МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

УО «Полесский государственный университет»

Кафедра высшей математики и информационных технологий

Курсовой проект по дисциплине

**КОРПОРАТИВНЫЕ ИНФОРМАЦИОННЫЕ СИСТЕМЫ**

на тему:

Программный модуль оценки качества обслуживания процессов банка аутсорсингвой компании

Исполнитель:

студент группы 15ИТ-3с\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /Деденко Е.С./

Руководитель:

Доцент, к.э.н.\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /Дунько Э.М./

Пинск 2017

**АННОТАЦИЯ**

Пояснительная записка содержит: 34 страницу, 10 рисунков, 14 использованных источников.

Объектом исследования является бизнес-процессы аутсорсинговой компании.

Предметом исследования является оценка качества предоставляемых услуг.

Цель исследования – разработка программного модуля оценки качества обслуживания процессов банка аутсорсингвой компании.

В ходе проектирования была изучена предметная область — бизнес-процессы аутсорсинговой компании. После самостоятельного изучения предметной области задачи курсового проекта были выполнены.

Результатом курсового проекта является полноценный функционирующий программный модуль оценки качества обслуживания процессов банка аутсорсингвой компании. В проекте реализованы следующие задачи:

1. изучена предметная область, определен объект автоматизации;
2. разработаны бизнес-требования к информационной системе;
3. разработана математическая модель и алгоритмы решения;
4. разработано техническое задание;
5. обоснован выбор средств проектирования и разработки;
6. спроектирована информационная система;
7. разработан программный модуль.

СОДЕРЖАНИЕ

[Перечень условных обозначений и сокращений 5](#_Toc482590356)

[Введениe 6](#_Toc482590357)

[1 Анализ предметной области и выявление объекта автоматизации 7](#_Toc482590358)

[1.1 Методы анализа и проектирования информационных систем 8](#_Toc482590359)

[1.2 Основные понятия, используемые в объектно-ориентированном   
подходе……………………………………………………………………………….9](#_Toc482590360)

[2 Бизнес требование к информационной системе 12](#_Toc482590361)

[3 Математическая модель реализации функциональных задач проектируемой информационной системы 13](#_Toc482590362)

[4 Техническое задание на разработку информаци-онной системы 15](#_Toc482590363)

[5 Обоснование выбора средств проектирования и последующей разработки информационной системы 16](#_Toc482590364)

[6 Информационная модель предметной области и структура базы данных 19](#_Toc482590365)

[6.1 IDEF0 модель 19](#_Toc482590366)

[6.2 Диаграмма классов 23](#_Toc482590367)

[6.3 Структура базы данных 24](#_Toc482590368)

[7 Разработка программного модуля 26](#_Toc482590369)

[8 Выявление потенциальных рисков и предвари-тельная оценка эффектов 28](#_Toc482590370)

[Заключение 29](#_Toc482590371)

[Список использованных источников 30](#_Toc482590372)

[Приложение А 32](#_Toc482590373)

# Перечень условных обозначений и сокращений

В настоящей пояснительной записке применяются следующие термины, обозначения и сокращения.

CASE-средства (Computer - Aided Software Engineering) –– это инструмент, который позволяет автоматизировать процесс разработки информационной системы и программного обеспечения.

БД –– База Данных.

БП –– Бизнес-Процесс.

ИТ –– Информационные Технологии.

ПО –– Программное Обеспечение.

MVC –– Model View Controller [1].

ООП –– Объектно-Ориентированное Программирование.

BPM (Business Process Management) –– это класс программных продуктов, которые помогают управлять бизнес процессами организации.

SQL (Structured Query Language — язык структурированных запросов) — язык программирования, применяемый для создания, модификации и управления данными в реляционной базе данных, управляемой соответствующей системой управления базами данных.

UML (Unified Modeling Language — унифицированный язык моделирования) — [язык](https://ru.wikipedia.org/wiki/%D0%AF%D0%B7%D1%8B%D0%BA_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F) [графического](https://ru.wikipedia.org/wiki/%D0%92%D0%B8%D0%B7%D1%83%D0%B0%D0%BB%D0%B8%D0%B7%D0%B0%D1%86%D0%B8%D1%8F) описания для [объектного моделирования](https://ru.wikipedia.org/w/index.php?title=%D0%9E%D0%B1%D1%8A%D0%B5%D0%BA%D1%82%D0%BD%D0%BE%D0%B5_%D0%BC%D0%BE%D0%B4%D0%B5%D0%BB%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5&action=edit&redlink=1) в области [разработки программного обеспечения](https://ru.wikipedia.org/wiki/%D0%A0%D0%B0%D0%B7%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%BA%D0%B0_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D0%BE%D0%B3%D0%BE_%D0%BE%D0%B1%D0%B5%D1%81%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%B8%D1%8F), [моделирования бизнес-процессов](https://ru.wikipedia.org/w/index.php?title=%D0%91%D0%B8%D0%B7%D0%BD%D0%B5%D1%81-%D0%BC%D0%BE%D0%B4%D0%B5%D0%BB%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5&action=edit&redlink=1), [системного проектирования](https://ru.wikipedia.org/wiki/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B5%D0%BA%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5) и отображения [организационных структур](https://ru.wikipedia.org/wiki/%D0%9E%D1%80%D0%B3%D0%B0%D0%BD%D0%B8%D0%B7%D0%B0%D1%86%D0%B8%D0%BE%D0%BD%D0%BD%D0%B0%D1%8F_%D1%81%D1%82%D1%80%D1%83%D0%BA%D1%82%D1%83%D1%80%D0%B0) [2].

# [ВВЕДЕНИE](#_Toc226110007)

В настоящее время ИТ-сфера является самой быстрорастущей и активно-развивающейся сферой человеческой деятельности. Причина такого быстрого обусловлена тем, что именно ИТ при правильном подходе к автоматизации помогает другим отраслям экономики оптимизировать труд человека, снизить рутинные задачи до минимума, повысить скорость выпуска продукции или оказания услуги, исключить или снизить до минимума влияние человеческого фактора, а также, что не маловажно, снизить производственные затраты. Выше была перечислена лишь малая доля тех преимуществ, которые предоставляют автоматизированные системы. Руководители любого предприятия осознают эти преимущества и стремятся максимально автоматизировать бизнес-процессы, так как только в этом случае можно получить конкурентное преимущество. Но абсолютно каждое предприятие (вне зависимости от размера) при построении и дальнейшей поддержке автоматизированных систем сталкивается с рядом проблем. Первой проблемой является нехватка квалифицированных кадров: лишь специалист высокой квалификации с аналитическим складом ума способен разработать систему, которая будет предусматривать все потребности предприятия и при этом будет соответствовать требованиям безопасности и надежности. Второй крупной проблемой является оплата труда работников, которые разрабатывают и поддерживают данные автоматизированные системы, так специалист высокой квалификации стоит не малых денежных затрат, хотя на этапе поддержки не требуется ежедневное присутствие специалиста такого уровня. Выше сказанное наводит на мысль, что содержать отдел ИТ-специалистов не всегда целесообразно в плане денежных затрат.

В XIX веке очень развит капитализм, эта идеология построена на том, что если у потребителя есть проблема, то ее можно решить за денежное вознаграждение. Эта правило не минуло и проблем, связанных с автоматизацией бизнес процессов, что привело к возникновению на рынке широкого секатора аутсорсинговых компании.

Таким образом, данный курсовой проект будет связанна непосредственно с изучением проблематики аутсорсинговых компаний по части менеджмента и обратной связи с клиентом в банковской сфере.

Целью данного курсового проекта является разработка программного модуля оценки качества обслуживания процессов банка аутсорсингвой компании [3].

# 1 АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ И ВЫЯВЛЕНИЕ ОБЪЕКТА АВТОМАТИЗАЦИИ

ИТ-аутсорсинг (англ. IT outsourcing) –– частичная или полная передача работ по поддержке, обслуживанию и модернизации ИТ-инфраструктуры в руки компаний, специализирующихся на абонентском обслуживании организаций и имеющих штат специалистов различной квалификации. Для них выполнение подобных работ является профильным направлением деятельности [4].

ИТ-аутсорсинг предполагает делегирование внешней специализированной компании решение вопросов, связанных с разработкой, внедрением и сопровождением информационных систем, как целиком на уровне инфраструктуры предприятия (сопровождение оборудования или ПО), так и объёмов работ, связанных с развитием и/или поддержкой функционирования отдельных участков системы (программирование, хостинг, тестирование и т.д.)

Стоит отметить, что аутсорсиноговые компании имеют клиентскую базу, а не одного клиента, за счет чего могут получать прибыль от нескольких предприятий и как правило их услуги гораздо дешевле нежели содержание собственного штата сотрудников.

Аутсорсинговые компании решают проблемы других предприятий, но и сами сталкиваются с некоторыми трудностями. В рамках данного курсового проекта будет рассмотрена только одна проблема, связанная с менеджментом. Эта проблема имеет следующую формулировку: оценки качества обслуживания процессов. Стоит отметить, что эта проблема будет рассмотрена в контексте банковской системы.

Главная проблематика состоит в том, что если отдел менеджмента и контроля качества не будет отслеживать качество предоставляемых услуг, то аутсорсинговая компания рискует потерять клиентов, что несомненно отразиться на прибыли компании.

Для решения этой задачи компании необходимо собирать отзывы клиентов, жалобы, замечания, а также статистику. Данный функционал могут обеспечить продвинутые CRM-системы. Но CRM –– это очень обширное понятие и рассмотреть все возможности этих систем в рамках курсового проекта невозможно, поэтому было принято решения разработать программный модуль, отвечающий за сбор отзывов клиентов, принятие жалоб и замечаний. Удобнее всего реализовать такой модуль с помощью веб-интерфейса, т.к. такой подход не предполагает лишних действий со стороны клиента (например, загрузка и установка стороннего программного обеспечения).

Таким образом объектом автоматизации является процесс взаимодействия с клиентом, в данном случае в качестве клиента выступает банк, который воспользовался услугами аутсорсинга.

## Методы анализа и проектирования информационных систем

Чтобы перейти непосредственно к реализации и разработке информационной системы необходимо определиться с подходом к её построению.

Существует два наиболее популярных подхода (парадигмы) к анализу и проектированию информационных систем: структурный и объектно-ориентированный.

Первое отличие этих подходов друг от друга заключается в принципах декомпозиции и структурной организации элементов (компонентов, модулей) системы. Согласно этим принципам система представляет собой структуру, состоящую из четко выраженных модулей, связанных между собой определенными отношениями.

При использовании структурного подхода выполняется функциональная (процедурная, алгоритмическая) декомпозиция системы, т. е. она представляется в виде иерархии (дерева) взаимосвязанных функций. На высшем уровне система представляется единым целым с наивысшей степенью абстракции и по мере детализации (добавления уровней) разбивается на функциональные компоненты с более конкретным содержанием.

Второй вид декомпозиции – объектно-ориентированный. В рамках этого подхода система разбивается на набор объектов, соответствующих объектам реального мира, взаимодействующих между собой путем посылки сообщений.

Вторым отличием является объединение в объекте как атрибутивных данных (характеристики, свойства), так и поведения (функции, методы). В функционально-ориентированных системах функции и данные хранятся (существуют) отдельно.

Третье отличие двух подходов заключается в структурной организации внутри модулей системы. В структурном подходе модуль состоит из функций, иерархически связанных между собой отношением композиции (англ. Part of – часть-целое), т. е. функция состоит из подфункций, подфункция из подподфункций и т.д. В объектно-ориентированном подходе иерархия выстраивается с использованием двух отношений: композиции и наследования (англ. IS A – это есть). При этом в объектно-ориентированном подходе «объект-часть» может включаться сразу в несколько «объектов-целое». Таким образом, модуль в структурном подходе представляется в виде дерева, а в объектно-ориентированном подходе – в виде ориентированного графа, т. е. с помощью более общей структуры.

Наиболее популярными методологиями, поддерживающими данный подход, в настоящий момент являются:

– унифицированный процесс (Unified Process, UP);

– экстремальное программирование (eXtreme Programming, XP);

– гибкое моделирование (Agile Modeling, AM).

Базовым средством фиксации (документирования) результатов проектирования систем посредством этих методологий является Унифицированный язык моделирования (Unified Modeling Language, UML).

Также объектно-ориентированный подход наиболее удобен для разработки, поддержки и тестирования крупных проектов, что сыграло ключевую роль при выборе подхода разработки.

## Основные понятия, используемые в объектно-ориентированном подходе

Термин «объект» или эквивалентные ему понятия появились практически независимо в различных областях, связанных с компьютерами, в процессе их разработки.

Объект – это абстракция реальной или воображаемой сущности с четко выраженными концептуальными границами, индивидуальностью (идентичностью), состоянием и поведением.

Абстракция – форма познания, основанная на мысленном выделении существенных свойств и связей предмета и отвлечении от других, частных его свойств и связей. При этом «существенное» и «частное» должны рассматриваться с точки зрения решаемой задачи (предметной области). В объектно-ориентированном подходе абстракция – это модель сущности, описывающая ее свойства и поведение.

Примерами реальных (физических, осязаемых) сущностей могут служить поезд, стрелочный перевод или инженер службы пути, а воображаемых – технология проведения капитального ремонта пути или оптимальная траектория движения поезда (режимы и скорость в зависимости от текущего положения поезда на участке).

Индивидуальность – это свойство сущности, с помощью которого ее можно отличить от других.

Для концептуальной группировки однотипных объектов в объектно-ориентированном подходе используется понятие «класс». Класс – это множество объектов, имеющих общую структуру и поведение. Таким образом, класс – это шаблон, на основе которого генерируются (создаются) однотипные объекты. В качестве синонима понятия «объект» часто употребляют понятие «экземпляр класса».

Каждый класс и соответственно объект характеризуются строго определенным набором атрибутов и методов. Текущие значения атрибутов четко определяют текущее состояние объекта. Набор методов и их алгоритмическая реализация определяют поведение объекта (класса объектов).

Говоря об объектно-ориентированном подходе, в первую очередь отмечают наследование, инкапсуляцию и полиморфизм. Эти механизмы и принципы проектирования более естественно и полно реализованы в объектно-ориентированном подходе по сравнению со структурным.

Наследование — один из четырёх важнейших механизмов объектно-ориентированного программирования (наряду с инкапсуляцией, полиморфизмом и абстракцией), позволяющий описать новый класс на основе уже существующего (родительского), при этом свойства и функциональность родительского класса заимствуются новым классом. Применительно к классам это означает, что дочерний класс (узкая категория) полностью включает в себя (наследует) все атрибуты и методы, определенные в родительском классе (общей категории). При этом в дочернем классе могут быть определены дополнительные атрибуты и методы.

Инкапсуляция (информационная закрытость) – принцип, в соответствии с которым содержание внутреннего устройства элементов системы должно быть скрыто друг от друга. Этот принцип предписывает обмен информацией между объектами системы только в минимально необходимом объеме, ограничение доступа к атрибутам и методам объектов (классов) со стороны других объектов (классов) и полное скрытие алгоритмической реализации методов от других объектов (классов).

Полиморфизм – принцип построения элементов модели так, чтобы они могли принимать различные внешние формы или функциональность (поведение) в зависимости от обстоятельств. Например, методы draw() (нарисовать) или calculateS() (рассчитать площадь) для классов «круг» и «ромб», определенных путем наследования атрибутов и методов родительского класса «фигура», алгоритмически должны быть реализованы по-разному.

С учетом приведенных выше определений сущность объектно-ориентированного подхода к анализу и проектированию информационных систем заключается в декомпозиции системы на классы, которые соответствуют однотипным объектам предметной области, и построении из них иерархии в виде ориентированного графа с использованием отношений композиции и наследования [5].

Базовыми составляющими объектно-ориентированного подхода являются:

* унифицированный процесс;
* унифицированный язык моделирования;
* шаблоны проектирования.

Унифицированный процесс – это процесс разработки программного обеспечения (ПО), который обеспечивает упорядоченный подход к распределению задач и обязанностей в организации-разработчике. Унифицированный процесс охватывает весь жизненный цикл ПО, начиная с определения требований и заканчивая сопровождением, и представляет собой обобщенный каркас (шаблон, скелет), который может быть применен (специализирован) для разработки и сопровождения широкого круга систем.

Неотъемлемой частью унифицированного процесса является UML – язык (система обозначений) для определения, визуализации и конструирования моделей системы в виде диаграмм и документов на основе объектно-ориентированного подхода. Следует отметить, что Унифицированный процесс и UML разрабатывались совместно.

На стадиях анализа и проектирования часто используются так называемые шаблоны (паттерны) проектирования. Шаблон – это именованная пара «проблема/решение», содержащая готовое обобщенное решение типичной проблемы. Как правило, шаблон помимо текстового описания содержит также одну или несколько диаграмм UML (например, диаграммы классов, последовательности и/или коммуникации), графически иллюстрирующих состав и структуру классов, а также особенности их взаимодействия при решении поставленной проблемы. Шаблоны разрабатываются опытными профессионалами и являются проверенными, эффективными (порой оптимальными) решениями. Применение шаблонов может резко сократить затраты и повысить качество разработки ПО.

В отличие от структурного подхода объектно-ориентированный имеет ряд преимуществ:

* описание системы в виде объектов больше соответствует содержательному смыслу предметной области. Например, при использовании структурного подхода БД должна удовлетворять требованиям нормализации, в соответствии с которыми данные по одному и тому же объекту могут храниться в нескольких таблицах;
* сущности реального мира, как правило, обладают поведением, что в объектно-ориентированном проектировании отражается с помощью определения методов класса. В структурном подходе данные (атрибуты) и алгоритмы (методы) существуют отдельно друг от друга;
* объединение атрибутов и методов в объекте (классе), а также инкапсуляция позволяют добиться большей внутренней и меньшей внешней связности между компонентами системы. Это облегчает решение проблем: o адаптации системы к изменению существующих или появлению новых требований; o сопровождения системы на разных стадиях жизненного цикла; o повторного использования компонентов.
* объектно-ориентированный подход позволяет легче организовать параллельные вычисления, так как каждый объект обладает собственными значениями характеристик (атрибутов) и поведением, за счет чего можно добиться его автономной работы;
* Case-средства, поддерживающие объектно-ориентированный подход, на основе информации об объектах позволяют достичь большей степени автоматизации кодогенерации. Case-средства, поддерживающие структурный подход, хорошо справляются с генерацией структур БД. Однако следует отметить, что эта структура должна удовлетворять требованиям нормализации. В связи с чем автоматическая кодогенерация (например, экранов или функций обработки данных) возможна лишь в редких случаях.

При проектировании программного модуля поддержки принятия решений по выбору процессов для первоочередной оптимизации будет использоваться объектно-ориентированный подход, так как он имеет ряд преимуществ в сравнении с структурной парадигмой, а также поддерживается средой проектирования «IBM Rational Rose».

# 2 БИЗНЕС ТРЕБОВАНИЕ К ИНФОРМАЦИОННОЙ СИСТЕМЕ

Бизнес требования к информационной системе рассмотрим на трех уровнях: бизнес-требования, уровень требований пользователей, функциональные требования [6].

Бизнес-требования:

– сбор отзывов, жалоб, замечаний и предложений клиентов;

– обработка отзывов;

– определение низкокачественных услуг;

– определение отдела с низкими показателями.

Уровень требований пользователей:

– система должна иметь интуитивно понятный пользовательский интерфейс;

– веб-приложение должно иметь кросс-браузерную верстку запускаться и выглядеть одинаково во всех современных веб-браузерах, в том числе и Internet Explorer не ниже 9 версии;

– веб-приложение должно работать на мобильных гаджетах (планшеты, смартфоны, ультрабуки и нетбуки с низким разрешением экрана).

Функциональные требования:

– сохранение результатов опроса в БД;

– сохранение пользовательских жалоб, замечаний и предложений в БД;

– вывод на экран отделов и сопутствующих услуг;

– обработка качественного показателя;

– обработка количественного показателя;

– обработка интегрального (общего) показателя;

– вывод результатов эффективности работы отделов и качества предоставляемых услуг.

Данные требования являются частью технического задания, которое будет применяться при работки автоматизированной системы.

# 3 МАТЕМАТИЧЕСКАЯ МОДЕЛЬ РЕАЛИЗАЦИИ ФУНКЦИ-ОНАЛЬНЫХ ЗАДАЧ ПРОЕКТИРУЕМОЙ ИНФОРМАЦИОННОЙ СИСТЕМЫ

Для оценки бизнес-процессов определены следующие критерии:

1. Качественный критерий –– критерий, который основывается на оценке пользователем качества предоставляемой услуги по пятибалльной шкале.
2. Количественный критерий –– данный критерий основан на количестве жалоб, отправленных клиентом. Чем меньше количество жалоб, тем эффективнее работает отдел.
3. Интегральный критерий –– это суммарный критерий, согласно которому дается общая характеристика на основании среднего значения качественной и количественной оценки.

Оценка качества услуг осуществляется разработанной автоматизированной системой на основании данных полученных в результате сбора информации у клиентов. По результатам, полученных в ходе работы программного продукта, руководитель или отдел менеджмента и контроля качества будут принимать решение, связанное с улучшением качества предоставления услуг.

Расчет качественного критерия осуществляется по формуле 3.1.

 (3.1)

где  *Xi* –– оценка *i*-го пользователя;

*i* –– количественный индекс (номер) пользователя.

Сущность этого показателя заключается в определении средней оценки клиентов по конкретной услуге.

Расчет количественного критерия заключается всего лишь в суммировании количества жалоб клиентов на качество предоставляемых услуг. Данный расчет производиться по формуле 3.2.

 (3.2)

где *Xi ––* фактически подданная жалоба (*Xi* < 5);

*i ––* номер жалобы.

Данный критерий показывает фактическое количество жалоб клиента на качество предоставляемой услуги, этот критерий должен стремиться к 0 для достижения наилучшего качества предоставления услуг.

Интегральный критерий демонстрирует эффективность работы отдела, который предоставляет определенный спектр услуг. Для расчета интегрального критерия используется формула 3.3.

 (3.2)

где *Yi ––* Оценка качества предоставляемой услуги по выбранному отделу

(*Yi* < 5);

*i ––* количественный индекс (номер) оценки услуг.

Стоит отметить, что вычислять интегральный показатель по всем отделам сразу не имеет смысла, так как суть разрабатываемого программного продукта заключается именно в нахождении отдела с низкими показателями эффективности.

# ТЕХНИЧЕСКОЕ ЗАДАНИЕ НА РАЗРАБОТКУ ИНФОРМАЦИ-ОННОЙ СИСТЕМЫ

Основной задачей курсового проета является программный модуль оценки качества обслуживания процессов банка аутсорсингвой компании. Разрабатываемый программный модуль должен обладать интуитивно понятным графическим интерфейсом, высокой степенью надежности, не создавать дискомфорт пользователю при работе в программной подсистеме [7].

Разрабатываемый программный модуль должен осуществлять следующие необходимые задачи:

– сохранение результатов опроса в БД;

– сохранение пользовательских жалоб, замечаний и предложений в БД;

– вывод на экран отделов и сопутствующих услуг;

– обработка качественного показателя;

– обработка количественного показателя;

– обработка интегрального (общего) показателя;

– вывод результатов эффективности работы отделов и качества предоставляемых услуг.

Все функции, способы реализации и прочие технические тонкости, которые не оговорены в данном техническом задании остаются на усмотрение размотчика.

# 5 ОБОСНОВАНИЕ ВЫБОРА СРЕДСТВ ПРОЕКТИРОВАНИЯ И ПОСЛЕДУЮЩЕЙ РАЗРАБОТКИ ИНФОРМАЦИОННОЙ СИСТЕ-МЫ

Выбор инструментария разработки программного продукта во многом зависит от среды (операционной системы), в которой он будет использоваться.

В качестве средства проектирования выбор сделан в пользу CASE-средства верхнего уровня «BPwin», поддерживающее методологии «IDEF0» (функциональная модель), «IDEF3» (WorkFlow Diagram) и «DFD» (DataFlow Diagram). Методология «IDEF0» предписывает построение иерархической системы диаграмм –– единичных описаний фрагментов системы. Сначала проводится описание системы в целом и ее взаимодействия с окружающим миром (контекстная диаграмма), после чего проводится функциональная декомпозиция –– система разбивается на подсистемы, и каждая подсистема описывается отдельно (диаграммы декомпозиции). Затем каждая подсистема разбивается на более мелкие и так далее до достижения нужной степени подробности. Такая технология создания модели позволяет построить модель, адекватную предметной области на всех уровнях абстрагирования [8].

Стоит отметить, что приложение имеет три основных части:

– клиентская часть (Frontend), отвечает за отображение веб-страницы и придание ей интерактивности;

– серверная часть (Backend), отвечает за обработку бизнес-логики и записи результатов в базу данных;

– сервер базы данных (DataBase Server), обеспечивает многопользовательский доступ к базе данных и хранение в ней данных.

В качестве языка программирования обработки серверных сценариев был выбран PHP версии 7.1.4, с использованием потерна проектирования MVC. Средой разработки (IDE) выступает «Oracle NetBeans 8.2». Выбор был сделан в пользу данной среды разработки по причине её бесплатности, а также она имеет все необходимые инструменты для написания и отладки кода. Весом аргументом еще был, тот факт, что в данную IDE включена система контроля версий Git. Основным конкурентом данной IDE является «JetBrains PhpStorm», который также имеет все необходимые инструменты, но он был отсеян при выборе по причине его дороговизны (цены на данный программный продукт начинаются от 199 долларов США). Данные IDE имеет один существенный недостаток, который присутствует у обоих, он заключается в том, что эти инструменты очень требовательны к аппаратной части рабочего ПК. Этот недостаток не сыграл значимой роли при выборе, так как современные ПК справляются с такими нагрузками, а также такой недостаток свойствен практически всем профессиональным инструментам [9].

Выбор обоснован тем, что язык «PHP», тем что он наиболее подходит для разработки простых веб-приложений и имеет не сложный синтаксис для изучения. Также он имеет конкурентное преимущество над такими платформами как «C# ASP .NET» и «Java Enterprise Edition», так как он и все программные серверы, необходимые для интерпретации кода, являются OpenSource продуктами и могут работать под управлением свободно распространяемых операционных системах, таких как Linux, BSD, Unix, что в свою очередь отражается на стоимости хостинга и обеспечивает финансовую выгоду компании.

Для удобной организации данных был выбран паттерн проектирования и структурирования данных Model–View–Controller.

Model–View–Controller (MVC, «Модель–Представление–Контроллер», «Модель–Вид–Контроллер») — схема разделения данных приложения, пользовательского интерфейса и управляющей логики на три отдельных компонента: модель, представление и контроллер — таким образом, что модификация каждого компонента может осуществляться независимо:

Модель (Model) предоставляет данные и реагирует на команды контроллера, изменяя своё состояние;

Представление (View) отвечает за отображение данных модели пользователю, реагируя на изменения модели;

Контроллер (Controller) интерпретирует действия пользователя, оповещая модель о необходимости изменений.

Схематичное представление работы MVC изображено на рисунке 5.1.



Рисунок 5.1 –– Схема работы MVC.

Стоит отметить, что модель работает с базой данных с помощью системы подготовленных запросов PDO. Эта технология позволяет в сочетании с MVC Route (подсистема адресации и организации ссылок) позволяет защитить сайт от SQL-инъекций. Таким образом обеспечивается защита от передачи в СУБД вредоносных SQL-запросов и всех вытекающих последствий, таких как попадание лишних данных, несанкционированное получение и подменна данных, перегрузка сервера базы данных и многое другое. Данная технология требует подключение в конфигурационном файле интерпретатора PHP модуля драйвера выбранной СУБД [10].

Следующим компонентом системы является клиентская часть, которая разлизана на Twitter Bootstrap 3 Framework и популярной библиотеки jQuery 3. Также в проекты был добавлен бесплатный сборник векторных иконок Font Awesome, который представляют собой набор шрифтов и CSS-классов [11].

Последней частью разрабатываемой системы является сервер базы данных и СУБД, в качестве данного компонента на этапе разработки использовался «Oracle MySQL» по причине того, что эта СУБД способна работать с большим объемом данных, является полностью бесплатным, а также идеально подходит при работе с использованием PDO. На этапе внедрения MySQL можно заменить на MariaDB, эти СУБД имеет схожую архитектуру и общее ядро, поэтому в 99% случаев являются взаимозаменяемыми [12].

В следствие можно сделать вывод, что был выбран полностью бесплатный, что позволит снизить себестоимость разрабатываемого программного продукта.

# 6 ИНФОРМАЦИОННАЯ МОДЕЛЬ ПРЕДМЕТНОЙ ОБЛАСТИ И СТРУКТУРА БАЗЫ ДАННЫХ

На данном этапе курсового проекта необходимо разработать информационную модель предметной области, а также модель структуры базы

## 6.1 IDEF0 модель

Взаимодействие работ между собой и внешним миром описывается с помощью модели Business Process (IDEF0-диаграммы).

IDEF0 –– методология функционального моделирования. С помощью наглядного графического языка IDEF0, изучаемая система предстает перед разработчиками и аналитиками в виде набора взаимосвязанных функций (функциональных блоков - в терминах IDEF0). Как правило, моделирование средствами IDEF0 является первым этапом изучения любой системы [13].

Графический язык IDEF0 удивительно прост и гармоничен. В основе методологии лежат четыре основных понятия.

Первым из них является понятие функционального блока (Activity Box). Функциональный блок графически изображается в виде прямоугольника и олицетворяет собой некоторую конкретную функцию в рамках рассматриваемой системы.

Каждая из четырех сторон функционального блока имеет своё определенное значение (роль), при этом:

– верхняя сторона имеет значение “Управление” (Control);

– левая сторона имеет значение “Вход” (Input);

– правая сторона имеет значение “Выход” (Output);

– нижняя сторона имеет значение “Механизм” (Mechanism) [14].

Контекстная (или функциональная) диаграмма является вершиной древовидной структуры диаграмм и представляет собой самое общее описание системы и ее взаимодействия с внешней средой. В данном курсовом проекте главным функциональным блокам является разрабатываемый программный модуль оценки качества обслуживания процессов банка аутсорсингвой компании. Он представлена на рисунке 6.1.

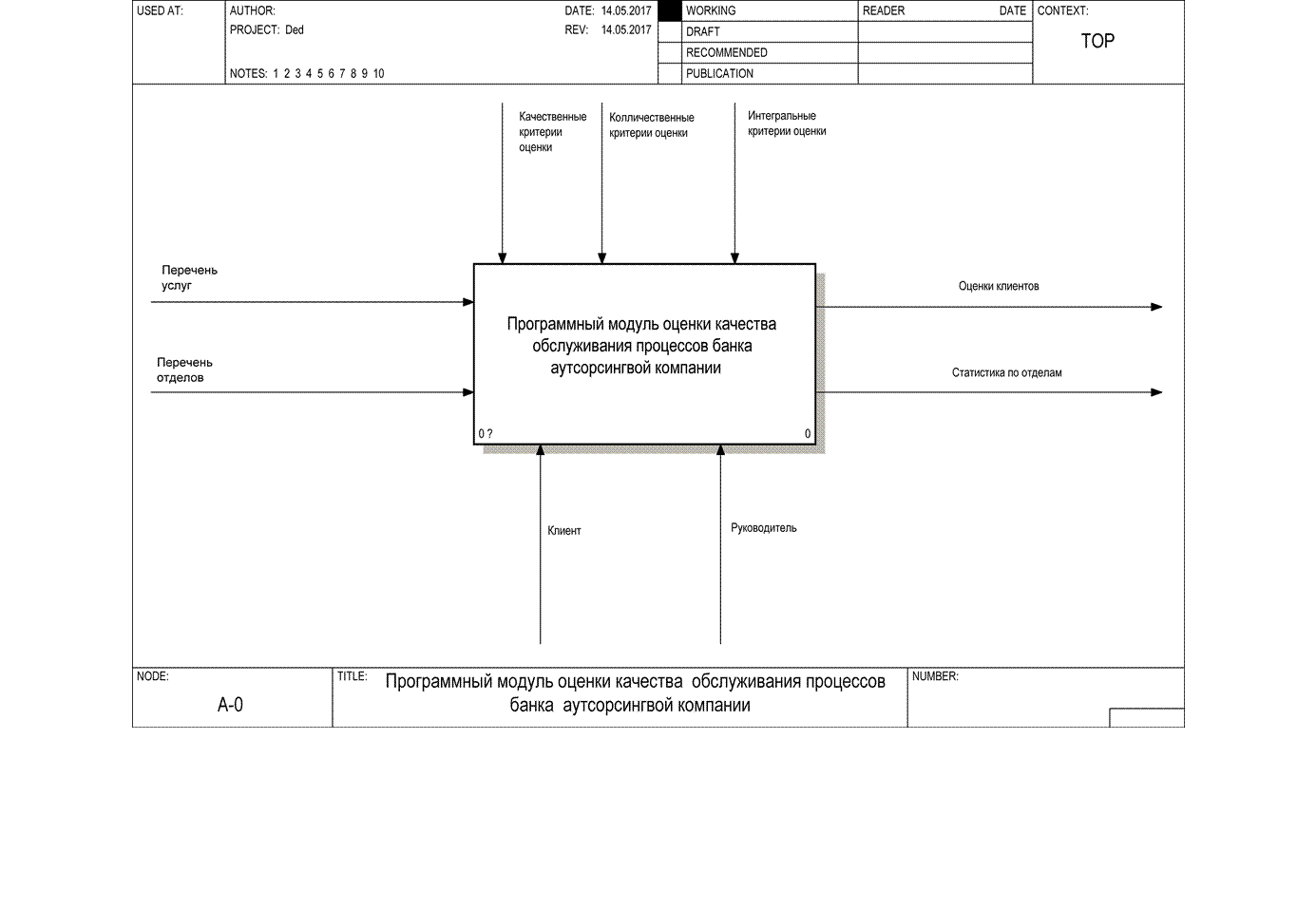


Рисунок 6.1 –– Функциональная схема программный модуль оценки качества обслуживания процессов банка аутсорсингвой компании

В данной системе входными параметрами являются перечень услуг и отделов. В качестве действующих лиц выступает клиент и руководитель или отдел менеджмента и контроля качества. Управляющими документами и нормативными актами являются критерии оценки качества предоставляемых услуг. Выходными данными являются непосредственно оценки критериев. Далее производится декомпозиция общей функциональной схемы на отдельные функции. Общая схема декомпозиции первого уровня представлена на рисунке 6.2.

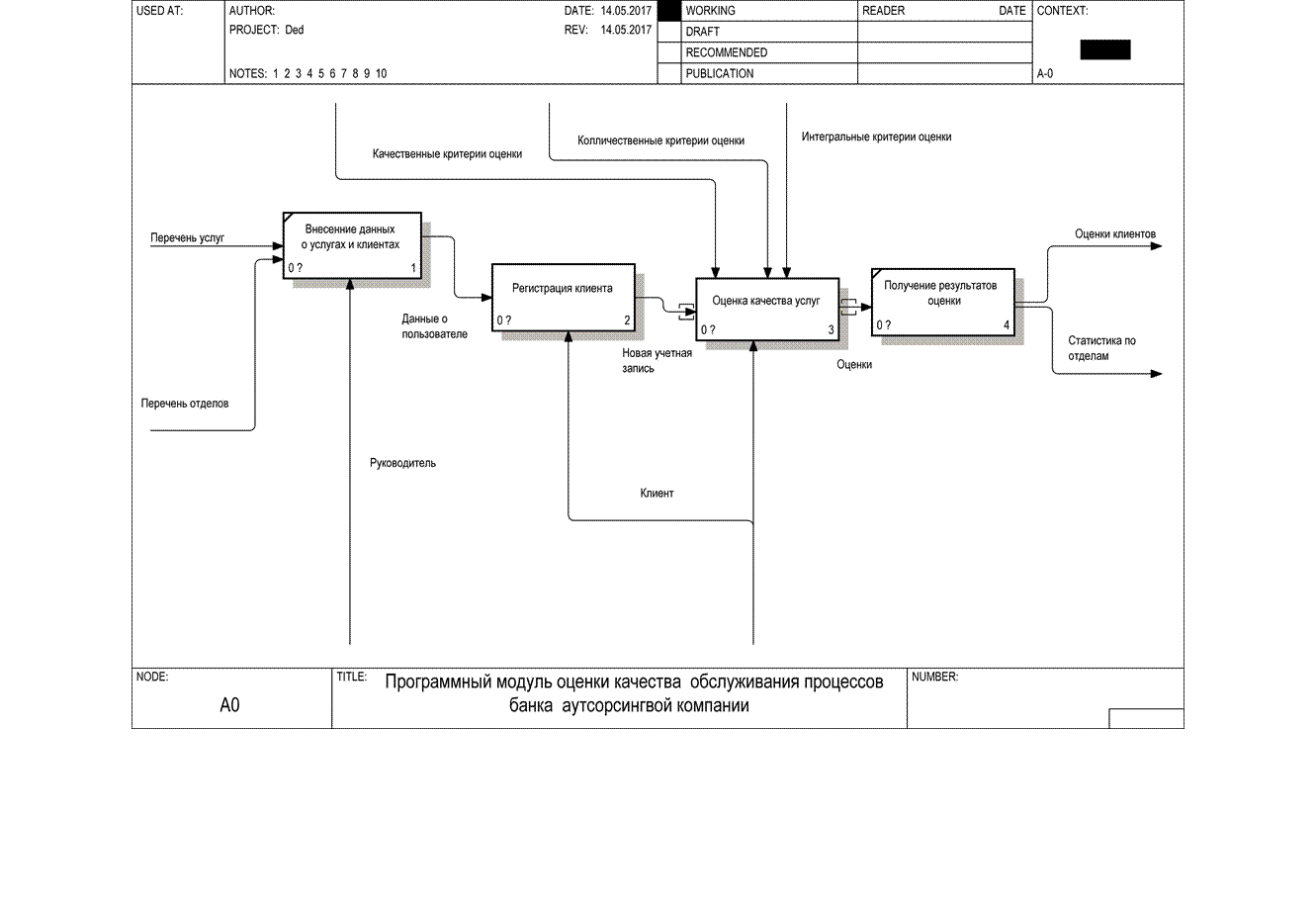


Рисунок 6.2 –– Общая схема декомпозиции процесса оценки качества аутсорсинговых услуг

На данной схеме видно, что процесс оценки качества услуг делиться на четыре основных этапа:

–– внесение перечня услуг и отделов аутсорсиноговой компании;

–– регистрация клиента;

–– оценка качества услуг;

–– получение результатов оценки.

Каждый из этих этапов имеет более низкоуровневую декомпозицию.

Так как процесс внесения данных о услугах и отделах принципиально прост и понятен, то не имеет смысла делать его полную декомпозицию. Процесс регистрации пользователя имеет более сложную ступенчитую структуры и поэтому имеет декомпозицию второго уровня. Данная декомпозиция изображена на рисунке 6.3.

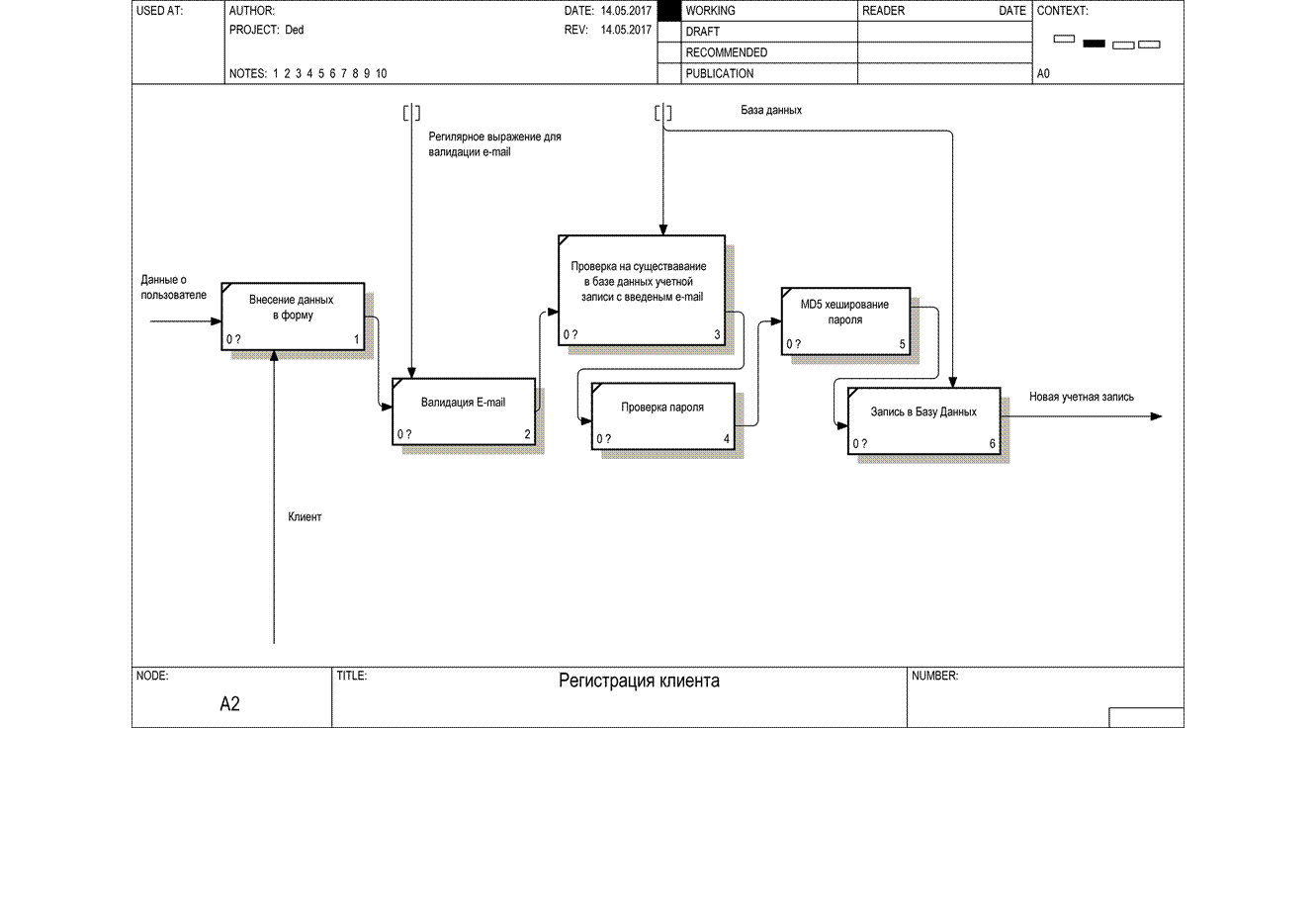


Рисунок 6.3 –– Декомпозиция процесса регистрации пользователя

На данной схеме видно, что в начале процесса регистрации перед пользователем отрывается форма ввода учетных данных. После ввода и подтверждения отправки данных, последние проходят процесс валидации. Валидация –– это процесс, подтверждающий, что введенные данные соответствуют заданным критериям. Сперва сверяется e-mail согласно регулярному выражению, согласно которому в строке электронной почты должен содержаться символ «@» и символ точки в домене почтового провайдера. Далее проверяется наличии пользователя в базе данных с таким адресом электронной почты. Он не должен присутствовать, так как адрес является уникальным идентификатором пользователя. Далее проверяется пароль и если он удовлетворяет условиям проверки, «хэшируется» с помощью алгоритма MD5 и сохраняется в базу данных.

Последней декомпозицией является процесс оценки качества услуг. На рисунке 6.4 изображена декомпозиция данного процесса.

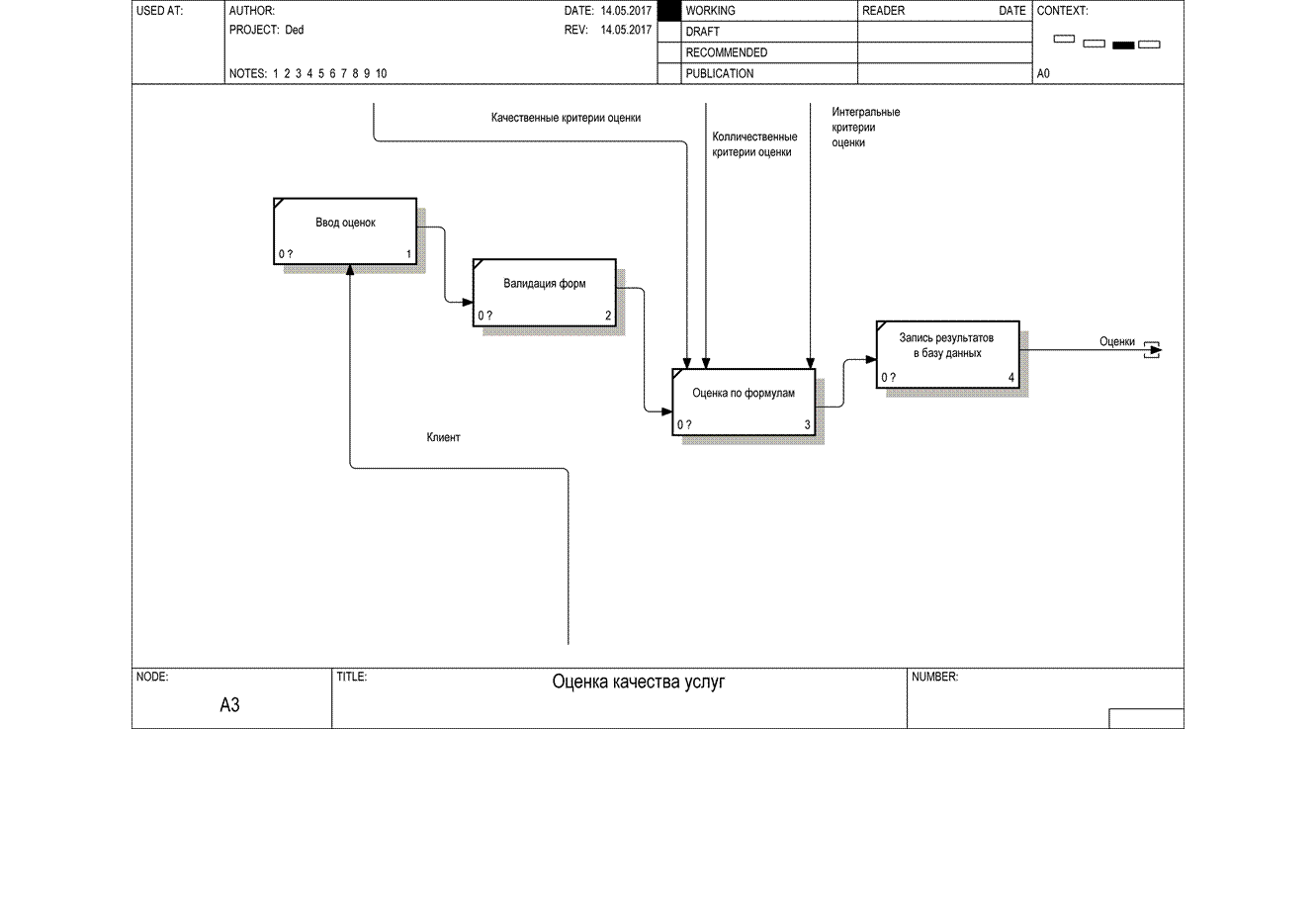


Рисунок 6.4 –– Декомпозиция процесса оценки качества услуг

Из этой схемы видно, что пользователь обязан оценить каждую услугу по 5-ти балльной шкале. Если пользователь не оценит какую-либо услугу, то данные не пройдут процесс валидации и результат не будет сохранен, а пользователь получить соответствующее уведомление на экране. В противном случае данные будут записаны в базу данных и могут быть использованы в дальнейшем для получения статистики.

## Диаграмма классов

Для удобства совместной разработки паттерн MVC предполагает четкое разбиение приложений на классы, каждый контроллер и модель –– это отдельный класс. Согласно этому утверждению в приложении присутствует множество классов, показанных на рисунке 6.5.

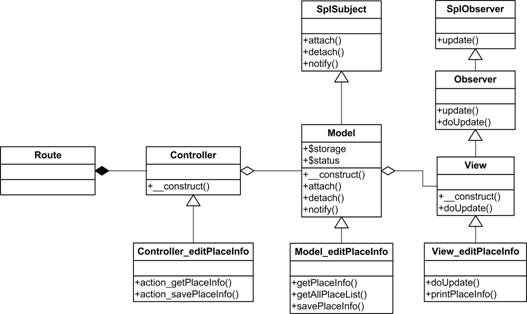


Рисунок 6.5 –– Диаграмма классов

## 6.3 Структура базы данных

Для реализации программного модуля была выбрана СУБД «Oracle MySQL». MySQL представляет собой сервер базы данных и консольное приложение для администрирования этого сервера, то есть не имеет встроенного графического интерфейса для работы с БД. Для удобства создания и редактирования таблиц было использовано стороннее ПО «Oracle MySQL WorkBench» [15]. С помощью этого инструмента была создана структура базы данных и сохранена на сервер. На рисунке 6.6 изображена схема базы данных.

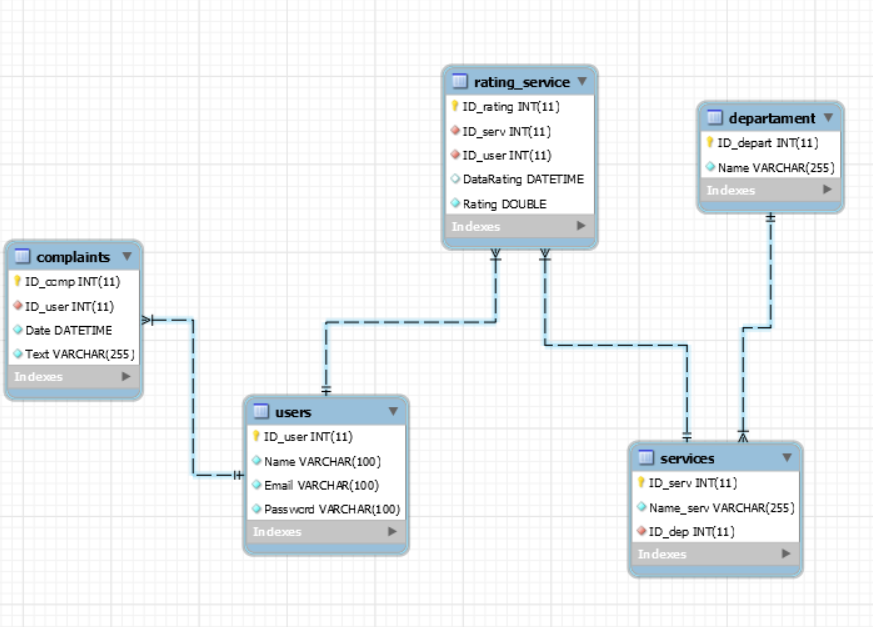


Рисунок 6.6 –– Схема базы данных

Далее будет рассмотрена каждая таблица в отдельности:

–– таблица «Users» хранит информацию о пользователе;

–– таблица «Departament» хранит список отделов;

–– таблица «Services» хранит список услуг с привязкой к отделу;

–– таблица «Complaints» –– хранит жалобы от пользователей.

–– таблица «Rating\_service» –– хранит оценки пользователей.

# 7 РАЗРАБОТКА ПРОГРАММНОГО МОДУЛЯ

Для начала нужно установить все необходимый инструментарий и библиотеки для разработки.

Перечень инструментария представлен ниже:

–– Oracle NetBeans 8.2;

–– PHP 7.1.4;

–– Oracle MySQL Server;

–– Oracle MySQL WorkBench;

–– NodeJS;

–– Git.

Далее необходимо скачать следующие Frontend библиотеки:

–– jQuery 3;

–– Twitter Bootstrap 3;

–– Font Awesome;

MVC приложение содержит структуру представленную на рисунке 7.1.

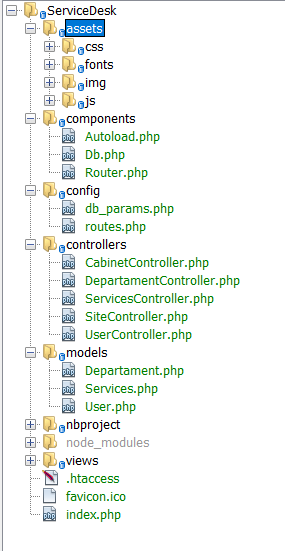


Рисунок 7.1 –– Структура MVC приложения

Ниже будет описаны основные файлы и директории MVC приложения:

–– Директория «Assets» хранит статический контент, то есть JS-скрипты, каскадные таблицы стилей, файлы изображений, шрифты и прочие статические файлы.

–– Директория «Components» хранит вспомогательные классы, например, класс маршрутизации на сайте, подключение к базе данных и тому подобное.

––Директория «Controllers» хранит серверные сценарии, отвечающие за обработку бизнес логики.

–– Директория «Models» хранит классы, отвечающие за работу с данными.

–– Директория «Views» хранит файлы представления, то есть файлы, содержавшие HTML разметку и отвечающие за внешний вид приложения.

–– Директория «Config» предназначена для хранения конфигурационных файлов приложения. В ней могут храниться, например, маршруты или учетные данные для подключения к базе данных.

–– Файл «index.php» –– точка входа в приложение.

–– Файл «Db.php» предназначен для установки соединения с базой данных.

Код для работы с базой данных представлен на рисунке 7.2.

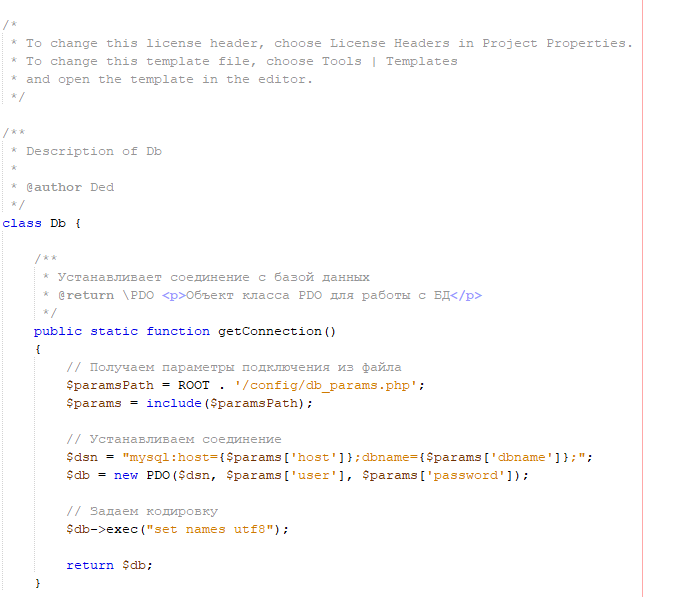


Рисунок 7.2 –– Класс для работы с базой данных с помощью технологии PDO

# 8 ВЫЯВЛЕНИЕ ПОТЕНЦИАЛЬНЫХ РИСКОВ И ПРЕДВАРИ-ТЕЛЬНАЯ ОЦЕНКА ЭФФЕКТОВ

Список главных причин провала программного проекта следующий:

1. не определена стратегия развития процесса;
2. нет планирования;
3. информация искажена / несвоевременна / отсутствует;
4. ИТ-система не обеспечивает в полном объеме потребности бизнеса;
5. отсутствует формально назначенное ответственное лицо / подразделение за результат бизнес-процесса;
6. отсутствие регламентации бизнес-процесса;
7. разногласия в экспертных оценках;
8. противоправные либо халатные действия работников в отношении компании;
9. возникновение незапланированных прямых операционных расходов.

Эффекты:

1. Отображение мнение клиентов о качестве предоставляемых услуг.
2. Выявление неэффективных отделов.
3. Выявление ошибок в предоставлении услуг.
4. Исправление ошибок при предоставлении услуг.
5. Недопущение ошибок в будущих периодах.
6. Сохранение клиентов.
7. Увеличение клиентской базы за счет улучшения качества предоставляемых услуг.
8. Увеличение прибыли.

# ЗАКЛЮЧЕНИЕ

Целью курсового проекта была создания программного модуля оценки качества обслуживания процессов банка аутсорсингвой компании.

В ходе проектирования была изучена предметная область — бизнес-процессы аутсорсинговых компаний. После самостоятельного изучения предметной области задачи курсового проекта были выполнены.

Результатом курсового проекта является полноценный функционирующий программный модуль оценки качества обслуживания процессов банка аутсорсингвой компании. В проекте реализованы следующие задачи:

1. изучена предметная область, определен объект автоматизации;
2. разработаны бизнес-требования к разрабатываемой информационной системе;
3. разработана математическая модель и алгоритмы решения;
4. разработано техническое задание;
5. обоснован выбор средств проектирования и разработки;
6. спроектирована информационная система;
7. разработан программный модуль.

Программный модуль выполнен на уровне выше среднего, не имеет недостатков в плане своей работы, запускается быстро, имеет весьма удобный интерфейс, что позволяет пользователю быстро сориентироваться в рабочей среде программы.

Разработанный программный продукт позволяет достичь такие весомые бизнес-решения, как:

– отображение мнение клиентов о качестве предоставляемых услуг;

– выявление неэффективных отделов;

– выявление ошибок в предоставлении услуг;

– исправление ошибок при предоставлении услуг;

– недопущение ошибок в будущих периодах;

– сохранение клиентов;

– увеличение клиентской базы за счет улучшения качества предоставляемых услуг.

Также в ходе выполнения работы были получены новые теоретические знания и практические навыки в области разработки и проектирования корпоративных информационных систем.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Информационный Интернет-портал [электронный ресурс]: delaybiznes.com. Режим доступа: https://goo.gl/e4HIwK – Дата доступа 02.11.2016.

2. Информационный Интернет-портал [электронный ресурс]: itfstudio.ru. Режим доступа: https://goo.gl/KfbTlZ – Дата доступа 02.11.2016.

3. Информационный Интернет-портал [электронный ресурс]: news.tut.by. Режим доступа: https://goo.gl/B0Fahl – Дата доступа 05.11.2016.

4. Информационный Интернет-портал [электронный ресурс]: maxtarget.by. Режим доступа: https://goo.gl/1IorKr – Дата доступа 05.11.2016.

5. Брайтон, М. Объектно-ориентированное конструирование программных систем/ М. Брайтон. – Спб : Русская редакция, 2005. – 111 c.

6. Боггс У., Боггс М. UML и Rational Rose: Пер. с англ. – М.: Издательство "Лори", 2000. – 78 с.

7. Информационный Интернет-портал [электронный ресурс]: www.intuit.ru. Режим доступа: https://goo.gl/s97mSX – Дата доступа 13.11.2016.

8. Информационный Интернет-портал [электронный ресурс]: swsys-web.ru. Режим доступа: https://goo.gl/64YzQ6 – Дата доступа 13.11.2016.

9. Информационный Интернет-портал [электронный ресурс]: www.cfin.ru. Режим доступа: https://goo.gl/wthxMN– Дата доступа 14.11.2016.

10. Информационный Интернет-портал [электронный ресурс]: www.intuit.ru. Режим доступа: https://goo.gl/MLXGbc – Дата доступа 15.11.2016.

11. Информационный Интернет-портал [электронный ресурс]: www.informicus.ru. Режим доступа: https://goo.gl/jHA7bp – Дата доступа 15.11.2016.

12. Информационный Интернет-портал [электронный ресурс]: www.intuit.ru. Режим доступа: https://goo.gl/phcy9T – Дата доступа 15.11.2016.

13. Информационный Интернет-портал [электронный ресурс]: www.intuit.ru. Режим доступа: https://goo.gl/0Rilgp – Дата доступа 15.11.2016.

14. Информационный Интернет-портал [электронный ресурс]: www.intuit.ru. Режим доступа: [https://goo.gl/48Fu74 – Дата доступа 19.11.2016](https://goo.gl/48Fu74%20–%20Дата%20доступа%2019.11.2016).

1. Основы современных компьютерных технологий: Учебное пособие /Под ред. проф. Хомоненко А. Д. Авторы: Артамонов Б. Н., Брякалов Г. А., Гофман В. Э., Кадироб Я. Е., Компаниец Р. И., Липецких А.Г., Мальцев М. Г., Рыжиков Ю. И., Хомоненко А.Д., Цыганков В. М.-СПб.: КОРОНА принт,2002. – 448 с.

# ПРИЛОЖЕНИЕ А

(Обязательное)

**Листинг программы**

***Код класса «Db.php»***

*class Db {*

*/\*\**

*\* Устанавливает соединение с базой данных*

*\* @return \PDO <p>Объект класса PDO для работы с БД</p>*

*\*/*

*public static function getConnection()*

*{*

*// Получаем параметры подключения из файла*

*$paramsPath = ROOT . '/config/db\_params.php';*

*$params = include($paramsPath);*

*// Устанавливаем соединение*

*$dsn = "mysql:host={$params['host']};dbname={$params['dbname']};";*

*$db = new PDO($dsn, $params['user'], $params['password']);*

*// Задаем кодировку*

*$db->exec("set names utf8");*

*return $db;*

*}*

*}*

***Код класса «Router.php»***

*<?php*

*class Router*

*{*

*private $routes;*

*public function \_\_construct()*

*{*

*$routesPath = ROOT . '/config/routes.php';*

*$this->routes = include($routesPath);*

*}*

*// Get request string*

*private function GetURI()*

*{*

*if (!empty($\_SERVER['REQUEST\_URI']))*

*{*

*return trim($\_SERVER['REQUEST\_URI'], '/');*

*}*

*}*

*public function run()*

*{*

*// Get request string*

*$uri = $this->GetURI();*

*// Find request string in file routes.php*

*foreach ($this->routes as $uriPattern => $path)*

*{*

*if (preg\_match("~$uriPattern~", $uri))*

*{*

*$segments = explode('/', $path);*

*$controllerName = array\_shift($segments) . 'Controller';*

*$controllerName = ucfirst($controllerName);*

*$actionName = 'action' . ucfirst(array\_shift($segments));*

*$parameters = $segments;*

*// Include controller file*

*$controllerFile = ROOT . '/controllers/' .*

*$controllerName . '.php';*

*if (file\_exists($controllerFile))*

*{*

*include\_once($controllerFile);*

*}*

*// Create object and call metod*

*$controllerObject = new $controllerName;*

*$result = call\_user\_func\_array(array($controllerObject,*

*$actionName), $parameters);*

*if ($result != null)*

*{*

*break;*

*}*

*}*

*}*

*}*

*}*

***Код класса «autoload.php»***

*<?php*

*function \_\_autoload($class\_name)*

*{*

*$array\_paths = array(*

*'/models/',*

*'/components/'*

*);*

*foreach ($array\_paths as $path)*

*{*

*$path = ROOT . $path . $class\_name . '.php';*

*if (is\_file($path))*

*{*

*include\_once $path;*

*}*

*}*

*}*

***Код класса «departament.php»***

*class Departament {*

*public static function getList()*

*{*

*// Соединение с БД*

*$db = Db::getConnection();*

*// Текст запроса к БД*

*$sql = 'SELECT `ID\_depart`, `Name` FROM departament;';*

*// Получение результатов. Используется подготовленный запрос*

*$result = $db->prepare($sql);*

*$result->execute();*

*return $result->fetchAll(PDO::FETCH\_ASSOC);*

*}*

*public static function add($name)*

*{*

*// Соединение с БД*

*$db = Db::getConnection();*

*// Текст запроса к БД*

*$sql = 'INSERT INTO `departament` (Name) VALUES (:name);';*

*// Получение и возврат результатов. Используется подготовленный запрос*

*$result = $db->prepare($sql);*

*$result->bindParam(':name', $name, PDO::PARAM\_STR);*

*return $result->execute();*

*}*

*}*

***Код класса «Evaluation.php»***

*class Evaluation {*

*public static function add($id, $id\_user, $rating)*

*{*

*// Соединение с БД*

*$db = Db::getConnection();*

*// Текст запроса к БД*

*$sql = 'Insert Into rating\_service (ID\_serv, ID\_user, '*

*. 'DataRating, Rating) values (:id, :id\_user, :date\_now, :rating)';*

*$db->setAttribute(PDO::ATTR\_ERRMODE, PDO::ERRMODE\_EXCEPTION);*

*// Получение и возврат результатов. Используется подготовленный запрос*

*$result = $db->prepare($sql);*

*$result->bindParam(':id', $id, PDO::PARAM\_INT);*

*$result->bindParam(':id\_user', $id\_user, PDO::PARAM\_INT);*

*$date\_now = date("Y-m-d H:i:s");*

*$result->bindParam(':date\_now', $date\_now);*

*$result->bindParam(':rating', $rating);*

*return $result->execute();*

*}*

*public static function delete($id\_user)*

*{*

*// Соединение с БД*

*$db = Db::getConnection();*

*// Текст запроса к БД*

*$sql = 'delete from rating\_service where ID\_user = :id\_user;';*

*$db->setAttribute(PDO::ATTR\_ERRMODE, PDO::ERRMODE\_EXCEPTION);*

*// Получение и возврат результатов. Используется подготовленный запрос*

*$result = $db->prepare($sql);*

*$result->bindParam(':id\_user', $id\_user, PDO::PARAM\_INT);*

*return $result->execute();*

*}*

*public static function get()*

*{*

*}*

*}*