**Федеральное государственное бюджетное образовательное учреждение**

**высшего образования «Калининградский государственный**

**технический университет»**

Кафедра систем управления и вычислительной техники

Заведующий кафедрой СУ и ВТ,

к.т.н., доцент

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_В.А. Петрикин

«\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_ 2018 г.

Д**опущен к защите**

Декан факультета автоматизации

производства и управления

к.т.н., доцент

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ А.В. Калинин

«\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_ 2018 г.

**«Автоматизированная система организации производственного процесса в студии создания графического контента»**

Выпускная квалификационная работа

09.03.01 Информатика и вычислительная техника

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

ВКР.09.03.01.ОЧ.2018.723ОЧ.17.ПЗ

Руководитель

доцент

\_\_\_\_\_\_\_\_\_\_\_\_\_\_В.Г. Пожидаев

(подпись, дата)

Консультант по

экономическому разделу:

к.э.н., доцент

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_М.В. Соловей

(подпись, дата)

Работу выполнил

студент 14-ВТ

\_\_\_\_\_\_\_\_\_\_\_\_\_\_Саркисян Э.В.

(подпись, дата)

Норм контролёр

д.п.н., к.т.н., профессор

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_И.Д. Рудинский

(подпись, дата)

Калининград

2018

**Федеральное государственное бюджетное образовательное учреждение**

**высшего образования «Калининградский государственный**

**технический университет»**

Факультет Автоматизации Производства и Управления

Кафедра Систем Управления и Вычислительной техники

Направление подготовки Прикладная информатика

**УТВЕРЖДАЮ**

Зав. кафедрой СУ и ВТ

к.т.н., доцент

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ В.А. Петрикин

«\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2018 г.

**ЗАДАНИЕ**

**на выпускную квалификационную работу студенту**

Саркисяну Эдуарду Валерьевичу

1. Тема выпускной квалификационной работы: Автоматизированная система организации производственного процесса в студии создания графического контента. Утверждена приказом от 15 мая 2018 г. № 723ОЧ.
2. Срок сдачи студентом выпускной квалификационной работы \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
3. Исходные данные к выпускной квалификационной работе:

* документация приложения «Discord»;
* методические материалы по фреймворку «Qt 5.10».

1. Содержание расчетно-пояснительной записки (перечень подлежащих разработке вопросов):

* анализ объекта автоматизации и постановка задач выпускной квалификационной работы;
* предлагаемые проектные решения;
* реализация и внедрение Системы;
  1. Узловой вопрос выпускной квалификационной работы: Разработка автоматизированной системы организации производственного процесса в студии создания графического контента.

1. Перечень графического материала:

* организационная структура студии;
* функциональная структура студии;
* схемы технологических процессов обработки данных;
* примеры чертежей форм документов и видеокадров.

1. Консультанты по выпускной квалификационной работе (с указанием относящихся к ним разделов):

|  |  |  |  |
| --- | --- | --- | --- |
| Раздел | Консультант | Подпись консультанта; дата | |
| Задание выдал | Задание принял |
| Экономическая эффективность создания Системы | к. э. н. Соловей М. В. |  |  |

1. Дата выдачи задания \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Руководитель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_доцент / В.Г. Пожидаев/

Задание принял к исполнению \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/ Э.В. Саркисян /

АННОТАЦИЯ

Данная пояснительная записка содержит описание результатов выполнения выпускной квалификационной работы.

Объектом выполнения работы выступает автоматизированная система организации производственного процесса в студии создания графического контента.

Процесс выполнения работы включает анализ объекта автоматизации, постановку задачи, разработку путей её решения, а также выбор средств и реализацию предложенных решений.

Результатом выполнения работы являются работоспособная автоматизированная система организации производственного процесса в студии создания графического контента на базе фреймворка «Qt» и комплект проектной документации на ее внедрение и эксплуатацию.

Пояснительная записка содержит 30 страниц, иллюстрации, 11 таблиц, 3 приложения, 12 источников.

**ANNOTATION**

This explanatory note contains a description of the results of the final qualifying work.

The object of the work is an automated system for organizing the production process in the studio creating graphic content.

The process of performing the work includes the analysis of the object of automation, the formulation of the problem, the development of ways to solve it, as well as the choice of means and the implementation of the proposed solutions.

The result of the work is a workable automated system for organizing the production process in the studio creating graphic content based on the "Qt" framework and a set of project documentation for its implementation and operation.

**Оглавление**

[1 АНАЛИЗ ОБЪЕКТА АВТОМАТИЗАЦИИ 10](#_Toc515938192)

[1.1 Общая характеристика студий 10](#_Toc515938193)

[1.2 Стратегические цели студий 11](#_Toc515938194)

[1.3 Организационная структура студий 12](#_Toc515938195)

[1.3.1 Отдел моделирования 13](#_Toc515938196)

[1.3.2 Отдел текстурирования 14](#_Toc515938197)

[1.3.3 Отдел VFX 14](#_Toc515938198)

[1.3.4 Отдел сценария и концепта 14](#_Toc515938199)

[1.3.5 Отдел сетапа 15](#_Toc515938200)

[1.3.6 Отдел анимации 15](#_Toc515938201)

[1.3.7 Отдел рендеринга 15](#_Toc515938202)

[1.3.8 Отдел композитинга 16](#_Toc515938203)

[1.4 Бизнес-модель студии 16](#_Toc515938204)

[2 ВЫЯВЛЕНИЕ ПРОБЛЕМ И ПОСТАНОВКА ЗАДАЧ 17](#_Toc515938205)

[2.1 Проблемы студии 17](#_Toc515938206)

[2.1.1 Проблемы организационного характера 17](#_Toc515938207)

[2.1.2 Проблемы рабочего характера 17](#_Toc515938208)

[2.1.3 Влияние проблем на деятельность студии 17](#_Toc515938209)

[2.2 Задачи для решения проблем 18](#_Toc515938210)

[3 ТРЕБОВАНИЯ К СИСТЕМЕ 19](#_Toc515938211)

[3.1 Требования к структуре и функционированию системы. 19](#_Toc515938212)

[3.2 Требования к персоналу 20](#_Toc515938213)

[3.3 Требования безопасности 20](#_Toc515938214)

[3.4 Требования к защите информации от несанкционированного доступа 21](#_Toc515938215)

[3.5 Требования по сохранности информации при авариях. 21](#_Toc515938216)

[3.6 Требования по стандартизации и унификации 21](#_Toc515938217)

[3.7 Требования к видам обеспечения 22](#_Toc515938218)

[3.7.1 Требования к информационному обеспечению 22](#_Toc515938219)

[3.7.2 Требования к лингвистическому обеспечению 22](#_Toc515938220)

[3.7.3 Требования к методическому обеспечению. 22](#_Toc515938221)

[3.8 Описание автоматизируемых функций 22](#_Toc515938222)

[3.8.1 Распределение функций 22](#_Toc515938223)

[4 ОБЗОР ГОТОВЫХ РЕШЕНИЙ 25](#_Toc515938224)

[4.1 Система управления производственным процессом «Alienbrain» 25](#_Toc515938225)

[4.1.1 Структура «Alienbrain» 25](#_Toc515938226)

[4.1.2 Возможности «Alienbrain» 25](#_Toc515938227)

[4.2 Системы голосовых конференций 27](#_Toc515938228)

[4.2.1 Приложение «Skype» 27](#_Toc515938229)

[4.2.2 Приложение «Teamspeak» 27](#_Toc515938230)

[4.2.3 Приложение «Discord» 28](#_Toc515938231)

[4.3 Выводы обзора готовых проектных решений 29](#_Toc515938232)

[4.3.1 Выводы обзора системы «Alienbrain» 29](#_Toc515938233)

[4.3.2 Выводы обзора систем голосовых конференций 29](#_Toc515938234)

[5 ПРЕДЛАГАЕМЫЕ ПРОЕКТНЫЕ РЕШЕНИЯ 30](#_Toc515938235)

[5.1 Сетевые характеристики взаимодействия приложений 30](#_Toc515938236)

[5.2 Поддерживаемые ОС 30](#_Toc515938237)

[5.3 Описание информационной базы системы 31](#_Toc515938238)

[5.4 Комплекс технических средств 31](#_Toc515938239)

[5.4.1 Структура комплекса технических средств 31](#_Toc515938240)

[5.4.2 Список технических средств 33](#_Toc515938241)

[5.4.3 Описание функционирования КТС 33](#_Toc515938242)

[5.4.4 Конфигурация комплекса технических средств 33](#_Toc515938243)

[5.4.5 Методы защиты технических средств 33](#_Toc515938244)

[5.4.6 Аппаратура передачи данных 33](#_Toc515938245)

[5.4.7 Организация обслуживания комплекса технических средств 34](#_Toc515938246)

[5.5 Необходимые инструменты для реализации 34](#_Toc515938247)

[6 ТЕХНИКО-ЭКОНОМИЧЕСКОЕ ОБОСНОВАНИЕ СОЗДАНИЯ «АСОППС» 35](#_Toc515938248)

[6.1 Этапы разработки системы 35](#_Toc515938249)

[6.2 Оценка затрат на разработку системы 36](#_Toc515938250)

[6.3 Преимущества реализации системы 37](#_Toc515938251)

[6.4 Экономическое преимущество внедрения системы в студию 37](#_Toc515938252)

[7 РЕАЛИЗАЦИЯ «АСОППС» 38](#_Toc515938253)

[7.1 Обоснование выбора объектно-ориентированного подхода и средств разработки 38](#_Toc515938254)

[7.1.1 Объектно-ориентированный подход 38](#_Toc515938255)

[7.1.2 Язык программирования «Python» 39](#_Toc515938256)

[7.1.3 Qt-фреймворк 40](#_Toc515938257)

[7.2 Классы системы 41](#_Toc515938258)

[7.2.1 Сущности системы 42](#_Toc515938259)

[7.2.2 Исполняющие механизмы 47](#_Toc515938260)

[7.2.3 Графический интерфейс приложений 47](#_Toc515938261)

Введение

Эффективность деятельности предприятий в рыночных условиях напрямую зависит от оперативной работы всех его подразделений, скорости и качества выполнения различного рода работ, точности и своевременности расчетов с контрагентами.

Решение этих задач связано с необходимостью обработки значительных объемов информации в короткие сроки, организации деятельности многих людей, работающих с единым набором данных. Эффективным решением подобных задач является применение информационных систем, объединяющих в себе большую часть информационных процессов, происходящих на предприятии.

При внедрении систем автоматизации повышается уровень и скорость обслуживания, минимизируются ошибки и неточности при выполнении работ и расчетов с контрагентами, а менеджеры различных уровней и сам руководитель предприятия получают возможность в любой момент проанализировать как свою работу, работу своих подчиненных, так и работу всей компании.

Всё вышесказанное применимо и к сфере медийного бизнеса, который в настоящее время становится популярным в РФ. Даже на текущий момент, учитывая распространенность и многообразие информационных технологий, во многих студиях, занимающихся разработкой графического контента, сотрудники (художники) некорректным образом организовывают свою деятельность, ежедневно выполняют рутинную работу, которую следует поручить компьютеру, что значительно усложняет и тормозит общий процесс выполнения и внесение изменений в проект.

Такого рода предприятие действует как универсальная система, предоставляющая клиентам услуги в разработке видеороликов, изображений и прочих медиа-материалов.

Схемы функционирования предприятий данного характера очень схожи между собой и отличаются наличием тех или иных отделов или процессов производства. Данный факт позволяет не рассматривать конкретную студию и создать универсальную систему, которая будет легко внедряема на каждый объект.

В данной выпускной квалификационной работе (ВКР) описываются решения по созданию автоматизированной системы организации производственного процесса в студиях создания графического контента на базе фреймворка «Qt». Для простоты условимся, что студии, разрабатывающие графический контент, будем называть «CG-студии» или «студии», а разрабатываемую систему – «СG X Studio» или «АСОППС» (автоматизированная система организации производственного процесса в студии). Моделирование системы и бизнес-процессов разрабатывалось в пакете бизнес-моделирования «Business Studio». Всё отчеты, касающиеся модели, также были сгенерированы в данном пакете.

Данная ВКР содержит семь глав:

1. «Анализ объекта автоматизации»;
2. «Выявление проблем и постановка задач»;
3. «Требования к системе»;
4. «Обзор готовых решений»;
5. «Предлагаемые проектные решения»;
6. «Технико-экономическое обоснование создания АСОППС»;
7. «Реализация АСОППС»;
8. «Описание контрольного примера».

Первая глава содержит результаты обследования предприятия, его организационную и функциональную структуру. Построена детальная функциональная модель основных рабочих процессов в нотации IDEF0, заданы параметры ее элементов.

Вторая глава содержит описание выявленных в ходе анализа предприятия проблем и постановку задач для их решения.

В третьей главе приведены требования к автоматизируемой системе.

В четвертой главе рассматриваются готовые проектные решения.

В пятой главе описываются предлагаемые проектные решения, а также необходимый инструментарий для реализации.

В шестой главе приведено экономическое обоснование реализации системы, а также экономическая эффективность внедрения разрабатываемого продукта;

В седьмой главе детально рассмотрена реализация автоматизированной системы организации производственного процесса в студиях.

В восьмой главе приведен пример использования системы.

# АНАЛИЗ ОБЪЕКТА АВТОМАТИЗАЦИИ

## Общая характеристика студий

CG-Студии – это предприятия, специализирующиеся на комплексном решении сложных задач в области компьютерной графики для кино и рекламы. Они сопровождают своих клиентов в течение всего цикла производства проекта: от разработки художественных решений для картины или рекламного ролика, создания аниматиков и превизуализации, супервайзинга и контроля съемки кадров под графику на площадке до постпродакшена и цветокоррекции.

Спектр решаемых студиями задач включает в себя разработку:

концепт-арта;

дизайна и превизуализации проекта;

работу с визуальными эффектами (взрывы, разрушения, симуляции жидкостей, дымов, огня, тканей);

анимацию объектов и персонажей;

2D и 3D композитинг;

моушн-дизайн.

Для работы над проектами студии используют не только инструменты и программные пакеты, давно ставшие индустриальным стандартом, но и также могут развивать свои собственные разработки.

В РФ отсутствуют законы, регулирующие тематику и содержание конечного продукта студий компьютерной графики, поэтому руководители таких предприятий оперируют моральными принципами при выборе дизайна, стиля и темы проекта в качестве допустимого к производству.

Ниже приведен список нескольких крупных Российских студий.

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Местоположение | Направление | Персонал |
| ALGOUS Studio | Москва | Фильмы, реклама | До 100 |
| Amalgama VFX | Москва | Фильмы | Больше 100 |
| CGF | Москва | Фильмы, реклама, геймдев | До 100 |
| Asymmetric VFX | Санкт-Петербург | Мультфильмы, геймдев | Больше 100 |
| CineLab | Москва | Фильмы | До 100 |

## Стратегические цели студий

Стратегические цели представляют собой результаты, которых стремится достичь компания в перспективе.

В соответствии с методологией «системы сбалансированных показателей (Balanced Scorecard)» цели разбиваются на четыре блока:

* финансы;
* клиенты;
* бизнес-процессы;
* развитие и обучение.

Основной целью студии является увеличение чистой прибыли. Так как доход поступает только от проектной деятельности, каждая студия старается придерживаться конкурентоспособному качеству своего продукта, либо стать лидером в своей области.

Студий очень много, поэтому конкуренция заставляет поддерживать своих специалистов на высоком уровне, поэтому данный аспект является одной из важных целей.

Для обучения персонала требуется время и средства. Если сократить время выполнения проекта, можно выделить его на обучение начальников отделов или потенциальных художников, которые в дальнейшем позволят решать сложные задачи на порядок быстрее или смогут заниматься инновационными разработками, что в свою очередь тоже позволит увеличить эффективность выполнения проекта.

Все выше сказанное окажет влияние на улучшение качества конечного продукта студий, за счёт этого вырастет клиентская база и увеличится доход студии.

В приложении А приведена стратегическая карта студий.

## Организационная структура студий

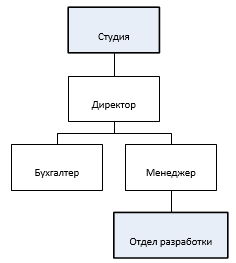


Рис. 1.1 Организационная структура студий

Из рисунка 1.1 «Организационная структура студий» видно, что студии имеют не сложную структуру.



Рис. 1.2 Неотъемлемая часть организационной структуры

отдела разработки

Полная организационная структура отдела разработки приведена в приложении Б.

Стоит обратить внимание на отдел разработки, количество подотделов которого может варьироваться.

На рисунке 1.2 «Неотъемлемая часть организационной структуры» приведена структура подотделов, которая присутствует в отделе разработки каждой студии. Дополнительными подотделами (далее по тексту просто «отдел») могут стать отделы:

* инновационных разработок;
* моушн дизайна;
* специальных эффектов;
* звуковых эффектов;
* пластического грима.

Также студия может организовать не стандартный отдел для реализации задач, которые, по ее мнению, являются важнейшими при разработке проекта.

Каждым отделом управляет начальник отдела.

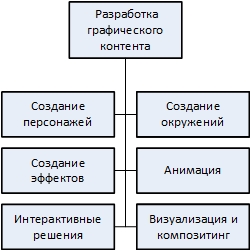


Рис. 1.3 Функциональная структура студии

Функциональная структура студии, изображенная на рисунке 1.3, во многом совпадает с его организационной структурой.

### Отдел моделирования

Данный отдел по эскизам (концепт-артам) создает цифровые модели персонажей и элементов окружений. Именно так закладываются будущие возможности героев, их способности двигаться и переживать эмоции.

Специалисты отдела моделирования обладают объемным мышлением и в совершенстве владеют программным обеспечением. Цифровому скульптору необходимо не только добиться внешнего сходства модели с прообразом, но и технически правильно воплотить модель, учесть поведенческие особенности персонажа во время анимации, деформацию мускулов, тканей и много другое.

3D моделлеры работают на начальном этапе производственного цикла и создают фундамент для успешной и комфортной работы многих других отделов.

### Отдел текстурирования

Данный отдел в соответствии с эскизами (концепт-артами) рисуют текстуры для объектов окружений и персонажей. Технические требования, такие как размер текстур, формат, детализация, устанавливаются начальником отдела.

Для каждого объекта создается определенный набор текстур – текстуры цвета, рельефа, бликов, иногда используются фотографии поверхностей различных предметов. С их помощью специалисты имитируют физические свойства любого материала.

### Отдел VFX

VFX (от англ. *Visual Effects*) – визуальные эффекты. Данный отдел занимается разработкой визуальных эффектов, которые невозможно смоделировать или создать в реальной жизни. Например, невозможно создать разрушение целого города на киносъемочной площадке. Ниже приведены несколько примеры эффектов:

* взрывы;
* разрушения;
* симуляции жидкостей;
* симуляции дымов;
* симуляции огня;
* тканей;
* нетривиальное поведение частиц;
* световые эффекты.

### Отдел сценария и концепта

Данный отдел является ключевым на этапе производства. Он занимается разработкой сценария фильма, рекламы, игры или небольшого ролика и отрисовкой концептов в заданной заказчиком тематике и стилистике. Стоит отметить, что продукт отдела является регламентом (в плане визуального представления) для всех остальных отделов. Художники не могут создавать модели, текстуры, эффекты и многое другое на своё усмотрение, они должны строго придерживаться концепт-артам, созданным в данном отделе.

### Отдел сетапа

Данный отдел занимается подготовкой персонажей и предметов к анимации. Создают анимационный скелет, делают для него элементы управления, создают мускульную систему и привязывают ее к модели, а также подготавливают одежду к просчету коллизий и динамики. Для каждого персонажа необходим индивидуальный подход на сетапе, так как в зависимости от действий, которые по сюжету совершает персонаж в кадре, используются различные комплексы технологических решений.

Одна из основных задач сетапа-художника – это достижение удобства работы аниматора с моделью. Отделы анимации и сетапа работают в тесном сотрудничестве.

### Отдел анимации

Данный отдел занимается анимированием персонажей и объектов. Используются все существующие методы создания анимации:

* key frames (ручной метод);
* motion capture (захват движения).

На первом этапе анимация еще сырая и требует доработки.

На втором этапе прорабатывается поведение одежды персонажей, волос, меха, тканей, окружающих объектов, взаимодействующих друг с другом в соответствии с законами физики.

На третьем этапе (завершающим) производится доскональная проверка анимации в каждом кадре – cleanup. Проверяется каждая сцена на предмет наличия ошибок в анимации, нестыковок по монтажу, наличия всех объектов в сцене и многих других мелочей.

### Отдел рендеринга

Данный отдел занимается постановкой света в сцене, настройкой материалов и шейдеров, а также производит конечную визуализацию сцены. Отдел текстурирования функционально подчиняется отделу рендеринга, так как все текстуры используются в материалах объектов или в создаваемых шейдерах. Главной целью отдела является сырая, но максимально качественная картинка, которая отправится в отдел композитинга на ручную доработку для придания её большей реалистичности.

### Отдел композитинга

Данный отдел собирает все отрендеренные сцены, дорабатывает их, создаёт 2D визуальные эффекты, компонует всё с отснятым материалом и производит финальную цветокоррекцию. После цветокоррекции монтажёр готовит финальный видеоролик, и вставляет в него звуковую дорожку.

## Бизнес-модель студии

Для описания информационных бизнес-процессов прекрасно подходит нотация IDEF0. Инструментом моделирования бизнес-процессов стала CASE-система «Business Studio». Все диаграммы и регламентирующие документы были сгенерированы в данном программном комплексе.

Диаграммы процессов приведены в приложении В, а регламенты в приложении Г.

Каждый документ описывает все важные атрибуты процессов, такие как:

содержание деятельности процесса;

владельцев процесса;

исполнителей процесса;

начало выполнения процесса;

результаты процесса;

требования к срокам выполнения процесса;

документация процессов;

взаимодействие с другими процессами;

описание подпроцессов.

Также в конце документов находится таблица состава набора объектов и ролей, участвующих в выполнении процесса.

Стоит отдельно описать главный процесс «А0 Создание графического контента», содержанием деятельности которого является разработка (по требованию заказчика) комплекса 3D-моделей, текстур, анимаций и симуляций, скомпонованных в единый видеоролик или же статическое изображение для применения в рекламе, кинофильме, мобильном приложении или видео игре. Владельцем процесса является директор – он несет полную ответственность перед заказчиком за конечный продукт. В роли исполнителей выступают менеджер и отдел разработки. Начало выполнения проекта определяется в общем техническом задании, согласованным с заказчиком.

# ВЫЯВЛЕНИЕ ПРОБЛЕМ И ПОСТАНОВКА ЗАДАЧ

## Проблемы студии

Проанализировав модель студии и диаграммы рабочих процессов сотрудников (приложение Д), были выявлены 2 группы проблем:

организационные;

рабочие.

### Проблемы организационного характера

Данная группа проблема связана с созданием и настройкой проекта, а также с передачей данных между сотрудниками отдела разработки.

К этой группе относятся следующие проблемы:

1. трудоемкое создание и изменение структуры проекта на серверах отделов;
2. трудоемкое распределение задач по художникам и контроль проделанной работы (начальниками над работой художников);
3. трудоемкая передача данных между сотрудниками студии;
4. отсутствии возможности быстрого доступа к готовым асетам и результатам проекта;
5. отсутствие моментальной связи между художниками и начальниками.

### Проблемы рабочего характера

Данная группа проблема связана с рабочей деятельностью художников и начальников всех отделов.

К этой группе относятся следующие проблемы:

1. длительное создание файловой структуры локального проекта;
2. частые сбои и вылет используемых приложений;
3. долгая настройка приложения перед его запуском;
4. отсутствие логической связи между настройками используемых приложений и текущим проектом;
5. отсутствие автоматизированного инструментария управления личными ресурсами и ресурсами локального проекта.

### Влияние проблем на деятельность студии

Рассмотрим более детально основные выявленные проблемы, приводящие к различного рода потерям.

Во-первых, из-за трудоемкого создания и изменения структуры проекта на серверах отделов невозможно быстро вносить изменения в проект, что приводит к длительному перераспределению файлов проекта. Это приводит к остановке производства более чем 20 минут.

Во-вторых, каждый художник использует большое количество различных приложений в своей рабочей деятельности. Каждое из приложений имеет массу параметров, которые художник настраивает под себя, из этого следует, что после запуска приложения (иногда и перед запуском) нужно уделить около 5 минут на этот процесс. Отрицательной особенность является то, что все эти приложений часто аварийно завершают работу и после повторного запуска требуется настройка параметров. Сохранять настроенные параметры нельзя, так как на рабочем месте могут работать до трех художников (в разные смены). Так же художник вынужден тратить около 5 минут на создание файловой структуры своего локального проекта, описательные создавать файлы для задач.

В-третьих, художники часто консультируются друг у друга и у начальника по вопросам различного рода, для это им приходится покидать свое рабочее место, либо использовать социальные сети, но этот подход не эффективен, потому что социальные сети не оповещают в должной форме о новых сообщениях и не позволяют продемонстрировать свой экран. Из-за этого часто прерывается рабочий процесс многих сотрудников.

В-четвертых, менеджер не имеет доступа к готовым асетам проекта, для презентации заказчику. Менеджер должен отвлекать начальников отдела разработки, чтобы получить финальный результат проекта, или же визуализацию асета.

В совокупности из-за всех выявленных проблем сотрудники студии часто выполняют рутинную работу, которая сильно тормозит разработку проекта.

## Задачи для решения проблем

Для решения выявленных проблем необходимо реализовать систему, которая обеспечила бы:

1. автоматизированное создание и настройку проекта на серверах отдела;
2. общий доступ к исходным данным проекта для каждого начальника;
3. хранение данных проекта в виде ассетов;
4. управление личными ресурсами художникам;
5. возможность внутри студийных голосовых конференций;
6. кастомизацию рабочих мест художников;
7. автоматизирование рутинных процессов;
8. быстрый просмотр готовых асетов и результатов проекта.

# ТРЕБОВАНИЯ К СИСТЕМЕ

## Требования к структуре и функционированию системы.

Система должна быть децентрализованной, то есть состоять из:

* трех подсистем;
* одного модуля;
* одного самостоятельного приложения.

|  |  |
| --- | --- |
| Модуль | Пояснение |
| CGX Pipeline | Конвейер передачи данных |
| Подсистема | Пояснение |
| CGX Artist | АРМ художника |
| CGX Supervisor | АРМ начальника отдела |
| CGX Manager | Мобильное приложение менеджера |
| Приложение | Пояснение |
| CGX Conference | Приложение для голосовых конференций |

Все подсистемы и модуль являются несамостоятельными приложениями, которые должны запускаться на разных машинах. На рисунке 3.1 приведены зависимости всех компонентов системы.

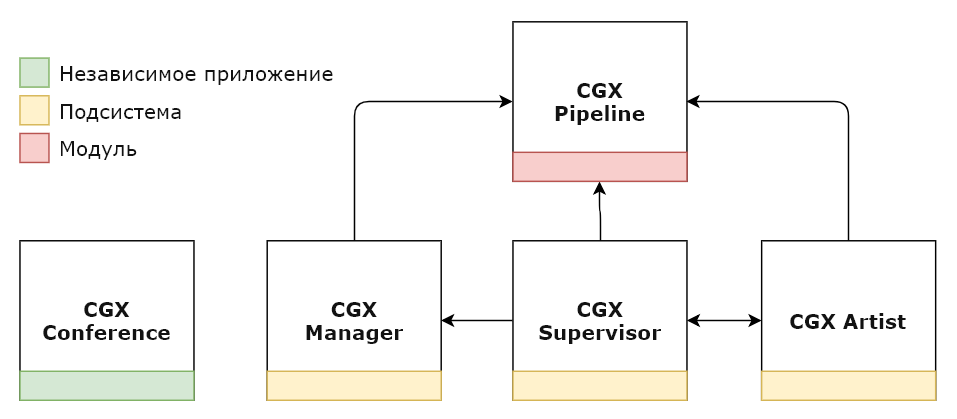


Рис. 3.1 Диаграмма зависимостей компонентов системы

|  |  |
| --- | --- |
| Компонент системы | Реализует |
| CGX Pipeline | связь между подсистемами |
| общий доступ к данным |
| CGX Artist | управление личной библиотекой |
| настройку программ под проект |
| управление локальным инстансом проекта |
| взаимодействие с задачами проекта |
| CGX Supervisor | управление локальным проектом |
| управление задачами проекта |
| контроль выполнения проекта |
| CGX Manager | доступ к готовым асетам и результатам проекта |
| CGX Conference | внутристудийное общение |

## Требования к персоналу

Пользователи требуемой системы должны быть разделены на 3 группы:

* начальники отделов;
* художники;
* менеджеры.

Для всех групп пользователей требуются базовые навыки работы на ПК и чёткое понимание студийного производственного процесса.

## Требования безопасности

Защита технических средств от воздействий электрического тока, электромагнитных полей, акустических шумов и т.п. должна осуществляться в соответствии с требованиями по эксплуатации, предъявляемыми к оборудованию его разработчиками.

Устанавливаемое оборудование должно соответствовать требованиям электробезопасности по ГОСТ Р МЭК 60065-2002.

Допустимый уровень электромагнитных полей на рабочих местах должен отвечать требованиям ГОСТ 12.1.006-84.

Факторы, оказывающие вредные воздействия на здоровье, (в том числе инфракрасное, ультрафиолетовое, рентгеновское и электромагнитное излучения, вибрация, шум, электростатические поля, ультразвук строчной частоты и т.д.), не должны превышать действующих норм (СанПиН 2.2.2. /2.4.1340-03 от 03.06.2003 г. с изменениями от 25 апреля 2007 г.).

Комплекс технических средств должен соответствовать требованиям техники безопасности, основными из которых являются:

* все внешние элементы технических устройств, находящиеся под напряжением, должны иметь защитное заземление;
* технические устройства должны быть установлены в местах, обеспечивающих свободный и безопасный доступ к ним при эксплуатации и проведении профилактического обслуживания;
* сотрудники, которые работают на технических средствах, должны проходить обучение, инструктаж, проверку знаний правил, норм и инструкций по технике безопасности;
* в помещении, предназначенном для эксплуатации технических средств, должны быть обеспечены противопожарные меры безопасности.

## Требования к защите информации от несанкционированного доступа

Работа с системой должна осуществляться только после аутентификации. Регистрация художника и начальника происходит на локальной машине (на той, где установлено приложение).

## Требования по сохранности информации при авариях.

В приложении управления проектом должен быть предусмотрен механизм резервного копирования текущих проектов. Процедура резервного копирования должна производиться начальниками отделов вручную 1 раз в конце рабочего дня для незавершенных проектов.

## Требования по стандартизации и унификации

Система должна быть кроссплатформенной, а стилистика приложений не должна зависеть от платформы. Ниже приведена таблица соответствия приложения и операционных систем, на которых оно должно работать.

## Требования к видам обеспечения

### Требования к информационному обеспечению

Все данные системы должны храниться на физическом носителе «Жёсткий диск», подключенном к серверному компьютеру, в структурированном виде под управлением реляционной СУБД. Исключения составляют файлы данных, предназначенные для просмотра (изображения, анимации, документы и т.п.). Такие файлы сохраняются в файловой системе, а в БД размещаются ссылки на них.

### Требования к лингвистическому обеспечению

Графические интерфейсы приложений должны быть выполнены на русском и английском языках.

### Требования к методическому обеспечению.

В рамках выбора системы должны быть учтены соответствующие административные регламенты, в которых определены процессы деятельности и функции сотрудников, их права, обязанности и ответственность при использовании системы.

Приложения должны включать справочную систему и подсказки в графическом интерфейсе.

## Описание автоматизируемых функций

Согласно этой структуре, Система должна состоять из 5 приложений, каждое из которых содержит комплекс функций.

Схема функциональной структуры и полное описание автоматизируемых функций приведены в приложении Д. Это структура подсистем и ТЗ?

### Распределение функций

#### Подсистема «CGX Supervisor»

Данная подсистема предназначена начальникам отделов. Она взаимодействует с подсистемой «CGX Artist» через модуль «CGX Pipeline» и реализует следующие функции:

* быстрая фильтрация, сортировка и поиск по проекту;
* отсылка готового результата на главный сервер;
* отсылка файлов своим подчиненным;
* подключение к существующему проекту;
* просмотр активных художников;
* управление задачами своего отдела.

#### Подсистема «CGX Artist»

Данная подсистема предназначена художникам отделов. Она взаимодействует с подсистемой «CGX Supervisor» через модуль «CGX Pipeline» и реализует следующие функции:

* создание личного аккаунта;
* автоматическая настройка переменных проекта;
* быстрая фильтрация, сортировка и поиск по личным ресурсам и ресурсам проекта;
* взаимодействие с задачами проекта;
* запуск приложений в окружении проекта;
* быстрый просмотр hdr и exr файлов;
* определение пользовательских настроек;
* отсылка файлов коллегам по отделу;
* подключение к существующему проекту;
* подключение личной библиотеки к аккаунту.

#### Подсистема «CGX Manager»

Данная подсистема предназначается менеджеру отдела разработки, она дает ему возможность создать проект, над которым будет работать его отдел, просматривать асеты проекта и финальные результаты. Так же эта подсистема позволит ему отправлять скриншоты асетов и готовые результаты заказчику.

Функции подсистемы:

* отправка данных заказчику;
* просмотр асетов;
* просмотр готовых результатов.

#### Приложение «CGX Conference»

Мессенджер с поддержкой видеоконференций для пользователей ПК и гаджетов. Данное приложение способно организовывать голосовые конференции с настройкой канала связи и работать по принципу push-to-talk (нажми, чтобы сказать), создавать публичные и приватные чаты для обмена текстовыми сообщениями.

Функции приложения:

* Организация голосовых конференций
* Приватные чаты

#### Модуль «CGX Pipeline»

Данный модуль обеспечивает связь между подсистемами.

Т.е эти приложения – это подсистемы (или модули) разрабатываемой системы?

# ОБЗОР ГОТОВЫХ РЕШЕНИЙ

Перед обзором готовых решений разделим рассматриваемые системы на два типа:

* системы организации производственного процесса в студии;
* системы организации голосовых конференций.

На сегодняшний день на рынке существует множество программных пакетов, позволяющих организовывать голосовые конференции, ниже будут рассмотрены некоторые из них.

С системами организации производственного процесса дела обстоят хуже – имеется лишь один вариант программного обеспечения – «Alienbrain» от компании «Avid».

## Система управления производственным процессом «Alienbrain»

Данная система разработана компанией «Avid» и позволяет сотрудникам студии осуществлять полный контроль над выполнением проекта.

Цена на студию из 10 сотрудников: 99 000 $.

### Структура «Alienbrain»

Данная система состоит из двух подсистем:

«Alienbrain server»;

«Alienbrain client».

Серверное приложение служит хранилищем данных проекта, а также связывает клиентов между собой.

Клиентское приложение является обозревателем контента, находящегося на сервере.

### Возможности «Alienbrain»

#### Централизованное хранение файлов

Все файлы проекта располагаются на центральном сервере студии, где художники могут просматривать их, извлекать или блокировать для редактирования. Данные функции осуществляются через клиентские приложения.

|  |  |
| --- | --- |
| Плюсы | Минусы |
| Все художники знают, где находятся важные файлы, и кто работает над ними в текущий момент. |  |
| Файлы никогда не удаляются случайно и не перезаписываются |  |

#### Контроль версий асетов

Сохраняется история версий всех асетов проекта. Возможно отслеживать изменения и, при необходимости, «откатываться» в любое время.

|  |  |
| --- | --- |
| Плюсы | Минусы |
| Уверенность в правильной версии асета |  |

#### Устойчивость к тяжелым нагрузкам

Большинство цифровых систем управления производственным процессом, рассчитаны на работу с небольшим количеством легких файлов. «Alienbrain» был разработан для совместной работы больших коллективов над массивными проектами, поэтому он продолжает стабильно функционировать даже при высоких нагрузках.

|  |  |
| --- | --- |
| Плюсы | Минусы |
| Стабильная передача данных между сотрудниками |  |

#### Управление проектами и его конфигурациями

«Alienbrain» дает администраторам полный контроль над проектами: от установки разрешений и привилегий, до модификации большого количества файлов, для резервного копирования и архивирования старого контента.

Можно управлять одним или несколькими проектами одновременно. Расширенные инструменты управления конфигурацией проекта позволяют легко поддерживать несколько преднастроенных проектов одновременно без дублирования его содержимого.

|  |  |
| --- | --- |
| Плюсы | Минусы |
| Гибкая организация конфигураций проектов |  |

#### Настройка, автоматизация, интеграция.

Наличие набора интерфейсов прикладного программирования (API) позволяют программно получать доступ к контенту и метаданным.

|  |  |
| --- | --- |
| Плюсы | Минусы |
| Возможность эффективно автоматизировать общие задачи и настраивать инструменты, представления и поведение «Alienbrain» в соответствии с конкретными потребностями проекта. | Высокое требование к навыкам программирования |
|  | Долгое изучение API «Alienbrain» |

## Системы голосовых конференций

### Приложение «Skype»

Бесплатное программное обеспечение с закрытым кодом, обеспечивающее текстовую, голосовую и видеосвязь через Интернет между компьютерами (IP-телефония), опционально используя технологии пиринговых сетей, а также платные услуги для звонков на мобильные и стационарные телефоны.

Программа также позволяет совершать конференц-звонки (до 25 голосовых абонентов, включая инициатора), видеозвонки (в том числе видеоконференции до 10 абонентов), а также обеспечивает передачу текстовых сообщений (чат) и передачу файлов. Есть возможность вместе с изображением с веб-камеры передавать изображение с экрана монитора, а также создавать и отправлять видеосообщения пользователям настольных версий программы.

### Приложение «Teamspeak»

Программа, предназначенная для голосового общения в сети Интернет посредством технологии VoIP. От классического телефона отличается практически неограниченным количеством абонентов, разговаривающих одновременно. Больше всего это похоже на многоканальную рацию, в которой можно одновременно пользоваться несколькими каналами. При этом доступны все опции, разработанные ранее для удобства использования раций в полевых (боевых) условиях. Программы этого типа предназначены прежде всего для геймеров, но могут использоваться везде, где необходима голосовая связь и координация большой группы людей.

Стоит отметить, что данное приложение работает только с аудио-данными и не имеет возможность демонстрации экрана.

### Приложение «Discord»

Бесплатный мессенджер с поддержкой VoIP и видеоконференций, изначально ориентированный для пользователей компьютерных игр. Настольное клиентское приложение реализовано для Windows (работает на Windows 7 и новее), macOS и Linux, мобильное приложение – для Android и iOS, существует также веб-клиент. Поддерживает 27 языков, включая русский.

Единая кодовая база для настольных, веб- и мобильных клиентов обеспечивается благодаря использованию фреймворка Electron. Серверы мессенджера размещены в 11 центрах обработки данных в разных частях мира. Также возможна организация локального сервера.

Приложение способно:

организовывать голосовые конференции с настройкой канала связи и работать по принципу push-to-talk;

создавать публичные и приватные чаты для обмена текстовыми сообщениями, а также оно обладает защитой от DDoS-атак.

Программа имеет браузерную версию, отличием от настольной версии является то, что режим push-to-talk работает только если вкладка с программой в фокусе.

При включении режима «стример» скрывается вся личная информация, отключаются звуки и уведомления на рабочий стол.

Включается режим автоматически (при запуске программ для трансляций, например, OBS Studio[en]), также режим можно включить или отключить вручную.

Для аудио используется кодек Opus, который имеет возможности эхоподавления, подавления посторонних шумов и автоматической регулировкой усиления. Для видео используется кодек VP8.

Поддерживается назначение «горячих клавиш».

Реализована функция отключения уведомлений, возможно отключить уведомления на мобильное устройство при бездействии на компьютере. Есть возможность включить озвучивание уведомлений.

В текстовом чате поддерживается присоединение файлов, картинок, вставка ссылок (для некоторых сайтов работает предпросмотр), форматирование текста и смайлики, размер вложения ограничен объёмом 8 МБ.

## Выводы обзора готовых проектных решений

### Выводы обзора системы «Alienbrain»

Данная система подходит для реализации практически всех поставленных задач. Очень гибкое проектное решение, так как разработчики данного пакета предоставляют API, которое можно использовать, как и в разрабатываемых приложениях, так и в скриптах.

Минусами данной системы являются:

* отсутствие возможности организации голосовых конференций;
* отсутствие запуска программ с определенными переменными окружения.

Но, так как выше было сказано про открытое API, данные проблемы решаемы.

### Выводы обзора систем голосовых конференций

Из рассмотренных выше приложений «Discord» является отличной готовой системой для решения поставленных коммуникационных задач. Данный программный комплекс будет интегрирован в систему «CGX Studio».

# ПРЕДЛАГАЕМЫЕ ПРОЕКТНЫЕ РЕШЕНИЯ

Технические задания на реализацию системы находятся в приложении

## Сетевые характеристики взаимодействия приложений

Так как система должна быть децентрализованной, в качестве протокола взаимодействия между приложениями на транспортно-сетевом уровне необходимо использовать протокол TCP/IP. Для организации информационного обмена между приложениями также должен использоваться протокол TCP/IP.

Ниже в таблице приведены соответствия приложений с понятиями «клиент-сервер»:

|  |  |
| --- | --- |
| Клиент | Сервер |
| CGX Artist | CGX Pipeline |
| CGX Supervisor |  |
| CGX Manager |  |
| CGX Conference |  |

## Поддерживаемые ОС

Так как система должна быть кроссплатформенной, ниже приведена таблица соответствия приложений и операционных систем, на которых они должны работать.

|  |  |
| --- | --- |
| Приложение | ОС |
| CGX Pipeline | Linux Mint |
| KUbuntu. |
| CGX Artist | Windows 10 |
| Linux Mint |
| Ubuntu |
| CGX Supervisor | Windows 10 |
| Linux Mint |
| Ubuntu |
| Android (от 8.0 и выше) |
| iOS (от 9.3.5 и выше) |
| CGX Manager | Android (от 8.0 и выше) |
| iOS (от 9.3.5 и выше) |
| CGX Conference | Windows 10 |
| Linux Mint |
| Ubuntu |
| Android (от 8.0 и выше) |
| iOS (от 9.3.5 и выше) |

## Описание информационной базы системы

Все данные системы будут храниться на физическом носителе «Жёсткий диск», подключенном к серверному компьютеру, в структурированном виде под управлением реляционной СУБД SQLite 3.0. Исключения составляют файлы данных, предназначенные для просмотра (изображения, анимации, документы и т.п.). Такие файлы сохраняются в файловой системе, а в БД размещаются ссылки на них.

## Комплекс технических средств

### Структура комплекса технических средств

Техническую составляющую будут выполнять локальные сервера отделов студии и главный сервер.

Сервера отделов отвечают за передачу данных между начальником отдела и его подчиненными, а также отправку на главный сервер. К главному серверу подключаются сервера отделов и Wi-Fi роутер, который обеспечивает беспроводное соединение приложения «CGX Manager».

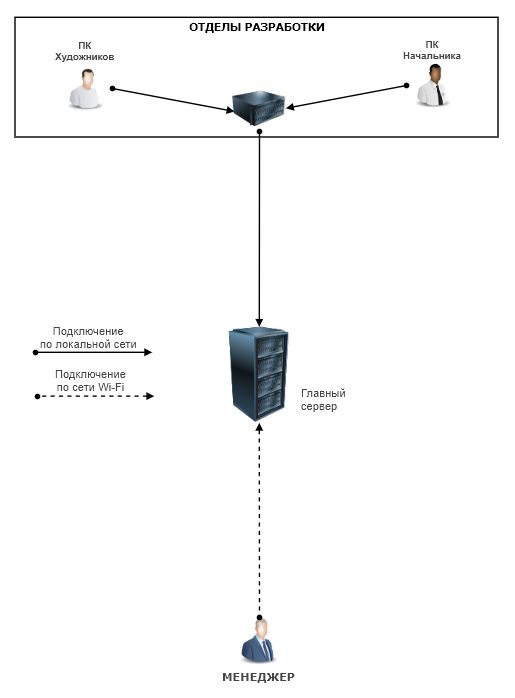


Рис. 5.1 Схема структуры комплекса технических средств

### Список технических средств

* сервера отделов разработки;
* главный сервер;
* ПК начальников отдела;
* ПК художников;
* планшет менеджера.

### Описание функционирования КТС

При нормальном режиме работы все сервера работают непрерывно. Плановая перезагрузка делается автоматически каждую неделю, в 4:00, каждое воскресенье.

При пусковом режиме работы определяется целостность всех модулей серверов и разделов жестких дисков. Все приложения системы доступны для работы только после завершения пускового режима.

При перезагрузке производится быстрая проверка сервера. Проверка сервера осуществляется утилитой «Active Directory Health Profiler».

При аварийном режиме работы, например, при отключении электричества, происходит плавное отключение серверов благодаря наличию устройства бесперебойного питания. После устранения проблем происходит полная проверка целостности всех модулей серверов и разделов жесткого диска.

Описание решений по размещению КТС на объекте внедрения

Так как студия имеет серверную комнату готовую к эксплуатации, все сервера будут размещаться в ней.

### Конфигурация комплекса технических средств

Спецификации требуемого комплекса технических средств приведены в приложении Ж.

### Методы защиты технических средств

Для защиты серверов от проблем с электричеством предусмотрено использование источника бесперебойного питания.

### Аппаратура передачи данных

Для успешного функционирования системы требуется обеспечение пропускной способности сети не менее 10 Гбит/c, чтобы не возникало длительных (свыше 5 мин.) задержек передачи данных больших размеров (до 200 ГБайт).

### Организация обслуживания комплекса технических средств

* Для своевременного и качественного облуживания системы предусмотрено наличие администратора. Администратор должен обладать квалификацией инженера АСОИУ, а также навыками настройки и администрирования серверов, знаниями таких технологий, как ЯП Python и SQL.
* Учитывая, что создателем данной системы является студент, и в дальнейшем производить её сопровождение не сможет, поддержкой будут заниматься работники, нанятые студией.

## Необходимые инструменты для реализации

Языком реализации приложений должен стать «С++» или «Python» с использованием фреймворка «Qt». Графический интерфейс будет написан на языке QML.

Для разработки системы потребуется среда разработки QtCreator в случае выбора языка «С++». Если же выбор падёт на язык Python, то необходимыми станут:

* среда разработки PyCharm от компании Jet Brains
* библиотека PyQt 5.10;
* библиотека xnetwork;
* библиотека send2trash;
* библиотека PyInstaller 3.0.

# ТЕХНИКО-ЭКОНОМИЧЕСКОЕ ОБОСНОВАНИЕ СОЗДАНИЯ «АСОППС»

## Этапы разработки системы

В соответствии с ГОСТ 34.601-90 «Автоматизированные системы. Стадии создания» сформирован план работ по созданию «АСОППС», который представлен в виде таблицы.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Стадии | Исполнитель | Начало | Окон-  чание | Длите-  льность |
| Формирование требований к системе | Системный аналитик, Системный архитектор | Пт. 01.06.18 | Чт. 07.06.18 | 5 дней |
| Разработка концепции системы | Системный аналитик | Пт. 08.06.18 | Пн. 11.06.18 | 2 дней |
| Формирование технического задания | Системный архитектор | Вт. 12.06.18 | Пн. 18.06.18 | 5 дней |
| Эскизное проектирование | Разработчик | Вт 19.06.18 | Чт. 21.06.18 | 3 дней |
| Разработка технического проекта | Программист | Пт. 22.06.18 | Чт. 28.06.18 | 5 дней |
| Ввод в действие | Тестировщик | Пт. 29.06.18 | Вт 03.07.18 | 3 дней |

Планирование проекта производилась в среде управления проектами Microsoft Project 2013. Диаграмма Ганта, наглядно отражающая сроки и последовательность реализации каждого из этапов работ представлена на рис. 6.1.

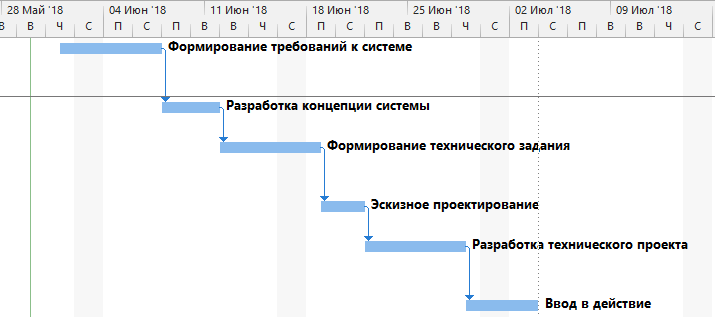


Рис. 6.1 Диаграмма Ганта стадий создания «АСОППС»

## Оценка затрат на разработку системы

Единовременные затраты на разработку и внедрение системы представлены в таблице 4.2

|  |  |
| --- | --- |
| Затраты, связанные с разработкой, внедрением и эксплуатацией | Значение, руб. |
| Установка необходимого оборудования | 0 |
| Разработка структуры системы | 25 000 |
| Документирование системы | 25 000 |
| Кодирование системы | 150 000 |
| Разработка методического материала | 10 000 |
| Разработка дизайна системы | 40 000 |
| Тестирование и отладка | 50 000 |
| Итого: 300 000 | |

## Преимущества реализации системы

В главе «Обзор готовых решений» была рассмотрена система «Alienbrain», которая реализует практически все требуемые задачи. Цена данного продукта на годовую лицензию для студии с составом 50 сотрудников составит 495 000 $ (31 007 790 рублей на момент 30.05.2018). С учетом того, что размеры студии могут превышать число 50, цена на «Alienbrain» будет выше.

Данный аспект позволяет рассматривать реализацию собственной системы более выгодным решением.

## Экономическое преимущество внедрения системы в студию

Разрабатываемая система позволит снизить время разработки проекта, что станет ключевым фактором увеличения дохода.

Появившееся время можно использовать для:

повышения квалификации персонала;

увеличения количества выполняемых проектов.

Повышение квалификации персонала является более эффективным шагом, данный аспект позволит сотрудникам решать более сложные задачи, делать более эффектные визуализации, но также данное решение потребует инвестиций и время.

# РЕАЛИЗАЦИЯ «АСОППС»

## Обоснование выбора объектно-ориентированного подхода и средств разработки

### Объектно-ориентированный подход

Объектно-ориентированное программирование (ООП) – методология программирования, основанная на представлении программы в виде совокупности объектов, каждый из которых является экземпляром определённого класса, а классы образуют иерархию наследования.

Идеологически ООП – подход к программированию как к моделированию информационных объектов, решающий на новом уровне основную задачу структурного программирования: структурирование информации с точки зрения управляемости, что существенно улучшает управляемость самим процессом моделирования, что в свою очередь особенно важно при реализации крупных проектов.

Управляемость для иерархических систем предполагает минимизацию избыточности данных (аналогичную нормализации) и их целостность, поэтому созданное удобно управляемым – будет и удобно пониматься. Таким образом через тактическую задачу управляемости решается стратегическая задача – транслировать понимание задачи программистом в наиболее удобную для дальнейшего использования форму.

Основные принципы структурирования в случае ООП связаны с различными аспектами базового понимания предметной задачи, которое требуется для оптимального управления соответствующей моделью:

**абстрагирование** для выделения в моделируемом предмете важного для решения конкретной задачи по предмету, в конечном счете – контекстное понимание предмета, формализуемое в виде класса;

**инкапсуляция** для быстрой и безопасной организации собственно иерархической управляемости: чтобы было достаточно простой команды «что делать», без одновременного уточнения как именно делать, так как это уже другой уровень управления;

**наследование** для быстрой и безопасной организации родственных понятий: чтобы было достаточно на каждом иерархическом шаге учитывать только изменения, не дублируя все остальное, учтенное на предыдущих шагах;

**полиморфизм** для определения точки, в которой единое управление лучше распараллелить или наоборот – собрать воедино.

То есть фактически речь идет о прогрессирующей организации информации согласно первичным семантическим критериям: «важное/неважное», «ключевое/подробности», «родительское/дочернее», «единое/множественное». Прогрессирование, в частности, на последнем этапе дает возможность перехода на следующий уровень детализации, что замыкает общий процесс.

Обычный человеческий язык в целом отражает идеологию ООП, начиная с инкапсуляции представления о предмете в виде его имени и заканчивая полиморфизмом использования слова в переносном смысле, что в итоге развивает выражение представления через имя предмета до полноценного понятия-класса.

### Язык программирования «Python»

Python это язык программирования общего назначения, нацеленный в первую очередь на повышение продуктивности самого программиста, нежели кода, который он пишет. Говоря простым человеческим языком, на Python можно написать практически что угодно (веб-/настольные приложения, игры, скрипты по автоматизации, комплексные системы расчёта, системы управления жизнеобеспечением и многое многое другое) без ощутимых проблем. Более того, порог вхождения низкий, а код во многом лаконичный и понятный даже тому, кто никогда на нём не писал. За счёт простоты кода, дальнейшее сопровождение программ, написанных на Python, становится легче и приятнее по сравнению с Java или C++. А с точки зрения бизнеса это влечёт за собой сокращение расходов и увеличение производительности труда сотрудников.

Одним из составляющих успеха любой технологии является сообщество, созданное вокруг неё. Именно оно предопределяет будущий вектор развития путём совместных усилий.

Сообщество вокруг Python одно из самых сильных в мире IT. Это сложный хорошо организованный и постоянно развивающийся организм. Помимо сотни тысяч индивидуальных разработчиков и небольших компаний, Python поддерживают такие гиганты IT как:

* Google;
* Dropbox;
* Mozilla;
* Facebook;
* Yandex;
* Red Hat;
* Microsoft (с недавних пор очень активно, в частности с Visual Studio);
* Intel (активно ведёт исследовательскую работу в области параллельных вычислений на Python).

Данный факт свидетельствует о том, что крупные корпорации не боятся строить свой бизнес вокруг Python, они уверены в том, что технология будет жить, а, следовательно, проблем с поиском специалистов ждать не стоит. Более того, разнообразие приложений также радует, что свидетельствует о широком круге задач, которые Python решает мастерски.

### Qt-фреймворк

Многие привыкли считать, что Qt – лишь средство для создания только интерфейса пользователя. Это не так – Qt представляет собой полный инструментарий для программирования, который состоит из отдельных модулей и предоставляет:

поддержку двух- и трехмерной графики (фактически, являясь стандартом для платформонезависимого программирования на OpenGL);

возможность интернационализации, которая позволяет значительно расширить рынок сбыта программ;

использование формата XML (extensible Markup Language);

поддержку стандартных протоколов ввода/вывода;

классы для работы с сетью;

поддержку программирования баз данных, включая:

* 1. Oracle;
  2. Microsoft SQL Server;
  3. IBM;
  4. DB2;
  5. MySQL;
  6. SQLite;
  7. Sybase;
  8. PostgreSQL;

и многое другое.

Qt – полностью объектно-ориентированная библиотека. Новая концепция ведения межобъектных коммуникаций, именуемая «сигналы и слоты», полностью заменяет былую, не вполне надежную модель обратных вызовов. Имеется также возможность обработки событий – например, нажатия клавиш клавиатуры, перемещения мыши и т. д.

Предоставляемая система расширений (plug-ins) позволяет создавать модули, расширяющие функциональные возможности создаваемых приложений. Эти расширения пользователи программы могут получать разных разработчиков.

Несмотря на то, что библиотека Qt изначально создавалась для языка программирования «C++», это вовсе не означает, что ее использование невозможно в других языках. Напротив, во многих языках программирования существуют модули для работы с этой библиотекой – например:

Qt# в С#;

PerlQt в Perl;

PyQt в Python;

РНР и т. д.

Программы, реализованные с помощью Qt, могут использовать язык сценариев QtScript. Эта технология позволяет пользователям приложения расширить возможности без изменения исходного кода и без перекомпоновки самого приложения изменить «поведение» приложения.

Qt прекрасно документирована, благодаря чему с помощью программы «QtAsisstant» всегда можно почерпнуть о ней любую интересующую информацию.

Если быть предельно кратким, то библиотеку Qt можно охарактеризовать в трех словах: простота, быстрота, мощность.

## Классы системы

Для придания принадлежности кода системы к конкретно данной системе, было создано пространство имен (namespace) «CGX». Это означает, что все классы, переменные, функции, интерфейсы и пр. будут иметь префикс пространства имен. Данный подход избавит дальнейшее расширение системы (плагины или доработка) от несанкционированного повреждения всех сущностей языка программирования.

Выделенные классы для реализации системы можно разделить на две группы:

* сущности (структуры данных);
* исполняющие механизмы.

Первый тип классов описывает типы данных, используемые в системе, например, тип «CGXProject». Все эти классы описывают сущности, а их функции являются лишь вспомогательным инструментарием интерфейсов. Второй тип классов манипулирует сущностями и выполняет какие-либо действия, например, запускает программу.

Так как язык Python имеет позднее связывание (динамическую типизацию), все значения параметров функций требуют проверку на соответствие типа.

### Сущности системы

Для реализации системы было выделено 8 сущностей:

|  |  |  |
| --- | --- | --- |
| Сущность | Название класса | Описание |
| Проект | CGXProject | Данная сущность описывает проект, создаваемый менеджером отдела разработки. Далее атрибуты проекта дополняют начальники отделов. К проекту подключаются художники всех отделов |
| Метаинформация | CGXMetainfo | Данная сущность является описательной частью сущности «Проект». Она содержит информацию о заголовке, описании и сроков сдачи проекта |
| Дедлайн | CGXDeadline | Данная сущность хранит сроки создания и сдачи проекта |
| Заказчик | CGXCustomer | Данная сущность описывает заказчика проекта. |
| Контакты | CGXContacts | Данная сущность содержит данные по контактам заказчика: вебсайты, социальные сети, телефоны, мейлы, а так заметки |
| Сотрудник | CGXEmployee | Данная сущность описывает сотрудника студии. Поля «логин» и «пароль» заполняются не для каждого типа сотрудника (например, менеджер не имеет логина и пароля) |
| Программа | CGXProgram | Данная сущность описывает подключенные внешние программы. Были выделены важнейшие атрибуты, которые обеспечивают гибкость при работе с внешними приложениями |
| Переменная | CGXVariable | Данная сущность описывает переменную, которую пользователи будут использовать в различных аспектах системы |

На рисунке 7.1 приведена схема логической структуры связей сущностей

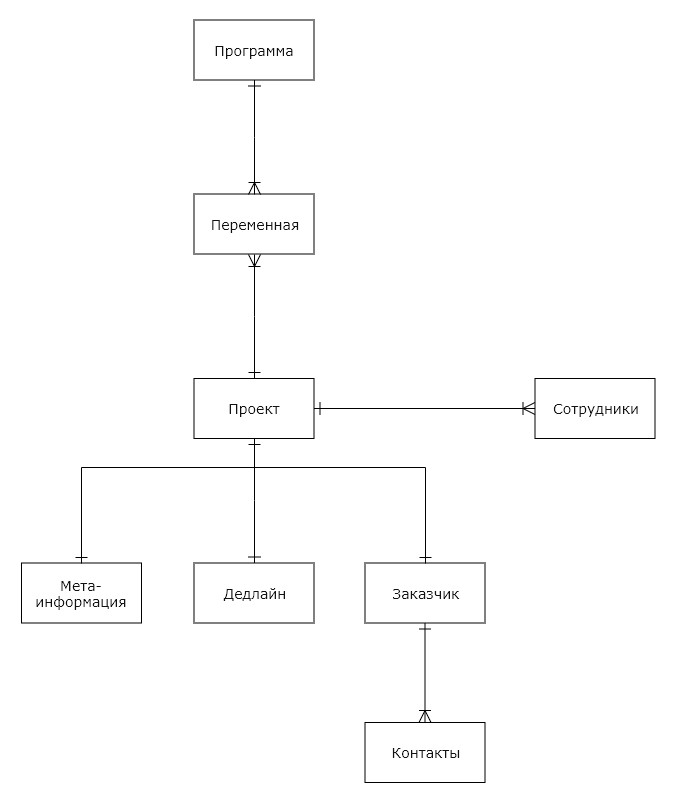


Рис. 7.1 Логическая структура связей сущностей

#### Описание сущностей системы

|  |  |  |  |
| --- | --- | --- | --- |
| Контакты (CGXContacts) | | | |
| Атрибут | В классе | рy-тип | Описание |
| Телефоны | phones | dict{str: str} | Содержит пары { название : номер }. Данное поле имеет тип словарь и хранит все телефоны заказчика. Ключом является название телефона (например «Основной»), а значением – номер. |
| Соц. сети | socialNetworks | dict{str: str} | Содержит пары { название : ссылка }. Данное поле имеет тип словарь и хранит ссылки на страницы социальных сетей заказчика. Ключом является название соц. сети (например «Facebook»), а значением – ссылка. |
| Мессенджеры | messengers | dict{str: str} | Содержит пары { название : ссылка }. Данное поле имеет тип словарь и хранит ссылки на страницы месенджеров заказчика. Ключом является название месенджера (например, «Skype»), а значением – ссылка. |
| Электронная почта | emails | dict{str: str} | Содержит пары { название : ссылка }. Данное поле имеет тип словарь и хранит электронные почты заказчика. Ключом является название (например, «Основная»), а значением – почта. |
| Вебсайты | websites | list(str) | Данное поле имеет тип список и хранит ссылки на вебсайты заказчика. Ссылка хранится в строке. |
| Заметки | notes | str | HTML-cтрока, которая хранит заметки по заказчику |

|  |  |  |  |
| --- | --- | --- | --- |
| Дедлайн (CGXDeadline) | | | |
| Атрибут | В классе | рy-тип | Описание |
| Начало | creationDate | QDate | Хранит дату создания проекта |
| Срок сдачи | deadline | QDate | Хранит дату сдачи проекта |

|  |  |  |  |
| --- | --- | --- | --- |
| Метаинформация (CGXMetainfo) | | | |
| Атрибут | В классе | рy-тип | Описание |
| Заголовок | title | str | Данное поле хранит заголовок проекта |
| Описание | description | str | HTML-cтрока, которая хранит описание проекта |
| Сроки выполнения | deadline | CGXDeadline | Объект, который хранит сроки начала создания проекта и его дедлайн |

|  |  |  |  |
| --- | --- | --- | --- |
| Метаинформация (CGXMetainfo) | | | |
| Атрибут | В классе | рy-тип | Описание |
| Заголовок | title | str | Данное поле хранит заголовок проекта |
| Описание | description | str | HTML-cтрока, которая хранит описание проекта |
| Сроки выполнения | deadline | CGXDeadline | Объект, который хранит сроки начала создания проекта и его дедлайн |

|  |  |  |  |
| --- | --- | --- | --- |
| Заказчик (CGXCustomer) | | | |
| Атрибут | В классе | рy-тип | Описание |
| Имя | name | str | Данное поле содержит имя заказчика |
| Фамилия | surname | str | Данное поле содержит фамилию заказчика |
| Компания | company | str | Данное поле содержит название организации заказчика |
| Контакты | contacts | CGXContacts | Данное поле хранит полную информацию о контактах заказчика |

|  |  |  |  |
| --- | --- | --- | --- |
| Сотрудник (CGXEmployee) | | | |
| Атрибут | В классе | рy-тип | Описание |
| Имя | name | str | Данное поле содержит имя сотрудника |
| Фамилия | surname | str | Данное поле содержит фамилию сотрудника |
| Логин | company | str | Данное поле содержит логин сотрудника (для подсистемы «CGX Artist» и «CGX Supervisor») |
| Пароль | contacts | CGXContacts | Данное поле содержит пароль сотрудника (для подсистемы «CGX Artist» и «CGX Supervisor») |
| Должность | post | str | Содержит должность сотрудника |
| Отдел | department | str | Содержит отдел сотрудника (для подсистемы «CGX Artist» и «CGX Supervisor») |

|  |  |  |  |
| --- | --- | --- | --- |
| Переменная (CGXVariable) | | | |
| Атрибут | В классе | рy-тип | Описание |
| Название | name | str | Данное поле содержит название переменной |
| Значение | value | str | Данное поле содержит значение переменной |
| Состояние | state | bool | Данное поле является флагом активации переменной |
| Описание | description | str | Данное поле содержит описание переменной |
| Категория | category | str | Данное поле содержит категорию переменной (например, «Системная») |

|  |  |  |  |
| --- | --- | --- | --- |
| Программа (CGXProgram) | | | |
| Атрибут | В классе | рy-тип | Описание |
| Название | name | str | Данное поле содержит название программы |
| Исполн. файл | exe | str | Данное поле содержит полный путь до исполняемого файла программы |
| Расширения | extensions | list(str) | Данное поле имеет тип список и хранит расширения файлов, которые может запустить программа. Расширение хранится в виде строки. |
| Аргументы | arguments | list(str) | Данное поле имеет тип список и хранит аргументы запуска программы. Аргумент хранится в виде строки. |

### Исполняющие механизмы

Поскольку различные аспекты системы должны настраиваться извне, было разработано гибкое решение данной задачи. В системе реализованы переменные, которые пользователи сами создают и инициализируют нужными значениями. Это очень схоже с переменными окружения операционной системы. Обработкой этих переменных, поиском ошибок, развертыванием занимается модуль под названием «VariableProcessing» (обработчик переменных). Этот модуль, содержит три класса:

| Класс | Описание |
| --- | --- |
| CGXVariableParser | Данное класс обеспечивает парсинг строк, содержащих переменные. Возможны 3 способа выделения переменных:   * внутренние переменные; * переменные системы; * все переменные.   Также можно проверить строку на наличие тех или иных видов переменных. |
| CGXRegexps | Данный класс является структурой, которая хранить шаблоны регулярных выражений для:   * системных переменных; * внутренних переменных; * все типы переменных.   Данная структура платформонезависимая, и возвращает правильные шаблоны для системных переменных. |
| CGXVariableProcessor | Данный класс является главным в модуле. Отвечает за манипулирование переменными в строках.  **Позволяет:**   * проверить переменную на наличие ошибок; "ScanError" * развернуть переменные в строке; "expandString, expandStringOS" * развернуть значение переменной; "expandVariable, expandVariableOS" * построить графики для строки; "stringToGraphs" * построить граф для переменной; "VariableToGraph"   **Исключения:**  NameError - если встречается недопустимое имя переменной  ReplicationError - если переменная содержит себя (или содержит переменные, которые содержат сами или переменные на уровне выше)  Функция «nameByError» позволяет получить имя переменной по коду ошибки. |

Все остальные исполняющие механизмы имеют типизированную структуру и не требуют рассмотрения. К ним относятся:

* модуль межсетевого взаимодействия (протокол TCP/IP);
* модуль запуска программ;
* модуль взаимодействия с базой данных (SQLite 3.0);

Модули связываются между собой с помощью сигналов и слотов, которые поддерживает фреймворк Qt.

### Вспомогательные классы

Для облегчения написания кода, а также для уменьшения его количества, были разработаны вспомогательные классы-обертки для работы с направленными графами, и операционной системой:

* DiGraphUtil;
* OSUtil.

Класс DigraphUtil обеспечивает полный функционал для работы с направленными графами. Он позволяет создавать граф, манипулировать им, исполнять алгоритмы поиска, а также выводить его структуру в текстовом или графическом виде. Данный класс используется в процессоре переменных (для глубокого парсинга строк, содержащих переменные) и при работе с древовидными структурами. Основан на библиотеке «xnetwork».

Класс OSUtil использует возможности класса DiGraphUtil для сохранения древовидной структуры на жесткий диск. Основными задачами данного класса стали: файловый ввод/вывод в разных форматах, контроль правильности имен директорий и файлов, работа с путями. Данный класс используется во многих частях системы. Основан на библиотеке «os».

### Графический интерфейс приложений

Фреймворк «Qt» позволяет реализовывать графические интерфейсы двумя методами:

* на используемом языке программирования за счет модуля «QtWidgets» (в нашем случае – «Python», а вообще – «С++»);
* на языке сценариев «QML».

Первый метод является более трудоемким в аспекте анимаций и лейаутинга, поэтому не будет рассматриваться.

Для реализации графических интерфейсов приложений системы был выбран второй метод – язык сценарием «QML».

QML – это описательный язык, он описывает, как выглядят и как взаимодействуют друг с другом элементы пользовательского интерфейса. В силу своего описательного характера этот язык не предполагает в себе использования конструкций программирования – например, циклов. И если требуется переместить с одного места на другое какой-либо объект, то этот процесс нужно описать. Впрочем, использовать те же циклы возможно благодаря встроенному в QML языку JavaScript. Для перехода к QML требуется небольшой переворот в сознании, чтобы понять философию создания приложений на QML.

QML – простой, легко осваиваемый и, в то же время, мощный язык программирования, который обладает элегантным синтаксисом. Он очень гибок, и рассчитан на создание пользовательских интерфейсов с использованием анимации. С его помощью можно создавать и воплощать собственные идеи и экспериментировать.

В QML встроен доступ ко всем имеющимся технологиям Qt – таким как, например, Qt/C++, QtwebKit, QtMobility, имеется также и доступ к метаинформации объектов – например, к свойствам, вызываемым методами (invokable methods), декларируемыми при помощи макроса Q\_INVOKABLE и т. п. QML обладает системой версионализации модулей.

Интегрированная среда разработки Qt Creator предоставляет хорошую поддержку этого языка.

Данный метод позволяет полностью отделить графический интерфейс от функциональной части приложения, что дает невероятную гибкость, как и при проектировании, так и при реализации.

Как я понял, ниже будет описание конкретных результатов реализации части системы? Т.е.то железо, что Вы произвели.

Ну и еще одна глава, о которой мы говорили: Описание контрольного примера. Описание действий пользователя и соотверствующих им скриншотов. Что будет проговорено и продемонстрировано в докладе.

То что представлено производит очень приятное впечатление. Для полного представления жду завершающей части и всех приложений.

Обсудим в среду, 6.06, в 11-00.