**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**



**МОСКОВСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

**ВЫСШАЯ ШКОЛА ПЕЧАТИ И МЕДИАИНДУСТРИИ**

***Институт Принтмедиа и информационных технологий***

***Кафедра Информатики и информационных технологий***

**направление подготовки**

**09.03.01 «Информатика и вычислительная техника»**

**КУРСОВАЯ РАБОТА**

**Дисциплина:** *Информационные технологии и системы в принтмедиа\_\_\_\_\_\_\_*\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Тема:** Разработка help desk системы для сервисной службы общежития

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Выполнил(а): студент(ка) группы** ДЦасБ 4 - 1

Кузин Дмитрий Сергеевич

(Фамилия И.О.)

**Дата, подпись** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  ***\_\_\_\_\_\_\_\_\_\_\_***

(Дата) (Подпись)

**Проверил:** Арсентьев Д.А. доцент к.т.н.

(Фамилия И.О., степень, звание)

**Дата, подпись** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  ***\_\_\_\_\_\_\_\_\_\_\_***

(Дата) (Подпись)

**Замечания: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Москва 2019 г**

**Реферат**

Курсовая работа содержит 37 страниц, 1 иллюстрацию.

Тема курсовой работы – “ Разработка help desk системы для сервисной службы общежития”

Целью курсовой работы является разработка концепции help desk системы для управления заявками на ремонт в общежитии.

Объектом исследования в данной работе являются аналоги этой системы и заявки.

Созданный проект может быть использован в следующих направлениях:

* Создание заявок на ремонт проживающими
* Распределение заявок между работниками
* Наглядное представление задачи исполнителю

Результатом выполнения проекта стала концепция веб-приложения для управления заявками в общежитии, а также изучение структуры баз данных.

В ходе работы были исследованы аналоги, разработаны требования к разрабатываемой системе, выбраны наиболее подходящие инструменты для разработки данного проекта и создан приятный интерфейс для работы пользователя.

**Оглавление**

[**Введение** 4](#_Toc6818551)

[**Глава 1. Анализ предметной области** 6](#_Toc6818552)

[**1.1** **Service desk системы** 6](#_Toc6818553)

[**1.2** **Программное обеспечение** 8](#_Toc6818554)

[**1.3** **Анализ имеющихся готовых решений на рынке** 10](#_Toc6818555)

[**1.4** **Принцип работы** 14](#_Toc6818556)

[**1.5** **Базы знаний** 17](#_Toc6818557)

[**1.6** **Программный агент** 23](#_Toc6818558)

[**1.7** **Веб-система** 27](#_Toc6818559)

[**Глава 2. Разработка проектного решения** 29](#_Toc6818560)

[**2.1** **Основные положения разрабатываемой системы** 29](#_Toc6818561)

[**2.2** **Графический интерфейс** 29](#_Toc6818562)

[**2.3** **База данных** 30](#_Toc6818563)

[**2.4** **База знаний** 31](#_Toc6818564)

[**2.5** **Управление пользователями** 32](#_Toc6818565)

[**2.6** **Управление заявками** 32](#_Toc6818566)

[**2.7** **Обоснование выбора программных средств** 32](#_Toc6818567)

[**Заключение** 37](#_Toc6818568)

**Введение**

Сервисная служба является неотъемлемой частью общежития. Она отвечает за состояние таких необходимых для комфортного проживания студентов элементов как электричество, водопровод и мебель. От ее деятельности зависит престиж университета, поэтому очень важно добиться максимальной эффективности ее работы.

Сейчас, в век всеобщего внедрения информационных технологий, современное развитие IT-технологий значительно изменило их значение в организации бизнес-процессов любой̆ компании и ее внутренних, и внешних коммуникаций. Чаще всего следствием автоматизации деловых процессов компании выступают:

значительное повышение производительности труда и снижение трудозатрат;

повышение скорости выполнения задач, связанных с разного рода учетом;

снижение количества ошибок в документации, отчетах и т. д.

повышение эффективности учета, приводящее к увеличению рентабельности предприятия;

удобство осуществления деловых процессов и документооборота.

Каждый из перечисленных выше пунктов — весомый аргумент в пользу того, чтобы обратить внимание на возможность автоматизации бизнес-процессов компании. В их необходимости и эффективности которого убедились руководители большинства успешных компаний не только Европы и Америки, но также и России.

Мы уже привыкли к тому, что можем нажатием кнопки в смартфоне подать заявление, записаться к врачу или открыть банковский счет. Это позволяет нам экономить время на более важные дела, чем ожидание в очереди, телефонные звонки и поездки в отделения для решения тех или иных вопросов. Также не хочется тратить время на звонки в сервисную службу для проведения какого-либо ремонта и описывать случившуюся ситуацию, особенно без знания названия элементов, вышедших из строя, когда можно просто оставить заявку, прикрепив к ней изображение и получать информацию об изменениях в вашей заявке. На данный момент работа сервисной службы оставляет желать лучшего, так как нет никакой системы мотивации работников, заявки, отправленные в службу не хранятся и нет гарантии что конкретная заявка будет закрыта и про нее не забудут.

Разрабатываемая система поможет повысить эффективность работы сервисной службы за счет закрепления заявок за определенными работниками и дальнейшего подсчета выполненных заявок, что будет являться показателем эффективности персонала. Также эта системы повысит удобство работы с сервисной службой ее пользователям - они смогут оставлять свои заявки с компьютера или телефона, прикрепляя изображения, отслеживать статус выполнения и общаться с исполнителем при помощи онлайн-чата.

**Глава 1. Анализ предметной области**

* 1. **Service desk системы**

Существующий способ приема заявок способ учета связан с большой трудоемкостью, разрозненностью сведений, что с большей вероятностью ведет к их утере или неправильной интерпретации. На сегодняшний день невозможно получить сведения об общем количестве заявок, провести анализ основных причин возникновения проблемных вопросов у клиентов и проанализировать причины обращения.

Кроме того, в отчетный период необходимо составление аналитических отчетов, включающих в себя анализ работы за определенный период, что очень затруднительно. Поэтому встает вопрос о внедрении системы Service desk

Service desk в настоящее время является необходимой и одной из основных составляющих хорошего обслуживания.

По сути, она является центральным пунктом, в который сообщают о проблемах или неисправностях, и впоследствии решают и координируют. С более широкой точки зрения, это основная часть для сервисной службы, которая включает несколько ресурсов для решения проблем.

Система позволяет организации или центру поддержки отслеживать проблемы, которые имеют пользователи. Роль системы service desk - поддержка сотрудников внутри организации, где важно рлучить решение проблемы как как можно скорее.

Система service desk разработана, чтобы помочь организации эффективно общаться, организовывать, отслеживать и сообщать о проблемах. С помощью веб-системы справочной службы организация может иметь доступ к информации в любое время и в любом месте.

История программного обеспечения сервисной службы восходит к 20-му веку, когда компании полагались в основном на личное взаимодействие для решения проблем клиентов. Клиенты должны были посетить магазин или офис компании с продуктом, чтобы решить свои проблемы.

С изобретением телефона в 1876 году и телефонного коммутатора в 1890-х годах справочная служба предложила лучший подход. Клиенты смогли связаться со своей компанией и озвучить свои проблемы по телефонной сети. В эпоху 20-го века компании использовали в основном оборудование, такое как диктофоны, пишущие машинки и немые терминалы с доступом к мэйнфрейму, для решения проблем клиентов. Самое раннее использование компьютеров для обслуживания клиентов было сделано с помощью программного обеспечения мэйнфреймов. Клиенты отправляли бумажные формы или сообщали о своей проблеме по телефону агентам по обслуживанию клиентов, которые искали пути решения проблем.

В 1960-х годах компании начали создавать колл-центры, а также обучать персонал, чтобы получать и обрабатывать запросы клиентов организованно и эффективно. Это была эра интерактивного голосового ответа (IVR), которая стала большим стимулом для системы обслуживания клиентов по телефону. Позже настольные ПК и электронная почта значительно улучшили системы справочной службы. Клиенты могли сообщить о своих проблемах по электронной почте, минуя бумажные формы. Агенты службы поддержки могли также предоставлять обновления статуса и разрешения по электронной почте.

Между тем, фактическое внедрение систем справочной службы началось в 1980 году, когда Интернет был официально доступен для общего пользования. Многие компании начали аутсорсинг своего отдела обслуживания клиентов. Это привело к массовому использованию систем электронной почты и чатов в 1990-х годах. Эта новая разработка позволила нескольким американским компаниям передать свою справочную службу на аутсорсинг в такие недорогие страны, как Индия и Филиппины.

В любом случае настоящее изобретение программного обеспечения для службы поддержки появилось в 2000-х годах, когда компании начали использовать разнообразные пакеты программного обеспечения для решения вопросов обслуживания клиентов. Это привело к массовому производству различных видов программного обеспечения справочной службы в Интернете и во всем мире. В последнее время Интернет и сетевые системы делают программное обеспечение справочной службы более интерактивным и доступным для клиентов и агентов. Теперь клиент может легче отправлять и отслеживать свои проблемы.

Системы программного обеспечения для обслуживания клиентов и службы поддержки становятся все более популярными в последнее время. Согласно недавнему отчету, объем продаж программного обеспечения для управления взаимоотношениями с клиентами (CRM), который включает в себя программное обеспечение службы поддержки, по всему миру значительно возрос.

Service desk системы появились относительно недавно, но успели зарекомендовать себя как крайне полезный инструмент повышения эффективности.

* 1. **Программное обеспечение**

Под программным обеспечением с Service desk понимается компьютерная программа, которая позволяет операторам по обслуживанию клиентов отслеживать запросы пользователей и решать другие вопросы, связанные с обслуживанием клиентов. Именно это делает обслуживание клиентов эффективным и удобным.

Как правило, программное обеспечение службы поддержки является частью общей категории, называемой службой поддержки, которая включает управление активами и управление услугами. Часто оба термина используются взаимозаменяемо. Тем не менее, программное обеспечение службы поддержки относится к системе, которая обрабатывает запросы клиентов.

Типичная Service desk система может эффективно выполнять несколько функций. Она обеспечивает единую (или множественную) точку контакта для пользователей, чтобы получить помощь в устранении неполадок, получить ответы на вопросы и решить известные проблемы. Служба поддержки обычно управляет своими запросами с помощью программного обеспечения, такого как системы отслеживания проблем. Эти системы часто включают использование «локального средства отслеживания ошибок» (LBT). Эта система позволяет службе поддержки отслеживать и сортировать запросы пользователей с помощью уникального номера и часто может классифицировать проблемы по пользователям, компьютерным программам или аналогичным категориям. Многие приложения доступны для поддержки функции справочной службы. Некоторые предназначены для справочной службы уровня предприятия, а некоторые - для нужд департамента.

В середине 1990-х годов в исследовании Iain Middleton из Университета Роберта Гордона изучалась ценность автоматизированных систем организации. Он обнаружил, что ценность была получена не только от реагирования на проблемы пользователей, но и от уникальной позиции службы поддержки ежедневного общения с многочисленными клиентами или сотрудниками. Информация, полученная в таких областях, как технические проблемы, предпочтения пользователей и удовлетворенность, может быть полезна для планирования и разработки других подразделений информационных технологий.

Программное обеспечение справочной службы автоматизирует обслуживание клиентов различными способами. Обычно оно состоит как минимум из трех частей. К ним относятся управление заявками, пакет автоматизации и отчеты / оптимизация.

Программное обеспечение службы поддержки имеет контактную точку для клиентов, чтобы отправлять свои запросы, в систему заявок, которая отслеживает и организует проблемы для более быстрого решения. Оно также может иметь функцию, которая объединяет и организует запросы и ответы в базе знаний, такой как часто задаваемые вопросы или инструкции. Оно может вместить несколько точек соприкосновения, рабочая панель приборов и раздел аналитики. ПО также может иметь функцию, которая позволяет агентам поднимать проблемы на более высокий уровень.

Более продвинутые приложения службы поддержки включают онлайн-чат, аналитику, автоматизированные процессы, несколько каналов контактов, инструменты отчетности, инструменты совместной работы и функцию CRM.

* 1. **Анализ имеющихся готовых решений на рынке**

На данный момент имеется огромное множество готовых service desk систем. Здесь мы рассмотрим некоторые из них.

[HubSpot Help Desk Software](https://www.hubspot.com/products/service/help-desk?_ga=2.106333696.2062359231.1543500359-836228720.1531858988)

Стоимость: 24 000 руб/мес

HubSpot Service Hub особенно эффективен, когда он используется с другими продуктами HubSpot для продаж и CRM. Интегрируя автоматизацию со всеми вашими маркетинговыми данными и данными CRM, Service Hub помогает обеспечить комплексное обслуживание клиентов, которое помогает оставлять их счастливыми.

Этот инструмент включает в себя программное обеспечение для создания заявок, которое сочетает в себе диалоговую папку входящих сообщений, автоматизацию, и ботов для эффективной маршрутизации, назначения и реагирования на заявки, а также составления отчетов, помогая командам поддержки постоянно совершенствоваться и выполнять итерации для достижения лучших результатов. Инструменты HubSpot были разработаны для малого и среднего бизнеса, чтобы помочь крупным компаниям обеспечить исключительную поддержку клиентов.

[Zendesk Support](https://www.zendesk.com/support/pricing/#pricing)

Стоимость: от 300 руб/мес за каждого работника

Служба поддержки Zendesk - это программный пакет, который позволяет организации объединять различные взаимодействия с клиентами в одну доступную базу данных. Он имеет функции для эффективного рабочего процесса, включая веб-виджеты, возможность поиска в истории клиента и предопределенные ответы на заявки. Еще одна особенность - интегрированные опросы для оценки удовлетворенности клиентов в сочетании с аналитикой и инструментальными панелями производительности для отслеживания текущей производительности. Служба поддержки Zendesk также работает с другими компонентами семейства программного обеспечения Zendesk, такими как Zendesk Talk, которое является их колл-центром.

Для Zendesk существует пять основных платных уровней: Essential, Team, Professional, Enterprise и Elite. Как и ожидалось, это связано с увеличением как цены, так и возможностей. Тем не менее, существует бесплатная пробная версия для использования Zendesk, так что вы можете получить представление о том, какой реальный уровень вам может понадобиться для вашего бизнеса. Базовый тарифный план Essential оплачивается по 300 рублей за работника в месяц, позволяя использовать электронную почту, предварительно определенные бизнес-правила и историю взаимодействия. Профессиональный план среднего уровня стоит 3000 рублей за работника в месяц и позволяет настраивать бизнес-правила, панели мониторинга производительности, интеграции, многоязычный контент и опросы CSAT.

[Zoho Desk](https://www.zoho.com/desk/pricing.html)

Стоимость: бесплатно до 3 работников

Zoho Desk - это облачное программное обеспечение службы поддержки, которое ориентировано на понимание контекста. Он включает в себя такие функции, как расстановка приоритетов более важных или просроченных заявок, информационные панели для отслеживания метрик качества и поддержка для создания базы знаний для более простых проблем, которые можно обслуживать самостоятельно. Zoho Desk могут использовать агенты с мобильными приложениями для iOS и Android. Поддержка использования Zoho Desk осуществляется через ряд направлений, включая активные форумы пользователей, серию вебинаров, портал самообслуживания, руководство пользователя и блоги.

В отличие от своих конкурентов, сильной стороной Zoho Desk является их упрощенное число уровней, которое составляет всего три, причем нижняя часть представляет собой бесплатный план с разумным ограничением в три пользователя, а также наличие бесплатных 15-дневных пробных версий для других уровней. Их самый популярный план - средний профессиональный план, который включает в себя «Рейтинги удовлетворенности клиентов» и облачную телефонию по доступной цене 700 рублей за оператора в месяц, оплачиваемой ежегодно.

Даже самый верхний корпоративный план, который добавляет настраиваемые шаблоны заявок, контроль доступа на основе ролей, межотраслевые отчеты и вариант поддержки в режиме реального времени в чате, стоит 2500 рублей на оператора в месяц, что меньше, чем в большинстве других планов высшего уровня.

OTRS

Стоимость: бесплатно

Бесплатная open-source система заявок на Perl. Отличная интеграция с Email. Поддерживает различные платформы, БД, LDAP. Автоматизированы многие мелочи. Гибкая система отчетов, большие возможности поиска, управления пользователями. Возможна интеграция с уже имеющимися БД клиентов и сотрудников. Система легко раcширяема через дополнительные модули: база знаний/FAQ, календарь, файловый менеджер, ITSM и др.

Интерфейс явно отличается от других коммерческих help desk -сервисов – нельзя назвать интуитивным, но он удобен после должного изучения. Список заявок отображается как в виде очереди заявок, так и в виде отдельных заявок с их текущим статусом. Между двумя вариантами отображения необходимо переключаться. Доступно групповое редактирование заявок, а также блокировка заявки сотрудником, взявшим его в работу. При этом можно указать время обработки заблокированного заявки, по истечении которого заявка вновь становится доступна для других сотрудников (если взявший не успел ее закрыть).

Freshservice

Стоимость: бесплатно

Это очень популярная облачная платформа поддержки клиентов, основанная с целью предоставления компаниям всех размеров возможности предоставлять отличное оказание услуг. Она предоставляет ITIL «включай и работай», который соответствует передовым практикам без необходимости дорогостоящих консультаций или экспертных мнений.

Его основные функции включают в себя надежную систему создания заявок, доступную базу знаний, обнаружение активов и CMDB. Кроме того, программное обеспечение оснащено такими возможностями, как управление поставщиками, управление инцидентами, управление проблемами, управление изменениями и управление выпусками.

Основным модулем системы является управление инцидентами, разработанное таким образом, чтобы ни один клиентский запрос не проходил мимо. С помощью службы поддержки, доступной по нескольким каналам, вы можете автоматизировать все повседневные задачи и позволить своей команде сотрудничать, чтобы найти лучшее решение для каждого запроса.

Freshservice также проделает большую работу по анализу и решению проблем, прежде чем они окажут какое-либо влияние на производительность вашего бизнеса, поскольку она связывает внезапные инциденты с общими проблемами, исследует основную причину и предлагает ценные обходные пути, а также сохраняет все проблемы в известной форме. ошибка базы данных, пока вы не найдете постоянное решение.

Другим конкурентным преимуществом этой простой в использовании системы является планирование, отслеживание и откат изменений, отсутствие которых, как известно, вызывает примерно 80% всех проблем в области ИТ. С Freshservice в вашем инструментарии вы сможете правильно отслеживать изменения и ресурсы и, таким образом, сможете анализировать риск и потенциальное воздействие для каждого из них. Таким образом вы можете подготовить свой собственный план отката к непредвиденным и неожиданным изменениям и отправить запросы на одобрение CAB с той же платформы.

Наконец, что не менее важно, Freshservice размещается в облаке и требует практически нулевого обслуживания при сохранении безопасности данных и полной автоматизации системы. Гибкие тарифные планы масштабируются в соответствии с ростом и потребностями клиентов, при этом предоставляется бесплатный план (Sprout) для обслуживания групп, в которых может быть до 3 агентов.

В результате изучения имеющихся программных решений стало ясно, что все эти системы ориентированы на работы исключительно IT-отдела, а не сервисной службы, поэтому они не подходят для решения нашей задачи.

* 1. **Принцип работы**

Для разработки собственной service desk системы необходимо сначала понять как они работают.

Help desk функционирует в качестве точки доступа для предоставления пользователю консультаций, информации и действий по устранению неполадок. Он также выступает в качестве посредника в сборе и анализе данных, которые могут трансформироваться в более активную роль. Центральное компьютерное и телекоммуникационное агентство подчеркивает, что в обязанности Help desk входит поддержка первой линии инцидента в случае сбоя в ИТ, повседневная связь между ИТ-отделом и пользователем, поддержка бизнес-систем и формирование отчетов о качестве обслуживания. Многие ссылаются на важность Help desk для оказания технической помощи пользователям в случае сбоя оборудования или программного обеспечения, связанного с компьютером. Тем самым, это первое место контакта для пользователя по всем вопросам поддержки ИТ. Как правило, связанные с ИТ вопросы поддержки включают в себя:

1) Запрос на использование программного обеспечения / приложений / аппаратных средств / устройств передачи данных / телекоммуникационных устройств

2) установка программного обеспечения / оборудования / устройства передачи данных / телекоммуникационного устройства

3) ремонт, устранение неисправностей и настройка

4) настройка учетной записи пользователя

5) проблема безопасности

6) интернет / поддержка по электронной почте

7) услуга покупки товара

8) управление запасами

9) обучение

Децентрализованная модель Help desk была очень популярна в 1980-х годах. В этой модели организация часто имеет более одного центра управления, где различные центры были созданы департаментами, филиалами и рабочими группами ИТ. В компаниях где было девять разных центров, пользователь должен был определить, к какому обращаться, в зависимости от того, где была проблема, в чем была проблема и когда возникла проблема. Децентрализованная модель разделяет убеждение, что различные вопросы поддержки можно легко отнести к связанным центрам, чтобы можно было своевременно получить ответ. Эта концепция тогда хорошо работала, потому что компьютерная составляющая была простой. В то время структура компьютеров была слабой и состояла только из терминалов, мэйнфреймов, принтеров и простых автономных прикладных программ. По мере усложнения ИТ-инфраструктуры, систем в масштабах всей организации с большим количеством взаимосвязанного аппаратного и программного обеспечения классификация проблемных областей стала менее четкой. В такой ситуации пользователи путались с несколькими help desk системами, и их часто требовали переносить с одной системы на другую, прежде чем получить правильное решение. К счастью, развитие help desk продолжалось. Чтобы восстановить свою репутацию, организации начали применять централизованную модель help desk. Идея состоит в том, чтобы объединить различные help desk в один, и пользователю нужно запомнить только один контактный номер для всех запросов, связанных с ИТ, что делает help desk первой и единственной точкой контакта. Эта модель не только консолидирует контактную точку, она также помогает консолидировать и стандартизировать различные политики и процедуры поддержки, соглашения об уровне обслуживания, а также инструменты поддержки. Другие стимулы для этой модели включают более эффективное распределение ресурсов, повышение скорости разрешения и связь между подразделениями.

В настоящее время некоторые глобальные корпорации с офисами, расположенными по всему миру, реализуют другую концепцию, называемую распределенной или виртуальной моделью help desk. Хотя эта модель поддерживает несколько физических местоположений, пользователь все же может связаться с системой, используя один контактный номер с помощью современной технологии маршрутизации вызовов. Таким образом, она может работать 24 часа в сутки, 7 дней в неделю, независимо от местоположения. Например, Morgan Stanley, один из крупнейших инвестиционных банков в мире, состоит из четырех центров в разных местах (США, Англия, Япония и Гонконг), которые позволяют им предоставлять услуги в масштабах всего предприятия. В настоящее время help desk классифицируются как внутренние и внешние. Первые поддерживают только пользователей организации, тогда как вторые поддерживает внешних клиентов и обычно устанавливается поставщиками программного и аппаратного обеспечения или интернет-провайдерами.

Появление системы управления Help desk является важным шагом для её автоматизации. Обычно в неё входит четыре основные функции, такие как запись информации о вызовах, изменение статусов заявок, хранение заявок, а также составление отчетов. Функция регистрации звонков позволяет оператору записывать личные данные пользователя, подробности, и описание проблемы в функции хранения заявки или репозитория заявок. Персонал всегда ссылается на эти записи. Как только пользователь звонит, чтобы запросить техническую поддержку, оператор должен открыть заявку, заполнить детали, а затем сохранить её в базу хранения. Если проблема требует дальнейшей эскалации, оператор может переслать или назначить заявку конкретному аналитику или рабочей группе. Аналитик или рабочая группа, которая обслуживает заявку отвечает за обновление всех последующих действий, в ходе выполнения разрешения, в хранилище билетов. Когда проблема будет решена, заявка будет закрыта.

Функция отчетности позволяет супервизору или менеджеру создавать отчеты с разными параметры, такие как высокоприоритетная заявка, непогашенная, тип проблемы и т. д. Отчетность - очень мощная функция для управления ежедневными операциями Help desk. Например, если есть слишком много срочных заявок, ожидающих решения, это может быть указанием нанять больше персонала. В другом случае, если есть огромное количество заявок, связанных с программной или аппаратной проблемой, тогда может потребоваться тщательная проверка соответствующей системы.

* 1. **Базы знаний**

Еще в середине 1980-х годов, инструменты и методы управления, такие как общее качество управления, сокращение и реинжиниринг бизнес-процессов были разработаны западными компаниями для того, чтобы помочь восстановить долю рынка автомобильной и электронной техники индустрия которой была захвачена японскими компаниями. Тем не менее, оба ввод и улучшение являются краткосрочными, потому что эти подходы к решению являются общими и легко доступны для всех конкурирующих компаний. Когда подход показывает себя успешным, конкурирующая компания дублирует и принимает ту же практику. Практика сокращения, аутсорсинга и реинжиниринга бизнес-процессов, которые направлены на оптимизацию процессов, а также на экономию затрат и времени, привели к потерям многих опытных сотрудников наряду с их способностями и знаниями, которые в свою очередь забрали бесценное вдохновение и креативность организации. Следовательно, организациям приходиться платить в дальнейшем высокую цену, в обмен на временную выгоду. Хуже всего после нескольких лет сокращений и реинжиниринга бизнес-процессов, компании конкурируют друг с другом на равных уровнях стоимости, качества и производительности. Это означает, что у компании есть трудности в дифференциации со своими конкурентами. Что усиливает и без того жестокое поле битвы, так это наличие дешевой рабочей силы в Азиатских и других развивающихся странах. Таким образом, концепция базы знаний возникла для поддержания долгосрочного конкурентного преимущества путем сохранения организационных знаний. Знание теперь признано одним из наиболее важных активов управления, потому что знания позволяют организациям использовать и развивать ресурсы, повышать их фундаментальную конкурентоспособность и развивать устойчивое конкурентное преимущество. Другими словами, знания позволяют организации делать что-либо лучше, чем ее конкуренты.

Прежде чем продолжить обсуждение базы знаний, важно уточнить значение знания. Знание — это не редкое слово. Тем не менее, многие люди до сих пор путают различия между данными, информацией и знаниями. Данные — это необработанные факты, а информация — это данные, которые были уточнены, обработаны и организованы для принятия решения. Большинство данных представлены в виде числовой базовой информации или наблюдения за производственной деятельностью, которые могут быть определены количественно, в то время как информация представляет собой данные с актуальностью, целью, а также контекстом. Информация не имеет большого значения до того, как человек не применит её для извлечения значения или использования в работе. С другой стороны, знание появляется в виде фактов, взглядов, мнений, проблем, ценностей, теорий, причин, процессов, инструментов, отношений, рисков и вероятностей. Знание часто рассматривается как информация, которая содержит конкретные свойства. Некоторые определяют знания как информацию, которая релевантна, действенна и основана по крайней мере частично на опыте далее или как оправданное истинное убеждение, которое рационально, динамично, гуманистически и контекстно-специфическое. Информация станет знанием, только если присутствует личная интерпретация опыта, убеждений и обязательств. Пока одни рассматривают информацию как своего рода предварительную стадию знания, другие утверждают, что знания и информация связаны друг с другом через процессы интернализации информации в личные знания и экстернализации личных знаний в информацию. Знания делят на молчаливые и явные. Молчаливое знание, которое приобретается благодаря индивидуальному пониманию со временем, является личным, сложным для распространения, а также формализации, потому что оно находится в человеческом уме, основывается на точке зрения, убеждений, предположений, поведения, восприятия, действий, процедур, обязательств, идеалов, ценностей и эмоций. И наоборот, явное знание структурировано и относительно простое. Оно может быть зафиксировано, записано, задокументировано, систематизировано и распространено с использованием формального и систематического языка в формах учебных пособий, патентов, отчетов, документов, оценок, баз данных, научных формул и т.д.

Знание определяется как оправданное истинное убеждение, которое является рациональным, динамичным, гуманистическим и специфичным для контекста и может проявляться в форме фактов, взглядов, мнений, проблем, ценностей, теорий, причин, процессы, инструменты, отношения, риски и вероятности.

База знаний пытается управлять знаниями, которые накапливаются в рабочей сфере. Это достигается путем организации формального и прямого процесса создания, хранения, сохранения, оценки и распространения знаний для будущей выгоды организации. Функционал базы знаний реализует создание, оценку, хранение, распространение и использование знаний. (рисунок 1)

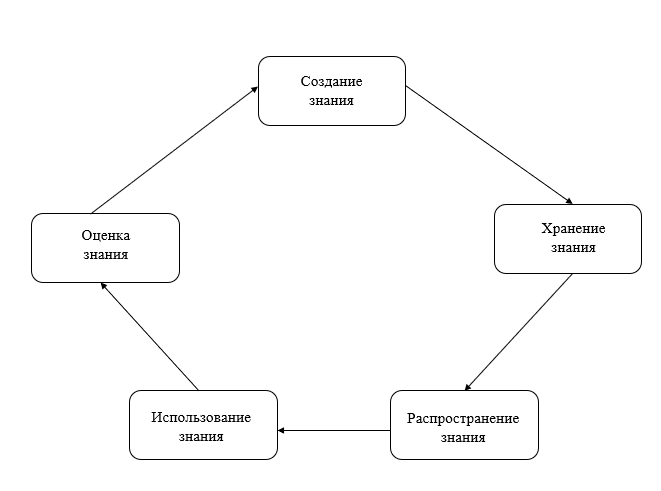


Рисунок 1 – Основные функции базы знаний

Существует четыре метода создания организационных знаний посредством взаимодействия между явным и неявным знанием. Первый метод - социализация. Это процесс разработки нового молчаливого знания из негласных знаний, встроенных в человека или организацию через обмен опытом, наблюдение и традиционное обучение. Второй метод называется экстернализацией. Это процесс превращения молчаливого знания в новые явные знания просто путем преобразования молчаливых знаний в форму документа, такого как руководство и отчет. Третий метод - комбинация. Это процесс слияния и редактирования явных знаний из нескольких источников в новый набор более всеобъемлющих и систематических явных знаний. Последний называется интернализация. Это процесс воплощения явных знаний в молчаливые знания путем обучения, поглощения и интеграции явных знаний в базу негласных знаний человека. Второй и третьи этапы базы знаний, хранение и распространение часто связаны с технологиями. Созданные явные знания собираются и хранятся в какой-либо базе данных или базе знаний, в которую пользователи имеют право доступа, используя поиск и извлечение, используя интранет, веб-доступ и приложения, программное обеспечение для групп и т. д. Вместо того, чтобы реагировать на возникшие трудности, знания должны использоваться упреждающим образом.

Использование знаний может быть приведено в соответствие с четырьмя аспектами управления:

1) непосредственный руководитель, кто сосредоточен на существующем управлении производительностью, отвечает за повышение текущего процесса с помощью операционных знаний;

2) старший функциональный менеджер, кто использует знания о функциональных требованиях, производительности ожидания, ресурсы и технические возможности, а также потенциал для реализации и координирует организационную стратегию;

3) старшее должностное лицо, которое может развиваться и использовать потенциальную стратегию используя внешние знания для прогнозирования тенденций в неопределенном будущем;

4) технический специалист, отвечающий за улучшение будущих и текущих эксплуатационных характеристик используя революционные специальные знания для внесения своего вклада в процессы, продукты, услуги и проблемы конкретного бизнеса.

Компания может эффективно использовать знания, способные предоставлять новые рыночные ценности и определять цены на мировом рынке.

Оценка знаний исключает неверные или устаревшие знания. Другими словами, организации должны продолжать создавать новые знания и заменить любые знания, которые стали недействительными.

Следовательно, база знаний, в отличие от других универсальных решений, способна выдержать долгосрочное конкурентное преимущество, но как этого достичь? Компания все еще может дублировать и подражать процессу базы знаний или даже его технологии, но они никогда не смогут скопировать знания и навыки сотрудников. Принцип базы знаний побуждает организации постоянно создавать и использовать знания, а также проявлять инициативу в инновациях и улучшении услуг, продуктов и операций.

Хотя база знаний является фундаментальным фактором успеха компании, некоторые проблемы все же присутствуют. Меньшинство рассматривает их как очередную переупаковку ИТ-проекта и даже путают KMS с IS, потому что их концепции и функции похожи.

KMS — это основанная на ИТ система, предназначенная для использования в процессе базу знания: создание, хранение / поиск, распространение и применение знаний.

Технология — это всего лишь инструмент, используемый для хранения и распространения знания, но сама технология не добавляет ценности знаниям.

Сотрудничество и взаимодействие между группами, отдельными лицами и лидерами в передаче знаний и обмен ими. Вот что может повысить ценность знаний. Уровень доверия, время, наличие участия лидеров, обстановка, организационная структура и денежные, а также неденежные вознаграждения являются ключами для мотивации передачи знаний и обмена ими.

* 1. **Программный агент**

Компьютерная программа отошла от самой ранней стадии компьютерно-специфического приложения к объектно-ориентированной парадигме, которая состоит из концепции объекта, абстрактных типов данных, полиморфизма, наследования и инкапсуляции. Объектно-ориентированный подход очень популярен главным образом из-за его повторного использования, расширяемости, гибкости, а также способности строить и абстрагировать сложную систему.

К сожалению, понятия и механизма объектов, классов и модулей в объектно-ориентированном программировании недостаточно для моделирования сложных задач реального мира из-за пассивного характера объекта и негибкости выбора действий в вызванный метод.

Инновационное понятие под названием программный агент разработана, чтобы справиться со сложностью реального мира. Преимущества агентно-ориентированного подхода включают в себя:

1) естественность модульных компонентов с точки зрения

цели, которых они достигают,

2) способность контролировать и определять свои собственные действия

3) существенное сокращение проблем связаных с соединением компонентов из-за использования агента высокого уровня

4) заметное сокращение проблем, связанных с управлением отношениями между программными компонентами за счет использования восходящего межагентского взаимодействия.

Идея программного агента основана на параллельной модели Карла Хьюитта, которая предложил концепцию автономного, интерактивного и одновременно выполняемого объекта или субъекта, в котором его внутреннее состояние инкапсулировано и обладает способностью реагировать на сообщения от других подобных субъектов. Исследователи и ученые до сих пор не могут уступить определению программного агента. Это неизбежно, что спор будет продолжаться некоторое время, пока они не пойдут на компромисс с широко принятой интерпретацией. Агент это тот, кто делает вещи и действует от имени кого-то или чего-то. Компьютерная программа, которая ведет себя как человек и способна на самостоятельные действия в достижении конкретной цели.

Кроме того, программный агент должен функционировать непрерывно, гибко и разумно, чтобы общаться, отвечать, определять, прогнозировать и сотрудничать в особой среде без вмешательства человека.

Это необходимо, особенно когда проблемы связаны с несколькими агентами, потому что агенты должны будут взаимодействовать друг с другом либо для достижения их отдельных целей или управлять зависимостями.

Некоторые не согласны с вышеуказанными соглашениями и утверждают, что программный агент не больше, чем кусок обычной прикладной программы. Они также жалуются, что большинство продуктов торгового агента - просто трюк продаж и не имеют существенных отличий от существующих технологий.

Чтобы бороться с его определением, рассмотрим программный агент как общий термин, охватывающий ряд более конкретных типов агентов, а затем продолжим список и определим другие типы агентов.

В «Топологии агентов», классифицируются существующие программные агенты на семь категорий: совместные агенты, интерфейсные агенты, мобильные агенты, информационные / Интернет агенты, реактивные агенты, гибридные агенты и умные агенты. Нет сомнений, что есть и другие методы классификации программных агентов, но общепринятые агенты должны обладать хотя бы одной из следующих характеристик: автономность, реактивность, проактивность, совместность, мобильность, адаптивность, личность, временная преемственность, коммуникабельность, гибкость, обучаемость и интеллект.

Личность относится к способности проявить признаки настоящего человека. Степень личности в основном зависит от того, с чем агент чаще всего взаимодействует. Если агент должен регулярно взаимодействовать с человеком, то лучше включать высокую степень индивидуальности, чтобы обеспечить «гладкость» взаимодействия и уменьшить недопонимание. Еще одна характеристика программного агента временная непрерывность. Здесь временная непрерывность означает постоянство идентичности и состояние в течение длительного периода времени. Так как агентская деятельность обычно

включает последовательность действий, которая длится в течение определенного периода времени, стабильность агента является ключом для поддержания целостности всего процесса. Гибкость — это способность выбрать подходящие действия в правильной последовательности в ответ на состояние внешнего окружающей среды. Эта характеристика отделяет технологию агента от традиционного программного приложения таким образом, что, когда метод в традиционном программном обеспечении вызывается, все действия выполняются. В отличие от этого, агент может решить наиболее соответствующее действие, соответствующее текущей ситуации.

Автономность - это способность выполнять свою задачу без прямого контроля или только с минимальным надзором. Чтобы достичь поставленной цели, ожидается, что автономный агент будет «ощущать, когда начинать», действовать, реагировать, а также принимать свое собственное решение в соответствии с окружающей средой без получения одобрения от человека. Однако, чтобы гарантировать, что агент находится под контролем, пользователь должен обеспечить степень автономности достаточно, чтобы агент выполнил задачу. Реактивность относится к способности воспринимать окружающую среду и отвечать на нее соответствующим образом. Реактивный агент разделяется на истинный реактивный агент и совещательный агент. Чтобы реагировать с окружающей средой, первый имеет подходящий датчик, в то время как последний обладает собственной внутренней моделью среды и из чего он может сделать свой собственный вывод.

Напротив, проактивность - это способность достичь своей цели проектирования в динамичной и непредсказуемой среде. Другими словами, активный агент способен принимать инициативу с целью достижения заранее определенной цели.

Другой характеристикой агента является совместность. Это способность сотрудничать с другими агентами для выполнения задач в открытых и ограниченных по времени мультиагентных средах. Агенту невозможно кодировать каждый сценарий из-за сложности реального мира, но совместность характеристик позволяет агенту преодолевать трудности путем обмена информацией и договариваются о специализированном обслуживании друг с другом.

Мобильность - способность передавать себя в другую среду через сеть. Если агент не обладает достаточными ресурсами или необходимыми услугами для достижения цели, возможность мобилизации позволяет агенту перемещаться по сети, пока он не достигнет целевого хоста, в котором агент мобильности может воспользоваться необходимыми услугами или ресурсами. Несмотря на преимущества, присутствуют проблемы по вопросам безопасности, конфиденциальности данных и управления.

Коммуникационная способность - это способность общаться с другими агентами, а также людьми. Коммуникационные способности имеют решающее значение, когда ресурсы или способность не в состоянии устранить барьер. При этом обстоятельстве агент может общаться с другими агентами или людьми-пользователями для получения информации, ресурсов, услуги или разрешение для решения проблемы.

Агент, который обладает адаптивностью, может корректировать свое поведение в соответствии с новыми целями и другими изменениями окружающей среды. Такое действие выполняется автоматически и принципиально на основе предыдущего опыта. Способность к обучению относится к способности учиться с целью улучшения принятия решений алгоритмом. Способность к обучению тесно связанны с адаптивностью. Например, если агент обнаруживает новые ресурсы, в то время как некоторые из его собственные ресурсы оказываются устаревшими и имеют ограниченное использование, агент должен учиться и адаптировать свое поведение соответственно. Разумный агент должен обладать как способностью к обучению, так и способностью к адаптации, чтобы агент мог учиться и адаптироваться, как ему следует действовать или взаимодействовать с внешней средой. Интеллект означает степень способности рассуждать и учиться, выполнять задачу в соответствии с целью пользователя. Таким образом, его производительность будет увеличивается со временем.

* 1. **Веб-система**

World Wide Web (WWW) полностью затмила другие Интернет приложения и становится крупнейшим потребителем пропускной способности магистрали Интернета. WWW изначально разработан, чтобы позволить людям находить или просматривать информацию на статических веб-страницах, нажав на соответствующий URL, но потенциал сети ушёл далеко от этого. В настоящее время веб-технологии используются для обеспечения широкого разнообразия гибких, динамичных и интерактивных действий, которые варьируются от простых приложений, мультимедийных веб-страниц, сложных бизнес-систем, сложных программных приложений. Быстрое развитие беспроводной сети еще больше распространило веб-приложения. В настоящее время мобильные устройства, такие как ноутбук и 4G мобильный телефон могут принимать участие в веб-мероприятиях. Вышеуказанная среда, несомненно, помогает ускорить популярность веб-системы. Web-система тесно связана с появлением электронной коммерции, электронной торговли, электронного бизнеса и других электронных услуг. В определенной степени, web-система даже рассматривается как один из жизненно важных элементов в нашей повседневной жизни, и никто не может провести день, не используя их хотя бы один раз. Веб-приложения распространились почти на каждый аспект в обществе и в широком диапазоне включают в себя торговые системы, системы HD, системы здравоохранения, бизнес системы, системы моделирования, системы поддержки принятия правовых решений и образовательные системы. Хотя существует множество типов веб-систем, выделяют некоторую архитектуру. Во-первых, разработчик должен выбрать тонкий или толстый клиент, другими словами, это выбор того, будет ли пользовательский интерфейс приложений храниться и выполняться удаленно, или локально. Во-вторых, разработчик должен решить количество уровней для организации приложения в рамках клиентской и серверной парадигм. В-третьих, разработчик должен определить общее количество серверов по вычислительной мощности. Наконец, разработчик должен решить, насколько разные приложения или уровни приложений выделятся, то есть стоит ли размещать несколько приложений на одном и том же или отдельном компьютере. Независимо от того, какие альтернативы дизайнер выбирает, веб-система в основном состоит из веб-клиента, веб-сервера, приложения и сервера базы данных. Веб-клиент обеспечивает пользовательский интерфейс для веб-системы, тогда как веб-сервер отвечает за взаимодействие с веб-клиентом и сервером приложений. Чтобы быть точным, веб-сервер захватывает запрос от веб-клиента и доставляет запрос на сервер приложений, на котором находится сервер базы данных, использует для поддержки поиска информации с целью подготовки ответа для рендеринга пользовательского интерфейса.

**Глава 2. Разработка проектного решения**

* 1. **Основные положения разрабатываемой системы**

После определения требований к системе необходимо начать разработку архитектуры, то есть проанализировать требования к системе и к его дизайну. Необходимо разработать то, как и какие компоненты системы будут взаимодействовать между собой.

Выделим необходимые компоненты:

1) Графический интерфейс.

2) База данных

3) База знаний

4) Управление пользователями

5) Управление заявками

В целом, разрабатываемая система основывается на таком принципе: на сервере запущено приложение, которое осуществляет создание, хранение изменение и распространение заявок.

* 1. **Графический интерфейс**

Графический интерфейс для пользователей будет состоять из таблицы заявок и меню, расположенного сверху. Для модераторов также будет возможность распределения заявок. Для администраторов добавиться возможность управления пользователями.

Целевые пользователи данной веб-системы - пользователи с низким и средним техническим навыком. Поэтому дизайн системы должен быть простым и удобным для пользователя. Простой динамический пользовательский интерфейс будет позволять пользователям просматривать и создавать заявки, выбирая их типы из имеющихся. Хотя такие системы довольно часто используют поиск по ключевым словам в качестве внешнего интерфейса, динамический пользовательский интерфейс подходит намного лучше для начинающих пользователей, чтобы было возможно использовать соответствующие и правильные слова для описания проблемы.

* 1. **База данных**

База данных будет хранить в себе заявки и их статусы, пользователей и их роли.

В таблице заявок будут содержатся следующие колонки:

1. Уникальный идентификатор
2. Название заявки
3. Пользователь, создавший заявку
4. Исполнитель заявки
5. Состояние
6. Приоритет
7. Описание
8. Тип
9. Жизненный цикл
10. Комментарий

Уникальный идентификатор будет использоваться как первичный ключ и нужен для поиска и отличия одинаковых заявок друг от друга.

В названии заявки будет записываться короткий текст.

Дата и время создания будет автоматически записываться в таблицу программой.

Исполнителя заявки назначает модератор в дальнейшем, после её создания.

Для заявки будут возможны состояния:

1. Не распределена
2. Распределена
3. Пауза
4. Закрыта успешно
5. Закрыта неуспешно

Для заявки существуют приоритеты:

* 1. Низкая
  2. Обычная
  3. Высокая

В описании будет максимально подробный текст о неисправности.

Тип заявки выставляет пользователь. Будет доступно 3 типа:

1. Сантехника
2. Электрика
3. Мебель

В жизненный цикл заявки будут записываться изменения состояния заявки со временем.

В таблице пользователей будут такие колонки:

Уникальный идентификатор

1. Имя
2. Логин
3. Пароль
4. Роль

В программе предусмотрены роли пользователей:

1. Администратор
2. Модератор
3. Исполнитель
4. Пользователь
   1. **База знаний**

База знаний – электронный репозиторий решений. При помощи нее исполнитель сможет быстро найти верное решение случившейся типовой проблемы и решить ее быстро и правильно.

* 1. **Управление пользователями**

Управление пользователями будет доступно только роли администратора. Этот модуль предусматривает:

* Создание пользователей
* Изменение пользователей
* Удаление пользователей

Также создание пользователей будет доступно самой программе при регистрации нового пользователя.

* 1. **Управление заявками**

Функция управления заявками доступна всех ролям, но в разной степени.

Пользователь, создавший заявку, может её удалить, изменить название, описание или тип. К заявкам других пользователей он не имеет доступа.

Исполнитель имеет доступ только к заявкам, назначенным ему. Он сможет изменять в них только состояние и писать комментарии.

* 1. **Обоснование выбора программных средств**

На сегодняшний день существует огромное множество языков программирования, подходящих для реализации нашей задачи. Самые популярные из них:

* PHP
* C#
* Java
* Perl

Для реализации приложения будет использован язык программирования C#, среда разработки Visual Studio 2017 платформа .NET core 2.2 и библиотека Dapper.

С учетом того, что было изучено и с чем приходилось сталкиваться на протяжении обучения в университете, выполняя практические и курсовые работы по объектно-ориентированному программированию, принято решение реализовать программу на языке C#.

С# – основной язык разработки приложений для платформы Microsoft.NET. Простой, современный, объектно-ориентированный, обеспечивающий безопасность типов язык программирования. В частности:

– обеспечивает языковые конструкции, непосредственно поддерживающие концепции объектно-ориентированного программирования;

– обеспечивает поддержку компонентно-ориентированного программирования;

– основан на строгой компонентной архитектуре и реализует передовые механизмы обеспечения безопасности кода;

– обладает свойствами, помогающими создавать надежные и устойчивые приложения, такими как: сборка мусора автоматически освобождает память, занятую неиспользуемыми объектами; обработка исключительных ситуаций обеспечивает структурированный и расширяемый подход к обнаружению ошибок и восстановлению; структура языка, обеспечивающая безопасность типов, делает невозможным получать значения неинициализированных переменных, индексировать массивы вне их границ или выполнять бесконтрольное приведение типов;

– имеет единую систему типов;

– поддерживает определяемые пользователем ссылочные типы и типы‑значения, что позволяет осуществлять динамическое распределение памяти под объекты наряду с компактным хранением небольших структур в стеке;

– особое внимание уделяется управлению версиями;

– программы и библиотеки последующих версий C# совместимы с предыдущими;

– функциональные возможности языка, такие как обобщенные типы и анонимные делегаты, динамические значения, сделали программы проще, их стало легче читать и поддерживать.

Профессионалы называют среди прочих такие преимущества языка C# как:

– множество полностью готовых к применению библиотек, уже встроенных в .NET Framework;

– комфортная среда разработки;

– стремительный захват языком новых платформ и операционных систем, благодаря развитию таких проектов, как [*mono*](http://ru.wikipedia.org/wiki/Mono) (реализация платформы .NET на Unix-подобных операционных системах);

– исключение некоторых моделей, зарекомендовавших себя как проблематичные при разработке программных систем, например, не поддерживается множественное наследование классов (в отличие от C++).

Кроме этого, C# упрощает работу с базами данных за счет использования таких библиотек как Dapper или Entity Framework.

.NET Core - это универсальная платформа разработки с открытым исходным кодом, поддерживаемая Microsoft и сообществом .NET на GitHub. Он кроссплатформенный (поддерживает Windows, macOS и Linux) и может использоваться для создания устройств, облачных приложений и приложений.

.NET Core был построен как очень похожий, но уникальный продукт по сравнению с другими продуктами .NET. Он был разработан для обеспечения широкой адаптируемости к новым платформам и рабочим нагрузкам. Он имеет несколько доступных портов ОС и ЦП и может быть перенесен на многие другие.

Продукт разбит на несколько частей, что позволяет различным частям адаптироваться к новым платформам в разное время. Среда выполнения и основанные на платформе базовые библиотеки должны быть портированы как единое целое. Библиотеки, независимые от платформ, должны работать как есть на всех платформах. Существует предвзятость проекта к сокращению реализаций, специфичных для платформы, для повышения эффективности работы разработчика, предпочитая нейтральный для платформы код C # всякий раз, когда алгоритм или API могут быть реализованы полностью или частично таким способом.

Люди часто спрашивают, как реализовано .NET Core для поддержки нескольких операционных систем. Обычно они спрашивают, существуют ли отдельные реализации или используется условная компиляция. И то и другое с сильным уклоном в сторону условной компиляции.

Entity Framework - это набор технологий в ADO.NET, которые поддерживают разработку программных приложений, ориентированных на данные. Архитекторы и разработчики приложений, ориентированных на данные, столкнулись с необходимостью достижения двух совершенно разных целей. Они должны моделировать сущности, отношения и логику бизнес-задач, которые они решают, и они также должны работать с механизмами данных, используемыми для хранения и извлечения данных. Данные могут охватывать несколько систем хранения, каждая со своими собственными протоколами; даже приложения, которые работают с единой системой хранения, должны сбалансировать требования системы хранения с требованиями написания эффективного и поддерживаемого кода приложения.

Entity Framework позволяет разработчикам работать с данными в форме доменных объектов и свойств, таких как клиенты и адреса клиентов, не заботясь о таблицах и столбцах базовой базы данных, где хранятся эти данные. С Entity Framework разработчики могут работать на более высоком уровне абстракции, когда они имеют дело с данными, и могут создавать и поддерживать ориентированные на данные приложения с меньшим количеством кода, чем в традиционных приложениях. Поскольку Entity Framework является компонентом .NET Framework, приложения Entity Framework могут работать на любом компьютере, на котором установлен .NET Framework, начиная с версии 3.5 с пакетом обновления 1 (SP1).

Dapper - это микро ORM, или это простая структура отображения объектов, которая помогает отобразить исходный вывод запроса в класс домена или класс C#. Это высокопроизводительная система доступа к данным, созданная командой StackOverflow и выпущенная с открытым исходным кодом. Если требуется писать хранимые процедуры или писать собственные запросы, а не использовать полноценные инструменты ORM, такие как EntityFramework или NHibernate, тогда Dapper - очевидный выбор для вас. Используя Dapper, очень легко запустить SQL-запрос к базе данных и получить результат, сопоставленный с классом домена C#.

**Заключение**

В ходе выполнения дипломной работы мною был разработан и реализован веб-приложение для повышения эффективности работы сервисной службы. Для этого были решены следующие задачи:

Анализ предметной области, а именно исследование help desk систем и заявок, обзор базы знаний. Данный анализ показал, что задача эффективной системы управления заявками не является решенной на сегодняшний день. Количество и сложность заявок постоянно растет, выполнять такое количество заявок становится все труднее, как из-за постоянного увеличения числа обращений и утечки высококвалифицированного персонала. Поэтому необходимость своевременно сформировывать и принимать меры по решению проблемы, напрямую связана с системой управления.

Разработка концепции системы управления заявками. Для решения этой задачи была разработана структура базы данных, графический интерфейс системы, выбраны средства разработки программного продукта.