**Содержание**

[1. ТЕХНИЧЕСКОЕ ЗАДАНИЕ 3](#_Toc107518563)

[1.1. Назначение и цели внедрения облачной платформы 3](#_Toc107518564)

[1.1.1. Назначение облачной платформы 3](#_Toc107518565)

[1.1.2. Цели внедрения облачной платформы 3](#_Toc107518566)

[1.2. Характеристика объекта автоматизации 4](#_Toc107518567)

[1.2.1. Общее описание 4](#_Toc107518568)

[1.2.2. Состав и порядок функционирования 4](#_Toc107518569)

[1.3. Общие требования к облачной платформе 11](#_Toc107518570)

[1.3.1. Требования к структуре и функционированию облачной платформы 11](#_Toc107518571)

[1.3.2. Дополнительные требования 15](#_Toc107518572)

[1.3.2.1. Требования к способам и средствам связи для информационного обмена между компонентами модуля информационной системы 15](#_Toc107518573)

[1.3.2.2. Требования к способу связи для информационного обмена между компонентами системы, в которую входит облачная платформа 15](#_Toc107518574)

[1.3.2.3. Требования к надежности 16](#_Toc107518575)

[1.3.2.4. Требования к безопасности 16](#_Toc107518576)

[1.3.2.5. Требования к эргономике и технической эстетике 17](#_Toc107518577)

[1.3.2.6. Требования к защите информации от несанкционированного доступа 18](#_Toc107518578)

[1.4. Требования к функциям, выполняемым облачной платформой 19](#_Toc107518579)

[1.4.1. Функция «Авторизироваться в системе» 20](#_Toc107518580)

[1.4.2. Функция «Тестирование кода» 21](#_Toc107518581)

[1.4.3. Функция «Загрузка ОС в облачную платформу» 22](#_Toc107518582)

[1.5. Требования к видам обеспечения 23](#_Toc107518583)

[1.5.1. Требования к информационному обеспечению 23](#_Toc107518584)

[1.5.2. Требования к алгоритмическому обеспечению 23](#_Toc107518585)

[1.5.3. Требования к программному обеспечению 24](#_Toc107518586)

[Приложение к Б1. Анализ аналогичных разработок 25](#_Toc107518587)

1. ТЕХНИЧЕСКОЕ ЗАДАНИЕ

1.1. Назначение и цели внедрения облачной платформы

1.1.1. Назначение облачной платформы

Облачная платформа предназначена для автоматизации разработки специализированного ПО по средствам размещение виртуальных машин и сетей на ФНПЦ АО «НПО Марс» для выполнения следующих процессов:

* создание виртуальной машины для связи с разрабатываемым репозиторием для его отладки;
* создание виртуальных локальных сетей для взаимодействия виртуальных машин и АРМ
* перенос разрабатываемых проектов в закрытую среду;

Облачная платформа должна быть развернута на ОС Astra Linux 1.6 “Смоленск”.

1.1.2. Цели внедрения облачной платформы

Облачная платформа внедряется с целью автоматизации разработки ПО, сокращения времени разработки, сохранности данных, разгрузки АРМ сотрудников.

Для реализации поставленных целей облачная платформа должна решать следующие задачи:

* создание виртуальных машин;
* создание виртуальных сетей для связи с АРМ;
* организация и хранение образов ОС;
* обеспечение шифрования трафика и данных

Облачная платформа должна использоваться в рамках предприятия ФНПЦ АО «НПО Марс» для использования информационным отделом при разработке специализированного ПО и для откладки систем.

1.2. Характеристика объекта автоматизации

1.2.1. Общее описание

В наше время многим предприятиям необходимо проводить оптимизацию производства для экономии времени и ресурсов. Для этого предприятия начинают осваивать информационные технологии, такие как облачные платформы. Облачная платформа должна быть гибкой, оперативно подстраиваться под новые и неожиданные ситуации.

Облачная платформа будет основана на Opennebula с использованием GitLab для контроля версии.

Инженеры и системные администраторы могут сконфигурировать облачную платформу под нужды программистов, для лучшего результата и для минимального простоя производства.

С каждым годом требование к ПО меняются, поэтому технологию нужно выбирать расширяемую, чтобы при возникновении форс-мажорных ситуаций не нужно было менять всю систему целиком.

1.2.2. Состав и порядок функционирования

Бизнес-процессы до внедрения облачной платформы представлены на диаграмме в нотации IDEF0 (Рисунок 1.1 - 1.3). Вербальное описание бизнес-процессов приведено в таблице 1.1.

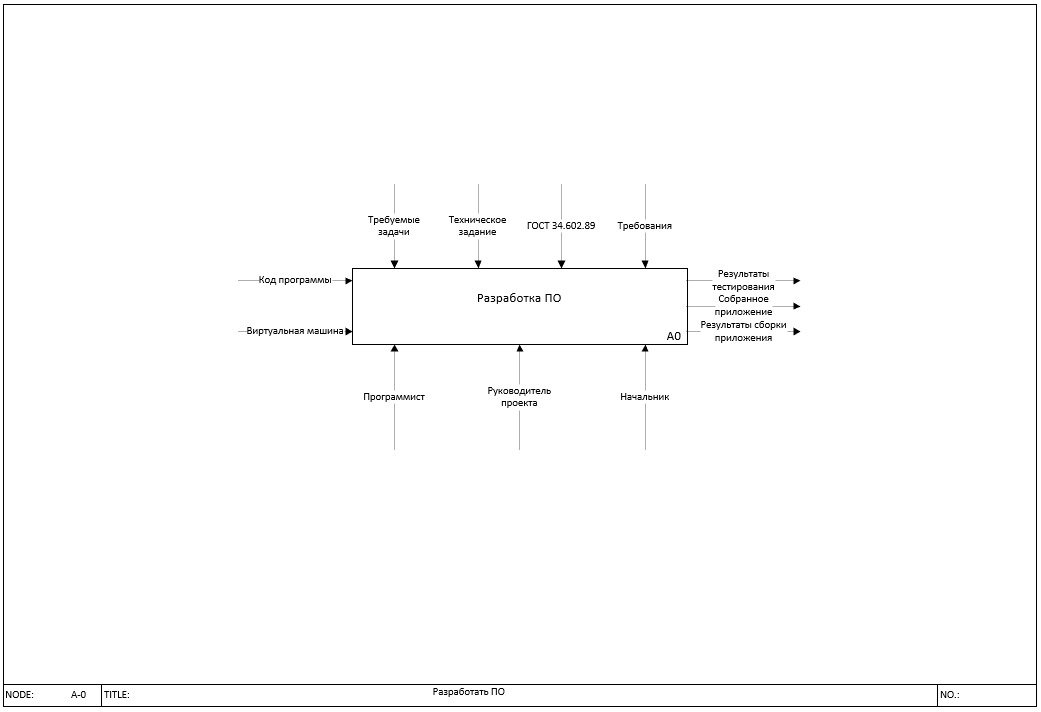


Рисунок 1.1 – IDEF0 до внедрения ОП, узел А-0

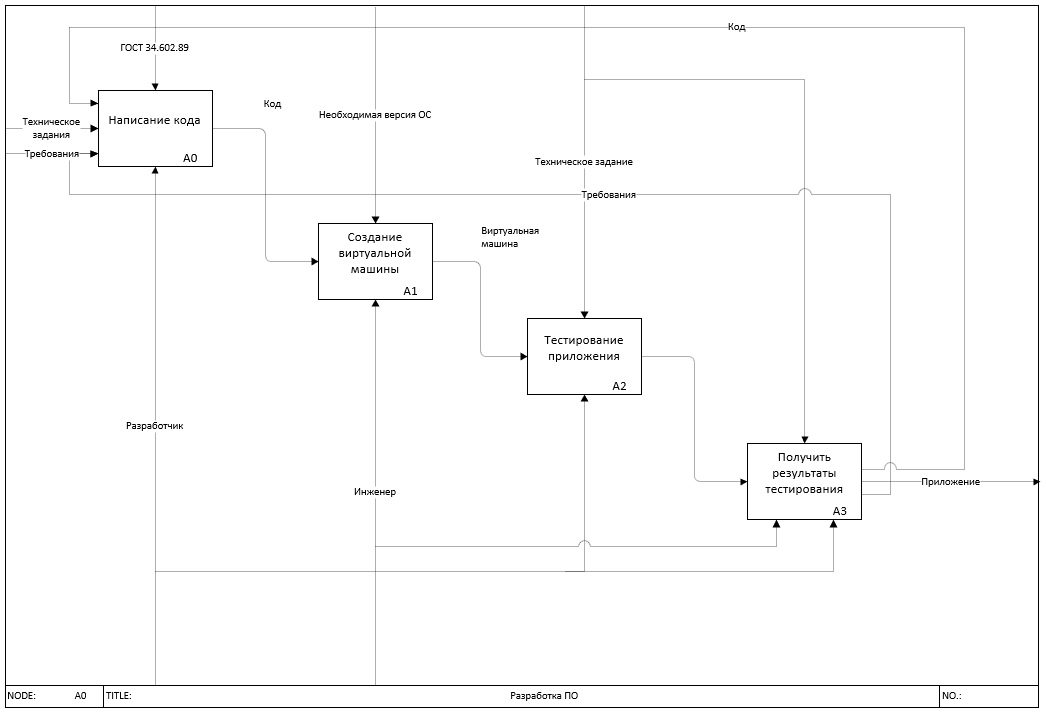


Рисунок 1.2 – IDEF0 до внедрения ОП, узел А0

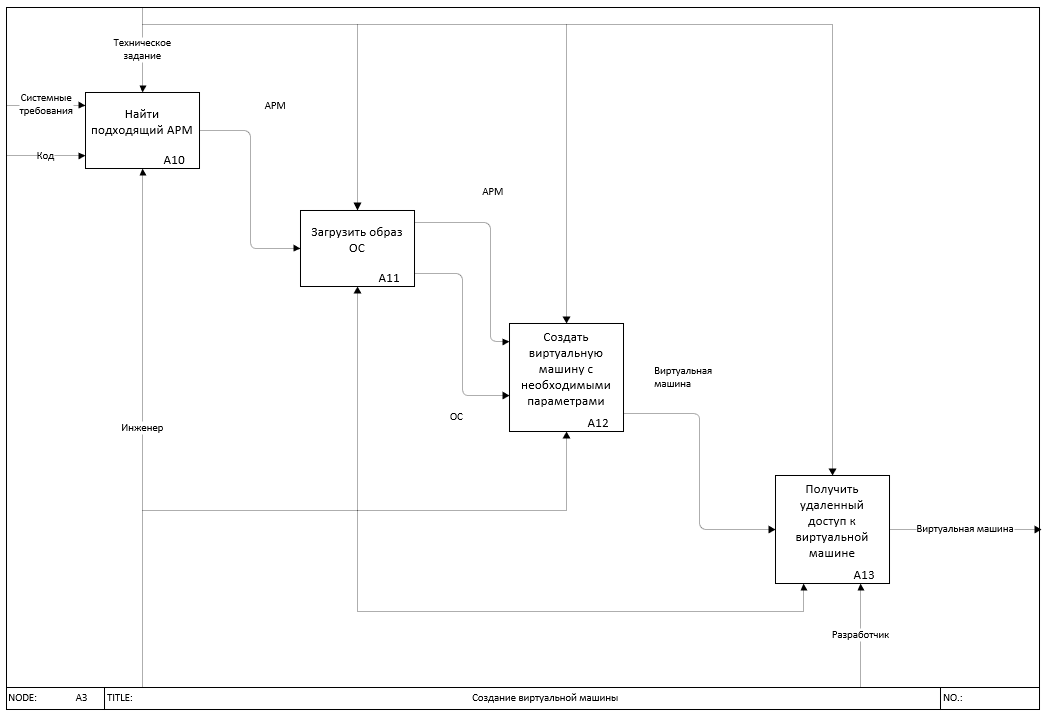


Рисунок 1.3 – IDEF0 до внедрения ОП, узел А3

Таблица 1.1

Вербальное описание бизнес-процессов до внедрения облачной платформы

|  |  |  |
| --- | --- | --- |
| **№ блока** | **Наименование блока** | **Описание блока** |
| **1** | **2** | **3** |
| А0 | Написать код | Руководитель проекта раздает задания программистам. Они изучают техническое задания и требования к оборудованию и начинают разрабатывать приложение |
| А1 | Создать виртуальную машину | Разработчики отправляют запрос инженерам на создание виртуальной машины с соответствием требованием и техническому заданию |
| А2 | Протестировать приложение | Разработчик, руководитель проекта и инженеры загружают код в виртуальную машину и проводят тестирование приложения – смотрят на поведение среды и ищут ошибки. |
| А3 | Получить результаты тестирования | Разработчики и инженеры сообщают о результатах тестирования руководителю проекта. Если результаты тестирования не удовлетворяют требованиям, то разработчики идут переписывать код. В другом случае – разработка заканчивается. |

Все вышеописанные бизнес-процессы полностью неавтоматизированные и выполняются вручную. Бизнес-процессы А1, А2, А3 должны быть автоматизированы.

Объектами автоматизации являются процессы создания ВМ, тестирования приложения и получения результатов после тестирования.

Процессы создания ВМ включают в себя:

* нахождение свободного АРМ;
* загрузки ОС на него;
* создание ВМ с необходимыми параметрами;
* получение удаленного доступа к ВМ.

Данные процессы осуществляются следующими специалистами:

* разработчиками
* инженерами

Бизнес-процессы после внедрения ОП представлены на диаграмме в нотации IDEF0 (Рисунок 1.4 – 1.6). Вербальное описание бизнес-процессов приведено в таблице 1.2.

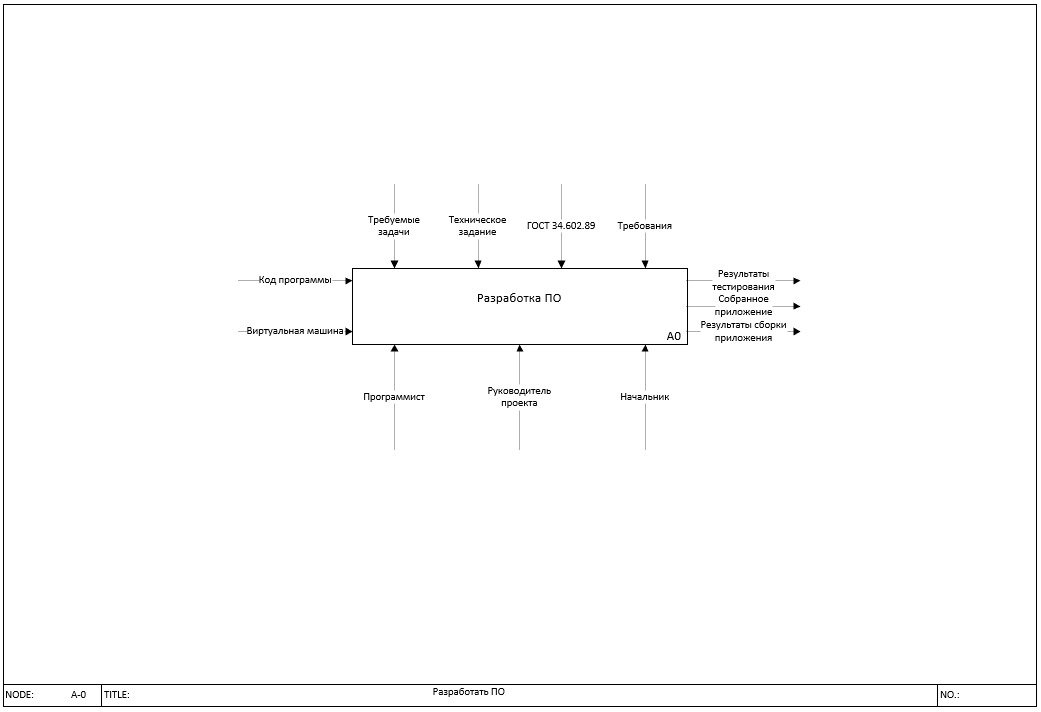


Рисунок 1.4 – IDEF0 после внедрения ОП, узел А-0

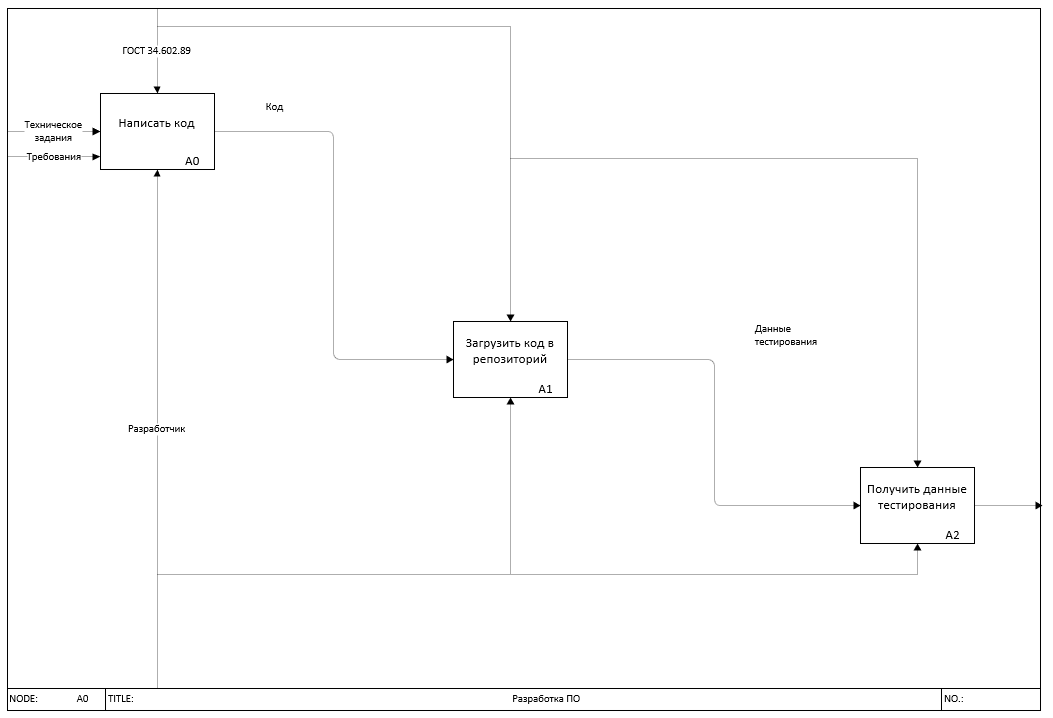


Рисунок 1.5 – IDEF0 после внедрения ОП, узел А0

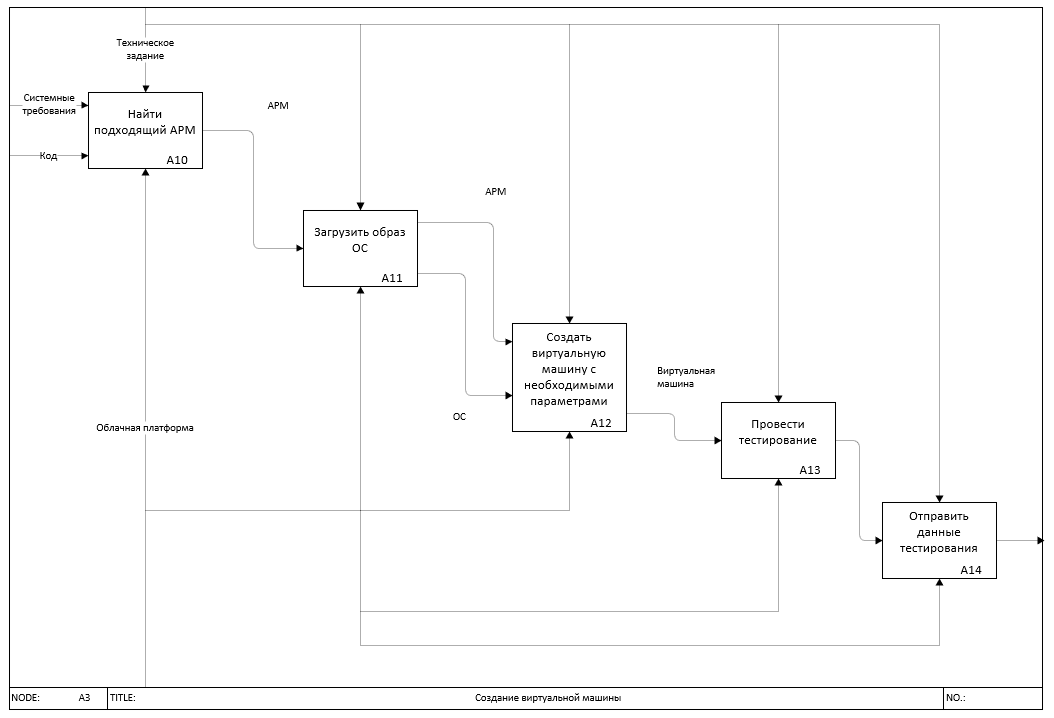


Рисунок 1.6 – IDEF0 после внедрения ОП, узел А3

Таблица 1.2

Вербальное описание бизнес-процессов после внедрения облачной платформы

|  |  |  |
| --- | --- | --- |
| **№ блока** | **Наименование блока** | **Описание блока** |
| А0 | Написать код | Руководитель проекта распределяет дисциплины по преподавателям. Преподаватели заполняют программную форму, выбирают учебную литературу из списка рекомендованной Руководителем ООП. |
| А1 | Загрузить код в репозиторий | После написания кода разработчик загружает его в репозиторий для контроля версий и для последующего тестирования |
| А2 | Получить данные тестирования | После загрузки кода в репозиторий облачная платформа производит необходимые операции и отправляет результат напрямую разработчику |

1.3. Общие требования к облачной платформе

1.3.1. Требования к структуре и функционированию облачной платформы

Структура взаимодействия облачной платформы с разработчиком и инженером приведена на диаграмме вариантов использования Use Case (Рисунок 1.7 – 1.8).

Ниже приведены типичные ходы событий для каждого из агентов системы в таблицах 1.3 - 1.5.

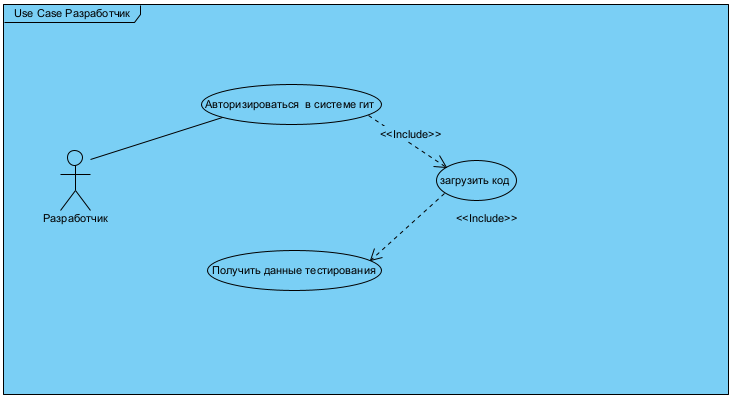


Рисунок 1.7 – Диаграмма Use Case, агент Разработчик

Таблица 1.3

Типичный ход событий агента – Разработчик

|  |  |
| --- | --- |
| **Действия исполнителя** | **Отклик системы** |
| **1** | **2** |
| 1. Разработчик запускает систему гит | 2. Приложение запускается на ПК |
| 3. Разработчик вводит логин и пароль и входит в систему | 4. Система анализирует введённые данные, подключается к БД, после успешной проверки предоставляет программную форму для дальнейшей работы Разработчик. |

Окончание таблицы 1.3

|  |  |
| --- | --- |
| **1** | **2** |
| 5. Разработчик выбирает действие «Загрузить код» | 6. Система открывает программную форму для загрузки кода |
| 7. Разработчик загружает код | 8. Система проверяет заполнение всех полей:  а) все заполнено, система добавляет данные в БД;  б) есть пропуски, система оповещает об этом |
| 9. Разработчик получает данные о тестировании | 10. Система считывает код и проверяет его на ошибки |

Дополнительная информация:

1. Если время выполнения программы для разработчика велико, то он прерывает процесс выполнения.

2. Необходимо обеспечить возможность выхода из варианта на любом этапе.

Таблица 1.4

Типичный ход событий – оповещение об ошибке

|  |  |
| --- | --- |
| **Действия исполнителя** | **Отклик системы** |
| 1. Пользователь пропускает поля для заполнения | 2. Система отмечает эти поля и просит заполнить их |
| 3. Пользователь заполняет поля | 4. Система снимает с полей метки и продолжает работу |
| 5. Пользователь вводит недопустимое/неверное значение | 6. Система отмечает эти поля и просит заполнить их |

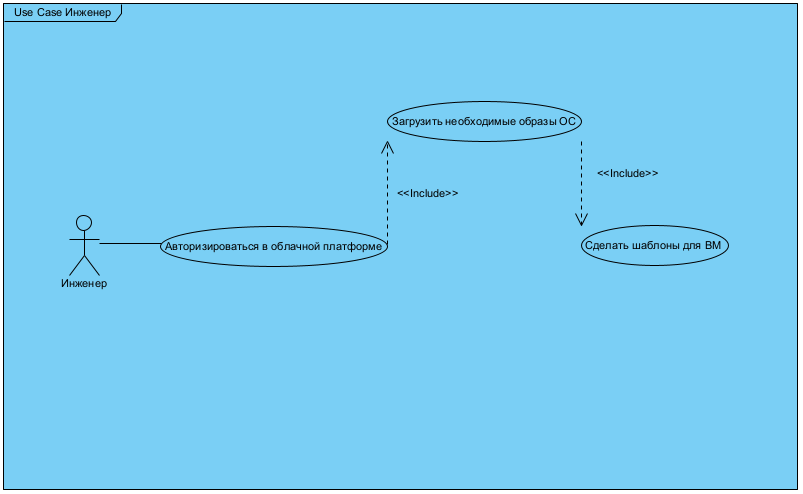


Рисунок 1.8 – Диаграмма Use Case, агент инженер

Таблица 1.5

Типичный ход событий агента – инженер

|  |  |
| --- | --- |
| **Действия исполнителя** | **Отклик системы** |
| 1. Инженер запускает приложение | 2. Приложение запускается на ПК |
| 3. Инженер вводит логин и пароль и входит в систему | 4. Система анализирует введённые данные, подключается к БД, после успешной проверки предоставляет программную форму для дальнейшей работы преподавателя |
| 5. Инженер загружает образы ОС | 6. Система подключается к БД, загружает образы в систему |
| 7. Инженер делает шаблоны для ВМ | 8. Система конвертирует список в рабочие данные и сохраняет изменения |

Для системы определены следующие режимы функционирования:

* нормальный режим функционирования;
* аварийный режим функционирования.

Основным режимом функционирования системы является нормальный режим. В нормальном режиме функционирования системы:

* клиентское программное обеспечение и технические средства пользователей и администратора системы обеспечивают возможность функционирования в любое время суток;
* серверное программное обеспечение и технические средства северов обеспечивают возможность круглосуточного функционирования, с перерывами на обслуживание;
* исправно работает оборудование, составляющее комплекс технических средств;
* исправно функционирует системное, базовое и прикладное программное обеспечение системы.

Для обеспечения нормального режима функционирования системы необходимо выполнять требования и выдерживать условия эксплуатации программного обеспечения и комплекса технических средств системы, указанные в соответствующих технических документах.

Аварийный режим функционирования системы характеризуется отказом одного или нескольких компонент программного, технического обеспечения. В случае перехода в предаварийный режим необходимо:

* завершить работу всех приложений, с сохранением данных;
* выключить рабочие компьютеры и все периферийные устройства;
* выполнить резервное копирование БД и ВМ.

После этого необходимо выполнить комплекс мероприятий по устранению причины перехода системы в аварийный режим.

1.3.2. Дополнительные требования

1.3.2.1. Требования к способам и средствам связи для информационного обмена между компонентами модуля информационной системы

Для обмена информацией, входящих в состав облачного хранилища, должна существовать локальная сеть с минимальной скорость 1гбит/с. У АРМ с клиентской частью облачной платформы должно быть 2 Ethernet разъема

1.3.2.2. Требования к способу связи для информационного обмена между компонентами системы, в которую входит облачная платформа

Связь модулей информационных систем, входящих в состав системы, представлена на диаграмме пакетов (Рисунок 1.10)

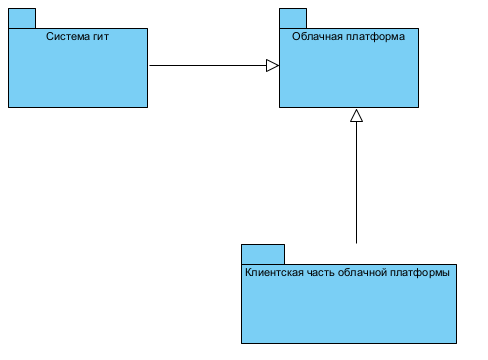


Рисунок 1.10 – Диаграмма пакетов

1.3.2.3. Требования к надежности

Система должна сохранять работоспособность и обеспечивать восстановление своих функций при возникновении следующих внештатных ситуаций:

* при сбоях в системе электроснабжения аппаратной части, приводящих к перезагрузке ОС, восстановление программы должно происходить после перезапуска ОС и запуска исполняемого файла системы;
* при ошибках в работе аппаратных средств восстановление функции системы возлагается на ОС;
* при ошибках, связанных с программным обеспечением (ОС и драйверы устройств), восстановление работоспособности возлагается на ОС.

Для защиты аппаратуры от бросков напряжения и коммутационных помех должны применяться сетевые фильтры [2].

1.3.2.4. Требования к безопасности

Система электропитания должна обеспечивать защитное отключение при перегрузках и коротких замыканиях в цепях нагрузки, а также аварийное ручное отключение.

Общие требования пожарной безопасности должны соответствовать нормам на бытовое электрооборудование. В случае возгорания не должно выделяться ядовитых газов и дымов. После снятия электропитания должно быть допустимо применение любых средств пожаротушения.

Факторы, оказывающие вредные воздействия на здоровье со стороны всех элементов системы (в том числе инфракрасное, ультрафиолетовое, рентгеновское и электромагнитное излучения, вибрация, шум, электростатические поля, ультразвук строчной частоты и т.д.), не должны превышать действующих норм.

1.3.2.5. Требования к эргономике и технической эстетике

Интерфейс системы должен быть понятным и удобным, не должен быть перегружен графическими элементами и должен обеспечивать быстрое отображение экранных форм. Навигационные элементы должны быть выполнены в удобной для пользователя форме. Средства редактирования информации должны удовлетворять принятым соглашениям в части использования функциональных клавиш, режимов работы, поиска, использования оконной системы. Ввод-вывод данных системы, прием управляющих команд и отображение результатов их исполнения должны выполняться в интерактивном режиме. Интерфейс должен соответствовать современным эргономическим требованиям и обеспечивать удобный доступ к основным функциям и операциям системы.

Интерфейс должен быть рассчитан на преимущественное использование манипулятора типа «мышь», то есть управление системой должно осуществляться с помощью набора экранных меню, кнопок, значков и т. п. элементов. Клавиатурный режим ввода должен применяться главным образом при заполнении и/или редактировании текстовых и числовых полей экранных форм.

Все надписи экранных форм, а также сообщения, выдаваемые пользователю (кроме системных сообщений), должны быть на русском языке.

Экранные формы должны проектироваться с учетом требований унификации:

* все экранные формы пользовательского интерфейса должны быть выполнены в едином графическом дизайне, с одинаковым расположением основных элементов управления и навигации;
* для обозначения сходных операций должны использоваться сходные графические значки, кнопки и другие управляющие (навигационные) элементы;
* внешнее поведение сходных элементов интерфейса должны реализовываться одинаково для однотипных элементов.

Система должна соответствовать требованиям эргономики.

Система должна обеспечивать корректную обработку аварийных ситуаций, вызванных неверными действиями пользователей, неверным форматом или недопустимыми значениями входных данных. В указанных случаях система должна выдавать пользователю соответствующие сообщения, после чего возвращаться в рабочее состояние, предшествовавшее неверной команде или некорректному вводу данных.

1.3.2.6. Требования к защите информации от несанкционированного доступа

Компоненты подсистемы защиты от несанкционированного доступа должны обеспечивать:

* идентификацию пользователя;
* проверку полномочий пользователя при работе с системой;
* разграничение доступа пользователей на уровне задач и информационных массивов;
* «слепые» пароли.

1.4. Требования к функциям, выполняемым облачной платформой

Программное обеспечение облачной платформы должно выполнять следующие функции:

* Создание ВМ
* Тестирование кода;
* Отправление данных тестирования.

Вышеуказанные функции подробно разобраны в пунктах 1.4.1. – 1.4.3. и представлены на диаграммах вариантов использования в нотации Use Case с точки зрения разработчика (Рисунок 1.11 – 1.13). Вербальное описание и типичный ход событий на каждую из данных функций представлены в таблицах 1.7 – 1.9.

1.4.1. Функция «Авторизироваться в системе»

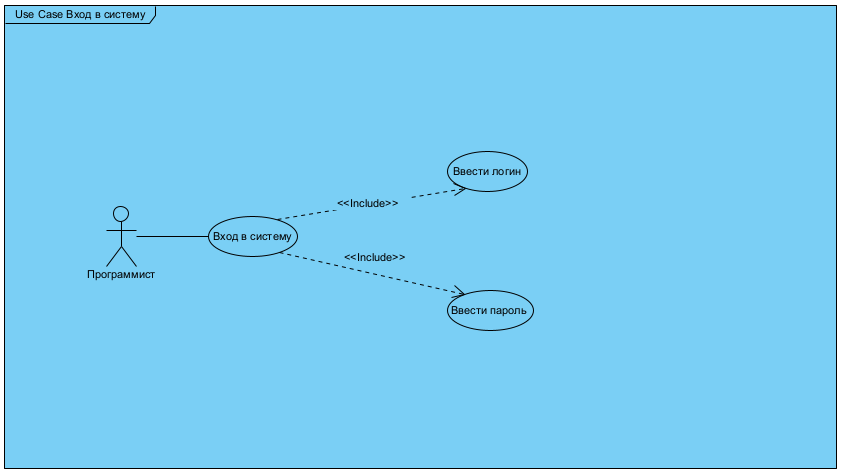


Рисунок 1.11 – Авторизироваться в системе

Таблица 1.7

Типичный ход событий

|  |  |
| --- | --- |
| **Действия исполнителя** | **Отклик системы** |
| 1. Пользователь запускает приложение | 2. Приложение запускается на ПК |
| 3. Пользователь вводит логин и пароль и входит в систему | 4. Система анализирует введённые данные, подключается к БД, после успешной проверки предоставляет программную форму для дальнейшей работы |

1.4.2. Функция «Тестирование кода»

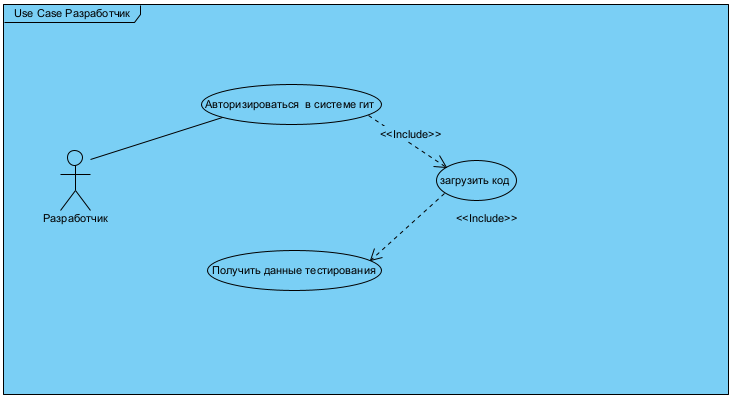


Рисунок 1.12 – Диаграмма Use Case, агент Разработчик

Таблица 1.8

Типичный ход событий агента – Разработчик

|  |  |
| --- | --- |
| **Действия исполнителя** | **Отклик системы** |
| **1** | **2** |
| 1. Разработчик запускает систему гит | 2. Приложение запускается на ПК |
| 3. Разработчик вводит логин и пароль и входит в систему | 4. Система анализирует введённые данные, подключается к БД, после успешной проверки предоставляет программную форму для дальнейшей работы Разработчик. |
| 5. Разработчик выбирает действие «Загрузить код» | 6. Система открывает программную форму для загрузки кода |
| 7. Разработчик загружает код | 8. Система проверяет заполнение всех полей:  а) все заполнено, система добавляет данные в БД;  б) есть пропуски, система оповещает об этом |
| 9. Разработчик получает данные о тестировании | 10. Система считывает код и проверяет его на ошибки |

1.4.3. Функция «Загрузка ОС в облачную платформу»

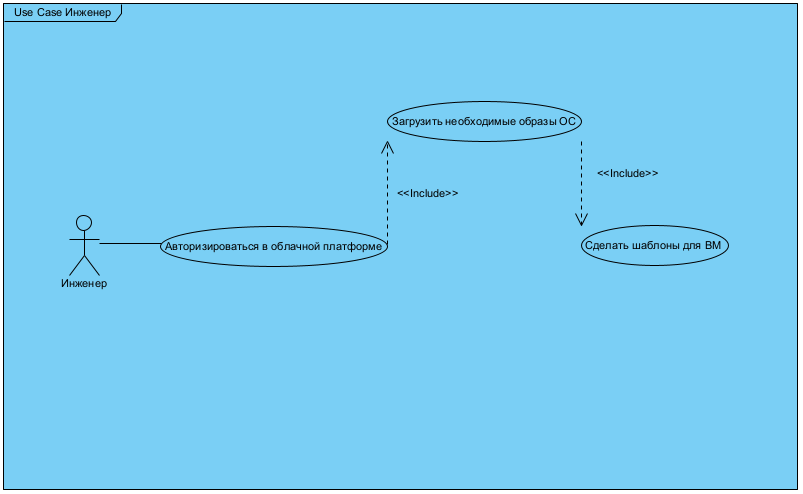


Рисунок 1.13 – Диаграмма Use Case, агент инженер

Таблица 1.9

Типичный ход событий агента – инженер

|  |  |
| --- | --- |
| **Действия исполнителя** | **Отклик системы** |
| 1. Инженер запускает приложение | 2. Приложение запускается на ПК |
| 3. Инженер вводит логин и пароль и входит в систему | 4. Система анализирует введённые данные, подключается к БД, после успешной проверки предоставляет программную форму для дальнейшей работы преподавателя |
| 5. Инженер загружает образы ОС | 6. Система подключается к БД, загружает образы в систему |
| 7. Инженер делает шаблоны для ВМ | 8. Система конвертирует список в рабочие данные и сохраняет изменения |

1.5. Требования к видам обеспечения

1.5.1. Требования к информационному обеспечению

Уровень хранения данных в системе должен быть построен на основе современных реляционных или объектно-реляционных СУБД.

Средства СУБД, а также средства используемых операционных систем должны обеспечивать документирование и протоколирование обрабатываемой в системе информации.

Структура базы данных должна поддерживать кодирование хранимой и обрабатываемой информации в соответствии с общероссийскими классификаторами.

Структура базы данных должна быть организована рациональным способом, исключающим единовременную полную выгрузку информации, содержащейся в базе данных системы. Технические средства, обеспечивающие хранение информации, должны использовать современные технологии, позволяющие обеспечить повышенную надежность хранения данных и оперативную замену оборудования.

В состав системы должна входить специализированная подсистема резервного копирования и восстановления данных.

Внедряемая БД должна интегрироваться со всеми модулями информационной системы, имеющимися в организации.

1.5.2. Требования к алгоритмическому обеспечению

Особые алгоритмические способы и функции отсутствуют.

1.5.3. Требования к программному обеспечению

На этапе проектирования и разработки системы необходимо максимально эффективным образом использовать ранее закупленное программное обеспечение, как серверное, так и для рабочих станций.

Используемое при внедрении программного обеспечения и библиотек программных кодов должны иметь широкое распространение, быть общедоступными. Базовой программной платформой может являться любая операционная система [2].

Программное обеспечение делится на клиентскую и серверную часть.

К требованиям клиентского ПО относятся: ОС Linux с версией ядра не меньше чем 5.0.

К требованиям серверного ПО относится: сервер с СУБД SQLite, OC Астра Linux 1.6 «Смоленск», библиотека libvirt, программный пакет Opennebula 6.2.

Приложение к Б1. Анализ аналогичных разработок

Проанализировав аналогичные разработки, был выявлены аналоги облачной платформы Opennebula. Одним из них является Proxmox. Proxmox предназначен для организации виртуальных машин в облаке, как и Opennebula, но у Proxmox есть ограничения. Главное из них – количество развертываемых виртуальных машин. Proxmox может оперировать до 100 виртуальных машин, что мало для предприятия связанный с военно-морским флотом Российской Федерации. Opennebula в свою очередь может контролировать более 1000 виртуальных машин в одно время, при условии мощных серверов, на основе которых будет находиться облако. Этого вполне достаточно для решения задач. Следующее отличие в типе виртуальных машин. Proxmox поддерживает только KVM и LXC. Opennebula может поддерживать KVM, LXC, LXD, VMWare и firecracker, что может покрыть любые требования.  
 Второй аналогичной разработкой является OpenStack. OpenStack является облачной платформой для организации виртуальных машин и хранилищ. Главное отличие OpenStack и Opennebula состоит в том, что Opennebula сертифицирована в Российской Федерации. Это значит, что Opennebula можно использовать в Министерстве обороны Российской Федерации.

Сравнив данные разработки, можно выявить следующие достоинства:

- возможность оперирования различными технологиями виртуализирования

- большое количество поддерживаемых виртуальных машин в одно время

- сертификация в Российской Федерации

- работа со сторонними сервисами

Таким образом, облачная платформа Opennebula имеет множество плюсов с аналогичными разработками. Оно является универсальным средством для разработки и виртуализации.