РЕФЕРАТ

Выпускная квалификационная работа содержит 116 страниц текстового документа формата А4, включающего 30 рисунков, 3 таблицы, 33 использованных источника, 1 приложение.

ИНФОРМАЦИОННАЯ СИСТЕМА, ТЕХНИЧЕСКАЯ ПОДДЕРЖКА ПОЛЬЗОВАТЕЛЕЙ, БАЗА ДАННЫХ, ЭКОНОМИЧЕСКАЯ ЭФФЕКТИВНОСТЬ, ИНФОРМАЦИОННО-ЛОГИЧЕСКАЯ МОДЕЛЬ, БИЗНЕС-ПРОЦЕССЫ.

Целью выпускной квалификационной работы является исследование информационного отдела КГБУЗ «Городская клиническая поликлиника №3» на наличие проблем в процессах технической поддержки пользователей.

Объект исследования – деятельность отдела информатизации КГБУЗ «Городская клиническая поликлиника №3» МЗ ХК

Предмет исследования – улучшение эффективности технической поддержки пользователей путем внедрения информационной системы поддержки пользователей.

В процессе работы над выпускной квалификационной работой было проведено исследование важности использования информационных систем в организациях, а также важность служб технической поддержки и современные методы работы таких служб. Было проведено исследование процесса технической поддержки пользователей в организации.

Результатом выполненной работы является разработка информационной системы поддержки пользователей. Внедрение информационной системы должно на треть сократить «невидимые» потери организации от простоев пользователей, вызванных незапланированными сбоями в работе ИТ-инфраструктуры организации.

Информационной базой послужили методологические материалы,

учебная, научная, методическая литература по рассматриваемому вопросу,

статьи в средствах массовой информации, электронные ресурсы.

СОДЕРЖАНИЕ

|  |  |
| --- | --- |
| ВВЕДЕНИЕ | 5 |
| 1 АКТУАЛЬНЫЕ ВОПРОСЫ РАЗРАБОТКИ ИНФОРМАЦИОННЫХ СИСТЕМ НА СОВРЕМЕННОМ ЭТАПЕ | 7 |
| 1.1 Сущность и экономическое значение информационных систем | 7 |
| 1.2 Методические основы систем поддержки пользователей | 18 |
| 1.3 Бизнес-процессы поддержки пользователей и оценка эффективности использования системы | 26 |
| 2 ПРОЕКТНАЯ ЧАСТЬ | 37 |
| 2.1. Постановка задач и требований к информационной системе | 37 |
| 2.2 Выбор средств реализации информационной системы | 45 |
| 2.3 Разработка информационной системы | 56 |
| 3 РАСЧЕТ ЭКОНОМИЧЕСКОЙ ЭФФЕКТИВНОСТИ | 66 |
| 3.1 Понятие экономической эффективности | 66 |
| 3.2 Методика расчета экономической эффективности | 75 |
| 3.3 Расчет экономической эффективности внедрения информационной системы | 86 |
| ЗАКЛЮЧЕНИЕ | 95 |
| СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ | 99 |
| ПРИЛОЖЕНИЕ А Программный код | 102 |

ВВЕДЕНИЕ

В современном мире информационные технологии играют очень важную роль. Информационные технологии используются повсеместно и играют особую роль в автоматизации бизнес-процессов, протекающих в организациях и корпорациях во всем мире, что в условиях постоянно нарастающей конкуренции является жизненной необходимостью для обеспечения конкурентоспособности любой организации.

Вместе с автоматизацией, информационными технологиями и системами приходит так же необходимость в их обслуживании. Каждая организация выбирает свой способ для решения этой проблемы. На текущем этапе развития информационных технологий для обеспечения бесперебойности их работы может привлекаться как один специалист, так и целые отделы специалистов, в некоторых случаях такие процессы и вовсе выносятся за пределы организации и передаются другим компаниям. У каждого варианта имеется ряд своих преимуществ и недостатков.

Службы технической поддержки так же, не стоят на месте и очень быстро развиваются вместе с новыми технологиями. Создаются новые способы и стратегии для эффективного обслуживания информационных технологий и поддержки пользователей. Техническая поддержка часто подразделяется на уровни с целью улучшить обслуживание организации или базы клиентов. Количество уровней определяется потребностями и желаниями бизнеса или же ставится в зависимость от возможностей эффективно помочь клиентам или пользователям. Успешность организационной структуры технической поддержки связана с пониманием техническими специалистами своих зон ответственности и обязанностей, времени, в течение которого эти обязательства перед клиентами исполняются, а также от особенностей эскалирования проблемы между уровнями технической поддержки.

С практической стороны, отделы информатизации зачастую замкнуты внутри организации и сотрудничают с другими ИТ-отделами только в случае крайней необходимости. Работа технической поддержки зачастую остаётся незамеченной, из-за чего также слабо изучается и редко освещается в СМИ. Это приводит к тому, что даже при появлении новых методов работы, они зачастую остаются незамеченными, что в свою очередь ведёт к спаду эффективности работы таких служб на фоне развития информатизации.

Проблемы в процессах служб технической поддержки часто остаются незаметными, ведь если всё работает, то принято считать, что и проблем нет. К сожалению, зачастую в работе информационных систем происходят сбои, которые негативно отражаются на работе всей организации. Такие сбои не вызывают опасений, так как являются кратковременными или не критичными, однако в сумме наносят значительный ущерб организации.

В КГБУЗ «Городская клиническая поликлиника №3» ежедневно поступает большое количество сообщений о сбоях разного характера, что вероятно является следствием недостаточной эффективности текущих бизнес-процессов.

Поэтому целью дипломной работы является исследование бизнес-процессов отдела и поиск проблемных мест. Это позволит повысить эффективность работы отдела и снизить потери от проблем подобного рода.

Для достижения данной цели необходимо решить следующие задачи: провести теоретический анализ и обработка практической и методической литературы по теме технической поддержки пользователей, анализ особенностей проблемы незапланированных простоев пользователей, проектирование и реализация приемлемой для предприятия информационной системы поддержки пользователей, направленной на реализацию этой проблемы.

Проведение обследования деятельности отдела в частности: составление моделей технической поддержки: «как есть» и «как будет», выявление проблемных бизнес-процессов, поиск оптимальных решений проблемы, определение их решения и разработка информационной системы.

1 АКТУАЛЬНЫЕ ВОПРОСЫ РАЗРАБОТКИ ИНФОРМАЦИОННЫХ СИСТЕМ НА СОВРЕМЕННОМ ЭТАПЕ

* 1. Сущность и экономическое значение информационных систем

На современном этапе развития информационные технологии имеют достаточно большое влияние на окружающий нас мир, уже сейчас информационные системы используются повсеместно почти во всех отраслях человеческой деятельности. Особую роль информационные технологии играют в автоматизации бизнес-процессов, протекающих в организациях и корпорациях во всем мире, что в условиях постоянно нарастающей конкуренции является жизненной необходимостью для обеспечения конкурентоспособности любой организации. Организации, с грамотно организованной информационно-технической структурой, при помощи информационных систем способны упорядочивать все организационные процессы в одну единую систему. Основываясь на точных и достоверных данных о результатах деятельности сотрудников всех уровней и подразделений, поставленных задачах и достигнутых целях, такой подход позволяет анализировать положение дел на предприятии, что способствует принятию точных и взвешенных управленческих решений и грамотному управлению организацией.

Несмотря на повсеместное использование информационных систем, само определение информационных систем достаточно размыто и интерпретируется разными источниками по-разному. Так, например, Федеральным законом Российской Федерации от 27 июля 2006 г. № 149-ФЗ «Об информации, информационных технологиях и о защите информации» понятие информационной системы как совокупность содержащейся в базах данных информации и обеспечивающих её обработку информационных технологий и технических средств. /26/ Так же достаточно широкое определение информационной системы даётся авторами книги The Information System Consultant's Handbook. Systems Analysis and Design, в которой утверждается, что информационная система определяется её неотъемлемыми компонентами, такими как данные, техническое и программное обеспечение, а также персонал и организационное обеспечение. /33/ Среди известных российских ученных в сфере информатики широким образом информационную систему определяет Михаил Рувимович Когаловский в глоссарии по информационному обществу. Согласно глоссарию по информационному обществу информационная система – это компьютерная система, включающая вычислительное и коммуникационное оборудование, программное обеспечение, данные и метаданные, лингвистические средства, а также системный персонал, и обеспечивающая поддержку информационной модели некоторой части реального мира для удовлетворения информационных потребностей пользователей. /8/

Все определения, приведённые выше имеют очень обширное понимание информационной системы. Более узкое определение, в свою очередь, ограничивает её состав данными, программами и аппаратным обеспечением. Интеграция данных компонентов позволяет автоматизировать как процессы управления информацией, так и целенаправленную деятельность конечных пользователей, направленную на получение, хранение и модификацию информации. Подтверждением этого является российский стандарт ГОСТ РВ 51987. Согласно этому стандарту информационная система характеризуется как система, результатом функционирования которой является представление выходной информации для последующего использования. /9/ В свою очередь, в ГОСТ Р 53622-2009 используется такой термин как информационно-вычислительная система. Исходя из текста ГОСТ Р 53622-2009 термин информационно-вычислительной системы обозначает совокупность данных (или баз данных), систем управления базами данных, а также прикладных программ, функционирующих на вычислительных средствах, как единое целое для решения определённых задач. /11/

В деятельности организации информационная система рассматривается как программное обеспечение, реализующее деловую стратегию организации. При этом целью является создание и развертывание единой [корпоративной информационной системы](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D1%80%D0%BF%D0%BE%D1%80%D0%B0%D1%82%D0%B8%D0%B2%D0%BD%D0%B0%D1%8F_%D0%B8%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%86%D0%B8%D0%BE%D0%BD%D0%BD%D0%B0%D1%8F_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0), удовлетворяющей информационные потребности всех сотрудников, служб и подразделений организации. Однако на практике создание такой всеобъемлющей информационной системы слишком затруднено или даже невозможно, вследствие чего на предприятии обычно функционируют несколько различных систем, решающих отдельные группы задач: [управление производством](https://ru.wikipedia.org/wiki/MES), [финансово-хозяйственная деятельность](https://ru.wikipedia.org/wiki/ERP),

[электронный документооборот](https://ru.wikipedia.org/wiki/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D0%B0%D0%B2%D1%82%D0%BE%D0%BC%D0%B0%D1%82%D0%B8%D0%B7%D0%B0%D1%86%D0%B8%D0%B8_%D0%B4%D0%BE%D0%BA%D1%83%D0%BC%D0%B5%D0%BD%D1%82%D0%BE%D0%BE%D0%B1%D0%BE%D1%80%D0%BE%D1%82%D0%B0) и т. д. Часть задач бывает «покрыта» одновременно несколькими информационными системами, часть задач — вовсе не автоматизирована. Такая ситуация получила название «лоскутной автоматизации» и является довольно типичной для многих предприятий. /17/

Таким образом, опираясь на вышеупомянутые определения, можно сделать заключение, что информационную систему можно охарактеризовать как совокупность программных и технических средств, данных и баз данных, с которыми они взаимодействует персонал для решения определённых задач. В деятельности организаций, чаще всего под информационной системой подразумевается специализированное программное обеспечение, способствующее реализации деловой стратегии организации.

Для того, чтобы обозначить сущности информационных систем необходимо рассмотреть основные задачи, свойства и классификацию информационных систем. Прежде всего рассмотрим основные задачи информационных систем, для этого обратимся к учебному пособию по информационным системам под авторством Бурцевой Е.В. /13/ Согласно учебному пособию информационные системы имеют 7 основных задач:

* Осуществление поиска, обработки и хранения информации, которая накапливается в течение большого периода времени, имеет большую ценность.

ИС предназначены для более быстрой и надёжной обработки информации, чтобы люди не тратили время, чтобы избежать свойственных человеку случайных ошибок, чтобы сэкономить расходы, чтобы сделать жизнь людей более комфортной.

* Хранение данных разной структуры. Не существует развитой ИС, работающей с одним однородным файлом данных. Более того, разумным требованием к информационной системе является то, чтобы она могла развиваться. Могут появиться новые функции, для выполнения которых требуются дополнительные данные с новой структурой. При этом вся накопленная ранее информация должна остаться сохранной. Теоретически можно решить эту задачу путём использования нескольких файлов внешней памяти, каждый из которых хранит данные с фиксированной структурой. В зависимости от способа организации используемой системы управления файлами эта структура может быть структурой записи файла или поддерживаться отдельной библиотечной функцией, написанной специально для данной ИС. Известны примеры реально функционирующих ИС, в которых хранилище данных планировалось основывать на файлах. В результате развития большинства таких систем в них выделился отдельный компонент, который представляет собой разновидность системы управления базами данных (СУБД).
* Анализ и прогнозирование потоков информации различных видов и типов, перемещающихся в обществе. Изучаются потоки с целью их минимизации, стандартизации и приспособления для эффективной обработки на вычислительных машинах, а также особенности потоков информации, протекающей через различные каналы распространения информации.
* Исследование способов представления и хранения информации, создание специальных языков для формального описания информации различной природы, разработка специальных приёмов сжатия и кодирования информации, аннотирования объёмных документов и реферирования их. В рамках этого направления развиваются работы по созданию банков данных большого объёма, хранящих информацию из различных областей знаний в форме, доступной для вычислительных машин.
* Построение процедур и технических средств для их реализации, с помощью которых можно автоматизировать процесс извлечения информации из документов, не предназначенных для вычислительных машин, а ориентированных на восприятие их человеком.
* Создание информационно-поисковых систем, способных воспринимать запросы к информационным хранилищам, сформулированные на естественном языке, а также специальных языках запросов для систем такого типа.
* Создание сетей хранения, обработки и передачи информации, в состав которых входят информационные банки данных, терминалы, обрабатывающие центры и средства связи.

Так же как обширно понятие информационных систем, обширен и перечень основных задач, решаемых ими. Чаще всего, конкретные задачи, решаемые информационной системой, напрямую зависят от прикладной области, для которой предназначена информационная система.

Для определения сущности информационной системы необходимо рассмотреть свойства, которыми она определяется. Согласно учебному пособию по информационным системам под авторством Бурцевой Е.В, информационная система определяется следующими свойствами: /13/

* Структура ИС, её функциональное назначение должны соответствовать поставленным целям.
* ИС предназначена для производства достоверной, надёжной, своевременной и систематизированной информации, основанной на использовании БД, экспертных систем и баз знаний. Так как любая ИС предназначена для сбора, хранения и обработки информации, то в основе любой ИС лежит среда хранения и доступа к данным. Среда должна обеспечивать уровень надёжности хранения и эффективность доступа, которые соответствуют области применения ИС.
* ИС должна контролироваться людьми, ими пониматься и использоваться в соответствии с основными принципами, реализованными в виде стандарта организации на ИС. Интерфейс пользователя ИС должен быть легко понимаем на интуитивном уровне.
* Любая информационная система может быть подвергнута анализу, по

строена и управляема на основе общих принципов построения систем.

* Любая ИС является динамичной и развивающейся.
* При построении ИС используются сети передачи данных.

Проводя анализ свойств информационной системы, приведенных в учебном пособии по информационным системам, под авторством Бурцевой Е.В, можно заключить, что существование любой информационной системы невозможно, без наличия определённой и чёткой структуры, соответствующей поставленным целям. Так как в любой ИС первостепенной задачей является работа с данными и их обработка, то и наличие надёжного и эффективного места для хранения необходимой информации, тоже является немаловажным элементом ИС. Информационная система предназначена для использования людьми, вследствие чего должна пониматься и управляться ими, посредством наличия интуитивно понятного пользовательского интерфейса.

Информационные системы, хоть и имеют общие черты, могут очень сильно различаться между собой, в зависимости от поставленных пред ними задач. Разнообразие задач, решаемых с помощью информационных систем, стало причиной появления множества разнотипных систем, отличающимися между собой заложенными в них принципами построения самих систем и правилами обработки информации. Согласно монографии “Проектирование информационных систем” под авторством Остроух А.В. /22/ ИС могут классифицироваться по степени охвата функций объекта, по степени автоматизации обрабатываемой информации, по виду единицы хранимой информации. Для более наглядного представления рассмотрим виды классификаций информационных систем на рисунке 1.

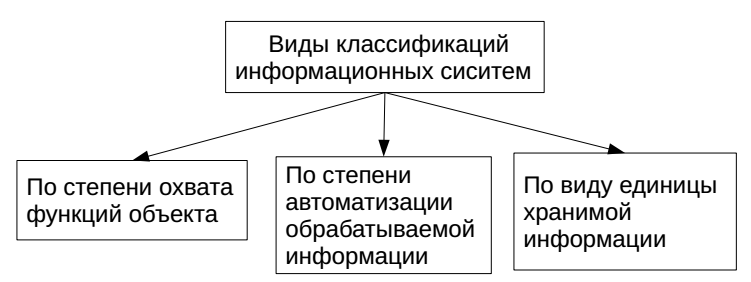


Рисунок 1 – Виды классификаций информационных систем

Разберём каждый вид классификаций более подробно. По степени охвата функций объекта выделяют три возможных вида информационных систем:

* Автоматизированное рабочее место (АРМ)
* Локальная ИС
* Корпоративная ИС

Автоматизированное рабочее место предназначено для поддержания деятельности одного пользователя, подразумевая под пользователем всех сотрудников, которые являются такими пользователями. В отличии от АРМ, локальная ИС рассчитана на поддержку выполнения определённых функций организации, как правило такие системы используются несколькими пользователями одного отдела на разных рабочих местах. Корпоративные информационные системы предназначены для автоматизации деятельности всей организации, такие информационные системы позволяют сотрудникам разных отделов более эффективно работать друг с другом и совместно выполнять необходимые задачи. Такие системы состоят из множества модулей, каждый из которых автоматизирует определённый бизнес-процесс. Основными требованиями к системам подобного класса является согласованность информационной системы с потребностями, структурой и культурой компании, так же такие системы должны быть масштабируемы и при необходимости расширяться, путём включения в них новых модулей.

По степени автоматизации информационные системы бывают:

* Ручные
* Автоматизированные
* Автоматические

Главной особенностью ручных информационных систем является то, что все операции по обработке данных в ней, выполняются человеком. Соответственно автоматизированные системы тоже имеют часть функций, которые выполняются пользователем вручную, другая же часть функций выполняется информационной системой в автоматическом режиме. Третий, и последний вид систем, классифицируемых по степени автоматизации – автоматические системы. Для данного типа систем характерна полная автоматизация всех функций управления и обработки данных с помощью технических средств без необходимости участия человека. Например, такая система может быть использована для управления технологическими процессами.

Так же используется классификация информационных систем по виду единицы хранимой информации. Классификация информационных систем по виду единицы хранимой информации приведена в учебнике «Основы информационных технологий. Учебник» под авторством Остроуха А.В. /21/ В этом учебнике определены следующие виды информационных систем:

* Фактографическая
* Документальная
* Геоинформационная
* Мультимедийная

Кратко рассмотрим каждый вид по отдельности. Фактографическая информационная система характерна тем, что в ней накапливаются и хранятся данные в виде множества экземпляров одного или нескольких типов структурных элементов. Совокупность таких экземпляров структурных элементов, и каждый из них в отдельности, отражают сведения по какому-либо факту или событию, вычлененному от всех прочих сведений и фактов, в отдельности от всех прочих сведений и фактов. Документальные информационные системы, в отличие от фактографических, содержит документы, не структурированные на более мелкие элементы. Примером таких документов могут выступать различные статьи, книги, рефераты и прочие текстовые, звуковые и графические объекты, снабженные тем или иным формализованным аппаратом поиска. Как правило целью такой системы является выдача ответа на запрос пользователя, зачастую такой ответ состоит из списка документов или объектов, удовлетворяющих условиям запроса, сформированного пользователем. Геоинформационная система, сокращённо называемая ГИС, используется для сбора, хранения, анализа и графической визуализации пространственных данных и информацией о необходимых объектах, связанной с ними. Главной особенностью ГИС, выделяющей её среди других видов ИС, является наличие в составе системы специфических методов анализа пространственно-координированной информации. Концепция ГИС-технологии заключается в создании многослойной электронной карты, опорный слой которой описывает географию территории, а каждый из остальных слоёв – один из аспектов состояния территории. Таким образом ГИС-технологии определяют специфическую область работы с информацией. Мультимедийная информационная система характеризуется одновременным хранением и представлением различных медиа, включая звуковые объекты, объекты анимированной компьютерной графики и видеоряда.

В статье «Целесообразность использования информационных систем на предприятии» под авторством Агафоновой М. С., Кулешова И. П. и Зелепукина В. А. /2/, посвященной рассмотрению роли информатизации в деятельности предприятия, отображению значимых преобразований в организации за счёт использования информационных систем, можно сделать вывод, что наличие информационной системы на предприятии является конкурентным преимуществом. В свою очередь, отказ от использования информационных технологий ведёт ведет организацию к негативным последствиям. Отсутствие на предприятиях опыта анализа оценки эффективности информационных технологий и широкого применения их, ведет к негативным последствиям таким, как снижение конкурентного преимущества, а в дальнейшем и полная потеря своей позиции на рынке. Процесс перехода от индустриального общества к информационному путем насыщения современными информационными технологиями экономической, социальное и политической деятельности получил название информатизация. Процесс информатизации представляет собой организационный, социально-экономический и научно-технический процесс по созданию оптимальных условий для удовлетворения информационных потребностей и реализации прав граждан, органов государственной власти, органов местного самоуправления, организаций, общественных объединений на основе формирования и использования информационных ресурсов. /1/ На данный момент сложно недооценить на сколько велико влияние информационных систем на культуру управления организации, менеджмент и общество. С тех пор, как предприниматели использовали первые быстродействующие компьютеры преимущественно для автоматизации процессов, выполняющихся вручную большим количеством сотрудников с невысокой квалификацией, прошло относительно немного времени. Однако, в нынешнее время, при столь быстром развитии информационных технологий, информационные технологии используются для реализации новых идей, новых способов получения конкурентного преимущества, а не только для выполнения операций по обработке данных. Благодаря сетевым технологиям и распределенным системам мир сузился до размеров экрана монитора и рабочего стола, что положительно отразилось на работе организаций, так как предоставило возможность быстрого и простого использования больших объемов информации и её обработки, с помощью специального инструментария.

Внедрение информационной системы в организацию имеет свои положительные стороны, для наглядности представим положительные эффекты от внедрения информационных систем на рисунке 2, а за тем разберём каждое из них более подробно.

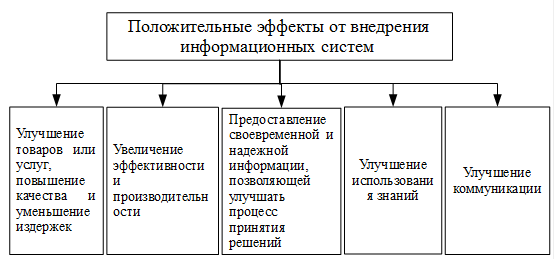


Рисунок 2 – Положительные эффекты от информационных систем

Первым положительный эффект от внедрения ИС – улучшение товаров или услуг, повышение качества и уменьшение издержек. Это особенно хорошо прослеживается в компаниях, ориентированных на работу с клиентами. Применение информационных технологий позволяет быстрее обслуживать клиентов, увеличивая тем самым «пропускную способность» организации, что несомненно положительно сказывается на выручке компании.

Второй положительный эффект – это увеличение эффективности и производительности. Примером данного эффекта может служить информационная система на производстве, оповещающая специалиста о неисправностях оборудования. Благодаря этому в значительной степени уменьшается количество брака, время простоя оборудования и другие возможные последствия, вызванные работой неисправного оборудования, что положительно сказывается на качестве продукции.

Третий положительный эффект – предоставление своевременной и надежной информации, позволяющей улучшать процесс принятия решений. Как пример, при помощи информационной системы менеджер может контролировать запасы продукции на складе и при резком падении продаж – отредактировать количество необходимой продукции, тем самым предотвратив финансовые потери организации.

Четвёртым положительным эффектом, представленным на рисунке 1.2, является улучшение использования знаний. Такой эффект часто наблюдается при использовании экспертных систем. Например, в экспертных системах по налогообложению, в которых содержатся знания лучших экспертов фирмы, что позволяет консалтинговым фирмам оказывать своим клиентам помощь по планированию налогов. /11/

Улучшение коммуникации. Улучшение коммуникации при использовании информационной системы может выражаться несколькими способами. Одним из примеров такого эффекта является ситуация, когда сотрудник компании находится в командировке. Не смотря на территориальную удалённость от рабочего места, сотрудник не утрачивает возможность обмениваться деловой почтой со своими коллегами, просматривать рабочие файлы и работать с ними.

Из этого можно сделать вывод, что сущность информационных систем заключается в наличии чёткой структуры, базы данных - надёжного и эффективного места для хранения необходимой информации, данных и персонала, взаимодействующего с ней для решения поставленных задач. Экономическим значением информационных систем является обеспечение организаций возможностью эффективной работы с информацией, одним из важнейших ресурсов нашего времени. Что в свою очередь позволяет компаниям обретать конкурентное преимущество перед другими компаниями, а игнорирование важности информационных систем ведёт к потере позиций на рынке.

* 1. Методические основы систем поддержки пользователей

Вместе с автоматизацией, информационными технологиями и системами приходит так же необходимость в их обслуживании. В ходе третьего исследования исследования Global Data Protection Index компании Dell EMC /25/, результаты которого были опубликованы 22 марта 2019 года – рост объемов данных составил 569% (9,70 петабайта) по сравнению с 2016 (1,45 петабайта) годом. В исследовании приняли участие 2200 руководителей, принимающих решения в области ИТ, как из государственных, так и из частных организаций с количеством сотрудников от 250 человек и выше в 18 странах и 11 отраслях промышленности. Исходя из данных исследования более чем три четверти (76%) респондентов во всем мире сталкивались с определенными видами сбоев в течение 12-месячного периода, а 27% не смогли восстановить данные с помощью имеющегося решения для их защиты, что превышает показатели 2016 года почти вдвое (14%). Незапланированные простои систем стали наиболее распространенным видами сбоев (43%) для тех, кто пользуется системами двух или более вендоров. За этим следовали атаки в целях вымогательства, которые становились препятствием к доступу данных (32%) и их потере (29%).

Хотя незапланированные простои систем носят всё более массовый характер, потеря данных обходится намного дороже. Например, те, кто столкнулся с простоями, наблюдали в среднем 20 часов простоя за последние 12 месяцев, что стоило им 526 845 долларов, в то время как те, кто столкнулся с потерей данных, потеряли в среднем 2,13 терабайта на общую сумму около 1 миллиона долларов. Кроме того, многие из тех, кто столкнулся со сбоями, также отмечали, что они имели далеко идущие последствия для предпринимательской деятельности, начиная от доверия клиентов к торговой марке и заканчивая производительностью труда работников, а также многие другие. Для решения этой проблемы в компаниях используются службы технической поддержки. Как правило, в организациях, специфика которых не направлена на предоставление информационных услуг, данными задачами занимается информационно технический отдел. В ряде случаев, задачи таких отделов передаются аутсорсинговым компаниям.

Для понимания сущности технической поддержки пользователей необходимо определить, что подразумевается под понятием технической поддержки пользователей. Для этого обратимся интернет-энциклопедии «Академик».

Служба технической поддержки – это сервисная структура, разрешающая проблемы пользователей с компьютерами (как аппаратным, так и программным обеспечением) и оргтехникой. Важная функциональная составляющая ITIL (библиотека инфраструктуры информационных технологий), позволяющая выявить проблемные участки инфраструктуры ИТ, оценить эффективность работы подразделения ИТ. /12/ Как видно из определения, под службой поддержки пользователей, чаще всего подразумевается целый отдел специалистов, основной задачей которых является максимально быстрое устранение проблем в ИТ-инфраструктуре.

Как правило, такие отделы включают в себя несколько специалистов, отвечающих за разные задачи. В грамотно организованной службе технической поддержки используется несколько линий поддержки пользователей.

Техническая поддержка часто подразделяется на уровни с целью улучшить обслуживание организации или базы клиентов. Количество уровней определяется потребностями и желаниями бизнеса или же ставится в зависимость от возможностей эффективно помочь клиентам или пользователям. Успешность организационной структуры технической поддержки связана с пониманием техническими специалистами своих зон ответственности и обязанностей, времени, в течение которого эти обязательства перед клиентами исполняются, а также от особенностей эскалирования проблемы между уровнями технической поддержки. /32/ Как правило, типичная структура технической поддержки предполагает наличие трёх уровней/линий, хотя возможно и другое их количество.

Первоначально, обращаясь в техническую службу, пользователь общается с первой линией технической поддержки. Зачастую она представлена в виде колл-центров, где операторы регистрируют обращения и используя специальные инструкции по устранению наиболее распространённых проблем пытаются удалённо помочь пользователю в решении его проблемы. Если решение проблемы невозможно средствами первой линии поддержки пользователей, то заявка передаётся на следующую, вторую линию. Вторая линия поддержки состоит из специалистов разных специальностей, в неё могут входить как системные администраторы, так и специалисты поддержки специального программного обеспечения, и специалисты поддержки специального оборудования. Если и в данном случае проблема пользователя не решена, то обращение передаётся в наивысший, третий уровень трёхуровневой модели поддержки пользователей. Специалисты этого уровня отвечают за решение наиболее сложных проблем. Специалисты этого уровня ответственны не только за помощь специалистам предыдущих уровней поддержки, но и за исследования и развитие решений для новых, появляющихся, неизвестных ранее проблем. Кроме того, иногда проблемы находятся вне компетенций технической поддержки, и локализуются в стороннем оборудовании, используемом компанией. Тогда техническая поддержка третьего уровня или же специализированный отдел обращается к поставщику, или к первичным разработчикам для углубленного анализа и конструирования решений. /31/

В IT-отделе чёткое распределение обязанностей между сотрудниками и позволяет чётко понимать, зоны охвата отдела. И ответственность каждого сотрудника за свой участок работы. Иначе говоря, каждая ошибка обзаводится своей фамилией и, если надо, стоимостью.

Как правило, в отдел назначается руководитель, руководящий действиями сотрудников и несущий прямую ответственность за деятельность своего отдела. Зачастую, руководители информационных отделов являются не только руководителями и хорошо разбираются в вверенной области, в небольших отделах зачастую именно руководитель является последней линией поддержки пользователей.

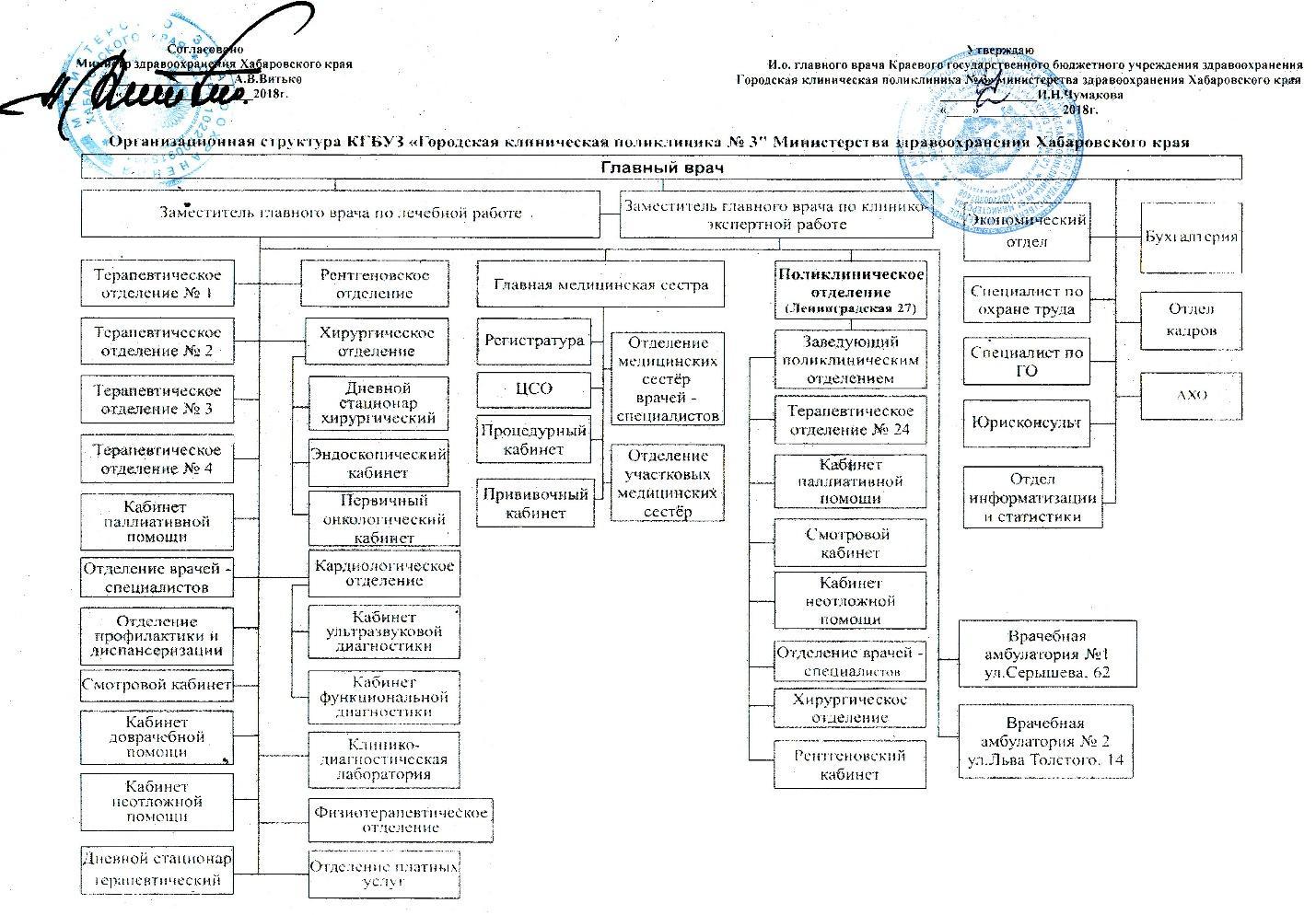
На примере КГБУЗ «Городская клиническая поликлиника №3» МЗ ХК рассмотрим строение информационного отдела. Для этого необходимо рассмотреть общую организационную структуру предприятия (рисунок 3), чтобы понимать какую позицию занимает отдел информатизации в организации.

Рисунок 3 – Организационная структура КГБУЗ «Городская клиническая поликлиника №3» МЗ ХК

Как видно из рисунка 3 в КГБУЗ «Городская клиническая поликлиника №3» МЗ ХК информационный отдел совмещён с отделом статистики и имеет одного общего руководителя, это обусловлено спецификой работы медицинской организации и необходимо для упрощения взаимодействия между информационным и статистическим отделом, необходимого для выгрузки объемов оказанных медицинских услуг. Учитывая тот момент, что в данном контексте совмещение двух отделов не является целью исследования, рассмотрим отдел информатизации отдельно. Для рассмотрения используем официально утверждённую министерством здравоохранения Хабаровского края организационную структур, размещённую на сайте поликлиники. /23/ Структуру отдела информатизации можно более детально рассмотреть на рисунке 4.

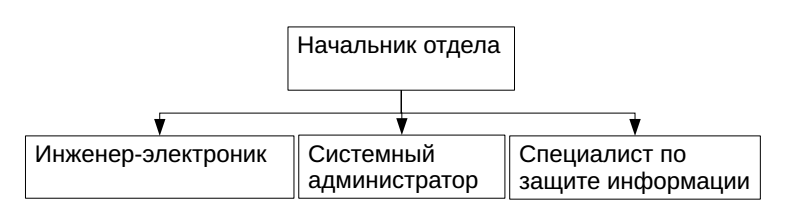


Рисунок 4 – Структура отдела информатизации КГБУЗ «Городская клиническая поликлиника №3» МЗ ХК

Как видно из рисунка 4, отдел информатизации представлен 4 специалистами, каждый из которых имеет свою область ответственности.

Инженер-электроник – отвечает за исправность и бесперебойность работы компьютерного оборудования, офисной техники. Осуществляет складской учёт и модульный ремонт неисправной техники.

Системный администратор – отвечает за бесперебойность работы локально-вычислительных сетей, информационных систем, используемых в организации, администрирует их и обучает персонал работе с ними, отвечает за поддержку пользователей.

Специалист по защите информации – отвечает за выполнение функций по защите информации, хранящейся на серверном оборудовании и других накопителях учреждения. Делает все возможное для защиты данных **от незаконного взлома и проникновения**.

Начальник отдела – отвечает за работу всего отдела, контролирует выполнение должностных обязанностей сотрудниками отдела, распределяет задачи между сотрудниками если это необходимо. Осуществляет работу во всех направлениях отдела, занимается разработкой программного обеспечения для удовлетворения нужд поликлиники.

Проанализировав структуру данного отдела, можно сделать вывод, что в данной структуре отсутствует первая линия технической поддержки и сотрудники сразу попадают на одного из специалистов. Если инцидент носит общий характер, не требующий узкоспециализированных знаний, равно как и инциденты, решение которых можно отнести к проблемам первого уровня, то такая проблема решается специалистом, принявшим сообщение об инциденте и не перенаправляется другому специалисту. Такая схема работы позволяет уменьшить время отклика на инциденты и значительно снизить сроки их устранения. Всё это сделано для обеспечения основной цели информационного отдела – поддержка пользователей и обеспечение бесперебойности их работы. Тут стоит отметить тот факт, что при возникновении инцидента при работе врача, время приёма текущего пациента и время ожидания для других пациентов, записанных к этому врачу увеличивается, что может негативно отразиться как на количестве оказанных услуг, так и на их качестве.

Зачастую, в своей работе службы технической поддержки пользователей используют специализированные информационные системы. Однозначно, целесообразность использования таких систем в первую очередь зависит от масштаба организации и количества, обслуживаемых автоматизированных рабочих мест. В небольших компаниях, где техническую поддержку осуществляет один или несколько специалистов, и рабочие места сотрудников территориально не удалены друг от друга, эффективность такой системы может быть не очень высокой. Если более крупные предприятия, особенно если рабочие помещения этого предприятия значительно удалены друг от друга, то в таком случае становится целесообразно использование специализированные систем для обеспечения бесперебойности работы предприятия. Информационные системы, предназначенные для автоматизации процесса поддержки пользователей выгодны собственникам бизнеса, менеджерам, ИТ подразделениями и конечным пользователям. Подобные сервисы в первую очередь направлены на устранение проблем, связанных с медленной реакцией ИТ служб на запросы. Также повышается качество услуг организаций, которые имеют несколько офисов и подразделений в разных регионах. Это положительно сказывается, на стабильности и безопасности ИТ инфраструктуры компании.

На сегодняшний день одной из самых популярных и эффективных методик работы в сфере информационных технологий является методология “Айтил” (ITIL) IT Infrastructure Library библиотека инфраструктуры информационных технологий. Она была разработана британской правительственной организацией для упорядочивания отношений с подрядчиками ИТ услуг.

Библиотека состоит из семи 7 книг, в которых описан весь набор процессов, необходимых для постоянного поддержания высокого качества ИТ-сервисов и повышения уровня удовлетворенности пользователей. Из 7 книг, составляющих библиотеку инфраструктуры информационных технологий, основными являются «Предоставление услуг» и «Поддержка» (Service Delivery и Service Support). В них были описаны основные процессы, составлящие ядро ITIL, в том числе процесс управления инцидентами (Incident management). Главным назначением этого процесса служит максимально быстрая ликвидация аварий, проблем с оборудованием и других проблем в ИТ-инфраструктуре. Для реализации этого процесса в организации и создается специальная служба, посредством которой происходит контакт между сотрудниками и ИТ подразделением. Эта служба может называться Центр обслуживания пользователей (Service Desk) или Центр поддержки пользователей (Help Desk).

Help desk система имеет достаточно узкое понятие, и представляет собой инструмент технической поддержки пользователей. Help desk – это информационная система, куда сотрудник может обратиться за помощью по всем вопросам и рассчитывать на быстрый отклик. Особенностями HelpDesk систем является то, что они являются частью IT-отдела и необходимы для его коммуникации с сотрудниками других отделов и клиентов компании. В первую очередь HelpDesk направлена на работу с отдельными инцидентами. Сервис принимает запрос об инциденте после чего он попадает в обработку сотрудникам ИТ-отдела и в ходе его обработки принимаются действия, направленные на решение поставленной задачи.

В отличие от систем HelpDesk, задачей Service Desk является предоставление услуг. Не смотря на схожесть результата в некоторых случаях, Service Desk имеет более широкие цели, чем в случае HelpDesk, эти цели могут быть связаны не только с функциями оказания ИТ-услуг, но и бизнес-услуг для внутренних и/или внешних целей. Service Desk предполагает значительную формализацию работы и концентрируется на контроле соглашения об уровне оказания услуг.

Подводя итог, можно заключить, что Help Desk – это первичная структура, являющаяся инструментом и необходимая для функционирования ИТ-подразделения, а Service Desk – это структура следующего уровня, в которую со временем может перерасти Help Desk. Тут важно понимать, что HelpDesk ставит только решение поступающих запросов от сотрудников, Service Desk направлена на решение значительно большего объема информации и является частью стратегической концепции компании, направленной на увеличение качества оказываемых услуг, лояльности как внешних, так и внутренних клиентов к компании, поддержку работоспособности продуктов компании.

На текущий момент современные системы поддержки пользователей придерживаются метода трехлинейной поддержки пользователей. Этот метод очень эффективен и как правило используется в больших организациях. В небольших организациях чаще всего сотрудники информационных отделов являются универсальными. Это снижает эффективность решения каждого конкретного инцидента с одной стороны, но позволяет решать большее количество проблем, меньшим штатом с другой. Внутренняя структура информационной системы должна быть согласована с используемой моделью поддержки пользователей, иначе это может негативно отразиться на работе отдела.

* 1. Бизнес-процессы поддержки пользователей и оценка эффективности использования системы

Для наглядности рассмотрим бизнес-процессы поддержки пользователей на примере отдела информатизации КГБУЗ «Городская клиническая поликлиника №3». В ходе изучения организационной структуры предприятия, структуры отдела и определения функций выбранного отдела и сотрудников, задействованных в нем, было получено представление о протекающих в данном отделе бизнес-процессах. Основным бизнес-процессом отдела является «Техническая поддержка пользователей». Используя методологию функционального моделирования и графической нотации IDEF0, произведём описание этого процесса (Рисунок 5).

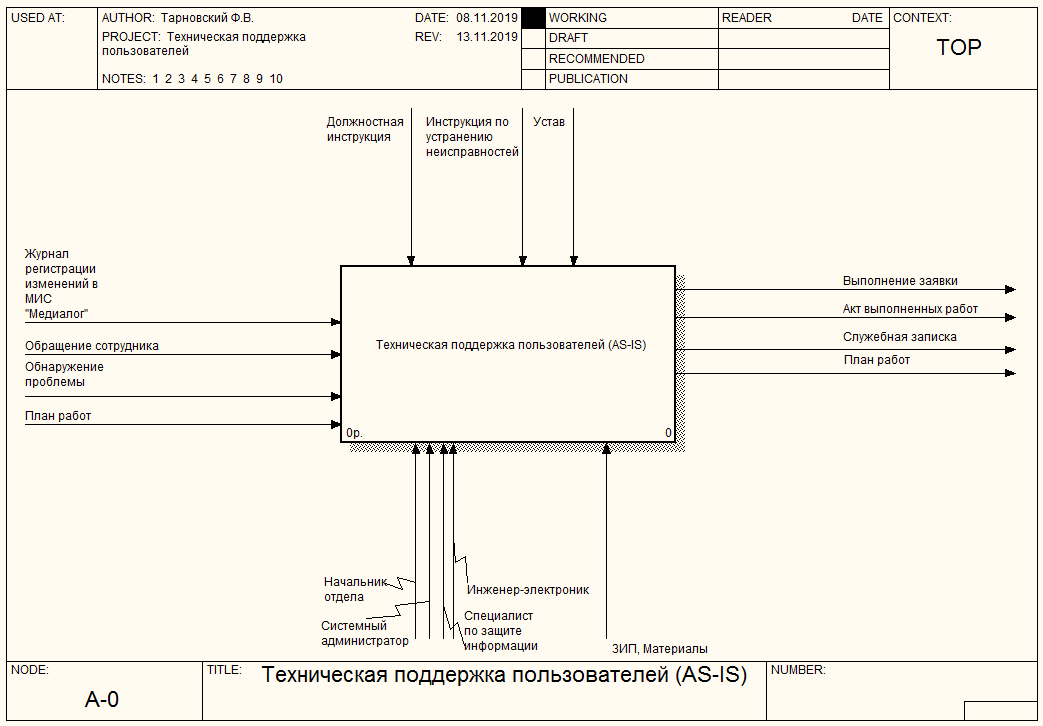


Рисунок 5 – Контекстная диаграмма бизнес-процесса технической поддержки пользователей (AS-IS)

При создании модели сначала необходимо изобразить самый высокий уровень – действие контекста. Наименование действия описывает систему непосредственно и, как правило, состоит из одного активного глагола в сочетании с обобщающим существительным, которое разъясняет цель деятельности с точки зрения самого общего взгляда на систему, в данном случае это процесс «Техническая поддержка пользователей». (Рисунок 5).

Контекстная диаграммаизображает деятельность самого верхнего уровня и обозначает границу моделирования относительно цели, возможностей и точки зрения. На контекстной диаграмме специфицируемая система представляется в виде одного единственного процесса, связанного с внешними сущностями потоками данных. Контекстная диаграмма представляет требования к системе на самом верхнем уровне – уровне взаимодействия с окружением.

Для детализации функций на составляющие части используем декомпозицию данного бизнес-процесса (Рисунок 6). Декомпозиция используется при моделировании информационных систем для разделения функций на составляющие части. 

Рисунок 6 – Декомпозиция бизнес-процесса технической поддержки пользователей (AS-IS)

Диаграммы декомпозиции предназначены для детализации функций и получаются при разбиении контекстной диаграммы на крупные подсистемы (функциональная декомпозиция) и описывающие каждый подсистему и их взаимодействие. Единственная функция, представленная на контекстной диаграмме верхнего уровня, может быть разложена на основные подфункции посредством создания дочерней диаграммы. В свою очередь, каждая из этих подфункций может быть разложена на составные части посредством создания дочерней диаграммы следующего, более низкого уровня, на которой некоторые или все функции также могут быть разложены на составные части. Каждая дочерняя диаграмма содержит дочерние блоки и стрелки, обеспечивающие дополнительную детализацию родительского блока.

Исходя из данной модели становится видно, что работа информационного отдела в КГБУЗ «Городская клиническая поликлиника №3» Министерства Здравоохранения Хабаровского Края на данный момент состоит из 4 подпроцессов. Не стоит забывать, что процесс технической поддержки отражает лишь одно из направлений работы отдела. Сам по себе данный бизнес-процесс не имеет сложной структуры, однако требует подробного анализа для понимания исследуемой предметной области. На рисунке 6 можно наблюдать 4 подпроцесса, в совокупности составляющих бизнес-процесс «Техническая поддержка пользователя». Как видно на рисунке 6 первым «Обработка обращений» является первым бизнес-процессом. Этот этап подразумевает принятие обращение сотрудников или же обнаружение проблемы специалистом отдела. Данный процесс является подготовительным и необходим для формирования заявки в дальнейшем. Прежде всего при обработке обращения необходимо определить место проблемы. Местом проблемы, или её локализацией может быть, как кабинет, рабочее место, так и определённое место в программе. Следующим этапом является определение сущности проблемы, так как обращение сотрудника зачастую не содержит четкой формулировки проблемы. Определение заявителя является заключительным процессом в обработке обращения. Этот процесс необходим для уточнения необходимой информации, касаемо проблемы, при возникновении такой необходимости. Более наглядно и схематично можно рассмотреть эту информацию на рисунке 7.

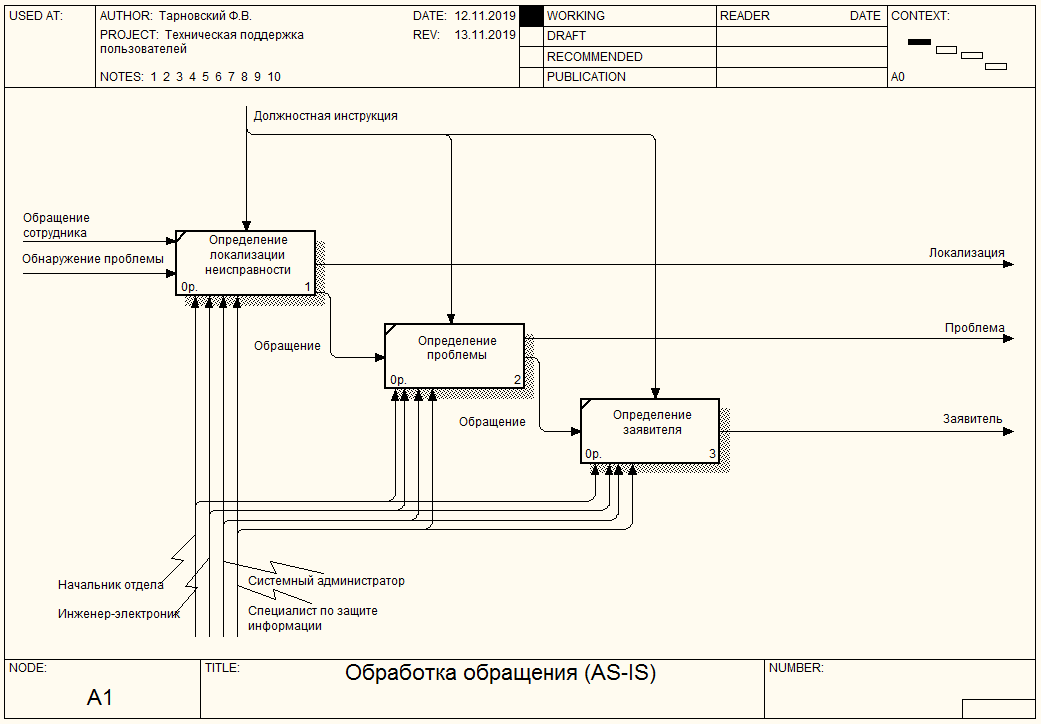


Рисунок 7 – Декомпозиция бизнес-процесса «Обработка обращений». (AS-IS)

Согласно рисунку 6, следующим бизнес-процессом является формирование заявки. Формирование заявки – очень важный процесс, в результате этого и предыдущего процесса сотрудник получает максимально полное представление об инциденте, что немаловажно для эффективного решения проблемы. Рассмотрим процесс формирования заявки подробнее на рисунке 8. Как видно из рисунка 8, на данном этапе первым делом происходит классификация проблемы. Не все проблемы требуют вмешательства профильного специалиста, незначительные и проблемы общего характера могут быть решены любым свободным сотрудником отдела. Однако, есть и проблемы, решением которых должен заниматься определённый специалист. Для этого определим класс проблемы – заявка на поддержку, заявка на обслуживание, технический сбой, программный сбой, проблемы информационной безопасности. После определения типа проблемы, обращение передаётся сотруднику, согласно его специализации.

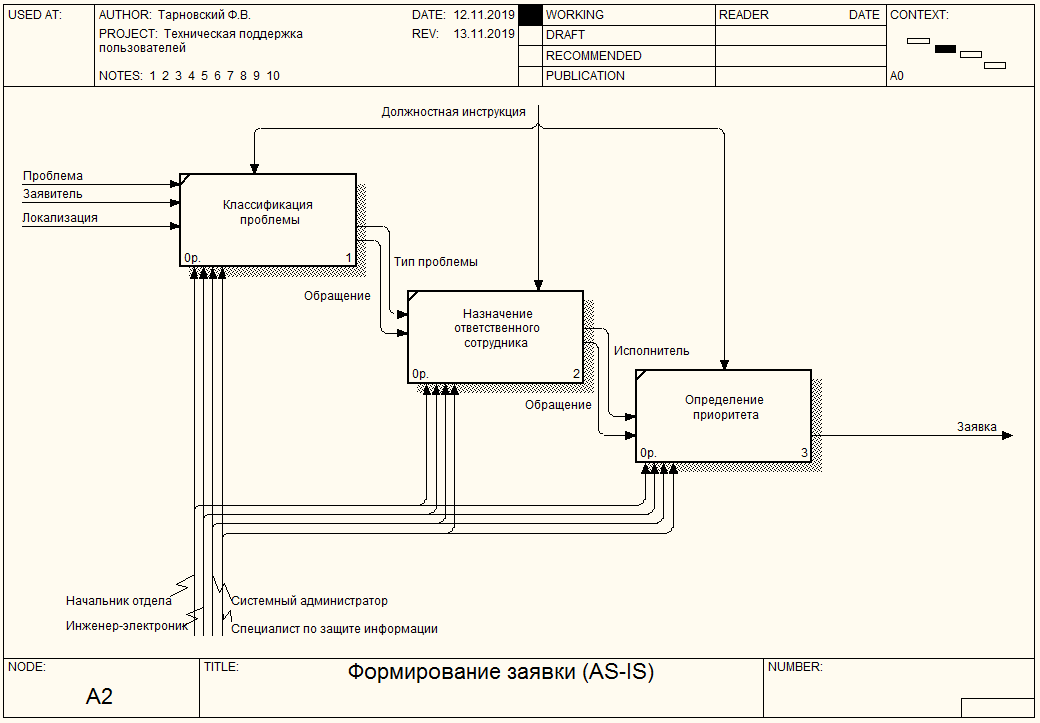


Рисунок 8 – Декомпозиция бизнес-процесса «Формирование заявки» (AS-IS)

На следующем этапе, будучи наиболее осведомленным в своей области, сотрудник определяет приоритет задачи, что в свою очередь является последним этапом формирования заявки. После того, как заявка окончательно сформирована, она переходит в стадию работы (Рисунок 9). На этапе выполнения заявки, сотрудник первым делом определяется с алгоритмом решения проблемы. Для определения алгоритма решения проблемы чаще всего используются рекомендации производителя по устранению неисправностей. После того, как алгоритм сформирован, проводятся подготовительные работы. Сюда относится как подготовка необходимых инструментов, материалов, так и запасных частей. Завершающим этапом выполнения заявки является проведение технических работ. На этапе проведения технических работ, сотрудник проводит работы, направленные на устранение проблемы. Как правило, выполнение заявки на этом этапе заканчивается.

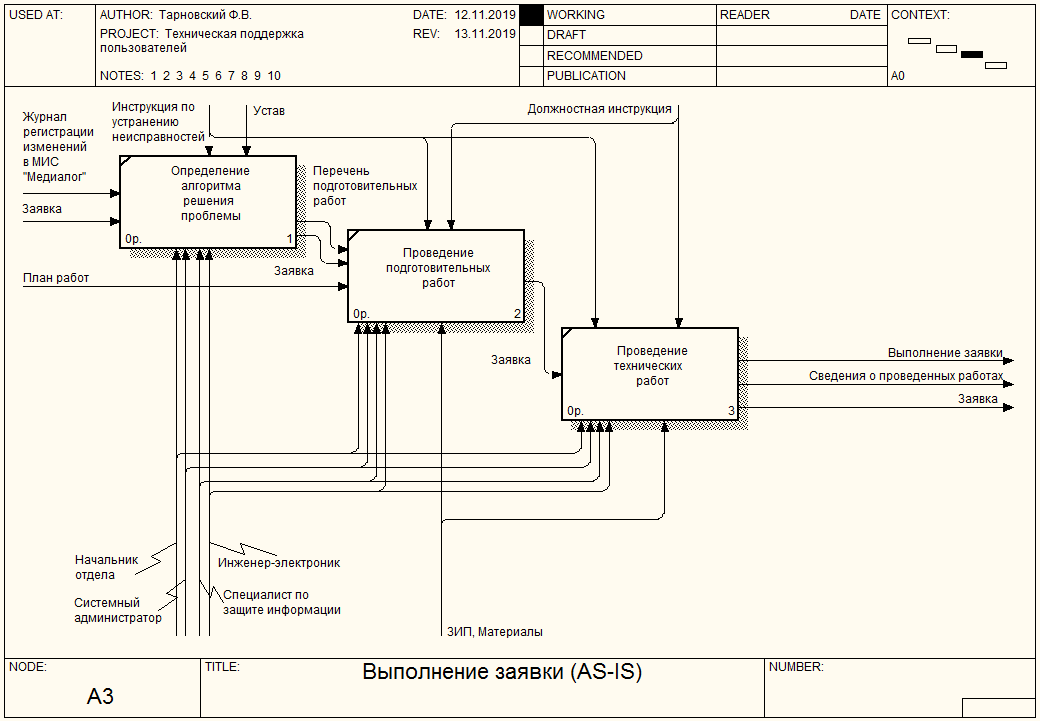


Рисунок 9 – Декомпозиция процесса «Выполнение заявки» (AS-IS)

Заключительным этапом является формирование отчётной документации. Стоит отметить, что не все заявки требуют формирование отчётной документации. Так, например, выполнение заявок на поддержку пользователей не подразумевает ведение отчётной документации. В большинстве заявок, связанных с техническими проблемами, и требующими использования запасных материалов, формируется акт о выполненных работах.

Рассмотрим изменения бизнес-процессов после внедрения информационной системы поддержки пользователей на рисунке 10. Как видно на основной контекстной диаграмме главным изменением является замена телефонного обращения, на заявку. Это позволяет регистрировать заявки и отслеживать ход их исполнения.

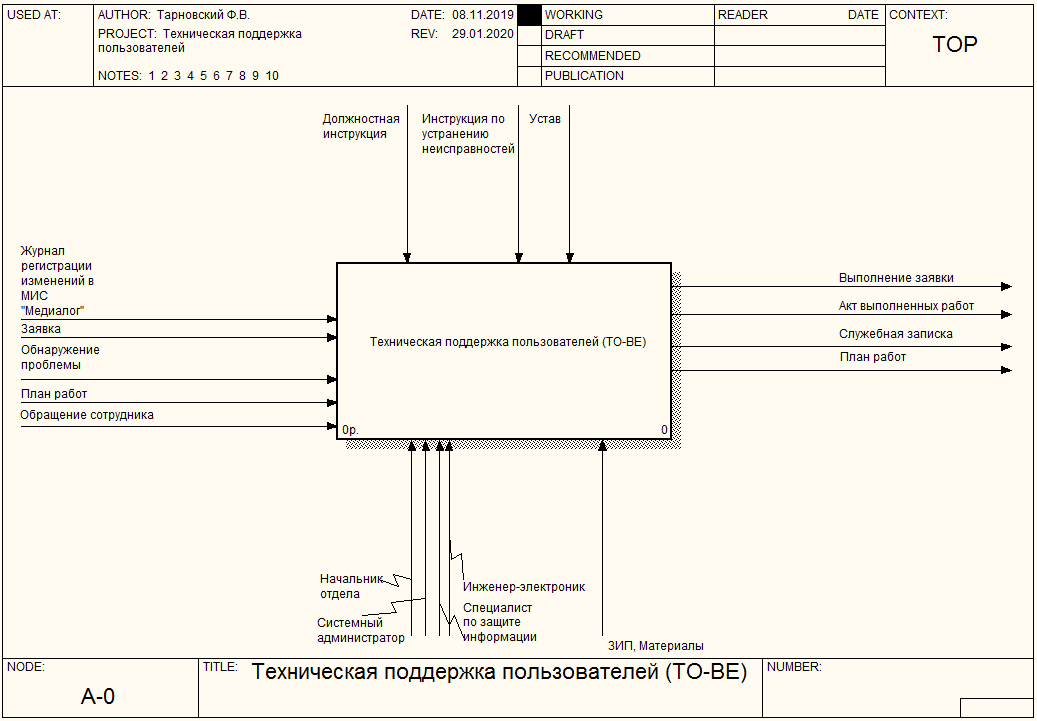


Рисунок 10 –Диаграмма процесса поддержки пользователей (TO-BE)

Так как заявка уже несёт в себе значительную часть информации, то информационная система убирает процесс обработки обращения и формирования заявки. Благодаря строгой, но в то же время простой для пользователя форме заявки информация попадает в ИС уже в готовом виде. Весь процесс в общем виде можно подробно рассмотреть на рисунке 11, а на рисунке 12 более подробно изображены процессы службы технической поддержки внутри самой ИС. Как видно из рисунка 12, после того как заявка поступает в ИС, ей назначается ответственный сотрудник, который в свою очередь определяет приоритет проблемы и пути её решения. После этого сотрудник устанавливает статус заявки, и обновляет его в зависимости от текущего состояния заявки. Так, например, заявка может иметь статус

«В работе», «Решена», «Заморожена» или «Отклонена». Статус «Заморожена» используется для заявок, для выполнения которых требуется результат действия третьих лиц. «Отклонена» используется в случаях, если заявленная проблема не является ошибкой или неисправностью.

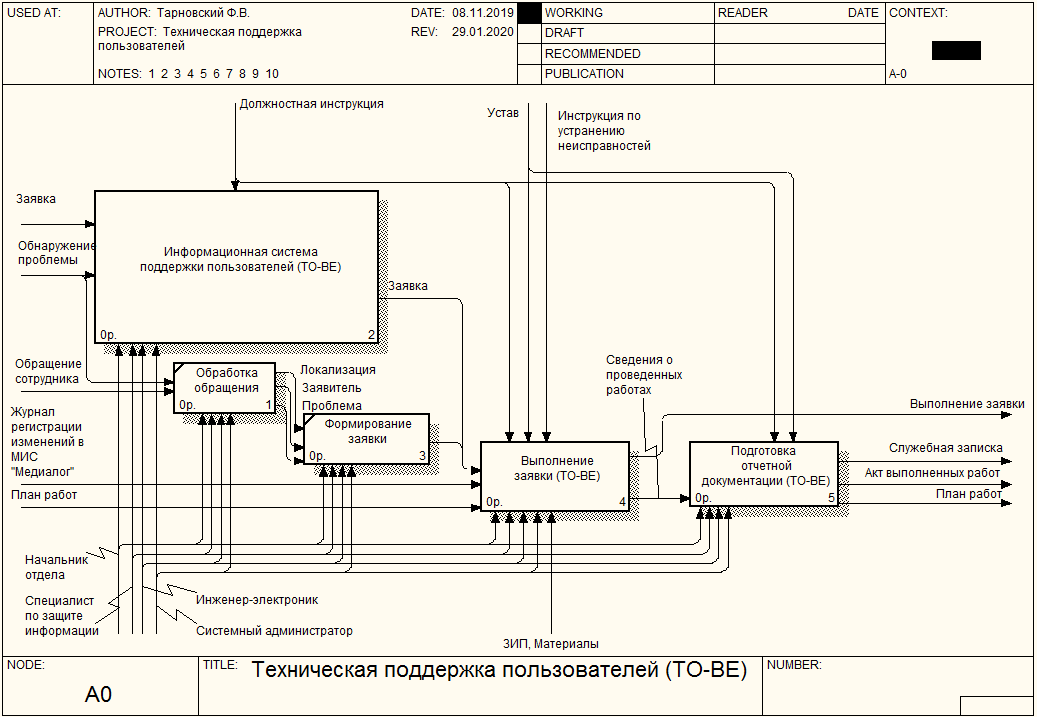


Рисунок 11 – Декомпозиция процесса технической поддержки (TO-BE)



Рисунок 12 – Декомпозиция ИС поддержки пользователей (TO-BE)

На рисунке 13 представлен переработанный процесс выполнения заявки. Как видно данный бизнес-процесс немного изменился. Главным изменением в этом случае является новый процесс «Обновление статуса заявки». Этот процесс является окончательным в жизненном цикле заявки.

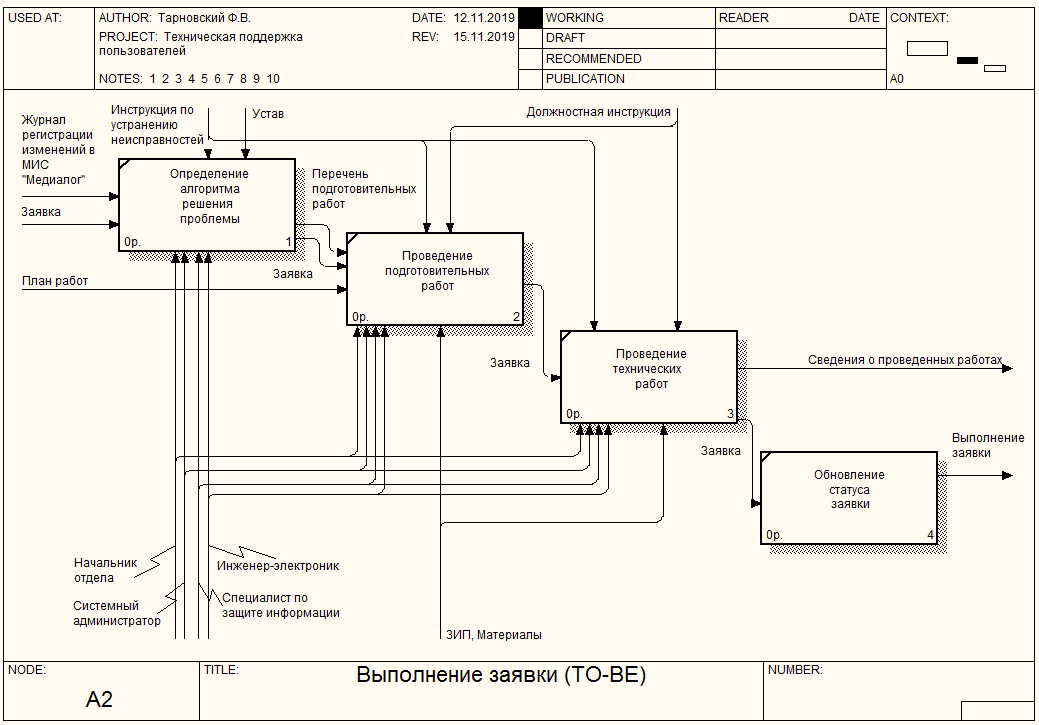


Рисунок 13 – Декомпозиция процесса выполнения заявки (TO-BE)

Внедрение информационной системы в КГБУЗ «Городская клиническая поликлиника №3» МЗ ХК позволит упростить бизнес-процессы обработки обращений и формирования заявок, а также позволит регистрировать поступающие заявки, отслеживать их выполнение, что должно положительным образом сказаться на качестве работы службы поддержки.

Как становится видно из контекстных диаграмм, информационная система забирает на себя этапы обработки обращения и частично этап формирования заявки, однако это не является основной целью внедрения информационной системы. На текущий момент, ежедневно отдел информатизации принимает от 15 до 30 заявок от сотрудников, количество заявок зависит от количества работающих сотрудников, сезонности. Несмотря на то, что количество заявок не выглядит внушительным, стоит понимать, что для решения некоторых заявок может потребоваться значительное количество времени. Так, например, полное изменение расписания приема одного терапевтического отделения может занимать до 3х часов.

В совокупности, бизнес-процессы поддержки пользователей составляют все процессы начиная от возникновения инцидента до формирования отчётной документации. В первую очередь, это конечно же процессы формирования обращения, обработки обращения, устранения инцидента и формирования отчётной документации. При этом формирование отчётной документации не во всех случаях является обязательным процессом поддержки пользователей. Для оценки эффективности использования информационных систем пользователей стоит понимать какие именно из процессов помогают решать такие системы. Рассмотренный в этой главе пример внедрения подобной системы позволит автоматизировать процессы обработки обращения и формирования заявки. Но конкретно в данном случае целью информационной системы не является уменьшение нагрузки на отдел информатизации, а возможность регистрации обращений в нерабочее время отдела. Это позволит более быстро реагировать на инциденты и избегать части простоев. Исходя из этого можно сделать вывод, что эффективность использования информационной системы поддержки пользователей заключается в сокращении затрат, вызванных сбоями в работоспособности автоматизированных рабочих мест и увеличении стабильности ИТ-инфраструктуры.

1. ПРОЕКТНАЯ ЧАСТЬ

2.1 Постановка задач и требований к информационной системе

Для того, чтобы определить структуру информационной системы необходимо сформировать перечень задач и требований к информационной системе. Эта информация позволит точно понять какие элементы необходимы в информационной системе, и её работа без них не представляется возможной, а какие могут быть избыточными.

Для начала определим перечень задач, с которые должна решать информационная система, для этого обратимся к учебному пособию «Информатика для медиков» под авторством Г. А. Хай, /27/, а также воспользуемся результатами исследований, приведённых в предыдущей главе этой работы.

Для того, чтобы определить задачи, сначала сформируем цели создания информационной системы. Для формирования целей, рассмотрим обнаруженные проблемы. Согласно проведённому в предыдущей главе исследованию бизнес-процесса технической поддержки пользователей имеют место быть следующие проблемы:

* Невозможность приема обращения;
* Долгое время ожидания реакции на обращение пользователя;
* Сложность отслеживания статуса заявок;
* Потеря обращений;
* Отсутствие статистических данных;

Основываясь на выявленных проблемах, сформируем перечень задач, необходимых для решения:

* Обеспечение круглосуточной регистрации обращений – для решения проблемы невозможности приема обращений, потери обращений и сокращения времени реакции на обращения пользователя.
* Улучшение качества работы службы технической поддержки – для отслеживания статуса заявок и формирования статистических данных, необходимых для расчета потребностей и выявления слабых мест в ИТ-структуре организации.

Существует значительное количество АИС и данные АИС существенно различаются между собой. Как следствие, выбор АИС для предприятия – достаточно нетривиальная задача. Для того, чтобы успешно её решить, необходимо хорошо знать объект внедрения (автоматизируемое предприятие), особенности его деятельности, стратегию развития и многие другие аспекты, предопределяющие характеристики закупаемой АИС. Указанные знания в конечном итоге формализуются в документе требований к АИС, на основе которого и осуществляется выбор и последующая настройка АИС. В ещё большей степени требования к АИС важны при разработке АИС на заказ. Подробнее об этом – в следующих лекциях. Определение понятия требования Л.Новиков в русской редакции нотации RUP приводит следующее определение: «Требование – это условие или возможность, которой должна соответствовать система». /19/ В IEEE Standard Glossary of Software Engineering Terminology (1990) данное понятие трактуется шире. Требование – это:

1. Условия или возможности, необходимые пользователю для решения проблем или достижения целей;
2. Условия или возможности, которыми должна обладать система или системные компоненты, чтобы выполнить контракт или удовлетворять стандартам, спецификациям или другим формальным документам;
3. Документированное представление условий или возможностей для пунктов 1 и 2. /7/

Введём ещё одно определение. Требования – это исходные данные, на основании которых проектируются и создаются автоматизированные информационные системы. Первичные данные поступают из различных источников, характеризуются противоречивостью, неполнотой, нечёткостью, изменчивостью. Требования нужны в частности для того, чтобы Разработчик мог определить и согласовать с Заказчиком временные и финансовые перспективы проекта автоматизации. Поэтому значительная часть требований должна быть собрана и обработана на ранних этапах создания АИС. Однако собрать на ранних стадиях все данные, необходимые для реализации АИС, удаётся только в исключительных случаях. На практике процесс сбора, анализа и обработки растянут во времени на протяжении всего жизненного цикла АИС, зачастую нетривиален и содержит множество подводных камней;

Существует значительное количество различных методов классификации требований /3,7/, рассмотрим наиболее существенные, такие как требования к продукту и процессу. В своей основе требования – это то, что формулирует заказчик. Цель, которую он преследует – получить хороший конечный продукт: функциональный и удобный в использовании. Поэтому требования к продукту являются основополагающим классом требований. Более подробно требования к продукту детализируются в следующих ниже классификациях. Вопросы формулирования требований к проекту, т.е. к тому, как Разработчик будет выполнять работы по созданию целевой системы, казалось бы, не лежат в компетенции Заказчика. Без регламентации процесса Заказчиком легко можно было бы обойтись, если бы все проекты всегда выполнялись точно и в срок. Однако, к сожалению, мировая статистика результатов программных проектов говорит об обратном. Заказчик, вступая в договорные отношения с Разработчиком, несёт различные риски, главными из которых является риск получить продукт с опозданием, либо ненадлежащего качества. Основные мероприятия по контролю и снижению риска – регламентация процесса создания программного обеспечения и его аудит. Насколько подробно Заказчику следует регламентировать требования к проекту – вопрос риторический. Ответ на него зависит о множества факторов, таких, как ценность конечного продукта для Заказчика, степень доверия Заказчика к Разработчику, сумма подписанного контракта, увязка срока сдачи продукта в эксплуатацию с бизнес-планами Заказчика. /16,18,29/ Однако, со всей определённостью можно сказать следующее:

* Регламентация процесса Заказчиком позволяет снизить его риски;
* Мероприятия Заказчика по регламентации процесса приводят к дополнительным накладным расходам.

Требуется найти разумный компромисс между степенью контроля рисков и величиной расходов. В качестве требований к проекту могут быть внесён регламент отчётов Разработчика, совместных семинаров по оценке промежуточных результатов, определены характеристики компетенций участников рабочей группы, исполняющих проект, их количество, указана методология управления проектом. Ниже сформулирован пример формулировки требования к оффшорному проекту (Заказчик и Разработчик физически находятся в разных государствах) – в этой ситуации Заказчику требуется жёсткий контроль над Разработчиком.

* Разработчик представляет Заказчику согласованный план работ c детализацией (WorkBreakdownStructure - WBS) с точностью до конкретных исполнителей.
* Разработчик осуществляет ежедневные сборки, регрессионное тестирование компонент разрабатываемого продукта и тестирование продукта в целом.
* Все управленческие и проектные артефакты, исходные коды и тестовые примеры размещаются в режиме online в интегрированной среде разработки Rational ClearCase® с возможностью для Заказчика осуществления online-мониторинга на базе web-технологий.

Уровни требований Внедрение ИС на предприятии всегда преследует конкретные бизнес-цели – такие, как, например, повышение прозрачности бизнеса, сокращение сроков обработки информации, экономия накладных расходов и т.д. Современные информационные системы – это крупные программные системы, содержащие в себе множество модулей, функциональных, интерфейсных элементов, отчётов и т.д. Как охватить единым взором такие разнородные вещи, как цели, преследуемые топ-менеджером предприятия Заказчика, описание интерфейса пользователя и характеристики модуля, осуществляющего расчёт себестоимости изделия? К счастью, человечество уже давно изобрело приёмы борьбы со сложностью, широко применяемые в моделировании сложных объектов – абстракцию и декомпозицию. Применительно к дисциплине анализа требований к программным системам эти принципы работают следующим образом. Требования разделяются по уровням. Уровни требований связаны, с одной стороны, с уровнем абстракции системы, с другой – с уровнем управления на предприятии. Обычно выделяют три уровня требований. На верхнем уровне представлены так называемые бизнес-требования (business requirements). Примеры бизнес-требования: система должна сократить срок оборачиваемости обрабатываемых на предприятии заказов в три раза. Бизнес-требования обычно формулируются топ-менеджерами, либо акционерами предприятия. Следующий уровень – уровень требований пользователей (user requirements). Пример требования пользователя: система должна представлять диалоговые средства для ввода исчерпывающей информации о заказе, последующей фиксации информации в базе данных и маршрутизации информации о заказе к сотруднику, отвечающему за его планирование и исполнение. Требования пользователей часто бывают плохо структурированными, дублирующимися, противоречивыми. Поэтому для создания системы важен третий уровень, в котором осуществляется формализация требований. Третий уровень – функциональный (functional requirements). Пример функциональных требований (или просто функций) по работе с электронным заказом: заказ может быть создан, отредактирован, удалён и перемещён с участка на участок. Существуют объективные противоречия между требованиями различных уровней. Так, очевидным бизнес-требованием является требование о полноте информации, собираемой на рабочих местах пользователей в единую базу данных. Чем полнее информация – тем глубже база для анализа деятельности и принятия решений. С другой стороны, конкретному пользователю системы вполне может быть достаточно использования только той части информации, которая влияет на выполнение его основных функций. Важные правила внедрения и использования АИС на предприятии – «Одна точка сбора», «Данные собираются там, где они появляются». Использование этих правил позволяет избежать затрат на необоснованное дублирование информации и, что важнее – потерь от ошибок учёта, неизбежно возникающих при дублировании точек ввода. Внедрение АИС на предприятии приводит к необходимости оснащения всех точек ввода информации автоматизированными рабочими местами (АИС), обучению персонала и, зачастую, оптимизации и повышению уровня формализации рабочих процессов, выполняемых персоналом. Поэтому внедрение АИС – непростой процесс, часто требующий «перекройки человеческого материала» и встречающий сопротивление со стороны пользователей, которые не готовы, либо не хотят работать по-новому. Системные требования и требования к программному обеспечению существуют различные трактовки понятия «Системные требования» (system requirements). К. Вигерс формулирует данный термин, как «высокоуровневые требования к продукту, которые содержат многие подсистемы, то есть системе» /7/. При этом под системой понимается программная, программно-аппаратная, либо человеко-машинная система. Данная система является сложной, структурированной системой и системные требования являются подмножеством функциональных требований к продукту. В данное подмножество целесообразно относить наиболее важные, существенные требования, которые относятся в целом к системе и не содержат избыточной детализации. INCOSE (International Council on Systems Engineering) даёт более детальное определение системы: «комбинация взаимодействующих элементов, созданная для достижения определенных целей; может включать аппаратные средства, программное обеспечение, встроенное ПО, другие средства, людей, информацию, техники (подходы), службы и другие поддерживающие элементы». Таким образом, происходит разделение между системными требованиями, как обобщающему понятию и требованиями к программному обеспечению, как выделенному подмножеству системных требований, направленных исключительно на программные компоненты системы. Этот же подход прослеживается в стандарте ГОСТ Р ИСО/МЭК 12207/99: работы, связанные с системой в целом и с программным обеспечением, выделяются в отдельные группы в целях удобства оперирования. /10/ В практике компьютерной инженерии бытует другой, более узкий контекст использования данного понятия: под системными требованиями в узком смысле понимается требования, выдвигаемые прикладной программной системой (в частности – информационной) к среде своего функционирования (системной, аппаратной). Пример таких требований – тактовая частота процессора, объём памяти, требования к выбору операционной системы. Функциональные, нефункциональные требования и характеристики продукта Функциональные требования регламентируют функционирование или поведение системы (behavioral requirements). Функциональные требования отвечают на вопрос «что должна делать система» в тех или иных ситуациях. Функциональные требования определяют основной «фронт работ» Разработчика, и устанавливают цели, задачи и сервисы, предоставляемые системой Заказчику. Функциональные требования записываются, как правило, при посредстве предписывающих правил: «система должна позволять кладовщику формировать приходные и расходные накладные». Другим способом являются так называемые варианты использования (uses cases) – популярный и весьма продуктивный способ представления требований. Нефункциональные требования, соответственно, регламентируют внутренние и внешние условия или атрибуты функционирования системы. К. Вигерс /7/ выделяет следующие основные группы нефункциональных требований:

* Внешние интерфейсы (External Interfaces)
* Атрибуты качества (Quality Attributes)
* Ограничения (Constraints).

Среди внешних интерфейсов в большинстве современных АИС наиболее важным является интерфейс пользователя (User Interface, UI). Кроме того, выделяются интерфейсы с внешними устройствами (аппаратные интерфейсы), программные интерфейсы и интерфейсы передачи информации (коммуникационные интерфейсы). Основные атрибуты качества:

* + Применимость
  + Надежность,
  + Производительность,
  + Эксплуатационная пригодность,

Ограничения - формулировки условий, модифицирующих требования или наборы требований, сужая выбор возможных решений по их реализации. /7/ выбор платформы реализации и/или развертывания (протоколы, серверы приложений, баз данных, ...), которые, в свою очередь, могут относиться, например, к внешним интерфейсам.

К.Вигерс формулирует характеристику продукта, как набор логически связанных функциональных требований, которые обеспечивают возможности пользователя и удовлетворяют бизнес-цели. /7/ Существует и более общий взгляд на данное понятие: характеристики продукта могут быть как относящимся к функциональным, так и к нефункциональным требованиям и могут изменяться от версии к версии продукта». С.Орлик отмечает, что «с точки зрения инженерии требований, характеристики продуктов являются самостоятельным артефактом, который может быть соотнесен как с функциональными требованиями, так и с нефункциональными». /20/ Роль характеристик проявляется в первую очередь в области маркетинга: не всякий потенциальный потребитель продукта станет читать его функциональные описания, а набор ключевых характеристик, характеризующих конкурентные преимущества, можно сделать лаконичным и уместить на одной странице рекламной листовки, либо напечатать на компакт-диске.

Основными задачами в данном случае выступают улучшение качества работы службы технической поддержки и обеспечение круглосуточной регистрации обращений. Это значит, что бизнес-требованием в данном случае является снижение времени простоя пользователей. Для обеспечения этих требований необходимо сформировать требования пользователей к информационной системе. Очевидно, что для удобства использования информационная система должна иметь пользовательский интерфейс, максимально простой по своей структуре. Все действия в информационной системе должны быть интуитивно понятны простому пользователю. Это необходимо, чтобы использование информационной системы не приводило к ещё большим простоям. Для обеспечения круглосуточной регистрации обращений, необходимо наличие простой формы для быстрой отправки обращений. Для специалистов отдела должна присутствовать возможность просмотра списка активных заявок с возможностью просмотра всех, новых и только своих заявок. Необходимой также является функция просмотра подробных сведений о выбранной заявке и возможность изменения её статуса, исполнителя, приоритета и типа заявки. Так как система подразумевает работу нескольких администраторов одновременно, то необходимо наличие авторизации для администраторов. Система должна позволять управлять пользователями посредством внутренних инструментов. В управление пользователями входят такие функции как «Добавление нового пользователя», «Изменение пользователя» «Просмотр всех пользователей». Учитывая тот момент, что доступ к системе должен быть обеспечен с любого рабочего места, то информационная система должна быть кроссплатформенной.

* 1. Выбор средств реализации информационной системы

Исходя из поставленных целей и задач, и функционала информационной системы поддержки пользователей, сформулируем требования к программной реализации информационной системы.

Так как предполагается использование данной программной реализации в различных кабинетах, основным требованием к разрабатываемому программному обеспечению стала кроссплатформенность. Конфигурацию машины, на которой будет использоваться данный программный продукт, предугадать сложно, так как в организации используется больше количество разных по конфигурации машин. Так же в ином случае установка и обслуживание приложения на всех 170-и АРМ очень трудозатратный процесс, поэтому информационная система должна обеспечивать возможность работы на разных операционных системах без предварительных настроек и редактирования исходного кода.

Опираясь на вышесказанное, становится очевидно, что не все машины будут соответствовать современным стандартам системных требований, поэтому требовательность программной реализации к аппаратной части так же является приоритетной задачей.

Так же стоит отметить, что для целостности рабочего процесса и привлекательности конечного продукта важна возможность запуска разработанного приложения без длительного процесса установки или предварительной настройки и компактный размер.

Опираясь на поставленные требования, был сделан выбор реализовать проект в формате web-приложения. Такой формат соответствует всем поставленным требованиям: web-приложение может быть запущен на любой платформе, поддерживающей браузер; оно обладает небольшим размером и не требует установки или настройки; возможна реализация, которая будет не требовательна к машине, на которой будет запущен программный продукт.

Среди прочих преимуществ web-платформы можно отметить удобство модернизации, доступность исходного кода и возможности простой сборки отдельных модулей в общий проект средствами общей страницы-оглавления, что особенно актуально учитывая специфику данной работы.

На сегодняшний день существует несчетное количество средств работы с web, однако легко можно выделить ряд средств, являющийся неотъемлемой частью разработки web-приложений. Для соблюдения сформулированных выше требований было принято решение придерживаться базовых инструментов и использовать дополнительные инструменты только в случае острой необходимости.

После того, как было принято решение о реализации проекта в формате web-приложения, нужно определиться с инструментарием, посредством которого это приложение будет разрабатываться.

HTML (HyperText Markup Language) - основной язык разметки Всемирной паутины. Первоначально HTML был в основном разработан как язык для семантического описания научных документов. Однако его общий дизайн позволил ему адаптироваться в последующие годы для описания ряда других типов документов и приложений. На сегодняшний день HTML является основным языком разметки для создания веб-страниц и web-приложений, и, вполне вероятно, самый используемый язык разметки в мире сегодня.

Гипертекст - это текст который может содержать содержащий ссылки на другие тексты, а также графику, видео и звук.

Язык разметки (Markup Language) - система для комментирования документа тегами, синтаксически отличающихся от текста. Например, пробелы между словами указывают границы слов, запятые указывают границы фразы, а периоды указывают границы предложения.

Тег – именованная метка, дескриптор. Текст, содержащийся между начальным и конечным тегом, отображается и размещается в соответствии со свойствами, указанными в начальном теге.

Кроме повсеместного использования HTML обязан браузерам еще одним достоинством. Так как язык разметки HTML интерпретируется браузерами, результат интерпретации может быть отображен экране любого устройства и любой системы, поддерживающей интернет-браузер, что обеспечивает работу практически на любом устройстве.

Протокол HTML подразумевает передачу страниц от сервера в виде простого текста, однако использование протокола HTTPS позволяет использовать шифрование в процессе передачи текста.  
 Для обеспечения удобного и гибкого управления внешнего вида веб-страниц целсообразно использование каскадных таблиц стилей. Cascading Style Sheets — каскадные таблицы стилей. CSS является формальным языком описания внешнего вида документа, реализованного с помощью языка разметки. Он преимущественно используется как средство для описания и оформления внешнего вида web-страниц, реализованных с помощью языков разметки HTML и XHTML, но также может применяться к любым XML-документам.

CSS используется разработчиками веб-страниц для задания шрифтов, цветов и расположения отдельных блоков и других аспектов представления внешнего вида этих веб-страниц. Основной целью разработки CSS являлось отделение описания логической структуры веб-страницы (которое производится с помощью HTML или других языков разметки) от описания внешнего вида этой веб-страницы (которое теперь производится с помощью формального языка CSS). Такое разделение может увеличить доступность документа, предоставить большую гибкость и возможность управления его представлением, а также уменьшить сложность и повторяемость в структурном содержимом.

Кроме того, CSS позволяет представлять один и тот же документ в различных стилях или методах вывода, таких как экранное представление, печатное представление, чтение голосом (специальным голосовым браузером или программой чтения с экрана), или при выводе устройствами, использующими шрифт Брайля.

Правила CSS пишутся на формальном языке CSS. Правила могут располагаться как в самом веб-документе, внешний вид которого они описывают, так и во внешних файлах, имеющих формат CSS. Формат CSS — это текстовый файл, в котором содержится перечень правил CSS и комментариев к ним.

Исходя из поставленных задач было принято решение о проведении процесса обработки информации преимущественно на стороне сервера. Таким образом, все вычисления можно будет проводить на стороне сервера. Учитывая небольшой поток заявок (до 30 в день), нет необходимости в выделении больших мощностей, для реализации проекта.

Среди существующих языков разработки клиентской и серверной части web-приложений лидирующие позиции занимает JavaScript. На сегодняшний день сложно сказать, что у JavaScript есть серьезные соперники. Альтернативные варианты либо компилируют свой код в JavaScript (CoffeeScript, ClojureScript, Opal, Kaffeine) либо сильно уступают ему в доступном функционале в связи со своей новизной (Dart).

Рассмотрим подробнее JavaScript. JavaScript (часто просто JS) — это легковесный, интерпретируемый или JIT-компилируемый, объектно-ориентированный язык. Наиболее широкое применение находит как язык сценариев веб-страниц, но также используется и в других программных продуктах, например, node.js или Apache CouchDB. JavaScript это прототипно-ориентированный, мультипарадигменный язык с динамической типизацией, который поддерживает объектно-ориентированный, императивный и декларативный (например, функциональное программирование) стили программирования.

Как уже упоминалось ранее JavaScript может запускаться на стороне клиента Интернета и используется для программирования того, как веб-страницы будут вести себя при наступлении каких-либо событий. JavaScript прост в изучении, а также это мощный скриптовый язык, широко используемый для контролирования поведения веб-страниц.

Для упрощения работы с JavaScript и возможности реализации серверной и клиентской части программного кода на одном языке программирования используем программную платформу Node.js. Node.js это программная платформа, основанная на движке V8 (транслирующем JavaScript в машинный код), превращающая JavaScript из узкоспециализированного языка в язык общего назначения. Node.js добавляет возможность JavaScript взаимодействовать с устройствами ввода-вывода через свой API (написанный на C++), подключать другие внешние библиотеки, написанные на разных языках, обеспечивая вызовы к ним из JavaScript-кода. Node.js применяется преимущественно на сервере, выполняя роль веб-сервера, но есть возможность разрабатывать на Node.js и десктопные оконные приложения (при помощи NW.js, AppJS или Electron для Linux, Windows и macOS) и даже программировать микроконтроллеры (например, tessel и espruino). В основе Node.js лежит событийно-ориентированное и асинхронное (или реактивное) программирование с неблокирующим вводом/выводом.

Несмотря на то, что разрабатывать web-приложения можно в любом текстовом редакторе, правильный выбор инструмента может существенно облегчить и ускорить процесс разработки.

Visual Studio Code – это бесплатный редактор кода с открытым исходным кодом, оптимизированный для создания и отладки современных веб-приложений и облачных приложений.

Visual Studio Code выделяется на фоне конкурентов благодаря системе собственной разработке Microsoft IntelliSense, которая обеспечивает подсветку синтаксиса, интеллектуальные доработки на основе типов переменных, определений функций и импортированных модулей, а также стоит отметить интеграцию с git, встроенный дебаггер и большое сообщество, разрабатывающее новые расширения и модули.

В связи с необходимостью разрабатывать проект на нескольких АРМ, возникла необходимость в удобной системе контроля версий, которая бы позволяла следить за прогрессом проделанной работы. В качестве такой системы был выбран git.

Git – это бесплатная система управления распределенной версией с открытым исходным кодом, предназначенная для быстрой и эффективной работы с проектами как небольших, так и очень крупных проектов.

Git маленький и простой в обучении клиент, отличающийся молниеносной быстротой. Он выгодно выделяется на фоне других систем контроля версий, таких как Subversion, CVS, Perforce и ClearCase простотой локальных ветвлений, наличием удобной области индексации и возможностью ведения нескольких рабочих процессов одновременно.

В качестве хранилища репозитория был выбран сервис GitHub.

GitHub — крупнейший веб-сервис для хостинга IT-проектов и их совместной разработки. Основан на системе контроля версий Git и разработан на Ruby on Rails и Erlang компанией GitHub, Inc. Сервис абсолютно бесплатен для проектов с открытым исходным кодом и предоставляет им все возможности.

GitHub обладает широким функционалом API, что позволяет интегрировать его в другие приложения. В процессе данной работы была произведена интеграция GitHub в мессенджер Slack, что позволило удобно отслеживать изменения репозитория.

База данных – это совокупность структурированных и взаимосвязанных данных и методов, обеспечивающих добавление выборку и отображение данных.

Рассматривая общий смысл понятия БД, стоит обратить внимание на то, что с самого начала развития вычислительной техники образовались два основных направления ее использования. Первое направление - применение вычислительной техники для выполнения численных расчетов, которые слишком долго или вообще невозможно производить вручную. Становление этого направления способствовало интенсификации методов численного решения сложных математических задач, развитию класса языков программирования, ориентированных на удобную запись численных алгоритмов, становлению обратной связи с разработчиками новых архитектур ЭВМ.

Второе направление, возникшее несколько позже первого, которое непосредственно касается нашей темы, это использование средств вычислительной техники в автоматических или автоматизированных информационных системах. В самом широком смысле информационная система представляет собой программный комплекс, функции которого состоят в поддержке надежного хранения информации в памяти компьютера, выполнении специфических для данного приложения преобразований информации и/или вычислений, предоставлении пользователям удобного и легко осваиваемого интерфейса. Обычно объемы информации, с которыми приходится иметь дело таким системам, достаточно велики, а сама информация имеет достаточно сложную структуру. Классическими примерами информационных систем являются банковские системы, системы резервирования авиационных или железнодорожных билетов, мест в гостиницах.

Из этого следует, что при создании базы данных следует особое внимание уделить выбору СУБД. При выборе СУБД необходимо учитывать множество аспектов, каждый из которых накладывает определенные ограничения на выбор продукта. Это не только технические показатели, но и экономические соображения, учет рынка, а также определенные культурологические аспекты, связанные с базой данных, поэтому основными требованиями к СУБД является:

Хранение, извлечение и обновление данных. СУБД должна предоставлять пользователям возможность сохранять, извлекать и обновлять данные в базе данных. Это самая фундаментальная функция СУБД. Из предыдущего ясно, что способ реализации этой функции в СУБД должен позволять скрывать от конечного пользователя внутренние детали физической реализации системы (например, файловую организацию или используемые структуры хранения).

Каталог, доступный конечным пользователям. СУБД должна иметь доступный конечным пользователям каталог, в котором хранится описание элементов данных. Ключевой особенностью архитектуры ANSI-SPARC является наличие интегрированного системного каталога с данными о схемах, пользователях, приложениях и т.д. Предполагается, что каталог доступен как пользователям, так и функциям СУБД. Системный каталог, или словарь данных, является хранилищем информации, описывающей данные в базе данных (по сути, это - метаданные). Появляются новые возможности организации поддержки целостности данных.

Сервисы управления параллельностью. СУБД должна иметь механизм, который гарантирует корректное обновление базы данных при параллельном выполнении операций обновления многими пользователями. При этом параллельный доступ сравнительно просто организовать, если все пользователи выполняют только чтение данных, поскольку в этом случае они не могут помешать друг другу. Однако, когда несколько пользователей одновременно получают доступ к БД, конфликт с нежелательными последствиями легко может возникнуть, например, если хотя бы один из них попытается обновить данные.

СУБД должна гарантировать, что при одновременном доступе к базе данных многих пользователей подобных конфликтов не произойдет.

Сервисы восстановления. При обсуждении поддержки транзакций упоминалось, что при сбое транзакции база данных должна быть возвращена в непротиворечивое состояние, что должно гарантироваться возможностями СУБД.

Сервисы контроля доступа к данным. СУБД должна иметь механизм, гарантирующий возможность доступа к базе данных только санкционированных пользователей. Термин "безопасность" относится к защите базы данных от преднамеренного или случайного несанкционированного доступа. Предполагается, что СУБД обеспечивает механизмы подобной защиты данных.

Поддержка обмена данными. СУБД должна обладать способностью к интеграции с коммуникационным программным обеспечением с целью организации доступа удаленных пользователей к централизованной БД (в рамках системы распределенной обработки).

Службы поддержки целостности данных. СУБД должна обладать инструментами контроля за тем, чтобы данные и их изменения соответствовали заданным правилам.

Целостность базы данных означает корректность и непротиворечивость хранимых данных и выражается в виде ограничений или правил сохранения непротиворечивости данных, которые не должны нарушаться в базе.

Службы поддержки независимости от данных. СУБД должна обладать инструментами поддержки независимости программ от структуры базы данных.

Понятие независимости от данных уже рассматривалось выше. Обычно она достигается за счет реализации механизма поддержки представлений или подсхем. Физическая независимость от данных достигается довольно просто, так как обычно имеется несколько типов допустимых изменений физических характеристик базы данных, которые никак не влияют на представления. Однако добиться полной логической независимости от данных сложнее. Как правило, система легко адаптируется к добавлению нового объекта, атрибута или связи, но не к их удалению. В некоторых системах вообще запрещается вносить любые изменения в уже существующие компоненты логической схемы.

Вспомогательные службы. СУБД должна предоставлять некоторый набор различных вспомогательных служб. Вспомогательные утилиты обычно предназначены для оказания помощи АБД в эффективном администрировании базы данных. Одни утилиты работают на внешнем уровне, а потому они, в принципе, могут быть созданы самим АБД, тогда как другие функционируют на внутреннем уровне системы и потому должны быть предоставлены самим разработчиком СУБД.

Всем этим требованиям в полной мере удовлетворяет нереляционная база данных MongoDB. MongoDB в отличии от SQL является новой технологией, стремительно набирающей популярность в web-разработке. Причина такого стремительного роста популярности MongoDB проста – в MongoDB используется гибкий JSON-формат документов, что позволяет решать простые задачи просто и удобно. Также в MongoDB встроена достаточно простая масштабируемость с использованием технологии шардинга. Конечно, MongoDB не поддерживает транзакции, но он поддерживает атомарные операции над документом. Это значит, что с точки зрения одного документа операция будет атомарна.

Паттерн MVC так же поможет упростить разработку программного продукта. Он является одним из распространенных паттернов, применяемых в веб-приложениях. В том числе он применяется и в приложениях на Node.js.

Паттерн MVC включает ряд компонентов:

* Модели определяют структуру и логику используемых данных
* Представления (views) определяет визуальную часть, как данные будут отображаться.
* Контроллеры обрабатывают входящие http-запросы, используя для обработки модели и представления, и отправляет в ответ клиенту некоторый результат обработки, нередко в виде html-кода.
* Система маршрутизация как дополнительный компонент сопоставляет запросы с маршрутами и выбирает для обработки запросов определенный контроллер.

В общем случае, когда к приложению приходит запрос, система маршрутизации выбирает нужный контроллер для обработки запроса. Контроллер обрабатывает запрос. В процессе обработки он может обращаться к данным через модели и для рендеринга ответа использовать представления. Результат обработки контроллера отправляется в ответ клиенту. Нередко ответ представляет html-страницу, которую пользователь видит в своем браузере.

Выбор средств реализации информационной системы, равно как и проектирование играет огромную роль в сложном процессе разработки. Правильный выбор средств реализации очень важен и имеет прямое влияние на качество разработки программного продукта. Учитывая то, что одним из требований к информационной системе была кроссплатформенность, то целесообразным будет реализация программного продукта в виде web-приложения. Web-приложение может быть запущено на любой платформе, поддерживающей браузер. В качестве языка разметки в работе используется HTML, а для удобства управления внешним видом страниц – CSS. Наиболее распространённым из-за ряда преимуществ язык программирования JavaScript был создан для разработки web-приложений, из-за чего является оптимальным вариантом для разработки информационной системы. Использование программной платформы Node.js позволит реализовывать серверную и клиентской части программного кода на одном языке программирования, что положительным образом отразится на процессе разработки. В качестве базы данных, для большей гибкости и удобства разработки используем MongoDB, использование нереляционной базы данных также позволит существенно сократить время разработки. Для работы с программным кодом используем Visual Studio Code так как это бесплатный, удобный и хорошо оптимизированный редактор кода с широким функционалом для создания и отладки современных веб-приложений и облачных приложений.

* 1. Разработка информационной системы

Для работы с программным кодом используем Visual Studio Code. Для разработки приложения используем паттерн MVC, поэтому первым делом определяем такие каталоги как “models”, “views”, “controllers” и “routes” они станут каркасом проекта. Создадим основной файл приложения – app.js, для работы приложения так же потребуются модули “express”, “mongoose”, “express-session”. Для их хранения используем каталог “node\_modules”. В итоге проект будет выглядеть следующим образом:

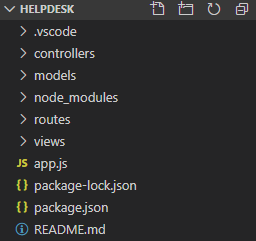


Рисунок 14 - Состав проекта

Следующим этапом будет создание сервера и подключение базы данных. Для более удобной работы с сервером используем модуль “Express”, а для работы MongoDB воспользуемся модулем “Mongoose”.



Рисунок 15 - Подключение сервера

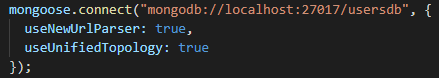


Рисунок 16 - Подключение базы данных

Следующим шагом определим начальную страницу проекта, для этого создадим в каталоге views файл homepage.hbs. После подключения к серверу главная страница проекта будет иметь следующий вид:

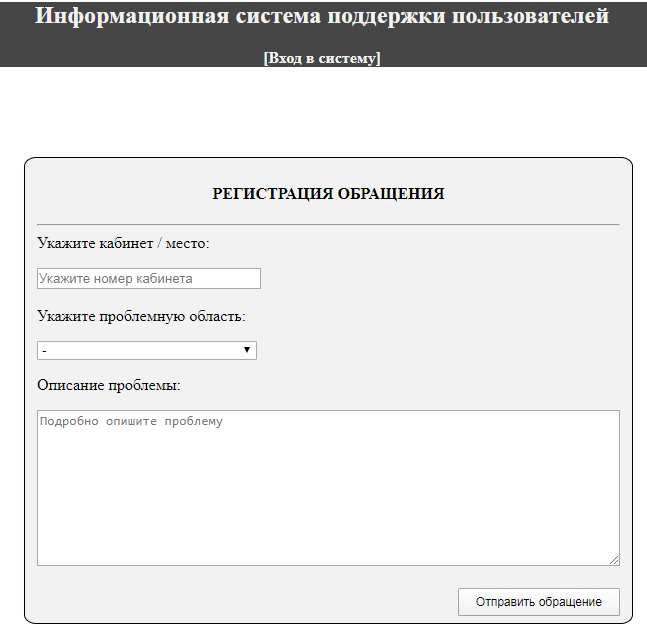


Рисунок 17 - Домашняя страница

Главной задачей в реализации программного продукта является возможность быстро формировать обращения. Поэтому на главной странице сразу размещена форма для регистрации обращения. Форма имеет простой, интуитивно понятный интерфейс и позволяет с быстро и легко формировать обращения. Для формирования обращения достаточно указать номер кабинета, проблемную область и дать описание проблемы, после чего нажать кнопку «Отправить обращение». Поскольку в программе появилась форма, которая отправляет POST-запрос, то на стороне сервера мы добавляем маршрутизатор, который «ловит» запрос и передаёт его в соответствующий роутер. Роутер в свою очередь определяет необходимый контроллер и перенаправляет туда запрос. Контроллер же занимается непосредственной обработкой запроса и сохранением запроса в базе данных. Для хранения запроса необходимо задаем модель для документов типа «supportRequest», что и продемонстрировано на рисунке 18.

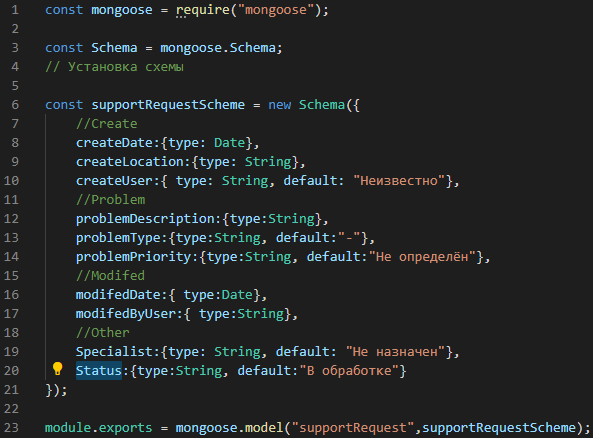


Рисунок 18 - Установка схемы запроса

На текущий момент мы имеем запрос, создаваемый пользователем, следующим шагом нужно добавить возможность обрабатывать эти запросы. Но простой возможности обработки запросов будет недостаточно, нам важно иметь возможность распределять запросы между сотрудниками. Для этого необходима возможность авторизации в системе.

На домашней странице добавляем возможность перехода на страницу авторизации (рисунок 19). Формируем страницу для авторизации пользователей.

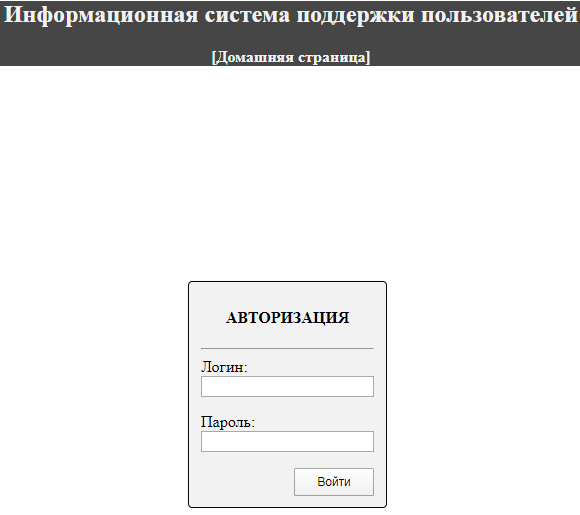


Рисунок 19 - Страница авторизации в системе

Для хранения информации о пользователях, аналогично хранению запросов создаём модель данных. Так как форма отправляет POST-запрос на сервер, то обрабатываем запрос на сервере. Учитывая то, что далее нам понадобится различать между собой пользователей – создаём сессии и присваиваем их во время авторизации.

После успешной авторизации, администратор попадает на основную рабочую страницу программы – workspace. На этой странице реализован функционал для просмотра списка заявок, просмотра подробной информации по каждой из заявок и её корректировке в зависимости от текущего состояния заявки. Для наглядности внешний вид основной рабочей страницы представлен на рисунке 20.

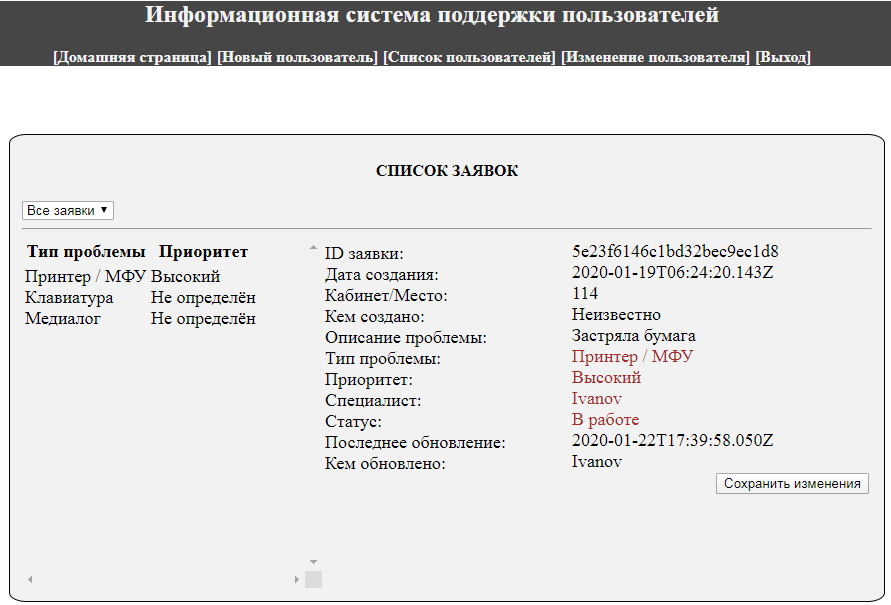


Рисунок 20 - Основная рабочая страница

Как видно из рисунка 20, страница состоит из навигационного блока (вверху) и формы «Список заявок». На форме «Список заявок» отображены все активные заявки. Заявки со статусом «Решено» на форме не отображаются. Так же при помощи фильтра можно выбрать режим просмотра заявок «Все заявки», «Новые» - отображаются заявки со статусом «Не определён» и «Мои» - заявки, в которых в графе «Специалист» указан текущий пользователь. Кликнув по заявке, на форме отображается более подробная информация о ней, там же и происходит управление заявками путём изменения значений полей «Тип проблемы», «Приоритет», «Специалист» и «Статус». После того как нужные изменения внесены, их нужно сохранить при помощи кнопки «Сохранить изменения».

Механизм работы этой страницы таков, что основные операции с данными происходят в «фоновом» режиме, благодаря технологии «Ajax». Изначально, страница не имеет никаких сведений о текущих заявках и производит их запрос после полной загрузки страницы. Это происходит так быстро, что почти незаметно при использовании. На основании списка запросов генерируется и другие элементы страницы, отвечающие за отображение полного описания запроса. Изменение сортировки и сохранение запроса, фактически использует те же самые механизмы, но с другими параметрами. В результате чего получается достигнуть максимально простой, интуитивно понятной и быстрой реализации поставленной задачи.

Помимо управления заявками, через основную страницу так же реализуются такие функции по управлению пользователями как «Создание нового пользователя», «Просмотр списка пользователей». Название этих функций говорит само за себя, на рисунках 21 и 23 соответственно продемонстрирован внешний вид этих страниц.

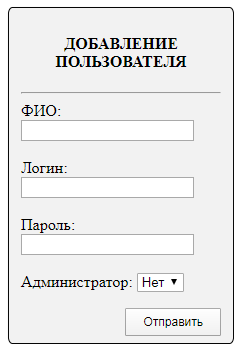


Рисунок 21 - Меню добавления нового пользователя

Благодаря меню «Добавление пользователя» возможно добавление новых пользователей в средствами самой системы. Форма имеет простой и понятный интерфейс. Важным является наличие администраторских прав, если учётная запись создаётся для сотрудника ИТ-отдела, в ином случае ряд основных функци пользователю будет недоступен. С внутренней стороны форма реализована аналогично форме «Авторизация» и использует ту же коллекцию документов – users. Посредством POST-запроса информация передаётся в маршрутизатор, затем в роутер и в конце концов в контроллер (Рисунок 22).

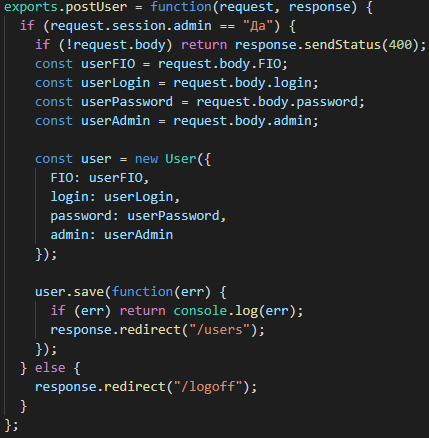


Рисунок 22 - Программная реализация функции добавления пользователей

Как уже было сказано ранее, для выполнения некоторых функций требуется важно наличие администраторской учётной записи, создание пользователей – одна из таких функций.

Вопрос создания неадминистраторских учетных записей остаётся открытым и требует дополнительного согласования с администрацией медицинского учреждения.

На данный момент, для создания заявки не требуется какая-либо авторизация, это сделано для упращения работы пользователей с системой. Пароль от входа в информационную систему может быть забыт или утерян и это не должно делать регистрацию обращений невозможной. Целесообразно добавления учётных записей для заведующих отделениями. Благодаря этому будет возможно вынести такой процесс как «Изменение расписания» в информационную систему. Сейчас для того чтобы внести изменения в расписания врачей, заведующий отделением должен оставить соответствующую заметку в специальном журнале, что не всегда удобно как для заведующих отделения, так и для администраторов. К сожалению, на данный момент введение такой функции не было одобренно руководящим составом поликлиники.

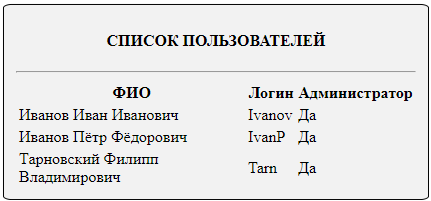


Рисунок 23 - Список пользователей

Отображение списка пользователей (Рисунок 24) имеет схожую структуру с добавлением пользователей, фактически различаются только контроллеры. Контроллер, реализующий обработку запроса списка пользователей представлен на рисунке 24.

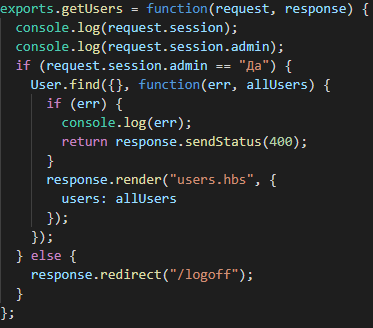


Рисунок 24 - Программная реализация функции «Список пользователей»

Полученная в ходе разработки информационная система имеет максимально простой интерфейс. Проект выполнен в виде web-приложения, что делает приложение кроссплатформенным. В приложении реализованы такие функции как «Регистрация обращений», «Авторизация», «Управление заявками», «Управление пользователями». Одним из главных преимуществ системы является возможность легко, быстро и без авторизации зарегистрировать сведения об инциденте. С другой стороны, предоставлен гибкий функционал для работы с заявками. Имеется возможность быстро и в удобном виде получать подробную информацию о заявках, обновлять статус заявок в несколько кликов. Управление пользователями реализовано достаточно просто. Информационная система соответствует всем предъявленным требованиями, а также имеет ряд возможностей для дальнейшей модернизации. Одним из примеров такой модернизации может служить создание личных кабинетов для заведующих отделений и реализация функции «Создание заявок об изменении расписания».

1. РАСЧЕТ ЭКОНОМИЧЕСКОЙ ЭФФЕКТИВНОСТИ
   1. Понятие экономической эффективности

Одним из важнейших критериев, который стоит учитывать при разработке информационной системы является её экономическая эффективность. Для того, чтобы сформировать понятие экономической эффективности, необходимо детально изучить понятие экономической эффективности и отражение этого показателя в других сферах деятельности.

На данный момент, проблема эффективности решается на всех уровнях экономики. Эта проблема затрагивается в самых различных уровнях экономики, начиная от общества в целом и заканчивая деятельностью отдельных фирм и их подразделений. Если рассмотреть эффективность деятельности фирмы, то она предполагает постоянное сопоставление результата и вызвавших его затрат, а также выбор наиболее эффективного варианта действий. От того насколько эффективно строится деятельность фирмы сегодня зависят её возможности завтра. На сегодняшний день в экономической литературе не существует однозначной трактовки понятия экономической эффективности, вследствие чего можно выделить три подхода к определению экономической эффективности: Бархатов В.И., Ковалев В.В., Плетнев Д.А., Хачатуров Т.С. и другие дают определение экономической эффективности как отношение результата, эффекта к затратам или ресурсам. /4,5/. Базылев Н.И. и Гурко С.П., Гукасьян Г.М., Шеремет А.Д. и другие понятие экономической эффективности рассматривают в рамках производства. Абалкин Л.И, Борисов А.Б. дают определение экономической эффективности через понятие результат в соответствии с предыдущими подходами. Проанализировав теоретические и методологические положения об экономической эффективности фирмы, необходимо отметить, что существующие подходы не позволяют объяснить поведение фирмы в процессе трансформирования российской экономики. Проблема заключается в том, что имеющиеся в западной литературе модели объясняют данные явления с позиции эволюции и развития западных рыночных экономик и отражают степень развития фирмы на определённом этапе истории. С другой стороны, советские учёные исследовали плановую экономику, где основным хозяйствующим субъектом было государство, а народное хозяйство рассматривалось как единое целое. На макроэкономическом уровне критерием эффективности считалось максимальное удовлетворение в общественном благе людей. Таким образом, проведенный анализ подходов к изучению эволюции взглядов на экономическую эффективность зарубежных и отечественных ученых позволяет уточнить содержание категории «экономическая эффективность». Таким образом, под экономической эффективностью будет корректным понятие отношения результата к затратам деятельности фирм, зависящее от создания оптимальных условий производства качественного продукта, его сбыта характеризующееся достижением высоких финансовых показателей.

Если рассматривать особенности экономической эффективности фирм, то можно выделить следующие факторы: Во-первых, в условиях низкого технического уровня оснащенности и высокого износа оборудования наблюдается снижение экономической эффективности фирм. Деятельность фирмы осуществляется тем эффективней, чем совершеннее на ней техническая оснащенность производства, под которой понимается комплекс конструкторских, технологических и организационных мероприятий, обеспечивающих разработку и освоение производства различных видов продукции, а также совершенствование выпускаемой продукции. Во-вторых, наличие четких конкурентных стратегий в изменяющихся условиях повышает уровень экономической эффективности фирм. Не малую роль имеет специфика продукции предприятий, находящихся на территории города или конкретного субъекта, отсутствие стабильной тенденции ее развития, трудности производства и реализации ставит перед предприятиями сложную проблему выбора не только конкурентной среды и непосредственных конкурентов на рынке, но и ее конкурентной стратегии, а также, разработанные стратегии на случаи различных изменений внешней среды. Осуществление выбора конкурентной стратегии для фирмы в первую очередь зависит от доли рынка, занимаемой фирмой, характеристик производства и продвижения товаров на рынке, наличия преимущественных особенностей продукции. Для многих фирм, особенно крупных, основой построения конкурентной стратегии является дифференциация производимой продукции. Те фирмы, дифференциация продукции которых практически не может быть осуществлена, используют ценовую конкуренцию, что возможно при снижении издержек производства. От того насколько правильно выбрана и конкурентная стратегия зависит эффективное функционирование фирм, находящихся на конкретной территории. В-третьих, зависимость от платежеспособного спроса населения снижает уровень экономической эффективности фирм. Неразвитая инфраструктура рынка негативно отражается на формировании справедливых цен на социально важные виды продукта и решении вопроса их физической доступности для населения. Низкие доходы большей части населения страны не позволяют потреблять продовольствие в достаточных количествах и разнообразного ассортимента. Таким образом, низкий платежеспособный спрос населения снижает и экономическую эффективность фирм, находящихся на этой территории. В-четвертых, привлечение инвестиций позволяет повысить уровень экономической эффективности фирм. Ключевым фактором является то, что для предприятий инвестиции выступают основным фактором, создающим условия для расширенного воспроизводства и роста потенциала конкурентоспособности. В-пятых, недостаточная реализация законодательных гарантий, прав собственности и контрактов приводит к снижению экономической эффективности фирм. Необходимо проводить мероприятия по управлению развитием промышленности на региональном уровне. В последнее время в регионах утверждаются специальные программы, на реализацию которых предоставляется финансовая поддержка из бюджетов субъектов Российской Федерации. Однако руководству многих субъектов не удается осуществлять эффективное управление промышленностью, а также другими сферами жизнедеятельности на своих территориях. Поставка некачественного сырья, товаров, количества, которое не соответствует заявленному в контракте, нарушение упаковки и маркировки товара, нарушение срока поставок, нарушение сроков платежа и другие факты могут послужить причиной снижения экономической эффективности фирм. Особенности экономической эффективности фирмы определяются тем, что фирма рассматривается как самостоятельная единица. Финансово-хозяйственная деятельность представляет собой эффективное использование имеющегося у фирмы экономического потенциала. Оценка целесообразности и эффективности управления экономической эффективности фирмы может быть дана в рамках комплексной оценки, а также, охарактеризована тремя взаимосвязанными блоками (ресурсы – производственно-технологический процесс – результат). Ресурсы следует изучать по трем направлениям, а именно, наличие и состояние ресурсов; привлечение и выбытие ресурсов; эффективность использования ресурсов. Основная цель управления экономической эффективностью фирмы это обеспечение эффективности расходов и целесообразности затрат в различных разрезах. Управление экономической эффективностью фирмы может быть оценено системой критериев. Аналитическое обоснование и оценка ресурсов, процесса и результата экономической эффективности фирмы могут быть выполнены в рамках внутреннего или внешнего анализа. Главное в управлении экономической эффективностью фирмы — это увязка и системность показателей. Результатом выступают показатели эффективности управления финансово – хозяйственной деятельностью фирмы с учетом действия всех факторов. При поведении управления экономической эффективностью фирмы следует учитывать общие и специфические особенности, связанные с функционированием фирмы в условиях финансового кризиса. Таким образом, представлена система показателей, посредством которых можно определить экономическую эффективность фирм, находящихся на территории города Шадринска и определить влияние на неё указанных факторов. Методика основана на оценке эффективности использования активов фирмы. Инвестиционная и финансовая деятельность фирмы рассматривается в условиях влияния факторов: технической оснащенности фондов, сроков окупаемости инвестиций, темпа роста продаж. В основу разработки методики оценки эффективности фирмы заложены следующие концептуальные моменты: Во-первых, экономическая эффективность фирмы зависит от уровня технической оснащенности. Во-вторых, приток инвестиций дает возможность увеличить экономическую эффективность фирмы. В-третьих, платежеспособный спрос населения влияет на эффективное функционирование фирмы.

Как уже говорилось в этой главе – экономическая эффективность информационной системы один из важнейших критериев, который учитывается при разработке программного продукта. Как любой бизнес-проект, программный продукт создается на базе понимания его эффективности с точки зрения востребованности и прибыльности. Интересным является то, что преимущества информационных технологий у руководящего состава предприятий чаще всего не вызывают сомнений. ИТ-решения часто окупаются и это признает большинство представителей топ-менеджмента компаний, однако, единой формулы подсчета эффективности информационных систем на настоящий момент не существует.

Как бы парадоксально это не звучало, но для большинства руководителей компаний возврат на инвестицию в информационные технологии не является главнейшим критерием для принятия решения о реализации проектов и это очень важный момент. Чаще всего оценка эффективности систем рассматривается с точки зрения повышения производительности труда. Однако, на практике это не единственный подход к расчету экономической эффективности. В зависимости от множества факторов, сложилось несколько различных методологических подходов к оценке эффективности от эксплуатации информационных систем.

Инвестиции в информационные технологии дают отдачу в виде роста рыночной капитализации компании за счет её большей управляемости, прозрачности, новых компетенций, производственной культуры, привлекательности для клиентов и сотрудников, уменьшения бизнес-рисков. В долгосрочной перспективе инвестиции в ИТ снижают дисконт на поток наличности от операционной деятельности компании, повышая её биржевую стоимость, а также снижают ставку банковского процента за счет уменьшения рискованности бизнеса.

Информационные технологии являются структурным элементом системы корпоративного управления, обеспечивая потоки внешней и внутренней информации для менеджмента компании, и всех лиц так или иначе заинтересованных в содержании управленческой информации компании. Информационные технологии являются основным источником такой информации и решают задачи по её формированию, сохранению и воспроизведению, обеспечивая конкурентоспособность, непрерывность и развитие бизнеса.

Инвестиции в ИТ являются основным инструментом для поддержания конкурентоспособности предприятия. Гарантия конкурентоспособности для предприятия – это прежде всего правильное применение информационных технологий в области формирования, поддержания и развития продуктовых линеек, цепочек поставок и отношений с клиентами в их динамике.

Благодаря инвестициям в информационные технологии в компании формируются такие следующие конкурентно способные качества:

* Сокращение сроков поставок продуктов заказчикам;
* Сокращение сроков ввода в производство новых продуктовых линеек;
* Гибкость в планировании производства продукции за счет автоматизации управления материальными потоками;
* Возможность управления себестоимостью продукции;
* Автоматизация отношений с клиентами (CRM).

Внедрение информационной системы способно успешно разрешить проблемные места на уровне функциональных подразделений в уже фактически сложившейся системе отношений. Каждое подразделение имеет свой собственный набор параметров эффективности работы системы. Так, например, функциональное подразделение технологической подготовки производства увеличивает производительность труда технологов, маркетинг получает контроль над исполнением заказов, снабжение получает операционное планирование закупок, ориентированное на материальное обеспечение производства.

Для оценки проектов по внедрению информационных технологий используется несколько подходов, наиболее популярными являются «Портфельный подход», «Бюджетный подход» и «Проектный подход». Рассмотрим каждый из подходов в отдельности, чтобы определить сильные и слабые стороны каждого.

Наиболее часто используемый подход оценки проектов по внедрению информационных технологий в компании – это так называемый портфельный подход. Его форма представляет собой простую таблицу правильно составленного ИТ-портфеля для предприятия. Такая таблица содержит исчерпывающий перечень бизнес-процессов компании с указанием всевозможных средств их автоматизации и оптимизации в сравнении. Портфельный подход применяется для оценки эффективности информационной технологии руководством компании, полагаясь на оценку, данную специалистами ИТ-подразделения. Оценка эффективности ИТ-портфеля осуществляется, как правило, с точки зрения производительности труда. Так же в таблице содержатся сведения о стоимости проектов по внедрению и поддержке ИТ-решений. Портфельный подход создан для руководителя предприятия, который в простой и доступной форме получает всю минимальную и достаточную информацию для выбора стратегического направления для развития ИТ на предприятии.

В отличии от портфельного подхода, бюджетный подход применяется на основе предпосылок о гарантированной эффективности ИТ при правильно построенных процедурах бюджетирования ИТ, мотивации персонала и контроля за расходованием средств. Данный подход применяется компаниями с уже сформировавшимся ИТ-хозяйством, когда большая часть ИТ-бюджета уходит не на внедрение новых ИТ-решений, а на поддержание уже внедренных ИТ (более 70% от бюджета). Как правило, компании определяют долю в процентах от, например, дохода компании которая уходит на инвестиции в ИТ. При этом ключевым параметром в обосновании для формирования такого бюджета является рост производительности труда.

Стоит заметить, что на предприятиях, не освоивших ИТ-системы, такой подход не применим, так как бюджетировать расходы на ИТ в непроизводительный труд не имеет никакого смысла, сначала необходимо изменить суть бизнес-процессов, привести предприятие в соответствие с современными требованиями к ИТ-оснащенности. Инвестиции в ИТ распределяются по функциональным подразделениям, которые при должной мотивации формируют обоснование применения, соответствующего ИТ-решения в привязке к росту производительности труда. Часто ИТ-бюджет осваивается функциональными подразделениями по принципу внутреннего подряда к ИТ-подразделению. Каждое из подразделений оценивает, какие решения в области ИТ являются обоснованными и необходимыми и, используя свой бюджет на ИТ, «заказывает» разработку у ИТ-подразделения. Таким образом, при внедрении ИТ-решений достигается эффективное участие в ИТ-проекте и персонала со стороны функционального подразделения, и работников ИТ-подразделения. В свою очередь, ИТ-подразделение, осваивая бюджеты от внутреннего подряда, привлекает внешних субподрядчиков для закупки/интеграции ИТ-решений.

Основываясь на таком подходе, многие крупные консорциумы в последние годы практикуют заключение сделок на аутсорсинг ИТ-подразделения. ИТ-бюджеты таких крупных консорциумов, как J.P. Morgan Chase или Bank of America составляют несколько миллиардов долларов, поэтому аутсорсинг или ауттаскинг (вынесение задач ИТ за пределы компании) в таких крупных компаниях имеет самое практическое значение. С другой стороны, например, в Российской Федерации ауттаскинг имеет самое широкое применение среди небольших компаний. Причина проста – вынесение ИТ-бюджета за пределы компании позволяет даже самым небольшим компаниям конкурировать с гигантами отрасли, сосредоточившись на основных функциях, не занимаясь поддержкой (развитием) информационных систем. Для белорусских компаний состояние зрелости в ИТ за редким исключением пока в далекой перспективе, хотя некоторые функции информационных систем, например, поддержка сети и парка компьютеров, уже бюджетируются, исходя из принципов, изложенных выше.

Современная финансовая теория признает четыре основных способа расчета эффективности проекта и его ценности для компании: срок окупаемости, возврат на инвестиции, внутренняя рентабельность и чистая прибыль от проекта с учетом стоимости капитала, приведенная к сегодняшнему дню.

Проблема заключается в том, что расчет NVP или внутренней рентабельности требует учета многих параметров (стоимость капитала, свободные потоки наличности, эффект от налогов, остаточная стоимость и т.п.), которые при отсутствии уже освоенной на предприятии информационной системы получить сложно (а зачастую и невозможно). В связи с этим наиболее распространенной методологией оценки информационных систем является ROI с точки зрения наглядности и простоты для руководителей компании, и инвесторов. ROI, как правило, рассчитывается по функциональным подразделениям, включенным в проект внедрения информационной системы. Недостаток данной методологии заключается в том, что в рамках горизонта функционального подразделения очень сложно количественно оценить качественное изменение в сути бизнес-процессов или же важное качественное изменение может быть просто не замечено. В связи с этим такая оценка зачастую бывает неточной и может быть проигнорирована, если проводится самостоятельно функциональными службами без участия специалистов финансового подразделения.

Оценка ROI, проведенная в совокупности с оценкой рисков внедрения информационной системы в компании, выдает показатели вероятности того или иного значения ROI (например, 85% вероятности успеха на 50% ROI, или 30% вероятности успеха на 70% ROI).

Для простоты расчета ROI имеет смысл разделить эффекты от внедрения информационной системы на три вида:

Расчетный эффект – рассчитывается все до копейки (снижение незавершенного производства при внедрении ERP-системы на миллион долларов, за счет этого экономия банковского процента на сто восемьдесят тысяч, экономия бумаги на производство справочников службы снабжения или сбыта на десять тысяч долларов в год). Как правило, такой расчет наглядно демонстрирует финансовым руководителям рост производительности капитала.

Эффект времени и производительности труда за счет более быстрого исполнения сотрудниками своих функций (например, на 15 минут в день для формирования отчетов о производстве основы для начальников смен, 8 часов в месяц для начальников складов и бухгалтеров для инвентаризации). В конце расчета этот эффект трансформируется в тысячи трудодней, обладающих объективной и внушительной стоимостью.

«Тонкие» эффекты – рассчитываются, исходя из специфики каждой компании. Например, можно рассчитать эффект от внедрения ERP-системы на производстве для получения управленческой информации, которая позволит принять стратегические решения в отношении более эффективного использования производственных мощностей, или замены неэффективных рабочих мест на новые, более эффективные.

Как правило, основной эффект от внедрения информационных систем – это рост производительности труда:

* Экономия рабочего времени определенного рода менеджеров;
* Эффективное применение человеческих ресурсов на предприятии;
* Сокращение стоимости осуществления той или иной трансакции на предприятии.

Как любой бизнес-проект, программный продукт создается на базе понимания его эффективности с точки зрения востребованности и прибыльности. И экономическая эффективность проекта - один из основных показателей, отражающий целесообразность внедрения информационной системы. Как бы парадоксально это не звучало, но для большинства руководителей компаний прибыльность самой информационной системы не является главнейшим критерием для принятия решения о её реализации. На сегодняшний день существует несколько различных методов оценки эффективности информационной системы. При этом, чаще всего оценка эффективности систем производится не с точки зрения получения «прямой» прибыли, а с точки зрения развития таких конкурентоспособных качеств предприятия, как повышение производительности труда, сокращение издержек, улучшения качества конечного продукта и увеличения гибкости в производстве.

* 1. Методика расчета экономической эффективности.

Вопрос обоснования расходов в сфере ИТ часто становится очень острым. Зачастую в организациях, напрямую не связанных с ИТ сферах, выделяемый бюджет не всегда покрывает даже базовые потребности организации в ИТ обеспечении. Поэтому, для реализации проекта необходимо наличие твёрдого обоснования.

Для полноценной, качественной оценки результата следует сделать упор на то, ради чего осуществляется внедрение ИТ-проекта. Такое целеполагание должно быть выполнено сверху донизу и органичным образом интегрировано в процесс проектирования ИТ-системы.

Для того чтобы правильно выбрать метод расчёта экономической эффективности информационной системы, необходимо рассмотреть существующие методы более детально.

Метод оценки совокупной стоимости владения информационной системой иликонцепция общей стоимости владения (TCO) ИТ была разработана компанией Gartner Group ещё в конце 80-х годов. Данная концепция позволяет оценивать совокупные затраты на ИТ, анализировать их и, соответственно, управлять ими для достижения наилучшей отдачи. В данном случае общая стоимость владения информационной системой является важнейшим критерием при рассмотрении будущих проектов, так как является ключевым фактором в определении их экономической эффективности и обоснованности. Помимо выявления избыточных статей расхода, основной целью расчёта данного показателя является оценка возможности возврата вложенных в проект средств. /28/ Так же ключевым моментом в этом случае является то, что ТСО своего предприятия сравнивается с ТСО других компаний аналогичного профиля. При оценке экономического эффекта зачастую возникают проблемы при оценке прямого экономического эффекта от информационной технологии. Сравнение показателей ТСО позволяет отследить являются ли экономические показатели проект средними по отрасли или даже лучше их. Зачастую подобное сравнение проводится со средними по отрасли аналогичными компаниями и с «лучшими в группе». Даже если прямой экономический эффект от внедрения ИТ определен, его всегда надо сравнить с затратной частью, то есть с ТСО.

Основу модели ТСО составляют две категории затрат: прямые и косвенные. Категория прямых затрат напрямую связана со следующими отделами:

* ИТ-отдел компании, ответственный за развитие и поддержку ИС, корпоративной сети;
* Группы по поддержке и развитию ИТ, имеющиеся внутри производственных и административных подразделений компании;
* Отдельные группы специалистов, обеспечивающих специализированные виды услуг, например, услуг связи и передачи данных.

Прямые расходы включают в себя:

* Капитальные затраты - аппаратное и программное обеспечение (АО и ПО);
* Расходы на управление ИТ;
* Расходы на техническую поддержку АО и ПО;
* Расходы на разработку прикладного ПО внутренними силами;
* Расходы на аутсорсинг;
* Командировочные расходы;
* Расходы на услуги связи;
* Другие группы расходов.

По этим группам прямых расходов определяют составляющие ТСО. Например, при определении капитальных затрат на оборудование расходы должны включать:

* Расходы на приобретение нового оборудования и его замену;
* Средства, вырученные от продажи или передачи оборудования;
* Амортизацию оборудования;
* Затраты на сетевое оборудование и соединения (кабели, концентраторы, карты, которые, как правило, не амортизируются);
* Расходы на приобретение периферийных устройств;
* Расходы на приобретение дополнительной оперативной памяти (при этом следует учитывать амортизацию оборудования);
* Расходы на дополнительные дисковые устройства (учитывается амортизация оборудования);
* Расходы на замену оборудования;
* Прочие расходы по оборудованию.

Самой простой группой для расчетов ТСО является расходы по оборудованию. Аналогичным образом рассматриваются и другие группы прямых расходов, такие как программное обеспечение, техническая поддержка, управление. Как правило перечень составляет до десятка различных групп, при этом каждая группа имеет определённую специфику расчетов. Наиболее трудоёмкой для расчётов группой можно считать расходы на управление. В эту статью входят расходы на проектирование, управление проектами, администрирование сетей, работа с чрезвычайными ситуациями, настройка систем, управление контрактами на закупку и управление поставками.

В расчёты включаются и косвенные расходы. Всего выделяют две группы по источникам их возникновения. Первая группа достаточно скрытая и отражается в неправильном проектировании ИС. При таком раскладе нарушается стабильность работы системы, что в свою очередь вызывает непроизводительное расходование времени у пользователей и даже потери в бизнесе компании. Как правило, косвенные расходы трудно определить напрямую. Однако их следует учитывать при проектировании ИС и организации технической поддержки. Важно различать плановое и сверхнормативное время неработоспособности. Ко второй группе косвенных расходов относится недостаточная поддержка со стороны штатных сотрудников ИТ-отдела, что приводит к необходимости самостоятельного восстановления работоспособности системы конечными пользователями, что значительно снижает продуктивность их труда. Косвенные расходы находятся за рамками бюджетов на ИТ, однако они могут играть существенную роль в оценке решения по проектам. При этом первая их группа ("неработоспособность системы") может быть рассмотрена с использованием метода определения производственных потерь. Вторая группа ("непроизводительные усилия конечного пользователя"), связанная с информационными технологиями, определяется с помощью полевых и статистических исследований. /15/

Показатель совокупной стоимости владения ИС рассчитывается по формуле:

TCO = DE + IC1 + IC2, (1)

где DE (direct expenses) – прямые расходы

IC1,2 (indirect costs) – косвенные расходы первой и второй группы соответственно.

При этом:

DE = DE1 + DE2 + DE3 + DE4 + DE5 + DE6 + DE7 + D8, (2)

где DE1 - капитальные затраты;

DE2 - расходы на управление ИТ;

DE3 - расходы на техническую поддержку АО и ПО;

DE4 - расходы на разработку прикладного ПО внутренними силами;

DE5 - расходы на аутсорсинг;

DE6 - командировочные расходы;

DE7 - расходы на услуги связи;

DE8 - другие группы расходов.

ТСО необходимо не только рассчитывать при рассмотрении нового проекта, но и постоянно отслеживать в дальнейшем. /24/

Общая стоимость владения информационными технологиями - это качественная ключевая характеристика, отображающая экономические аспекты состояния ИТ в компании и показывающая эффективность их работы.

Вложения информационные технологии могут рассматриваться не с точки зрения затрат, а как инвестиции в основной бизнес. Таким образом, для оценки экономической эффективности используются те же инструменты и процедуры, что и в любом инвестиционном проекте.

Как правило, внедрение средств вычислительной техники для обработки информации направлено на повышение эффективности производства через совершенствование процессов управления, что проявляется в показателях производственно-хозяйственной деятельности объекта управления. Реализация мероприятия по использованию средств вычислительной техники, средств связи и оргтехники требует занчительных финансовых затрат, то и расходы по внедрению проектов обслуживания ИТ инфраструктуры должны окупаться в короткие сроки. При создании АИС, предприятие несёт единовременные затраты на её разработку и приобретение необходимых технических и программных средств. Текущие затраты на функционирование системы, её подготовку и обучение сотрудников организации. Экономия от функционирования программного продукта определяется от с учетом затрат на её эксплуатацию. /14/

Стандартная методика расчета показателей экономической эффективности включает в себя расчет суммы годовой экономии, коэффициента экономической эффективности капитальных вложений и срока окупаемости капитальных вложений.

Сумма годовой экономии от сокращения ручного труда по обработке информации рассчитывается по формуле:

S = OC1 – OC2 (3)

где S – сумма годовой экономии от сокращения ручного труда по обработке информации, руб.;

OC1 – годовые эксплуатационные затраты при ручной обработке информации, руб.

 (4)

где Z1i – месячная основная заработная плата i-го работника, руб.;

T1i – месячные трудовые затраты i-го работника на решение задачи, человеко-дни;

Q – среднее количество рабочих дней в месяц, дни;

α – коэффициент накладных расходов;

β – коэффициент дополнительной заработной платы (отчисления на социальное страхование, в различные фонды и т.п.).

OC2 = С1+С2+С3 (5)

С1 – годовые затраты машинного времени на решение задачи, руб.;

С2 – годовые затраты на заполнение документов, анализ и корректировку данных (ручные операции), руб.;

С3 – годовые затраты на обучение персонала, адаптацию и настройку оборудования, руб.

 (6)

где Sq – себестоимость часа работы оборудования q при решении задачи, руб.;

Tq – время работы оборудования q при решении задачи в течение месяца, машино-часы;

C2 рассчитывается по аналогичной формуле.

С3 = К3 \* γ (7)

где К3 – годовые единовременные затраты на обучение персонала, адаптацию, настройку оборудования при решении задачи, руб.

γ - коэффициент настройки оборудования.

К3 = К31+К32+К33 (8)

К31 – годовые единовременные затраты по заработной плате персонала на обучение, адаптацию и настройку оборудования для решения задачи, руб.

 (9)

где Z2i – месячная основная заработная плата работника i, руб.;

T2i – месячные трудовые затраты работника i на обучение, настройку оборудования и т.п., человеко-дни;

К32 – годовые единовременные затраты машинного времени

** (10)

где T2q – время работы оборудования q на обучение персонала, адаптацию и настройку оборудования, машино-часы.

К33 – прочие единовременные расходы, руб.:

 (11)

где h – коэффициент прочих расходов, к прочим расходам относятся: расходы на приобретение машинных носителей, бумаги, краски и т.п.

К – единовременные затраты на решение задачи, руб.

К = К1+К2+К3 (12)

где К1 – единовременные затраты на проектирование, руб.:

K1 =  (13)

где T2i – месячные трудовые затраты специалиста на проектирование решения задачи, человеко-часы.

n – длительность проектирования.

К2 – единовременные затраты, связанные с использованием различных видов оборудования, руб.

K2 =  (14)

где BV – балансовая стоимость комплекта техники или ПЭВМ, руб.;

t – длительность эксплуатации ПЭВМ до начала решения задачи, годы;

r – годовая норма на реновацию оборудования (около 10%);

T - время работы оборудования при решении задачи в течении месяца, машино-часы;

α – коэффициент, определяющий стоимость вспомогательного оборудования;

F – планируемый годовой фонд времени работы ПЭВМ (оборудования);

F=tc\*Tc (15)

tc – среднесуточная фактическая загрузка ПЭВМ (оборудования), часы;

Tc – среднее количество дней работы ПЭВМ (оборудования) в году.

Коэффициент экономической эффективности Er рассчитывается по формуле:

Er = S/К (16)

Если Er ≥ Еnce, то технология является эффективной.

где Еnce – нормативный коэффициент эффективности капитальных вложений для вычислительной техники, его значение определяет нижнюю границу годовой экономии, которую можно получить на один рубль капитальных затрат.

Т - срок окупаемости затрат на решение задачи.

Метод чистой приведенной стоимости (NPV). Этот метод основан на сопоставлении величины исходных инвестиций (IC) с общей суммой дисконтированных чистых денежных поступлений, генерируемых в течение прогнозируемого срока. Поскольку приток денежных средств распределен во времени, он дисконтируется с помощью коэффициента r, устанавливаемого аналитиком (инвестором) самостоятельно исходя из ежегодного процента возврата, который он хочет или может иметь на инвестируемый им капитал.

Допустим, делается прогноз, что инвестиции (IC) будут генерировать в течение n лет, годовые доходы в размере P1, P2, ..., Рn. Общая накопленная величина дисконтированных доходов (PV) и чистый приведенный эффект (NPV) соответственно рассчитываются по формулам:

, (17)

. (18)

Очевидно, что если

NPV > 0, то проект принесет прибыль сверх ожидаемой;

NPV < 0, то по проекту ожидается «убыток»;

NPV = 0, то проект соответствует ожиданиям инвестора.

При прогнозировании доходов по годам необходимо по возможности учитывать все виды поступлений как производственного, так и непроизводственного характера, которые могут быть ассоциированы с данным проектом. Так, если по окончании периода реализации проекта планируется поступление средств в виде ликвидационной стоимости оборудования или высвобождения части оборотных средств, они должны быть учтены как доходы соответствующих периодов.

Если проект предполагает не разовую инвестицию, а последовательное инвестирование финансовых ресурсов в течение m лет, то формула для расчета NPV модифицируется следующим образом:

, (19)

где i — прогнозируемый средний уровень инфляции.

Необходимо отметить, что показатель NPV отражает прогнозную оценку изменения экономического потенциала предприятия в случае принятия рассматриваемого проекта. Этот показатель аддитивен во временном аспекте, т. е. NPV различных проектов можно суммировать. Это очень важное свойство, выделяющее этот критерий из всех остальных и позволяющее использовать его в качестве основного при анализе оптимальности инвестиционного портфеля. При помощи NPV-метода можно определить не только коммерческую эффективность проекта, но и рассчитать ряд дополнительных показателей.

Однако корректное использование NPV-метода возможно только при соблюдении ряда условий:

Объем денежных потоков в рамках инвестиционного проекта должен быть оценен для всего планового периода и привязан к определенным временным интервалам. Денежные потоки в рамках инвестиционного проекта должны рассматриваться изолированно от остальной производственной деятельности предприятия, т.е. характеризовать только платежи и поступления, непосредственно связанные с реализацией данного проекта. Принцип дисконтирования, применяемый при расчете чистого приведенного дохода, с экономической точки зрения подразумевает возможность неограниченного привлечения и вложения финансовых средств по ставке дисконта. Использование метода для сравнения эффективности нескольких проектов предполагает использование единой для всех проектов ставки дисконта и единого временного интервала (определяемого, как правило, как наибольший срок реализации из имеющихся).

При расчете NPV, как правило, используется постоянная ставка дисконтирования, однако в зависимости от обстоятельств (например, ожидается изменение уровня процентных ставок) ставка дисконтирования может дифференцироваться по годам. Если в ходе расчетов применяются различные ставки дисконтирования, то, во-первых, формулы (19) и (20) неприменимы и, во-вторых, проект, приемлемый при постоянной ставке дисконтирования, может стать неприемлемым.

Классическим методом для оценки эффективности проекта предполагается сравнение «доходной» и «затратной». Расчёт затрат обычно не составляет большого труда, а вот расчёт результатов остается сложной кардинально не решенной проблемой, особенно это касается материального количественного учета социального эффекта. Основная сложность заключается в оценке эффектов от реализации ИТ-проекта, т. е. оценки "доходной" части. Часто прибыль определяется путем экспертной оценки и по аналогии с другими подобными системами, а социальный эффект количественно вовсе не поддается оценке. /6/

Выбор методологии расчета экономической эффективности является ключевым моментом в расчёте экономической эффективности системы и зачастую зависит от цели внедрения ИТ-проекта. Учитывая тот факт, что целью внедрения информационной системы поддержки пользователей является сокращение потерь от простоев пользователей, то для отражения экономической эффективности системы потребуется расчет суммы капитальных вложений, годовой экономии и срока окупаемости капитальных вложений. Этим условиям полностью удовлетворяет стандартная методика расчета показателей экономической эффективности, что делает оправданным её использование для расчёта экономической эффективности проекта.

* 1. Расчет экономической эффективности внедрения информационной системы.

Для расчётов используем стандартную методику расчета показателей экономической эффективности. Эта методика включает в себя расчет суммы годовой экономии и срока окупаемости капитальных вложений. Так как информационная система поддержки пользователей не является основным инструментом в работе медперсонала и основной целью ставит уменьшение издержек от переработок, вызванных внеплановыми сбоями в работе программных и технических средств, используемых в КГБУЗ «Городская клиническая поликлиника №3». Для расчёта эффективности информационной системы сначала определим «затратную часть». В неё входят все необходимые затраты на оборудование разработку информационной системы.

Таблица 1 – Общая стоимость оборудования

|  |  |  |  |
| --- | --- | --- | --- |
| Состав оборудования | Количество, шт. | первоначальная стоимость, руб. | Общая стоимость, руб. |
|
|
| Офисный ПК | 1 | 25000 | 25000 |
| Итого: | | | 25000 |

Так как информационная система поддержки пользователей – клиент-серверное приложение, то для его работы необходимо наличие сервера, который мог бы принимать запросы от клиентов в режиме 24/7. Ввиду малых объемов выполняемой работы, на роль сервера подойдёт любой офисный компьютер с под управлением операционной системы Windows 10 Pro.

Расчет затрат на электроэнергию осуществляется по формуле 20

Зэл=P\*T\*Z (20)

где P – общая мощность оборудования,

T – общее время работы оборудования,

Z – цена одного киловатта в час (4,55 руб).

Данные для расчета общей мощности оборудования предоставлены в таблице 2. Источником данных являются средние показатели мощности ЭВМ и комплектующих.

|  |  |  |  |
| --- | --- | --- | --- |
| Состав оборудования | Количество, шт. | Мощность, КВт/ч | Общая мощность, КВт/ч |
|
|
| Офисный ПК | 1 | 0,4 | 0,4 |
| Итого: | | | 0,4 |

Таблица 2 – Общая мощность оборудования

1

Месячный полезный фонд времени работы ЭВМ рассчитывается по формуле 21.

(21)

где Треж – режимный фонд времени, час;

Ррем – процент плановых потерь на профилактику и ремонт (10%).

Произведём расчёт затрат электричества на поддержание работы сервера в течение 1 года. Так как работоспособность сервера необходимо поддерживать в режиме 24/7, то режимный фонд времени оборудования Треж = 24\*365 = 8760 часов.

Используя формулу 21 вычислим общее время работы оборудования:

Зная общее время работы оборудования по формуле 20 вычислим затраты на электричество за 1 год.

Зэл = 0,4\*7784\*4,55 = 14166,18 рублей

Произведём расчёт стоимости разработки и внедрения информационной системы. Сначала рассчитаем заработную плату руководителя.

Заработная плата руководителя выпускной квалификационной работы рассчитывается по следующей формуле 22.

(22)

где t – время на консультацию по ВКР;

Тч – тарифная ставка руководителя за час.

ЗПрук = 10 \* 200 = 2000 руб.

Для установки и настройки оборудования потребуется привлечение системного администратора. Рассчитаем заработную плату системного администратора по формуле 23.

(23)

где t – время на установку сервера и настройку ПО;

Тч – тарифная ставка системного администратора в час.

ЗПса = 5 \* 180 = 900 руб.

Итого, за первый год затраты на оборудование, электроэнергию, разработку и внедрение составят 42016, 18 руб.

Главным эффектом от внедрения информационной системы поддержки пользователей является сокращение затрат потерей от простоев. Потери от простоев пользователей — это потери для организации, связанные с простоем пользователя вследствие перерыва того или иного сервиса ИТ. Простой может быть плановым, то есть связанным с регламентными работами по обновлению оборудования и версии ПО, переносу данных и т.д. Также простой может быть внеплановым в связи с каким-либо инцидентом, то есть внеплановым нарушением сервиса. Простой обычно измеряется в единицах рабочего времени, потерянного пользователями.

В среднем за один рабочий день в отдел информатизации поступает 23 новые задачи, так или иначе связанных с поддержанием работоспособности ИТ-структуры организации. Девять из которых требуют немедленного разрешения, так как «останавливают» приём пациентов.

Рисунок 25 – Процентное соотношение обычных и срочных заявок

Конечно, разные по сложности проблемы требуют разных трудозатрат и разную длительность, решение некоторых из них может достигать нескольких часов. Точно оценить среднее время простоя очень сложная задача. Для начала определим составляющие обычного инцидента: самоподдержка, подготовка специалиста к устранению инцидента и сами технические работы. Самоподдержка означает, что пользователь самостоятельно разрешает инциденты с информационными системами на своем рабочем месте, не прибегая к помощи службы ИС. В данном случае мы не учитываем случаи, когда пользователи удачно устраняли инцидент, так как данные о таких случаях не передаются в ИТ отдел. На рисунке 26 рассмотрен инцидент с замятием бумаги во время печати врачебного заключения.

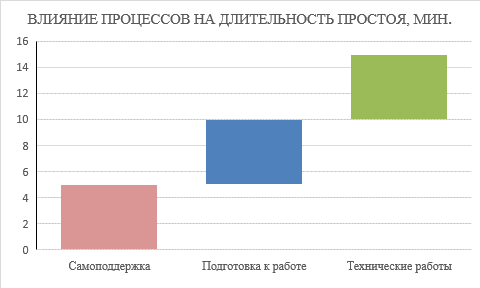


Рисунок 26 – Влияние процессов на длительность простоя

В этот самый момент начинается время простоя, с большой вероятностью пользователь попытается извлечь замятую бумагу, перезапустит принтер и попытается распечатать заключение повторно. Так как пользователь устранил следствие, а не причину, то и второй лист будет замят и только после этого в ИТ отдел поступит заявка о неисправности. Самоподдержка в этом случае добавит 5 минут к простою, ещё 5 минут потребуется специалисту на сбор инструментов, запчастей и прибытие к месту поломки и 5 минут дополнительно на устранение самой неисправности. Поэтому для расчётов установим среднее время простоя равным 15 минутам, этот показатель несомненно меньше фактического, но в случае его превышения врач может продолжить свою работу на другом компьютере до устранения проблемы. Эта информация относится к тому моменту, когда пользователь может получить поддержку сразу.

Ещё одной проблемой является то, что не всё время работы поликлиники покрывается поддержкой со стороны ИТ отдела (Рисунок 27). поликлиника работает с понедельника по пятницу по 12 часов в день (с 08:00 до 20:00) и ещё 6 часов в субботу (с 09:00 до 14:00). Не сложно посчитать, что за полную рабочую неделю в течение 66 часов осуществляет приём посетителей, из них 45 часов в неделю (9 часов в день, 5 дней в неделю) отдел информатизации обеспечивает поддержку пользователей.

Рисунок 27 – Доступность ИТ-поддержки

Как было описано выше, в среднем за рабочий день в 9 часов работы ИТ отдела обрабатывается 9 срочных заявок, из чего следует что в среднем каждый час происходит 1 инцидент. Таким образом за рабочую неделю поликлиники (66 часов) случается примерно 66 инцидентов, вызывающих потери от простоя. Из них оперативно решаются только 45 инцидентов, которые происходят в период работы ИТ-отдела (45 часов). Остальные инциденты (21), которые происходят в внерабочее для ИТ-отдела наиболее существенны, так как вызывают два простоя - в вечернее время, так как тратят время врача, работающего в вечернее время на смену рабочего места (15 минут) и врача, работающего в утреннюю смену на сообщение об инциденте и его устранение (15 минут). Для наглядности эта информация графически отображена на рисунке 28.

Рисунок 28 – Общее количество простоев, вызванных инцидентами

Внедрение информационной системы поддержки позволит сотрудникам ИТ-отдела узнавать о проблемах и устранять их последствия ещё до начала приёма врача, что позволит избежать до 21 случая простоя пользователей в неделю или до 1092 таких случаев в год.

Как было описано ранее, при длительности одного простоя в 15 минут, 1092 случаев простоя приводят к 16380 минутам или 273 часам простоя за один год. Если взять среднюю заработную плату медицинского работника равной 33000 рублей, то при продолжительности рабочего дня в 7,6 часов и 22 рабочим дням в месяц, что равно 167,2 часам в месяц, то средняя заработная плата медицинского работника составит 179,5 рублей в час. Не сложно подсчитать, что 273 часа простоя при такой заработной плате приводят к потерям, эквивалентным 49003,5 рублей ежегодно.

Таблица 3 - Экономический эффект от внедрения ИС

|  |  |  |  |
| --- | --- | --- | --- |
| Длительность эксплуатации ИС | 1-й год | 2-й год | 3-й год |
| Общая экономия от использования ИС, руб. | 49003,5 | 98007 | 147010,5 |
| Общие затраты на ИС, руб. | 42016,18 | 56182,36 | 70348,54 |
| Экономический эффект, руб. | 6987,32 | 41824,64 | 76661,96 |

Как было рассчитано ранее, общие затраты на внедрение и эксплуатацию ИС в первый год составят 42016,18. Каждый последующий год эксплуатации информационной системы потребует дополнительных затрат в размере 14166,18 рублей. Более наглядно эта информация представлена на рисунке 29

Рисунок 29 - Сравнение затрат и экономии от использования ИС за три года

Экономия от использования информационной системы составит 49003,5 рублей в первый год и столько же за каждый последующий год или 4083,62 руб. ежемесячно. При таком темпе уже на 11 месяц экономия от использования информационной системы составит 44919,87 руб., что позволит окупить начальные вложения.

Это позволит сэкономить на потерях от простоев пользователей 6987,32 рублей в первый год, 41824,64 рублей – за два года и 76661,91 рубль за три года эксплуатации информационной системы поддержки пользователей.

Рисунок 30 - Общий экономический эффект от эксплуатации ИС за 3 года

По результатам расчётов становится видно, что несмотря на большие затраты в первый год в размере 42016,18 рублей и дополнительные затраты 14166,18 рублей за каждый последующий год эксплуатации информационной системы, экономия от использования системы поддержки пользователей за счёт сокращения потерь за каждый год эксплуатации составит до 49003,5 рублей. Первоначальные вложения окупятся уже на 11 месяц, а использование информационной системы позволит сэкономить за три года сумму в размере 76661,91 рублей, что подтверждает эффективность разработанного продукта.

ЗАКЛЮЧЕНИЕ

В результате изучения сущности и экономической значимости информационных систем на современном этапе, можно сделать вывод, что сущность информационных систем заключается в наличии чёткой структуры, данных и базы данных для её хранения, а также персонала, взаимодействующего с информационной системой для решения поставленных задач. Экономическая значимость информационных систем заключается в возможности эффективной работы с одним из важнейших ресурсов нашего времени – информацией. Эффективное использование информационных систем невозможно без технического сопровождения.

Техническое сопровождение и обеспечение бесперебойности работы информационных систем осуществляется техническими специалистами. В зависимости от объемов, обслуживанием может заниматься как один специалист, так и целый отдел. В некоторых случаях для обслуживания информационных систем могут использоваться целые компании. Для увеличения эффективности работы таких служб используются информационные системы. Современным подходом к технической поддержке является метод трехлинейной поддержки пользователей, однако чаще всего использование такой информационной системы в небольших отделах нецелесообразно. Внутренняя структура информационной системы должна быть согласована с используемой моделью поддержки пользователей, иначе это может негативно отразиться на работе отдела.

К бизнес-процессам поддержки пользователей можно отнести все процессы по управлению инцидентами, начиная от возникновения инцидента и заканчивая формированием отчётной документации. Рассмотренный в этой работе пример внедрения подобной системы позволит автоматизировать процессы обработки обращения и формирования заявки, это позволит регистрировать обращения в нерабочее время отдела, что позволит избежать части простоев пользователей. Исходя из этого можно сделать вывод, что эффективность использования информационной системы поддержки пользователей заключается в сокращении затрат, вызванных сбоями в работоспособности автоматизированных рабочих мест и увеличении стабильности ИТ-инфраструктуры.

Основное бизнес-требование к системе заключается в снижении времени простоя пользователей, что является прямым следствием снижения времени реагирования на инцидент. Для выполнения этого требования информационная система должна соответствовать таким требованиям пользователей как наличие пользовательского интерфейса, простота и удобность в использовании. Все действия в информационной системе должны быть интуитивно понятны простому пользователю. Это необходимо, чтобы использование информационной системы не приводило к ещё большим простоям. Для обеспечения круглосуточной регистрации обращений, необходимо наличие простой формы для быстрой отправки обращений. Для специалистов отдела должна присутствовать возможность просмотра списка активных заявок с возможностью просмотра всех, новых и только своих заявок. Необходимой также является функция просмотра подробных сведений о выбранной заявке и возможность изменения её статуса, исполнителя, приоритета и типа заявки. Так как система подразумевает работу нескольких администраторов одновременно, то необходимо наличие авторизации для администраторов. Система должна позволять управлять пользователями посредством внутренних инструментов. Ещё одним важным требованием к системе является кроссплатформенность. Кроссплатформенность необходима для возможности доступа к ИС с любого рабочего места в организации.

Равно как и проектирование, выбор средств реализации информационной системы имеет большое влияние в процессе разработки. От правильного выбора средств реализации зависит качество конечного продукта. Одним из требований к информационной системе является кроссплатформенность, поэтому было решено разработать ИС в виде Web-приложения. Такой подход к разработке позволит получать доступ к ИС любой платформе, поддерживающей браузер. В качестве языка разметки в работе используется HTML, а для удобства управления внешним видом страниц – CSS. Приложение написано на языке программирования JavaScript, так как он содержит ряд возможностей, направленных на упрощение разработки Web-приложений. Для реализации серверной части использована программная платформа Node.js. Для хранения данных в проекте задействована нереляционная база данных MongoDB, что позволило ускорить процесс разработки. В качестве редактора программного кода был задействован Visual Studio Code так как это бесплатный, удобный и хорошо оптимизированный редактор кода с широким функционалом для создания и отладки современных веб-приложений.

Результатом разработки стало создание информационной системы поддержки пользователей. Разработанная система реализована в виде Web-приложения и имеет максимально простой интерфейс. В приложении реализованы такие функции как «Регистрация обращений», «Авторизация», «Управление заявками», «Управление пользователями». Главным преимуществом системы является возможность легко и быстро зарегистрировать сведения об инциденте. Для работы с заявками предоставлен простой и гибкий функционал. Имеется возможность быстро и в удобном виде получать подробную информацию о заявках, обновлять статус заявок в несколько кликов. Управление пользователями реализовано достаточно просто. Информационная система соответствует предъявленным требованиям, а также имеет ряд возможностей для дальнейшей модернизации. Одним из вариантов модернизации системы может быть добавление функции «Создание заявок об изменении расписания».

Как любой бизнес-проект, программный продукт создается на базе понимания его эффективности с точки зрения востребованности и прибыльности. Чаще всего руководители компаний не рассматривают прибыльность как главный критерий в реализации ИС, в таком случае оценивается развитие таких конкурентоспособных качеств предприятия, как повышение производительности труда, сокращение издержек, улучшения качества конечного продукта и увеличения гибкости в производстве. Разработанная система как раз преследует одну из таких целей – сокращение издержек, поэтому для отражения экономической эффективности системы используются такие показатели как расчет суммы капитальных вложений, годовой экономии и срока окупаемости капитальных вложений.

Согласно результатом расчётов, затраты в первый год составят 42016,18 рублей и дополнительно 14166,18 рублей за каждый последующий год эксплуатации информационной системы. С другой стороны, экономия от использования системы поддержки пользователей за счёт сокращения потерь за каждый год эксплуатации составит до 49003,5 рублей. Первоначальные вложения окупятся уже на 11 месяц, а использование информационной системы позволит сэкономить за три года сумму в размере 76661,91 рублей, что подтверждает эффективность разработанного продукта.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Агафонова М.С., Агафонов П.В. Современные наукоемкие технологии // Успехи современного естествознания. –2013. –No 10-1. –С. 130.
2. Агафонова М. С., Кулешова И. П., Зелепукина В. А. Целесообразность использования информационных систем на предприятии // Научно-методический электронный журнал «Концепт». – 2017. – Т. 39. – С. 371–375. – URL: <http://e-koncept.ru/2017/970401.htm>.
3. Алистер Коберн. Современные методы описания функциональных требований к системам. М.: издательство «Лори», 2002. – 263 с.
4. Бархатов В.И., Плетнёв Д.А. Институционально-экономическая эффективность корпорации: содержание, критерии, показатели и факторы // Учёные записки Тамбовского регионального отделения Вольного экономического общества России. Том 5, вып.1. – Тамбов: Издательство Тамбовского государственного университета им. Г.Р.Державина, 2003. – С. 36-46.
5. Бархатов В.И., Плетнёв Д.А., Бородкин В.П., Подшивалов Д.В. Экономическая эффективность корпорации в транзитивной экономике. Челябинск: Издательство ЮУрГУ, 2004, 312 с.
6. Ватолина О. В. Эффективность информационных технологий: курс лекций. Хабаровск, ТОГУ, 2012. <http://dotogu.ru/mod/data/view.php?id=24233>
7. Вигерс Карл Разработка требований к программному обеспечению/Пер, с англ. — М.:Издательско-торговый дом «Русская Редакция», 2004. —576с.: ил.
8. Глоссарий по информационному обществу / Под общ. ред. Ю. Е. Хохлова. — М.: Институт развития информационного общества, 2009. — 160 с. http://www.iis.ru/docs/is.glossary.2009.pdf
9. ГОСТ РВ 51987[35; 131]
10. ГОСТ Р ИСО/МЭК 12207/99. Государственный стандарт РФ. Информационная технология. Процессы жизненного цикла информационных систем. Издание официальное. – М., 1999.
11. ГОСТ Р 53622-2009. Информационные технологии. Информационно-вычислительные системы. Стадии и этапы жизненного цикла, виды и комплектность документов
12. Интернет энциклопедия «Академик» <https://dic.academic.ru/dic.nsf/ruwiki/164510>
13. Информационные системы : учеб. пособие / Е.В. Бурцева, И.П. Рак, А.В. Селезнев, А.В. Терехов, В.Н. Чернышов. – Тамбов : Изд-во Тамб. гос. техн. ун-та, 2009. – 128 с. – 150 экз. – ISBN 978-5-8265-0874-9
14. Кадушин А., Михайлова Н., [Информационные технологии](http://www.iteam.ru/publications/it/): [внедрение и эффективность](http://www.iteam.ru/publications/it/section_53/), http://www.iteam.ru
15. Козаченко В.Е. Управление общей стоимостью владения КИС <http://www.cfin.ru>
16. Леффингуелл Д., Уидриг Д. Принципы работы с требованиями к программному обеспечению. М.: ИД “Вильямс”, 2002.
17. Лисин Н. -  Лоскутная автоматизация, или как управлять «зоопарком» программ //BYTE Россия, 2009 http://www.bytemag.ru/articles/detail.php?ID=14862
18. Мацяшек Лешек. Анализ требований и проектирование систем. Разработка информационных: с Диалектика-Вильямс
19. Новиков. Л. - Введение в Rational Unified Process. <http://www.interface.ru/rational/interface/151199/rup/main.htm>
20. Орлик С., Булуй Ю. Введение в программную инженерию и управление жизненным циклом ПО Программная инженерия. Программные требования. Copyright © Сергей Орлик, 2004-2005. <http://www.sorlik.ru/swebok/3-1-software_engineering_requirements.pdf>
21. Остроух А. В. «Основы информационных технологий. Учебник» Издательство: "Академия" (2014)Формат: Твердая глянцевая, 208 стр.ISBN: 9785446805884
22. Остроух А.В., Суркова Н.Е. - Проектирование информационных систем : монография /. — Санкт-Петербург : Лань, 2019. — ISBN 978-5-8114-3404-6. — Текст : электронный // Электронно-библиотечная система «Лань» : [сайт]. — URL: https://e.lanbook.com/book/118650 (дата обращения: 30.10.2019). — Режим доступа: для авториз. пользователей. — С. 9.).
23. Официальный сайт КГБУЗ «Городская клиническая поликлиника №3» <https://www.gkp3.ru/chapters/show/about>
24. Пятков М. Экономика информационных технологий, М., 2001.
25. Результаты третьего исследования Dell EMC <https://comptek.ru/news/dell_emc/5126>
26. Федеральный закон "Об информации, информационных технологиях и о защите информации" от 27.07.2006 N 149-ФЗ <https://duma.consultant.ru/documents/878565?items=1&page=2>
27. Хай Г.А. - Информатика для медиков : учебное пособие - 2009. - 223 с. <http://vmede.org/sait/?id=Informatika_xai_2009&menu=Informatika_xai_2009&page=15>
28. Цыгалов Ю. Экономическая эффективность инвестиций в ИТ: оптимальный метод оценки, <http://www.pcweek.ru>
29. IEEE Standard Glossary of Software Engineering Terminology/IEEE Std 610.12-1990
30. IEEE Guide to the Software Engineering Body of Knowledge (1) - SWEBOK®, 2004. – <http://www.swebok.org>
31. Kajko-Mattsson, Mira. Problems within front-end support (англ.) // [Journal of Software Maintenance and Evolution: Research and Practice](https://en.wikipedia.org/wiki/Journal_of_Software:_Evolution_and_Process) (англ.)[русск.](https://ru.wikipedia.org/w/index.php?title=Journal_of_Software:_Evolution_and_Process&action=edit&redlink=1) : journal. — Vol. 16, no. 4/5. — P. 309—329. — [DOI](https://ru.wikipedia.org/wiki/Doi):[10.1002/smr.298](https://dx.doi.org/10.1002%2Fsmr.298).
32. Walker, Gary. IT Problem Management (Harris Kern’s Enterprise Computing Institute Series). — Upper Saddle River : [Prentice Hall](https://ru.wikipedia.org/w/index.php?title=Prentice_Hall&action=edit&redlink=1), 2001. — P. 85–113.
33. William S. Davis, David C. Yen. The Information System Consultant's Handbook. Systems Analysis and Design. — CRC Press, 1998. — 800 с. — [ISBN 0849370019](https://ru.wikipedia.org/wiki/%D0%A1%D0%BB%D1%83%D0%B6%D0%B5%D0%B1%D0%BD%D0%B0%D1%8F:%D0%98%D1%81%D1%82%D0%BE%D1%87%D0%BD%D0%B8%D0%BA%D0%B8_%D0%BA%D0%BD%D0%B8%D0%B3/0849370019).

ПРИЛОЖЕНИЕ А

(Обязательное)

Программный код

const express = require("express");

const mongoose = require("mongoose");

const session = require("express-session");

var MongoDBStore = require("connect-mongodb-session")(session);

const app = express();

const bodyParser = require("body-parser");

const userRouter = require("./routes/userRouter.js");

const homeRouter = require("./routes/homeRouter.js");

const workSpaceRouter = require("./routes/workSpaceRouter.js");

const supportRequestRouter = require("./routes/supportRequestRouter.js");

var User = mongoose.model("User");

mongoose.connect("mongodb://localhost:27017/usersdb", {

  useNewUrlParser: true,

  useUnifiedTopology: true

});

app.use(

  session({

    store: new MongoDBStore({

      // MongoDB connection string .

      uri: "mongodb://localhost:27017/usersdb",

      collection: "mySessions"

    }),

    resave: true,

    saveUninitialized: true,

    secret: "DjrureEur8#7666hf9W",

    cookie: { maxAge: 36000000 }

  })

);

app.set("view engine", "hbs");

app.use(bodyParser.urlencoded({ extended: false }));

app.use(bodyParser.json()); // Для парсинга json

app.use("/createNewSupportRequest", supportRequestRouter, function(req, res) {

  console.log(req.session);

  console.log("[APP] createNewSupportRequest");

});

app.use("/logoff", function(req, res) {

  console.log("[APP] Logoff");

  req.session.destroy();

  res.redirect("/login");});

Продолжение приложения А

app.get("/login", function(req, res) {

  console.log("[APP] Get /login");

  res.render("login");

});

app.post("/login", function(req, res) {

  console.log(req.body);

  User.findOne({ login: req.body.login, password: req.body.password }, function(

    err,

    currentUser

  ) {

    if (err) {

      console.log(err);

      return response.sendStatus(400);

    }

    if (currentUser) {

      req.session.userLogin = currentUser.login;

      req.session.admin = currentUser.admin;

      res.render("workspace.hbs", {

        user: currentUser

      });

    } else {

      res.render("login.hbs");

    }

  });

  console.log(req.session);

  console.log(req.session.userLogin);

});

app.use("/workspace", workSpaceRouter, function(req, res) {

  console.log("[APP] workSpaceRouter");

});

app.use("/users", userRouter, function(req, res) {

  console.log("[APP] userRouter");

});

app.use("/homepage", homeRouter, function(req, res) {

  console.log(req.session);

  console.log("[APP] homeRouter");

});

app.use(function(req, res, next) {

  console.log("Redirected to homepage | URL: " + req.url);

  res.redirect("/homepage");

});

app.listen(3000, function() {

Продолжение приложения А

  console.log("Сервер ожидает подключения...");

});

const express = require("express");

const homeController = require("../controllers/homeController.js");

const homeRouter = express.Router();

homeRouter.get("/about", homeController.about, function(req, res) {

  console.log("[homeRouter] /about");

});

homeRouter.get("/", homeController.homePage, function(req, res) {

  console.log("[homeRouter] /");

});

module.exports = homeRouter;

const express = require("express");

const supportRequestController = require("../controllers/supportRequestController.js");

const supportRequestRouter = express.Router();

supportRequestRouter.post(

  "/modify",

  supportRequestController.modify,

  function() {

    console.log("[supportRequestRouter] /");

  }

);

supportRequestRouter.post(

  "/",

  supportRequestController.createSupportRequest,

  function(req, res) {

    console.log("[supportRequestRouter] /");

  }

);

module.exports = supportRequestRouter;

const express = require("express");

const userController = require("../controllers/userController.js");

const userRouter = express.Router();

userRouter.use("/postuser",userController.postUser,function(){

    console.log("[userRouter] /postuser");

});

userRouter.use("/create", userController.addUser,function(){

    console.log("[userRouter] /create");});

Продолжение приложения А

userRouter.use("/",userController.getUsers,function(){

    console.log("[userRouter] /");

});

module.exports = userRouter;

const express = require("express");

const workSpaceController = require("../controllers/workSpaceController.js");

const workSpaceRouter = express.Router();

workSpaceRouter.get("/getRequests",workSpaceController.getRequests,function(request,response){

    console.log("[workSpaceRouter] /getRequests");

});

workSpaceRouter.use("/",workSpaceController.main,function(request,response){

    console.log("[workSpaceRouter] /");

});

module.exports = workSpaceRouter;

exports.homePage = function(request,response){

    response.render("homePage.hbs");

};

exports.about = function (request, response) {

    response.send("О сайте");

};

const supportRequest = require("../models/supportRequest.js");

exports.createSupportRequest = function(request, response) {

  if (!request.body) return response.sendStatus(400);

  let now = new Date();

  const location = request.body.location;

  const description = request.body.supportRequestDescription;

  const pType = request.body.problemType;

  const sRequest = new supportRequest({

    createLocation: location,

    problemDescription: description,

    сreateDate: now,

    problemType: pType

  });

  sRequest.save(function(err) {

    if (err) return console.log(err);  });

Продолжение приложения А

  console.log("[Controller] createSupportRequest");

  response.render("RequestInfo.hbs", { sRequest });

};

exports.modify = function(request, response) {

  console.log("[Contoller] Modify");

  console.log(request.body);

  let now = new Date();

  supportRequest.updateOne(

    { \_id: request.body.\_id },

    {

      $set: {

        problemType: request.body.problemType,

        problemPriority: request.body.problemPriority,

        modifedDate: now,

        modifedByUser: request.session.userLogin,

        Specialist: request.body.Specialist,

        Status: request.body.Status

      }

    },

    function(err, uRequest) {

      if (err) {

        console.log(err);

        return response.sendStatus(400);

      }

      console.log(uRequest);

    }

  );

};

exports.postUser = function(request, response) {

  if (!request.body) return response.sendStatus(400);

  const userName = request.body.name;

  const userAge = request.body.age;

  const user = new User({ name: userName, age: userAge });

  user.save(function(err) {

    if (err) return console.log(err);

    response.redirect("/users");

  });

};

const User = require("../models/user.js");

exports.addUser = function(request, response) {

  response.render("create.hbs");

};

Продолжение приложения А

exports.getUsers = function(request, response) {

  console.log(request.session);

  console.log(request.session.admin);

  if (request.session.admin == "Да") {

    User.find({}, function(err, allUsers) {

      if (err) {

        console.log(err);

        return response.sendStatus(400);

      }

      response.render("users.hbs", {

        users: allUsers

      });

    });

  } else {

    response.redirect("/logoff");

  }

};

exports.postUser = function(request, response) {

  if (request.session.admin == "Да") {

    if (!request.body) return response.sendStatus(400);

    const userFIO = request.body.FIO;

    const userLogin = request.body.login;

    const userPassword = request.body.password;

    const userAdmin = request.body.admin;

    const user = new User({

      FIO: userFIO,

      login: userLogin,

      password: userPassword,

      admin: userAdmin

    });

    user.save(function(err) {

      if (err) return console.log(err);

      response.redirect("/users");

    });

  } else {

    response.redirect("/logoff");

  }

};

const supportRequest = require("../models/supportRequest.js");

exports.getRequests = function(request, response){

    supportRequest.find({Status:{$ne:'Решено'}}, function(err, allSupportRequests){

        if(err) {

Продолжение приложения А

  console.log(err);

            return response.sendStatus(400);

        }

        console.log('getRequests');

        //console.log(allSupportRequests);

        response.send(allSupportRequests);

    });

};

exports.main = function(req, res){

    console.log(req.session.userLogin);

    if(req.session.userLogin != "" && req.session.userLogin != undefined ){

        res.render("workspace.hbs", {

            user: req.session.userLogin

        });

    }else{

        res.render("login.hbs");

    }

};

const mongoose = require("mongoose");

const Schema = mongoose.Schema;

// Установка схемы

const supportRequestScheme = new Schema({

    //Create

    сreateDate:{type: Date},

    createLocation:{type: String},

    createUser:{ type: String, default: "Неизвестно"},

    //Problem

    problemDescription:{type:String},

    problemType:{type:String, default:"-"},

    problemPriority:{type:String, default:"Не определён"},

    //Modifed

    modifedDate:{ type:Date},

    modifedByUser:{ type:String},

    //Other

    Specialist:{type: String, default: "Не назначен"},

    Status:{type:String, default:"В обработке"}

});

module.exports = mongoose.model("supportRequest",supportRequestScheme);

const mongoose = require("mongoose");

Продолжение приложения А

const Schema = mongoose.Schema;

// Установка схемы

const userScheme = new Schema({

    FIO: String,

    login: String,

    password: String,

    admin: String,

});

module.exports = mongoose.model("User",userScheme);

<!DOCTYPE html>

<html>

<head>

  <meta http-equiv="Content-Type" content="text/html;charset=UTF-8" />

  <meta name="viewport" content="width=device-width,initial-scale=1.0" />

  <title>Список заявок</title>

  <style>

    body {

      background-color: white;

      font-family: 'Times New Roman', Times, serif

    }

    nav {

      background-color: rgb(70, 70, 70);

      text-align: center;

      color: rgb(242, 242, 242);

      height: 20%;

    }

    A:link,

    A:visited,

    A:active,

    A:hover {

      font-weight: bold;

      color: rgb(242, 242, 242);

      text-decoration: none;

    }

    .mainForm {

      position: fixed;

      /\* or absolute \*/

      top: 35%;

      left: 50%;

      width: 350px;

Продолжение приложения А

 height: 850px;

      margin: -175px 0 0 -425px;

    }

    .requestForm {

      background-color: rgb(242, 242, 242);

      height: 450px;

      width: 850px;

      border-radius: 2%;

      border: 1px solid

    }

    #display {

      // background-color: rgb(30, 119, 192);

    }

    .selectedDisplay:hover {

      background-color: blueviolet;

    }

    .requestList1 {

      background-color: rgb(242, 242, 242);

      font-size: large;

      float: left;

      width: 300px;

      height: 350px;

      overflow: scroll;

    }

    .requestList2 {

      background-color: rgb(242, 242, 242);

      font-size: large;

      width: 550px;

      overflow: auto;

    }

    .requestList3 {

      width: 250px;

      overflow: auto;

    }

    .requestList4 {

      width: 300px;

      overflow: auto;

    }

    .divButton {

      text-align: right;

Продолжение приложения А

    }

    H4 {

      text-align: center;

    }

    #loginDiv {

      text-align: right;

      color: rgb(70, 70, 70);

    }

    #modifyButton {}

    .modify {

      color: brown

    }

  </style>

</head>

<body onload="loaded()">

  <nav>

    <h1>Информационная система поддержки пользователей</h1>

    <div><a href="/homepage">[Домашняя страница]</a> <a href="/users/create">[Новый пользователь]</a> <a

        href="/users">[Список пользователей]</a> <a href="/logoff">[Изменение пользователя]</a> <a

        href="/logoff">[Выход]</a></div>

  </nav>

  <div id="loginDiv">{{user.FIO}} [{{user.login}}]

  </div>

  </p>

  <div class="mainForm">

    <fieldset class="requestForm">

      <h4>СПИСОК ЗАЯВОК</h4>

      <select onchange="loaded()" id="listFilter">

        <option value="all">Все заявки</option>

        <option value="new">Новые</option>

        <option value={{user.login}}>Мои</option>

      </select>

      <hr>

      </hr>

      <div class="requestList1">

        <table>

          <tr>

            <th>Тип проблемы</th>

Продолжение приложения А

            <th>Приоритет</th>

          </tr>

          <tr>

            <td>

              <div id="displayType"></div>

            </td>

            <td>

              <div id="displayPriority"></div>

            </td>

          </tr>

        </table>

      </div>

      <div class="requestList2">

        <table>

          <td class="requestList3">

            <div class="des">ID заявки: </div>

            <div class="des">Дата создания: </div>

            <div class="des">Кабинет/Место: </div>

            <div class="des">Кем создано: </div>

            <div class="des">Описание проблемы: </div>

            <div class="des">Тип проблемы: </div>

            <div class="des">Приоритет: </div>

            <div class="des">Специалист: </div>

            <div class="des">Статус: </div>

            <div class="des">Последнее обновление: </div>

            <div class="des">Кем обновлено: </div>

            </p>

          </td>

          <td class="requestList4">

            <div id="displayID"></div>

            <div id="displayCreateDate"></div>

            <div id="displayCreateLocation"></div>

            <div id="displayCreateUser"></div>

            <div id="displayProblemDescription"></div>

            <div id="displayProblemType" contenteditable='true' class="modify"></div>

            <div id="displayProblemPriority" contenteditable='true' class="modify"></div>

            <div id="displaySpecialist" contenteditable='true' class="modify"></div>

            <div id="displayStatus" contenteditable='true' class="modify"></div>

            <div id="displayModifedDate"></div>

Продолжение приложения А

            <div id="displayModifedByUser"></div>

            <div class="divButton"><input id="modifyButton" type="button" value="Сохранить изменения" onclick="modify()"

                hidden /></div>

          </td>

        </table>

      </div>

    </fieldset>

  </div>

  <script>

    var listDisplayType = document.getElementById('displayType');

    var listDisplayPriority = document.getElementById('displayPriority');

    function modify() {

      let modifyJson = {

        \_id: document.getElementById('displayID').textContent,

        problemType: document.getElementById('displayProblemType').textContent,

        problemPriority: document.getElementById('displayProblemPriority').textContent,

        Specialist: document.getElementById('displaySpecialist').textContent,

        Status: document.getElementById('displayStatus').textContent

      }

      let options = {

        method: 'POST',

        body: JSON.stringify(modifyJson),

        headers: {

          'Content-Type': 'application/json'

        }

      }

      fetch('http://localhost:3000/createNewSupportRequest/modify', options)

        .then(res => res.json())

        .then(res => console.log(res));

      loaded();

    };

    function loaded() {

      fetch('http://localhost:3000/workspace/getRequests')

        .then(response => response.json()) // преобразуем ответ в json

Продолжение приложения А

        .then(data => {

          // Зачистка списка запросов

          while (listDisplayType.firstChild) {

            listDisplayType.removeChild(listDisplayType.firstChild);

          }

          while (listDisplayPriority.firstChild) {

            listDisplayPriority.removeChild(listDisplayPriority.firstChild);

          }

          var i = 0;

          while (i < data.length) {

            if (document.getElementById("listFilter").value == "new" && data[i].Status != 'В обработке') {

              i++;

              continue;

            };

            if (document.getElementById("listFilter").value != "all" && document.getElementById("listFilter").value != "new" && data[i].Specialist != document.getElementById("listFilter").value) {

              i++;

              continue;

            };

            let div = document.createElement('div');

            let div2 = document.createElement('div');

            div.textContent = data[i].problemType;

            div2.textContent = data[i].problemPriority;

            div.setAttribute('ID', data[i].\_id)

            div.setAttribute('сreateDate', data[i].сreateDate);

            div.setAttribute('CreateLocation', data[i].createLocation);

            div.setAttribute('CreateUser', data[i].createUser);

            div.setAttribute('problemDescription', data[i].problemDescription);

            div.setAttribute('ProblemType', data[i].problemType);

            div.setAttribute('ProblemPriority', data[i].problemPriority);

            div.setAttribute('ModifedDate', data[i].modifedDate);

            div.setAttribute('ModifedByUser', data[i].modifedByUser);

            div.setAttribute('Specialist', data[i].Specialist);

            div.setAttribute('Status', data[i].Status);

            listDisplayType.appendChild(div);

            listDisplayPriority.appendChild(div2);

            div.classList = "selectedDisplay"

            div2.classList = "selectedDisplay"

            div.onclick = () => {

              clicked(div);

            }

            div2.onclick = () => {

              clicked(div);

            }

            i++;

          }

        })

        .catch(error => console.error(error))

      clearInfo();

    }

    function clicked(div) {

      document.getElementById("displayID").textContent = div.getAttribute('ID');

      document.getElementById("displayCreateDate").textContent = div.getAttribute('сreateDate');

      document.getElementById("displayCreateLocation").textContent = div.getAttribute('CreateLocation');

      document.getElementById("displayCreateUser").textContent = div.getAttribute('CreateUser');

      document.getElementById("displayProblemDescription").textContent = div.getAttribute('problemDescription');

      document.getElementById("displayProblemType").textContent = div.getAttribute('ProblemType');

      document.getElementById("displayProblemPriority").textContent = div.getAttribute('ProblemPriority');

      document.getElementById("displayModifedDate").textContent = div.getAttribute('ModifedDate');

      document.getElementById("displayModifedByUser").textContent = div.getAttribute('ModifedByUser');

      document.getElementById("displaySpecialist").textContent = div.getAttribute('Specialist');

      document.getElementById("displayStatus").textContent = div.getAttribute('Status');

      document.getElementById("modifyButton").hidden = false;

    }

    function clearInfo(div) {

Продолжение приложения А

      document.getElementById("displayID").textContent = '';

      document.getElementById("displayCreateDate").textContent = '';

      document.getElementById("displayCreateLocation").textContent = '';

      document.getElementById("displayCreateUser").textContent = '';

      document.getElementById("displayProblemDescription").textContent = '';

      document.getElementById("displayProblemType").textContent = '';

      document.getElementById("displayProblemPriority").textContent = '';

      document.getElementById("displayModifedDate").textContent = '';

      document.getElementById("displayModifedByUser").textContent = '';

      document.getElementById("displaySpecialist").textContent = '';

      document.getElementById("displayStatus").textContent = '';

      document.getElementById("modifyButton").hidden = true;

    }

  </script>

</body>

<html>