Учреждение образования

«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

**Отчёт**

по предмету «Проектирование программного обеспечения»

Лабораторная работа №2

«Проектирование информационных систем»

Студент: Лопатнюк П.В.

ФИТ 3 курс 1 группа

Преподаватель: Якубенко К. Д

Минск 2024

# Теоретические сведения

Структурный подход — это методология разработки программного обеспечения, которая позволяет создавать сложные системы, разбивая их на более мелкие, взаимосвязанные части. Этот подход основан на чёткой структуре и иерархическом разбиении, что делает его относительно простым для понимания и использования. Основные принципы:

Модульность: Система разделена на модули, каждый из которых решает отдельную задачу. Это позволяет разрабатывать и тестировать каждый модуль отдельно, что упрощает процесс разработки и поддерживает изменения в будущем.

Иерархичность: Модули организованы по уровням иерархии, причём более сложные модули состоят из более простых. Это обеспечивает последовательность разработки и даёт ясный вид общей структуры системы.

Абстракция: Используются абстрактные модели, которые представляют сущность системы, не вдаваясь в детали реализации. Это позволяет создать общее представление о системе и упростить ее понимание.

Пошаговая детализация: Разработка ведётся поэтапно, сначала создаётся общее представление, а затем детализируется каждый модуль. Это позволяет постепенно углубиться в детали системы и создать более точную модель.

Преимущества:

* Понятность: Структурный подход позволяет легко понять логику системы.
* Удобство разработки: Разбиение на модули упрощает процесс разработки и тестирования.
* Изменяемость: Изменения в одном модуле не влияют на работу других.
* Повторное использование: Модули могут быть использованы в других проектах.

Для создания моделей, которые помогают визуализировать и структурировать сложные системы, используются различные методологии и инструменты:

* DFD (Диаграмма потоков данных): Графическая нотация для описания движения данных в системе. Показывает как данные перемещаются между различными модулями и как они трансформируются в процессе обработки.
* SADT (Structured Analysis and Design Technique) - Техника структурированного анализа и проектирования: Методология, основанная на иерархическом разбиении задачи на более мелкие подзадачи.
* IDEF (ICAM Definition for Function Modeling): Набор методологий для моделирования бизнес-процессов и систем, включая различные модели, такие как:

1. IDEF0 (Функциональное моделирование): Описывает функции системы и их взаимосвязи.
2. IDEF1 (Моделирование данных): Описывает структуру данных и их взаимосвязи.
3. IDEF2 (Моделирование динамики): Описывает поведение системы во времени.
4. IDEF3 (Моделирование процессов): Описывает последовательность шагов в процессе.
5. IDEF4 (Моделирование предметной области): Описывает концептуальную модель области.
6. IDEF5 (Моделирование онтологии): Описывает отношения между концептами в предметной области.

Базовые принципы моделирования в IDEF0:

* Иерархичность: Разбиение системы на более мелкие функции по уровням иерархии.
* Графическое представление: Используется специальная нотация для визуализации функций и их взаимосвязей.
* Описание входа, выхода, управления и механизма: Каждая функция описывается через её входные данные, выходные данные, управляющие факторы и механизм реализации.
* Сосредоточение на функциях: IDEF0 концентрируется на функциях системы, а не на её реализации.

«Как есть» (As-Is): Модель существующей системы, которая описывает её текущее состояние. Используется для анализа и выявления недостатков системы. Это позволяет понять, как система работает в реальности и где есть проблемы. Используется при анализе существующих процессов, для выявления узких мест, ошибок и возможностей для улучшения.

«Как будет» (To-Be): Модель желаемого состояния системы, которая описывает её после внесения изменений. Используется для проектирования новой системы или улучшения существующей. Это позволяет представить, как система будет работать после внесения изменений. Используется при проектировании новой системы, для определения ее функциональности и структуры, а также при реинжиниринге существующей системы.

Структурный подход является фундаментом для разработки сложных систем, обеспечивая понятность, удобство разработки и изменяемость кода. Он помогает создать структурированные, модульные и легко поддерживаемые системы.

# 2. Постановка задачи

## 2.1 Описание функциональных требований

Проект «Интерфейс электронной почты» нацелен на создание удобного и безопасного программного обеспечения для работы с электронной корреспонденцией. В процессе разработки были определены функциональные требования, которые должны быть реализованы в программе. Ниже представлены основные функциональные требования:

* Отправка и получение писем — это основное требование для любого почтового приложения. Пользователь должен иметь возможность добавлять файлы (документы, изображения и т.д.) к письму. Приложение должно автоматически проверять наличие новых сообщений. Возможность узнать, было ли письмо успешно отправлено или доставлено. Реализация поддержки различных протоколов, таких как IMAP/POP3 для получения почты и SMTP для отправки.
* Управление контактами — важная часть работы с электронной почтой. Пользователь должен иметь возможность создавать новые контакты с полями для ввода имени, адреса электронной почты, номера телефона и другой информации (например, адреса, заметки). Опция редактирования информации о существующих контактах и удаления ненужных.
* Организация почтового ящика играет важную роль в управлении поступающей корреспонденцией. Необходима возможность сортировки писем по дате, отправителю, теме и статусу (прочитанные, непрочитанные). Функция архивирования старых писем для освобождения места в почтовом ящике без их удаления. Удобное удаление ненужных или нежелательных писем с возможностью восстановления из корзины.
* Эффективный поиск и фильтрация писем являются критически важными для управления электронной почтой. Пользователь может вводить одно или несколько ключевых слов, и система должна находить соответствующие письма по содержимому, отправителю, получателю или теме.

Эти функциональные требования обеспечивают пользователям эффективное и комфортное использование программы «Интерфейс электронной почты», делая её инструментом для продуктивного управления корреспонденцией.

## 3. Описание программных средств

Visual Paradigm — это многофункциональное программное обеспечение для моделирования и проектирования, которое широко используется как в IT, так и в бизнес-средах. Программа позволяет создавать различные типы диаграмм, такие как UML (Unified Modeling Language), BPMN (Business Process Model and Notation), DFD (Data Flow Diagram), ERD (Entity Relationship Diagram) и многие другие. Она предназначена для визуализации сложных процессов, архитектуры систем и бизнес-структур, что делает её полезной для аналитиков, разработчиков и менеджеров проектов.  
 Программа разработана компанией Visual Paradigm International и доступна как в виде десктопного приложения для платформ Windows, macOS и Linux, так и онлайн через веб-интерфейс. Это дает возможность пользователям работать с диаграммами как на локальных компьютерах, так и в облаке, что упрощает совместную работу над проектами. Visual Paradigm поддерживает функционал для создания моделей в режиме реального времени, что особенно полезно для командного взаимодействия и управления проектами.  
 Одной из ключевых особенностей Visual Paradigm является её интеграция с популярными средами разработки, такими как Eclipse, NetBeans и IntelliJ IDEA, что позволяет seamlessly переносить диаграммы в код или управлять проектами на разных этапах жизненного цикла разработки. Также программа поддерживает экспорт диаграмм в популярные форматы, включая PNG, PDF, SVG, и легко интегрируется с инструментами для версионирования, такими как Git.  
 Visual Paradigm предлагает гибкий подход к управлению процессами, позволяющий вам адаптировать его под свои индивидуальные нужды. Наш принцип Just-in-Time дает вам свободу выбора: вы можете использовать готовые процессы или создавать собственные, используя богатую библиотеку компонентов.

Visual Paradigm предоставляет гибкий подход к моделированию, который подходит для использования как в малых командах, так и в крупных предприятиях.

# 4. Описание практического задания

Контекстная диаграмма (или диаграмма уровня 0) является верхнеуровневой диаграммой потоков данных (DFD), которая отображает общее взаимодействие системы с внешними сущностями, показывая, какие данные поступают в систему, как они обрабатываются, и какие данные отправляются обратно во внешние системы. Контекстная диаграмма описывает всю систему одним процессом и показывает основные внешние сущности, взаимодействующие с ней.

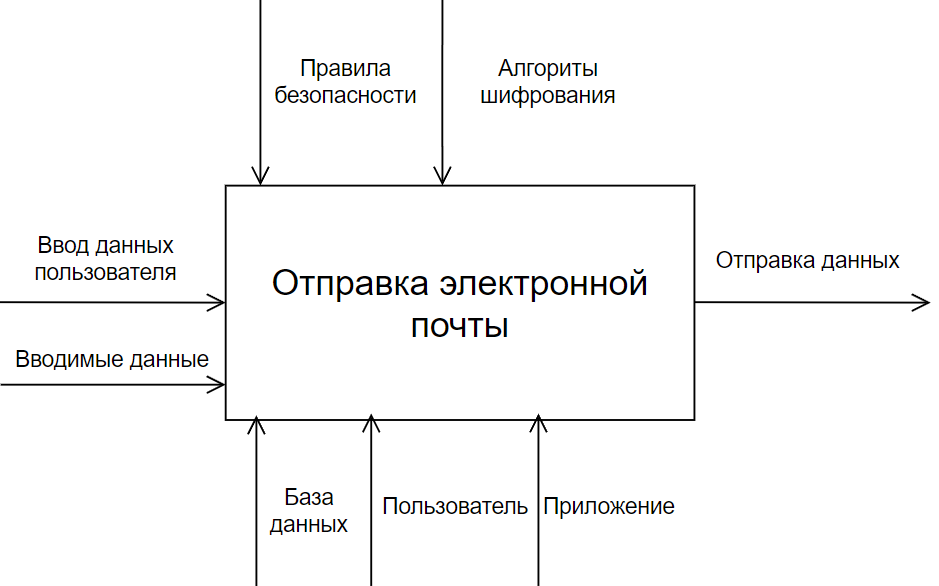


Рисунок 4.1 – Диаграмма отправки и получения почты

Центральный элемент диаграммы — процесс «Отправка и получение электронной почты». Этот процесс связан с различными внешними элементами и потоками данных.  
 Настройки учётной записи и письмо от пользователя подаются на вход системы, что означает, что пользователь настраивает учётную запись и отправляет письмо через систему.  
 Правила безопасности и Файлы вложений также поступают в систему, указывая, что система учитывает правила безопасности и обрабатывает вложенные файлы.  
 На выходе из системы:

* Отправленное письмо, которое выходит из системы как результат процесса отправки.
* Полученные письма, поступающие обратно пользователю.
* Фильтрованные сообщения (спам и нежелательные письма), которые система обрабатывает и фильтрует.
* Уведомления, которые могут быть отправлены пользователю, когда происходят определённые события, связанные с почтовыми сообщениями

Диаграмма 1-го уровня декомпозиции (DFD уровня 1) — это детализация контекстной диаграммы, которая разбивает основной процесс на более мелкие подпроцессы, чтобы показать внутренние действия системы и их взаимодействие. В отличие от контекстной диаграммы, которая отображает только взаимодействие с внешними сущностями, диаграмма уровня 1 более подробно описывает процессы внутри системы, их взаимосвязи и потоки данных между ними.

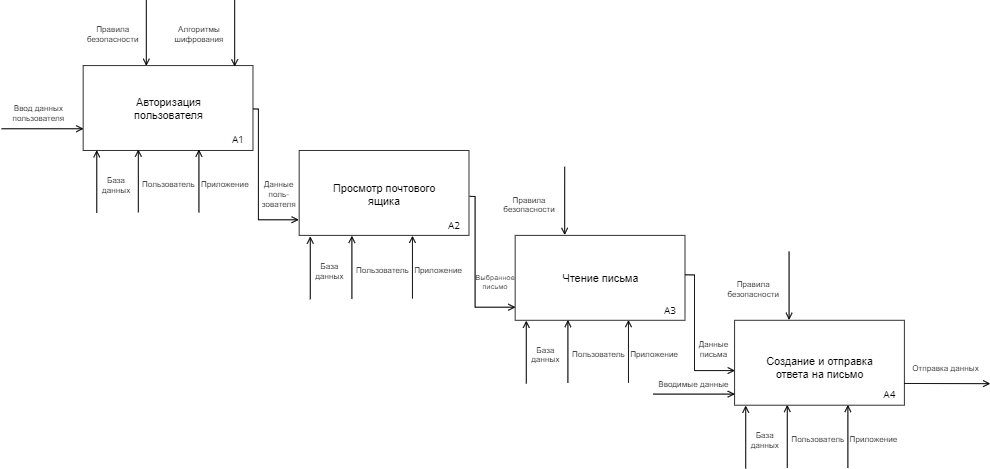


Рисунок 4.2 – Диаграмма 1-го уровня декомпозиции

Входы (левая сторона):

* Ввод данных пользователя — информация, которую пользователь вводит для регистрации и аутентификации, такая как логин, пароль, или другие идентификационные данные.

Управление (верхняя сторона):

* Стандарты верификации — проверяют, что введённые пользователем данные соответствуют заданным критериям безопасности и правильности.
* Алгоритмы поиска и фильтрации — помогают ускорить поиск нужных писем и разделение сообщений на важные и спам.
* Правила безопасности — применяются на всех этапах взаимодействия, контролируя процессы чтения и управления письмами, а также отправку новых сообщений.
* Правила форматирования писем — применяются при создании и отправке новых писем, гарантируя соответствие стандартам оформления.

Процессы (центральная часть):

* Регистрация и аутентификация пользователя — отвечает за проверку  
  учётных данных пользователя и регистрацию нового аккаунта в системе. Здесь используется база данных пользователей для сравнения введённых данных с существующими.
* Просмотр почтового ящика — позволяет пользователю видеть список писем, используя систему фильтрации и сортировки для отображения нужных сообщений.
* Чтение и управление письмами — процесс, позволяющий открывать письма, просматривать содержимое, удалять, архивировать или помечать их как важные.
* Создание и отправка нового письма — пользователь создаёт новое сообщение, форматирует его в соответствии с правилами и отправляет через систему.

Выходы (правая сторона):

* Выбранное письмо — результат поиска и сортировки писем, выбранное пользователем для чтения.
* Опция создания нового письма — возможность начать процесс написания нового письма, которое затем будет отправлено через SMTP.
* Отправка данных через SMTP — новый процесс отправки сообщения, передаваемого через почтовый протокол.
* Получение данных через IMAP/POP3 — процесс получения новых писем через почтовые протоколы для загрузки в почтовый ящик.

Механизмы (нижняя сторона):

* База данных пользователей — хранит информацию о пользователях, включая их учётные записи, для аутентификации.
* База данных писем — хранит все письма, отправленные и полученные, для доступа к ним через почтовый ящик.
* Получение данных через IMAP/POP3 — протоколы, которые используются для доступа к сообщениям с почтового сервера.
* Отправка данных через SMTP — отвечает за передачу новых сообщений по почтовому протоколу.

Таким образом, эта диаграмма описывает детализированные внутренние процессы системы отправки и получения почты, показывая, как данные поступают в систему, как они обрабатываются, и каким образом результаты выводятся обратно пользователю.