**Лабораторная работа 2**

**Организационные меры по внедрению программного обеспечения**

*Цель 1:* научиться выполнять разработку программного продукта в команде с различным жизненным циклом

Для создания программного обеспечения "АРМ администратора аэропорта" можно выбрать следующую модель работ:

1. **Анализ требований**:
   * Проведение собеседований с администраторами аэропорта для определения конкретных потребностей и функциональных требований к системе.
   * Составление подробного технического задания.
2. **Проектирование системы**:
   * Разработка архитектуры системы, включая базу данных, пользовательский интерфейс и бизнес-логику.
   * Создание схемы взаимодействия различных модулей системы.
3. **Разработка программного обеспечения**:
   * Написание кода для фронтенда и бекенда системы.
   * Создание модулей для управления данными о рейсах, пассажирах, багаже, безопасности и других аспектов аэропорта.
4. **Интеграция с существующими системами**:
   * Обеспечение взаимодействия с уже существующими системами управления аэропортом, такими как системы бронирования, багажной обработки и безопасности.
5. **Разработка пользовательского интерфейса**:
   * Создание интуитивного и удобного интерфейса для администраторов аэропорта, который позволит им эффективно управлять различными аспектами работы аэропорта.
6. **Тестирование и отладка**:
   * Проведение тестирования на разных этапах разработки, включая модульное, интеграционное и приемочное тестирование.
   * Отладка и исправление ошибок.
7. **Внедрение и обучение персонала**:
   * Внедрение программного обеспечения в рабочую среду аэропорта.
   * Обучение администраторов и другого персонала работе с новой системой.
8. **Поддержка и обновление**:
   * Постоянная поддержка системы, включая регулярное обновление и устранение выявленных проблем.
9. **Мониторинг и оптимизация**:
   * Настройка мониторинга производительности системы и ее оптимизация для обеспечения эффективной работы.
10. **Соблюдение стандартов и безопасности**:
    * Гарантирование соблюдения всех необходимых стандартов и мер безопасности для обработки чувствительных данных в аэропорту.
11. **Документация**:
    * Создание подробной документации, описывающей функциональность системы и процедуры ее обслуживания.
12. **Мониторинг и обновление**:
    * Регулярный мониторинг производительности системы и ее обновление для обеспечения безопасной и эффективной работы.

Эти шаги позволят разработать и внедрить программное обеспечение для "АРМ администратора аэропорта" с учетом всех необходимых требований и стандартов.

Контрольные вопросы

Жизненный цикл программного обеспечения (ПО) включает в себя различные модели и методологии, которые описывают этапы создания, развития и поддержки программного продукта. Вот некоторые из наиболее распространенных моделей жизненного цикла ПО:

1. **Каскадная модель (Waterfall)**: Эта модель предполагает линейное выполнение этапов, начиная с определения требований, затем проектирования, разработки, тестирования, развертывания и поддержки. Каждый этап завершается, прежде чем начинается следующий.
2. **Итеративная модель**: В этой модели разработка ПО разбивается на циклы (итерации), каждый из которых включает в себя фазы, аналогичные каскадной модели. Однако разработчики могут вернуться к предыдущим итерациям для внесения изменений.
3. **Спиральная модель (Spiral)**: Эта модель комбинирует итерации с элементами анализа рисков. Процесс разработки ПО развивается в спираль, включая планирование, анализ рисков, инженерное проектирование, разработку и тестирование в каждой итерации.
4. **Инкрементальная модель**: В этой модели программное обеспечение разрабатывается по частям (инкрементам), и каждая новая версия добавляет дополнительные функции или улучшения к предыдущей версии.
5. **Прототипирование**: Здесь сначала создается прототип или предварительная версия ПО, которая используется для сбора обратной связи от заказчика и рефинирования требований. Затем разрабатывается окончательное ПО на основе полученных знаний.
6. **DevOps**: DevOps — это философия, объединяющая разработку (Dev) и операции (Ops). Она акцентирует автоматизацию, непрерывную интеграцию и непрерывное развертывание для более быстрой и надежной поставки ПО.
7. **Scrum**: Scrum - это фреймворк для гибкой разработки ПО, основанный на итеративной и инкрементальной методологии. Включает в себя роли, события и артефакты, например, спринты, стендапы и бэклог продукта.

Вот сравнительная характеристика каждой из перечисленных моделей жизненного цикла программного обеспечения:

1. **Каскадная модель (Waterfall)**:
   * Преимущества:
     + Хорошо подходит для проектов с жестко определенными требованиями.
     + Простота управления и отчетности.
   * Недостатки:
     + Мало гибкости при изменении требований.
     + Долгий цикл разработки, что может привести к задержкам в поставке.
2. **Итеративная модель**:
   * Преимущества:
     + Позволяет корректировать требования и функциональность в процессе разработки.
     + Улучшенное взаимодействие с заказчиком.
   * Недостатки:
     + Требует дополнительного управления и контроля над итерациями.
     + Возможна необходимость пересмотра ранних итераций.
3. **Спиральная модель (Spiral)**:
   * Преимущества:
     + Учитывает анализ рисков, что помогает предотвращать проблемы.
     + Гибкость в разработке и изменении требований.
   * Недостатки:
     + Требует больше времени и ресурсов из-за анализа рисков.
     + Сложно предсказать точные сроки завершения.
4. **Инкрементальная модель**:
   * Преимущества:
     + Постепенное внедрение новых функций.
     + Ранний доступ к базовой функциональности.
   * Недостатки:
     + Возможно появление несовместимостей между инкрементами.
     + Требует дополнительного управления версиями.
5. **Прототипирование**:
   * Преимущества:
     + Позволяет заказчику лучше понять и определить требования.
     + Сокращает риск разработки неправильного продукта.
   * Недостатки:
     + Прототип может потребовать дополнительной работы для превращения в окончательный продукт.
     + Не всегда подходит для крупных проектов.
6. **DevOps**:
   * Преимущества:
     + Ускоренная поставка и обновление ПО.
     + Автоматизация процессов снижает вероятность ошибок.
   * Недостатки:
     + Требует значительной культурной и организационной трансформации.
     + Не подходит для всех видов проектов.
7. **Scrum**:
   * Преимущества:
     + Высокая гибкость и участие заказчика в процессе.
     + Регулярные стендапы и инспекции способствуют качеству.
   * Недостатки:
     + Не всегда подходит для проектов с жесткими сроками.
     + Требует дисциплинированного управления бэклогом продукта.
8. **Kanban**:
   * Преимущества:
     + Подходит для проектов с непрерывной поставкой.
     + Улучшает видимость и контроль над рабочими процессами.
   * Недостатки:
     + Может потребовать изменений в культуре и методах работы команды.
     + Не всегда подходит для проектов с жесткими сроками.

Каждая модель имеет свои сильные и слабые стороны, и выбор определенной модели зависит от конкретных потребностей и характеристик проекта.

*Цель 2:* научиться распределять задания по рабочим группам, разрабатывать техническое задание.

Контрольные вопросы

1)Внедрение программного обеспечения (ПО) может иметь различные цели и задачи в зависимости от конкретного контекста и бизнес-потребностей. Вот несколько возможных целей и задач внедрения ПО:

Цели внедрения ПО:

1. **Автоматизация бизнес-процессов:** Целью может быть улучшение эффективности и продуктивности бизнеса путем автоматизации рутинных операций.
2. **Улучшение доступности и удобства:** Создание ПО с целью сделать информацию и сервисы более доступными для клиентов или пользователей.
3. **Увеличение безопасности:** Внедрение ПО для улучшения защиты данных и обеспечения информационной безопасности.
4. **Улучшение аналитики и принятия решений:** Использование ПО для анализа данных и предоставления лучшей аналитики для принятия решений.
5. **Увеличение конкурентоспособности:** Целью может быть создание уникальных функций или сервисов, которые позволят компании выделиться на рынке.
6. **Снижение затрат:** Внедрение ПО может направляться на оптимизацию бизнес-процессов и снижение операционных затрат.
7. **Улучшение пользовательского опыта:** Создание удобных и интуитивных интерфейсов для повышения удовлетворенности пользователей.

Задачи внедрения ПО:

1. **Определение требований:** Сбор и анализ требований к ПО.
2. **Выбор технологии:** Выбор наиболее подходящих технологий для разработки и внедрения.
3. **Разработка ПО:** Создание и настройка программного продукта в соответствии с требованиями.
4. **Тестирование:** Проверка ПО на наличие ошибок и дефектов.
5. **Обучение и внедрение:** Обучение пользователей и внедрение ПО в рабочий процесс.
6. **Поддержка и обновление:** Поддержание ПО, регулярные обновления и исправление ошибок.
7. **Мониторинг и анализ:** Слежение за производительностью и анализ эффективности ПО.
8. **Соответствие законодательным и регуляторным требованиям:** Обеспечение соблюдения законодательных норм и стандартов.
9. **Управление изменениями:** Управление изменениями и сопровождение процесса адаптации.

Эти цели и задачи могут варьироваться в зависимости от конкретного проекта и отрасли. Важно четко определить их перед началом внедрения ПО, чтобы обеспечить успешное достижение поставленных целей.

Команда разработчиков обычно состоит из различных рабочих групп, каждая из которых выполняет свои функции, специализируясь на определенных аспектах разработки программного обеспечения. Основные рабочие группы в команде разработчиков могут включать:

1. **Разработчики (Developers):** Это основные специалисты, которые пишут код и создают программное обеспечение. Разработчики могут быть разделены на подгруппы, такие как фронтенд-разработчики (Front-End Developers), бэкенд-разработчики (Back-End Developers), и т. д., в зависимости от их функциональных обязанностей.
2. **Тестировщики (Testers):** Эта группа занимается тестированием программного обеспечения, выявлением ошибок и дефектов, а также обеспечением качества продукта. Они могут включать в себя тестировщиков, автоматизированных тестировщиков и тест-инженеров.
3. **Архитекторы (Architects):** Архитекторы разрабатывают общую архитектуру программного продукта, определяют структуру и взаимодействие его компонентов, обеспечивая согласованность и масштабируемость.
4. **Дизайнеры (Designers):** Дизайнеры отвечают за создание пользовательского интерфейса (UI/UX) и внешнего вида приложения, обеспечивая удобство и привлекательность для пользователей.
5. **Аналитики (Business Analysts):** Аналитики занимаются анализом бизнес-требований, определением функциональных и нефункциональных требований, и взаимодействием с заказчиками.
6. **Менеджеры проекта (Project Managers):** Менеджеры проекта отвечают за планирование, управление сроками и ресурсами, а также координацию работы команды разработчиков.
7. **Инженеры по развертыванию (DevOps Engineers):** Инженеры по развертыванию занимаются настройкой и автоматизацией процессов развертывания и управления программным продуктом.
8. **Системные администраторы (System Administrators):** Эта группа отвечает за управление инфраструктурой и обеспечение надежности и безопасности серверов и сетей.
9. **Техническая поддержка (Support Team):** Группа технической поддержки обеспечивает обслуживание и поддержку конечных пользователей после развертывания продукта.
10. **Специалисты по безопасности (Security Specialists):** Эти специалисты занимаются анализом уязвимостей и обеспечением безопасности программного продукта.

Конечно, структура команды разработчиков может варьироваться в зависимости от размера и сложности проекта, а также от его спецификаций и требований. Важно правильно организовать и согласовать работу этих рабочих групп для успешного развития и внедрения программного обеспечения.

Функциональная характеристика каждой из основных групп в команде разработчиков:

1. **Разработчики (Developers):** Эта группа отвечает за создание кода и разработку программного продукта. Фронтенд-разработчики работают над пользовательским интерфейсом, а бэкенд-разработчики занимаются серверной частью приложения, базами данных и логикой.
2. **Тестировщики (Testers):** Задачей этой группы является тестирование программного обеспечения, выявление ошибок, дефектов и недоработок. Они также участвуют в создании тест-кейсов и тест-планов.
3. **Архитекторы (Architects):** Архитекторы определяют общую архитектуру программного продукта. Они решают, какие компоненты будут включены, как они будут взаимодействовать и как будет обеспечена масштабируемость и производительность.
4. **Дизайнеры (Designers):** Дизайнеры отвечают за создание пользовательского интерфейса (UI) и внешнего дизайна приложения. Их цель - сделать продукт привлекательным и удобным для пользователей.
5. **Аналитики (Business Analysts):** Аналитики анализируют бизнес-требования и взаимодействуют с заказчиками, чтобы понять, какое функциональное и нефункциональное ПО необходимо разработать.
6. **Менеджеры проекта (Project Managers):** Менеджеры проекта занимаются планированием, управлением ресурсами и координацией работы команды. Они следят за сроками и бюджетом проекта.
7. **Инженеры по развертыванию (DevOps Engineers):** Инженеры DevOps отвечают за автоматизацию процессов развертывания и управления программным продуктом, что позволяет быстро и надежно внедрять изменения.
8. **Системные администраторы (System Administrators):** Системные администраторы управляют серверами, сетями и инфраструктурой, обеспечивая их надежность, производительность и безопасность.
9. **Техническая поддержка (Support Team):** Группа технической поддержки предоставляет помощь и обслуживание конечным пользователям после развертывания программного продукта. Они решают проблемы и отвечают на вопросы пользователей.
10. **Специалисты по безопасности (Security Specialists):** Специалисты по безопасности обеспечивают анализ уязвимостей и внедрение мер безопасности, чтобы защитить программный продукт от угроз.

Каждая из этих групп выполняет ключевые функции в рамках разработки программного обеспечения и специализируется на определенных аспектах, гарантируя, что проект будет успешно завершен и продукт будет соответствовать требованиям заказчика и стандартам качества.

Цель 3: научиться управлять внедрение программных продуктов.

Контрольные вопросы

1. Методологии внедрения представляют собой структурированные наборы процессов, планов и руководств, разработанные для управления и координации внедрения новых технологий, программного обеспечения или изменений в организационной структуре.
   * Agile (гибкие методологии)
   * Waterfall (классический метод)
   * PRINCE2
   * PMI/PMBOK
   * Scrum
   * Lean
   * Six Sigma
   * ITIL
   * и многие другие.
2. Управление сроками проекта - это процесс определения, планирования и контроля временных рамок для завершения проекта. Оно необходимо для обеспечения выполнения работ в срок, чтобы проект прошел успешно и не превысил бюджет.
3. Процессы управления сроками проекта включают в себя:
   * Определение целей и задач проекта.
   * Разработка расписания и графика работ.
   * Оценка рисков и идентификация критических путей.
   * Мониторинг выполнения работ и коррекция плана, если необходимо.
4. Управление стоимостью - это процесс планирования, мониторинга и контроля бюджета проекта. Оно включает в себя управление расходами, чтобы проект оставался в рамках бюджета.
5. Процессы управления стоимостью проекта включают в себя:
   * Оценка бюджета и ресурсов.
   * Управление затратами и контроль бюджета.
   * Идентификация и управление рисками, связанными с финансами проекта.
6. Управление рисками - это процесс определения, анализа, мониторинга и управления рисками, которые могут повлиять на успешность проекта. Оно направлено на минимизацию негативных воздействий рисков и максимизацию возможностей.

*Цель 4:* научиться предвидеть результаты внедрения, видеть цель во внедрении.

Внедрение программного обеспечения - это процесс настройки программного обеспечения под определенные условия использования, а также обучения пользователей работе с программным продуктом.

Эффекты от внедрения. Эффекты можно разделить на две большие категории: эффект, получаемый на стратегическом уровне, т.е. на уровне бизнеса, и тактический эффект, который получат все участники процесса.

Стратегический эффект:

- обеспечение прозрачности и измеряемости достижения стратегических целей, т.е. коэффициент попадания по стратегическим целям возрастает;

- снижение вероятности проявления рисков в сфере информационных технологий, так как все проекты собираются в общую базу знаний, которая в дальнейшем используется как источник лучших практик компании в управлении проектами и рисками.

Тактический эффект:

- уменьшение сроков и снижение стоимости обработки новых запросов, т.е. обработки новых требований, новых потребностей бизнеса и снижение себестоимости исправления дефектов;

- улучшение качества IT-услуг – это вовремя выполненные работы с заранее определенным качеством;

- увеличение эффективности используемых ресурсов – хорошо организованный и формализованный процесс позволяет четко определить роли и ответственности каждого участника этого процесса.

Контрольные вопросы

1. Внедрение - это процесс внедрения, реализации или внесения чего-либо (какой-либо технологии, изменения, системы, метода и т. д.) в действие или практику. Этот термин часто используется в контексте бизнеса, информационных технологий, военных операций и других областях. Внедрение может включать в себя планирование, тестирование, оценку эффективности и мониторинг результатов для успешного внедрения нового элемента или процесса.
2. Внедрение программного обеспечения (ПО) может иметь различные эффекты, и результаты зависят от конкретного ПО, его целей и контекста. Вот несколько общих эффектов, которые можно ожидать от внедрения ПО:

* Увеличение производительности: Правильно выбранное и настроенное ПО может повысить производительность организации. Например, автоматизированные системы управления могут улучшить процессы и снизить человеческий труд.
* Улучшение качества и точности: ПО может помочь устранить ошибки и недочеты, связанные с ручными процессами. Это способствует улучшению качества продукции или услуг.
* Экономия времени и ресурсов: Автоматизация и оптимизация процессов с помощью ПО могут сократить расходы на рабочую силу и уменьшить время, требуемое для выполнения задач.
* Увеличение доступности данных: ПО может обеспечить легкий доступ к данным, что может улучшить принятие решений и сотрудничество в организации.

1. Внедрение программного обеспечения (ПО) может привести к разнообразным результатам, которые зависят от целей и конкретных потребностей организации. Вот некоторые из возможных результатов от внедрения ПО:

* Автоматизация процессов: ПО может автоматизировать множество задач, что сокращает ручной труд и уменьшает вероятность ошибок.
* Увеличение эффективности: Правильно выбранное ПО может повысить эффективность бизнес-процессов, ускоряя выполнение задач.
* Улучшение точности: Автоматизация и использование ПО могут снизить вероятность ошибок, что важно в критических операциях, таких как финансовые расчеты или медицинская диагностика.
* Экономия времени и ресурсов: ПО может сократить расходы на рабочую силу и уменьшить затраты на бумагу, офисное пространство и другие ресурсы.

Цель 5: научиться рационально распределять ответственность на этапах внедрения.

Контрольные вопросы

1. **Ключевые лица, принимающие решения по управлению проектом**, включают:
   * **Спонсор проекта**: Обеспечивает финансирование и поддержку проекта.
   * **Менеджер проекта**: Отвечает за планирование, организацию и выполнение проекта.
   * **Заказчик (клиент)**: Определяет требования к проекту и оценивает его успешное завершение.
   * **Исполнитель (команда проекта)**: Работает над выполнением проекта, осуществляет задачи и достигает целей.
2. **Спонсор проекта** несет ответственность за следующее:
   * Обеспечение финансирования проекта.
   * Утверждение бюджета и изменений бюджета.
   * Обеспечение ресурсами и поддержкой для проекта.
   * Оценка успешности проекта и достижение его целей.
3. **Менеджер проекта со стороны заказчика** отвечает за:
   * Коммуникацию с заказчиком.
   * Управление требованиями заказчика.
   * Обеспечение выполнения проекта в соответствии с ожиданиями заказчика.

**Менеджер проекта со стороны исполнителя** отвечает за:

* + Управление командой проекта.
  + Планирование и выполнение проекта.
  + Контроль бюджета и расписания.

1. **Обязанности бизнес-менеджера** варьируются в зависимости от конкретного проекта, но в целом включают в себя:
   * Определение бизнес-целей проекта.
   * Управление бизнес-стратегией проекта.
   * Обеспечение максимальной ценности для бизнеса от проекта.
   * Работу с клиентами и заинтересованными сторонами.
2. **Ключевые лица в команде управления проектом** могут включать:
   * Спонсора проекта.
   * Менеджера проекта.
   * Заказчика (клиента).
   * Менеджера проекта со стороны исполнителя.
   * Бизнес-менеджера.
   * Экспертов и консультантов.
   * Членов команды проекта.
3. **Менеджер по качеству** несет ответственность перед всей командой за обеспечение качества продукта или услуги проекта. Это включает в себя:
   * Разработку и внедрение системы управления качеством.
   * Оценку соответствия продукта или услуги стандартам и требованиям.
   * Организацию тестирования и контроля качества.
4. **В крупных проектах** могут быть организованы различные комитеты, такие как:
   * **Управляющий комитет**: Принимает стратегические решения о проекте.
   * **Комитет по управлению рисками**: Оценивает и управляет рисками проекта.
   * **Комитет по изменениям**: Рассматривает и утверждает изменения в проекте.
5. **Документирование распределения ролей и ответственности** членов команды проекта может осуществляться разными способами, включая:
   * **Организационная структура проекта**: Диаграмма, отражающая иерархию и роли в команде.
   * **Ролевые описания**: Документы, описывающие обязанности и задачи каждого члена команды.
   * **Матрица ответственности (RACI)**: Матрица, указывающая, кто ответственен, кто участвует, кто консультирует и кто информируется по конкретным задачам и ролям.
   * **Соглашения и документы проекта**: Включение информации о ролях и ответственностях в документацию проекта.