МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение высшего образования
«КУЗБАССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ИМЕНИ Т. Ф. ГОРБАЧЕВА»
Филиал КузГТУ в г. Белово

Кафедра инженерно-экономическая

# УП.06.01 «Сопровождение информационных систем»

Методические рекомендации
по выполнению учебной практики
для специальности
09.02.07 «Информационные системы и программирование»

Составитель: Витвицкий М.Н.
Рассмотрены и утверждены на заседании кафедры
Протокол № 5 от 17.01.2026 г.
Рекомендовано учебно-методической комиссией специальностей СПО в качестве электронного издания для использования в учебном процессе
Протокол № 5 от 20.01.2026 г.



Белово 2026

**СОДЕРЖАНИЕ**

# ОРГАНИЗАЦИЯ УЧЕБНОЙ ПРАКТИКИ

Первоначальные профессиональные навыки обучающиеся по основным профессиональным образовательным программам получают во время прохождения учебных и производственных практик. Практика имеет целью комплексное освоение обучающимися всех видов профессиональной деятельности по специальности (профессии) среднего профессионального образования, формирование общих и профессиональных компетенций, а также приобретение необходимых умений и опыта практической работы по специальности (профессии).

Практика по профилю специальности направлена на формирование у обучающегося общих и профессиональных компетенций, приобретение практического опыта и реализуется в рамках профессиональных модулей ОП СПО по каждому из видов профессиональной деятельности, предусмотренных ФГОС СПО по специальности.

Учебная практика проводится в филиале КузГТУ в г.Белово в специально оборудованных аудиториях. Организацию и руководство практикой по профилю специальности (профессии) осуществляют руководители практики из числа профессорско-преподавательского состава филиала КузГТУ в г.Белово.

Руководитель практики из числа лиц, относящихся к профессорско-преподавательскому составу филиала:

- разрабатывает содержание и планируемые результаты практики;
- осуществляет руководство практикой;
- контролирует реализацию программы практики и условия проведения практики, в том числе требования охраны труда, безопасности жизнедеятельности и пожарной безопасности в соответствии с правилами и нормами, в том числе отраслевыми;
- формирует группы в случае применения групповых форм проведения практики;
- определяет процедуру оценки общих и профессиональных компетенций обучающегося, освоенных им в ходе прохождения практики;
- разрабатывает формы отчетности и оценочный материал прохождения практики.

Обучающиеся в период прохождения практики обязаны:

- выполнять задания, предусмотренные программами практики;

- соблюдать действующие в филиале КузГТУ в г.Белово правила внутреннего трудового распорядка;

- соблюдать требования охраны труда и пожарной безопасности.

Результаты практики определяются программами практики, разрабатываемыми руководителями практики из числа профессорско-преподавательского состава филиала КузГТУ в г.Белово. По результатам практики руководителем практики формируется аттестационный лист, содержащий сведения об уровне освоения обучающимся профессиональных компетенций, а также характеристика на обучающегося по освоению профессиональных компетенций в период прохождения практики.

В период прохождения практики обучающимся ведется дневник практики. По результатам практики обучающимся составляется отчет. В качестве приложения к дневнику практики обучающийся оформляет графические, аудио-, фото-, видеоматериалы, подтверждающие практический опыт, полученный на практике.

Аттестация по итогам учебной практики проводится с учетом (или на основании) результатов ее прохождения, подтверждаемых аттестационным листом.

Отчет по практике является основным документом, характеризующим работу обучающегося во время практики. Отчет составляется в соответствии с программой практики и содержит следующие разделы:

1. Титульный лист
2. Задание на учебную практику
3. Введение
4. Теоретические основы в соответствии с темами практики
5. Реализация поставленной задачи
6. Выводы
7. Список используемой литературы

Комплект документов, оформляемых при прохождении учебной практики, а также титульный лист отчета по учебной практике приведены в Приложении А-Д.

# КОНТРОЛЬ РЕЗУЛЬТАТОВ ВЫПОЛНЕНИЯ ЗАДНИЙ ПО УЧЕБНОЙ ПРАКТИКЕ

Контроль результатов по учебной практике студентов осуществляться в пределах времени, отведенного на обязательные учебные занятия, может проходить в письменной, устной или смешанной форме, с представлением продукта деятельности учащегося.

В качестве форм и методов контроля работы обучающихся, могут быть использованы, зачеты, тестирование, самоотчеты по этапам практики, практические задания, и др., которые могут осуществляться на учебном занятии или вне его.

Общими критериями оценки результатов работы обучающегося являются:

- уровень освоения учащимся учебного материала;
- умение обучающегося использовать теоретические знания при выполнении практических задач;
- сформированность общих и профессиональных компетенций;
- обоснованность и четкость изложения ответа;
- оформление материала в соответствии с требованиями.

# МЕТОДИЧЕСКИЕ МАТЕРИАЛЫ

**УЧЕБНАЯ ПРАКТИКА 06.01 «Сопровождение информационных систем».**

Учебная практика — это форма организации учебного процесса, нацеленная на формирование и закрепление практических умений и навыков через непосредственное выполнение профессиональных или учебно-профессиональных действий. Её суть — в переносе теоретических знаний в плоскость реального (или моделируемого) практического действия.

**Цель учебной практики 06.01 «Сопровождение информационных систем».**

Комплексное освоение обучающимися вида профессиональной деятельности «Сопровождение информационных систем», формирование общих и профессиональных компетенций, приобретение необходимого опыта практической работы по специальности.

**Задачи учебной практики 06.01 «Сопровождение информационных систем».**

Осуществление инсталляции, настройки и сопровождения одной из информационных систем.

Выполнение регламентов по обновлению, техническому сопровождению и восстановлению данных информационной системы.

Сохранение и восстановление базы данных информационной системы.

Организация доступа пользователей к информационной системе в рамках компетенции конкретного пользователя.

Модификация отдельных модулей информационной системы.

Взаимодействие со специалистами смежного профиля при разработке методов, средств и технологий применения объектов профессиональной деятельности.

**Учебная практика осуществляется в двух формах:**

1. Проработка теоретической части рассматриваемой задачи (теоретическая часть работы);
2. Проработка практической части рассматриваемой задачи (выдается преподавателем индивидуально согласно перечню тем заданий).

**Задания по учебной практике**

| № раздела (темы) | Вопросы, выносимые на изучение | Количество часов |
|---|---|---|
| Тема 1. Сопровождение информационной системы. | Задание 1.1. Подбор и настройка конфигурации программного обеспечения компьютерных систем | 2 |
| | Задание 1.2. Использование методов защиты программного обеспечения компьютерных систем | 4 |
| | Задание 1.3. Проведение инсталляции программного обеспечения компьютерных систем | 4 |
| | Задание 1.4. Настройка отдельных компонентов программного обеспечения компьютерных систем | 4 |
| | Задание 1.5. Анализ рисков и характеристик качества программного обеспечения | 4 |
| | Задание 1.6. Выполнение отдельных видов работ на этапе поддержки программного обеспечения компьютерной системы | 4 |
| Тема 2. Проектирование части информационной системы. | Задание 2.1. Разработка модели бизнес-процессов информационной системы. Методология IDEF0 | 4 |
| | Задание 2.2. Разработка модели бизнес-процессов информационной системы. Методология DFD | 4 |
| | Задание 2.3. Построение диаграмм вариантов использования при проектировании ИС | 6 |
| | Задание 2.4. Построение диаграмм | 6 |

| | | |
|---|---|---|
| | деятельности для описания поведения разрабатываемой системы | |
| | Задание 2.5. Построение диаграмм последовательности для описания разрабатываемой системы | 6 |
| | Задание 2.6. Разработка состава требований к проектируемой ИС | 4 |
| | Задание 2.7. Разработка логической и физической модели данных информационной системы | 4 |
| | Задание 2.8. Разработка диаграммы классов для описания структуры выбранного решения | 4 |
| | Задание 2.9. Разработка моделей интерфейсов пользователей | 4 |
| | Задание 2.10. Выбор технологий для разработки ИС | 4 |
| Тема 3. Надёжность и качество информационной системы. | Задание 3.1. Определение показателей безотказности системы | 4 |
| | Задание 3.2. Определение показателей долговечности системы | 4 |
| | Задание 3.3. Определение комплексных показателей надежности системы | 4 |
| | Задание 3.4. Определение единичных показателей достоверности информации в системе | 4 |
| Тема 4. Безопасность информационных систем. | Задание 4.1. Основные угрозы | 2 |
| | Задание 4.2. Защита от несанкционированного доступа | 4 |
| Тема 5. Разработка технического задания на | Задание 5.1. Стандарты на разработку и внедрение информационной системы. | 2 |

| | | |
|---|---|---|
| внедрение ИС. | Задание 5.1. Построение графика разработки и внедрения информационной системы. | 4 |
| | Задание 5.2. Разработка сценария внедрения информационной системы для рабочего места. | 4 |
| | Задание 5.3. Разработка пояснительной записки к внедрению информационной системы. | 4 |
| | Задание 5.4. Написание плана вывода информационной системы из эксплуатации. | 4 |
| Промежуточная аттестация в форме: дифференцированного зачета | | |

**Типовые вопросы на защиту отчета**

1. Назовите основные этапы жизненного цикла информационной системы?

2. Каковы преимущества и недостатки облачных технологий перед традиционными?

3. Опишите механизм работы нейронной сети?

4. Классификация информационных систем

5. Этапы внедрения информационной системы

6. Основные задачи сопровождения информационной системы

7. Способы идентификации ошибок, возникающих в процессе эксплуатации системы

8. Способы коррекции ошибок, возникающих в процессе эксплуатации системы

9. Способы резервного копирования информации

10. Способы настройки сетевого оборудования

11. Особенности эксплуатации облачных информационных систем

12. Методы разработки сценариев внедрения информационных систем

13. Методы разработки технического задания

14. Методы разработки календарных графиков разработки и внедрения информационных систем

15. Состав эксплуатационной документации на информационную систему

16. Требования к оформлению эксплуатационной документации на информационную систему

17. Состав обучающей документации на информационную систему

18. Методы разработки эксплуатационной документации на информационную систему.

## ТЕМА 1. СОПРОВОЖДЕНИЕ ИНФОРМАЦИОННОЙ СИСТЕМЫ.

Сопровождение информационных систем представляет собой комплекс мероприятий, направленных на поддержание эффективной работы информационных систем в организации. Это важнейший этап жизненного цикла любой информационной системы, обеспечивающий её бесперебойное функционирование и развитие.

Актуальность темы обусловлена следующими факторами:

постоянное развитие информационных технологий требует регулярного обновления и поддержки систем;

рост числа пользователей информационных систем увеличивает нагрузку на техническую поддержку;

необходимость обеспечения безопасности и целостности данных;

потребность в оптимизации работы систем под изменяющиеся бизнес-процессы;

Основные направления сопровождения информационных систем включают:

техническое обслуживание и поддержка работоспособности системы;

обновление программного обеспечения и баз данных;

восстановление данных при сбоях;

разработка и актуализация технической документации;

обучение пользователей;

мониторинг производительности системы;

Практическая значимость сопровождения информационных систем заключается в:

обеспечении стабильной работы бизнес-процессов организации;

снижении рисков потери данных;

повышении эффективности работы пользователей;

оптимизации затрат на ИТ-инфраструктуру;

соответствии современным требованиям безопасности;

В процессе практики студенты получат возможность:

изучить реальные информационные системы;

приобрести практические навыки сопровождения;

развить профессиональные компетенции;

сформировать представление о работе специалиста по сопровождению информационных систем;

получить опыт взаимодействия с пользователями системы;

## Задание 1.1. Подбор и настройка конфигурации программного обеспечения компьютерных систем.

Т: Программное обеспечение (ПО) — это совокупность программ, данных и документации, необходимых для функционирования компьютерной системы. Оно делится на несколько категорий:

Системное ПО управляет ресурсами компьютера, обеспечивает взаимодействие между аппаратурой и прикладными программами. К нему относятся операционные системы, драйверы, утилиты для обслуживания системы.

Прикладное ПО решает конкретные задачи пользователя: обработка текстов, создание графики, анализ данных и т. д..

Инструментальное ПО используется для разработки новых программ (системы программирования, отладчики и т. п.).

Конфигурация ПО — это набор программных компонентов, их версии, параметры настройки и взаимосвязи, которые определяют функциональность и производительность системы.

Этапы подбора ПО

Анализ требований и задач. Определяются цели использования системы, типы задач, которые она должна решать, объём данных, количество пользователей и другие параметры. Учитываются текущие и перспективные потребности.

Выбор ПО. Подбираются программные продукты, соответствующие требованиям. Учитываются такие факторы, как функциональность, совместимость с аппаратурой и другими программами, лицензия, репутация разработчика, наличие технической поддержки. Для бизнеса важно также оценить интеграцию с

существующими системами, масштабируемость, стоимость владения.

inzheterra.ru +2

Проверка совместимости. Убеждаются, что выбранное ПО совместимо с аппаратной платформой, операционной системой, другими программами и сетевыми настройками.

Тестирование и пилотный запуск. Перед полным внедрением ПО тестируется на небольшом участке или в демо-режиме, чтобы выявить возможные проблемы.

Настройка ПО

После установки ПО требуется его настройка — изменение параметров для оптимальной работы в конкретной среде. Это может включать:

Конфигурацию параметров системы. Например, настройку прав доступа, параметров сети, параметров безопасности.

Интеграцию с другими системами. Настройку обмена данными между различными программными продуктами, синхронизацию баз данных.

Оптимизацию производительности. Настройку параметров памяти, процессора, дискового пространства, чтобы система работала эффективно.

Настройка пользовательского интерфейса. Адаптацию интерфейса под потребности пользователей, добавление или удаление элементов управления, изменение цветовой схемы и т. п.

Для настройки часто используются специальные инструменты:

Средства управления системой (например, «Диспетчер задач» в Windows для контроля процессов, «Управление компьютером» для администрирования ресурсов).

Консольные утилиты для тонкой настройки параметров.

Графические интерфейсы настроек в программах и операционной системе.

Факторы, влияющие на конфигурацию

Аппаратные ресурсы. Мощность процессора, объём оперативной памяти, ёмкость дисков ограничивают выбор и настройку ПО.

Тип задач. Для ресурсоёмких задач (например, 3D-моделирования, обработки видео) требуется специализированное ПО и его тщательная оптимизация.

Количество пользователей. В многопользовательских системах важна настройка прав доступа, одновременной работы, балансировки нагрузки.

Требования к безопасности. Необходимо настроить защиту данных,

аутентификацию, шифрование и другие меры безопасности.

Обновления. Регулярные обновления ПО требуют планирования и тестирования, чтобы не нарушить работу системы.

Инструменты для мониторинга и поддержки

Для контроля состояния системы и ПО используются:

Журналы событий (Event Viewer в Windows), где фиксируются ошибки, предупреждения и другие события.

Системы мониторинга, которые отслеживают нагрузку на систему, доступность сервисов, состояние аппаратных компонентов.

Утилиты для диагностики, например, программы для проверки состояния жёстких дисков, тестирования памяти.

Риски и проблемы

При подборе и настройке ПО могут возникнуть следующие проблемы:

Конфликты между программами из-за несовместимости версий или настроек.

Снижение производительности из-за неправильной настройки или избыточного количества ПО.

Угрозы безопасности при использовании нелицензионного или устаревшего ПО.

Сложности с поддержкой при выборе малоизвестных или узкоспециализированных продуктов.

П: Подобрать конфигурацию и описать настройки для установки виртуальной машины с RedOS.

## Задание 1.2. Использование методов защиты программного обеспечения компьютерных систем.

Т: Использование методов защиты программного обеспечения (ПО) компьютерных систем — это комплекс мер, направленных на предотвращение несанкционированного доступа, копирования, модификации, анализа и распространения ПО, а также на защиту от вредоносных программ и других угроз. Эти меры включают технические, организационные и юридические подходы, которые дополняют друг друга для создания многоуровневой системы безопасности.

Основные угрозы для программного обеспечения

Вредоносное ПО (вирусы, трояны, черви, шпионские программы, программы-

вымогатели). Они могут повредить систему, украсть данные или заблокировать работу.

Уязвимости в коде. Ошибки в программном коде или использование устаревших версий ПО создают «дыры», которые злоумышленники могут использовать для проникновения в систему. Примеры: SQL-инъекции, межсайтовый скриптинг (XSS), переполнение буфера.

Атаки на отказ обслуживания (DDoS). Направлены на перегрузку системы и делают её недоступной для пользователей.

Фишинг и социальная инженерия. Методы обмана пользователей для получения доступа к системам или данным.

Несанкционированный доступ. Попытки получения доступа к информации или функциям системы без соответствующих прав.

Утечки данных. Непреднамеренное или преднамеренное раскрытие конфиденциальной информации.

Технические методы защиты

Шифрование. Используется для защиты данных при передаче (например, с помощью SSL/TLS) и хранении (шифрование баз данных, жёстких дисков). Криптографические методы делают информацию недоступной для посторонних даже при её перехвате.

Аутентификация и авторизация. Аутентификация проверяет подлинность пользователей (например, с помощью паролей, биометрии, многофакторной аутентификации — MFA). Авторизация определяет права доступа на основе ролей пользователей (ролевое управление доступом — RBAC).

Управление уязвимостями. Включает регулярное обновление ПО для устранения известных уязвимостей, сканирование системы на предмет слабых мест с помощью сканеров уязвимостей (например, OpenVAS).

Безопасное кодирование. Использование лучших практик при написании кода для избежания распространённых ошибок. Включает рецензирование кода (code review) и статический/динамический анализ кода (SAST/DAST).

Антивирусное ПО. Обнаруживает и нейтрализует вредоносные программы с помощью сигнатурного, эвристического и поведенческого анализа. Современные антивирусы также защищают от фишинга, программ-вымогателей и других угроз.

Песочницы (Sandbox). Изолированная виртуальная среда для безопасного запуска непроверенного кода. Позволяет анализировать поведение программ без риска для основной системы.

Межсетевые экраны (Firewall). Фильтруют сетевой трафик, блокируя несанкционированный доступ к системе. NGFW (межсетевые экраны нового поколения) дополнительно анализируют содержимое пакетов и могут включать функции антивирусов и IDS/IPS.

Системы обнаружения и предотвращения вторжений (IDS/IPS). Анализируют сетевой трафик и поведение системы, выявляя подозрительную активность.

Защита от реверса и анти-отладка. Методы, усложняющие анализ кода (обфускация, встраивание ложных фрагментов, использование промежуточного байт-кода). Активная защита (anti-tamper, anti-debug) предотвращает несанкционированное изменение кода и отладку.

Цифровые подписи и проверка целостности. Генерация хэшей для контроля неизменности кода и данных. При запуске ПО проверяется соответствие текущего состояния изначальному.

Организационные меры

Политика информационной безопасности. Регламентирует правила работы с данными, доступ к системам, использование паролей, обработку инцидентов.

Обучение сотрудников. Тренинги по киберграмотности, распознаванию фишинга, правилам работы с конфиденциальной информацией.

Контроль доступа. Реализация принципа минимальных привилегий: сотрудники получают только те права, которые необходимы для выполнения обязанностей.

Управление версиями и сборка ПО. Использование хранилищ секретов (например, Рутокен Vault, SecretHub) для хранения конфиденциальных данных (ключей, паролей).

Резервное копирование данных. Создание дубликатов важных файлов для восстановления после сбоев или атак.

Юридические методы

Лицензирование и DRM (Digital Rights Management). Контроль использования ПО, предотвращение несанкционированного копирования и распространения.

Договоры и EULA (End User License Agreement). Юридическая база для защиты прав разработчика и доказательства нарушения.

Современные тенденции

Использование ИИ и машинного обучения для обнаружения новых угроз и анализа поведения системы.

Контейнеризация и оркестрация. Управление безопасностью в облачных и контейнерных средах с помощью инструментов вроде Luntry.

Zero Trust-архитектура. Принцип «никогда не доверяй, всегда проверяй» для всех пользователей и устройств.

Эффективная защита ПО требует комплексного подхода, сочетающего технические меры (шифрование, антивирусы, контроль уязвимостей), организационные мероприятия (обучение, политики безопасности) и юридические инструменты. Ключевой принцип — многоуровневая защита, которая затрудняет взлом и делает его экономически нецелесообразным для злоумышленников.

П: Описать реализованные методы защиты виртуальной машины с RedOS.

## Задание 1.3. Проведение инсталляции программного обеспечения компьютерных систем.

Т: Основные этапы инсталляции

Предварительная подготовка: проверка системных требований, создание резервной копии важных данных, закрытие фоновых приложений, проверка целостности установочных файлов.

Процесс установки: запуск установочного файла, принятие лицензионного соглашения, выбор компонентов для установки, определение пути установки, настройка параметров программы.

Завершающие операции: проверка корректности установки, настройка параметров безопасности, создание резервной копии конфигурации, обучение пользователей, типы установочных файлов.

Основные форматы: EXE и MSI — стандартные установочные файлы для Windows, DMG — для macOS, DEB и RPM — для Linux, ZIP, RAR — архивные файлы с исполняемыми компонентами, ISO — образы дисков.

Системные требования: аппаратные компоненты: процессор, оперативная память, место на диске, программное окружение: версия ос, наличие зависимостей,

сетевые условия: доступ в интернет для активации, права доступа: административные привилегии, методы установки.

Основные подходы: ручная установка — последовательное выполнение всех этапов, автоматическая установка — использование скриптов и пакетных менеджеров, удаленная установка — установка через сеть, групповая установка — одновременная установка на несколько компьютеров, проверка работоспособности.

Тестирование включает: запуск программы, проверка основных функций, тестирование интеграции с другими системами, проверка безопасности, оценка производительности, документирование процесса.

Необходимая документация: план установки, журнал изменений, протоколы тестирования, инструкции по использованию, сведения о проблемах и их решениях, устранение проблем.

Типичные проблемы и решения: конфликты с другим ПО — проверка совместимости, ошибки установки — проверка системных требований, проблемы с активацией — проверка лицензий, сбои при запуске — проверка зависимостей, рекомендации по установке.

Важные аспекты: использование официальных источников дистрибутивов, проверка целостности файлов, создание точек восстановления системы, регулярное обновление антивирусного ПО, документирование всех этапов установки, безопасность при установке.

Меры предосторожности: проверка источников ПО, использование антивирусного контроля, резервное копирование данных, контроль доступа к установочным файлам, соблюдение политик безопасности организации.

Успешная инсталляция программного обеспечения требует тщательного планирования, подготовки и документирования всех этапов процесса. Это обеспечивает надежную работу системы и минимизирует риски возникновения проблем в будущем.

П:Установить дополнительный модуль в виртуальную машину с RedOS.

## Задание 1.4. Настройка отдельных компонентов программного обеспечения компьютерных систем.

Т: Компонент программного обеспечения — это независимый элемент системы, выполняющий определенные функции и взаимодействующий с другими

компонентами.

Настройка ПО — это процесс конфигурирования параметров программного обеспечения для оптимальной работы в конкретной среде.

Классификация компонентов ПО

Системные компоненты: операционные системы, драйверы устройств, системные утилиты, компоненты безопасности.

Прикладные компоненты: бизнес-приложения, офисные программы, специализированное ПО, средства разработки.

Сервисные компоненты: антивирусные системы, архиваторы, программы очистки, средства мониторинга, этапы настройки компонентов.

Предварительная подготовка: анализ системных требований, проверка совместимости, создание резервной копии, подготовка необходимых инструментов.

Процесс настройки: установка компонентов, конфигурация параметров, настройка взаимодействия между компонентами, тестирование работоспособности.

Постнастройка: документирование изменений, обучение пользователей, мониторинг работы, внесение корректировок, основные параметры настройки, системные параметры: настройки безопасности, параметры производительности, сетевые настройки, права доступа, пользовательские параметры: интерфейс, рабочие профили, личные настройки, предпочтения, методы настройки.

Ручная настройка: через графический интерфейс, консольные команды, редактирование конфигурационных файлов.

Автоматическая настройка: использование мастеров настройки, применение групповых политик, скриптовое управление, инструменты настройки.

Системные утилиты: диспетчер задач, панель управления, системный конфигуратор, средства диагностики.

Специализированные инструменты: Конфигураторы, Мониторы производительности, Средства аудита, Системы мониторинга.

Технические проблемы: конфликты компонентов, несовместимость версий, проблемы с производительностью, ошибки конфигурации.

Организационные проблемы: отсутствие документации, недостаточная квалификация персонала, нарушение процедур, несогласованность действий, рекомендации по настройке.

Базовые принципы: последовательный подход, документирование изменений, тестирование после каждой настройки, резервное копирование, контроль безопасности, контроль качества настройки.

Методы проверки: функциональное тестирование, нагрузочное тестирование, проверка безопасности, мониторинг производительности, анализ журналов событий.

Успешная настройка компонентов ПО требует комплексного подхода, включающего тщательное планирование, выполнение необходимых процедур и постоянный мониторинг результатов. Важно учитывать специфику каждого компонента и его взаимодействие с другими элементами системы.

П:Настройте удаленное подключение к виртуальной машины с RedOS.

## Задание 1.5. Анализ рисков и характеристик качества программного обеспечения.

Т: Риск — это вероятность возникновения негативных событий, которые могут помешать успешному внедрению или функционированию ПО. Анализ рисков включает их идентификацию, оценку вероятности и потенциального воздействия на проект, а также разработку стратегий минимизации.

Основные категории рисков при внедрении ПО:

Технические риски:

Ошибки в коде, которые не были выявлены на стадии тестирования, могут привести к сбоям в работе системы.

Несоответствие функциональности заявленным требованиям.

Сложности с интеграцией с другими системами (базами данных, сервисами, сторонними приложениями).

Невыполнение требований производительности (например, система не выдерживает высокие нагрузки).

Использование нестабильных технологий, которые ещё не прошли апробацию.

Организационные риски:

Отсутствие подготовки пользователей, что может снизить эффективность работы.

Неопределённость требований на стадии планирования, что приводит к изменениям в проекте и дополнительным затратам.

Нехватка ресурсов (квалифицированных специалистов, времени).

Риски управления проектом, связанные с отсутствием навыков проектного менеджмента у менеджера.

Экономические риски:

Перерасход бюджета из-за дополнительных работ по настройке, доработке системы, обучению пользователей.

Задержки в сроках внедрения, влияющие на операционную деятельность компании.

Риски безопасности:

Уязвимости в ПО, позволяющие злоумышленникам получить доступ к системе.

Проблемы с конфиденциальностью данных, ведущие к утечкам информации.

Методы анализа рисков:

SWOT-анализ помогает определить сильные и слабые стороны системы, а также выявить возможности и угрозы.

Анализ ошибок и дефектов через различные виды тестирования (функциональное, стресс-тестирование, тестирование на проникновение).

Количественные и качественные модели оценки рисков — расчёт вероятности возникновения рисков и их воздействия на проект.

Риск-матрица позволяет приоритизировать риски на основе вероятности и влияния.

Метод FTA (анализ дерева отказов) помогает понять причинно-следственные связи в системе.

Монте-Карло симуляция — количественный метод расчёта вероятности различных исходов.

Характеристики качества программного обеспечения

Качество ПО — это комплекс характеристик, определяющих способность программного продукта выполнять возложенные на него функции. Для оценки качества используются различные критерии, которые могут варьироваться в зависимости от типа ПО и требований пользователей.

Согласно международному стандарту ISO/IEC 25010:2011, к основным характеристикам качества ПО относятся:

Функциональность — способность ПО выполнять требуемые функции и

соответствовать заявленным потребностям пользователей. Включает правильность работы, совместимость компонентов.

Надёжность — бесперебойное выполнение задач в заданных условиях в течение установленного времени. К атрибутам надёжности относятся безошибочность, правильность, устойчивость к ошибкам, безотказность, пригодность к восстановлению.

Юзабилити (удобство использования) — степень удобства ПО для пользователей, его наглядность, лёгкость эксплуатации и изучения. Включает понятность и лёгкость обучения.

Эффективность — степень обеспечения необходимой производительности при заданных условиях.

Удобство сопровождения — простота анализа, тестирования, коррекции компонентов ПО, его обслуживания, а также степень адаптации к новым условиям.

Портативность — степень лёгкости переноса ПО на другую платформу.

Совместимость — способность программных компонентов взаимодействовать друг с другом.

Защищённость — минимизация угроз, связанных с несанкционированным доступом к программам и данным.

Ранее использовался стандарт ГОСТ Р ИСО/МЭК 9126-93, который также выделял характеристики качества, включая функциональные возможности, надёжность, практичность, эффективность, сопровождаемость и мобильность.

Методы оценки качества ПО

Функциональное тестирование проверяет соответствие ПО функциональным требованиям.

Тестирование производительности оценивает эффективность работы системы под нагрузкой.

Интеграционное тестирование проверяет взаимодействие компонентов ПО.

Тестирование совместимости убеждается в корректной работе ПО в разных средах.

Юзабилити-тестирование оценивает удобство использования.

Анализ метрик качества — использование количественных показателей (например, количество ошибок на тысячу строк кода, время отклика системы).

Оценка удовлетворённости пользователей через опросы и обратную связь.

Эффективный анализ рисков и контроль характеристик качества позволяют снизить вероятность неудач при внедрении ПО, обеспечить его стабильную работу и соответствие ожиданиям пользователей.

П: Опишите риски и характеристики виртуальной машины с RedOS.

## Задание 1.6. Выполнение отдельных видов работ на этапе поддержки программного обеспечения компьютерной системы.

Т: Основные цели этапа поддержки: обеспечение стабильной работы программного продукта, устранение возникающих ошибок и недоработок, модернизация системы в соответствии с требованиями пользователей, поддержание актуальности программного обеспечения, обеспечение безопасности системы, виды работ на этапе поддержки.

Базовое сопровождение включает: обучение пользователей работе с системой, консультирование по вопросам использования по, настройка и адаптация системы под конкретное окружение, профилактика возможных сбоев, устранение критических ошибок, мониторинг работоспособности.

Расширенное сопровождение предусматривает: модификацию существующих сервисов, обновление баз данных, добавление нового функционала, оптимизацию производительности, интеграцию с другими системами, основные направления работ.

Техническая поддержка: обработка обращений пользователей, диагностика проблем, устранение неисправностей, восстановление работоспособности системы.

Модернизация системы: добавление новых функций, улучшение существующих возможностей, оптимизация производительности, обновление интерфейсов.

Обеспечение безопасности: установка обновлений безопасности, проверка на уязвимости, настройка систем защиты, резервное копирование, этапы выполнения работ.

Анализ проблемы: сбор информации о возникшей ситуации, определение причин возникновения проблемы, оценка влияния на работу системы.

Планирование решения: разработка плана действий, оценка необходимых ресурсов, согласование сроков выполнения.

Реализация: внесение изменений в код, тестирование исправлений, проверка работоспособности.

Документирование: фиксация выполненных работ, описание внесенных изменений, обновление документации, методы работы с ошибками.

Классификация ошибок: критические (требуют немедленного исправления), существенные (влияют на работу основных функций), незначительные (не влияют на основные функции).

Процесс работы с ошибками: регистрация ошибки, анализ причин возникновения, разработка решения, тестирование исправления, внедрение решения, закрытие ошибки, инструменты поддержки.

Основные инструменты: системы отслеживания ошибок, средства диагностики, инструменты профилирования, системы контроля версий, средства резервного копирования, организация процесса поддержки.

Структура службы поддержки: первая линия (прием обращений), вторая линия (техническая поддержка), третья линия (решение сложных технических проблем), контроль качества поддержки.

Показатели эффективности: время реакции на обращение, время решения проблемы, процент успешно решенных проблем, удовлетворенность пользователей, документация при поддержке.

Необходимые документы: журнал регистрации обращений, база знаний по решению типовых проблем, документация по внесенным изменениям, отчеты о выполненных работах.

Успешное выполнение работ на этапе поддержки требует четкой организации процесса, квалифицированного персонала и современных инструментов сопровождения. Это обеспечивает стабильную работу системы и удовлетворение потребностей пользователей.

П: Создайте алгоритм обучения пользователей по работе с виртуальной машины с RedOS.

# ТЕМА 2. ПРОЕКТИРОВАНИЕ ЧАСТИ ИНФОРМАЦИОННОЙ СИСТЕМЫ.

Проектирование информационных систем — это упорядоченная совокупность методологий и средств создания или модернизации информационных систем,

обеспечивающая эффективное решение задач пользователей.

Актуальность проектирования обусловлена следующими факторами:

постоянное развитие информационных технологий;

необходимость адаптации систем к меняющимся бизнес-процессам;

рост требований к производительности и безопасности;

потребность в масштабируемости систем;

важность оптимизации затрат на разработку и поддержку;

основные аспекты проектирования;

Ключевые направления проектирования включают:

проектирование объектов данных;

разработка программной составляющей;

создание пользовательского интерфейса;

учет технологической инфраструктуры;

обеспечение интеграции компонентов;

компоненты проектирования;

Основные элементы проектирования:

концептуальное проектирование — определение общей архитектуры и основных компонентов;

логическое проектирование — разработка детальной структуры данных и процессов;

физическое проектирование — реализация проектных решений на конкретных платформах;

методологические основы;

Современные подходы к проектированию базируются на:

использовании case-технологий;

применении унифицированного языка моделирования (uml);

методологиях структурного анализа;

объектно-ориентированном подходе;

agile-методологиях;

этапы проектирования;

типовые этапы проектирования:

анализ требований и определение границ проекта;

разработка концептуальной модели;

создание логической модели;

проектирование физической реализации;

документирование проектных решений;

верификация и валидация проекта;

требования к проектированию;

Основные требования к проектируемой части ИС:

функциональная полнота;

производительность;

надежность;

безопасность;

удобство использования;

масштабируемость;

совместимость с существующей инфраструктурой;

практическая значимость;

результаты освоения темы позволят:

разрабатывать проектные решения для информационных систем;

создавать техническую документацию;

оценивать эффективность проектных решений;

прогнозировать последствия изменений в системе;

обеспечивать качество разрабатываемых компонентов.

В процессе изучения темы студенты получат практические навыки проектирования, необходимые для дальнейшей профессиональной деятельности в области информационных технологий.

## Задание 2.1. Разработка модели бизнес-процессов информационной системы. Методология IDEF0.

Т: IDEF0 — методология функционального моделирования, предназначенная для создания детальных моделей бизнес-процессов и информационных систем. Является стандартом для проектирования и анализа сложных систем.

Основные характеристики IDEF0: высокий уровень детализации процессов, иерархический подход к отображению элементов, четкая структура документации, универсальность применения, основные элементы нотации, функциональный блок,

изображается прямоугольником, содержит название процесса (глагол или отглагольное существительное), имеет уникальный номер.

Стрелки (4 типа): входы (левая сторона) — данные или объекты, преобразуемые функцией, выходы (правая сторона) — результаты выполнения функции, управление (верхняя сторона) — правила и процедуры, механизмы (нижняя сторона) — ресурсы выполнения.

**Структура моделирования.**

Контекстная диаграмма (A-0): отображает систему как один блок, определяет границы моделирования, устанавливает связи с внешней средой.

Декомпозиция: разделение процессов на подпроцессы, иерархическая структура, детализация до необходимого уровня, правила построения модели, на одном уровне должно быть не менее 2 и не более 8 блоков, чтение диаграмм происходит сверху вниз и слева направо, блоки располагаются по диагонали от верхнего левого к нижнему правому углу.

**Этапы разработки модели.**

Подготовительный этап: определение целей моделирования, сбор информации, формирование команды.

Построение контекстной диаграммы: определение основной функции, выделение входов/выходов, указание управления и механизмов.

Декомпозиция: разделение процессов, детализация функций, проверка целостности модели.

Валидация модели: проверка корректности, согласование с экспертами, внесение корректировок, преимущества методологии, высокая детализация процессов, четкая структура документации, универсальность применения, наглядность представления, ограничения методологии, сложность освоения, трудоемкость создания детальных моделей, отсутствие механизмов обработки исключений, ограниченные возможности автоматизации, практическое применение.

Области использования: анализ и оптимизация бизнес-процессов, проектирование информационных систем, документирование процессов, обучение персонала, внедрение изменений, рекомендации по применению, начинать с построения контекстной диаграммы, соблюдать правила нумерации блоков, обеспечивать полноту описания процессов, поддерживать актуальность модели,

вовлекать экспертов предметной области.

Методология IDEF0 позволяет создать детальную модель бизнес-процессов информационной системы, обеспечивающую понимание взаимосвязей между функциями, данными и ресурсами. Это помогает в анализе, оптимизации и дальнейшем развитии системы.

Пример создания функциональной модели IDEF0.

Для того чтобы понять, как работать с функциональным моделированием, я приведу пример процесса написания статьи.

Основной блок – «Написать статью».

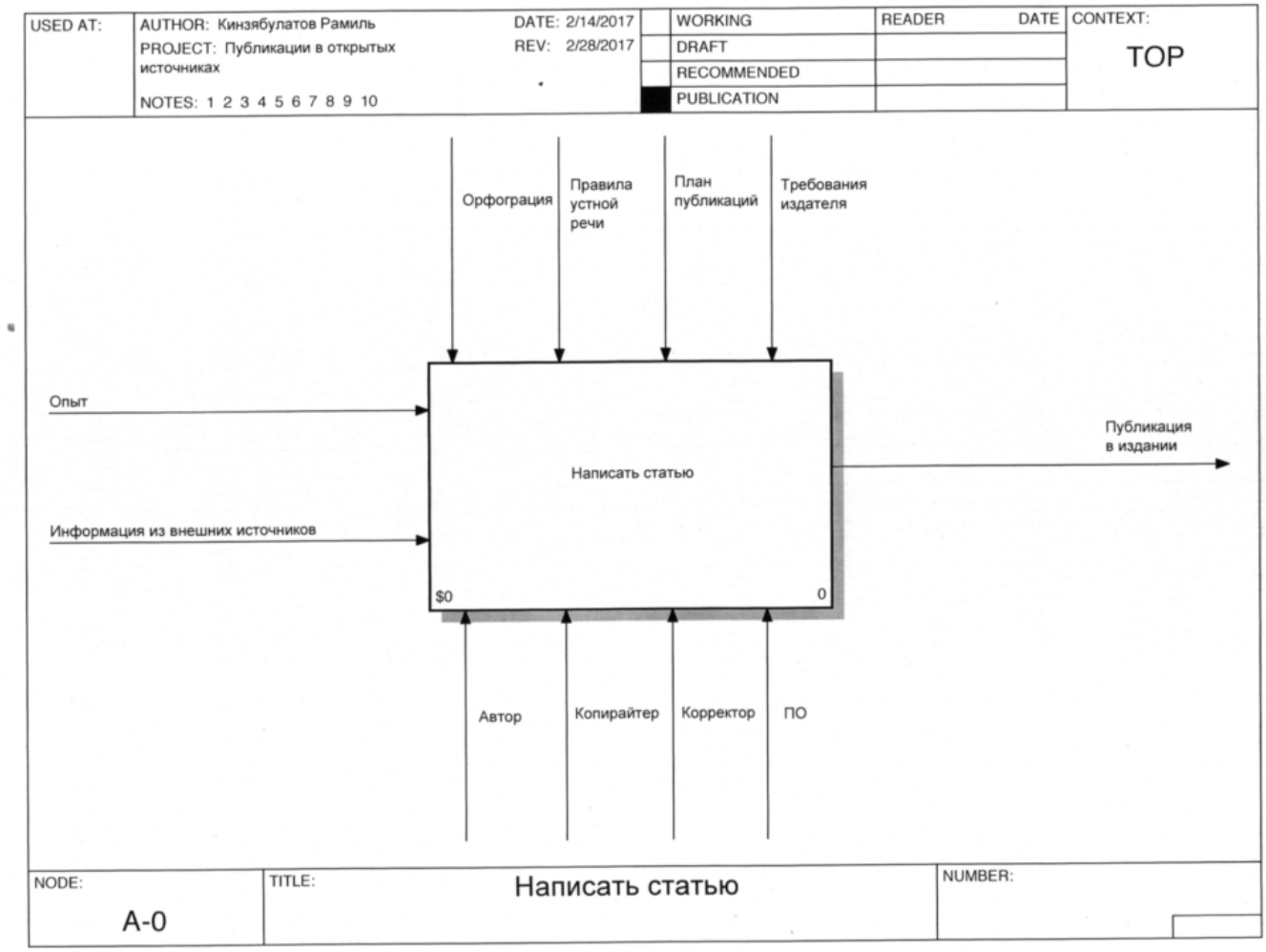



Рисунок 1.

Входящие стрелки – «Опыт», «Информация из сторонних источников». Это те вводные, которые необходимы для начала работы.

Управляющие для написания статьи – это «План публикации», «Требования издателя», «Правила русского языка».

А в роли «Механизмов» выступают автор, копирайтер, корректор и программное обеспечение. В данном случае автор создает аудиоматериал, в котором

собирает все мысли и идеи, которые должны быть отражены в статье. Копирайтер – это человек, который создает на основе этого материала, руководствуясь требованиями издателя, планом публикации и правилами русского языка, готовый текст статьи. Корректор проверяет материал на ошибки. А программное обеспечение – это те инструменты, которые используют в работе все участники процесса.

Таким образом, я задал основные параметры процесса, его вход, выход, а также все необходимое для успешного проведения процесса. Но это – только основные рамки процесса. Так описывается общая схема работы компании в целом.

На самом деле, процесс создания статьи, как и любой бизнес-процесс можно и нужно детализировать. Для этого я декомпозирую общий блок «написать статью» на связанные между собой элементы.

В нашем случае работа делится на 4 основных этапа:

Подготовить аудио.

Подготовить текст

Подготовить текст к публикации.

Разместить статью в издании.

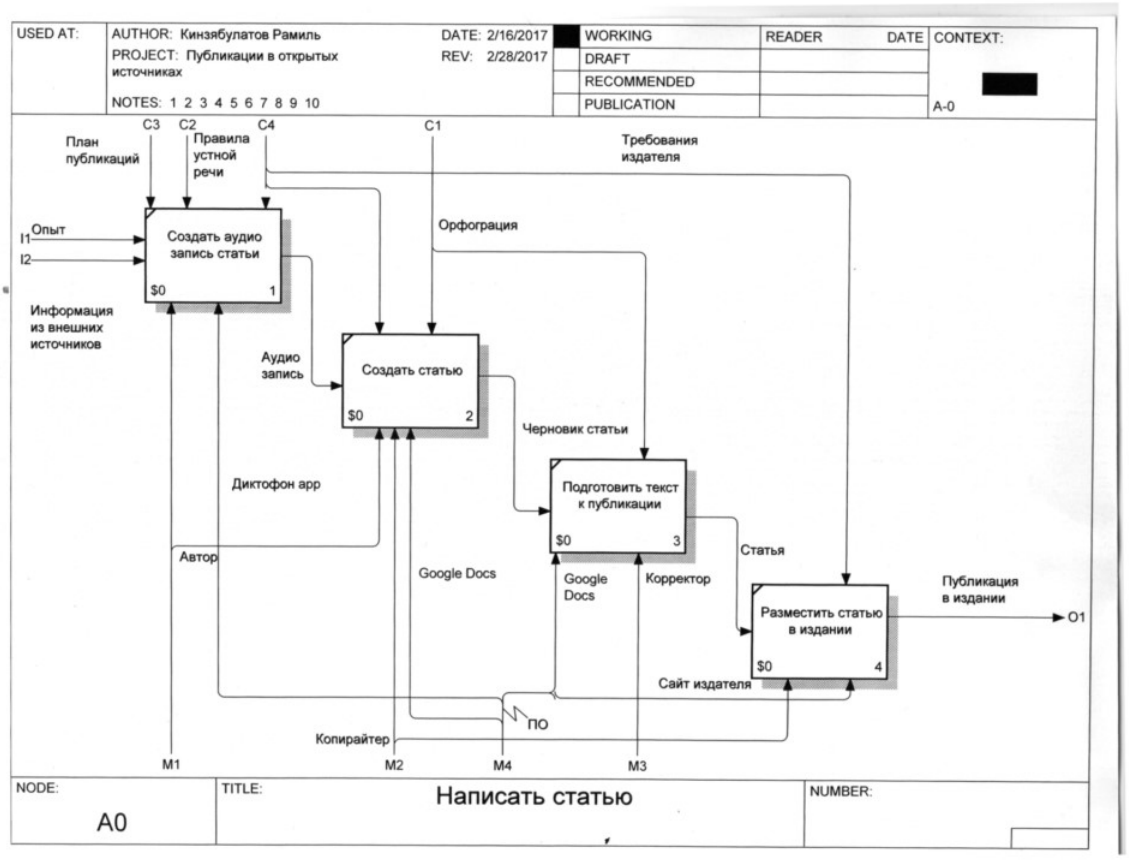



Рисунок 2.

На схеме наглядно видно, на каком этапе какие управляющие элементы и какие механизмы задействованы.

Так, автор при создании аудио использует свои знания и опыт, при этом руководствуется планом публикации и требованиями издателя. Копирайтер получает на входе аудиозапись, из которой, руководствуясь правилами русского языка, создает текст. Корректор получает текст и проверяет его, также руководствуясь правилами русского языка. Для размещения статьи в издании необходимо специальное программное обеспечение.

П: Разработать по выданному варианту IDEF0 модель.

## Задание 2.2. Разработка модели бизнес-процессов информационной системы. Методология DFD.

Т: DFD (Data Flow Diagrams) — методология моделирования информационных потоков в системе. Используется для визуализации процессов обработки данных и их движения между различными компонентами системы.

Базовые компоненты DFD :

Процесс — действие или функция, преобразующая входные данные в выходные;

Поток данных — информация, передающаяся между процессами, хранилищами и внешними сущностями;

Хранилище данных — место хранения данных;

Внешняя сущность — объект за пределами моделируемой системы.

**Уровни детализации DFD**

Контекстная диаграмма (уровень 0):

показывает систему как единый процесс, отображает основные внешние сущности, демонстрирует главные потоки данных.

Диаграмма уровня 1: детализирует контекстную диаграмму, разбивает основной процесс на подпроцессы, показывает основные потоки между ними.

Диаграмма уровня 2 и ниже: дальнейшая детализация процессов, более глубокое описание функций, конкретизация потоков данных, правила построения DFD, каждый процесс должен иметь уникальный номер, потоки данных должны быть именованы существительными, хранилища данных должны иметь как минимум один входной и один выходной поток, внешние сущности должны быть

связаны с процессами через потоки данных.

**Этапы моделирования.**

Подготовительный этап: определение границ системы, выявление основных процессов, сбор информации о потоках данных.

Создание контекстной диаграммы: определение главной функции, выделение внешних сущностей, построение основных потоков.

Детализация процессов: разбивка процессов на подпроцессы, описание потоков данных, создание хранилищ данных.

Валидация модели: проверка корректности связей, согласование с экспертами, внесение корректировок, преимущества методологии, наглядность представления потоков данных, возможность детализации процессов, четкое отображение взаимодействий, универсальность применения, поддержка различных уровней детализации, ограничения методологии, сложность отображения сложных логических условий, отсутствие временных характеристик, трудности при моделировании итеративных процессов, ограниченные возможности для описания поведения системы, практическое применение.

Области использования: анализ потоков данных в системе, проектирование информационных систем, документирование процессов, оптимизация бизнес-процессов, внедрение изменений, рекомендации по построению, начинать с создания контекстной диаграммы, соблюдать правила именования элементов, обеспечивать полноту описания процессов, поддерживать логическую последовательность, использовать понятные названия для потоков данных, инструменты моделирования.

Популярные средства:

MS Visio

ERWin Data Modeler

BPWin

Ramus

StarUML

Примеры использования.

Типичные сценарии: моделирование процессов обработки заказов, анализ документооборота, проектирование систем учета, оптимизация бизнес-процессов,

разработка информационных систем.

Методология DFD позволяет создать наглядную модель потоков данных в информационной системе, что помогает в анализе, проектировании и оптимизации бизнес-процессов организации.

Пример DFD диаграммы.

**1. Контекстный уровень.** На этом уровне специалист в общих чертах описывает системы и процессы. Как правило, вся система представлена как один процесс. Диаграммы контекстного уровня используются редко, прежде всего они нужны, чтобы в понятном виде презентовать заказчику проект.

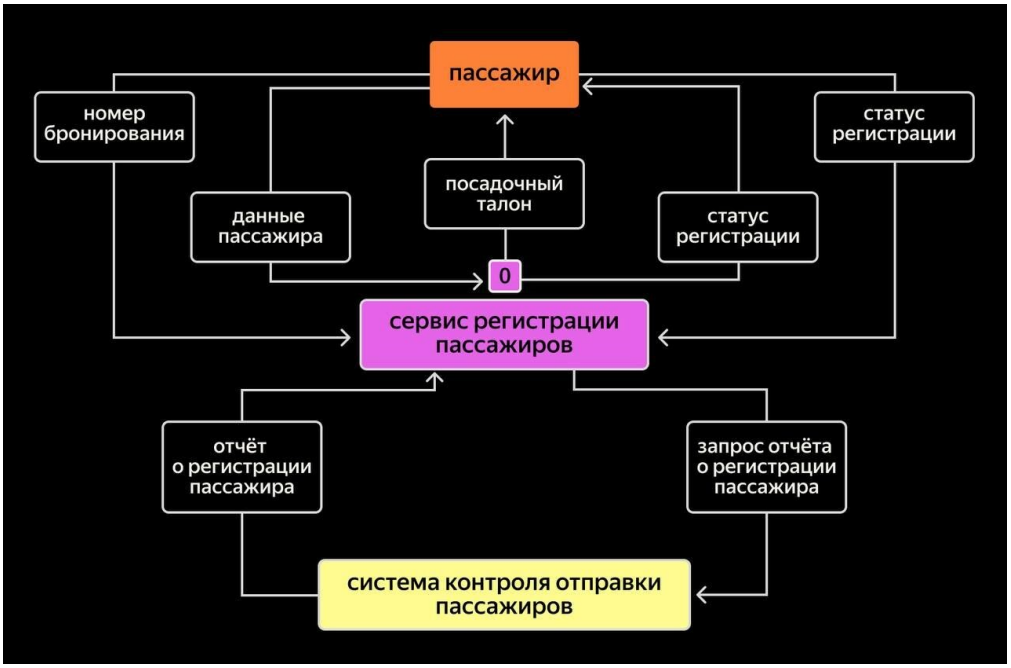



Рисунок 3

Контекстная диаграмма DFD — лёгкий способ начать работать с диаграммами, если есть страх белого листа

**2. Логический уровень.** Более подробно описывает процессы, которые происходят в системе, какие входящие и выходящие данные нужны для каждого из процессов.

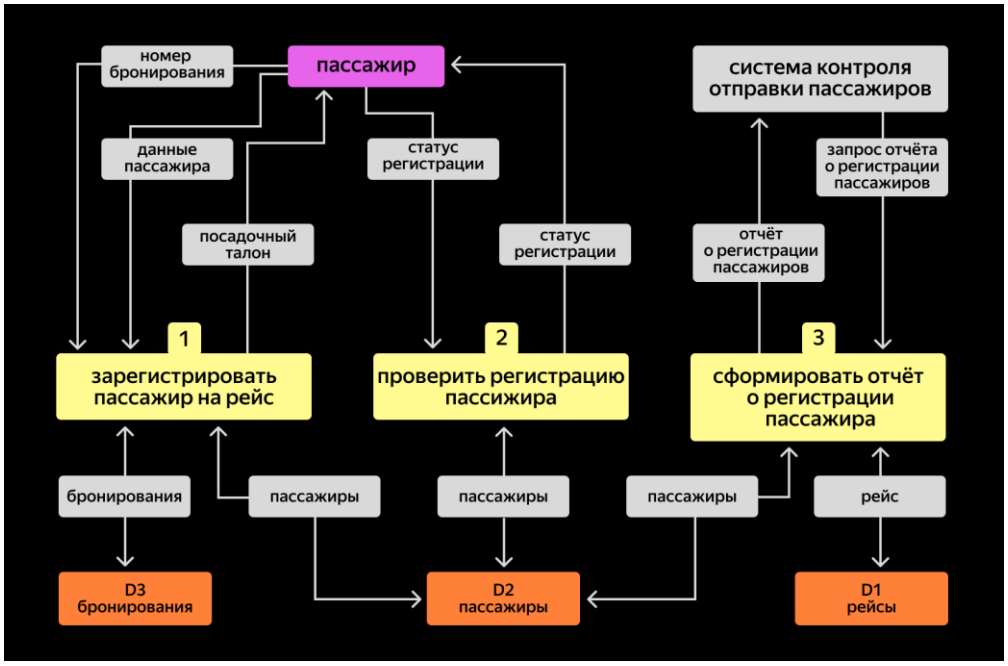



Рисунок 4

Если разбить обобщенный процесс контекстной диаграммы на подпроцессы, получится детализировать потоки данных

**3. Физический уровень.** Более детализированный логический уровень, где подробно раскрываются все входящие и выходящие данные, появляется подробное описание баз данных, которые используются в работе, и способ реализации всех элементов модели.

Если DFD-уровни становятся слишком сложными, вводятся слои. Они позволяют собрать выпадающие уровни и навести порядок в схеме. Аналитики работают со схемами на логическом уровне.

Разберём, как составить диаграмму потоков данных, которые участвуют в работе системы регистрации пассажиров на поезд:

1. Выделить сущности.
2. Нужно определить все сущности, которые используют систему.

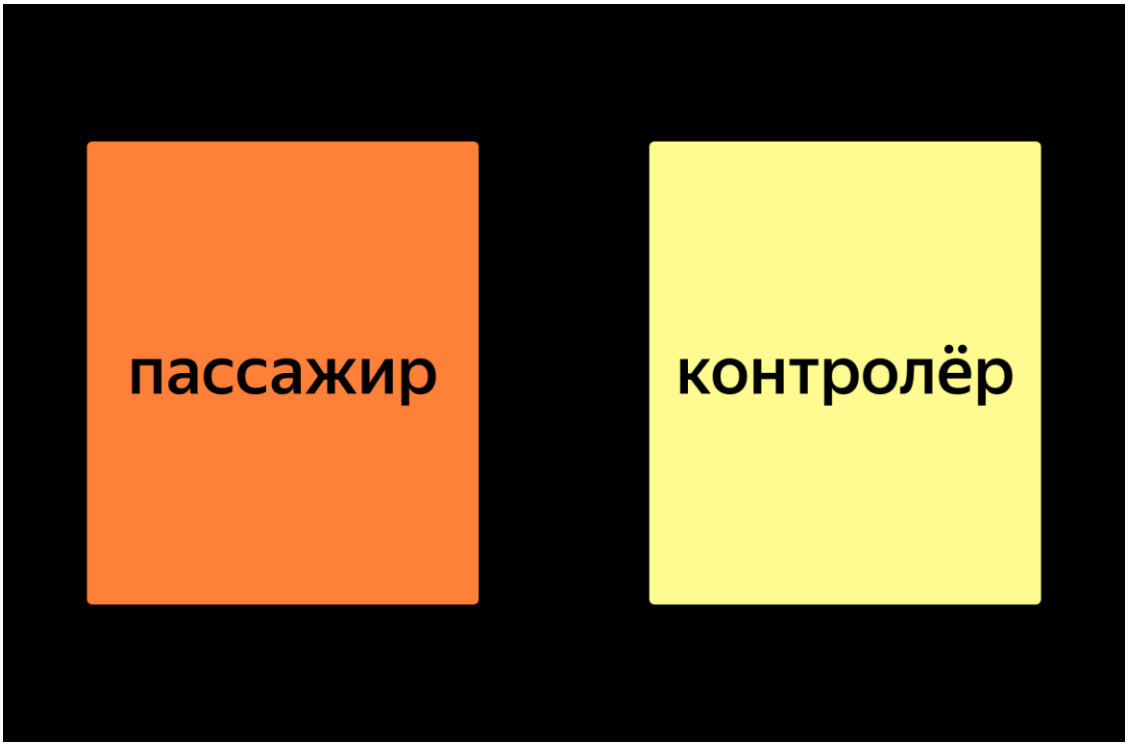



Рисунок 5

В этой предметной области есть две сущности, которые используют систему: пассажир и контролёр

**4. Определить процессы.**

Пассажиры используют систему, чтобы регистрироваться на рейс. А контролёры используют систему, чтобы проверить статус регистрации каждого пассажира.

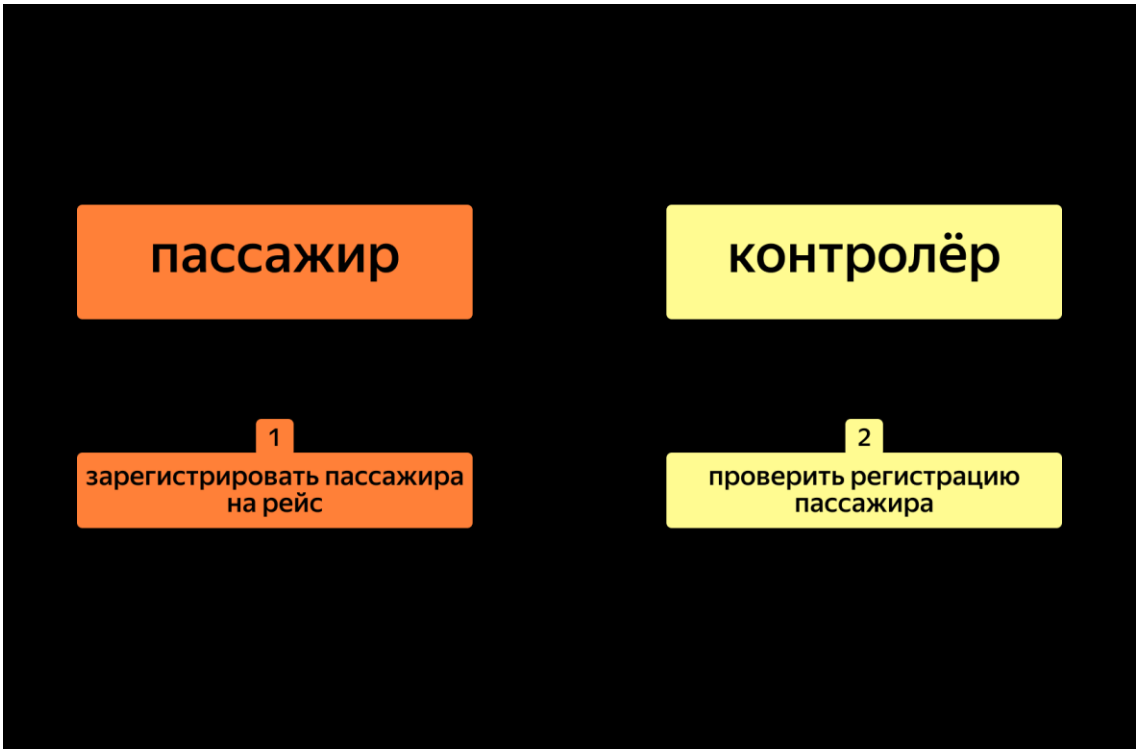



Рисунок 6

Система выполняет две функции (два процесса): «Зарегистрировать пассажира на рейс» и «Проверить регистрацию пассажира»

**5. Указать потоки данных между сущностями и процессами.** Для того чтобы зарегистрироваться на рейс, пассажир должен предоставить номер бронирования и свои данные. Например, ФИО, серию и номер паспорта. Стрелки с этими данными идут от сущности в процесс. После регистрации на рейс пассажир получает посадочный талон. Эти данные уже направляются от процесса к сущности.


Рисунок 7

Чтобы проверить регистрацию пассажира на рейс, контролёр должен внести его данные. А после проверки контролёр получит статус регистрации пассажира

**6. Определить хранилища данных и потоки к ним.**

Когда стало понятно, какие данные получает система на входе и какие данные она должна отдать на выходе, на диаграмму нужно добавить хранилища. В них будут направляться данные, там они будут храниться и извлекаться по требованию пользователя. Вся информация, которая нужна системе для выполнения двух процессов, сосредоточена в трёх хранилищах: «Бронирования», «Пассажиры» и «Рейсы».

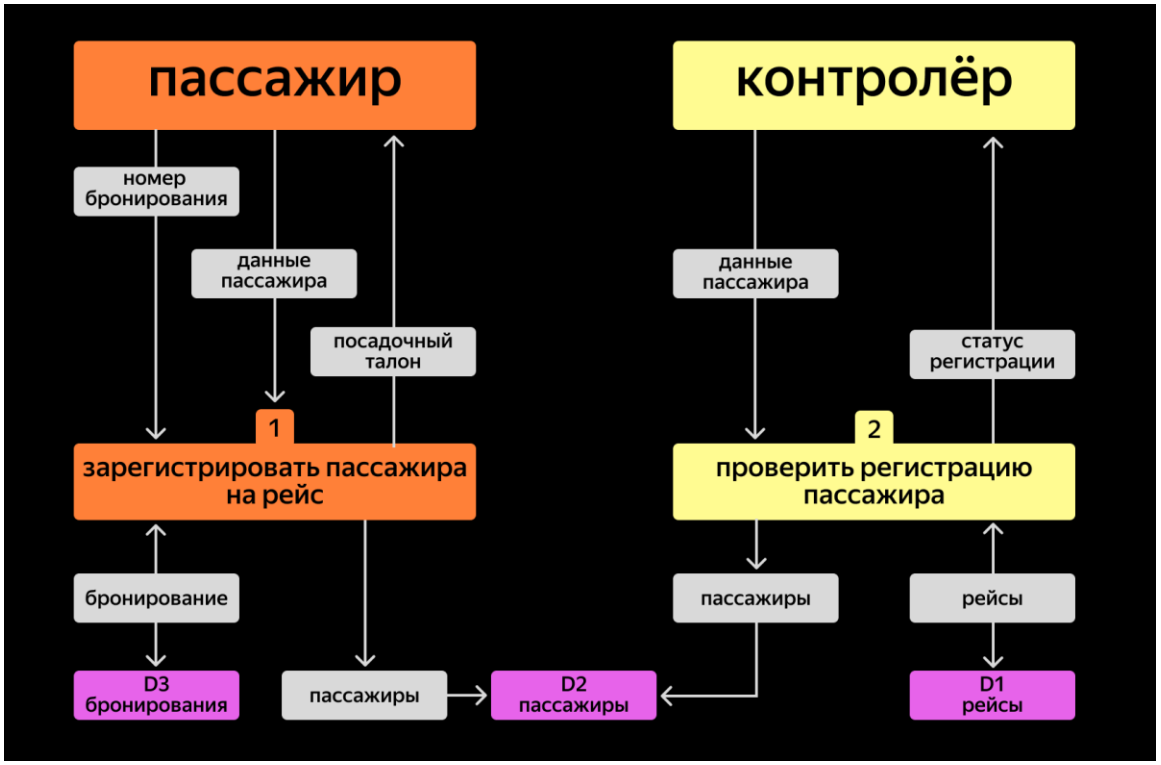



Рисунок 8

Когда система регистрирует пассажира на рейс, она получает номер бронирования и проверяет его наличие в хранилище «Бронирование». Данные пассажира передаются в хранилище «Пассажиры». Для проверки регистрации контролёр задействует то же самое хранилище — «Пассажиры»

П: Разработать по выданному варианту DFD модель.

## Задание 2.3. Построение диаграмм вариантов использования при проектировании ИС.

Т: Диаграмма вариантов использования (Use Case Diagram) — это инструмент UML (Unified Modeling Language), который визуализирует взаимодействие пользователей (акторов) с системой. Она описывает, какие функции системы доступны разным группам пользователей, и помогает определить границы системы, сформулировать требования к её функциональному поведению, а также обеспечить взаимопонимание между разработчиками и заказчиками.

Диаграмма вариантов использования (use case diagram) uml.

Основные элементы диаграммы

Актёр (Actor) — внешняя по отношению к системе сущность, которая взаимодействует с ней. Это может быть человек, другое приложение, устройство или система. На диаграмме актёр изображается в виде фигурки «человечка», под которой указывается его имя. Актёр представляет роль, а не конкретного человека

или устройство. Один физический пользователь может выступать в роли разных актёров в зависимости от контекста использования системы.

Прецедент (Use Case, вариант использования) — описание последовательности действий, выполняемых системой в ответ на событие, инициируемое актёром. Прецедент приводит к наблюдаемому результату, который важен для актёра. Изображается эллипсом, внутри которого указывается название в форме глагола с пояснительными словами (например, «Оформить заказ»).

Граница системы (System Boundary) — прямоугольник, который очерчивает границы системы. Внутри него размещаются прецеденты, а актёры располагаются снаружи.

Связи (отношения) — стрелки, которые показывают взаимодействие между актёрами и прецедентами, а также между прецедентами.

Диаграмма вариантов использования языка UML 2 - презентация онлайн

Типы отношений между элементами

Тип отношения Описание

Ассоциация Связь между актёром и прецедентом, показывающая, что актёр взаимодействует с этим прецедентом.

Обобщение (Generalization) Указывает, что один элемент (актёр или прецедент) является специальным случаем другого. Например, актёр «Сотрудник отдела продаж» может быть обобщён до актёра «Сотрудник».

Включение (Include) Указывает, что один прецедент обязательно включает в себя другой как составную часть. Например, прецедент «Сохранить документ» может включать «Выбрать формат сохранения».

Расширение (Extend) Указывает, что один прецедент может расширять другой при выполнении определённых условий. Например, прецедент «Оформить заказ» может расширяться прецедентом «Применить скидку», если выполняется условие наличия промокода.

Этапы построения диаграммы

Определение границ системы. Нужно чётко обозначить, что входит в систему, а что является внешним по отношению к ней. Это делается с помощью прямоугольника, обозначающего границу системы.

Идентификация актёров. Следует перечислить все внешние сущности,

которые будут взаимодействовать с системой. Для каждого актёра важно определить его роль.

Определение прецедентов. Нужно выявить основные функции системы, которые будут доступны актёрам. Каждый прецедент должен иметь чёткое название и описывать результат, который получит актёр.

Установление связей. Следует определить отношения между актёрами и прецедентами, а также между прецедентами (если есть включения, расширения или обобщения).

Проверка и валидация. Диаграмму нужно показать заказчикам и другим заинтересованным сторонам, чтобы убедиться, что она точно отражает требования к системе.

Рекомендации

Не перегружайте диаграмму деталями. На этом этапе не стоит углубляться в технические реализации — диаграмма должна давать общее представление о функционале.

Используйте понятные названия. Имена актёров и прецедентов должны быть краткими и отражать суть.

Учитывайте разные типы актёров. Различают первичных (инициируют взаимодействие с системой) и вторичных (реагируют на запросы системы) актёров.

Проверяйте целостность модели. Убедитесь, что каждый прецедент имеет хотя бы одного актёра, а отношения между элементами корректны.

Инструменты для создания диаграмм

Для построения диаграмм вариантов использования можно использовать:

Специализированные UML-инструменты, например, StarUML, Visual Paradigm.

Онлайн-сервисы, такие как Lucidchart, Draw.io (diagrams.net), Creately.

Microsoft Visio — часть пакета Office, поддерживает создание различных типов диаграмм, включая Use Case.

Видео с подробным объяснением построения диаграммы вариантов использования:

Диаграмма вариантов использования — важный инструмент на этапе сбора требований и анализа системы. Она помогает сформировать общее представление о

функционале, упростить коммуникацию между участниками проекта и заложить основу для дальнейшей детализации системы.

Пример:


Основные элементы диаграммы - участник (actor) и прецедент (вариант).

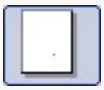
**Участник**- это множество логически связанных ролей, исполняемых при взаимодействии с прецедентами или сущностями (система, подсистема или класс). Участником может быть человек или другая система, подсистема или класс, которые представляют нечто вне сущности. Графически участник изображается “человечком”.

**Прецедент (use case)**- описание множества последовательных событий (включая варианты), выполняемых системой, которые приводят к наблюдаемому участником результату. Прецедент представляет поведение сущности, описывая взаимодействие между участниками и системой. Прецедент не показывает, “как” достигается некоторый результат, а только “что” именно выполняется. Прецеденты обозначаются очень простым образом - в виде эллипса, внутри которого указано его название.

Основные элементы диаграммы вариантов использования

На диаграмме вариантов использования можно отобразить следующие элементы нотации UML, доступные в панели элементов:

| **Элемент/Нотация** | **Предназначение** |
|---|---|
|  | Участник (Actor) |

| Элемент/Нотация | Предназначение |
|---|---|
|  | Вариант (Use case) |
|  | Граница (Boundary |
|  | Ненаправленная ассоциация (Undirected communication association) |
|  | Направленная ассоциация (Directed communication association) |
|  | Обобщение (Generalization) |
|  | Зависимость (Dependency) |
|  | Точка изгиба связей (Point) |
|  | Комментарий (Note) |
|  | Коннектор комментария (Note connector) |

П: Разработать по выданному варианту диаграмму вариантов использования.

## Задание 2.4. Построение диаграмм деятельности для описания поведения разрабатываемой системы.

Т: Диаграмма деятельности (Activity Diagram) в UML — это поведенческая диаграмма, которая визуализирует поток управления и данных между действиями в системе. Она показывает, как процесс развивается от начала до конца, включая последовательные, параллельные и условные ветви выполнения. Используется для описания логики работы функций, бизнес-процессов, алгоритмов и взаимодействия компонентов системы.

Основные элементы диаграммы деятельности

Элемент Описание Графическое представление

Начальный узел (Initial Node) Обозначает начало выполнения процесса.

Заполненный чёрный круг.

Действие (Action)Конкретное действие или задача в процессе.

Прямоугольник с закруглёнными углами.

Поток управления (Control Flow) Показывает направление выполнения процесса. Стрелка.

Решение/ветвление (Decision Node) Точка выбора между несколькими вариантами развития процесса. Ромб. Внутри или рядом с ним указываются условия ветвления.

Объединение (Merge Node) Слияние нескольких веток в одну. Позволяет вернуться к общему сценарию после выполнения альтернативных путей. Ромб.

Параллельное выполнение (Fork Node) Разделяет процесс на несколько параллельных действий. Линия, из которой выходят несколько стрелок.

Конечный узел (Final Node) Обозначает окончание процесса. Чёрный круг с обводкой.

Дорожки (Swimlanes) Используются для группировки действий, выполняемых разными участниками или подразделениями. Помогают распределить ответственность. Горизонтальные или вертикальные полосы, разделяющие диаграмму на секции.

Узел передачи сигнала (Send Signal) Действие, которое создаёт экземпляр сигнала и передаёт его объекту-получателю. Специальный символ.

Узел приёма события (Receive Signal) Ожидает наступления определённого события. Специальный символ.

Диаграмма деятельности Uml FBD

Особенности построения

Диаграмма должна иметь один начальный узел и один или несколько конечных узлов.

Действия следуют сверху вниз, начальное состояние располагается в верхней части диаграммы, конечное — в нижней.

Для параллельных потоков используются точки разделения (Fork Node) и точки слияния (Join Node). Из точки разделения выходят несколько потоков, выполняющихся параллельно, а точка слияния синхронизирует их.

При ветвлении (Decision Node) в точку ветвления входит один переход, а

выходит два или более. Для каждого исходящего перехода задаётся булевское выражение (условие), которое вычисляется при входе в точку ветвления.

Дорожки (Swimlanes) позволяют распределить действия по участникам или подразделениям. Каждая деятельность принадлежит ровно одной дорожке, но переходы могут пересекать их границы.

Этапы построения диаграммы деятельности

Определение процесса. Чётко сформулируйте, какой процесс или прецедент вы моделируете. Диаграмма деятельности часто используется для детализации шагов, которые система выполняет после инициирования прецедента.

Идентификация действий. Перечислите все действия, которые входят в процесс. Каждое действие изображается прямоугольником с закруглёнными углами.

Определение последовательности. Установите порядок выполнения действий. Используйте потоки управления (стрелки) для отображения последовательности.

Добавление ветвлений и условий. Если процесс включает условия, добавьте узлы решения (ромбы) с указанием условий. Убедитесь, что условия взаимоисключающие и покрывают все возможные варианты.

Учёт параллельных потоков. Если есть действия, выполняющиеся одновременно, используйте точки разделения (Fork Node) и слияния (Join Node).

Распределение по дорожкам (при необходимости). Если нужно показать, кто выполняет те или иные действия, разделите диаграмму на дорожки (Swimlanes).

Добавление конечных узлов. Обозначьте завершение процесса конечными узлами.

Проверка и валидация. Убедитесь, что диаграмма логична, не содержит зависших подпроцессов, незавершённых ветвлений или циклов. Покажите диаграмму заинтересованным сторонам для обратной связи.

Примеры использования

Моделирование бизнес-процессов. Например, процесс оформления заказа в интернет-магазине: выбор товаров, добавление в корзину, оформление доставки, оплата.

Описание алгоритмов. Диаграмма может визуализировать логику работы функции, например, алгоритм проверки корректности данных.

Анализ сценариев использования. Диаграмма помогает детализировать шаги,

которые система выполняет при инициировании прецедента.

Рекомендации

Используйте коннекторы (полые кружки с названием внутри), если нужно упростить диаграмму, избегая загромождения связями.

Для отображения прерывания процесса (например, отмены пользователем) можно использовать Interruptible Edge — область со штриховой связью, прерывающее событие и связь-молнию.

Старайтесь не смешивать понятия состояния и действия. Состояние отражает этап в жизненном цикле объекта, а действие — конкретный шаг в процессе.

Диаграммы деятельности — гибкий инструмент, который помогает сделать сложные процессы более понятными как для разработчиков, так и для заказчиков. Они особенно полезны при работе с системами, где важно учитывать параллельные потоки и условные ветвления.

По существу, эта диаграмма представляет собой блок-схему, которая наглядно показывает, как поток управления переходит от одной деятельности к другой.

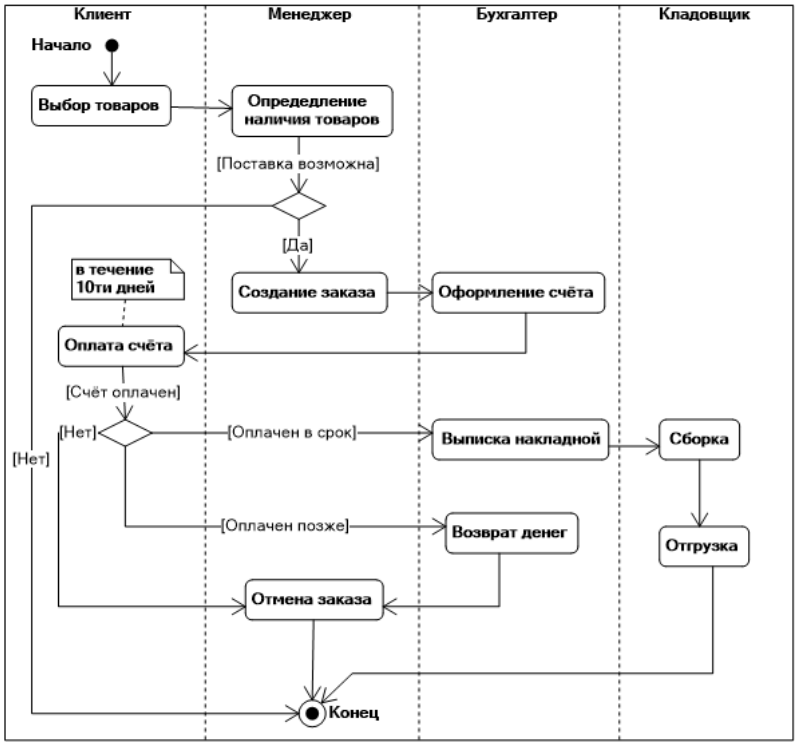


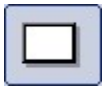

Рисунок 9

Активности на диаграмме “разбросаны” по беговым дорожкам, каждая из которых соответствует поведению одного из объектов (например, клиента, менеджера, веб-сервера, сервера БД и т.п.). Благодаря этому легко определить,

каким из объектов выполняется каждая из активностей. Дорожка - часть области диаграммы деятельности, на которой отображаются только те активности, за которые отвечает конкретный объект. Предназначены дорожки для разбиения диаграммы в соответствии с распределением ответственности за действия. Имя дорожки может означать роль или объект, которому она соответствует.

Основные элементы диаграммы активностей

На диаграмме активностей можно отобразить следующие элементы нотации UML, доступные в панели элементов:

| Элемент/Нотация | Предназначение |
|---|---|
|  | Принятие решения (Decision) |
|  | Активное состояние (Active state) |
|  | Начальное состояние (Start state) |
|  | Конечное состояние (Final state) |
|  | Синхронизатор/разветвитель (Complex transition) |
|  | Объект в состоянии (Object in state) |
|  | Получение сигнала (Signal receipt) |
|  | Отправка сигнала (Signal sending) |
|  | Переход (Transition) (Object in state) |

| Элемент/Нотация | Предназначение |
|---|---|
|  | Изменение объекта (Object flow) |
|  | Раздел (Partition) |
|  | Разделитель плавательных дорожек (Swimlane separator) |
|  | Точка изгиба связей (Point) |
|  | Комментарий (Note) |
|  | Коннектор комментария (Note connector) |

П: Разработать по выданному варианту диаграмму деятельности.

## Задание 2.5. Построение диаграмм последовательности для описания разрабатываемой системы.

Т: Диаграмма последовательности (Sequence Diagram) в UML — это поведенческая диаграмма, которая визуализирует взаимодействие между объектами системы во времени. Она показывает порядок обмена сообщениями между объектами, их жизненный цикл и хронологию событий в рамках конкретного сценария использования (прецедента). Используется для моделирования взаимодействий между компонентами системы, описания интеграций, анализа сценариев обработки запросов и других динамических процессов.

Как нарисовать UML-диаграмму последовательности: простой пример

Основные элементы диаграммы последовательности

Элемент Описание Графическое представление

Объект Участник взаимодействия (пользователь, компонент системы, внешний сервис). Изображается прямоугольником с названием. Может быть

актором (Actor), границей (Boundary), контроллером (Control) или сущностью (Entity). Прямоугольник с именем объекта или класса (например, «Пользователь», «Сервер», «База данных»).

Линия жизни (Lifeline) Вертикальная линия, представляющая период существования и активности объекта. Начинается с появления объекта на диаграмме и заканчивается его удалением или завершением взаимодействия.Пунктирная вертикальная линия, от которой отходят сообщения.

docs.system-analyst-base.ru +1

Сообщение Взаимодействие между объектами в виде передачи данных или вызова метода. Изображается горизонтальной стрелкой. Стрелка, направленная от отправителя к получателю. Может быть синхронной, асинхронной, возвратной и др..

Активация (Activation Bar) Узкий прямоугольник на линии жизни, показывающий период активности объекта при обработке сообщения. Прямоугольник на линии жизни, ограниченный сверху и снизу, обозначающий длительность выполнения действия.

Фреймы (комбинированные фрагменты) Используются для группировки сообщений и описания условных конструкций, циклов, параллельных процессов. Прямоугольная рамка с оператором в левом верхнем углу (например, alt, opt, loop, par).

Тип сообщения Описание Графическое представление

Синхронное Отправитель передаёт управление получателю и ждёт завершения действия. Пока получатель не обработает сообщение, отправитель не может выполнять другие действия. Сплошная линия со стрелкой в виде закрашенного треугольника.

Асинхронное Отправитель передаёт сообщение и сразу может выполнять другие действия, не дожидаясь ответа. Сплошная линия с открытой стрелкой.

Возвратное (ответное) Ответ на синхронное сообщение, обычно содержит возвращаемое значение. Пунктирная линия с открытой стрелкой.

Самосообщение Сообщение, которое объект отправляет самому себе (например, рекурсивный вызов метода). Стрелка, начинающаяся и заканчивающаяся на одной линии жизни.

Как построить диаграмму последовательности

Определите сценарий использования (прецедент). Диаграмма строится для конкретного сценария, например, «Авторизация пользователя», «Обработка заказа», «Интеграция с внешним сервисом».

Идентифицируйте объекты. Перечислите все объекты, которые участвуют во взаимодействии (пользователь, компоненты системы, внешние сервисы и т. д.). Расположите их горизонтально в порядке участия в процессе, объект-инициатор обычно помещают слева.

Нарисуйте линии жизни. Для каждого объекта проведите вертикальную пунктирную линию, обозначающую его существование во времени.

Добавьте сообщения. Изобразите обмен сообщениями между объектами в виде горизонтальных стрелок. Учитывайте порядок событий: сообщения располагаются сверху вниз в хронологической последовательности.

Укажите активации. На линиях жизни отметьте периоды активности объектов при обработке сообщений в виде узких прямоугольников.

Используйте фреймы для сложных сценариев. Если есть условные ветвления, циклы, параллельные процессы, заключите соответствующие группы сообщений в фреймы с операторами (alt, opt, loop, par и др.).

Добавьте комментарии и примечания при необходимости для пояснения деталей взаимодействия.

Рекомендации

Упрощайте диаграмму. Не перегружайте её мелкими деталями — это затруднит чтение.

Соблюдайте хронологию. Сообщения должны располагаться строго сверху вниз в порядке их возникновения.

Проверяйте согласованность. Убедитесь, что диаграмма соответствует архитектуре системы и не содержит противоречий.

Используйте инструменты. Для создания диаграмм можно использовать графические редакторы (например, draw.io), специализированные инструменты (PlantUML) или плагины в IDE.

Диаграммы последовательности особенно полезны при проектировании интеграций, анализе взаимодействий в распределённых системах и

документировании сложных бизнес-процессов. Они помогают визуализировать динамику системы и упростить коммуникацию между разработчиками, аналитиками и другими участниками проекта.

Пример:

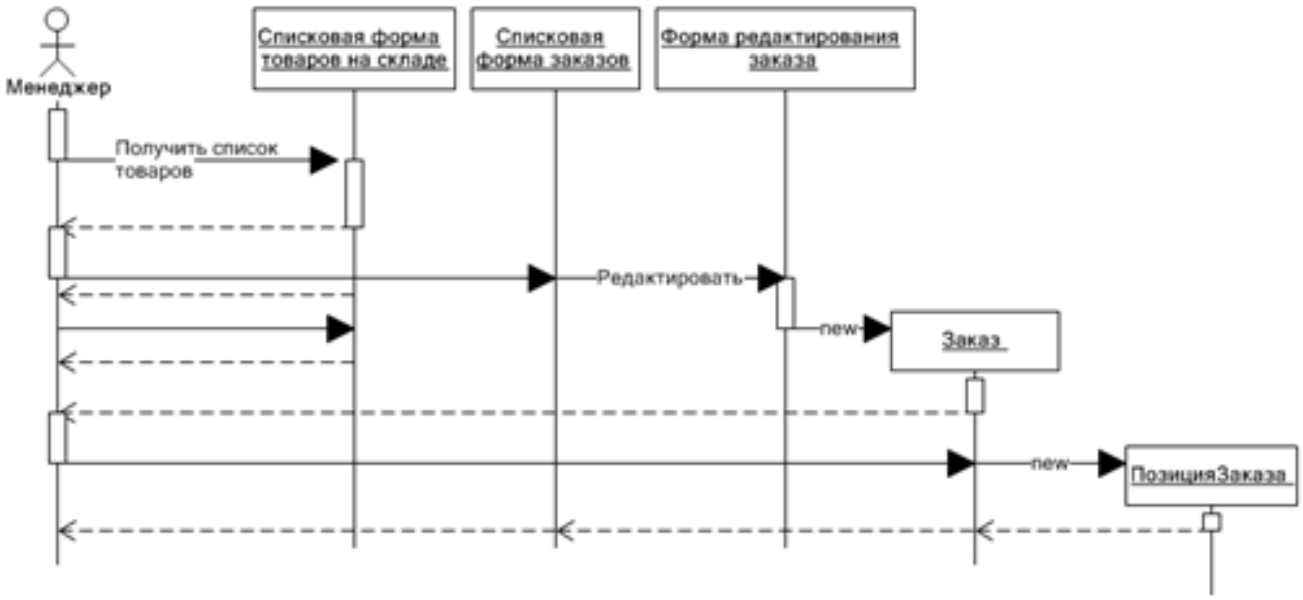



Рисунок 10

**Диаграммы последовательностей используются для уточнения диаграмм прецедентов**, более детального описания логики сценариев использования. Это отличное средство документирования проекта с точки зрения сценариев использования!

Диаграммы последовательностей обычно содержат **объекты**, которые **взаимодействуют в рамках сценария**, **сообщения**, которыми они обмениваются, и **возвращаемые результаты**, связанные с сообщениями. Впрочем, часто возвращаемые результаты обозначают лишь в том случае, если это не очевидно из контекста.

**Объекты** обозначаются прямоугольниками с подчеркнутыми именами (чтобы отличить их от классов).

**Сообщения (вызовы методов)** - линиями со стрелками.

**Возвращаемые результаты** - пунктирными линиями со стрелками.

Прямоугольники на вертикальных линиях под каждым из объектов показывают **“время жизни” (фокус) объектов**. Впрочем, довольно часто их не изображают на диаграмме, все это зависит от индивидуального стиля проектирования.

Основные элементы диаграммы последовательности

На диаграмме последовательности можно отобразить следующие элементы нотации UML, доступные в панели элементов:

| Элемент/Нотация | Предназначение |
|---|---|
|  | Участник (Actor) |
|  | Объект (Object) |
|  | Активный объект (Active object) |
|  | Терминатор (Terminator) |
|  | Вызов процедуры (Procedure call) |
|  | Сообщение (Flat message) |
|  | Асинхронное сообщение (Async message) |
|  | Сообщение с результатом (Return message) |
|  | Временной интервал (In scope) |
|  | Временное ограничение (Time constraint) |
|  | Точка изгиба связей (Point) |
|  | Комментарий (Note) |
|  | Коннектор комментария (Note connector) |

П: Разработать по выданному варианту диаграмму последовательности.

## Задание 2.6. Разработка состава требований к проектируемой ИС.

Т: Сущность требований к ИС

Требования к ИС — это документированные характеристики, функции и ограничения, определяющие работу системы, решаемые задачи и условия функционирования.

**Основные этапы разработки требований.**

Подготовительный этап: определение целей создания ИС, анализ текущей ситуации, выявление проблем и потребностей, определение границ проекта.

Сбор требований: опрос заинтересованных сторон, анализ документации, изучение бизнес-процессов, выявление ограничений.

Формализация требований: структурирование полученной информации, описание функциональных возможностей, определение нефункциональных характеристик, документирование результатов, категории требований.

Функциональные требования: описание основных функций системы, алгоритмы обработки данных, бизнес-процессы, интерфейсы взаимодействия, интеграции с другими системами.

Нефункциональные требования: производительность, надёжность, безопасность, удобство использования, масштабируемость, совместимость, методы сбора требований, интервьюирование пользователей и экспертов, анкетирование заинтересованных сторон, наблюдение за рабочими процессами, анализ документации существующей системы, мозговой штурм с командой проекта.

**Структура документа требований.**

Введение: цели и задачи проекта, определения и сокращения, ссылки на документы.

Общее описание: обзор системы, функциональное назначение, характеристики, ограничения, детальные требования:, функциональные требования, нефункциональные требования, требования к интерфейсам, требования к надёжности, требования к безопасности, процесс управления требованиями, валидация требований, верификация соответствия, отслеживание изменений, управление приоритетами, документирование решений, критерии качества требований, полнота — охват всех необходимых функций, однозначность —

отсутствие двусмысленности, проверяемость — возможность верификации, непротиворечивость — отсутствие конфликтов, реализуемость — техническая возможность выполнения, документация требований.

Основные документы: техническое задание, спецификация требований, пользовательская документация, тестовые сценарии, план внедрения, риски при разработке требований, неполное выявление потребностей, некорректная формулировка требований, изменение требований в процессе разработки, отсутствие согласования с заинтересованными сторонами, несоответствие реальным потребностям, рекомендации по работе с требованиями, регулярное обновление документации, вовлечение пользователей в процесс, документирование всех изменений, контроль версий требований, проверка на соответствие целям проекта.

Успешная разработка требований — ключевой фактор создания эффективной информационной системы, отвечающей потребностям пользователей и решающей поставленные задачи.

П: Разработать по выданному варианту состав требований к ИС.

## Задание 2.7. Разработка логической и физической модели данных информационной системы.

Т: Моделирование данных — это процесс создания формального описания структуры данных, необходимых для функционирования информационной системы.

**Этапы моделирования данных.**

Концептуальное моделирование: анализ предметной области, выявление сущностей и их взаимосвязей, создание общей модели без привязки к конкретной СУБД, логическое моделирование, детализация концептуальной модели, определение атрибутов сущностей, установление ограничений целостности.

Физическое моделирование: реализация логической модели в конкретной СУБД, определение типов данных, создание индексов и ограничений.

Логическая модель данных: сущности — основные объекты предметной области, атрибуты — характеристики сущностей, связи — отношения между сущностями, ограничения целостности — правила, которым должны соответствовать данные.

Элементы логической модели: сущность — набор однотипных объектов, экземпляр сущности — конкретный объект, атрибут — характеристика сущности,

ключ — уникальный идентификатор записи, связь — ассоциация между сущностями.

Компоненты физической модели:, таблицы и их структура, типы данных полей, индексы, ограничения, физическое размещение данных.

**Процесс разработки моделей.**

Анализ требований: определение целей системы, выявление информационных потребностей, сбор данных о предметной области.

Проектирование логической модели: создание ER-диаграммы, определение сущностей и атрибутов, установление связей, нормализация данных.

Проектирование физической модели: выбор СУБД, определение типов данных, проектирование файловой структуры, оптимизация производительности.

Инструменты моделирования: CASE-средства для проектирования баз данных, ER-диаграммы для визуализации моделей, UML-инструменты для объектного моделирования, специализированные редакторы схем БД.

**Требования к моделям данных.**

Основные характеристики: целостность данных — корректность и непротиворечивость, эффективность — оптимальное использование ресурсов, масштабируемость — возможность расширения, безопасность — защита данных, документация моделей.

Необходимые документы: описание логической модели, спецификация физической модели, словарь данных, описание ограничений целостности, документация по нормализации, проверка моделей.

Этапы верификации: проверка корректности связей, анализ производительности, тестирование целостности данных, оценка масштабируемости, проверка безопасности, оптимизация моделей, методы оптимизации:, нормализация данных, создание индексов, оптимизация запросов, сжатие данных, разделение данных, типичные ошибки, избыточность данных, нарушение целостности, неэффективные связи, отсутствие индексации, неправильная нормализация.

Рекомендации: регулярное обновление моделей, документирование изменений, тестирование на реальных данных, оптимизация под конкретные задачи, соблюдение стандартов проектирования.

Успешная разработка моделей данных — ключевой фактор создания

эффективной информационной системы, обеспечивающей надежное хранение и обработку данных.

Пример:

Концептуальная модель хранилища данных представляет собой описание главных (основных) сущностей и отношений между ними. Концептуальная модель является отражением предметных областей, в рамках которых планируется построение хранилища данных.

При проектировании концептуальной модели структурируют данные и выявляют взаимосвязи между ними, без рассмотрения особенностей реализации и вопросов эффективности обработки.

Для разработки концептуальной модели системы нужно выделить информационные объекты. В нашем случае это:

- номера;
- типы номеров;
- клиенты;
- типы клиентов;
- бронирование;
- проживание;
- выезд;
- пользователь.

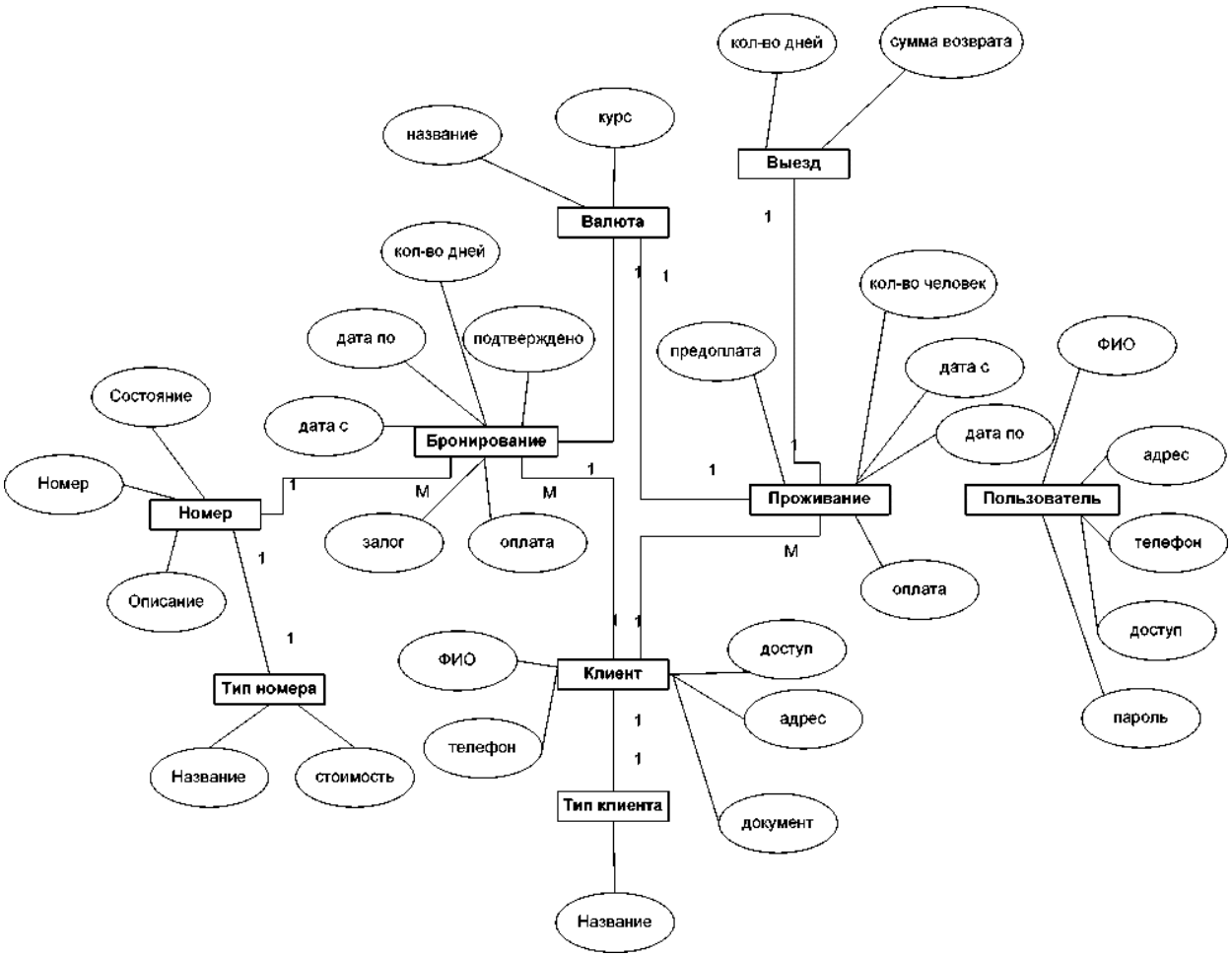



Рисунок 11. Концептуальная модель АРМ

В схемах в квадратах отражены сущности, а в овалах - атрибуты (рисунок 1). Связи между сущностями:

- Номер <относится> к Типу номера, связь 1 к 1;
- Клиент <относится> к Типу клиента, связь 1 к 1;
- Бронирование <относится> к Номеру, связь М к 1;
- Проживание <относится> к Номеру, связь М к 1;
- Бронирование <оформляется> на Клиента, связь М к 1;
- Проживание <оформляется> на Клиента, связь М к 1;
- Выезд <относится> к Проживанию, связь 1 к 1;
- Бронирование <оплачивается> в Валюте, связь 1 к 1;
- Проживание <оплачивается> в Валюте, связь 1 к 1.

Логическая модель расширяет концептуальную путем определения для сущностей их атрибутов, описаний и ограничений, уточняет состав сущностей и взаимосвязи между ними.

Концептуальная модель изменяется так, чтобы она могла быть обеспечена конкретной моделью данных.

В результате формируется логическая модель.

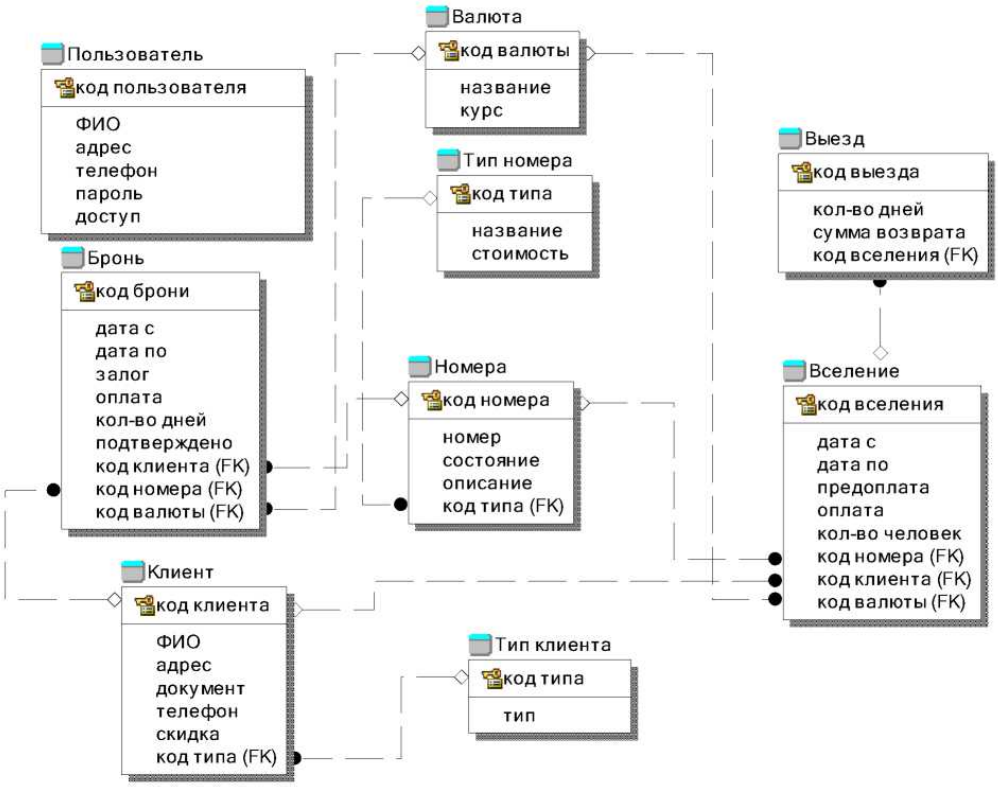



Рисунок 12. Логическая модель АРМ

Логическая модель отражает логические связи между элементами данных вне зависимости от их содержания и среды хранения.

Логическая модель может быть реляционной, иерархической или сетевой.

В качестве способа организации информационной базы выбрана реляционная база данных. Именно такой способ хранения всех данных является наиболее подходящим для проектируемого АРМ по следующим причинам:

- наглядность модели для пользователя: все данные в реляционной модели представлены в табличной форме;
- независимость данных от программного продукта для их обработки;
- реляционные базы данных являются наиболее распространенными среди разработчиков ПО, следовательно, использование этих баз позволит сэкономить время и бюджет на внедрение нового типа БД.

Физические модели данных служат для отображения моделей данных. Основными понятиями модели данных являются поле, логическая запись, логический файл. Слово "логический" введено, чтобы отличать понятия, относящиеся к логической модели данных, от понятий, относящихся к физической модели данных.

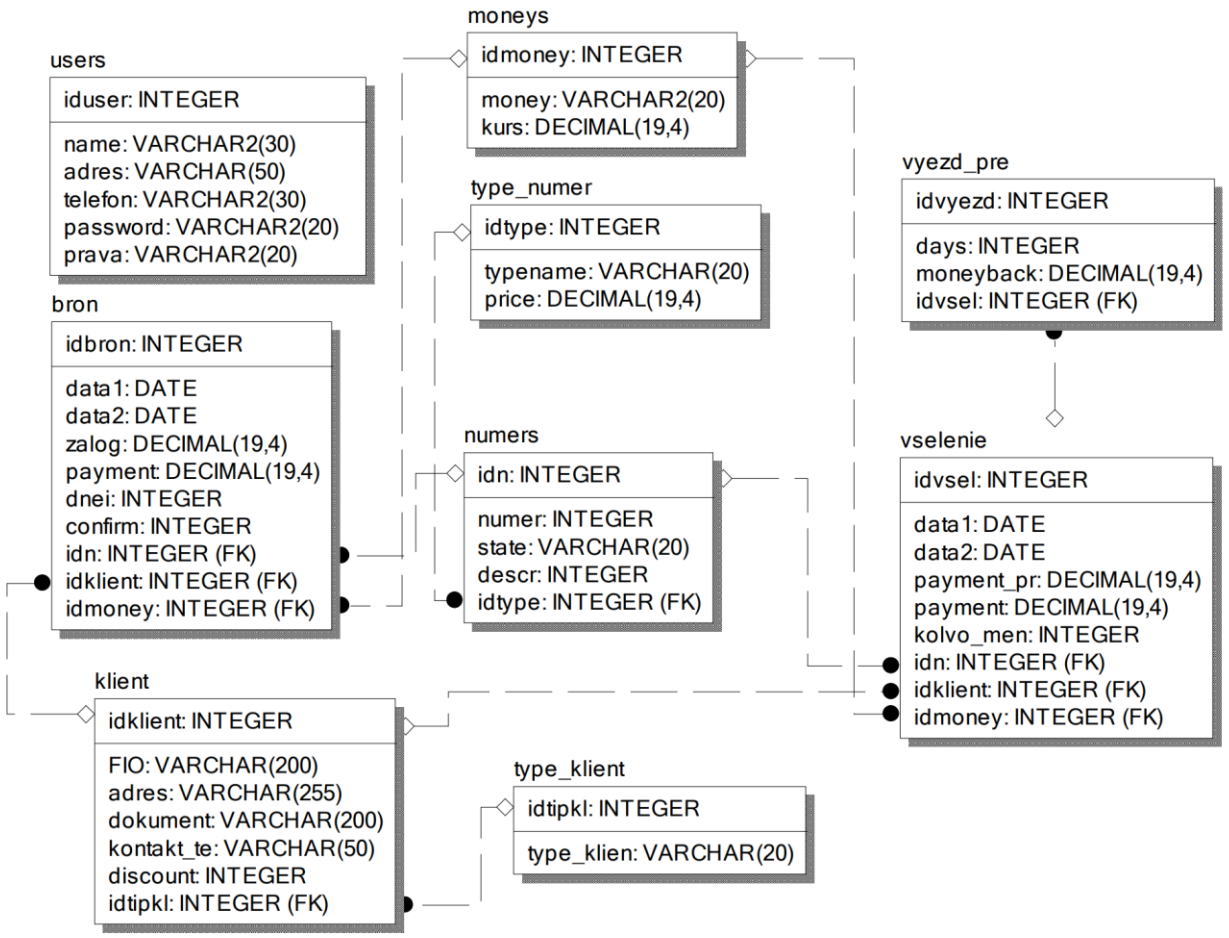



Рисунок 13. Физическая модель АРМ

П: Разработать по выданному варианту физическую и логическую модель ИС.

## Задание 2.8. Разработка диаграммы классов для описания структуры выбранного решения.

Т: Диаграмма классов — это структурная диаграмма UML, которая отображает статическую структуру системы, включая классы, их атрибуты, методы и взаимосвязи между ними.

Основные элементы диаграммы: класс, имя класса, атрибуты (поля), методы (операции), видимость элементов, связи между классами, ассоциация, наследование, агрегация, композиция, зависимость.

Этапы создания диаграммы классов: анализ предметной области, выявление основных понятий, определение объектов и их характеристик, анализ взаимосвязей между объектами, идентификация классов, определение первичных объектов системы, выделение атрибутов классов, определение методов, определение отношений, анализ связей между классами, установление типов отношений, определение кратности связей, документирование, описание атрибутов, описание методов, спецификация интерфейсов, типы отношений между классами, ассоциация,

двунаправленная связь между классами, может иметь кратность.

Пример: “студент учится в группе”, наследование отношение “является”., иерархия классов, агрегация отношение “часть-целое”., слабая связь части могут существовать отдельно., композиция сильная форма агрегации, части не имеют смысла существовать отдельно.

**Стандартный формат класса.**

Рекомендации по построению: структурирование, разбивка на пакеты, группировка связанных классов, использование поддиаграмм, читаемость, логичное расположение элементов, минимизация пересечений связей, использование цветов для группировки, детализация, постепенное добавление деталей, начало с высокоуровневого представления, постепенное углубление в детали.

Инструменты моделирования: Visual Studio Class Designer, PlantUML, Draw.io, Enterprise Architect, StarUML.

Используется: проверка корректности, валидация структуры, проверка целостности связей, анализ циклических зависимостей, оценка сложности архитектуры, тестирование, проверка реализуемости, оценка производительности, анализ масштабируемости, документирование, обязательные элементы документации:, описание каждого класса, спецификация атрибутов, описание методов, детализация связей, ограничения и инварианты, типичные ошибки, избыточное наследование, сложные иерархии, некорректные связи, отсутствие документации, нарушение принципов ООП, практические советы, принцип единственной ответственности, инкапсуляция данных, минимальные зависимости, понятные имена классов, логичная структура.

Диаграмма классов является фундаментальным инструментом проектирования, позволяющим создать четкую структуру будущей системы и обеспечить эффективное взаимодействие всех её компонентов.

Диаграмма классов состоит из трех основных частей, как показано на рисунке ниже:

1. Название класса
2. Атрибуты класса
3. Операции класса

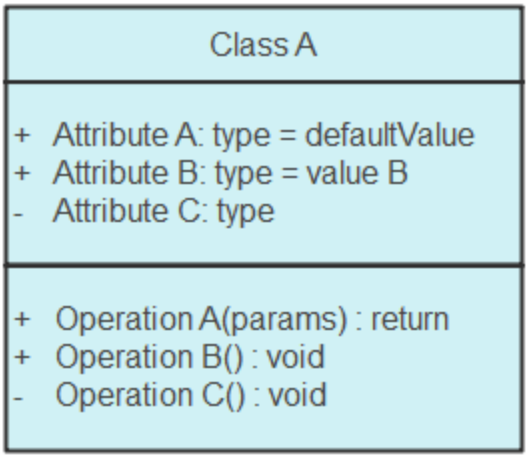



Рисунок 14

Для представления класса используется один прямоугольник, как показано выше. Прямоугольник разделен на три части, самая верхняя из них - название класса, атрибуты располагаются в середине, а операции внизу.

**Название класса**

Название класса важно для графического представления. В этом случае текст выделяется жирным шрифтом в верхнем поле и начинаться с заглавной буквы. Кроме того, абстрактный класс должен быть выделен курсивом.

**Атрибуты**

Атрибуты записываются в средней части и содержат список всех свойств моделируемого объекта. Вы можете просто добавить новые атрибуты или вывести новые атрибуты из уже перечисленных атрибутов. Атрибуты должны быть значимыми, обычно их используют с коэффициентом видимости, который описывает доступность атрибута.

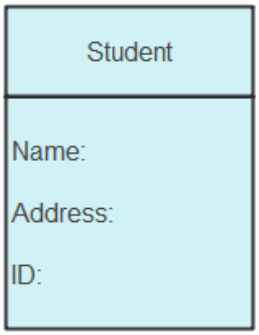



Рисунок 15

**Операции** - это процессы, выполнение которых известно классу. Каждая операция соответствует методам класса. Вам не нужно показывать операции, которые похожи на атрибуты, данные могут быть выведены из информации.

**Отношения классов**

Следующий шаг при создании диаграммы классов - построение отношений или связей. Здесь есть три основных типа отношений:

Обобщения

Ассоциации

Зависимости

**Обобщения** часто называют **наследованием**, поскольку они связывают подкласс с его суперклассом. Диаграмма классов позволяет подклассу брать данные от нескольких суперклассов, но не может быть использована для моделирования реализации интерфейса. Например, расчетные, сберегательные и кредитные счета обобщаются с помощью учетной записи

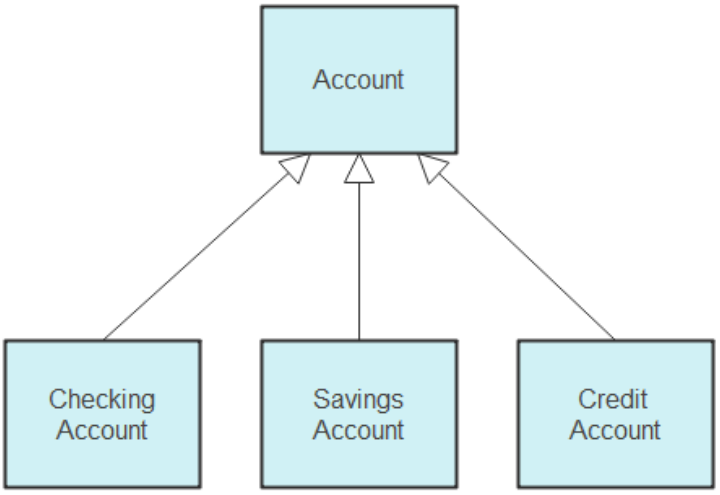



Рисунок 16

**Ассоциация** показывает статическую взаимосвязь между двумя объектами. Связь между учащимся и школой - это учеба.

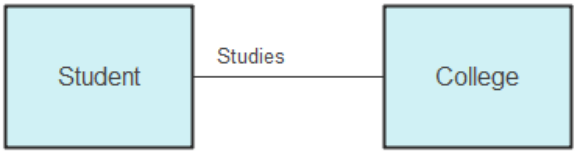



Рисунок 17

Коэффициент множественности в ассоциации показывает, во сколько раз умножается атрибут. В случае, если в организации работает 100 человек, то атрибут умножается в 100 раз.

При **агрегировании** два класса имеют отношение "целое-часть". Например, если сотрудник не придет, организация останется на месте.

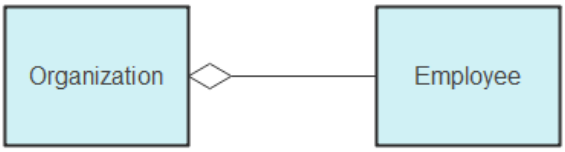



Рисунок 18

У агрегации есть еще один особый тип, называемый **композицией.** В композиции класс настолько тесно связан с другим классом, что без него он перестанет функционировать. Например, если организация закрывается, всем сотрудникам придется уволиться.

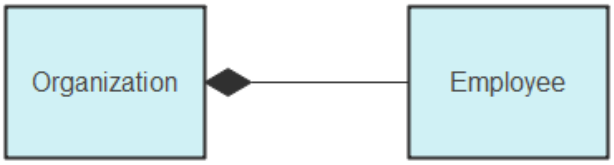



Рисунок 19

**Зависимость** показывает зависимость одного класса от другого. Изменение в одном классе приведет к изменению в другом классе. Например, сотрудник зависит от организации.

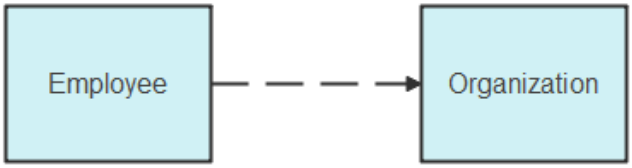



Рисунок 20

Диаграмма классов - примеры распространенных сценариев схема классов для системы управления отелем.

Диаграмма классов управления отелями тщательно связывает все классы, соединяя их стрелками, чтобы показать взаимосвязь между ними. С легкостью настройте диаграмму классов управления отелями и добавить дополнительные классы по желанию.

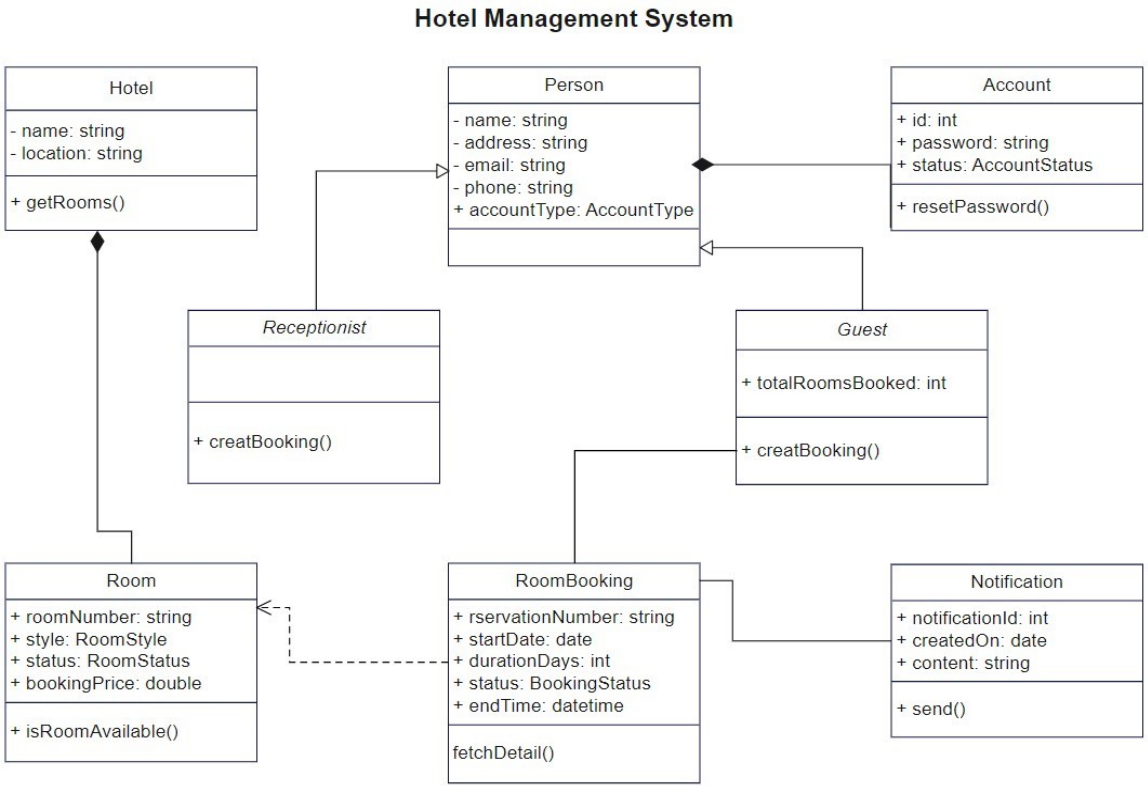



Рисунок 21

П: Разработать по выданному варианту диаграмму классов.

## Задание 2.9. Разработка моделей интерфейсов пользователей.

Т: Пользовательский интерфейс — это совокупность средств и методов взаимодействия пользователя с информационной системой. Качественно разработанный интерфейс обеспечивает эффективное и удобное использование системы.

Основные принципы проектирования интерфейсов: простота и интуитивность — интерфейс должен быть понятным без дополнительных инструкций, последовательность — единообразие элементов и их поведения, обратная связь — система должна информировать пользователя о результатах действий, гибкость — возможность адаптации под разные сценарии использования, эффективность — минимизация усилий пользователя для достижения цели.

**Этапы разработки интерфейсов.**

подготовительный этап, анализ целевой аудитории, изучение требований к системе, сбор информации о конкурентах, определение основных сценариев использования, проектирование, создание карты пользовательских путей (CJM), разработка структуры интерфейса, проектирование навигации, создание прототипов, дизайн, разработка визуальной концепции, создание макетов экранов, проработка стилей и компонентов, определение цветовой схемы, разработка интерактивного

прототипа, создание кликабельного прототипа, тестирование сценариев, сбор обратной связи, внесение корректировок, компоненты интерфейса, основные элементы управления, кнопки и ссылки, поля ввода, списки и выпадающие меню, чекбоксы и радиокнопки, слайдеры и шкалы, визуальные компоненты, иконки и изображения, цветовые схемы, типографика, анимации.

Методы разработки интерфейсов: проектирование на основе сценариев — разработка под конкретные пользовательские задачи, дизайн-система — создание набора стандартных компонентов, MVP-подход — разработка базового функционала с последующим развитием, DATA-DRIVEN DESIGN — проектирование на основе анализа поведения пользователей.

Тестирование интерфейсов: юзабилити-тестирование — проверка удобства использования, функциональное тестирование — проверка корректности работы, кроссплатформенное тестирование — проверка работы на разных устройствах, стресс-тестирование — проверка устойчивости к нагрузкам.

Документация интерфейса: техническое задание — описание требований и ограничений, дизайн-спецификация — детальное описание компонентов, гайдлайны — правила использования элементов, пользовательская документация — инструкции по работе.

Типичные ошибки проектирования: нелогичная навигация — запутанная структура меню, перегруженность интерфейса — избыточное количество элементов, отсутствие обратной связи — пользователь не понимает результат действий, неинтуитивные элементы — нестандартное поведение компонентов.

Инструменты разработки.

Для прототипирования: Figma, Sketch, Adobe XD, Balsamiq.

Для дизайна: Photoshop, Illustrator, Figma, UXPin.

Для тестирования: UsabilityHub, UserTesting, Hotjar.

Рекомендации по разработке: фокус на пользователе — проектирование под реальные потребности, итеративный подход — постоянное улучшение на основе обратной связи, тестирование на ранних этапах — выявление проблем до реализации, документация стандартов — создание единой системы компонентов.

Успешная разработка пользовательских интерфейсов требует комплексного подхода, учитывающего как технические аспекты, так и потребности конечных

пользователей.

П: Разработать по выданному варианту модель интерфейса.

## Задание 2.10. Выбор технологий для разработки ИС.

Т: Основные критерии выбора технологий: соответствие целям проекта, масштабируемость решения, стоимость разработки и поддержки, наличие специалистов, интеграционные возможности, безопасность, производительность, технологический стек.

**Языки программирования.**

Backend: Java (Spring Boot, Jakarta EE), Python (Django, Flask), JavaScript (Node.js), PHP (Laravel, Symfony), .NET (ASP.NET Core).

Frontend: JavaScript (React, Angular, Vue.js), TypeScript, HTML5, CSS3.

**Базы данных.**

Реляционные: PostgreSQL, MySQL, Microsoft SQL Server, NoSQL:, MongoDB, Cassandra, Redis, phpMyAdmin.

**Среды разработки.**

IDE: IntelliJ IDEA, Visual Studio Code, Eclipse, PyCharm.

Инструменты сборки: Maven, Gradle, npm/yarn, Composer.

**Инфраструктура и развертывание.**

Облачные платформы: AWS, Azure, Google Cloud, Yandex Cloud.

Контейнеризация: Docker, Kubernetes.

Инструменты разработки.

Версионирование: Git, SVN.

Тестирование: JUnit, Selenium, Postman, pytest.

Мониторинг: Prometheus, Grafana, ELK Stack.

Рекомендации по выбору:

анализ требований проекта: определение ключевых функций, оценка нагрузки, требования к безопасности, оценка команды: опыт разработчиков, доступность специалистов, необходимость обучения.

Бюджет проекта: стоимость лицензий, расходы на поддержку, инфраструктурные затраты.

**Этапы внедрения технологий.**

Исследование и выбор: анализ существующих решений, тестирование

технологий, создание прототипов.

Планирование архитектуры: проектирование системы, выбор паттернов, определение компонентов.

Разработка и внедрение: создание документации, настройка окружения, внедрение CI/CD, риски и их минимизация.

Технологические риски: устаревшие технологии, сложность поддержки, отсутствие специалистов.

Меры снижения рисков: регулярное обновление стека, документирование решений, обучение команды, документация и стандарты.

Техническая документация: архитектура системы, описание API, инструкции по развертыванию, стандарты разработки, Code Review, тестирование, безопасность/

Успешный выбор технологического стека зависит от комплексного анализа всех аспектов проекта и тщательного планирования каждого этапа разработки. Важно учитывать не только текущие требования, но и перспективы развития системы.

П: Выбрать и описать по выданному варианту технологию разработки ИС.

# ТЕМА 3. НАДЁЖНОСТЬ И КАЧЕСТВО ИНФОРМАЦИОННОЙ СИСТЕМЫ.

Качество информационной системы — это совокупность свойств системы, определяющих возможность её использования для удовлетворения потребностей в соответствии с её назначением.

Актуальность темы обусловлена следующими факторами:

постоянное развитие информационных технологий требует поддержания высокого уровня качества ИС;

рост значимости информационных систем в бизнес-процессах организаций;

необходимость обеспечения безопасности и целостности данных;

важность соответствия систем современным стандартам качества.

Надежность ИС — это свойство системы сохранять во времени в установленных пределах значения всех параметров, характеризующих способность выполнять требуемые функции в заданных режимах и условиях применения.

Качество ИС определяется следующими основными показателями:

надежность;

достоверность;

безопасность;

эффективность;

компоненты надежности.

Основные составляющие надежности информационной системы:

безотказность — способность системы сохранять работоспособное состояние;

ремонтопригодность — приспособленность к предупреждению и обнаружению отказов;

долговечность — способность сохранять работоспособность при установленном техническом обслуживании;

методы обеспечения надежности.

Технические меры:

резервирование технических средств;

использование стандартных протоколов;

применение специализированных средств защиты.

Программные меры:

тщательное тестирование;

использование стандартных библиотек;

модульное построение программ;

контроль качества.

Основные виды контроля:

профилактический;

рабочий;

синтаксический;

семантический;

прагматический;

показатели качества.

Ключевые показатели:

единичные — характеризуют одно свойство надежности;

комплексные — описывают несколько свойств одновременно;

экономические — характеризуют целесообразность затрат;

стандарты качества.

Современные стандарты:

ISO 9000 — управление качеством;

ISO 14000 — экологические требования;

ISO 9126 — модель качества программного обеспечения.

Результаты освоения темы позволят:

оценивать качество информационных систем;

выявлять и устранять проблемы надежности;

обеспечивать защиту данных;

контролировать работоспособность системы;

оптимизировать процессы сопровождения ис.

В процессе изучения темы студенты получат практические навыки оценки и обеспечения качества информационных систем, необходимые для профессиональной деятельности в сфере информационных технологий.

## Задание 3.1. Определение показателей безотказности системы.

Т: Методика расчёта:

Определение типа элементов и их характеристик. Необходимо чётко сформулировать, что считается отказом системы, и учесть только те элементы, выход из строя которых приводит к отказу системы.

Выбор метода расчёта. В зависимости от структуры системы (последовательное, параллельное соединение элементов, резервирование и т. д.) выбирается соответствующий метод расчёта. Например, для последовательного соединения используется произведение вероятностей безотказной работы элементов, для параллельного — суммирование вероятностей отказов с учётом резервирования.

Определение интенсивностей отказов элементов. Эти данные могут быть получены из справочников, статистических данных испытаний или эксплуатации. Учитываются также внешние факторы (температура, вибрации, нагрузки), которые влияют на надёжность элементов.

Расчёт суммарной интенсивности отказов системы. Для последовательного соединения элементов интенсивности отказов суммируются:

Расчёт других показателей (средняя наработка до отказа, гамма-процентная наработка и т. д.) с использованием полученных данных.

П: Определить показатели безотказности ИС по вашему варианту.

## Задание 3.2. Определение показателей долговечности системы.

Т: Определение показателей долговечности системы. теперь для этой

Долговечность системы — это её способность сохранять работоспособное состояние до наступления предельного состояния при установленной системе технического обслуживания и ремонта. Для оценки долговечности используются показатели, связанные с ресурсом и сроком службы.

cyberleninka.ru +1

Основные показатели долговечности

Ресурс — суммарная наработка объекта от начала эксплуатации (или её возобновления после капитального ремонта) до наступления предельного состояния, оговорённого технической документацией.

Различают:

доремонтный ресурс (до первого капитального ремонта);

межремонтный ресурс (между капитальными ремонтами);

полный ресурс (до списания);

назначенный ресурс — суммарная наработка, при достижении которой эксплуатация объекта должна быть прекращена независимо от его технического состояния;

остаточный ресурс — суммарная наработка объекта от момента контроля его технического состояния до перехода в предельное состояние.

Срок службы — календарная продолжительность эксплуатации объекта от её начала (или возобновления после капитального ремонта) до наступления предельного состояния. Аналогично ресурсу, различают доремонтный, межремонтный, полный и назначенный сроки службы.

Средний ресурс — математическое ожидание ресурса. Статистическая оценка среднего ресурса определяется по формуле:

Средний срок службы — математическое ожидание срока службы. Рассчитывается аналогично среднему ресурсу, но для календарных сроков.

Гамма-процентный ресурс — суммарная наработка, в течение которой изделие не достигнет предельного состояния с заданной вероятностью

Гамма-процентный срок службы — календарная продолжительность

эксплуатации, в течение которой объект не достигнет предельного состояния с вероятностью

Дополнительные понятия

Гарантийный ресурс — наработка объекта, до завершения которой изготовитель гарантирует и обеспечивает выполнение определённых требований к объекту при условии соблюдения потребителем правил эксплуатации.

Гарантийный срок службы (срок гарантии) — календарный период, в течение которого изготовитель гарантирует и обеспечивает выполнение определённых требований к объекту при условии соблюдения потребителем правил эксплуатации.

Методика расчёта

Определение критерия предельного состояния системы. Он зависит от допустимого уровня снижения эффективности функционирования системы, заданного в техническом задании.

Сбор исходных данных.

Для расчётов необходимы:

перечень компонентов системы;

гамма-процентные ресурсы компонентов (если применимо);

«критические» параметры компонентов и их значения в рабочем и предельно допустимом режимах;

временные графики работы компонентов.

Выбор метода расчёта. Используются аналитические методы (на основе теоретических распределений) или статистические методы обработки данных испытаний и эксплуатации.

Расчёт среднего ресурса или срока службы. Проводится по формулам, учитывающим наработку или календарное время для каждого объекта в выборке.

Определение гамма-процентного ресурса или срока службы. Требует знания функции распределения ресурса/срока службы. Аналитический расчёт зависит от вида распределения (нормальное, Вейбулла и т. д.).

Учёт внешних факторов. На долговечность влияют условия эксплуатации, технологические погрешности производства, нагрузки и другие параметры.

П: Определить показатели долговечности ИС по вашему варианту.

## Задание 3.3. Определение комплексных показателей надежности системы.

Т: Комплексные показатели надежности информационной системы

Комплексные показатели надежности характеризуют готовность системы к применению, эффективность использования и сохранение работоспособности.

Коэффициент готовности (Кг).Определяет вероятность того, что система окажется работоспособной в произвольный момент времени.

Показатели затрат на обслуживание: трудоемкость технического обслуживания, средняя суммарная трудоемкость за определенный период, учитывается для всех видов работ, стоимость обслуживания, средние суммарные затраты на:, техническое обслуживание, ремонтные работы, запасные части, работы специалистов, методика расчета комплексных показателей, сбор исходных данных:, время безотказной работы, продолжительность простоев, время на обслуживание, затраты на ремонт.

Анализ данных: группировка по периодам, классификация простоев, оценка эффективности работ.

Расчет показателей: определение коэффициентов готовности, вычисление затрат, анализ динамики показателей.

Комплексные показатели используются для: оценки эффективности системы, планирования технического обслуживания, расчета необходимых ресурсов, оптимизации затрат на поддержку, принятия решений о модернизации.

Факторы влияния, на значения комплексных показателей влияют: качество компонентов системы, условия эксплуатации, квалификация персонала, организация технического обслуживания, своевременность профилактических работ.

Регулярный мониторинг комплексных показателей позволяет своевременно выявлять проблемы в работе системы и принимать меры по их устранению.

П: Определить комплексные показатели надежности ИС по вашему варианту.

## Задание 3.4. Определение единичных показателей достоверности информации в системе.

Т: Единичные показатели достоверности информации характеризуют отдельные аспекты безошибочности, полноты и целостности данных в информационной системе. Они позволяют количественно оценить качество

информации и выявить потенциальные проблемы в её обработке, хранении и передаче.

Основные единичные показатели достоверности информации

Доверительная вероятность необходимой точности (достоверность) — D = 1 − Рош. Это вероятность того, что в пределах заданной наработки (информационной совокупности — массива, показателя, реквизита, кодового слова, символа или иного информационного компонента) отсутствуют грубые погрешности, приводящие к нарушению необходимой точности. Здесь Рош — вероятность ошибки.

Средняя наработка информации на ошибку — Q = 1/P. Это отношение объёма информации, преобразуемой в системе, к математическому ожиданию количества ошибок, возникающих в информации.

Вероятность ошибки (параметр потока ошибок) — Рош. Это вероятность появления ошибки в очередной информационной совокупности. При передаче дискретных сообщений Рош можно оценить как отношение числа ошибочно принятых элементов сообщения к общему числу переданных элементов.

Вероятность коррекции в заданное время — Ркорр(τ). Это вероятность того, что время, затрачиваемое на идентификацию и исправление ошибки, не превысит заданного τ.

Среднее время коррекции информации — Ти. Это математическое ожидание времени, затрачиваемого на идентификацию и исправление ошибки.

Дополнительные аспекты оценки достоверности.

Достоверность данных рассматривается в синтаксическом аспекте и измеряется вероятностью отсутствия ошибок в данных. В отличие от достоверности информации, к снижению достоверности данных приводят любые погрешности, а не только грубые.

Контроль — один из ключевых методов обеспечения достоверности информации. Это процесс получения и обработки информации с целью оценки соответствия фактического состояния объекта предъявляемым к нему требованиям и выработки соответствующего управляющего решения. При обнаружении ошибки должны быть приняты меры для её устранения или выработки рекомендаций по локализации и идентификации ошибки.

Избыточность информации может способствовать повышению достоверности.

Например, введение избыточности данных и смысловой избыточности позволяет проводить контроль достоверности.

Особенности применения показателей

При оценке достоверности информации важно учитывать контекст её использования. Например, в некоторых случаях допустимый уровень ошибок может варьироваться в зависимости от критичности данных и последствий их искажения. Также стоит различать грубые погрешности, которые нарушают необходимую точность, и менее значимые ошибки, которые могут не влиять на общую достоверность информации.

Для комплексной оценки часто используют сочетание единичных показателей с другими метриками, например, с показателями полноты, своевременности и целостности данных. Это позволяет получить более полное представление о качестве информации в системе.

П: Определить единичные показатели достоверности ИС по вашему варианту.

## ТЕМА 4. БЕЗОПАСНОСТЬ ИНФОРМАЦИОННЫХ СИСТЕМ.

Безопасность информационных систем— это комплекс мер, направленных на обеспечение конфиденциальности, целостности и доступности информации в информационной системе.

Актуальность темы обусловлена следующими факторами:

рост числа киберугроз и информационных атак;

увеличение объема обрабатываемых данных;

развитие новых технологий и методов взлома;

повышение требований к защите персональных данных;

необходимость соответствия международным стандартам безопасности;

основные понятия безопасности.

Ключевые аспекты информационной безопасности:

конфиденциальность — защита от несанкционированного доступа;

целостность — обеспечение неизменности данных;

доступность — гарантия доступа авторизованным пользователям;

аутентификация — подтверждение личности пользователя;

авторизация — предоставление прав доступа;

Компоненты системы безопасности.

основные элементы защиты:

технические средства — оборудование и программно-аппаратные комплексы;

программные решения — антивирусные системы, межсетевые экраны;

организационные меры — политики безопасности, регламенты;

правовые механизмы — нормативные документы и стандарты;

Угрозы безопасности.

основные виды угроз:

внутренние угрозы — действия персонала, технические сбои;

внешние угрозы — хакерские атаки, вирусы, вредоносное по;

природные факторы — стихийные бедствия, аварии;

экономические риски — финансовые потери от нарушений безопасности;

методы защиты.

Современные подходы к обеспечению безопасности:

криптографическая защита данных;

системы аутентификации и идентификации;

межсетевое экранирование;

антивирусная защита;

резервное копирование;

мониторинг безопасности;

стандарты и нормативы.

Базовые стандарты:

ISO/IEC 27001 — системы менеджмента информационной безопасности;

ГОСТы по защите информации;

PCI DSS — защита платежных данных;

ФЗ-152 «О персональных данных»;

Результаты освоения темы позволят:

оценивать риски информационной безопасности;

разрабатывать политики безопасности;

внедрять системы защиты информации;

проводить аудит безопасности;

управлять инцидентами безопасности.

В процессе изучения темы студенты получат практические навыки

обеспечения безопасности информационных систем, необходимые для профессиональной деятельности в сфере информационных технологий. Особое внимание будет уделено практическому применению современных методов защиты и анализу эффективности внедряемых мер безопасности.

## Задание 4.1. Основные угрозы.

Т: Угрозы для информационных систем (ИС) можно классифицировать по разным критериям.

Основные типы угроз и их примеры:

**По источнику возникновения.**

Внешние угрозы — исходят от злоумышленников, не связанных с организацией.

Примеры: хакерские атаки. попытки взлома системы с целью кражи данных, повреждения инфраструктуры или других вредоносных действий. DDoS-атаки (Distributed Denial of Service). Перегрузка сети или сервера большим количеством запросов, что приводит к временной недоступности инфраструктуры, фишинг. мошеннические схемы для получения личных или платёжных данных, рассылка писем с поддельными веб-ресурсами, маскирующимися под настоящие.

Внутренние угрозы — инициируются сотрудниками организации.

Примеры: несанкционированный доступ к ресурсам и базам данных. часто происходит из-за избыточных полномочий у сотрудников. умышленные утечки. целенаправленная передача конфиденциальной информации третьим лицам, случайные утечки или повреждение данных. например, когда сотрудник случайно удаляет важные файлы или программу, что приводит к потере данных и нарушению работы системы.

**По способу реализации.**

Технические угрозы — связаны с использованием уязвимостей в программном обеспечении, оборудовании или сетевых протоколах. Примеры: эксплуатация уязвимостей в ПО, использование ошибок разработчиков для взлома системы или кражи данных, несовершенные механизмы аутентификации, устаревшие компоненты, проблемы с алгоритмами вредоносное ПО, вирусы, трояны, шифровальщики, шпионские программы, кейлогеры, трояны маскируются под легитимные программы, шифровальщики парализуют систему, кейлогеры

регистрируют нажатия клавиш. , SQL-инъекции. использование вредоносных запросов к базам данных, которые могут привести к удалению важных файлов или краже информации.

Социальные угрозы (социальная инженерия) — основаны на психологических манипуляциях. Примеры: претекстинг -создание ложного предлога для получения доступа к информации, обещание рекламы в соцсетях в обмен на вход через аккаунт , BEC-атаки (Business Email Compromise), взлом или подделка почтовых аккаунтов руководителей с целью заставить сотрудников перевести деньги на контролируемые преступниками счета, целевой фишинг -точечная атака на конкретного сотрудника с использованием данных из его профиля в соцсетях, должности и рабочих задач.

Физические угрозы — связаны с непосредственным доступом к технике. Примеры: кража устройств (ноутбуков, серверов)., проникновение в серверную., уничтожение оборудования (например, при пожаре или затоплении).

**По цели.**

Угрозы данным — попытки несанкционированного доступа с целью кражи, изменения, порчи или удаления данных. Пример: кража персональных данных через взлом базы.

Угрозы программной среде — атаки на программное обеспечение, попытки внедрения вредоносного ПО или изменения программных компонентов. Пример: внедрение трояна, маскирующегося под обновление системы.

Угрозы аппаратному обеспечению — физическое повреждение оборудования, кража или неправомерный доступ к аппаратным компонентам. Пример: повреждение серверов при перегреве.

Угрозы поддерживающей инфраструктуре — атаки на сетевую инфраструктуру, серверы и базы данных. Пример: DDoS-атака на DNS-серверы.

По преднамеренности.

Преднамеренные угрозы — умышленные действия злоумышленников. Примеры:

целенаправленные хакерские атаки, кража данных, внедрение вредоносного ПО.

Непреднамеренные угрозы — случайные действия или ошибки.

Примеры:

удаление важных файлов администратором по ошибке;

сбой сервера из-за перегрева;

установка сотрудником программ, не относящихся к работе, что приводит к нарушению системы.

**По объекту воздействия.**

Угрозы могут направляться на разные объекты:

Автоматизированные рабочие места (АРМ), заражение компьютера сотрудника вредоносным ПО.

Локальные и распределённые информационные системы, атака на корпоративную сеть.

Средства обработки данных (принтеры, плоттеры и т. п.).

Каналы передачи данных, перехват информации при передаче по сети.

Дополнительные категории.

Угрозы целостности — влияют на полноту и достоверность информации, изменение данных о клиентах в базе перед увольнением сотрудника.

Угрозы конфиденциальности — приводят к утечкам информации, взлом сайта оператора связи с утечкой данных более 100 тысяч клиентов.

Угрозы доступности — нарушают доступ к информации или сервисам, блокирование системы программами-вымогателями.

Естественные угрозы — не зависят от человека, связаны с природными явлениями (ураганы, наводнения, пожары).

Для минимизации рисков важно использовать комплексный подход, включающий технические меры (антивирусы, фаерволы, шифрование), организационные меры (обучение сотрудников, контроль доступа) и регулярные аудиты безопасности.

П: Опишите основные угрозы для ИС по вашему варианту.

## Задание 4.2. Защита от несанкционированного доступа.

Т: Защита от несанкционированного доступа (НСД) — комплекс мер, направленных на предотвращение незаконного проникновения в информационные системы, а также на ограничение доступа к данным и ресурсам для неавторизованных пользователей. Эти меры включают технические, организационные и административные методы.

**Технические меры.**

Аутентификация и идентификация

Аутентификация — проверка подлинности пользователя с помощью различных факторов.

Виды аутентификации: однофакторная — использует один метод подтверждения личности (например, пароль), многофакторная (MFA) — сочетает несколько факторов: знание (пароль), обладание (токен, смарт-карта) и свойство (биометрия — отпечатки пальцев, распознавание лица, голос), двухфакторная аутентификация (2FA) — комбинация двух разных факторов, например, пароля и одноразового кода из мобильного приложения.

Шифрование данных.

Криптографические методы преобразуют данные так, что они становятся недоступными для несанкционированного доступа.

Используются два основных метода: симметричный — один ключ для шифрования и расшифровки, асимметричный — пара ключей: открытый для шифрования и закрытый для расшифровки, протоколы TLS/SSL и IPsec обеспечивают защищённую передачу данных, шифруя трафик между клиентом и сервером.

Межсетевые экраны (файерволы).

Фильтруют сетевой трафик на основе заранее установленных правил, блокируя попытки несанкционированного доступа и атаки. Специализированные WAF (Web Application Firewall) защищают веб-приложения от SQL-инъекций и XSS-атак.

Системы обнаружения и предотвращения вторжений (IDS/IPS).

IDS отслеживают подозрительную активность и уведомляют администраторов, а IPS автоматически блокируют угрозы в реальном времени.

Антивирусное ПО.

Обнаруживает и блокирует вредоносное ПО, которое может использоваться для несанкционированного доступа.

Модули доверенной загрузки.

Аппаратные контроллеры, устанавливаемые на материнскую плату, работают только с доверенными пользователями и блокируют устройство при попытках

включения без владельца.

Организационные меры: разграничение доступа и принцип наименьших привилегий, каждый пользователь получает доступ только к тем ресурсам, которые необходимы для выполнения рабочих задач, администраторы должны использовать отдельные учётные записи, а операции без повышенных прав — выполнять с непривилегированных аккаунтов, ролевая модель доступа (RBAC) распределяет права в зависимости от роли пользователя (например, читатель, редактор, администратор).

Сегментация сети: разделение сети на логические или физические сегменты с разными уровнями доступа и контроля. Это ограничивает распространение угроз и упрощает мониторинг.

Политика безопасности: чёткие правила использования информационных ресурсов, порядок действий при инцидентах, требования к паролям, доступу к данным и т. д.

Обучение персонала.

Регулярные тренинги по кибербезопасности для сотрудников помогают снизить риски, связанные с человеческим фактором. Сотрудники учатся распознавать фишинговые атаки, соблюдать правила работы с данными.

**Административные меры.**

Аудит и мониторинг.

Регулярный анализ журналов событий, отслеживание аномальной активности, подозрительных попыток доступа. Системы мониторинга в реальном времени уведомляют администраторов о потенциальных угрозах.

Резервное копирование данных.

Позволяет восстановить информацию в случае её потери или повреждения из-за взлома, сбоя системы или вредоносного ПО.

Управление ключами и сертификатами.

Грамотная организация инфраструктуры криптографии, включая генерацию, распределение, хранение и своевременную замену криптографических ключей.

Пенетрейшн-тесты и сканирование уязвимостей.

Тестирование на проникновение (пентесты) и использование сканеров уязвимостей помогают выявить слабые места в системе безопасности до того, как их

обнаружат злоумышленники.

**Физические меры.**

Размещение серверов в защищённых помещениях с ограниченным доступом.

Установка видеонаблюдения в серверных комнатах.

Использование резервных источников питания.

Контроль доступа к оборудованию (например, с помощью систем контроля доступа в помещения).

Эффективная защита от НСД требует комплексного подхода, сочетающего технические, организационные и административные меры. Важно регулярно обновлять системы, проводить аудит безопасности и адаптировать меры защиты к новым угрозам.

П: Опишите основные методы защиты для ИС по вашему варианту от несанкционированного доступа.

# ТЕМА 5. РАЗРАБОТКА ТЕХНИЧЕСКОГО ЗАДАНИЯ НА ВНЕДРЕНИЕ ИС.

Техническое задание — это основополагающий документ, определяющий требования и порядок создания, внедрения и эксплуатации информационной системы.

Актуальность темы обусловлена следующими факторами:

необходимость формализации требований к разрабатываемой системе;

важность четкого определения границ проекта;

потребность в документировании всех аспектов внедрения;

значимость оценки ресурсов и сроков реализации;

структура технического задания.

Основные разделы ТЗ включают:

общие сведения;

назначение и цели создания системы;

характеристики объекта автоматизации;

требования к системе;

состав и содержание работ по созданию системы;

порядок контроля и приемки системы;

требования к документированию;

источники разработки;

основные требования к ИС.

Функциональные требования:

описание процессов автоматизации;

перечень функций и задач;

характеристики входных и выходных данных;

требования к интерфейсам.

Нефункциональные требования:

надёжность;

безопасность;

производительность;

масштабируемость;

удобство использования;

этапы разработки ТЗ.

Последовательность работ:

анализ текущего состояния;

определение целей и задач;

сбор требований от пользователей;

формализация требований;

согласование и утверждение;

методология разработки.

Ключевые подходы:

структурный анализ;

моделирование бизнес-процессов;

системный подход к проектированию;

учёт стандартов разработки;

Содержание разделов ТЗ:

Раздел 1. Общие сведения:

наименование системы;

заказчик и разработчик;

сроки и этапы разработки;

Источники финансирования.

Раздел 2. Назначение и цели:

функциональное назначение;

цели создания;

результаты внедрения.

Раздел 3. Характеристики объекта автоматизации:

описание предметной области;

существующая система;

процессы автоматизации.

Раздел 4. Требования к системе:

технические требования;

требования к надежности;

требования к безопасности;

требования к персоналу;

практическая значимость.

Результаты освоения темы позволят:

разрабатывать качественные технические задания;

формулировать четкие требования к ИС;

планировать этапы внедрения;

оценивать ресурсы проекта;

контролировать процесс реализации.

В процессе изучения темы студенты получат практические навыки разработки технических заданий, необходимые для профессиональной деятельности в сфере информационных технологий. Особое внимание будет уделено методологии формирования требований и их документальному оформлению.

## Задание 5.1. Стандарты на разработку и внедрение информационной системы.

Т: Стандарты на разработку и внедрение информационных систем (ИС) регулируют процессы проектирования, создания, эксплуатации и защиты систем. Они устанавливают требования к документации, жизненному циклу, безопасности, качеству и другим аспектам. Основные стандарты можно разделить на международные, российские и отраслевые.

Международные стандарты

ISO/IEC 12207 («Информационные технологии. Системная и программная инженерия. Процессы жизненного цикла программных средств») — базовый стандарт, определяющий процессы жизненного цикла программного обеспечения (ПО) как части автоматизированной системы. Охватывает все этапы: от концептуализации идей до завершения жизненного цикла. Стандарт динамичен и адаптивен: позволяет реализовывать разные модели жизненного цикла, исключать неприменимые в конкретном проекте процессы, виды деятельности и задачи.

ISO/IEC 29148:2011 (ранее — IEEE 830-1998) регламентирует работу с требованиями к ПО: их определение, моделирование, анализ, верификацию, валидацию, управление изменениями. Определяет структуру и содержание спецификации требований к системе (SyRS) и к ПО (SRS).

ISO/IEC 27001 («Информационная технология. Методы и средства обеспечения безопасности. Системы менеджмента информационной безопасности. Требования») устанавливает требования к системам менеджмента информационной безопасности.

ISO/IEC 27005 («Информационная технология. Методы и средства обеспечения безопасности. Менеджмент риска информационной безопасности») определяет подходы к управлению рисками в сфере информационной безопасности.

Российские стандарты

ГОСТ Р 59793-2021 («Информационные технологии. Комплекс стандартов на автоматизированные системы. Автоматизированные системы. Стадии создания») определяет стадии и этапы создания автоматизированных систем (АС), включая исследования, проектирование, разработку, внедрение и эксплуатацию.

ГОСТ Р 59795-2021 («Информационные технологии. Комплекс стандартов на автоматизированные системы. Автоматизированные системы. Требования к содержанию документов») устанавливает требования к содержанию документации, разрабатываемой при создании АС.

ГОСТ 34.601-90 («Автоматизированные системы. Стадии создания») — ранее действовавший стандарт, который также регламентировал этапы разработки АС, но в большей степени соответствовал каскадной модели жизненного цикла.

ГОСТ Р 56939-2024 («Защита информации. Разработка безопасного программного обеспечения. Общие требования») регулирует процессы разработки

безопасного ПО, включая планирование, реализацию, верификацию, валидацию, сопровождение и улучшение. Стандарт актуален для разработчиков средств защиты информации, которые проходят сертификацию ФСТЭК.

ГОСТ Р 59547-2021 («Защита информации. Мониторинг информационной безопасности. Общие положения») определяет требования к мониторингу информационной безопасности.

Стандарты в области безопасности

ГОСТ Р 52633.5-2011 устанавливает требования к индикации близости предъявленных биометрических данных к образу «Свой».

ГОСТ Р 56115-2014 классифицирует уязвимости информационных систем.

ГОСТ Р 58412-2019 описывает угрозы безопасности информации при разработке программного обеспечения.

ГОСТ Р 58833-2020 регламентирует идентификацию и аутентификацию.

ГОСТ Р 59453.1-2021 и ГОСТ Р 59453.2-2021 касаются формальных моделей управления доступом и их верификации.

Другие важные стандарты

IDEF (ICAM Definitions) — серия стандартов для моделирования различных аспектов систем:

IDEF0 — функциональное моделирование;

IDEF1X — моделирование баз данных на основе модели «сущность-связь»;

IDEF3 — документирование технологических процессов.

Корпоративные стандарты разрабатываются организациями для конкретных проектов и включают стандарты проектирования, оформления документации, пользовательского интерфейса и др.

**Особенности стандартизации.**

Стандарты могут быть официальными (международные, национальные, отраслевые, ведомственные) и «де-факто» (неофициальные, но фактически применяемые, например, язык программирования C или SQL). Для сложных проектов часто создаются корпоративные стандарты, адаптированные под конкретные условия.

При выборе стандартов важно учитывать специфику проекта, требования заказчика, используемые технологии и нормативные требования в отрасли.

Примеры документации и файлов при разработке ИС: проектная документация, техническое задание (ТЗ), описание целей и задач проекта, требования к системе, этапы разработки, сроки выполнения, бюджет проекта, проектная документация, архитектурная схема системы, диаграммы классов и последовательностей, ER-диаграммы, описание API, схема развертывания, техническая документация, документация по коду, описание основных модулей, комментарии к ключевым функциям, примеры использования, обработка ошибок, документация базы данных, структура таблиц, описание полей, связи между таблицами, индексы и ограничения, пользовательская документация, руководство пользователя, инструкции по установке, пошаговые руководства, часто задаваемые вопросы, устранение типовых проблем, справочная система, глоссарий терминов.

Эксплуатационная документация: документация администратора, настройка системы, мониторинг работы, резервное копирование, восстановление после сбоев, план аварийного восстановления, процедуры восстановления, необходимые ресурсы, сроки восстановления, роли участников.

Тестовая документация: тестовые планы, сценарии тестирования, критерии приемки, ожидаемые результаты, отчеты о тестировании, найденные дефекты, результаты тестов, рекомендации по исправлению.

Стандарты и регламенты: корпоративные стандарты, правила кодирования, соглашения по именованию, стандарты документации, процедуры управления, управление изменениями, контроль версий, управление релизами.

Примеры исходных файлов проекта:файлы кода (.java, .py, .js), файлы конфигурации (.xml, .json), скрипты развертывания, файлы документации (README.md, INSTALL.md, CHANGELOG.md, LICENSE.md).

Файлы тестирования: тестовые наборы (.test, .spec), тестовые данные, отчеты о тестировании.

Системы контроля версий: Git, SVN, Mercurial.

Системы документации: Confluence, Wiki, специализированные порталы, базы знаний, FAQ, база решений проблем, лучшие практики.

Все эти документы и файлы должны быть взаимосвязаны и составлять единую систему документации, обеспечивающую полный жизненный цикл информационной системы.

## Задание 5.2. Построение графика разработки и внедрения информационной системы.

Т: График разработки и внедрения ИС — это визуальное представление плана проекта, где каждая задача или этап представлены в виде полосы, протяжённость которой соответствует длительности выполнения.

Он включает:

список задач или этапов;

даты начала и окончания каждой задачи;

продолжительность выполнения;

зависимости между задачами (если одна задача не может начаться до завершения другой);

ресурсы, выделенные на каждую задачу (люди, оборудование, бюджет);

вехи — ключевые моменты или контрольные точки проекта.

Основой для графика служит жизненный цикл ИС — последовательность стадий от возникновения потребности в системе до её вывода из эксплуатации. Стандартные стадии включают:

Планирование и анализ требований — определение целей, сбор требований, технико-экономическое обоснование, разработка технического задания.

Проектирование — выбор проектных решений, разработка архитектуры системы, создание технического проекта.

Реализация (разработка) — программирование, настройка баз данных, создание инструкций.

Внедрение — тестирование, опытная эксплуатация, обучение персонала, сдача системы в промышленную эксплуатацию.

Эксплуатация и сопровождение — поддержка, модернизация, сбор обратной связи.

Модели жизненного цикла определяют подход к разбивке и выполнению этапов. Некоторые из них:

Каскадная модель (Waterfall) — строгая последовательность этапов без возможности возврата на предыдущие стадии.

Итеративная модель — проект разбивается на итерации, каждая из которых

добавляет функционал.

Спиральная модель — каждый виток включает анализ рисков, проектирование, тестирование и оценку.

V-модель — каждому этапу разработки соответствует этап тестирования.

Инструменты для построения графиков

Для создания и управления графиками используют специализированные программные средства, например:

MS Project — позволяет планировать, отслеживать и управлять проектами, создавать диаграммы Ганта.

Atlassian Jira — система для управления проектами и отслеживания задач, поддерживает визуализацию в виде досок и графиков.

Redmine, YouTrack, Trac — системы для управления проектами с возможностями планирования и контроля сроков.

Рекомендации:

Учитывать критический путь — последовательность задач, определяющую минимальную продолжительность проекта. Задержки на критическом пути влияют на общий срок.

Учитывать ресурсы — убедиться, что на все задачи выделено достаточно людей, оборудования и бюджета.

Закладывать резервы времени на непредвиденные задержки, особенно для сложных или не до конца изученных задач.

Регулярно обновлять график по мере выполнения задач и появления новой информации.

Выбор модели жизненного цикла и инструментов зависит от специфики проекта, требований к гибкости, срокам и ресурсам. Для крупных и сложных проектов часто используют гибридные подходы, сочетающие элементы разных моделей.

П: Построить график разработки и внедрения ИС по вашему варианту.

## Задание 5.3. Разработка сценария внедрения информационной системы для рабочего места.

Т: Основные этапы внедрения: подготовительный этап, анализ текущего состояния рабочего места, оценка существующих процессов, определение

требований к новой системе, формирование команды внедрения, планирование внедрения, разработка детального плана работ, распределение ресурсов, установка сроков реализации, определение критериев успеха, сценарий внедрения.

Этап 1: Подготовка рабочего места.

Техническое оснащение: проверка соответствия системных требований, подготовка необходимой инфраструктуры, установка базового ПО.

Организационная подготовка: назначение ответственных лиц, разработка графика внедрения, информирование пользователей.

Этап 2: Установка программного обеспечения.

Базовая установка: установка операционной системы, настройка сетевого подключения, установка антивирусного ПО.

Специализированное ПО: установка основной ИС, настройка параметров системы, интеграция с другими системами.

Этап 3: Настройка и адаптация.

Конфигурация системы: настройка пользовательских профилей, установка прав доступа, настройка рабочих процессов.

Персонализация: настройка интерфейса, создание необходимых шаблонов, настройка уведомлений.

Этап 4: Тестирование.

Функциональное тестирование: проверка основных функций, тестирование интеграции, проверка безопасности.

Нагрузочное тестирование: проверка производительности, оценка стабильности работы, выявление узких мест.

Этап 5: Обучение пользователей.

Базовое обучение: знакомство с интерфейсом, основные операции, работа с документацией.

Специализированное обучение: продвинутые функции, решение типовых задач, работа с отчетами.

Этап 6: Запуск в эксплуатацию.

Пилотный запуск: ограниченный запуск системы, сбор обратной связи, корректировка настроек.

Полный запуск: официальное внедрение, мониторинг работы, поддержка

пользователей.

Пример плана внедрения

| Этап | Действия | Сроки | Ответственные |
|---|---|---|---|
| 1 | Подготовка инфраструктуры | 2 дня | Системный администратор |
| 2 | Установка базового ПО | 1 день | IT-специалист |
| 3 | Установка ИС | 2 дня | Специалист по внедрению |
| 4 | Настройка системы | 3 дня | Администратор ИС |
| 5 | Тестирование | 2 дня | Тестировщик |
| 6 | Обучение | 1 день | Тренер |
| 7 | Запуск | 1 день | Команда внедрения |

Критерии успешного внедрения.

Технические показатели: стабильная работа системы, соответствие производительности требованиям, отсутствие критических ошибок.

Пользовательские показатели: положительная обратная связь, достижение целевых показателей эффективности, снижение количества ошибок, управление рисками.

Идентификация рисков: технические сбои, сопротивление пользователей, превышение сроков.

Меры по снижению рисков: резервное копирование, план отката изменений, обучение персонала, регулярный мониторинг.

Документация.

Техническая документация: руководство администратора, инструкция пользователя, план внедрения.

Эксплуатационная документация: журнал изменений, план поддержки, регламенты обслуживания.

Успешное внедрение ИС на рабочем месте требует тщательного планирования, координации всех участников процесса и постоянного контроля качества на каждом этапе.

П:Разработать сценарий внедрения для ИС по вашему варианту.

## Задание 5.4. Разработка пояснительной записки к внедрению информационной системы.

Т: Пояснительная записка к внедрению информационной системы

1. Введение

Цель разработки — описание процесса внедрения информационной системы (ИС) и обоснование необходимости её внедрения в организацию.

Задачи внедрения: анализ текущего состояния, разработка плана внедрения, организация процесса установки, обучение пользователей, оценка результатов внедрения.

2. Общая характеристика ИС.

Назначение системы: автоматизация бизнес-процессов, управление данными, оптимизация рабочих процессов, формирование отчетности.

Основные компоненты: программная часть, аппаратные средства, документация, обучающие материалы.

3. Подготовка к внедрению.

Предварительные работы: обследование объекта автоматизации, анализ существующих бизнес-процессов, определение требований к системе, формирование команды проекта.

Необходимые ресурсы: технические средства, программное обеспечение, кадровые ресурсы, финансовые средства.

4. План внедрения.

Этапы внедрения:

подготовительный этап:, разработка плана внедрения, подготовка документации, настройка инфраструктуры.

Установка системы:

монтаж оборудования, установка по, настройка параметров.

Настройка системы: конфигурация модулей, создание учетных записей, настройка прав доступа.

Тестирование: проверка функциональности, испытание производительности, выявление ошибок.

5. Организация работ.

Структура команды внедрения:

руководитель проекта, технический специалист, бизнес-аналитик, тренер, администратор системы.

График работ: разработка детального расписания, распределение задач,

контроль выполнения, корректировка плана.

6. Обучение пользователей.

Программа обучения: теоретическая подготовка, практические занятия, работа с реальными данными, консультации.

Формы обучения: индивидуальные занятия, групповые тренинги, самостоятельное изучение, дистанционное обучение.

7. Оценка результатов.

Критерии успешности: достижение целевых показателей, удовлетворенность пользователей, эффективность работы системы, экономический эффект.

Методы оценки: сбор обратной связи, анализ показателей, мониторинг работы, аудит безопасности.

8. Гарантийные обязательства.

Техническая поддержка: консультации пользователей, устранение ошибок, обновление системы, резервное копирование.

Документация:

руководство пользователя, техническая документация, инструкции администратора, регламенты обслуживания.

9. Заключение.

Ожидаемые результаты:

Повышение эффективности работысокращение временных затрат, улучшение качества данных, оптимизация бизнес-процессов.

Перспективы развития: расширение функциональности, интеграция с другими системами, модернизация компонентов, обновление технологий, приложения, план-график внедрения, смета затрат, акт приемки-передачи, протоколы испытаний, обучающие материалы.

## Задание 5.5. Написание плана вывода информационной системы из эксплуатации.

Т: 1. Общие положения

Цель плана — регламентация процесса вывода информационной системы (ИС) из эксплуатации с сохранением целостности данных и безопасности информации.

Основание для вывода из эксплуатации: технологическая устарелость,

экономическая неэффективность, смена бизнес-процессов, дублирование функциональности.

2. Подготовительный этап.

Предварительные мероприятия: формирование комиссии по выводу из эксплуатации, разработка календарного плана работ, информирование пользователей, подготовка технических средств.

3. Основные этапы вывода.

Этап 1: уведомление пользователей, публикация официального уведомления, информирование конечных пользователей, подготовка инструкций по переходу, сбор обратной связи.

Этап 2: подготовка системы, завершение текущих операций, архивация данных, проверка целостности информации, тестирование процедур резервного копирования.

Этап 3: отключение системы, прекращение доступа пользователей, блокировка учетных записей, отключение сервисов, деактивация сетевых подключений.

4. Технические мероприятия.

Работа с данными: создание резервных копий, проверка целостности архивов, классификация данных, определение сроков хранения.

Работа с инфраструктурой: деинсталляция по, очистка технических средств, удаление остаточных данных, проверка безопасности.

5. Документационное обеспечение.

Необходимая документация: акт о выводе из эксплуатации, отчет о состоянии данных, протокол проверки безопасности, документы по передаче данных, 6. План-график выполнения работ

Этап Мероприятие Срок выполнения Ответственный

1 Уведомление пользователей 5 рабочих дней Руководитель проекта

2 Подготовка системы 10 рабочих дней Технический специалист

3 Архивация данных 7 рабочих дней Администратор БД

4 Отключение сервисов 3 рабочих дня Системный администратор

5 Демонтаж ПО 5 рабочих дней Технический специалист

6 Документирование 5 рабочих дней Руководитель проекта

7. Обеспечение безопасности

Меры защиты: шифрование резервных копий, уничтожение данных на носителях, контроль доступа, мониторинг безопасности.

8. Контроль качества.

Критерии успешного завершения: полное удаление данных, сохранность архивной информации, отсутствие утечек данных, документальное оформление.

9. Заключительные положения.

Последующие действия: хранение резервных копий, обеспечение доступа к архивам, обновление документации, информирование заинтересованных сторон, приложения, форма акта о выводе из эксплуатации, инструкция по архивации данных, план обеспечения безопасности, регламент хранения данных.

П: Написать план вывода из эксплуатации для ИС по вашему варианту.

# СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ

**Основная литература**

1. Перлова О. Н. Соадминистрирование баз данных и серверов: учебное издание / Перлова О. Н., Ляпина О. П. - Москва : Академия, 2023. - 304 с. (Специальности среднего профессионального образования). - URL: https://academia-moscow.ru - Режим доступа: Электронная библиотека «Academia-moscow». - Текст : электронный.

2. Федорова, Г. Н. Разработка, внедрение и адаптация программного обеспечения отраслевой направленности : Учебное пособие / Г. Н. Федорова. – Москва : КУРС : ИНФРА-М, 2022. – 336 с. (Среднее профессиональное образование). – ISBN 9785906818416. – Текст : непосредственный.

**Дополнительная литература**

1. Компьютерные сети : учебник для среднего профессионального образования по специальностям 09.02.06 "Сетевое и системное администрирование", 09.02.07 "Информационные системы и программирование" / В. В. Баринов, И. В. Баринов, А. В. Пролетарский, А. Н. Пылькин ; В. В. Баринов [и др.]. – 4-е изд., испр. и доп.. - Москва : Академия, 2021. – 192 с. – ISBN 9785446899258. – URL: https://academia-moscow.ru/catalogue/4831/551458/. – Текст : электронный.

2. Гохберг Г.С. Информационные технологии: ЭУМК: учебное издание / Гохберг Г.С., Зафиевский А.В., Короткин А.А. - Москва : Академия, 2024. - 0 с. (Специальности среднего профессионального образования). - URL: https://academia-moscow.ru - Режим доступа: Электронная библиотека «Academia-moscow». - Текст : электронный.

3. Казанский, А. А.  Программирование на C# : учебное пособие для среднего профессионального образования / А. А. Казанский. — 3-е изд., перераб. и доп. — Москва : Издательство Юрайт, 2025. — 181 с. — (Профессиональное образование). — ISBN 978-5-534-21380-5. — Текст : электронный // Образовательная платформа Юрайт [сайт]. — URL: https://urait.ru/bcode/569863.

4. Семакин И.Г. Основы алгоритмизации и программирования: ЭУМК: учебное издание / Семакин И.Г., Шестаков А. П. - Москва : Академия, 2025. - 0 с. (Специальности среднего профессионального образования). - URL: https://academia-

moscow.ru - Режим доступа: Электронная библиотека «Academia-moscow». - Текст : электронный

5. Куприянов, Д. В. Информационное обеспечение профессиональной деятельности : учебник и практикум для среднего профессионального образования / Д. В. Куприянов. — 3-е изд., перераб. и доп. — Москва : Издательство Юрайт, 2025. — 236 с. — (Профессиональное образование). — ISBN 978-5-534-20826-9. — Текст : электронный // Образовательная платформа Юрайт [сайт]. — URL: https://urait.ru/bcode/558828.

6. Грекул, В. И. Проектирование информационных систем : учебник и практикум для среднего профессионального образования / В. И. Грекул, Н. Л. Коровкина, Г. А. Левочкина. — 2-е изд. — Москва : Издательство Юрайт, 2025. — 404 с. — (Профессиональное образование). — ISBN 978-5-534-19506-4. — Текст : электронный // Образовательная платформа Юрайт [сайт]. — URL: https://urait.ru/bcode/566739.

7. Проектирование информационных систем : учебник и практикум для среднего профессионального образования / Д. В. Чистов, П. П. Мельников, А. В. Золотарюк, Н. Б. Ничепорук. — 2-е изд., перераб. и доп. — Москва : Издательство Юрайт, 2025. — 273 с. — (Профессиональное образование). — ISBN 978-5-534-20362-2. — Текст : электронный // Образовательная платформа Юрайт [сайт]. — URL: https://urait.ru/bcode/562355.

**Приложение А. Бланк титульного листа**

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение \

высшего образования

«КУЗБАССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ

УНИВЕРСИТЕТ ИМЕНИ Т.Ф.ГОРБАЧЕВА»

Филиал КузГТУ в г.Белово

ОТЧЕТ ПО УЧЕБНОЙ ПРАКТИКЕ

ПМ.06 «Сопровождение информационных систем»

Выполнил:

Студент группы _____

/_____/_____

подпись

Руководитель:

/_____/_____

подпись

Оценка_____

«___»_____20___ г.

Белово

20__г.

**ПРИЛОЖЕНИЕ Б. Бланк задания по УП**

ЗАДАНИЕ

на учебную практику

по профессиональному модулю **ПМ.06 «Сопровождение информационных систем»**

студент _____

группы_____

специальности «09.02.07 Информационные системы и программирование»

Дата начала практики «___» _____ 20___г.

Дата окончания практики «___» _____ 20__г.

Дата сдачи практики «___» _____ 20__г.

**Виды работ, обязательных для выполнения:**

1. _____
2. _____
3. _____
4. _____

Задание выдал Руководитель учебной практики от

филиала КузГТУ в г.Белово /_____/_____

подпись

**ПРИЛОЖЕНИЕ В. Дневник по УП**

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«КУЗБАССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ИМЕНИ Т.Ф.ГОРБАЧЕВА»

Филиал КузГТУ в г.Белово

ДНЕВНИК УЧЕБНОЙ ПРАКТИКИ

профессионального модуля ПМ.06 «Сопровождение информационных систем»

студент _____

группы_____

специальности «09.02.07 Информационные системы и программирование»

Период практики с «___» _____ 20__г. по «___» _____ 20__.

База практики_____

М.П.

Руководитель учебной практики от

филиала КузГТУ в г.Белово

/_____/_____

подпись

Закончил практику «___» _____ 20__.

| Дата | Содержание работы | Оценка | Подпись |
|---|---|---|---|
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |

## ПРИЛОЖЕНИЕ Г. Бланк аттестационного листа по УП

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение \

высшего образования

«КУЗБАССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ

УНИВЕРСИТЕТ ИМЕНИ Т.Ф.ГОРБАЧЕВА»

Филиал КузГТУ в г.Белово

**АТТЕСТАЦИОННЫЙ ЛИСТ**

**по учебной практике**

по профессиональному модулю _____

*(индекс и наименование профессионального модуля)*

| | |
|---|---|
| Обучающийся | |
| Институт/факультет | |
| Специальность | |
| *(код специальности)* | |
| Курс | Группа |
| Вид практики | |
| Способ прохождения практики | |
| Период прохождения практики с | по |
| Профильная организация | |

*(наименование, местонахождение)*

Во время прохождения практики обучающимся были освоены следующие профессиональные и общие компетенции

| Наименование компетенции | Оценка | |
|---|---|---|
| | Освоена | Не освоена |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |

Руководитель учебной практики от филиала КузГТУ в г.Белово

/_____/_____

подпись

**ПРИЛОЖЕНИЕ Д. Бланк характеристики**

**на обучающегося в период прохождения УП**

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение \

высшего образования

«КУЗБАССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ

УНИВЕРСИТЕТ ИМЕНИ Т.Ф.ГОРБАЧЕВА»

Филиал КузГТУ в г.Белово

## ХАРАКТЕРИСТИКА

**на обучающегося по освоению общих и профессиональных компетенций**

**в период прохождения учебной практике**

по профессиональному модулю_____

*(индекс и наименование профессионального модуля)*

Обучающийся

Институт/факультет

Специальность

*(код специальности)*

| | |
|---|---|
| Курс | Группа |
| Вид практики | |
| Способ прохождения практики | |
| Период прохождения практики с | по |
| Профильная организация | |

*(наименование, местонахождение)*

Виды и качество выполненных работ:

| Виды работ | Критерии выполнения работ | | |
|---|---|---|---|
| | Выполнены полностью | Выполнены с незначительной помощью | Выполнены с помощью наставника |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |

Руководитель учебной практики от филиала КузГТУ в г.Белово

/_____/_____

подпись

Составитель

Витвицкий Максим Николаевич

Методические указания по учебной практике УП.06.01

для студентов очной формы обучения

по направлению специальности

09.02.07 «Информационные системы и программирование»

Публикуется в авторской редакции