**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра ИС**

Курсовая работа

**по дисциплине: Технология разработки программного обеспечения**

Тема: Разработка ИС «Железнодорожный переезд» средствами UML и Microsoft Visio

|  |  |  |
| --- | --- | --- |
| Студент гр. 9362 |  | Малютин Д. Л. |
| Руководитель |  | Соничев А. В. |

Санкт-Петербург

2023

#### Задание

Студент: Малютин Дмитрий

Группа: 9362

Тема работы: Разработка ИС «Железнодорожный переезд» средствами UML и MS Visio.

Исходные данные:

Необходимо разработать ИС «Железнодорожный переезд» средствами UML и любыми другими средствами планирования, сделать краткое описание, выбрать модель жизненного цикла. Модель жизненного цикла должна основываться на ГОСТ Р ИСО/МЭК 12207-2010.

Дата выдачи задания:

Дата сдачи курсовой работы:

Дата защиты курсовой работы:

|  |  |  |
| --- | --- | --- |
| Студент |  | Малютин Д. Л. |
| Преподаватель |  | Соничев А. В. |

#### Аннотация

В данной курсовой работе изложена методология разработки информационной системы «Железнодорожный переезд», в составе жизненного цикла, схем проекта в объектно-ориентированной парадигме.

#### Abstract

This term paper outlines the methodology of development of the information system "Railway crossing", as part of life cycle, project schemes in an object-oriented paradigm.

#### Содержание

[1. Введение 3](#_Toc135832804)

[1. Описание ИС 3](#_Toc135832805)

[2. Выбор модели жизненного цикла 3](#_Toc135832806)

[2. Разработка ИС 42](#_Toc135832807)

[3. Диаграмма вариантов использования 42](#_Toc135832808)

[1. Описание 42](#_Toc135832809)

[2. Актёры 42](#_Toc135832810)

[3. Варианты использования 42](#_Toc135832811)

[4. Диаграмма классов 43](#_Toc135832812)

[4. Описание 43](#_Toc135832813)

[5. Классы 44](#_Toc135832814)

[5. Диаграмма состояний 46](#_Toc135832815)

[6. Описание 46](#_Toc135832816)

[6. Состояния 47](#_Toc135832817)

[7. Переходы 48](#_Toc135832818)

[7. Диаграмма деятельности 50](#_Toc135832819)

[8. Описание 50](#_Toc135832820)

[9. Состояния действий 51](#_Toc135832821)

[8. Диаграмма последовательности 55](#_Toc135832822)

[10. Описание 55](#_Toc135832823)

[11. Объекты 55](#_Toc135832824)

[12. Сообщения 55](#_Toc135832825)

[9. Диаграмма объектов 57](#_Toc135832826)

[13. Описание 57](#_Toc135832827)

[14. Объекты 57](#_Toc135832828)

[10. Диаграмма EPC 58](#_Toc135832829)

[15. Описание 58](#_Toc135832830)

[11. Диаграмма BPMN 61](#_Toc135832831)

[12. Описание 61](#_Toc135832832)

[3. Выводы 62](#_Toc135832833)

[4. Литература 63](#_Toc135832834)

# Введение

## Описание ИС

Цель данной работы создание информационной системы железнодорожного переезда.

Информационная система железнодорожного переезда представляет собой комплекс технических средств, предназначенных для обеспечения безопасности движения на железнодорожных переездах. Она включает в себя различные системы оповещения: **светофоры**, **шлагбаумы**, **звуковые сигналы**, которые позволяют контролировать состояние железнодорожного переезда и своевременно предупреждать водителей о приближении поезда.

Систему сигналов контролирует **контролёр**, обладающий **расписанием** прибытия на железнодорожный переезд **поездов**. Ранее эту функцию выполнял человек, но данная задача может быть автоматизирована. Иногда поезд может опаздывать или приходить раньше. Для этого существуют **датчики** на железнодорожных путях, расположенные на достаточно большом удалении от переезда для того, чтобы контролёр успел переключить сигналы.

Важно отметить, что информационная система железнодорожного переезда должна быть постоянно обновляемой и поддерживаемой в рабочем состоянии. Это необходимо для обеспечения безопасности на железнодорожном транспорте и предотвращения аварий. Для этого периодически систему посещает **ревизор**, меняющий лампочки, динамики и проводящий другие работы.

## Выбор модели жизненного цикла

Информационная система железнодорожного переезда – это система, которая будет эксплуатироваться постоянно и допущение ошибок в её работе неприемлемо, так-как от этого зависят жизни людей. Данная ИС является большой и будет эксплуатироваться многократно на одинаковых объектах, поэтому для разработки программной части подойдёт модель «Waterfall» или «Каскадная модель». Используя данную модель возможно разработать систему устойчивую к отказам со сведением ошибок к их отсутствию.

После введения системы в эксплуатацию на том или ином объекте она должна будет регулярно проходить ревизию: замена лампочек в светофорах, смена динамиков в системах звукового оповещения и прочие. Для этого необходимо предусмотреть возможность произведения ревизии системы.

Основания для выбора модели жизненного цикла представлены в таблице 1.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| № | Характеристика требований | Waterfall | V | RAD | Inc | Fast prototype | Evolv |
| 1 | Требования к проекту легко определимы и реализуемы | Да | Да | Да | Нет | Нет | Нет |
| 2 | Требования могут быть сформулированы в начале ЖЦ | Да | Да | Да | Да | Нет | Нет |
| 3 | Требования в процессе проектирования будут часто меняться | Нет | Нет | Нет | Нет | Да | Да |
| 4 | Требования необходимо демонстрировать с целью их определения | Нет | Нет | Да | Нет | Да | Да |
| 5 | Требуется проверка концепции программного средства или системы | Нет | Нет | Да | Нет | Да | Да |
| 6 | С ростом сложности системы в процессе ЖЦ требования будут изменяться или уточняться | Нет | Нет | Нет | Да | Да | Да |
| 7 | На ранних этапах разработки нужно реализовать основные требования | Нет | Нет | Да | Да | Да | Да |
| 8 | Проблемная область является новой для разработчиков | Нет | Нет | Нет | Нет | Да | Да |
| 9 | Инструментальные средства, являются новыми для разработчиков | Да | Нет | Нет | Нет | Нет | Да |
| 10 | На протяжении ЖЦ изменяются роли участников проекта | Нет | Нет | Нет | Да | Да | Да |
| 11 | Структура процесса разработки более значима для разработчиков, чем ее гибкость | Да | Да | Нет | Да | Нет | Нет |
| 12 | Распределение человеческих ресурсов не составляет большой проблемы | Да | Да | Да | Да | Нет | Нет |
| 13 | В процессе разработки предусмотрены оценки и проверки команды разработчиков | Да | Да | Нет | Да | Да | Да |
| 14 | Участие пользователей в разработке ограничено | Да | Да | Нет | Да | Нет | Да |
| 15 | В процессе разработки состояние ПО будет оцениваться пользователями | Нет | Нет | Нет | Да | Да | Да |
| 16 | Пользователи будут вовлечены в процесс разработки | Нет | Нет | Да | Нет | Да | Нет |
| 17 | Заказчик будет отслеживать ход выполнения проекта | Нет | Нет | Нет | Нет | Да | Да |
| 18 | Направление ПО ново для организации | Нет | Нет | Нет | Да | Да | Да |
| 19 | Разрабатывается расширение существующей системы | Да | Да | Да | Да | Нет | Нет |
| 20 | Проект будет крупно или среднемасштабным | Нет | Нет | Нет | Да | Да | Да |
| 21 | Ожидается длительная эксплуатация ПО | Да | Да | Нет | Да | Нет | Да |
| 22 | Необходим высокий уровень проектируемого ПО | Нет | Да | Нет | Да | Нет | Да |
| 23 | В процессе ЖЦ предполагается эволюция ПО | Нет | Нет | Нет | Да | Да | Да |
| 24 | Велика вероятность изменения ПО в процессе сопровождения | Нет | Нет | Нет | Да | Да | Да |
| 25 | Необходима быстрая разработка | Нет | Нет | Да | Да | Да | Да |
| 26 | Предполагается повторное использование компонентов | Нет | Нет | Да | Да | Да | Да |
| 27 | Ресурсы (время, деньги, инструменты, персонал) достаточны | Нет | Нет | Нет | Нет | Да | Да |
|  | Итог | 21 | 20 | 18 | 12 | 6 | 5 |

Таблица 1 – выбор жизненного цикла

Диаграмма модели жизненного цикла представлена на рисунке 1.

Диаграмма модели жизненного цикла также представлена в приложении 1.

Изображение выглядит как текст, снимок экрана, линия, диаграмма

Автоматически созданное описание

Рисунок 1 – диаграмма модели жизненного цикла

В соответствии с ГОСТ Р ИСО-МЭК 12207-2010 в модели жизненного цикла представлены следующие процессы:

7.1.2 Процесс анализа требований к программным средствам

7.1.2.1 Цель

Цель процесса анализа требований к программным средствам заключается в установлении требова­ний к программным элементам системы.

7.1.2.2 Выходы

В результате успешного осуществления процесса анализа требований к программным средствам:

a) определяются требования к программным элементам системы и их интерфейсам;

b) требования к программным средствам анализируются на корректность и тестируемость;

c) осознается воздействие требований к программным средствам на среду функционирования;

d) устанавливается совместимость и прослеживаемость между требованиями к программным средствам и требованиями к системе;

e) определяются приоритеты реализации требований к программным средствам;

f) требования к программным средствам принимаются и обновляются по мере необходимости;

д) оцениваются изменения в требованиях к программным средствам по стоимости, графикам работ и техническим воздействиям;

h) требования к программным средствам воплощаются в виде базовых линий и доводятся до сведения заинтересованных сторон.

7.1.2.3 Виды деятельности и задачи

При реализации проекта необходимо выполнять следующие виды деятельности и задачи в соответствии с принятыми в организации политиками и процедурами в отношении процесса анализа требований к программным средствам.

7.1.2.3.1 Анализ требований к программным средствам

Для каждого программного элемента (или элемента конфигурации, если он определен) данный вид деятельности состоит из решения следующих задач:

7.1.2.3.1.1 Исполнитель должен установить и документально оформить следующие требования к программным средствам (включая спецификации характеристик качества):

a) спецификации функциональных характеристик и возможностей, включая эксплуатационные, физические характеристики и условия окружающей среды, при которых будет применяться программная со­ ставная часть;

b) внешние интерфейсы к программной составной части;

c) квалификационные требования;

d) спецификации по безопасности, включая те спецификации, которые относятся к методам функционирования и сопровождения, влиянию окружающей среды и ущербу для персонала;

e) спецификации по защите, включая спецификации, связанные с угрозами для чувствительной ин­ формации;

f) спецификации эргономических факторов, включая спецификации, связанные с ручными операция­ ми, взаимодействием человека с оборудованием, ограничениями по персоналу и областям, требующим концентрации внимания и чувствительным к ошибкам человека и уровню его обученности;

д) описание данных и требования к базам данных;

h) инсталляция и требования к приемке поставляемого программного продукта в местах функционирования и сопровождения;

i) требования к документации пользователя;

j) операции пользователя и требования к их выполнению;

k) пользовательские требования к сопровождению.

7.1.2.3.1.2 Исполнитель должен оценить требования к программным средствам, учитывая критерии, перечисленные ниже. Результаты оценок должны быть документально оформлены.

a) прослеживаемость к системным требованиям и к системному проекту;

b) внешняя согласованность с системными требованиями;

c) внутренняя согласованность;

d) тестируемость;

е) осуществимость программного проекта;

f) осуществимость функционирования и сопровождения.

7.1.2.3.1.3 Исполнитель должен проводить ревизии в соответствии с 7.2.6.

7.1.3 Процесс проектирования архитектуры программных средств

7.1.3.1 Цель

Цель процесса проектирования архитектуры программных средств заключается в обеспечении проекта для программных средств, которые реализуются и могут быть верифицированы относительно требований.

7.1.3.2 Выходы

В результате успешной реализации процесса проектирования архитектуры программных средств:

a) разрабатывается проект архитектуры программных средств и устанавливается базовая линия, описывающая программные составные части, которые будут реализовывать требования к программным средствам;

b) определяются внутренние и внешние интерфейсы каждой программной составной части;

c) устанавливаются согласованность и прослеживаемость между требованиями к программным средствам и программным проектом.

7.1.3.3 Виды деятельности и задачи

При реализации проекта необходимо осуществлять следующие виды деятельности в соответствии с принятыми в организации политиками и процедурами в отношении процесса проектирования архитектуры программных средств.

7.1.3.3.1 Проектирование архитектуры программных средств

Для каждого программного элемента (или элемента конфигурации, если он определен) данный вид деятельности состоит из решения следующих задач:

7.1.3.3.1.1 Исполнитель должен преобразовать требования к программным составным частям в архитектуру, которая описывает верхний уровень его структуры и идентифицирует программные компоненты. Необходимо гарантировать, что все требования к программным составным частям распределяются по программным компонентам и в дальнейшем уточняются для облегчения детального проектирования. Архитектуру программной составной части необходимо документировать.

7.1.3.3.1.2 Исполнитель должен разработать и документально оформить проект верхнего уровня для внешних интерфейсов программной составной части и интерфейсов между ней и программными компонентами.

7.1.3.3.1.3 Исполнитель должен разработать и документально оформить проект верхнего уровня для базы данных.

7.1.3.3.1.4 Исполнитель должен разработать и документально оформить предварительные версии пользовательской документации.

7.1.3.3.1.5 Исполнитель должен определить и документировать требования к предварительному тестированию и график работ по комплексированию программных средств.

7.1.3.3.1.6 Исполнитель должен оценить архитектуру программной составной части, проекты по интерфейсам и базе данных, учитывая следующие критерии:

a) прослеживаемость к требованиям программной составной части;

b) внешняя согласованность с требованиями программной составной части;

c) внутренняя согласованность между программными компонентами;

d) приспособленность методов проектирования и используемых стандартов;

e) осуществимость детального проектирования;

f) осуществимость функционирования и сопровождения. Результаты оценок следует оформлять документально.

7.1.3.3.1.7 Исполнитель должен проводить ревизии в соответствии с 7.2.6.

7.1.4 Процесс детального проектирования программных средств

7.1.4.1 Цель

Цель процесса детального проектирования программных средств заключается в обеспечении проекта для программных средств, которые реализуются и могут быть верифицированы относительно установленных требований и архитектуры программных средств, а также существенным образом детализируются для последующего кодирования и тестирования.

7.1.4.2 Выходы

В результате успешного осуществления процесса детального проектирования программных средств: а) разрабатывается детальный проект каждого программного компонента, описывающий создаваемые программные модули;

b) определяются внешние интерфейсы каждого программного модуля и

c) устанавливается совместимость и прослеживаемость между детальным проектированием, требованиями и проектированием архитектуры.

7.1.4.3 Виды деятельности и задачи

При реализации проекта необходимо осуществлять следующие виды деятельности в соответствии с принятыми в организации политиками и процедурами в отношении процесса детального проектирования программных средств.

7.1.4.3.1 Детальное проектирование программных средств

Для каждой программной составной части (или составной части конфигурации, если она определена) данный вид деятельности состоит из решения следующих задач:

7.1.4.3.1.1 Исполнитель должен разработать детальный проект для каждого программного компонента программной составной части. Программные компоненты должны быть детализированы на более низком уровне, включающем программные блоки, которые могут быть закодированы, откомпилированы и проверены. Следует гарантировать, что все требования к программным средствам распределяются от программных компонентов к программным блокам. Детальный проект должен быть документально оформлен.

7.1.4.3.1.2 Исполнитель должен разработать и документально оформить детальный проект для внешних интерфейсов к программным составным частям, между программными компонентами и между программными блоками. Необходимо, чтобы детальный проект для интерфейсов позволял проводить кодирование без потребности в получении дополнительной информации.

7.1.4.3.1.3 Исполнитель должен разработать и документально оформить детальный проект базы данных.

7.1.4.3.1.4 Исполнитель должен совершенствовать пользовательскую документацию по мере необходимости.

7.1.4.3.1.5 Исполнитель должен определять и документировать требования к тестированию и графики работ по тестированию программных блоков. Необходимо, чтобы требования к тестированию включали в себя проведение проверок программных блоков при граничных значениях параметров, установленных в требованиях.

7.1.4.3.1.6 Исполнитель должен обновлять требования к тестированию и графики работ по комплексированию программных средств.

7.1.4.3.1.7 Исполнитель должен оценивать детальный проект для программных средств и требования к тестированию по следующим критериям:

a) прослеживаемость к требованиям программной составной части;

b) внешняя согласованность с архитектурным проектом;

c) внутренняя согласованность между программными компонентами и программными блоками;

d) соответствие методов проектирования и используемых стандартов;

e) осуществимость тестирования;

f) осуществимость функционирования и сопровождения. Результаты оценки должны быть документально оформлены.

7.1.4.3.1.8 Исполнитель должен проводить ревизии в соответствии с 7.2.6.

7.1.5 Процесс конструирования программных средств

7.1.5.1 Цель

Цель процесса конструирования программных средств заключается в создании исполняемых программных блоков, которые должным образом отражают проектирование программных средств.

7.1.5.2 Выходы

В результате успешного осуществления процесса конструирования программных средств:

a) определяются критерии верификации для всех программных блоков относительно требований;

b) изготавливаются программные блоки, определенные проектом;

c) устанавливается совместимость и прослеживаемость между программными блоками, требования­ ми и проектом;

d) завершается верификация программных блоков относительно требований и проекта.

7.1.5.3 Виды деятельности и задачи

При реализации проекта необходимо выполнять следующие виды деятельности и задачи в соответствии с принятыми в организации политиками и процедурами в отношении процесса конструирования программных средств.

7.1.5.3.1 Конструирование программных средств

Для каждой программной составной части (или составной части конфигурации, если она определена) данный вид деятельности состоит из решения следующих задач:

7.1.5.3.1.1 Исполнитель должен разработать и документально оформить:

a) каждый программный блоки базу данных;

b) процедуры тестирования и данные для тестирования каждого программного блока и базы данных.

7.1.5.3.1.2 Исполнитель должен тестировать каждый программный блок и базу данных, гарантируя, что они удовлетворяют требованиям. Результаты тестирования должны быть документально оформлены.

7.1.5.3.1.3 Исполнитель должен улучшать документацию пользователя при необходимости.

7.1.5.3.1.4 Исполнитель должен совершенствовать требования к тестированию и графики работ по комплексированию программных средств.

7.1.5.3.1.5 Исполнитель должен оценивать программный код и результаты испытаний, учитывая следующие критерии:

a) прослеживаемость к требованиям и проекту программных элементов;

b) внешнюю согласованность с требованиями и проектом для программных составных частей;

c) внутреннюю согласованность между требованиями к блокам;

d) тестовое покрытие блоков;

e) соответствие методов кодирования и используемых стандартов;

f) осуществимость комплексирования и тестирования программных средств; д) осуществимость функционирования и сопровождения.

Результаты оценки должны быть документально оформлены.

7.1.6 Процесс комплексирования программных средств

7.1.6.1 Цель

Цель процесса комплексирования программных средств заключается в объединении программных блоков и программных компонентов, создании интегрированных программных элементов, согласованных с проектом программных средств, которые демонстрируют, что функциональные и нефункциональные требования к программным средствам удовлетворяются на полностью укомплектованной или эквивалентной ей операционной платформе.

7.1.6.2 Выходы

В результате успешного осуществления процесса комплексирования программных средств:

a) разрабатывается стратегия комплексирования для программных блоков, согласованная с программным проектом и расположенными по приоритетам требованиями к программным средствам;

b) разрабатываются критерии верификации для программных составных частей, которые гарантируют соответствие с требованиями к программным средствам, связанными с этими составными частями;

c) программные составные части верифицируются с использованием определенных критериев;

d) программные составные части, определенные стратегией комплексирования, изготавливаются; е) регистрируются результаты комплексного тестирования;

f) устанавливаются согласованность и прослеживаемость между программным проектом и программными составными частями;

д) разрабатывается и применяется стратегия регрессии для повторной верификации программных составных частей при возникновении изменений в программных блоках (в том числе в соответствующих требованиях, проекте и кодах).

7.1.6.3 Виды деятельности и задачи

При реализации проекта необходимо выполнять следующие виды деятельности и задачи в соответствии с принятыми в организации политиками и процедурами в отношении процесса комплексирования программных средств.

7.1.6.3.1 Комплексирование программных средств

Для каждой программной составной части (или составной части конфигурации, если она определена) данный вид деятельности состоит из решения следующих задач:

7.1.6.3.1.1 Исполнитель должен разработать план комплексирования для объединения программных блоков и программных компонентов в программную составную часть. План должен включать в себя требо­вания к тестированию, процедуры, данные, обязанности и графики работ. План должен быть оформлен документально.

7.1.6.3.1.2 Исполнитель должен объединить программные блоки, программные компоненты и тесты, поскольку они разрабатываются в соответствии с планом комплексирования. Должны быть гарантии в том, что каждое такое объединение удовлетворяет требованиям к программной составной части и что составная часть комплексируется при завершении выполнения данной задачи. Результаты комплексирования и тестирования должны быть оформлены документально.

7.1.6.3.1.3 Исполнитель должен обновлять пользовательскую документацию по мере необходимости.

7.1.6.3.1.4 Исполнитель должен разработать и документально оформить для каждого квалификационного требования к программной составной части комплект тестов, тестовых примеров (входов, результатов, критериев тестирования) и процедур тестирования для проведения квалификационного тестирования программных средств. Разработчик должен гарантировать, что после комплексирования программная состав­ ная часть будет готова к квалификационному тестированию.

7.1.6.3.1.5 Исполнитель должен оценить план комплексирования, проект, код, тесты, результаты тестирования и пользовательскую документацию, учитывая:

a) прослеживаемость к системным требованиям;

b) внешнюю согласованность с системными требованиями;

c) внутреннюю согласованность;

d) тестовое покрытие требований к программной составной части;

e) приспособленность используемых методов и стандартов тестирования;

f) соответствие ожидаемым результатам;

д) осуществимость квалификационного тестирования программных средств; h) осуществимость функционирования и сопровождения.

Результаты оценки должны быть оформлены документально.

7.1.6.3.1.6 Исполнитель должен проводить ревизии в соответствии с 7.2.6.

7.1.7 Процесс квалификационного тестирования программных средств

7.1.7.1 Цель

Цель процесса квалификационного тестирования программных средств заключается в подтверждении того, что комплексированный программный продукт удовлетворяет установленным требованиям.

7.1.7.2 Выходы

В результате успешного осуществления процесса квалификационного тестирования программных средств:

a) определяются критерии для комплексированных программных средств с целью демонстрации соответствия с требованиями к программным средствам;

b) комплексированные программные средства верифицируются с использованием определенных критериев;

c) записываются результаты тестирования;

d) разрабатывается и применяется стратегия регрессии для повторного тестирования комплексированного программного средства при проведении изменений в программных составных частях.

7.1.7.3 Виды деятельности и задачи

При реализации проекта необходимо осуществлять следующие виды деятельности в соответствии с принятыми в организации политиками и процедурами в отношении процесса квалификационного тестирова­ ния программных средств.

7.1.7.3.1 Квалификационное тестирование программных средств

Для каждой программной составной части (или составной части конфигурации, если она определена) данный вид деятельности состоит из решения следующих задач:

7.1.7.3.1.1 Исполнитель должен проводить квалификационное тестирование в соответствии с квалификационными требованиями к программному элементу. Должна обеспечиваться гарантия того, что реализация каждого требования к программным средствам тестируется на соответствие. Результаты квалификационного тестирования должны быть документально оформлены.

7.1.7.3.1.2 Исполнитель должен обновлять пользовательскую документацию по мере необходимости.

7.1.7.3.1.3 Исполнитель должен оценивать проект, код, тесты, результаты тестирования и пользовательскую документацию, учитывая следующие критерии:

a) тестовое покрытие требований к программной составной части;

b) соответствие с ожидаемыми результатами;

c) осуществимость системного комплексирования и тестирования, если они проводятся;

d) осуществимость функционирования и сопровождения. Результаты оценки должны быть документально оформлены.

7.1.7.3.1.4 Исполнитель должен поддерживать проведение аудитов в соответствии с 7.2.7. Результаты аудитов должны быть документально оформлены. Если и технические, и программные средства разрабатываются или комплексируются, то аудиты могут быть отсрочены до тех пор, пока не будет выполнено системное квалификационное тестирование.

7.1.7.3.1.5 После успешного завершения аудитов (если они проводились) исполнитель должен обновить и подготовить поставляемый программный продукт для системного комплексирования, системного квалификационного тестирования, инсталляции программных средств или поддержки приемки программных средств.

7.2.1 Процесс менеджмента документации программных средств

7.2.1.1 Цель

Цель процесса менеджмента документации программных средств заключается в разработке и сопровождении зарегистрированной информации по программным средствам, созданной некоторым процессом.

7.2.1.2 Выходы

В результате успешного осуществления процесса менеджмента документации программных средств:

a) разрабатывается стратегия идентификации документации, которая реализуется в течение жизненного цикла программного продукта или услуги;

b) определяются стандарты, которые применяются при разработке программной документации;

c) определяется документация, которая производится процессом или проектом;

d) указываются, рассматриваются и утверждаются содержание и цели всей документации;

e) документация разрабатывается и делается доступной в соответствии с определенными стандартами;

f) документация сопровождается в соответствии с определенными критериями.

7.2.1.3 Виды деятельности и задачи

При реализации проекта необходимо осуществлять следующие виды деятельности в соответствии с принятыми в организации политиками и процедурами в отношении процесса менеджмента документации программных средств.

7.2.1.3.1 Реализация процесса

Данный вид деятельности состоит из решения следующей задачи:

7.2.1.3.1.1 Необходимо разрабатывать, документально оформлять и выполнять план, определяющий документы, которые производятся в течение жизненного цикла программного продукта. Идентифицированная документация должна включать в себя:

a) заголовок или название;

b) цели и содержание;

c) круг пользователей, которым она предназначена;

d) процедуры и ответственность при формировании исходных данных, разработке, ревизиях, модификации, утверждении, производстве, хранении, распределении, сопровождении и менеджменте кон­ фигурации;

e) графики создания промежуточных и окончательных версий.

7.2.1.3.2 Проектирование и разработка

Данный вид деятельности состоит из решения следующих задач:

7.2.1.3.2.1 Каждый идентифицированный документ должен быть разработан в соответствии с подходящими стандартами на документацию, регламентирующими носители, форматы, описание содержания, нумерацию страниц, размещение рисунков и таблиц, пометки о правах собственности и секретности, упаковку и другие элементы представления.

7.2.1.3.2.2 Источники правомерность использования исходных данных для документов должны быть подтверждены. Могут применяться автоматизированные средства поддержки документирования.

7.2.1.3.2.3 Подготовленные документы должны быть рассмотрены и отредактированы по формату, техническому содержанию и стилю представления в соответствии со стандартами на документацию. Перед выпуском адекватность этих документов должна быть подтверждена уполномоченным персоналом.

7.2.1.3.3 Производство

Данный вид деятельности состоит из решения следующих задач:

7.2.1.3.3.1 Документы должны изготавливаться и поставляться в соответствии с планом. При производстве и распределении документов может использоваться бумага, электронные или другие носители. Важные материалы должны храниться в соответствии с требованиями по содержанию записей, защищенности, сопровождению и резервированию.

7.2.1.3.3.2 В соответствии с процессом менеджмента конфигурации программных средств (см. 7.2.2) должны быть установлены необходимые средства управления.

7.2.1.3.4 Сопровождение

Данный вид деятельности состоит из решения следующих задач:

7.2.1.3.4.1 Должны выполняться задачи процесса сопровождения программных средств, которые необходимы при изменениях в документации (см. 6.4.10). Для документов, находящихся под воздействием менеджмента конфигурации, изменения должны проводиться в соответствии с процессом менеджмента конфигурации программных средств (см. 7.2.2).

7.2.2 Процесс менеджмента конфигурации программных средств

7.2.2.1 Цель

Цель процесса менеджмента конфигурации программных средств заключается в установлении и со­ провождении целостности программных составных частей процесса или проекта и обеспечении их доступ­ ности для заинтересованных сторон.

7.2.2.2 Выходы

В результате успешного осуществления процесса менеджмента конфигурации программных средств: а) разрабатывается стратегия менеджмента конфигурации программных средств;

b) составные части, порождаемые процессом или проектом, идентифицируются, определяются и вводятся в базовую линию;

c) контролируются модификации и выпуски этих составных частей;

d) обеспечивается доступность модификаций и выпусков для заинтересованных сторон;

e) регистрируется и сообщается статус составных частей и модификаций;

f) гарантируются завершенность и согласованность составных частей;

д) контролируются хранение, обработка и поставка составных частей.

7.2.2.3 Виды деятельности и задачи

Проект должен осуществлять следующие виды деятельности в соответствии с принятыми в организации политиками и процедурами в отношении процесса менеджмента конфигурации программных средств.

7.2.2.3.1 Реализация процесса

Данный вид деятельности состоит из решения следующей задачи:

7.2.2.3.1.1 Должен быть разработан план менеджмента конфигурации программных средств. План должен описывать: действия менеджмента конфигурации; процедуры и графики работ для выполнения этих действий; организацию (организации), ответственную за выполнение этих действий, и ее отношения с другими организациями, например разрабатывающими или сопровождающими программные средства. План должен быть документально оформлен и реализован.

7.2.2.3.2 Идентификация конфигурации

Данный вид деятельности состоит из решения следующей задачи:

7.2.2.3.2.1 Должна быть установлена схема для идентификации программных составных частей, а их версии должны контролироваться в рамках проекта. Для каждой программной составной части и ее версий должны быть определены документация, устанавливающая базовую линию, ссылки на версии и другие детали идентификации.

7.2.2.3.3 Управление конфигурацией

Данный вид деятельности состоит из решения следующей задачи:

7.2.2.3.3.1 Должны быть выполнены: идентификация и регистрация заявок на изменения; анализ и оценка изменений; принятие или отклонение заявок; реализация, верификация и выпуск модифицирован­ ной составной части. Должны проводиться проверочные испытания, на основании которых можно прослеживать каждую модификацию, ее причины и полномочия на проведение изменений. Должно осуществляться управление и аудит всего доступа к контролируемым программным составным частям, связанным с выполнением критических функций по безопасности или защите.

7.2.2.3.4 Отслеживание состояния конфигурации

Данный вид деятельности состоит из решения следующей задачи:

7.2.2.3.4.1 Должны выполняться записи менеджмента и отчеты о состоянии, которые отражают состояние и историю управляемых программных элементов, включая базовую линию. В отчеты о состоянии следует включать число изменений для проекта, последние версии программных составных частей, идентификаторы выпусков, номера выпусков и сравнение выпусков.

7.2.2.3.5 Оценка конфигурации

Данный вид деятельности состоит из решения следующей задачи:

7.2.2.3.5.1 Должны быть определены и гарантированы: функциональная завершенность программных составных частей относительно заданных требований и их физическая завершенность (отражают ли их структура и код текущее техническое описание).

7.2.2.3.6 Поставка и менеджмент выпуска

Данный вид деятельности состоит из решения следующей задачи:

7.2.2.3.6.1 Выпуск и поставка программных продуктов и документации должны официально управляться. Важные копии кодов и документации должны поддерживаться в течение срока жизни программного продукта. Код и документация, относящиеся к критическим функциям по безопасности и защите, должны обрабатываться, храниться, паковаться и доставляться в соответствии с политиками организаций, участвующих в этих процессах.

7.2.4 Процесс верификации программных средств

7.2.4.1 Цель

Цель процесса верификации программных средств заключается в подтверждении того, что каждые программный рабочий продукт и (или) услуга процесса или проекта должным образом отражают заданные требования.

7.2.4.2 Выходы

В результате успешного осуществления процесса верификации программных средств:

a) разрабатывается и осуществляется стратегия верификации;

b) определяются критерии верификации всех необходимых программных рабочих продуктов;

c) выполняются требуемые действия по верификации;

d) определяются и регистрируются дефекты;

e) результаты верификации становятся доступными заказчику и другим заинтересованным сторонам.

7.2.4.3 Виды деятельности и задачи

При реализации проекта необходимо осуществлять следующие виды деятельности и задачи в соответствии с принятыми в организации политиками и процедурами в отношении процесса верификации программных средств.

7.2.4.3.1 Реализация процесса

Данный вид деятельности состоит из решения следующих задач:

7.2.4.3.1.1 Должны быть определены условия реализации процесса, если проектом предусматриваются работы по верификации и необходима определенная степень организационной независимости этих работ. Требования проекта должны быть проанализированы на критичность. Критичность может быть оценена в терминах:

a) потенциального наличия необнаруженной ошибки в требованиях к системе или программным средствам, приводящей к гибели или травматизму персонала, невыполнению задания, финансовому ущербу, катастрофической утрате или повреждению оборудования;

b) степени отработки технологии программных средств и рисков, связанных с ее применением;

c) доступности фондов и ресурсов.

7.2.4.3.1.2 Если проектом предусматриваются работы по верификации, то должен быть установлен процесс верификации для проверки программного продукта.

7.2.4.3.1.3 Если проектом предусматриваются работы по независимой верификации, то должна быть выбрана квалифицированная организация, ответственная за проведение верификации. Данной организацией должны гарантироваться независимость и полномочия для проведения работ по верификации.

7.2.4.3.1.4 Должны быть определены программные продукты, требующие верификации, и конечные цели действий в течение жизненного цикла, основанные на области их применения, размерах, сложности и анализе критичности. Виды деятельности и задачи верификации, определенные в 7.2.4.3.2, включая соот­ветствующие методы, технические приемы и инструментарий для выполнения задач, должны быть выбраны в зависимости от конечных целей действий в течение жизненного цикла и программных продуктов.

7.2.4.3.1.5 Должен быть разработан и документально оформлен план проведения верификации на основе установленных задач верификации. План должен содержать действия в течение жизненного цикла и предмет верификации программных продуктов, необходимые задачи по верификации для каждого действия в течение жизненного цикла и программного продукта, связанные с ними ресурсы, ответственность и графики проведения работ. План должен предусматривать процедуры направления отчетов о верификации приобретающей стороне и другим заинтересованным организациям.

7.2.4.3.1.6 Должен быть реализован план проведения верификации. Проблемы и несоответствия, обнаруженные при проведении верификации, должны служить входами в процесс решения проблем (см. 7.2.8). Все возникшие проблемы должны быть решены, а обнаруженные несоответствия устранены. Результаты действий по верификации должны быть доступны приобретающей стороне и другим заинтересованным организациям.

7.2.4.3.2 Верификация

Данный вид деятельности состоит из решения следующих задач:

7.2.4.3.2.1 Верификация требований. Требования должны быть верифицированы с учетом следующих

критериев:

a) системные требования являются согласованными, выполнимыми и тестируемыми;

b) системные требования соответственно распределены по техническим, программным элементам и

ручным операциям согласно критериям проекта;

c) требования к программным средствам согласованы, выполнимы, проверяемы и точно отражают системные требования;

d) требования к программным средствам, связанные с безопасностью, защитой и критичностью, являются корректными, что показано соответствующими строгими методами.

7.2.4.3.2.2 Верификация проекта

Проект должен быть верифицирован с учетом следующих критериев:

a) проект корректируется, согласуется с требованиями и обеспечивает прослеживаемость к ним;

b) проект осуществляет надлежащую последовательность событий, входы, выходы, интерфейсы, логические связи, назначение сроков и размеров финансирования, а также обнаружение ошибок, локализацию и восстановление;

c) выбранный проект может быть выведен из требований;

d) проект корректно реализует требования по безопасности, защищенности и другим критическим свойствам, как показано соответствующими строгими методами.

7.2.4.3.2.3 Верификация кода

Код должен быть верифицирован с учетом следующих критериев:

a) код является следствием проекта и требований тестируемости, правильности и соответствует установленным требованиям и стандартам, относящимся к кодированию;

b) код осуществляет надлежащую последовательность событий, согласованные интерфейсы, корректные данные и поток команд управления, завершений, адекватного распределения времени и размеров финансирования, а также определение ошибок, локализацию и восстановление;

c) выбранный код может следовать из проекта или требований;

d) код корректно реализует требования по безопасности, защищенности и другим критическим свойствам, как показано соответствующими строгими методами.

7.2.4.3.2.4 Верификация комплексирования

Комплексирование должно быть верифицировано с учетом перечисленных ниже критериев:

a) программные компоненты и модули каждого программного элемента полностью и корректно комплексируются в программный элемент.

b) технические и программные элементы, а также ручные операции системы комплексируются в систему;

c) задачи комплексирования выполняются в соответствии с планом комплексирования.

7.2.4.3.2.5 Верификация документации

Документация должна быть верифицирована с учетом перечисленных ниже критериев:

a) документация является адекватной, полной и согласованной;

b) подготовка документации осуществляется своевременно;

c) менеджмент конфигурации документов следует установленным процедурам.

7.2.5 Процесс валидации программных средств

7.2.5.1 Цель

Цель процесса валидации программных средств заключается в подтверждении того, что требования выполняются для конкретного применения рабочего программного продукта.

7.2.5.2 Выходы

В результате успешного осуществления процесса валидации программных средств:

a) разрабатывается и реализуется стратегия валидации;

b) определяются критерии валидации для всей требуемой рабочей продукции;

c) выполняются требуемые действия по валидации;

d) идентифицируются и регистрируются проблемы;

e) обеспечиваются свидетельства того, что созданные рабочие программные продукты пригодны для

применения по назначению;

f) результаты действий по валидации делаются доступными заказчику и другим заинтересованным

сторонам.

7.2.5.3 Виды деятельности и задачи

При реализации проекта необходимо выполнять следующие виды деятельности и задачи в соответствии с принятыми в организации политиками и процедурами в отношении процесса валидации программных средств.

7.2.5.3.1 Реализация процесса

Данный вид деятельности состоит из решения следующих задач:

7.2.5.3.1.1 Должны быть определены условия реализации процесса, если проектом предусматриваются работы по валидации и необходима определенная степень организационной независимости этих работ.

7.2.5.3.1.2 Если проект предусматривает работы по валидации, то должен быть установлен процесс валидации для подтверждающей проверки системного или программного продукта. Должны быть выбраны задачи валидации, определенные ниже, в том числе связанные с ними методы, технологии и инструментарий.

7.2.5.3.1.3 Если проект предусматривает независимые работы по валидации, то должна быть выбрана квалифицированная организация, ответственная за проведение работ. Эта организация должна гарантировать независимость и полномочия при выполнении задач валидации.

7.2.5.3.1.4 Должен быть разработан и документально оформлен план валидации. План должен включать в себя, по крайней мере:

a) элементы, подвергаемые валидации;

b) задачи валидации, которые будут выполняться;

c) ресурсы, ответственности и графики выполнения работ по валидации;

d) процедуры передачи отчетов приобретающей стороне и другим сторонам.

7.2.5.3.1.5 План валидации должен быть выполнен. Проблемы и несоответствия, обнаруженные в процессе работ по валидации, должны быть переданы процессу решения проблем в программных средствах (см. 7.2.8). Все проблемы и несоответствия должны быть устранены. Результаты действий по валидации должны быть доступны приобретающей стороне и другим заинтересованным организациям.

7.2.5.3.2 Валидация

Данный вид деятельности состоит из решения следующих задач:

7.2.5.3.2.1 Готовить выбранные требования к тестированию, тестовые примеры и спецификации для анализа результатов тестирования.

7.2.5.3.2.2 Гарантировать, что требования к тестированию, тестовые примеры и спецификации отражают частные требования для конкретного применения.

7.2.5.3.2.3 Провести проверки выполнения 7.2.5.3.2.1 и 7.2.5.3.2.2, включая:

a) тестирование в условиях повышенной нагрузки, граничных значений параметров и необычных входов;

b) тестирование программного продукта на его способность изолировать и минимизировать влияние ошибок; то есть осуществлять плавную деградация после отказов, обращение к оператору за помощью в условиях повышенной нагрузки, граничных значений параметров и необычных входов;

c) тестирование того, что основные пользователи могут успешно решать намеченные задачи, используя данный программный продукт.

7.2.5.3.2.4 Подтвердить, что программный продукт удовлетворяет своему назначению.

7.2.5.3.2.5 Провести тестирование программного продукта в выбранных областях заданной среды

применения по назначению.

7.3.3 Процесс менеджмента повторного применения программ

7.3.3.1 Цель

Цель процесса менеджмента повторного применения программ заключается в планировании, создании, руководстве, управлении и мониторинге повторного применения программ в организации при систематическом использовании возможностей повторного применения.

7.3.3.2 Выходы

В результате успешного осуществления процесса менеджмента повторного применения программ:

a) определяется стратегия повторного применения программ в организации, в том числе назначение, область применения, конечные и промежуточные цели;

b) идентифицируются домены для потенциальных возможностей повторного применения;

c) оценивается возможность систематического повторного применения организацией;

d) оцениваются потенциальные возможности повторного применения каждого домена;

e) оцениваются предложения повторного применения для гарантии того, что повторно используемый продукт пригоден для предложенного приложения;

f) реализуется стратегия повторного применения в организации;

д) устанавливаются обратная связь, коммуникации и механизмы оповещения, которые функционируют между взаимодействующими сторонами;

h) контролируется и оценивается повторное применение программ.

7.3.3.3 Виды деятельности и задачи

При реализации проекта необходимо осуществлять следующие виды деятельности в соответствии с принятыми в организации политиками и процедурами в отношении процесса менеджмента повторного применения программ.

7.3.3.3.1 Инициация

Данный вид деятельности состоит из решения следующих задач:

7.3.3.3.1.1 Повторное применение программ в организации должно быть инициировано установлением стратегии организации в части повторного применения, которая включает в себя конечные цели, назначение, промежуточные цели и область применения.

7.3.3.3.1.2 Должен быть назван спонсор повторного применения.

7.3.3.3.1.3 Должны быть идентифицированы участники повторного применения программ и обозначены их роли.

7.3.3.3.1.4 Должна быть установлена функция, регулирующая повторное применение, для принятия полномочий и обязанностей по повторному применению программ в организации.

7.3.3.3.1.5 Должна быть установлена функция поддержки повторного применения программ.

7.3.3.3.2 Идентификация домена

Данный вид деятельности состоит из решения следующих задач:

7.3.3.3.2.1 Администратор повторного применения программ, которому помогают соответствующий менеджер, разработчики доменов, пользователи и разработчики программных средств, должен идентифицировать и документировать домены для исследования возможностей повторного применения или для осуществления намерения организации практиковать повторное применение.

7.3.3.3.2.2 Администратор повторного применения программ, которому помогают соответствующие менеджеры, разработчики доменов, пользователи и разработчики программных средств, должен оценить домены для гарантии точного отражения стратегии повторного применения в организации.

7.3.3.3.2.3 Администратор повторного применения программ должен проводить ревизии в соответствии с процессом ревизий. Разработчики программных средств, разработчики доменов и пользователи должны принимать участие в этих ревизиях.

7.3.3.3.2.4 Когда получение более обширной информации о доменах и планах организации относительно будущих программных продуктов становится доступным или когда проводится анализ доменов, сами домены могут быть уточнены, а область их распространения пересмотрена администратором повторного применения программ.

7.3.3.3.3 Оценки повторного применения

Данный вид деятельности состоит из решения следующих задач:

7.3.3.3.3.1 Администратор повторного применения программ должен оценивать возможности систематического повторного применения в организации.

7.3.3.3.3.2 Администратор повторного применения программ должен оценивать каждый домен, подлежащий рассмотрению, для определения потенциального успеха повторного применения в домене.

7.3.3.3.3.3 Администратор повторного применения программ должен выдавать рекомендации по уточнению стратегии и плана реализации повторного применения программ в организации, основанному на результатах оценок повторного применения.

7.3.3.3.3.4 Администратор повторного применения программ совместно с соответствующими приобретающими сторонами, поставщиками, разработчиками, операторами, сопровождающими сторонами, менеджерами активов, разработчиками доменов должен с приращением улучшать навыки, технологии, процессы повторного применения, структуру организации, а также показатели, которые вместе включают в

себя инфраструктуру повторного применения.

7.3.3.3.4 Планирование

Данный вид деятельности состоит из решения следующих задач:

7.3.3.3.4.1 Должен быть создан, документально оформлен и поддерживаться план реализации повторного применения программ для определения ресурсов и процедур по осуществлению повторного применения программ.

7.3.3.3.4.2 План должен анализироваться и оцениваться на полноту, осуществимость выполнения, а также на способность реализовать стратегию повторного применения в организации. К проведению оценки плана следует привлекать представителей, осуществляющих функцию регулирования повторного приме­ нения.

7.3.3.3.4.3 Принятие и поддержка плана реализации повторного применения программ должны вытекать из функции регулирования повторного применения и функций соответствующих менеджеров.

7.3.3.3.4.4 Администратор повторного применения программ должен проводить ревизии в соответствии с процессом ревизий. Представители от функции регулирования повторного применения и соответствующие менеджеры должны принимать участие в этих ревизиях.

7.3.3.3.5 Выполнение и управление

Данный вид деятельности состоит из решения следующих задач:

7.3.3.3.5.1 Действия, предусмотренные планом реализации повторного применения программ, должны выполняться в соответствии с планом.

7.3.3.3.5.2 Администратор повторного применения программ должен осуществлять мониторинг процесса продвижения повторного применения программ в соответствии со стратегией повторного применения программ в организации, а также проводить необходимые корректировки плана для реализации этой стратегии.

7.3.3.3.5.3 Проблемы и несоответствия, которые возникают в процессе выполнения плана реализации повторного применения программ, должны быть зарегистрированы и устранены.

7.3.3.3.5.4 Администратор повторного применения программ должен периодически подтверждать финансовую поддержку менеджмента, поддержку и обязательства по программе повторного применения.

7.3.3.3.6 Ревизии и оценивание

Данный вид деятельности состоит из решения следующих задач:

7.3.3.3.6.1 Администратор повторного применения программ должен периодически оценивать повтор­ но применяемые программы для достижения стратегии повторного применения в организации, продолжающейся пригодности и результативности повторного применения программ.

7.3.3.3.6.2 Администратор повторного применения программ должен представлять результаты оценок и информацию об извлеченных уроках для реализации функции регулирования повторного применения и соответствующим менеджерам.

7.3.3.3.6.3 Администратор повторного применения программ должен давать рекомендации и проводить изменения в повторно применяемых программах, соответственно расширяя и улучшая эти про­ граммы.

6.4.7 Процесс инсталляции программных средств

6.4.7.1 Цель

Цель процесса инсталляции программных средств заключается в установке программного продукта, удовлетворяющего заданным требованиям, в целевую среду применения.

6.4.7.2 Выходы

В результате успешного осуществления процесса инсталляции программных средств:

a) разрабатывается стратегия инсталляции программных средств;

b) разрабатываются критерии для инсталляции программных средств, предназначенные для демонстрации соответствия с требованиями к инсталляции программных средств;

c) программный продукт инсталлируется в целевую среду;

d) обеспечивается готовность программного продукта для использования в среде его применения.

6.4.7.3 Виды деятельности и задачи

В процессе реализации проекта должны осуществляться следующие виды деятельности и задачи в соответствии с принятыми в организации политиками и процедурами в отношении процесса инсталляции программных средств.

6.4.7.3.1 Инсталляция программных средств

Данный вид деятельности состоит из решения следующих задач:

6.4.7.3.1.1 Исполнитель должен разработать план инсталляции программного продукта вереду его применения, как определено в контракте. Ресурсы и информация, необходимые для инсталляции программного продукта, должны быть определенны и быть в наличии. Как установлено в контракте, исполнитель должен содействовать приобретающей стороне при проведении установки. Если инсталлируемый программный продукт заменяет существующую систему, то исполнитель должен поддерживать любые параллель­ но выполняемые действия, которые требуются в соответствии с контрактом. План инсталляции должен быть документирован.

6.4.7.3.1.2 Разработчик должен инсталлировать программный продукт в соответствии с планом инсталляции. Необходимо гарантировать, что базы данных и программный код инициализируются, выполняются и отменяются, как установлено в контракте. События, происшедшие при инсталляции, и их результаты должны документироваться.

6.4.9.1 Цель

Цель процесса функционирования программных средств заключается в применении программного продукта в предназначенной для него среде и обеспечении поддержки заказчиков программного продукта.

6.4.9.2 Выходы

В результате успешного осуществления процесса функционирования программных средств:

a) определяется стратегия функционирования;

b) определяются и оцениваются условия корректного функционирования программных средств в пред­ назначенной для них среде;

c) программные средства тестируются и настраиваются в предназначенной для них среде;

d) программные средства функционируют в предназначенной для них среде;

e) обеспечиваются содействие и консультации заказчикам программных продуктов в соответствии с условиями соглашения.

6.4.9.3 Виды деятельности и задачи

При реализации проекта необходимо осуществлять следующие виды деятельности и решать задачи в соответствии с принятыми в организации политиками и процедурами в отношении процесса функционирования программных средств.

6.4.9.3.1 Подготовка к функционированию

Данный вид деятельности состоит из решения следующих задач:

6.4.9.3.1.1 Оператор должен разрабатывать план и определять эксплуатационные стандарты для выполнения действий и задач этого процесса. План должен быть документирован и реализован.

6.4.9.3.1.2 Оператор должен определять процедуры для получения, регистрации, решения, прослеживания проблем и обеспечения обратной связи. Всякий раз, когда возникают проблемы, они должны быть зарегистрированы и введены в процесс решения проблем программных средств (см. 7.2.8).

6.4.9.3.1.3 Оператор должен устанавливать процедуры тестирования программного продукта в среде его эксплуатации для включения отчетов по проблемам, заявок на модификацию процесса сопровождения программных средств (см. 6.4.10) и реализации выпуска программного продукта для его функционального применения.

6.4.9.3.2 Активизация и контроль функционирования

Данный вид деятельности состоит из решения следующих задач:

6.4.9.3.2.1 Для каждого выпуска программного продукта оператор должен выполнить тестирование на соответствие функциональным требованиям и при условии удовлетворения заданных критериев выпустить программный продукт для применения по назначению.

6.4.9.3.2.2 Оператор должен гарантировать, что программный код и база данных инициируются, реализуются и заканчивают свое действие, как указано в плане.

6.4.9.3.2.3 Оператор должен активизировать систему в предназначенной для нее функциональной среде, чтобы представить образцы услуг или показать непрерывность предоставления услуг согласно их целевому назначению.

6.4.9.3.3 Применение по назначению

Данный вид деятельности состоит из решения следующих задач:

6.4.9.3.3.1 Система должна функционировать в предназначенной для нее среде согласно пользовательской документации.

6.4.9.3.4 Поддержка заказчика

Данный вид деятельности состоит из решения следующих задач:

6.4.9.3.4.1 Оператор должен обеспечивать содействие и консультации пользователей по их просьбе. Эти заявки и последующие действия должны быть зарегистрированы и проконтролированы.

6.4.9.3.4.2 Оператор должен направлять заявки пользователей (по мере необходимости) для выполнения в процессе сопровождения программных средств (см. 6.4.10). Эти заявки должны направляться по назначению, а сведения о действиях, которые планируются и предпринимаются, должны сообщаться инициаторам заявок. Все решения должны контролироваться для заключения об их результативности.

6.4.9.3.5 Решение проблем функционирования

Данный вид деятельности состоит из решения следующих задач:

6.4.9.3.5.1 Оператор должен направлять возникшие проблемы в процесс решения проблем в программных средствах для их устранения.

6.4.9.3.5.2 Если проблема, приведенная в отчете, относится к временному рабочему окружению перед тем, как будет реализовано постоянное решение, то инициатор отчета о проблеме должен представить вариант его применения в этом окружении. Текущие исправления используемого программного продукта, выпуски, включающие в себя ранее пропущенные функции или свойства, а также улучшения системы должны проводиться через процесс сопровождения программных средств (см. 6.4.10).

Данные пункты ГОСТа Р ИСО/МЭК 12207–2010 «Процессы жизненного цикла программных средств» должны бить выполнены. В случае неприменимости подпунктов они могут быть пропущены.

# Разработка ИС

## Диаграмма вариантов использования

### Описание

Главными действующими лицами в системе железнодорожного переезда являются транспорт в виде поездов и авто, и контролёр переезда.

Транспорт движется по своим путям: железной дороге или автодороге, а контролёр, используя датчики манипулирует сигнальными устройствами для указания транспорту действий.

### Актёры

Транспорт:

Авто

Поезд

Контролёр

Ревизор

### Варианты использования

Транспорт:

Совершить переезд

Ожидать

Контролёр:

Разрешить движение

Запретить движение

Получить время прибытия поезда

Ревизор:

Произвести ревизию

Диаграмма вариантов использования представлена на рисунке 2.

Диаграмма вариантов использования также представлена в приложении 2.

Изображение выглядит как снимок экрана, дизайн, шаблон

Автоматически созданное описание

Рисунок 2 – диаграмма вариантов использования

## Диаграмма классов

### Описание

Диаграмма классов является ключевой в построении информационной системы.

Диаграмма классов содержит классы и их отношения. Сами классы содержат поля и методы. Публичные поля и методы обозначены символом «+», приватные – символом «-».

В данной работе используется 3 вида отношения между классами: отношение зависимости (обозначается штрих-линией с пустой стрелкой), отношение агрегации (обозначается пустым ромбом со стороны целого и пустой стрелкой со стороны части), отношение обобщения (полая стрелка со стороны суперкласса).

### Классы

Далее представлены классы и их методы.

* Ревизор
  + Произвести ревизию() – сообщает контролёру о произведении ревизии, чтобы контролёр запретил всё движение на это время.
* Контролёр
  + Разрешить движение авто() – переводит сигналы в разрешающий для авто режим.
  + Разрешить движение поездов() – переводит сигналы в разрешающий для поездов режим.
  + Ревизия() – переводит сигналы в запрещающий режим для произведения ревизии системы.
  + Таймер() – устанавливает таймер приезда следующего поезда.
* Расписание – класс-обёртка для базы данных времени приезда поездов.
  + Получить время приезда поезда() – выдаёт следующее ближайшее время приезда поезда.
* Динамик
  + Включить() – включает звуковой сигнал.
  + Выключить() – выключает звуковой сигнал.
* Шлагбаум
  + Поднять() – поднимает шлагбаум.
  + Опустить() – опускает шлагбаум.
* Светофор
  + Включить зелёный() – переключает цвет светофора на зелёный.
  + Включить красный() – переключает цвет светофора на красный.
* Светофор для авто
  + Включить зелёный() – переключает цвет светофора авто на зелёный.
  + Включить красный() – переключает цвет светофора авто на красный.
* Светофор для поездов
  + Включить зелёный() – переключает цвет светофора поездов на зелёный.
  + Включить красный() – переключает цвет светофора поездов на красный.
* Транспорт
  + Двигаться() – совершить переезд.
  + Ожидать() – ожидать открытия переезда.
* Авто
  + Двигаться() – совершить переезд.
  + Ожидать() – ожидать открытия переезда.
* Поезд
  + Двигаться() – совершить переезд.
  + Ожидать() – ожидать открытия переезда.
  + Приблизится() – подаёт сигнал на датчик о приближении поезда.
* Датчик путей
  + Оповестить() – оповещает контролёра о неплановом приезде поезда.

Динамик, шлагбаум, светофоры, транспорт, датчик путей, контролёр содержат поле «состояние», которое описывает состояние объекта и используется для автоматического исправления ошибок. У контролёра состояния 3, так как присутствует дополнительное состояние произведения ревизии, у остальных состояния включён и выключен.

Контролёр содержит поле «время след поезда», используемое при произведении ревизии. Оно содержит последнее полученное от расписания время

Диаграмма классов представлена на рисунке 3.

Диаграмма классов представлена также в приложении 3.

Изображение выглядит как текст, снимок экрана, Шрифт, дизайн

Автоматически созданное описание

Рисунок 3 – диаграмма классов

## Диаграмма состояний

## Описание

Каждая прикладная система характеризуется не только структурой составляющих ее элементов, которая представляется, например, диаграммой классов, но и некоторым поведением или функциональностью. Для общего представления функциональности моделируемой системы предназначены диаграммы вариантов использования, которые на концептуальном уровне описывают поведение системы в целом. Для более детального представления поведения системы, которое обеспечивает необходимую функциональность, используются диаграммы состояний.

Диаграммы состояний позволяют описать возможные последовательности состояний и переходов, которые в совокупности характеризуют поведение системы в целом или ее элемента, как некоторого объекта, и представляют динамическое поведение сущностей, на основе спецификации их реакции на восприятие некоторых конкретных событий .

Основными элементами диаграммы состояний являются состояние и переход.

### Состояния

* Начало работы
  + Контролёр::запретить движение авто()
  + Светофор для авто::включить красный()
  + Светофор для поездов::включить зелёный()
  + Шлагбаум::опустить()
  + Динамик::включить()
  + Авто::ожидать()
  + Поезд::двигаться()
* Подготовка к движению авто
  + Контролёр::таймер()
  + Расписание::получить время приезда поезда()
* Открыть переезд для авто
  + Контролёр::разрешить движение авто()
  + Светофор для авто::включить зелёный()
  + Светофор для поездов::включить красный()
  + Шлагбаум::поднять()
  + Динамик::выключить()
* Переезд для авто открыт
  + Авто::двигаться()
  + Поезд::ожидать()
* Неплановое прибытие поезда
  + Датчик путей::оповестить()
  + Контролёр::запретить движение авто()
* Открыть переезд для поездов
  + Контролёр::запретить движение авто()
  + Светофор для авто::включить красный()
  + Светофор для поездов::включить зелёный()
  + Шлагбаум::опустить()
  + Динамик::включить()
* Переезд для поездов открыт
  + Авто::ожидать()
  + Поезд::двигаться()
* Подготовка ревизии
  + Контролёр::запретить движение авто()
  + Контролёр::ревизия()
  + Светофор для авто::включить красный()
  + Светофор для поездов::включить красный()
  + Шлагбаум::опустить()
  + Динамик::выключить()
* Ревизия
  + Ревизор::произвести ревизию()
  + Авто::ожидать()
  + Поезд::ожидать()

### Переходы

Основной цикл работы системы состоит в переходе от состояния «Переезд поездов открыт» к состоянию «Переезд для авто открыт» через состояния открытия переездов для авто или поездов.

Существует состояние «подготовка к движению авто», которое уточняет время прибытия следующего поезда.

Существует состояние «Неплановое прибытие поезда», которое переводит систему в состояние «Открыть переезд для поездов», минуя необходимость ожидания таймера прибытия поезда.

Существует состояние «Подготовка ревизии», запрещающее любое движение и выключающее все системы сигналов для их ревизии, а также переводящее систему в состояние «Ревизия», в котором ревизор выполняет необходимые ему действия.

В итоге, когда система морально устареет, будет произведена последняя ревизия, после чего система окончит функционирование.

Диаграмма состояний представлена на рисунке 4.

Диаграмма состояний также представлена в приложении 4.

Изображение выглядит как текст, снимок экрана, диаграмма, Шрифт

Автоматически созданное описание

Рисунок 4 – диаграмма состояний

## Диаграмма деятельности

### Описание

При моделировании поведения проектируемой или анализируемой системы возникает необходимость не только представить процесс изменения ее состояний, но и детализировать особенности алгоритмической и логической реализации операций, выполняемых системой. В используемых для этой цели традиционных блок-схемах алгоритмов отсутствуют средства для представления параллельных процессов и их синхронизации.

Для моделирования процесса выполнения операций в языке UML используются диаграммы деятельности. Применяемая в них графическая нотация во многом похожа на нотацию диаграммы состояний, поскольку на диаграммах деятельности также присутствуют обозначения состояний и переходов. Отличие заключается в семантике состояний и в отсутствии на переходах описаний событий. Каждое состояние на диаграмме деятельности соответствует выполнению некоторой элементарной операции, а переход в следующее состояние срабатывает только при завершении этой операции в предыдущем состоянии. Графически диаграмма деятельности представляется в форме графа, вершинами которого являются состояния действия, а дугами - переходы от одного состояния действия к другому. При этом каждое состояние может являться выполнением операции некоторого класса или ее части, что позволяет использовать диаграммы деятельности для описания реакций на внутренние события системы.

### Состояния действий

Каждому набору действий соответствует состояние. Некоторое количество состояний инкапсулируется в набор, и один или несколько таких наборов инкапсулируются в состояние.

* Начало работы
  + Закрыть движение авто
    - Контролёр::запретить движение авто()
    - Светофор для авто::включить красный()
    - Шлагбаум::опустить()
    - Динамик::включить()
  + Открыть движение поездов
    - Светофор для поездов::включить зелёный()
  + Ждать для авто
    - Авто::ожидать()
  + Двигаться для поезда
    - Поезд::двигаться()
* Подготовка к движению авто
  + Запустить таймер
    - Контролёр::таймер()
  + Проверить расписание
    - Расписание::получить время приезда поезда()
* Открыть переезд для авто
  + Открыть движение авто
    - Контролёр::разрешить движение авто()
    - Светофор для авто::включить зелёный()
    - Шлагбаум::поднять()
    - Динамик::выключить()
  + Закрыть движение поездов
    - Светофор для поездов::включить красный()
* Переезд для авто открыт
  + Двигаться для авто
    - Авто::двигаться()
  + Ждать для поезда
    - Поезд::ожидать()
* Неплановое прибытие поезда
  + Сигнал датчика
    - Датчик путей::оповестить()
    - Контролёр::запретить движение авто()
* Открыть переезд для поездов
  + Закрыть движение авто
    - Контролёр::запретить движение авто()
    - Светофор для авто::включить красный()
    - Светофор для поездов::включить зелёный()
    - Шлагбаум::опустить()
    - Динамик::включить()
* Переезд для поездов открыт
  + Ждать для авто
    - Авто::ожидать()
  + Двигаться для поезда
    - Поезд::двигаться()
* Подготовка ревизии
  + Запретить всё движение
    - Контролёр::запретить движение авто()
    - Контролёр::запретить движение поездов()
  + Включить режим ревизии
    - Контролёр::ревизия()
    - Светофор для авто::включить красный()
    - Светофор для поездов::включить красный()
    - Шлагбаум::опустить()
    - Динамик::выключить()
* Ревизия
  + Выполнить ревизию
    - Ревизор::произвести ревизию()
  + Ждать для всего транспорта
    - Авто::ожидать()
    - Поезд::ожидать()

Диаграмма деятельности представлена на рисунке 5.

Диаграмма деятельности также представлена в приложении 5.

Изображение выглядит как снимок экрана, диаграмма, текст, линия

Автоматически созданное описание

Рисунок 5 – диаграмма деятельности

## Диаграмма последовательности

### Описание

Кроме последовательности изменения состояний и алгоритма функционирования система определяется еще и взаимодействием образующих ее элементов. В UML такими элементами являются экземпляры классов – объекты. В процессе взаимодействия объекты обмениваются некоторой информацией в форме сообщений. Для описания этого процесса используются диаграммы последовательности.

### Объекты

* Ревизор
* Авто
* Поезд
* Контролёр
* Шлагбаум
* Светофор для авто
* Светофор для поездов
* Динамик
* Датчик
* Расписание

### Сообщения

В случае проектируемой системы сообщения ми являются вызовы функций классов, в частном случае при обращении к расписанию, сообщение содержит целочисленное значение, которое возвращается после обращения к расписанию.

Список сообщений:

* Динамик::включить()
* Динамик::выключить()
* Шлагбаум::поднять()
* Шлагбаум::опустить()
* Светофор для авто::включить красный()
* Светофор для авто::включить зелёный()
* Светофор для поездов::включить красный()
* Светофор для поездов::включить зелёный()
* Датчик путей::оповестить()
* Поезд::двигаться()
* Поезд::ожидать()
* Авто::двигаться()
* Авто::ожидать()
* Контролёр::разрешить движение авто()
* Контролёр::запретить движение авто()
* Контролёр::ревизия()
* Контролёр::таймер()
* Ревизор::произвести ревизию()
* Расписание::получить время приезда()
* return int

Диаграмма последовательности представлена на рисунке 6.

Диаграмма последовательности также представлена в приложении 6.

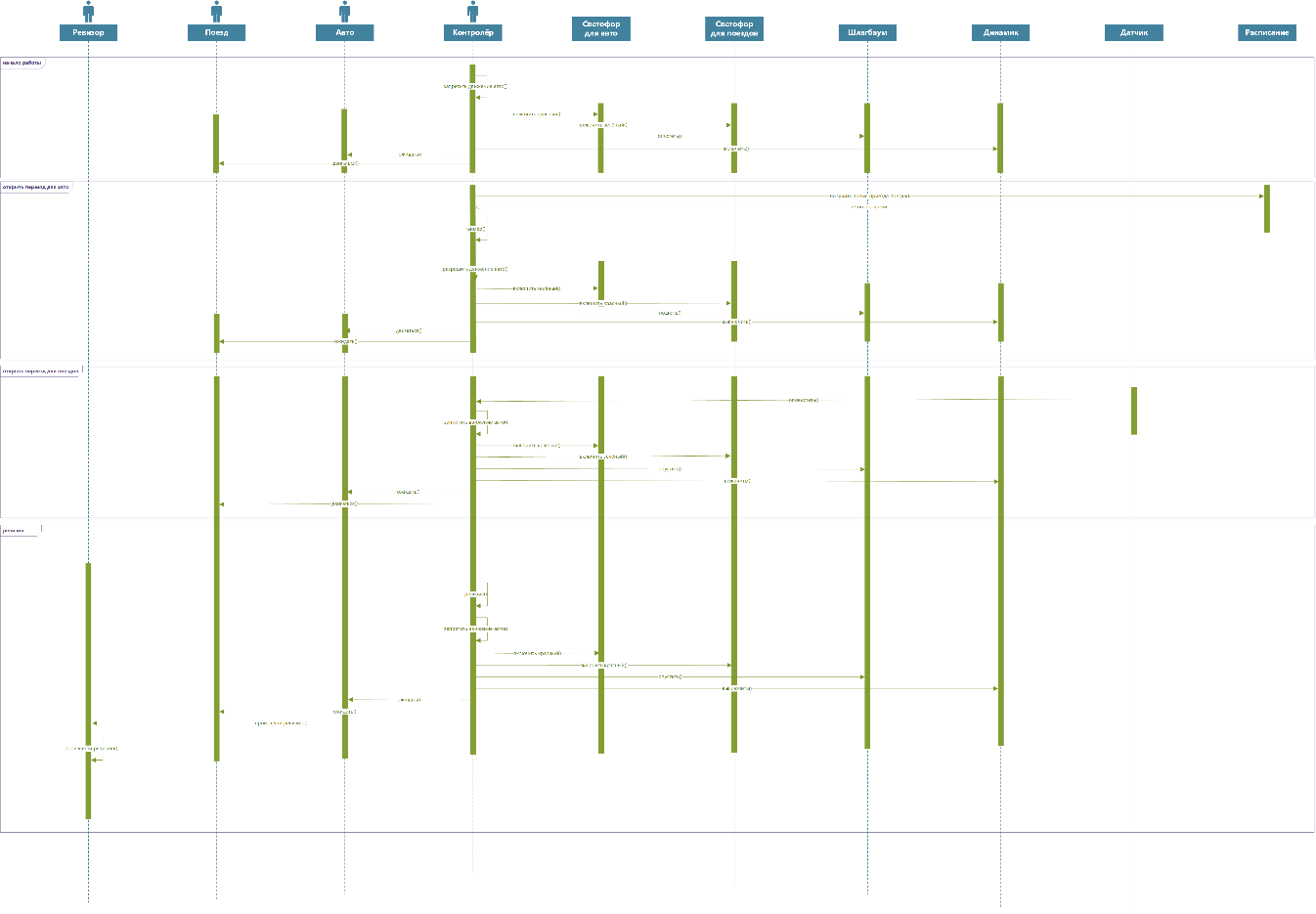


Рисунок 6 – диаграмма последовательности

## Диаграмма объектов

### Описание

В отличие от диаграмм классов, диаграммы объектов не позволяют полностью описать объектную структуру системы. У класса может быть большое количество различных экземпляров, а при наличии нескольких классов, связанных друг с другом отношениями, число возможных конфигураций объектов многократно возрастает, и диаграмма объектов, таким образом, есть отображение множества объектов и отношений между ними лишь в некоторый момент времени.

### Объекты

В случае данной информационной системы большинство объектов присутствует постоянно. Исключением являются: авто, поезда, ревизор. Авто и поезда появляются и пропадают в системе циклически, так-как в физическом плане являются пользователями системы. Ревизор в физическом плане является проверяющим, который появляется только при необходимости.

Диаграмма объектов представлена на рисунке 7.

Диаграмма объектов также представлена в приложении 7.

Изображение выглядит как диаграмма, снимок экрана, линия, План

Автоматически созданное описание

Рисунок 7 – диаграмма объектов

## Диаграмма EPC

### Описание

Нотация EPC (Event-Driven Process Chain – событийная цепочка процессов) используется для описания процессов нижнего уровня. Диаграмма процесса в нотации EPC, представляет собой упорядоченную комбинацию событий и функций. Для каждой функции могут быть определены начальные и конечные события, участники, исполнители, материальные и документальные потоки, сопровождающие её, а также проведена декомпозиция на более низкие уровни. Декомпозиция может производиться в нотациях EPC или BPMN.

Обозначения

Красный шестиугольник – событие, зелёный прямоугольник – процесс, – логическое и, – логическое или, XOR – исключающее или.

Диаграмма EPC представлена на рисунке 8.

Диаграмма EPC также представлена в приложении 8.

Изображение выглядит как текст, диаграмма, снимок экрана, Красочность

Автоматически созданное описание

Рисунок 8 – диаграмма EPC

## Диаграмма BPMN

## Описание

Нотация BPMN (Business Process Model and Notation – модель бизнес-процессов и нотация) используется для описания процессов нижнего уровня. Диаграмма процесса в нотации BPMN представляет собой алгоритм выполнения процесса. На диаграмме могут быть определены события, исполнители, материальные и документальные потоки, сопровождающие выполнение процесса. Каждый процесс может быть декомпозирован на более низкие уровни. Декомпозиция может производиться в нотациях BPMN или EPC.

В нотации BPMN выделяют пять основных категорий элементов:

1. соединяющие элементы;
2. зоны ответственности (пулы и дорожки);
3. объекты потока управления (события, процессы и шлюзы);
4. данные (объекты данных и базы данных);
5. артефакты.

Диаграмма BPMN представлена на рисунке 9.

Диаграмма BPMN также представлена в приложении 9.

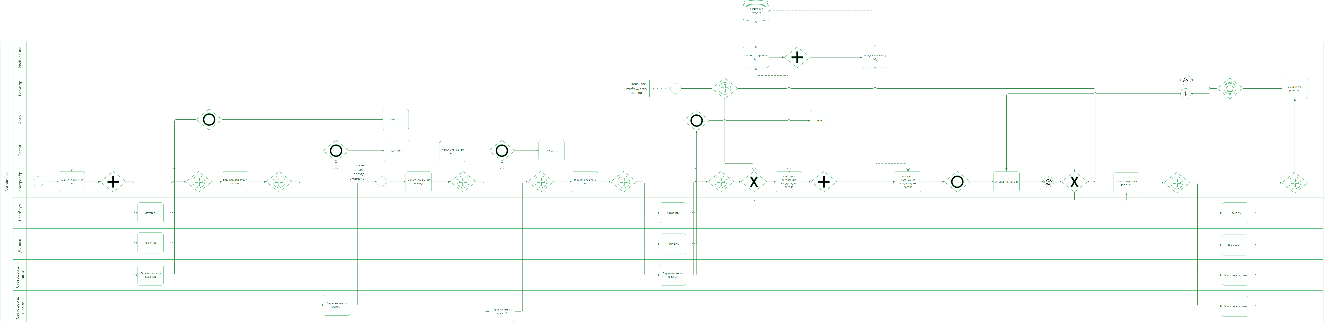


Рисунок 9 – диаграмма BPMN

# Выводы

В ходе выполнения курсовой работы была выбрана каскадная модель с частичной обратной связью. Разработанная модель жизненного цикла основывается на ГОСТ Р ИСО/МЭК 12207-2010. Были разработаны схемы для диаграммы вариантов использования, диаграммы классов, диаграммы состояний, диаграммы деятельности, диаграммы последовательности, диаграммы объектов, диаграммы EPC, диаграммы BPMN.

# Литература

1. Разработка программного обеспечения средствами UML и Aris: учебно-методическое пособие/ сост.: С.С. Егоров, А.В. Соничев. СПб.: Изд-во СПбГЭТУ «ЛЭТИ», 2016
2. ГОСТ Р ИСО/МЭК 12207-99
3. ГОСТ Р ИСО/МЭК 12207-2010
4. ГОСТ Р ИСО/МЭК 15271-2002
5. ГОСТ 10.701-90