Учреждение образования

«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

**Отчёт**

по предмету «Проектирование программного обеспечения»

Лабораторная работа №7

«Проектирование информационных систем»

Студент: Лопатнюк П.В.

ФИТ 3 курс 1 группа

Преподаватель: Якубенко К. Д

Минск 2024

# Теоретические сведения

Диаграммы поведения в UML используются для моделирования динамических аспектов системы, показывая, как она функционирует во времени. К основным видам таких диаграмм относятся:

* Диаграмма случаев использования (Use Case Diagram): Показывает взаимодействие пользователей с системой через функциональные сценарии. Она отображает, какие функции доступны пользователям и как они их используют, но не детализирует внутреннюю реализацию.
* Диаграмма последовательностей (Sequence Diagram): Описывает порядок взаимодействий между объектами в определённом сценарии. Особое внимание уделяется времени, демонстрируя, в каком порядке объекты обмениваются сообщениями и как это влияет на их состояние.
* Диаграмма коммуникаций (Communication Diagram): Фокусируется на структурных связях между объектами, участвующими в обмене сообщениями. В отличие от диаграммы последовательностей, она акцентирует внимание на том, какие объекты взаимодействуют, а не на временном порядке их взаимодействий.
* Диаграмма состояний (State Machine Diagram): Показывает жизненный цикл объекта через его состояния и переходы между ними. Это полезно для объектов с чётко определёнными фазами работы, таких как устройства или сложные программные компоненты.
* Диаграмма деятельности (Activity Diagram): Отображает рабочие процессы, включая последовательность действий, условия выполнения операций и параллельные процессы. Она позволяет моделировать алгоритмы, бизнес-процессы и потоки управления в системе.
* Диаграмма временных характеристик (Timing Diagram): Включает временной контекст, демонстрируя, как состояния объекта или системы изменяются во времени. Используется в сценариях, где важна синхронизация событий.

Эти диаграммы взаимосвязаны: они дополняют друг друга, предоставляя различные уровни детализации. Например, диаграмма последовательностей и диаграмма коммуникаций конкретизируют взаимодействия, представленные на диаграммах случаев использования, а диаграмма состояний описывает внутреннее поведение объектов, участвующих в этих взаимодействиях.

Диаграмма деятельности – одна из наиболее универсальных диаграмм поведения. Её основное назначение:

* Моделирование рабочих процессов и алгоритмов: показывает последовательность шагов, необходимых для выполнения задачи.
* Отображение бизнес-процессов: детализирует, как задачи распределяются между участниками или подразделениями.
* Представление параллельных процессов: с помощью развилок (fork) для разделения потоков и узлов синхронизации (join) для их объединения.
* Управление потоками данных и управления: визуализирует, как информация и задачи перемещаются между элементами системы.

На диаграмме состояний моделируется поведение объекта через:

* Состояния: указывают, в каком положении находится объект в определённый момент (например, «Активен», «Ожидает подтверждения»).
* Начальное состояние: изображается чёрным кругом и обозначает стартовый пункт.
* Конечное состояние: представлено кругом с окружностью и показывает завершение жизненного цикла объекта.
* Переходы: стрелки, соединяющие состояния, описывают изменения объекта.
* События: инициируют переходы между состояниями (например, «Получен запрос»).
* Гарды (условия): записываются в квадратных скобках и определяют, при каких условиях переход возможен.

На диаграмме последовательностей моделируются связи между объектами через сообщения, которые могут быть:

* Синхронные сообщения: вызывающий объект ожидает завершения операции перед продолжением.
* Асинхронные сообщения: объект отправляет сообщение, но не ждёт ответа.
* Сообщения создания: указывают на создание нового объекта.
* Сообщения удаления: отображают завершение существования объекта.
* Возвратные сообщения: пунктирные стрелки, показывающие возвращаемый результат.

Для моделирования процессов с параллельным выполнением лучше всего подходит диаграмма деятельности. Она включает специальные элементы, такие как:

* Развилки (fork), которые делят поток выполнения на несколько параллельных ветвей.
* Узлы синхронизации (join), объединяющие параллельные ветви в один поток.

Эти возможности делают её незаменимой при проектировании многозадачных систем и сложных рабочих процессов, где требуется учесть взаимодействие нескольких компонентов, выполняющих задачи одновременно.

# Постановка задачи

Целью данной работы является построение диаграмм поведения для интерфейса электронной почты. Это включает в себя создание диаграмм активности, диаграмм состояний и диаграмм взаимодействия, которые наглядно отображают ключевые сценарии использования системы, изменение её состояний и передачу сообщений между компонентами. Построение таких диаграмм позволит глубже понять логику работы интерфейса, ключевые этапы выполнения процессов и взаимодействие между пользователями, сервисами и элементами системы.

Диаграмма активности служит для отображения последовательности действий, выполняемых системой или пользователем. В контексте интерфейса электронной почты это может включать такие процессы, как авторизация пользователя, создание и отправка сообщения, фильтрация входящих писем, добавление контактов или управление папками.

Диаграмма состояний описывает изменение состояния объектов системы под воздействием событий. Например, состояние сообщения может изменяться от «черновика» до «отправлено», а затем до «прочитано». Также диаграмма может отображать изменения статусов пользователя, таких как «в сети», «неактивен» или «заблокирован». Это помогает выявить, какие события инициируют изменения и какие условия должны быть выполнены для перехода из одного состояния в другое.

Диаграмма взаимодействия сосредотачивается на обмене сообщениями между компонентами интерфейса. Для электронной почты это может включать обмен данными между клиентским приложением, сервером почты и базой данных.

Задача работы заключается в построении и анализе диаграмм поведения для интерфейса электронной почты. Это включает:

* Определение ключевых сценариев использования системы, таких как отправка и получение писем, управление контактами и организация сообщений.
* Выявление состояний объектов и событий, которые инициируют их изменения.
* Определение последовательности взаимодействий между компонентами системы и пользователей.

Результатом работы станут три типа диаграмм поведения, которые обеспечат подробное понимание процессов и взаимодействий в системе. Диаграммы помогут уточнить требования к системе, выявить возможные проблемы в проектировании, улучшить процессы тестирования и разработки, а также подготовить документацию для дальнейшего использования. Визуализация поведения системы облегчит её масштабирование, поддержку и интеграцию с внешними сервисами.

# Описание программных средств

Для создания моделей в данной работе использовалось программное средство Draw.io, которое является универсальным инструментом для проектирования диаграмм и графических схем. Этот инструмент широко применяется для разработки UML-диаграмм, блок-схем, интеллект-карт, бизнес-процессов, а также схем сущностей и отношений. Благодаря своей функциональности и удобству, Draw.io активно используется как студентами, так и профессиональными разработчиками.

Draw.io представляет собой онлайн-сервис и настольное приложение с открытым исходным кодом, доступное для всех пользователей бесплатно. Это делает программу привлекательной для широкого круга задач, связанных с визуализацией и проектированием. В ней предусмотрен интуитивно понятный интерфейс, который позволяет быстро создавать схемы любой сложности.

Инструмент обладает богатым функционалом, таким как:

* Более 500 шаблонов элементов и фигур.
* Облегчённый интерфейс, в котором за короткий промежуток времени можно создать готовый проект.
* Поддержка горячих клавиш, задействованных в большинстве графических редакторов.
* Экспорт в форматы: JPG, PNG, SVG, VDSX.
* Возможность совместной работы.
* Наличие различных фоновых тем.
* Мультиязычный интерфейс.

Программа поддерживает мультиязычный интерфейс и предоставляет различные темы оформления, что позволяет адаптировать её под личные предпочтения.

Отдельного внимания заслуживает возможность совместной работы, что особенно полезно для командных проектов. Также Draw.io поддерживает импорт файлов из других инструментов, включая Microsoft Visio (.vsdx), Gliffy™ и Lucidchart™, что облегчает работу с уже существующими диаграммами.

Таким образом, Draw.io выступает надежным и функциональным инструментом, обеспечивающим эффективное создание и редактирование диаграмм в рамках проектирования программных систем и других областей.

# Практическое задание

Диаграмма охватывает все основные этапы процесса, включая проверки на корректность данных и действия в случае ошибок. Это обеспечивает наглядность и помогает лучше понять последовательность выполнения задач в системе.

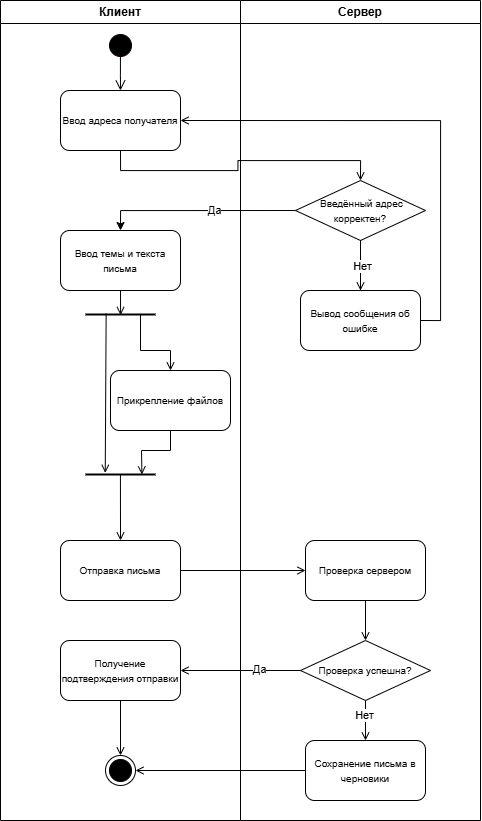


Рисунок 4.1 – Диаграмма деятельности

Описание процесса:

* Ввод адреса получателя: Процесс начинается с действия пользователя, который вводит email-адрес в соответствующее поле.
* Проверка корректности адреса: Если адрес некорректен, отображается сообщение об ошибке, после чего пользователь возвращается к вводу адреса. Если адрес корректен, процесс продолжается.
* Ввод темы и текста письма: Пользователь заполняет текстовое поле «Тема» и вводит содержимое письма.
* Прикрепление файлов (опционально): Пользователь может прикрепить файлы, которые будут отправлены вместе с письмом.
* Отправка письма: Пользователь нажимает кнопку «Отправить», инициируя процесс отправки.
* Проверка сервером: Система отправляет запрос на сервер для проверки введённых данных. Если сервер обнаруживает ошибку, письмо сохраняется в черновиках. Если все данные корректны, письмо успешно отправляется.
* Подтверждение отправки: Система уведомляет пользователя об успешной отправке письма.



Рисунок 4.2 – Диаграмма состояния

Диаграмма состояния описывает различные этапы, через которые проходит система во время процесса создания и отправки письма, а также переходы между этими состояниями:

* Начальное состояние: Процесс начинается в момент, когда пользователь инициирует создание нового письма, например, нажимает кнопку «Создать письмо» в интерфейсе.
* Состояние «Редактирование письма»: Пользователь вводит данные письма. Это состояние длится до тех пор, пока пользователь не завершит ввод и не нажмёт кнопку «Отправить».
* Состояние «Проверка данных»: Система проверяет введённые данные.
* Состояние «Ожидание отправки»: После нажатия кнопки "Отправить" система переходит в ожидание ответа от сервера для подтверждения возможности отправки письма.
* Состояние «Отправка письма»: При успешной проверке система передаёт письмо почтовому серверу. На этом этапе происходит обработка отправки, включая взаимодействие с сервером SMTP.
* Состояние «Ошибка»: Если во время отправки письма произошла ошибка (например, сервер недоступен, превышен лимит вложений или некорректен адрес получателя), система показывает пользователю сообщение об ошибке.
* Состояние «Письмо отправлено»: Если письмо успешно отправлено, система уведомляет пользователя об успешной отправке.
* Конечное состояние: Процесс завершён. Пользователь может продолжить работу с почтовым ящиком, вернувшись к общему интерфейсу.

Диаграмма состояния подчёркивает ключевые этапы работы системы, включая проверки, ошибки, возвраты к предыдущим шагам и подтверждение успешного завершения задачи. Она помогает разработчикам и пользователям понять логику работы системы на уровне состояний.