

АННОТАЦИЯ

Название работы: «Разработка элементов Complaint Management System для управления информационными потоками в городской инфраструктуре».

Автор: Меркулов В. Д., гр. ПИп-1501.

Тематика работы: описание процесса разработки системы управления жалобами, поступающими в органы администрации города от его жителей.

Объектом исследования является процесс взаимодействия горожанина с органами городского управления в случае обнаружения горожанином определенной проблемы, решением которой должны заниматься соответствующие службы городской администрации. **Предметом исследования** является оптимизация описанного выше процесса с точки зрения повышения простоты понимания горожанином, какие шаги он должен совершить для достижения желаемого результата – решения обнаруженной им проблемы.

Целью данной работы является разработка информационной системы интеллектуальной обработки обращений от жителей города к органам городского управления. Основное назначение данной системы – сделать процесс взаимодействия между простым жителем города и органами городского управления максимально простым и удобным.

Работа разбита на 3 главы, описывающие 3 стадии разработки программного решения. **Первая глава** посвящена концептуальному и функциональному моделированию деятельности департаментов городской администрации в процессе решения поступающих от граждан обращений. **Во второй главе** описан этап логического проектирования разрабатываемого программного продукта с использованием объектно-ориентированного подхода. **В третьей главе** рассмотрен этап физического проектирования ПО. Работа завершается описанием разработанного программного продукта, что включает в себя диаграмму классов программного кода, а также экранные формы, демонстрирующие примеры работы системы.

Общий объем работы составляет 45 страниц, в том числе 16 рисунков, 2 таблицы и 2 приложения.

ABSTRACT

The title of the bachelor's thesis is «Development of the Complaint Management System elements to handle information flows in city infrastructure».

This bachelor's thesis deals with the process of developing an information system that manages complaints received from city residents. The object of the bachelor's thesis is the interaction process between a citizen and the city government in case the citizen has found some problem that the city government must solve. The subject of the bachelor's thesis is reengineering of this process to make it more convenient and simple for the citizen.

The aim of the work is to develop an intellectual information system able to manage citizens' complaints. This intellectual information system can make the interaction process between a citizen and the city government as convenient as possible for the citizen.

The bachelor's thesis is divided into three logically connected parts which represent three stages of the software development. We start with conceptual and functional modelling of the city government business-processes connected to complaint management. Next, we describe the software logical design process with the object-oriented approach. After that, we present the software physical design. Finally, we present the developed software by describing the software class diagram and giving some work examples.

The work is of interest to a wide circle of readers, especially for the ones engaged in software engineering and application of machine learning technologies for solving practical problems.

The bachelor's thesis consists of an explanatory note on 45 pages, including 16 figures, 2 tables, the list of 31 references including 5 foreign sources and 2 appendices.

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ.....	7
Глава 1 ФУНКЦИОНАЛЬНОЕ МОДЕЛИРОВАНИЕ ДЕЯТЕЛЬНОСТИ ГОРОДСКИХ СЛУЖБ.....	10
1.1 Анализ деятельности городских служб.....	10
1.2 Концептуальное моделирование взаимодействия горожанина и органа городского управления.....	11
1.2.1 Выбор технологии концептуального моделирования для решаемой задачи.....	11
1.2.2 Разработка и анализ модели бизнес-процесса «КАК ЕСТЬ».....	13
1.2.3 Обоснование необходимости автоматизированного варианта решения и формирование требований к новой технологии.....	13
1.3 Анализ существующих разработок на предмет соответствия сформулированным требованиям.....	14
1.3.1 Определение критериев анализа.....	14
1.3.2 Сравнительная характеристика существующих разработок.....	15
1.4 Постановка задачи на разработку программного обеспечения.....	17
1.5 Выводы по главе 1.....	20
Глава 2 ЛОГИЧЕСКОЕ ПРОЕКТИРОВАНИЕ COMPLAINT MANAGEMENT SYSTEM.....	21
2.1 Выбор используемых подходов к проектированию.....	21
2.2 Разработка логической модели программного обеспечения.....	21
2.3 Логическая модель базы данных.....	24
2.4 Выводы по главе 2.....	26
Глава 3 ФИЗИЧЕСКОЕ ПРОЕКТИРОВАНИЕ COMPLAINT MANAGEMENT SYSTEM.....	27
3.1 Используемая сетевая архитектура.....	27
3.2 Выбор используемых в процессе кодирования технологий.....	28
3.3 Выбор используемой системы управления базами данных.....	30
3.4 Разработка физической модели данных.....	32

3.5 Представление разработанного программного обеспечения	33
3.6 Принцип обеспечения модульности	36
3.7 Выводы по главе 3	38
ЗАКЛЮЧЕНИЕ	39
СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ	41
ПРИЛОЖЕНИЕ А Диаграммы последовательности	44
ПРИЛОЖЕНИЕ Б Диаграмма классов	46

ВВЕДЕНИЕ

Термин «умный город» (англ. «smart city») описывает городское пространство, в которое интегрированы современные информационные и компьютерные технологии, что позволяет использовать их для оптимизации значительной части процессов, качественного управления потоками информацией и рационального использования ресурсов. Сам термин известен с 80-х годов XX века и динамически развивался вместе с прогрессом в сфере информационных технологий, и на текущий момент в мире уже существуют примеры реализации концепции «умного города». В той или иной степени, идеи «умного» города реализованы в Амстердаме, Барселоне, Дубае, Дублине, Мадриде, Нью-Йорке и в других крупных городах.

В рамках проектов «умных городов» реализовано достаточно много различных технологий, вследствие чего дать четкое определение самому понятию достаточно сложно. Однако определены [29] четыре основных признака, которые в значительной степени определяют «умный город»:

1. Применение большого набора электронных и цифровых технологий в обществе и городах.
2. Использование информационно-компьютерных технологий (далее ИКТ) для трансформации жизни и рабочей среды в пределах региона.
3. Внедрение таких технологий в государственные системы.
4. Практика территориализации, которая объединяет ИКТ и людей, для того, чтобы повысить инновации и знания, которые они предлагают.

Вовлечение жителей города в его усовершенствование может происходить многими способами. Один из них – наличие постоянного и прямого контакта между городским населением и городской администрацией. Жители города первыми узнают о локальных городских проблемах и заинтересованы в их скорейшем решении. Кроме того, горожане заинтересованы в всестороннем развитии городского пространства. Способствовать этому они могут путем предложения различных идей и предложений. Администрация города, в свою очередь, имеет все необходимые

ресурсы и средства как для решения жалоб горожан, так и для реализации поступающих предложений.

Таким образом, можно констатировать, что в «умном» городе используются ИКТ для удовлетворения потребностей рынка (жителей города), и что наличие постоянного контакта между жителями города и его органами управления — это одно из необходимых условий успешно развития городской среды. [7]. Важнейшим условием реализации концепции «умного» города является непрерывный и структурированный диалог органов власти с гражданами и организациями бизнеса [3].

Объектом исследования является процесс взаимодействия горожанина с органами городского управления в случае обнаружения горожанином определенной проблемы, решением которой должны заниматься соответствующие службы городской администрации.

Предметом исследования является оптимизация описанного выше процесса с точки зрения повышения простоты понимания горожанином, какие шаги он должен совершить для достижения желаемого результата – решения обнаруженной им проблемы.

Целью данной работы является разработка интеллектуальной информационной системы обработки обращений от жителей города к органам городского управления. Основное назначение данной системы – сделать процесс взаимодействия между простым жителем города и органами городского управления максимально простым и удобным.

Для достижения поставленной цели требуется решение следующих **задач**:

1. Выделить основные бизнес-процессы, возникающих в ходе взаимодействия жителей города и городских органов управления, и провести их анализ.
2. Выявить основные направления работы (сферы деятельности) городских органов управления, разработать концептуальную модель.
3. Рассмотреть существующие аналогичные программные решения.

4. Сформулировать постановку задачи на разработку ПО с описанием требований.
5. Провести проектирование ПО.
6. Выбрать средства реализации ПО, разработать программный код.
7. Сформулировать выводы по проделанной работе и дальнейшим перспективам.

В первой главе дается общее описание сфер деятельности органов городского управления и хозяйства, строится концептуальное представление решаемой задачи, проводится анализ модели бизнес-процесса подачи обращения «КАК ЕСТЬ», на основании которого ставится задача на разработку ПО и происходит определение требований к нему; глава завершается проведением анализа аналогичных решений и построением модели бизнес-процесса подачи обращения «КАК ДОЛЖНО БЫТЬ».

Во второй главе описан этап проектирования разрабатываемого программного продукта: на основе построенной в первой главе концептуальной модели строится логическая модель, включающая в себя диаграммы вариантов использования, классов и последовательности. Разрабатывается логическая модель схемы базы данных, приводится ее описание.

В третьей главе рассмотрен этап физического проектирования ПО: описывается процесс выбора используемой архитектуры, а также используемых при программной реализации средств и технологий. Также приводится физическая модель схемы базы данных с учетом используемой СУБД. Глава завершается описанием разработанного программного продукта, что включает в себя диаграмму классов программного кода, а также экранные формы, демонстрирующие примеры работы системы.

Данная работа состоит из пояснительной записки на 49 страниц, включая 14 рисунков, 2 таблицы, список литературы из 31 источника и 2 приложения.

Глава 1 ФУНКЦИОНАЛЬНОЕ МОДЕЛИРОВАНИЕ ДЕЯТЕЛЬНОСТИ ГОРОДСКИХ СЛУЖБ

1.1 Анализ деятельности городских служб

Городские администрации в различных городах имеют различную структуру, которая утверждается соответствующими органами. Это связано с тем, что сферы деятельности в различных городах могут не совпадать, либо некоторые из них могут быть более значимы в рамках города. Структурными единицами городской администрации могут быть департаменты, отделы, управления и так далее, однако каждая единица несет ответственность за конкретную сферу деятельности (или хозяйства) в городе.

Чтобы избежать привязки к конкретной структуре органов городского управления, перейдем к определению сфер ответственности, существующих в рамках практически любого города. К ним можно отнести:

- дорожное хозяйство и транспортная инфраструктура;
- экология;
- общественный транспорт;
- городская инфраструктура;
- вопросы жилищно-коммунального хозяйства.

Существуют и другие сферы, для осуществления работы в которых могут создаваться структурные единицы городской администрации, однако именно с вышеперечисленными сферами чаще всего сталкиваются жители города.

Каждый отдел городской администрации может быть ответственен за одну или несколько сфер. Структурная схема в конкретном городе может быть достаточно сложной и учитывать взаимосвязи между отделами, но в общем случае такую схему можно представить в следующем виде (рисунок 1).



Рисунок 1 – Общий вид структуры органов городского управления

Схема демонстрирует, что каждый отдел работает с конкретным кругом задач и проблем. Основным источником обращений в подобные отделы являются жители города.

Рассмотрев общее описание сфер деятельности органов городского управления, можно переходить к концептуальному моделированию для создания более формализованного описания.

1.2 Концептуальное моделирование взаимодействия горожанина и органа городского управления

1.2.1 Выбор технологии концептуального моделирования для решаемой задачи

В течение работы над проектом для более формального и структурированного представления имеющейся информации используют концептуальное моделирование. В среде профессиональных разработчиков принята точка зрения, что концептуальная модель системы имеет большее значение, чем модели, создаваемые на следующих шагах проектирования (логическая, физическая). Это связано с тем, что корректно смоделированная концептуальная модель с помощью современных инструментальных средств может быть легко переведена в программную реализацию, что является

основой концепции «архитектура, управляемая моделью» (англ. Model Driven Architecture) [27].

Для создания моделей исследуемой системы существует множество подходов и базирующихся на них технологий, из них наиболее наглядными и популярными можно назвать графические нотации [6].

Графическая нотация – это набор символов и правил их взаимодействия, используемых для визуального моделирования бизнес-процесса (системы, структуры). Формализованный бизнес-процесс поддается анализу и оптимизации, поэтому графическая нотация является инструментом описания и/или моделирования бизнес-процессов [19].

К классическим графическим нотациям описания процессов можно отнести IDEF0, однако ее распространенность в настоящее время уменьшается из-за морального устаревания и определенных недостатков [1].

Более современной нотацией для описания бизнес-процессов является BPMN (англ. “Business Process Model and Notation”, нотация и модель бизнес-процессов) [8]. Главное достоинство нотации – модели, построенные с ее помощью, могут быть легко интерпретированы как техническими специалистами, так и бизнес-пользователями. Кроме того, нотация использует интуитивно понятный набор графических элементов, разделенных по категориям. В отличие от IDEF0, нотация BPMN хорошо ориентирована на описание архитектуры программного обеспечения, что является значимым преимуществом, поскольку конечная цель проекта – разработка программной реализации.

Таким образом, при концептуальном моделировании будет использоваться нотация BPMN для описания бизнес-процессов и нотация Чена для составления диаграммы «сущность-связь».

1.2.2 Разработка и анализ модели бизнес-процесса «КАК ЕСТЬ»

Основной рассматриваемый процесс – подача горожанином обращения о какой-либо ситуации. Диаграмма данного процесса в нотации BPMN представлена на рисунке 2.

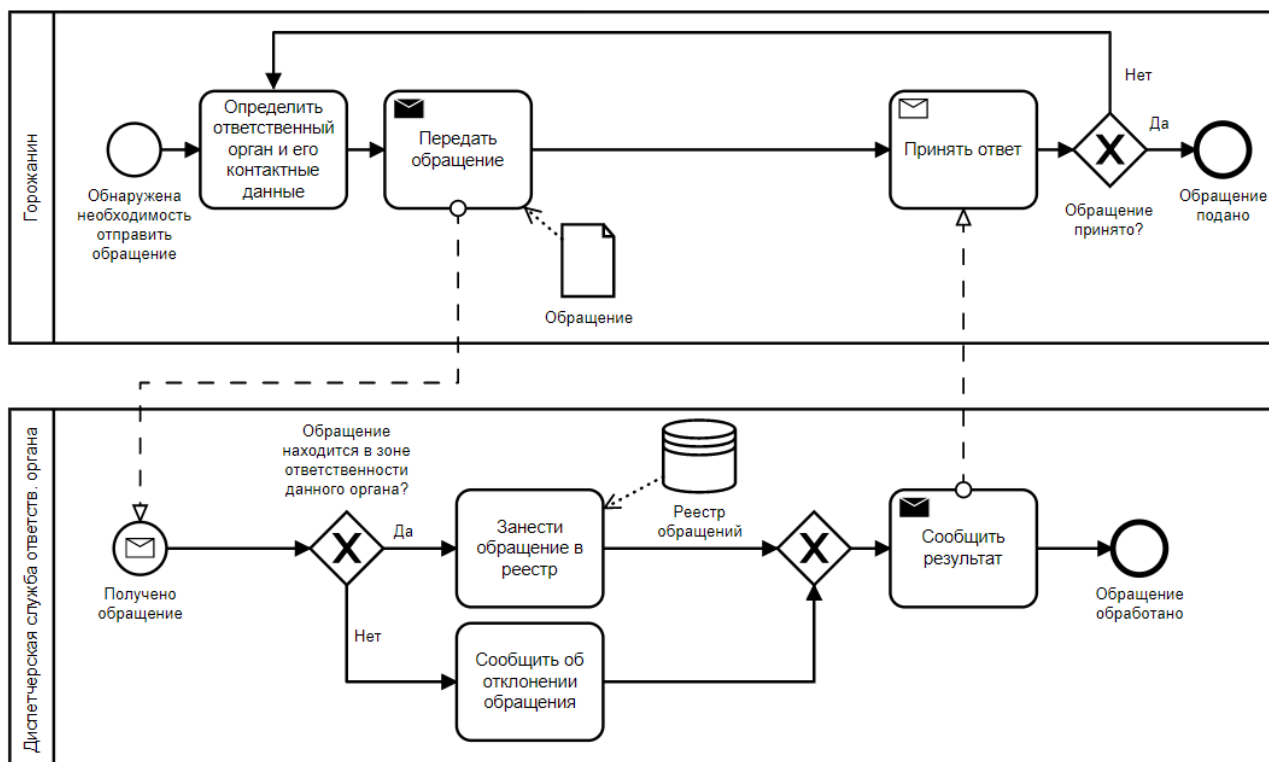


Рисунок 2 – Модель «КАК ЕСТЬ» бизнес-процесса подачи обращения

Как видно из результатов моделирование бизнес-процесса, горожанину для подачи обращения нужно совершить довольно много лишних действий, что значительно усложняет сам процесс. С другой стороны, диспетчерская служба в основном имеет лишь одну обязанность – реагирование на входящие сигналы и определение, входит ли тематика данного обращения в зону ответственности органа городской администрации.

1.2.3 Обоснование необходимости автоматизированного варианта решения и формирование требований к новой технологии

Анализ бизнес-процесса «КАК ЕСТЬ» показывает, что подача обращения традиционным методом имеет следующие недостатки:

1. Сложность определения ответственного органа – горожанин может не знать, к зоне ответственности какого органа относится его вопрос, а найти данную информацию может быть не всегда возможно.

2. Спектр задач диспетчерской службы сводится лишь к приему обращения, что является довольно простой для автоматизации задачей.

3. При использовании традиционных каналов связи (телефон) отсутствует возможность прикрепить к обращению дополнительные материалы (фото, видео и т.п.).

Для устранения данных недостатков, предлагается использовать автоматизированную систему обработки обращений от населения. При этом в основе требований к системе будет лежать автоматизация процесса определения тематики обращения, и затем – ответственного органа, что приведет к решению первых двух недостатков бизнес-процесса в существующем виде. Третий же недостаток возможно решить, добавив возможность подачи электронной формы обращения, к которой можно будет прикрепить (помимо текста) различную медиа-информацию.

Рассмотрев концептуальную модель системы и базовые требования к ней, следует проанализировать существующие разработки, реализующие схожие концепции.

1.3 Анализ существующих разработок на предмет соответствия сформулированным требованиям

1.3.1 Определение критериев анализа

В Российской Федерации сформировался класс городских информационных систем, призванных упростить взаимодействие между жителями города и городской администрацией. Основной функционал таких систем как правило включает возможность подачи обращения, отслеживание его статуса и просмотр обращений других пользователей. Зачастую можно встретить также привязку к геоданным, то есть отображение карты города с наложенным на него «слоем» объектов-обращений.

Среди значимых критериев при анализе подобных решений можно выделить:

1. Необходимость обязательной регистрации для работы с системой.
2. Автоматический процесс классификации обращения.
3. Возможность отслеживания статуса обращения.
4. Добавление мультимедийного содержимого.
5. Возможность работы с мобильного устройства (без использования браузера), то есть наличие мобильного приложения или схожего решения.

Анализу будут подвергнуты только работающие системы, применяемые в городах Российской Федерации. Иностраные аналоги не рассматриваются по причине отсутствия достоверных сведений о данных системах в открытых источниках и невозможности проверки функционала систем. Например, в городе Мумбаи (Индия) функционирует система OCMS (Online Complaint Monitoring System), однако более подробных сведений о ней в проверенных источниках не присутствует.

1.3.2 Сравнительная характеристика существующих разработок

Среди существующих были выделены системы, функционирующие в Москве, Санкт-Петербурге, Чебоксарах, Самаре и Тольятти. Сравнительная характеристика систем с учетом определенных критериев анализа приведена в таблице 1.

Таблица 1 – Характеристика существующих разработок

Наименование аналогичного решения	Требуется регистрация	Автоматическая классификация	Отслеживание статуса обращения	Добавление мультимедиа - содержимого	Работа с мобильного устройства
Наш город Москва	Да	Нет	Да	Да	Да
Открытый город Чебоксары	Да	Нет	Да	Да	Да

Наименование аналогичного решения	Требуется регистрация	Автоматическая классификация	Отслеживание статуса обращения	Добавление мультимедиа - содержимого	Работа с мобильного устройства
Открытый город Тольятти	Нет	Нет	Да	Да	Нет
Муниципальный геопортал Самары	Да	Нет	Да	Да	Да
Единый портал обращений граждан (г. Санкт-Петербург)	Нет	Нет	Да	Да	Нет

Исходя из полученных результатов, можно сделать следующие выводы:

1. Все системы реализуют базовый функционал подачи обращения – то есть позволяют прикрепить дополнительную информацию и отслеживать статус обращения. Следовательно, и в реализуемой системе данные требования должны обязательно выполняться.

2. Некоторые системы ориентированы в том числе на использование с мобильных устройств, что может говорить о востребованности подобных решений со стороны пользователей.

3. Не все системы требуют регистрации. Необязательность регистрации зачастую повышает удобство пользователя.

4. Ни одна из систем не реализует автоматической классификации обращений.

Рассмотрев существующие разработки, постановку задачи на разработку ПО следует производить на основании полученных выводов и созданной ранее концептуальной модели.

1.4 Постановка задачи на разработку программного обеспечения

Назначением (целью) разрабатываемой системы является улучшение процесса работы с поступающими от горожан обращениями путем внедрения технологии автоматической классификации текста обращения.

С точки зрения функциональных возможностей разрабатываемое программное обеспечение должно включать в себя реализацию следующих требований:

1. Предоставлять жителям города возможность подачи электронного обращения в городскую администрацию по существующим городским проблемам.

2. Фиксировать поступающие обращения путем присвоения уникального идентификатора и ведения хранилища данных.

3. Осуществлять автоматическую классификацию поступающих обращений по их тематике, для дальнейшей отправки поступившего обращения в ответственный орган городской администрации.

4. Иметь возможность отслеживания статуса поданного обращения.

5. Обращение может быть подано как с мобильного устройства, так и с помощью веб-браузера на персональном компьютере.

6. Прикрепление к обращению мультимедийного содержимого (изображения).

7. Добавление геометки при создании обращения.

С учетом определенных требований к разрабатываемому ПО можно разработать улучшенную версию приведенной ранее модели бизнес-процесса «КАК ЕСТЬ», которая будет являться основополагающим элементом концепции разрабатываемой системы. Модель бизнес-процесса подачи обращения с учетом планируемых изменений представлена на рисунке 3. Как видно из диаграммы, при использовании программного решения отсутствует необходимость в сотруднике диспетчерской службы. Разрабатываемая система будет сама принимать обращения от жителей города, после чего автоматически определять ответственный орган и отправлять обращение на его адрес.

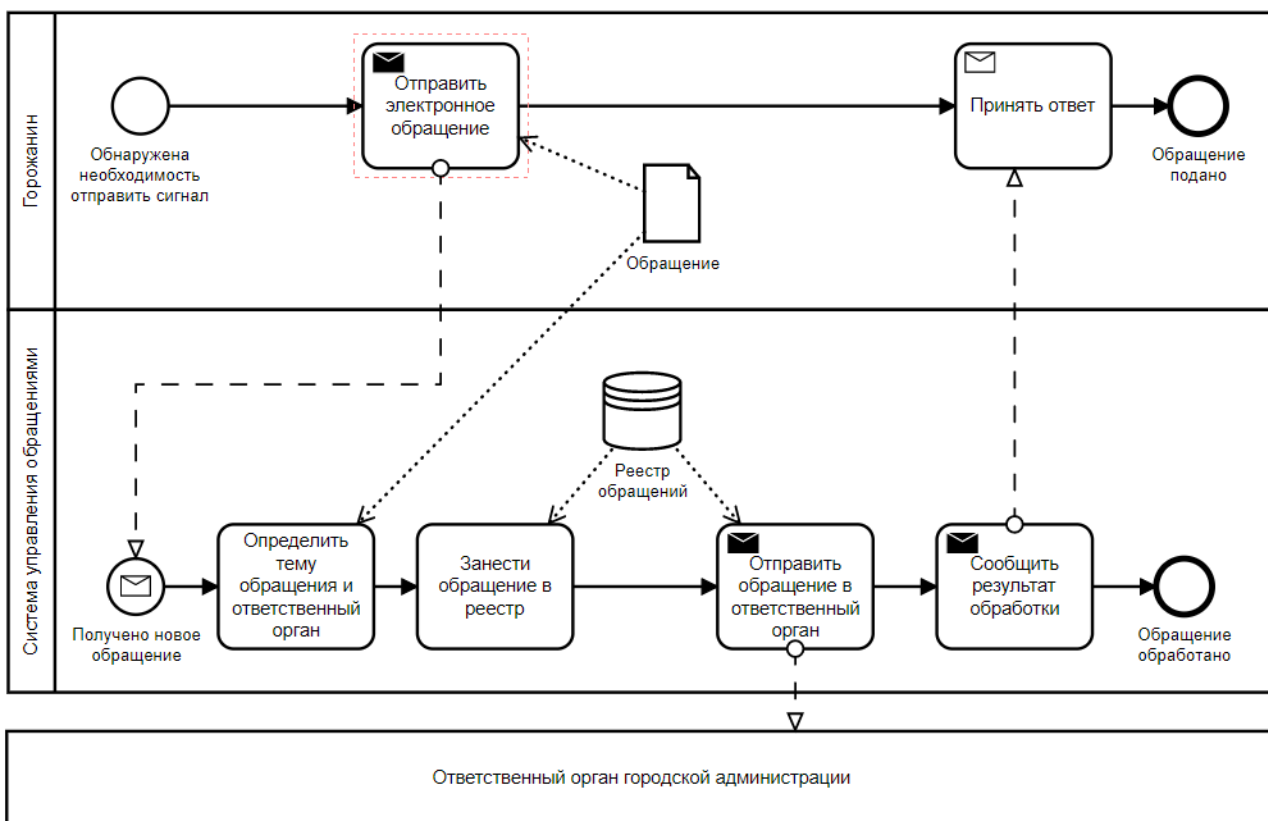


Рисунок 3 – Модель «КАК ДОЛЖНО БЫТЬ» бизнес-процесса подачи обращения

Концептуальную модель можно представить в виде диаграммы «сущность-связь» (рисунок 4).

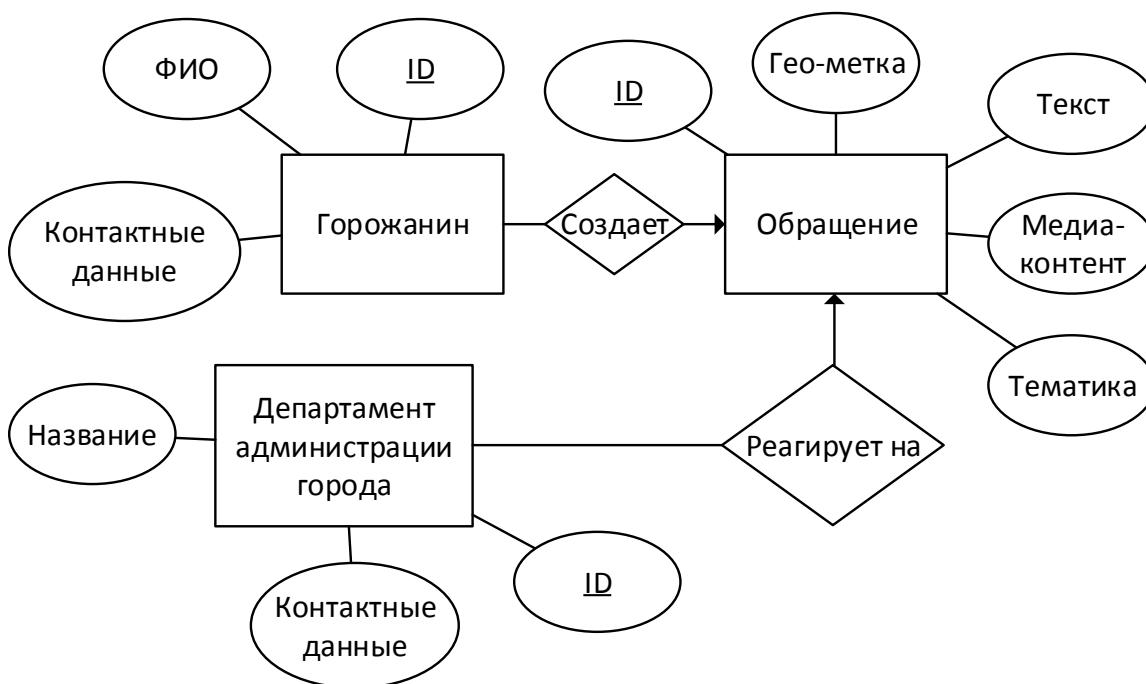


Рисунок 4 – Диаграмма «сущность-связь»

В системе существуют следующие виды сущностей:

1. Горожанин – основной пользователь системы, так как именно он является создателем обращений, без которых функционирование системы не имеет смысла. Данный концепт может быть описан несколькими атрибутами, например, ФИО и контактные данные.

2. Департамент администрации города. Данный концепт описывает структурное подразделение городской администрации, которое реагирует на поступающие сигналы жителей города.

3. Обращение – абстрактное представление поступающего в систему сигнала. Может содержать в себе различную информацию: текст, мультимедиа, географические координаты и прочее.

Используя разработанную диаграмму, в дальнейшем можно будет выделить основных пользователей системы (актеров) и разработать схему хранилища данных.

Разрабатываемая система должна быть достаточно гибкой с точки зрения использования технологии классификации текста, то есть иметь возможность настройки классов документов-обращений. Следование данному принципу позволит упростить процесс внедрения разрабатываемой системы за счет уменьшения количества необходимых изменений и настроек, большая часть которых будет сведена к изменениям элементов, которые напрямую работают с типами документов (классами).

Указанное требование можно удовлетворить, если система не будет зависеть от технологии классификации текстовых данных. Таким образом, технология классификации текста должна быть выполнена в виде заменяемого модуля. При выполнении данного условия, если в процессе внедрения системы появится необходимость изменения набора классов документов, по которым делаются обращения, потребуется только переработка технологии классификации, без значительного изменения остальных модулей системы.

1.5 Выводы по главе 1

После разработки и анализа концептуальной модели выявлена необходимость в разработке программного обеспечения, направленного на оптимизацию бизнес-процесса подачи обращения в отделы городской администрации.

На основе обзора аналогичных программных разработок, которые включают в себя городские информационные системы различных городов Российской Федерации, были выделены основные функциональные возможности систем подобного класса. С учетом созданных концептуальных моделей и проведенного анализа аналогов были сформулированы основные требования к разрабатываемому ПО, цель разработки системы, а также определены необходимые функциональные возможности и особенности работы.

В конце главы создана модель улучшенного бизнес-процесса подачи обращения «КАК ДОЛЖНО БЫТЬ» с учетом применения в нем разрабатываемого ПО.

Глава 2 ЛОГИЧЕСКОЕ ПРОЕКТИРОВАНИЕ COMPLAINT MANAGEMENT SYSTEM

2.1 Выбор используемых подходов к проектированию

Поскольку большая часть современных программных решений реализуется с использованием объектно-ориентированных языков программирования, то использование объектно-ориентированных подходов к проектированию является наиболее оправданным.

Графическая нотация стандарта UML включает в себя широкий набор диаграмм, позволяющих описать требуемые аспекты разрабатываемой системы. Набор диаграмм, используемых при разработке конкретного программного продукта, определяется разработчиком, в зависимости от требований к проекту и необходимого уровня полноты описания. Использование диаграмм нотации UML при разработке является общепринятым решением, поскольку они обеспечивают достаточно подробное описание требуемых спецификаций при сохранении простоты перевода моделей в программный код.

В данной работе для общего описания функционала продукта и существующих ролей используется диаграмма вариантов использования; для уточнения разработанной концептуальной модели и создания спецификации разрабатываемого программного продукта используется диаграмма классов; для отображения взаимодействия во времени между модулями системы используется диаграмма последовательности; также в главе 3 для описания физической модели развертывания аппаратных компонентов используется диаграмма развертывания.

2.2 Разработка логической модели программного обеспечения

Первым шагом моделирования является создание концептуальной модели программного обеспечения. Разработанная диаграмма вариантов использования, показывающая основной доступный функционал программного решения, представлена на рисунке 5.

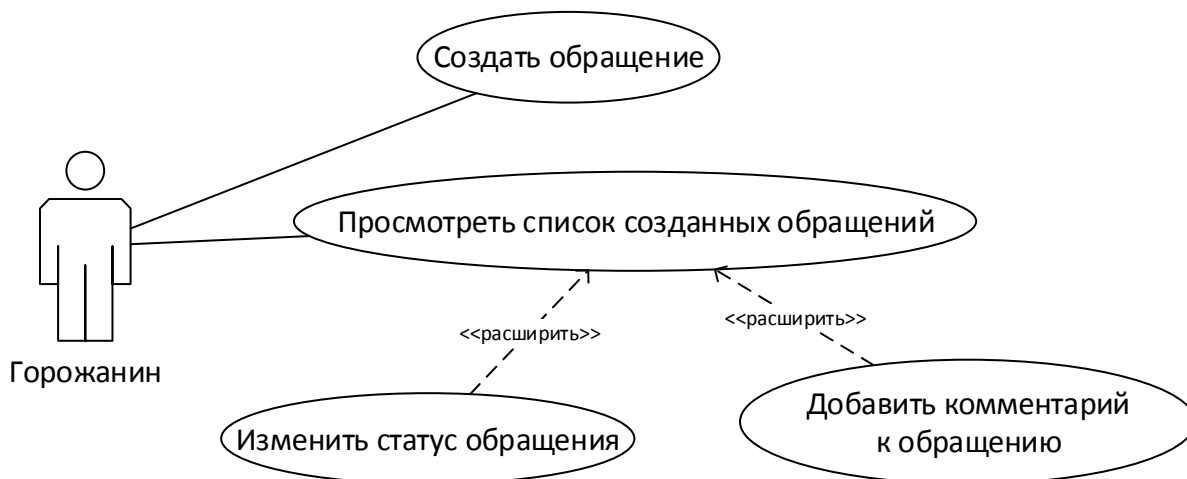


Рисунок 5 – Диаграмма вариантов использования

Горожанин – единственный выделенный вид пользователей системы – имеет возможность создавать обращения в системе и отслеживать их состояние, изменять их статус и добавлять комментарий к уже созданным обращениям. Стоит отметить, что актер Горожанин имеет доступ только к обращениям, автором которых является сам.

Следующий этап проектирования – создание диаграммы классов программного продукта на основе созданной ранее концептуальной модели в виде ER-диаграммы, что позволит уточнить концептуальную модель. Разработанная диаграмма классов представлена на рисунке 6.

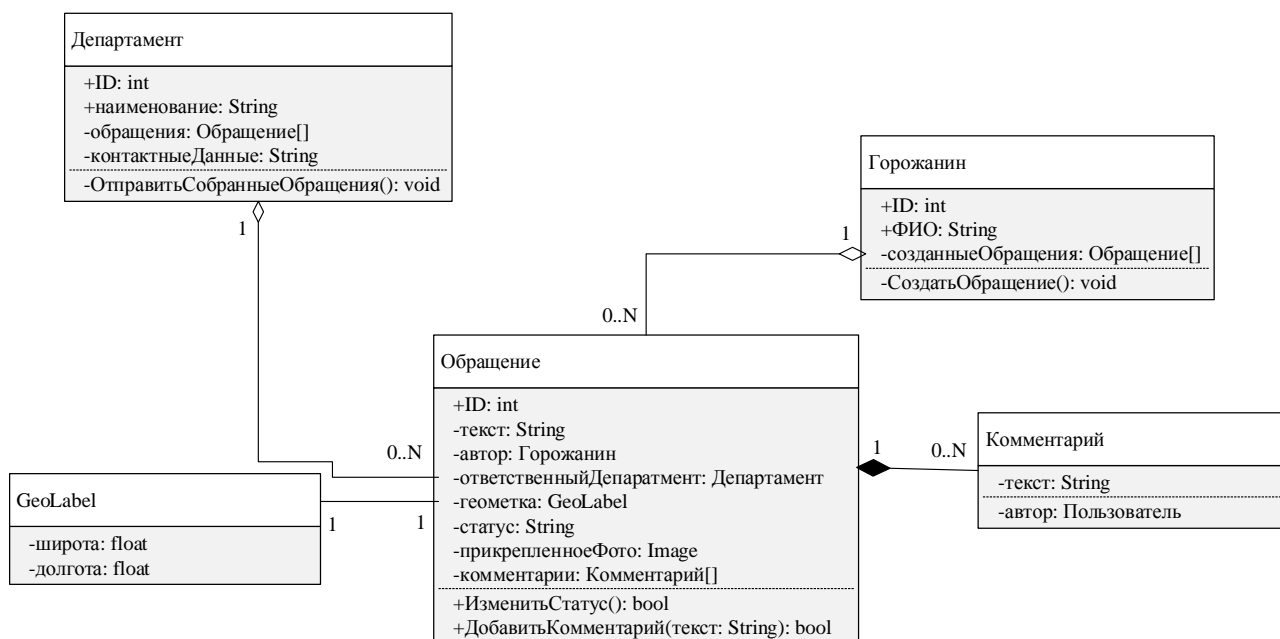


Рисунок 6 – Диаграмма классов

Классы в диаграмме классов являются более детальным описанием выявленных при построении ER-модели концептов, при этом были выделены некоторые отдельные классы, которые отсутствовали в ER-диаграмме, а также уточнены имеющиеся атрибуты, связи и добавлены методы:

- метод «Создать обращение» класса «Горожанин» позволяет Горожанину создавать новые обращения в системе;
- метод «ОтправитьСобранныеОбращения» класса «Департамент» позволяет отправить имеющиеся в системе обращения данного департамента по его контактному данным, которые могут представлять собой адрес сервера обработки данных данного департамента, либо его электронный адрес (в зависимости от имеющихся условий эксплуатации);
- ссылки на объекты класса «Обращения» имеются как у объектов класса Горожанин (что показывает созданные Горожанином обращения), так и у объектов класса «Департамент» (обращения, которые находятся в зоне ответственности данного Департамента);
- класс «GeoLabel» – географическая метка с двумя координатами (широта и долгота), добавляемая пользователем в момент создания обращения и прикрепленная к нему;
- класс «Комментарий» описывает концепт добавляемого комментария к обращению, которое возможно после создания обращения в системе.

Для представления динамического аспекта логической модели разработаны диаграммы последовательности UML для всех рассмотренных вариантов использования. На рисунке 7 приведена диаграмма последовательности для варианта использования «создать обращение». Диаграммы последовательности для остальных вариантов использования приведены в приложении А.

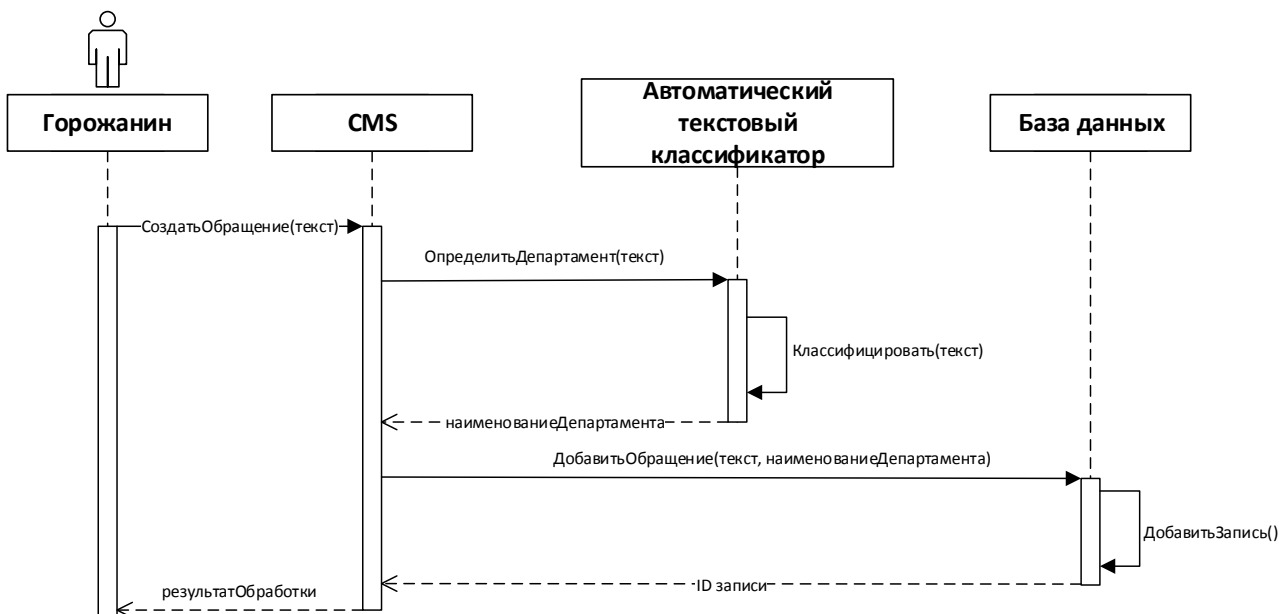


Рисунок 7 – Диаграмма последовательности для варианта использования «создать обращение»

Разработанные диаграммы показывают динамический компонент разрабатываемого решения, то есть схему их взаимодействия в ходе выполнения какого-либо процесса. Данные диаграммы будут полезны при программной реализации функциональных возможностей системы.

Следующим шагом логического проектирования должна стать разработка логической модели схемы базы данных, которая будет использоваться в программном продукте.

2.3 Логическая модель базы данных

Для использования в программном продукте выбраны реляционные базы данных, что является стандартным решением и не противоречит условиям решаемой задачи.

Разработанная логическая модель базы данных с указанием мощности связи и ее типа представлена на рисунке 8.

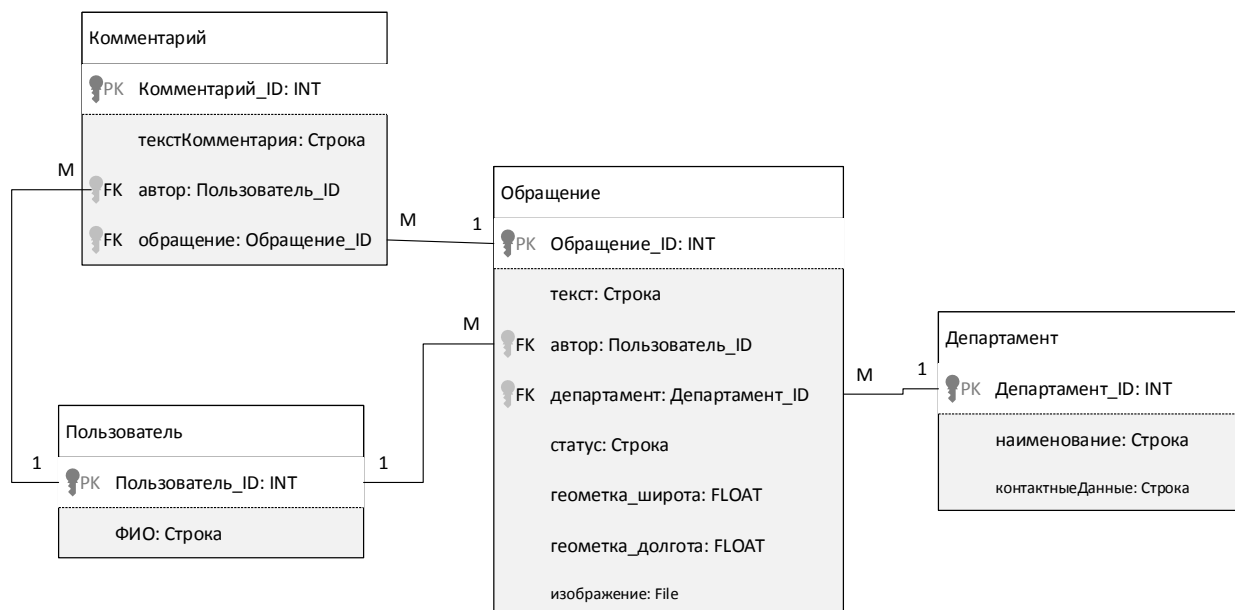


Рисунок 8 – Логическая модель данных

Спецификация связей и таблиц в схеме:

1. Таблица «Пользователь»: первичный ключ по ID.
2. Таблица «Департамент»: первичный ключ по ID; в строке «контактныеДанные» указывается электронный адрес, на который должны отправляться собранные системой обращения (адрес сервера, электронная почта и т.п.).
3. Таблица «Обращение»: первичный ключ по ID; внешний ключ по полю «Пользователь_ID» указывает на создавшего обращение пользователя; внешний ключ по полю «Департамент_ID» указывает на ответственный департамент; строки «геометка_долгота» и «геометка_широта» задают координаты географической метки (при их наличии); в строка «изображение» хранится прикрепленное в момент создания обращения изображение.
4. Таблица «Комментарий»: первичный ключ по ID; внешний ключ по полю «Пользователь_ID» указывает на создавшего комментарий пользователя; внешний ключ по полю «Обращение_ID» указывает на обращение, к которому относится комментарий.

2.4 Выводы по главе 2

В главе была создана логическая модель разрабатываемого программного продукта, позволяющая проанализировать его аспекты с различных точек зрения.

Разработанная диаграмма вариантов использования позволила выделить два основных вида пользователей системы – Горожанин и Сотрудник, а также спектр доступного им функционала.

На основе разработанной ранее концептуальной модели в виде ER-диаграммы была создана диаграмма классов, позволившая уточнить существующие объекты в системе, их атрибуты и взаимосвязи. Кроме этого, были выделены некоторые новые объекты.

Разработанные диаграммы последовательности для всех рассмотренных вариантов использования программного продукта являются опорой при разработке функциональных возможностей системы, поскольку моделируют порядок обмена сообщений между субъектами внутри системы.

В заключение этапа логического проектирования диаграмма классов была переведена в логическую модель схемы реляционной базы данных с указанием всех ключей и связей.

Разработанные диаграммы, демонстрирующие логический уровень программного обеспечения, позволят упростить написание программного кода. Схема базы данных легко может быть переведена в физическую модель, которая будет учитывать специфику конкретной используемой системы управления базами данных.

Глава 3 ФИЗИЧЕСКОЕ ПРОЕКТИРОВАНИЕ COMPLAINT MANAGEMENT SYSTEM

3.1 Используемая сетевая архитектура

При разработке системы в качестве физической модели аппаратного и программного обеспечения используется клиент-серверная архитектура.

Клиентское приложение, которое обеспечит функциональные возможности для пользователей, имеющих роль «горожанин» по диаграмме вариантов использования, не имеет строгой привязки к физической платформе и может быть реализован как для настольных компьютеров, так и для мобильных устройств. Однако выбирается вариант с использованием мобильных устройств, так как в главе 1 в процессе анализа подобных решений и требований к системе было выявлено, что мобильные устройства являются более предпочтительной платформой при имеющихся условиях.

Программное обеспечение серверной части в таком случае должно корректно обрабатывать запросы от приложений-клиентов. Непосредственно на физической машине-сервере будет развертываться модуль для приема и обработки запросов, поступающих от клиентов через сеть Интернет по выбранному транспортному протоколу (HTTP, либо TCP). Данный модуль реализует основную бизнес-логику системы. Далее, при использовании двухуровневой архитектуры «клиент-сервер», сервер баз данных будет развертываться на той же физической машине, что и модуль бизнес-логики, а при использовании трехуровневой архитектуры «клиент-сервер» - на отдельной машине. Система может быть развернута в обоих вариантах, в зависимости от отсутствия или наличия отдельной машины для использования в качестве сервера баз данных, а также качества канала связи между сервером приложения и сервером баз данных.

Использование трехуровневой архитектуры возможно при соблюдении следующих условий:

- наличие отдельной машины для ее использования в качестве сервера баз данных;

- возможность обеспечения стабильного канала связи между машиной с сервером приложения и машиной с сервером баз данных;
- повышенные требования к производительности системы в целом, что обеспечивается выделением сервера баз данных на отдельную машину.

В целом при наличии возможности соблюсти данные условия при развертывании системы в реальных условиях, разумно использовать трехуровневую архитектуру, что обеспечит лучшую масштабируемость и производительность, а также простоту дальнейшей поддержки и эксплуатации базы данных. При невыполнении данных условий, система также будет работоспособна при развертывании и сервера приложений, и сервера баз данных на одной машине.

3.2 Выбор используемых в процессе кодирования технологий

Мобильное приложение разумно реализовать для двух наиболее распространенных операционных систем Android и iOS, поскольку в совокупности мобильные устройства, работающие на данных операционных системах, составляют 97,79% [10] от рынка мобильных устройств.

Весь функционал, который должен содержаться в клиентском мобильном приложении, может быть реализован с использованием технологии создания ботов в мессенджере Telegram. Данный выбор обосновывается тем фактом, что основной функционал клиентского приложения – это отправка текстовых сообщений (обращений, комментариев) с возможностью прикрепления дополнительного содержимого (фото, геометка), а также просмотр созданных обращений и изменение статуса конкретного обращения. В связи с этим, к клиентскому приложению не могут быть предъявлены какие-либо требования, которые невозможно реализовать на основе функционала чат-ботов.

При этом реализуется основное требование к разрабатываемому продукту – удобство и простота использования сервиса для конечного пользователя (горожанина), так как для создания обращения будет достаточно открыть диалог с ботом в мессенджере и написать текст обращения.

Еще один плюс использования ботов мессенджера Telegram – отсутствие привязки к определенной платформе, как и необходимости разрабатывать отдельные приложения для каждой мобильной операционной системы. Бизнес-логика чат-бота реализуется на сервере приложений, при этом на клиентской части используется только легковесный клиент мессенджера Telegram.

Кроме этого, боты имеют возможность отправления уведомлений пользователю, которые могут содержать информацию об изменениях в его обращениях. При этом отправка уведомлений происходит через API, что облегчает написание программного кода. С другой стороны, например, задача программной реализации пуш-уведомлений в приложении на ОС Android является довольно сложной и трудоемкой [12].

Последним преимуществом данного подхода является отсутствие необходимости в разработке плана дистрибуции обновлений клиентского приложения на устройствах пользователей, так как бизнес-логика реализуется полностью на стороне сервера, и изменение в ее структуре и алгоритмах работы не требует проведения каких-либо изменений на стороне клиента.

Бизнес-логика ботов реализуется на языке Python с использованием библиотеки “python-telegram-bot”, содержащей в себе набор высокоуровневых компонентов для работы с Telegram API [13].

С учетом выбранных технологий реализации продукта, диаграмма развертывания может быть представлена следующим образом (рисунок 9). На стороне департаментов располагаются сервера с развернутым ПО для взаимодействия с системой управления жалобами. Система отправляет собранные обращения на сервер конкретного департамента, определяя нужный исходя из тематики обращения. Канал связи между системой и данными серверами, как и используемое ПО, зависит от конкретных условий эксплуатации, поскольку в различных городах имеющаяся инфраструктура будет различной.

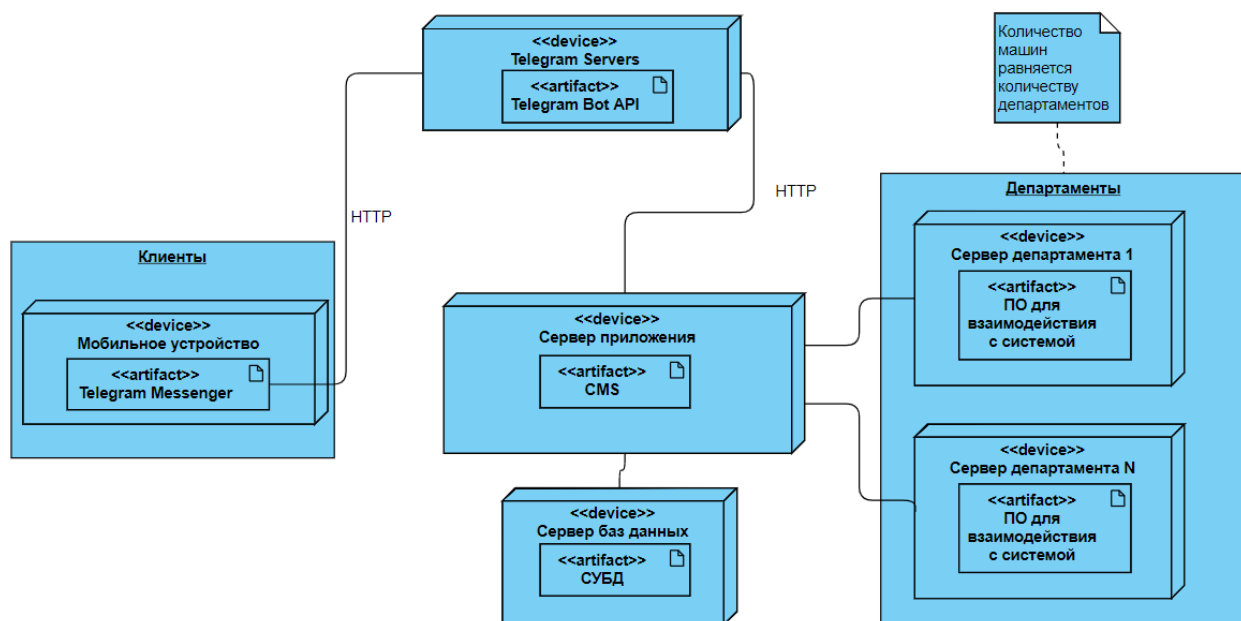


Рисунок 9 – Диаграмма развертывания для трехуровневой архитектуры «клиент-сервер» с учетом использования Telegram API

Со стороны клиентов используется мессенджер Telegram для работы с системой путем взаимодействия через интерфейс чат-бота. С сервером Telegram напрямую работает приложение мессенджера и программный код, запущенный на сервере приложений.

Стоит отметить, что при использовании двухуровневой архитектуры «клиент-сервер» диаграмма развертывания будет выглядеть схожим образом, за исключением того, что СУБД будет развернута на сервере приложений.

3.3 Выбор используемой системы управления базами данных

Для сравнения выбраны три реляционные СУБД: MySQL, MariaDB и PostgreSQL. Критерии сравнения:

1. Разработка СУБД ведется в соответствии с принципами открытого ПО. Возможные оценки: 0 – нет; 1 – частично; 2 – да.
2. Распространенность – вероятность быстро найти решение возникшей проблемы, либо какую-либо инструкцию гораздо выше, если СУБД

широко используется разработчиками. Возможные оценки: 2 – высокая; 1 – средняя; 0 – низкая.

3. Скорость работы СУБД (в среднем в режиме чтения и записи). Возможные оценки: 2 – высокая; 1 – средняя.

К СУБД не предъявляются требования по функциональности и повышенной надежности проведения транзакций из-за специфики хранимой информации – вероятность ошибки, которая может привести к потере информации, достаточно мала у любой СУБД, при этом сама информация, хранимая в базе данных, не несет значимой ценности и при этом может быть восстановлена из резервных копий.

Результаты сравнения приведены в таблице 2.

Таблица 2 – Сравнительная характеристика СУБД

Критерий СУБД	MySQL	MariaDB	PostgreSQL
Свободное ПО	1	2	2
Распространенность	2	2	1
Скорость работы	2	2	1
Итого баллов	5	6	4

MariaDB является ответвлением от MySQL, вследствие чего обладает схожими достоинствами, высокой совместимостью с MySQL и широкой распространенностью. Однако разработка данной СУБД ведется в соответствии с принципами свободного программного обеспечения, что является преимуществом перед MySQL.

По результатам сравнения для использования выбрана СУБД MariaDB, так как ее разработка полностью ведется по принципу открытого программного обеспечения, в отличие от СУБД MySQL [26], а использование СУБД PostgreSQL нерационально с точки зрения требуемой скорости работы.

В качестве программного интерфейса языка Python для работы с СУБД будет использоваться MySQL-Python – бесплатное ПО, совместимое с Python database API version 2.0 (PEP-0249).

3.4 Разработка физической модели данных

Диаграмма физической модели данных, созданной в СУБД MariaDB, представлена на рисунке 10.

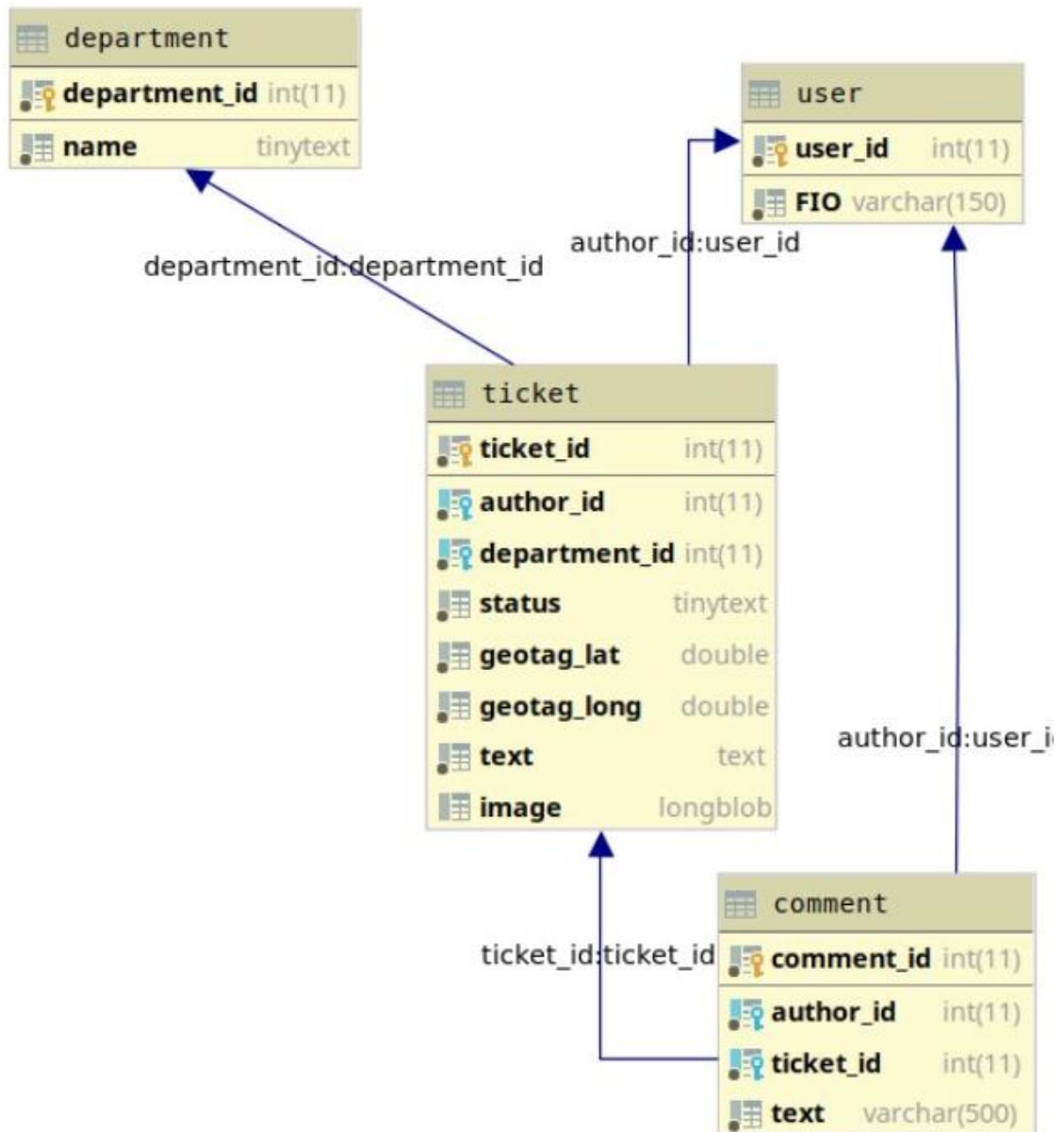


Рисунок 10 – Физическая модель данных

Спецификация таблиц и связей полностью соответствует разработанной ранее логической схеме, однако добавлено указание используемых типов данных с учетом выбранной СУБД.

3.5 Представление разработанного программного обеспечения

В рамках данной работы реализован программный код бота, обрабатывающего поступающие обращения от горожан. Диаграмма состояний бота представлена на рисунке 11.

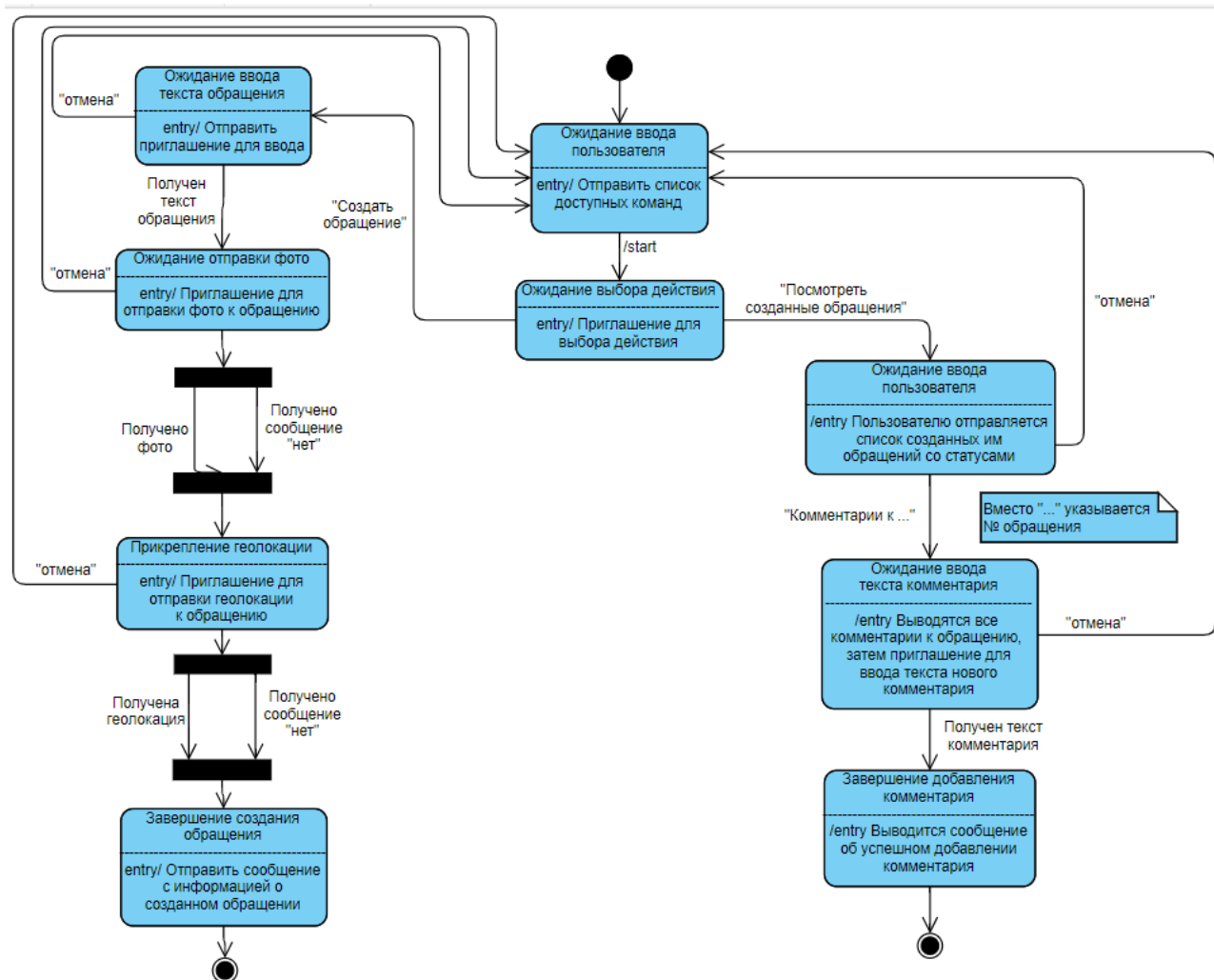


Рисунок 11 – Диаграмма состояний бота

Процесс общения пользователя с ботом осуществляется либо отправкой боту специальных команд, начинающихся со знака косой черты, либо с помощью текстовых сообщений. После ввода команды `"/start"`, пользователь получает список доступных действий, включающий в себя создание нового обращения и просмотр существующих обращений, созданных пользователем. Сообщением «отмена» пользователь может вернуться в начало диалога из любого состояния.

При выборе операции «создать обращение», бот переходит в состояние ожидания ввода текста обращения, а затем, последовательно, в состояния ожидания отправки фото и геометки. Если же пользователь хочет пропустить этап отправки фото или геометки, ему достаточно ответить «нет» на сообщение-приглашение бота отправить фото или геометку. По завершению заполнения обращения, бот сообщает ID созданного обращения и наименование ответственного департамента администрации.

При выборе варианта «посмотреть созданные обращения», бот отправляет пользователю список созданных им обращений с указанием текущих статусов. Если же пользователь хочет посмотреть комментарии к какому-то обращению, он должен отправить сообщение вида «Комментарии к ...», где вместо многоточия указывается ID обращения. В ответ на это бот отправляет комментарии к указанному обращению, а также приглашение для ввода нового комментария. При получении текста в ответ, это считается новым комментарием и добавляется к текущему обращению, при получении сообщения «отмена», происходит возврат в начальное состояние.

Примеры взаимодействия с ботом представлены на рисунках 12, 13.

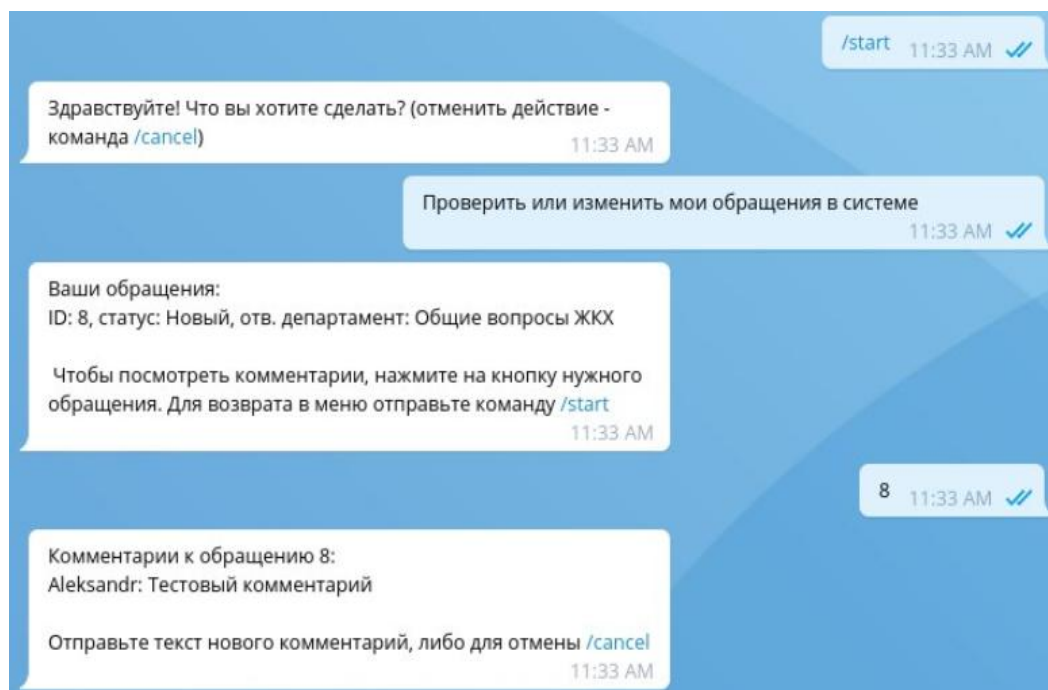


Рисунок 12 – Пример просмотра обращений пользователя и комментариев к его обращениям

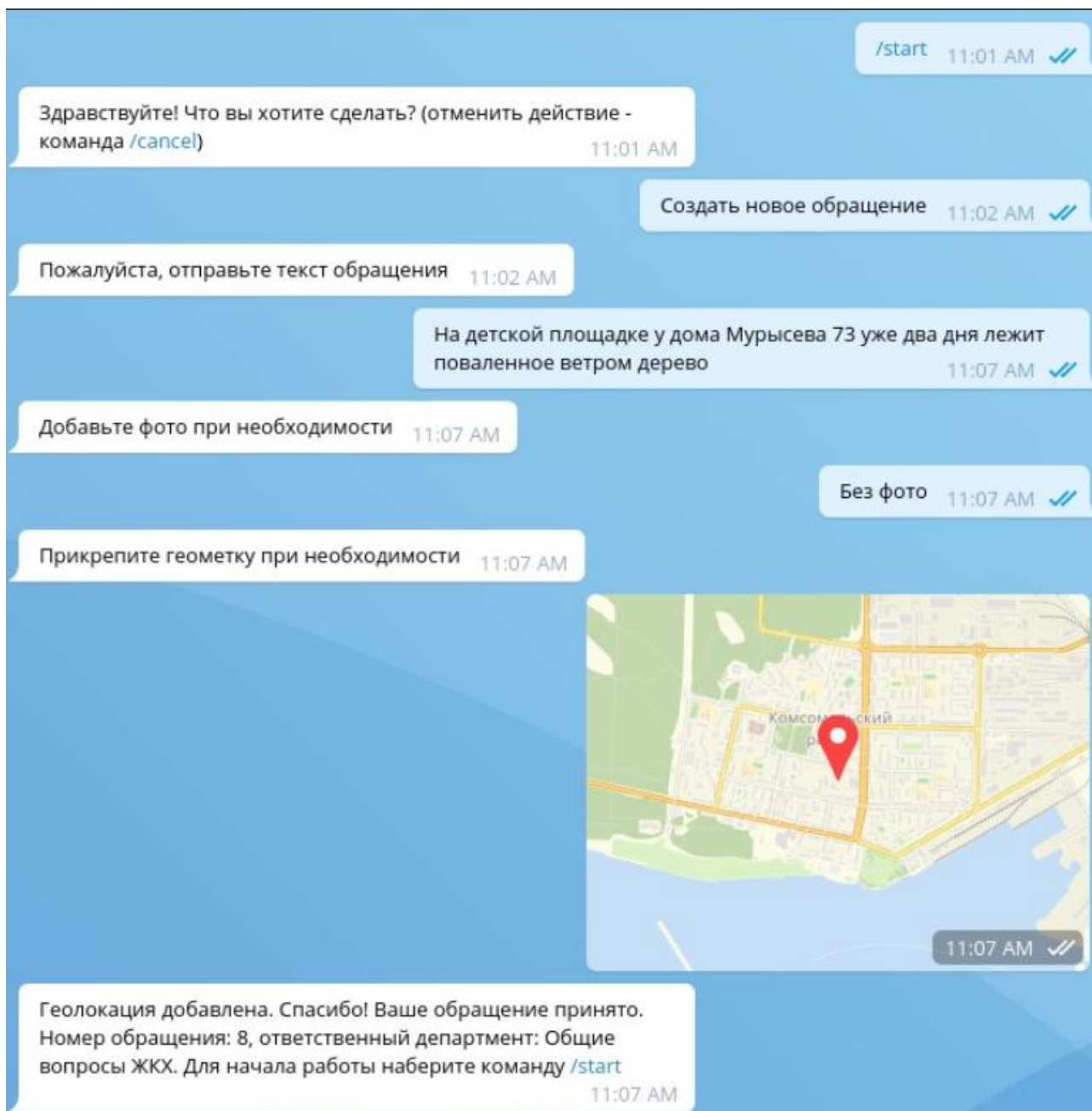


Рисунок 13 – Пример добавления обращения

Диаграмма класса разработанного программного обеспечения бота приведена в приложении Б.

Полученные системой обращения вносятся в базу данных (рисунок 14).

ticket_id	author_id	department_id	status	geotag_lat	geotag_long	text
1	5	112493866	9 Новый	0	0	Нанесена разметка «Двойная сплошная» по улице Дзержинског
2	6	112493866	9 Новый	0	0	Маршрутка 124 ходит очень редко, с интервалом 30-40 мин.
3	7	112493866	6 Новый	0	0	По адресу ул.Банькина,66, где находятся организации Росре
4	8	112493866	6 Новый	0	0	20й квартал. Не стригут траву по всему Рябиновому бульвар
5	9	112493866	8 Новый	53.485018	49.48439	С вечера 30.10 по настоящее время в районе ул.Ярославской
6	10	112493866	5 Новый	53.478075	49.366881	Комзина 2 не горит фонарь уличного освещения
7	11	112493866	6 Новый	53.477639	49.470586	на детской площадке дома Мурысева 73 уже два дня лежит по
8	12	112493866	3 Новый	53.509442	49.448082	на прилегающей территории МБУ школы №1 складированы строи

Рисунок 14 – Пример хранения обращений в базе данных

Собранные системой обращения в дальнейшем отправляются по департаментам электронным способом. На рисунке 15 представлен пример заполненной таблицы базы данных, в которую внесена имеющаяся структура департаментов в городе. Дополнив записи IP-адресами серверов департаментов, по которым будет осуществляться рассылка, и адаптировав используемое ПО, система сможет осуществлять рассылку собранных обращений.

	department_id	name
1	1	Вопросы УК-ЖЭУ
2	2	Департамент городского хозяйства
3	3	Департамент образования
4	4	Департамент общественной безопасности
5	5	Инженерная инфраструктура
6	6	Общие вопросы ЖКХ
7	7	Управление потребительского рынка
8	8	Управление экологического контроля
9	9	Департамент дорожного хозяйства и транспорта

Рисунок 15 – Пример заполнения таблицы «Департамент»

Разработанный бот удовлетворяет всем требованиям, которые были предъявлены в процессе постановки задачи на разработку ПО: реализована возможность подачи электронного обращения; прикрепления к нему фотографии и геометки; отслеживания статусов поданных обращений; с учетом того, что имеются версии мессенджера Telegram и для настольных компьютеров, обращение может быть подано как с мобильного устройства, так и с компьютера.

Помимо этого, разработанные программные решения (база данных и бот) имеют большой потенциал для внедрения в различные условия эксплуатации за счет обеспечения принципа модульности.

3.6 Принцип обеспечения модульности

Разработанное программное решение не привязано к конкретной структуре отделов администрации и может адаптироваться под различные

условия. На рисунке 16 изображена схема зависимостей системы управления жалобами в виде диаграммы компонентов.

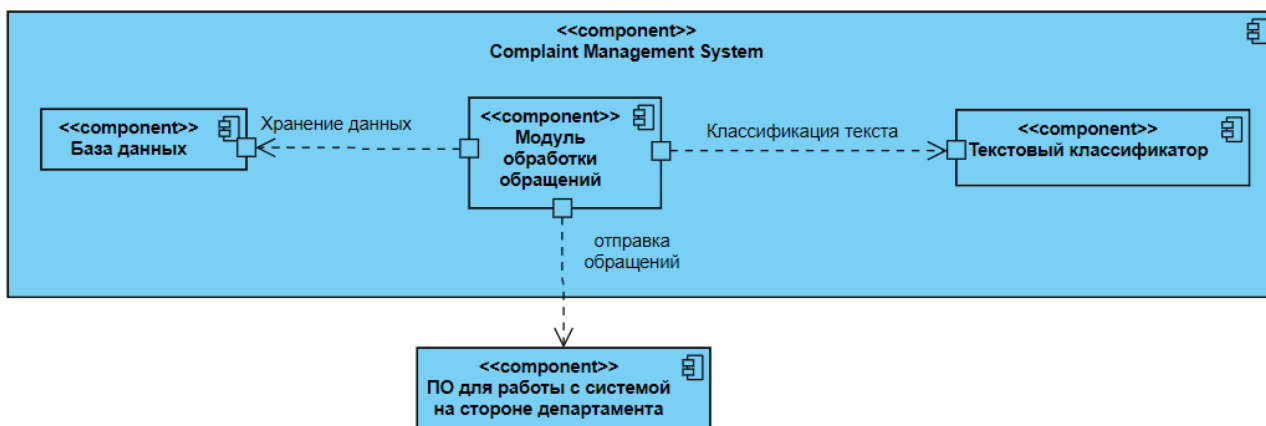


Рисунок 16 – Схема зависимостей компонентов

Модуль обработки обращений в процессе работы имеет три внешние зависимости:

1. Текстовый классификатор определяет перечень департаментов, работающих в конкретном городе, при этом его функционал – определить тему обращения и вернуть в модуль обработки обращений наименование ответственного департамента за данное обращение.

2. В базе данных должны иметься записи для обеспечения схемы данных, которая будет соответствовать перечню департаментов конкретного города.

3. Особенности реализации ПО для приема обращений из системы на стороне департаментов.

Таким образом, при развертывании разработанной системы на практике, для ее адаптации под конкретные условия эксплуатации потребуется:

1. Определить перечень департаментов и их сфер ответственности в населенном пункте.

2. Адаптировать работу текстового классификатора под имеющиеся условия.

3. Занести в базу данных сведения о департаментах.

4. Настроить инфраструктуру для отправки обращений из системы в имеющиеся департаменты.

При этом остальные элементы системы изменять не требуется. На основании этого можно выделить одно из значимых преимуществ разработанного программного продукта – возможность его применения в разных населенных пунктах, которые могут иметь отличающуюся структуру отделов администрации, о чем говорилось в первой главе. При этом на этапе внедрения программного решения потребуется выполнить только указанные выше изменения.

3.7 Выводы по главе 3

В главе был рассмотрен этап физического проектирования разрабатываемой системы, в процессе которого была выбрана используемая архитектура программного обеспечения, а также соответствующие средства реализации. При физическом проектировании была создана физическая модель схемы базы данных для выбранной СУБД, а также разработано программное обеспечение на основании созданной ранее логической модели.

Результатом проекта является созданная система, способная настраиваться под различные условия эксплуатации без существенных изменений, что достигается за счет следования принципу модульности в процессе разработки.

ЗАКЛЮЧЕНИЕ

В работе был последовательно описан процесс разработки системы управления жалобами жителей города, в процессе которого:

1. Выявлены основные направления работы (сферы деятельности) городских органов управления, на основании чего разработана концептуальная модель «сущность-связь» рассматриваемой предметной области.

2. Выделен основной бизнес-процесс, возникающих в ходе взаимодействия жителей города и городских органов управления – процесс подачи обращения жителем города. Построена модель «КАК ЕСТЬ» бизнес-процесса в нотации BPMN.

3. На основе анализа модели бизнес-процесса подачи обращения сформулирована возможность для автоматизации, заключающаяся в использовании автоматической классификации текста обращения, что позволяет ускорить проведение бизнес-процесса в целом, а также сократить затраты на персонал, деятельность которого может быть заменена автоматическим классификатором.

4. Рассмотрены существующие аналогичные программные решения – городские порталы по приему и обработке жалоб населения. Выделены особенности, заключающиеся в возможности прикрепления жителем к обращению фотографий и геометок, помимо текста.

5. С учетом планируемых изменений, вносимых в бизнес-процесс подачи обращения, построена модель «КАК ДОЛЖНО БЫТЬ» данного процесса. Сформулированы требования к разрабатываемому ПО и поставлена задача на его разработку.

6. Проведено логическое проектирование разрабатываемого программного решения, в процессе которого с учетом построенных моделей бизнес-процессов и концептуальной схемы «сущность-связь» выделены основные требования к логической структуре ПО и вариантам взаимодействия пользователей с системой.

7. На основе логической модели ПО проведен этап физического проектирования: разработана физическая схема базы данных в выбранной СУБД; диаграмма развертывания, описывающая доступные варианты для развертывания системы на практике; описаны используемые программные средства реализации, наиболее значимым из которых является использование перспективного подхода к реализации взаимодействия с пользователем через диалог с ботом.

8. Представлены результаты написания программного кода: примеры взаимодействия с ботом, а также его диаграмма состояний.

В целом можно констатировать успех достижения поставленной цели, поскольку все поставленные задачи были выполнены, а полученные решения действительно позволяют сделать процесс взаимодействия между простым жителем города и органами городского управления максимально простым и удобным.

Работа выполнена в рамках проектной деятельности по направлению «Умный город» центра «IT-Student» Тольяттинского государственного университета. Результаты, полученные в процессе выполнения работы, были представлены на конференциях «V Международная научно-практическая конференции (школа-семинар) молодых ученых «Прикладная математика и информатика: современные исследования в области естественных и технических наук» и «Студенческие Дни науки в ТГУ (2019 г.)». Представленные на конференциях материалы были высоко оценены, в результате чего получены грамоты за участие на конференциях и победу в конкурсе докладов («Студенческие Дни науки в ТГУ»), что говорит об успешной апробации работы.

Итоговая система, после проведения описанных ранее соответствующих процедур адаптации и настроек, может быть использована в структуре органов администрации населенного пункта любого уровня, от небольших до крупных городов.

СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ

Научная и методическая литература

1. Аксенова О.П. Анализ графических нотаций для имитационного моделирования бизнес-процессов предприятия / Аксенова О.П., Аксенов К.А., Антонова А.С., Смолий Е.Ф. // Современные проблемы науки и образования. – 2013. – № 4.; URL: <http://www.science-education.ru/ru/article/view?id=9745> (дата обращения: 15.03.2019).

2. Барабанова И.А. Преимущества трехуровневой архитектуры "клиент-сервер" // Молодежная наука в развитии регионов. - Пермь: Пермский национальный исследовательский политехнический университет, 2017.

3. Прокопенкова В. Особенности управления ресурсами в интеллектуальном пространстве Smart city // Недвижимость: экономика, управление. - 2017. - №3.

4. Инюшкина О.Г. Проектирование информационных систем (на примере методов структурного системного анализа). - Екатеринбург: «Форт-Диалог Исеть», 2014.

5. Максимов С.Н. «Умный город»: к вопросу о понятии и концепции // Экономические проблемы регионов и отраслевых комплексов. - 2017. - №10. - С. 117-120.

6. Матвеев А. С. Разработка рекомендаций перехода от нотации моделирования бизнес-процессов IDEF0 к нотации BPMN / Матвеев А.С., Руденко А. Ю., Прочухан В. В. // Бизнес. Образование. Право. Вестник Волгоградского института бизнеса. 2016. № 3 (36). С. 176–182.

Электронные ресурсы

7. A vision of smarter cities [Электронный ресурс] // IBM. URL: <https://www.ibm.com/services/us/gbs/bus/html/smarter-cities.html> (дата обращения: 10.03.2019).

8. BPMN Specification - Business Process Model Notation [Электронный ресурс]. URL: <http://www.bpmn.org/> (дата обращения: 20.03.2019).

9. Component Diagram Tutorial [Электронный ресурс] // Visual Paradigm. URL: <https://online.visual-paradigm.com/tutorials/component-diagram-tutorial> (дата обращения: 20.05.2019).
10. Mobile Operating System Market Share Worldwide [Электронный ресурс] // Statcounter. URL: <http://gs.statcounter.com/os-market-share/mobile/worldwide> (дата обращения: 10.05.2019).
11. MySQL-Python [Электронный ресурс]. URL: <https://pypi.org/project/MySQL-python/> (дата обращения: 25.05.2019).
12. Push уведомления в Android [Электронный ресурс] // Хабр. URL: <https://habr.com/ru/post/274169/> (дата обращения: 25.05.2019)
13. Python-telegram-bot [Электронный ресурс]. URL: <https://python-telegram-bot.org/> (дата обращения: 25.05.2019)
14. Querying a MariaDB or MySQL database using Python [Электронный ресурс]. URL: <http://support.moonpoint.com/languages/python/queryMariaDB.php> (дата обращения: 20.05.2019)
15. SQL Statements [Электронный ресурс] // MariaDB Knowledge Base. URL: <https://mariadb.com/kb/en/library/sql-statements/> (дата обращения: 20.05.2019)
16. State machine diagram tutorial [Электронный ресурс] // Visual Paradigm. URL: <https://online.visual-paradigm.com/tutorials/state-machine-diagram-tutorial/> (дата обращения: 20.05.2019)
17. The MariaDB Foundation [Электронный ресурс]. URL: <https://mariadb.org/> (дата обращения: 20.05.2019)
18. UML Class Diagram Tutorial [Электронный ресурс] // Visual Paradigm. URL: <https://www.visual-paradigm.com/guide/uml-unified-modeling-language/uml-class-diagram-tutorial> (дата обращения: 20.05.2019)
19. Графическая нотация [Электронный ресурс] // «ПитерСофт» Официальный сайт компании. URL: http://piter-soft.ru/automation/more/glossary/process/graficheskaya_notatsiya/ (дата обращения: 20.03.2019).

20. Документация Telegram: API [Электронный ресурс]. URL: <https://tigrm.ru/docs/bots/api> (дата обращения: 25.05.2019).
21. Единый портал обращений граждан Санкт-Петербург [Электронный ресурс]. URL: <https://letters.gov.spb.ru/> (дата обращения: 10.04.2019).
22. Муниципальный геопортал Самары [Электронный ресурс]. URL: <https://map.samadm.ru/> (дата обращения: 10.04.2019).
23. Наш город Москва [Электронный ресурс]. URL: <https://gorod.mos.ru/index.php> (дата обращения: 10.04.2019).
24. Открытый город – Чебоксары [Электронный ресурс]. URL: <https://www.og21.ru/home> (дата обращения: 10.04.2019).
25. Открытый город Тольятти [Электронный ресурс]. URL: <http://open.tgl.ru/icity/open> (дата обращения: 10.04.2019).
26. Сравнение MySQL vs MariaDB [Электронный ресурс] // Losst. URL: <https://losst.ru/sravnenie-mysql-vs-mariadb> (дата обращения: 20.05.2019).

Литература на иностранном языке

27. A. Olivé. Conceptual Modeling of Information Systems // Universitat Politècnica de Catalunya, - January 2007.
28. K.Bittner, I. Spence. Use Case Modeling. – Addison-Wesley Professional, 2003.
29. M. Deakin. From intelligent to smart cities // Intelligent Buildings International. - 2011. - Volume 3.
30. M. Lutz. Python. Pocket reference. - 4 edition. - Sebastopol: O'Reilly Media, 2009.
31. Sunil Kr Pandey, G.P. Singh, Dr. Vineet Kansal. Study of Object Oriented Analysis and Design Approach // Journal of Computer Science. - 2011. - №7.

ПРИЛОЖЕНИЕ А

Диаграммы последовательности

Ниже представлены разработанные диаграммы последовательности для всех выделенных вариантов использования, отображенных на use-case диаграмме, кроме варианта использования «отправить обращение».

Диаграмма последовательности для варианта использования «просмотреть список обращений» с точки зрения актера Горожанин:

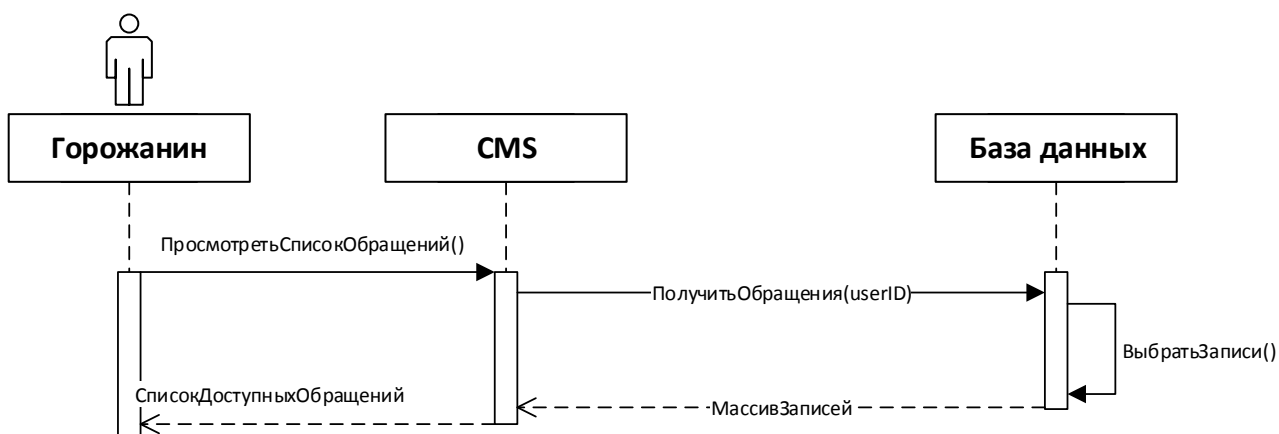


Диаграмма последовательности для варианта использования «добавить комментарий к обращению»:

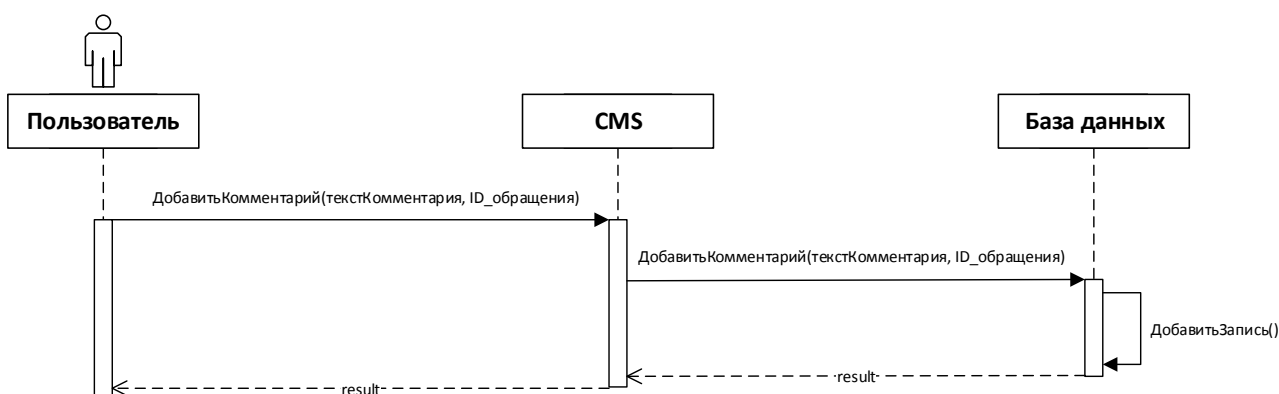
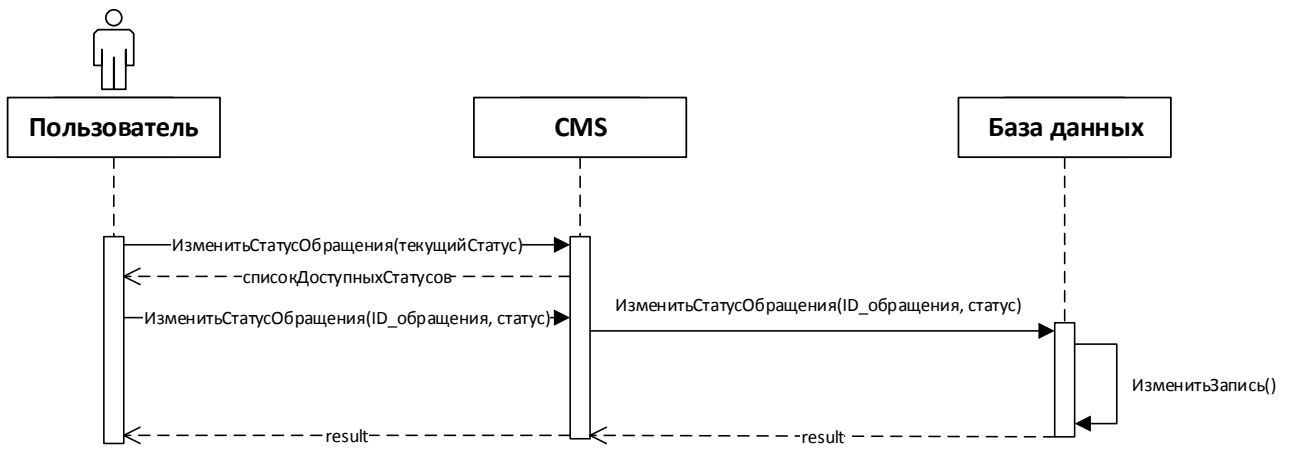


Диаграмма последовательности для варианта использования «изменить статус обращения»:



ПРИЛОЖЕНИЕ Б

Диаграмма классов

Ниже приведена диаграмма классов разработанных элементов системы управления жалобами. Класс `ComplaintClassifier`, являющийся модулем системы для классификации текста поступающих обращений, разработан вне данной выпускной работы.

