  
МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ  
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ  
ВЫСШЕГО ОБРАЗОВАНИЯ  
**«Дальневосточный федеральный университет»**

**ИНСТИТУТ МАТЕМАТИКИ И КОМПЬЮТЕРНЫХ ТЕХНОЛОГИЙ  
Департамент программной инженерии и искусственного интеллекта**

**РАЗРАБОТКА СИСТЕМЫ УПРАВЛЕНИЯ НАВЫКАМИ ГОЛОСОВОГО ПОМОЩНИКА**

КУРСОВОЙ ПРОЕКТ  
по дисциплине «Технологии коллективной промышленной разработки информационных систем» по образовательной программе подготовки бакалавров по направлению 09.03.04 «Программная инженерия»

Выполнили:  
студенты гр. Б9120-09.03.04прогин  
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Бронников А.А.  
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Воронова Д.В.  
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Лысенко А.А.   
Руководитель:  
старший преподаватель департамента ПИиИИ  
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Иваненко Ю.С.

г. Владивосток  
2024

**Оглавление**

[**Введение** 3](#_Toc161867053)

[**1 План проекта** 4](#_Toc161867054)

[**2 Регламент проведения инспекции** 5](#_Toc161867055)

[**2.1** **Критерии отнесения к формальной / неформальной инспекции для различных типов рабочих продуктов** 5](#_Toc161867056)

[**2.2** **Перечень ролей участников инспекции и их обязанности, число участников в зависимости от объёма рабочего продукта** 6](#_Toc161867057)

[**2.3** **Этапы инспекции** 6](#_Toc161867058)

[**2.4** **Порядок организации** 6](#_Toc161867059)

[**2.5** **Порядок подготовки к инспекции** 8](#_Toc161867060)

[**2.6** **Порядок проведения инспекции** 9](#_Toc161867061)

[**2.7** **Перечень статусов и степени важности замечаний** 11](#_Toc161867062)

[**2.8** **Порядок верификации учёта замечаний** 12](#_Toc161867063)

[**3 Модель состояний задач** 13](#_Toc161867064)

[**4 Презентация проекта** 15](#_Toc161867065)

[**5 Требования к проекту** 17](#_Toc161867066)

[**6 Разработка архитектуры проекта** 27](#_Toc161867067)

[**9 Требования к коду** 34](#_Toc161867068)

**Введение**

Промышленная разработка информационных систем включает в себя множество этапов, начиная от разработки плана проекта, заканчивая его тестированием, для чего необходимо множество специалистов различных профилей, а также унифицированные методы коммуникации между ними, с помощью которых можно разделить обязанности членов команды по их специализации.

Исходя из описанного выше, необходимо использовать определенные технологии коллективной разработки для повышения эффективности работы в группе и соответствия конечного продукта заявленным требованиям.

В данной курсовой работе рассматривается задача коллективной разработки программного средства «Система управления навыками голосового помощника» и составление технической документации к нему.

Таким образом, целью курсовой работы является разработка программного средства с использованием подходов коллективной промышленной разработки.

Для достижения поставленной цели необходимо решить следующие задачи:

1. Разработать план проекта.
2. Разработать регламент проведения инспекции.
3. Разработать модель состояний задач.
4. Разработать презентацию проекта.
5. Разработать требования к проекту.
6. Разработать архитектуру проекта.
7. Разработать измерения проекта.
8. Разработать перечень задач проекта.
9. Разработать рекомендации по кодированию.
10. Разработать план тестирования проекта.
11. Протестировать проект.

**1 План проекта**

План проекта – это документ, содержащий подробную информацию о проекте: исполнителях, задачах и сроках. Документ является конечным результатом этапа планирования, утверждается до начала любых работ и становится самым главным и достоверным источником информации о грядущем проекте.

В нашем случае исполнителями являются следующие лица:

* Team Leader – Лысенко Алексей;
* Coder 1 – Воронова Диана;
* Coder 2 – Бронников Алексей;
* Build Engineer – Бронников Алексей;
* Technical Writer – Воронова Диана.

На рисунке 1 представлен перечень задач для выполнения и примерные сроки их реализации.

Изображение выглядит как снимок экрана, текст, линия, График

Автоматически созданное описание

Рисунок 1 – План проекта

**2 Регламент проведения инспекции**

Верификация рабочих продуктов является неотъемлемой частью процесса по обеспечению их качества. Современной технологией программирования выработаны специальные стандарты, подходы и механизмы проведения верификаций рабочих продуктов, в формате так называемых инспекций.

Инспекция – это мероприятие по обеспечению качества рабочих продуктов проектов по разработке ПО и иной деятельности, которая проводится разработчиками, возможно – с участием представителей заказчика.

Концептуально инспекция имеет следующие цели:

* обнаружить ошибки в функциях, логике, содержании или реализации рабочих продуктов на ранних этапах их разработки и предотвратить их наследование;
* рационально донести замысел или реализацию продукта до всех заинтересованных лиц (через их участие);
* оптимизировать, оценить или улучшить рабочий продукт.

1. **Критерии отнесения к формальной / неформальной инспекции для различных типов рабочих продуктов**

Критерии отнесения к формальной / неформальной инспекции для различных типов рабочих продуктов представлены в таблице 1.

Таблица 1 – Критерии относения к формальной / неформальной инспеции

|  |  |
| --- | --- |
| **Рабочий продукт** | **Размер продукта или изменения**  **(не менее)** |
| Требования | 5 страниц  или  10% рабочего продукта |
| Документы дизайна | 5 страниц  или  10% рабочего продукта |
| Завершение таблицы 1 | |
| **Рабочий продукт** | **Размер продукта или изменения**  **(не менее)** |
| Код | 20 NCLOC (модуль проекта)  или  60 NCLOC (весь проект) |
| Тесты | 5 страниц  или  10% рабочего продукта |

1. **Перечень ролей участников инспекции и их обязанности, число участников в зависимости от объёма рабочего продукта**

Автор (Author) – Лысенко Алексей.

Председатель (Moderator) – Воронова Диана.

Секретарь (Recorder) – Воронова Диана.

Ведущий (Presenter) – Бронников Алексей.

Инспектор (Inspector) – Бронников Алексей.

\* Роли могут меняться в зависимости от стадии проекта, возможностей участников.

1. **Этапы инспекции**

**Этапы формальной инспекции:**

1. Планирование инспекции.
2. Назначение инспекции.
3. Обзорное собрание (опционально).
4. Подготовка к инспекции.
5. Собрания по инспекции.
6. Завершение инспекции.
7. **Порядок организации**

**Функции каждого участника:**

Функции автора:

1. Инициировать формальную инспекцию рабочего продукта. Для этого автор должен оповестить руководителя проекта о готовности рабочего продукта к формальной инспекции и зафиксировать промежуточную стабильную версию рабочего продукта. С момента предоставления рабочего продукта на формальную инспекцию до собрания включительно автор не имеет права вносить изменения.
2. Запросить руководителя проекта об имеющихся ресурсах на роль председателя инспекции.
3. Провести обзорное собрание, если это необходимо. Данное собрание имеет целью ввести участников формальной инспекции в курс дела. Изложить суть и, возможно, какие-то особенности рабочего продукта, вызывающие непонимание или которые необходимо проверить более тщательно.

Функции председателя:

1. Убедиться, что рабочий продукт удовлетворяет критерию готовности к формальной инспекции. Данный критерий формулируется обычно в виде соответствующего типу рабочего продукта чек-листа.
2. Определить необходимость проведения обзорного собрания.
3. Проверить, что все обязательные участники формальной инспекции приняли приглашение либо прислали отказ с обязательным указанием причин и/или предложением своих сроков проведения инспекции. В случае получения хотя бы одного отказа, председатель должен проанализировать сложившуюся ситуацию на предмет того, может ли быть проведена инспекция без отказавшегося участника. Если это невозможно (количество участников ограничено и участие каждого строго обязательно или специалист, приславший отказ, является заинтересованным в данном рабочем продукте лицом), то формальная инспекция не проводится.

Автор / Председатель должны обеспечить:

1. Определить материально – техническую сторону проведения инспекции: место, время, дату и продолжительность собрания.

Место, дату и время собрания необходимо согласовать с участниками.

Продолжительность собрания формальной инспекции не должна превышать 2 часов. Если рабочий продукт настолько большой, что продолжительность собрания получается больше двух часов, то проведение формальной инспекции необходимо запланировать на несколько собраний.

1. Подобрать команду участников и распределить роли. При подборе команды необходимо руководствоваться следующими правилами:

Команда участников формальной инспекции должна включать 3 человека (включая автора).

При назначении инспекций кода рекомендуется приглашать в качестве инспекторов разработчиков инспектируемых модулей и разработчиков вызываемых модулей.

1. Оповестить всех участников формальной инспекции. В приглашении необходимо правильно указать размер инспектируемого рабочего продукта, а также указать рекомендуемое время подготовки, исходя из размера инспектируемого рабочего продукта.

Функции секретаря, ведущего и инспектора:

1. Принять или отклонить с обязательным указанием причин приглашение на формальную инспекцию. Рекомендуется, чтобы каждый участник формальной инспекции ограничивал себя двумя инспекциями в день.
2. **Порядок подготовки к инспекции**

**Сроки и время**

Данный этап необходимо проводить за сутки до проведения инспекции. Ответы на приглашения на инспекцию должны быть получены до 17:00 в течение суток.

**Функции каждого участника**

Функции автора:

1. До собрания ознакомиться с содержанием полученных от инспекторов протоколов подготовки к формальной инспекции, проанализировать изложенные в них замечания.

Функции председателя:

1. На основе полученных от инспекторов протоколов подготовки к формальной инспекции принять одно из следующих решений:

* Провести инспекцию.
* Перенести инспекцию.
* Отменить инспекцию.

1. Во всех случаях сразу после принятия решения председатель должен оповестить всех участников формальной инспекции об изменениях.

Функции секретаря:

1. Проанализировать замечания, зафиксированные инспекторами в протоколах подготовки к формальной инспекции, выявить повторяющиеся. На данном этапе секретарь должен собрать полный список проблем рабочего продукта.

Функции инспектора:

1. Самостоятельно независимо от других участников инспекции изучить предоставленный для инспекции рабочий продукт, используя накопленный опыт, стандарты, руководства, контрольные списки.
2. Заполнить необходимые поля протокола подготовки к формальной инспекции и отправить его председателю формальной инспекции, секретарю и автору. В протоколе подготовки к инспекции указывается, как правило, время подготовки к инспекции и описание найденных ошибок.
3. **Порядок проведения инспекции**

**Сроки и время**

На проведение инспекции отводится от 10 до 20 минут.

**Функции каждого участника**

Функции автора:

1. По мере необходимости давать четкие и ясные ответы на вопросы инспекторов без попыток оценить корректность рабочего продукта. Во время собрания автор не имеет права принимать решения относительно замечаний и изменять рабочий продукт. На собрании коллегиально может быть принято решение о необходимости проведения повторной формальной инспекции рабочего продукта.

Функции председателя:

* 1. Огласить цель данного собрания, представить участников инспекции (если это необходимо) и уточнить их роли.
  2. Убедиться, что у каждого инспектора есть его замечания и попросить ведущего начать представление инспектируемого рабочего продукта.
  3. Следить за ходом и темпом собрания: пресекать попытки решения выявленной в рабочем продукте проблемы и просить ведущего продолжать представление рабочего продукта; разрешать конфликты; пресекать попытки ухода от темы собрания; следить, чтобы внимание было сосредоточено на рабочем продукте и выявлении его недостатков.
  4. По ходу собрания принимать решение, действительно ли конкретное замечание является проблемой и каков его статус в случае, если инспекторы не могут прийти к единому мнению.
  5. Прервать собрание, если его длительность превысила запланированное время, а часть замечаний еще не охвачена.
  6. Принять решение о переносе собрания в случае нарушения участниками инспекции установленного приглашением порядка проведения инспекции или возникновения непредвиденных обстоятельств.
  7. Принять окончательное решение о назначении проверяющего (Verifier). Любой участник формальной инспекции, за исключением автора, может быть назначен на роль проверяющего. Проверяющий контролирует изменения в рабочем продукте, сделанные согласно замечаниям.

Функции секретаря:

* 1. Огласить для всех участников окончательную формулировку каждого замечания, его статус и местоположение. Это необходимо делать, потому что в ходе собрания замечание (а также его статус и местоположение) может быть переформулировано, уточнено, исправлено. Секретарь должен зафиксировать все эти изменения и огласить окончательную формулировку, а участники инспекции – проконтролировать.
  2. Вести протокол собрания. Обнаруженные в ходе собрания недостатки рабочего продукта необходимо сформулировать в виде замечаний, указать их местоположение в рабочем продукте, определить статус и занести в протокол инспекции.

Функции ведущего:

* + 1. Представить рабочий продукт для участников собрания, выбрав при этом наиболее эффективный способ. От выбора способа представления конкретного рабочего продукта во многом зависит эффективность проведения формальной инспекции.

Функции инспектора:

1. Задавать вопросы или оглашать найденные в рабочем продукте проблемы.
2. По ходу инспекции высказать свое мнение о статусе замечания. Значения статуса каждого замечания должны быть уже зафиксированы в протоколе подготовки к формальной инспекции каждого инспектора, но они могут быть откорректированы во время собрания.
3. В конце инспекции высказать свое мнение по поводу необходимости проведения повторной формальной инспекции рабочего продукта.
4. **Перечень статусов и степени важности замечаний**

**Допустимые значения статуса замечания:**

* Ошибка (Error).
* Предупреждение (Warning).
* Комментарий (Comment).

Допустимые значения степени серьёзности замечания:

* Критическая (Critical).
* Особо важная (Major).
* Средняя (Moderate).
* Мелкая, незначительная (Minor).
* Другие (Other).

1. **Порядок верификации учёта замечаний**

**Функции каждого участника**

Функции автора:

1. Проанализировать все замечания, зафиксированные в протоколе собрания, и исправить все недостатки рабочего продукта.
2. Решение по каждому замечанию зафиксировать в протоколе инспекции.

Функции председателя:

1. Проверить протокол инспекции, а именно: убедиться в правильности и полноте заполнения протокола, включая как формулировки так и статусы замечаний; убедиться, что протокол разослан/оповещены все участники формальной инспекции и заинтересованные лица.

Функции секретаря:

1. Разослать всем участникам протокол формальной инспекции. Секретарь должен оповестить всех участников формальной инспекции и заинтересованных лиц о готовности протокола, приложив его к письму или указав место его нахождения. В случае если во время подготовки к формальной инспекции и собрания не было выявлено замечаний к рабочему продукту, то в протокол необходимо поместить соответствующую информацию.

Функции лица, назначенного на роль проверяющего:

1. Проверить следующее: все замечания учтены в обновленном рабочем продукте корректно. Именно после вердикта проверяющего формальная инспекция считается завершённой.

**3 Модель состояний задач**

Каждая задача, являясь отражением делового процесса, проходит определенные состояния. Сначала идет создание задачи, потом идет выполнение работ по задаче, после выполнения задача завершается.

**Перечень состояний задач:**

1. New – новая подзадача.
2. Analysis – в процессе анализа. В это состояние подзадачу переводит сотрудник после того, как начнёт её анализ.
3. Forward – в данном случае имеет значение «переданный на дальнейшую разработку». В это состояние задача переводится CCB после анализа при назначении задачи на разработку конкретному сотруднику.
4. Coding – кодирование. В это состояние задача переводится сотрудникомразработчиком, при начале работы по кодированию, связанному с задачей.
5. Inspected – проинспектировано. В это состояние задача переводится сотрудником-разработчиком после завершения кодирования и инспектирования изменений рабочего продукта.
6. Integrated – заинтегрировано. Переводится сотрудником, осуществляющим интеграцию изменений в основную ветку рабочего продукта после успешной интеграции этих изменений.
7. Tested – протестировано. Переводится сотрудником, осуществляющим тестирование изменений в рабочий продукт (tester).
8. Closed – закрыто. В это состояние задача переводится CCB по результатам отчёта о тестировании сделанных изменений.

**Правила создания новой задачи**

Созданием новых задач могут заниматься все участники команды. Происходит это на начальном этапе, когда необходимо организовать структурированную работу над проектом и в процессе работы над проектом в случае присутствия в плане слишком объемных задач, требующих упрощения. Также, когда необходимо организовать баг-фиксинг или разработать новый функционал.

**Правила перехода задачи из состояния в состояние**

Состояния задач всегда идут последовательно друг за другом, в некоторых случаях пункты могут опускаться или повторяться. Схема перехода из состояния в состояние показана на рисунке 2.

Изображение выглядит как диаграмма, текст, снимок экрана, линия

Автоматически созданное описание

Рисунок 2 – Состояния задачи

**4 Презентация проекта**

На рисунке 3 представлена титульная страница презентации.

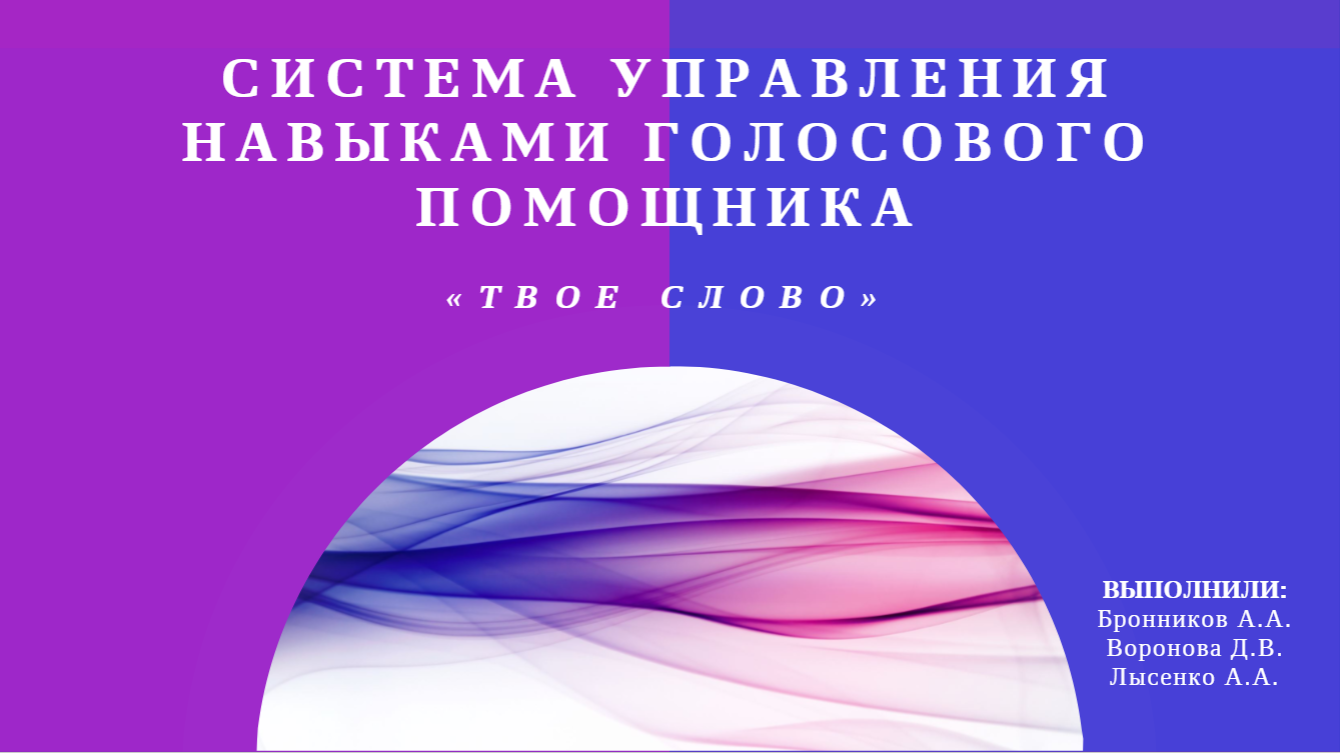


Рисунок 3 – Титульная страница презентации

На рисунке 4 представлены данные о проблеме, существующей в предметной области.

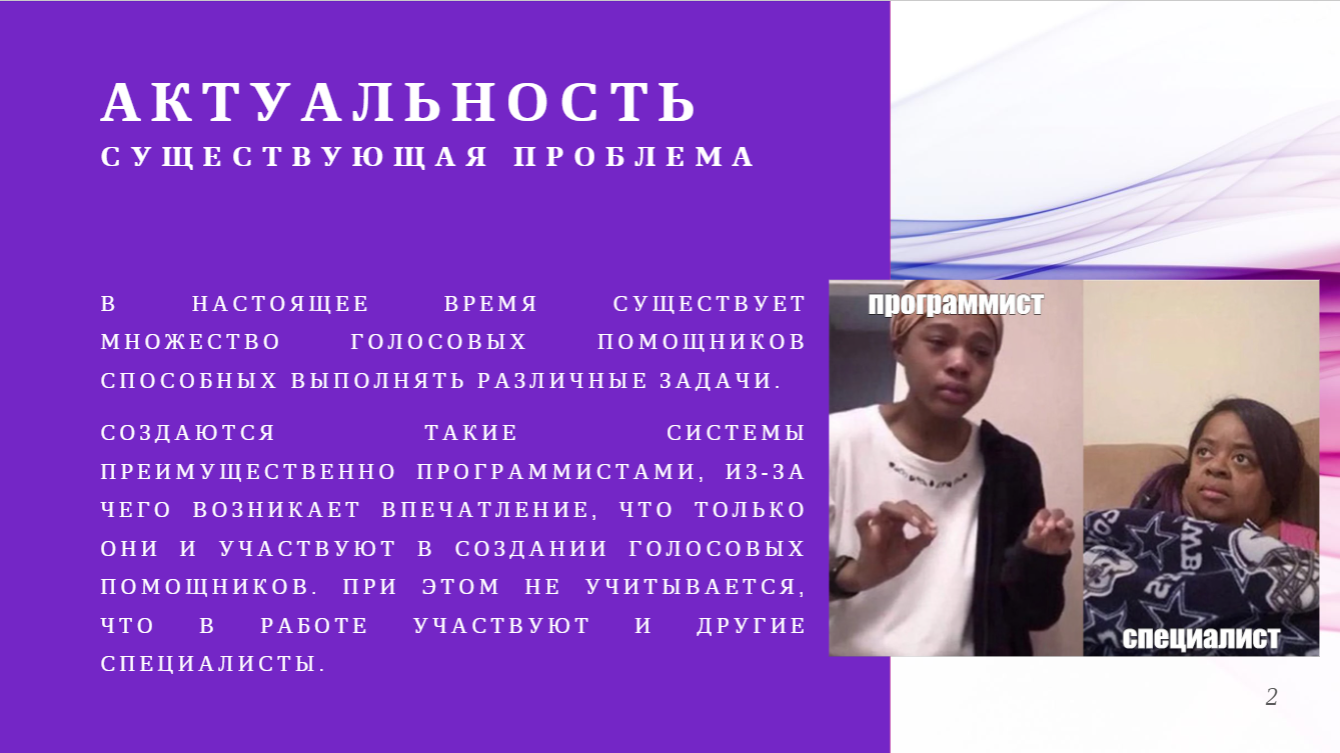
****

Рисунок 4 – Слайд об актуальности в презентации

На рисунке 5 представлена информация о целях проекта.

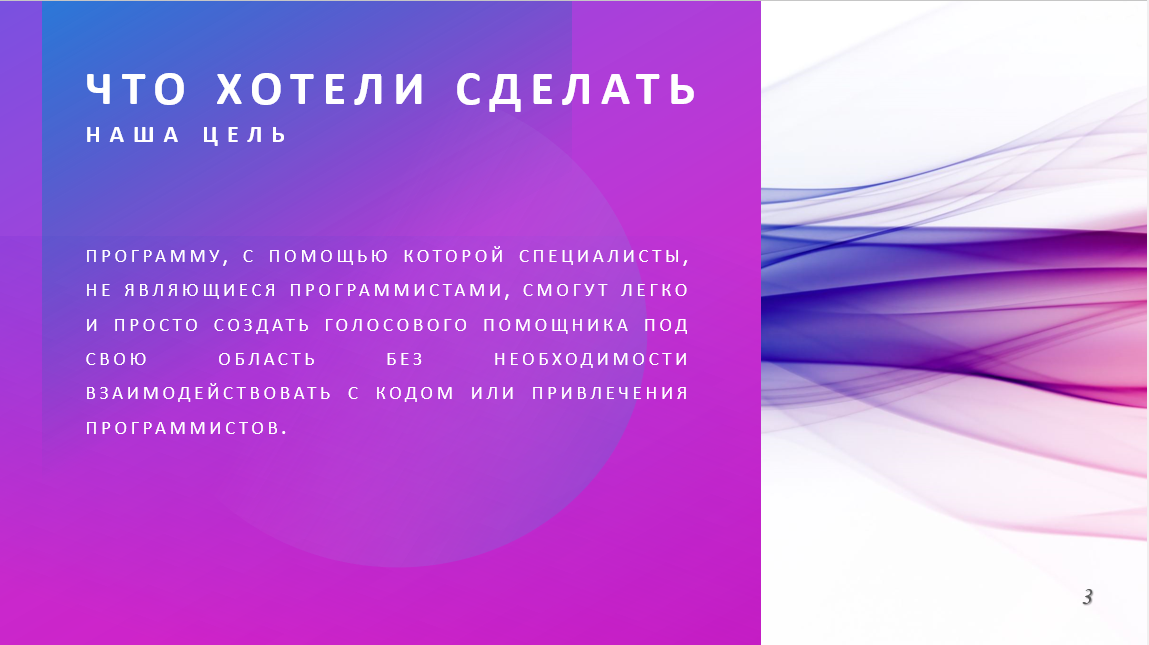
****

Рисунок 5 – Слайд с данными о целях проекта

На рисунках 6-12 представлены данные о результатах работы над проектом, а также принципе работы программного продукта.

**5 Требования к проекту**

**Требования к системе (CS):**

* REQ-CS-1 Система должна позволять пользователю задавать вопрос голосом или письменно в поле чата.
* REQ-CS-2 Система должна генерировать пользователю ответ в виде синтезированной речи и текста.
* REQ-CS-3 Система должна генерировать ответ на основе данных, занесенных в систему.

**Создаваемая система состоит из следующих модулей:**

* Модуль диалога (DM).
* Модуль тестирования (TM).
* Модуль семантической сети (SNM).

**Требования к модулю диалога**

Требования диалога подразделяются на:

* DTr – Требования, касающиеся дерева диалога.
* DI – Требования, касающиеся ввода.
* DO – Требования, касающиеся вывода.
* Dtemp – Требования, касающиеся шаблонов сцен.

Функциональные требования к модулю диалога:

* REQ-DI-1 Модуль диалога должен считывать речь пользователя и выделять из неё интенты и их значения.
* REQ-DO-1 Модуль диалога должен синтезировать речь при ответе пользователю.
* REQ-DI-2 Модуль диалога должен отправлять модулю **графа знаний** вычисленные из речи пользователя **интенты** и их значения.
* REQ-DTr-1 Модуль диалога должен быть представлен **деревом** **сцен** с возможностью добавления, удаления и изменения узлов.
* REQ-DTr-2 Модуль диалога должен хранить **контекст** для возможности перехода по сценам.
* REQ-DTemp-1 В сцене должны быть прописаны **шаблоны ответов**, заполняемые полученными из графа знаний **интентами** и их значениями при выводе пользователю.
* REQ-DTemp-2 В сцене должны быть прописаны **шаблоны уточняющих вопросов**, вызываемых при недостатке данных, введённых пользователем.
* REQ-DTr-3 **Сцена** должна иметь ссылки для перехода к следующей **сцене** диалога при выполнении требований (найденные в речи пользователя интенты).
* REQ-DI-3 В системе должно быть поле чата.

Пользовательские требования к модулю диалога:

* REQ-DI-4 У пользователя должна быть возможность в любой момент задать вопрос голосовому помощнику.
* REQ-DI-5 Пользователь должен иметь возможность задать вопрос в письменной форме в чате.
* REQ-DO-2 Пользователь должен иметь возможность получить ответ на все вопросы, заданные в устной форме после ключевого слова или включения микрофона в окне чата.
* REQ-DI-6 У пользователя должна быть возможность задать вопрос в свободной форме.
* REQ-DI-7 У пользователя должна быть возможность дополнить заданный ранее вопрос.
* REQ-DO-3 Пользователь должен получить уведомление от модуля диалога в случае, если он не способен вернуть ответ на поставленный им вопрос.

**Требования к модулю тестирования**

Требования к модулю тестирования подразделяются на:

* Требования, касающиеся тестирования модуля диалога (DMT).
* Требования, касающиеся тестирования семантической сети (SNT).
* Требования, касающиеся логирования (L).

Функциональные требования к модулю тестирования:

* REQ-L-1 Модуль тестирования должен заполнять три файла **логирования**: с информацией о корректных сеансах работы с системой; о сеансах работы, во время которых система выдавала неправильные ответы; о сеансах работы, во время которых система не смогла дать ответ.
* REQ-L-2 В файлах **логирования** должны быть представлены не только реплики, но и места в диалоговом **дереве**, где они были получены.
* REQ-L-3 В файле с информацией о корректной работе цепочки диалогов между пользователем и системой должны быть представлены полностью.
* REQ-SNT-1 Модуль должен предлагать варианты возможных вопросов, не добавленных в систему управления **навыками**.
* REQ-DMT-1 Модуль тестирования должен позволять проводить автоматическое функциональное тестирование системы управления **навыками**.

Пользовательские требования к модулю тестирования:

* REQ-L-4 Пользователь должен иметь возможность получить доступ к файлам **логирования**, чтобы понимать причину, если система управления **навыками** работает ошибочно.
* REQ-DMT-2 Пользователь должен иметь возможность провести автоматическое функциональное тестирование системы управления **навыками** голосового помощника, чтобы убедиться в корректности ее работы.
* REQ-DMT-3 При проведении функционального тестирования пользователь должен видеть всю цепочку диалога, вместе с указанием того, где был получен тот или иной ответ, чтобы понять причину ошибки при необходимости.
* REQ-SNT-2 Пользователь должен видеть варианты вопросов, не добавленные в систему управления **навыками**, для того, чтобы добавить их при необходимости.

**Требования к модулю графа знаний**

Функциональные требования к модулю графа знаний подразделяются на:

* Требования семантической сети (SN).
* Требования данных (D).
* Требования взаимодействия систем (SI).
* Требования взаимодействия пользователя (UI).

Функциональные требования к модулю графа знаний:

* REQ-SN-1 Граф знаний должен обновлять информацию о предметной области.
* REQ-SN-2 Граф знаний должен хранить информацию о предметной области в сокращенной форме в виде семантической сети.
* REQ-D-1 Граф знаний должен хранить целостную информацию.
* REQ-D-2 Граф знаний должен содержать минимальное количество дублированной информации.
* REQ-SI-1 Граф знаний должен принимать на вход интенты и их значения в виде заранее определенных структур (кортежей, массивов).
* REQ-SI-2 Граф знаний должен возвращать интенты и их значения в виде заранее определенных структур (кортежей, массивов).
* REQ-SI-3 Граф знаний должен выдавать ответ на запросы других систем, даже если информация отсутствует (возвращать None).
* REQ-SI-4 Граф знаний должен выдавать все варианты ответа, если таковых вариантов несколько.

Пользовательские требования к модулю графа знаний:

* REQ-UI-1 Пользователь должен взаимодействовать с **графом знаний** через другие модули, чтобы легко и эффективно работать.

**Требования к интерфейсу (I)**

Функциональные требования к интерфейсу подразделяются на:

* Требования к интерфейсу чата голосового помощника (С).
* Требования к интерфейсу системы управления навыками голосового помощника (M).

Функциональные требования к интерфейсу чата:

* REG-I-C-1 При нажатии на кнопку микрофона система должна записывать голос пользователя и отправлять его реплику в модуль диалога для дальнейшей обработки.
* REG-I-C-2 При нажатии на кнопку отправки система должна отправлять текст, введенный пользователем, в модуль диалога для дальнейшей обработки.

Пример пользовательского интерфейса представлен на рисунке 13.

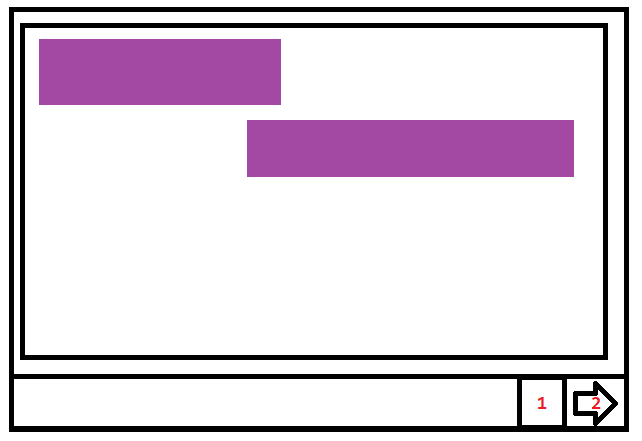


Рисунок 13 – Вариант пользовательского интерфейса

Функциональные требования к интерфейсу системы управления навыками голосового помощника:

* REG-I-M-1При нажатии на соответствующую кнопку меню должен открываться раздел с результатами работы модуля тестирования.
* REQ-I-M-2 При нажатии на соответствующую кнопку меню должен открываться раздел для работы с модулем диалога.
* REQ-I-M-3 При нажатии на соответствующую кнопку меню должен открываться раздел для работы с графом знаний.

Пример интерфейса для управления навыками голосового помощника представлен на рисунке 14.

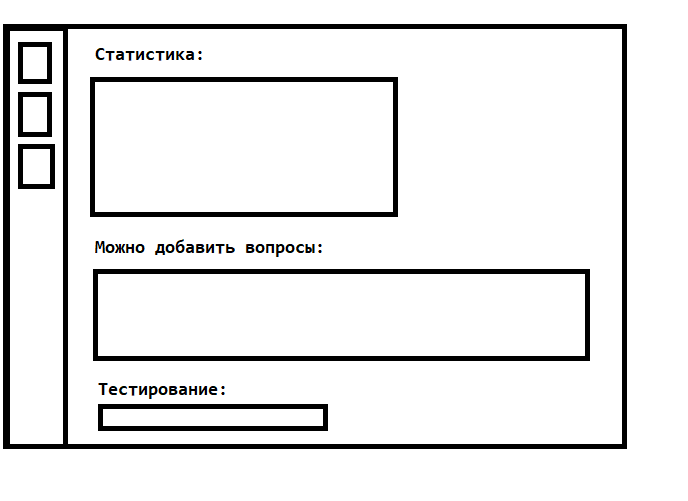


Рисунок 14 – Вариант интерфейса для создателей голосового помощника

**Матрица покрытия требований**

Матрица покрытия требований модуля семантической сети представлена в таблице 2.

Таблица 2 – Матрица покрытия требований модуля семантической сети.

|  |  |  |
| --- | --- | --- |
| **Идентификатор тестовой ситуации** | **Идентификатор покрытого требования** | **Покрытие** |
| SN | | |
| TEST\_SN\_1\_1  TEST\_SN\_1\_2  TEST\_SN\_1\_3 | REQ-SN-1 | Не покрыт |
| Завершение таблицы 2 | | |
| **Идентификатор тестовой ситуации** | **Идентификатор покрытого требования** | **Покрытие** |
| TEST\_SN\_2 | REQ-SN-2 | Не покрыт |
| D | | |
| TEST\_D\_1 | REQ-D-1 | Покрыт |
| TEST\_D\_2 | REQ-D-2 | Не покрыт |
| SI | | |
| TEST\_SI\_1 | REQ-SI-1 | Покрыт |
| REQ-SI-2 |
| REQ-SI-3 |
| REQ-SI-4 |
| UI | | |
| TEST\_UI\_1\_1  TEST\_UI\_1\_2 | REQ-UI-1 | Не покрыт |

TEST\_SN\_1\_1 – Добавление нового непустого документа содержащего минимум одну строку текста.

TEST\_SN\_1\_2 – Удаление устаревшего документа содержащего минимум одну строку текста.

TEST\_SN\_1\_3 – Обновление существующего документа с измененной строкой текста.

TEST\_SN\_2 – Сокращение информации при получение множества однотипных строк.

TEST\_D\_1 – Хранение и представление информации в графовой форме.

TEST\_D\_2 – Сокращение информации при получение множества дублированной информации.

TEST\_SI\_1 – Граф знаний принимает запрос и возвращать на него ответ в виде заранее определенного картежа.

TEST\_UI\_1\_1 – Получение запроса (запрос информации) от внешней системы и выдача ответа на запрос.

TEST\_UI\_1\_2 – Получение запроса (запрос узлов / связей) от внешней системы и выдача ответа на запрос.

Матрица покрытия требований модуля диалога представлена в   
таблице 3

Таблица 3 – Матрица покрытия требований модуля диалога.

|  |  |  |
| --- | --- | --- |
| **Идентификатор тестовой ситуации** | **Идентификатор покрытого требования** | **Покрытие** |
| DTr | | |
| TEST\_DTr1 | REQ-DTr-1, REQ-DTr-3 | Покрыт |
| TEST\_DTr2 | REQ-DTr-2 | Не покрыт |
| DI | | |
| TEST\_DI\_1 | REQ-DI-1, REQ-DI-2, REQ-DI-4, REQ-DI-6 | Покрыт |
| TEST\_DI\_2 | REQ-DI-2, REQ-DI-3, REQ-DI-4, REQ-DI-5, REQ-DI-6 | Покрыт |
| TEST\_DI\_3 | REQ-DI-4, REQ-DI-6, REQ-DI-7 | Не покрыт |
| DO | | |
| TEST\_DO\_1 | REQ-DO-1, REQ-DO-2 | Покрыт |
| TEST\_DO\_2 | REQ-DO-3 | Покрыт |
| DTemp | | |
| TEST\_DTemp\_1 | REQ-DTemp-1, REQ-DTemp-2 | Не покрыт |

TEST\_DTr1 – Составление дерева сцен со сценами main (отвечающая на шаблонные вопросы по проходному баллу направления), sub1 (потомок main, отвечающий на вопросы направлений, связанные с годом), sub2 (потомок main, отвечающий на вопросы направлений, связанные со сравнением баллов направлений).

TEST\_DTr2 – После вопроса «Какой проходной балл на направление Программная инженерия?» в контексте должны быть указаны интенты «балл» и «направление», а также значение «Программная инженерия».

TEST\_DI\_1 – Ввод голосом, сообщение «Какой проходной балл на направление Программная инженерия?».

TEST\_DI\_2 – Ввод текстом, сообщение «Какой проходной балл на направление Программная инженерия?».

TEST\_DI\_3 – Дополнение вопроса «Какой проходной балл на направление Программная инженерия?» уточнением «А в 2020 году?».

TEST\_DO\_1 – Получение ответа на вопрос «Какой проходной балл на направление Программная инженерия?».

TEST\_DO\_2 – На сообщение «В каком году появилось направление Программная инженерия?» получить ответ «Такой информации нет в системе или не хватает данных».

TEST\_DTemp\_1 – Создание сцены с шаблоном ответов «Какой (интент) на (интент) (значение)?» и уточняющим вопросом «Не хватает данных, про какое направление вы хотите узнать больше?».

**Матрица покрытия требований модуля тестирования представлена в таблице 4.**

Таблица 4 – Матрица покрытия требований модуля тестирования

|  |  |  |
| --- | --- | --- |
| **Идентификатор тестовой ситуации** | **Идентификатор покрытого требования** | **Покрытие** |
| SNT | | |
| TEST-SNT-1 | REQ-SNT-1, REQ-SNT-2 | Не покрыт |

|  |  |  |
| --- | --- | --- |
| Продолжение таблицы 4 | | |
| **Идентификатор тестовой ситуации** | **Идентификатор покрытого требования** | **Покрытие** |
| DMT | | |
| TEST-DMT-1 | REQ-DMT-1, REQ-DMT-2 | Не покрыт |
| TEST-DMT-2 | REQ-DMT-3 | Не покрыт |
| L | | |
| TEST-L-1 | REQ-L-1, REQ-L-2, REQ-L-3 | Не покрыт |
| TEST-L-2 | REQ-L-4 | Покрыт |

TEST-SNT-1 – Открыт раздел «Возможные вопросы» окна «Модуль тестирования».

TEST-DMT-1 – Нажата кнопка «Начать автоматическое тестирование» окна «Модуль тестирования».

TEST-DMT-2 – Открыт отчет по автоматическому тестированию.

TEST-L-1 – На вход модулю тестирования подаются по очереди успешные, неуспешные, неудачные цепочки диалогов.

TEST-L-2 – Нажата кнопка «Посмотреть логи» окна «Модуль тестирования».

**6 Разработка архитектуры проекта**

Архитектура программного обеспечения относится к фундаментальным структурам программной системы и дисциплине создания таких структур и систем. Каждая структура включает элементы программного обеспечения, отношения между ними, а также свойства как элементов, так и отношений.

Архитектура программной системы – это метафора, аналогичная архитектуре здания. Он функционирует как план для системы и проекта разработки, в котором излагаются задачи, которые должны быть выполнены командами разработчиков.

UML-диаграмма связей между объектами дерева диалога представлена на рисунке 15.

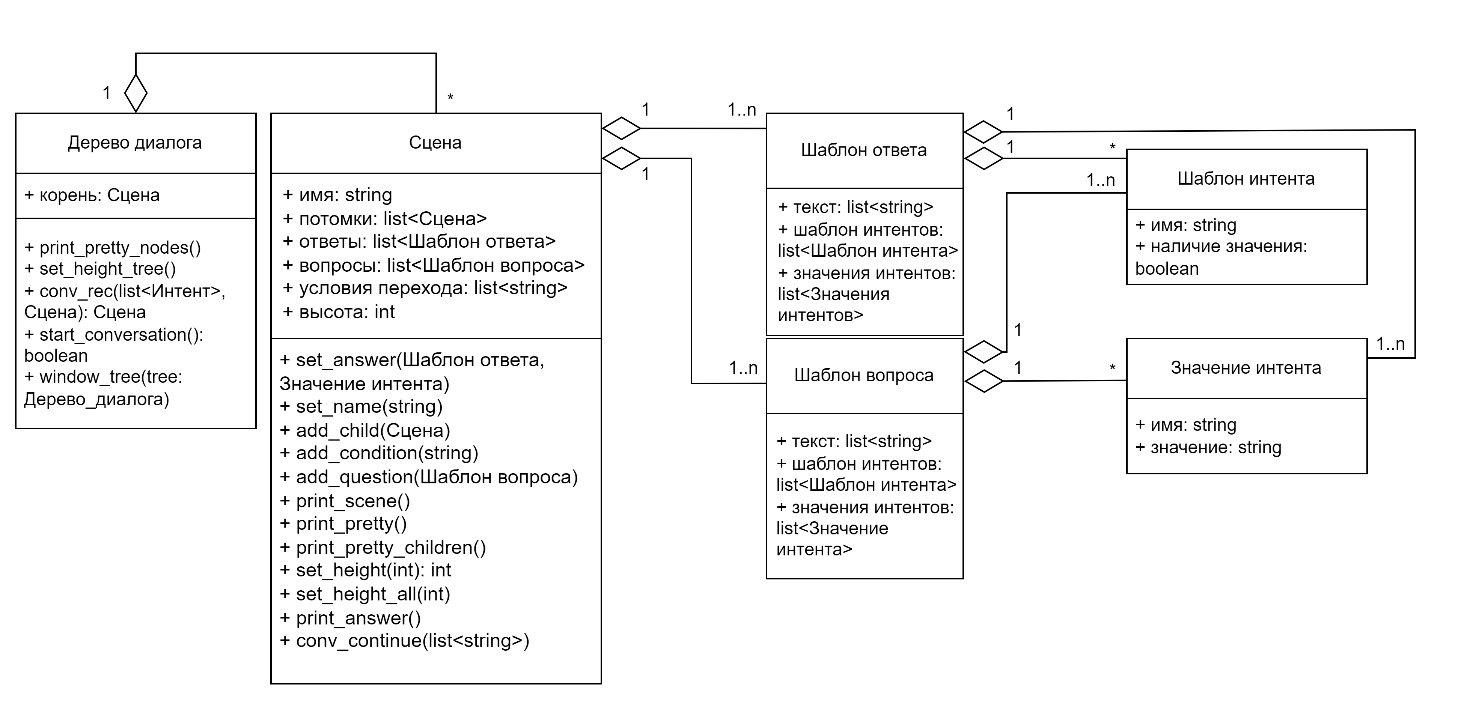


Рисунок 15 – Диаграмма связей между объектами дерева диалога

UML-диаграмма связей между объектами семантической сети представлена на рисунке 16.



Рисунок 16 – UML-диаграмма связей между объектами семантической сети

Use-Case диаграмма модуля диалога представлена на рисунке 17.

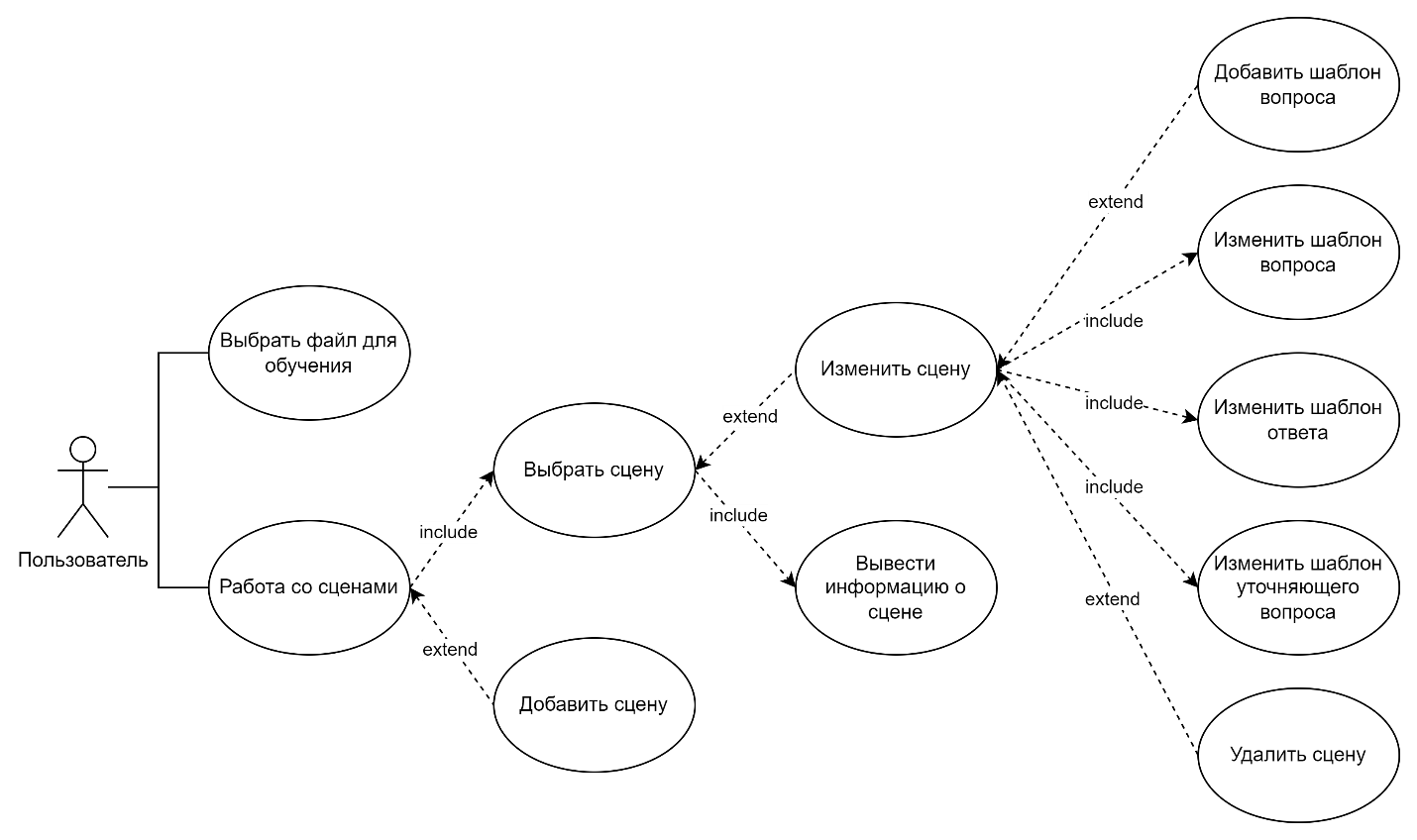


Рисунок 17 – Use-Case диаграмма модуля диалога.

Use-Case диаграмма модуля диалога представлена на рисунке 18.

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 18 – Use-Case диаграмма модуля семантической сети.

Use-Case диаграмма модуля тестирования представлена на рисунке 19.

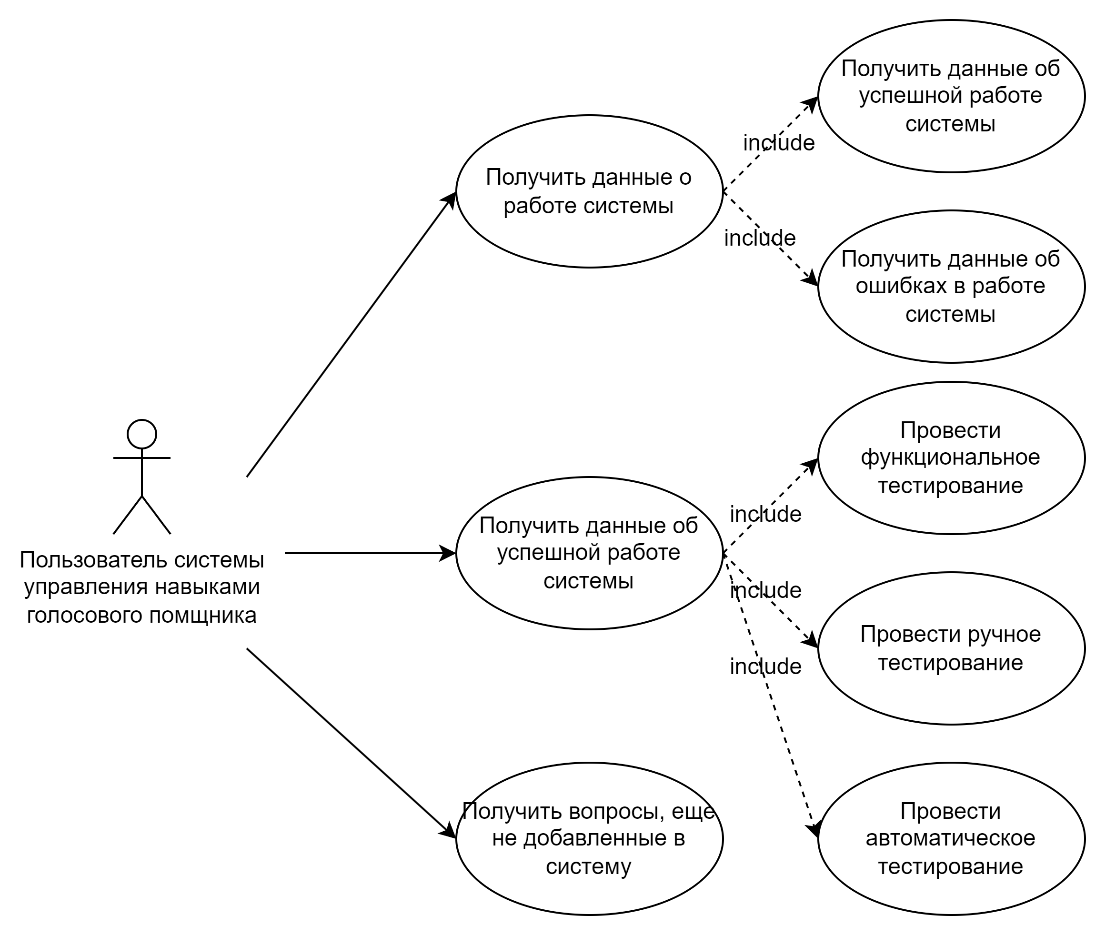


Рисунок 19 – Use-Case диаграмма модуля тестирования.

Архитектурно-контекстная диаграмма модуля тестирования представлена на рисунке 20.

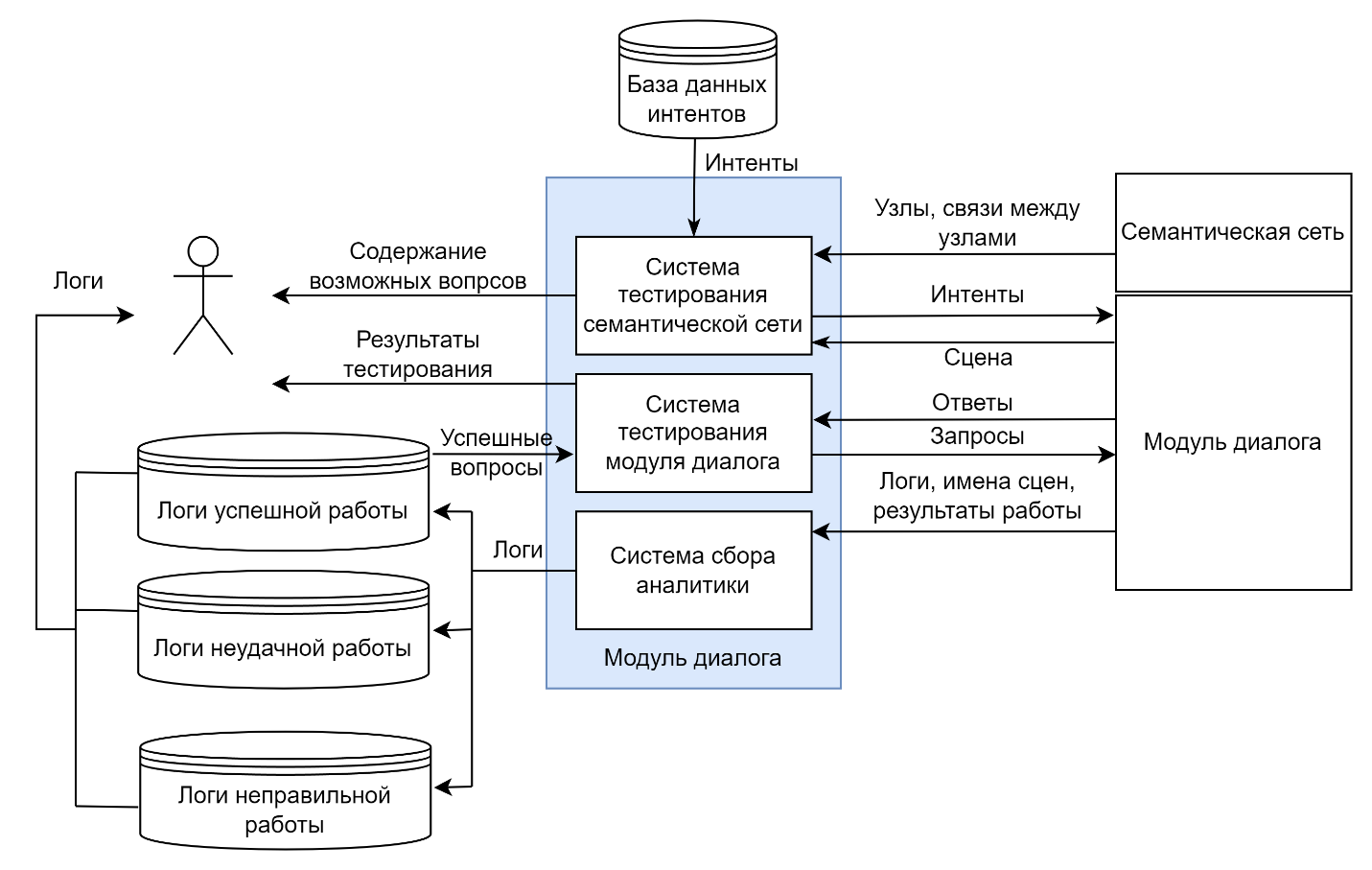


Рисунок 20 – АКД-диаграмма модуля тестирования.

Архитектурно-контекстная диаграмма модуля диалога представлена на рисунке 21.

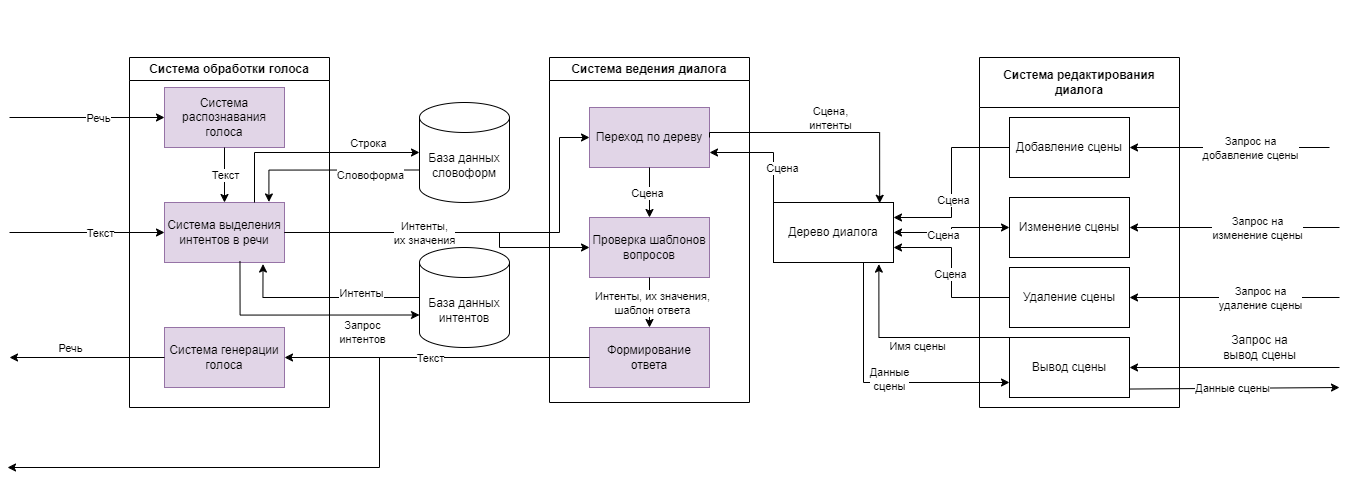


Рисунок 21 – Архитектурно-контекстная диаграмма модуля диалога.

Общая архитектурно-контекстная диаграмма представлена на рисунке 22.

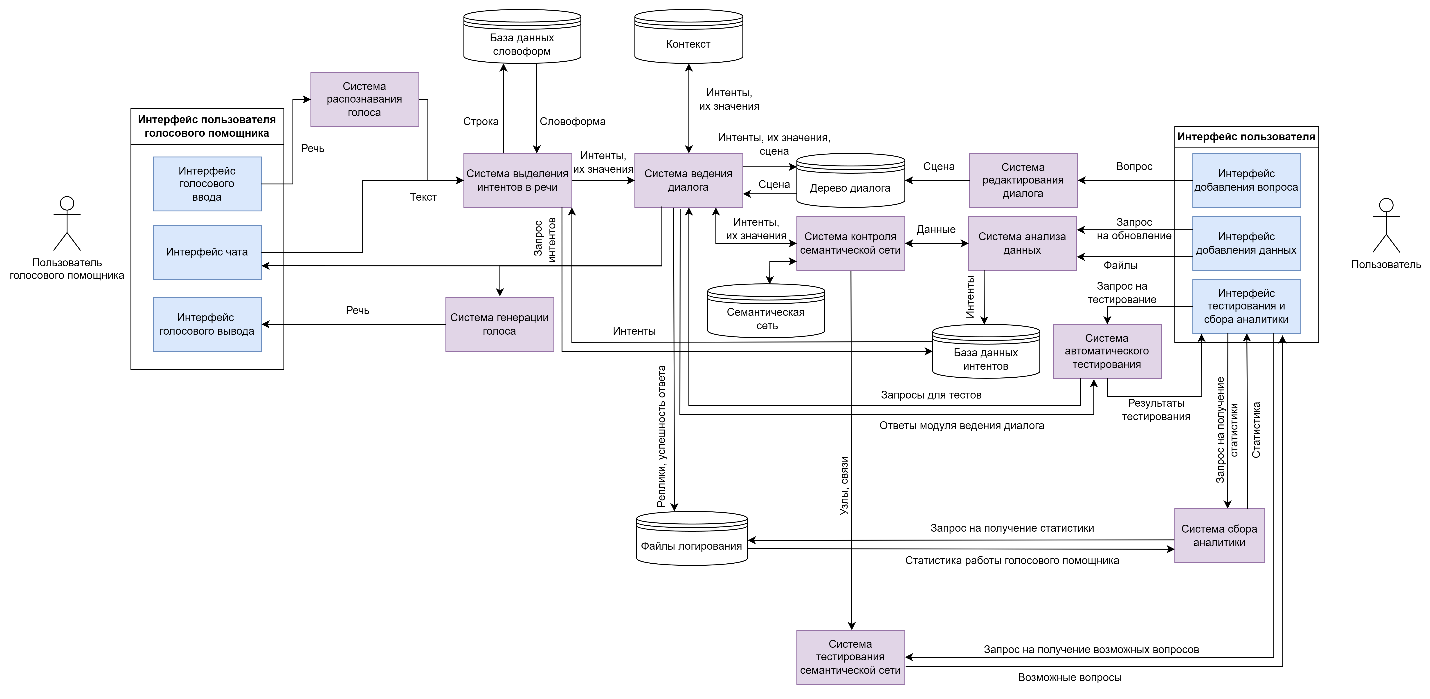


Рисунок 22 – АКД-диаграмма системы.

Граф сценария модуля диалога представлен на рисунке 23.

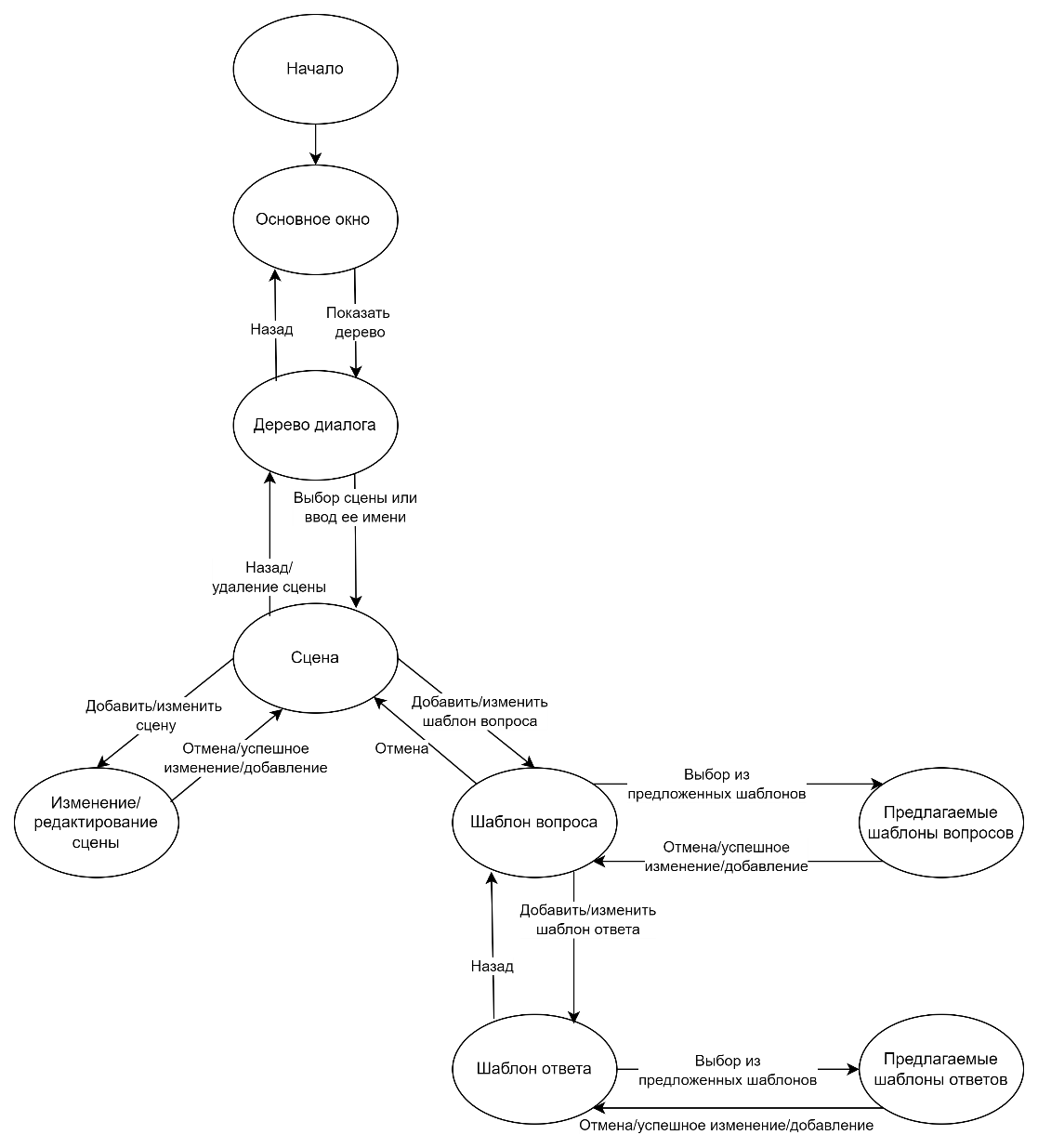


Рисунок 23 – Граф сценария модуля диалога.

**7 Измерения проекта**

Контроль над производственным процессом и его результатами является ключевым видом деятельности на современном предприятии, производящем программное обеспечение на заказ. В силу специфики такого продукта, как программное обеспечение, для оценки эффективности процесса и качества конечного продукта применяются особые методы.

Комплекс мероприятий, направленных на количественную оценку эффективности работы, называется программой измерений.

Метрики, применяемые для проекта:

* Inspection Fault Density (IFD).

(Количество найденных ошибок / Размер рабочего продукта).

* Inspection Preparation Rate (IPR).

(Количество инспекторов \* Размер продукта) / Общее время подготовки.

* Inspection Rate (IR).

Размер продукта / Общее время инспектирования.

**8 Перечень задач проекта**

**Подсистема DM**

**TASK-1 DTR**

Структура дерева

До 1.12 разработать структуру дерева сцен, функции их добавления, удаления, редактирования – REQ TR-1.

**TASK-2 DTR**

Контекст для перехода

До 22.12 разработать метод сохранения контекста диалога, добавить в сцены условия перехода – REQ TR-2, REQ TR-3

**TASK-1 DI**

Обработка речи

До 1.12 разработать методы перевода речи в текст, выделения из текста интентов и значений – REQ I-1.

**TASK-2 DI**

Связь с графом

До 22.12 разработать методы отправки выделенных значений и интентов в граф – REQ I-2.

**TASK-3 DI**

Поле чата

До 22.12 создать в интерфейсе отдельное поле для ввода вопросов с выводом последних вопросов и ответов – REQ I-3, REQ I-4, REQ I-5, REQ I-6.

**TASK-3 DI**

Уточняющий вопрос

До 22.12 добавить метод добавления контекста к вопросу при недостатке интентов для существующих шаблонов или при уточнении вопроса – REQ I-7.

**TASK-DO**

Генерация ответа

До 22.12 добавить кнопку включения микрофона, функции генерации голоса для ответа или при отсутствии ответа – REQ O-1, REQ O-2, REQ O-3.

**TASK-DTemp**

Шаблоны в сценах

До 22.12 добавить в сцены шаблоны ответов и уточняющих вопросов, заполняемых при выводе интентами или их значениями – REQ DTemp-1, REQ DTemp-2.

**Подсистема TM**

**TASK-1 L**

Структура лога

До 22.12 разработать структуру файлов логирования, структуру логов, метод разделения логов по файлам – REQ L-1, REQ L-2, REQ L-3.

**TASK-2 L**

Заполнение логов

До 22.12 разработать методы для модуля диалога для работы с файлами логирования – REQ L-1, REQ L-3.

**TASK-3 SNT**

Поиск состава возможных вопросов

До 22.12 разработать метод для поиска цепочек интентов из графа, которым не соответствуют сцены в дереве диалога – REQ SNT-1.

**TASK-4 DMT**

Метод для автоматического тестирования модуля диалога

До 22.12 разработать метод для автоматического тестирования модуля диалога по составу успешного файла логирования – REQ DMT-1.

**TASK-5 L**

Открытие файлов логирования

До 22.12 разработать метод для доступа пользователям файлов логирования – REQ L-4.

**TASK-5 DMT**

Запуск автоматического тестирования

До 22.12 разработать способ для запуска пользователями автоматического тестирования модуля диалога и получения результатов тестирования – REQ DMT-2.

**TASK-6 DMT**

Создание файла логирования автоматического тестирования

До 22.12 разработать метод для заполнения файлов логирования во время автоматического тестирования ­– REQ-DMT-3.

**TASK-7 SNT**

Вывод состава возможных вопросов

До 22.12 создать способ для доступа пользователей к составу возможных вопросов – REQ-SNT-2.

**Подсистема SNM**

**TASK-1 SN**

Разработать методы для анализа документов и преобразования их в данные – REQ-SN-1, REQ-SN-2, REQ-D-1, REQ-D-2, REQ-SI-1, REQ-SI-2, REQ-SI-3, REQ-SI-4, REQ-UI-1.

**TASK-2 SN**

Разработать методы для добавления и обновления данных в семантическую сеть – REQ-SN-1, REQ-SN-2, REQ-D-1, REQ-D-2.

**TASK-3 SN**

Разработать метод для удаления данных из семантической сети – REQ-SN-1, REQ-SN-2, REQ-D-1, REQ-D-2.

**TASK-1 D**

Разработать структуру для хранения семантической сети – REQ-SN-1, REQ-SN-2, REQ-D-1, REQ-D-2, REQ-SI-1, REQ-SI-2, REQ-SI-3, REQ-SI-4, REQ-UI-1.

**TASK-1 SI**

Разработать метод для поиска информации в семантической сети – REQ-SI-1, REQ-SI-2, REQ-SI-3, REQ-SI-4, REQ-UI-1.

**9 Требования к коду**

Требования к коду:

* Следует использовать 4 пробела или табуляцию для отступов, аргументы нужно располагать под аргументами.
* Массивы объявляются таким образом, чтобы скобки находились отдельно от элементов массива.
* Максимальная длина строки составляет 79 символов.
* При переносе математических выражений новые строки должны начинаться с математических операторов.
* Объявление функций и классов должно быть отделено двумя пустыми строками.
* Импортирование каждой библиотеки должно находиться на новой строке.
* Должны быть использованы двойные кавычки. Можно использовать одинарные кавычки внутри двойных.
* Запятые и аргументы не должны окружаться дополнительными пробелами.
* Логические операторы (==, <, >, !=, <>, <=, >=, in, not in, is, is not) должны окружаться пробелами, в выражении пробелы должны быть только вокруг оператора с наименьшим приоритетом.
* Комментарии должны быть полными предложениями. Первое слово должно быть написано с заглавной буквы, если только оно не является идентификатором, начинающимся со строчной буквы.
* Модули должны иметь короткие имена, состоящие только из строчных букв. В названии модуля можно использовать символы подчеркивания.

Рекомендации к коду:

* Сравнения с одиночными элементами, такими как None, всегда следует выполнять с помощью is или is not, но никогда с операторами равенства.
* В функции допустимо использование нескольких return, однако не допустимо одновременное использование возвратных и невозвратных return. Возвратный return возвращает значение, невозвратный return не возвращает значения, поэтому он должен возвращать None.

**10 Разработка плана тестирования проекта**

**Подсистема TM-SNT**

* Тест TEST-SNT-1

Тестируемые требования: REQ-SNT-1, REQ-SNT-2

Описание теста: открыть раздел «Возможные вопросы» окна «Модуль тестирования».

Ожидаемый результат: возможный состав вопросов выводится в нужном разделе. Состав вопросов выводится корректно, соответствует задаче.

**Подсистема TM-DMT**

* Тест TEST-DMT-1

Тестируемые требования: REQ-DMT-1, REQ-DMT-2

Описание теста: нажать на кнопку «Автоматическое тестирование» окна «Модуль тестирования»

Ожидаемый результат: результат тестирования выводится и соответствует настоящему состоянию системы.

* Тест TEST-DMT-2

Тестируемые требования: REQ-DMT-3

Описание теста: открыт отчет по автоматическому тестированию.

Ожидаемый результат: в отчете по автоматическому тестированию по успешным логам цепочки диалогов выводятся полностью.

**Подсистема TM-L**

* TEST-L-1

Тестируемые требования: REQ-L-1, REQ-L-2, REQ-L-3

Описание теста: на вход модулю тестирования подаются по очереди успешные, неуспешные, неудачные цепочки диалогов.

Ожидаемый результат: в каждом из файлов логирования появляется по одному логу, логи отображены корректно, интенты и значения выделены корректно.

TEST-DMT-2

* TEST-L-2

Тестируемые требования: REQ-L-4

Описание теста: нажата кнопка «Посмотреть логи» окна «Модуль тестирования».

Ожидаемый результат: открываются нужные файлы логирования, содержимое отображается корректно.

**11 Тестирование проекта**

**Подсистема TM-SNT**

* Тест TEST-SNT-1

Тестируемые требования: REQ-SNT-1, REQ-SNT-2

Описание теста: открыть раздел «Возможные вопросы» окна «Модуль тестирования».

Ожидаемый результат: совпадает с предполагаемым.

Резюме: тест пройден.

**Подсистема TM-DMT**

* Тест TEST-DMT-1

Тестируемые требования: REQ-DMT-1, REQ-DMT-2

Описание теста: нажать на кнопку «Автоматическое тестирование» окна «Модуль тестирования»

Ожидаемый результат: совпадает с предполагаемым.

Резюме: тест пройден.

* Тест TEST-DMT-2

Тестируемые требования: REQ-DMT-3

Описание теста: открыт отчет по автоматическому тестированию.

Ожидаемый результат:

Резюме:

**Подсистема TM-L**

* TEST-L-1

Тестируемые требования: REQ-L-1, REQ-L-2, REQ-L-3

Описание теста: на вход модулю тестирования подаются по очереди успешные, неуспешные, неудачные цепочки диалогов.

Ожидаемый результат: в каждом из файлов логирования появляется по одному логу, логи отображены корректно, интенты и значения выделены корректно.

TEST-DMT-2

* TEST-L-2

Тестируемые требования: REQ-L-4

Описание теста: нажата кнопка «Посмотреть логи» окна «Модуль тестирования».

Ожидаемый результат: совпадет с предполагаемым.

Резюме: тест пройден.

**Заключение**

В рамках курсовой работы было разработано программное средство «Система управления навыками голосового помощника» с использованием подходов коллективной промышленной разработки, для чего были решены следующие поставленные задачи:

1. Разработан план проекта.
2. Разработан регламент проведения инспекции.
3. Разработана модель состояний задач.
4. Разработана презентация проекта.
5. Разработаны требования к проекту.
6. Разработана архитектура проекта.
7. Разработаны измерения проекта.
8. Разработан перечень задач проекта.
9. Разработаны рекомендации по кодированию.
10. Разработан план тестирования проекта.
11. Протестирован проект.

Таким образом, цель данной курсовой работы была достигнута.

По окончанию всех работ посчитаем метрики, описанные в 7 главе. Эффективность процесса разработки программного средства – 7 дней работы. Качество продукта выразилось в 28% плотности неполадок.

**Список литературы**