ВВЕДЕНИЕ

Цель курсового проекта состоит в углубленном изучении студентами учебного материала по курсу «Теория информационных процессов и систем», практическом освоении объектно-ориентированной методологии проектирования с использованием Унифицированного языка моделирования (UML).

Задачами курсовой работы является формирование у студентов навыков применения языка UML, правил формирования требований к разрабатываемой информационной системе, принципов проектирования программных средств.

Для достижения поставленной выше цели, студентам предлагается по описанию предметной области построить объектно-ориентированную модель с помощью пакета MS Visio (можно также использовать любое другое CASE-средство для построения UML-диаграмм). В результате выполнения курсового проекта студенты осваивают приемы практического использования объектно-ориентированного подхода в проектировании и создании программного обеспечения информационных систем.

Результат выполнения курсового проекта оформляется в виде пояснительной записки, содержащей постановку задачи и описание этапов разработки каждой диаграммы, а также обоснование принятых решений.

1 АНАЛИЗ ПРЕДПРИЯТИЯ ООО «ИМЦ»

Специализацией компании «Информационно-медицинский центр» (ИМЦ) является информатизация здравоохранения, включая комплексные решения для медицинских организаций, органов управления здравоохранением и территориальных фондов ОМС. Автоматизируется также финансово-хозяйственная работа бюджетных учреждений, деятельность ВУЗов и библиотек.

Создана компания «ИМЦ» со специализацией в области разработки, внедрения и сопровождения медицинских информационных систем в 2005 году. До этого специалисты ИМЦ начинали свою трудовую деятельность в отделе информационных систем компании «ПАРУС» с основной специализацией - автоматизация здравоохранения и ОМС с 1994 года.

Адрес предприятия ООО «ИМЦ»: 443099, г. Самара, ул. князя Григория Засекина (Карбюраторная), д. 1, литера «З»

Руководитель предприятия ООО «ИМЦ»: директор Новиков Олег Васильевич.

ООО «ИМЦ» осуществляет разработку компьютерного программного обеспечения (62.01).

С момента организации предприятие стремительно развивается, специализируясь на разработке программных продуктов являющимся комплексными решениями для медицинских организаций, органов управления здравоохранением и территориальных фондов ОМС.

Постоянно улучшающий практический опыт разработки, внедрения и сопровождения информационных систем, хорошее знание особенностей действующих нормативных документов, положений, стандартов и технологий.

ООО «ИМЦ» имеет возможность адаптировать систему под нужды организации, их эволюционное развитие в зависимости от совершенствования технической инфраструктуры.

Основная информация об ООО «ИМЦ» представлена в таблице 1.

Таблица 1 – Общие сведения о ООО «ИМЦ»

|  |  |
| --- | --- |
| Наименование: | ООО «ИМЦ» |
| Основной ОКВЭД: | Разработка компьютерного программного обеспечения |
| Страна: | РОССИЯ |
| Регион: | Самарская область |
| ИНН: | 6317059075 |
| ОКПО или др.: | 78250264 |
| Данные госрегистрации: | Межрайонная инспекция ФНС России № 18 по Самарской области Дата постановки на учет: 20 июня 2005 г. |
| Юридический адрес: | 443099, г. Самара, ул. князя Григория Засекина (Карбюраторная), д. 1, литера «З» |

Целью предприятия ООО «ИМЦ» является Разработка компьютерного программного обеспечения для выполнения требований потребителя.

Задачами предприятия ООО «ИМЦ» являются:

- получение дохода владельцам ООО «ИМЦ»;

- обеспечение потребителей программным обеспечением в соответствии с договорами;

- обеспечение персонала предприятия заработной платой, нормальными условиями труда и возможностью профессионального роста;

- создание рабочих мест для населения, в пределах муниципального округа.

К основному виду деятельности ООО «ИМЦ» относится: Разработка компьютерного программного обеспечения.

К дополнительным видам деятельности ООО «ИМЦ» относятся:

- торговля оптовая компьютерами, периферийными устройствами к компьютерам и программным обеспечением;

- торговля оптовая неспециализированная;

- торговля розничная компьютерами, периферийными устройствами к ним и программным обеспечением в специализированных магазинах;

- деятельность консультативная и работы в области компьютерных технологий;

- деятельность, связанная с использованием вычислительной техники и информационных технологий, прочая;

- деятельность по обработке данных, предоставление услуг по размещению информации и связанная с этим деятельность;

- деятельность по созданию и использованию баз данных и информационных ресурсов;

- деятельность по оказанию консультационных и информационных услуг;

- научные исследования и разработки в области естественных и технических наук прочие;

- деятельность по предоставлению прочих вспомогательных услуг для бизнеса, не включенная в другие группировки;

- ремонт компьютеров и периферийного компьютерного оборудования.

1.1 Задачи и функции информационной системы

Информационная система будет разработана для информационно-технической деятельности отдела техническая поддержка ООО «ИМЦ» для оптимизации процесса – обработка обращении от клиентов.

Информационная система должна решать задачу систематизировать поступающие обращение от клиентов и оптимизировать процесс их заполнения.

Входными данными информационной системы является клиент и обращение клиентов.

Выходные данные информационной системы является проделанные работы исполнителя об выполненных работ, описанных в обращение клиента;

Сущности информационной системы:

* клиент;
* исполнитель;
* обращение(задача);
* организация клиента.

Информационная система выполняет следующие функции:

* оформление обращении клиентов об какой-либо ошибке или доработке системы;
* добавление клиента;
* получение списка клиентов;
* получение информации об клиенте;
* добавление исполнителя;
* редактирование исполнителя;
* получение списка исполнителей;
* добавление организации клиента;
* получение списка организации клиентов;
* редактирование организации;
* возможность авторизации клиента и исполнителя;
* возможность аутентификация клиента и исполнителя;
* возможность просмотра всех обращении;
* создание обращения;
* изменение статуса обращения;
* фильтрация по обращениям;
* изменение обращения;
* валидация изменения статусов обращения.

1.2 Выбор методологии моделирования

Сущность структурного подхода к разработке ИС заключается в ее декомпозиции (разбиении) на автоматизируемые функции: система разбивается на функциональные подсистемы, которые в свою очередь делятся на подфункции, подразделяемые на задачи и так далее.

В качестве средств структурного анализа и проектирования, наиболее распространены следующие нотации:

* SADT. применяется для определения требований (функций) для разработки системы, реализующей выделенные функции. Для уже существующих - IDEF0 может быть использована для анализа функций, выполняемых системой. Модель в нотации IDEF0 представляет собой совокупность иерархически упорядоченных и взаимосвязанных диаграмм;
* DFD (Data Flow Diagrams) диаграммы потоков данных. Диаграммы DFD обычно строятся для наглядного изображения текущей работы системы документооборота организации. Как правило, диаграммы DFD используют в качестве дополнения модели бизнес-процессов, выполненной в IDEF0;
* IDEF3. Методология моделирования IDEF3 позволяет описать процессы, фокусируя внимание на течении этих процессов, позволяет рассмотреть конкретный процесс с учетом последовательности выполняемых операций;
* ER (Entity-Relationship Diagrams) диаграммы "сущность-связь". Методология описания данных (IDEF1X).

Общие выводы: применение универсальных графических языков моделирования IDEF0, IDEF3 и DFD обеспечивает логическую целостность и полноту описания, необходимую для достижения точных и непротиворечивых результатов на этапе анализа.

Принципиальное различие между структурным и объектно-ориентированным (ОО) подходом заключается в способе декомпозиции системы. ОО подход использует объектную декомпозицию, при этом статическая структура системы описывается в терминах объектов и связей между ними, а поведение системы описывается в терминах обмена сообщений между объектами.

UML предоставляет средства для создания визуальных моделей, которые единообразно понимаются всеми разработчиками, вовлеченными в проект, и являются средством коммуникации в рамках проекта. Диаграмма в UML - это графическое представление набора элементов. Диаграммы рисуют для визуализации системы с разных точек зрения. При визуальном моделировании на UML используются восемь видов диаграмм, каждая из которых может содержать элементы определенного типа.

Выбор объектно-ориентированного подхода вместо структурного обоснован следующими причинами:

* объектно-ориентированные системы лучше моделируют предметную область. Они проще адаптируются к изменяющимся условиям, легче изменяются, устойчивее и позволяют создавать более крупные проекты;
* объектная декомпозиция уменьшает размер программных систем. Это достигается за счёт повторного использования общих механизмов, что приводит к существенной экономии выразительных средств;
* объектно-ориентированные системы снижают риск при создании сложной программной системы. Она развивается из меньших систем, в которых уже уверены;
* использование объектного подхода повышает уровень унификации разработки и пригодность для повторного использования.

В данной главе был проведен анализ предметной области, определены задачи и цели компании, определена задача и функции разрабатываемой информационной системы, произведено изучение и сравнение структурного подхода и объектно-ориентированного подхода и определены причины выбора объектно-ориентированного подхода.

2 РАЗРАБОТКА UML-ДИАГРАММ

Отличительная особенность UML — это возможность напрямую связать модели с языками программирования, благодаря чему нотацию можно рассматривать в качестве верхнеуровневого инструмента разработки. UML можно рассматривать как наследницу идей объектно-ориентированного анализа и проектирования.

Ключевые понятия объектно-ориентированного подхода к проектированию:

* объект — простейшая сущность, базовый строительный блок.
* класс — чертеж объекта, его условное описание.
* абстракция — отражение поведения сущности в реальном мире.
* инкапсуляция — механизм связывание данных и их сокрытия от внешнего мира.
* Наследование — механизм получения новых классов на основе уже существующих
* Полиморфизм — механизм образования новых форм из существующих элементов.

В UML объекты содержат данные и методы их контроля. Данные описывают состояние объекта. Классы описывают объекты и образуют иерархию, которая отражает реально существующую систему. Объекты — это сущности реального мира, и UML использует для их отображения такие методы, как абстракция, инкапсуляция, наследование и полиморфизм. Таким образом диаграммы UML по сути являются объектно-ориентированным представлением.

2.1 Диаграмма вариантов использования

Диаграмма вариантов использования (use case diagram) — это диаграмма, на которой изображаются отношения между актерами и вариантами использования.

В терминологии UML, актёр обозначает любые сущности, использующие систему. Этими сущностями могут быть люди, технические устройства или даже другие системы.

Каждая группа пользователей на диаграмме вариантов использования обозначается человечком, под которым записывается имя группы людей, которую он обозначает.

На диаграмме вариантов использования функция системы изображается эллипсом, внутри которого записывается имя функции в форме глагола с пояснительными словами.

Связи между элементами:

* ассоциация – это связь между двумя классификаторами, такими как действующее лицо и варианты использования, которая описывает причину связи и управляющие ею правила.
* обобщения - означает, что некоторый актёр (вариант использования) может быть обобщён до другого актёра (варианта использования). Стрелка направлена от частного случая(специализации) к общему случаю;
* расширения - определяет взаимосвязь одного варианта использования с некоторым другим вариантом использования, функциональность или поведение которого задействуется первым не всегда, а только при выполнении некоторых дополнительных условий;
* включения – в диаграмме вариантов использования специфицирует тот факт, что некоторый вариант использования содержит поведение, определённое в другом варианте использования.

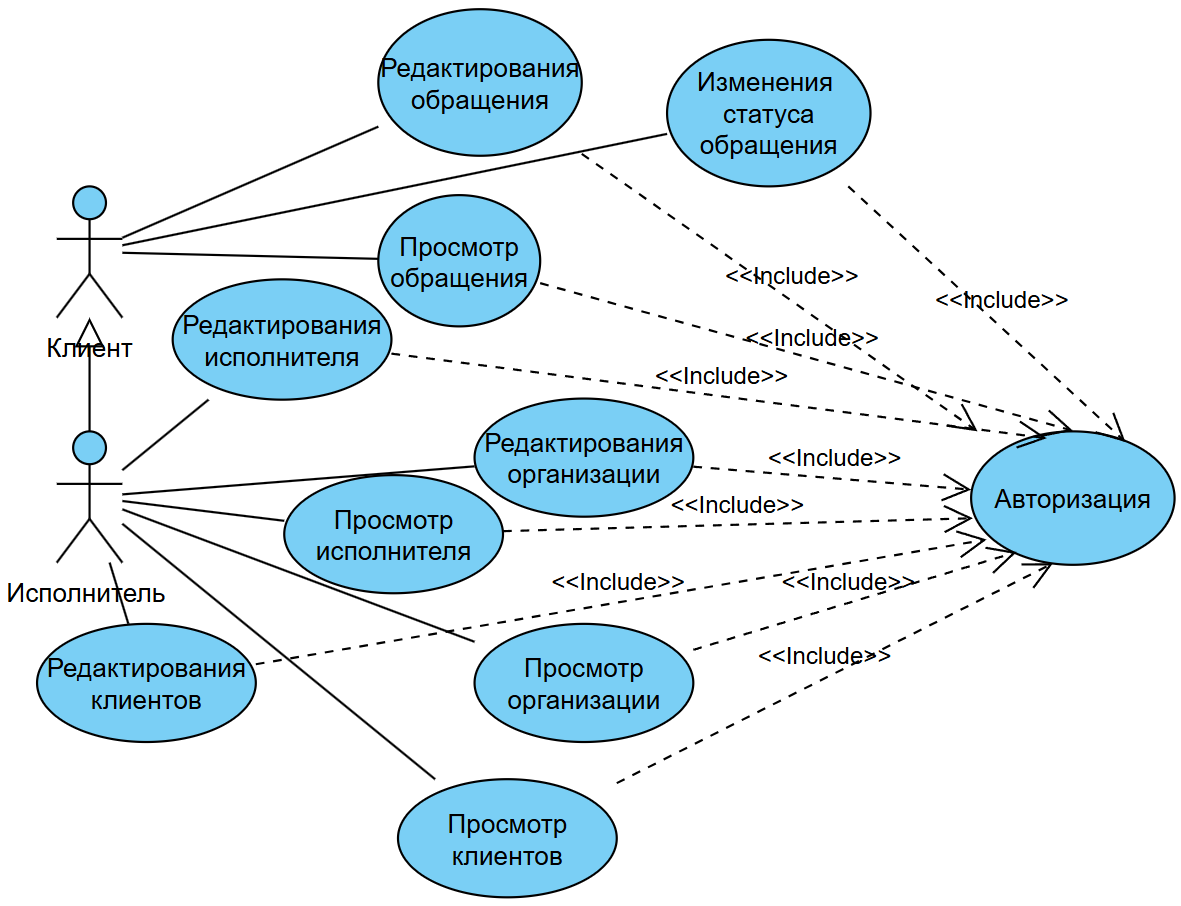
Ниже на рисунке 1 представлена диаграмма вариантов использования. 

Рисунок 1 - Диаграмма вариантов использования.

Ниже в таблице 2 представлено множество действующих лиц (актеров).

Таблица 2 - Множество действующих лиц (актеров).

|  |  |  |
| --- | --- | --- |
| № п.п. | Имя | Описание |
| 1 | клиент | пользователь, который имеет доступ к технической поддержки, оформляют обращение на доработку функционала. |
| 2 | исполнитель | сотрудник, который выполняет обращение клиентов и актуализирует информацию клиентов, организации и исполнителей. |

Ниже в таблице 3 представлено множество вариантов использования.

Таблица 3 - Множество вариантов использования

|  |  |  |  |
| --- | --- | --- | --- |
| № п.п | Имя | Описание | Точки расширения |
| 1 | Авторизация | Ввод логина и пароля |  |
| 2 | Редактирование обращения | Создание или изменения обращения |  |
| 3 | Изменение статуса обращения | Смена статуса обращения |  |
| 4 | Просмотр обращения | Просмотр и поиск обращении |  |
| 5 | Просмотр исполнителя | Просмотр и поиск исполнителей |  |
| 6 | Редактирование исполнителя | Создать или изменить информацию о исполнителе |  |
| 7 | Просмотр клиентов | Просмотр и поиск клиентов |  |
| 8 | Редактирование клиента | Создать или изменить информацию о клиента |  |
| 9 | Просмотр организации | Просмотр и поиск организации |  |
| 10 | Редактирование организации | Создание или изменения организации |  |

2.2 Диаграмма последовательности

Диаграмма последовательности — это UML-диаграмма, на которой для некоторого набора объектов на единой временной оси показан жизненный цикл объекта (создание, деятельность, уничтожение) и взаимодействие акторов (действующих лиц) информационной системы в рамках прецедента.

Основные элементы диаграммы последовательности:

* обозначения объектов (прямоугольники с названиями объектов);
* вертикальные «линии жизни», отображающие течение времени;
* прямоугольники, отражающие деятельность объекта или исполнение им определенной функции;
* стрелки, показывающие обмен сигналами или сообщениями между объектами.

На рисунке 2 представлена диаграмма последовательности.

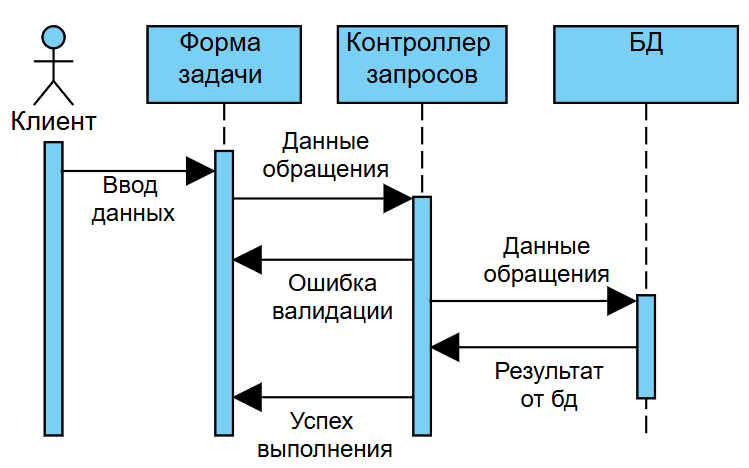


Рисунок 2 - Диаграмма последовательности

2.3 Диаграмма классов

Диаграмма классов (class diagram) — это структурная диаграмма языка моделирования UML, демонстрирующая общую структуру иерархии классов системы, их коопераций, атрибутов (полей), методов, интерфейсов и взаимосвязей (отношений) между ними.

Основными элементами являются классы и связи между ними. Классы характеризуются при помощи атрибутов и операций.

Атрибуты описывают свойства объектов класса. Большинство объектов в классе получают свою индивидуальность из-за различий в их атрибутах и взаимосвязи с другими объектами. Однако, возможны объекты с идентичными значениями атрибутов и взаимосвязей. Т.е. индивидуальность объектов определяется самим фактом их существования, а не различиями в их свойствах. Имя атрибута должно быть уникально в пределах класса.

Операция есть функция или преобразование. Операция может иметь параметры и возвращать значения.

Ниже на рисунке 3 представлена диаграмма классов.

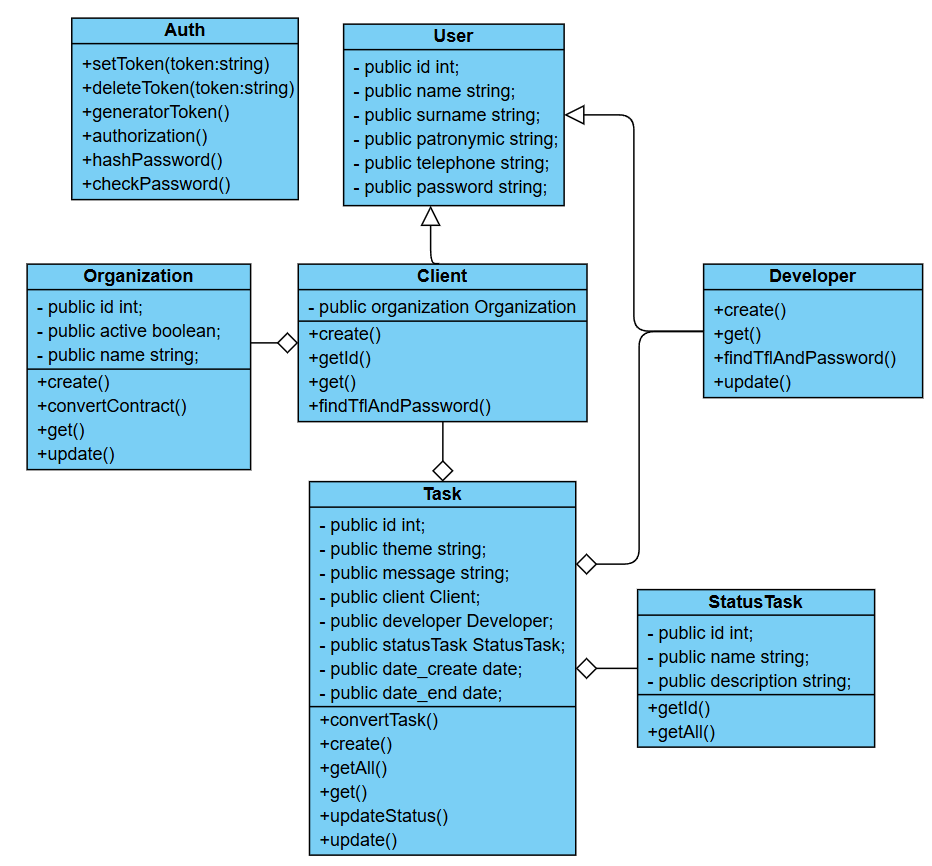


Рисунок 3 - Диаграмма классов

Ниже в таблице 4 представлено описание классов.

Таблица 4 - Описание классов

|  |  |  |
| --- | --- | --- |
| № п/п | Имя класса | Краткое описание |
| 1 | User | Обобщенный класс пользователей для наследования. |
| 2 | Client | Класс представляющий обработку клиентов. |
| 3 | Developer | Класс представляющий обработку исполнителей. |
| 4 | Task | Класс представляющий обработку задач. |
| 5 | StatusTask | Класс представляющий обработку статусов задачи. |
| 6 | Organization | Класс представляющий обработку организации клиентов |
| 7 | Auth | Класс представляющий функционал авторизации |

Ниже в таблице 5 представлено описание атрибутов классов.

Таблица 5 - Описание атрибутов классов

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| User | | | | |
| Атрибут | Видимость | Тип атрибута | Множественность | Описание |
| id | public | int | - | Первичный ключ |
| name | public | string | - | Имя пользователя |
| surname | public | string | - | Фамилия пользователя |
| patronymic | public | string | - | Отчество пользователя |
| telephone | public | string | - | Номер телефона |
| password | public | string | - | Пароль пользователя |
| Client | | | | |
| organization | public | Organization | + | Класс организации |
| Organization | | | | |
| id | public | int | - | Первичный ключ |
| active | public | boolean | - | Активность поддержки организации |
| name | public | string | - | Название организации |
| StatusTask | | | | |
| id | public | int | - | Первичный ключ |
| name | public | string | - | Название статуса |
| description | public | string | - | Описание статуса |
| Task | | | | |
| id | public | int | - | Первичный ключ |
| theme | public | string | - | Тема |

Продолжение таблицы 5

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| message | public | string | - | Сообщение |
| client | public | Client | - | Класс клиент |
| developer | public | Developer | - | Класс исполнитель |
| date\_create | public | date | - | Дата создание |
| date\_end | public | date | - | Дата завершения |

Ниже в таблице 6 представлены методы классов

Таблица 6 - Методы классов

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Client | | | | | | |
| Операция | Видимость | Параметры | | | Тип возвращаемого  значения | Описание |
| Имя | | Тип |
| create | public | сlientData | | Сlient | void | Метод создания нового клиента |
| getId | public | Id | | Integer | Сlient | Получить клиента по id |
| get | public |  | |  | Сlient[] | Получить список клиента |
| update | public | сlientData | | Сlient | void | Изменить клиента |
| Developer | | | | | | |
| create | public | developer | Developer | | void | Метод создания  нового исполнителя |
| get | public |  |  | | Developer [] | Получить список  исполнителя |
| update | public | developer | Developer | | void | Изменить  Исполнителя |
| Organization | | | | | | |
| create | public | organization | Organization | | void | Метод создания  нового исполнителя |

Продолжение таблицы 6

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| get | public |  |  | Organization [] | Получить список  организации |
| update | public | organization | Organization | void | Изменить  Организацию |
| StatusTask | | | | | |
| get | public |  |  | StatusTask[] | Получить список  статусов задачи |
| getId | public |  |  | StatusTask | Получить статус задачи по id |
| Auth | | | | | |
| setToken | public | token | string | string | Сохранение токена |
| deleteToken | public | token | string | string | Удаление токена |
| generatorToken | public |  |  | string | Генерация токена |
| authorization | public | login  password | string  string | void | Авторизация пользователя |
| Task | | | | | |
| create | public | task | Task | void |  |
| getAll | public | filter | Filter | Task[] | Получить задачи для исполнителя |
| get | public | filter | Filter | Task[] | Получить задачи для клиента |
| updateStatus | public | id  idStatus | integer  integer | void | Изменение статуса  задачи |
| update | public | task | Task | void | Изменение задачи |

2.4 Диаграмма компонентов

Диаграмма компонентов - это структурная диаграмма языка унифицированного моделирования, она описывает особенности физического представления системы. Диаграмма компонентов позволяет определить архитектуру разрабатываемой системы, установив зависимости между программными компонентами.

Диаграмма компонентов предоставляет общую картину архитектуры системы, помогает разработчикам и архитекторам лучше понять ее структуру и взаимосвязи, а также является полезным инструментом для коммуникации и документирования архитектурных решений.

* диаграмма компонентов разрабатывается для следующих целей:
* визуализация общей структуры исходного кода программной системы;
* спецификация исполнимого варианта программной системы;
* обеспечение многократного использования отдельных фрагментов программного кода;
* представление концептуальной и физической схем баз данных.

Компонент представляет отдельную часть системы, которая выполняет определенную функцию или имеет определенную роль. Он может быть программным модулем, классом, библиотекой, сервисом, физическим устройством и т. д. Он обычно отображается в виде прямоугольника с его именем или обозначением.

На рисунке 4 ниже представлена диаграмма компонентов.

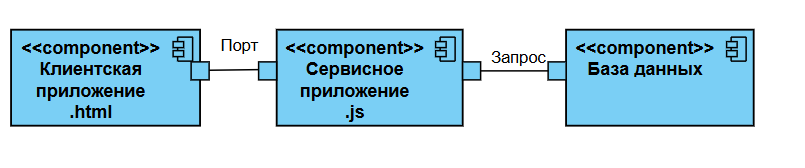


Рисунок 4 - Диаграмма компонентов

2.5 Диаграмма развертывания

Диаграмма развертывания (deployment diagram) предназначена для представления общей конфигурации или топологии распределенной программной системы и содержит изображение размещения различных артефактов по отдельным узлам системы. Диаграммы развертывания используются в разных целях.

Диаграмма развертывания предназначена для визуализации элементов и компонентов программы, существующих лишь на этапе ее исполнения (runtime). При этом представляются только компоненты-экземпляры программы, являющиеся исполнимыми файлами или динамическими библиотеками. Те компоненты, которые не используются на этапе исполнения, на диаграмме развертывания не показываются.

Например, с их помощью можно:

* наглядно показать, какие программные элементы развертываются на тех или иных аппаратных компонентах;
* проиллюстрировать обработку процессов исполнения аппаратными компонентами;
* провести обзор топологии аппаратного комплекса.

На рисунке 5 представлена диаграмма развертывания.

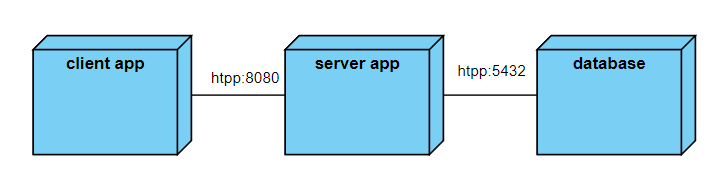


Рисунок 5 - Диаграмма развертывания

ЗАКЛЮЧЕНИЕ

В ходе выполнение курсового проекта была достигнута цель изучения учебного материала по курсу «Теория информационных процессов и систем», освоено объектно-ориентированной методологии проектирования с использованием Унифицированного языка моделирования (UML).

Выполнены следующие задачи курсовой работы:

* приобретён навыков применения языка UML, правил формирования требований к разрабатываемой информационной системе, принципов проектирования программных средств;
* выполнен анализ предприятия ООО «ИМЦ»;
* определены задачи и функции информационной системы;
* проведен выбор методологии моделирования разрабатываемой информационной системы;
* разработана диаграмма вариантов использования;
* разработана диаграмма последовательности;
* разработана диаграмма классов;
* разработана диаграмма компонентов;
* разработана диаграмма развертывания.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Романов, П.С. Математические основы теории систем. Практикум: учебное пособие / П.С. Романов, И.П. Романова. — Санкт-Петербург: Лань, 2021. — 172 с.

2. Флегонтов, А.В. Моделирование информационных систем. Unified Modeling Language : учебное пособие / А.В. Флегонтов, И.Ю. Матюшичев. — 2-е изд., стер. — Санкт-Петербург : Лань, 2019. — 112 с.

3. Певзнер Л.Д. Практикум по математическим основам теории систем. [Электронный ресурс] / Л.Д. Певзнер - Москва: Лань, 2023. – 400с.

4. Юмагулов М.Г. Введение в теорию динамических систем [Электронный ресурс] / М.Г. Юмагулов - Москва: Лань, 2021. – 272с.

5. Качала В.В. Основы теории систем и системного анализа: Учебное пособие для вузов - 2-е изд.,испр. - ("Учебное пособие для высших учебных заведений-Специальность") (ГРИФ) Гор. линия-Телеком, 2022

6. Волкова В.Н. Теория информационных процессов и систем: Учебник и практикум для академического бакалавриата - ("Бакалавр. Академический курс") Юрайт, 2020

7. Вдовин В.М. Теория систем и системный анализ: Учебник / В.М. Вдовин, Л.Е. Суркова, В.А. Валентинов. – М: Дашков и К, 2024. – 640с.

8. Советов Б.Я. Теория информационных процессов и систем: учебник для студентов высших учебных заведений / Б.Я. Советов, В.А. Дубенецкий, В.В. Цехановский и др. – М.: Издательский центр «Академия», 2020. – 432с.