedge \mathrm{OK}_{\mathrm{spark}}(\texttt{spark-submit}) \, .\end{aligned}$$

Для проверки производительности вводится порог Δ_{hdp}. Контрольный вход имеет размер S и разбивается на P блоков. Задаётся серия запусков длиной K. Условие прохождения этапа CI формулируется как

$$\overline{T}_{\mathrm{job}} + z_{1-\alpha} \frac{s_T}{\sqrt{K}} \leqslant \Delta_{\mathrm{hdp}} \, ,$$

где $\overline{T}_{\mathrm{job}}$ и s_T — выборочные средние и стандартные отклонения времени выполнения, а $z_{1-\alpha}$ — квантиль уровня α. В приложении Д приводятся декларативные утверждения для автоматических проверок.

Стратегия разбиения на разделы основана на размере блока HDFS. Пусть $B = 128\,\mathrm{MB}$ или $B = 256\,\mathrm{MB}$ в зависимости от политики хранилища. В этом случае P соответствует механизму локальности данных и снижает нагрузку на сеть во время чтения. Для операций объединения количество разделов может быть увеличено после фильтрации и шифрования для уменьшения неравномерности ключей. Для задач, чувствительных ко времени, используются разделы, основанные на временных интервалах в графе дорожной сети. Это обеспечивает стабильность окна в задачах ИТС.

В окончательной реализации входящий поток данных объединяется с пакетной обработкой в Spark. Внешняя система получает результат после завершения критического пути приложения Spark. Метрики и журналы доступны через телеметрию кластера. Ограничения по времени выполнения и ресурсной мощности позволяют выбирать конфигурации кластера и параметры заданий, соответствующие соглашению об уровне обслуживания (SLA) транспортного домена.

4.1.5 Исполнение модели с использованием S3 в ИТС

Итоговая реализация объединяет входной информационный поток и пакетное вычисление в Spark. Внешняя система получает результат после завершения критического пути Spark-приложения. Метрики и журналы при этом доступны через телеметрию кластера. Ограничения на время выполнения и объём ресурсов позволяют подбирать конфигурацию кластера и параметры задания с учётом SLA транспортной предметной области.

Архитектура взаимодействия компонентов представлена на рисунке 4.7. На схеме выделены внешняя система, объектное хранилище, прокси, сервис модели, контроллер хранения и подсистема журналирования. Реализация в Kubernetes с отдельными подами для прокси, модели, контроллера хранения и журналирования показана на рисунке 4.8. Информационные потоки приведены в таблице 32.

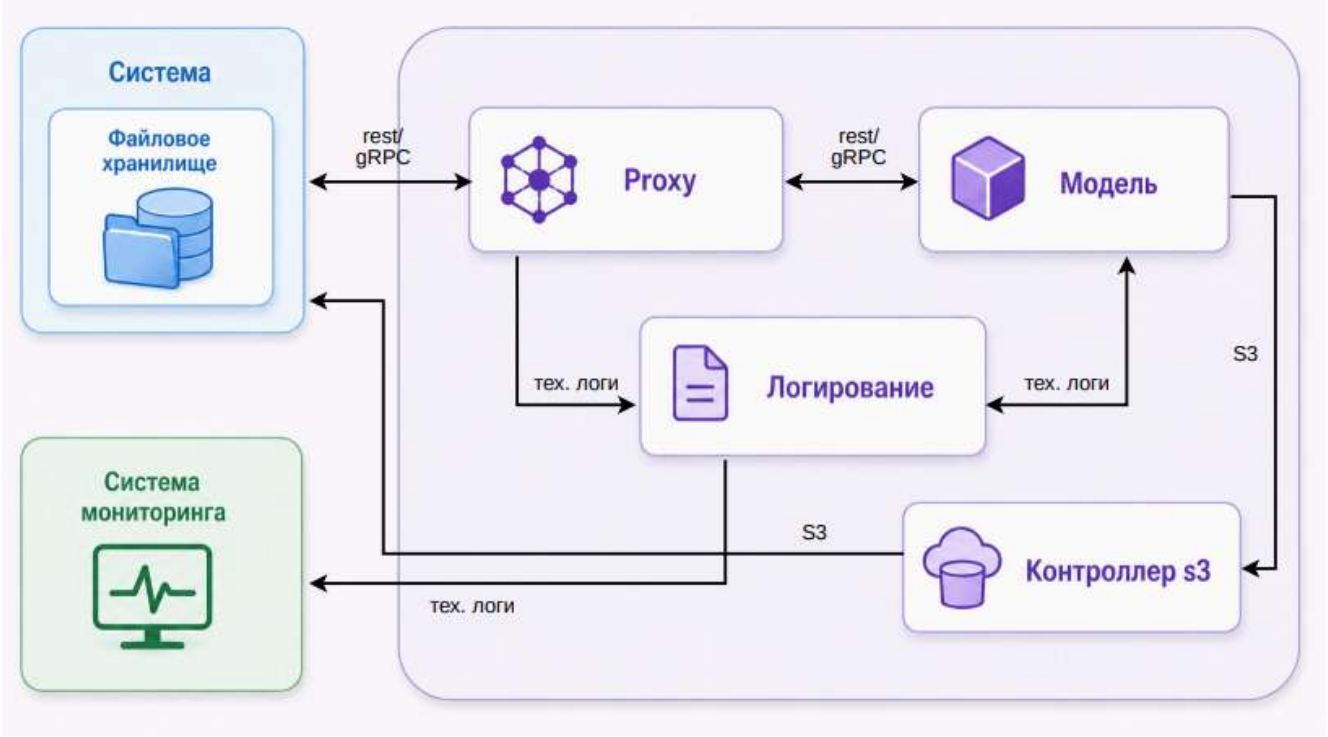

Рисунок 4.7 — Архитектура исполнения модели с использованием S3 в контуре ИТС

Производительность такого варианта определяется скоростью чтения объекта из хранилища и передачей фрагментов в сервис модели. Пусть размер

файла равен S байт. Объект читается частями размера c, а уровень параллелизма чтения равен k. Тогда число частей определяется как

$$m = \left\lceil \frac{S}{c} \right\rceil .$$

Для одной части учитываются три составляющие: время установления запроса T_{req}, время сетевой передачи c/β_{net} и накладные расходы на аутентификацию и подпись T_{sig}. Тогда время получения одной части оценивается как

$$T_{\mathrm{part}}(c) \approx T_{\mathrm{req}} + \frac{c}{\beta_{\mathrm{net}}} + T_{\mathrm{sig}} .$$

При параллельном чтении несколько частей загружаются одновременно, а затем собираются контроллером хранения. Эффективная пропускная способность в таком режиме оценивается выражением

$$\mu_{\mathrm{S3}}(c,k) \approx \min\left(\beta_{\mathrm{net}},\ \frac{k\,c}{T_{\mathrm{part}}(c)}\right).$$

Параметры c и k ограничены доступной памятью контроллера и сервиса модели. Пусть M_{ctrl} и M_{mdl} обозначают соответствующие объёмы оперативной памяти. Тогда должно выполняться условие

$$k\,c \leqslant \min(M_{\mathrm{ctrl}}, M_{\mathrm{mdl}}).$$

Сквозная задержка чтения объекта оценивается как

$$L_{\mathrm{obj}}(S; c,k) \approx \frac{S}{\mu_{\mathrm{S3}}(c,k)} + \delta_{\mathrm{proc}},$$

где δ_{proc} учитывает сериализацию и предобработку на стороне контроллера хранения. В потоковом режиме контроллер передаёт модели не весь объект сразу, а последовательность окон размера w. Для устойчивой подачи данных в буфер модели используется условие

$$\frac{w}{\mu_{\mathrm{S3}}(c,k)} \leqslant \Delta_{\mathrm{win}},$$

где Δ_{win} задаёт допустимое время поступления одного окна.

Безопасность доступа строится на двух уровнях. На границах ML-приложения используется взаимная аутентификация, а для доступа к bucket применяются клиентские ключи. Пусть Σ_{S3} обозначает набор секретов, включающий

access key, secret key и материалы шифрования. Корректность конфигурации задаётся условием

$$\mathrm{Creds}(\text{controller-storage}) \in \Sigma_{\mathrm{S3}} \quad \wedge \quad \mathrm{Scope}(\text{controller-storage}) = \{\text{read}\},$$

которое фиксирует режим только на чтение. Для сетевого канала вводится предикат $\mathrm{TLS}(e) = 1$ для всех рёбер e, включая обмены между моделью и контроллером хранения.

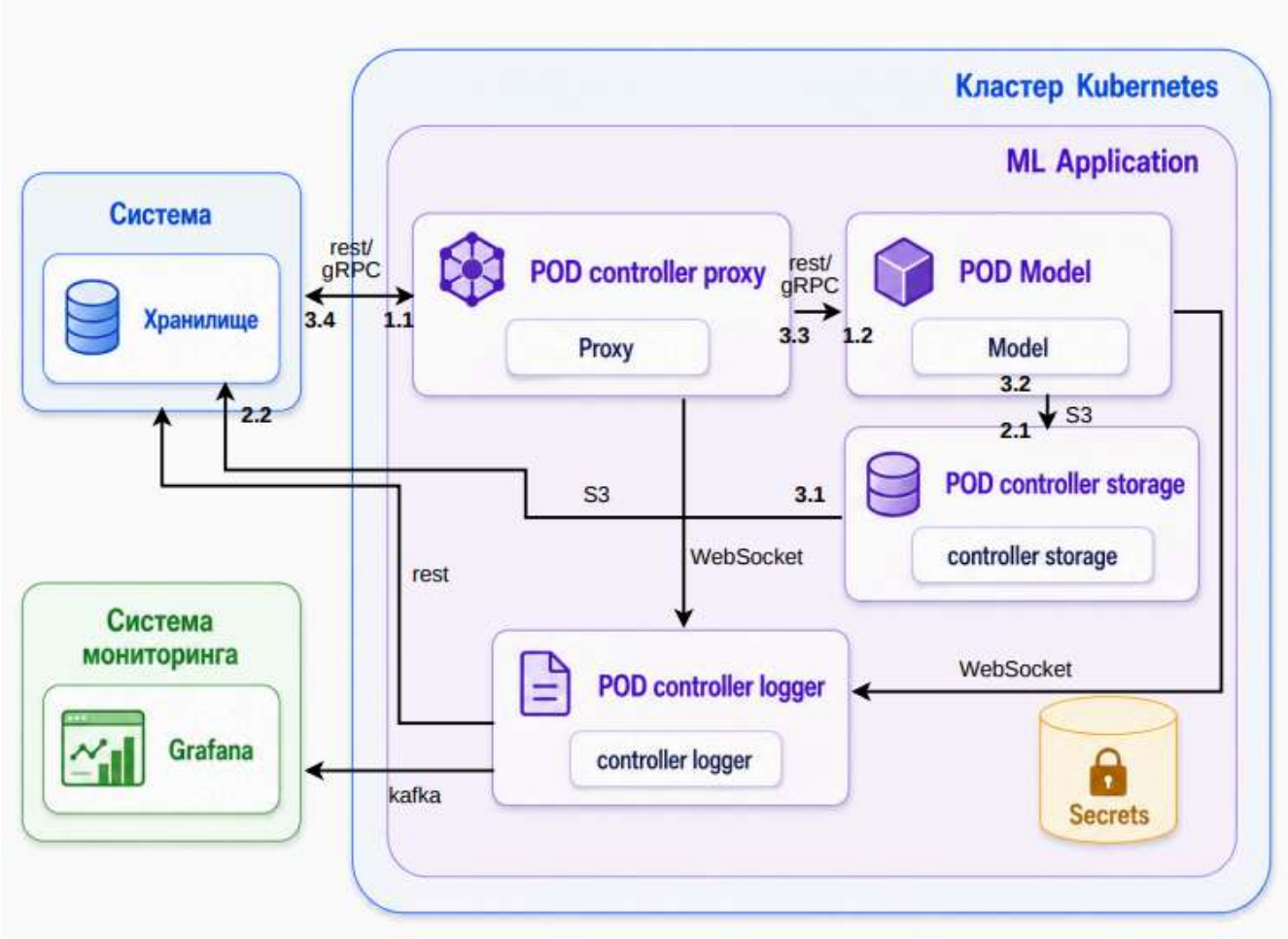

Рисунок 4.8 — Информационные потоки взаимодействия с S3 в кластере Kubernetes

Для видеоданных S3-контур используется как источник фрагментов записи. Пусть модель принимает тензор кадров $X \in \mathbb{R}^{b\times c\times h\times w}$, а объект в S3 представляет фрагмент видеозаписи размером S. Контроллер хранения разбивает видеопоток на окна длины τ, согласованные с размером батча b. Условие синхронизации записывается как

$$\frac{S_\tau}{\mu_{\mathrm{S3}}(c,k)} \leqslant T_{\mathrm{prep}},$$

где S_τ — объём данных одного окна, а T_{prep} — время подготовки признаков на CPU. Для изображений и аудио применяется тот же принцип фрагментации.

Сводная спецификация потоков взаимодействия приведена в таблице 32. Нумерация потоков согласована с обозначениями на рисунке 4.8.

Таблица 32 — Информационные потоки взаимодействия при исполнении модели с использованием S3

Код	Описание взаимодействия	Инициатор	Получатели	Сущности в потоке	Механизм реализации
1	Передача данных для запуска модели				
1.1	Входной вызов в ML-приложение	Система	Controller –Proxy	вектор данных	REST / gRPC (mTLS)
1.2	Доставка запроса в сервис модели	Controller –Proxy	Модель	вектор данных	REST / gRPC (mTLS)
2	Чтение данных из файлового хранилища				
2.1	Запрос модели на чтение по ссылке в bucket	Модель	Controller –Storage	вектор данных	S3 (mTLS)
2.2	Чтение объекта из S3 и проксирование в сторону модели	Controller –Storage	Хранилище S3	вектор данных	S3 (mTLS)
3	Возврат результата				
3.1	Передача данных от S3 в контроллер хранения	Хранилище S3	Controller –Storage	вектор данных	S3 (mTLS)
3.2	Доставка данных из контроллера хранения в модель	Controller –Storage	Модель	вектор данных	S3 (mTLS)
3.3	Публикация результата исполнения модели в прокси	Модель	Controller –Proxy	вектор данных	REST / gRPC (mTLS)
3.4	Возврат результата инициатору	Controller –Proxy	Система	вектор данных	REST / gRPC (mTLS)

Контроллер хранения включается только для тех моделей, которым требуется чтение крупных объектов из S3. В рамках ML-приложения используется один экземпляр контроллера, что упрощает управление ключами и политиками

доступа. Контроллер проксирует данные между S3 и моделью и поддерживает потоковую выдачу фрагментов. Библиотека MinIO Java Client используется как адаптер к S3. Через неё выполняются получение атрибутов объекта, чтение диапазонов и параллельная выборка частей. Режим работы контроллера ограничен операциями чтения. Расположение целевого объекта определяется по ссылке, которую внешняя система передаёт в запросе к модели. Формат данных не фиксируется заранее и определяется содержимым объекта. Технические журналы и бизнес-метрики передаются через специализированный сервис журналирования.

Нефункциональные условия задают границы интеграции с объектным хранилищем. Модель обращается к контроллеру хранения через внутрикластерный адрес типа ClusterIP. Учётные данные хранятся в Kubernetes Secrets и предоставляются администраторами проекта. Клиентская запись S3 включает пару ключей доступа и идентификатор пользователя, используемые для входа в bucket. Настройка учётных данных для области вызывающей системы выполняется владельцем, после чего параметры конфигурируются в секретах кластера. Тип объекта развёртывания контроллера согласуется с эксплуатационной практикой и соответствует DeploymentConfigs.

Размер части c и уровень параллелизма k выбираются с учётом памяти и допустимой задержки. Пусть C задаёт верхнюю границу на объём буферов, а Δ обозначает допустимую задержку чтения объекта. Тогда задача настройки записывается как

$$\min_{c,k} \; L_{\mathrm{obj}}(S; c,k) \quad \text{при условиях} \quad kc \leqslant C \;, \quad \mathbb{P}\{L_{\mathrm{obj}} \leqslant \Delta\} \geqslant 1 - \varepsilon.$$

Для сценариев ИТС с онлайн-видеорядом дополнительно выбирается окно τ и вводится ограничение на дрожание выдачи $\mathrm{Jitter}(\tau) \leqslant J_{\max}$. Это ограничение нужно для стабильной подачи кадров в конвейер исполнения.

Верификация взаимодействия с S3 в конвейере CI

В CI для S3-контура проверяются конфигурация доступа и производительность чтения. Рассматривается путь $\pi_{\mathrm{S3}} = (1.1 \circ 1.2 \circ 2.1 \circ 2.2 \circ 3.1 \circ 3.2 \circ 3.3 \circ 3.4)$.

Для него задаются предикаты готовности сервисов Ready$(\cdot)$, наличия эндпоинтов EP$(\cdot)$, защищённости каналов TLS(e), валидности секретов Creds$(\cdot)$ и доступности bucket BucketOK$(\cdot)$. Инвариант доступности имеет вид

$$\begin{aligned}\mathcal{I}_{\mathrm{S3}}(\pi_{\mathrm{S3}}) = &\Big(\bigwedge_{e \in \pi_{\mathrm{S3}}} \mathrm{TLS}(e) \Big) \\ &\wedge \mathrm{Ready}(\text{controller-storage}) \\ &\wedge \mathrm{EP}(\text{controller-storage}) \\ &\wedge \mathrm{Creds}(\text{controller-storage}) \in \Sigma_{\mathrm{S3}} \\ &\wedge \mathrm{BucketOK}(\text{bucket}) \\ &\wedge \mathrm{Scope}(\text{controller-storage}) = \{\text{read}\}\,.\end{aligned}$$

Производительность проверяется микро-бенчмарком на объекте размера S с параметрами c и k. По K прогонам вычисляются средняя задержка $\overline{L}_{\mathrm{obj}}$ и дисперсия s^2. Критерий прохождения этапа CI при пороге Δ_{S3} и уровне доверия α задаётся как

$$\overline{L}_{\mathrm{obj}} + z_{1-\alpha}\frac{s}{\sqrt{K}} \leqslant \Delta_{\mathrm{S3}}.$$

Целостность чтения проверяется по контрольной сумме. Пусть ETag обозначает хэш объекта в S3, а $\mathrm{MD5}_{\mathrm{loc}}$ — локально вычисленную сумму. Если объект не был загружен многокомпонентно, используется условие ETag $\equiv \mathrm{MD5}_{\mathrm{loc}}$. Если $m > 1$, применяется функция сверки частей. В приложении Е приведены декларативные утверждения для автоматических проверок.

Если задержка превышает порог Δ_{S3}, корректируются размер части c и уровень параллелизма k. При этом сохраняются ограничения по памяти и доступной пропускной способности сети.

4.1.6 Исполнение модели с использованием In-Memory БД в ИТС

In-Memory БД используется в тех случаях, когда сервису модели требуется быстрый доступ к состояниям, признакам и промежуточным результатам. В рассматриваемой реализации такой слой построен по принципам In-Memory Data Grid на базе Apache Ignite [219]. Для ИТС это важно при каскадном исполнении моделей, повторном использовании признаков дорожных сегментов и

хранении текущих состояний симуляторов или ограничений. Доступ к данным ускоряется за счёт размещения ключей в оперативной памяти, локализации партиций на узлах и кэширования часто используемых значений. Внутрикластерный обмен выполняется по защищённым каналам и остаётся внутри границ приложения.

Архитектура взаимодействия представлена на рисунке 4.9. В ней выделены внешняя система, граничный прокси, сервисы моделей, слой журналирования и кластер In-Memory БД. Этот кластер используется как общая оперативная память для быстрых чтений и записей. Реализация в Kubernetes показана на рисунке 4.10. На схеме отдельно обозначены координатор кластера, ведомые узлы, входной прокси и выходной прокси уровня приложения.

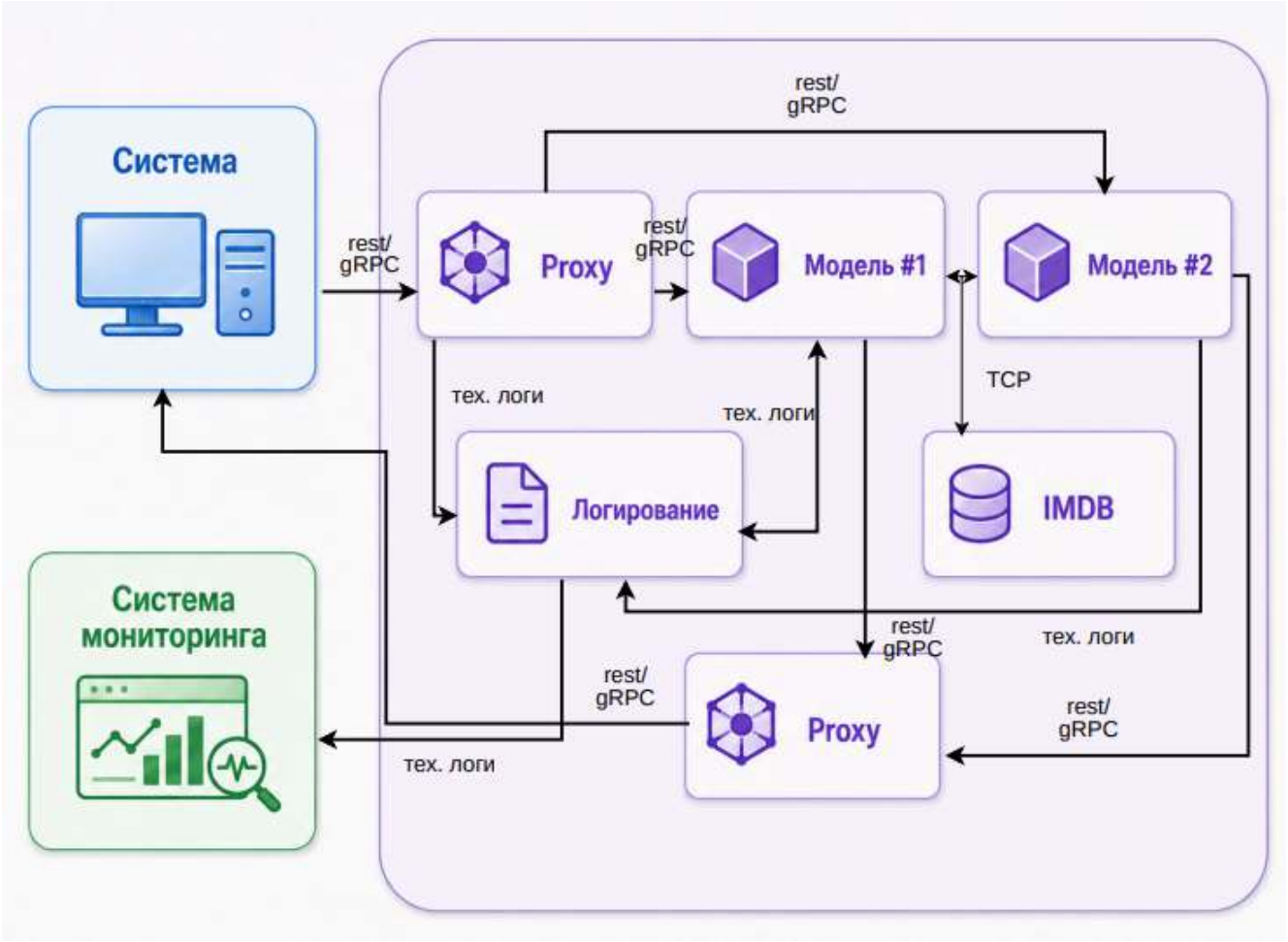

Рисунок 4.9 — Архитектура исполнения моделей с использованием In-Memory БД в контуре ИТС

Основные информационные потоки сведены в таблицу 33. Входной запрос проходит через прокси приложения и передаётся в сервис модели. Если модели нужны признаки, состояние или промежуточный результат, она обращается

к кластеру In-Memory БД по TCP с взаимной аутентификацией. Ответ модели возвращается инициатору через тот же граничный прокси. Для внешней автоматизированной системы допускается доступ через тонкий клиент и специализированный контроллер. Такой клиент не входит в состав кластера и не хранит партиции данных.

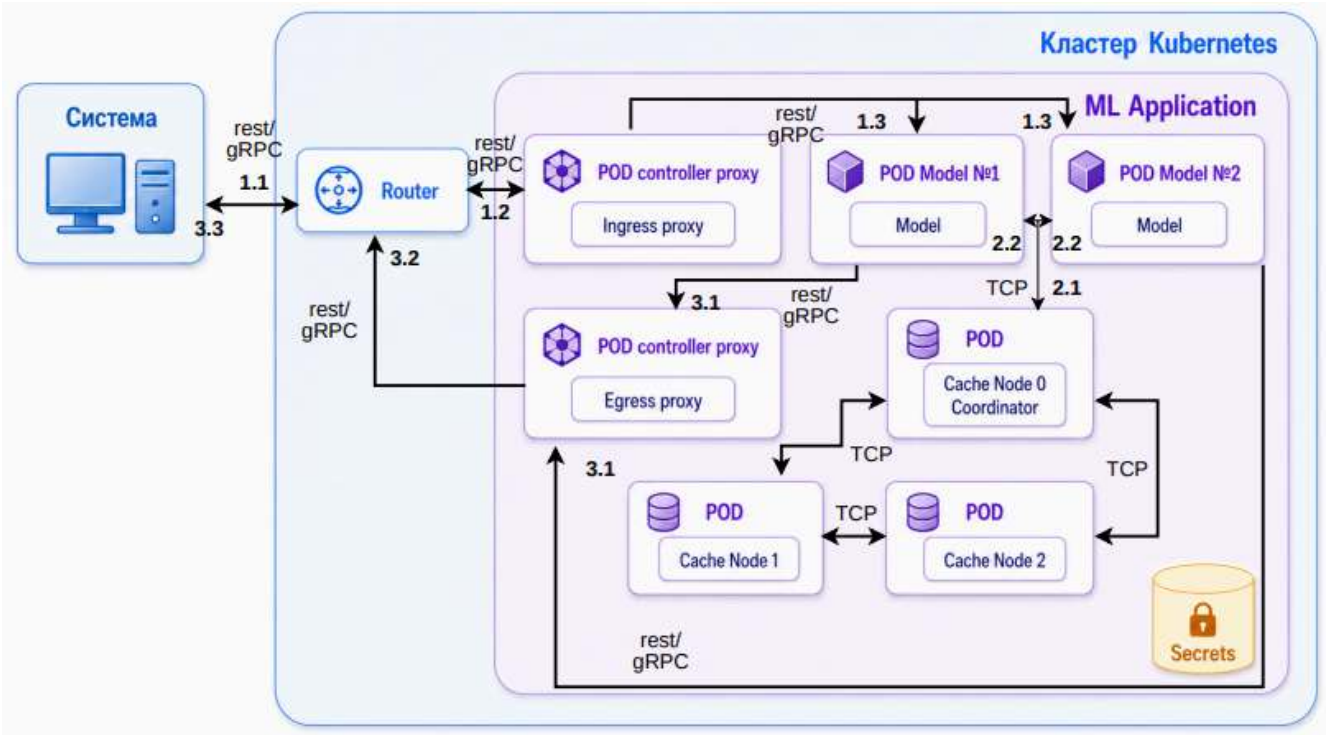

Рисунок 4.10 — Реализация кластера In-Memory БД в Kubernetes с координатором и ведомыми узлами

Таблица 33 — Информационные потоки взаимодействия при исполнении модели с использованием In-Memory БД

Код	Описание взаимодействия	Инициатор	Получатели	Сущности в потоке	Механизм реализации
1	Передача данных для исполнения модели				
1.1	Входной вызов с параметрами инференса	Система	Router	вектор данных	REST / gRPC (mTLS)
1.2	Доставка в граничный прокси и терминация сертификата	Router	Controller –Proxy	вектор данных	REST / gRPC (mTLS)

продолжение на следующей странице

Код	Описание взаимодействия	Инициатор	Получатели	Сущности в потоке	Механизм реализации
1.3	Проксирование запроса в сервис модели	Controller –Proxy	Модель	вектор данных	REST / gRPC (mTLS)
2	Операции чтения и записи в In-Memory БД				
2.1	Доступ к кэшу и хранилищу состояний	Модель	In-Memory DB	вектор данных	TCP (mTLS)
2.2	Возврат результата операции чтения или подтверждение записи	In-Memory DB	Модель	вектор данных	TCP (mTLS)
3	Возврат результата инициатору				
3.1	Публикация результата в прокси	Модель	Controller –Proxy	вектор данных	REST / gRPC (mTLS)
3.2	Доставка результата в Router	Controller –Proxy	Router	вектор данных	REST / gRPC (mTLS)
3.3	Возврат результата внешней системе	Router	Система	вектор данных	REST / gRPC (mTLS)

Размер кластера определяется через объём данных и параметр репликации r. В Apache Ignite используется партиционированное хранение, при котором для каждой партиции создаётся $r-1$ резервных копий. Пусть n обозначает объём пользовательских данных в гигабайтах без учёта копий. На каждом узле доступно $M_{\text{node}} = 32$ ГБ оперативной памяти, а служебные расходы платформы обозначаются m_0. Тогда для числа узлов k должно выполняться условие вместимости

$$k\big(M_{\text{node}} - m_0\big) \geqslant rn\,.$$

Минимально допустимое значение вычисляется как

$$k^{\star} = \left\lceil \frac{rn}{M_{\text{node}} - m_0} \right\rceil ,$$

Эта запись задаёт минимальное число узлов с учётом копий и служебных расходов. Полный выделяемый объём памяти равен $m = k^{\star} M_{\text{node}}$. При $m_0 = 0.3$ ГБ и $r = 3$ получается оценка $k^{\star} = \left\lceil \frac{3n}{31.7} \right\rceil$. Она эквивалентна проверке неравенства $3n + km_0 \leqslant kM_{\text{node}}$, но удобнее для прямого выбора размера кластера.

Задержка чтения зависит от того, находится ли нужная партиция на том же узле, где выполняется вычисление. Пусть функция $\mathsf{h} : \mathcal{K} \to \{1,\dots,P\}$ распределяет ключи по P партициям, а отображение $\mathsf{a} : \{1,\dots,P\} \to \{1,\dots,k\}$ связывает партиции с узлами кластера. Если сервис модели размещён с учётом аффинности к партициям, доля локальных обращений возрастает. Без такой привязки при равномерном распределении ключей вероятность локального доступа приближённо равна $1/k$. Средняя задержка чтения записывается как

$$\mathbb{E}[T_{\mathrm{get}}] = p_{\mathrm{loc}}\,T_{\mathrm{loc}} + (1 - p_{\mathrm{loc}})\left(T_{\mathrm{net}} + T_{\mathrm{rem}}\right),$$

где p_{loc} есть вероятность локального доступа, T_{loc} есть время чтения из памяти процесса, T_{net} есть стоимость однохоповой передачи по TCP в кластере, T_{rem} есть время чтения на удалённом узле. Пропускная способность каждого узла определяется

$$\mu_{\mathrm{node}} \approx \frac{1}{\mathbb{E}[T_{\mathrm{get}}]}, \qquad \mu_{\mathrm{cluster}} \approx k\,\mu_{\mathrm{node}},$$

если сеть и процессорные ресурсы не становятся узким местом [220]. Для записи используется асинхронная репликация. Она уменьшает задержку, видимую клиентом, но полная консолидация копий завершается позже. Пусть T_{ack} обозначает время подтверждения клиенту, а T_{rep} описывает фоновую доставку на резервные узлы. Тогда

$$T_{\mathrm{ack}} \approx T_{\mathrm{loc}}, \qquad T_{\mathrm{full}} \approx T_{\mathrm{loc}} + (r-1)\,T_{\mathrm{net}},$$

что позволяет отделить метрику клиентской задержки от полной длительности консолидации копий. Вероятность потери данных при независимых отказах оценивается величиной $p_{\mathrm{loss}} \approx p^{r}$, где p есть вероятность отказа одного узла на временном интервале репликации.

В ИТС такой слой удобно использовать для признаков дорожных сегментов, состояний симуляторов, списков ограничений доступа и промежуточных результатов каскадных моделей. Пусть $\mathcal{F}(e,t)$ обозначает вектор признаков для сегмента e в момент t. Если для кэша задан коэффициент попаданий h, то ожидаемая задержка получения признаков имеет вид

$$\mathbb{E}[T_{\Phi}] = h\,T_{\mathrm{get}} + (1-h)\left(T_{\mathrm{src}} + T_{\mathrm{put}}\right),$$

где T_{src} описывает получение данных из внешнего источника, T_{put} есть запись в кэш. Выбор TTL и объёма партиций проводится по условию $\mathbb{P}\{T_{\Phi} \leqslant \Delta\} \geqslant 1 - \varepsilon$ при данных SLA.

Кластер использует координатор, избрание лидера и звёздную топологию связи. При отказе координатора выполняется переизбрание, после чего кворум подтверждает новую метадату. Тонкий клиент сохраняет семантику операций чтения и записи, но не хранит партиции и не участвует в продвижении журнала топологии. Поэтому модель может обращаться к кластеру напрямую, а внешняя система — через специализированный контроллер тонкого клиента. Долговечность данных внутри данного варианта обеспечивается репликацией в памяти узлов. Внешние диски в этой схеме не используются.

Мониторинг кластера включает события топологии, операции чтения и записи, ошибки репликации и метрики распределения ключей. Технические журналы передаются через выделенный сервис в систему мониторинга. Для сетевой безопасности используются взаимная аутентификация и шифрование транспорта. Доступ к сервисам ограничивается пространством имён приложения. Все обращения к In-Memory БД фиксируются в журнале аудита.

Верификация кластера In-Memory БД в конвейере CI

В CI проверяется, что кластер In-Memory БД готов к запуску модели и удовлетворяет требованиям безопасности. Пусть множество узлов $\mathcal{N}$ содержит координатор и ведомые узлы. Для проверки вводятся предикаты $\mathrm{Ready}(v)$ для подов, $\mathrm{EP}(s)$ для сервисов, $\mathrm{TLS}(e)$ для защищённых каналов, $\mathrm{Topo}(k)$ для размера кластера и $\mathrm{Repl}(r)$ для числа резервных копий. Условие доступности записывается как

$$\Big(\bigwedge_{v\in\mathcal{N}} \mathrm{Ready}(v)\Big) \wedge \mathrm{EP}(\text{imdb-coordinator}) \wedge \mathrm{TLS}(e)\ \forall e \wedge \mathrm{Topo}(k) \wedge \mathrm{Repl}(r)$$

и дополняется проверкой вместимости

$$k\big(M_{\text{node}} - m_0\big) \geqslant rn\,.$$

В приложении Ж приведены декларативные утверждения в стиле unit-проверок.

Задержка чтения и записи проверяется микро-бенчмарком. Пусть K обозначает число запросов. По измерениям $\{T_i\}_{i=1}^{K}$ вычисляются средняя $\overline{T}$ и

стандартное отклонение s. Условие SLA по задержке имеет вид

$$\overline{T} + z_{1-\alpha}\frac{s}{\sqrt{K}} \leqslant \Delta_{\text{imdb}}.$$

Оно задаёт вероятностную гарантию для операций чтения или записи при уровне доверия $1 - \alpha$. Если условие не выполняется, корректируются размещение вычислений относительно партиций, число узлов k, TTL записей и политика кэширования.

4.1.7 Обращение к модели по HTTPS/gRPC в ИТС

Синхронное обращение используется тогда, когда внешней системе нужен ответ модели в рамках одного запроса. Такой режим подходит для онлайн-прогноза, проверки состояния дорожного сегмента, оценки инцидента и других сценариев, где результат должен быть получен сразу после передачи входных признаков. Обмен выполняется по REST поверх HTTPS либо по gRPC поверх HTTP/2 с взаимной аутентификацией [221]. В текстовом варианте полезная нагрузка передаётся как JSON-представление входного вектора. В бинарном варианте используется Protobuf, что уменьшает объём сообщения и снижает затраты сериализации. Граничный прокси принимает запрос, завершает TLS-сеанс, выбирает модель по имени и передаёт вызов в сервис модели.

Архитектура обмена представлена на рисунке 4.11. Развёртывание этого варианта в Kubernetes показано на рисунке 4.12.

Сквозная задержка одного синхронного вызова складывается из сетевых, протокольных и вычислительных составляющих:

$$L_{\text{e2e}} = T_{\text{tcp}} + T_{\text{tls}} + T_{\text{hdr}} + T_{\text{body}} + T_{\text{app}} + T_{\text{resp}}\,,$$

где T_{tcp} соответствует установлению транспортного соединения, T_{tls} — рукопожатию TLS и взаимной аутентификации, T_{hdr} — обработке заголовков, T_{body} — передаче полезной нагрузки, T_{app} — вычислению модели, а T_{resp} — возврату ответа. Если соединение переиспользуется, составляющая $T_{\text{tcp}}+T_{\text{tls}}$ для последующих запросов не возникает. В HTTP/2 часть расходов на заголовки и передачу тела может перекрываться за счёт мультиплексирования потоков.

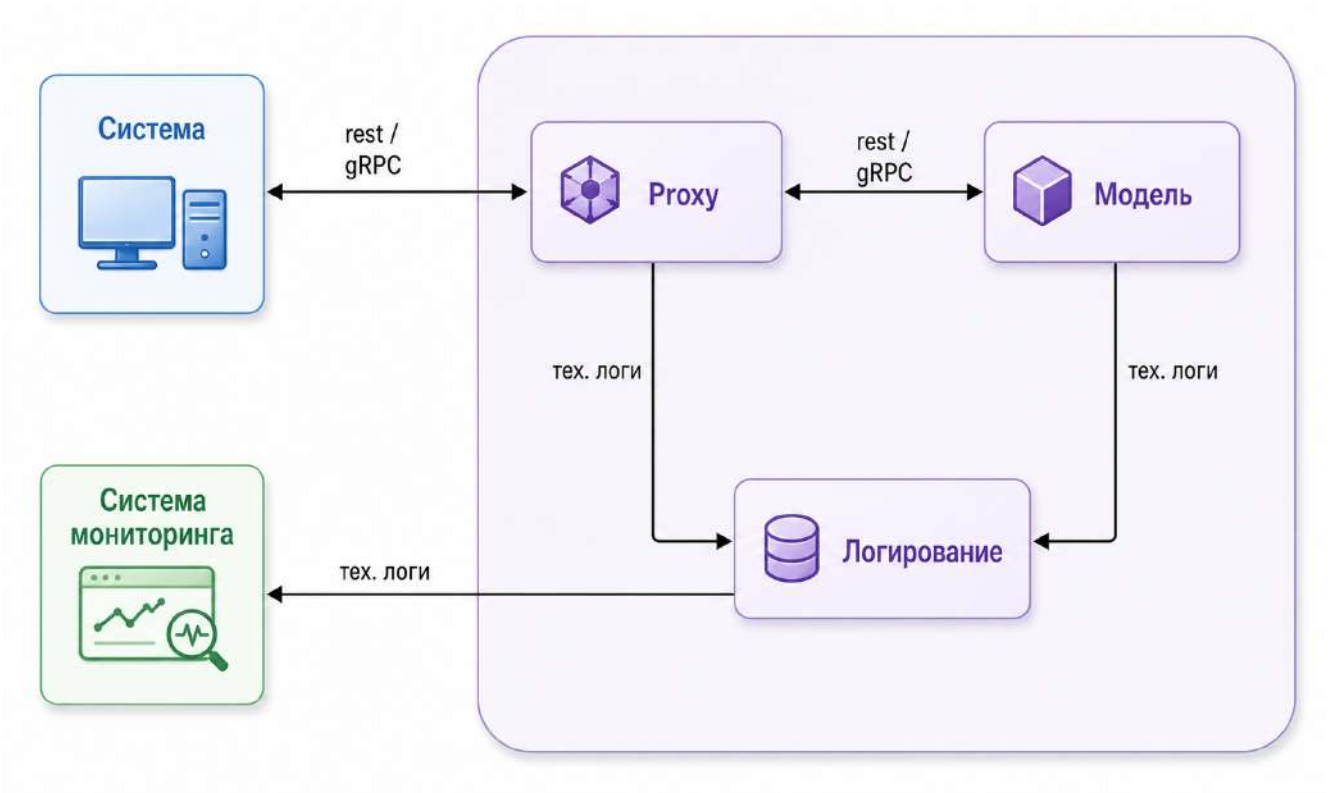

Рисунок 4.11 — Синхронное обращение к модели через граничный прокси

Транспортная часть зависит от размера полезной нагрузки и числа промежуточных обработчиков. Пусть размер запроса равен S_{in}, а размер ответа — S_{out}. Для канала с пропускной способностью β и задержками обработки на входе, в сервисе модели и на выходе путь проксирования оценивается как

$$T_{\text{net}} \approx \delta_{\text{ing}} + \frac{S_{\text{in}}}{\beta} + \delta_{\text{mdl}} + \frac{S_{\text{out}}}{\beta} + \delta_{\text{eg}}\,.$$

При JSON-сериализации размер входного сообщения обозначается $S_{\text{in}}^{\text{json}}$, а при Protobuf сериализации — $S_{\text{in}}^{\text{pb}}$. Обычно $S_{\text{in}}^{\text{pb}} < S_{\text{in}}^{\text{json}}$, поэтому бинарный формат уменьшает вклад передачи и разбора сообщения. Вычислительная часть задаётся временем инференса $T_{\text{app}} = T_{\text{inf}}$ и зависит уже от архитектуры модели и выбранной аппаратной платформы.

В gRPC тот же прикладной смысл реализуется униарным вызовом метода предсказания, где полезная нагрузка передаётся сообщением Protobuf. Для обоих вариантов задаётся одно и то же ограничение SLA:

$$\mathbb{P}\{L_{\text{e2e}} \leqslant \Delta\} \geqslant 1 - \varepsilon\,,$$

где Δ задаёт допустимую задержку для сценария ИТС, а ε фиксирует допустимую долю нарушений при пиковых нагрузках.

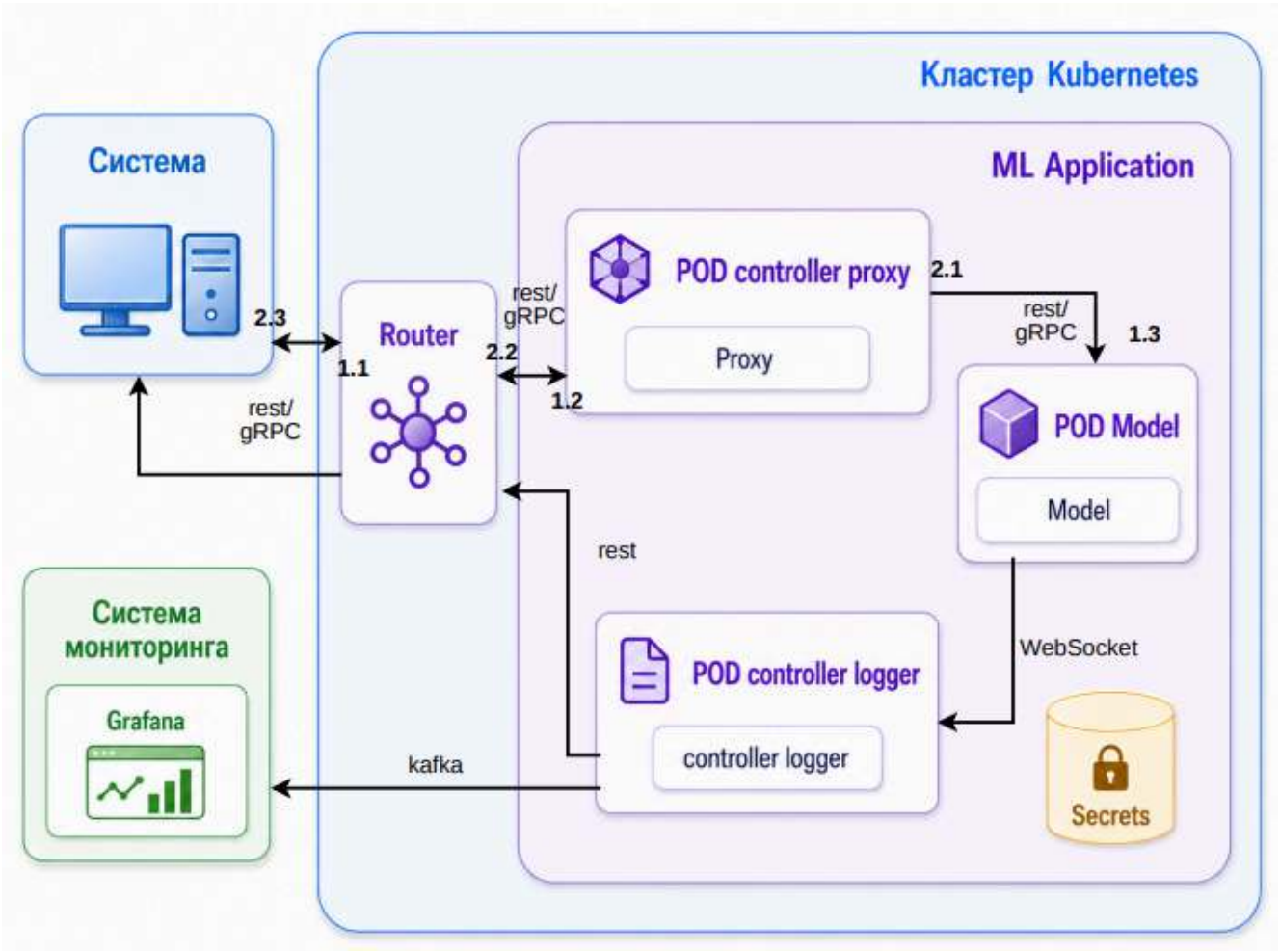

Рисунок 4.12 — Реализация синхронного доступа по HTTPS/gRPC в кластере Kubernetes

В синхронном режиме ключевыми параметрами становятся переиспользование соединений и допустимый параллелизм. Пусть на один экземпляр модели одновременно приходится m активных запросов. Тогда в устойчивом режиме средняя пропускная способность сервиса может быть оценена как

$$\mu_{\text{srv}} \approx \frac{m}{\mathbb{E}[L_{\text{e2e}}]}.$$

HTTP/2 позволяет обслуживать несколько потоков в рамках одного соединения. Поэтому рост m не всегда приводит к линейному росту сетевых накладных расходов. Для gRPC поверх HTTP/2 дополнительно задаётся минимальное число постоянных соединений на клиента, чтобы не создавать новые TLS-сеансы при каждом запросе.

Граничный прокси включается параметром конфигурации и работает как единая точка входа в ML-приложение. Он завершает TLS-сеанс, проверяет

сертификат внешней системы, использует материалы аутентификации из секретов кластера и маршрутизирует запрос по имени модели. Внутри приложения межсервисный обмен также выполняется по защищённым каналам. Формат полезной нагрузки выбирается по контракту вызова: JSON для текстового REST-варианта или Protobuf для gRPC. Технические журналы и бизнесовые метрики передаются в сервис журналирования с разделением по каналам доставки в систему наблюдаемости.

Нефункциональные ограничения фиксируют способ промышленного развёртывания. Конфигурация задаётся объектами DeploymentConfigs с декларативным описанием ресурсов, политик и переменных окружения. Сервис модели получает доступ к прокси только внутри кластера. Секреты с материалами шифрования и хранилищами доверия подготавливаются администраторами и не включаются в поставляемые артефакты модели. Контур безопасности строится на взаимной аутентификации, авторизации и шифровании трафика [44].

Описание информационных потоков согласовано с обозначениями на рисунке 4.12 и приведено в таблице 34.

Таблица 34 — Потоки взаимодействия при обращении к модели по HTTPS/gRPC

Код	Описание взаимодействия	Инициатор	Получатели	Сущности в потоке	Механизм реализации
1	Передача данных для исполнения модели				
1.1	Входной вызов в ML-приложение с полезной нагрузкой	Система	Router	вектор данных	REST / gRPC (mTLS)
1.2	Доставка запроса в граничный прокси	Router	Controller –Proxy	вектор данных	REST / gRPC (mTLS)
1.3	Проксирование в сервис модели и завершение TLS на границе	Controller –Proxy	Модель	вектор данных	REST / gRPC (mTLS)
2	Возврат результата работы модели				
2.1	Публикация результата в граничный прокси	Модель	Controller –Proxy	вектор данных	REST / gRPC (mTLS)

продолжение на следующей странице

Код	Описание взаимодействия	Инициатор	Получатели	Сущности в потоке	Механизм реализации
2.2	Доставка результата в маршрутизатор	Controller –Proxy	Router	вектор данных	REST / gRPC (mTLS)
2.3	Возврат ответа внешней системе	Router	Система	вектор данных	REST / gRPC (mTLS)

Вклад TLS зависит от того, создаётся ли новое соединение или используется уже открытый канал. Пусть T_{hs} обозначает длительность первичного рукопожатия с проверкой клиентского и серверного сертификатов. При переиспользовании соединения эта стоимость для очередного запроса не добавляется. Эффективная задержка записывается как

$$L_{\mathrm{https}} \approx T_{\mathrm{hs}}\,\mathbf{1}_{\{\mathrm{new}\}} + T_{\mathrm{net}} + T_{\mathrm{inf}}\,,$$

где $\mathbf{1}_{\{\mathrm{new}\}}$ указывает, требуется ли новое рукопожатие. Для соблюдения SLA задаётся минимальное число постоянных соединений c, удовлетворяющее ограничению

$$\frac{\lambda}{c}\,\mathbb{E}[L_{\mathrm{https}}] \leqslant 1\,,$$

где λ обозначает интенсивность входного потока. Это ограничение используется для того, чтобы нагрузка не создавала растущую очередь перед сервисом модели.

Верификация контура HTTPS/gRPC в конвейере CI

В CI проверяется, что синхронный маршрут доступен, защищён и укладывается в заданный порог задержки. Для этого рассматривается путь $\pi_{\mathrm{https}} = (1.1 \circ 1.2 \circ 1.3 \circ 2.1 \circ 2.2 \circ 2.3)$. Для него задаются предикаты Ready для подов, EP для сервисов, TLS(e) для защищённости каналов, Secret для наличия материалов аутентификации и Route для доступности маршрута `/predict_online`.

Критерий прохождения стадии CI записывается как

$$\Big(\bigwedge_{e\in\pi_{\text{https}}} \text{TLS}(e)\Big) \wedge \text{Ready(proxy)} \ \wedge \ \text{Ready(model)}$$
$$\wedge\, \text{EP(proxy)} \ \wedge \ \text{EP(router)}$$
$$\wedge\, \text{Secret(proxy)} \ \wedge \ \text{Route(model, /predict_online)}$$
$$\wedge\, \overline{L}_{\text{e2e}} + z_{1-\alpha}\frac{s}{\sqrt{K}} \leqslant \Delta\,.$$

где $\overline{L}_{\text{e2e}}$ и s — выборочные среднее и стандартное отклонение задержки по K прогонам, Δ — порог SLA, а $z_{1-\alpha}$ — квантиль нормального распределения.

В приложении З приведены декларативные утверждения в стиле unit-проверок.

Рассмотренный вариант подходит для сценариев, где внешняя система должна получить ответ модели сразу после запроса. Для высокой частоты вызовов критичны пул постоянных соединений, HTTP/2 и контроль числа параллельных запросов. Эти параметры уменьшают долю сетевых накладных расходов и делают задержку на пути от внешней системы до модели и обратно более предсказуемой.

4.1.8 Обращение к модели через Kafka в ИТС

Асинхронное обращение используется в тех случаях, когда внешней системе не требуется держать открытое соединение до завершения инференса. Внешняя система публикует сообщение во входной топик Kafka, а результат затем появляется в выходном топике. Модель при этом работает не напрямую с инициатором запроса, а через транспортную очередь. Это удобно для потоков телеметрии, событий датчиков, результатов детекции и агрегированных прогнозов, где важны упорядочивание, буферизация и возможность повторной обработки.

Доступ к брокеру ограничивается на уровне топиков и групп потребителей. Для входного топика внешняя система получает право публикации, а для выходного — право чтения результата. Соединения с брокером выполняются по защищённому каналу с взаимной аутентификацией. Такой вариант проще

проверять в CI, поскольку права доступа, наличие топиков и параметры TLS можно тестировать независимо от кода модели.

Архитектура асинхронного обмена представлена на рисунке 4.13.

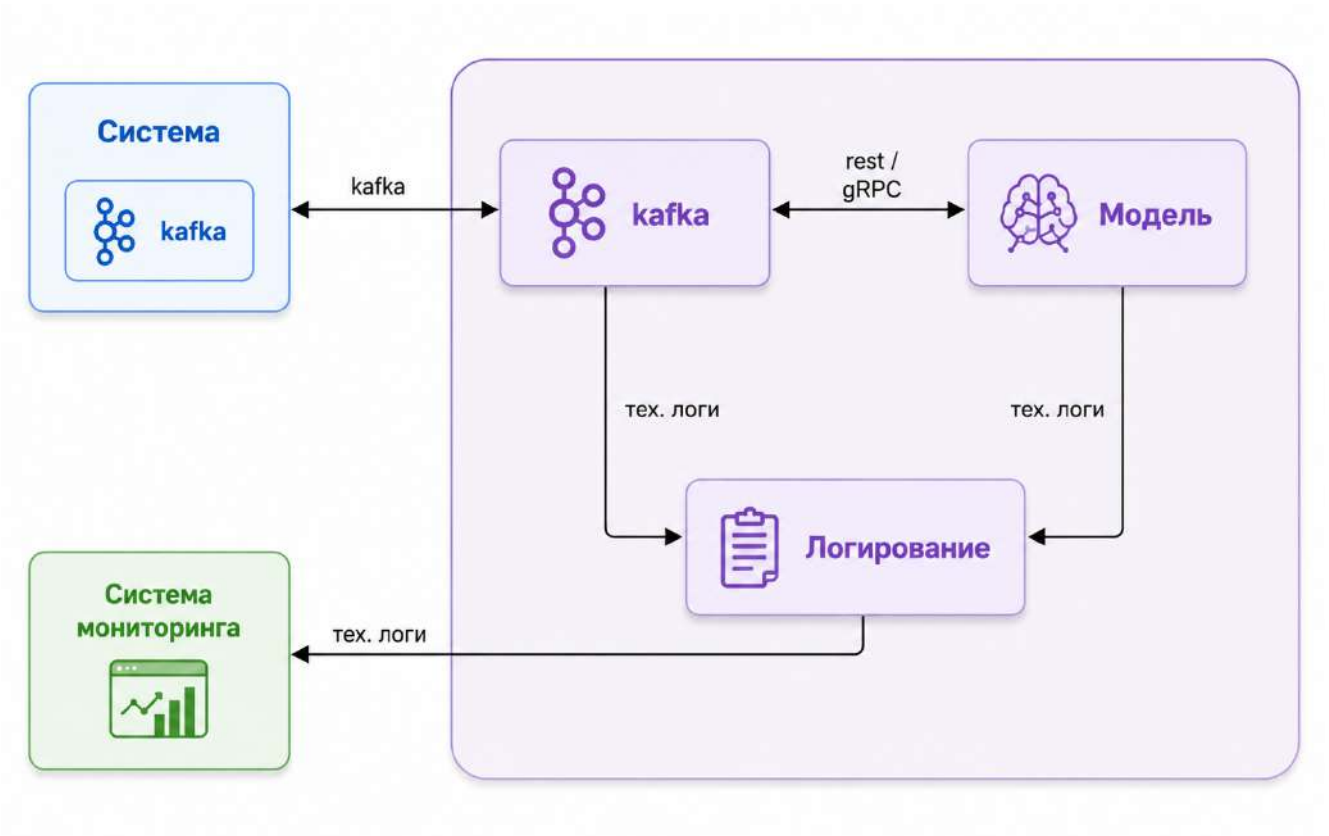

Рисунок 4.13 — Архитектура асинхронного обращения к модели через Kafka

Пусть входной топик обозначается IN, а выходной топик — OUT. Число секций во входном топике равно P_{in}, число секций в выходном топике равно P_{out}. Для обоих топиков задаётся фактор репликации r_k. Сервис модели читает сообщения в составе группы потребителей. Для одного пода число параллельных потоков ограничивается величиной $c \leqslant 10$. Это ограничение вводится не как свойство Kafka, а как эксплуатационный предел конкретного сервиса модели. Согласование ключей сообщений с секциями задаётся отображением $h : \mathcal{K} \to \{1, \ldots, P_{\text{in}}\}$, что обеспечивает упорядоченность по ключу внутри секции и детерминирует балансировку между потоками. Ключ сообщения используется для сохранения порядка внутри выбранной секции. Для ИТС это важно, если несколько сообщений относятся к одному дорожному сегменту, одному датчику или одному корреляционному идентификатору. В таком случае обработчик получает события в воспроизводимом порядке и может связать входной запрос с последующим ответом.

Потоковая производительность рассматривается покомпонентно. Пусть средний размер входного сообщения равен S_{in} байт, а выходного S_{out} байт. Эффективная полоса для клиента контролируется минимумом

$$\beta_{\text{path}} = \min\{\beta_{\text{cli}}, \beta_{\text{net}}, \beta_{\text{broker}}\}\,.$$

Скорость чтения по одной секции аппроксимируется выражением

$$\mu_{\text{sec}} \approx \frac{1}{T_{\text{poll}} + T_{\text{deser}} + \frac{S_{\text{in}}}{\beta_{\text{path}}}}\,,$$

где T_{poll} — средняя длительность выборки пакета записей потребителем, T_{deser} — стоимость десериализации. Эффективная скорость группы потребителей оценивается как

$$\mu_{\text{grp}} \approx \min\{P_{\text{in}}, c\}\,\mu_{\text{sec}}\,.$$

Вычислительная производительность сервиса модели определяется параметрами исполнения. Пусть средняя длительность обработки одной записи равна T_{inf}, тогда

$$\mu_{\text{mdl}} = \frac{1}{T_{\text{inf}}}\,.$$

Условие устойчивости очереди формулируется как

$$\lambda < \min\{\mu_{\text{grp}},\ \mu_{\text{mdl}}\}\,,$$

где λ — интенсивность входа. Для нескольких реплик сервиса модели с одинаковой группой потребителей выполняется линейное масштабирование по числу подов до предела P_{in}.

Сквозная задержка складывается из времени ожидания в очереди Kafka, времени передачи и вычислений

$$L_{\text{e2e}} \approx W_{\text{kaf}} + T_{\text{deser}} + T_{\text{inf}} + T_{\text{ser}} + W_{\text{out}}\,.$$

Для оценки ожидания используется приближение Кингмана для одной секции и одного потока

$$W_{\text{kaf}} \approx \frac{\rho}{1-\rho}\,\frac{c_a^2 + c_s^2}{2}\,\frac{1}{\mu_{\text{eff}}}\,,$$

где $\rho = \lambda_{\text{sec}}/\mu_{\text{eff}}$, $\mu_{\text{eff}} = \min\{\mu_{\text{sec}},\mu_{\text{mdl}}\}$, c_a и c_s — коэффициенты вариации межприходных интервалов и времени обслуживания. При пакетной выборке размером b задержка выборки уменьшается за счёт амортизации накладных расходов, что отражается в снижении T_{poll} и T_{deser}.

Надёжность доставки зависит от репликации и параметра подтверждения. При r_k и подтверждении `acks = all` сообщение считается записанным только после подтверждения реплик согласно политике брокера. В упрощённой оценке вероятность потери уменьшается с ростом фактора репликации:

$$p_{\text{loss}} \approx p^{r_k},$$

где p обозначает вероятность недоступности узла на интервале репликации. Такая оценка используется как инженерный ориентир. Она не заменяет проверку настроек брокера, подтверждений продюсера и политики восстановления после отказа. Для уменьшения риска повторной публикации результата используется идемпотентный продюсер и транзакционный режим чтения-записи. Идентификатор корреляции ξ используется как ключ сообщения. По нему связываются входной запрос, вызов модели и запись ответа в выходной топик. Если ответ имеет размер S и не должен передаваться одной записью, он разбивается на части размера c. Число фрагментов равно

$$m = \left\lceil \frac{S}{c} \right\rceil,$$

где сохраняется упорядочивание по ключу ξ и номеру фрагмента, что гарантирует корректную сборку на стороне потребителя. Для сборки результата потребителю достаточно сохранить общий ключ ξ, номер фрагмента и общее число частей. Порядок внутри секции при этом остаётся связанным с ключом сообщения.

Компонент `controller-kafka` выполняет роль адаптера между брокером и сервисом модели. Со стороны Kafka он читает входной топик и публикует ответ в выходной топик. Со стороны модели он выполняет обычный внутренний вызов по REST или gRPC. Поэтому асинхронность остаётся на границе системы, а внутри ML-приложения сохраняется простой синхронный контракт вызова модели. Задержка внутреннего вызова в этом случае определяется временем инференса и сериализации. Для крупных сообщений используется фрагментация и последующая сборка результата.

Нефункциональные требования здесь сводятся к двум проверяемым группам: защите канала и трассируемости сообщения. Подключения к брокеру выполняются с взаимной аутентификацией. Закрытые ключи, сертификаты и материалы доверия хранятся в подсистеме секретов и монтируются только в под `controller-kafka`. Для трассировки используется корреляционный идентификатор. Он передаётся в заголовках сообщений и сохраняется в ключе топика, чтобы по журналам можно было восстановить путь конкретного запроса.

Схема развёртывания в Kubernetes приведена на рисунке 4.14. В ней `controller-kafka` показан как адаптер между брокером сообщений и сервисом модели. Сводная спецификация информационных потоков приведена в таблице 35.

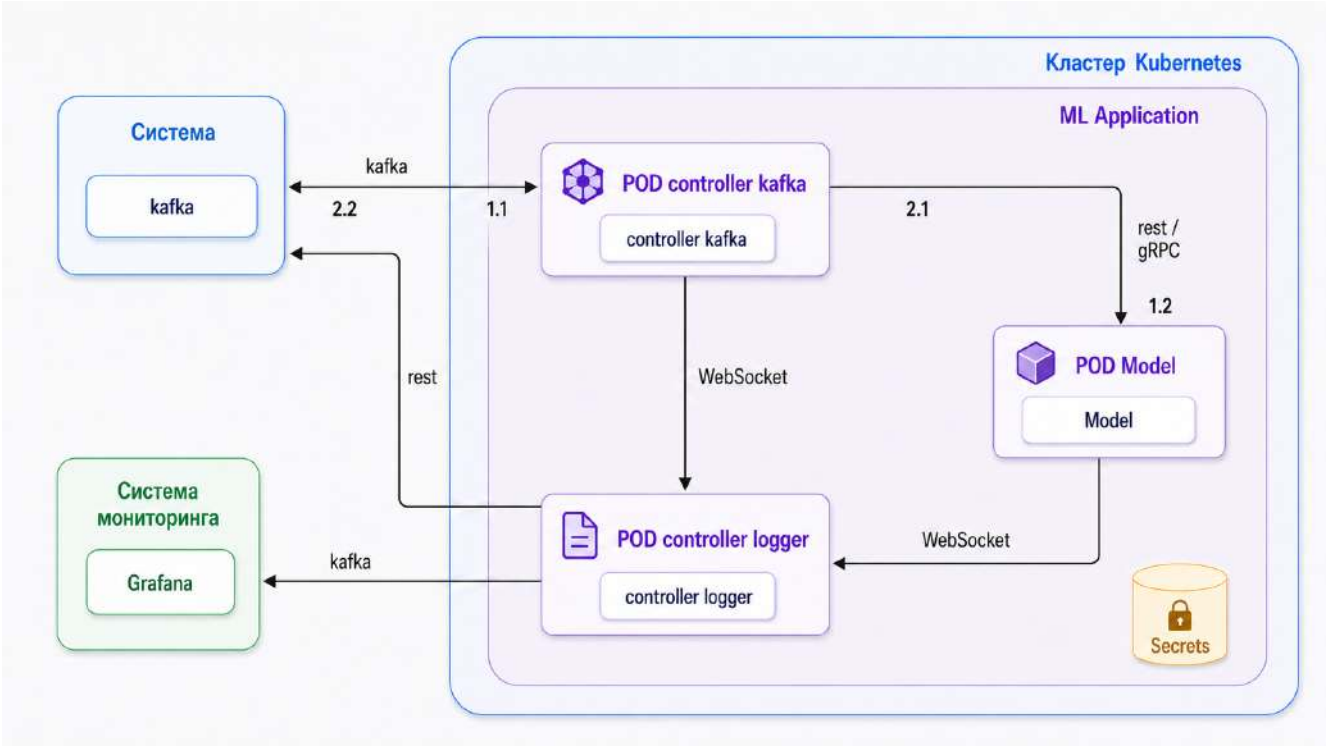

Рисунок 4.14 — Реализация контура Kafka в кластере Kubernetes

Таблица 35 — Потоки взаимодействия при обращении к модели через Kafka

Код	Описание взаимодействия	Инициатор	Получатели	Сущности в потоке	Механизм реализации
1	Подача данных для исполнения модели				

продолжение на следующей странице

Код	Описание взаимодействия	Инициатор	Получатели	Сущности в потоке	Механизм реализации
1.1	Публикация сообщения во входной топик	Система	Controller –Kafka (consumer)	вектор данных	Kafka (mTLS)
1.2	Передача полезной нагрузки в модель	Controller –Kafka (handler)	Модель	вектор данных	REST / gRPC (mTLS)
2	Возврат результата работы модели				
2.1	Получение результата от модели и подготовка записи	Модель	Controller –Kafka (handler)	вектор данных	REST / gRPC (mTLS)
2.2	Публикация ответа в выходной топик для системы	Controller –Kafka (producer)	Система	вектор данных	Kafka (mTLS)

После таблицы потоков проверяются два ограничения. Первое относится к задержке ответа. Для него используется вероятностное условие

$$\mathbb{P}\{L_{\mathrm{e2e}} \leqslant \Delta_{\mathrm{kaf}}\} \geqslant 1 - \varepsilon\,.$$

Второе ограничение относится к устойчивости потребления сообщений. Для каждой секции входной поток не должен превышать меньшую из двух величин: скорость обработки секции и скорость вызова модели.

Для устойчивой работы группы потребителей требуется ограничение

$$\lambda_{\mathrm{sec}} < \min\{\mu_{\mathrm{sec}}, \mu_{\mathrm{mdl}}\} \quad \text{на каждой секции}\,,$$

что достигается выбором числа секций P_{in}, числа подов и числа потоков на под. При фиксированном P_{in} верхняя граница масштабирования определяется равенством $\min\{P_{\mathrm{in}}, c_{\mathrm{tot}}\}$, где c_{tot} — суммарное число потоков по всем репликам сервиса.

Верификация контура Kafka в конвейере CI

В CI проверяется полный путь асинхронного вызова. Сначала контролируется готовность `controller-kafka` и сервиса модели. Затем проверяются входной и выходной топики, число секций, фактор репликации, ACL, TLS и режим идемпотентной публикации. Для проверки используется путь

$$\pi_{\text{kaf}} = (1.1 \circ 1.2 \circ 2.1 \circ 2.2).$$

Для этого пути вводятся предикаты Ready, Topic, ACL, TLS и EOS. Предикат EOS относится к идемпотентности продюсера и транзакционному режиму чтения-записи.

$$\begin{aligned}
&\text{Ready}(\text{controller-kafka}) \wedge \text{Ready}(\text{model}) \\
&\wedge \text{Topic}(\mathsf{IN}, P_{\text{in}}, r_k) \wedge \text{Topic}(\mathsf{OUT}, P_{\text{out}}, r_k) \\
&\wedge \text{ACL}(\text{system}, \mathsf{IN} \to \text{write}) \wedge \text{ACL}(\text{system}, \mathsf{OUT} \to \text{read}) \\
&\wedge \text{TLS}(\text{kafka}) \wedge \text{EOS}(\text{producer}) \\
&\wedge \overline{L}_{\text{e2e}} + z_{1-\alpha}\frac{s}{\sqrt{K}} \leqslant \Delta_{\text{kaf}}\,.
\end{aligned}$$

В приложении И приведены декларативные утверждения в стиле unit-проверок.

Kafka-контур целесообразно использовать там, где входной поток может приходить неравномерно, а результат модели допустимо получать через выходной топик. Такой вариант подходит для событий датчиков, асинхронных уведомлений о результатах детекции и доставки агрегированных прогнозов. Его преимущество состоит не в минимальной задержке отдельного запроса, а в управлении очередью, повторной обработке сообщений и масштабировании потребителей по секциям.

4.2 Разработка методологии построения архитектуры исполнения моделей искусственного интеллекта в ИТС

В параграфе 4.1 были последовательно рассмотрены несколько вариантов исполнения моделей ИИ в составе ИТС: реляционная БД, GPU, Hadoop/Spark,

S3, In-Memory БД, HTTPS/gRPC и Kafka. На практике эти варианты различаются не только используемой технологией. Меняется сам путь прохождения запроса: где выполняется чтение данных, где возникает очередь, где происходит сериализация, какой компонент отвечает за безопасность и в какой точке измеряется задержка. Поэтому один и тот же сервис модели в разных схемах получает разные эксплуатационные свойства. [33]

Из этого следует ограничение, которое существенно для проектирования ИТС. Выбор архитектуры исполнения нельзя сводить к названию платформы или к скорости отдельного узла. Проверять нужно полный маршрут: источник входного события, транспортный адаптер, сервис модели, доступ к данным, запись результата и передачу технических метрик. Если этот маршрут не выделен как самостоятельный объект анализа, то задержка, доступность, права доступа и отказоустойчивость оцениваются по отдельности и могут не совпасть на сквозном пути. [32]

Далее маршрут исполнения описывается в графовой форме. Это нужно для того, чтобы явно отделить вычисление модели от передачи данных, доступа к хранилищам, работы брокера и журналирования. Вершины в такой записи соответствуют сервисам, контроллерам, брокерам, хранилищам и вычислительным узлам. Рёбра задают конкретные каналы обмена, для которых можно измерять задержку, пропускную способность и требования к защищённому соединению. После этого инварианты связности и безопасности, а также ограничения по задержке и доступности задаются уже не для абстрактной архитектуры, а для конкретного маршрута ИТС. [2]

Методология рассчитана на несколько конфигураций исполнения, а не на один заранее выбранный вариант. Это позволяет сравнивать конфигурации по одинаковым признакам таким как состав маршрута, место появления очереди, используемые ресурсы, проверяемые инварианты и параметры масштабирования. [28; 35] Для такой записи используется ориентированный мультиграф $\mathcal{G} = (V,E)$, в котором вершины обозначают сервисы модели, контроллеры доступа к данным, транспортные адаптеры и элементы наблюдаемости, а рёбра описывают типизированные каналы. [6] Для ребра e фиксируются задержка T_e, скорость обслуживания μ_e, протокол обмена и требования к защищённому каналу. Для вершины v задаются вычислительная способность μ_v, доступная память и политика масштабирования. Сценарий задаётся входным процессом $\lambda(t)$, размером и типом полезной нагрузки, а также требованиями к реакции сервиса.

Сквозной путь $\pi \subseteq E$ задаёт один конкретный маршрут обработки запроса. В него попадают сетевые переходы, вызовы сервисов, обращения к контроллерам и операции адаптеров. В инженерной проверке такой путь удобен тем, что для него можно явно указать, где появляется задержка ожидания, где выполняется передача данных и где возникает вычислительная обработка. Если стадии идут последовательно и перекрытие не используется, верхняя оценка задержки пути задаётся формулой

$$L(\pi) \leqslant \sum_{e \in \pi} T_e + \sum_{v \in V(\pi)} W_v. \tag{4.8}$$

Для конвейерного исполнения отдельно учитывается этап заполнения. Он возникает, когда первые сообщения уже прошли начальные стадии, но весь маршрут ещё не вышел на устойчивый темп обработки. Для серии из K сообщений суммарное время обработки оценивается как

$$T_{\mathrm{tot}}(K) \approx T_{\mathrm{fill}} + \frac{K}{\mu_{\mathrm{bottleneck}}}, \tag{4.9}$$

что приводит к стационарной задержке одной записи

$$L_{\mathrm{steady}} \approx \frac{1}{\mu_{\mathrm{bottleneck}}} + \sum_{v \in V(\pi)} W_v. \tag{4.10}$$

Для полного маршрута пропускная способность определяется не средним значением по компонентам, а самой медленной стадией. Поэтому используется минимум по рёбрам и вершинам:

$$\mu(\pi) = \min\Big(\min_{e \in \pi} \mu_e, \min_{v \in V(\pi)} \mu_v \Big). \tag{4.11}$$

Для эксплуатационного режима важна не только расчётная пропускная способность, но и запас до насыщения. В ИТС входная интенсивность меняется неравномерно, поэтому конфигурация должна выдерживать пиковый интервал без роста очередей. Это требование задаётся коэффициентом загрузки ρ:

$$\sup_{t \in \mathcal{T}_{\mathrm{peak}}} \lambda(t) \leqslant \rho\, \mu(\pi), \qquad 0 < \rho < 1, \tag{4.12}$$

либо через перцентильную оценку $\lambda_q \leqslant \rho\, \mu(\pi)$ на уровне q в интервале пиковой нагрузки [222]. После проверки устойчивости задаётся ограничение по задержке:

$$\mathbb{P}\{L(\pi) \leqslant \Delta\} \geqslant 1 - \varepsilon. \tag{4.13}$$

Синтез конфигураций архитектуры выполняется после того, как задан маршрут и известны его контролируемые пути. На этом этапе выбирается не отдельная технология, а набор параметров, определяющих размещение сервисов, параллелизм обработки, режим передачи данных и резервирование. Конфигурация должна одновременно проходить проверки безопасности, доступности, памяти и задержки. Набор управляемых параметров записывается как

$$x = \big(n, c, b, k, P, r, h, p\big),$$

где n обозначает число реплик сервисов модели, c — суммарную параллельность потоков обработки, b — размер микропакета или окна, k — уровень параллелизма при чтении крупных объектов, P — число секций в топиках брокера, r — коэффициенты репликации для памяти и хранилищ, h — класс аппаратной платформы, p — режим вычислительной точности. Стоимость ресурсов задаётся функцией $C(x)$, а доступность обозначается $A(x)$. Задача выбора конфигурации формулируется как многокритериальная оптимизация:

$$\min_{x,\ \Pi} \ J(x,\Pi) = \alpha_1 \, \mathbb{E}\left[\max_{\pi \in \Pi} L(\pi)\right] + \alpha_2 \, C(x) + \alpha_3 \, E(x), \tag{4.14}$$

при ограничениях

$$\mathbb{P}\{L(\pi) \leqslant \Delta_\pi\} \geqslant 1 - \varepsilon_\pi \quad \forall \pi \in \Pi, \tag{4.15}$$

$$A(x) \geqslant A_{\min}, \tag{4.16}$$

$$M(x) \geqslant M_{\text{need}}, \tag{4.17}$$

$$\sup_{t \in \mathcal{T}_{\text{peak}}} \lambda(t) \leqslant \rho \, \mu(\pi) \quad \forall \pi \in \Pi. \tag{4.18}$$

Формулы (4.14)–(4.18) определяют пространство поиска. Конфигурация может быть рассмотрена далее только в том случае, если она одновременно удовлетворяет ограничению SLA, требованию доступности, условию памяти и запасу стабильности относительно входной нагрузки.

Далее, оценки задаются отдельно для разных типов маршрутов. Для конвейера на графическом процессоре сначала определяется этап, который фактически определяет скорость обработки микропакета:

$$T_{\text{st}}(b) = \max\big\{T_{\text{h2d}}(b),\, T_{\text{kern}}(b),\, T_{\text{d2h}}(b)\big\}, \tag{4.19}$$

что даёт стационарную пропускную способность

$$\mu_{\text{gpu}}^{\max}(b,s) = \min\left\{\frac{s\,b}{T_{\text{st}}(b)}\ ,\ \frac{\beta_{\text{pcie}}}{\eta}\ ,\ \frac{\beta_{\text{mem}}}{\eta}\right\}. \tag{4.20}$$

В этой записи выделены два существенных ограничения. Первое связано с конвейером обработки — а именно, с количеством потоков s, размером микропакета b и временем выполнения самого медленного этапа. Второе уже не связано с моделью, а с аппаратной средой: связью через PCIe и доступом к памяти устройства. Поэтому увеличение s или b выгодно только до тех пор, пока передача данных не начнет доминировать. Параметр η определяет объем данных на элемент микропакета.

Для Kafka-контура расчёт начинается с одной секции входного топика. Это важно, поскольку порядок сообщений сохраняется именно внутри секции, а не во всём топике. Скорость чтения одной секции оценивается как

$$\mu_{\text{sec}} \approx \frac{1}{T_{\text{poll}} + T_{\text{deser}} + \frac{S_{\text{in}}}{\beta_{\text{path}}}}, \tag{4.21}$$

после чего определяется скорость группы потребителей:

$$\mu_{\text{grp}} \approx \min\{P_{\text{in}},\, c_{\text{tot}}\}\, \mu_{\text{sec}}. \tag{4.22}$$

Величина $\min\{P_{\text{in}},c_{\text{tot}}\}$ показывает практическое ограничение масштабирования. Если потоков обработки больше, чем секций, часть потоков не получает самостоятельной работы. Если секций больше, чем потоков, часть секций ожидает освобождения обработчика.

При неравномерном входном потоке дополнительно оценивается ожидание сообщения в секции. Для этого используется приближение Кингмана [197]

$$W_{\text{kaf}} \approx \frac{\rho}{1-\rho} \cdot \frac{c_a^2 + c_s^2}{2} \cdot \frac{1}{\mu_{\text{eff}}}, \tag{4.23}$$

где $\rho = \lambda_{\text{sec}}/\mu_{\text{eff}}$, а $\mu_{\text{eff}} = \min\{\mu_{\text{sec}},\mu_{\text{mdl}}\}$. Здесь учитывается, что задержка определяется не только брокером, но и скоростью вызова модели. Поэтому оценки (4.21)–(4.23) проверяются совместно с условием устойчивости (4.12).

Для S3-контура сначала оценивается не скорость самого хранилища, а полоса всего пути чтения. В этот путь входят сетевой интерфейс, клиентская библиотека, серверная сторона хранилища и дисковая подсистема:

$$\beta_{\text{path}} = \min\{\beta_{\text{nic}},\, \beta_{\text{cli}},\, \beta_{\text{srv}},\, \beta_{\text{disk}}\}, \tag{4.24}$$

Для одной части объекта размера c учитываются накладные расходы запроса и время передачи данных:

$$T_{\text{part}}(c) \approx T_{\text{oh}} + \frac{c}{\beta_{\text{path}}}, \tag{4.25}$$

Сквозное время чтения объекта размера S зависит от выбранного размера части и числа параллельных чтений:

$$L_{\text{obj}}(S; c,k) \approx T_{\text{conn}} + \frac{S}{\min\left\{\beta_{\text{path}},\ \frac{k\,c}{T_{\text{part}}(c)}\right\}} + \delta_{\text{proc}}. \tag{4.26}$$

При этом параллельное чтение ограничено буферами контроллера и сервиса модели:

$$k\,c \leqslant \min(M_{\text{ctrl}}, M_{\text{mdl}}). \tag{4.27}$$

Формулы (4.24)–(4.27) показывают, что параметры c и k нельзя выбирать независимо. Увеличение размера части снижает долю накладных расходов, но требует большего буфера. Увеличение k ускоряет чтение только до тех пор, пока не достигается одно из ограничений пути или памяти.

Для реляционной БД различаются два режима передачи результата. Потоковый режим уменьшает задержку первого фрагмента и снижает ожидание в очередях в (4.8). Буферный режим, напротив, полезен при умеренном объёме выборки, поскольку сокращает число сетевых обменов между сервисом модели и БД.

Для In-Memory БД главным фактором становится локальность доступа. Если нужная партиция находится на том же узле, где выполняется сервис модели, чтение близко к локальной операции. Если партиция размещена на другом узле, к задержке добавляются сетевой переход и обработка удалённого запроса. Поэтому такой слой повышает пропускную способность пути только при достаточной доле локальных обращений и устойчивом коэффициенте попаданий в кэш.

Процедура конфигурирования начинается с паспорта сценария. В нём указываются профиль входной нагрузки, допустимая задержка, требования к доступности и ресурсные ограничения. После этого из канонических элементов собирается прототип мультиграфа. Для его путей рассчитываются оценки задержки и пропускной способности по (4.8)–(4.10) и (4.11)–(4.13). Затем выбираются параметры конфигурации через постановку (4.14)–(4.18). Полученный вариант переводится в манифесты размещения, где уже фиксируются реплики, лимиты ресурсов, параметры транспортных адаптеров и настройки доступа. Перед выпуском дополнительно проверяются инварианты безопасности и доступности с учётом ограничения (4.12).

Проверки безопасности и мониторинга привязываются к тем же путям, для которых рассчитываются задержка и пропускная способность. Для рёбер контролируется наличие защищённого канала. Для вершин проверяются готовность подов и доступность endpoint. Для подсистем доступа к данным отдельно проверяются секреты, роли и область разрешённых операций. Для маршрутов дополнительно фиксируется существование сервисных точек и корректность связей между компонентами. Эти проверки выполняются в CI вместе со статистической проверкой SLA на контрольной нагрузке. Поэтому ошибка в маршруте, секрете, TLS-настройке или endpoint выявляется до промышленного запуска, а не во время обработки реального потока.

Размещение и масштабирование задаются отдельно для каждого типа маршрута. Для обычного сервиса основным действием остаётся изменение числа реплик. Для GPU-контура вместе с числом реплик подбираются размер микропакета и число параллельных потоков, поскольку они влияют на загрузку устройства и передачу данных. Для Kafka-контура регулируются число потоков в группе потребителей и число секций входного топика. При этом сохраняется порядок сообщений по ключу. Для S3-контура настраиваются размер части и параллелизм чтения с учётом доступной памяти. Для In-Memory БД контролируются заполнение кэша и время жизни записей, так как от них зависит доля локальных обращений и попаданий в кэш.

Формальное описание разработанной методологии

Методология синтеза конфигурации архитектуры потокового исполнения моделей ИИ в ИТС задаёт порядок перехода от требований сценария к проверяемой конфигурации исполнения. В отличие от простого выбора платформы, здесь отдельно фиксируются маршрут данных, параметры вычислительных узлов, ограничения SLA и набор автоматических проверок.

1. **Формализация сценария и требований.** На первом шаге задаётся входной процесс $\lambda(t)$, множество контролируемых путей Π и пороги $\{\Delta_\pi, \varepsilon_\pi\}_{\pi \in \Pi}$ для вероятностного SLA:

$$P\{L(\pi) \leqslant \Delta_\pi\} \geqslant 1 - \varepsilon_\pi, \qquad \forall \pi \in \Pi,$$

дополнительно фиксируются требования по доступности $A_{\min}$, стоимости $C_{\max}$, памяти M_{need} и качеству $\text{Acc}_{\min}$, если для выбранной модели вводится ограничение на точность.

2. **Конфигурирование структуры мультиграфа исполнения.** По выбранному сценарию строится ориентированный мультиграф $\mathcal{G} = (V,E)$. В него включаются только те элементы, через которые реально проходит запрос или поток данных. Это могут быть реляционная БД, S3, Ignite, GPU-контур, Hadoop/Spark, HTTPS/gRPC или Kafka. После выбора элементов задаются контролируемые пути Π. Затем сервисы, контроллеры, хранилища, брокеры и каналы обмена сопоставляются вершинам и рёбрам графа.
3. **Параметризация конфигурации.** Для выбранной структуры формируется вектор параметров

$$x = (n,c,b,k,P,r,h,p),$$

где n задаёт число реплик, c — параллельность обработки, b — размер микропакета или окна, k — параллелизм чтения, P — число секций Kafka, r — коэффициенты репликации, h — класс аппаратной платформы, а p — режим вычислительной точности [202; 203; 210]. Такая параметризация нужна для сравнения вариантов исполнения по управляемым настройкам, а не по названиям используемых технологий.
4. **Оценивание пропускной способности и задержки.** Для каждого пути $\pi \in \Pi$ рассчитываются пропускная способность $\mu(\pi)$ по (4.11) и задержка $L(\pi)$ по (4.8)–(4.10). Если в путь входят GPU, Kafka, S3 или БД, дополнительно применяются специализированные оценки (4.19)–(4.27). На этом шаге проверяется не отдельный сервис, а весь маршрут прохождения сообщения с учётом передачи данных, ожидания, вычисления и доступа к хранилищу.
5. **Проверка устойчивости по нагрузке.** Для пикового интервала $\mathcal{T}_{\text{peak}}$ проверяется, остаётся ли входной поток ниже допустимой загрузки маршрута. Запас задаётся коэффициентом $\rho \in (0,1)$:

$$\sup_{t \in \mathcal{T}_{\text{peak}}} \lambda(t) \leqslant \rho\, \mu(\pi), \qquad \forall \pi \in \Pi,$$

это условие не допускает работы пути у предела пропускной способности. При отсутствии буфера даже небольшие колебания входного

потока приводят к увеличению очередей и задержек в конце отрезка времени.

6. **Выбор параметров как задача оптимизации.** После расчёта задержек, пропускной способности и устойчивости выполняется выбор параметров x и набора путей Π. Этот выбор рассматривается как многокритериальная задача. Целевая функция определяется в (4.14), а ограничения SLA, доступности, памяти и стабильности определяются в (4.15)-(4.18). Результатом является пара $x^\star$,$\Pi^\star$, которая определяет допустимую конфигурацию и набор путей выполнения.
7. **Переход к развёртываемой конфигурации.** На основе параметров $x^\star$,$\Pi^\star$ генерируются конфигурации, которые могут быть развернуты в производственной среде. Они включают в себя количество реплик, ограничения ЦП и памяти, правила HPA, параметры брокера и шардинга, а также настройки безопасности. mTLS, RBAC и секреты настраиваются отдельно, поскольку ошибка в этих параметрах влияет не только на доступность, но и на корректность всего пути выполнения.
8. **Верификация инвариантов в CI и нагрузочная проверка SLA.** Для каждого пути $\pi \in \Pi^\star$ в CI проверяются инварианты доступности и безопасности: Ready, TLS, RBAC, EP, HP, MNT и OK. Эти проверки показывают, что маршрут существует, его сервисы готовы, каналы защищены, права доступа заданы корректно, а необходимые точки подключения доступны. После этого выполняется статистическая проверка задержки на контрольной нагрузке. Если нарушен хотя бы один инвариант или критерий SLA, конфигурация не допускается к выпуску.
9. **Эксплуатационный контур наблюдаемости и адаптации.** После развёртывания контролируются $\lambda(t)$, $L(\pi)$, глубина очередей, доля нарушений SLA и утилизация ресурсов. Если нагрузка или задержка выходят за допустимые границы, изменяются параметры $x(t)$, например $n(t)$, $b(t)$, $c(t)$ или $P(t)$. Такая адаптация не должна нарушать инварианты безопасности и ограничения SLA. Поэтому изменение параметров рассматривается как управляемая корректировка конфигурации, а не как произвольное масштабирование.

В результате методология задаёт последовательный цикл требования — конфигурирование — оптимизация — развёртывание — верификация — эксплуатация. В этом цикле каждое требование должно быть связано с па-

раметром конфигурации и проверяемой метрикой. Поэтому результат можно проследить от исходного сценария ИТС до конкретных настроек размещения и тестов в CI.

Итоговая конфигурация получается не путём ручного выбора технологии, а через формальное описание маршрута и его параметров. Разные варианты исполнения приводятся к единому набору управляемых величин. Автоматическая проверка фиксирует, выполняются ли требования по задержке, доступности, безопасности и устойчивости. За счёт этого архитектурное решение можно воспроизвести на другом стенде и затем адаптировать под иной профиль транспортной нагрузки. городскими агломерациями с разными профилями транспортной нагрузки.

Эффективность разработанной методологии и метрики её оценки

Эффективность методологии оценивается через поведение полученной конфигурации на заданном профиле нагрузки. Для этого используются три базовые проверки: выполнение вероятностного SLA (4.13), выполнение условия устойчивости (4.12) и значение целевой функции (4.14). Сама по себе абсолютная величина этих показателей малоинформативна, поэтому сравнение выполняется относительно референсной конфигурации $x^{\mathrm{ref}}, \Pi^{\mathrm{ref}}$. Синтезированная и референсная конфигурации проверяются на одинаковых входных потоках, с одинаковыми границами SLA и по одной процедуре измерений. Далее вводятся метрики, позволяющие оценить конфигурацию x,Π по задержке, пропускной способности, стоимости, доступности и полноте автоматической проверки.

Доля запросов, уложившихся в SLA, рассчитывается по каждому контролируемому пути $\pi \in \Pi$. Пусть $L_i(\pi)$ обозначает задержку i-го запроса на этом пути. Тогда для отдельного маршрута используется оценка

$$\Xi_{\mathrm{SLA}}(\pi) = \frac{1}{N} \sum_{i=1}^{N} \mathbf{1}\{L_i(\pi) \leqslant \Delta_\pi\}, \tag{4.28}$$

а для конфигурации в целом выбирается минимальное значение по всем путям:

$$\Xi_{\mathrm{SLA}}(x,\Pi) = \min_{\pi \in \Pi} \ \Xi_{\mathrm{SLA}}(\pi). \tag{4.29}$$

Такой способ агрегации используется из-за инженерного смысла SLA. Если один из маршрутов регулярно превышает допустимую задержку, конфигурация не должна считаться успешной за счёт быстрых соседних маршрутов. Поэтому условие эффективности записывается как $\Xi_{\mathrm{SLA}}(x,\Pi) \geqslant 1 - \varepsilon_{\max}$ и согласуется с ограничением (4.13).

Пропускная способность, соответствующая SLA, указывает на максимальную входящую нагрузку, при которой маршрут остается в допустимых пределах. Здесь оценивается не скорость отдельного сервиса, а весь путь π, включая передачу, ожидание, вывод и доступ к данным. Обозначим эту величину как

$$\mu_{\mathrm{SLA}}(x,\Pi) = \sup\left\{\lambda \,:\, \mathbb{P}\{L(\pi) \leqslant \Delta_\pi\} \geqslant 1 - \varepsilon_\pi \;\; \forall \pi \in \Pi\right\}. \tag{4.30}$$

Полученное значение дополнительно проверяется с запасом по загрузке в соответствии с (4.12). Для сопоставления с референсной конфигурацией используется коэффициент

$$\Upsilon_\mu = \frac{\mu_{\mathrm{SLA}}(x,\Pi)}{\mu_{\mathrm{SLA}}(x^{\mathrm{ref}},\Pi^{\mathrm{ref}})}. \tag{4.31}$$

Для задержки отдельно рассматривается хвост распределения. Среднее значение может оставаться приемлемым даже тогда, когда часть запросов заметно выходит за допустимое время ответа [196; 197]. Поэтому для каждого пути используется перцентиль q:

$$\Theta_q(\pi) = Q_q(L(\pi)), \tag{4.32}$$

где Q_q есть эмпирическая квантиль уровня q. На уровне конфигурации выбирается наибольшее значение среди контролируемых путей:

$$\Theta_q(x,\Pi) = \max_{\pi\in\Pi} \; \Theta_q(\pi), \tag{4.33}$$

а выигрыш относительно референсной конфигурации задаётся отношением

$$\Upsilon_L = \frac{\Theta_q(x^{\mathrm{ref}},\Pi^{\mathrm{ref}})}{\Theta_q(x,\Pi)}. \tag{4.34}$$

Значение q выбирается согласованно с вероятностным ограничением (4.13). Максимум по путям используется для того, чтобы самый медленный маршрут не был скрыт более быстрыми участками конфигурации.

Экономическая оценка связывает выбранную архитектуру с затратами на её эксплуатацию. В качестве стоимости используется $C(x)$, то есть стоимость ресурсов, необходимых для работы конфигурации x. Поскольку сравниваются варианты с разной пропускной способностью, стоимость нормируется на тысячу запросов в секунду при выполнении SLA:

$$\kappa_{\mathrm{kTPS}}(x,\Pi) = \frac{C(x)}{\mu_{\mathrm{SLA}}(x,\Pi)/10^3}. \tag{4.35}$$

Для сопоставления с референсной конфигурацией используется отношение

$$\Upsilon_C = \frac{\kappa_{\mathrm{kTPS}}(x^{\mathrm{ref}},\Pi^{\mathrm{ref}})}{\kappa_{\mathrm{kTPS}}(x,\Pi)}. \tag{4.36}$$

При $\Upsilon_C > 1$ единица полезной пропускной способности обходится дешевле, чем в референсном варианте. При $\Upsilon_C < 1$ улучшение по задержке, доступности или устойчивости достигается за счёт более дорогой конфигурации ресурсов.

Фактическая доступность рассчитывается на интервале наблюдения T. В рабочее состояние включаются только те моменты, когда одновременно выполняются условие устойчивости (4.12) и ограничение SLA (4.13). Пусть $\mathcal{T}_{\mathrm{ok}} \subseteq [0,T]$ обозначает множество таких моментов. Тогда

$$\widehat{A}(x,\Pi) = \frac{|\mathcal{T}_{\mathrm{ok}}|}{T}. \tag{4.37}$$

Эта запись намеренно связывает доступность с качеством обслуживания. Сервис может быть формально запущен, но не считаться доступным для выбранного сценария, если он не выдерживает задержку или работает без запаса по нагрузке. Согласование с целевым уровнем из (4.16) оценивается отношением $\widehat{A}(x,\Pi)/A_{\min}$.

Развернутая конфигурация дополнительно сравнивается с решением оптимизационной задачи (4.14)–(4.18). Такое сравнение нужно потому, что найденный оптимум не всегда переносится в промышленную среду без изменений. На практике могут появиться ограничения по доступным ресурсам, политике безопасности, версии платформы или допустимым шаблонам развёртывания. Пусть $J^\star$ обозначает минимальное значение целевой функции в допустимой области, а $\widehat{x},\widehat{\Pi}$ — фактически развернутую конфигурацию. Тогда относительный разрыв задаётся как

$$\Gamma = \frac{J(\widehat{x},\widehat{\Pi}) - J^\star}{J^\star}. \tag{4.38}$$

Чем меньше Γ, тем ближе промышленная конфигурация к рассчитанному варианту. Большое значение Γ показывает, что итоговое развёртывание заметно отклонилось от решения, выбранного на этапе оптимизации.

Для некоторых сценариев ИТС важно учитывать не только задержку и стоимость, но и энергопотребление вычислительной схемы [200]. Это особенно заметно при выполнении задач на графическом процессоре, потоковой обработке и режимах с постоянно высокой нагрузкой.Пусть $E(x)$ обозначает среднюю мощность конфигурации при нагрузке λ, а N — число сообщений, обработанных за интервал T. Тогда энергия на один обработанный запрос определяется как

$$\varepsilon_{\mathrm{req}}(x,\Pi) = \frac{E(x)\,T}{N}, \tag{4.39}$$

а относительное улучшение по сравнению с референсом записывается как $\varepsilon_{\mathrm{req}}(x^{\mathrm{ref}},\Pi^{\mathrm{ref}})/\varepsilon_{\mathrm{req}}(x,\Pi)$. Эта величина используется вместе с энергетическим слагаемым целевой функции (4.14).

Диагностическая пригодность показывает, насколько полно конфигурация покрыта автоматическими проверками. Пусть $\mathcal{I}$ обозначает множество инвариантов. В него входят защищённость каналов, готовность сервисов, корректность маршрутов, наличие секретов и допустимость прав доступа. Пусть $\mathcal{I}_1(x,\Pi)$ — подмножество инвариантов, которые выполняются на всём горизонте измерений. Тогда

$$\Omega_{\mathrm{CI}}(x,\Pi) = \frac{|\mathcal{I}_1(x,\Pi)|}{|\mathcal{I}|}. \tag{4.40}$$

Эта метрика показывает, какая часть эксплуатационных условий контролируется без ручной проверки. Чем выше Ω_{CI}, тем меньше риск, что ошибка в маршруте, секрете, TLS-настройке или endpoint будет обнаружена уже после запуска реального потока.

Для итогового сопоставления вводится мультипликативный индекс. Он нужен не для замены отдельных метрик, а для компактной фиксации общего результата сравнения с референсной конфигурацией. Перед агрегированием используются нормированные множители:

$$\begin{aligned}
m_1 &= \Xi_{\mathrm{SLA}}(x,\Pi), \quad m_2 = \Upsilon_{\mu}, \quad m_3 = \Upsilon_{L},\\
m_4 &= \Upsilon_{C}, \quad m_5 = \frac{\widehat{A}(x,\Pi)}{A_{\min}}, \quad m_6 = 1-\Gamma,\\
m_7 &= \frac{\varepsilon_{\mathrm{req}}(x^{\mathrm{ref}},\Pi^{\mathrm{ref}})}{\varepsilon_{\mathrm{req}}(x,\Pi)}, \quad m_8 = \Omega_{\mathrm{CI}}(x,\Pi)\,.
\end{aligned}$$

Каждый множитель отвечает за отдельную сторону эксплуатационного качества. m_1 связан с выполнением SLA, m_2 — с допустимой пропускной способностью, m_3 — с хвостовой задержкой, m_4 — со стоимостью, m_5 — с доступностью, m_6 — с отклонением от оптимального решения, m_7 — с энергозатратами, а m_8 — с полнотой автоматических проверок. Такая декомпозиция позволяет видеть, за счёт какого свойства конфигурация выигрывает или проигрывает референсу.

Индекс эффективности записывается как

$$\mathcal{E}(x,\Pi) = \prod_{i=1}^{8} m_i^{w_i}, \qquad w_i \geqslant 0, \quad \sum_{i=1}^{8} w_i = 1. \tag{4.41}$$

Коэффициенты w_i задают приоритеты конкретного внедрения. Например, для контура оперативного инференса больший вес может получить m_3, а для длительной пакетной обработки — m_4 или m_7. Условие $\mathcal{E}(x,\Pi) > 1$ означает, что при выбранных весах синтезированная конфигурация лучше референсной по совокупности нормированных критериев. Связь (4.41) с постановкой оптимизации (4.14) сохраняется через компоненты m_3,m_4,m_5,m_7,m_6, которые отражают задержку, стоимость, доступность, энергию и разрыв относительно найденного решения.

Метрики рассчитываются по одной и той же процедуре для всех сравниваемых конфигураций. Сначала задаются профиль нагрузки и границы $\Delta_\pi,\varepsilon_\pi$ по (4.13). Затем выбирается перцентиль q для (4.32) и запускаются нагрузочные испытания. По результатам испытаний оцениваются μ_{SLA} по (4.30) и перцентильные задержки по (4.33). После этого вычисляются стоимость κ_{kTPS} по (4.35), энергия на запрос по (4.39), фактическая доступность по (4.37) и покрытие инвариантов по (4.40). Далее решается задача (4.14)-(4.18), определяется $J^\star$, вычисляется разрыв (4.38) и формируется индекс (4.41). Такой порядок нужен для того, чтобы различия между вариантами объяснялись их параметрами, а не разной методикой измерений.

В таблице 36 приведено качественное сопоставление вариантов исполнения по введённым метрикам. Шкала используется как инженерная оценка ожидаемого профиля варианта при типичных нагрузках ИТС. Обозначение H означает благоприятное влияние на соответствующую метрику, M указывает на зависимость результата от параметров конфигурации, а L фиксирует неблаго-

приятный профиль. Формальная интерпретация метрик задана выражениями 4.28-4.40.

Таблица 36 — Сравнение канонических архитектур исполнения моделей по метрикам эффективности

Вариант (раздел)	$\Xi_{\mathbf{SLA}}$	$\mu_{\mathbf{SLA}}$	Θ_q	$\kappa_{\mathbf{kTPS}}$	$\hat{\boldsymbol{A}}$	$\varepsilon_{\mathbf{req}}$	$\Omega_{\mathbf{CI}}$
Реляционная БД: HTTP streaming	M	M	M	M	M	M	H
Реляционная БД: gRPC (непотоковый SELECT)	M	M	M	M	M	M	H
Kubernetes-контур с контроллерами (инварианты CI)	H	M	M	M	H	M	H
GPU-инференс	H	H	M	M	M	M	M
Hadoop/Spark	L	H	L	M	M	M	M
S3-объектное хранилище	M	M	M	M	H	M	H
In-Memory Data Grid (Apache Ignite)	H	H	H	M	M	M	M
Синхронный HTTPS/gRPC	M	M	M	M	H	M	H
Асинхронный Kafka	H	H	M	M	H	M	M

Обоснование качественных оценок. Оценки в табл. 36 следует читать как инженерную интерпретацию ожидаемого профиля варианта, а не как абсолютный рейтинг технологии. Каждая отметка зависит от того, какой участок маршрута становится ограничивающим при типичной нагрузке ИТС.

1. Для Ξ_{SLA} и Θ_q главным фактором является хвост задержки. Если в маршруте доминирует сеть, диск или очередь без жёсткого ограничения, доля выполнения SLA снижается, а q-перцентиль задержки растёт. При локализации данных и устранении лишних магистральных передач профиль обычно становится более благоприятным.
2. Для μ_{SLA} существенна не только номинальная скорость компонента, но и положение узкого места на полном пути. Если самая медленная

стадия имеет достаточно большую $\mu(\pi)$ в (4.11), конфигурация выдерживает больший входной поток. GPU и Kafka получают высокую оценку именно в тех режимах, где параллелизм и разделение скоростей обработки реально проявляются на маршруте, см. (4.20)-(4.23).

3. Для κ_{kTPS} (4.35) неблагоприятны дорогие ресурсы и избыточные инфраструктурные накладные расходы. Поэтому GPU-контур может проигрывать по стоимости при малой загрузке. В то же время при заметном росте μ_{SLA} (4.30) стоимость единицы полезной пропускной способности может снизиться.
4. Для $\hat{A}$ положительно влияют репликация, развязка компонентов и отсутствие единственной критической точки отказа. Поэтому Kafka и S3 получают более устойчивый профиль в сценариях, где допустима асинхронность или объектное хранение. Дополнительный вклад даёт формализация инвариантов доступности и безопасности в CI (4.40), поскольку часть ошибок конфигурации выявляется до запуска.
5. Для Ω_{CI} выше оцениваются варианты, в которых свойства можно проверить декларативно: TLS, RBAC, Secrets, Routes и Topics. Ниже оцениваются контуры, где существенны аппаратные и низкоуровневые зависимости. Для GPU это драйверы и CUDA-окружение, для Hadoop - Kerberos, роли узлов и состояние кластера. Такие проверки возможны, но требуют более широкого набора тестов.

После таблицы 36 уточняется, какие параметры сильнее всего влияют на отдельные метрики. Для GPU-контура это прежде всего b, p, h и n. Параметр b задаёт размер микропакета, p определяет режим вычислительной точности, h фиксирует класс аппаратной платформы, а n задаёт число реплик сервиса. Их влияние на μ_{SLA} и Θ_q проявляется через время инференса, передачу данных между хостом и устройством и ограничения памяти GPU (4.19)-(4.20).

Для Kafka-контура основными параметрами являются P, c, n и r. Параметр P задаёт число секций входного топика, c определяет параллельность обработки, n влияет на число экземпляров сервиса, а r связан с репликацией сообщений. Эти параметры задают скорость разбора очереди и величину ожидания в секциях. Их влияние описывается оценками (4.21)-(4.23).

Для S3-контура главными параметрами являются k и c. Первый задаёт параллелизм чтения, второй определяет размер части объекта. Если c слишком мало, возрастает доля накладных расходов на запросы. Если c слишком велико,

увеличивается нагрузка на буферы контроллера и модели. Поэтому μ_{S3} и L_{obj} оцениваются через (4.24)-(4.26) при обязательном учёте ограничения по памяти.

Для In-Memory БД существенны коэффициент репликации, число узлов и политика аффинного размещения. Эти параметры определяют, какая доля обращений будет выполнена локально, а какая потребует сетевого перехода к другому узлу. Поэтому влияние In-Memory БД на Θ_q и $\mu(\pi)$ проявляется через локальность доступа и коэффициент попаданий в кэш.

Для реляционной БД основным остаётся режим передачи результата. Потоковая выдача полезна, когда важно уменьшить задержку первого фрагмента и быстрее начать обработку данных в сервисе модели. Буферизация, напротив, оправдана при умеренном объёме выборки, когда выгоднее сократить число сетевых обменов.

4.3 Выводы по главе

В главе решалась задача выбора конфигурации исполнения моделей ИИ для контуров ИТС, в которых результат модели должен быть получен с заданными ограничениями по времени, доступности и пропускной способности. Показано, что такую конфигурацию нельзя описывать только через выбранную технологию запуска модели. Существенным является весь маршрут данных: от источника запроса и доступа к хранилищу до вычисления, передачи результата и журналирования. Поэтому в рассмотрение включены несколько вариантов исполнения: REST/gRPC-вызов, Kafka-контур, реляционная БД, S3, In-Memory Data Grid, GPU-ускорение и пакетное исполнение в Hadoop/Spark. Для каждого варианта выделены параметры, которые непосредственно влияют на задержку, масштабирование, доступность и возможность автоматической проверки.

Методология опирается на ориентированный мультиграф исполнения. Вершинами в нём являются компоненты, которые реально участвуют в обработке запроса: сервис модели, контроллер доступа к данным, транспортный адаптер, брокер, хранилище, вычислительный узел или компонент наблюдаемости. Рёбра задают каналы передачи данных между этими компонентами. Для каждого ребра фиксируются измеряемые характеристики, включая задержку T_e, пропускную способность μ_e, протокол обмена и требования к защищённо-

му каналу. Поэтому разные варианты исполнения сравниваются не по общему описанию технологии, а по составу маршрута и параметрам его рёбер.

Для разных типов каналов в главе использованы разные расчётные модели. В GPU-контуре отдельно учитываются передача данных из памяти хоста на устройство, выполнение ядра и возврат результата. В Kafka-контуре существенны число секций, группа потребителей и ожидание сообщения в очереди. В S3-контуре задержка определяется чтением объекта частями и ограничением буферов. Для реляционной БД различаются потоковая передача результата и буферная передача при умеренном объёме выборки.

Введённый формальный аппарат нужен для описания маршрутов, которые имеют разную технологическую природу, но проверяются по сходным эксплуатационным требованиям. В одном случае маршрут проходит через синхронный вызов модели. В другом случае он включает брокер сообщений, объектное хранилище, распределённую память или пакетное вычисление. Представление через параметризованный мультиграф позволяет не смешивать эти случаи, но задавать для них единые признаки: вершины, рёбра, задержки, пропускную способность, требования безопасности и параметры масштабирования.

Для оценки маршрутов используются конвейерные модели задержки и пропускной способности. В расчёт включаются не только логические стадии обработки, но и ограничения среды исполнения. Для GPU существенны передача через PCIe, память устройства и время выполнения ядра. Для Kafka важны секции, группа потребителей и ожидание в очереди. Для Spark учитывается организация задания и чтение данных. Поэтому задержка в главе трактуется как характеристика полного пути исполнения, а не как изолированное время работы одного сервиса модели.

Выбор конфигурации сформулирован как многокритериальная задача. Управляемыми параметрами выступают число реплик, размер микропакета, уровень параллелизма, коэффициент репликации и другие настройки исполнения. Целевая функция учитывает время ответа и стоимость ресурсов. Допустимая область задаётся ограничениями по SLA, устойчивости входного потока, доступности и ресурсам.

Проверка архитектуры вынесена в CI как отдельный этап работы с конфигурацией. До выпуска контролируются готовность сервисов, наличие защищённых каналов, корректность прав доступа, состав секретов и связность маршрутов. Затем выполняется нагрузочная проверка, на которой измеряется

выполнение SLA. Если маршрут не проходит хотя бы один инвариант или показывает недопустимую задержку, такая конфигурация не должна попадать в эксплуатационный контур.

Для сопоставления вариантов исполнения введены метрики, которые можно получить повторным измерением. Используются доля запросов, уложившихся в SLA, пропускная способность при сохранении SLA, стоимость тысячи запросов, фактическая доступность и покрытие инвариантов CI. Эти показатели позволяют сравнивать конфигурации по свойствам полного маршрута обработки, а не по названию применённой технологии.

Разработанная методология смещает акцент с самой модели на условия её исполнения. Модель рассматривается вместе с маршрутом данных, вычислительными ресурсами, контроллерами доступа и проверками эксплуатации. Поэтому к обычному жизненному циклу модели добавляются этапы выбора схемы исполнения, расчёта ограничений, подготовки конфигурации и проверки перед выпуском. Такое представление позволяет заранее связать архитектурное решение с его измеряемыми последствиями. Для ИТС это важно, поскольку нарушение временных характеристик может привести не только к снижению качества сервиса, но и к ухудшению устойчивости управления транспортным потоком.

Архитектурные шаблоны для реляционных БД, S3, In-Memory Grid и Kafka описаны как компоненты, развёртываемые в Kubernetes. Для каждого шаблона фиксируются границы сервиса, параметры подключения, правила масштабирования и состав секретов. Код модели при этом не должен содержать логику подключения к конкретному хранилищу или брокеру. Эту функцию выполняет соответствующий контроллер доступа.

Требования безопасности задаются отдельными настройками конфигурации. Межсервисный обмен выполняется с использованием mTLS. Разграничение полномочий задаётся через RBAC. Ключи, сертификаты и параметры подключения размещаются в Kubernetes Secrets. Такая организация позволяет менять параметры инфраструктуры без изменения кода модели и одновременно сохранять проверяемость маршрута исполнения.

Итогом главы является процедура, которая переводит описание сценария ИТС в проверяемую конфигурацию исполнения модели. Сначала фиксируется маршрут данных и его критические участки. Затем выбираются параметры вычислительной среды, транспортного контура, хранилищ и механизмов

масштабирования. После этого конфигурация проверяется по задержке, доступности, устойчивости и инвариантам безопасности. В таком виде архитектура исполнения задаётся не описательно, а через набор параметров и проверок, которые можно повторить при развёртывании и затем скорректировать при изменении нагрузки.

Глава 5. Разработка методологии построения инфраструктуры интеллектуальных транспортных систем

5.1 Разработка метода децентрализованного взаимодействия подсистем ИТС

Постановка задачи

Рассмотрим множество серверов ИТС $S = \{s_1, \ldots, s_n\}$, размещённых в разных сетевых зонах. Пусть сетевое расстояние между узлами определяется дву-сторонней задержкой $\mathrm{RTT}(s_i, s_j)$, образующей веса полного графа $G = (S, E)$.

Требуется найти разбиение S на подкластеры $= \{C_1, \ldots, C_k\}$, $C_m \subseteq S$, и множество лидеров $L = \{\ell_1, \ldots, \ell_k\}$, где $\ell_m \in C_m$, при ограничениях:

$$C_m \cap C_{m'} = \varnothing, \quad \bigcup_{m=1}^{k} C_m = S, \tag{5.1}$$

$$|C_m| \geqslant 3 \quad \text{(минимум для большинства в Raft)}, \tag{5.2}$$

$$\forall s_i \in C_m : \ \mathrm{assign}(s_i) = \ell_m. \tag{5.3}$$

Определим функционал качества топологии:

$$J(,L) = \alpha \sum_{m=1}^{k} \sum_{s_i \in C_m} \mathrm{RTT}(s_i, \ell_m) + \beta \mathbb{E}[T_{\text{failover}}(,L)] + \gamma T_{\text{setup}}(,L),$$

где $\alpha, \beta, \gamma \geqslant 0$— весовые коэффициенты; T_{failover}— время переизбрания лидера при отказах; T_{setup}— время включения нового узла.

Целью является минимизация $J(C, L)$ при соблюдении согласованности данных в каждом C_m согласно Raft (безопасность, лог-матчинг, лидерство) и обеспечить динамическую реконфигурацию при изменении и состава без ручного вмешательства администратора. [26; 51]

Задачей данного раздела является подробная проработка архитектурных вопросов функционирования интеллектуальных транспортных систем с использованием технологий обработки больших данных. Предлагаемый вариант

эталонной архитектуры системы обработки больших данных для интеллектуальной транспортной системы позволяет определить пути планирования, разработки и развертываний приложений в рассматриваемой предметной области и облегчить внедрение решений для анализа больших данных для ИТС. Предлагаемая архитектура ИТС содержит описание разворачиваемой системы, включая стек технологий и протоколы интеграции (этапы сбора, хранения, извлечения, обработки и использования данных), согласованные с доменными службами ИТС [21; 30; 223—226].

Референсная архитектура описывается представлениями, каждое из которых отражает проблемы конкретной системы. Представления облегчают обобщение и обсуждение архитектурных вопросов заинтересованными сторонами. В частности, представления, включенные в предлагаемую архитектуру ИТС, являются функциональными, технологическими и интеграционными.

Функциональное представление архитектуры представлено на рисунке 5.1. Данное представление описывает службы (наборы общих функций), коннекторы (коммуникация между службами) и группы служб. Зона представления (А) определяет, как службы предоставляют информацию по каналам.

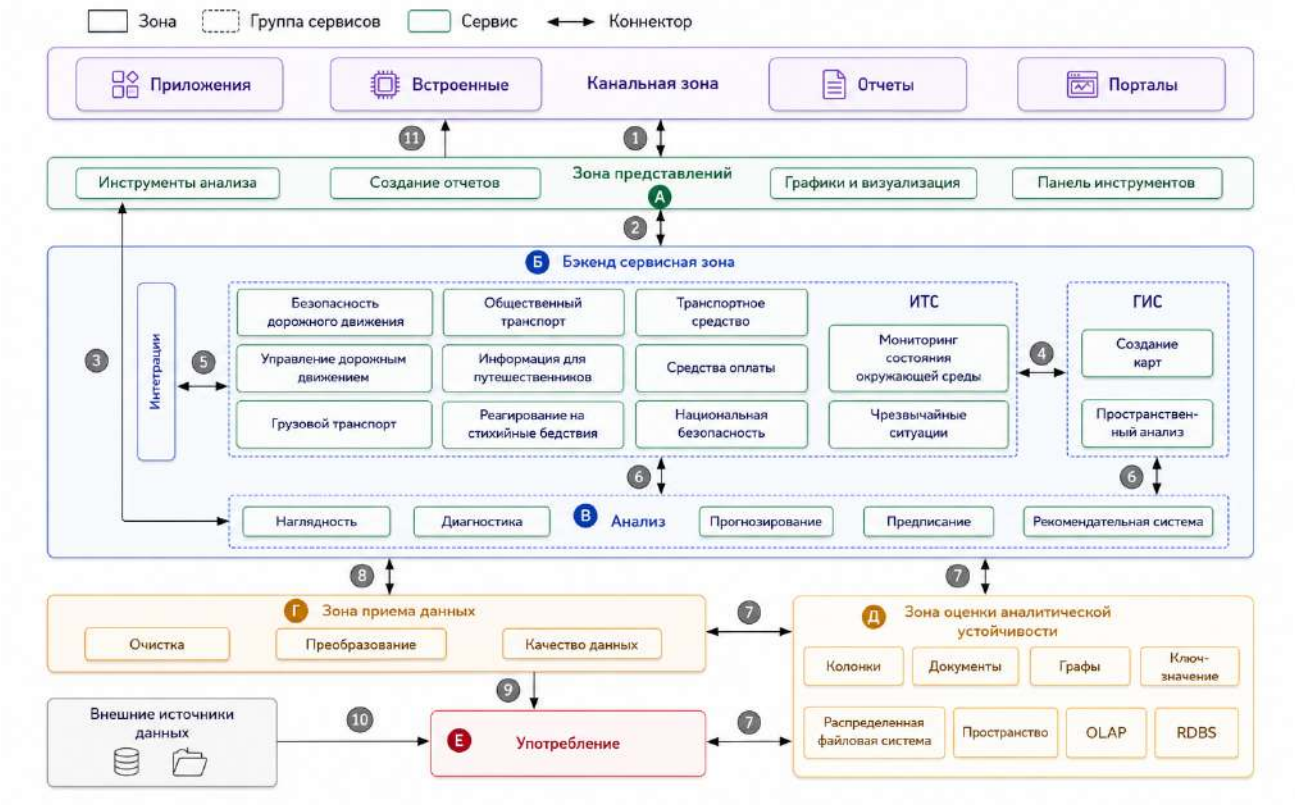

Рисунок 5.1 — Функциональное представление архитектуры ИТС

Зона (Б) описывает серверные службы, которые объединяют рассмотренную эталонную архитектуру ИТС с геоинформационными сервисами (ГИС). Зона аналитики (В) описывает типы анализа, которые должны выполняться и поддерживаться в Зоне хранения данных (Г) и Зоне оценки аналитической устойчивости (Д). Зона оценки аналитической устойчивости предлагает сервисы такие, как распределенная файловая система, SQL, NoSQL хранилище и др [223; 227—230]. Зона потребления (Е) отвечает за сбор данных из внешних источников и их перенаправление соответствующим потребителям [224; 231; 232].

Интеграционное представление используется для описания каждого типа коннектора (пронумерованного от 1 до 11 на рисунке 5.1) на транспортном, распределительном, посредническом и прикладном уровне. На рисунке 5.2 представлен коннектор 10-го типа, определяющий интеграцию между внешними данными и службой загрузки. Данный коннектор описывает рекомендованные протоколы для связи внешних ресурсов на интеграционном и транспортном уровне. К ним относятся протоколы: HTTP, FTP, WebSocket и MQTT. Протоколы должны выбираться исходя из схемы коммуникаций (ориентированной на подключение или работающей по системе подписка/публикация) и их конечных точках (веб приложение, БД, устройства Интернета вещей, FTP сервер и др.) [233—237].

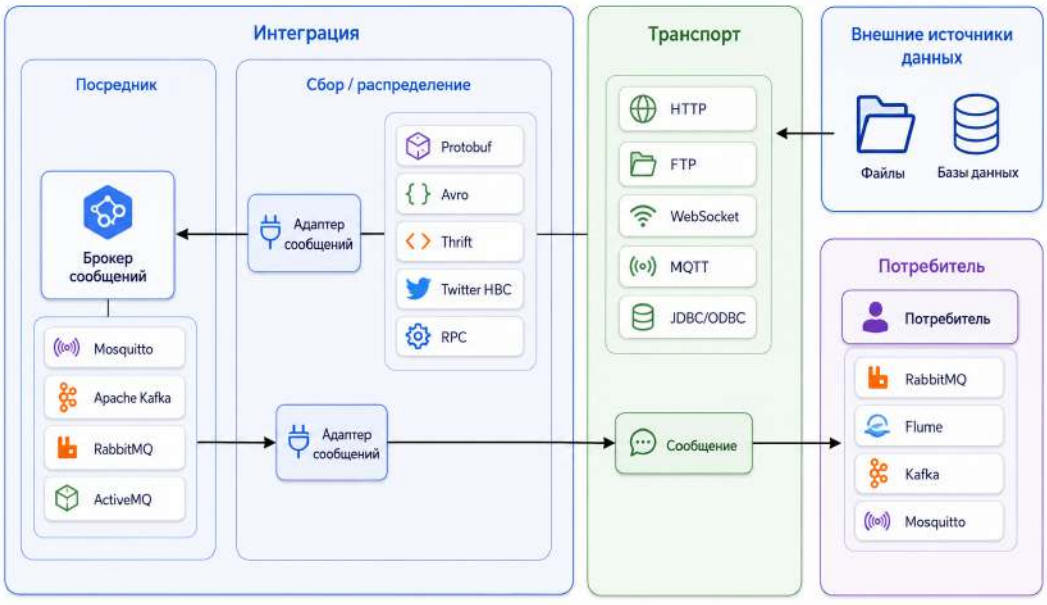

Рисунок 5.2 — Интеграционное представление коннектора 10-го типа

Представление выбора технологии содержит список рекомендаций по конкретным продуктам, предлагающим возможности, необходимые для службы

или группы служб, которые необходимо создать. Это представление является своего рода пособием при выборе решений в конкретной реализации.

Рисунок 5.3 описывает технологическое представление для продуктов с открытым исходным кодом. Другие варианты данного представления могут фокусироваться на продуктах конкретных поставщиков.

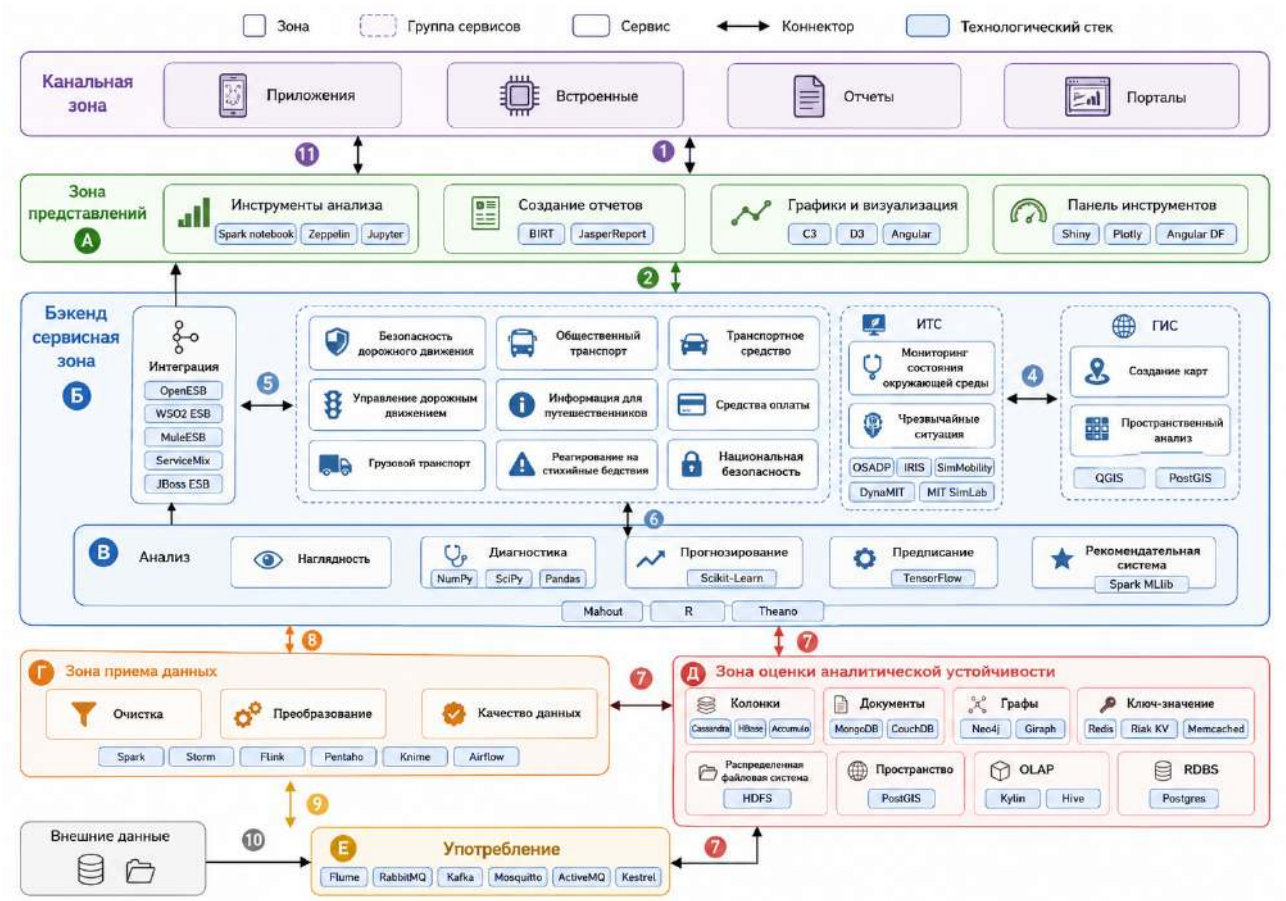

Рисунок 5.3 — Представление выбора технологий с открытым исходного кода

5.2 Разработка метода кластеризации серверов в ИТС

В данном пункте иллюстрируется и подтверждается реализуемость и удобство использования предлагаемой архитектуры на трех частных случаях с конкретной архитектурой в рамках ИТС: Ч1, Ч2, Ч3. В таблице 37 кратко описывается каждый частный случай, а также применяемые тактики и шаблоны. Первый частный случай описывает интеллектуальную систему для анализа дорожно-транспортных происшествий и их связи со штрафами за нарушения правил дорожного движения. Второй пример описывает интеллектуальную си-

стему, способную прогнозировать задержки автобусов практически в режиме реального времени. Пример №3 описывает подсистему ИТС, позволяющая анализировать транспортную мобильность в связи с данными о транспортной обстановке, собранными датчиками.

Таблица 37 — Описание частных случаев

Название	**Описание**	**Тактики**	**Шаблоны (проектные архитектуры)**
Ч1	Анализ дорожных происшествий	Удаление ненужных данных, выбор и извлечение признаков	Обработка данных, хранение
Ч2	Прогнозирование задержек автобусов	Параллельная обработка, фрагментация, повторное использование предыдущих результатов, использование данных, хранящихся на разных серверах	Лямбда архитектура, интерактивный механизм запросов, брокер распределенных сообщений
Ч3	Кратчайшие пути на основе анализа	Удаление ненужных данных, выбор и извлечение признаков, параллельное программирование, использование данных, хранящихся на разных серверах	Сборщик данных, обработка данных

Описание частного случая №1. Данный случай описывает ИТС для анализа дорожно-транспортных происшествий и их связи со штрафами за нарушения правил дорожного движения. Для визуализации анализа и мониторинга аварий рекомендуется. Модуль анализа обрабатывает данные о дорожно-транспортных происшествиях и их связи со штрафами за нарушения правил дорожного движения для поддержки принятия решений в области безопасности дорожного движения. Модуль мониторинга операций отслеживает перемещения автобусов в системе общественного транспорта с целью применения моделей мобильности. Данные модули объединены под единым интерфейсом, позволяющим пользователям получать информацию о несчастных случаях на дорогах, автобусных маршрутах, а также данные их геолокации и скорости движения. Прогнозирование включает в себя вычисление индекса риска на каждом участке дороги и корреляции аварий и штрафов за нарушение правил дорожного движения.

В результате управление ИТС имеет возможность анализировать дорожную обстановку и принимать решения по обеспечению безопасности дорожного движения.

На рисунке 5.4 представлена программная архитектура, разработанная на основе предлагаемой архитектуры ИТС, а в таблице 38 показаны связи между компонентами и сервисами. Происшествия, штрафы за нарушение правил и дорожные сети являются внешними источниками данных, загружаемыми посредством компонента AccidentsETL, который является частью сервисов зоны Е и Г предлагаемой архитектуры ИТС и реализуется с использованием программных средств Python, Pandas и PostGIS. По количеству и качеству данные включает в себя: штрафы за нарушение правил, дорожные происшествия, данные GPS-трекинга автобусов, а также дорожные сети и их граф. При поступлении новых данных происходит их фильтрация и сохранение в базе данных MondoDB, что соответствует зоне Д. Помимо этого данные о работе автобусов загружаются и объединяются посредством компонента OperETL, выполненного программным средством Spark в зоне Е и Г. Результирующие объединенные данные также сохраняются в базе данных MongoDB. В зоне Б компонент AccidentBackend и OperBackend осуществляют доступ и агрегирование предварительно обработанных данных о происшествиях, используя Python, и затем предоставляет результаты обработки фронтэнд компоненту посредством REST. Фронтэнд компоненты AccidentDashboard и OperMonitoring в свою очередь реализуются с использованием средства Angular Dashboard Framework (ADF), AngularJS, C3, D3 и Leaflet в сервисной зоне А. [227; 230; 238—241]

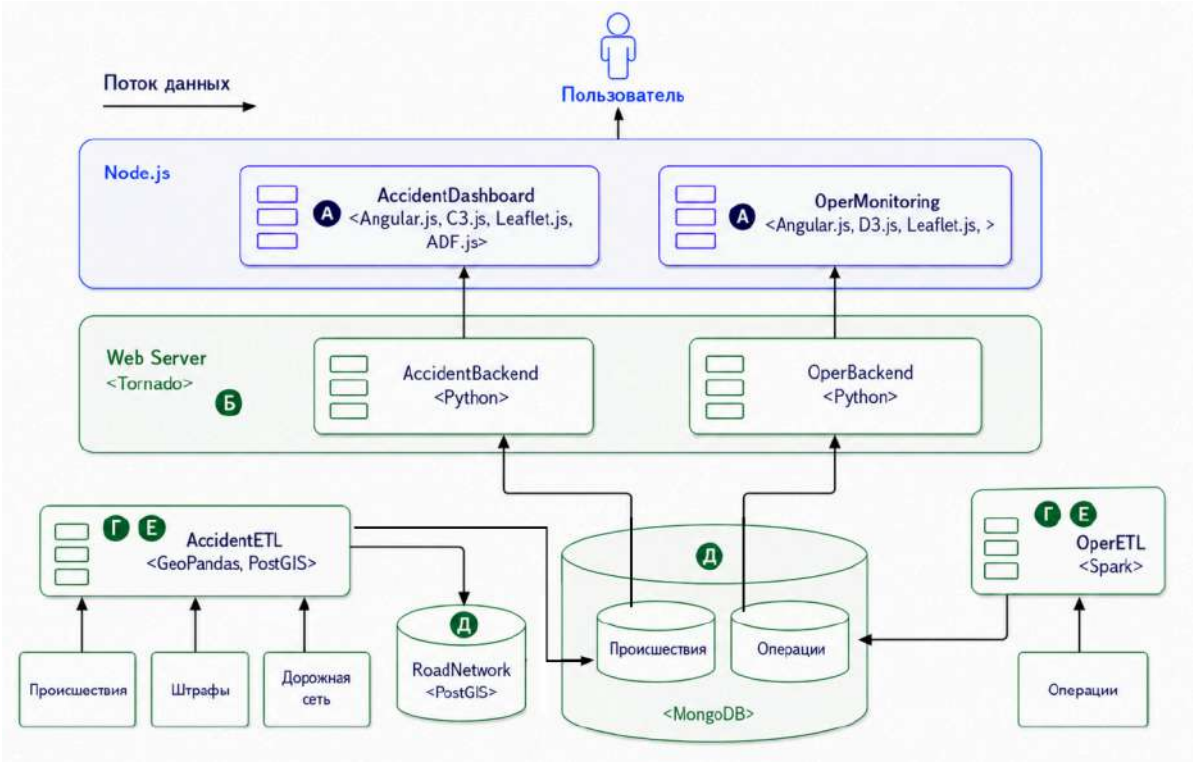

Рисунок 5.4 — Программная архитектура для частного случая № 1

Таблица 38 — Сопоставление компонентов с функциональным видом предлагаемой архитектуры ИТС

Компоненты	**Зона**	**Сервисы**
AccidentsETL	Г, Е	Употребление, очистка, трансформация
OperETL	Г, Е	Употребление, очистка, трансформация
Accidents, Operations	Д	Хранение документов
RoadNetwork	Д	Пространственная база данных
AccidentBackend	Б, В	Описательный анализ безопасности дорожного движения
OpenBackend	Б, В	Описательный анализ управления дорожным движением
Accident and operation UI	А	Визуализация панели инструментов

Описание частного случая №2. Частный случай №2 (Ч2) описывает интеллектуальную систему, позволяющую во времени близком к реальному предсказывать задержки движения автобусов. В данной системе обновления

о поездках собираются и агрегируются по маршрутам автобусов для расчета среднего значения задержки за временное окно, которое составляет пять минут. Обработка данных производится с низкой задержкой и ограниченной точностью ввиду наличия микропакетов данных. С другой стороны, пакетная обработка, хотя и позволяет построить точную модель, не обеспечивает достаточного времени отклика. Данный вариант позволяет пользователям получать более точную оценку задержек движения транспорта, чтобы планировать свои поездки, и управлениям ИТС более эффективно выстраивать графики движения транспорта.

Предлагаемая архитектура разработана с учетом совместимости как показано на рисунке 5.5. В таблице 39 показаны связи между компонентами и сервисами архитектуры. Обновленные данные о поездках в режиме реального времени преобразуются в специализированный формат. Каждый пакет такого формата генерируется раз в минуту и передается для обработки. Каждое обновление является бинарным файлом типа protobuf (размером 56 Кб в среднем), который необходимо преобразовать в JSON формат. Обновляемые данные затем принимаются с использованием компонента Apache Kafka (сервис зоны Е), чтобы обеспечить низкую задержку и высокую доступность. На скоростном уровне (зоне В), компонент прогнозирования задержек принимает данные из Kafka для агрегации и прогнозирования ожидаемых задержек за временное окно с использованием Spark Streaming и сохранения скорости просмотров в HDFS. Кроме того, каждое необработанное обновление поездки сохраняется в HDFS для дальнейшей обработки на пакетном уровне. Пакетный уровень реализован с использованием Spark и SparkML Lib, где компонент TripAggregator (зона Г) обрабатывает данные и сохраняет их в HDFS. DelayPredictor (зона В) оценивает ожидаемую задержку за временное окно и пути движения автобусов, сохраняя результаты в виде пакетов в HDFS.На уровне сервиса (зоны А и В) интегрированы как произвольные, так и пакетные запросы с использованием интерактивных пользовательских запросов через компонент запросов, который использует HiveQL. [224; 231; 232; 236; 242]

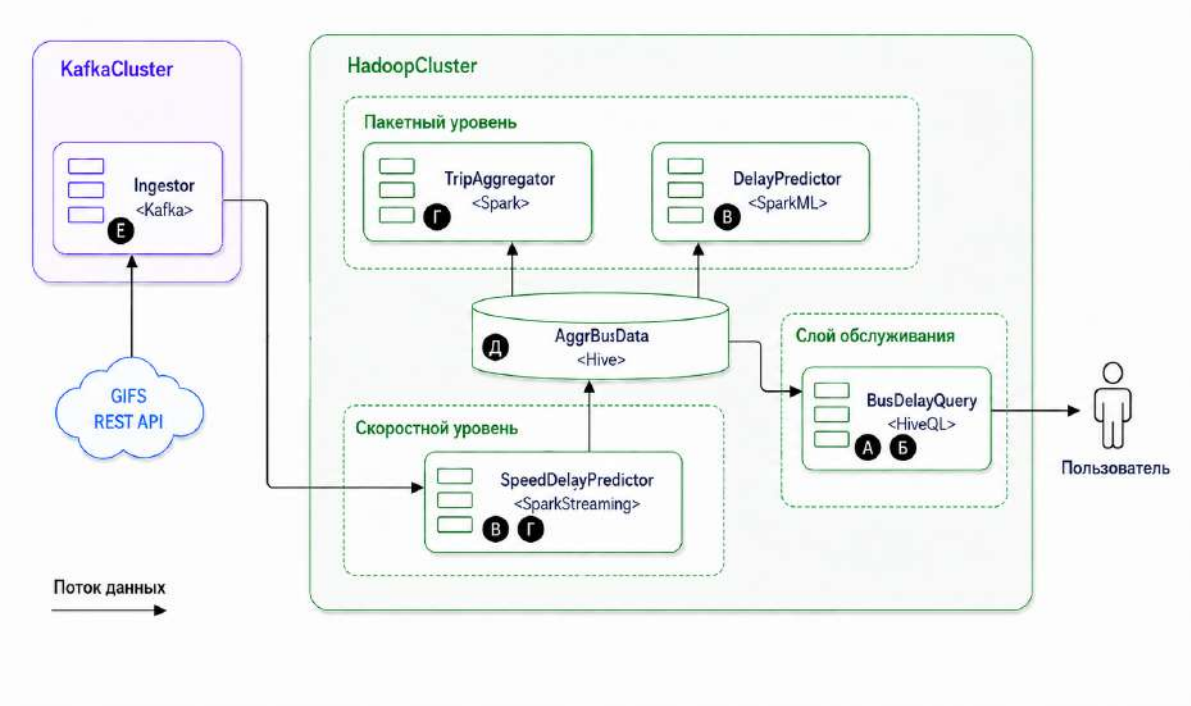

Рисунок 5.5 — Программная архитектура для частного случая № 2

Таблица 39 — Сопоставление компонентов с функциональным видом предлагаемой архитектурой ИТС

Компоненты	**Зона**	**Сервисы**
Ingestor	Е	Употребление
SpeedDelayPredictor	В, Г	Преобразование, прогнозирование и анализ информации о путешественниках
TripAggregator	Г	Преобразование
DelayPredictor	В	Прогнозирование информации о путешественниках
AggrBusData	Д	OLAP
BusDelayQuery	А, Б	Информация для путешественников, инструменты интерактивной аналитики

Описание частного случая №3. В данном примере описывается ИТС, позволяющая анализировать транспортную мобильность на основе данных собранных датчиками. Транспортная мобильность, обеспечиваемая интеллектуальными технологиями, предлагает решение проблемы транспортных заторов в условиях быстрой урбанизации. Информация о дорожных заторах собирается каждые пять минут в полуструктурированном виде в формате CSV

Open Data Aarhus. Относительно набора сетевых сенсорных данных датчиков строится график для расчета кратчайших путей между местоположениями пользователей. Затем рассчитываются наименее загруженные пути на основе предварительно рассчитанных кратчайших путей, собранных сетью датчиков с использованием кластера Hadoop. Рассматриваемое решение позволяет конечным пользователям выбирать более безопасные пути, а управлениям ИТС отслеживать переменные среды для улучшения качества принятия обоснованных мер.

На рисунке 5.6 показана программная архитектура частного случая №3. В таблице 40 отображены связи между компонентами и сервисами предлагаемой архитектурой ИТС. Наборы данных о парковке, погоде, дорожном движении загружаются и предварительно обрабатываются загрузчиком файлов, реализованным при помощи языка R (зона Е). Эти предварительно обработанные данные принимаются двумя компонентами (зона Г) для построения графа сети и вычисления кратчайших путей в HDFS, зона Д. Кратчайшие пути обрабатываются с помощью Hadoop для вычисления наиболее рационального пути между двумя точками, зона В. В Shiny, бэкенд компоненты AnalysisServer и SmartMobilityServer анализируются предварительно обработанные данные и наиболее рациональные пути для их дальнейшего отображения в графическом интерфейсе. Бэкенд компоненты находятся в сервисной зоне Б рассматриваемой архитектуры. Согласно представлению зоны А, компоненты AnalysisUI и MobilityUI предоставляют панель инструментов на сервере Shiny.

Таблица 40 — Сопоставление компонентов с функциональным видом предлагаемой архитектурой ИТС

Компоненты	**Зона**	**Сервисы**
FileLoader	Е	Употребление
PollutionGraph	Г	Очистка, преобразование
ShortestPath	Г	Преобразование
Healthiest	В	Описательный анализ
HealthiestPaths, Preprocessed data	Д	Распределенная файловая система
AnalysisServer, SmartMobility	Б	Информация для путешественников, инструменты интерактивной аналитики
AnalysisUI, SmartMobilityUI	А	Визуализация панели инструментов

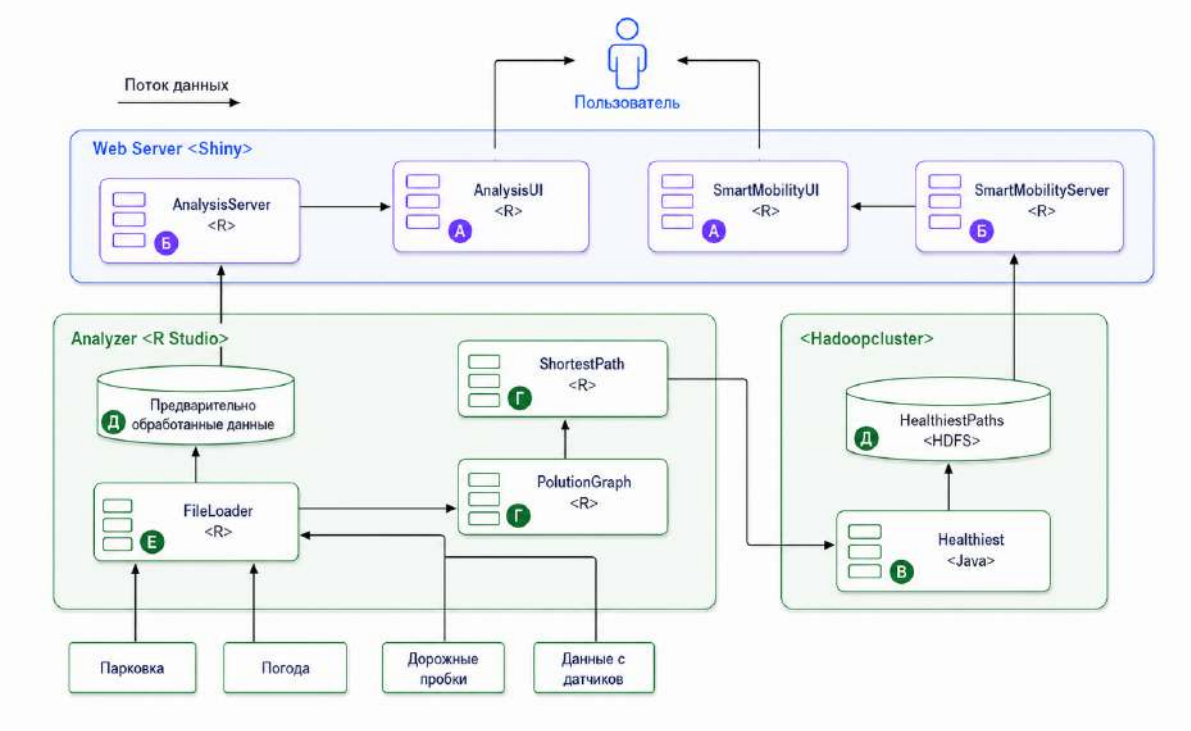

Рисунок 5.6 — Программная архитектура для частного случая № 3

В случае организации интеллектуальной транспортной системы большого масштаба (государственной, региональной или крупной агломерации) встает вопрос о рациональном распределении серверных ресурсов, необходимых для функционирования ИТС. Видится целесообразным проработать методический аппарат кластеризации серверов ИТС, который позволяет контролировать консистентность (согласованность) системы в целом. В случае решения задач для ИТС строгая консистентность играет немаловажную роль, поскольку несколько равноценных серверов находятся в едином подкластере, соответственно, существует необходимость достижения консенсуса. Каждый географический регион может иметь несколько серверов, объединенных в подкластер, находящийся внутри другого кластера. Именно набор нескольких подкластеров позволяет создать децентрализованную систему ИТС, где каждый подкластер обладает необходимым функционалом и работает независимо друг от друга [243—246].

Современные методы кластеризации имеют несколько проблем, одной из которых является вероятное наличие больших сетевых задержек, при которых некоторые серверы данного кластера не могут эффективно обрабатывать данные или поддерживать новые задачи и поэтому кластер теряет свою ценность. В большинстве случаев архитектура построения кластеров является одноуровневой, т.е. кластер состоит из нескольких серверов и один из них – главный сервер, в задачи которого входит распределение поставленных новых задач на

другие серверы. В общем случае при одноуровневом построении один кластер функционирует как один большой сервер. Каждый новый сервер присоединяется константно к кластеру и отключается от кластера, если сервер был удален или вышел из строя. Если серверы расположены в разных географических (сетевых) зонах, они продолжают получать задачи от основного сервера, который может находиться в другой географической зоне. Исходя из этого формируются следующие недостатки: константное подключение сервера к кластеру не обеспечивает полную работоспособность данного сервера, при больших задержках сервер не принимает новые задачи, большие задержки в получении новых задач. [243—245; 247—249]

Для устранения указанных недостатков и решения задачи проработки современного кластерного подхода, в данном исследовании предлагается использовать следующие новые элементы: сервис Agent, метрика, представляющая собой сетевое расстояние.

Благодаря наличию данного сервиса существует возможность разрешения конфликтов в кластере и решения задачи консенсуса. Сервис Agent может приобретать одно из трех состояний. При запуске сервис имеет начальное состояние Последователь. При отсутствии Лидера в кластере сервис через 10 секунд меняет состояние на Кандидат. Если количество голосов от других серверов больше 50% от всех имеющихся голосов в кластере, то сервис изменяет свое состояние на Лидер, иначе на состояние Последователь.

В рамках исследования разработан и внедрён метод кластеризации серверов, основанный на совместном использовании метрики сетевого расстояния (RTT), протокола консенсуса Raft и автоматического выбора лидера, реализуемого сервисом Agent. Сетевое расстояние – время, в течение которого сервер получает ответ на свой запрос от другого сервера. На рисунке 5.7 графически представлены значения сетевого расстояния для различных пар серверов.

Сервер добавляется к определенному Лидеру, выбирая его по минимальному сетевому расстоянию, что позволяет избежать каких-либо дополнительных операций и действий со стороны системного администратора. Каждый Лидер контролирует своих Последователей и, если сервер не выполняет все функции с полной эффективностью, Лидер удаляет его из подкластера. Для принудительного удаления сервера может быть несколько причин: длительная задержка сети, сервис Agent работает некорректно, сервер работает некорректно.

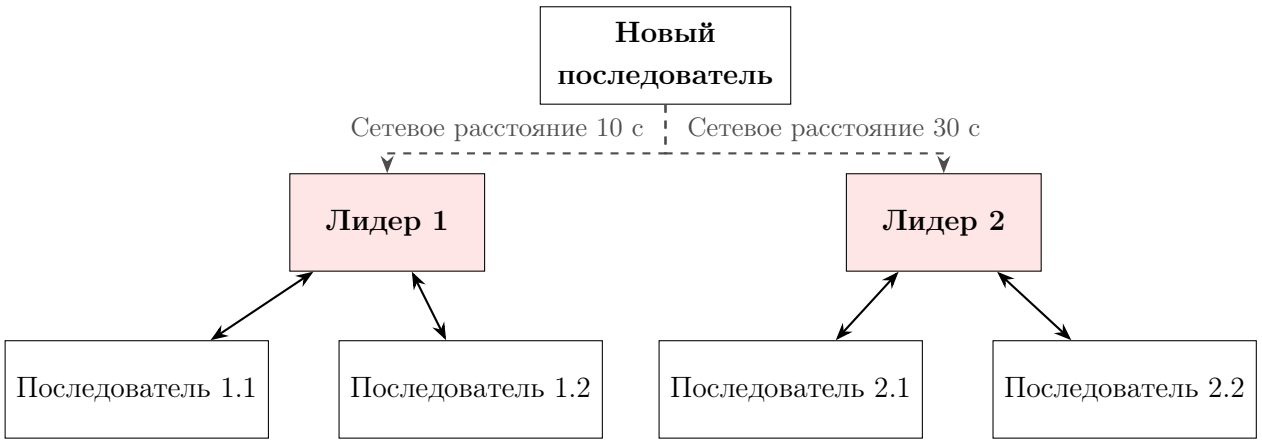

Рисунок 5.7 — Динамическое добавление нового сервера

На рисунке 5.8 представлена схема, поясняющая порядок смены Лидера и удаления Последователя, соответственно.

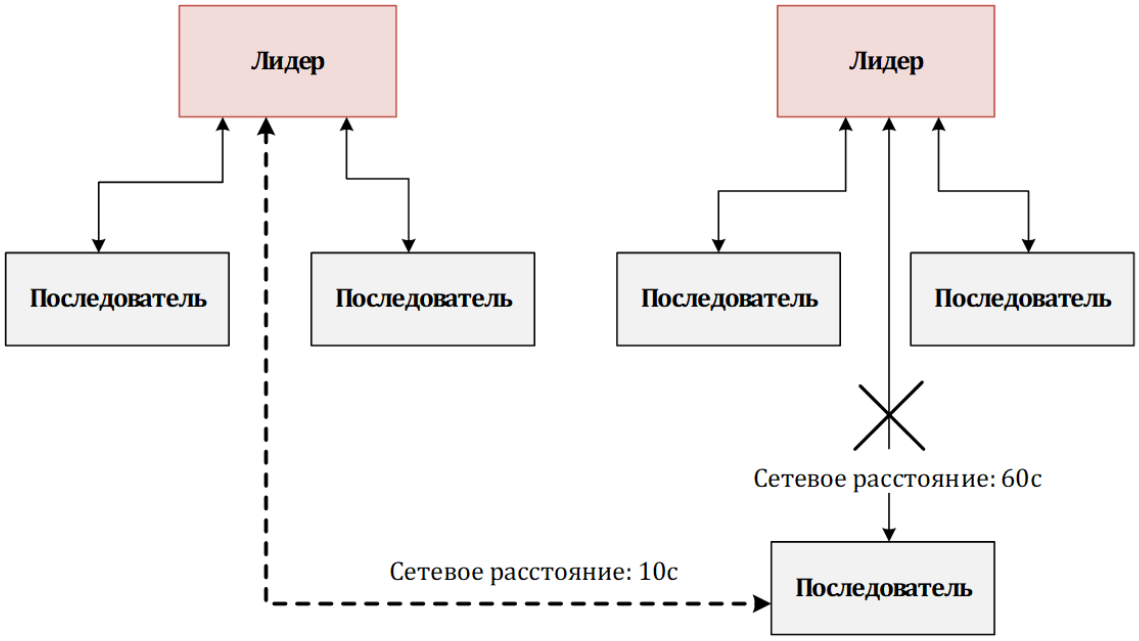

Рисунок 5.8 — Схема смены Лидера

Наличие вышепредставленных процедур необходимо для оптимизации вычислительных ресурсов.

Далее представлено, как выглядит автоматическая кластеризация данных после внедрения разработанного метода. Пусть существует шесть серверов, которые не сгруппированы в кластеры (рисунок 5.9).

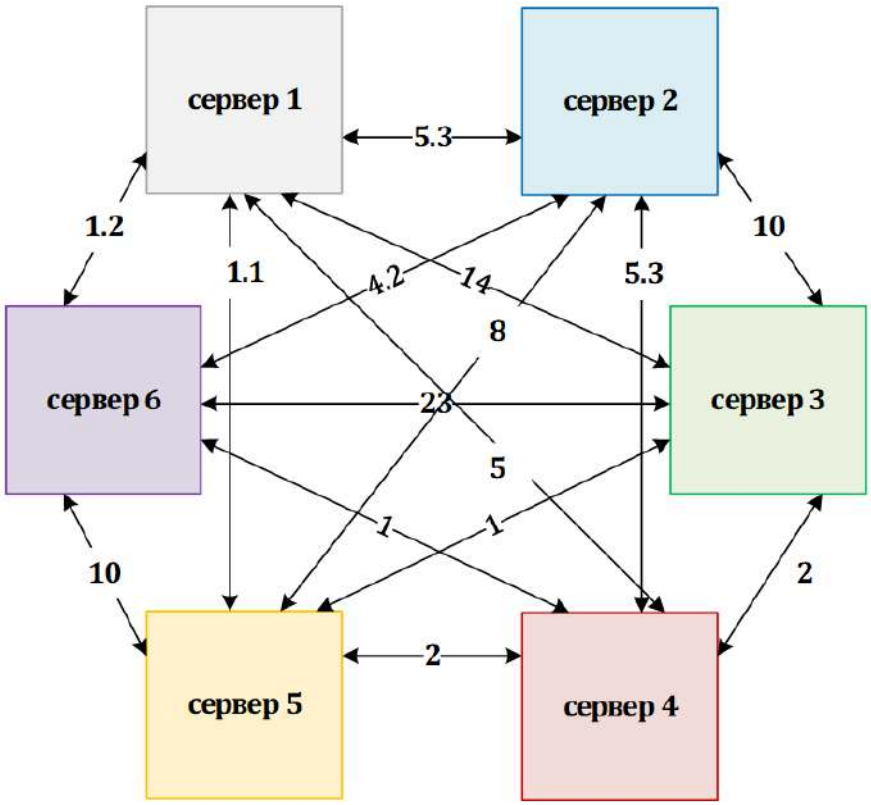

Рисунок 5.9 — Схема взаимодействия серверов без кластеризации

Каждый из серверов производит опрос друг друга и детерминируют сетевое расстояние друг между другом. В дальнейшем процедура заключается в том, что каждый из серверов (на котором работает тот же сервис Агент) выбирает те серверы, которые являются ближайшими к нему по сетевому расстоянию, что позволяет кластеризировать их динамически без вмешательства сетевого администратора. Далее шесть серверов группируются в два подкластера (рисунок 5.10).

Каждый сервер понимает, что в подкластере может быть не менее трех серверов. Поэтому если сервер не нашел ближайшего к себе сервера, он изменяет гипотезу о сетевом расстоянии от двух до четырех секунд и так далее, пока не будет найден ближайший сервер, что обеспечивает гибкость и работоспособность подхода.

После группировки серверов в подкластеры, в каждом из них выбирается Лидер, а все остальные серверы, принимаемые в одном подкластере принимают статус – Последователи. Голосование происходит следующим образом. Серверы делают запрос на заданный порт, который слушает агент. Агент понимает, что в кластере находятся три сервера и, если пришел голос только от одного сервера, значит что-то некорректно с другим сервером, или уже существует Лидер.

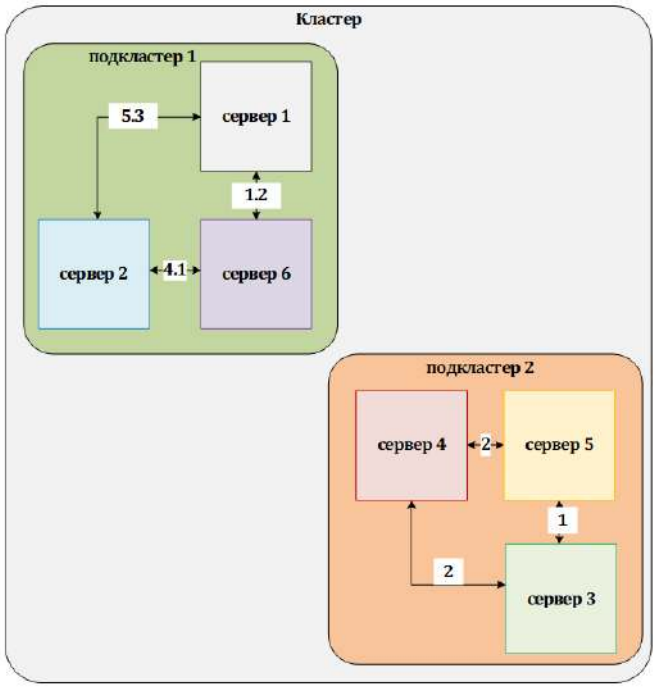

Рисунок 5.10 — Серверы, сгруппированные в подкластеры

Когда сервер перешел в состояние Лидера, он отправляет запрос Последователям с целью проинформировать их, что появился новый Лидер. Таким образом, все остальные серверы становятся автоматически Последователями. На рисунке 5.11 представлена схема определения Лидеров и Последователей.

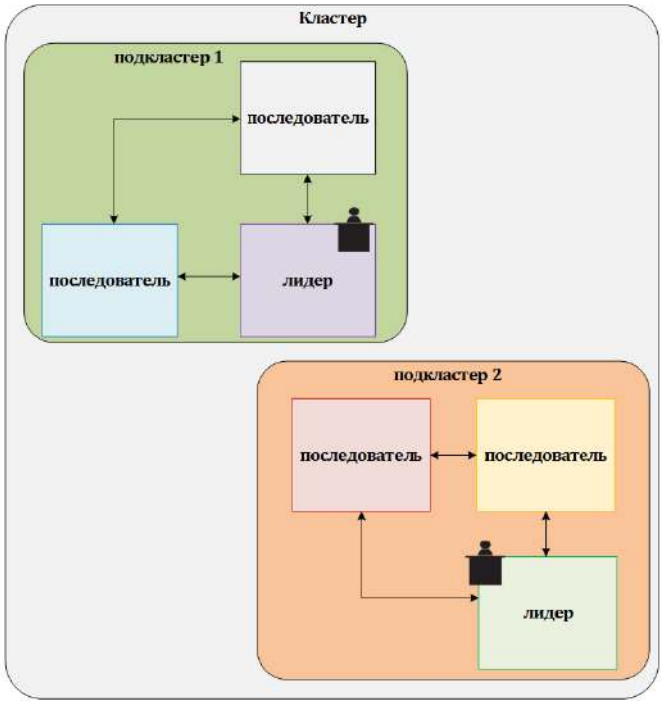

Рисунок 5.11 — Схема определения Лидеров и Последователей

После проведения голосования, каждому Лидеру подкластера необходимо настроить связь с другими Лидерами, что позволит объединить все независи-

мые компоненты (т.е. подкластеры) и обеспечит создание децентрализованной системы. Каждый Лидер знает, какие данные хранятся в его подкластере и эта информация дублируется также на другие серверы, а если Лидер выходит из строя, значит необходимо повторное голосование. Лидер играет важную роль в подкластере, в его функции входит разрешение конфликтов и потому все новые запросы или данные приходят именно Лидеру. Лидер конкретного подкластера не может общаться с Последователями другого Лидера. На рисунке 5.12 представлена схема взаимодействия в децентрализованной системе.

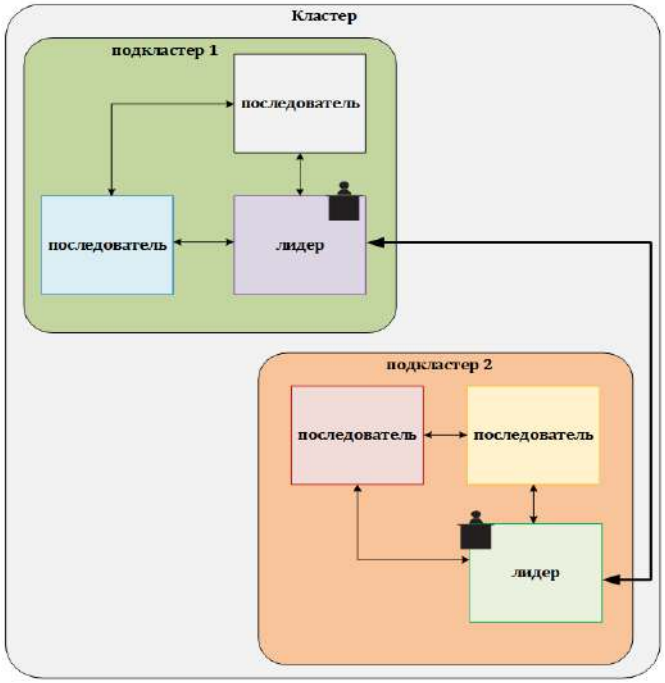

Рисунок 5.12 — Схема взаимодействия в децентрализованной системе

Преимущества использования нового предложенного метода заключается в следующем:

– обеспечивается динамичность кластера
– каждый новый сервер находит ближайшего к нему Лидера по сетевому расстоянию
– поиск нового Лидера для уже подключенного сервера с его текущим Лидером осуществляется путем удаления сервера из кластера и повторного добавления его в кластер
– благодаря контролю Последователей, Лидеры могут самостоятельно удалять Последователей, которые находятся далеко за пределами сетевого расстояния, что гарантирует устойчивость кластера

– при предлагаемом подходе каждый Лидер может связываться только с ближайшими к нему Последователями

В таблице 41 приведены сравнительные характеристики методов кластеризации.

Таблица 41 — Сравнительные характеристики методов кластеризации

Функционал	**Существующие методы**	**Предлагаемый метод**
Подготовка сервера перед объединением с кластером.	Ручная установка дополнительных инструментов. Запускает процессы администратор. Сервер подключается постоянно.	Копирование библиотеки сервиса Agent на сервер и его запуск. Сервер самостоятельно детерминирует ближайшего Лидера.
Проверка серверов путем отзыва.	Отсутствует или проверяется только сетевая доступность с кластером.	Каждый Лидер следит за Последователями. В случае плохого отзыва — Лидер удаляет их.
Смена медоидов для последователей.	Отсутствует или возможна только при вмешательстве системного администратора.	Динамическое определение Лидера для Последователей, которые откликаются дольше всех. Отсутствие вмешательства со стороны системного администратора.

Разработка алгоритмического обеспечения децентрализированной ИТС

Для эффективной разработки и обслуживания, децентрализованная ИТС, управляемая через алгоритм Raft, имеет микросервисную архитектуру. Микросервисная архитектура – вариант сервис-ориентированной архитектуры программного обеспечения, направленный на взаимодействие насколько это возможно небольших, слабо связанных и легко изменяемых модулей — микросервисов. [235; 250—253] На рисунке 5.13 представлена общая схема функционирования отдельных элементов предлагаемой децентрализованной архитектуры ИТС с пятью основными сервисами: client, storage, auth, database, agent.

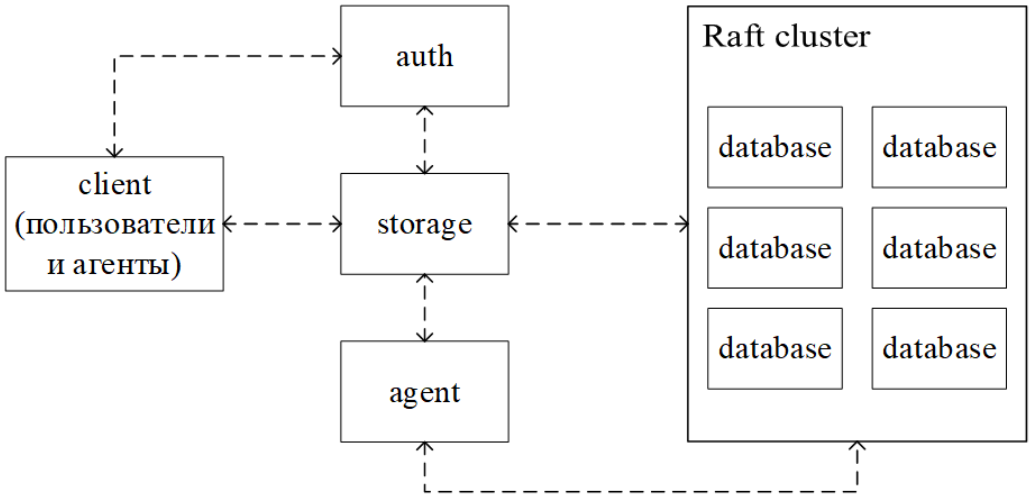

Рисунок 5.13 — Общая схема функционирования элементов децентрализованной ИТС

Сервис storage, который может быть запущен как на отдельном сервере, так и на любом сервере с базой данных, обеспечивает выполнение двух основных задач:

– получение пользователями целевой информации о состоянии транспортной сети из ИТС
– сохранение в ИТС неструктурированных данных от различных источников (агентов) ИТС (смартфоны, мультимедийные устройства автомобилей и общественного транспорты, умные светофоры, видеокамеры и пр.) различного формата.

Сервис storage передает неструктурированные блоки данных в сервис database для сохранения. Находится во взаимодействии с сервисами auth, database, agent, client.

Сервис auth, отвечает за аутентификацию и авторизацию в децентрализованной ИТС. Данный сервис позволяет регистрироваться новым пользователям в системе, производить сервису storage авторизацию пользователей с помощью выдачи токенов, подключать агентов ИТС (смартфоны, мультимедийные устройства автомобилей и общественного транспорты, умные светофоры, видеокамеры и пр.). Находится во взаимодействии с сервисами storage, client.

Сервис database отвечает за хранение данных в децентрализованной ИТС. Для обеспечения надежности функционирования реализует алгоритм Raft. Находится во взаимодействии с сервисом storage.

Сервис agent отвечает за добавление новых серверов ИТС для сервиса database. Позволяет запрашивать у облачного провайдера дополнительный сервер и запускать на нем новый сервис database. Оповещает сервис storage о добавлении нового сервера. Находится во взаимодействии с сервисами database и storage.

Сервис client включает в себя два основных модуля:

– клиентское приложение, предоставляющее возможность конечным пользователям получать целевую информацию о состоянии транспортной сети из ИТС
– библиотеки агентов ИТС, обеспечивающие сохранение неструктурированных данных.

Для того чтобы реализовать кластер с использованием алгоритма Raft, необходимо реализовать протокол связи в децентрализованной ИТС. Основной особенностью является то, что каждый агент должен работать в обе стороны и в любой момент времени может быть как в состоянии Последователя, так и в состоянии Лидера, уже относительно его состояния должны изменяться инструкции его функционала [243—246; 254]. Блок-схема перехода состояний агента изображена на рисунке 5.14.

В прототипе программного обеспечения элементов децентрализованной ИТС заложены в архитектуру пять микросервисов: client, storage, auth, database, agent. Данные макросервисы достаточно разместить на одном сервере, однако для улучшения быстродействия децентрализованной ИТС каждый из сервисов рекомендуется вынести на каждый сервер отдельно. Полная схема взаимодействия сервисов показана на рисунке 5.15. Сплошные линии отображают запросы, а прерывистые ответы сервисов. Цифрами на стрелках обозначены порядок запросов и ответов по временной шкале при типичной работе пользователя. Запросы и ответы содержат следующую информацию:

– запрос пользователя (агента) на аутентификацию (стрелка 1 на рисунке 5.15).
– при успешной аутентификации отправка пользователю/агенту (сервису client) двух токенов (стрелка 2 на рисунке 5.15). Первый токен необходим для работы с сервисом storage. Он не имеет ограничений по количеству использования, но действителен лишь в течении некоторого времени. Второй токен (одноразовый), не имеющий ограничений по времени, нужен для получения новой пары токенов от сервиса auth.

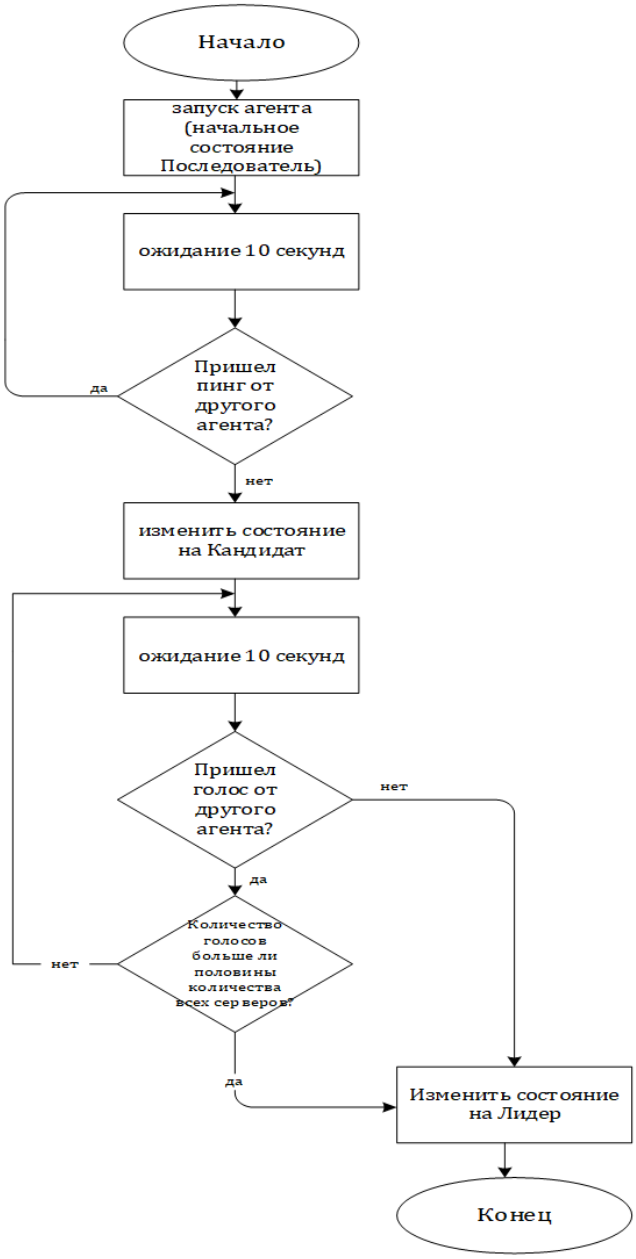

Рисунок 5.14 — Блок-схема перехода состояний агента

– запрос пользователя на получение целевой информации о состоянии транспортной сети из ИТС (стрелка 3 на рисунке 5.15). Запрос отправляется в виде формализованного сообщения.
– проверка полученного от пользователя токена через сервис auth (стрелка 4 на рисунке 5.15).
– отправка сервису storage данных о пользователе (к каким данным имеет доступ, какие операции может выполнять) в случаи успешной авторизации (стрелка 5 на рисунке 5.15).

– в зависимости от запроса пользователя сервис storage отправляет запрос к сервису database на получение целевой информации (стрелка 6 на рисунке 5.15).
– в зависимости от запроса пользователя сервис database либо отправляет сервису storage запрашиваемые данные, либо загружает и сохраняет неструктурированные данные от агентов (стрелка 7 на рисунке 5.15).

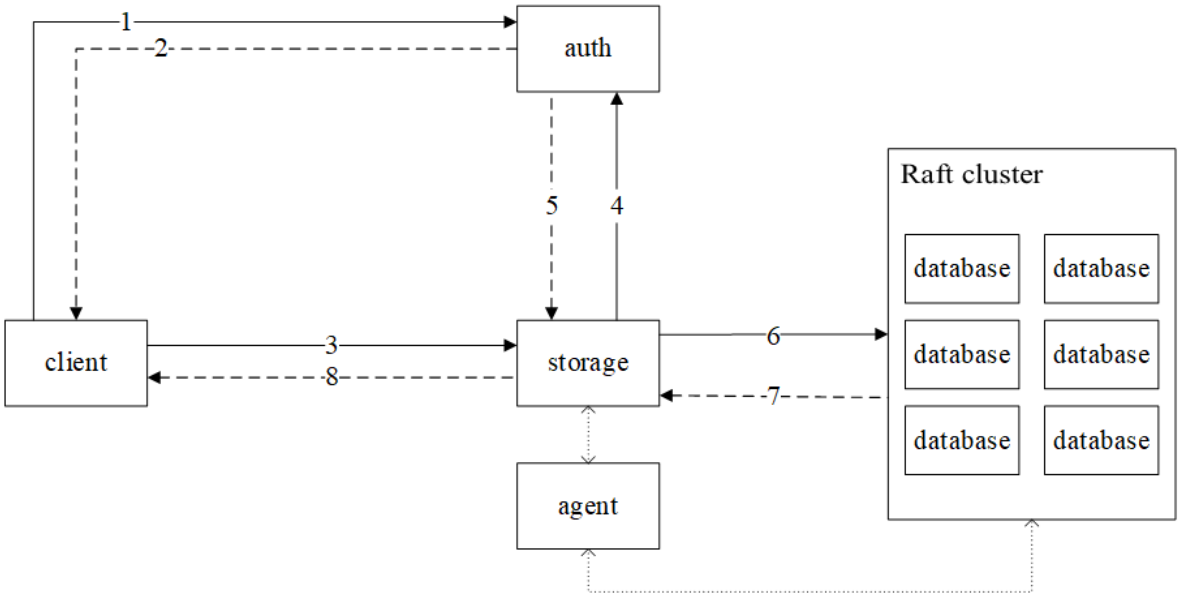

Рисунок 5.15 — Общая схема взаимодействия сервисов в децентрализованной ИТС

Описание функций и процедур основных сервисов. Для управления сервисом storage был реализован графический интерфейс пользователя. На рисунке 5.16 представлена UML диаграмма основных классов и их методов.

Класс MainWidget является основным классом управления в программе сервиса storage. Он отвечает за отображение пользовательского интерфейса и позволяет запускать и останавливать сервис storage. Управление происходит за счет вызова соответствующих методов:

– метод on_start_btn() – запускает сервер. Запуск сервера происходит за счет делегирования этого вызова классу Server.
– метод on_stop_btn() – останавливает сервер. Остановка происходит за счет делегирования этого вызова классу Server.

Класс Server отвечает за непосредственно взаимодействие с сервисом client и auth. Данный класс имеет следующие компоненты. AuthManager – отвечает за авторизацию пользователя (в общем случае и агента ИТС) с помощью взаимодействия с сервисом auth. Проверка прав пользователя происходит за счет

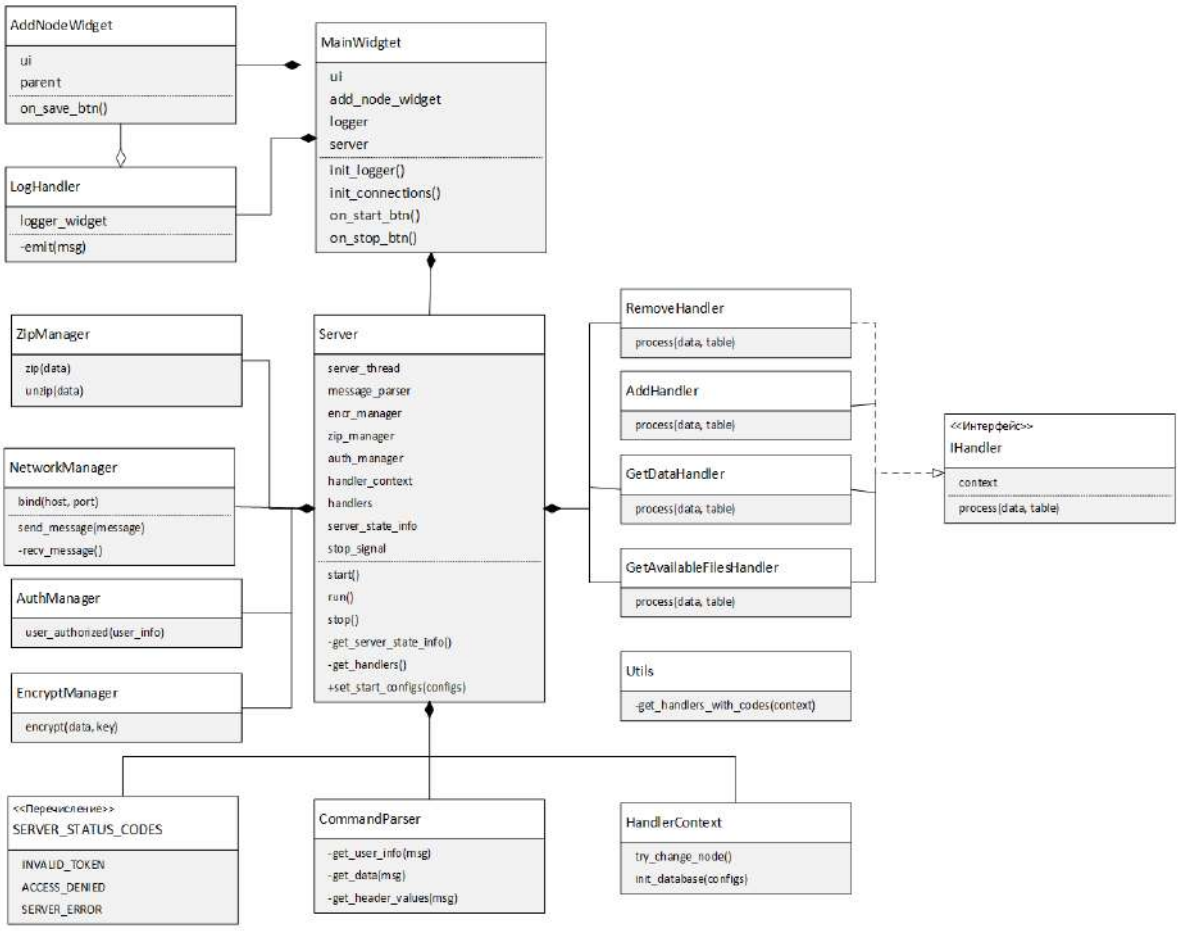

Рисунок 5.16 — UML-диаграмма классов сервиса storage

отправки сервису auth логина пользователя с предоставленным токеном. Класс NetworkManager – отвечает за взаимодействие с сервисом client посредством использования методов send_data(msg), recv_data(), bind(host, port).

Класс CommandParser производит непосредственную обработку данных, полученных от пользователя по сети. Выделяет из этого сообщения имя пользователя, токен, данные, код команды для сервера и т.д.

Класс HandlerContext отвечает за создание контекста связи с базой данных. Также отвечает за переключение между узлами базы данных при выходе из строя одного из них (метод try_change_node()). Инициализируется словарем доступных узлов базы данных вместе с данными необходимыми для подключения (имя пользователя базы данных и паролем при необходимости). Смена текущего узла базы данных происходит за счет попытки подключения к другому серверу из списка доступных серверов.

Все поддерживаемые сервисом storage операции реализованы с помощью классов, наследующих абстрактный класс IHandler. Он имеет всего один метод process (table, data), который принимает имя таблицы в базе данных и дан-

ные запроса, отправленные пользователем. Возвращает два значения: первое типа bool, означающее успешность выполнения операции, а второе данные, которые вернул сервер базы данных ИТС (по умолчанию возвращается значение None, если данных нет).

Сервис client позволяет конечному пользователю взаимодействовать с децентрализованной ИТС. На рисунке 5.17 представлены UML диаграммы основных классов сервиса client.

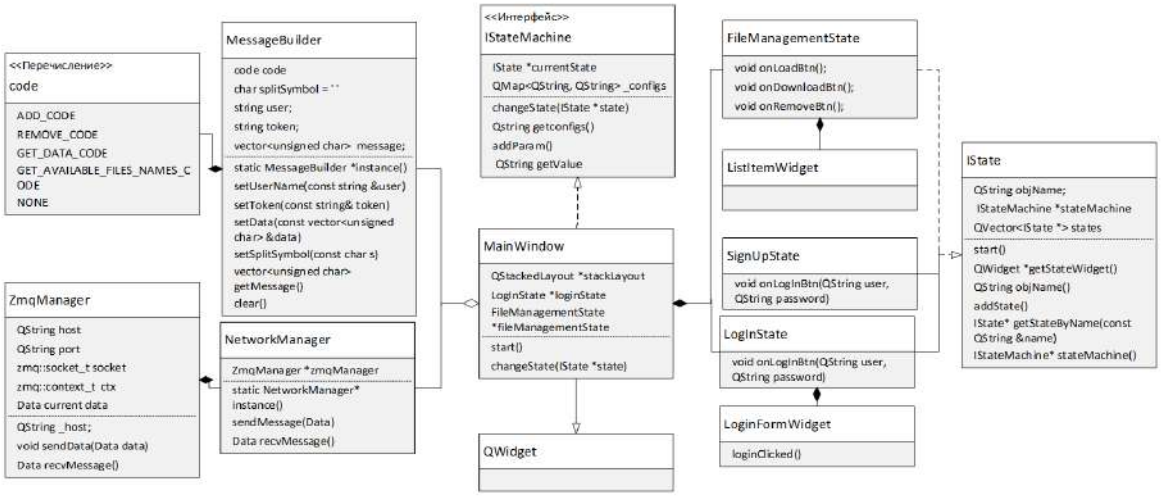

Рисунок 5.17 — UML-диаграмма классов сервиса client

Сервис client основан на реализации паттерна «State Machine». В каждый момент времени клиентская программа может находится в одном конечном состоянии. Всего реализовано три состояния:

– состояние регистрации
– состояние входа в систему
– состояние получения целевых данных от ИТС.

Паттерн «State Machine» состоит из класса состояние которого изменяется в зависимости от различных условий и классов, которые реализуют это состояние. В данной реализации основным классом чье состояние будет меняться является MainWindow. Классы конкретных состояний должны реализовывать методы абстрактного класса IState. Смена состояний класса MainWindow происходит за счет вызова метода changeState(IState *).

Сервис database отвечает за непосредственное хранение неструктурированных данных от различных источников (агентов) ИТС (смартфоны, мультимедийные устройства автомобилей и общественного транспорты, умные светофоры, видеокамеры и пр.) различного формата и реализацию алгоритма

Raft. Исходя из перечисленных задач в качестве основной базы данных для реализации прототипа программного обеспечения была выбрана RethinkDB. RethinkDB – распределенная документно-ориентированная СУБД с открытым исходным кодом, сохраняющая данные в бессхемном JSON-формате. Ориентирована на применение для веб-приложений, требующих интенсивных обновлений базы данных. Отличительная особенность – поддержка реактивного стиля: база данных позволяет клиентскому приложению подписываться на изменения тех или иных данных. [227—230; 255; 256]

RethinkDB состоит из двух программных компонентов: непосредственно сервера базы данных и клиентского драйвера. На данный момент официально выпущены клиентские драйверы для языков программирования: Java, JavaScript, Python, Ruby. Сервис storage использует драйвер для языка программирования Python. Сервер базы данных поставляется уже в скомпилированном виде, то есть в виде отдельной программы для операционных систем семейства Linux.

Реализация не имеет графического пользовательского интерфейса, а управление происходит за счет передачи аргументов командной строки. В данной главе будет рассмотрена одна конфигурация сервера RethinkDB, а именно в режиме кластера с применением алгоритма Raft. Для того чтобы база данных работала в режиме кластера, необходимо как минимум три сервера базы данных, из которых один должен быть запушен в режиме «Лидер», а остальные как «Последователи». В разработанном прототипе децентрализированной ИТС сервис database состоит из шести серверов, из которых один запушен как «Лидер», а остальные как «Последователи». В случаи выхода из строя одного из серверов, RethinkDB с помощью алгоритма Raft сам переназначит «Лидера».

Сервис agent, является дополнительным сервисом. Он необходим для повышения надежности хранения неструктурированных данных децентрализованной ИТС. Сервис database функционирует в штатном режиме только при работоспособности половины серверов кластера. Сервис agent осуществляет мониторинг сервиса database и в случаи выхода из строя половины серверов включает в кластер резервный сервер. После чего происходит оповещение сервиса storage о включении нового сервера. Алгоритм работы при включении резервного сервера представлен на рисунке 5.18.

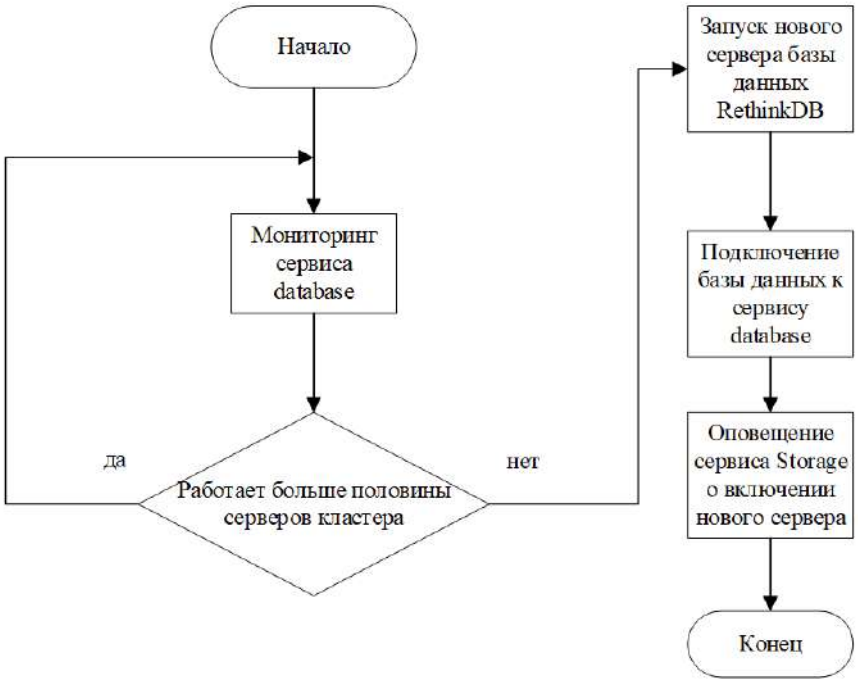

Рисунок 5.18 — Алгоритм работы при включении резервного сервера в основной кластер

После того как работоспособность сервиса database будет восстановлена сервис agent самостоятельно отключит резервный сервер от основного кластера, оповестит об этом сервис storage и перейдет в режим мониторинга. Таким образом, полный алгоритм работы представлен на рисунке 5.19.

Сервис «Agent» состоит из четырех основных классов:

– «Agent» - является основным, классом управляющим ходом выполнения программы
– «CommandParser» - отвечает за обработку аргументов командной строки
– «DatabaseManager» - отвечает за организацию связи с основным кластером сервиса database, подключения и отключения резервного сервера
– StorageManager» - необходим для оповещения сервиса storage о включении или отключении резервного сервера.

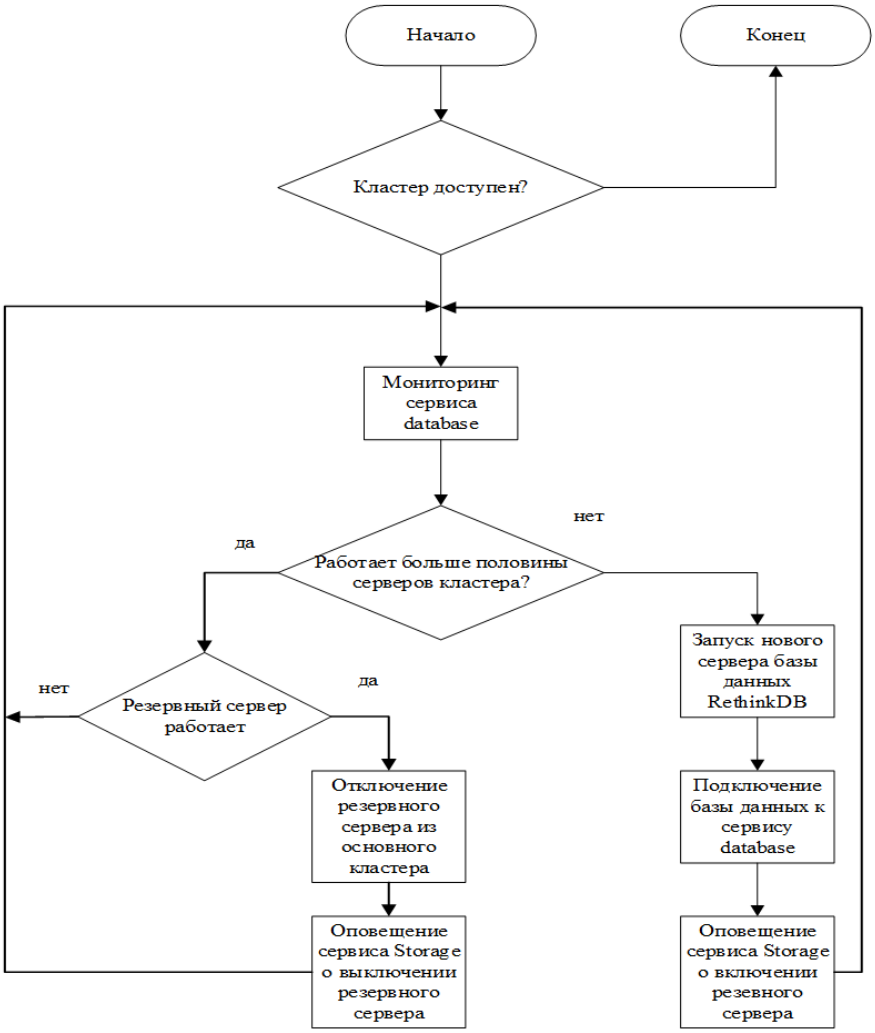

Рисунок 5.19 — Алгоритм работы сервиса agent

Класс «AgentServer» реализует вышеописанный алгоритм работы сервиса agent. Для запуска мониторинга сервиса database необходимо инициализировать объект класса данными об резервном сервере (ip-адрес, порт, логин, пароль) и вызвать метод start(). Цикл обработки не прекратится пока внутренняя переменная self.stop_signal не будет установлена в состояние «False».

Тестирование разработанного алгоритмического обеспечения. Прототип развёрнут на 7 физических хостах на виртуальных машинах с операционной системой Ubuntu 22.04. Запущены: 7 серверов database (RethinkDB), сервисы storage и auth, 7 клиентов client (имитаторы агентов ИТС). Обмен сообщениями осуществляется через ZeroMQ. В качестве сценариев выбрана нормальная работа (одиночные загрузки файлов), нагрузочное тестирование (7 клиентов непрерывно записывают данные) и отказоустойчивость (отключение двух из 7 серверов database на пике нагрузки).

Для тестирования пропускной способности децентрализованной системы ИТС реализована имитация передачи данных от агентов ИТС в виде загрузки файлов произвольного размера. После того как сервис client был запущен, система готова ко второму этапу тестирования, а именно к загрузке данных ИТС на сервер в обычном режиме. Производительность работы сервиса storage представлена на рисунке 5.20. На рисунке по оси абсцисс отложено время эксперимента, а по оси ординат - количество операций в секунду, выполняемых сервисом storage при загрузке данных в децентрализованную ИТС. Начальный участок графика, на котором значения близки к нулю, соответствует этапу развертывания и выхода компонентов на установившийся режим, а именно запуску клиента, установлению сетевых соединений, инициализации сессий обмена, а также подготовке сервиса хранения к приему потока данных. После завершения этого этапа наблюдается резкий переход к рабочему режиму, при котором система начинает обрабатывать поток загрузки и формируется характерное плато производительности. Колебания значений на рабочем участке графика обусловлены спецификой распределенной записи. Данные передаются неравномерными порциями, подтверждение операций зависит от сетевых задержек и текущего состояния узлов кластера, а внутренняя репликация и синхронизация между экземплярами database вносят дополнительную вариативность во времени завершения операций. Таким образом, график на рисунке 5.20 отражает типичный переход распределенной системы от стадии инициализации к стационарному режиму с последующими флуктуациями производительности около некоторого среднего уровня. Это согласуется с используемым сценарием тестирования, в котором после запуска сервиса client выполнялась имитация передачи данных от агентов ИТС в виде загрузки файлов произвольного размера.

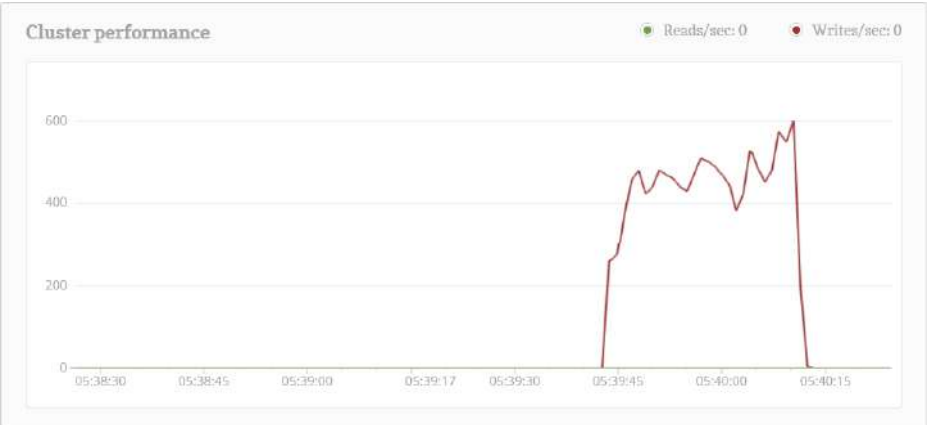

Рисунок 5.20 — Производительность работы сервиса storage при загрузке данных на сервер

Для проведения третьего этапа тестирования было запущено пять экземпляров сервиса пользователей. Запись данных на сервер осуществлялась в потоковом режиме. Для тестирования в произвольный момент отключали два из семи серверов, и проводилась проверка работоспособности системы. На рисунке 5.21 представлено состояние кластера database в штатном режиме при активной нагрузке с пятью работающими клиентами в нормальном режиме работы. В центральной части панели отображается временной график производительности кластера. По оси абсцисс на нем отложено время, а по оси ординат - число операций записи и чтения в секунду. Рост кривой в начале интервала связан с тем, что к моменту построения графика нагрузка на систему уже была приложена, и кластер перешел из состояния низкой активности в режим интенсивной обработки запросов от пяти клиентов. Последующий колебательный характер кривой является нормальным для распределенной системы, поскольку фактическая скорость записи зависит от сетевых задержек, внутренней очереди запросов, подтверждения репликации и текущего распределения ролей между узлами кластера.

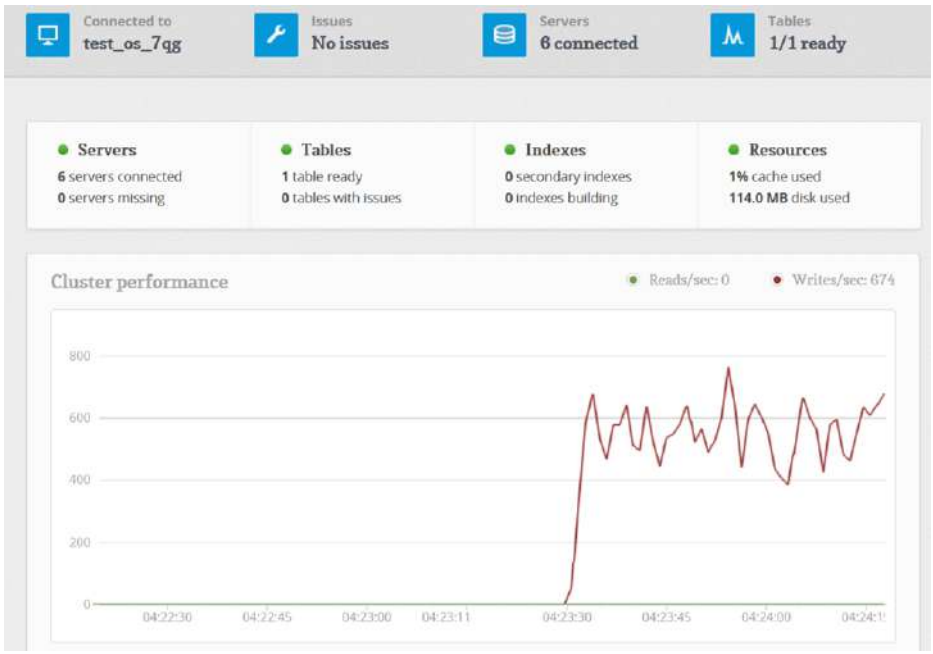

Рисунок 5.21 — Состояние кластера сервисов database, функционирующем в штатном режиме

На рисунке 5.22 показано состояние того же кластера после отключения двух из семи серверов database на пике нагрузки. Несмотря на отказ части узлов, форма графика остается близкой к наблюдаемой в штатном режиме, поскольку система не потеряла кворум и сохранила возможность обработки

записей. Кратковременная задержка или переходный участок после отказа объясняются необходимостью обнаружения недоступности узлов, выполнением процедуры переизбрания лидера и перенастройкой маршрутизации запросов внутри кластера. После завершения этой переходной фазы производительность восстанавливается до уровня, сопоставимого с исходным, что подтверждает сохранение работоспособности системы в условиях частичного отказа. Именно этим объясняется вывод о том, что количество операций записи в секунду после отключения двух серверов существенно не уменьшилось. Дополнительно это согласуется с зафиксированным по логам временем смены лидера порядка 730 мс, то есть переходный процесс имеет малую длительность по сравнению с общим временем тестового сценария.

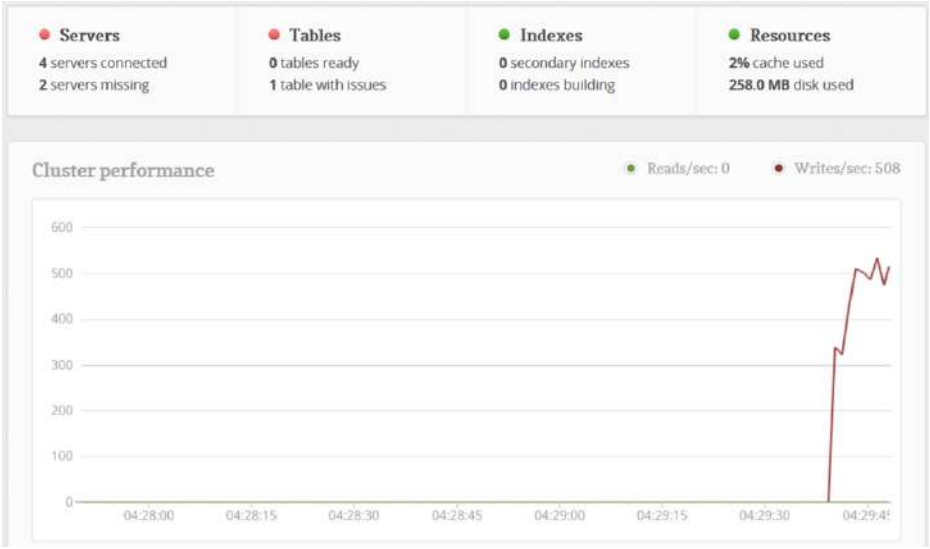

Рисунок 5.22 — Состояние кластера сервисов database, при отключении двух серверов

Сервис storage позволяет осуществить логирование основных событий системы. Благодаря этому можно вычислить время, затраченное на смену «Лидера» в сервисе database. Для этого необходимо вычислить разницу времени между логами об ошибках записи и первом успешном логе записи. Таким образом время, затрачиваемое системой на смену «Лидера», составляет 730 мс.

Сквозная производительность при отключении двух узлов не снизилась — кворум был сохранён, а переизбрание прошло прозрачно для клиентов. По логам storage разница между первым неуспешным и первым успешным записями составила около 730 мс. Время подготовки сервера уменьшилось примерно с 121 минут до 15 минут за счёт самоподключения агента, а загрузка 10 ГБ

данных ускорилась с 46,2 до 24,2 минут, что согласуется с уменьшением внутрикластерных задержек. В таблице 42 приведены показатели эффективности улучшенного предлагаемого метода кластеризации.

Таблица 42 — Усредненные показатели эффективности предлагаемого метода

Показатель	**Существующие подходы**	**Предлагаемый подход**	**Улучшения %**
Время подготовки сервера ИТС	121 минута (установка дополнительных инструментов, конфигурация, запуск процессов администратором)	15 мин (копирование библиотеки сервиса Agent, запуск администратором или перезагрузка сервера)	87.5
Загрузка 10 Гб разнородных данных ИТС	46.2 мин.	24.2 мин	47.6
Failover лидера	-	$\sim$ 730 мс	-

Сокращение времени подготовки обусловлено отказом от ручной конфигурации и использованием механизма автоматического присоединения по RTT. Ускорение загрузки данных объясняется локализацией репликации и минимизацией пути подтверждения команд. Время переключения лидера порядка 100 миллисекунд соответствует типичным значениям для Raft при корректно выбранных тайм-аутах. Ограничениями исследования являются лабораторная среда и ограниченный профиль нагрузок. В том числе для географически распределённых сетей с переменным джиттером и потерями потребуется дополнительная настройка параметров и расширение сценариев тестирования.

5.3 Разработка метода динамического распределения нагрузки для повышения отказоустойчивости ИТС

Кластерные вычислительные системы, принадлежащие к классу процессорных вычислительных систем (МПВС), играют ключевую роль в современной информатике, находя широкое применение в самых различных областях, от научных исследований до управления критически важными инфраструктурами. Одной из таких областей являются интеллектуальные транспортные системы (ИТС), которые все активнее интегрируются в повседневную жизнь, обеспечивая безопасность, эффективность и управление транспортными потоками

[5; 22; 257—259]. Учитывая сложность задач обработки данных и высокие требования к надежности таких систем, разработка методов повышения отказоустойчивости становится одной из первоочередных задач для кластерных вычислительных систем, используемых в ИТС. [21; 49]

Однако обеспечение высокой надежности кластерных вычислительных систем в условиях быстрого роста их сложности не всегда возможно только за счет использования высоконадежной элементной базы. [39] Частые аппаратные и программные сбои, а также неопределенность в характере нагрузки, характерная для ИТС, требуют применения более гибких и интеллектуальных подходов. В таких системах отсутствие отказоустойчивости может не только снизить производительность, но и привести к критическим сбоям, которые могут напрямую повлиять на безопасность и функциональность городской инфраструктуры. Это особенно важно в контексте ИТС, где сбои в работе могут нивелировать преимущества их внедрения, такие как снижение заторов, оптимизация транспортного потока и предотвращение аварий. [247; 260—265]

Перспективным подходом к повышению отказоустойчивости таких систем является разработка методов эффективного управления вычислительными процессами на основе отказоустойчивого планирования. Эти методы учитывают вероятность отказа вычислительных модулей и позволяют динамически перераспределять задачи между операционными модулями [29; 38]. Цель таких методов состоит не только в обеспечении надежности системы, но и в поддержании необходимого уровня производительности и качества работы даже в случае сбоев. Важно отметить, что использование присущей многопроцессорным системам избыточности ресурсов вычислительных модулей может значительно снизить потребность в резервных ресурсах. Вместо жестких стратегий резервного копирования используются гибкие механизмы адаптации, позволяющие найти баланс между производительностью и надежностью. [248; 249; 266—268]

В этом разделе рассматриваются подходы к обеспечению отказоустойчивости кластерных вычислительных систем для задач обработки информации, связанных с ИТС. Основное внимание уделяется методам, позволяющим перераспределять ресурсы и задачи в случае сбоев, минимизируя их влияние на общую производительность системы, что особенно важно для работы ИТС в режиме реального времени.

Обеспечение отказоустойчивости ВС с помощью перераспределения задач

Один из методов обеспечения отказоустойчивости вычислительных систем заключается в перераспределении задач между операционными вычислительными модулями в случае отказа, при этом допускается определенная степень снижения производительности системы. Этот подход предполагает переконфигурацию архитектуры системы: вышедшие из строя сервера удаляются, а задачи перераспределяются. Системы, использующие этот метод, называются системами с постепенной деградацией. Когда доступно восстановление виртуальной машины, процесс включает обнаружение неисправностей, замену модулей и перераспределение задач. Такой подход важен для стабильной работы критически важных приложений. [247—249; 260; 265; 266; 269]

В данной главе рассматриваются вычислительные системы с не восстанавливаемыми модулями, где при отказах перераспределение задач осуществляется только между существующими работными модулями без введения дополнительных ресурсов. Системы данного типа классифицируются как избыточные, используя аппаратную или программную избыточность для повышения отказоустойчивости. Предполагается, что вычислительные модули могут находиться в одном из двух состояний: работоспособности или отказа. Рассмотрение вычислительных систем с восстанавливаемыми модулями оговаривается отдельно.

Далее используется следующее понятие структурного состояния системы:

Структурное состояние (s-состояние) вычислительной системы, которое состоит из n вычислительных модулей, представляется в виде вектора:$s_v = (\sigma_1, \ldots, \sigma_n)$, где $\sigma_i = 0$, если i-й элемент работоспособен, $\sigma_i = 1$, если i-й элемент отказал $i = 1, \ldots, n$, $\nu = 1, \ldots, 2^n$.

Выделим следующие s-состояния:

- $s^0 = (0, \ldots, 0)$ – начальное (все вычислительные модули работоспособны);
- s_v – искажённое – любое s-состояние, не являющееся начальным: $s_v \neq s^0$;
- $s^1 = (1, \ldots, 1)$ – конечное (отказ всех ВМ).

В таком случае множество всех состояний ВС будет: $\Omega_s = \{s_v\}, \quad v = 1, \ldots, 2^n$.

На основе указанного критерия отказа вычислительная система делится на непересекающиеся подмножества. Одно включает работоспособные модули, продолжающие выполнять задачи, а второе — модули, находящиеся в отказе и неспособные к выполнению задач: $\Omega_s = S \cup S^F$, где S – подмножество работоспособных s-состояний, называемое в дальнейшем структурной областью работоспособности системы, S^F – подмножество неработоспособных s-состояний (то есть соответствующих отказу ВС).

Для вычислительной системы без механизмов отказоустойчивости множество S ограничивается стартовым состоянием s^0 — система функционирует без сбоев. В отказоустойчивой системе в S присутствует минимум одно нарушенное состояние s_ω, не тождественное s^1, которое представляет собой состояние полной неработоспособности. Такие состояния отражают способность системы функционировать с допустимой деградацией в условиях отказов: $S = \{s^0\} \cup S_r^\omega$, где S_r^ω — некоторое множество искажённых работоспособных s-состояний.

Очевидно, что в общем случае при возможности отказов любых ВМ система в произвольный момент времени t может находиться в любом из s-состояний: $s_v = (\sigma_1, \ldots, \sigma_n), \quad \nu = 1, \ldots, 2^n$.

Отказоустойчивость достигается перераспределением задач отказавших модулей на работоспособные модули. Рассматривается система с фиксированным множеством задач $\Lambda^{(0)}$ без задания порядка выполнения и взаимосвязей. $\Lambda^{(0)}$ — все задачи, которые в начальном состоянии s^0 могут быть выполнены при имеющихся ресурсах.

Считаем, что для состояния s^0 задано начальное распределение задач (РЗ) между всеми ВМ, при котором каждый модуль M_i способен выполнять некоторое подмножество задач: $\Lambda_i^{(0)} = \{U_1^i, U_2^i, \ldots, U_{m_i}^i\}$, которые называются собственными задачами модуля M_i.

При отказе каких-либо ВМ система переходит в одно из искажённых состояний s_ω и утрачивает способность выполнять собственные задачи отказавших ВМ. Сохранить способность системы выполнять эти задачи можно с помощью такого перераспределения выполняемых системой задач по оставшимся работоспособным ВМ, при котором собственные задачи отказавших ВМ будут выполняться в работоспособных ВМ.

Статическое и отказоустойчивое перераспределение задач

Есть множество S работоспособных состояний системы, определяющих требуемую степень отказоустойчивости вычислительной системы, которая должна быть достигнута. [270]

Согласно формуле, S_r^{ω} - множество искажённых работоспособных s-состояний. Для того чтобы сохранить работоспособность ВС в каждом из состояний множества S_r^{ω}, то есть обеспечить выполнение системой задач отказавших модулей при выполнении заданных требований, необходимо для каждого из этих состояний осуществлять определённое перераспределение выполняемых системой задач между работоспособными ВМ всякий раз, когда система после отказов вычислительных модулей попадает в данное состояние.

При подготовке к работе для каждого состояния s_r^{ω}, принадлежащего заданному множеству S_r^{ω} искажённых работоспособных состояний, вычисляется такой план D_r^{ω} распределения задач по работоспособным ВМ, при котором достигается наилучшее значение некоторого заданного показателя качества функционирования системы при заданных ограничениях на другие выбранные показатели качества и на ресурсы системы.

Указанный план представляет собой план рационального назначения задач в данном состоянии. Все задачи, подлежащие запуску на виртуальной машине M_i согласно любому из планов D_r^{ω}, в сумме образуют множество $\Omega_i^{(\omega)}$.

Размещение задачи U_j на ВМ M_i определяется её принадлежностью множеству $\Omega_i^{(\omega)}$. При этом неважно, хранится ли модуль задачи в памяти машины или подгружается динамически.

Под основным размещением задач (РмЗ) будем понимать такое распределение, при котором каждая задача U_j из начального множества $\Omega^{(0)}$ закрепляется ровно за одним вычислительным модулем. Задачи множества $U_j \in \Omega^{(0)}$ назовём основными. План основного РмЗ обозначим матрицей D^0 — это двоичная матрица размера $L \times n$: $D^0 = \|d_{ji}^0\|_{L\times n}$, где $d_{ji}^0 = 1$, если задача U_j назначена к выполнению на ВМ M_i и размещена в ней; $d_{ji}^0 = 0$ — в противном случае. Считаем, что начальному состоянию s^0 соответствует основное РмЗ.

Под дополнительным РмЗ понимаем матрицу $Y = \|y_{ji}\|_{L\times n}$, описывающую размещение резервных копий задач. Она получается путём логического объединения планов $D_r^{\omega} = \|d_{ji}^{\omega}\|_{L\times n}$, рассчитанных для всех искажённых, но работоспособных состояний s_r^{ω}: $y_{ji} = \bigcup_{s_r^{\omega}\in S_r^{\omega}} d_{ji}^{\omega}$.

Матрица ОУ-РмЗ $Z = \|z_{ij}\|_{L\times n}$ строится из двух компонентов — основного $D^0 = \|d^0_{ji}\|_{L\times n}$ и дополнительного $Y = \|y_{ji}\|_{L\times n}$. Элемент Z определяется правилом: $z_{ij} = x_{ij} \oplus y_{ij}$. Здесь операция $\oplus$ используется в смысле объединения информативных признаков с целью формирования расширенного представления состояния системы, обеспечивающего учет нескольких источников данных в единой модели.

Такое решение обеспечивает резервирование: одна и та же задача получает копии на разных ВМ системы, при этом в каждой ВМ допускается не более одной копии данной задачи. В исходном состоянии s^0 запускаются лишь основные задачи согласно начальному плану размещения D^0 (основному РмЗ). [271]

Если из-за отказов части ВМ система переходит в искажённое, но работоспособное состояние s^r_ω, режим работы следует перестроить по плану РмЗ D^ω_r: активировать резервные копии задач на доступных ВМ и выполнять их вместе с основными.

Если в системе с восстановлением ВМ неисправные модули возвращаются к работе, увеличивается состав доступных ВМ, и система переключается в состояние с меньшим числом отказов, активируя план размещения для этого состояния. [272]

Система и её ВМ снабжены ресурсами, достаточными для выполнения задач как по начальному РмЗ, так и по каждому плану D^ω_r. Все полученные планы D^ω_r хранятся в базе данных планов РмЗ и используются процедурами ПЗ/ОУ. При переходе системы в любое из состояний s^r_ω формируется запрос к этой базе данных и из неё извлекается план РмЗ для данного состояния, в соответствии с которым происходит выполнение.

Динамическое распределение задач с приоритетами

Кластерная вычислительная система (КВС) относится к классу систем с клиент-серверной архитектурой, где главный узел системы (MasterClient) выполняет функции клиента, а N вычислительных узлов (Nodes) представляют собой серверный массив. Взаимодействие между узлами осуществляется через протокол прикладного уровня HTTP в формате JSON.

Полагаем, что система состоит из N идентичных узлов с одинаковым объёмом оперативной памяти, и известно множество $W = \{U_j, j = 1, \ldots, L\}$ всех задач, предназначенных для выполнения в этой системе. Для каждой задачи U_j

этого множества известны значения определённых параметров, которые могут использоваться при динамическом распределении задач по узлам.

В данной диссертации разрабатывается процедура динамического распределения задач, при которой учитываются вес задачи ω_j и время t_j, необходимое для её решения в одном узле (т.е. время выполнения в узле соответствующего процесса).

Вес задачи ω_j — это параметр, характеризующий приоритет выполнения задачи от пользователя. Этот параметр критически важен для ИТС, где, например, обработка данных о ДТП или нарушениях имеет гораздо более высокий приоритет, чем статистический анализ транспортного потока.

Значение параметра t_j, используемое при разработке динамического распределения задач, определяется на этапе тестирования КВС. На каждой итерации тестирования фиксируется время, которое выполнялась каждая задача. Далее для каждой такой задачи находится среднее арифметическое время её выполнения, полученное значение является параметром t_j.

Решение задач в КВС может осуществляться в виде ряда сессий. Каждой сессии соответствует решение определённого набора A задач, выделенного из множества W.

При описании процесса решения задач в КВС используется понятие состояния узла в некоторый момент времени, которое характеризуется его готовностью к работе, а также суммарным значением t_j назначенных задач, выполняемых на данном узле. Это состояние позволяет оценить загруженность узла и его способность принимать новые задачи. [241]

Информация о состояниях всех узлов КВС хранится в памяти главного узла (клиента) и обновляется в определенные моменты времени, такие, как посылка запросов от клиента к узлам на решение задач. Обновление информации клиента о состоянии каждого узла осуществляется на основе ответа узла на запрос клиента о его состоянии.

Рассмотрим кластерную вычислительную систему, состоящую из главного узла C и множества вычислительных узлов N. Состояние вычислительного узла n_i в момент времени t характеризуется:

- $\text{status}(n_i) \in \{\text{Active}, \text{Online}, \text{Offline}\}$ – текущий статус узла;
- $T_{\text{зат.}}(n_i)$ – суммарное время, затраченное узлом на выполнение уже назначенных ему задач;
- $\text{task}_{\text{pod}}(n_i)$ – множество обработчиков, загруженных в память узла.

Главный узел C хранит глобальное состояние системы в виде множества объектов состояния.

Процесс функционирования кластерной вычислительной системы для сессии A:

1. **Шаг 1. Инициализация сессии.** Клиент формирует список задач на сессии и сохраняет список в локальной памяти.
2. **Шаг 2. Обнаружение узлов.**
   а) Клиент инициирует процедуру обнаружения путём отправки запроса на каждый потенциальный узел.
   б) Каждый узел, способный ответить, возвращает клиенту статус.
   в) Клиент обновляет своё глобальное состояние системы, формируя множество работоспособных узлов.
3. **Шаг 3. Распространение задач.**
4. **Шаг 4. Динамическое распределение и выполнение задач.**
5. **Шаг 5. Пока $A \neq \varnothing$:**
   а) Клиент обновляет информацию о состоянии всех узлов и применяет алгоритм динамического распределения.
   б) Сортирует список не назначенных задач по убыванию их веса.
   в) Сортирует множество доступных узлов по возрастанию $T_{\text{зат.}}(n_i)$.
   г) Отправляет запрос на узел n_i, и задача удаляется из списка.
   д) Каждый узел n_i при получении задач выполняет её и отправляет результат выполнения клиенту по запросу.
   е) Клиент сохраняет полученные результаты в локальную базу данных.

Алгоритм динамического распределения задач с приоритетами

Рассмотрим кластерную вычислительную систему в начальный момент времени сессии решения задач. Система состоит из множества вычислительных узлов: $N = \{n_1, n_2, \ldots, n_N\}$, где каждый узел $n_i \in N$ характеризуется своим состоянием. В рамках данной процедуры узлы считаются идентичными по вычислительной мощности.

Пусть A – множество задач, предназначенных для решения в текущей сессии, где $A \subseteq W$. Тогда $T_A(s_v)$ – необходимое время для выполнения заданного набора задач A при распределении, определяемом принятым алгоритмом, когда система находится в состоянии s_v. Получим, что T_A^0 – наименьшее достижимое время выполнения набора задач A в состоянии s^0: $T_A^0 = \min\{T_A(s^0)\}$.

Также для системы задаётся максимально допустимое время выполнения заданного набора задач A, обозначаемое как $T_A^* > T_A^0$. Эти значения определяются требованиями пользователя и спецификой приложения (ограничения на режим работы в реальном времени). [263]

Областью работоспособности системы S является множество всех её состояний s_i, в которых время выполнения набора задач A не превышает T_A^*: $S = \{s_i \mid T_A(s_i) \leqslant T_A^*\}$.

Для отказоустойчивой системы S включает начальное состояние s^0 и множество искажённых состояний S^ω: $S = \{s^0\} \cup S^\omega$, где $S^\omega = \{s_v \neq s^0 \mid T_A(s_v) \leqslant T_A^*\}$.

Событие считается отказом кластерной вычислительной системы, если в результате отказов узлов система переходит в состояние s_v, такое что $T_A(s_v) > T_A^*$, т.е. $s_v \notin S$.

Предположение 1. Отказы узлов являются независимыми событиями, происходящими с одинаковой интенсивностью λ.

Вероятность P_T того, что КВС будет функционировать без отказа в течение интервала времени T, определяется как вероятность того, что все состояния s_i, в которых система пребывает в течение T, принадлежат области работоспособности S: $P_T = P_T(s^0) + \sum_{s_v \in S^\omega} P_T(s_v)$, где $P_T(s^0)$ – вероятность пребывания в начальном состоянии s^0 в течение времени T, $P_T(s_v)$ – вероятность пребывания в искажённом состоянии $s_v \in S^\omega$ в течение времени T.

Если все искажённые состояния s_v, в которых число отказавших узлов k удовлетворяет условию $1 \leqslant k \leqslant d$, принадлежат S^ω $(T_A(s_v) \leqslant T_A^*)$, то вероятность безотказной работы системы за время T определяется формулой: $P_T = \sum_{k=0}^{d} C_N^k p^k (1-p)^{N-k}$, где вероятность отказа одного узла за время T: $p = 1 - e^{-\lambda T}$.

Требуется разработать алгоритм динамического распределения задач, который будет гарантировать функционирование КВС в пределах заданной области работоспособности S, обеспечивая требуемый уровень отказоустойчивости.

Формальное описание алгоритма динамического распределения задач с приоритетами.

1. **Шаг 1. Инициализация.** Формирование списка задач A для сессии.
2. **Шаг 2. Отсортировать список задач A по убыванию весов ω_j,** получив упорядоченный список U, где $\omega_{j_1} \geqslant \omega_{j_2} \geqslant \ldots \geqslant \omega_{j_{|A|}}$.
3. **Шаг 3. Цикл распределения задач (пока $U \neq \varnothing$):**
   а) Обновить информацию о состоянии всех узлов N. Состояние каждого узла n_i включает его текущий статус (работоспособен/отказавший) и суммарное затраченное им время.
   б) Отфильтровать отказавшие узлы.
   в) Определить множество работоспособных узлов N_{work}.
   г) Отсортировать множество N_{work} по возрастанию затраченного времени $T_{\text{зат.}}(n_i)$, получив упорядоченный список $M = (n_{i_1}, n_{i_2}, \ldots, n_{i_{|N_{\text{work}}|}})$, где $T_{\text{зат.}}(n_{i_1}) \leqslant T_{\text{зат.}}(n_{i_2}) \leqslant \ldots \leqslant T_{\text{зат.}}(n_{i_{|N_{\text{work}}|}})$.
   д) Назначить первую задачу из списка U на выполнение первому узлу n_{i_1} из списка M.
4. **Шаг 4. Завершить при $U = \varnothing$.** Задачи распределены по узлам в соответствии с приоритетами важности и текущей загруженности.

В начальном состоянии s^0 распределение задач может выполняться с помощью любого алгоритма, направленного на минимизацию времени выполнения заданного набора задач и позволяющего обеспечить «наилучшее» время T_A^0, в частности с помощью алгоритма, выполняющего распределение задач на основе затрачиваемого процессорного времени и соответствующей сортировки задач и узлов по этому критерию, как это предусмотрено в данном алгоритме. Это даёт возможность распределить задачи таким образом, чтобы затрачиваемое каждым узлом процессорное время было примерно одинаковым. Такой подход, по сравнению со случайным распределением задач по узлам, обеспечивает меньшее общее время выполнения всех задач.

Процедуры тестирования кластерной ВС

Процедура тестирования кластерной ВС направлена на определение допустимого числа отказавших модулей, при котором происходит отказ всей системы. Отказ системы фиксируется, если превышается установленное до-

пустимое время выполнения всех задач. Также в процессе тестирования вычисляется среднеарифметическое время выполнения каждой задачи.

Алгоритм тестирования заключается в поэтапном принудительном отключении узлов (моделировании отказов) до момента превышения допустимого времени выполнения задач. При этом значение числа отключённых узлов, вызвавших отказ системы, фиксируется и принимается как минимальное количество отказов, приводящих к полной недееспособности системы.

Формальное описание алгоритма тестирования кластерной вычислительной системы с разработанным модулем динамического распределения задач с приоритетами.

Входные данные:

– N: Множество вычислительных узлов.
– A: Множество задач, предназначенных для выполнения в сессии.
– T_A^*: Максимально допустимое время выполнения задач A.
– Разработанный алгоритм динамического распределения задач с приоритетами.
– Процедура определения ожидаемого времени выполнения задач.

Выходные данные:

– d: Максимальное число отказавших узлов.

Алгоритм:

1. **Инициализация**
    а) $\text{status}(n_i) \leftarrow$ Active. Установить все узлы в работоспособное состояние.
    б) Выполнить процедуру определения ожидаемого времени выполнения всех задач.
    в) Установить счётчик отказавших узлов: $f \leftarrow 0$.
    г) Установить флаг завершения: test_complete $\leftarrow$ False.
2. **Цикл тестирования, пока test_complete = False:**
    а) Проверка условия завершения: Если $f = N$ (все узлы отключены), то test_complete $\leftarrow$ True. Перейти к шагу 3.
    б) Распределение и выполнение задач. Запустить сессию выполнения задач A с использованием разработанного алгоритма динамического распределения задач с приоритетами. Запустить таймер.

в) Имитация отказа. Принудительно отключить один произвольный работоспособный узел n_i и установить $\text{status}(n_i) \leftarrow$ Offline. Установить счётчик: $f \leftarrow f + 1$.

г) Наблюдение. Дождаться либо выполнения всех задач, либо превышения времени T_A^*.

д) Оценка результата. Зафиксировать фактическое время выполнения в текущем состоянии с отказавшими узлами.
Если $T_A(s_v) \leqslant T_A^*$, то: система работоспособна при f отказах. Продолжить цикл, вернувшись к шагу 2.1.
Иначе $T_A(s_v) > T_A^*$: Система считается отказавшей в состоянии s_v. Установить test_complete $\leftarrow$ True. Установить $d \leftarrow f - 1$. (Максимальное число отказавших узлов, при котором система ещё работоспособна.)

3. **Завершение:** Вернуть значение d.

Для тестирования разработанных программных модулей используем КВС, состоящую из 8 узлов и 3 различных наборов задач, включающих в себя задачи обработки данных с камер наблюдения: ДТП, парковка в запрещённом месте, проезд на красный свет:

– 1 набор состоит из 800 задач,
– 2 набор состоит из 500 задач,
– 3 набор состоит из 300 задач.

Возьмём 4 сценария функционирования КВС:

1. Распределение по случайной логике, все узлы активны.
2. Распределение по разработанной логике, все узлы активны.
3. Распределение по случайной логике, 4 узла отказывают постепенно.
4. Распределение по разработанной логике, 4 узла отказывают постепенно.

В качестве порогового значения временем отказа примем время выполнения задач при распределении по случайной логике, когда все узлы системы находятся в активном состоянии. В случае, если КВС достигает этого порогового время, все оставшиеся задачи отбрасывают, распределение останавливается. Результаты тестирования по указанным сценариям представлены в таблице 43.

Исходя из проведенных тестов можно сделать вывод о том, что разработанный алгоритм распределения задач повышает отказоустойчивость системы с динамическим распределением задач. В случае данного теста разработанный

Таблица 43 — Результаты тестирования.

Кол-во задач	Сценарий №1	Сценарий №2	Сценарий №3	Сценарий №4
800 задач	96 секунд (все задачи решены)	42 секунды (все задачи решены)	96 секунд (отброшено 322 задач)	96 секунд (отброшено 37 задач)
500 задач	66 секунд (все задачи решены)	29 секунд (все задачи решены)	66 секунд (отброшено 126 задач)	66 секунд (отброшено 12 задач)
300 задач	36 секунд (все задачи решены)	20 секунд (все задачи решены)	36 секунд (отброшено 84 задачи)	34 секунды (все задачи решены)

динамический алгоритм распределения с приоритетами на специализированных задачах справился в среднем примерно на 38% быстрее, чем при случайном распределении, а следовательно, при отказах вычислительных узлов, за некоторый промежуток времени КВС решит больше задач с максимальным значением веса.

Существующие универсальные и сложные платформы оркестрации Kubernetes и Apache Mesos предназначены для широкого спектра задач: от масштабирования веб-приложений до управления сложными микросервисными архитектурами. Их основная цель эффективное и отказоустойчивое управление жизненным циклом множества разнородных приложений.

Разработанный метод представляет из себя специализированную систему с четко определенными допущения предметной области (ИТС). Основной отличительной особенностью является эффективная работа в сценариях, где приоритетом является быстрое выполнение независимых задач с разным уровнем важности (весом) в условиях сбоев с минимальными накладными расходами.

Приведем основные сценарии в которых разработанный подход динамического распределения задач с приоритетами имеет преимущества перед системами Kubernetes и Apache Mesos:

1. **Обработка потоков независимых задач с приоритетами в реальном времени.** Система обработки данных от дорожных камер для распознавания различных инцидентов, где каждое видео-событие это отдельная задача. Разработанный алгоритм динамического распределения задач в явном виде учитывает приоритет выполнения. Например,

задача по распознаванию опасного нарушения или ДТП будет обработана в первую очередь, даже если поступила позже. При отказе узлов система фокусируется на завершении наиболее важных задач в течение допустимого времени, отбрасывая наименее приоритетные. В то время как системы Kubernetes и Apache Mesos в первую очередь балансируют нагрузку и обеспечивают доступность обработчиков, но не имеют встроенной, простой и эффективной модели для присвоения и учёта весов каждой отдельной задачи на уровне планировщика.

2. **Простые и легковесные кластерные системы с ограниченными ресурсами.** Распределённая обработка данных на базе небольшого кластера 8–16 серверов, где требуется быстрая развёртываемость и минимальная сложность. Разработанная система построена на простом клиент-серверном взаимодействии. В то время как системы Kubernetes и Apache Mesos являются большими и сложными. Их установка, настройка и поддержка требует значительных ресурсов и экспертизы. Для решения описанных задач они являются избыточными и приводят к дополнительным накладным расходам на администрирование и ресурсы самой платформы.
3. **Системы с предсказуемым набором задач и фиксированным временем выполнения.** Пакетная обработка, где заранее известен набор задач, и для каждой задачи можно определить среднее время выполнения. Разработанный алгоритм использует `operateTime` для сортировки узлов по загруженности. Планировщики Kubernetes оптимизируют размещение обработчиков на узлах, но не всегда фокусируются на минимизации общего времени выполнения конкретного пакета независимых задач.

В таблице 44 приведены ключевые отличия разработанного метода и определяющие сценарии.

Исходя из выше изложенного можно сделать вывод, что разработанный подход не является конкурентом Kubernetes и Apache Mesos в их основной нише. Он представляет специализированное решение для конкретного класса задач. Область применения в сценариях: небольшие кластеры, выполняющие пакетную или потоковую обработку большого количества независимых задач, где критически важная скорость обработки и низкие накладные расходы. Именно

Таблица 44 — Ключевые отличия разработанного метода.

Характеристика	Разработанный подход	Kubernetes / Mesos
Архитектура	Легковесный, простой клиент-сервер (HTTP/JSON)	Сложная, многоуровневая оркестрация (API-сервер, etcd, kubelet и т.д.)
Сложность	Низкая, легко развернуть	Высокая, требует значительной экспертизы
Основной критерий оптимизации	Минимизация времени выполнения набора задач с учетом веса и времени	Обеспечение доступности, балансировка ресурсов, масштабируемость
Учет приоритета (веса)	Явный, встроенный параметр в алгоритме распределения	Возможен (через приоритеты обработчиков), но не является основой алгоритма
Сценарий отказа	Фокус на выполнении максимального числа важных задач в течение допустимого времени	Фокус на перезапуске обработчиков и восстановлении состояния сервиса
Тип задач	Независимые, однородные (по типу)	Разнородные, сложные (микросервисы, базы данных, батч-задачи)

такие требования зачастую предъявляются к платформам обработки данных в ИТС.

5.4 Разработка метода генерации синтетических данных транспортного потока для проведения тестирования ИТС

При разработке распределенной интеллектуальной системы мониторинга транспортных потоков сталкиваются с серьезной проблемой: отсутствием подходящих методик тестирования. Это ограничение в первую очередь связано с

трудностями в получении значительных объемов аутентичных данных о транспортных потоках, что приводит исследователей к поиску синтетических данных в качестве альтернативного решения. [50]

В ходе данного исследования было обнаружено несколько существующих генераторов тестовых данных, однако, ни один из них не удовлетворял специфическим требованиям предметной области. [10] Можно выделить основные четыре критерия для удовлетворения потребностей создания синтетических наборов данных о состоянии транспортных потоков:

1. Модели данных, которые в точности повторяют реальные сценарии
2. Точное представление эксплуатационных характеристик транспортного средства
3. Возможность одновременной генерации нескольких уникальных параметров
4. Непрерывная генерация потока данных в заданные сроки

Основная проблема в данной области характерна для исследователей и специалистов по изучению данных, которые часто сталкиваются с ситуациями, когда реальные данные либо недоступны, либо ограничены в связи с правилами конфиденциальности [9; 226]. Генерирование синтетических данных предлагает эффективное решение, позволяющее создавать искусственные наборы данных, которые эффективно имитируют характеристики реальных данных, сохраняя при этом требования конфиденциальности. [36]

Синтетические данные, по определению, представляют собой искусственно созданную информацию, которая имитирует реальные модели данных, не будучи полученной из реальных событий. Хотя для создания синтетических данных существуют различные инструменты и алгоритмы, их основная ценность заключается в проверке моделей и сравнительном анализе поведения реальных и смоделированных данных. Ключ к успешной реализации заключается в том, чтобы синтетические данные в точности соответствовали статистическим свойствам и закономерностям подлинных данных, сохраняя при этом полезность для целей тестирования и разработки. [273]

Создание синтетических наборов данных осуществляется на основе систематического подхода, основанного на моделировании характеристик реальных данных. Эти модели призваны отразить и воспроизвести как статистические свойства, так и основные закономерности, обнаруженные в исходных наборах

данных [274]. В основе создания синтетических данных лежат две основные методологии:

1. Статистические методы
   - Сосредоточены на оценке выборочных распределений по исходным данным
   - Генерирование синтетических данных на основе статистических свойств
2. Методы машинного обучения
   - Использование реальных наборов данных для обучения моделей
   - Создание синтетических данных на основе изученных закономерностей и взаимосвязей

Ключевой особенностью генерации синтетических данных является их не детерминированный характер - каждый процесс генерации создает уникальные наборы данных, сохраняя при этом основные характеристики исходных данных. Эта изменчивость в сочетании с не идентифицируемым характером синтетических данных делает их особенно ценными в ситуациях, когда необходимо соблюдать правила защиты конфиденциальности. В настоящее время синтетические данные в значительной степени освобождены от действия правил защиты конфиденциальности, что еще больше повышает их полезность в самых разных областях применения.

Синтетические данные можно разделить на три различных типа, каждый из которых отвечает различным требованиям к полезности: полностью синтетические данных (не содержат исходных данных, практически исключает риск повторной идентификации, обеспечивают максимальную защиту конфиденциальности благодаря полному синтезу), частично синтетические данные (заменяет синтетическими значениями только чувствительные элементы, использует множественную интерполяцию и подходы на основе моделей, эффективно для обработки недостающих данных в реальных наборах данных), гибридные синтетические данные (сочетает реальные и синтетические элементы данных).

Каждая категория предлагает уникальные преимущества и компромиссы между конфиденциальностью данных, удобством использования и вычислительными требованиями, что позволяет выбрать подход, наилучшим образом соответствующий конкретным потребностям.

Несмотря на то, что синтетические данные предоставляют значительные преимущества для анализа данных, необходимо решить ряд важных проблем: сложность генерации (создание высококачественных синтетических данных требует сложных методологий, сложные взаимосвязи в реальных данных трудно воспроизвести в точности); несоответствие данных (поддержание согласованности данных по нескольким переменным представляет значительные трудности); проблемы валидации (заинтересованные стороны часто сомневаются в надежности тестирования на основе синтетических данных); ограничения производительности (алгоритмы, обученные на синтетических данных, могут иметь скрытые предубеждения).

Эти проблемы подчеркивают важность тщательной разработки и тестирования стратегий при использовании синтетических данных в производственных средах. Принимая решение о внедрении синтетических данных, организации должны сопоставить эти ограничения с преимуществами.

Существующие инструменты генерации синтетических данных

DATPROF предоставляет множество инструментов для управления данными, но здесь мы сосредоточимся исключительно на «DATPROF Privacy». Этот инструмент примечателен тем, что позволяет маскировать тестовые данные и генерировать синтетические наборы данных, что особенно актуально для данной работы.

К достоинствам DATPROF Privacy можно отнести эффективность работы с большими наборами данных, удобный интерфейс, отличную службу поддержки и совместимость с многочисленными системами управления базами данных, такими как Oracle, MySQL и MariaDB.[227; 229; 240]

Однако у системы есть и недостатки. Она может потребовать настройки для оптимизации производительности, бесплатные пробные версии доступны только для сотрудников компаний, а многократное применение одних и тех же инструментов маскировки ко всей базе данных может потребовать дополнительных усилий.

Другой инструмент служащий для генерации синтетических наборов данных SQL Data Generator, который позволяет генерировать большой объем данных с возможностью применения различных генераторов, при этом обеспечивая согласованность данных в различных таблицах.

Следующее программное обеспечение для генерации синтетических данных, который стоит отметить – IBM InfoSphere, предназначенный для оптимизации и автоматизации процессов управления тестовыми данными, ускоряющий этап тестирования при разработке программного обеспечения. Его основная функция — эффективно генерировать тестовые данные и выявлять потенциальные проблемы, которые могут привести к сбоям в работе приложения.

Однако этот инструмент имеет существенные ограничения. Он совместим исключительно с операционными системами Windows, что ограничивает его использование в различных вычислительных средах. Кроме того, его сложная архитектура может создавать проблемы при внедрении и сопровождении.

Учитывая специфические требования к генерации реалистичных данных о транспортном потоке и поддержке взаимодействия клиент-сервер, разработка собственного решения представляется наиболее подходящим подходом. Это решение особенно актуально для распределенной системы мониторинга дорожного движения с высокой нагрузкой, требующей таких специфических данных, как телеметрия, скорость, направление движения и местоположение транспортных средств.

В данном исследовании для генерации набора данных о транспортных потоках используется статистическая модель. Статистический метод обеспечивает более надежную основу для создания необходимых моделей и характеристик транспортных потоков, гарантируя при этом согласованность и реалистичность данных. Разработка индивидуального решения позволяет точно контролировать процесс генерации данных и гарантирует выполнение всех специфических требований системы мониторинга дорожного движения, избегая при этом ограничений, присущих существующим коммерческим решениям.

Архитектура модуля мониторинга интеллектуальной транспортной системы

Начнем с определения ключевых требований к модулю мониторинга ИТС, который должен обеспечивать масштабируемость и отказоустойчивость.

Приложение должно использовать контейнеризацию для упрощения развертывания в облачных средах. Kubernetes следует использовать в качестве оркестратора для обеспечения быстрого масштабирования системы. Поскольку система интегрируется с облачными вычислениями, важно рассмотреть подходы к построению вычислительной архитектуры в рамках Kubernetes.

Для повышения отказоустойчивости система должна включать механизмы репликации данных и контроля наличия протокола heartbeat у компонентов. Компоненты приложения должны быть модульными, чтобы обеспечить гибкое горизонтальное масштабирование вместо монолитной структуры. Обработка данных должна быть устроена для легкой адаптации к росту входящего потока. Для защиты данных приложение должно поддерживать идентификацию, аутентификацию и авторизацию пользователей и сервисов. [237; 245; 253; 275]

Инфраструктура разработанной интеллектуальной транспортной системы развернута внутри кластера Kubernetes, что способствует лучшей облачной интеграции и автоматизации. Процесс мониторинга системы включает две роли: пользователей и сервисов-источников данных (модуль генерации синтетических данных), действующих по RESTful-архитектуре через HTTP/HTTPS. Схема взаимодействия модулей с системой изображена на рисунке 5.23.

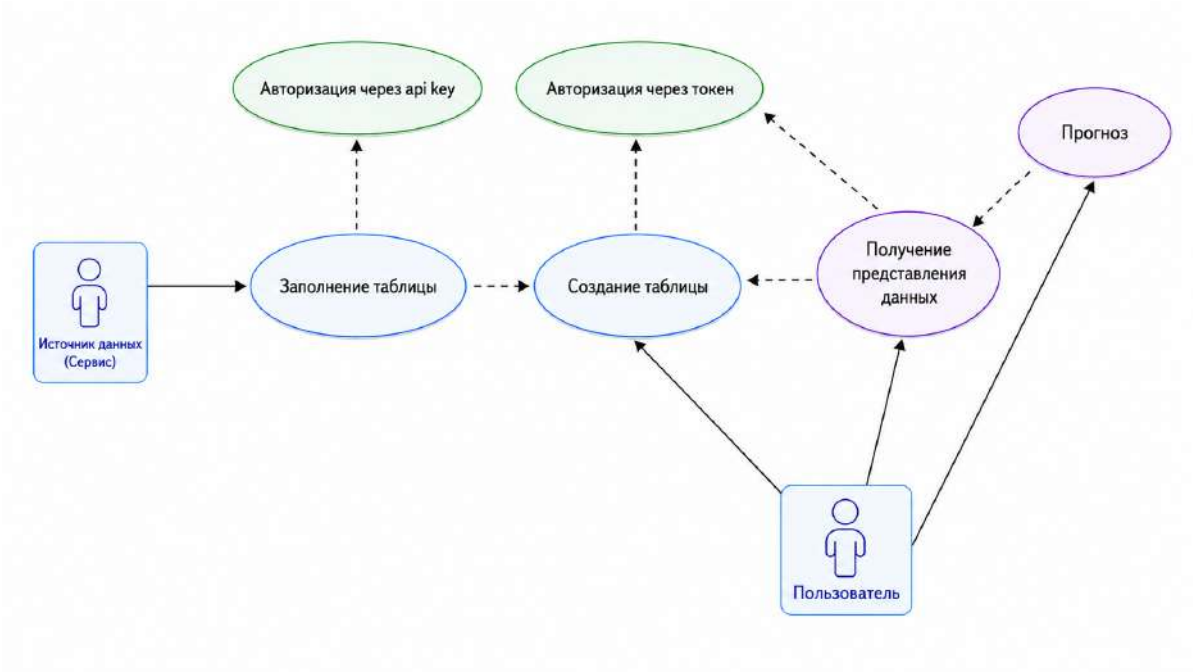

Рисунок 5.23 — Сценарии взаимодействия с системой

Система включает три ключевых модуля: авторизация, создание таблиц и отправка метрик, а также обработка метрик. Модуль авторизации обеспечивает безопасность через обработку ключей, переданных пользователями или источниками данных. Пользователи используют токены, параметры которых задаются в конфигурации системы, в то время как сервис применяет API ключ, который хранится в базе данных с метаданными и может быть обновлен по запросу. Модуль обрабатывает создание пользователей, получение токенов,

управление API ключами и предоставляет информацию о пользователях и сервисах.

Модуль создания таблиц и отправки метрик отвечает за запросы на создание таблиц - абстракций, объединяющих метрики. Этот модуль передает метрики от сервиса в очередь для дальнейшей валидации другим модулем. Интерфейс предоставляет доступ к информации о таблицах и сохраненным метрикам.

Модуль обработки метрик отвечает за валидацию и сохранение данных в базе. При возрастании загрузки, связанной с увеличением потока метрик, он может горизонтально масштабироваться. Загрузка увеличивает эффективность обработки метрик из очереди. Модули обработки метрик и создания таблиц работают на общем сервере из-за их тесного взаимодействия.

К другим важным компонентам системы относится хранилище данных, включающее данные о пользователях, сервисах и таблицах. Оно поддерживает горизонтальное масштабирование и репликацию, размещаясь в отдельном кластере внутри Kubernetes для удобной облачно ориентированной интеграции. Очередь метрик, в совокупности с репликацией обработчиков, поддерживает стабильную скорость обработки. Организация компонентов системы и их взаимодействие показаны на схеме рисунка 5.24.

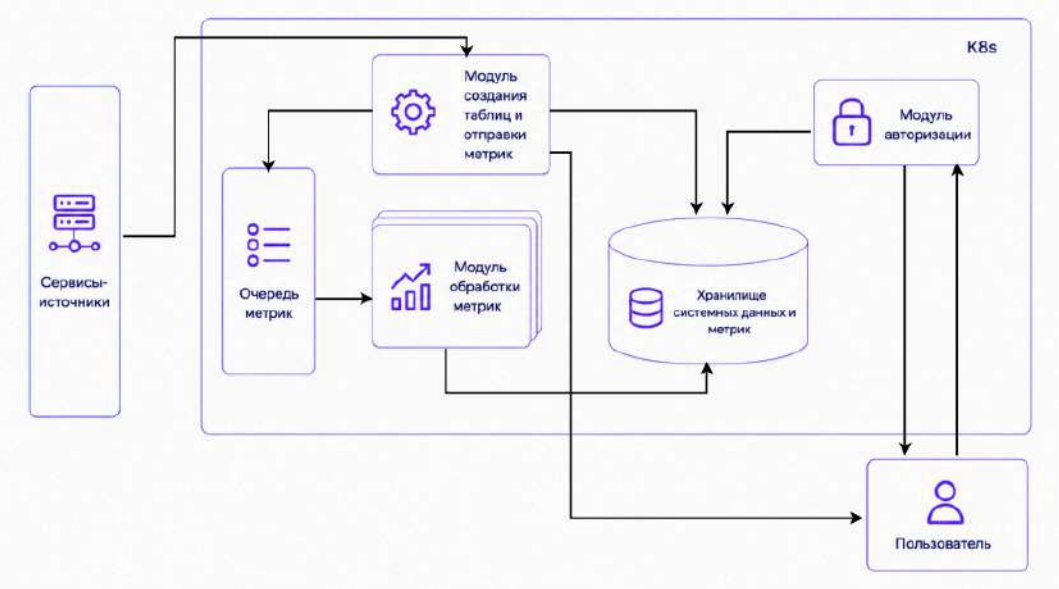

Рисунок 5.24 — Организация компонентов системы

Сценарии горизонтального масштабирования модулей

Можно выделить три разных случая масштабирования компонентов системы. При росте нагрузки на очередь получаемых метрик и большом скоплении

данных в ней масштабироваться будет модуль обработки метрик. Очередь метрик, организованная с помощью брокеров сообщений, распределяет данные между репликами модуля обработки метрик. Эти реплики воспринимаются как подписчики (консьюмеры) очереди. В результате масштабирования обработчиков скорость разбора очереди увеличится, за счет параллельной обработки N метрик, где N - это количество обработчиков. [231; 232]

С ростом скорости обрабатываемых данных может наблюдаться рост объема сохраняемых метрик и, соответственно, деградация скорости получения метрик и их визуализации. В таком случае логично добавлять в кластер реплики модуля таблиц, ответственного за получение метрик и визуализацию. Для того, чтобы нагрузка от запросов на получение метрик равномерно распределялась между копиями модуля таблиц, необходимо наличие балансировщика нагрузки. Kubernetes поддерживает целый ряд подобных решений (например, в рамках различных реализаций компонента Ingress). [241; 276]

По мере увеличения скорости обработки очереди метрик увеличится и количество операций записи в хранилище. Чтобы сбалансировать нагрузку на каждый узел хранилища, можно увеличить количество реплик. Это означает добавление дополнительных узлов в кластер хранилища, а не просто дублирование баз данных или отдельных таблиц. Однако такое масштабирование оправдано только тогда, когда кластер хранилища имеет достаточное количество частей данных (шардов). Если на каждом узле хранилища находится по одному шарду из каждой таблицы, потребуется переконфигурация кластера хранилища. В Kubernetes существуют решения, которые позволяют проводить такие изменения без простоя системы. [230; 277]

В системе мониторинга база данных организована в виде таблиц, каждая из которых следует одной из четырех структурных моделей: пользователь, сервис, таблица и экземпляр таблицы. Это не значит, что количество таблиц совпадает с числом моделей. Наоборот, для каждой записи в таблице Table создается отдельная таблица с уникальным названием в формате “instance<Идентификатор записи из таблицы Table>”. Такой подход требует поддержания целостности между данными в таблице Table и соответствующими экземплярами. Поэтому операции по добавлению записи в Table и созданию таблицы на основе модели Instance выполняются в рамках одной транзакции. Удаление таблиц происходит по аналогичной схеме. Описательные поля колонок таблиц из Table и конкретных элементов из Instance содержат динамические

объекты в формате JSON, и проверка их соответствия производится модулем обработки входящих метрик.

Доменные модели системы мониторинга во многом соответствуют моделям данных из хранилища. Основных модели так же четыре: пользователь, сервис, таблица и ряд. Ряд представляет собой значение, хранящееся в экземпляре таблицы. Ключевое отличие между моделями - это наличие жестко типизированных представлений описаний колонок и их значений в ряду (в отличие от динамических JSON в хранилище). Жесткая типизация этих полей позволяет валидировать добавляемый ряд на соответствие описанию колонок в Table. Стоит также отметить, что для каждой из основных доменных моделей (кроме пользователя) проставляются значения createdBy (идентификатор пользователя-создателя экземпляра сущности) и created (дата создания экземпляра сущности). Идентификатор для каждой сущности представляет собой UUID и генерируется на стороне модулей авторизации и таблиц.

Для создания хранилища системы мониторинга требуется платформа, которая способна справляться с постоянно увеличивающейся нагрузкой. Одним из таких решений является Hadoop.

Основой работы Hadoop служит HDFS — распределенная файловая система, и модель распределенных вычислений MapReduce. MapReduce позволяет программам автоматически распределяться по кластеру аппаратного обеспечения. HDFS спроектирована для обработки больших объемов данных и обеспечивает отказоустойчивость путем репликации, сохранения кодов ошибок и копирования метаданных на вторичный узел имен. Система также диагностирует себя, следит за состоянием узлов и отличается высокими показателями производительности среди решений для распределенных вычислений.

Тем не менее, у Hadoop есть недостатки, заставляющие искать альтернативы. Прежде всего, он неэффективен для вычислений в реальном времени, что требует использования дополнительных решений вроде Spark, усложняющих развертывание.

Yugabyte — это высокопроизводительная распределенная СУБД с поддержкой реляционной модели с транзакционностью, NoSQL и хранилища в оперативной памяти. Она может быть полностью развернута в определенных облачных кластерах Kubernetes, что делает Yugabyte привлекательным выбором для облачных проектов, так как позволяет централизованно управлять экосистемой приложения, не жертвуя производительностью. Yugabyte созда-

ет кластер хранилища внутри Kubernetes и позволяет масштабировать его с помощью параметра replication factor, используя стандартные компоненты инфраструктуры Kubernetes.

Ключевая черта Yugabyte — поддержка реляционных баз данных, NoSQL баз, и быстродействующего хранилища ключ-значение, использующего оперативную память. Эти решения доступны через интерфейсы PostgreSQL, Cassandra и Redis.

Кроме того, Yugabyte обеспечивает отказоустойчивость, безопасный доступ к данным и мониторинг внутренних компонентов. Кластер может распределяться по различным геолокациям, а встроенная балансировка нагрузки равномерно распределяет шарды по всему кластеру, что повышает скорость обработки без необходимости вмешательства администратора. [47] По этим причинам хранилище данных в разрабатываемой системе основано на Yugabyte. [228; 255]

Разработка модуля генерации синтетических данных о транспортных потоках (сервис-источник)

В архитектуре системы реализован клиент-серверный RESTful API для обеспечения бесперебойной связи между сервисом-источником и сервером назначения. Для обеспечения непрерывной подачи синтетических данных был разработан REST-клиент, способный автоматически генерировать и отправлять POST-запросы через заданные интервалы. Каждый запрос содержит JSON-данные, включающие параметры, описывающие текущее состояние транспортного средства.

Основой этой реализации является многопоточная генерация данных. Мощные возможности многопоточности Java позволяют нескольким программным компонентам работать параллельно, значительно повышая общую эффективность системы. Такой подход к параллельной обработке позволяет выполнять различные задачи одновременно, максимально используя вычислительные ресурсы и повышая производительность.

Многопоточная архитектура выполняет две важные функции. Во-первых, она позволяет системе генерировать большие объемы тестовых данных, что крайне важно для оценки стабильности целевой системы в условиях высоких нагрузок. Во-вторых, она облегчает одновременную работу нескольких генераторов, каждый из которых реализует различные модели поведения

транспортного средства. Такой подход к параллельной генерации обеспечивает всесторонние сценарии тестирования при сохранении оптимальной производительности.

Управление жизненным циклом потоков осуществляется в рамках структурированной системы, в которой каждый поток работает независимо, но при этом координируется в рамках общей архитектуры системы. Каждый поток управляет собственными процессами генерации и передачи данных, в то время как основное приложение организует их выполнение и управляет распределением ресурсов. Такой подход обеспечивает эффективную генерацию данных, поддерживая стабильность системы и предотвращая конфликты ресурсов. Данная реализация эффективно уравновешивает необходимость генерации больших объемов данных с управлением системными ресурсами, создавая надежное и масштабируемое решение для генерации синтетических данных о дорожном движении.

Входными данными параметры начального состояние транспортного средства. Выходные данные генерируется на основе начальных ограничений и зависят от входных данных. Параметрами модели является распределение дискретной случайной величины, которая является функцией перехода. Эти параметры гарантируют, что генерируемые синтетические данные точно отражают реалистичное движение транспортных средств благодаря учету установленных правил и статистических вариаций.

Придерживаясь этих предположений и ограничений, генератор синтетических данных стремится получить надежные и реалистичные данные о транспортных средствах, отвечающие специфическим потребностям системы мониторинга движения.

Работа модели основана на дискретных 1-секундных интервалах, в пределах которых выполняются все расчеты. Транспортное средство в модели может либо продолжать движение прямо, либо выполнить поворот. Повороты разрешены только тогда, когда скорость транспортного средства находится в диапазоне от 10,8 км/ч до 32,4 км/ч. Вероятность принятия решения о повороте: продолжение движения прямо – 80 %; поворот в любом направлении одинаково вероятен и имеет вероятность 10 % для каждого. Если скорость транспортного средства не попадает в заданный диапазон для поворота, оно будет продолжать движение только прямо. Модель учитывает специфические принципы ускорения и замедления, основанные на эмпирически определенных ограничениях скорости.

Любые изменения скорости должны соответствовать этим ограничениям, чтобы обеспечить реалистичное поведение автомобиля. Условия зависят от текущего диапазона скорости автомобиля: 57,6 км/ч - 61,2 км/ч (55% шансов сохранить текущую скорость, 45% шанс замедлиться); 21,6 км/ч - 57,5 км/ч (40% шанс сохранить текущую скорость, 30% шанс ускориться, шанс замедлиться); 10,9 км/ч - 21,5 км/ч (45% шанс сохранить текущую скорость, 35% шанс ускориться, 20% шанс замедлиться); 3,6 км/ч - 10,8 км/ч (55% шанс сохранить текущую скорость, 45% шанс ускориться).

Разработка сервиса-источника для генерации данных о транспортном потоке

Согласно объекто-ориентированной парадигме программирования было создано 2 класса CarInfo и CoordInfo. Соответственно первый класс содержит идентификационный ключ, получаемый от сервера, номер и скорость транспортного средства, а также метод обновления координат. Второй класс содержит координаты широты, долготы и направление движения. [13; 41]

Программный интерфейс приложения представляет собой REST API, в качестве протокола передачи данных используется HTTP/HTTPS. Интерфейс предоставляется двумя узлами кластера системы: auth-node и tables-node. Для того, чтобы внешние запросы попадали на нужные узлы необходимо добавить в кластер контроллер трафика (ingress) и правильно настроить правила распределения запросов. Почти весь функционал, предоставляемый интерфейсом узла авторизации (за исключением входа по логину и паролю) требует токен пользователя для аутентификации, указываемый в заголовке запроса. Все действия, за исключением добавления записи в таблицу требуют указать в запросе токен пользователя. Запросы на добавление записей в таблицу разрешены сервисам и разрешаются только при предоставлении валидного apikey. Перед началом использования системы, пользователь должен быть зарегистрирован администратором системы. Пользователи, зарегистрированные в системе, получают доступ просмотру созданных ими и их сервисами таблиц, созданию новых таблиц, созданию сервисов-источников данных, редактированию таблиц (удалению записей) и к поиску собранных в их таблицы метрик. Авторизация прав доступа происходит после валидации токена пользователя, который тот получает после корректного ввода логина и пароля. Созданным пользователем сервисам

присваивается уникальные ключ к API, с помощью которого сервисы получают доступ к заполнению таблиц пользователя. Диаграмма последовательности добавления пользователем сервиса показана на рисунке 5.25.

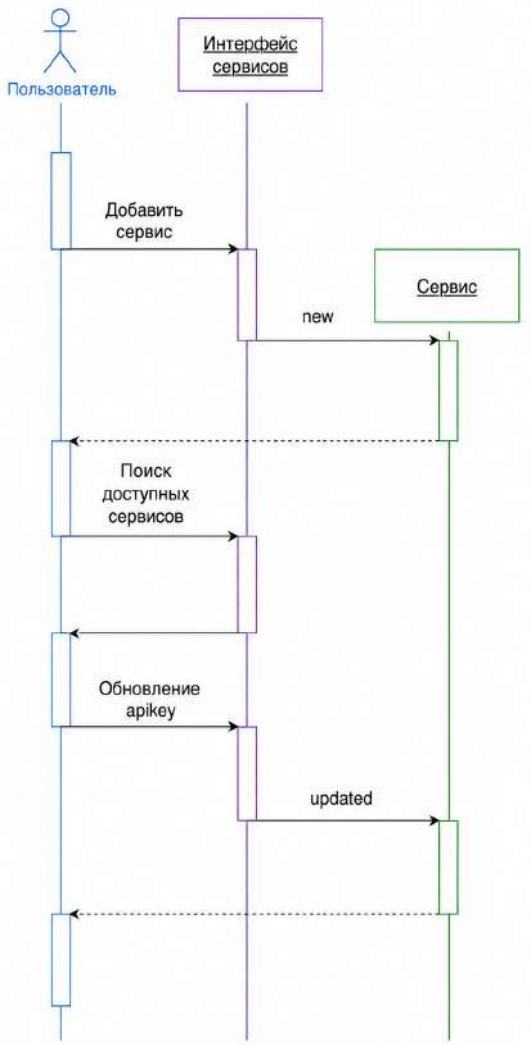

Рисунок 5.25 — Диаграмма последовательности добавления пользователем сервиса

Отправка сервисом метрик подразумевает асинхронность, связанную с отправкой метрики в очередь. После успешной передачи метрики в очередь, модуль таблиц отправляет сервису ответ со статусом 202 Accepted, давая понять, что на данном этапе метрика еще не сохранилась в соответствующую таблицу. Далее обработчик очереди, получив очередную метрику проводит валидацию ее структуры, проверяя соответствие структуре таблицы. После успешного прохождения валидации метрика сохраняется в том экземпляре таблицы, в который была отправлена. Добавление пользователем новой таблицы и отправка сервисом очередной метрики представлены соответственно на рисунке 5.26.

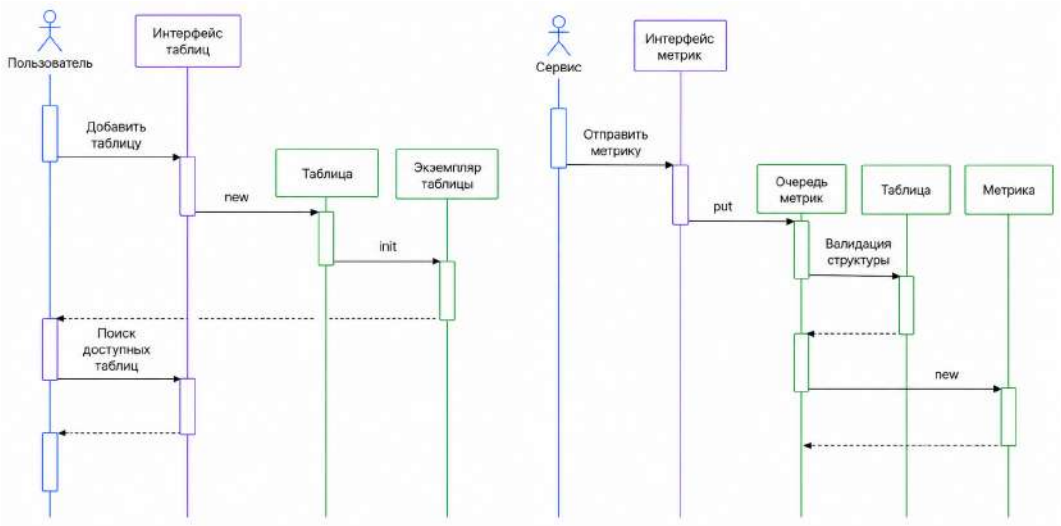

Рисунок 5.26 — Последовательность создания таблицы и отправки новой метрики

Тестирование основных сценариев

Два основные сценария тестирования системы мониторинга – обработка нагрузки на хранилище данных и обработка нагрузки на обработчиках. Оба эти сценария должны решаться за счет автоматического масштабирования системы и равномерным распределением вычислительной нагрузки между узлами.

Ввиду ограниченности вычислительных ресурсов тестового стенда, распределение системы тестировалось на относительно небольшом количестве вычислительных узлов, однако в данном вопросе интересует сам масштаб изменения нагрузки на каждый узел. Для тестирования первого сценария был использован разработанный сервис-источник, генерирующий синтетические данные о состоянии транспортного потока. Нагрузка создаваемая на хранилище, выполняющая 650 операций чтения из одной таблицы в секунду на одном узле, содержащем 4 шарда таблицы. Для кластера, состоящего из трех узлов, каждый из которых содержал по 4 четыре шарда таблицы, средняя нагрузка составляла 181 операцию в секунду. После увеличения коэффициента репликации узлов хранения нагрузка постепенно распределялась по всем узлам, в среднем составляя 129 операций чтения в секунду. В таблицах 45 и 46 приведены примеры распределения нагрузки в кластерах с тремя и четырьмя узлами. Стоит отметить, что масштабирование может быть эффективным способом снижения нагрузки на отдельный узел, если количество узлов меньше общего числа сегментов таблиц в кластере. Это иллюстрируется примером

горизонтального масштабирования кластера от одного узла с четырьмя сегментами до пяти узлов. На рисунке 5.27 показано распределение нагрузки в этом сценарии. Если количество узлов в кластере равно количеству сегментов таблицы, нагрузку на операции записи в эту таблицу можно уменьшить путем переконфигурации системы в новой версии.

Таблица 45 — Распределение нагрузок с 3 узлами.

Статус	**Пользователи Reers / Leaders**	**RAM**	**Число SST файлов**	**Размер несжатых файлов SST**	**Общий размер SST файлов**	**Чтение опер./сек.**	**Запись опер./сек.**
ALIVE	12 / 4	53.09 MB	0	0 B	0 B	190.08	28.29
ALIVE	12 / 4	42.96 MB	0	0 B	0 B	178.85	29.67
ALIVE	12 / 4	43.25 MB	0	0 B	0 B	176.62	34.45

Таблица 46 — Распределение нагрузок с 4 узлами.

Статус	**Пользователи Reers / Leaders**	**RAM**	**Число SST файлов**	**Размер несжатых файлов SST**	**Общий размер SST файлов**	**Чтение опер./сек.**	**Запись опер./сек.**
ALIVE	9 / 3	52.43 MB	3	246.60 KB	381.53 KB	34.20	22.63
ALIVE	9 / 3	38.80 MB	9	748.24 KB	1.18 MB	137.49	21.06
ALIVE	9 / 3	62.76 MB	2	168.77 KB	271.10 KB	123.72	20.42
ALIVE	9 / 3	61.41 MB	0	0 B	0 B	123.49	20.02

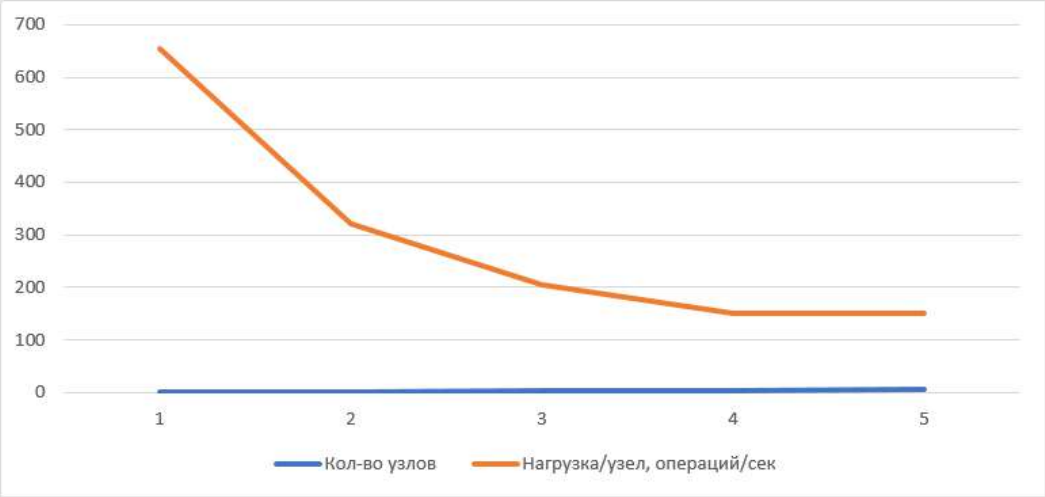

Рисунок 5.27 — Распределение нагрузки

В ходе второго этапа тестирования была измерена скорость полной обработки записи для экземпляра таблицы с момента поступления на сервер и

до сохранения в базе. Структура тестовых данных была сформирована таким образом, чтобы имитировать информацию от источников метрик дорожного движения.

Средняя скорость обработки составила 0,2 записи в секунду, что главным образом связано с ожиданием записи в очереди. На этапе извлечения из очереди одна запись обрабатывалась примерно за 100 миллисекунд. С увеличением числа реплик обработчиков скорость обработки на уровне отдельного узла осталась прежней, однако общая производительность потока возросла. Следует учитывать, что важную роль в эффективности обработчиков очереди играло количество партиций обрабатываемого задания. Рост производительности прекращался, когда число реплик обработчиков равнялось количеству партиций в очереди.

5.5 Выводы по главе

В данной главе предложена и обоснована комплексная методология построения инфраструктуры ИТС, ориентированная на обеспечение отказоустойчивости, масштабируемости и гибкости при обработке больших объёмов данных. Разработанная методология охватывает как архитектурные, так и методологические аспекты проектирования распределённых ИТС. Методология включает в себя три ключевых направления: децентрализованное взаимодействие подсистем, динамическое распределение нагрузки с учётом приоритетов задач и генерацию синтетических данных для тестирования. Разработанная методология включает в себя описанные ниже методы.

Разработана архитектура системы сбора, хранения и обработки больших данных для ИТС, которая структурирована по функциональным зонам. Архитектура включает в себя зоны сбора и потребления данных, аналитики, хранения, обработки и визуализации. Предлагаемая архитектура обеспечивает чёткое разделение ответственности между компонентами, упрощает интеграцию с внешними системами и поддерживает гибкое развёртывание в облачных средах. Особое внимание уделено вопросам протокольной совместимости, адаптированной к типу источника и сценарию взаимодействия.

Предложен метод кластеризации серверов ИТС на основе метрики сетевого расстояния (RTT) и алгоритма консенсуса Raft с автоматическим определением Лидера и переконфигурацией системы. В отличие от традиционных одноуровневых архитектур, где все узлы зависят от одного лидера. Разработанный метод использует многоуровневую децентрализованную топологию, в которой каждый географический регион образует независимый подкластер с собственным лидером. Это минимизирует влияние сетевой задержки, автоматически исключает не отвечающие узлы и динамически добавляет новые серверы без вмешательства администратора. Сервис `Agent` разработан для реализации логики состояний "Последователь–Кандидат–Лидер"и обеспечения самоконфигурирующейся топологии кластера.

Разработан специализированный метод динамического распределения приоритетных задач, учитывающий как вес задачи (например, обработка дорожно-транспортных происшествий имеет более высокий приоритет, чем статистический анализ дорожного движения), так и текущую нагрузку на вычислительные узлы. В отличие от универсальных оркестраторов (Kubernetes, Mesos), которые ориентированы на балансировку ресурсов и доступность сервисов, предлагаемый метод фокусируется на минимизации времени выполнения набора задач при заданном пороговом значении. Метод гарантирует сохранение работоспособности системы даже при отказе части узлов, перераспределяя наиболее критичные задачи на оставшиеся ресурсы.

Разработан метод генерации синтетических данных о транспортном потоке на основе статистического моделирования поведения транспортных средств. Метод учитывает реалистичные ограничения, такие как диапазоны скоростей, радиусы поворотов, вероятности маневров и динамика ускорения. Генератор реализован в виде многопоточного REST-клиента, способного имитировать одновременную работу сотен агентов. Предложенное решение устраняет необходимость в реальных данных, которые могут быть недоступны или конфиденциальны, и позволяет проводить стресс-тестирование в контролируемых условиях.

Заключение

В диссертационной работе получены следующие основные результаты, соответствующие поставленным задачам исследования.

1. Проведён анализ современного состояния интеллектуальных транспортных систем, рассматриваемых как сложные социотехнические распределённые системы с многослойной архитектурой и сквозными функциями управления. Установлено, что ключевыми ограничениями являются отсутствие единого формализованного аппарата синтеза архитектур эксплуатации моделей ИИ с учётом требований SLA, а также методологический разрыв между задачами прогнозирования, моделирования поведения и обеспечения эксплуатационной надёжности. Выявленные ограничения сформировали научную проблему и обосновали направление дальнейших исследований.
2. Разработан метод прогнозирования макроскопических характеристик транспортного потока PTSTG (PatchTST-Graph), основанный на декомпозиции временного и пространственного моделирования. Метод обеспечивает однопроходное формирование прогноза без авторегрессионного развёртывания, учитывает пространственные зависимости через обучаемую матрицу смежности и демонстрирует конкурентоспособную точность на общепринятых наборах данных при сниженной вычислительной сложности. Экспериментально подтверждена эффективность метода по метрикам MAE и RMSE на различных горизонтах прогнозирования.
3. Разработан метод прогнозирования микроскопических характеристик транспортного потока, основанный на гибридной архитектуре, сочетающей модели, основанные на данных, и классические физически интерпретируемые модели. Предложен гибридный метод на базе Bi-LSTM-LSTM-Informer и внешнего ограничителя на базе модели SSSR-Micro с динамической коррекцией параметров, обеспечивающая устойчивость при замкнутом применении и подавление аномальных предсказаний. Экспериментально показана устойчивость метода к накоплению ошибок и сдвигу распределения, а также его применимость в реальных условиях.

4. Разработана и формализована методология конфигурирования архитектуры исполнения моделей ИИ в интеллектуальных транспортных системах, основанная на представлении системы в виде ориентированного мультиграфа с параметризованными характеристиками рёбер и вершин. Методология обеспечивает синтез архитектурных конфигураций с учётом ограничений по задержке, пропускной способности и доступности, включает формальные модели оценки производительности и механизм верификации SLA в контуре непрерывной интеграции.
5. Предложена архитектура системы обработки больших данных для интеллектуальных транспортных систем, структурированная по функциональным зонам, включающим сбор, передачу, хранение, обработку и визуализацию данных. Архитектура обеспечивает разделение ответственности компонентов, поддержку интеграции с внешними системами и возможность масштабируемого развёртывания в распределённых вычислительных средах.
6. Разработана и обоснована методология создания инфраструктуры интеллектуальных транспортных систем, ориентированная на отказоустойчивость, масштабируемость и адаптивность. Предложены методы децентрализованной кластеризации с использованием алгоритма консенсуса, динамического распределения задач с учётом приоритетов и генерации синтетических данных для стресс-тестирования. Проведена экспериментальная верификация, подтвердившая работоспособность и эффективность предложенных решений в условиях нестационарной нагрузки.

Полученные результаты в совокупности формируют целостную методологическую и технологическую основу для построения интеллектуальных транспортных систем, обеспечивающих согласованное решение задач прогнозирования, моделирования поведения и эксплуатации моделей машинного обучения в условиях реального времени и нестационарной нагрузки.

Список сокращений и условных обозначений

АСУДД : Автоматизированная система управления дорожным движением

ГКРЧ : Государственная комиссия по радиочастотам

ГОСТ : Государственный стандарт

ИТС : Интеллектуальная транспортная система

КИИ : Критическая информационная инфраструктура

МПУ : Интеллектуальный контур мониторинг — прогноз — управление

ОДД : Организация дорожного движения

ПДД : Правила дорожного движения

ПДн : Персональные данные

РЭС : Радиоэлектронные средства

СФ : Светофорный объект / светофор

ТС : Транспортное средство

УДС : Улично-дорожная сеть

УЦ : Удостоверяющий центр

ЭРА-ГЛОНАСС : Система экстренного реагирования при авариях

ADAS : Advanced Driver Assistance Systems (Улучшенные системы помощи водителю)

API : Application Programming Interface (Программный интерфейс приложения)

APTS : Advanced Public Transportation System (Система общественного транспорта и приоритизации подвижного состава)

ARC-IT : Architecture Reference for Cooperative and Intelligent Transportation (Американская архитектура ИТС)

ATIS : Advanced Traveler Information System (Система информирования участников дорожного движения)

ATMS : Advanced Traffic Management System (Система управления дорожным движением)

AVL : Automatic Vehicle Location (Автоматическое определение местоположения транспортных средств)

BSM : Basic Safety Message (Базовое сообщение безопасности)

C-ITS : Cooperative Intelligent Transport Systems (Кооперативные интеллектуальные транспортные системы)

CAM : Cooperative Awareness Message (Кооперативное сообщение об осведомленности)

CBA : Cost-Benefit Analysis (Анализ затрат и выгод)

CRL : Certificate Revocation List (Список отозванных сертификатов)

DSRC : Dedicated Short-Range Communications (Выделенная связь ближнего действия)

DENM : Decentralized Environmental Notification Message (Децентрализованное уведомление о дорожной обстановке)

DWH : Data Warehouse (Хранилище данных)

ETC : Electronic Toll Collection (Электронное взимание платы за проезд)

ETSI : European Telecommunications Standards Institute (Европейский институт телекоммуникационных стандартов)

GNSS : Global Navigation Satellite System (Глобальная навигационная спутниковая система)

GTFS : General Transit Feed Specification (Общий формат данных о маршрутах общественного транспорта)

HMI : Human-Machine Interface (Человеко-машинный интерфейс)

HSM : Hardware Security Module (Аппаратный модуль безопасности)

IAM : Identity and Access Management (Система управления доступом)

ISO : International Organization for Standardization (Международная организация по стандартизации)

LCCA : Life Cycle Cost Analysis (Анализ жизненного цикла стоимости)

LTE : Long-Term Evolution (Стандарт беспроводной высокоскоростной передачи данных)

MaaS : Mobility as a Service (Мобильность как услуга)

MAP : Topology/Intersection Description Message (Сообщение с описанием геометрии перекрестка)

MLOps : Machine Learning Operations (Методология управления жизненным циклом моделей машинного обучения)

MPC : Model Predictive Control (Модели предиктивного управления)

NAP : National Access Point (Национальная точка доступа к данным)

NeTEx : Network and Timetable Exchange (Европейский стандарт обмена транспортными данными)

NTP : Network Time Protocol (Протокол сетевого времени)

OBU : On-Board Unit (Бортовое устройство транспортного средства)

OCSP : Online Certificate Status Protocol (Протокол проверки статуса сертификата в реальном времени)

PC5 : Side link interface for C-V2X (Интерфейс прямой связи между устройствами в сотовых сетях V2X)

PET : Post-Encroachment Time (Показатель времени до потенциального столкновения)

PKI : Public Key Infrastructure (Инфраструктура открытых ключей)

PTI : Planning Time Index (Показатель надежности маршрута)

PTP : Precision Time Protocol (Протокол точного времени)

RSU : Road Side Unit (Дорожная станция)

RTT : Round-Trip Time (Время приема-передачи сигнала)

RWIS : Road Weather Information System (Дорожная метеорология / метеостанция)

SAE : Society of Automotive Engineers (Сообщество автомобильных инженеров)

SIEM : Security Information and Event Management (Система управления событиями безопасности)

SIRI : Service Interface for Real Time Information (Интерфейс для передачи информации о движении транспорта в реальном времени)

SLA : Service Level Agreement (Соглашение об уровне обслуживания)

SPaT : Signal Phase and Timing (Сообщение о фазах и временных интервалах светофора)

TPM : Trusted Platform Module (Доверенный платформенный модуль)

TSP : Transit Signal Priority (Приоритизация общественного транспорта)

TTC : Time-to-Collision (Время до столкновения)

TTTR : Travel Time Reliability Index (Показатель надежности времени в пути)

V2X : Vehicle-to-Everything (Кооперативные транспортные коммуникации «автомобиль — ко всему»)

VMS : Variable Message Sign (Табло переменной информации)

gRPC : Google Remote Procedure Call (Высокопроизводительный протокол удаленного вызова процедур)

LSTM : Long Short-Term Memory (Долговременная кратковременная память)

GPU : Graphics Processing Unit (Графический процессор)

ПИМ : Пакетное исполнение моделей

ГИС : Геоинформационная система

Список литературы

1. *Mkrtchian, G.* Patch-Based Transformer–Graph Framework (PTSTG) for Traffic Forecasting in Transportation Systems [Текст] / G. Mkrtchian, M. Gorodnichev // Applied Sciences (Switzerland). — 2025. — Т. 15, № 19. — С. 10468. — (Scopus Q1).
2. On the Automated Selection of Neural Network Architectures in the Task of Pattern Recognition for Intelligent Transportation Systems [Текст] / A. Moseva [и др.] // 2025 Systems of Signals Generating and Processing in the Field of on Board Communications. — IEEE. Moscow, Russian Federation, 2025. — С. 1—5.
3. *Moseva, M.* Robust Neural Network-Based Object Detection Method for Drone Images [Текст] / M. Moseva, K. Kharrasov, M. Gorodnichev // International Journal of Artificial Intelligence. — 2024. — Т. 22, № 1. — С. 1—6. — (Scopus Q2).
4. *Gorodnichev, M.* On the Problem of Traffic Forecasting Using Graph Neural Networks and Dynamic Adjacency Matrix [Текст] / M. Gorodnichev // 2025 Intelligent Technologies and Electronic Devices in Vehicle and Road Transport Complex (TIRVED). — IEEE. 2025. — С. 1—4.
5. *Polyantseva, K.* Ensuring the Reliability of a Highly Loaded Vehicle Monitoring and Traffic Control Platform [Текст] / K. Polyantseva, M. Gorodnichev, M. Moseva // Systems of Signals Generating and Processing in the Field of on Board Communications. Т. 6. — IEEE. 2023. — С. 365—372.
6. Development of a system for collecting and processing big data received from terminal devices [Текст] / M. Gorodnichev [и др.] // 2021 Systems of Signal Synchronization, Generating and Processing in Telecommunications (SYNCHROINFO). — IEEE. 2021. — С. 9488337.
7. *Degtyareva, V.* Study of Machine Learning Techniques for Transport [Текст] / V. Degtyareva, M. Gorodnichev, M. Moseva // Smart Technologies for Society, State and Economy. — 2021. — С. 1585—1595.

8. The study of the applicability of clustering methods to identify patterns in traffic flows [Текст] / M. Gorodnichev [и др.] // 2020 Wave Electronics and its Application in Information and Telecommunication Systems (WECONF). — IEEE. 2020. — С. 9131435.

9. *Menshih, P.* Efficiency analysis of neural networks ensembles using synthetic data [Текст] / P. Menshih, S. Erokhin, M. Gorodnichev // 2020 Wave Electronics and its Application in Information and Telecommunication Systems (WECONF). — IEEE. 2020. — С. 9131495.

10. Программное обеспечение генерации синтетических данных о состоянии транспортного потока [Текст]. — заявл. 07.05.2025 ; опубл. 22.05.2025.

11. Мобильное приложение многопараметрического сбора основных характеристик транспортных потоков под управлением ОС iOS [Текст]. — заявл. 03.02.2020 ; опубл. 13.02.2020.

12. Программная реализация иерархической кластеризации поведенческого фактора водителей методами машинного обучения [Текст]. — заявл. 11.02.2020 ; опубл. 25.02.2020.

13. Программная платформа мониторинга и управления движением беспилотных транспортных средств [Текст]. — заявл. 03.02.2020 ; опубл. 25.02.2020.

14. Программа автоматизированного определения скорости и габаритов транспортных средств на основании изображений, получаемых со стереокамер [Текст]. — заявл. 15.11.2019 ; опубл. 28.11.2019.

15. *Харрасов, К. Р.* Исследование устойчивости нейронных сетей в задаче распознавания образов [Текст] / К. Р. Харрасов, М. С. Мосева, М. Г. Городничев // Информационные технологии. — 2025. — Т. 31, № 2. — С. 87—92. — (Перечень рецензируемых научных изданий №1459 от 09.12.2024).

16. Development of Software Module for Recognizing Traffic Flows Through Deep Learning [Текст] / M. Gorodnichev [и др.] // Journal of Industrial Integration and Management: Innovation and Entrepreneurship. — 2023. — Т. 08, № 02. — С. 175—199. — (Scopus Q1).

17. On the Problem of Restoring and Classifying a 3D Object in Creating a Simulator of a Realistic Urban Environment [Текст] / M. Gorodnichev [и др.] // Sensors. — 2022. — Т. 22, № 14. — С. 5199. — (Scopus Q1).

18. *Gorodnichev, M.* A Road Accident Detection Method Utilizing Deep Learning and Fast Fourier Transform [Текст] / M. Gorodnichev, K. Kharrasov, M. Moseva // Engineering, Technology and Applied Science Research. — 2025. — Т. 15, № 6. — С. 30191—30195. — (Scopus Q2).

19. *Gorodnichev, M.* On the Problem of Interpretability of Neural Algorithms for Predicting Microscopic Traffic Flow Data [Текст] / M. Gorodnichev // 2026 Systems of Signals Generating and Processing in the Field of on Board Communications. — IEEE. 2026. — С. 1—6.

20. *Gorodnichev, M.* On the Applicability of Reinforcement Learning in the Traffic Flow Control Problem [Текст] / M. Gorodnichev // 2025 Systems of Signals Generating and Processing in the Field of on Board Communications. — IEEE. Moscow, Russian Federation, 2025. — С. 1—6.

21. *Gorodnichev, M.* On the problem of developing a fault-tolerant high-loaded cluster of support for an intelligent transportation system [Текст] / M. Gorodnichev, M. Moseva // Wave Electronics and Its Application in Information and Telecommunication Systems. — 2023. — Т. 6, № 1. — С. 122—127.

22. *Gorodnichev, M.* Architecture development for a high-load vehicle tracking application [Текст] / M. Gorodnichev, M. Moseva, D. Belousov // 2021 Wave Electronics and its Application in Information and Telecommunication Systems (WECONF). — IEEE. 2021. — С. 9470576.

23. On methods for recognizing unstructured road maintenance objects using deep learning algorithms [Текст] / M. Gorodnichev [и др.] // 2020 International Conference on Engineering Management of Communication and Technology (EMCTECH). — IEEE. 2020. — С. 9261568.

24. On the task of recognizing road infrastructure on devices with low computing power [Текст] / M. Moseva [и др.] // 2020 International Conference on Engineering Management of Communication and Technology (EMCTECH). — IEEE. 2020. — С. 9261559.

25. Neural network methods in real-time recognition of road infrastructure objects [Текст] / M. Gorodnichev [и др.] // IOP Conference Series: Materials Science and Engineering. — 2020. — Т. 918. — С. 012077.

26. *Gorodnichev, M.* Development of Platform for Confirming and Storing Supply Data Using Blockchain Technology [Текст] / M. Gorodnichev, A. Nazarova, M. Moseva // 2019 IEEE International Conference on Quality Management, Transport and Information Security (IT&QM&IS). — IEEE. 2019. — С. 182—185.

27. On the Applicability of Neural Networks in the Tasks of Detecting Dangerous Movement [Текст] / K. Polyantseva [и др.] // 2019 Wave Electronics and its Application in Information and Telecommunication Systems (WECONF). — IEEE. 2019. — С. 8840654.

28. The Research of Blockchain Technology for Data Protection in IoT Devices [Текст] / M. Gorodnichev [и др.] // International Conference Technology & Entrepreneurship in Digital Society (TEDS). — 2019. — С. 94—97.

29. Программный модуль динамического распределения ресурсов в подсистемах интеллектуальных транспортных систем мониторинга и управления [Текст]. — заявл. 07.05.2025 ; опубл. 27.05.2025.

30. Автоматизированная информационная система поддержки функционирования интеллектуальной транспортной системы с учетом формата обрабатываемой и хранимой информации [Текст]. — заявл. 19.05.2025 ; опубл. 02.06.2025.

31. *Полянцева, К. А.* Архитектура распределенной системы хранения и обработки больших данных на основе Apache Ozone и Argo Workflows [Текст] / К. А. Полянцева, А. В. Комлев, М. Г. Городничев // Известия Кабардино-Балкарского научного центра РАН. — 2026. — Т. 28, № 2. — С. 34—50. — (Перечень рецензируемых научных изданий №1328 от 17.02.2026).

32. *Городничев, М. Г.* Механизм управления алгоритмами построения витрин данных в интеллектуальных транспортных системах [Текст] / М. Г. Городничев, К. А. Полянцева, Н. А. Грязнов // Computational Nanotechnology. — 2026. — Т. 13, № 1. — С. 175—185. — (Перечень рецензируемых научных изданий №27 от 17.02.2026).

33. *Иевлев, К. О.* Сравнительный анализ открытых табличных форматов: Apache Iceberg, Apache Hudi, Delta Lake, Apache Paimon [Текст] / К. О. Иевлев, М. Г. Городничев // Вестник Российского нового университета. Серия: Сложные системы: модели, анализ, управление. — 2026. — № 1. — С. 170—188. — (Перечень рецензируемых научных изданий №167 от 25.11.2025).

34. *Иевлев, К. О.* Сравнительный анализ систем хранения данных HDFS и Apache Ozone [Текст] / К. О. Иевлев, М. Г. Городничев // Computational Nanotechnology. — 2025. — Т. 12, № 1. — С. 26—33. — (Перечень рецензируемых научных изданий №27 от 09.12.2024).

35. *Городничев, М. Г.* О задаче построение независимых архитектур обработки данных в интеллектуальных транспортных системах [Текст] / М. Г. Городничев, Д. В. Титов, А. Д. Липатова // Инженерный вестник Дона. — 2025. — 11(131). — С. 62—92. — (Перечень рецензируемых научных изданий №1398 от 01.07.2025).

36. On the Task of Developing Algorithms for Generating Synthetic Data for Testing Intelligent Transportation Systems [Текст] / A. Lipatova [и др.] // Systems of Signal Synchronization, Generating and Processing in Telecommunications. Т. 7. — IEEE. 2024. — С. 267—271.

37. On the problem of developing a realistic road infrastructure simulator for reinforced learning [Текст] / K. Polyantseva [и др.] // Wave Electronics and Its Application in Information and Telecommunication Systems. — 2023. — Т. 6, № 1. — С. 340—347.

38. Providing Fault Tolerance of Cluster Computing Systems Based on Fault-Tolerant Dynamic Computation Planning [Текст] / E. Tourouta [и др.] // Lecture Notes in Information Systems and Organisation. — 2022. — С. 143—150.

39. *Tourouta, E.* Ensuring Fault Tolerance of a Computing System with a Client-Server Architecture Based on Fault-Tolerant Computing Planning [Текст] / E. Tourouta, M. Gorodnichev, K. Polyantseva // Wave Electronics and Its Application in Information and Telecommunication Systems. — 2022. — Т. 5, № 1. — С. 496—500.

40. Intelligent System for Recognizing the Main Characteristics of the Traffic Flow [Текст] / M. Gorodnichev [и др.] // 2021 Intelligent Technologies and Electronic Devices in Vehicle and Road Transport Complex (TIRVED). — IEEE. 2021.

41. Development of a module for controlling and predicting the movement of unmanned vehicles [Текст] / M. Gromov [и др.] // 2021 Systems of Signal Synchronization, Generating and Processing in Telecommunications (SYNCHROINFO). — IEEE. 2021. — С. 9488374.

42. Development of a Platform for Road Infrastructure Digital Certification [Текст] / M. Moseva [и др.] // 2021 Intelligent Technologies and Electronic Devices in Vehicle and Road Transport Complex (TIRVED). — IEEE. 2021.

43. *Gorodnichev, M.* Informational Aspects of Automated Monitoring of Dangerous Driving [Текст] / M. Gorodnichev, K. Polyantseva, M. Yashina // 2019 Systems of Signals Generating and Processing in the Field of on Board Communications (SOSG). — IEEE. 2019. — С. 8706721.

44. Features of the development of information systems for working with blockchain technology [Текст] / M. Gorodnichev [и др.] // Journal of Physics: Conference Series. — 2019. — Т. 1399. — С. 033039.

45. Интеллектуальная платформа адаптивного управления светофорами [Текст]. — заявл. 15.01.2026 ; опубл. 22.01.2026.

46. Программная реализация метода поиска нейросетевых алгоритмов управления технологическим процессом на транспорте [Текст]. — заявл. 07.05.2025 ; опубл. 27.05.2025.

47. Программная реализация архитектуры, обеспечивающей информационный процесс передачи информации между одиночными транспортными средствами (V2V) [Текст]. — заявл. 07.05.2025 ; опубл. 09.06.2025.

48. RouteGenius — интеллектуальная платформа для планирования маршрутов автотуристов [Текст]. — заявл. 04.06.2025 ; опубл. 24.06.2025.

49. *Городничев, М. Г.* О некоторых подходах повышения отказоустойчивости вычислительного кластера интеллектуальной транспортной системы с учетом приоритетов выполнения задач [Текст] / М. Г. Городничев // Модели, системы, сети в экономике, технике, природе и обществе. —

2026. — 1(XX). — С. XX—XX. — (Перечень рецензируемых научных изданий №1791 от 25.11.2025).

50. *Городничев, М. Г.* О задаче применимости синтетических данных при тестировании интеллектуальных транспортных систем [Текст] / М. Г. Городничев // Computational Nanotechnology. — 2025. — Т. 12, № 1. — С. 105—115. — (Перечень рецензируемых научных изданий №27 от 09.12.2024).

51. О задаче построения децентрализованной интеллектуальной транспортной системы на основе протокола Raft и кластеризации по сетевому расстоянию [Текст] / М. Г. Городничев [и др.] // Вестник Рязанского государственного радиотехнического университета. — 2025. — № 94. — С. 59—67. — (Перечень рецензируемых научных изданий №735 от 25.11.2025).

52. *Gorodnichev, M.* On the Task of Designing Efficient Machine Learning Operations Architectures [Текст] / M. Gorodnichev // 2025 Systems of Signal Synchronization, Generating and Processing in Telecommunications (SYNCHROINFO). — IEEE. Tyumen, Russian Federation, 2025. — С. 1—5.

53. Research on Driver Monitoring Methods [Текст] / M. Gorodnichev [и др.] // Systems of Signals Generating and Processing in the Field of on Board Communications. Т. 7. — IEEE. 2024. — С. 123—127.

54. *Gorodnichev, M.* On the applicability of reinforced learning in the route selection task of an unmanned vehicle [Текст] / M. Gorodnichev // Systems of Signal Synchronization, Generating and Processing in Telecommunications. Т. 6. — IEEE. 2023. — С. 107—115.

55. On Automated Safety Distance Monitoring Methods by Stereo Cameras [Текст] / M. Gorodnichev [и др.] // 2020 Systems of Signals Generating and Processing in the Field of on Board Communications. — IEEE. 2020. — С. 9078616.

56. Программная реализация метода моделирования транспортных потоков с учетом физических характеристик одиночных транспортных средств [Текст]. — заявл. 07.05.2025 ; опубл. 07.05.2025.

57. Автоматизированная информационная система распознавания и восстановления объектов дорожной инфраструктуры [Текст]. — заявл. 07.05.2025 ; опубл. 23.05.2025.

58. Программный модуль определения основных характеристик транспортного потока (интенсивность, плотность, качественный состав) на основе видеопотока [Текст]. — заявл. 21.07.2021 ; опубл. 30.07.2021.

59. Мобильное приложение многопараметрического сбора основных характеристик транспортных потоков под управлением ОС Android [Текст]. — заявл. 03.02.2020 ; опубл. 25.02.2020.

60. *Городничев, М. Г.* О применении обучения с подкреплением в задаче выбора оптимальной траектории движения [Текст] / М. Г. Городничев // Известия Кабардино-Балкарского научного центра РАН. — 2025. — Т. 27, № 2. — С. 86—102. — (Перечень рецензируемых научных изданий №1319 от 05.02.2025).

61. Benchmarking Compact VLMs for Clip-Level Surveillance Anomaly Detection Under Weak Supervision [Текст] / K. Borodin [и др.] // J. Imaging. — 2025. — Т. 11. — С. 400. — (Scopus Q1).

62. End-to-End Train Horn Detection for Railway Transit Safety [Текст] / V.-T. Tran [и др.] // Sensors. — 2022. — Т. 22. — С. 4453. — (Scopus Q1).

63. *Gorodnichev, M.* Simulation model of traffic flow based on agent-based modeling [Текст] / M. Gorodnichev // Heritage and Sustainable Development. — 2022. — Т. 4, № 2. — С. 195—190. — (Scopus Q1).

64. *Jarašūnienė, A.* RESEARCH INTO INTELLIGENT TRANSPORT SYSTEMS (ITS) TECHNOLOGIES AND EFFICIENCY [Текст] / A. Jarašūnienė // Transport. — 2007. — Т. 22, № 2. — С. 61—67.

65. *Qi, L.* Research on Intelligent Transportation System Technologies and Applications [Текст] / L. Qi // Proceedings of the Workshop on Power Electronics and Intelligent Transportation System (PEITS). — 2008. — С. 529—531.

66. *An, S.-h.* A Survey of Intelligent Transportation Systems [Текст] / S.-h. An, B.-H. Lee, D.-R. Shin // Proceedings of the Third International Conference on Computational Intelligence, Communication Systems and Networks (CICSyN). — 2011. — С. 332—337.

67. *Qureshi, K. N.* A Survey on Intelligent Transportation Systems [Текст] / K. N. Qureshi, A. H. Abdullah // Middle-East Journal of Scientific Research. — 2013. — Т. 15, № 5. — С. 629—642.

68. Data-Driven Intelligent Transportation Systems: A Survey [Текст] / J. Zhang [и др.] // IEEE Transactions on Intelligent Transportation Systems. — 2011. — Т. 12, № 4. — С. 1624—1639.

69. *Комаров, В. В.* Архитектура и стандартизация телематических и интеллектуальных транспортных систем. Зарубежный опыт и отечественная практика [Текст] / В. В. Комаров, С. А. Гараган. — НТБ «Энергия», 2012.

70. *Евстигнеев, И. А.* Интеллектуальные транспортные системы на автомобильных дорогах федерального значения России [Текст] / И. А. Евстигнеев. — Издательство «Перо», 2015.

71. *Жанказиев, С. В.* Интеллектуальные транспортные системы: учебное пособие [Текст] / С. В. Жанказиев. — МАДИ, 2016.

72. *Душкин, Р. В.* Интеллектуальные транспортные системы [Текст] / Р. В. Душкин. — ДМК Пресс, 2020.

73. *Singh, B.* Recent trends in intelligent transportation systems: a review [Текст] / B. Singh, A. Gupta // Journal of Transport Literature. — 2015. — Апр. — Т. 9. — С. 30—34.

74. *Dudhwala, R.* Intelligent Transportation System (ITS) - A Review [Текст] / R. Dudhwala, D. J. R. Pitroda // International Journal of Constructive Research in Civil Engineering. — 2020. — Т. 6, № 3. — С. 29—34.

75. *Yadav, S.* A Systematic and Critical Analysis of the Developments in the Field of Intelligent Transportation System [Текст] / S. Yadav, R. Rishi // Advances in Dynamical Systems and Applications. — 2021. — Т. 16, № 2. — С. 901—912.

76. *Garg, T.* A Systematic Review on Intelligent Transport Systems [Текст] / T. Garg, G. Kaur // Journal of Computational and Cognitive Engineering. — 2023. — Т. 2, № 3. — С. 175—188.

77. *Mishra, A.* A Comprehensive Study on Intelligent Transportation Systems [Текст] / A. Mishra, A. Priya, B. Scholar // SMART MOVES JOURNAL IJOSCIENCE. — 2018. — Т. 4. — С. 10.

78. Special Issue on Future Intelligent Transportation System (ITS) for Tomorrow and Beyond [Текст] / S. H. Nengroo [и др.] // Applied Sciences. — 2022. — Т. 12, № 11. — С. 5468.

79. Guest Editorial Introduction to the Special Issue on Applications and Developments of Novel Technologies in Intelligent Transportation Infrastructure Systems [Текст] / Y. Hou [и др.] // IEEE Transactions on Intelligent Transportation Systems. — 2022. — Т. 23, № 11. — С. 22015—22019.

80. *Raman, S. P.* Intelligent Transport System in India: State of the art [Текст] / S. P. Raman, J. Rodda // International Journal of Advance Research, Ideas and Innovations in Technology. — 2018. — Т. 4, № 5. — С. 26—30.

81. Development of Intelligent transportation system from the perspective of local government units, national government agencies and transportation cooperatives [Текст] / V. Sunio [и др.] // Innovatus: A Journal on Computing Technology Innovations. — 2022. — Т. 5, № 2.

82. Justification of the synergy of innovative intelligent transport systems in local conditions [Текст] / S. Mirzaliev [и др.] // 6th International Conference on Future Networks Distributed Systems (ICFNDS). — 2022. — С. 182—193.

83. *Krobot, Z.* Intelligent transport systems in the Czech Republic [Текст] / Z. Krobot, P. Stodola, J. Stodola // 2021 International Conference on Military Technologies (ICMT). — 2021. — С. 1—5.

84. *Stodola, J.* Intelligent Transport Systems [Текст] / J. Stodola, P. Stodola, J. Furch // Challenges to national defence in contemporary geopolitical situation. — 2022. — Т. 2022. — С. 41—49.

85. *Zhao, H.* Research on the Application of Intelligent Transportation System in Transportation Management [Текст] / H. Zhao, S. Sun // Academic Journal of Science and Technology. — 2023. — Т. 4. — С. 1—4.

86. *Odtalla, M.* Smart Transportation System [Текст] / M. Odtalla // Multi-Knowledge Electronic Comprehensive Journal For Education And Science Publications (MECSJ). — 2018. — № 8.

87. Ontology-Based Architecture for Intelligent Transportation Systems Using a Traffic Sensor Network [Текст] / S. Fernandez [и др.] // Sensors. — 2016. — Т. 16, № 8. — С. 1287.

88. *Zornić, D.* Infrastructure for Intelligent Transport Systems [Текст] / D. Zornić, D. Radovanović, N. Denic //. — 2016. — С. 192—196.

89. Information Management and Applications of Intelligent Transportation System [Текст] / C.-C. Lo [и др.] // Mathematical Problems in Engineering. — 2015. — Т. 2015. — С. 1—2.

90. Model Study for Intelligent Transportation System with Big Data [Текст] / J. Zeyu [и др.] // Procedia Computer Science. — 2017. — Т. 107. — С. 418—426.

91. *Su, H.* Research about the construction of intelligent transportation system under the big data of vehicle networking [Текст] / H. Su // IOP Conference Series: Materials Science and Engineering. Т. 688. — 2019. — С. 022011.

92. *Neelam, S.* Smart Vehicular Traffic Management: An Edge Cloud centric IoT based Framework [Текст] / S. Neelam, S. Sood // Internet of Things. — 2021. — Т. 14. — С. 100140.

93. *Vaidya, R.* Intelligent transportation system using IOT: A Review [Текст] / R. Vaidya, S. Kulkarni, V. Didore. — 2022.

94. *Anisi, M. H.* Efficient Data Reporting in Intelligent Transportation Systems [Текст] / M. H. Anisi, A. H. Abdullah // Networks and Spatial Economics. — 2015. — Т. 16, № 2. — С. 623—642.

95. Big Data Analytics in Intelligent Transportation Systems: A Survey [Текст] / L. Zhu [и др.] // IEEE Transactions on Intelligent Transportation Systems. — 2019. — Т. 20, № 1. — С. 383—398.

96. *Dabiri, S.* Transport-domain applications of widely used data sources in the smart transportation: A survey [Текст] / S. Dabiri, K. Heaslip // arXiv preprint. — 2018.

97. *Сайфиева, М. К.* Использование больших данных в интеллектуальных транспортных системах [Advantages of using big data in the intelligent transport systems] [Текст] / М. К. Сайфиева, И. А. Зарайченко // Сборник статей III международной конференции "Тенденции развития логистики и управления цепями поставок". — 2022. — С. 122—128.

98. Big data analytics: Computational intelligence techniques and application areas [Текст] / R. Iqbal [и др.] // Technological Forecasting and Social Change. — 2020. — Т. 153. — С. 119253.

99. *Vlahogianni, E. I.* Computational Intelligence and Optimization for Transportation Big Data: Challenges and Opportunities [Текст] / E. I. Vlahogianni // Engineering and Applied Sciences Optimization. Т. 38. — 2015. — С. 107—128.

100. The role of big data in smart city [Текст] / I. A. T. Hashem [и др.] // International Journal of Information Management. — 2016. — Т. 36, № 5. — С. 748—758.

101. The role of big data analytics in Internet of Things [Текст] / E. Ahmed [и др.] // Computer Networks. — 2017. — Т. 129, № 2. — С. 459—471.

102. *Ge, M.* Big data for internet of things: A survey [Текст] / M. Ge, H. Bangui, B. Buhnova // Future Generation Computer Systems. — 2018. — Т. 87. — С. 601—614.

103. *Shadroo, S.* Systematic Survey of Big Data and Data Mining in Internet of Things [Текст] / S. Shadroo, A. M. Rahmani // Computer Networks. — 2018. — Т. 139. — С. 19—47.

104. Exploiting IoT and Big Data Analytics: Defining Smart Digital City using Real-Time Urban Data [Текст] / M. M. Rathore [и др.] // Sustainable Cities and Society. — 2018. — Т. 40. — С. 600—610.

105. *Gohar, M.* SMART TSS: Defining Transportation System Behavior using Big Data Analytics in Smart Cities [Текст] / M. Gohar, M. Muzammal, A. U. Rahman // Sustainable Cities and Society. — 2018. — Т. 41. — С. 114—119.

106. *Hopkins, J.* Big data analytics and IoT in logistics: a case study [Текст] / J. Hopkins, P. Hawking // The International Journal of Logistics Management. — 2018. — Т. 29, № 2. — С. 575—591.

107. A Data Platform for the Highway Traffic Data [Текст] / R. Mian [и др.] // IEEE 8th International Symposium on the Maintenance and Evolution of Service-Oriented and Cloud-Based Systems (MESOCA). — 2014. — С. 37—42.

108. Towards a Multi-cluster Analytical Engine for Transportation Data [Текст] / M. Shtern [и др.] // IEEE International Conference on Cloud and Autonomic Computing (ICCAC). — 2014. — С. 249—257.

109. H-TDMS: A System for Traffic Big Data Management [Текст] / X. Hua [и др.] // Advanced Computer Architecture. Т. 626. — 2016. — С. 85—96.

110. Big Traffic Data Processing Framework for Intelligent Monitoring and Recording Systems [Текст] / Y. Xia [и др.] // Neurocomputing. — 2016. — Т. 181, № –. — С. 53—61.

111. *Babar, M.* Real-time data processing scheme using big data analytics in internet of things based smart transportation environment [Текст] / M. Babar, F. Arif // Journal of Ambient Intelligence and Humanized Computing. — 2018. — Т. 10. — С. 4167—4177.

112. *Shi, Q.* Big Data applications in real-time traffic operation and safety monitoring and improvement on urban expressways [Текст] / Q. Shi, M. Abdel-Aty // Transportation Research Part C: Emerging Technologies. — 2015. — Т. 58. — С. 380—394.

113. *Amin, M. A.* Big Data Role in Improving Intelligent Transportation Systems Safety: A Survey [Текст] / M. A. Amin, S. Hadouej, T. S. J. Darwish. — 2019.

114. *El Faouzi, N.-E.* Data Fusion for ITS: Techniques and Research Needs [Текст] / N.-E. El Faouzi, L. A. Klein //. Т. 15. — 2016. — С. 495—512.

115. *Haydari, A.* Deep Reinforcement Learning for Intelligent Transportation Systems: A Survey [Текст] / A. Haydari, Y. Yilmaz // IEEE Transactions on Intelligent Transportation Systems. — 2022. — Т. 23, № 1. — С. 11—32.

116. *Salitula, P.* Real Time Traffic Control System Using Wavelet Neural Networks Signal for Wireless Communication Technology of Future High Speed Railways Performance in Thailand with Intelligent Transportation Systems [Текст] / P. Salitula, C. Jaipradidtham // Proceedings of the 17th International Conference on Control, Automation and Systems (ICCAS). — 2017. — С. 391—396.

117. *Bäumler, M.* Categorizing Data-Driven Methods for Test Scenario Generation to Assess Automated Driving Systems [Текст] / M. Bäumler, F. Linke, G. Prokop // IEEE Access. — 2024. — Т. 12. — С. 52030—52050.

118. *Chen, W.* A Survey of Traffic Data Visualization [Текст] / W. Chen, F. Guo, F.-Y. Wang // IEEE Transactions on Intelligent Transportation Systems. — 2015. — Т. 16, № 6. — С. 2970—2984.

119. *Wang, X.* Research and Application of Traffic Visualization Based on Vehicle GPS Big Data [Текст] / X. Wang, S. Zhao, L. Dong // Proceedings of the Second International Conference on Intelligent Transportation. Т. 53. — 2017. — С. 293—302.

120. *Dilek, E.* Computer Vision Applications in Intelligent Transportation Systems: A Survey [Текст] / E. Dilek, M. Dener // Sensors. — 2023. — Т. 23, № 6. — С. 2938.

121. *Galea, S.* A Survey of Intelligent transportation systems based Modern Object Detectors Under Night-time Conditions [Текст] / S. Galea, D. Seychell, M. Bugeja // 3rd International Conference on Intelligent Sustainable Systems (ICISS). — 2020. — С. 265—270.

122. Survey of smartphone-based sensing in vehicles for intelligent transportation system applications [Текст] / J. Engelbrecht [и др.] // IET Intelligent Transport Systems. — 2015. — Т. 9, № 10. — С. 924—935.

123. Connected Vehicular Transportation: Data Analytics and Traffic-Dependent Networking [Текст] / C. Chen [и др.] // IEEE Vehicular Technology Magazine. — 2017. — Т. 12, № 3. — С. 42—54.

124. *Javed, M. A.* Data analytics for Cooperative Intelligent Transport Systems [Текст] / M. A. Javed, S. Zeadally, E. Ben Hamida // Vehicular Communications. — 2019. — Т. 15. — С. 63—72.

125. A Survey on Federated Learning in Intelligent Transportation Systems [Текст] / R. Zhang [и др.] // IEEE Transactions on Intelligent Vehicles. — 2025. — Т. 10, № 5. — С. 3043—3059.

126. *Hochreiter, S.* Long Short-Term Memory [Текст] / S. Hochreiter, J. Schmidhuber // Neural Computation. — 1997. — Т. 9, № 8. — С. 1735—1780.

127. Learning Phrase Representations Using RNN Encoder–Decoder for Statistical Machine Translation [Текст] / K. Cho [и др.] // Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP). — 2014. — С. 1724—1734.

128. *Yu, B.* Spatio-Temporal Graph Convolutional Networks: A Deep Learning Framework for Traffic Forecasting [Текст] / B. Yu, H. Yin, Z. Zhu // Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence (IJCAI-2018). — 2018. — С. 3634—3640.

129. Diffusion Convolutional Recurrent Neural Network: Data-Driven Traffic Forecasting [Текст] / Y. Li [и др.] // arXiv preprint. — 2017. — arXiv: 1707.01926.

130. Attention Is All You Need [Текст] / A. Vaswani [и др.] // arXiv preprint. — 2017. — arXiv: 1706.03762.

131. Informer: Beyond Efficient Transformer for Long Sequence Time-Series Forecasting [Текст] / H. Zhou [и др.] // Proceedings of the AAAI Conference on Artificial Intelligence. Т. 35. — 2021. — С. 11106—11115.

132. Autoformer: Decomposition Transformers with Auto-Correlation for Long-Term Series Forecasting [Текст] / H. Wu [и др.] // Advances in Neural Information Processing Systems 34 (NeurIPS 2021). Т. 34. — 2021. — С. 22419—22430.

133. A Time Series is Worth 64 Words: Long-Term Forecasting with Transformers [Текст] / Y. Nie [и др.] // arXiv preprint. — 2022. — arXiv: 2211.14730.

134. FEDformer: Frequency Enhanced Decomposed Transformer for Long-Term Series Forecasting [Текст] / T. Zhou [и др.] // Proceedings of the International Conference on Machine Learning (ICML). — 2022. — С. 27268—27286.

135. *Zhang, Y.* Crossformer: Transformer Utilizing Cross-Dimension Dependency for Multivariate Time Series Forecasting [Текст] / Y. Zhang, J. Yan // The Eleventh International Conference on Learning Representations. — 2023.

136. Non-stationary Transformers: Exploring the Stationarity in Time Series Forecasting [Текст] / D. Liu [и др.] // Proceedings of the International Conference on Learning Representations (ICLR). — 2022.

137. ETSformer: Exponential Smoothing Transformers for Time-Series Forecasting [Текст] / G. Woo [и др.] // arXiv preprint. — 2022. — arXiv: 2202.01381.

138. Are Transformers Effective for Time Series Forecasting? [Текст] / A. Zeng [и др.] // Proceedings of the AAAI Conference on Artificial Intelligence. Т. 37. — 2023. — С. 11121—11128.

139. TripChain2RecDeepSurv: A Novel Framework to Predict Transit Users' Lifecycle Behavior Status Transitions for User Management [Текст] / C. Yu [и др.] // Transportation Research Part C: Emerging Technologies. — 2024. — Т. 167. — С. 104818.

140. Spatial-Temporal Synchronous Graph Convolutional Networks: A New Framework for Spatial-Temporal Network Data Forecasting [Текст] / C. Song [и др.] // Proceedings of the AAAI Conference on Artificial Intelligence. Т. 34. — 2020. — С. 914—921.

141. Graph WaveNet for Deep Spatial-Temporal Graph Modeling [Текст] / Z. Wu [и др.] // Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI). — 2019. — С. 1907—1913.

142. Connecting the Dots: Multivariate Time Series Forecasting with Graph Neural Networks [Текст] / Z. Wu [и др.] // Proceedings of the 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining. — 2020. — С. 753—763.

143. Adaptive Graph Convolutional Recurrent Network for Traffic Forecasting [Текст] / L. Bai [и др.] // Neural Information Processing Systems (NeurIPS 2020). — 2020.

144. GMAN: A Graph Multi-Attention Network for Traffic Prediction [Текст] / C. Zheng [и др.] // Proceedings of the AAAI Conference on Artificial Intelligence. Т. 34. — 2020. — С. 1234—1241.

145. Spectral Temporal Graph Neural Network for Multivariate Time-Series Forecasting [Текст] / D. Cao [и др.] // Advances in Neural Information Processing Systems 33 (NeurIPS 2020). — 2020. — С. 17766—17778.

146. Long-Range Transformers for Dynamic Spatiotemporal Forecasting [Текст] / J. Grigsby [и др.] // arXiv preprint. — 2021. — arXiv: 2109.12218.

147. Spatial-Temporal Transformer Networks for Traffic Flow Forecasting [Текст] / K. Xu [и др.] // arXiv preprint. — 2020. — arXiv: 2001.02908.

148. *Feng, A.* Adaptive Graph Spatial-Temporal Transformer Network for Traffic Flow Forecasting [Текст] / A. Feng, L. Tassiulas // Proceedings of the 31st ACM International Conference on Information and Knowledge Management (CIKM). — 2022. — С. 3933—3937.

149. Attention based spatiotemporal graph attention networks for traffic flow forecasting [Текст] / Y. Wang [и др.] // Information Sciences. — 2022. — Т. 607. — С. 869—883.

150. STAEformer: Spatio-Temporal Adaptive Embedding Makes Vanilla Transformer SOTA for Traffic Forecasting [Текст] / H. Liu [и др.] // Proceedings of the 32nd ACM International Conference on Information and Knowledge Management. — 2023.

151. Navigating Spatio-Temporal Heterogeneity: A Graph Transformer Approach for Traffic Forecasting [Текст] / J. Zhou [и др.] // arXiv preprint. — 2024. — arXiv: 2408.10822.

152. Dynamic Graph Convolutional Recurrent Network for Traffic Prediction: Benchmark and Solution [Текст] / F. i [и др.] // ACM Transactions on Knowledge Discovery from Data. — 2023. — Т. 17, № 1. — С. 1—21.

153. *Shang, C.* Discrete Graph Structure Learning for Forecasting Multiple Time Series [Текст] / C. Shang, J. Chen, J. Bi // arXiv preprint. — 2021. — arXiv: 2101.06861.

154. *Li, M.* Spatial-Temporal Fusion Graph Neural Networks for Traffic Flow Forecasting [Текст] / M. Li, Z. Zhu // Proceedings of the AAAI Conference on Artificial Intelligence. Т. 35. — 2021. — С. 4189—4196.

155. Decoupled Dynamic Spatial-Temporal Graph Neural Network for Traffic Forecasting [Текст] / Z. Shao [и др.] // Proceedings of the VLDB Endowment. — 2022. — Т. 15, № 11. — С. 2733—2746.

156. Efficient Large-Scale Traffic Forecasting with Transformers: A Spatial Data Management Perspective [Текст] / Y. Fang [и др.] // Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining. — 2025. — С. 307—317.

157. *He, Z.* Research Based on High-Fidelity NGSIM Vehicle Trajectory Datasets: A Review [Текст] / Z. He. — 2017. — URL: 10.13140/RG.2.2.11429.60643/1. ResearchGate.

158. *Wahlström, J.* Smartphone-Based Vehicle Telematics: A Ten-Year Anniversary [Текст] / J. Wahlström, I. Skog, P. Händel // IEEE Transactions on Intelligent Transportation Systems. — 2017. — Т. 18, № 10. — С. 2802—2825.

159. *Ahmed, H. U.* A Review of Car-Following Models and Modeling Tools for Human and Autonomous-Ready Driving Behaviors in Micro-Simulation [Текст] / H. U. Ahmed, Y. Huang, P. Lu // Smart Cities. — 2021. — Т. 4, № 1. — С. 314—335.

160. Car-Following Models: A Multidisciplinary Review [Текст] / T. T. Zhang [и др.] // IEEE Transactions on Intelligent Vehicles. — 2025. — С. 92—116.

161. FollowNet: A Comprehensive Benchmark for Car-Following Behavior Modeling [Текст] / X. Chen [и др.] // Scientific Data. — 2023. — Т. 10, № 1. — С. 828.

162. Car-Following Models: A Multidisciplinary Review [Текст] / T. Zhang [и др.] // arXiv preprint arXiv:2304.07143. — 2023. — arXiv: 2304.07143 [eess.SY].

163. *Pipes, L. A.* An Operational Analysis of Traffic Dynamics [Текст] / L. A. Pipes // Journal of Applied Physics. — 1953. — Т. 24, № 3. — С. 274—281.

164. *Gazis, D. C.* Nonlinear Follow-the-Leader Models of Traffic Flow [Текст] / D. C. Gazis, R. Herman, R. W. Rothery // Operations Research. — 1961. — Т. 9, № 4. — С. 545—567.

165. *Helly, W.* Simulation of Bottlenecks in Single Lane Traffic Flow [Текст] / W. Helly // Proceedings of the Symposium on Theory of Traffic Flow. — New York : General Motors Research Laboratories, 1959. — С. 207—238.

166. *Newell, G. F.* Nonlinear Effects in the Dynamics of Car Following [Текст] / G. F. Newell // Operations Research. — 1961. — Т. 9, № 2. — С. 209—229.

167. *Jiang, R.* Full Velocity Difference Model for a Car-Following Theory [Текст] / R. Jiang, Q. Wu, Z. Zhu // Physical Review E. — 2001. — Т. 64, № 1. — С. 017101.

168. *Treiber, M.* Congested Traffic States in Empirical Observations and Microscopic Simulations [Текст] / M. Treiber, A. Hennecke, D. Helbing // Physical Review E. — 2000. — Т. 62, № 2. — С. 1805—1824.

169. *Gipps, P. G.* A Behavioural Car-Following Model for Computer Simulation [Текст] / P. G. Gipps // Transportation Research Part B: Methodological. — 1981. — Т. 15, № 2. — С. 105—111.

170. A CNN-LSTM Car-Following Model Considering Generalization Ability [Текст] / P. Qin [и др.] // Sensors. — 2023. — Т. 23, № 2.

171. *Mo, Z.* A Physics-Informed Deep Learning Paradigm for Car-Following Models [Текст] / Z. Mo, R. Shi, X. Di // Transportation Research Part C: Emerging Technologies. — 2021. — Т. 130. — С. 103240.

172. Toward Personalized Car-Following Behaviors From Driver Data Using Learned and Hybrid Control Strategies [Текст] / C. T. Huang [и др.] // 2024 International Conference on System Science and Engineering (ICSSE). — IEEE, 2024. — С. 1—6.

173. *Treiber, M.* Validation of Traffic Flow Models with Respect to the Spatiotemporal Evolution of Congested Traffic Patterns [Текст] / M. Treiber, A. Kesting // Transportation Research Part C: Emerging Technologies. — 2012. — Т. 21, № 1. — С. 31—41.

174. *Montanino, M.* Trajectory Data Reconstruction and Simulation-Based Validation Against Macroscopic Traffic Patterns [Текст] / M. Montanino, V. Punzo // Transportation Research Part B: Methodological. — 2015. — Т. 80. — С. 82—106.

175. A Data-Driven Car-Following Model Based on the Random Forest [Текст] / H. Shi [и др.] // World Journal of Engineering and Technology. — 2021. — Т. 9, № 3. — С. 503—515.

176. Multivariate Analysis on Dynamic Car-Following Data of Non-Lane-Based Traffic Environments [Текст] / S. Das [и др.] // Transportation in Developing Economies. — 2019. — Т. 5. — С. 1—13.

177. *Liu, Y.* A Real-Time Car-Following Data Collecting Method Based on Binocular Stereo Vision [Текст] / Y. Liu, H. Wang, Q. Chen // CICTP 2019. — 2019. — С. 2258—2269.

178. A Car-Following Data Collecting Method Based on Binocular Stereo Vision [Текст] / Y. Liu [и др.] // IEEE Access. — 2020. — Т. 8. — С. 25350—25363.

179. Rapid Driving Style Recognition in Car-Following Using Machine Learning and Vehicle Trajectory Data [Текст] / Q. Xue [и др.] // Journal of Advanced Transportation. — 2019. — Т. 2019, № 1. — С. 9085238.

180. *Abdulrazzaq, A.* Comparative Evaluation of the Moving Car Method for Traffic Data Collection on Multilane Highways [Текст] / A. Abdulrazzaq, M. Y. Taha // Jurnal Teknologi (Sciences & Engineering). — 2025. — Т. 87, № 2. — С. 381—387.

181. Investigating the Long- and Short-Term Driving Characteristics and Incorporating Them into Car-Following Models [Текст] / X. Chen [и др.] // Transportation Research Part C: Emerging Technologies. — 2020. — Т. 117. — С. 102698.

182. *James, R. M.* Methods to Obtain Representative Car-Following Model Parameters from Trajectory-Level Data for Use in Microsimulation [Текст] / R. M. James, B. E. Hammit, S. D. Boyles // Transportation Research Record. — 2019. — Т. 2673, № 7. — С. 62—73.

183. A Car-Following Model to Assess the Impact of V2V Messages on Traffic Dynamics [Текст] / T. Li [и др.] // Transportmetrica B: Transport Dynamics. — 2020. — Т. 8, № 1. — С. 150—165.

184. *El Zorkany, M.* Vehicle to Vehicle "V2V" Communication: Scope, Importance, Challenges, Research Directions and Future [Текст] / M. El Zorkany, A. Yasser, A. I. Galal // The Open Transportation Journal. — 2020. — Т. 14, № 1. — С. 86—98.

185. *Talebpour, A.* Influence of Connected and Autonomous Vehicles on Traffic Flow Stability and Throughput [Текст] / A. Talebpour, H. S. Mahmassani // Transportation Research Part C: Emerging Technologies. — 2016. — Т. 71. — С. 143—163.

186. *Surya, H. R.* Stability Analysis of Mixed Traffic Flow using Car-Following Models on Trajectory Data [Текст] / H. R. Surya, N. Raju, S. S. Arkatkar // 2021 International Conference on COMmunication Systems & NETworkS (COMSNETS). — IEEE, 2021. — С. 656—661.

187. Physical Model versus Artificial Neural Network (ANN) Model: A Comparative Study on Modeling Car-Following Behavior at Signalized Intersections [Текст] / L. Yang [и др.] // Journal of Advanced Transportation. — 2022. — Т. 2022, № 1. — С. 8482846.

188. An Attention-Based Car-Following Model Based on Fused Data [Текст] / L. Z. Wang [и др.] // IEEE Access. — 2023. — Т. 11. — С. 51368—51381.

189. Driver Age and Its Effect on Key Driving Metrics: Insights from Dynamic Vehicle Data [Текст] / A. Joshi [и др.]. — 2025. — eprint: arXiv:2501.06918. — URL: https://arxiv.org/abs/2501.06918. arXiv preprint.

190. TransFollower: Long-Sequence Car-Following Trajectory Prediction through Transformer [Текст] / M. Zhu [и др.] // arXiv preprint arXiv:2202.03183. — 2022. — URL: https://arxiv.org/abs/2202.03183.

191. A Safety Control Method of Car-Following Trajectory Planning Based on LSTM [Текст] / X. Chen [и др.] // Promet – Traffic&Transportation. — 2023. — Т. 35, № 3. — С. 380—394.

192. Investigating the long- and short-term driving characteristics and incorporating them into car-following models [Текст] / X. Chen [и др.] // Transportation Research Part C: Emerging Technologies. — 2020. — Т. 117. — С. 102698.

193. *Montanino*, *M.* Trajectory data reconstruction and simulation-based validation against macroscopic traffic patterns [Текст] / M. Montanino, V. Punzo // Transportation Research Part B: Methodological. — 2015. — Т. 80. — С. 82—106.

194. Horizontal Pod Autoscaling in Kubernetes for Elastic Container Orchestration [Текст] / T.-T. Nguyen [и др.] // Sensors. — 2020. — Т. 20, № 16. — С. 4621.

195. *Augustyn*, *D. R.* Tuning a Kubernetes Horizontal Pod Autoscaler for Meeting Performance and Load Demands in Cloud Deployments [Текст] / D. R. Augustyn, Ł. Wyciślik, M. Sojka // Applied Sciences. — 2024. — Т. 14, № 2. — С. 646.

196. *Dean*, *J.* The Tail at Scale [Текст] / J. Dean, L. A. Barroso // Communications of the ACM. Т. 56. — 2013. — С. 74—80.

197. *Harchol-Balter*, *M.* Performance Modeling and Design of Computer Systems: Queueing Theory in Action [Текст] / M. Harchol-Balter. — Cambridge University Press, 2013.

198. The Dataflow Model: A Practical Approach to Balancing Correctness, Latency, and Cost in Massive-Scale, Unbounded, Out-of-Order Data Processing [Текст] / T. Akidau [и др.] // Proceedings of the VLDB Endowment. Т. 8. — 2015. — С. 1792—1803.

199. Apache Flink: Stream and Batch Processing in a Single Engine [Текст] / P. Carbone [и др.] // IEEE Data Engineering Bulletin. — 2015. — Т. 38, № 4. — С. 28—38.

200. Efficient Memory Management for Large Language Model Serving with PagedAttention [Текст] / W. Kwon [и др.] // Proceedings of the 29th ACM Symposium on Operating Systems Principles (SOSP). — 2023. — arXiv:2309.06180.

201. *Kolluru, S.* Comparative Analysis of Large Language Model Inference Serving Systems: A Performance Study of vLLM and HuggingFace TGI [Текст] / S. Kolluru // arXiv preprint arXiv:2511.17593. — 2025.

202. INFaaS: Automated Model-less Inference Serving [Текст] / F. Romero [и др.] // 2021 USENIX Annual Technical Conference (USENIX ATC 21). — USENIX Association, 2021. — С. 397—411.

203. Serving DNNs like Clockwork: Performance Predictability from the Bottom Up [Текст] / A. Gujarati [и др.] // arXiv preprint arXiv:2006.02464. — 2020.

204. *Merkel, D.* Docker: Lightweight Linux Containers for Consistent Development and Deployment [Текст] / D. Merkel // Linux Journal. — 2014.

205. *Dean, J.* MapReduce: Simplified Data Processing on Large Clusters [Текст] / J. Dean, S. Ghemawat // Communications of the ACM. Т. 51. — 2008. — С. 107—113.

206. Borg, Omega, and Kubernetes [Текст] / B. Burns [и др.] //. Т. 59. — 2016. — С. 50—57.

207. Large-scale Cluster Management at Google with Borg [Текст] / A. Verma [и др.] // Proceedings of the 10th European Conference on Computer Systems (EuroSys). — 2015. — С. 1—17.

208. *Rescorla, E.* The Transport Layer Security (TLS) Protocol Version 1.3 [Текст] / E. Rescorla // RFC 8446. — 2018.

209. TensorFlow-Serving: Flexible, High-Performance ML Serving [Текст] / C. Olston, R. Fiedel, J. Gorovoy [и др.] // Proceedings of the 2017 USENIX Conference on Operational Machine Learning (OpML). — 2017.

210. Clipper: A Low-Latency Online Prediction Serving System [Текст] / D. Crankshaw [и др.] // Proceedings of the 14th USENIX Symposium on Networked Systems Design and Implementation (NSDI). — 2017. — С. 613—627.

211. Multi-model Machine Learning Inference Serving with GPU Spatial Partitioning [Текст] / S. Choi [и др.] // arXiv preprint arXiv:2109.01611. — 2021.

212. *Zhou, Y.* Exploring TensorRT to Improve Real-Time Inference for Deep Learning [Текст] / Y. Zhou, K. Yang // 2022 IEEE 24th Int Conf on High Performance Computing Communications; 8th Int Conf on Data Science & Systems; 20th Int Conf on Smart City; 8th Int Conf on Dependability in Sensor, Cloud & Big Data Systems & Application (HPCC/DSS/SmartCity/DependSys). — 2022. — С. 2011—2018.

213. *NVIDIA*. Triton Inference Server Release 20.03 Release Notes [Текст] / NVIDIA. — 2020. — NVIDIA Triton Inference Server Documentation.

214. The Hadoop Distributed File System [Текст] / K. Shvachko [и др.] // 2010 IEEE 26th Symposium on Mass Storage Systems and Technologies (MSST). — 2010. — С. 1—10.

215. Apache Hadoop YARN: Yet Another Resource Negotiator [Текст] / V. K. Vavilapalli [и др.] // Proceedings of the 4th Annual Symposium on Cloud Computing (SoCC). — 2013. — С. 1—16.

216. Spark: Cluster Computing with Working Sets [Текст] / M. Zaharia [и др.] // Proceedings of the 2nd USENIX Conference on Hot Topics in Cloud Computing (HotCloud). — 2010.

217. Resilient Distributed Datasets: A Fault-Tolerant Abstraction for In-Memory Cluster Computing [Текст] / M. Zaharia [и др.] // Proceedings of the 9th USENIX Symposium on Networked Systems Design and Implementation (NSDI). — 2012. — С. 15—28.

218. Spark SQL: Relational Data Processing in Spark [Текст] / M. Armbrust [и др.] // Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data. — 2015. — С. 1383—1394.

219. *Tapekhin, A.* Analysis of Consistency for In Memory Data Grid Apache Ignite [Текст] / A. Tapekhin, I. Bogomolov, O. Velikanov //. — 2019. — С. 46—50.

220. *LeMay, M.* Perseus: Characterizing Performance and Cost of Multi-Tenant Serving for CNN Models [Текст] / M. LeMay, S. Li, T. Guo // 2020 IEEE International Conference on Cloud Engineering (IC2E). — 2020. — С. 66—72.

221. *gRPC Authors.* gRPC: A High Performance, Open Source Universal RPC Framework [Текст] / gRPC Authors // Technical Report. — 2016.

222. DEFT: SLO-Driven Preemptive Scheduling for Containerized DNN Serving [Текст] / Y. Hao [и др.] // Proceedings of the 20th USENIX Symposium on Networked Systems Design and Implementation (NSDI 23) (Poster). — 2023.

223. *Kastouni, M. Z.* Big data analytics in telecommunications: Governance, architecture and use cases [Текст] / M. Z. Kastouni, A. A. Lahcen // Journal of King Saud University - Computer and Information Sciences. — 2020.

224. *Fu, Y.* Real-time Data Infrastructure at Uber [Текст] / Y. Fu, C. Soman // Proceedings of the 2021 International Conference on Management of Data. — 2021. — С. 2503—2516.

225. Pinot: Realtime OLAP for 530 Million Users [Текст] / J.-F. Im [и др.] // Proceedings of the 2018 International Conference on Management of Data. — 2018. — С. 583—594. — URL: https://doi.org/10.1145/3183713.3190661.

226. *Olawoyin, A. M.* Open Data Lake to Support Machine Learning on Arctic Big Data [Текст] / A. M. Olawoyin, C. K. Leung, A. Cuzzocrea // 2021 IEEE International Conference on Big Data (Big Data). — IEEE. 2021. — С. 5215—5224.

227. *Manoj, V.* Comparative Study of NoSQL Document, Column Store Databases and Evaluation of Cassandra [Текст] / V. Manoj // International Journal of Database Management Systems. — 2014. — Т. 6, № 4. — С. 11—26. — URL: https://doi.org/10.5121/ijdms.2014.6402.

228. *Lakshman, A.* Cassandra: A Decentralized Structured Storage System [Текст] / A. Lakshman, P. Malik // ACM SIGOPS Operating Systems Review. — 2010. — Т. 44, № 2. — С. 35—40. — URL: https://doi.org/10.1145/1773912.1773922.

229. *Cattell, R.* Scalable SQL and NoSQL Data Stores [Текст] / R. Cattell // ACM SIGMOD Record. — 2010. — Т. 39, № 4. — С. 12—27.

230. Bigtable: A Distributed Storage System for Structured Data [Текст] / F. Chang, J. Dean, S. Ghemawat [и др.] // ACM Transactions on Computer Systems. — 2008. — Т. 26, № 2. — С. 1—26.

231. A Comparative Study on Streaming Frameworks for Big Data [Текст] / W. Inoubli [и др.] // Latin America Data Science Workshop (LADaS) in conjunction with the 44th International Conference on Very Large Data Bases (VLDB 2018). — 2018. — С. 17—24.

232. *Katsifodimos, A.* Apache Flink: Stream Analytics at Scale [Текст] / A. Katsifodimos, S. Schelter // 2016 IEEE International Conference on Cloud Engineering Workshop (IC2EW). — IEEE, 2016. — С. 193—193.

233. *Peng, W.* Exploration and Research of Service-oriented Architecture Model in the Course of Enterprise's Network Service Registration [Текст] / W. Peng, L. Huijuan // 2010 International Conference on Computer Design and Applications (ICCDA). — 2010. — С. V5-104—V5-107. — URL: https://doi.org/10.1109/ICCDA.2010.5540874.

234. Software as a Service: Configuration and Customization Perspectives [Текст] / W. Sun [и др.] // 2008 IEEE Congress on Services Part II (services-2 2008). — 2008. — С. 18—25. — URL: https://doi.org/10.1109/SERVICES-2.2008.29.

235. Dynamic Service Substitution in Service-Oriented Architectures [Текст] / M. Fredj [и др.] // 2008 IEEE Congress on Services - Part I. — 2008. — С. 101—104. — URL: https://doi.org/10.1109/SERVICES-1.2008.52.

236. *Amrani, A.* Enhance Journey Planner with Predictive Travel Information for Smart City Routing Services [Текст] / A. Amrani, K. Pasini, M. Khouadjia // 2020 Forum on Integrated and Sustainable Transportation Systems (FISTS). — IEEE. 2020. — С. 304—308.

237. *Arslan, H.* A New Approach Based on Blockchain for User Access Management Problem of IoT Environments [Текст] / H. Arslan, A. G. Yüksel, H. Aslan // 2018 International Conference on Artificial Intelligence and Data Processing (IDAP). — 2018. — С. 1—4. — URL: https://doi.org/10.1109/IDAP.2018.8620906.

238. *Денисов, В. Ю.* Методы многомерного data mining для анализа больших данных [Текст] / В. Ю. Денисов, И. С. Синева // Труды Северо-Кавказского филиала Московского технического университета связи и информатики. — 2016. — № 1. — С. 106—110.

239. *Воронова, Л. И.* Big Data. Методы и средства анализа [Текст] / Л. И. Воронова, В. И. Воронов. — Москва : Московский технический университет связи и информатики, 2016. — С. 33.

240. *Менщиков, А. А.* Основные проблемы использования больших данных в современных информационных системах [Текст] / А. А. Менщиков, В. Э. Перфильев, И. Р. Фабзиев // Столыпинский вестник. — 2022. — № 1.

241. *Zhu, L.* Improved Dynamic Load Balancing Algorithm Based on Least-Connection Scheduling [Текст] / L. Zhu, J. Cui, G. Xiong // 2018 IEEE 4th Information Technology and Mechatronics Engineering Conference (ITOEC). — 2018. — С. 1858—1862. — URL: https://doi.org/10.1109/ITOEC.2018.8740642.

242. Research and Implementation for the Distributed Bridge Structure Monitoring System Based on IEEE 1588 Protocol [Текст] / F. Liu [и др.] // 2010 International Conference on Intelligent Computation Technology and Automation (ICICTA). — 2010. — С. 1035—1038. — URL: https://doi.org/10.1109/ICICTA.2010.256.

243. Revisit Raft Consistency Protocol on Private Blockchain System in High Network Latency [Текст] / N. Cao [и др.] // International Conference on Artificial Intelligence and Security. — Springer, Cham, 2021. — С. 571—579.

244. *Nakagawa, T.* Resource management for raft consensus protocol [Текст] / T. Nakagawa, N. Hayashibara // International Journal of Space-Based and Situated Computing. — 2018. — Т. 8, № 2. — С. 80—87.

245. Incorporating the Raft consensus protocol in containers managed by Kubernetes: An evaluation [Текст] / H. Netto [и др.] // International Journal of Parallel, Emergent and Distributed Systems. — 2020. — Т. 35, № 4. — С. 433—453.

246. *Любимова, Т. В.* Алгоритм консенсуса в распределенных системах [Текст] / Т. В. Любимова // Университетская наука. — 2019. — № 1. — С. 136—139.

247. *Турута, Е. Н.* Комплекс методов обеспечения отказоустойчивости многопроцессорных систем на основе рационального перераспределения задач [Текст] / Е. Н. Турута // Труды 5-го Совещания по распределённым вычислительным системам и сетям (CDS–92). — Калининград, 1992.

248. *Кибедов, Р. А.* Обеспечение отказоустойчивости ВС на основе статического отказоустойчивого планирования задач [Текст] / Р. А. Кибедов, Е. Н. Турута // Телекоммуникации и информационные технологии. — 2019. — Т. 6, № 1. — С. 89—95.

249. *Богатырев, В. А.* Модель надёжности кластера с миграцией виртуальных машин и восстановлением на определённых стадиях деградации системы [Текст] / В. А. Богатырев, С. М. Алексанков, А. Н. Деркач // Волновая электроника и её применения в информационных и телекоммуникационных системах. — 2018. — С. 225—231.

250. *Al-Debagy, O.* A Comparative Review of Microservices and Monolithic Architectures [Текст] / O. Al-Debagy, P. Martinek // 2018 IEEE 18th International Symposium on Computational Intelligence and Informatics (CINTI). — 2018. — С. 149—154.

251. *Гольчевский, Ю. В.* Актуальность использования микросервисов при разработке информационных систем [Текст] / Ю. В. Гольчевский, А. В. Ермоленко // Вестник Сыктывкарского университета. Серия 1. Математика. Механика. Информатика. — 2020. — № 2. — С. 25—36.

252. *Качин, К.* Введение в надежное и безопасное распределенное программирование [Текст] / К. Качин, Р. Гуерру, Л. Родригес. — Москва : ДМК Пресс, 2015. — С. 513.

253. *Petrović, N.* Model-based Approach for Semantic-driven Deployment of Containerized Applications to Support Future Internet Services and Architectures [Текст] / N. Petrović // Serbian Journal of Electrical Engineering. — 2018. — Т. 16, № 1. — С. 21—44.

254. *Комбаров, В. В.* Применение программного обеспечения принятия консенсуса в системе децентрализованного управления распределительными энергетическими сетями [Текст] / В. В. Комбаров, И. В. Полозов, Е. Н. Соснина // Информационные системы и технологии ИСТ-2020. — 2020. — С. 349—355.

255. *Kaur, G.* In-Memory Data Processing Using Redis Database [Текст] / G. Kaur, J. Kaur // International Journal of Computer Applications. — 2018. — Март. — Т. 180. — С. 26—31.

256. *Lv, H.* Highly Reliable Control System based on Reactive Event Stream [Текст] / H. Lv // 2021 IEEE 4th Advanced Information Management, Communicates, Electronic and Automation Control Conference (IMCEC). — 2021. — С. 963—966. — URL: https://doi.org/10.1109/IMCEC51613.2021.9482011.

257. *Агуреев, И. Е.* К вопросу оптимизации маршрутной сети города с применением интеллектуальных транспортных систем [Текст] / И. Е. Агуреев, А. В. Ахромешин, В. А. Пышный // Современные технологии: актуальные вопросы, достижения и инновации. — 2018. — С. 45—50.

258. *Дерюгина, И. А.* Интеллектуальные транспортные системы и технологии [Текст] / И. А. Дерюгина, Н. Д. Берман // ТОГУ-старт: фундаментальные и прикладные исследования молодых. — 2021. — С. 243—249.

259. *Ильяшенко, О. Ю.* Формирование требований к архитектурному решению интеллектуальной транспортной системы с использованием Больших Данных [Текст] / О. Ю. Ильяшенко, В. М. Ильяшенко // Фундаментальные и прикладные исследования в области управления, экономики и торговли. — 2018. — С. 64—70.

260. *Турута, Е. Н.* Организация распределения задач в вычислительных системах, обеспечивающая их отказоустойчивость [Текст] / Е. Н. Турута // Автоматика и вычислительная техника. — 1985. — № 1. — С. 5—14.

261. *Tourouta, E. N.* The methods for ensuring fault-tolerance of distributed control systems [Текст] / E. N. Tourouta // Proceedings of IFAC Symposium SAFEPROCESS'94. Т. 2. — Helsinki, Finland, 1994.

262. *Randell, B.* System Structure for Software Fault-Tolerance [Текст] / B. Randell // IEEE Transactions on Software Engineering. — 1975. — Т. SE—1. — С. 220—232.

263. *Hecht, H.* Fault-Tolerant Software for Real-Time Applications [Текст] / H. Hecht // ACM Computing Surveys. — 1976. — Дек. — Т. 8, № 4. — С. 391—407.

264. *Дегтярев, А. Р.* Отказоустойчивые реконфигурирующиеся комплексы интегрированной модульной авионики [Текст] / А. Р. Дегтярев, С. К. Киселёв // Электротехнические и информационные комплексы и системы. — 2016. — Т. 12, № 1. — С. 88—99.

265. Distributed Fault-tolerant Platform for Web Applications [Текст] / M. M. Rovnyagin [и др.] // 2020 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (EIConRus). — 2020. — С. 482—485. — URL: https://doi.org/10.1109/EIConRus49466.2020.9039529.

266. *Сущенко, С. С.* Надёжность сложных технических систем с временной избыточностью, находящихся в резерве [Текст] / С. С. Сущенко, Е. В. Новиков // Современные информационные технологии. — 2014. — № 10. — С. 56—59.

267. *Грузенкин, Д. В.* Применение программной избыточности для повышения надёжности программного обеспечения [Текст] / Д. В. Грузенкин, С. С. Касымов // Новая наука: От идеи к результату. — 2016. — № 9—1. — С. 9—11.

268. *Мельник, В. А.* Информационная избыточность в узлах непозиционного спецвычислителя для телекоммуникационных устройств [Текст] / В. А. Мельник, Р. В. Кузьменко, В. П. Ирхин // Вестник Воронежского института МВД России. — 2015. — № 2. — С. 149—156.

269. *Богатырев, В. А.* Перераспределение запросов между вычислительными кластерами при их деградации [Текст] / В. А. Богатырев, А. В. Богатырев, С. В. Богатырев // Известия высших учебных заведений. Приборостроение. — 2014. — Т. 57, № 9.

270. *Харченко, А. С.* Оценка способов повышения надёжности отказоустойчивого кластера [Текст] / А. С. Харченко, В. Н. Ручкин // Известия Тульского государственного университета. Технические науки. — 2014. — № 9—2.

271. *Аниконов, А. В.* Мультиверсионное программное обеспечение информационно-управляющих систем [Текст] / А. В. Аниконов, В. А. Морозов, Р. Ю. Царев // Фундаментальные исследования. — 2006. — № 11. — С. 65.

272. *Кадочникова, Н. А.* Подходы к построению отказоустойчивых кластеров [Текст] / Н. А. Кадочникова // Научный альманах. — 2016. — № 10—3. — С. 158—162.

273. *Walker, C.* Personal data lake with data gravity pull [Текст] / C. Walker, H. Alrehamy // 2015 IEEE Fifth International Conference on Big Data and Cloud Computing. — IEEE. 2015. — С. 160—167.

274. *Neumann, P.* High-Performance Techniques for Big Data Processing [Текст] / P. Neumann, J. Kunkel // Knowledge Discovery in Big Data from Astronomy and Earth Observation. — 2020. — С. 137—158.

275. *Ларионова, О. Б.* Особенности облачных технологий [Текст] / О. Б. Ларионова, А. О. Васин, М. В. Семенова // Глобус. — 2019. — Т. 39, № 6. — С. 96—102.

276. *Farooq, M. U.* An Efficient Dynamic Round Robin Algorithm for CPU Scheduling [Текст] / M. U. Farooq, A. Shakoor, A. B. Siddique // IEEE International Conference on Communication, Computing and Digital Systems (C-CODE). — 03.2017. — С. 244—248.

277. KV-Direct: High-performance In-Memory Key-Value Store with Programmable NIC [Текст] / B. Li [и др.] // Proceedings of the 26th Symposium on Operating Systems Principles. — 2017. — С. 137—152.

Приложение A. Профили полей и ограничения

Таблица 47 — Полевой профиль V2X: общие параметры для SPaT/MAP/CAM/DENM

Параметр	Значение/правило
Синхронизация времени	UTC от GNSS; расхождение RSU–контроллер СФ 100 мс; на приёме для CAM/DENM допустимое смещение 500 мс
Система координат и проекции	WGS-84 (EPSG:4326), широта/долгота в 1e-7 градуса; высота при наличии
Безопасность сообщений	Подпись ETSI TS 103 097; обязательная верификация; шифрование не применяется
Ограничение размеров	SPaT 1200 байт; MAP стремится 1200 байт (с редукцией геометрии при превышении); CAM 1400 байт; DENM 800 байт
Частоты публикации	SPaT 10 Гц при изменениях фаз (иначе 2 Гц); MAP 1 Гц либо при изменениях; CAM 1–10 Гц по триггерам; DENM 1 Гц на время актуальности
Политика приёма	Отклонение неподписанных/просроченных кадров; контроль окон времени; устранение дубликатов по идентификатору/времени/хешу

Таблица 48 — SPaT (SAE J2735): профиль полей и ограничений

Элемент/Поле (J2735)	Тип/единицы	Статус	Ограничение/правило
Spat.msgID	enum	M	Идентификатор SPaT по SAE J2735
Spat.timeStamp	ms UTC	O	При наличии; расхождение с RSU 100 мс
Spat.intersections	list	M	Один `IntersectionState` на сообщение
IntersectionState.id.id	uint16	M	1–65535, уникален в городе
IntersectionState.states	list	M	Набор `MovementState` по группам
MovementState.signalGroup	uint8	M	1..63; согласован с контроллером
MovementState.state-time-speed.eventState	enum	M	Допустимые состояния: stop-And-Remain; stop-Then-Proceed; protected/permissive-Movement-Allowed; caution-Conflicting
MovementState.state-time-speed.timing.minEndTime	ms UTC	C	Обязательно при активной фазе; согласовано с `maxEndTime`
MovementState.state-time-speed.timing.maxEndTime	ms UTC	C	Допуск рассогласования с фактом 250 мс

Таблица 49 — MAP (SAE J2735): профиль полей и ограничений

Элемент/Поле (J2735)	Тип/единицы	Статус	Ограничение/правило
MapData.msgID	enum	M	Идентификатор MAP
Intersections.id.*	см. SPaT	M	Совпадает с SPaT
IntersectionGeometry. refPoint.lat, lon	1e-7 град	M	Центр опорной системы узла
laneWidth	1 см	O	При известной ширине
GenericLane.laneID	uint8	M	1..63
GenericLane.nodeList.xy	Node-LLmD	M	До 64 узлов/полоса; WGS-84
GenericLane.connectsTo	list	C	Определять для всех разрешённых манёвров; указывать целевую `laneID` и `signalGroup`

Таблица 50 — DENM (ETSI EN 302 637-3): профиль полей, события и ограничения

Элемент/Поле	Тип/единицы	Статус	Ограничение/правило
Management.actionID	stationID, seq	M	Уникализация источника
Management. detectionTime/ referenceTime	ms UTC	M	Смещение 500 мс
Management.termination	enum	C	`isCancelled` при окончании
Situation.causeCode/ subCauseCode	enum	M	Допустимые типы: accident, stationaryVehicle, roadWorks, hazardousLocation_surfaceCondition, wrongWayDriver, obstacleOnRoad
Location.eventPosition.lat/ lon	1e-7 град	M	WGS-84

Приложение Б. Примеры запросов

Листинг 1: HTTP запрос для потокового SELECT

```
POST /db/v1/select
Content-Type: application/json

{
  "queryId": "...",
  "queryParams": [ ... ]
}
```

Листинг 2: Формат записи результата для потокового SELECT

```
{
  "<column_1_name>": "<value_1>",
  "<column_2_name>": "<value_2>"
  // ...
}
```

Листинг 3: HTTP запрос для пакетного INSERT

```
POST /db/v1/batch-update
Content-Type: application/json

{
  "queryId": "...",
  "queryParamsBatch": [
    { ... },
    { ... }
  ]
}
```

Листинг 4: gRPC запросы `ExecutionRequest` и `BatchExecutionRequest`

```
message ExecutionRequest {
  string query_id = 1;
  ParameterSet parameter_set = 2;
}

message BatchExecutionRequest {
  string query_id = 1;
  repeated ParameterSet parameter_sets = 2;
```

```
}

message ParameterSet {
  map<string, string> values = 1;
}
```

Листинг 5: gRPC ответ `QueryExecutionResponse`

```
message QueryExecutionResponse {
  QueryResult query_result = 1;
  string message = 2;
}

message QueryResult {
  repeated Row rows = 1;        // результат SELECT
  int32 records_affected = 2;   // количество изменений для UPDATE, INSERT,
    DELETE
}

message Row {
  map<string, bytes> values = 1;  // байтовое представление значений колонок
}
```

Приложение В. Декларативные утверждения для unit-тестов конфигурации БД

Листинг 6: Assert-проверки для пути π_1 входного трафика

```
assert Ready("router") && Ready("controller-proxy") && Ready("model")
assert EP("router") && EP("controller-proxy") && EP("model")
assert TLS("1.1") && TLS("1.2") && TLS("1.3")
assert RBAC("1.2", sa="router-sa") && RBAC("1.3", sa="controller-proxy-sa")
t = http_probe(url="https://router/ml/ingress/health", mtls=true)
assert t.ok
t = grpc_health(target="controller-proxy:443", mtls=true)
assert t.ok
t = grpc_health(target="model:443", mtls=true)
assert t.ok
l11 = measure("1.1") ; l12 = measure("1.2") ; l13 = measure("1.3")
assert (l11 + l12 + l13) <= _inf
```

Листинг 7: Assert-проверки для пути π_2 обращения к БД

```
assert Ready("model") && Ready("controller-database") && Ready("system-db")
assert EP("controller-database") && EP("system-db")
assert TLS("2.1") && TLS("2.2") && TLS("3.1") && TLS("3.2")
assert RBAC("2.1", sa="model-sa") && RBAC("3.2", sa="controller-db-sa")
assert MNT("controller-database", secret="db-credentials")
t = grpc_health(target="controller-database:443", mtls=true)
assert t.ok
T1 = grpc_exec(target="controller-database:443",
               method="Execute",
               payload=SELECT 1,
               mtls=true)
assert T1.code == OK && T1.result.rows == [[1]]
l21 = measure("2.1") ; l22 = measure("2.2") ; l31 = measure("3.1") ; l32 =
   measure("3.2")
assert (l21 + l22 + l31 + l32) <= _db
```

Приложение Г. Декларативные утверждения для unit-тестов конфигурации GPU

Листинг 8: Assert-проверки окружения GPU

```
# Узлы с устройствами
assert exists(node in cluster where label("nvidia.com/gpu") && allocatable("
    nvidia.com/gpu") >= 1)

# Плагин устройств и драйвер
assert daemonset_ready("nvidia-device-plugin-daemonset")
assert driver_version(node) >= V_MIN

# Совместимость контейнера модели и видимость устройства
assert image_has_cuda(model_image) && image_has_cudnn(model_image)
assert cuda_driver_compatible(model_image, node)
assert pod_exec(model_pod, "nvidia-smi -L").ok
assert env(model_pod, "CUDA_VISIBLE_DEVICES") != ""

# Наличие MIG (если требуется)
if require_mig:
  assert mig_slices(node) >= K_MIN
  assert pod_bound_to_mig(model_pod)
```

Листинг 9: Микро-бенчмарк для оценки $\mathrm{T}_h 2d, T_k ern, T_d 2h L_s tep(b)$

```
# Конфигурация бенчмарка
b = BATCH_SIZE            # размер микропакета
dtype = "fp16" or "fp32"  # режим точности по политике
iters = 100

# Измерения при двойной буферизации и раздельных CUDA streams
times = []
for i in range(iters):
  t0 = now()
  h2d();              # асинхронная копия хост -> устройство
  k0 = now()
  kernel();           # вычисление ядра модели
  d0 = now()
  d2h();              # асинхронная копия устройство -> хост
  t1 = now()
  times.append({ "h2d": k0-t0, "kern": d0-k0, "d2h": t1-d0, "step": t1-t0 })
```

```

T_h2d = mean(t["h2d"] for t in times)
T_kern = mean(t["kern"] for t in times)
T_d2h = mean(t["d2h"] for t in times)
L_step = mean(t["step"] for t in times)
sL = std(t["step"] for t in times)

# Критерии прохождения
assert T_h2d <= TH_H2D && T_kern <= TH_KERN && T_d2h <= TH_D2H
assert L_step + z(1-alpha) * sL / sqrt(iters) <= _gpu
```

Приложение Д. Декларативные утверждения для автоматических проверок

Листинг 10: Assert-проверки готовности кластера Hadoop и Spark

```
# Доступность ролей Spark и HDFS
assert service_ready("spark-master") && endpoints("spark-master")
assert all(workers_ready())
assert service_ready("hdfs-namenode") && service_ready("hdfs-datanodes")

# Безопасность
assert tls_ok("router<->proxy") && tls_ok("proxy<->model")
assert tls_ok("model<->spark-master") && tls_ok("proxy<->system")
assert kerberos_kinit(keytab="hadoop.controller.keytab").ok

# Права и секреты
assert secret_present("hadoop-secrets") && sa_has_scope("controller-hadoop-sa
    ","hdfs:rw")

# Контрольная отправка задания
R = spark_submit(app="healthcheck.py", args="--size 1GB --partitions 8")
assert R.status == "SUCCEEDED"
```

Листинг 11: Измерение производительности на контрольном наборе

```
S = 1 * TB
B = 128 * MB            # размер блока HDFS
P = ceil(S / B)
E = executors()         # число executor
C = cores_per_executor()

# Рекомендованный уровень параллелизма
Q = min(P, E*C)
assert Q >= E*C

# Запуск и измерение
K = 5
times = []
for i in range(K):
  t = spark_submit(app="batch_inference.py",
                   args=f"--input hdfs:///its/data --partitions {P}",
                   conf={"spark.executor.instances": E,
```

```
                           "spark.executor.cores": C})
  assert t.status == "SUCCEEDED"
  times.append(t.duration)

Tbar = mean(times); s = std(times)
assert Tbar + z(0.95)*s/sqrt(K) <= _hdp
```

Приложение E. Декларативные утверждения для unit-тестов конфигурации S3

Листинг 12: Assert-проверки готовности и доступа к S3

```
assert service_ready("controller-storage") && endpoints("controller-storage")
assert secret_present("s3-credentials") && mount_ok("controller-storage","s3-credentials")
assert tls_ok("system<->proxy") && tls_ok("proxy<->model") && tls_ok("model<->controller-storage")
assert s3_bucket_exists("its-bucket")
assert s3_object_head("its-bucket","path/to/object").ok
```

Листинг 13: Микро-бенчмарк потокового чтения из S3

```
S = object_size("its-bucket","path/to/object")
c = CHUNK_SIZE; k = CONCURRENCY; iters = 5
times = []
for i in range(iters):
  t = s3_stream_read(bucket="its-bucket", key="path/to/object",
                     chunk=c, parallel=k, client="minio")
  assert t.etag == local_etag() or multipart_etag_match(t.parts)
  times.append(t.duration)
Lbar = mean(times); s = std(times)
assert Lbar + z(0.95)*s/sqrt(iters) <= _S3
```

Приложение Ж. Декларативные утверждения для unit-тестов конфигурации In-Memory БД

Листинг 14: Assert-проверки кластера In-Memory БД

```
# Топология и готовность
assert service_ready("imdb-coordinator") && endpoints("imdb-coordinator")
assert all(pod_ready(p) for p in pods(label="imdb-node"))
assert topology_size() == k && replication_factor(cache="its-cache") == r

# Безопасность и доступ
assert tls_ok("model<->imdb") && mTLS_version("model<->imdb") >= "1.2"
assert clusterIP_open(src="model", dst="imdb-coordinator")

# Проверка вместимости
assert k * (32.0 - 0.3) >= 3.0 * n

# Функциональные операции
put("its-cache", key="edge:123|t:456", value=payload)
val = get("its-cache", key="edge:123|t:456")
assert val == payload
```

Приложение З. Декларативные утверждения для unit-тестов контура HTTPS/gRPC

Листинг 15: Assert-проверки контура HTTPS/gRPC и маршрута predict$_o$*nline*

```
# Готовность сервисов и эндпоинтов
assert service_ready("controller-proxy") && endpoints("controller-proxy")
assert service_ready("model") && endpoints("model")
assert service_ready("router") && endpoints("router")

# Безопасность и секреты
assert secret_present("tls-proxy") && mount_ok("controller-proxy","tls-proxy
    ")
assert mTLS_version("system<->proxy") >= "1.2"

# Доступность маршрута и типы полезной нагрузки
r1 = https_post("https://host/model_name/predict_online",
                json={"vector":[...]}, mtls=True)
assert r1.code == 200 && content_type(r1) in {"application/json","application
    /protobuf"}

# Нагрузочная проверка SLA
K = 20
times = []
for i in range(K):
  t = https_post("https://host/model_name/predict_online",
                 protobuf=request_pb, mtls=True).duration
  times.append(t)
Lbar = mean(times); s = std(times)
assert Lbar + z(0.95) * s / sqrt(K) <= _https
```

Приложение И. Декларативные утверждения для unit-тестов конфигурации Kafka

Листинг 16: Assert-проверки контура Kafka и измерение SLA

```
# Топики, секции и ACL
assert topic_exists("ITS.IN") and partitions("ITS.IN") >= P_in and
   replication("ITS.IN") == r_k
assert topic_exists("ITS.OUT") and partitions("ITS.OUT") >= P_out and
   replication("ITS.OUT") == r_k
assert acl_allows("system","ITS.IN","WRITE") and acl_allows("system","ITS.OUT
   ","READ")

# Безопасность и транзакционная доставка
assert mtls_ok(broker="kafka-broker") and idempotent_producer_enabled()
assert transactional_id_configured("controller-kafka")

# Нагрузочная проверка
K = 50
times = []
for i in range(K):
  cid = uuid()
  produce(topic="ITS.IN", key=cid, value=payload, transactional=True)
  t = measure_until(lambda: consume(topic="ITS.OUT", key=cid))
  times.append(t)
Lbar = mean(times); s = std(times)
assert Lbar + z(0.95)*s/sqrt(K) <= _kaf
```

Приложение К. Свидетельства о государственной регистрации программ для ЭВМ

РОССИЙСКАЯ ФЕДЕРАЦИЯ

RU2026611997

ФЕДЕРАЛЬНАЯ СЛУЖБА
ПО ИНТЕЛЛЕКТУАЛЬНОЙ СОБСТВЕННОСТИ

ГОСУДАРСТВЕННАЯ РЕГИСТРАЦИЯ ПРОГРАММЫ ДЛЯ ЭВМ

Номер регистрации (свидетельства):
2026611997
Дата регистрации: **22.01.2026**
Номер и дата поступления заявки:
2026610353 15.01.2026
Дата публикации и номер бюллетеня:
22.01.2026 Бюл. № 2

Автор(ы):
Городничев Михаил Геннадьевич (RU),
Латыпов Эрнест Марселевич (RU),
Матюнина Дарья Дмитриевна (RU),
Полянцева Ксения Андреевна (RU)
Правообладатель(и):
Ордена Трудового Красного Знамени федеральное государственное бюджетное образовательное учреждение высшего образования «Московский технический университет связи и информатики» (RU)

Название программы для ЭВМ:
Интеллектуальная платформа адаптивного управления светофорами

Реферат:
Программа представляет собой адаптивную систему на основе искусственного интеллекта, предназначенную для оптимизации транспортных и пешеходных потоков в городской среде. Основные функциональные возможности: динамическое управление светофорами; интеграция компьютерного зрения для детекции транспортных средств и пешеходов; моделирование транспортных потоков в среде SUMO; сбор и обработка данных с датчиков движения для формирования входных векторов состояния среды; обучение с подкреплением для адаптации к изменяющимся условиям трафика; генерация аналитических отчетов с визуализацией результатов моделирования; поддержка работы в режиме реального времени с возможностью интеграции в системы мониторинга «умного города».

Язык программирования: Python

Объем программы для ЭВМ: 65,2 Мб

Стр.: 1

РОССИЙСКАЯ ФЕДЕРАЦИЯ

ФЕДЕРАЛЬНАЯ СЛУЖБА
ПО ИНТЕЛЛЕКТУАЛЬНОЙ СОБСТВЕННОСТИ

RU2025661566

ГОСУДАРСТВЕННАЯ РЕГИСТРАЦИЯ ПРОГРАММЫ ДЛЯ ЭВМ

Номер регистрации (свидетельства):
2025661566
Дата регистрации: **07.05.2025**
Номер и дата поступления заявки:
2025660681 07.05.2025
Дата публикации и номер бюллетеня:
07.05.2025 Бюл. № 5

Автор(ы):
Городничев Михаил Геннадьевич (RU)
Правообладатель(и):
Ордена Трудового Красного Знамени федеральное государственное бюджетное образовательное учреждение высшего образования «Московский технический университет связи и информатики» (RU)

Название программы для ЭВМ:
Программная реализация метода моделирования транспортных потоков с учетом физических характеристик одиночных транспортных средств

Реферат:
Программная реализация метода моделирования транспортных потоков с учетом физических характеристик одиночных транспортных средств представляет собой инструмент для анализа и оптимизации дорожного движения. Метод учитывает индивидуальные параметры транспортных средств. Благодаря этому, система способна более точно предсказывать динамику транспортного потока и его реакцию на различные внешние условия. Система интегрируется с различными платформами транспортного мониторинга и управления. Данный функционал важен для планировщиков, инженеров и управляющих транспортными сетями, так как позволяют заранее оценивать различные сценарии дорожной ситуации и принимать обоснованные решения.

Язык программирования: C#

Объем программы для ЭВМ: 24,3 Мб

РОССИЙСКАЯ ФЕДЕРАЦИЯ

ФЕДЕРАЛЬНАЯ СЛУЖБА
ПО ИНТЕЛЛЕКТУАЛЬНОЙ СОБСТВЕННОСТИ

RU2025662580

ГОСУДАРСТВЕННАЯ РЕГИСТРАЦИЯ ПРОГРАММЫ ДЛЯ ЭВМ

Номер регистрации (свидетельства):
2025662580
Дата регистрации: **22.05.2025**
Номер и дата поступления заявки:
2025661315 07.05.2025
Дата публикации и номер бюллетеня:
22.05.2025 Бюл. № 6
Контактные реквизиты:
нет

Автор(ы):
Городничев Михаил Геннадьевич (RU)
Правообладатель(и):
Ордена Трудового Красного Знамени федеральное государственное бюджетное образовательное учреждение высшего образования «Московский технический университет связи и информатики» (RU)

Название программы для ЭВМ:
Программное обеспечение генерации синтетических данных о состоянии транспортного потока

Реферат:
Программа предназначена для имитации и анализа различных сценариев дорожного движения, предоставляя исследователям и инженерам возможность тестирования алгоритмов без необходимости сбора реальных данных. Используя методы машинного обучения и моделирования, способна создавать детализированные и достоверные модели транспортных потоков, учитывающие широкий спектр переменных. Система генерирует данные, имитируя реальную динамику транспортных узлов, включая перекрестки, магистрали и городские улицы. Возможности настройки параметров генерации данных обеспечивают гибкость и адаптацию под различные географические и инфраструктурные условия. Программа поддерживает интеграцию с внешними аналитическими инструментами и платформами, что расширяет её применение в различных областях.

Язык программирования: Java

Объем программы для ЭВМ: 52 МБ

РОССИЙСКАЯ ФЕДЕРАЦИЯ

ФЕДЕРАЛЬНАЯ СЛУЖБА
ПО ИНТЕЛЛЕКТУАЛЬНОЙ СОБСТВЕННОСТИ

RU2025662817

ГОСУДАРСТВЕННАЯ РЕГИСТРАЦИЯ ПРОГРАММЫ ДЛЯ ЭВМ

Номер регистрации (свидетельства):
2025662817
Дата регистрации: **23.05.2025**
Номер и дата поступления заявки:
2025661298 07.05.2025
Дата публикации и номер бюллетеня:
23.05.2025 Бюл. № 6
Контактные реквизиты:
нет

Автор(ы):
Городничев Михаил Геннадьевич (RU)
Правообладатель(и):
Ордена Трудового Красного Знамени федеральное государственное бюджетное образовательное учреждение высшего образования «Московский технический университет связи и информатики» (RU)

Название программы для ЭВМ:
Автоматизированная информационная система распознавания и восстановления объектов дорожной инфраструктуры

Реферат:
Автоматизированная информационная система распознавания и восстановления объектов дорожной инфраструктуры предоставляет уникальные возможности для реконструкции и анализа инфраструктурных компонентов на основе монокулярных изображений. Используя методы компьютерного зрения и глубокого обучения, система преобразует двумерные фотоизображения в трехмерные модели ключевых объектов, включая дорожные знаки, барьеры, светофоры и другие элементы. В основе системы лежит алгоритм восстановления глубины и структуры сцены, который анализирует визуальную информацию и генерирует облако точек с высокой точностью. Это позволяет инженерам и планировщикам получать важные метрики, такие как размеры, объемы и пространственные расположения объектов, что играет критическую роль в управлении дорожной инфраструктурой.

Язык программирования: Python

Объем программы для ЭВМ: 212 МБ

РОССИЙСКАЯ ФЕДЕРАЦИЯ

ФЕДЕРАЛЬНАЯ СЛУЖБА
ПО ИНТЕЛЛЕКТУАЛЬНОЙ СОБСТВЕННОСТИ

RU2025663312

ГОСУДАРСТВЕННАЯ РЕГИСТРАЦИЯ ПРОГРАММЫ ДЛЯ ЭВМ

Номер регистрации (свидетельства):
2025663312
Дата регистрации: **27.05.2025**
Номер и дата поступления заявки:
2025661338 07.05.2025
Дата публикации и номер бюллетеня:
27.05.2025 Бюл. № 6
Контактные реквизиты:
нет

Автор(ы):
Городничев Михаил Геннадьевич (RU)
Правообладатель(и):
Ордена Трудового Красного Знамени федеральное государственное бюджетное образовательное учреждение высшего образования «Московский технический университет связи и информатики» (RU)

Название программы для ЭВМ:
Программный модуль динамического распределения ресурсов в подсистемах интеллектуальных транспортных систем мониторинга и управления

Реферат:
Программный модуль представляет собой интегрированное решение, разработанное для оптимизации и эффективного управления ресурсами в реальном времени. Модуль обеспечивает адаптивное распределение вычислительных ресурсов, учитывая переменные нагрузки и изменение внешних условий. Основной функционал включает в себя анализ данных с датчиков транспортных средств, прогнозирование загруженности маршрутов и обеспечение приоритета задач, критически важных для безопасности и оперативности принятия решений. Программный модуль минимизирует риски перегрузки системы, обеспечивая стабильную и бесперебойную работу подсистем интеллектуальной транспортной системы. Программный модуль может применяться в области построения современных умных городов, способствуя повышению уровня безопасности и оптимальности дорожного движения.

Язык программирования: Java

Объем программы для ЭВМ: 24,8 МБ

РОССИЙСКАЯ ФЕДЕРАЦИЯ

ФЕДЕРАЛЬНАЯ СЛУЖБА
ПО ИНТЕЛЛЕКТУАЛЬНОЙ СОБСТВЕННОСТИ

RU2025663314

ГОСУДАРСТВЕННАЯ РЕГИСТРАЦИЯ ПРОГРАММЫ ДЛЯ ЭВМ

Номер регистрации (свидетельства):
2025663314
Дата регистрации: **27.05.2025**
Номер и дата поступления заявки:
2025661275 07.05.2025
Дата публикации и номер бюллетеня:
27.05.2025 Бюл. № 6
Контактные реквизиты:
Нет

Автор(ы):
Городничев Михаил Геннадьевич (RU)
Правообладатель(и):
Ордена Трудового Красного Знамени федеральное государственное бюджетное образовательное учреждение высшего образования «Московский технический университет связи и информатики» (RU)

Название программы для ЭВМ:
Программная реализация метода поиска нейросетевых алгоритмов управления технологическим процессом на транспорте

Реферат:
Программа ориентирована на повышение эффективности и автоматизации управления в транспортной отрасли. Программа комбинирует методы машинного обучения и оптимизационные алгоритмы для создания адаптивных нейросетевых моделей, которые могут обучаться на основе реальных данных о транспортных процессах. Основной функцией программы является автоматизированный подбор и настройка параметров нейросетевых алгоритмов, включая архитектуру сети, функции активации и скорости обучения, что позволяет добиться максимальной точности и эффективности управления. Программа поддерживает интеграцию с различными транспортными платформами и системами управления, обеспечивая обмен данными в реальном времени и адаптацию к изменяющимся условиям эксплуатации. Внедрение программы способствует цифровизации транспортных процессов, улучшая общую эффективность отрасли.

Язык программирования: Python

Объем программы для ЭВМ: 71 МБ

РОССИЙСКАЯ ФЕДЕРАЦИЯ

ФЕДЕРАЛЬНАЯ СЛУЖБА
ПО ИНТЕЛЛЕКТУАЛЬНОЙ СОБСТВЕННОСТИ

RU2025664002

ГОСУДАРСТВЕННАЯ РЕГИСТРАЦИЯ ПРОГРАММЫ ДЛЯ ЭВМ

Номер регистрации (свидетельства):
2025664002
Дата регистрации: **02.06.2025**
Номер и дата поступления заявки:
2025662313 19.05.2025
Дата публикации и номер бюллетеня:
02.06.2025 Бюл. № 6

Автор(ы):
Городничев Михаил Геннадьевич (RU)
Правообладатель(и):
Ордена Трудового Красного Знамени федеральное государственное бюджетное образовательное учреждение высшего образования «Московский технический университет связи и информатики» (RU)

Название программы для ЭВМ:
Автоматизированная информационная система поддержки функционирования интеллектуальной транспортной системы с учетом формата обрабатываемой и хранимой информации

Реферат:
Программа представляет собой комплексное решение, предназначенное для интеграции, анализа и управления широким спектром транспортных данных. Программа обеспечивает эффективную обработку информации, поступающей от различных сенсоров, камер и других устройств, используемых для мониторинга транспортных потоков. Одной из ключевых особенностей является адаптивное управление форматами данных, что позволяет интегрировать информацию из разнородных источников, включая текстовые, числовые и мультимедийные данные, в единую платформу. Программа гарантирует надежное хранение данных. В результате реализуется более устойчивая транспортная среда, повышающая уровень безопасности и улучшая качество обслуживания участников дорожного движения.

Язык программирования: Python

Объем программы для ЭВМ: 105,8 МБ

РОССИЙСКАЯ ФЕДЕРАЦИЯ

ФЕДЕРАЛЬНАЯ СЛУЖБА
ПО ИНТЕЛЛЕКТУАЛЬНОЙ СОБСТВЕННОСТИ

RU2025664922

ГОСУДАРСТВЕННАЯ РЕГИСТРАЦИЯ ПРОГРАММЫ ДЛЯ ЭВМ

Номер регистрации (свидетельства):
2025664922
Дата регистрации: **09.06.2025**
Номер и дата поступления заявки:
2025661305 07.05.2025
Дата публикации и номер бюллетеня:
09.06.2025 Бюл. № 6

Автор(ы):
Городничев Михаил Геннадьевич (RU)
Правообладатель(и):
Ордена Трудового Красного Знамени федеральное государственное бюджетное образовательное учреждение высшего образования «Московский технический университет связи и информатики» (RU)

Название программы для ЭВМ:
Программная реализация архитектуры, обеспечивающей информационный процесс передачи информации между одиночными транспортными средствами (V2V)

Реферат:
Программная представляет собой решение для повышения безопасности и эффективности дорожного движения. Предлагаемая архитектура позволяет транспортным средствам обмениваться данными в режиме реального времени, создавая сетевую среду, способную оперативно реагировать на изменения дорожной обстановки. Основные функции системы включают обмен информацией о скорости, направлении движения, расстоянии между транспортными средствами и других параметрах, критически важных для предотвращения аварий. Архитектура обмена между одиночными транспортными средствами позволяет интеллектуальной транспортной системе адаптироваться к динамическим условиям дорожного движения, оптимизируя маршруты и минимизируя заторы. Данное решение способствует созданию более безопасной и интеллектуальной транспортной системы, снижая количество дорожно-транспортных происшествий и улучшая общую пропускную способность дорог.

Язык программирования: Python

Объем программы для ЭВМ: 12,3 МБ

РОССИЙСКАЯ ФЕДЕРАЦИЯ

ФЕДЕРАЛЬНАЯ СЛУЖБА
ПО ИНТЕЛЛЕКТУАЛЬНОЙ СОБСТВЕННОСТИ

RU2025666160

ГОСУДАРСТВЕННАЯ РЕГИСТРАЦИЯ ПРОГРАММЫ ДЛЯ ЭВМ

Номер регистрации (свидетельства):
2025666160
Дата регистрации: **24.06.2025**
Номер и дата поступления заявки:
2025664984 04.06.2025
Дата публикации и номер бюллетеня:
24.06.2025 Бюл. № 7
Контактные реквизиты:
нет

Автор(ы):
Мкртчян Грач Маратович (RU),
Павликов Артем Евгеньевич (RU),
Тимчук Андрей Васильевич (RU),
Мокин Александр Игоревич (RU),
Городничев Михаил Геннадьевич (RU)
Правообладатель(и):
Ордена Трудового Красного Знамени федеральное государственное бюджетное образовательное учреждение высшего образования «Московский технический университет связи и информатики» (RU)

Название программы для ЭВМ:
RouteGenius — интеллектуальная платформа для планирования маршрутов автотуристов

Реферат:
Программа предназначена для автоматизации создания персонализированных маршрутов для автотуристов. Программа реализует поиск достопримечательностей, сервисов (ночлег, заправки) и рекомендаций, используя LLM (Large Language Model) для интерпретации нечётких запросов пользователей и формирования оптимальных путей с учётом их предпочтений и условий поездки. Программа находит применение в туристических агентствах и онлайн-сервисах для путешественников, автоклубах и сообществах автотуристов, логистических компаниях, оптимизирующих маршруты грузоперевозок с учётом точек обслуживания, крупных агрегаторах туристических услуг, автопроизводителях, внедряющих умные системы навигации, а также государственных туристических ассоциациях и нацпарках.

Язык программирования: Python, Typescript

Объем программы для ЭВМ: 3,97 КБ

РОССИЙСКАЯ ФЕДЕРАЦИЯ

ФЕДЕРАЛЬНАЯ СЛУЖБА
ПО ИНТЕЛЛЕКТУАЛЬНОЙ СОБСТВЕННОСТИ

RU2021662524

ГОСУДАРСТВЕННАЯ РЕГИСТРАЦИЯ ПРОГРАММЫ ДЛЯ ЭВМ

Номер регистрации (свидетельства):
2021662524
Дата регистрации: **30.07.2021**
Номер и дата поступления заявки:
2021661503 21.07.2021
Дата публикации и номер бюллетеня:
30.07.2021 Бюл. № 8

Автор(ы):
Городничев Михаил Геннадьевич (RU),
Мосева Марина Сергеевна (RU),
Полянцева Ксения Андреевна (RU)
Правообладатель(и):
ОБЩЕСТВО С ОГРАНИЧЕННОЙ ОТВЕТСТВЕННОСТЬЮ "КИБЕР АЙ КЬЮ" (RU)

Название программы для ЭВМ:
Программный модуль определения основных характеристик транспортного потока (интенсивность, плотность, качественный состав) на основе видеопотока

Реферат:
Программа предназначена для определения посредством анализа видеопотока основных характеристик транспортного потока, таких как: интенсивность, плотность и качественный состав. Программа может использоваться для анализа транспортного потока на перегонах для управления и оптимизации движения транспорта. Функциональные возможности программы: расчет интенсивности транспортного потока, расчет плотности транспортного потока, определение типа транспортного средства, подсчет количества транспортных средств. ОС: Linux, MacOS, Windows.

Язык программирования: Python

Объем программы для ЭВМ: 100 КБ

РОССИЙСКАЯ ФЕДЕРАЦИЯ

ФЕДЕРАЛЬНАЯ СЛУЖБА
ПО ИНТЕЛЛЕКТУАЛЬНОЙ СОБСТВЕННОСТИ

RU2020612021

ГОСУДАРСТВЕННАЯ РЕГИСТРАЦИЯ ПРОГРАММЫ ДЛЯ ЭВМ

Номер регистрации (свидетельства):
2020612021
Дата регистрации: 13.02.2020
Номер и дата поступления заявки:
2020610981 03.02.2020
Дата публикации и номер бюллетеня:
13.02.2020 Бюл. № 2
Контактные реквизиты:
+7(495) 957-77-43, oiris.patent@gmail.com

Автор(ы):
Городничев Михаил Геннадьевич (RU),
Мосева Марина Сергеевна (RU),
Полянцева Ксения Андреевна (RU)
Правообладатель(и):
Ордена Трудового Красного Знамени федеральное государственное бюджетное образовательное учреждение высшего образования «Московский технический университет связи и информатики» (МТУСИ) (RU)

Название программы для ЭВМ:
Мобильное приложение многопараметрического сбора основных характеристик транспортных потоков под управлением ОС iOS

Реферат:
Программа осуществляет сбор следующих данных, необходимых для дальнейшего анализа основных характеристик транспортных потоков: данные акселерометра (по осям X, Y, Z), данные гироскопа (по осям X, Y, Z), географические координаты устройства (широта, долгота), скорость транспортного средства. Программа является клиентской частью распределенной интеллектуальной системы сбора и анализа качественных параметров движения частиц в транспортном потоке и будет использоваться для решения задач моделирования и управления транспортными потоками. Собираемые числовые и видеоданные могут использоваться в дальнейшем в качестве параметров для машинного обучения и классификации событий. ОС: iOS.

Язык программирования: Swift

Объем программы для ЭВМ: 34 Мб

РОССИЙСКАЯ ФЕДЕРАЦИЯ

ФЕДЕРАЛЬНАЯ СЛУЖБА
ПО ИНТЕЛЛЕКТУАЛЬНОЙ СОБСТВЕННОСТИ

RU2020612467

ГОСУДАРСТВЕННАЯ РЕГИСТРАЦИЯ ПРОГРАММЫ ДЛЯ ЭВМ

Номер регистрации (свидетельства):
2020612467
Дата регистрации: **25.02.2020**
Номер и дата поступления заявки:
2020611426 11.02.2020
Дата публикации и номер бюллетеня:
25.02.2020 Бюл. № 3
Контактные реквизиты:
+7(495) 957-77-43, oiris.patent@gmail.com

Автор(ы):
Городничев Михаил Геннадьевич (RU),
Мосева Марина Сергеевна (RU)
Правообладатель(и):
Ордена Трудового Красного Знамени федеральное государственное бюджетное образовательное учреждение высшего образования «Московский технический университет связи и информатики» (МТУСИ) (RU)

Название программы для ЭВМ:
Программная реализация иерархической кластеризации поведенческого фактора водителей методами машинного обучения

Реферат:
Разработанная программа представляет собой алгоритм кластеризации параметров транспортных потоков для решения задачи распознавания психологических типов водителей с целью усовершенствования и повышения эффективности управления транспортными потоками. Разработанный алгоритм может использоваться в качестве модуля интеллектуальной транспортной системы для мониторинга и управления транспортными потоками. ОС: Windows, Linux, macOS.

Язык программирования: Python

Объем программы для ЭВМ: 6 Мб

РОССИЙСКАЯ ФЕДЕРАЦИЯ

ФЕДЕРАЛЬНАЯ СЛУЖБА
ПО ИНТЕЛЛЕКТУАЛЬНОЙ СОБСТВЕННОСТИ

RU2020612482

ГОСУДАРСТВЕННАЯ РЕГИСТРАЦИЯ ПРОГРАММЫ ДЛЯ ЭВМ

Номер регистрации (свидетельства):
2020612482
Дата регистрации: **25.02.2020**
Номер и дата поступления заявки:
2020610977 03.02.2020
Дата публикации и номер бюллетеня:
25.02.2020 Бюл. № 3
Контактные реквизиты:
+7(495)957-77-43; oiris.patent@gmail.com

Автор(ы):
Городничев Михаил Геннадьевич (RU),
Мосева Марина Сергеевна (RU),
Полянцева Ксения Андреевна (RU)
Правообладатель(и):
Ордена Трудового Красного Знамени федеральное государственное бюджетное образовательное учреждение высшего образования «Московский технический университет связи и информатики» (МТУСИ) (RU)

Название программы для ЭВМ:
Программная платформа мониторинга и управления движением беспилотных транспортных средств

Реферат:
Целью программы является интеграция всех транспортных средств в единую киберсистему с целью обеспечения должного уровня безопасности и оптимальности передвижения транспортных потоков. Использование разработанного комплекса позволит прогнозировать и управлять движением транспортных средств, а также проводить оценку влияния задержки управляющего воздействия на безопасность движения в различных дорожных условиях. Инновационность программы будет обеспечиваться за счет создания многоуровневой системы мониторинга и прогнозирования. Тип ЭВМ: IBM PC-совмест. ПК; ОС: Windows; Mac OS; Linux.

Язык программирования: JavaScript

Объем программы для ЭВМ: 34 Мб

РОССИЙСКАЯ ФЕДЕРАЦИЯ

ФЕДЕРАЛЬНАЯ СЛУЖБА
ПО ИНТЕЛЛЕКТУАЛЬНОЙ СОБСТВЕННОСТИ

RU2020612483

ГОСУДАРСТВЕННАЯ РЕГИСТРАЦИЯ ПРОГРАММЫ ДЛЯ ЭВМ

Номер регистрации (свидетельства):
2020612483
Дата регистрации: **25.02.2020**
Номер и дата поступления заявки:
2020610979 03.02.2020
Дата публикации и номер бюллетеня:
25.02.2020 Бюл. № 3
Контактные реквизиты:
+7(495)957-77-43; oiris.patent@gmail.com

Автор(ы):
Городничев Михаил Геннадьевич (RU),
Мосева Марина Сергеевна (RU),
Полянцева Ксения Андреевна (RU)
Правообладатель(и):
Ордена Трудового Красного Знамени федеральное государственное бюджетное образовательное учреждение высшего образования «Московский технический университет связи и информатики» (МТУСИ) (RU)

Название программы для ЭВМ:
Мобильное приложение многопараметрического сбора основных характеристик транспортных потоков под управлением ОС Android

Реферат:
Программа осуществляет сбор следующих данных, необходимых для дальнейшего анализа основных характеристик транспортных потоков: данные акселерометра (по осям X, Y, Z), данные гироскопа (по осям X, Y, Z), географические координаты устройства (широта, долгота), скорость транспортного средства. Программа является клиентской частью распределенной интеллектуальной системы сбора и анализа качественных параметров движения частиц в транспортном потоке и будет использоваться для решения задач моделирования и управления транспортными потоками. Собираемые числовые и видео-данные могут использоваться в дальнейшем в качестве параметров для машинного обучения и классификации событий. ОС: Android.

Язык программирования: Java

Объем программы для ЭВМ: 21 Мб

РОССИЙСКАЯ ФЕДЕРАЦИЯ

ФЕДЕРАЛЬНАЯ СЛУЖБА
ПО ИНТЕЛЛЕКТУАЛЬНОЙ СОБСТВЕННОСТИ

RU2019665793

ГОСУДАРСТВЕННАЯ РЕГИСТРАЦИЯ ПРОГРАММЫ ДЛЯ ЭВМ

Номер регистрации (свидетельства):
2019665793
Дата регистрации: **28.11.2019**
Номер и дата поступления заявки:
2019664655 15.11.2019
Дата публикации и номер бюллетеня:
28.11.2019 Бюл. № 12
Контактные реквизиты:
+7(495) 957-77-43; oiris.patent@gmail.com

Автор(ы):
Городничев Михаил Геннадьевич (RU),
Полянцева Ксения Андреевна (RU),
Гематудинов Ринат Арифулаевич (RU),
Мосева Марина Сергеевна (RU),
Джабраилов Хизар Абубакарович (RU)

Правообладатель(и):
Ордена Трудового Красного Знамени федеральное государственное бюджетное образовательное учреждение высшего образования «Московский технический университет связи и информатики» (МТУСИ) (RU)

Название программы для ЭВМ:
Программа автоматизированного определения скорости и габаритов транспортных средств на основании изображений, получаемых со стереокамер

Реферат:
Программа предназначена для получения расстояния до движущегося объекта, его скорости и габаритов посредством изображений со стереокамер. Программа снижает общие затраты на настройку и обслуживание по сравнению с другими аналогичными системами, обеспечивает точные измерения скорости движущегося объекта. Для реализации метода используются следующие этапы: калибровка камер для устранения дисторсии, стереокалибровка, ректификация, построение карты глубины, вычисление расстояния до объекта, вычисление габаритов объекта, вычисление скорости движения объекта. Тип ЭВМ: IBM PC-совмест. ПК; ОС: Windows, Linux, MacOS.

Язык программирования: Python

Объем программы для ЭВМ: 22 Кб

Приложение Л. Акты о внедрении

«УТВЕРЖДАЮ»
Ректор ордена Трудового Красного
Знамени федерального государственного
бюджетного образовательного учреждения
высшего образования «Московский
технический университет связи и
информатики» (МТУСИ)

к.т.н., доцент Ерохин С.Д.
«12» 02 2026 г.

АКТ

об использовании результатов диссертационной работы Городничева М. Г.
на тему: «Методология создания информационной системы управления транспортными потоками» в учебном процессе кафедры «Математическая кибернетика и информационные технологии»

Комиссия в составе:

- заместителя проректора по учебной работе, Камаловой Гульназ Ильгамовны;
- доцента кафедры «Математическая кибернетика и информационные технологии», к.т.н., доц. Джабраилова Хизара Абубакаровича;

удостоверяет, что в учебном процессе кафедры «Математическая кибернетика и информационные технологии» (МКиИТ), при реализации образовательной программы «ТОП-ИИ: Инженерия систем искусственного интеллекта» (направление 09.03.01 «Информатика и вычислительная техника»), в рамках дисциплины «Технологии ИИ в интеллектуальных транспортных системах», используются результаты диссертации Городничева Михаила Геннадьевича, а именно: проведенный соискателем анализ современного состояния интеллектуальных транспортных систем, а также методология построения инфраструктуры ИТС, ориентированная на отказоустойчивость, масштабируемость и адаптируемость сбора, хранения и обработки больших объёмов разнородных данных.

Эффективность внедрения заключается в приобретении студентами знаний по перспективным направлениям развития науки и техники.

Заместитель проректора по учебной работе — Г.И. Камалова

Доцент кафедры МКиИТ — Х.А. Джабраилов

«УТВЕРЖДАЮ»
Первый проректор – проректор
по образовательной деятельности

Артемьев И.А.

«20» апреля 2026 г.

АКТ

об использовании результатов диссертационной работы Городничева Михаила Геннадьевича в учебном процессе кафедры «Инженерия и математика прикладных систем искусственного интеллекта» федерального государственного бюджетного образовательного учреждения высшего образования «Московский автомобильно-дорожный государственный технический университет (МАДИ)»

Настоящим удостоверяем, что результаты диссертационной работы Городничева Михаила Геннадьевича на тему «Методология создания информационной системы управления транспортными потоками» на соискание степени доктора технических наук по специальности 2.3.8. «Информатика и информационные процессы» были рассмотрены на заседании кафедры «Инженерия и математика прикладных систем искусственного интеллекта» от 21.04.2026 протокол №6. Научный консультант – заведующий кафедрой «Инженерия и математика прикладных систем искусственного интеллекта» доктор технических наук, кандидат физико-математических наук, доцент Яшина Марина Викторовна.

В качестве председателя заседания выступила доцент кафедры «Инженерия и математика прикладных систем искусственного интеллекта», кандидат технических наук Доткулова Анастасия Сергеевна с предложением поддержать работу и внедрить результаты исследования в учебный процесс кафедры «Инженерия и математика прикладных систем искусственного интеллекта» путем использования материалов диссертации при обучении студентов института Транспортных технологий и логистики, изучающих дисциплины «Интеллектуальные системы в моделировании транспортных потоков», «Интеллектуальные транспортные системы» (направление 01.03.04 «Прикладная математика», профили «Прикладная математика» и«Индустриальная математика и информационная безопасность») и «Управление проектированием информационных систем» (направление

09.04.01 «Информатика и вычислительная техника», профиль «Аналитика и управление распределенными данными») с 2026/27 учебного года.

Директор ИТТЛ
к.т.н., доцент С.А. Филатов

Доцент кафедры ИМПСИИ
к.т.н. А. С. Доткулова

Ученый секретарь кафедры ИМПСИИ И.А. Кутейников

Утверждаю
Генеральный директор АО "Минимакс-94"
Сергей Викторович Разживин
30 апреля 2026 г.

АКТ

о внедрении результатов диссертационной работы
Городничева Михаила Геннадьевича, представленной на соискание ученой степени доктора технических наук

Настоящим актом подтверждается, что основные результаты диссертационного исследования Городничева Михаила Геннадьевича «Методология создания информационной системы управления транспортными потоками» в настоящее время используется в деятельности АО "Минимакс-94», а именно:

- метод пространственно-временного прогнозирования макроскопических характеристик транспортного потока, применяемый для оценки и прогнозирования загрузки улично-дорожной сети;
- метод прогнозирования микроскопических характеристик транспортного потока, используемый при анализе поведения транспортных средств и моделировании дорожной ситуации;
- алгоритмы анализа транспортных данных, обеспечивающие мониторинг транспортной обстановки в режиме, близком к реальному времени;
- результаты моделирования, применяемые при поддержке принятия решений в задачах управления дорожным движением.

Внедрение указанных результатов позволило повысить точность прогнозирования транспортной ситуации, улучшить качество управления транспортными потоками и повысить оперативность реагирования на изменения дорожной обстановки.

Результаты находятся в стадии промышленной эксплуатации и используются в практической деятельности организации.

Генеральный директор С.В. Разживин

ТРАНС НАВИ СОФТ

ООО «ТранснавиСофт», 115114, г. Москва,
вн.тер.г. муниципальный округ Замоскворечье,
ул. Летниковская, д. 10, стр. 4, помещ. 1/1 (7-й этаж)
E-mail: trnoffice@trnsoft.ru

ОГРН 1157746821286
ИНН 7734362462
КПП 770501001

«19» мая 2026 г.

АКТ Внедрения
научных результатов диссертационной работы
Городничева Михаила Геннадьевича, представленной на соискание ученой степени доктора технических наук

Настоящим актом подтверждается, что основные результаты диссертационного исследования Городничева Михаила Геннадьевича «Методология создания информационной системы управления транспортными потоками» в настоящее время используется в деятельности ООО «Транснависофт», а именно:

- метод динамической коррекции коэффициентов классической модели следования за лидером, применяемый для адаптивного моделирования транспортных потоков в изменяющихся дорожных условиях;
- гибридный метод прогнозирования микроскопических характеристик транспортного потока, обеспечивающий устойчивость прогнозирования и снижение накопления ошибок при последовательном применении моделей;
- методы генерации и использования синтетических транспортных данных, применяемые при тестировании и верификации интеллектуальных транспортных систем;
- методологические решения по обеспечению воспроизводимого тестирования и оценки функционирования интеллектуальных транспортных систем в условиях ограниченности натурных данных.

Внедрение указанных результатов позволило повысить устойчивость моделирования транспортных процессов, обеспечить воспроизводимость оценки качества работы интеллектуальных транспортных систем и повысить эффективность анализа транспортных данных.

Результаты используются при исследовании, тестировании и совершенствовании интеллектуальных транспортных систем и программных средств обработки транспортной информации.

Заместитель генерального директора
ООО «Транснависофт», к.т.н. _____ Байтулаев А.М

LIMITED LIABILITY COMPANY
«RC-TECHNOLOGIES»
(Road Construction Technologies)

ООО «Эр Си Технолоджис»
119180, город Москва, ул. Большая Полянка, д. 42
стр. 1, помещ. 4/1
ИНН 7722465745, КПП 772201001,
ОГРН 1187746742842.
rc-technologies@yandex.ru
https://rc-technologies.ru/

Утверждаю
Генеральный директор ООО «Эр Си Технолоджис»
_____ К.Ю. Мамедов
_____ 2026 г.

АКТ

о внедрении результатов диссертационной работы
Городничева Михаила Геннадьевича, представленной на соискание ученой степени
доктора технических наук

Настоящим актом подтверждается, что основные результаты диссертационного исследования Городничева Михаила Геннадьевича «Методология создания информационной системы управления транспортными потоками» в настоящее время используется в деятельности ООО «Эр Си Технолоджис», а именно:

– методология построения архитектуры исполнения моделей искусственного интеллекта в интеллектуальных транспортных системах, применяемая при проектировании и эксплуатации программных сервисов;

– архитектурные решения по организации системы сбора, хранения и обработки транспортных данных;

– методы построения распределённой инфраструктуры интеллектуальных транспортных систем, обеспечивающие масштабируемость и отказоустойчивость;

– решения по распределению вычислительной нагрузки и обеспечению устойчивой работы сервисов обработки данных.

Внедрение указанных результатов позволило повысить надёжность функционирования информационных систем, обеспечить масштабируемость обработки данных и соблюдение требований к качеству сервиса.

Результаты используются при разработке и эксплуатации информационных систем и программно-аппаратных комплексов обработки транспортных данных.

Главный инженер / Технический директор
ООО «Эр Си Технолоджис» _____ Г.А. Захарьянц