МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение
высшего образования
«Тольяттинский государственный университет»

Институт математики, физики и информационных технологий
(наименование института полностью)

Кафедра Прикладная математика и информатика
(наименование)

09.03.03 Прикладная информатика
(код и наименование направления подготовки, специальности)

Прикладная информатика в социальной сфере
(направленность (профиль)/специализация)

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА (БАКАЛАВРСКАЯ РАБОТА)

на тему «Разработка элементов CRM-системы для гаражно-строительного предприятия (на примере ГСК № 17 «Весна»)»

Студент И.Р. Хорошев
(И.О. Фамилия) (личная подпись)

Руководитель к.т.н., доцент, Э.В. Егорова
(ученая степень, звание, И.О. Фамилия)

Тольятти 2020

Оглавление

Аннотация

С. 70, рис. 25, табл. 10, лит. 24 источника

CRM- СИСТЕМА, БИЗНЕС-ПРОЦЕСС, IDEF0, DFD, БАЗА ДАННЫХ.

Тема: «Разработка элементов CRM-системы для гаражно-строительного предприятия (на примере ГСК № 17 «Весна»»

Выпускную квалификационную работу выполнил студент Тольяттинского государственного университета, института математики, физики и информационных технологий, кафедры: прикладная математика и информатика, Хорошев Иван Романович.

Объект исследования – бизнес-процесс взаимодействия с клиентами гаражно-строительной организации. Предмет исследования – частичная автоматизация бизнес-процесса обслуживания клиентов.

Целью данной работы является разработка элементов CRM-системы управления взаимоотношениями с клиентами гаражно-строительной организации.

Для достижения поставленной цели в работе необходимо решить следующие задачи:

- провести анализ литературы по проблеме автоматизации взаимоотношений с клиентами гаражно-строительного предприятия;
- изучить основные бизнес-процессы гаражно-строительного предприятия с целью выявления процессов, требующих автоматизации;
- спроектировать информационную систему, реализующие функции CRM-системы;
- произвести выбор средств реализации CRM-системы;
- разработать элементы CRM-системы выбранными средствами реализации;
- протестировать разработанную систему с целью выявления недостатков и их устранения;
- рассчитать эффективность разработки и внедрения CRM-системы.

Работа состоит из трех глав, списка используемой литературы и приложения.

Первая глава посвящена анализу предметной области, разработке концептуальной модели будущей автоматизированной системы.

Вторая глава посвящена разработке логической и физической моделей данных, выбору средств реализации системы, проектированию и реализации автоматизированной системы взаимодействия с клиентами гаражно-строительного предприятия.

В третьей главе дано экономическое обоснование эффективности разработанной автоматизированной системы.

Выпускная квалификационная работа состоит из 81 страницы, из которых приложение составляет 11 страниц. В работе: рисунков - 25, таблиц - 10, список используемой литературы – 24 источника.

Abstract

Pages. 70, figures 25, tables 10, literature 24 sources

CRM SYSTEM, BUSINESS PROCESS, IDEF0, DFD, DATABASE.

Theme: "CRM system elements development for a garage construction company (the example: GSK №. 17 "Vesna")"

The final qualifying work was done by Ivan Romanovich Khoroshev, a student of Togliatti State University, Institute of Mathematics, Physics and Information Technologies, Department of Applied Mathematics and Computer Science.

The object of research work is a business process of clients' interaction in a garage construction organization.

The subject of the research is partial automation of customer service business process.

The purpose of this work is elements' development of CRM system for managing relations with customers in a garage construction organization.

To achieve this goal, you need to solve the following tasks:

– analyze the literature on the problem of automating relations with customers in a garage construction company;

– study the main business processes of a garage construction company in order to identify processes that require automation;

– design an information system that implements CRM system functions;

– select CRM system means of implementing;

– develop CRM system elements using the selected implementation tools;

– test the developed system in order to identify shortcomings and eliminate them;

– calculate the effectiveness of CRM system development and implementation.

The work consists of three Chapters, References and Appendix.

The first Chapter is devoted to the subject of area analysis, the conceptual model development of the future automated system.

The second Chapter is devoted to logical and physical data models development, the choice of means for implementing the system and an automated system design and implementation for interacting with garage construction company customers.

The third Chapter provides an economic justification for the developed automated system effectiveness.

The final qualifying work consists of 81 pages, where Appendix has 11 pages. There are 25 figures, 10 tables and 24 sources in References in this work.

Введение

Одним из наиболее важных элементов ведения бизнеса является взаимодействие компании с клиентами. Информационная система, обеспечивающая автоматизированное взаимодействие подразделений компании с клиентами, называется CRM-системой. Если компания хорошо знает свою целевую аудиторию, потребности, может четко сформулировать стратегию дальнейшего развития, то работа с клиентами становится более эффективной.

Однако, многие компании зачастую уделяют недостаточно внимания важнейшей части процесса – установлению контакта с клиентом. Как привлечь и удержать потенциального клиента? Как сохранить информацию о каждом клиенте и обеспечить индивидуальный подход? Все эти и многие другие вопросы помогает решить CRM-система. Немаловажным является также выбрать подходящую систему для конкретного предприятия, четко сформулировать цели внедрения.

Целью данной работы является разработка элементов CRM-системы управления взаимоотношения с клиентами гаражно-строительной организации.

Для достижения поставленной цели в работе необходимо решить следующие задачи:

- провести анализ литературы по проблеме автоматизации взаимоотношений с клиентами гаражно-строительного предприятия;
- изучить основные бизнес-процессы гаражно-строительного предприятия с целью выявления процессов, требующих автоматизации;
- спроектировать информационную систему, реализующие функции CRM-системы;
- произвести выбор средств реализации CRM-системы;
- разработать элементы CRM-системы выбранными средствами реализации;

- протестировать разработанную систему с целью выявления недостатков и их устранения;
- рассчитать эффективность разработки и внедрения CRM-системы.

Работа состоит из трех глав, списка используемой литературы и приложения.

Первая глава посвящена анализу предметной области, разработке концептуальной модели будущей автоматизированной системы.

Вторая глава посвящена разработке логической и физической моделей данных, выбору средств реализации системы, проектированию и реализации автоматизированной системы взаимодействия с клиентами гаражно-строительного предприятия.

В третьей главе дано экономическое обоснование эффективности разработанной автоматизированной системы.

Выпускная квалификационная работа состоит из 81 страницы, из которых приложение составляет 11 страниц. В работе: рисунков - 25, таблиц - 10, список используемой литературы – 24 источника.

Глава 1 Функциональное моделирование гаражно-строительной организации

1.1 Технико-экономическая характеристика ГСК № 17 «Весна»

Предприятие ГСК № 17 «Весна» представляет собой основанное на членстве объединение людей и организаций, созданное для достижения общих экономических и социальных целей, связанных с удовлетворением материальных или иных потребностей членов, внесших долю (пай) в созданный для этого фонд, признающих участие в рисках и результатах организации и участвующих в её функционировании в качестве пайщиков, управляя ею демократическим путём.

Основной целью данного предприятия является удовлетворение потребностей членов организации по совместной эксплуатации здания за счет собственных средств.

Предметом деятельности заведения является:

- аккумулирование финансовых и материальных ресурсов членов организации;

- надлежащее использование и содержание земельного участка, на котором расположено здание;

- обеспечение эксплуатируемого здания электрической и тепловой энергией, водоснабжением, вентиляцией и стоками;

- проведение мероприятий по телефонизации здания, организации текущего и капитального ремонта здания, а также канализационных и вентиляционных коммуникаций;

- оказание пайщикам предприятия услуг по вызову бытовых и вентиляционных отходов;

- осуществление иной деятельности, соответствующей уставным целям организации.

На рисунке 1.1 представлена структура гаражно-строительного предприятия.

Рисунок 1.1 - Структура гаражно-строительного предприятия

Специфика предметной области предполагает выполнение системой следующих функций:

- составление сметы, контроль расходов, сравнение со сметой;
- расчет платежей;
- оформление платежных извещений, прием оплаты услуг;
- распределение полученной оплаты согласно начислениям;
- оплата услуг поставщикам услуг;
- анализ взаиморасчетов потребителей с поставщиками.

Ряд обозначенных функций может быть автоматизировано с использованием информационной системы. При реализации продуманных функций на практике должна получиться работоспособная и функциональная система с возможностью автоматизированного взаимодействия с клиентами.

Построение эффективной автоматизированной информационной системы возможно лишь при правильном подходе специалиста. Первым этапом работы является исследование и формализация бизнес-процессов деятельности организации, при разработке и проведенном анализе которых будут выявлены положительные и отрицательные стороны

функционирования предприятия в работе с его членами до внедрения информационной системы.

Таким образом, был определен структурный отдел, деятельность которого должна быть автоматизирована.

1.2 Описание бизнес-процесса управления хозяйственной деятельностью гаражно-строительного предприятия

Построение эффективной автоматизированной системы взаимодействия с клиентами возможно лишь при правильном подходе специалиста. Первым этапом проектирования является исследование и формализация бизнес-процессов деятельности организации, при разработке и проведенном анализе которых будут выявлены положительные и отрицательные стороны функционирования предприятия в работе с его собственниками, а также регистрация новых собственников до внедрения информационной системы. Для проектирования автоматизированной системы применяется несколько типов моделей, каждая из которых предназначена для удовлетворения определенных целей. Их последовательность создает логическую модель данных [9].

На следующих рисунках представлены модели «AS-IS» и «TO-BE», построенные средствами пакета Ramus. Данный программный комплекс является одним из основных инструментов бизнес-аналитиков в проектах по построению или реорганизации бизнес-процессов предприятия.

Основными возможностями Ramus являются:

- моделирование процессов (согласно методологий IDEF0 и DFD);
- разработка систем классификации и кодирования предприятия с внутренними перекрёстными связями, которая также тесно увязывается и с моделями процессов;
- формирование отчётности по моделям и системе классификации, в том числе и отчётности в форме такой регламентирующей документации как должностные инструкции и регламенты процессов;

- генерация сайта, с обеспечением доступа к данным моделей процессов, системы классификации и кодирования, а также к разнообразнейшей отчётности через веб-интерфейс.

Таким образом, выбранное средство идеально подходит для проектирования и анализа бизнес-процессов организации. С его помощью можно описать процессы с позиции их анализа для наглядного отображения слабых и сильных сторон в бизнес-процессах.

1.2.1 Моделирование бизнес-процесса «как есть»

На начальном этапе необходимо выявить бизнес-процессы, которые совершаются на регулярной основе и являются базовыми для предприятия.

Модель бизнес-процесса «как есть» описывает принципы и механизмы функционирования компании как единого механизма (рисунок 1.2). Подобная информация является основой для комплексного, системного анализа процессов, поиска проблем и путей их преодоления. Декомпозиция деятельности гаражно-строительного предприятия представлена на рис. 1.3.

Рисунок 1.2 - Бизнес-процесс деятельности ГСП(«AS-IS»)

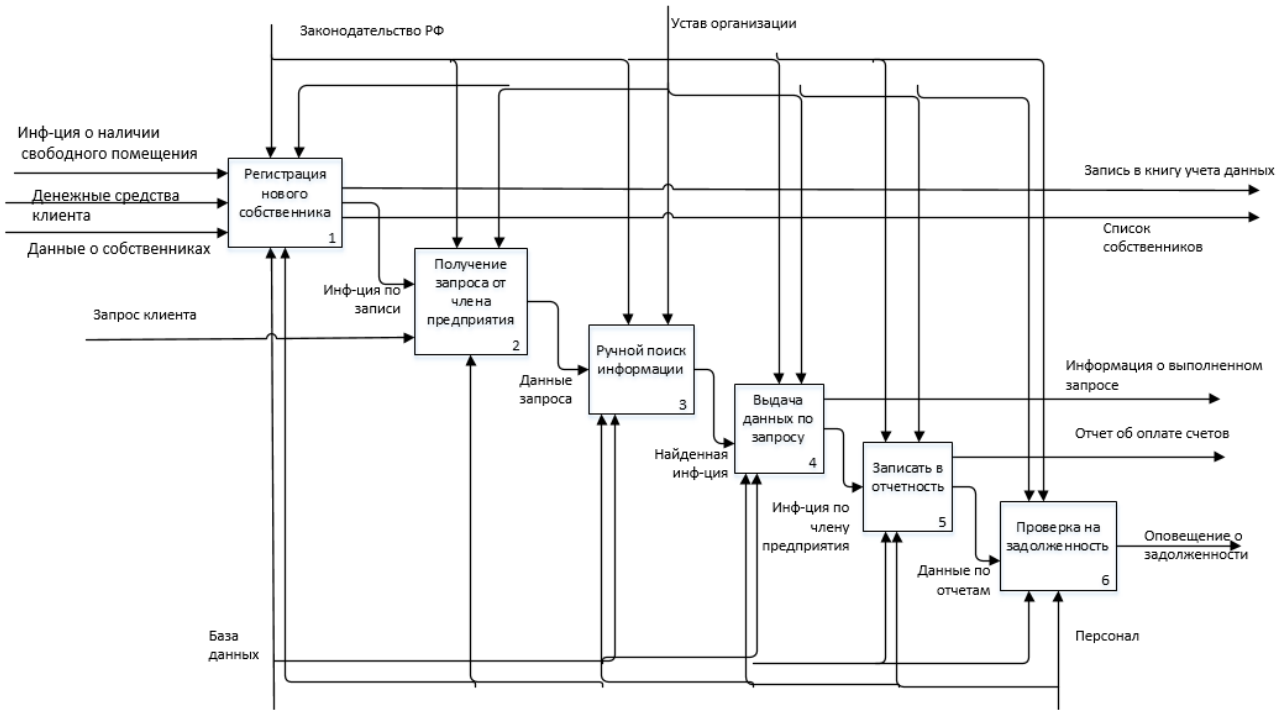

Рисунок 1.3 - Декомпозиция деятельности гаражно-строительного предприятия

Проанализировав диаграммы «как есть» можно определить, что система работает несовершенно: много труда затрачивается на ручной перебор бумаг, замедляется работа по исключению дебиторской задолженности в связи с ограниченными возможностями учета оплат, обновленные данные о собственниках имеют большую вероятность затеряться и отсутствует наглядный просмотр изменения тарифных ставок и затрат на услуги ЖКХ.

Данные недостатки можно будет решить при помощи разработки автоматизированной информационной системы управления гаражным кооперативом. В рамках данной работы будет произведена автоматизация деятельности подразделений организации, осуществляющие взаимодействие с клиентами.

Основным процессом в рассматриваемой гаражно-строительной организации является «Обработка новой заявки». Главной целью данного бизнес-процесса является регистрация нового собственника и оформление всех необходимых при этом документов и отчетности. Исполнителем будет являться менеджер по работе с клиентами. В процессе предусмотрено 2 типа выхода: «Договор заключен», «Заявка отменена»; входом будет служить новая заявка на сайте, которая требует подтверждения. Данный бизнес-процесс представлен на рисунке 1.4.

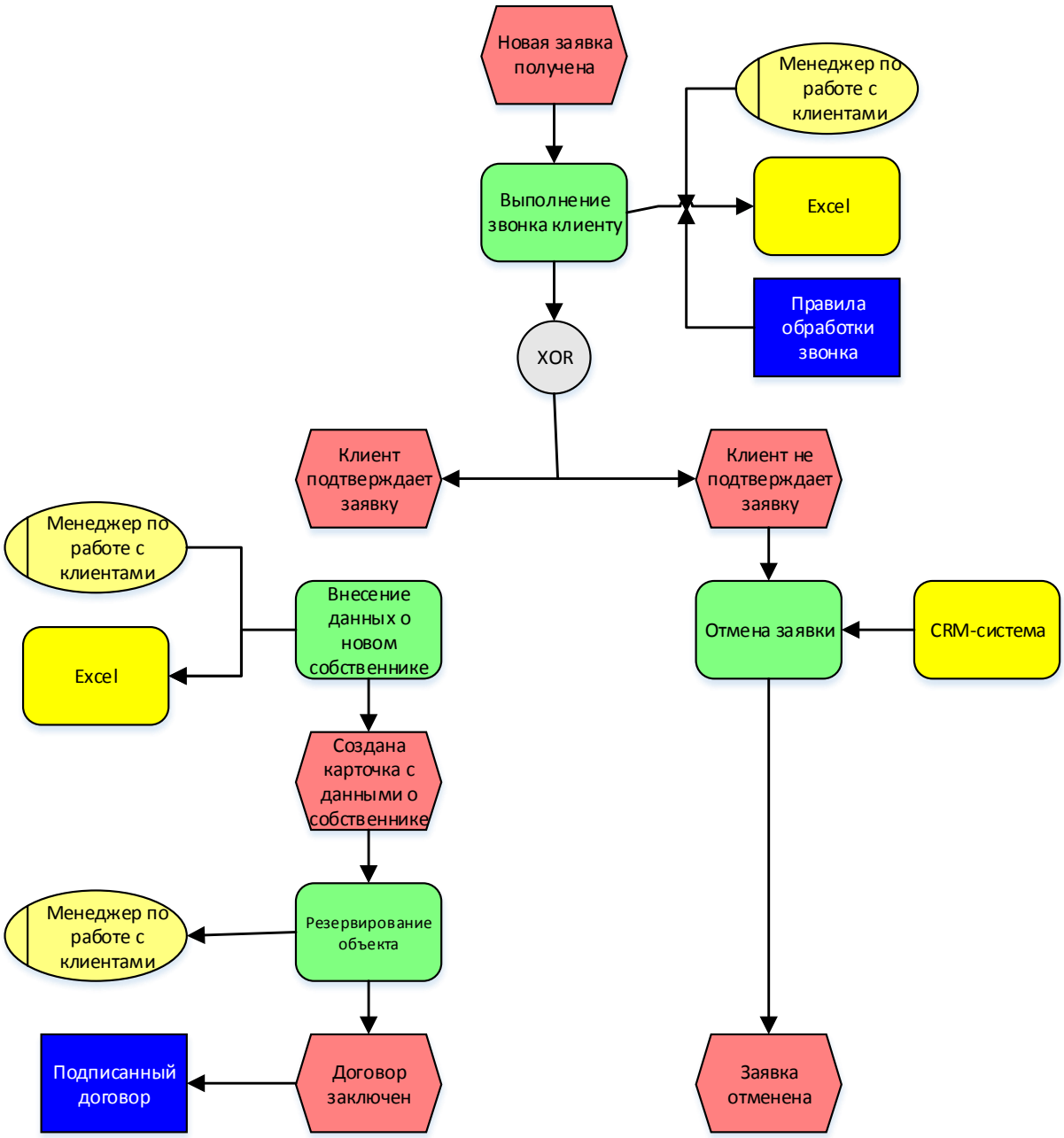

Рисунок 1.4 – Обработка новой заявки

Огромную часть работы ГСП составляет взаимодействие с клиентом посредством телефонных звонков и онлайн общение. CRM-система с интегрированной интернет-телефонией поможет не только повысить удобство взаимодействия с уже имеющимися собственниками, но и повысить эффективность формирования новых заявок и соответствующих отчетов.

Статистика новых запросов ведется в Excel по месяцам. В целом отчет в Excel сложно читаем для неподготовленного пользователя и не имеет сводной таблицы общих данных по году в целом, что усложняет процесс отслеживания общей динамики аренды и продажи помещений.

На данный момент в рассматриваемом ГСК CRM-система не используется. Как таковой клиентской базы не существует, соответственно отсутствует аналитика взаимоотношений с собственниками.

Это усложняет работу с клиентом и делает невозможным такой важный в нынешнее время личностный подход к каждому клиенту. Становится сложно отследить весь путь формирования заказа, а значит выявить возможные ошибки на каждом этапе обслуживания от регистрации клиента до непосредственно заключения договора и предоставления услуг. Соответственно, поиск, устранение этих ошибок и неполадок занимает лишнее время и ресурсы и приводит к сверхнормативным затратам. Кроме того, поиск потенциальных потребителей усложняется и конкурирующие фирмы получают преимущество.

Таким образом, были определены слабые места в процессах отдела, которые показали, что нужны новые эффективные методы и средства для того, чтобы сделать работу компании более рентабельной. Отсюда следует необходимость разработки системы «как должно быть» посредством внедрения автоматизированной системы управления взаимоотношения с клиентами.

1.2.2 Постановка задачи на проектирование автоматизированной системы

Конечным результатом исследования бизнес-процесса является выстроенная цепочка выполняемых функций работы персонала предприятия для достижения конкретной цели. В эти этапы включаются функции и результаты выполнения каждой из них. В результате исследования бизнес-процессов «как есть» становится очевидным, какие функции деятельности предприятия необходимо автоматизировать:

- оформление книги зарегистрированных собственников организации;
- ведение учета по различным показателям гаражно-строительного предприятия;
- оформление периодических отчетов;
- поиск запрашиваемой информации.

Автоматизация хозяйственной деятельности организации по работе с его собственниками позволит улучшить оперативность принятия решений, повысить производительность труда персонала, понизить суммы задолженностей посредством своевременного и оперативного вмешательства системы, вести необходимые данные о составе льготной группы среди членов предприятия и значимости проведенных работ для них, обеспечить эффективное и безопасное хранение и доступ к информации.

CRM-система предназначена для того, чтобы облегчить процессы взаимоотношений с клиентами и увеличить эффективность работы в целом, а также избежать множества сопутствующих ошибок. Анализ бизнес-процессов, определение составляющих карточки клиента, систематизация всей известной информации и создание справочников – вот далеко не полный список того, что в обязательном порядке следует подготовить во время второго этапа.

Таким образом, задачей автоматизированного решения является возможность разработки элементов CRM-системы.

1.2.3 Обоснование выбора методологии и технологии концептуального моделирования информационной системы

Основными нотациями для построения концептуальной модели являются IDEF0/IDEF3, ARIS, и UML [1].

IDEF0 – это нотация, применяемая для моделирования широкого класса систем. Результатом данного моделирования является модель системы, которая состоит из иерархии диаграмм, текста документации, которые связанны друг с другом при помощи перекрестных ссылок [2].

Нотация ARIS предполагает построение большого числа диаграмм для описания динамики и статистики. Данные диаграммы классифицируются по видам, типам, уровням и ракурсам описания [3].

Нотация UML – семейство графических нотаций, в основе которого лежит единая метамодель. Помогает в описании и проектировании программных систем, в особенности систем, которые построены с применением объектно-ориентированных технологий [4].

Результаты сравнения рассмотренных нотаций представлены в таблице 1.1.

Таблица 1.1 – Сравнительный анализ нотаций моделирования

Критерий	**Нотация IDEF0**	**Нотация ARIS**	**Нотация UML**
Легкость в изучении и понимании	Легок в освоении	Очень сложный в освоении	Сложный в освоении
Подход к проектированию	Функциональный	Процессный	Объектно-ориентированный
Области применения	Бизнес-процессы, программное обеспечение	Бизнес-процессы	Бизнес-процессы, программное обеспечение

Как видно из таблицы 1.1, наиболее подходящей является нотация IDEF0, поскольку нотации ARIS и UML достаточно сложны в освоении.

1.2.4 Разработка модели бизнес-процесса «КАК ДОЛЖНО БЫТЬ» и формулировка требований к информационной системе

Для формулировки основных требований к внедряемой ИС построим контекстную модель «КАК ДОЛЖНО БЫТЬ», которая показывает будущее предполагаемое состояние предметной области.

Модель «TO BE» (модель «как должно быть») представляет собой функциональную модель предприятия, отражающая его структуру, информационные потоки между подразделениями, внешние информационные связи. Контекстная модель показана на рисунке 1.5. На рисунке 1.6 представлена ее декомпозиция.

Рисунок 1.5 - Контекстная диаграмма автоматизации деятельности гаражно-строительного предприятия («TO-BE»)

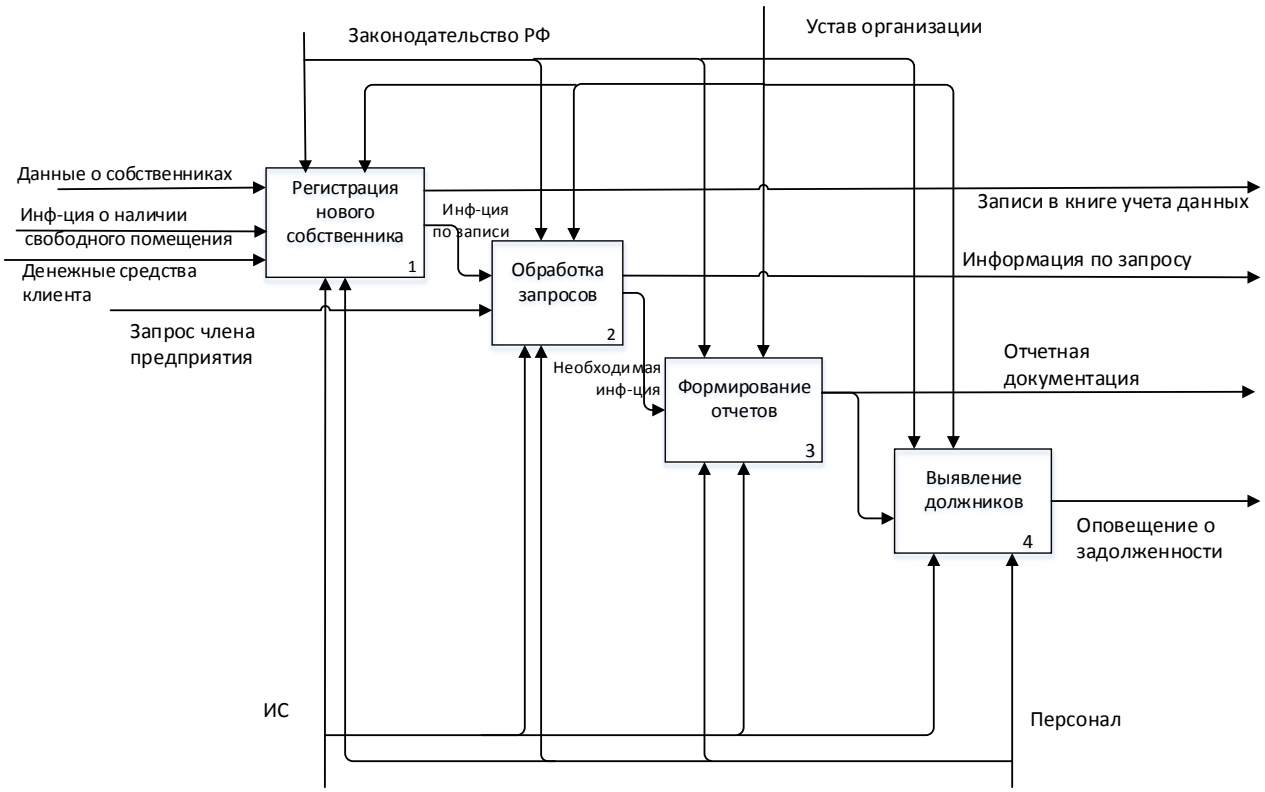

Рисунок 1.6 - Декомпозицияавтоматизации деятельности гаражно-строительного предприятия («ТО-ВЕ»)

Представленные диаграммы дают представление об эффективности реорганизации бизнес-процесса деятельности гаражно-строительного предприятия.

Исходя из рассмотренных бизнес-процессов, можно понять, что автоматизация функционирования подразделений по работе с клиентами приведет к экономии рабочего времени, затрачиваемого персоналом на обработку соответствующих документов и формирование отчетности, и исключит возможные ошибки, связанные с человеческим фактором.

На основе проведенного структурного анализа деятельности гаражного кооператива были определены основные функции, которые необходимо автоматизировать. Для более подробного процесса автоматизации с позиции использования возможных и необходимых хранилищ и движения потоков данных необходимо применить методологию графического структурного анализа DFD (DataFlowDiagrams), описывающую внешние по отношению к

системе источники и адресаты данных, логические функции, потоки данных и хранилища данных, к которым осуществляется доступ.

На рисунке 1.7 представлена диаграмма потоков данных деятельности гаражного кооператива.

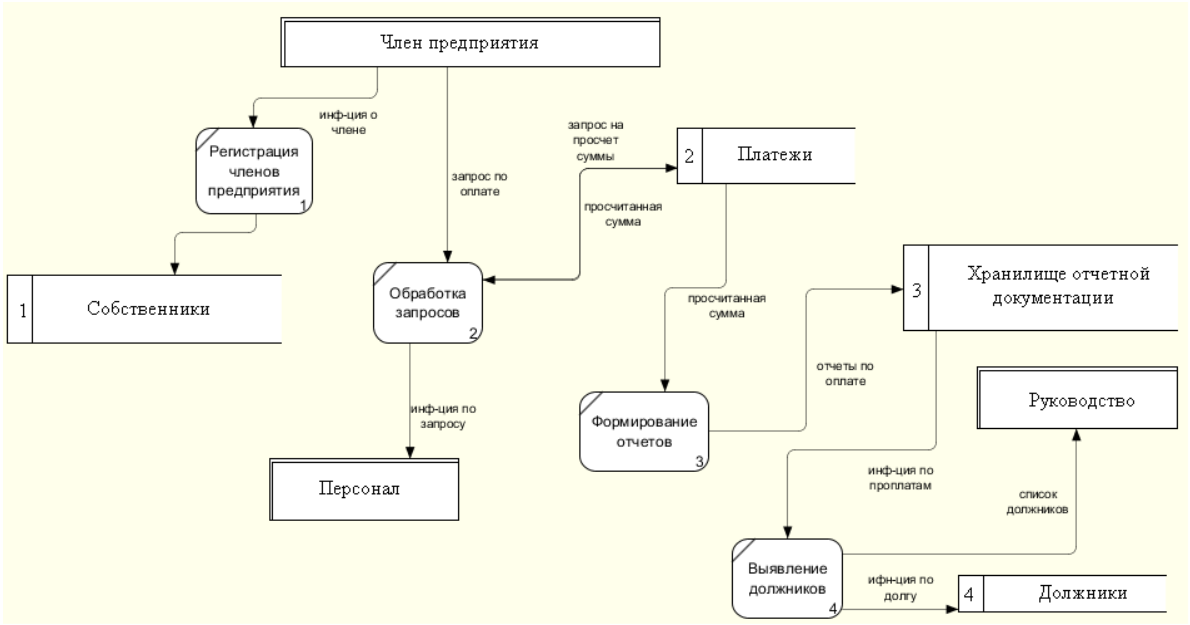

Рисунок 1.7 – Диаграмма потоков данных деятельности гаражного кооператива

Таким образом, использование диаграммы потоков данных помогло более подробно описать работу гаражного кооператива с демонстрацией основных процессов и возможных хранилищ данных, на основе которых можно реализовать концептуальную модель данных.

Одним из важнейших факторов, которые позволяют определиться с необходимостью внедрения автоматизированной системы, является правильный прогноз будущих изменений.

Произведенный анализ основных бизнес-процессов организации позволил выявить ряд основных проблем, требующих решения и предполагаемый положительный результат от внедрения CRM-системы.

1. Отсутствие сегментации клиентской базы. CRM-система предоставит возможность сегментировать потребителей по различным критериям. После внедрения: увеличится эффективность маркетинговых мероприятий и рекламных кампаний; повысится качество обслуживания

разных групп клиентов; появится грамотная коммуникация с различными сегментами клиентов.

2. Отсутствие инструмента планирования. Решение CRM-системы: предоставление статистики. После внедрения: точность прогнозов увеличится; происходит грамотное бюджетирование бизнеса; KPI менеджеров увечится.

Перечисленные выше изменения – далеко не все, что ждет организацию после внедрения CRM-системы. Также изменятся и бизнес-процессы, пусть и незначительно, но это сильно скажется на общем функционировании предприятия. Например, процесс «Поступление заявки от клиента», представленный на рисунке 1.8.

Как видно из бизнес-процесса, менеджер больше не должен вручную вносить данные о клиенте. CRM-система автоматически обрабатывает поступившие данные, записывает телефонные разговоры и помогает определить на каком этапе продажи произошел отказ клиента от заключения договора, если отказ имел место быть.

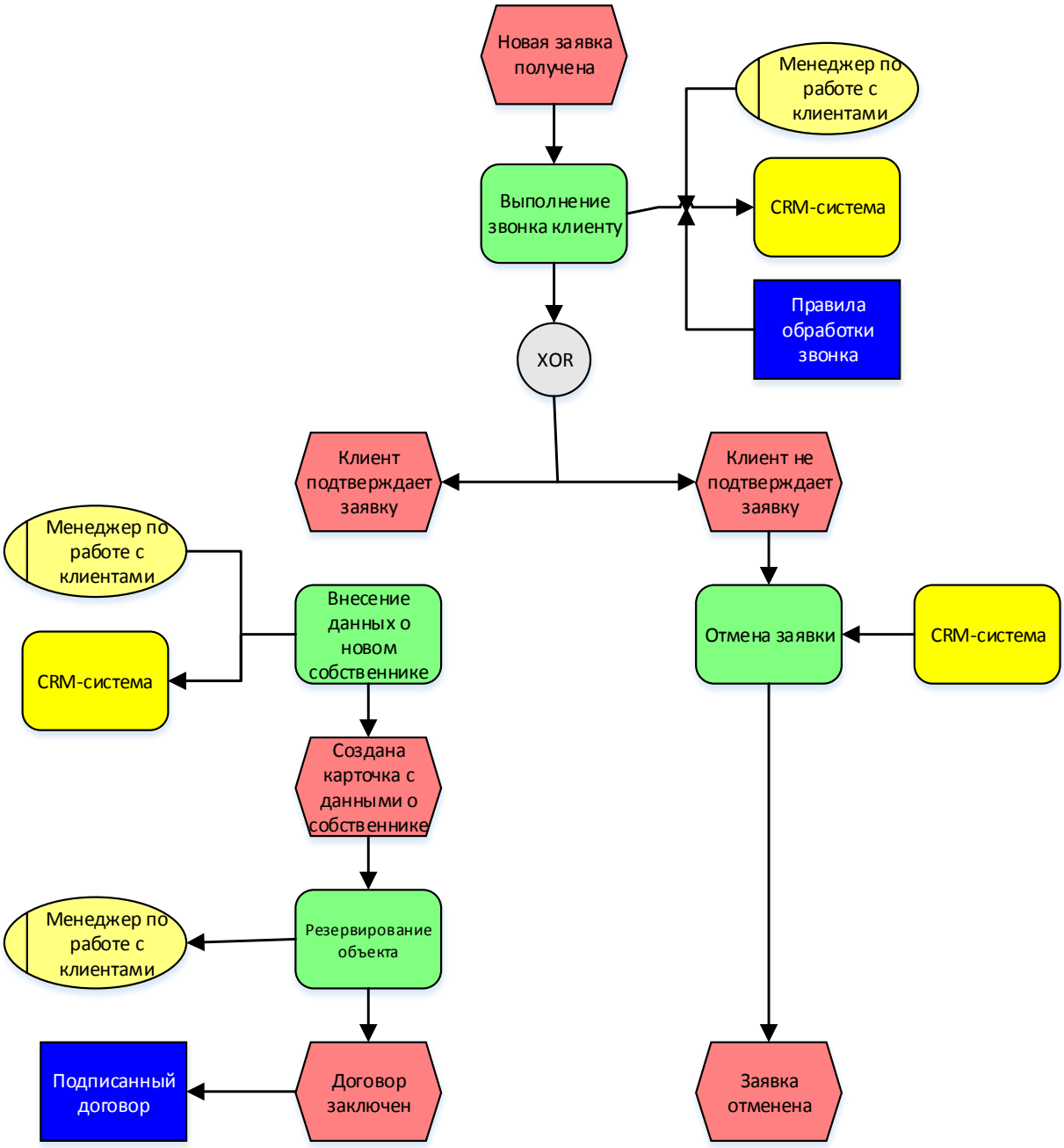

Рисунок 7 – Обработка новой заявки от клиента с использованием CRM-системы

В целом система дает возможность довести рутинные операции до автоматизма: SMS- и e-mail-рассылку, телефонные переговоры. Система помогает соблюдать регламент работы с клиентами. Все данные о клиентах и заключенных договорах хранятся в одной защищенной базе данных с разделяемым доступом: сотрудники имеют доступ к информации в

соответствии со своей ролью и полномочиями. Уменьшается вероятность искажения и фальсификации данных.

Система позволит учитывать индивидуальные особенности, предпочтения клиента, его значимость для организации. Четко видно, как осуществляется оформление нового собственника (по этапам). Каждый клиент взаимодействует с менеджером, который несет персональную ответственность за результаты работы, вместе с тем руководство в любую минуту (даже в отсутствие менеджера) может ознакомиться с положением дел.

Исходя из возможностей компании, необходимо определиться с тем, чего ожидать от внедрения стратегии CRM:

- создание базы данных о контрагентах: сюда включается занесение всех тех, с кем идет сотрудничество на постоянной и временной основе: поставщики, партнеры и сами клиенты.Во-первых, вся необходимая информация будет собрана в одном месте, во-вторых, экономится время на процессе ее поиске, а, в-третьих, новые сотрудники будут обладать собранными данными и смогут сразу приступить к работе;
- сохранение истории совершенных сделок. Благодаря этой функции CRM есть возможность проанализировать деятельность менеджеров по работе с клиентами, а также посчитать какие преимущества принесло компании сотрудничество с тем или иным клиентом;
- планирование дальнейшей работы. При решении использовать в своем бизнесе CRM появляется возможность рационально планировать время на проведение встреч и переговоров;
- разработка программ лояльности для постоянной клиентуры. Имея на руках подробную базу, составленную специально для организации CRM, можно выявить тех клиентов, которые могут вот-вот отказаться от сотрудничества, а также тех, которые уже на протяжении долгого времени остаются верными работе с одной компанией.

По итогам описанной проблемы, внедрение CRM-системы необходимо. Но для того, чтобы перейти к анализу рынка готовых решений, следует конкретизировать задачи, которые эта система будет выполнять:

1. Управление контактами и клиентской базой. Это одна из главных задач, которая должна быть решена в первую очередь. Структурированная база, простой ввод новой информации и быстрый доступ к ней любого отдела – это основные пункты, которые должны быть реализованы.

2. Управление бизнес-процессами. Процессный подход в реализации решения, анализ и возможность редактировать бизнес-процесс в любой момент времени.

3. Единые стандарты работы. Каждый сотрудник компании может следовать общему плану работы, что экономит время и облегчает взаимоотношения между работниками.

4. Анализ продаж. Четкая статистика и контроль каждого этапа продаж поможет выявить ошибки и устранить их.

5. Интеграция с почтой, IP-телефонией и SMS-рассылкой. Полная история общения с каждым клиентом облегчает реализовать личностный подход.

Таким образом, были выделены основные процессы проектируемого решения, которые показали, как может измениться деятельность компании с использованием информационной системы. Внедряемая система должна обеспечить максимально оперативный доступ к информации о связанных объектах учета разных блоков; обеспечить максимальную степень достоверности и полноты информации; обеспечение быстрого формирования отчетов и печатных форм по всей совокупности сведений, содержащихся в базе данных; обеспечение контроля доступа к информации. Кроме того, она должна показать себя как инструмент лояльности при взаимодействии с клиентами компании, как механизм, с помощью которого можно увеличить число потенциальных клиентов.

1.3. Анализ существующих автоматизированных информационных систем и их основные функции

При анализе предметной области были рассмотрены две аналогичные системы:

- «Автоматизированная информационная система жилищно-коммунального хозяйства» (АИС ЖКХ);
- «Автоматизированная Информационная Система Бюро технической инвентаризации» (АИС БТИ).

Конфигурация "АИС ЖКХ" предназначена для повышения эффективности работы предприятий, осуществляющих начисления за квартирную плату и коммунальные услуги, и прием платежей для любой формы собственности жилья [9].

Конфигурация одинаково применима для Единых расчетных центров, для абонентских отделов поставщиков услуг (водоканалы, тепловые сети, спецавтохозяйства и т.д.), на предприятиях жилищно-коммунального хозяйства (ЖЭК, ДЭЗ, ТСЖ).

В конфигурации заложены следующие возможности:

- поддержка лицевых счетов в актуальном состоянии по составу проживающих и количеству предоставленных услуг в разрезе поставщиков услуг;
- учет объема и номенклатуры предоставляемых услуг и расчет их стоимости (тарификация услуг);
- расчет услуг на основании показаний любых приборов учета (индивидуальных, коллективных, кустовых, учета расхода тепловой энергии);
- расчет услуги по нескольким индивидуальным приборам учета, принадлежащих одному лицевому счету, с поддержанием технических характеристик каждого;
- учет льготных категорий граждан в соответствии с федеральным и муниципальным законодательством как для снижения оплат за жилищно-

коммунальные услуги, так и для предоставления адресных компенсационных выплат;

- оповещение владельцев жилья о текущих начислениях и долговых обязательствах перед поставщиками услуг;
- прием платежей за жилищно-коммунальные услуги по квитанциям со штрих-кодом или без него, регистрация приема денег с помощью фискального регистратора. Для приема платежей для отдаленных пунктов применима работа в режиме распределенных баз данных;
- расчет пени при несвоевременных платежах за жилищно-коммунальные услуги;
- учет услуг в разрезе поставщиков и расщепление платежей по поставщикам за любой период времени (от 1 дня до месяца);
- учет субсидий для малоимущих слоев населения;
- перерасчет начисленных сумм за жилищно-коммунальные услуги при изменении нормативно-справочной информации за любой промежуток времени;
- получение разнообразных отчетов в разрезе адресов и обслуживающих организаций; получение диаграмм для сравнительного анализа.

Конфигурация позволяет оперативно производить настройку расчетов начислений за квартплату и коммунальные услуги любой сложности без изменения программного обеспечения, подключать правила расчета для льготных категорий граждан, опираясь на Жилищный кодекс РФ, рекомендации института жилищно-коммунального хозяйства, федеральное и муниципальное законодательства.

Выполнение функций, заложенных в конфигурации, закреплено за документами, отражающими реальную работу предприятия. Конфигурация "АИС ЖКХ" является оригинальным решением для "1С:Предприятия 8.0". Продукт не является самостоятельной программой и предназначен для

совместного использования с системой "1С:Предприятия 8.0" локальной, сетевой и SQL-версий.

В архиве «АИС БТИ» содержатся правовые и техническая информация о каждом объекте недвижимости - будь то жилой дом, дача или гаражный кооператив. Для каждого объекта недвижимости формируется пакет документов, в котором отражена вся его история и оценка его стоимости. При получении этой оценки учитывается все: размеры и этажность дома, из чего сделаны его стены и крыша, и насколько эта крыша прохудилась, есть ли удобства и где они расположены.

АИС БТИ создана в архитектуре клиент-сервер. Сервер обрабатывает запросы, поступающие от набора клиентов - автоматизированных рабочих мест (АРМ). Набор рабочих мест отражает основные направления деятельности БТИ. В него входят автоматизированные рабочие места техника, правовика, архивариуса, администратора. Набор может быть расширен и дополнен добавлением в систему новых “клиентов”. Современные аппаратные средства позволяют обрабатывать запросы в масштабе реального времени [9].

Наличие в АИС развитого функционала настроек, различных справочников, мощного инструментария АРМ Администратора позволяет внедрить и адаптировать Систему для работы практически в любое БТИ России. Главным отличием АИС от других подобных информационных систем, существующих на рынке, является развитая поддержка работы с правоустанавливающими документами, легкая адаптируемость, развиваемость и простота использования.

Возможности системы:

- комплексная автоматизация деятельности БТИ;
- информационная поддержка всех основных технологических процессов в БТИ;

- оперирование данными о физических и юридических лицах, в том числе паспортными данными физ. лиц, регистрационными данными юр. лиц, данными о прежних фамилиях и названиях всех владельцев;
- оперирование данными о всех правовых документах на недвижимость, включая быстрый поиск любого правового документа по указанному адресу, или номеру реестровой записи, или фамилии (названию организации) владельца. При этом возможен поиск по адресу со старым названием улицы, по старой фамилии владельца, поиск с указанием неполных данных;
- решение задач регистрации документов приватизации. При этом исключается возможность повторной приватизации жилья одним лицом.

Реализованы также возможности:

- расторжения договора приватизации;
- оперирование всей адресной информацией об объекте недвижимости (район, улица, номер, префикс, корпус);
- оперирование всей технической информацией об объекте недвижимости;
- получение оценки инвентаризационной стоимости объекта недвижимости без учета и с учетом износа, а так же с учетом коэффициента пересчета на указанный год;
- надежную сохранность данных не только при возникновении сбоев аппаратуры, но также их защиту от несанкционированного доступа. При любом изменении информации система фиксирует и хранит сведения об операторе, внесшем изменение, и дате этого изменения.

Исходя из перечисленных функций, была построена таблица сравнительного анализа подобных систем, оценивающая по трехбалльной системе (таблица 1.2).

Таблица 1.2 - Оценивание систем по трехбалльной шкале

Критерии	**«АИС ЖКХ»**	**«АИС БТИ»**
Удобство интерфейса	3	1

Критерии	**«АИС ЖКХ»**	**«АИС БТИ»**
Формирование отчетов	2	2
Безопасность	2	2
Доступность(цена)	1	1
Расширенный поиск	2	2
Удобство в эксплуатации	2	2
Итого (max=18)	12	10

Таким образом, из таблицы видно, что ни одна из проанализированных систем не набрала максимальное количество баллов по перечисленным функциям, что свидетельствует о существовании недостатков, и в разрабатываемой системе необходимо устранить их по максимуму. После рассмотрения выбранных систем были отобраны необходимые модули для автоматизации хозяйственной деятельности гаражно-строительного предприятия, что сделает систему удобной в эксплуатации и доступной по цене, по сравнению с рассмотренными программными продуктами. В системе необходимо учитывать все стороны работы персонала, а также необходимо учитывать все требования заказчика предприятия.

Вывод по главе: в ходе выполнения первой главы были построены диаграммы «бизнес-процессов» предприятия, выявлены недостатки в работе и предложены подходы к их устранению. Изучено функционирование систем-аналогов и приведена их сравнительная характеристика по выбранным критериям. Проведенный анализ показал, что компания требует изменений в бизнес-функционировании, и ИС может выступить инструментом для решения выявленных в деятельности предприятия проблем.

Глава 2 Проектирование CRM-системы гаражно-строительной организации

2.1 Постановка задачи проектирования

Автоматизированная система управления (АСУ) – это совокупность экономико-математических методов, технических средств и организационных комплексов, обеспечивающих рациональное управление сложным объектом (например, предприятием, технологическим процессом). Наиболее важная цель построения любой АСУ – резкое повышение эффективности управления объектом (производственным, административным и т.д.) на основе роста производительности управленческого труда и совершенствования методов планирования и гибкого регулирования управляемого процесса [7].

Общее представление схемы АСУ представлено на рисунке 2.1.

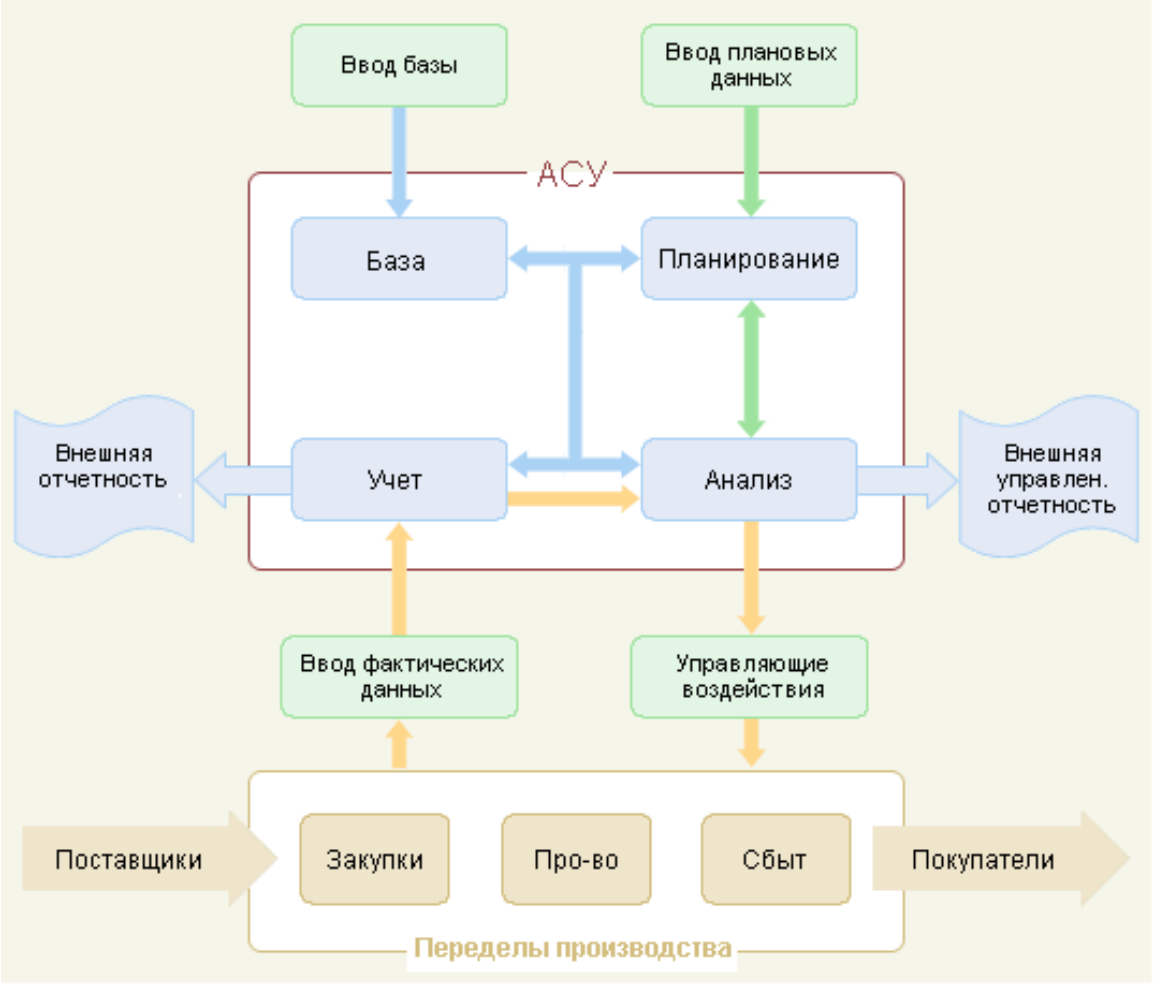

Рисунок 2.1 - Универсальная схема представления АСУ

В бакалаврской работе разрабатываются элементы CRM-системы взаимодействия с собственниками гаражно-строительного предприятия, основываясь на основных принципах общего построения АСУ. Подразумевая построение АСУ как следующий этап развития информационной системы.

Для эффективного решения поставленной задачи необходим комплекс технических средств, предназначенных для работы информационной системы [6]. Такой комплекс могут составлять компьютеры любых моделей, устройства сбора, накопления, обработки, передачи и вывода информации, устройства передачи данных и линий связи, оргтехника.

В качестве архитектуры была выбрана клиент-серверная организация системы с использованием веб-интерфейса (рисунок 2.2).

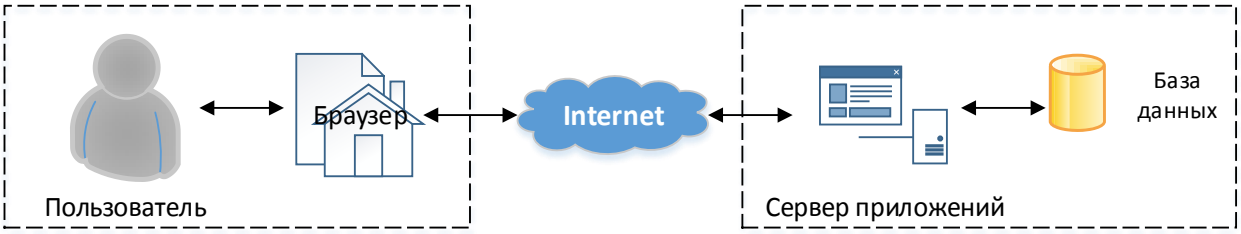

Рисунок 2.2 - Схема взаимодействия «Клиент-Сервер»

При выборе ТО необходимо учитывать архитектуру системы, а также руководствоваться рядом характеристик. К таким характеристикам относятся надежность, стоимость, производительность и другие. От значения указанных параметров зависит возможность работы с требуемыми программными средствами.

Для нормального функционирования разрабатываемой системы необходимо наличие ТО, указанного в таблице 2.1.

Таблица 2.1 - Техническое обеспечение системы

	Серверное обеспечение	**Клиентское обеспечение**
Тактовая частота процессора	2000 МГц и выше	1000 МГц и выше
Объем ОЗУ	не менее 2 Гбайт	не менее 512 Мбайт
Свободный объем	160 Гбайт	20 Мбайт

	Серверное обеспечение	**Клиентское обеспечение**
жесткого диска		
Сетевая карта	Ethernet 10/100 Мбит/с	Ethernet 10/100 Мбит/с
Монитор, видеоадаптер	минимальное разрешение экрана – 800 x 600, 64 Мбайт и выше	минимальное разрешение экрана – 800 x 600, 64 Мбайт и выше
Прочие устройства	клавиатура, мышь, принтер	клавиатура, мышь, принтер
Операционная система	Windows, UNIX, Solaris	Windows, UNIX, Solaris, MacOS
Веб-браузер	любой браузер	любой браузер

Следует отметить, что внедрение разрабатываемой системы на базе гаражно-строительного предприятия потребует установку требуемого технического обеспечения по причине отсутствия таковой в стандартной минимальной установке.

Согласно поставленной задаче, требуется реализовать веб-приложение, основанное на концепции «облачных вычислений», которое благодаря веб-архитектуре предоставляет наиболее простой способ доступа большого числа клиентов к информационной системе [13].

Для реализации системы был выбран язык программирования Javaплатформы J2EE. Java – объектно-ориентированный, структурный язык программирования. Он обладает рядом следующих преимуществ: преемственность языка, надежность и безопасность, независимость от архитектуры и переносимость, производительность, многопоточность.

Сервер приложений — это реализация системы в соответствии со спецификацией J2EE, обеспечивающая работу модулей с логикой конкретного приложения. Включает в себя как минимум следующие сервисы:

- EJB-контейнер, который поддерживает автоматическую синхронизацию Java объектов с базой;
- JMS — сервис доставки сообщений между компонентами и серверами;
- управление ресурсами (доступ к СУБД, файловой системе и т. д.);
- безопасность и защита данных;

- поддержка транзакций;
- веб-сервер;
- поддержка веб-сервисов.

На данный момент распространенным сервером приложений является JBossapplicationserver. Существует как Windows-, так и Unix-версия, практически не отличающиеся по возможностям. Преимуществами JBoss является его бесплатность даже для коммерческих решений (лицензия GNU LGPL) при значительной функциональности.

В качестве средства управления базами данных была выбрана MySQL. Применение СУБД MySQL идеально для Интернет-сайтов, как небольших, так и достаточно крупных. MySQL отличатся хорошей скоростью работы, надежностью, гибкостью. Работа с ней, как правило, не вызывает больших трудностей. MySQL распространяется на условиях общей лицензии GNU (GPL, GNU Public License).

Также при разработке дизайна системы использованы таблицы стилей, или CSS. Таблицы стилей (stylesheets) представляют собой абстракцию, в которой стиль документа определяется отдельно от его содержания и может быть легко модифицирован в зависимости от желания разработчика.

Таким образом данный проект реализован средствами языка программирования Java, таблицы стилей CSS, технологии Ajax, системы управления базами данных MySQL и сервера приложений JBoss.

Под технологическим обеспечением понимается совокупность средств и методов создания, актуализации, сохранения и эксплуатации ее информационных ресурсов. Операции сбора и регистрации данных осуществляются с помощью различных средств, отображающих реальное состояние объекта на носители информации [8].

Таким образом, наиболее эффективно использовать визуальный и программно-логический контроль.

Поскольку разрабатываемую информационную систему предполагается использовать на отечественном рынке, то в качестве языка

общения системы с пользователем был выбран русский язык. Таким образом, заголовки, пункты меню, всплывающие подсказки и обязательная информация указаны на выбранном языке. В качестве набора символов (кодировки), используемого системой, был выбран формат UTF-8 (от англ. UnicodeTransformationFormat – формат преобразования Юникода) [13]. Для защиты данных в разрабатываемой системе предусмотрено разграничение уровней доступа пользователей согласно их основной роли (пользователь, администратор).

Таким образом, был определен набор инструментов, с помощью которого может быть реализована информационная система. А также показана ее архитектура, которая должна позволить предприятию работать постоянно и тем самым, повышать свою привлекательность.

2.2 Разработка диаграмм прецедентов

Назначение создаваемой CRM-системы – автоматизация взаимодействия предприятия с его собственниками и новыми клиентами. В настоящее время эта работа производится вручную сотрудниками офиса. На рисунке 2.3 представлена модель деятельности в виде диаграммы прецедентов.

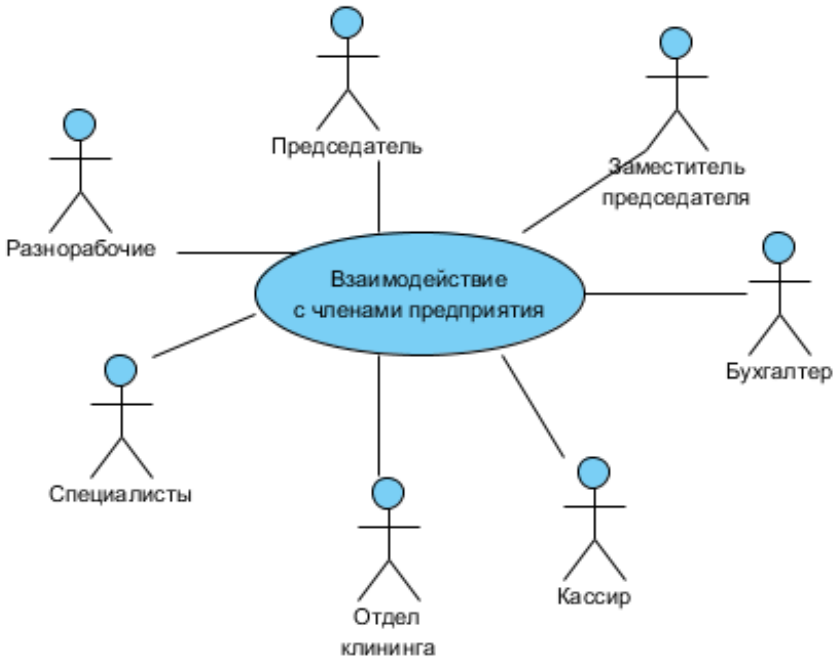

Рисунок 2.3 - Общая диаграмма деятельности предприятия по взаимодействию с его клиентами

Прецедент "Взаимодействие с членами предприятия" реализуется через множество других, более ограниченных прецедентов (рисунок 2.4), отражающих детализацию представления функционирования организации.

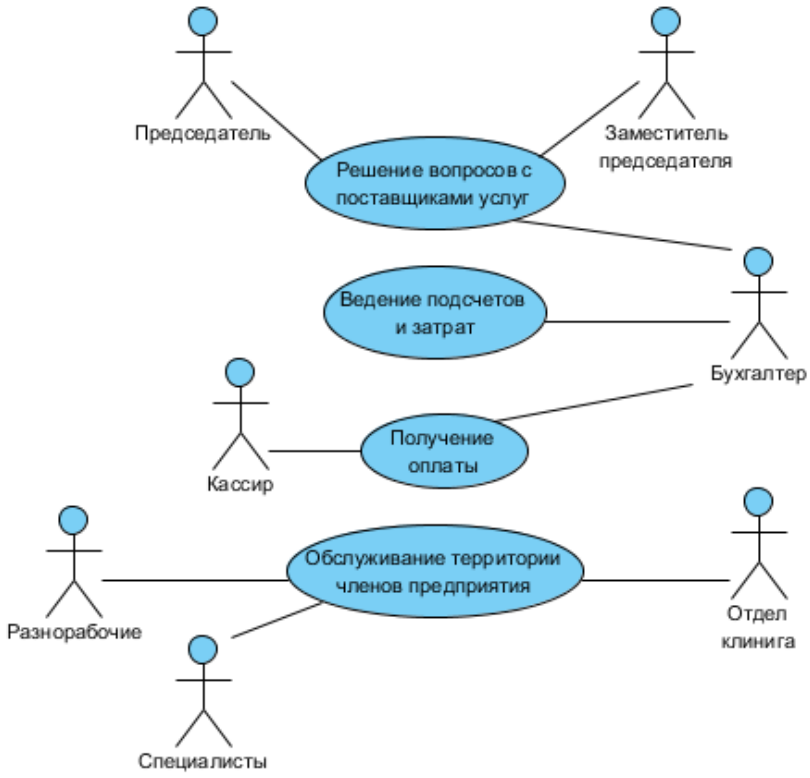

Рисунок 2.4 - Модель бизнес-прецедентов, составляющих взаимодействие с членами предприятия

Для включения в диаграмму выбранные прецеденты должны удовлетворять следующим критериям:

- прецедент должен описывать, **ЧТО** нужно делать, а не **КАК**;
- прецедент должен описывать действия с точки зрения **ИСПОЛНИТЕЛЯ**;
- прецедент должен возвращать исполнителю некоторое **СООБЩЕНИЕ**;

- последовательность действий внутри прецедента должна представлять собой одну **НЕДЕЛИМУЮ** цепочку.

Выполнение прецедента описывается с помощью диаграмм видов деятельности, которые отображают исполнителей и последовательность выполнения соответствующих бизнес-процессов

Таким образом, представив основные моменты деятельности организации через бизнес-прецеденты, теперь можно перейти к разработке концептуальной модели будущей системы.

2.3 Разработка информационной модели CRM-системы

Информационная модель (концептуальная модель) включает в себя совокупность входных и выходных документов, файлов входной оперативной, постоянной, промежуточной и результатной информации [8].

Ядром ИС является база данных (БД). Процесс проектирования базы данных начинается с установления концептуальных требований пользователей системы. Концептуальное проектирование – это процедура конструирования информационной модели, не зависящей от каких-либо физических условий реализации. Информационными объектами обычно являются сущности – обособленные объекты или события, информацию о которых необходимо сохранять, имеющие определенные наборы свойств – атрибутов.

Проектирование концептуальной модели основано на анализе предметной области и решаемых задач по обработке данных.

Для справочной информации были выделены сущности: Тип собственности, Тип льгот, Бухгалтерский учет, Тип расчета, Тип счетчика. Сущности Назначенные льготы, Льготная категория, Льгота, Схема расчета содержат информацию о всех возможных льготах и о способах получения этих льгот. Информация об индивидуальных средствах учета хранится в сущностях счетчик и Учет показаний. Сущности СОБСТВЕННИК и ЧЛЕН КООПЕРАТИВА отражают частную и юридическую информацию о

владельце гаража, в то время как ГАРАЖ отражает его собственность. ЗАТРАТА и СТАТЬЯ ЗАТРАТ и ПЛАТЕЖ используются для начисления платежей владельцам гаражей.

На рисунке 2.5 представлена концептуальная модель.

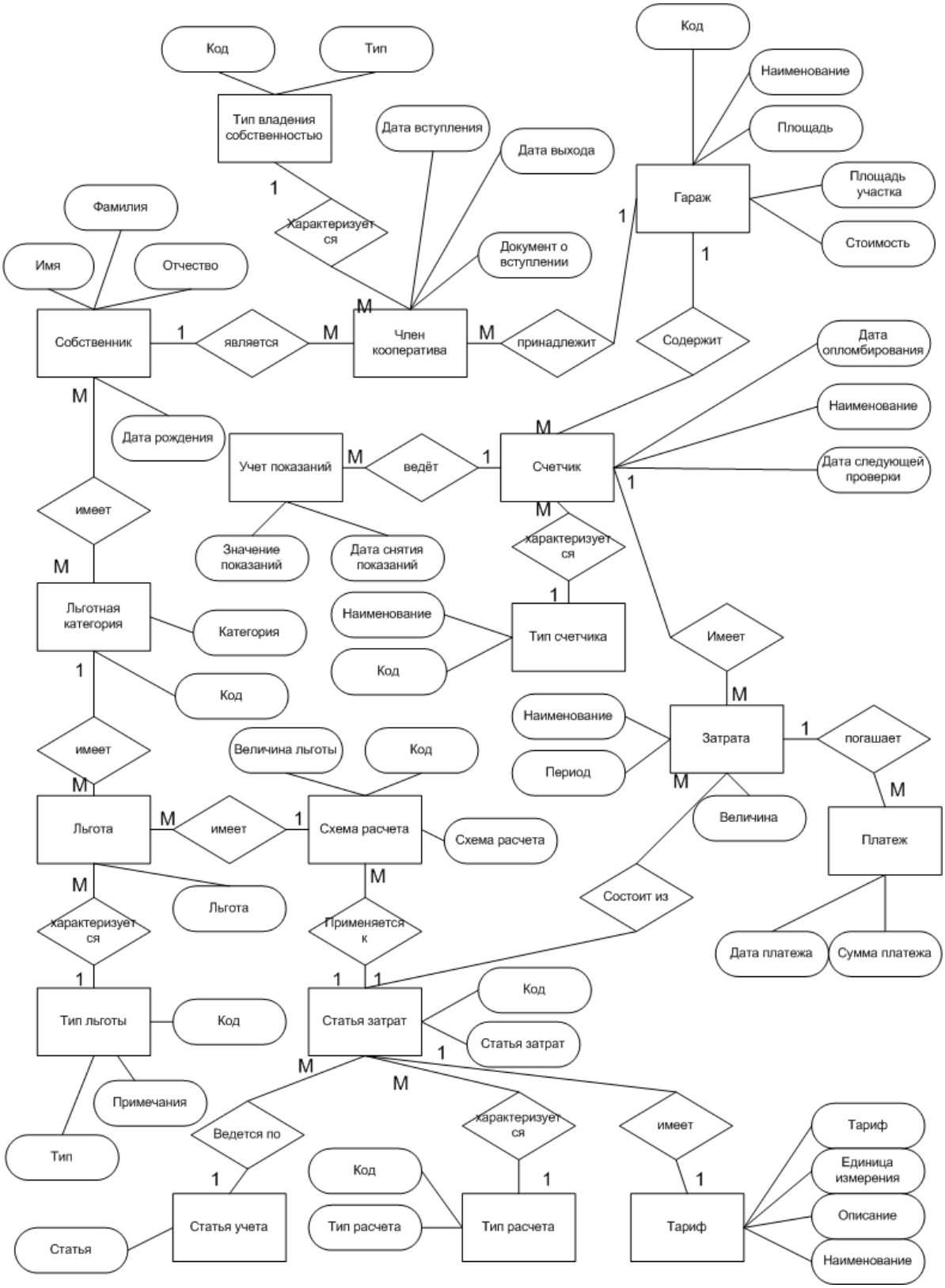

Рисунок 2.5 - Концептуальная модель данных

На концептуальной модели отображены описания объектов и их взаимосвязей, представляющих интерес в рассматриваемой предметной области и выявляемых в результате анализа данных. Данная модель инвариантна к структуре базы данных и отображает основные объекты и их характеристики, которые составляют сущность процессов хозяйственной деятельности гаражно-строительного предприятия.

Таким образом, имя концептуальное представление модели данных можно построить логическую модель данных разрабатываемой системы.

2.4 Проектирование логической и построение физической модели данных

Важнейшим этапом проектирования базы данных является разработка логической модели предметной области. Версия концептуальной модели, которая может быть обеспечена конкретной системой управления базой данных (СУБД), называется логической моделью [3].

Логическая модель отображает связи между информационными данными в концептуальной модели. Чем ближе логическая модель к концептуальной, тем проще и эффективнее работать с системой, тем легче процесс внедрения и тем больше отдача от автоматизации.

Основываясь на том, что физическая модель данных – это продолжение логической модели, следующим шагом является ее разработка и построение. Эта модель соответствует практической реализации базы данных и определяет те физические объекты, которые предстоит реализовать. При переходе от логической модели к физической сущности преобразуются в таблицы, а атрибуты — в поля (столбцы).

Физическая модель – это модель, определяющая размещение данных, методы доступа и технику индексирования, так же ее называют внутренней моделью системы. Этап физического проектирования базы данных заключается в связывании логической структуры данных и физической среды

хранения с целью отображении логической структуры данных в структуру хранения.

Физическая модель данных является полностью компьютерно-ориентированной и конечные пользователи, а порой и прикладные программисты, не имеют никакого представления о том, каким образом данные запоминаются и извлекаются или каким способом организуются индексы в таблицах для быстрого поиска или ссылочная целостность. Эти и множество других функций по методам доступа и поддержании баз данных на внешних носителях, а также способов поиска и доступа к данным в современных СУБД обеспечивается в основном ядром базы данных, что значительно облегчает задачу создания базы данных и их ведение.

Уникальность первичных ключей обеспечивается средствами СУБД путем установки для поля свойства Auto_increment (автоматический расчет уникального номера записи в таблице). Для обеспечения проверки целостности ссылочных данных были использованы внешние ключи.

Для хранения текстовых данных используются строки переменной длины типа VARCHAR. Для хранения чисел используется типы данных INT и DOUBLE. Тип данных DATETIME используется для хранения значений дат с указанием времени суток. Тип BOOLEAN (логическое поле) используется для обозначения статуса пользователя: активированный (1, true) или не активированный (0, false).

Модель разработана средством визуального проектирования баз данных DBDesigner 4. Система объединяет базы данных, моделирование, создание и поддержание единой цельной среды, сочетает в себе профессиональные возможности и простой пользовательский интерфейс. DBDesigner 4 разработан и оптимизирован с открытым исходным кодом базы данных MySQL. Средство является бесплатным и общедоступным.

Схема разработанной физической структуры данных представлена на рисунке 2.6.

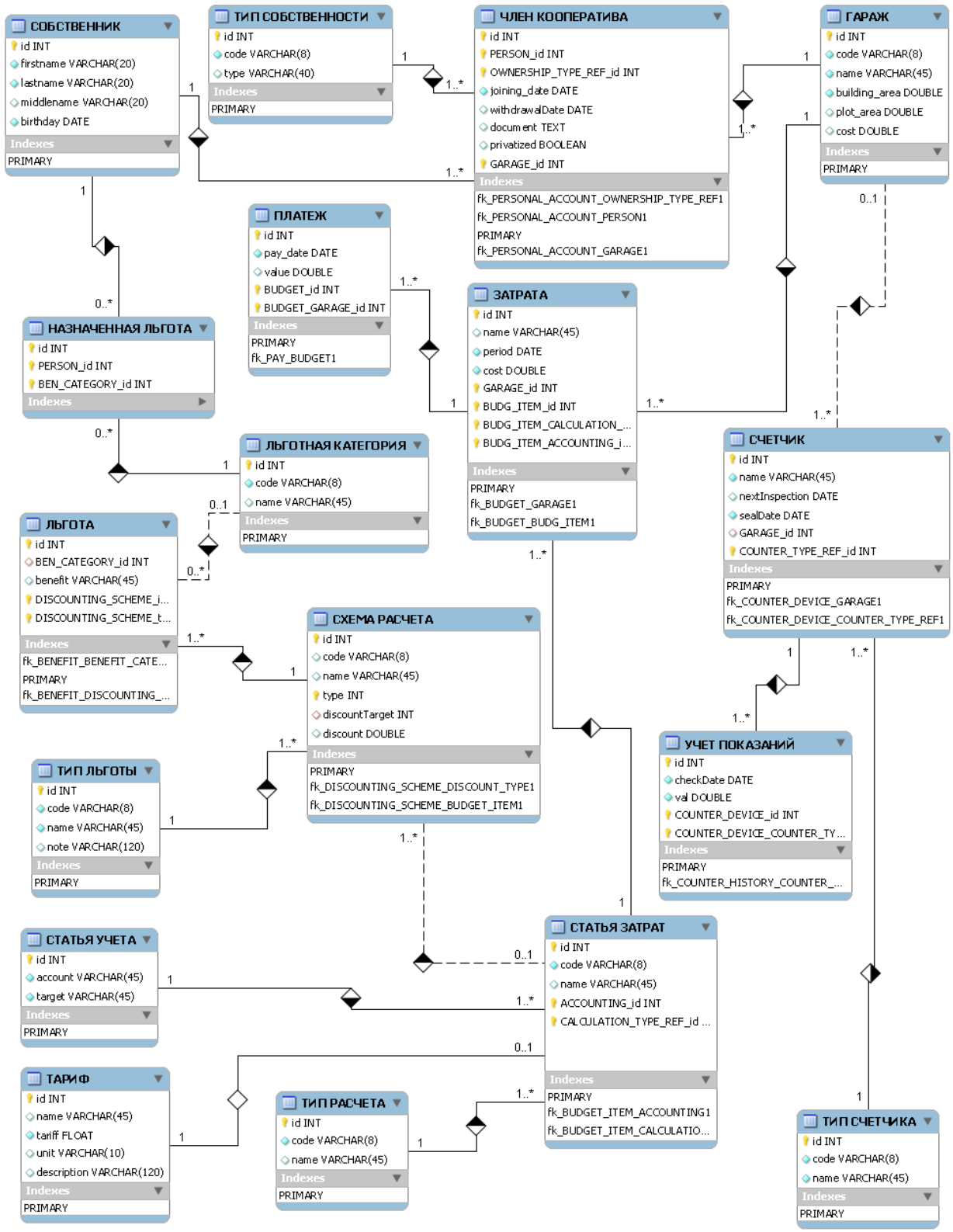

Рисунок 2.6 - Физическая модель данных

Физическая модель данных строится на базе логической модели и описывает данные уже средствами конкретной СУБД. Отношения, разработанные на стадии логического моделирования, преобразуются в

таблицы, атрибуты – в столбцы, домены – в типы данных, принятых в выбранной конкретной СУБД.

Исходя из построенных моделей данных, можно выделить основные сущности и их атрибуты, описывающие базу данных. Определив данные сущности, можно перейти к разработке автоматизированной системы.

Таким образом, была построена физическая модель данных, которая затем будет реализована с использованием ранее выбранной СУБД.

2.5 Описание создания элементов CRM-системы для гаражно-строительной организации

В процессе создания ядра приложения были созданы и использованы компоненты для управления классами сущностей, используемых в базе данных автоматизированного рабочего места (таблица 2.15).

Таблица 2.15 - Характеристика основных типов классов приложения

Класс	**Характеристика**
Entity	Представляет собой класс сущности, отражающей таблицу базы данных. Имя класса является именем таблицы, поля класса являются столбцами таблицы, а экземпляры класса — записями.
EntityHome	Представляет собой класс, предназначенный для управления классом Entity. Отвечает за создание, выборку, обновление и удаление записей (т.е. экземпляров класса сущности).
EntityList	Представляет собой класс, отвечающий за получение списка экземпляров сущности.

Рассмотрим каждый из этих классов более подробно:

1) Компонент Home.

Класс Home управляет экземплярами сущности за счет кэширования и координации CRUD операций, выполняемых с этими классами, с компонентом управления сохранением. Каждый Home компонент представлен классом Home, наследующегося от класса PersistenceController,

или любого подкласса Home. Диаграмма классов для EntityHome показана на рисунке 2.16.

Компоненты Home используются для управления существующими записями в таблице базы данных разработанного приложения. Для выполнения этой функции компоненту Home необходимо передать идентифицирующее значение сущности в метод setId(). Идентификатор сущности представляет собой уникальный экземпляр сущности, который сопоставляется уникальным ключом в соответствующей таблице базы данных через аннотацию @Id поля объекта класса.

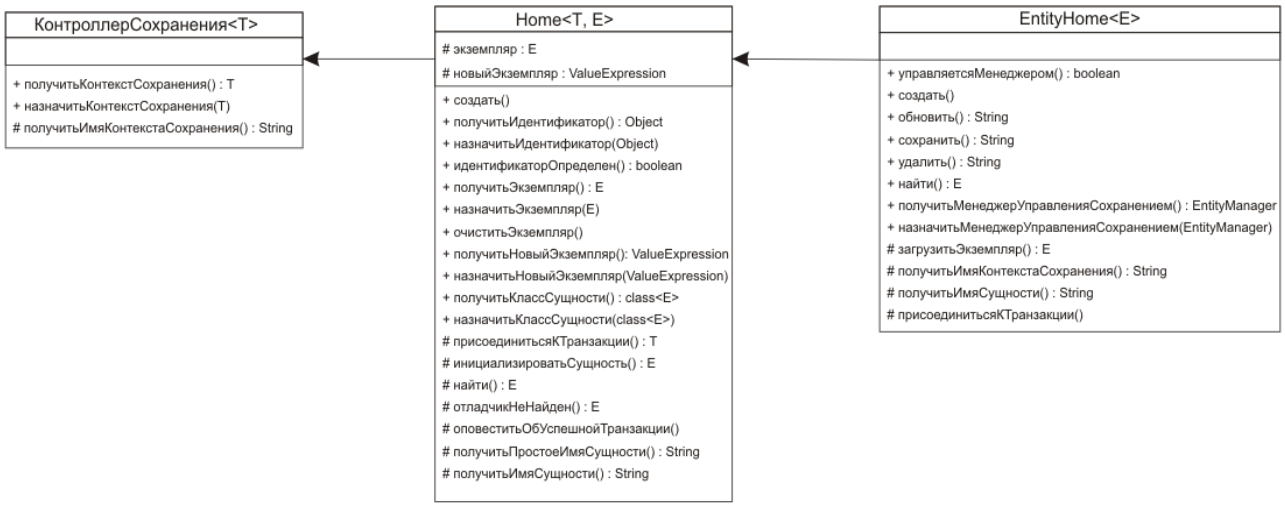

Рисунок 2.16 - Диаграмма классов EntityHome

Метод getInstance() компонента Home использует этот идентификатор для поиска сущности, вызывая метод find() менеджера управления сохранением, что проиллюстрировано на рисунке 2.17.

Когда компоненту Home назначается другой идентификатор, последующий вызов getInstance() выполняет поиск с целью обновления данных о сущности, получая экземпляр класса, связанный с идентификатором. Если идентификатор не был назначен до того, как метод getInstance() был вызван, то новый экземпляр сущности будет автоматически создан, путем вызова метода createInstance(). Для дальнейшего рассмотрения функциональных особенностей и применения компонента Home рассмотрим его модель.

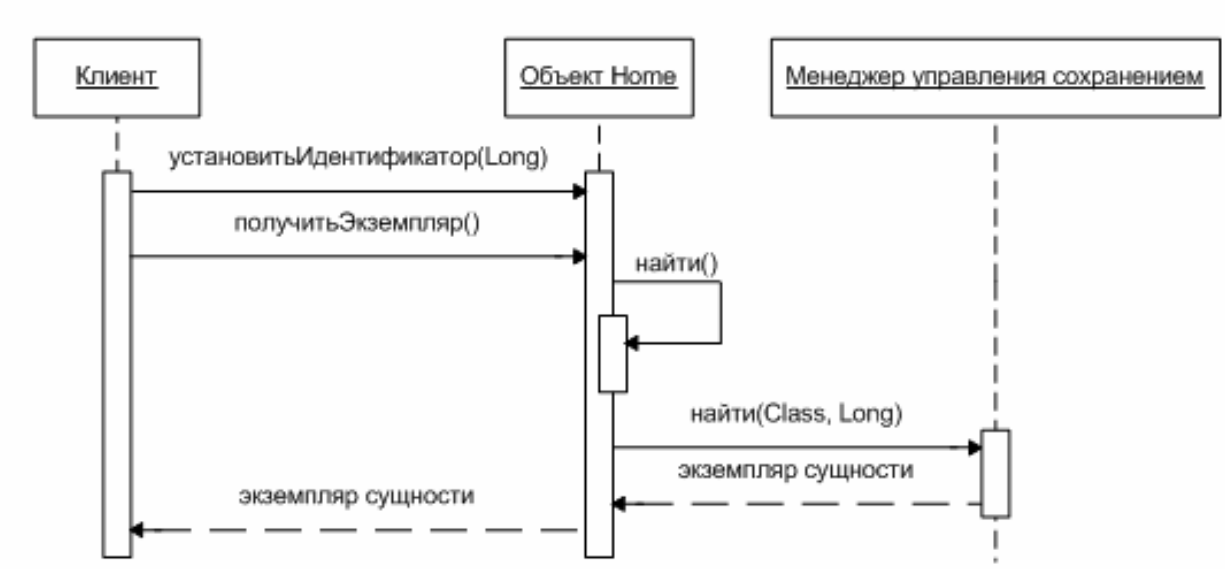

Рисунок 2.17 - Диаграмма последовательности, изображающая как компонент Home получает экземпляр класса сущности

2) Модель компонента Home.

Компонент Home включает в себя экземпляр сущности и менеджер сохранения, упрощая взаимодействие между ними, при этом оставляя их независимыми друг от друга, что показано на рисунке 2.18.

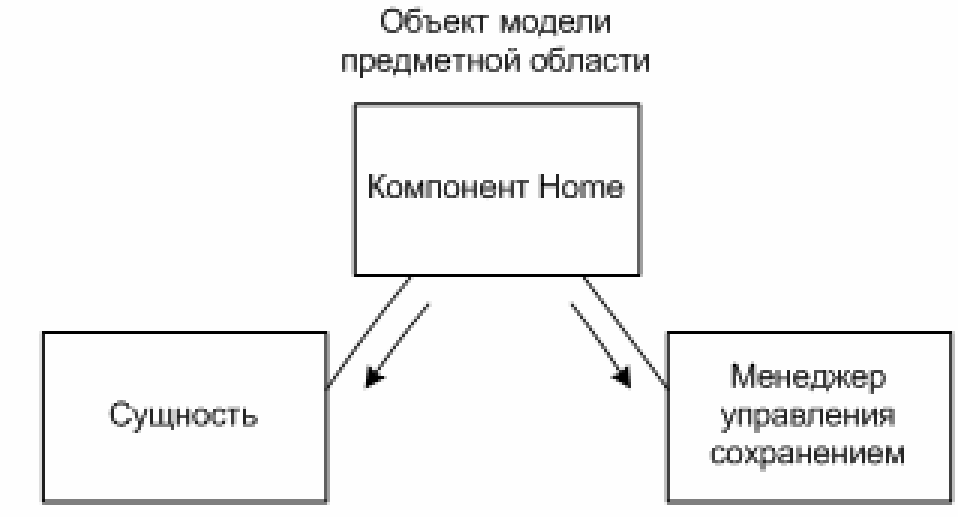

Рисунок 2.18 - Взаимодействие между Entity и Home

Шаблон проектирования, называемый доменный объект и примененный для создания компонента Home, выступает для внешнего мира

в качестве единого элемента, позволяя этому новому составному классу действовать, не заполняя классы сущности логикой доступа к базе данных.

Методы класса Home, выполняющие взаимодействие с менеджером сохранения перечислены в таблице 2.15.

Таблица 2.15 - Методы класса Home

Метод	**Описание**
getInstance()	Возвращает экземпляр сущности, управляемой объектом Home.
isManaged()	Проверяет, находится ли сущность во «временном» состоянии или оно уже сохранено в базу данных.
persist()	Сохраняет «временный» экземпляр сущности в базу данных.
update()	Синхронизирует состояние сохраненной сущности с записью в таблице базы данных.
remove()	Делает экземпляр сущности «временным», удаляя его из БД.

Класс Home позволяет помещать методы менеджера управления сохранениями (т.е. методы для работы с базой данных) в транзакции, объявляя их при помощи аннотации @Transactional, помещенной над методом класса. Ниже приведен отрывок метода persist() компонента EntityHome:

```
@Transactional
publicStringpersist() {
getEntityManager().persist(getInstance());
getEntityManager().flush();
...
return "persisted";
}
```

На примере этого метода можно увидеть, как компонент Home служит посредником между экземпляром сущности и менеджером управления сохранением. Кроме того, компонент Home предоставляет возможности кэширования, что приводит к устранению необходимости получать экземпляр сущности каждый раз при переходе на новую страницу, а также необходимости заливать в базу данных изменения экземпляра сущности каждый раз, когда они происходят. Рассмотрим перечисленные функциональные особенности на примере метода обновления базы данных измененной сущностью. Каждой сущности системы автоматизированного

рабочего места соответствует страница редактирования данных сущности. На этой странице расположена кнопка «Сохранить», связанная с методом update() соответствующего компонента Home. Ниже приведена рассматриваемы отрывок кода метода update() класса EntityHome:

```
@Transactional
public String update() {
joinTransaction();
getEntityManager().flush();
...
return "updated";
}
```

Метод Update() выполняет две операции менеджера управления сохранением, ни одна из которых не вызывает явного обновления. Данный отрывок демонстрирует, что менеджер управления сохранением сначала присоединяется к активной транзакции, а затем отправляет произведенные изменения синхронизации с базой данных. Эти изменения становятся постоянными, только тогда, когда транзакция успешно завершается. До этого они хранятся в хэшированной области изменений.

Update () метод не нуждается в выполнении операции слияния, даже если экземпляр сущности хранится не в долгосрочной работе в режиме conversation. Причина в том, что экземпляр сущности извлекается из базы данных в фазу UpdateModelValues обновления значений модели до того, как значения из формы применяются к нему. Таким образом, экземпляр гарантированно будет управлять при вызове Update() метода. Любые изменения, сделанные на этапе обновления значений модели определяются и синхронизируются с базой данных.

Таким образом, была решена задача создания модуля информационной системы, которая позволила автоматизировать некоторые функции предприятия. В основном, данные функции были ориентированы на взаимодействие с клиентами, позволяя дать наглядную информацию о совершаемых действиях.

2.6 Структура и функции разрабатываемой автоматизированной системы взаимодействия с клиентами организации

Автоматизированная система управления — это комплекс аппаратных и программных средств, предназначенный для управления различными процессами в рамках технологического процесса предприятия.

Автоматизированная система управления хозяйственной деятельностью гаражно-строительного предприятия позволит автоматизировать все текущие процессы ПЭГК. Оно обеспечит сбор, хранение, обработку, поиск, выдачу информации, необходимой пользователю. Система поможет сотрудникам, работающим с данными, повышать продуктивность и производительность работы предприятия. Структура информационной системы – это состав, порядок и принципы взаимодействия элементов системы, определяющие основные свойства системы. Взаимодействие пользователя с системой осуществляется в пользовательском режиме (рисунок 2.19).

В состав автоматизированной системы управления гаражно-строительного предприятия входят все данные необходимые пользователю для работы. Специализированный пользователь сможет выводить на печать необходимые отчеты и наблюдать изменения временных диаграмм.

Взаимодействие пользователя с системой осуществляется в основном пользовательском режиме. Сначала пользователь проходит авторизацию. После авторизации определяется роль пользователя.

Важная часть информационной системы – возможность контролировать доступ к данным. После авторизации приложение определяет возможность доступа к ресурсам. АСУ хозяйственной деятельностью гаражно-строительного предприятия представляет собой систему процедур и функций, взаимосвязанных между собой. Рассмотрим две операции:

- регистрация собственника;

- прием платежей.

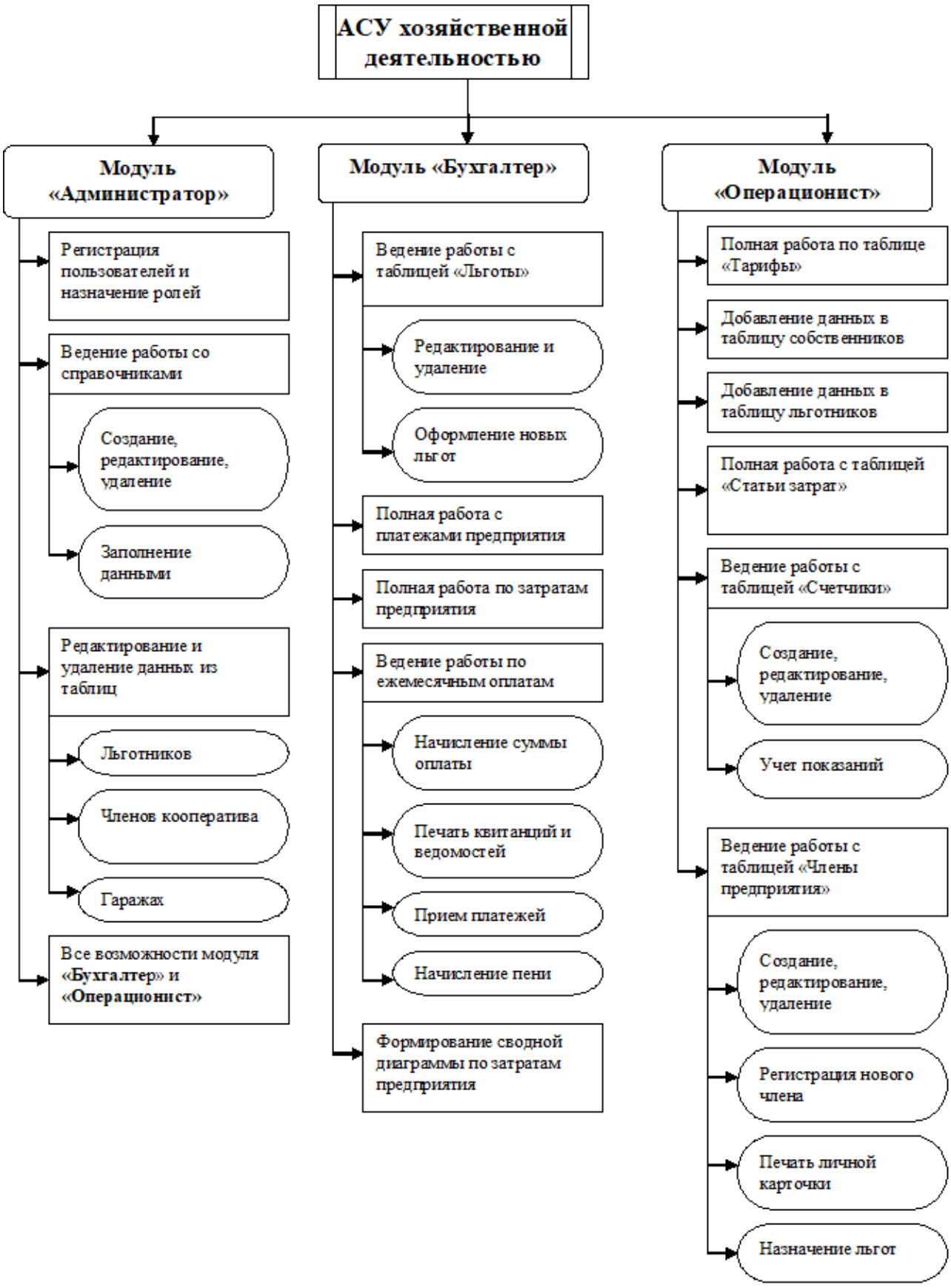

Рисунок 2.19 - Структура функционирования системы управления

На рисунке 2.20 представлен алгоритм регистрации собственника гаража.

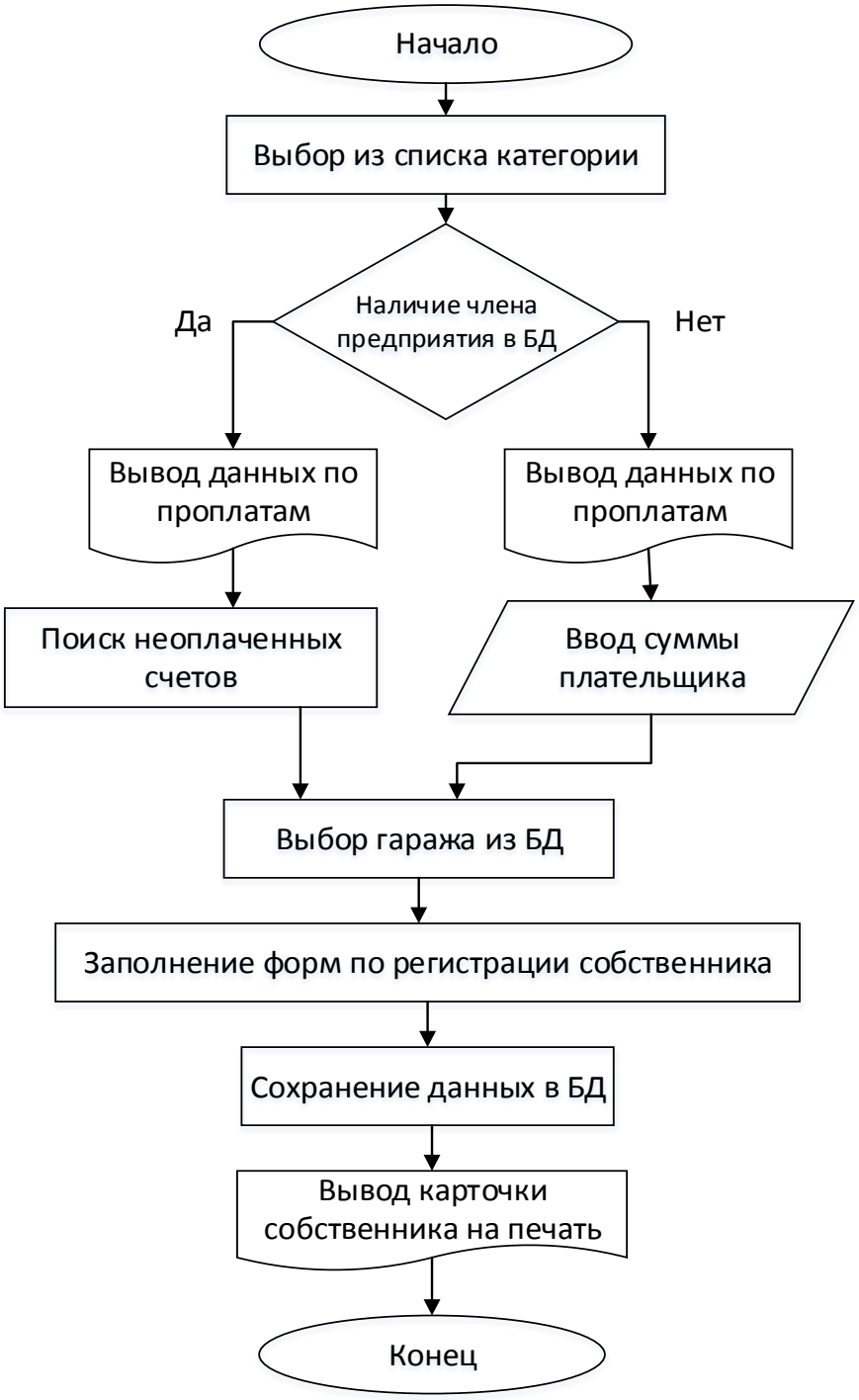

Рисунок 2.20 - Алгоритм регистрации собственника

Элемент кода, выполняющий функции, описанные в алгоритме:

```
private static final long serialVersionUID = 1L;
@Logger private Log log;
```

```
   @In EntityManagerentityManager;
   @In(create=true,required=false) /*@Out(required=false)*/
PersonHomepersonHome;
   @In(create=true,required=false)/* @Out*/
PersAccountHomepersAccountHome;
   @In(create=true,required=false) /*@Out*/
GarageHomegarageHome;

   //@In(create=true) //@Out
booleanpersonAlreadyExists = false;

   public booleangetPersonAlreadyExists() {
         log.info("trying to get");
               return personAlreadyExists;
         }
         public void setPersonAlreadyExists(booleanpersonAlreadyExists) {
               log.info("trying to set");
               this.personAlreadyExists = personAlreadyExists;
         }
         @Begin
   public String begin()
   {
log.info("beginning conversation");
log.info("personAlreadyExists #0", personAlreadyExists);
     if (personAlreadyExists)
         return "select";
     else return "createNewPerson";
   }
      public void load()
   {
         log.info("!!!start loading");
         if (personHome.isManaged())
         {
               personHome.load();
         }
         log.info("!!!end loading");
   }

   public void loadGar()
   {
               if (garageHome.isManaged())
         {
               garageHome.load();
         }           }
     public String selectGarage()
   {
         return "select";
   }
```

```
public void persistPerson()
{
        if (!personHome.isManaged())
        {
                /*String outcome = personHome.persist();*/
                try {
                        Person person = personHome.getInstance();
                        entityManager.persist(person);
                        personHome.setPersonIdentdoc(person.getIdentdoc());
                        log.info("Person persisted");
                } catch (Exception e) {
                        log.info("Error while persisting  Person");
                                e.printStackTrace();
                        }

        }
}
  @Transactional(TransactionPropagationType.REQUIRED)
public void register()
{
        persistPerson();
        log.info("wiring");
        persAccountHome.wire();
        log.info("persisting");
        persAccountHome.persist();
}
   @End
public String end()
{
        try{
                register();
        } catch (Exception e) {

                        log.info("Error while persisting  Account");
                e.printStackTrace();
                return "fuck";
                }
log.info("ending conversation");
     return "home";
}}
```

Таким образом, при регистрации собственника гаража система вначале проверяет уже имеющихся (зарегистрированных) и, в случае наличия такого, представляет форму, позволяющую прикрепить новый гараж данному собственнику; в случае отсутствия данной личности в базе, представляет

форму для регистрации нового собственника. Итогом работы является возможная для ввода на печать карточка собственника.

На рисунке 2.21 представлен алгоритм приема платежей системой в гаражно-строительном предприятии.

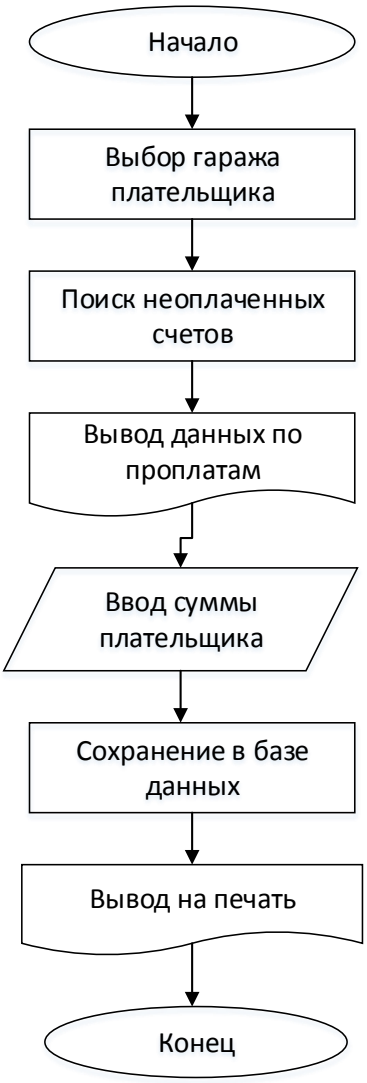

Рисунок 2.21 - Алгоритм приема платежей

Элемент кода, выполняющий функции, описанные в алгоритме:

```
public class Payer implements Serializable
{
   private static final long serialVersionUID = 1L;

        private static final String OWNER_TYPE = "P’";

        @Logger private Log log;
        @In EntityManagerentityManager;
        @In(create=true,required=false)
```

```
GarageHomegarageHome;
        @In(create=true,required=false)
BudgetHomebudgetHome;
        @In(create=true,required=false)
PayHomepayHome;

        @Out
        List<Budget> budgets = new LinkedList<Budget>();
        @Out(required = false)
        double toPay;
        @Out(required = false)
        double debtToPay;

   @Begin
   public String begin()
   {
log.info("beginning conversation");
    return "begin";
   }

   public void loadGar()
   {
        log.info("!!!start loading");
        if (garageHome.isManaged())
        {
                garageHome.load();
        }
        log.info("!!!end loading");
   }
     public String showSum()
   {
        return "ok";
   }
     public void calculatePay()
   {
        List<PersAccount> accounts = garageHome.getPersAccounts();
        PersAccount owner = null;
        for (PersAccountaccount : accounts)
        {
                if(account.getWithdrawalDate()                  ==                  null
&&account.getOwnershipTypeRef().getCode() == OWNER_TYPE)
                {
                        owner = account;
                        break;
                }       }
        String query = "select budget from Budget budget" +
                        " where budget.persAccount = :owner" +
                        " and (budget.payed = null or budget.payed = false)" +
                        " order by budget.id.period desc";
```

```
                budgets = entityManager.createQuery(query).setParameter("owner",
owner).getResultList();
                toPay = 0.0;
                debtToPay = 0.0;
                Date now = new Date();
                for (Budget budget : budgets)
                {
                        int delay = 0;
                        if (budget.getPayBefore().before(now))
                        {
                                delay = getNumberOfDaysBetween(budget.getPayBefore(), now);
                                budget.setDebt(delay*budget.getPenaltyFee());
                                debtToPay +=budget.getDebt();
                                toPay +=budget.getDebt();
                        }
                        toPay += budget.getCost();
                }
        }
        public int getNumberOfDaysBetween(Date startDay, Date endDate){
                GregorianCalendar start = new GregorianCalendar();
                GregorianCalendar end = new GregorianCalendar();
                if (startDay.before(endDate))
                {
                        start.setTime(startDay);
                        end.setTime(endDate);
                } else if (startDay.after(endDate))
                {
                        start.setTime(endDate);
                        end.setTime(startDay);
                } else {
                        return 0;       }

                start.get(Calendar.DAY_OF_YEAR);
                end.get(Calendar.DAY_OF_YEAR);

                start.get(Calendar.YEAR);
                end.get(Calendar.YEAR);

                if (start.get(Calendar.YEAR) <end.get(Calendar.YEAR))
                {
                        int numberOfDays = 0;
                        while (!start.after(end))
                        {
                                numberOfDays++;
                                start.add(Calendar.DAY_OF_YEAR, 1);
                        }
                        return numberOfDays;
                } else {
```

```
                return end.get(Calendar.DAY_OF_YEAR) -
start.get(Calendar.DAY_OF_YEAR);
            }   }
          public String pay()
        {
            for (Budget budget : budgets)
            {
                budgetHome.newInstance(budget);
                budgetHome.getInstance().setPayed(true);

                payHome.newInstance();
                Pay pay = payHome.getInstance();
                pay.setBudget(budget);
                pay.getId().setPayDate(new Date());
                pay.setValue(budget.getCost()+budget.getDebt());
                payHome.persist();
                budgetHome.persist();
                    }
            return "print";
        }
          @End
        public String end()
        {
      log.info("ending conversation");
          return "home";
        }}
```

Таким образом, основные функции разрабатываемой системы:

1. Каталогизация:

- ввод и редактирование данных о собственниках;

- возможность работы с различными типами учета (соответствие боксу, сделанные платежи, значения счетчиков и т.д.).

2. Доступ к АСУ:

- возможность доступа согласно категории пользователя и роли;

- обеспечение поиска посредством фильтра.

3. Отчеты:

- формирование списка зарегистрированных собственников;

- создание списка льготников гаражного кооператива;

- создание отчета по показаниям счетчиков;

- формирование отчета по произведенным платежам;

- вывод на печать квитанции на оплату собственникам боксов.

4. Диаграммы:

- формирование диаграмм по затратам ПЭГК;

- формирование диаграммы по потреблению электроэнергии.

Разработанные алгоритмы показывают, как функционируют основные модели в разрабатываемой системе управления хозяйственной деятельностью гаражно-строительного предприятия, которая в дальнейшем позволит руководству организации осуществлять мониторинг, контроль и быстрое принятие решений в своей деятельности.

2.7 Описание основных программных модулей

Для входа в систему пользователю необходимо авторизоваться (рисунок 2.22).

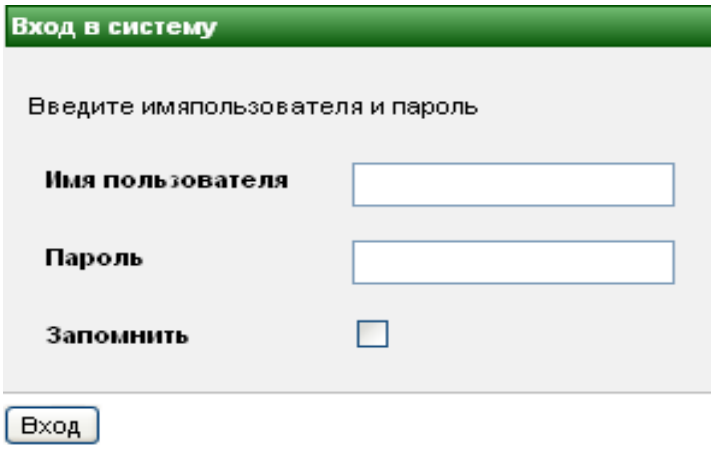

Рисунок 2.22 - Форма авторизации пользователя

После нажатия на кнопку программа зашифровывает введенный пароль, посылает запрос к таблице ПОЛЬЗОВАТЕЛЬ базы данных «business_plan» и проверяет, есть ли в ней введенные логин и пароль. Если такой пользователь нашелся, то в сессию заносится объект currentUser, являющийся объектным отображение таблицы БД. При успешной авторизации на главной странице, в правом верхнем углу, появится имя пользователя и ссылка «Выход» для завершения сеанса.

Если введенные логин и пароль не найдены в базе, выводится предупреждение «Такой логин не зарегистрирован, либо пароль неверный».

Авторизованному пользователю системы предоставлена возможность добавления данных. Для этого служат формы. Одна из них представлена на рисунке 2.23.

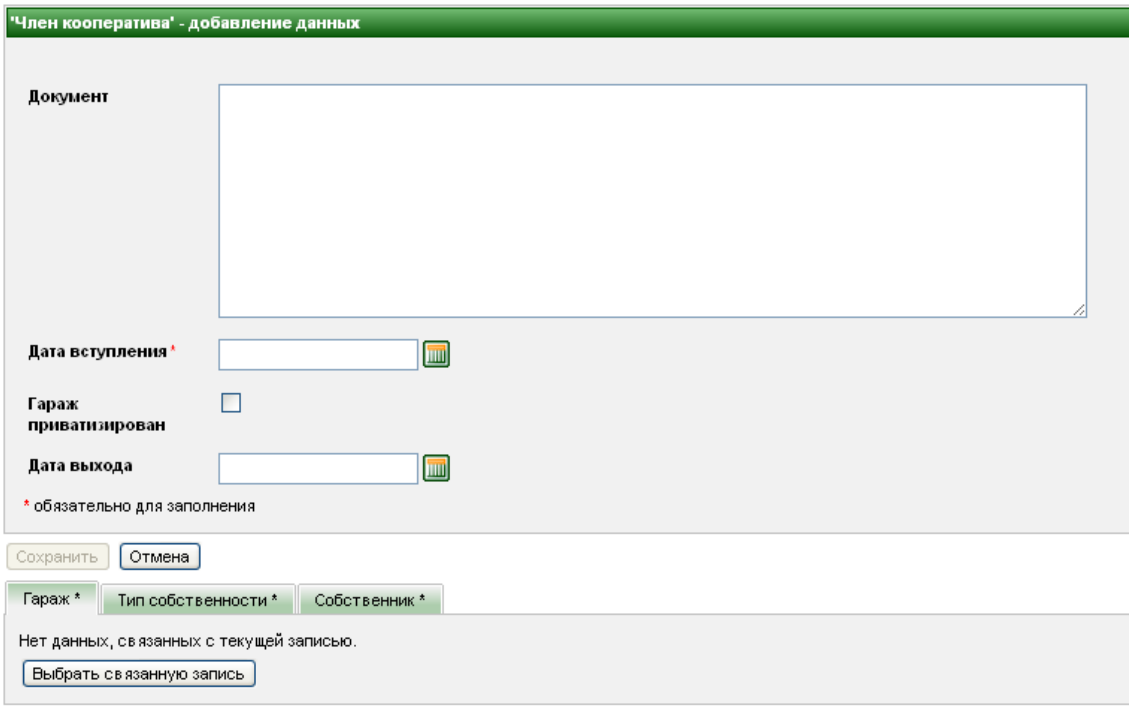

Рисунок 2.23 - Форма добавления новой записи (на правах администратора)

Добавление данных в Справочники возможно только создателем системы. Администратор же имеет право использовать их при заполнении разрешенных ему для работы форм.

Соответственно, и редактировать данные может только авторизованный администратор. Редактировать в данной системе возможно все то же, что добавляется и удаляется.

При осуществлении редактирования код программы будет выглядеть следующим образом:

```
action ="#{counterDeviceHome.persist}" //создаем сущность
disabled="#{!counterDeviceHome.wired}" //если заполнены все требуемые поля
rendered="#{!counterDeviceHome.managed}"/> //эта кнопка будет показана если сущность еще не существует, иначе будем использовать метод сохранения
<h:commandButton id="update"
```

```
value="Сохранить"
action="#{counterDeviceHome.update}" //метод сохранения
```

В определенной степени создание какой-либо записи – это своего рода редактирование, только начальные данные равны нулю. Программы, реализованные на Java, предусматривают данный подход.

Отчеты – это не сохраняемые в базе данных объекты, предназначенные для формирования печатных форм.

В данной системе существуют несколько видов отчетов. Один из них представлен на рисунке 2.24.

Льготники

Фамилия	Имя	Отчество	Дата рождения	Льготная категория	Описание льготы	Величина	Тип льготы	Применяется к статье затрат
Иванов	Иван	Иванович	14.11.2010					
				Участник ВОВ	Стандартный вычет из эл. энергии	300	Скидка указанной суммы	Расчеты с сетевой компанией гор энерго
				Участник ВОВ	Скидка на внутренние затраты	12	Скидка в процентах	Внеплановый ремонт крыши
				Пенсионер	Скидка пенсионерам на внутренние расходы	100	Скидка указанной суммы	Внеплановый ремонт крыши
Пересыпкин	Федор	Тихонович	14.12.1922					
				Участник ВОВ	Стандартный вычет из эл. энергии	300	Скидка указанной суммы	Расчеты с сетевой компанией гор энерго
				Участник ВОВ	Скидка на внутренние затраты	12	Скидка в процентах	Внеплановый ремонт крыши
				Пенсионер	Скидка пенсионерам на внутренние расходы	100	Скидка указанной суммы	Внеплановый ремонт крыши
				Пенсионер	Скидка пенсионерам на внутренние расходы	100	Скидка указанной суммы	Внеплановый ремонт крыши
Верещагин	Денис	Александрович	08.09.1982					
				Ветеран войны	Скидка указанной суммы из оплаты за свет	150	Скидка указанной суммы	Электроэнергия по счетчикам

Отчет в PDF | Отчет в Doc | Отчет в Xls

Рисунок 2.24 -Формирование отчета «Льготники»

Фрагмент кода генератора PDF отчетов:

```
<servlet>
<servlet-class>org.boxproject.birt.DocStoreServlet</servlet-class>
</servlet>
<servlet-mapping>
<servlet-name>PDF Document Store Servlet</servlet-name>
<url-pattern>*.pdf</url-pattern>
</servlet-mapping>

@Override
protected void doGet(final HttpServletRequest request,
        final HttpServletResponse response) throws ServletException,
        IOException {
                System.out.println("starting doget method");
                new ContextualHttpServletRequest(request) {
```

```
                @Override
                public void process() throws ServletException, IOException {
                        try {
                                executeReport(request, response);
                        } catch (EngineException e) {
                                log("executeReport error", e);
                        }
        }
        }.run();
}
void executeReport(HttpServletRequest request, HttpServletResponse response)
        throws EngineException {
        response.setContentType("application/pdf");
        response.setHeader("Content-Disposition", "inline; filename=test.pdf");

        String reportName = request.getServletPath();
        ServletContextsc = request.getSession().getServletContext();
        this.engine = BirtEngine.getBirtEngine(sc);
        IReportRunnable design;
        try {
                design = engine.openReportDesign(sc.getRealPath("./")
                                + reportName.substring(0, reportName.length() - 4)
                                + ".rptdesign");
                IRunAndRenderTask task = engine.createRunAndRenderTask(design);
                task.getAppContext().put(
                                EngineConstants.APPCONTEXT_CLASSLOADER_KEY,
                                DocStoreServlet.class.getClassLoader());
                PDFRenderOption options = new PDFRenderOption();
                options.setOutputFormat(HTMLRenderOption.OUTPUT_FORMAT_PDF);
                options.setOutputStream(response.getOutputStream());
                task.setRenderOption(options);
                task.run();
                task.close();
        } catch (IOException e) {
                e.printStackTrace();
                return;
        }
        System.out.println("Finished");
}
```

Существует возможность вывода сформированного отчета в текстовый редактор Word, табличный редактор Excel и документы формата PDF.

На всех вкладках существует возможность поиска (в данном случае – фильтра). Осуществить его можно в зависимости от знаний предметной области пользователя: поиск по всем полям и поиск по одному из полей. Пример представлен на рисунках 2.25 – 2.26.

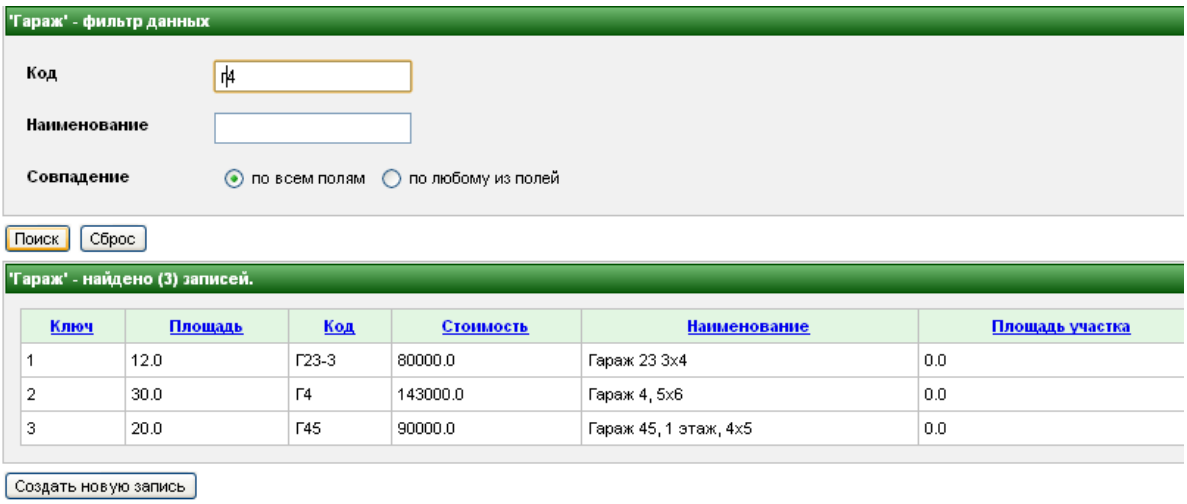

Рисунок 2.25 - Демонстрация использования фильтра по «коду гаража»

Рисунок 2.26 - Демонстрация результата после использования фильтра

Таким образом, рассмотрены все основные формы разработанной системы, необходимые для полноценной функциональности.

2.8. Тестирование разработанной CRM-системы

Тестирование разработанной системы производилось на базе гаражно-строительного предприятия ПЭГК № 74.

Существуют различные виды тестирования. Для данной системы было проведено два: функциональное тестирование и тестирование базы данных [13]. Цель функциональных тестов состоит в том, чтобы проверить

соответствие разработанных графических компонентов установленным требованиям.

Разработанная система на отлично прошла автоматизированное тестирование: все необходимые и задуманные функции работают исправно. Ручное тестирование также прошло успешно: при вводе данных некорректные записи подчеркиваются красным цветом и при нажатии на кнопку «Сохранить», система вначале проверяет логичность введенного. Благодаря реализации модулей справочников система подсказывает пользователю, что надо выбрать (например, «тип льгот» или «тип счетчика» и т.д.). Конечно, ввод грамматических ошибок допустим при вводе данных, но это – человеческий фактор.

Цель тестирования базы данных: убедиться в надежности методов доступа к базам данных, в их правильном исполнении, без нарушения целостности. По всем критериям данного тестирования система успешна и полностью функциональна. Это легко проверить, зайдя в любую из сформированных таблиц: вывод информации там основан на правильно структурированной базе данных и проектированных запросах.

Таким образом, было показано, что разработанная система не содержит ошибок и может быть использована в деятельности организации.

Вывод по главе: для проектирования компонентов системы были построены концептуальная, логическая и физическая модели данных, позволяющие более полно оценить специфику моделируемой предметной области. Выбран комплекс технических и программных средств реализации. Разработана автоматизированная система управления хозяйственной деятельностью гаражно-строительного предприятия.

Глава 3 Обоснование экономической эффективности CRM-системы

Под экономической эффективностью внедрения разработки понимают совокупность стоимостных, натуральных и качественных показателей, которые позволяют судить о целесообразности внедрения данной разработки.

Мероприятия по внедрению автоматизированной системы управления в гаражно-строительном предприятии, предназначенной для повышения финансовой точности работы с членами организации, ведения полной базы собственников и территориальных владений, снижения затрат рабочего времени, облегчения труда сотрудников офиса и увеличения оперативности получения информации связаны со значительными материальными затратами на разработку и функционирование системы.

Эффект, получаемый автоматизации с использованием CRM-систем позволяет спрогнозировать и описать экономические (измеримые) и организационные (неизмеримые) показатели улучшения бизнес-процессов предприятия. К измеримым экономическим прогнозируемым показателям эффективности внедрения автоматизированной системы взаимоотношениями с клиентами относятся:

- увеличение места для хранения документов, вследствие этого – уменьшение затрат на хранение;
- непроизводственные издержки уменьшаются, т.к. сокращаются затратына копирование, на доставку информации в печатном виде, на оборудование и на бумагу;
- скорость обработки информации увеличивается;
- рабочее время менеджеров используется рационально.

Качественными организационными прогнозируемыми показателями эффективности внедрения CRM-системы являются:

- оптимизирование рабочих процессов и регламентов компании;
- уровень прозрачности управления организацией увеличивается;
- повышение уровня доступности к информации сотрудникам;
- обеспечение надежной информационной безопасности;

- улучшение контроля над рабочими процессами;
- улучшение процесса поиска информации, увеличение его скорости;
- эффективность и качество организации работы сотрудников увеличивается.

Пользователями данной системы являются сотрудники офиса гаражно-строительного предприятия (в первую очередь: администратор, бухгалтер, кассир). В процессе проектирования будут задействованы: заместитель руководителя, бухгалтер, программист (в рамках данного предприятия - администратор). В калькуляцию себестоимости разработки ПО включаются следующие статьи затрат: основная зарплата; дополнительная зарплата; единый социальный налог; прочие прямые расходы; накладные расходы.

Трудовая неделя длится 5 дней, из этого следует, что месячный оклад для каждого участника проекта определен из расчета оплаты 23 рабочих дней (таблица 3.1).

Таблица 3.1 - Основная заработная плата исполнителей работ, проектный вариант

№	**Этап**	**Исполнители**	**Время работы, час**	**Часовая ставка, руб**	**Размер зарплаты, руб.**
1	Подготовительный	Заместительруководителя	2	80	160
		Бухгалтер	1	75,5	75,5
		Программист	20	65,5	1310
2	Основной	Программист	68	65,5	4454
3	Тестирование проекта	Программист	8	65,5	524
ИТОГО					**6523,5**

Таким образом, размер основной заработной платы (ОЗП) составляет 6523,5 руб.

Дополнительная заработная плата (ДЗП) штатным сотрудникам предприятия составляет 10% от ОЗП, что составляет 6523,5*0.1=652,35 руб.

Фонд оплаты труда (ФОТ) равен сумме основной и дополнительной заработных плат, что составляет 6523,5+652,35=7175,85 руб.

Отчисления в единый социальный налог (ЕСН) равны 26,3% от фонда оплаты труда, что составляет 7175,85*0.263= 1887 руб.

Накладные расходы составляют (НР) 40 % от фонда оплаты труда, что равно 7175,85*0.4=2870,34 руб.

Прочие прямые расходы (ППР) состоят из расходов на обслуживание ЭВМ, платы за потребляемую электроэнергию, платы за доступ в Интернет.

Для реализации проекта необходимо 310 часов машинного времени (все этапы выполнялись на компьютерах), себестоимость 1 часа машинного времени – 6 рублей, потребляемая мощность 0,35 кВт, стоимость 1 кВт электроэнергии составляет 2,13 руб.

Тогда расходы на электроэнергию и машинное время составят: 310*6+310* 0,35 кВт * 2,13 =2091 руб.

Стоимость 1 часа доступа в Интернет равна 12 руб. Интернет используется для консультаций при разработке программы. При условии пользования сетью Интернет 4 часа в день, получим следующие расходы: 4*4*12 = 192 руб.Следовательно, ППР составляют 2091+192=2283 руб.

Сводим полученные результаты в таблице 3.2.

Таблица 3.2 - Расчет себестоимости внедрения проектного варианта

Статьи затрат	**Сумма затрат (руб.)**	**Удельный вес, %**
1. Основная зарплата	6523,5	45,9
2. Дополнительная зарплата (10 % от основной)	652,35	4,6
3. Единый социальный налог	1887	13,3
4. Прочие прямые затраты	2283	16
5. Накладные расходы	2870,34	20,2
ИТОГО	**14216,19**	**100**

Таким образом, себестоимость разработки программы в проектном варианте равна $С_{пр}$=14216,19 руб.

В компаниях, занимающихся разработкой и сопровождением ПО, принято оплачивать работу программистов, исходя из количества

затраченного на разработку времени в часах и стоимости нормочаса специалиста.

Используем для расчетов среднюю стоимость нормочаса специалистов компании «1С» (таблица 3.3).

Таблица 3.3 - Основная оплата труда исполнителей работ

Этапы	**Исполнители**	**Время работы, час**	**Часовая ставка, руб.**	**Размер зарплаты, руб.**
1. Подготовительный	Менеджер высшего звена	6	67,5	405
	Консультант по внедрению	36	157,5	5670
2. Основной	Консультант по внедрению	46	157,5	787,5
3. Основной	Программист	56	135	7560
	Системный инженер	25	135	3375
	Консультант по внедрению	200	157,5	31500
ИТОГО				**55597,5**

Так как рассматривается разработка и внедрение системы приглашенными специалистами, с заранее заготовленными программными пакетами, то остается просчитать только прочие прямые расходы (ППР). Они состоят из расходов на обслуживание ЭВМ и платы за потребляемую электроэнергию. Для установки проекта необходимо 12 часов машинного времени, себестоимость 1 часа машинного времени – 6 рублей, потребляемая мощность 0,35 кВт, стоимость 1 кВт электроэнергии составляет 2,13 руб. Тогда расходы на электроэнергию и машинное время составят: 12*6+12* 0,35 кВт * 2,13 =81 руб.

Сводим полученные результаты в таблице 3.4.

Таблица 3.4 - Расчет себестоимости внедрения базового варианта

Статьи затрат	Сумма затрат (руб.)	Удельный вес, %
1. Затраты на оплату труда	55597,5	71,4
4. Прочие прямые затраты	81	0,1
5. Накладные расходы	22239	28,5
ИТОГО	**77917,5**	**100**

Себестоимость разработки программы на условиях привлечения сторонних специалистов равна $С_{баз}$=77917,5 руб.

Формируем таблицу показателей эффективности (таблица 3.5)

Таблица 3.5 - Показатели эффективности от внедрения проекта автоматизации

Затраты		**Абсолютное изменение затрат**	**Коэфф-т изменения затрат**	**Индекс изменения затрат**
сравнительный вариант	**проектный вариант**			
$С_{баз}$ (руб.)	$С_{пр}$ (руб.)	$\Delta С = С_{баз} - С_{пр}$ (руб.)	$К_С = \Delta С / С_{баз} \times 100\%$	$У_С = С_{баз} / С_{пр}$
55597,5	**14216,19**	**41381,31**	**74%**	**3.91**

Сводные данные по экономическому эффекту от внедрения автоматизированной системы приведены на рисунке 3.1.

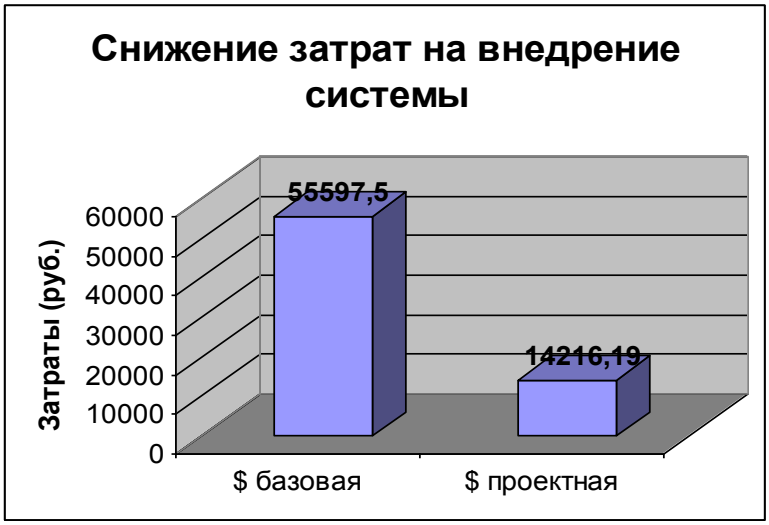

Рисунок 3.1 - Отражение разности стоимостных затрат при внедрении проекта в работу

Целесообразно также рассчитать срок окупаемости затрат на внедрение проекта:

$Т_{ок} = К_П / \Delta С$,

абсолютное снижение стоимостных затрат $\Delta С$ =41391 руб.

где $К_П$ - затраты на проектирование и внедрение системы $К_П$ =14216 руб.

$T_{ок}$ = 14216/41391= 0,3 года ~ 4 месяца

Таким образом, проведенная оценка эффективности показала, что система работает и дает определенную выгоду компании по сохранению и увеличению численности клиентов.

Вывод по главе: сравнение себестоимостей базового и проектного вариантов разработки элементов CRM-системы подтвердило целесообразность собственной разработки, в рамках которой выполнена настоящая бакалаврская работа.

Заключение

В ходе выполнения бакалаврской работы была разработана автоматизированная CRM-система управления хозяйственной деятельностью гаражно-строительного предприятия на базе ГСК № 17 «Весна».

Была подробно исследована предметная область, в которой были определены требования к функциональным характеристикам работы и выделены основные задачи, подлежащие автоматизации. Были изучены бизнес-процессы гаражно-строительного предприятия, дано описание объекта автоматизации, осуществлён анализ методов проектирования и сформулированы требования к программному обеспечению. Для проектирования компонентов системы были построены концептуальная, логическая и физическая модели данных, которые позволяют более полно оценить специфику моделируемой предметной области и избежать возможных ошибок на стадии проектирования схемы реляционной базы данных.

Использование CRM-системы позволяет:

- вести учет зарегистрированных собственников гаражей;
- сократить время на ведение данных по задолженностям и платежам;
- снизить затраты рабочего времени, облегчить труд сотрудников;
- вести учет состояния льготной группы предприятия;
- увеличить оперативность получения информации.

Автоматизированная система управления взаимоотношениями с клиентами гаражно-строительного предприятия разработана в соответствии со всеми требованиями заказчика.

Применяя в своей работе CRM-систему, в компании значительно вырастает эффективность работы с клиентами. При помощи CRM-системы компания получает полную информацию о своих клиентах, лучше понимает их желания и потребности, более детально позволяет изучить целевую аудиторию. Такая система упорядочивает информацию о клиентах.

Список используемой литературы

1. Быков С.Ю. Методы моделирования бизнес-процессов / С.Ю. Быков. - Компания ТЕМ consulting, 2016. - 16с.

2. Варзунов А.В. Анализ и управление бизнес-процессами [Электронный ресурс]: учеб.пособие / А.В. Варзунов, Е.К. Торосян, Л.П. Сажнева. – Санкт-Петербург: Университет ИТМО, 2016. – 114 с.

3. Грекул В.И. Проектирование информационных систем : учебник и практикум для академического бакалавриата / В.И. Грекул, Н.Л. Коровкина, Г.А. Левочкина. – Москва : Издательство Юрайт, 2019. – 385 с.

4. Григорьев М.В. Проектирование информационных систем : учеб. пособие для вузов / М.В. Григорьев, И.И. Григорьева. – Москва : Издательство Юрайт, 2018. – 318 с.

5. Золотов С.Ю. Проектирование информационных систем [Электронный ресурс] :учеб.пособие / С Ю. Золотов ; Томский гос. ун-т систем управления и радиоэлектроники. - Томск : Эль Учебное пособие Контент, 2013. - 86 с.

6. Проектирование информационных систем : учебник и практикум для академического бакалавриата / Д.В. Чистов, П.П. Мельников, А.В. Золотарюк, Н. Б. Ничепорук ; под общ. ред. Д. В. Чистова. – Москва : Издательство Юрайт, 2018. – 258 с.

7. Репин В.В. Бизнес-процессы. Моделирование, внедрение, управление / В.В. Репин. - М.: Манн, Иванов и Фербер, 2013. - 512 с.

8. Ротер М. Учитесь видеть бизнес-процессы: Построение карт потоков создания ценности. 4-е изд. / М. Ротер. - М.: Альпина Паблишер, 2015. - 136 с.

9. Самуйлов К.Е. Основы формальных методов описания бизнес-процессов. Учебное пособие / К.Е. Самуйлов, Н.В. Серебренникова, А.В. Чукарин – Москва: РУНД, 2013- 130с.

10. Фаулер М. UML. Основы, 3-е издание / М. Фаулер. – СПб.: Символ-Плюс, 2016. – 192с.

11. IDEF0: функциональное моделирование деловых процессов [Электронный ресурс] // АИС: Академия информационных систем. Учебный центр. URL: http://www.infosystem.ru/designing/methodology/sadt/sadt_for_bp.html (дата обращения: 26.06.2019)

12. Глоссарий процессного управления [Электронный ресурс] // ПитерСофт: современные технологии управления бизнесом. URL: http://piter-soft.ru/automation/more/glossary/process/as-is-model/ (дата обращения: 11.07.2019)

13. Диаграмма вариантов использования как концептуальное представление бизнес-системы в процессе ее разработки [Электронный ресурс] // НОУ Интуит. URL: https://www.intuit.ru/studies/courses/32/32/lecture/1004 (дата обращения: 12.07.2019)

14. Диаграмма классов: крупным планом [Электронный ресурс] // НОУ Интуит. URL: https://www.intuit.ru/studies/courses/1007/229/lecture/5956 (дата обращения: 26.06.2019)

15. Логическое моделирование [Электронный ресурс]. – Режим доступа: https://www.intuit.ru/studies/courses/3440/682/lecture/14036 (дата обращения: 21.06.2019)

16. Моделирование бизнес-процессов: методика, нотация, инструмент [Электронный ресурс]. – Режим доступа: http://emirs.miet.ru/oroks-miet/upload/ftp/pub/2015/11_0/563778e6738ac/Lecture2.pdf (дата обращения: 12.06.2019)

17. Построение диаграммы прецедентов [Электронный ресурс]. –Режим доступа: http://life-prog.ru/1_16788_ postroenie-diagrammi-pretsedentov.html (дата обращения: 21.06.2019)

18. Программа компьютерного моделирования BPwin [Электронный ресурс]. – Режим доступа: http://bourabai.kz/cm/bpwin.htm (дата обращения: 6.07.2019)

19. Функционально-ориентированные и объектно-ориентированные методологии описания предметной области [Электронный ресурс]. // НОУ Интуит. URL: https://www.intuit.ru/studies/courses/2195/55/lecture/1628?page=3 (дата обращения: 26.06.2019)

20. Brenda J. ,Saurabh S., Shevat A.: Designing Web APIs : Building APIs That Developers Love. 2018., – 200c.

21. Djirdeh H., Murray N., Lerner A.: Fullstack Vue: The Complete Guide to Vue.js., 2018. – 442c.

22. G. Heineman, G. Pollice and S. Selkow Algorithms in a Nutshell 2E [Текст]: O'Reilly Media 2015. – 435 c.: - Sorting Algorithms: c. 81 – 87. – Searching: c. 120 – 121. - ISBN: 063-6-920-03288-5

23. Introduction to C++/CLI Programming [Электронныйресурс]: Ajay Yadav. — Электрон. текстовыеданные. — Codeguru 2015. — Режимдоступа: https://www.codeguru.com/cpp/cpp/introduction-to-ccli-programming.html.

24. Raman R., Dewailly L.: Building RESTful Web Services with Spring 5 - Second Edition: Leverage the power of Spring 5.0, Java SE 9, and Spring Boot 2.0, 2018. — 228 p

Приложение А

Фрагмент программного кода

РедактированиясущностиСчетчик:

```
<h:formid="counterDevice" styleClass="edit">
<rich:panel>
        <f:facet name="header">
                'Счетчик' - #{counterDeviceHome.managed ? 'редактирование данных' : 'добавление данных'}
        </f:facet>
<s:decorate id="nameField" template="layout/edit.xhtml">
<ui:define name="label">Наименование</ui:define>
<h:inputText id="name"
 required="true" size="45" maxlength="45"
                value="#{counterDeviceHome.instance.name}">
<a:support event="onblur" reRender="nameField" bypassUpdates="true" ajaxSingle="true"/>
</h:inputText>
</s:decorate>


<s:decorate id="nextInspectionField" template="layout/edit.xhtml">
<ui:define name="label">Датаследующейпроверки</ui:define>
<rich:calendar id="nextInspection"
                value="#{counterDeviceHome.instance.nextInspection}" datePattern="MM/dd/yyyy" />
</s:decorate>
<s:decorate id="sealDateField" template="layout/edit.xhtml">
<ui:define name="label">Датаопломбирования</ui:define>
Используемоткрывающийсякалендарик<rich:calendar id="sealDate"  required="true"
value="#{counterDeviceHome.instance.sealDate}"                  datePattern="dd/MM/yyyy"
/>форматвремени
</s:decorate>
<div style="clear:both">
<spanclass="required">*</span>
          обязательно для заполнения
</div>
</rich:panel>
<div class="actionButtons">
<h:commandButton id="save"
                 value="Сохранить"
                action="#{counterDeviceHome.persist}" создаемсущность
               disabled="#{!counterDeviceHome.wired}"еслизаполненывсетребуемыеполя
```

rendered="#{!counterDeviceHome.managed}"/>эта кнопка будет показана если сущность еще не существует, иначе будем использовать метод сохранения

<h:commandButton id="update"

value="Сохранить"

action="#{counterDeviceHome.update}" методсохранения

rendered="#{counterDeviceHome.managed}"/>

<h:commandButton id="delete"

value="Удалить"

action="#{counterDeviceHome.remove}"удаляем

immediate="true" при удалении не делаем проверку корректности изменений в сущности, так как она все равно ужалиться

rendered="#{counterDeviceHome.managed}"/> кнопка доступна только если есть сущность и следовательно есть что удалять

<s:button id="cancelEdit"

value="Отмена"

propagation="end"

view="/CounterDevice.xhtml"

rendered="#{counterDeviceHome.managed}"/>

<s:button id="cancelAdd"

value="Отмена"

propagation="end"

view="/#{empty counterDeviceFrom ? 'CounterDeviceList' :counterDeviceFrom}.xhtml"

rendered="#{!counterDeviceHome.managed}"/>

* *

</h:form>

Добавление нового счетчика:

* *

<s:button

value="Добавитьзапись"

view="/CounterDeviceEdit.xhtml">

<f:param name="garageId"

value="#{garageHome.instance.id}"/>

<f:param name="CounterDeviceFrom" value="Garage"/>

</s:button>

* *

Код, отображающий список результатов запроса:

<rich:panel>

```
<f:facet name="header">'Счетчик' - найдено (#{empty counterDeviceList.resultList ? 0 :
(counterDeviceList.paginated ? counterDeviceList.resultCount : counterDeviceList.resultList.size)})
записей.</f:facet>
<div class="results" id="counterDeviceList">
<h:outputTextvalue="Поиск не дал результатов."
rendered="#{empty counterDeviceList.resultList}"/>
<rich:dataTable id="counterDeviceList"
              var="_counterDevice"
            value="#{counterDeviceList.resultList}"
          rendered="#{not empty counterDeviceList.resultList}">
            ...
<h:column>
<f:facet name="header">
<ui:includesrc="layout/sort.xhtml">
<ui:param name="entityList" value="#{counterDeviceList}"/>
<ui:param name="propertyLabel" value="Наименование"/>
<ui:param name="propertyPath" value="counterDevice.name"/>
</ui:include>
</f:facet>
<h:outputText value="#{_counterDevice.name}"/>
</h:column>
    ...
<h:column>
<f:facet name="header">
<ui:includesrc="layout/sort.xhtml">
<ui:param name="entityList" value="#{counterDeviceList}"/>
<ui:param name="propertyLabel" value="Датаопломбирования"/>
<ui:param name="propertyPath" value="counterDevice.sealDate"/>
</ui:include>
</f:facet>
<h:outputText value="#{_counterDevice.sealDate}">
<s:convertDateTime type="date" dateStyle="short"/>
</h:outputText>
</h:column>
</rich:dataTable>
```

Управление списком счетчиков:

```
@Name("counterDeviceList")
public class CounterDeviceList extends EntityQuery<CounterDevice> {
```

```
        private static final String EJBQL = "select counterDevice from CounterDevicecounterDevice";

        private static final String[] RESTRICTIONS = {"lower(counterDevice.name) like
lower(concat(#{counterDeviceList.counterDevice.name},'%'))",};

        private CounterDevicecounterDevice = new CounterDevice();

        public CounterDeviceList() {
                setEjbql(EJBQL);
                setRestrictionExpressionStrings(Arrays.asList(RESTRICTIONS));
                setMaxResults(25);
        }

        public CounterDevicegetCounterDevice() {
                return counterDevice;
        }
}
```

Код, генерирующий PDF отчеты

```
<servlet>
<servlet-class>org.boxproject.birt.DocStoreServlet</servlet-class>
</servlet>
<servlet-mapping>
<servlet-name>PDF Document Store Servlet</servlet-name>
<url-pattern>*.pdf</url-pattern>
</servlet-mapping>

@Override
protected void doGet(final HttpServletRequest request,
        final HttpServletResponse response) throws ServletException,
        IOException {
                System.out.println("starting doget method");
                new ContextualHttpServletRequest(request) {
                        @Override
                        public void process() throws ServletException, IOException {
                                try {
                                        executeReport(request, response);
                                } catch (EngineException e) {
                                        log("executeReport error", e);
```

```
                                }
        }
        }.run();
}
void executeReport(HttpServletRequest request, HttpServletResponse response)
        throws EngineException {
        response.setContentType("application/pdf");
        response.setHeader("Content-Disposition", "inline; filename=test.pdf");

        String reportName = request.getServletPath();
        ServletContextsc = request.getSession().getServletContext();
        this.engine = BirtEngine.getBirtEngine(sc);
        IReportRunnable design;
        try {
                design = engine.openReportDesign(sc.getRealPath("./")
                                + reportName.substring(0, reportName.length() - 4)
                                + ".rptdesign");
                IRunAndRenderTask task = engine.createRunAndRenderTask(design);
                task.getAppContext().put(
                                EngineConstants.APPCONTEXT_CLASSLOADER_KEY,
                                DocStoreServlet.class.getClassLoader());
                PDFRenderOption options = new PDFRenderOption();
                options.setOutputFormat(HTMLRenderOption.OUTPUT_FORMAT_PDF);
                options.setOutputStream(response.getOutputStream());
                task.setRenderOption(options);
                task.run();
                task.close();
        } catch (IOException e) {
                e.printStackTrace();
                return;
        }
        System.out.println("Finished");
}
```

Код, реализующийрегистрациюсобственника

```
private static final long serialVersionUID = 1L;
        @Logger private Log log;
   @In EntityManagerentityManager;
   @In(create=true,required=false) /*@Out(required=false)*/
```

Продолжение приложения А

```
PersonHomepersonHome;
   @In(create=true,required=false)/* @Out*/
PersAccountHomepersAccountHome;
   @In(create=true,required=false) /*@Out*/
GarageHomegarageHome;

   //@In(create=true) //@Out
booleanpersonAlreadyExists = false;

   public booleangetPersonAlreadyExists() {
         log.info("trying to get");
               return personAlreadyExists;
         }
         public void setPersonAlreadyExists(booleanpersonAlreadyExists) {
               log.info("trying to set");
               this.personAlreadyExists = personAlreadyExists;
         }
         @Begin
   public String begin()
   {
log.info("beginning conversation");
log.info("personAlreadyExists #0", personAlreadyExists);
      if (personAlreadyExists)
         return "select";
      else return "createNewPerson";
   }
       public void load()
   {
         log.info("!!!start loading");
         if (personHome.isManaged())
         {
               personHome.load();
         }
         log.info("!!!end loading");
   }

   public void loadGar()
   {
               if (garageHome.isManaged())
```

Продолжение приложения А

```
        {
                garageHome.load();
        }           }
    public String selectGarage()
{
        return "select";
}

public void persistPerson()
{
        if (!personHome.isManaged())
        {
                /*String outcome = personHome.persist();*/
                try {
                        Person person = personHome.getInstance();
                        entityManager.persist(person);
                        personHome.setPersonIdentdoc(person.getIdentdoc());
                        log.info("Person persisted");
                } catch (Exception e) {
                        log.info("Error while persisting  Person");
                                e.printStackTrace();
                        }

        }
}
  @Transactional(TransactionPropagationType.REQUIRED)
public void register()
{
        persistPerson();
        log.info("wiring");
        persAccountHome.wire();
        log.info("persisting");
        persAccountHome.persist();
}
    @End
public String end()
{
        try{
                register();
```

```
        } catch (Exception e) {

                        log.info("Error while persisting  Account");
                e.printStackTrace();
                return "fuck";
                }
log.info("ending conversation");
      return "home";
  }}
```

Код, реализующийприемплатежей

```
public class Payer implements Serializable
{
   private static final long serialVersionUID = 1L;

        private static final String OWNER_TYPE = "P’";

        @Logger private Log log;
        @In EntityManagerentityManager;
        @In(create=true,required=false)
GarageHomegarageHome;
        @In(create=true,required=false)
BudgetHomebudgetHome;
        @In(create=true,required=false)
PayHomepayHome;

        @Out
        List<Budget> budgets = new LinkedList<Budget>();
        @Out(required = false)
        double toPay;
        @Out(required = false)
        double debtToPay;

   @Begin
   public String begin()
   {
log.info("beginning conversation");
    return "begin";
   }
```

```
   public void loadGar()
   {
         log.info("!!!start loading");
         if (garageHome.isManaged())
         {
                  garageHome.load();
         }
         log.info("!!!end loading");
   }
      public String showSum()
   {
         return "ok";
   }
      public void calculatePay()
   {
         List<PersAccount> accounts = garageHome.getPersAccounts();
         PersAccount owner = null;
         for (PersAccountaccount : accounts)
         {
                  if(account.getWithdrawalDate() == null &&account.getOwnershipTypeRef().getCode()
== OWNER_TYPE)
                  {
                           owner = account;
                           break;
                  }         }
         String query = "select budget from Budget budget" +
                           " where budget.persAccount = :owner" +
                           " and (budget.payed = null or budget.payed = false)" +
                           " order by budget.id.period desc";
         budgets = entityManager.createQuery(query).setParameter("owner", owner).getResultList();
         toPay = 0.0;
         debtToPay = 0.0;
         Date now = new Date();
         for (Budget budget : budgets)
         {
                  int delay = 0;
                  if (budget.getPayBefore().before(now))
                  {
                           delay = getNumberOfDaysBetween(budget.getPayBefore(), now);
```

```
                    budget.setDebt(delay*budget.getPenaltyFee());
                    debtToPay +=budget.getDebt();
                    toPay +=budget.getDebt();
                }
                toPay += budget.getCost();
        }
    }
    public int getNumberOfDaysBetween(Date startDay, Date endDate){
        GregorianCalendar start = new GregorianCalendar();
        GregorianCalendar end = new GregorianCalendar();
        if (startDay.before(endDate))
        {
                start.setTime(startDay);
                end.setTime(endDate);
        } else if (startDay.after(endDate))
        {
                start.setTime(endDate);
                end.setTime(startDay);
        } else {
                return 0;           }

        start.get(Calendar.DAY_OF_YEAR);
        end.get(Calendar.DAY_OF_YEAR);

        start.get(Calendar.YEAR);
        end.get(Calendar.YEAR);

        if (start.get(Calendar.YEAR) <end.get(Calendar.YEAR))
        {
                int numberOfDays = 0;
                while (!start.after(end))
                {
                        numberOfDays++;
                        start.add(Calendar.DAY_OF_YEAR, 1);
                }
                return numberOfDays;
        } else {
                return end.get(Calendar.DAY_OF_YEAR) - start.get(Calendar.DAY_OF_YEAR);
        }    }
```

Продолжение приложения А

```
    public String pay()
  {
        for (Budget budget : budgets)
        {
                budgetHome.newInstance(budget);
                budgetHome.getInstance().setPayed(true);

                payHome.newInstance();
                Pay pay = payHome.getInstance();
                pay.setBudget(budget);
                pay.getId().setPayDate(new Date());
                pay.setValue(budget.getCost()+budget.getDebt());
                payHome.persist();
                budgetHome.persist();
                        }
        return "print";
  }
    @End
  public String end()
  {
log.info("ending conversation");
    return "home";
} }
```