|  |  |
| --- | --- |
|  | **Автономная некоммерческая организация профессионального образования**  **КАЛИНИНГРАДСКИЙ БИЗНЕС-КОЛЛЕДЖ** |

Кафедра информационных технологий

**Дипломный проект**

**На тему: «Разработка модуля рабочих заданий для инструмента формирования недельно-суточных заданий с внедрением диаграммы Ганта»**

Выполнил студент:

группы 17-ИСП-4

Зуев В.С.

Дипломный руководитель:

Дубинин А.В.

Оценка:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Калининград**

**2020**

# Содержание

[Содержание 2](#_Toc43360568)

[Введение 4](#_Toc43360569)

[1. Анализ проблемы и постановка задачи дипломного проекта 6](#_Toc43360570)

[1.1. Анализ объекта исследования 6](#_Toc43360571)

[1.1.1. Анализ неэффективного управления проектом 6](#_Toc43360572)

[1.1.2. Анализ влияния планирования на проект с целью улучшения его эффективности 8](#_Toc43360573)

[1.2. Постановка задачи дипломного проекта 11](#_Toc43360574)

[1.2.1. Анализ субъекта 11](#_Toc43360575)

[1.2.2. Постановка задачи 14](#_Toc43360576)

[1.3. Выбор языка и инструментария разработки 14](#_Toc43360577)

[1.3.1. Выбор языка 14](#_Toc43360578)

[1.3.2. Выбор инструментария 16](#_Toc43360579)

[2. Анализ требований и определение спецификаций 18](#_Toc43360580)

[2.1. Выбор парадигмы программирования 18](#_Toc43360581)

[2.2. Декомпозиция задачи 18](#_Toc43360582)

[2.3. Диаграмма прецедентов 20](#_Toc43360583)

[2.4. Диаграмма деятельности для расшифровки содержания прецедентов 21](#_Toc43360584)

[2.5. Диаграмма состояний 22](#_Toc43360585)

[2.6. Диаграмма классов 23](#_Toc43360586)

[2.7. Требования к программе 25](#_Toc43360587)

[3. Проектирование программного продукта 26](#_Toc43360588)

[3.1. Архитектура приложения 26](#_Toc43360589)

[3.1.1. Подсистема для работы с рабочими заданиями 26](#_Toc43360590)

[3.1.2. Подсистема команд 26](#_Toc43360591)

[3.1.3. Подсистема пользовательского интерфейса 27](#_Toc43360592)

[3.2. Диаграммы архитектуры приложения 28](#_Toc43360593)

[3.2.1. Диаграмма компонентов 28](#_Toc43360594)

[3.2.2. Диаграмма объектов 28](#_Toc43360595)

[3.2.3. Обзорная диаграмма взаимодействия 29](#_Toc43360596)

[3.2.4. Диаграмма пакетов 30](#_Toc43360597)

[4. Реализация 31](#_Toc43360598)

[4.1. Интерфейс пользователя 31](#_Toc43360599)

# Введение

Информационная система (далее ИС) – это система, предназначенная для хранения, поиска и обработки информации, а также соответствующие организационные ресурсы (человеческие, технические, финансовые и т.д.), обеспечивающие и распространяющие информацию.

Наиболее широко информационные системы и технологии используются в производственной, управленческой и финансовой деятельности. Например, в строительной отрасли.

Основной составляющей строительной отрасли (примерно 90% вложений) является стадия ведения проектов. В первую очередь это касается строительных и компаний-разработчиков, а также компаний, специализирующихся на развитии территорий и создании инфраструктуры строительства. Во многом проекты капитального строительства зданий и сооружений и создания сопутствующей инфраструктуры зависят от общего подхода к процессу. То, насколько качественно компания сможет управлять проектами, их рисками, сроками, бюджетом и многими другими составляющими, и будет влиять на бизнес в целом. Кроме человеческого фактора нельзя не принимать в расчет вспомогательные инструменты - информационные системы для управления строительством. Соответственно такие ИС должны являться интерактивным план-графиком подрядчика, в котором у пользователя будет возможность просматривать запланированные, действующие и выполненные работы, оценивать затраченные и необходимые в дальнейшем ресурсы, а также координировать работу строительных команд и выдавать заказчику требуемую информацию о состоянии строительства на различных его этапах. Однако, сама по себе ИС, работающая на основе таблиц, не так уж и удобна в использовании, поэтому для наглядности отслеживания процессов производства в проектах применяется диаграмма Ганта.

Диаграмма Ганта (a Gantt Chart) — это визуальный способ отображения запланированных задач. Горизонтальные графики широко используются для планирования проектов любых размеров в разных отраслях и сферах. Это удобный способ показать, какая работа планируется к выполнению в определенный день и время. Gantt Charts также помогают командам и менеджерам проектов контролировать даты начала и окончания любого проекта. Все в одном пространстве.

Большинство менеджеров проектов, знают о диаграммах Ганта. Диаграммы Ганта помогают повысить производительность и эффективность рабочих процессов и обеспечить своевременное выполнение задач. Такие графики могут быть применены к небольшим проектам и самым сложным.

Целью данной работы является разработка модуля рабочих заданий для инструмента формирования недельно-суточных заданий с внедрением диаграммы Ганта. Данный модуль будет позволять оптимально планировать затраты ресурсов и времени на выполнение поставленных задач, предоставление быстрого поиска и фильтрации по различным атрибутам работ, создание связей и цепочек работ для составления и отслеживания рабочего процесса.

# Анализ проблемы и постановка задачи дипломного проекта

# Анализ объекта исследования

# Анализ неэффективного управления проектом

Каждое предприятие своей целью ставит получение как можно большего дохода при минимальных затратах (такой подход называется бережливым производством), но добиться этого получается не всегда и зачастую появляются дополнительные расходы из-за различных внешних и внутренних факторов, основным из которых можно выделить недостаточное планирование и неэффективное управление проектом.

Основными проблемами, возникающими при реализации проекта, являются:

* Слабо-налаженные коммуникации между проектными отделами, что приводит к увеличению сроков разработки проекта;
* Нехватка имеющихся ресурсов;
* Конфликты, возникающие при распределении имеющихся ресурсов, если организация занимается осуществлением нескольких проектов одновременно;
* Невыполнение в срок;
* Превышение установленного бюджета;
* Потеря актуальности долгосрочных проектов к моменту их завершения;
* Несоответствие стратегиям организации;
* Недооценка влияния возможных рисков.

Эти проблемы, как последствие, приводят к потерям. Потери – это операции, на которые затрачиваются временные и материальные ресурсы без добавления ценности товару или услуге для конечного потребителя. В соответствии с концепцией бережливого производства в бизнес-процессах возникают следующие виды потерь:

* Неиспользование потенциала сотрудников. (Основополагающий фактор эффективности непрерывного совершенствования –вовлечение в этот процесс всего персонала компании. Знание особенностей сотрудников и грамотное использование их особенностей способно значительно увеличить скорость и качество производства. Так же к этому критерию можно и отнести бездействие сотрудников при отсутствии соответствующей их специализации задачи);
* Потери на транспортировку. (Потери в результате перемещения материалов и заготовок между производственными участками и цехами, не добавляющими ценности конечному продукту. По отношению к строительным проектам, это обычно несвоевременный подвоз материалов, инструментов и оборудования по причине некорректного планирования, что ведет к потерям на обслуживание запасов);
* Потери от брака, излишних отходов и переделок. (Типичными примерами могут служить ошибки в документах и доработки, некорректный ввод данных в информационные системы, несогласованность работы подразделений, дефекты и брак на этапе изготовления и после передачи изделия заказчику);
* Потери на обслуживание запасов. (Потери в результате приобретения сырья и материалов в объеме, превышающем необходимый на текущий плановый период, такие как затраты на складское хранение, убытки в результате ухудшения свойств материалов вследствие их длительного хранения, запасы готовой продукции, сырья, незавершенного производства между технологическими стадиями и на рабочем месте);
* Потери на перемещениях и движениях персонала. (Потери в результате нерациональной организации рабочего пространства, что отражается в виде лишних движений персонала для поиска документов, материалов, инструмента, отсутствия у предметов постоянного места, нерациональной организации рабочих мест);
* Потери от простоев. (Потери в результате простоя персонала или оборудования в ожидании информации, ресурсов, очередной технологической операции. Это могут быть сбои в логистической системе, отсутствие необходимой документации, ожидание указаний руководства по причине отсутствия текущих задач, недостаточный уровень делегирования полномочий);

# Анализ влияния планирования на проект с целью улучшения его эффективности

В настоящее время достаточно активно проявляется интерес к оценке проектной деятельности и эффективности управления проектами. Следовательно, совокупная оценка результатов бизнеса необходима не только в общем, но и по определенным программам и проектам. Таким образом, вопросы обеспечения оптимального проекта, определения долей проектов и их весов в затратах, рентабельности и сроках отвлечения средств на исполнение (в том числе и затраты на кадры) формируют особый подход к оценке эффективности проектного управления.

Наиболее важным фактором эффективного производства и управления проектами является грамотное планирование и сопровождение бизнес-процессов на всех его этапах.

Оптимизация планирования проекта строительства получила значительное внимание в течение последних 20 лет. В результате, множество методов и алгоритмов были разработаны для решения конкретных сценариев или проблем.

Планирование процессов выполнения для строительного проекта является сложной и трудной задачей. Выбор ресурсов (например, труда, основных средств) является наиболее важной частью планирования и должен рассматриваться по-разному, с ограничениями плана и работы, которая будет осуществляться в процессе выполнения данного плана. Так проекты являются уникальными по своей природе, создание графика строительных задач планировщиком, например, следует рассмотреть целый ряд условий, таких как: технологические, организационные методы и ограничения при выполнении, а также от наличия ресурсов для обеспечения того, чтобы потребности клиента и требования в отношении времени, стоимости и качества были выполнены. Ели эти правила не соблюдать, то можно понести ущерб из-за штрафов, прописанных в трудовых и временных условиях договора.

Процессы управления проектами включают в себя:

* Инициацию – принятие решения о начале выполнения проекта;
* Планирование – определение целей и критериев успеха проекта и разработка рабочих схем их достижения;
* Исполнение – координация людей и других ресурсов для выполнения плана;
* Анализ – определение соответствия плана и исполнение проекта и приятие решений о необходимости применения корректирующих воздействий;
* Управление – определение необходимых корректирующих воздействий, их согласование и применение;
* Завершение – формализация выполнения проекта и подведение его к упорядоченному финалу.

Планирование строительства связано с оптимальным уточнением деятельности в течение долгого времени и распределения ресурсов соответственно. Критерий продолжительности выполнения работ, является решающим фактором при учете клиентом оценки и выбором ставки, со сроком проекта. Подрядчики, почти всегда, стремятся свести к минимуму продолжительность выполнения проекта, чтобы получить преимущество при оценке тендерного предложения.



Рис.1.1. Составляющие системы планирования и контроля производства/строительства.

Любая деятельность в целом зачастую имеет зависимость от времени, затрат и ресурсов, которые являются ограниченными. Решение задачи оптимального планирования с учетом или без учета этих ограничений в течении времени меняются. Зависимость в ходе выполнения работ одно из самых основных ограничений, которое существует в строительных проектах. В процессе строительства, деятельность не может начаться, пока все его главные аспекты не будут завершены. Кроме того, время начала каждого вида деятельности не может быть позднее, чем последнего, чтобы закончить проект в рамках требуемого времени. Рабочее пространство всегда ограничено в проекте строительства. Рабочая зона может потребоваться в нескольких различных видах деятельности, в одно и тоже время. Таким образом, чтобы определить, как оптимально управлять рабочей зоной, чтобы облегчить планирование деятельности необходимо влиять на проект производства. Ресурсы являются наиболее влиятельными ограничениями в строительстве, так как они определяют целесообразность применения сроков реализации проекта, а также оптимальны ли они. Сжатое расписание сильно зависит от наличия и количества ресурсов. Ограниченная информация, которая состоит из чертежей, спецификаций, оценки риска и безопасности, разрешений для работы, также имеет значительное влияние на проблему планирования в строительстве. Информационный поток между деятельностью, часто упускается из виду.

# Постановка задачи дипломного проекта

# Анализ субъекта

В ходе дипломной работы необходимо разработать модуль создания и управления рабочими заданиями для десктоп программы инструмента формирования недельно-суточных заданий, который будет предназначен для использования строительным подрядчиком «АтомСтройЭкспорт». Данный программный продукт разрабатывается фирмой «Неолант-Тенакс».

С 2004 года группа компаний «НЕОЛАНТ» (Россия), поставщик № 1 решений для жизненного цикла промышленных объектов, осуществляет комплексную поддержку управления регионами и предприятиями в России и мире и предоставляет услуги по направлениям:

* Разработка и внедрение российских ГИС, САПР и PLM-систем.
* Цифровой инжиниринг: проектирование/сооружение, конструирование/изготовление.
* Разработка государственных и корпоративных информационных систем на заказ: мониторинг, анализ, прогнозирование.
* Информационное и имитационное моделирование территорий и предприятий.

Специалисты ГК «Неолант» с 2012 года активно разрабатывают информационные системы для атомной отрасли, в том числе и систему мониторинга и оперативного управления капитальным строительством, в которую входит инструмент формирования недельно-суточных заданий как приложение-клиент для взаимодействия с системой.

Система мониторинга и оперативного управления капитальным строительством – интернет-портал, предназначенный для поддержки принятия управленческих решений при капитальном строительстве (КС).

Целью создания инструментария является повысить эффективность управления капитальным строительством АЭС, что означает сократить сроки и стоимость, повысить качество проекта сооружения АЭС.

В ходе проекта специалисты «НЕОЛАНТ» и «НИАЭП» совместно разработали методологию мониторинга и оперативного управления капитальным строительством (КС) в атомной отрасли. В её основе – две идеи:

* Формирование системы аналитических показателей, отражающих возможные отклонения фактических показателей процессов сооружения АЭС от плановых. Эта система демонстрирует состояние процессов капитального строительства – проектирование, закупки, поставки, строительно-монтажные и пуско-наладочные работы – в разрезе сроков, стоимости и качества;
* Предоставление участникам проекта КС актуальной информации в реальном времени для поддержки принятия управленческих решений и обеспечения эффективных коммуникаций между всеми участниками проекта.

Далее «НЕОЛАНТ» реализовал ИТ-решение, воплотившее разработанную методологию:

* Для расчета аналитических показателей используется платформа бизнес-аналитики SAS BI. Реализованный на её основе аналитический модуль обеспечивает выявление «узких мест» в процессах сооружения АЭС, поддерживая декомпозицию показателей от сводных значений до исходных данных. Для удобства восприятия и анализа сводные значения отображаются по принципу «светофора» – красный цвет означает критическое отклонение фактических значений от плановых, зеленый – нахождение фактических значений в пределах допустимого отклонения от плана;
* Для доступа к информации по проекту и к информационным системам заказчика создан интернет-портал – единое информационное пространство для совместной и индивидуальной работы участников сооружения АЭС;
* Важнейшей составляющей IT-реализации методологии стала система визуализации. Она позволяет в цифрах, графиках и реалистичных изображениях ознакомиться с актуальной информацией: сравнить фактические показатели с плановыми, проектные 3D модели и сферические панорамы строительной площадки с календарно-сетевыми графиками и проектной документацией. В системе реализовано синхронное отображение инженерных 3D моделей объектов проекта КС – «как спроектировано», с их реальным видом «как построено» на основе сферических панорам. Это позволяет сравнить ситуацию на стройплощадке с запланированной и оперативно оценить ход работ.;
* Для удобства взаимодействия с субподрядными строительно-монтажными организациями в системе реализовано формирование недельно-суточных заданий, а также сбор данных о фактическом состоянии работ с помощью мобильных компьютеров/планшетов. Данные на планшетах синхронизируются с порталом, что обеспечивает участникам проекта получение информации о фактически выполненных работах в режиме реального времени.

# Постановка задачи

Для улучшения качества и эффективности планирования строительной деятельности и сопровождения бизнес-процессов была поставлена цель – разработать модуль рабочих заданий, представляющий собой подробный календарный план, отображающий графически последовательность запланированных, выполняемых и выполненных работ с информацией о расходах, для настольного клиентского приложения – инструмента формирования недельно-суточных заданий.

Создание данного модуля позволит:

* Формировать последовательность работ для каждого рабочего задания;
* Оптимизировать календарный план выполнения работ, что позволит улучшить эффективность проекта;
* Производить мониторинг выполнения работ и отслеживать все бизнес-процессы, связанные со строительством.

# Выбор языка и инструментария разработки

# Выбор языка

**C#** (произносится *си шарп*) — объектно-ориентированный язык программирования. Разработан в 1998—2001 годах группой инженеров компании Microsoft под руководством Андерса Хейлсберга и Скотта Вильтаумота как язык разработки приложений для платформы Microsoft .NET Framework. Впоследствии был стандартизирован как ECMA-334 и ISO/IEC 23270.

C# относится к семье языков с C-подобным синтаксисом, из них его синтаксис наиболее близок к C++ и Java. Язык имеет статическую типизацию, поддерживает полиморфизм, перегрузку операторов (в том числе операторов явного и неявного приведения типа), делегаты, атрибуты, события, свойства, обобщённые типы и методы, итераторы, анонимны функции с поддержкой замыканий, LINQ, исключения, комментарии в формате XML.

Переняв многое от своих предшественников — языков C++,  Delphi, Модула, Smalltalk и, в особенности, Java — С#, опираясь на практику их использования, исключает некоторые модели, зарекомендовавшие себя как проблематичные при разработке программных систем, например, C# в отличие от C++ не поддерживает множественное наследование классов (между тем допускается множественное наследование интерфейсов).

C# разрабатывался как язык программирования прикладного уровня для CLR и, как таковой, зависит, прежде всего, от возможностей самой CLR. Это касается, прежде всего, системы типов C#, которая отражает BCL. Присутствие или отсутствие тех или иных выразительных особенностей языка диктуется тем, может ли конкретная языковая особенность быть транслирована в соответствующие конструкции CLR. Так, с развитием CLR от версии 1.1 к 2.0 значительно обогатился и сам C#; подобного взаимодействия следует ожидать и в дальнейшем (однако, эта закономерность была нарушена с выходом C# 3.0, представляющего собой расширения языка, не опирающиеся на расширения платформы .NET). CLR предоставляет C#, как и всем другим .NET-ориентированным языкам, многие возможности, которых лишены «классические» языки программирования. Например, сборка мусора не реализована в самом C#, а производится CLR для программ, написанных на C# точно так же, как это делается для программ на VB.NET, J# и др.

Мною был выбран этот язык по 3 факторам:

1. Скорость разработки - синтаксис c# позволяет быстро создавать сложные механизмы при разработке
2. Технология WPF – разработанная для c# является одной из лучших технологий для создания настольных программ
3. Компилируемый язык – c# перенял скорость и мощность c++ и почти ей не уступает что позволяет совершать миллионы операций в секунду.

# Выбор инструментария

**Microsoft Visual Studio** — линейка продуктов компании Microsoft, включающих интегрированную среду разработки программного обеспечения и ряд других инструментальных средств. Данные продукты позволяют разрабатывать как консольные приложения, так и приложения с графическим интерфейсом, в том числе с поддержкой технологии Windows Forms, а также веб-сайты, веб-приложения, веб-службы как в родном, так и в управляемом кодах для всех платформ, поддерживаемых Windows, Windows Mobile, Windows CE, .NET Framework, Xbox, Windows Phone .NET Compact Framework и Silverlight.

Visual Studio включает в себя редактор исходного кода с поддержкой технологии IntelliSense и возможностью простейшего рефакторинга кода. Встроенный отладчик может работать как отладчик уровня исходного кода, так и отладчик машинного уровня. Остальные встраиваемые инструменты включают в себя редактор форм для упрощения создания графического интерфейса приложения, веб-редактор, дизайнер классов и дизайнер схемы базы данных.

Visual studio был выбран учитывая 2 фактора:

1. Это единственный инструмент где присутствует технология WPF и ее графический дизайнер
2. Это мощный инструмент с множеством функций и удобством разработки приложений на .NET платформе

# Анализ требований и определение спецификаций

# Выбор парадигмы программирования

При выборе между функциональным и объектно-ориентированным подходом были рассмотрены такие факторы как:

1. Объектно-ориентированное программирование (ООП) является более «традиционной» парадигмой. С её помощью разработано несчётное количество программ, в том числе огромные промышленные системы в финансовых организациях, телекоммуникации на производстве, складах, транспорте.
2. Функциональное программирование может, как облегчить, так и усложнить разработку ПО, особенно если программа не большая, ввиду того что требуется больше времени за удержанием всех функций чистыми и написанием высших функций.

Поэтому был выбран объектно-ориентированный подход с небольшим содержанием функционального программирования в виде чистых функций.

# Декомпозиция задачи

Основными частями будут являться интерфейс и логика программы. Для интерфейсной части возникают задачи как:

1. Проработка общего стиля программы – шрифтов, цветовой гаммы, элементов управления;
2. Реализация стандартов качества интерфейса, например, возможность перетаскивать работы влево/вправо по временной шкале на диаграмме Ганта, а не только при изменении дат старта и финиша работ, добавление всплывающих подсказок с актуальной информацией по работам;

Для логики программы стоят задачи:

1. Реализовать визуализацию загруженных рабочих заданий;
2. Реализовать создание новых рабочих заданий;
3. Реализовать создание цепочек последовательностей работ с различными типами связей;
4. Реализовать создание ограничений для старта работ («жесткий старт», «старт не раньше»)
5. Реализовать алгоритм выстраивания работ согласно их типам связей;
6. Реализовать поиск, сортировку и фильтрацию работ по их атрибутам;

# Диаграмма прецедентов

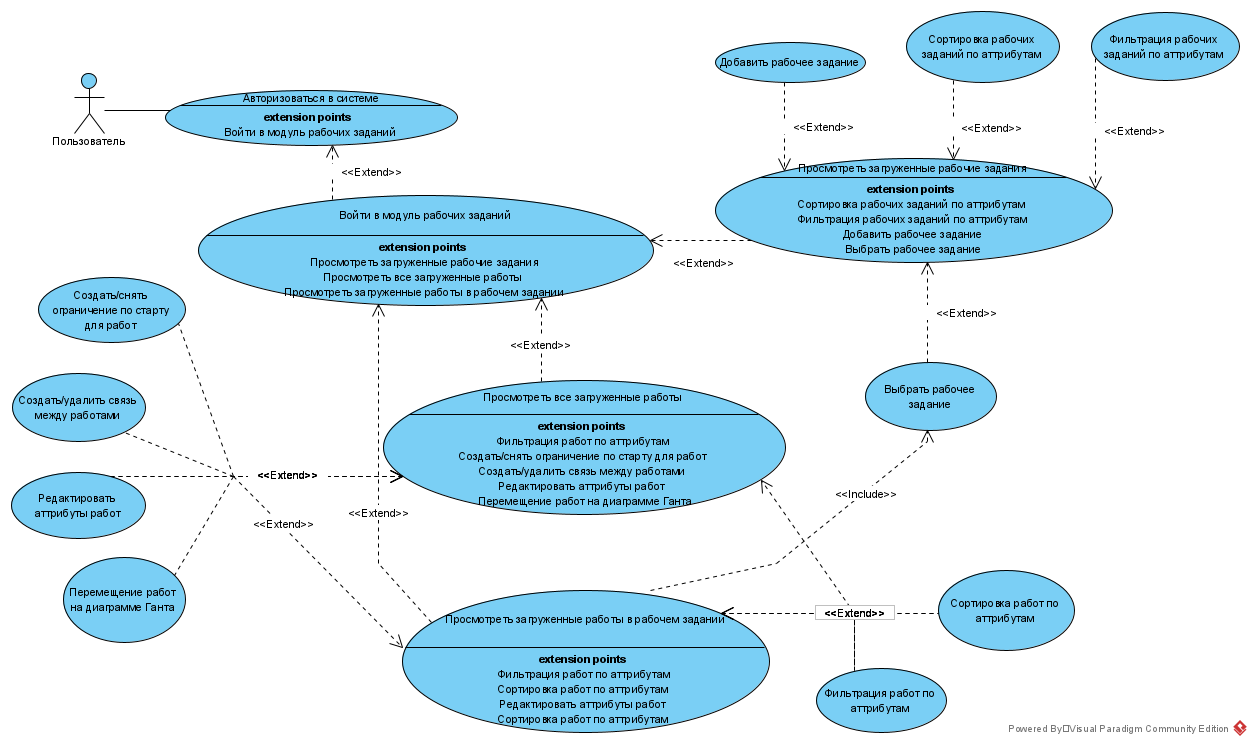


Рис. 2.1. Диаграмма прецедентов.

# Диаграмма деятельности для расшифровки содержания прецедентов

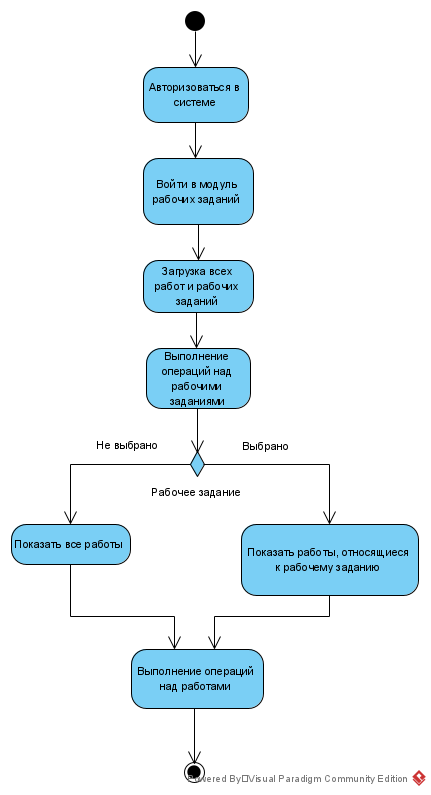
****

Рис. 2.2. Диаграмма деятельности.

# Диаграмма состояний

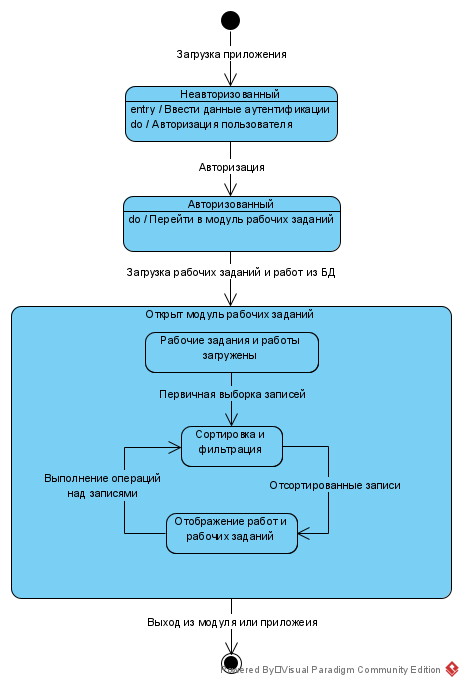


Рис. 2.4. Диаграмма состояний.

# Диаграмма классов

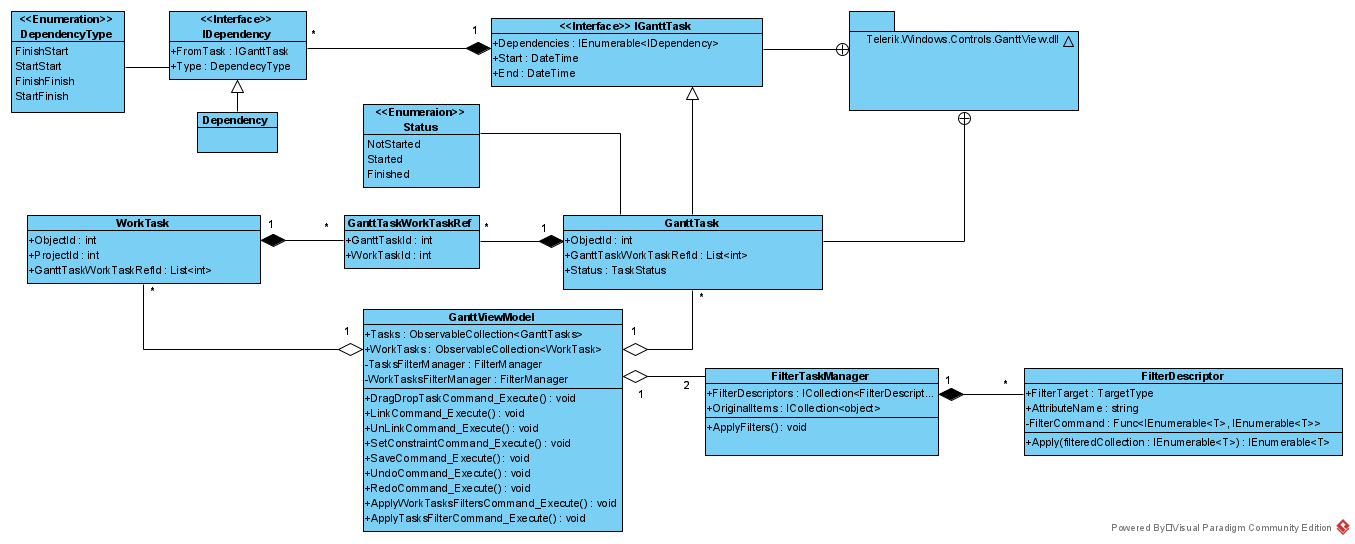


Рис. 2.5. Диаграмма классов.Диаграмма последовательностей

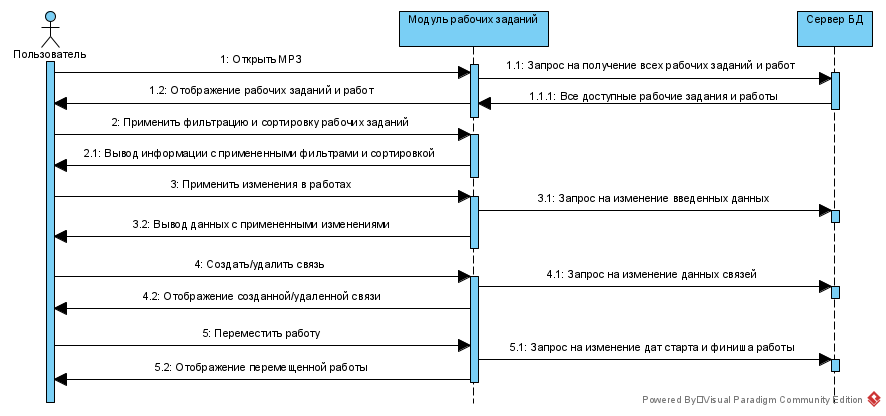


Рис. 2.5. Диаграмма последовательностей.

# Требования к программе

Разрабатываемый модуль должен удовлетворять следующим требованиям:

* Просматривать и редактировать все загруженные рабочие задания;
* Создавать новые рабочие задания;
* Просматривать все загруженные работы;
* Добавлять/удалять работы в рабочих заданиях;
* Создавать/удалять связи между работами;
* Редактировать доступные поля работ;
* Перемещать работы на диаграмме Ганта;
* Создавать ограничения по старту для работ;
* Корректно сохранять все изменения в БД.

# Проектирование программного продукта

# Архитектура приложения

Приложение «Инструмент формирования недельно-суточных заданий» состоит из 10 модулей, одним из которых и является разрабатываемый модуль рабочих заданий, который состоит из следующих подсистем:

* Основная подсистема для создания и редактирования рабочих заданий, сортировки и фильтрации и внесения изменений в атрибуты работ;
* Подсистема команд для отслеживания изменений, их отмены и сохранения в базу данных;
* Подсистема пользовательского интерфейса.

# Подсистема для работы с рабочими заданиями

Основная подсистема модуля, отвечающая за вычисление и взаимопередачу данных между пользовательским интерфейсом и базой данных. Подсистема для работы с рабочими заданиями является логической моделью всего, что видит на экране пользователь. Именно здесь формируются все команды от пользователя на изменение данных, которые в дальнейшем передаются в подсистему команд.

Основная задача – это формирование и редактирование списков рабочих заданий и работ для текущего комплексного сетевого графика.

# Подсистема команд

В связи с тем, что все элементы управления в пользовательском интерфейсе настроены на использование команд, то их функционал в проекте расширен глобально для возможности контролировать все изменения, совершаемые пользователем в процессе работы с приложением и модулем рабочих заданий в частности.

Данная подсистема включает в себя такие операции как «Undo» (отмена последнего в очереди изменения), «Redo» (восстановление отмененного в очереди изменения) и «Save» (сохранение текущих изменений). Так как некоторые действия пользователя могут неочевидно повлиять на зависимые параметры, то эти операции помогают ему сократить время на корректный ввод данных.

# Подсистема пользовательского интерфейса

Эта подсистема займет большое количество времени разработки, однако, именно она и будет определять удобство работы с программой. Будет использована технология WPF совместно с пакетом услуг Telerik. Эти технологии помогут в наименьший срок разработать дружелюбный, многофункциональный, масштабируемый интерфейс. Основное назначение этой подсистемы будет заключаться в предоставлении посредника между пользователем и логикой работы программы.

Основной задачей будет являться определение общего стиля интерфейса и возможность реализовать все функции других подсистем.

Преимущественной сложностью предвидится поддержание принципов паттерна MVVM.

# Диаграммы архитектуры приложения

# Диаграмма компонентов

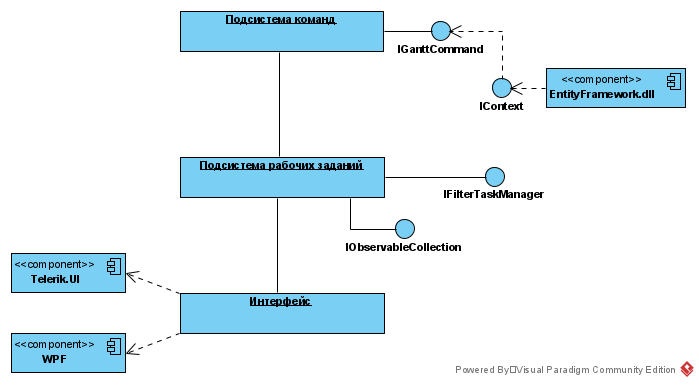


Рис. 3.1. Диаграмма компонентов.

# Диаграмма объектов

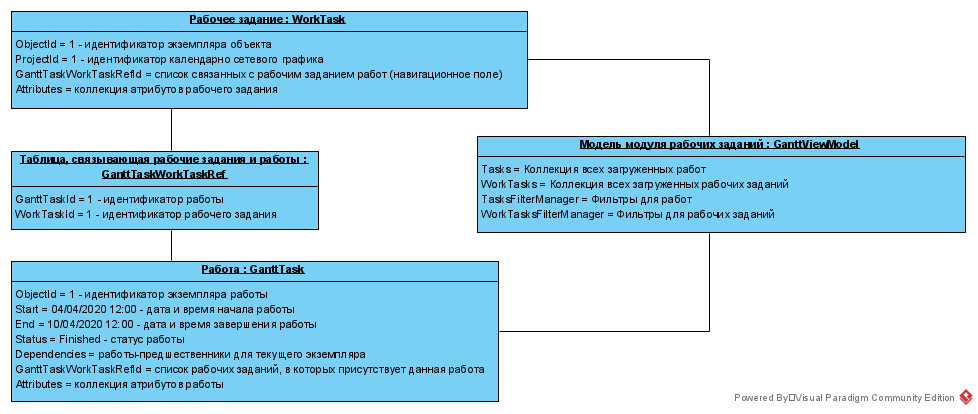


Рис. 3.2. Диаграмма объектов.

# Обзорная диаграмма взаимодействия

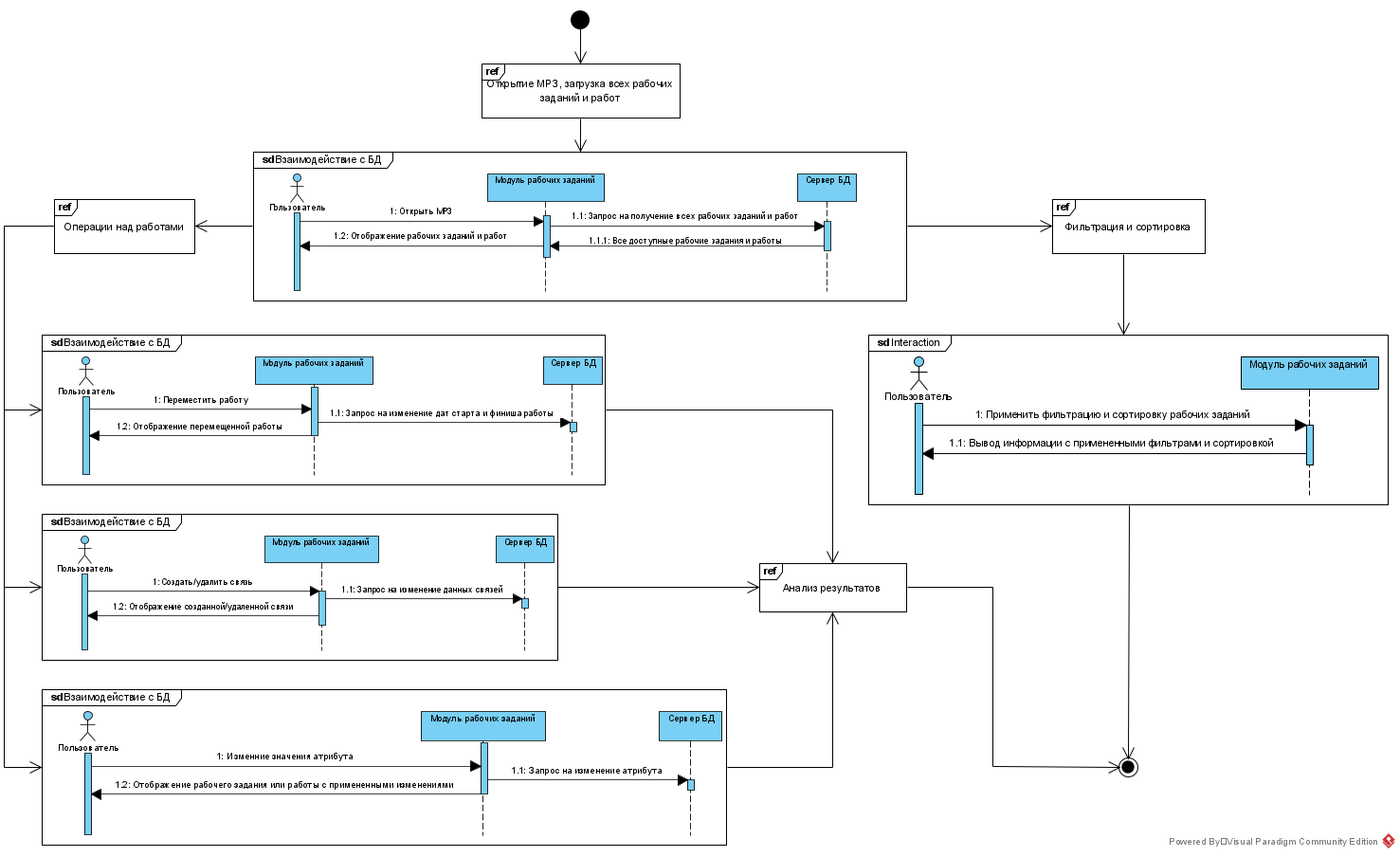


Рис. 3.4. Обзорная диаграмма взаимодействия

# Диаграмма пакетов

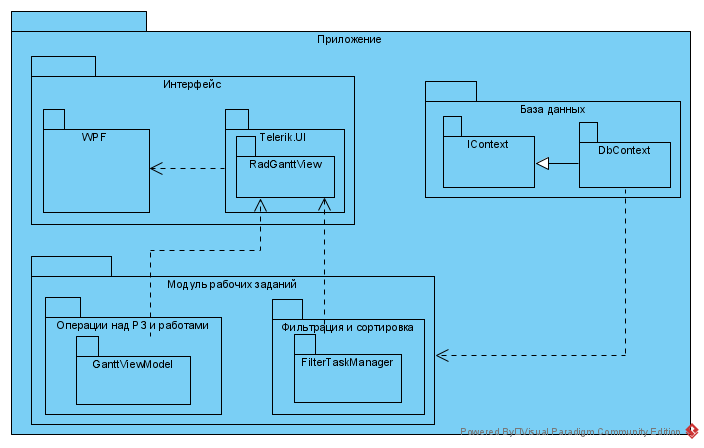


Рис. 3.5. Диаграмма пакетов

# Реализация

# Интерфейс пользователя