Лабораторная работа: Разработка перечня артефактов и протоколов проекта

Предмет: Инструментальные средства разработки программного обеспечения

Цель работы:

- •Получить практические навыки определения необходимых артефактов и протоколов для успешного ведения проекта разработки программного обеспечения.
- •Научиться классифицировать артефакты по стадиям жизненного цикла разработки.
- •Ознакомиться с принципами выбора и применения протоколов ведения проекта.

Задачи:

1. Выбрать условный проект разработки программного обеспечения (например, разработка веб-приложения для бронирования отелей, мобильного приложения для учета личных финансов, десктопного приложения для управления библиотекой).
2. Определить основные стадии жизненного цикла разработки для выбранного проекта (например, планирование, анализ требований, проектирование, разработка, тестирование, развертывание, поддержка).
3. Для каждой стадии определить перечень необходимых артефактов.
4. Определить необходимые протоколы для управления проектом и взаимодействия между членами команды.
5. Оформить результаты работы в виде отчета.

Теоретическое введение:

В процессе разработки программного обеспечения создается множество документов, моделей, исходных кодов и других материалов. Все эти материалы называются артефактами проекта. Артефакты помогают координировать работу команды, обеспечивают понимание целей и задач проекта, а также служат основой для дальнейшей разработки и поддержки.

Протоколы проекта – это набор правил и соглашений, определяющих способы взаимодействия между участниками проекта, процедуры принятия решений,

правила ведения документации и контроля версий. Наличие четких протоколов обеспечивает прозрачность и управляемость проектом.

Классификация артефактов по стадиям жизненного цикла:

- Планирование: Устав проекта, план проекта, оценка бюджета, оценка рисков, план управления ресурсами.
- Анализ требований: Спецификация требований (SRS), Use Case диаграммы, диаграммы прецедентов (UML), глоссарий терминов.
- Проектирование: Архитектурное описание, UML-диаграммы классов, диаграммы последовательностей, диаграммы состояний, ER-диаграммы (для работы с базами данных), описание пользовательского интерфейса.
- Разработка: Исходный код, документация к коду, модульные тесты.
- Тестирование: План тестирования, тестовые сценарии, отчеты о тестировании, журналы ошибок (bug reports).
- Развертывание: Инструкция по развертыванию, конфигурационные файлы, скрипты установки.
- Поддержка: Руководство пользователя, база знаний (FAQ), отчеты об инцидентах, планы по улучшению.

Примеры протоколов проекта:

- Протокол управления изменениями: определяет процедуру внесения изменений в требования, дизайн или код.
- Протокол контроля версий: определяет правила работы с системой контроля версий (например, Git).
- Протокол коммуникации: определяет каналы коммуникации между членами команды (например, Slack, email), частоту совещаний, формат отчетности.
- Протокол code review: определяет процедуру проверки кода другими разработчиками.
- Протокол оформления кода (coding style): устанавливает правила оформления кода для обеспечения единообразия.

•Протокол обработки ошибок (bug tracking): определяет процедуру регистрации, классификации и исправления ошибок.

Ход работы:

1.Выбор проекта: Опишите выбранный вами проект разработки программного обеспечения. Укажите его цели, целевую аудиторию, основные функциональные возможности. Например: “Разработка веб-приложения для бронирования номеров в отелях. Целевая аудитория: туристы, бизнес-путешественники. Основные функции: поиск отелей, просмотр информации об отелях, бронирование номеров, оплата бронирования, управление личным кабинетом.”

2.Определение стадий жизненного цикла: Перечислите основные стадии жизненного цикла разработки, которые вы будете использовать для выбранного проекта. Обоснуйте свой выбор. Например: “Для данного проекта будем использовать следующие стадии: планирование, анализ требований, проектирование, разработка, тестирование, развертывание, поддержка.”

3.Разработка перечня артефактов: Для каждой стадии жизненного цикла составьте список необходимых артефактов. Опишите кратко назначение каждого артефакта. Используйте таблицу для оформления результатов.

Стадия Жизненного Цикла	**Артефакт**	**Описание**
Планирование	Устав проекта	Документ, определяющий цели, задачи, рамки и заинтересованные стороны проекта.
Планирование	План проекта	Документ, определяющий расписание, бюджет, ресурсы и риски проекта.
Анализ требований	Спецификация требований (SRS)	Документ, описывающий функциональные и

Стадия Жизненного Цикла	**Артефакт**	**Описание**
		нефункциональные требования к разрабатываемой системе.
Анализ требований	Use Case диаграммы	Графическое представление взаимодействия пользователя с системой.
Проектирование	Архитектурное описание	Описание архитектуры системы, выбор технологических решений.
Проектирование	UML-диаграммы классов	Диаграмма, описывающая структуру классов и их взаимосвязи.
Разработка	Исходный код	Код программы на выбранном языке программирования.
Разработка	Документация к коду	Описание структуры кода, функций и классов.
Тестирование	План тестирования	Документ, определяющий стратегию тестирования, типы тестов и критерии приемки.
Тестирование	Тестовые сценарии	Описание шагов для проверки конкретной функциональности.
Развертывание	Инструкция по развертыванию	Документ, описывающий шаги для развертывания приложения на сервере или устройстве пользователя.
Поддержка	Руководство	Документ, описывающий как

Стадия Жизненного Цикла	Артефакт	Описание
	пользователя	пользоваться приложением.
Поддержка	Отчеты об инцидентах	Описание проблем, возникших в процессе эксплуатации приложения и способы их решения.

4.Определение протоколов проекта: Составьте список необходимых протоколов для управления проектом и взаимодействия между членами команды. Опишите кратко назначение каждого протокола. Используйте таблицу для оформления результатов.

Протокол	Описание
Протокол управления изменениями	Определяет процедуру внесения изменений в требования, дизайн или код. Все изменения должны быть зарегистрированы и одобрены.
Протокол контроля версий	Определяет правила работы с Git: ветвление, коммиты, pull requests. Используется ветка `main` для стабильной версии, `develop` для разработки.
Протокол коммуникации	Использование Slack для оперативной коммуникации, еженедельные совещания команды (онлайн или офлайн), email для официальной переписки.
Протокол code review	Каждый код должен быть проверен как минимум одним другим разработчиком перед слиянием в основную ветку.
Протокол оформления кода	Использование PEP 8 для Python, Google Java Style Guide для Java.

Протокол	Описание
Протокол обработки ошибок	Использование Jira для регистрации ошибок, присвоение приоритета и исполнителя, отслеживание статуса исправления.

5.Оформление отчета: Оформите результаты работы в виде отчета, включающего:

- •Титульный лист (название работы, предмет, ФИО студента, группа, дата).
- •Цель работы.
- •Задачи работы.
- •Теоретическое введение (кратко).
- •Описание выбранного проекта.
- •Перечень стадий жизненного цикла.
- •Таблицу с перечнем артефактов для каждой стадии.
- •Таблицу с перечнем протоколов проекта.
- •Выводы (краткое заключение о проделанной работе и полученных навыках).

Пример выводов:

“В ходе выполнения лабораторной работы были получены практические навыки определения необходимых артефактов и протоколов для успешного ведения проекта разработки программного обеспечения. Был выбран проект разработки веб-приложения для бронирования отелей, определены стадии жизненного цикла и для каждой стадии составлен перечень артефактов. Также был разработан перечень протоколов проекта, обеспечивающих эффективное взаимодействие между членами команды и управление процессом разработки. Данная работа позволила закрепить знания об инструментальных средствах разработки программного обеспечения и их применении на практике.”

Критерии оценки:

•Правильность выбора артефактов и протоколов для выбранного проекта.

•Полнота и обоснованность описания артефактов и протоколов.

•Структурированность и оформление отчета.

•Понимание теоретических основ.