# Отчет Лабораторная работа 4

**Необходимая теория из методички**

*Жизненный цикл ПО* — период от момента зарождения идеи ПО до:

* момента завершения его поддержки фирмой-разработчиком
* или завершения его сопровождения фирмой, выполнявшей сопровождение

В жизненном цикле ПО создаются и перерабатываются:

* модели предметной области, архитектура системы, описание требований
* прототипы системы и компонентов, исходный код
* проектная документация на систему в целом и на ее компоненты, пользовательская документация и документация администратора системы
* руководство по развертыванию, план проекта

Для этого привлекается *группа заинтересованных лиц*, включающая в себя разные профессии.

И для того, чтобы следовать концепции жизненного цикла ПО, существуют соответствующие стандарты.

В свою очередь, стандарты определяют элементы жизненного цикла, из которых выделяют *процессы*.

**Задание 1. Обосновать модель ЖЦ, наиболее подходящую для вашего проекта**

Мы выбирали модель жизненного цикла ПО двумя этапами:

В первую очередь мы сопоставили предложенные в тексте лабораторных работ стандарты, и отфильтровали стандарты методом исключения

* IEEE/EIA 12207-1997 подходит в случае, если проект оформляется по IEEE-документам. В нашем случае мы используем российское ТЗ.
* ISO/IEC 15288 слишком избыточен, потому что охватывает оборудование. А наш проект — это только комплекс программных средств.
* ISO/IEC 15504 (SPICE) может использоваться только для аудита.
* IEEE 1074-1997 – дополняющий стандарт для ISO/IEC 12207.
* Стандарты от ECMA/SEI/ANSI – второстепенные, нужны для описания конкретных средств реализации ПО. Например, как стандарт ECMA 262, являющийся спецификацией языка EcmaScript, который, в свою очередь, реализуют языки JavaScript и надстройка TypeScript, используя которые, можно реализовать конкретное ПО.

Остается только стандарт ISO/IEC 12207, который описывает весь цикл разработки и поддержки. Но, так как у нас российский проект с российским ТЗ, наиболее логично использовать стандарт ГОСТ Р ИСО/МЭК 12207-2010, который является переводом и адаптацией предыдущего стандарта.

Во вторую очередь мы уже подробно изучили реализованные в прошлых лабораторных работах техническое задание и ПО, и пришли к интересным наблюдениям.

Так как, в любом случае, разработка сложных систем требует управления сложностью, то разработанное нами ПО, используя архитектурные паттерны и средства языка C#, разделено на уровни абстракции с реализацией отдельных модулей, скрывая подробности реализации системы на более низких уровнях абстракции.

Помимо этого, в пункте 4.6. Специальные требования технического задания указано, что система должна иметь возможность *поэтапного расширения функционала*. Как раз для этого необходимо реализовывать модульную систему в соответствии с приведенной схемой управления сложностью выше.

Ставя точку в этом вопросе, мы подробно изучили ГОСТ Р ИСО/МЭК 12207-2010, и выявили следующее:

* В соответствии с пунктом 5.1.11 Декомпозиция процессов, процессы декомпозируют в отдельные процессы или процессы более низкого уровня с более четким уровнем детализации. Это соответствует нашим понятиям о разбиении системы на уровни абстракции с реализацией конкретных модулей.
* Построив гипотезу, что, концептуально наши идеи сходятся с тем, что описано в стандарте, мы обратились к главе 4. Термины и определения и выделили нужные нам понятия:  
  - Жизненный цикл: Развитие системы, продукта, услуги, проекта или других изготовленных человеком объектов, начиная со стадии разработки концепции и заканчивая прекращением применения.  
  - Стадия: Период в пределах жизненного цикла некоторого объекта, который относится к состоянию его описания и реализации.  
  - Модель жизненного цикла Структура процессов и действий, связанных с жизненным циклом, организуемых в стадии, которые также служат в качестве общей ссылки для установления связей и взаимопонимания сторон.  
  - Процесс: Совокупность взаимосвязанных или взаимодействующих видов деятельности, преобразующих входы в выходы.  
  - Система: Комбинация взаимодействующих элементов, организованных для достижения одной или нескольких поставленных целей.
* Далее, обратившись к пункту 5.1.2. Отношения между системами и программными средствами мы пришли к следующей концептуальной структуре стандарта. Для удобного представления опишем ее по знакомым нам уровням абстракции:  
  - Уровень абстракции — концепция жизненного цикла ПО. Состоит из стадий, которые включают в себя модели.  
  - Уровень абстракции — модель жизненного цикла ПО. Состоит из процессов и действий.  
  - Уровень абстракции — процессы. В свою очередь разбиваются на свои уровни абстракции: процессы жизненного цикла систем → процессы жизненного цикла ПО → более нижние детализирующие процессы.

Таким образом, мы пришли к выводу, что стандарт ГОСТ Р ИСО/МЭК 12207-2010 идеально подходит для нашего проекта.

**Задание 2. Выделить этапы выполнения проекта в соответствии с ЖЦ**

В соответствии с пунктом 5.1.12 Модели и стадии жизненного цикла, стандарт не требует использования какой-либо конкретной модели жизненного цикла, а также совокупности стадий, поэтому модель и стадии мы выбираем сами.

Для этого обратимся к календарному плану работ, сопоставим подходящие под реализации этапов процессы и выберем названия для этапов жизненного цикла, чтобы они соответствовали формулировкам стандарта.

|  |  |  |
| --- | --- | --- |
| **Этап календарного плана** | **Подходящие процессы из стандарта** | **Название этапа в ЖЦ** |
| Изучение предметной области.  Разработка предложений по реализации системы. | 6.4.1 Процесс определения требований правообладателей  6.4.2 Процесс анализа системных требований | Анализ и определение требований к системе |
| Проектирование структуры системы и программного модуля.  Разработка UML моделей проекта. | 6.4.3 Процесс проектирования архитектуры системы  7.1.3 Процесс проектирования архитектуры программных средств | Проектирование архитектуры системы и программных средств |
| Разработка программного модуля для контроля штрих-кодов товаров, учета расходования товара и коррекции склада. | 6.4.4 Процесс реализации  7.1.1 Процесс реализации | Реализация программных средств |
| Внедрение программного модуля в систему для одной кассы магазина.  Отладка модуля. | 7.2.4 Процесс верификации программных средств | Верификация программных средств |
| Тестирование модуля. | 7.2.5 Процесс валидации программных средств | Валидация программных средств |
| Внедрение программного модуля в систему для всех касс магазина. | 6.4.7 Процесс инсталляции программных средств | Инсталляция программных средств |
| Тестирование и отладка системы. | 6.4.6 Процесс квалификационного тестирования системы | Тестирование системы |
| Сдача системы заказчику.  Ввод системы в эксплуатацию. | 6.4.8 Процесс поддержки приемки программных средств | Приемка и эксплуатация системы |

|  |  |  |  |
| --- | --- | --- | --- |
| **Этап жизненного цикла** | **Процессы** | **Действия** | **Результаты** |
| Анализ и определение требований к системе | 6.4.1 Процесс определения требований правообладателей  6.4.2 Процесс анализа системных требований | * Подготовка и согласование ТЗ * Изучение предметной области (подходящих технологий) * Создание диаграммы с вариантами использования | * Техническое задание * Документ «Требования к системе» * Диаграмма с вариантами использования |
| Проектирование архитектуры системы и программных средств | 6.4.3 Процесс проектирования архитектуры системы  7.1.3 Процесс проектирования архитектуры программных средств | * Установка функциональных и нефункциональных системных требований * Создание UML-диаграмм классов, последовательностей, активностей и использования * Проектирование архитектуры программных средств по принципам Clean Architecture * Проектирование модулей * Определение интерфейсов и зависимостей | * UML-диаграммы классов, последовательностей, активностей и использования * Архитектурная спецификация * Описание интерфейсов |
| Реализация программных средств | 6.4.4 Процесс реализации  7.1.1 Процесс реализации | * Разработка кода в соответствии с архитектурой * Реализация модулей (сканирование, кассовые операции, отчеты, экспорт данных) * Настройка окружения и инфраструктуры (инструменты для разработки, а также внешние приложения, такие как базы данных) | * Исходный код программных средств * Файлы сборки * Отчет о реализации |
| Верификация программных средств | 7.2.4 Процесс верификации программных средств | * Отладка функциональности * Проверка соответствия архитектуры и кода требованиям * Статический анализ * Модульные тесты * Интеграционные тесты | * Отчет об отладке |
| Валидация программных средств | 7.2.5 Процесс валидации программных средств | * Тестирование системы в среде заказчика * Проверка корректности обработки операций (продажа, возврат, отчет) * Тестирование резервирования данных и устойчивости к сбоям | * Отчет о тестировании |
| Инсталляция программных средств | 6.4.7 Процесс инсталляции программных средств | * Развертывание системы на кассовых терминалах * Подключение сканеров и фиксальных регистраторов | * Акт внедрения системы |
| Тестирование системы | 6.4.6 Процесс квалификационного тестирования системы | * Проверка заказчиком системы на соответствие заявленным требованиям | * Отчет о тестировании системы |
| Приемка и эксплуатация системы | 6.4.8 Процесс поддержки приемки программных средств | * Корректировка выявленных дефектов системы * Повторное проведение приемочных испытаний * Подготовка руководств пользователя и администратора * Проведение обучения персонала | * Руководство пользователя * Руководство администратора * Отчет о тестировании системы * Акт приемки |

**Контрольные вопросы**

1. *Какие виды жизненного цикла программного обеспечения вы знаете?*

Мы знаем следующие:

* Каскадный — жизненный цикл, который мы реализовали. В нем строгая структура этапов с документированием.
* Инкрементальный — поэтапная разработка, где продукт можно частично использовать на ранних стадиях.
* Итеративный — цикл действий, в каждом из которых уточняются требования и улучшается функциональность
* На гибких методологиях (Agile, Scrum) – совмещение инкрементальной и итеративной разработки с минимальной документацией.
* DevOps – совмещение процессов разработки и эксплуатации. Непрерывная интеграция, доставка и мониторинг ПО.

Выбранный нами ГОСТ в соответствии с пунктом 5.1.12 Модели и стадии жизненного цикла не требует от нас конкретной модели жизненного цикла ПО и оставляет выбор за нами. Стандарт дает лишь набор правил, которым мы должны следовать при описании жизненного цикла ПО, а также процессов, через которые мы описываем.

1. *Понятие процесса. Виды процессов в соответствии с принятыми стандартами.*

Выбранный нами стандарт ГОСТ Р ИСО/МЭК 12207-2010 определяет процесс следующим образом: «Совокупность взаимосвязанных или взаимодействующих видов деятельности, преобразующих входы в выходы». Иными словами, процессы — это конкретные действия, которые принимаются группой заинтересованных лиц, т. е. людьми, реализующих жизненный цикл ПО, для решения конкретных задач разрабатываемого или сопровождаемого ПО или системы.

1. *Основные отличия ГОСТ Р ИСО/МЭК 12207-2010 от ГОСТ 19.102-77*

ГОСТ 19.102-77 — это стандарт, определяющий общие правила для документов, написанных по ГОСТ. ГОСТ Р ИСО/МЭК 12207-2010 — это стандарт, описывающий жизненный цикл ПО.