# Реферат

Курсовая работа: 41 страниц, 16 рисунков, 8 используемых источников.

Ключевые слова: ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ, ПРОЕКТИРОВАНИЕ, ПОСЛЕДОВАТЕЛЬНОСТИ, МОДЕЛЬ, КЛАСС, ТЕЛЕФОН, АТС, UML, BPMN, ГАНТ, EPC, FURPS+ , IDEF0, DFD, ДИАГРАММЫ.

Объектом исследования является программное обеспечение микропроцессора Телефона, который способен производить вызов абонента с помощью терминала.

Цель работы состоит в разработке проекта программного обеспечения «Телефон» с использованием диаграмм разного вида, в полной мере описывающих как внутреннее устройство исследуемой системы, так и всевозможные взаимодействия между её компонентами.

В результате были получены диаграммы, обладающие исчерпывающей информацией о программном обеспечение Телефона. К ним относятся: диаграмма Ганта, UML-диаграмма, IDEF0-диаграмма, DFD-диаграмма, EPC-диаграмма, BPMN «As-Is» и BPMN «To be», документ FURPS+.

# Содержание

[**Реферат 2**](#_Toc65460394)

[**Содержание 3**](#_Toc65460395)

[**Введение 4**](#_Toc65460396)

[**1. Формулировка задачи 5**](#_Toc65460397)

[**2. Диаграмма Ганта 7**](#_Toc65460398)

[**3. Семейства моделей IDEF 8**](#_Toc65460399)

[**3.1. Стандарты семейства IDEF 8**](#_Toc65460400)

[**3.2. Создание модели As-Is в стандарте IDEF0 9**](#_Toc65460401)

[**3.3. Модель IDEF1X 13**](#_Toc65460405)

[**3.4. Модель IDEF3 14**](#_Toc65460406)

[**4. Диаграмма потоков данных (DFD) 16**](#_Toc65460407)

[**5. UML 16**](#_Toc65460408)

[**6. EPC 23**](#_Toc65460409)

[**7. BPMN 25**](#_Toc65460410)

[**8. FURPS+ 26**](#_Toc65460411)

[**9. Результаты машинного тестирования программы 28**](#_Toc65460412)

[**10. Системные требования 30**](#_Toc65460413)

[**Заключение 32**](#_Toc65460414)

[**Приложение А – Проверка на антиплагиат 34**](#_Toc65460415)

[**Приложение Б – Диаграмма Ганта 35**](#_Toc65460416)

[**Приложение В – Листинг программы 36**](#_Toc65460417)

# Введение

В настоящее время сотовые телефоны служат одним из основных средств связи и работы на расстоянии, но их предшественниками были стационарные телефоны, работающие через АТС. Простота и удобство использования последних моделей телефонов достигнуты благодаря развитым технологиям, по сравнению со старыми громоздкими и телефонами, связь которых осуществляет АТС.

При создании новых моделей телефона сначала рассматривается тестовая модель или макет. Таким образом, определяются такие параметры как: удобство использования, надежность, производительность, качество звука, качество графики и т.д.

Данный проект – это макет телефона, в котором реализована функция звонка.

1. **Формулировка задачи**

Программное обеспечение встроенного микропроцессора для аппарата учрежденческой телефонной сети.

Аппарат подключен к линии связи, ведущей к мини-АТС. В его задачу входит прием и передача сигналов (в том числе и голосовых данных) мини-АТС. Аппарат имеет кнопочную панель управления, экран для отображения набираемых номеров, звонок и трубку, в которую встроены микрофон и громкоговоритель.

В начальном состоянии трубка телефона повешена, телефон не реагирует на нажатия кнопок. Телефон реагирует только на сигнал «Вызов» от мини-АТС, при этом включается звонок.

При снятии трубки на АТС подается сигнал «Трубка». При получении ответного сигнала «Тон» от АТС телефон воспроизводит звуковой тон « Готов» (длинный непрекращающийся гудок) в трубку. При получении сигнала «Занято», в трубке воспроизводится тон «Занято» (частые короткие гудки).

Пользователь, слыша в трубке тон «Готов», набирает трехзначный номер. Номер может быть набран при помощи кнопок с цифрами или нажатием на специальную кнопку « # ». При нажатии на кнопку с цифрой соответствующий ей сигнал «Цифра» передается АТС. Нажатия на кнопки с цифрами после третьего игнорируются. Во время набора номера введенные цифры отображаются на экране. Последний полностью набранный номер запоминается в памяти аппарата для того, чтобы можно было его воспроизвести при нажатии на кнопку « # ». При нажатии на эту кнопку номер из памяти аппарата высвечивается на экране, и АТС передается последовательность из трех сигналов «Цифра». В ответ на набранный номер от АТС приходит либо

сигнал «Занято», либо сигнал «Вызов». При получении сигнала «Вызов» телефон воспроизводит в трубку длинные гудки до того момента, когда АТС осуществит коммутацию и передаст сигнал «Данные».

Телефон воспроизводит данные, передаваемые с сигналом, в трубку. Ответ пользователя воспринимается микрофоном трубки, преобразуется в сигнал «Данные» и передается АТС. Обмен данными прерывается, если повешена трубка одного из телефонов, участвующих в обмене.

О том, что трубку повесил вызываемый абонент, сообщает сигнал «Занято», посылаемый АТС. После того, как трубка аппарата была повешена, телефон посылает АТС сигнал «Конец», и телефон переходит в начальное состояние.

# Диаграмма Ганта

Упростить планирование проекта со множеством задач поможет составление диаграммы Ганта. На ней будет виден не только список задач, но и кто, что и когда должен сделать.

Диаграмма Ганта — это горизонтальная столбчатая диаграмма с временной шкалой, которая используется для иллюстрации плана работ по проекту с привязкой ко времени. Ее придумал американский инженер Генри Гант в 1910 году, и в течение 20 века она широко использовалась для планирования проектов. Задачам, перечисленным слева от диаграммы, соответствуют ленты, ориентированные вдоль оси времени. Эти ленты изображают рабочий процесс в проекте. Диаграммы Ганта обычно содержат даты начала и завершения задач, контрольные точки, зависимости между задачами и исполнителей. Современное ПО развивает первоначальную идею:

* оно позволяет скрывать и показывать список задач;
* отображать критический путь;
* отслеживать индикаторы выполнения;
* работать с панелями;
* управления ресурсами и многое другое.

Ключевым понятием диаграммы Ганта является «веха» — метка значимого момента в ходе выполнения работ, общая граница двух или более задач. Вехи позволяют наглядно отобразить необходимость синхронизации, последовательности в выполнении различных работ. Вехи, как и другие границы на диаграмме, не являются календарными датами. Сдвиг вехи приводит к сдвигу всего проекта. Поэтому диаграмма Ганта не является, строго говоря, графиком работ. Кроме того, диаграмма Ганта не отображает значимости или ресурсоемкости работ, не отображает сущности работ (области действия). Для крупных проектов диаграмма Ганта становится чрезмерно тяжеловесной и теряет всякую наглядность.»

Диаграмма Ганта для проекта «Телефон» находится в «Приложении Б».

1. **Семейства моделей IDEF** 
   1. **Стандарты семейства IDEF**

В настоящий момент к семейству IDEF можно отнести следующие стандарты:

* IDEF0 - методология функционального моделирования. С помощью наглядного графического языка IDEF0, изучаемая система предстает перед разработчиками и аналитиками в виде набора взаимосвязанных функций (функциональных блоков - в терминах IDEF0). Как правило, моделирование средствами IDEF0 является первым этапом изучения любой системы;
* IDEF1 – методология моделирования информационных потоков внутри системы, позволяющая отображать и анализировать их структуру и взаимосвязи;
* IDEF1X (IDEF1 Extended) – методология построения реляционных структур. IDEF1X относится к типу методологий “Сущность-взаимосвязь” (ER – Entity-Relationship) и, как правило, используется для моделирования реляционных баз данных, имеющих отношение к рассматриваемой системе;
* IDEF2 – методология динамического моделирования развития систем. В связи с весьма серьезными сложностями анализа динамических систем от этого стандарта практически отказались, и его развитие приостановилось на самом начальном этапе. Однако в настоящее время присутствуют алгоритмы и их компьютерные реализации, позволяющие превращать набор статических диаграмм IDEF0 в динамические модели, построенные на базе “раскрашенных сетей Петри” (CPN – Color Petri Nets);
* IDEF3 – методология документирования процессов, происходящих в системе, которая используется, например, при исследовании технологических процессов на предприятиях. С помощью IDEF3 описываются сценарий и последовательность операций для каждого процесса. IDEF3 имеет прямую взаимосвязь с методологией IDEF0 – каждая функция (функциональный блок) может быть представлена в виде отдельного процесса средствами IDEF3;
* IDEF4 – методология построения объектно-ориентированных систем. Средства IDEF4 позволяют наглядно отображать структуру объектов и заложенные принципы их взаимодействия, тем самым позволяя анализировать и оптимизировать сложные объектно-ориентированные системы;
* IDEF5 – методология онтологического исследования сложных систем. С помощью методологии IDEF5 онтология системы может быть описана при помощи определенного словаря терминов и правил, на основании которых могут быть сформированы достоверные утверждения о состоянии рассматриваемой системы в некоторый момент времени. На основе этих утверждений формируются выводы о дальнейшем развитии системы и производится её оптимизация.
  1. **Создание модели As-Is в стандарте IDEF0**

Прежде чем пытаться выбрать существующую или создать собственную информационную систему, а затем внедрить ее, необходимо проанализировать, как работает система в настоящее время. Для этого строится функциональная модель AS-IS. Анализ этой функциональной модели позволяет понять, где находятся наиболее слабые места, в чем будут состоять преимущества новых бизнес-процессов. Детализация бизнес-процессов позволяет выявить недостатки. Найденные в модели AS-IS недостатки можно исправить при создании модели ТО-ВЕ - модели новой организации бизнес-процессов. Модель ТО-ВЕ нужна для оценки последствий внедрения информационной системы и анализа альтернативных путей выполнения работы и документирования того, как система будет функционировать в будущем.

IDEF0 – это графическая нотация, предназначенная для описания бизнес-процессов. Система, описываемая в данной нотации, проходит через декомпозицию или, иными словами, разбиение на взаимосвязанные функции. Для каждой функции существует правило сторон:

– стрелкой слева обозначаются ресурсы;

– стрелкой сверху обозначаются контролирующие нормы;

– стрелкой справа обозначает результат работы;

– стрелкой снизу обозначает механизм, с помощью которого осуществлялась работа.

Учитывая всё вышеперечисленное на рисунке 1 была составлена модель As-Is проекта «Телефон».

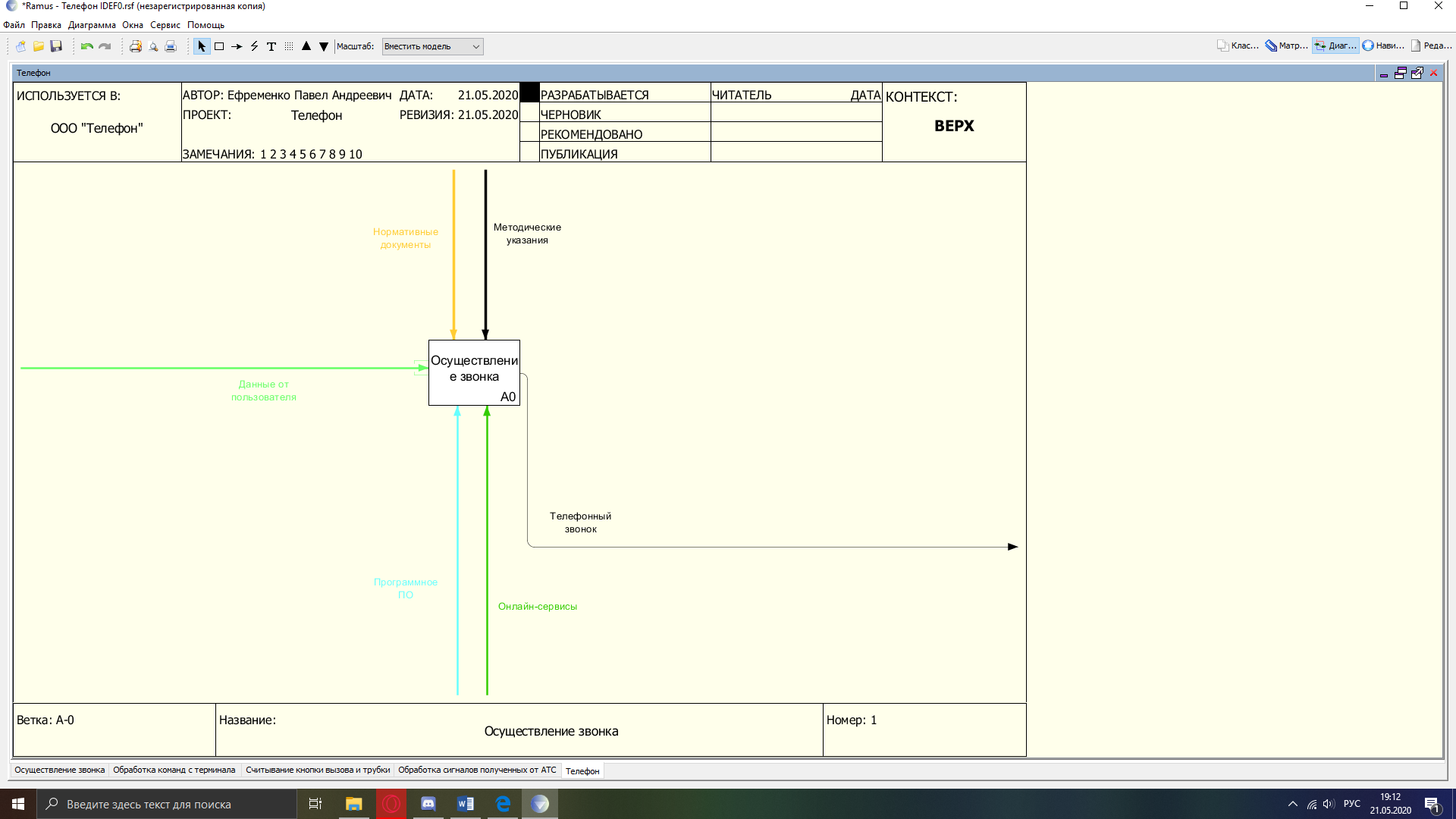


Рисунок 1 – Модель As-Is проекта «Телефон»

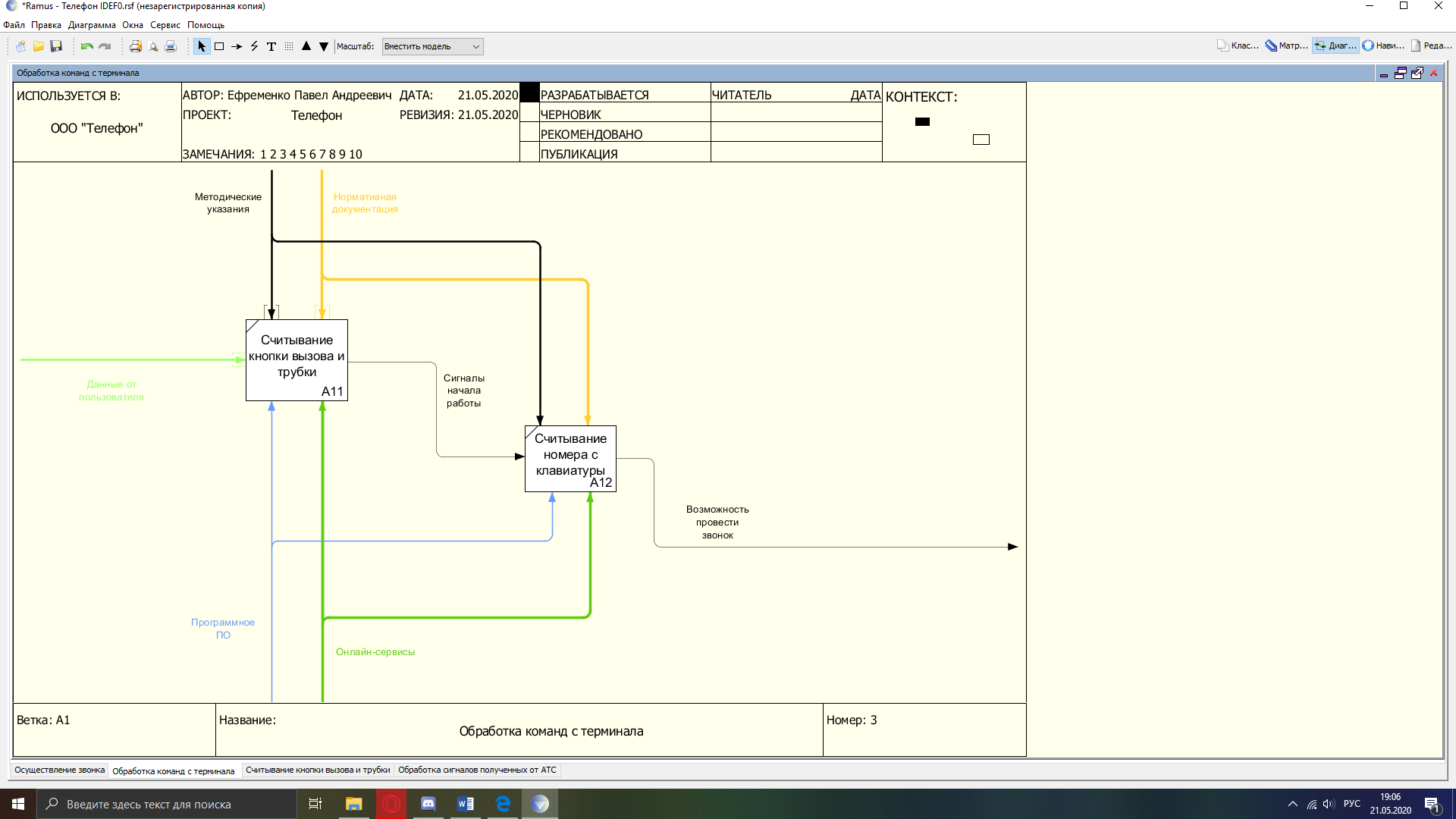


Рисунок 2 – Декомпозиция проекта «Телефон»

Полученная модель системы может быть представлена в более подробном виде путём разбиения на большее количество составных элементов.

На рисунке 3 и 4 можно видеть модель кода «Телефон» после декомпозиции.

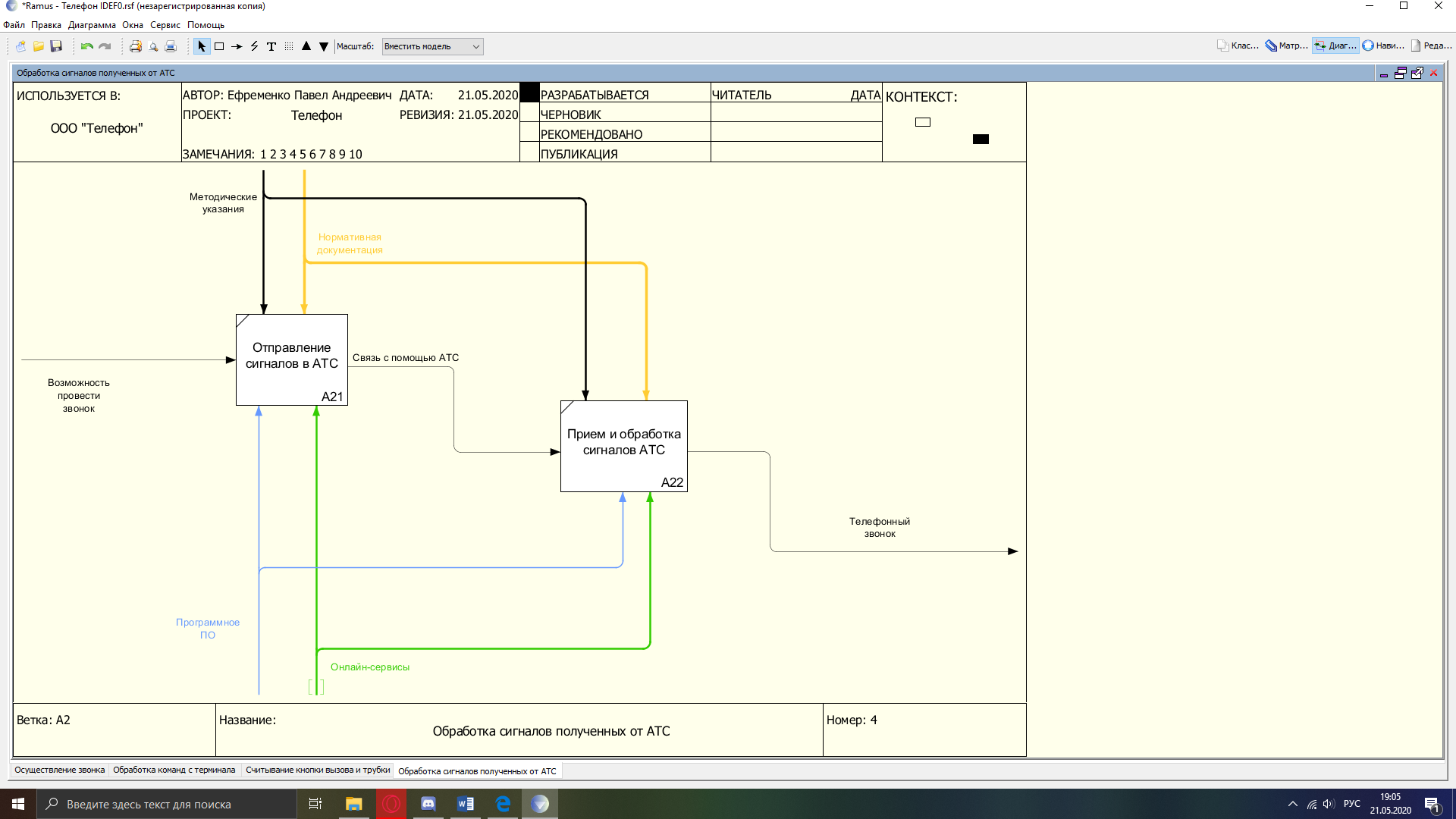


Рисунок 3 – Декомпозиция кода обработки сигналов от АТС.

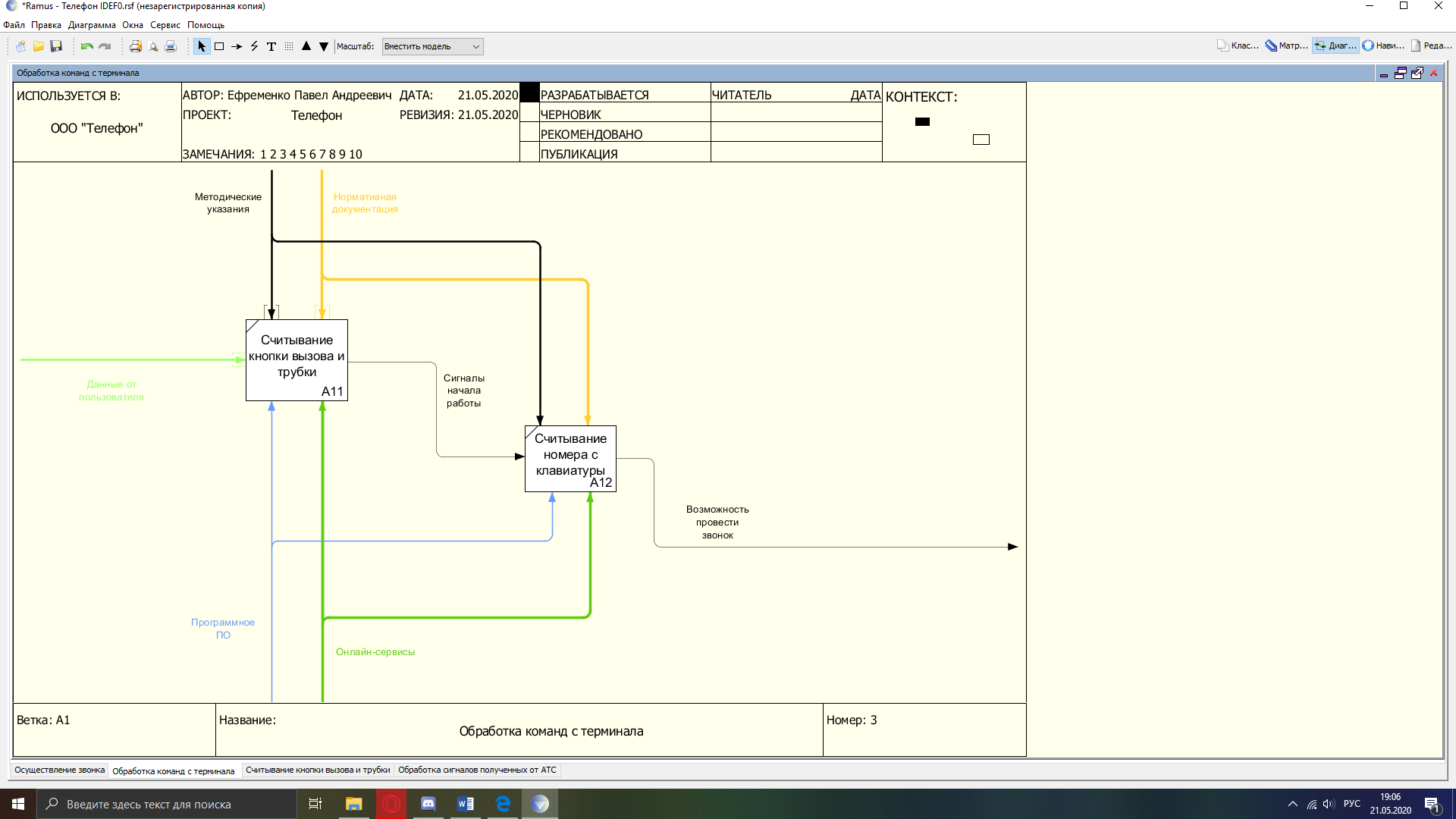


Рисунок 4 – Декомпозиция кода

На вход поступают данные с терминала, которые вводятся с терминала и обрабатываются телефоном. Эти данные служат командами к началу работы, а также номером абонента с которым необходимо связаться.

На выходе мы получаем сам звонок пользователя абоненту. Сигналы необходимые для связи регулирует АТС.

Таким образом работа, рассматриваемой системы разбивается на две основанных функции:

– обработка команд с терминала;

– обработка сигналов АТС;

1. 3. **Модель IDEF1X**

Данный метод моделирования используется для моделирования данных стандартным, последовательным и предсказуемым образом, чтобы управлять ими как ресурсом. Он может быть использован в проектах, требующих стандартных средств определения и анализа ресурсов данных внутри организации. Такие проекты включают в себя включение метода моделирования данных в методологию, управление данными как ресурсом, интеграцию информационных систем или проектирование компьютерных баз данных. Основные цели стандарта IDEF1X заключаются в том, чтобы предоставить:

* Средство для полного понимания и анализа информационных ресурсов организаций
* Общее средство представления и передачи сложности данных
* Методики общего представления данных, необходимых для работы предприятия
* Средства для определения независимого от приложения представления данных, которое может быть проверено пользователями и преобразовано в физический дизайн базы данных
* Методики получения интегрированного определения данных из существующих ресурсов данных.

Основной целью IDEF1X является поддержка интеграции. Подход к интеграции сфокусирован на захвате, управлении и использовании единого семантического определения ресурса данных, называемого "концептуальной схемой". "Концептуальная схема" обеспечивает единое интегрированное определение данных внутри предприятия, которое не предвзято по отношению к какому-либо одному применению данных и не зависит от того, как физически эти данные хранятся или доступны. Основная цель концептуальной схемы состоит в том, чтобы обеспечить последовательное определение значений и взаимосвязей между данными, которые могут быть использованы для интеграции, совместного использования и управления целостностью данных.

Пример данной модели можно увидеть на рисунке 5.

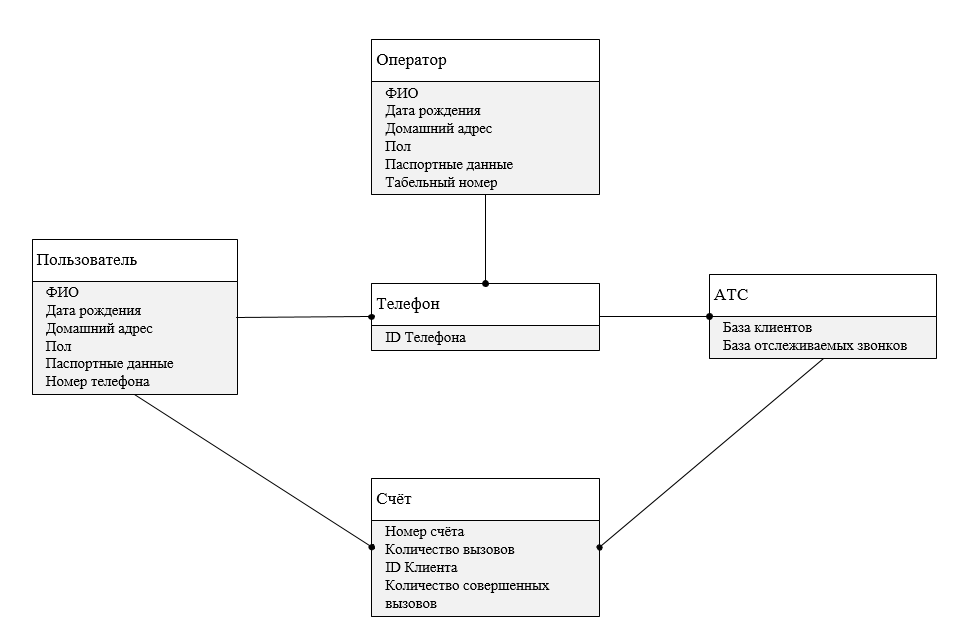


Рисунок 5 – Пример IDEF1X модели

* 1. **Модель IDEF3**

IDEF3 (англ. Integrated DEFinition for Process Description Capture Method) — методология моделирования и стандарт документирования процессов, происходящих в системе. Метод документирования технологических процессов представляет собой механизм документирования и сбора информации о процессах. IDEF3 показывает причинно-следственные связи между ситуациями и событиями в понятной эксперту форме, используя структурный метод выражения знаний о том, как функционирует система, процесс или предприятие.

IDEF3 широко применяется при разработке информационных систем. При этом используется инструмент визуального моделирования бизнес-процессов. Пример данной диаграммы находится на рисунке 6.

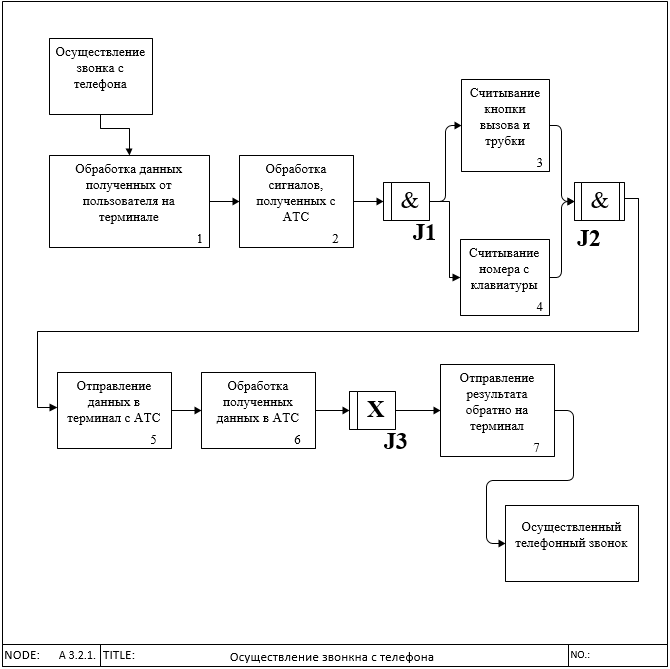


Рисунок 6 – Пример IDEF3 модели

1. **Диаграмма потоков данных (DFD)**

Диаграмма потоков данных DFD (DataFlowDiagrams) – «это методология графического структурного анализа, описывающая внешние по отношению к системе источники и адресаты данных, логические функции, потоки данных и хранилища данных, к которым осуществляется доступ. Диаграмма DFD – это один из основных инструментов структурного анализа и проектирования информационных систем, существовавших до широкого распространения UML.»

В результате декомпозиции системы «Телефон» была получена следующая диаграмма DFD (рис. 5).

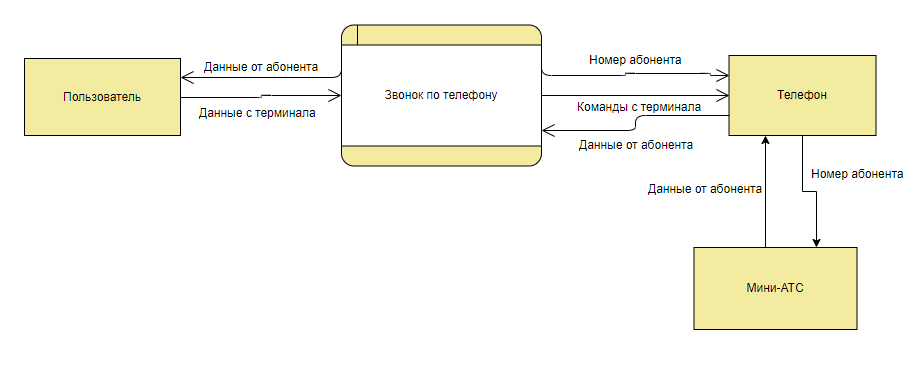


Рисунок 7 – Диаграмма DFD-системы «Телефон»

Внешними сущностями данной системы являются: пользователь, Телефон, Мини-АТС. Система не содержит баз данных.

1. **UML**

UML — язык графического описания для объектного моделирования в области разработки программного обеспечения, для моделирования бизнес-процессов, системного проектирования и отображения организационных структур.

UML является языком широкого профиля, это — открытый стандарт, использующий графические обозначения для создания абстрактной модели системы, называемой UML-моделью. UML был создан для определения, визуализации, проектирования и документирования, в основном, программных систем. UML не является языком программирования, но на основании UML-моделей возможна генерация кода.»

Преимущества UML:

UML объектно-ориентирован, в результате чего методы описания результатов анализа и проектирования семантически близки к методам программирования на современных объектно-ориентированных языках;

UML позволяет описать систему практически со всех возможных точек зрения и разные аспекты поведения системы;

Диаграммы UML сравнительно просты для чтения после достаточно быстрого ознакомления с его синтаксисом;

UML расширяет и позволяет вводить собственные текстовые и графические стереотипы, что способствует его применению не только в сфере программной инженерии;

UML получил широкое распространение и динамично развивается.

Давайте рассмотрим UML Диаграммы:

* + - Диаграмма классов – структурная диаграмма языка моделирования UML, демонстрирующая общую структуру иерархии классов системы, их коопераций, атрибутов (полей), методов, интерфейсов и взаимосвязей между ними. Пример представлен на рисунке 9.
    - Диаграмма компонентов – элемент языка моделирования UML, статическая структурная диаграмма, которая показывает разбиение программной системы на структурные компоненты и связи (зависимости) между компонентами. Пример представлен на рисунке
    - Диаграмма композитной/составной структуры – статическая структурная диаграмма, демонстрирует внутреннюю структуру классов и, по возможности, взаимодействие элементов (частей) внутренней структуры класса.
    - Диаграмма развёртывания – диаграмма, которая моделирует физическое развертывание артефактов на узлах.
    - Диаграмма объектов – диаграмма, которая предназначена для демонстрации совокупности моделируемых объектов и связей между ними в фиксированный момент времени.
    - Диаграмма пакетов – диаграмма, которая отображают зависимости между пакетами, составляющими модель. Пример представлен на рисунке
    - Диаграмма деятельности – диаграмма, на которой показаны действия, состояния которых описано на диаграмме состояний.
    - Диаграмма вариантов использования – диаграмма описывает взаимоотношения и зависимости между группами вариантов использования и действующих лиц, участвующими в процессе. Пример представлен на рисунке 8.
    - Диаграммы коммуникации и последовательности – диаграмма, на которой для некоторого набора объектов на единой временной оси показан жизненный цикл объекта (создание-деятельность-уничтожение некой сущности) и взаимодействие актеров (действующих лиц) информационной системы.
    - Диаграмма состояний – диаграмма, которая показывает, как объект переходит из одного состояния в другое. Диаграммы состояний служат для моделирования динамических аспектов системы. Пример представлен на рисунке

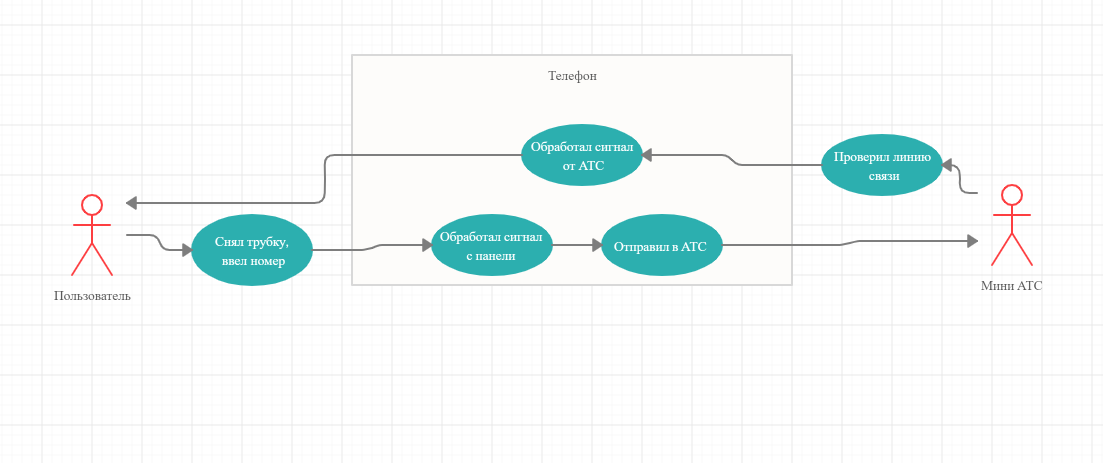


Рисунок 8 – UML-диаграмма системы «Телефон»

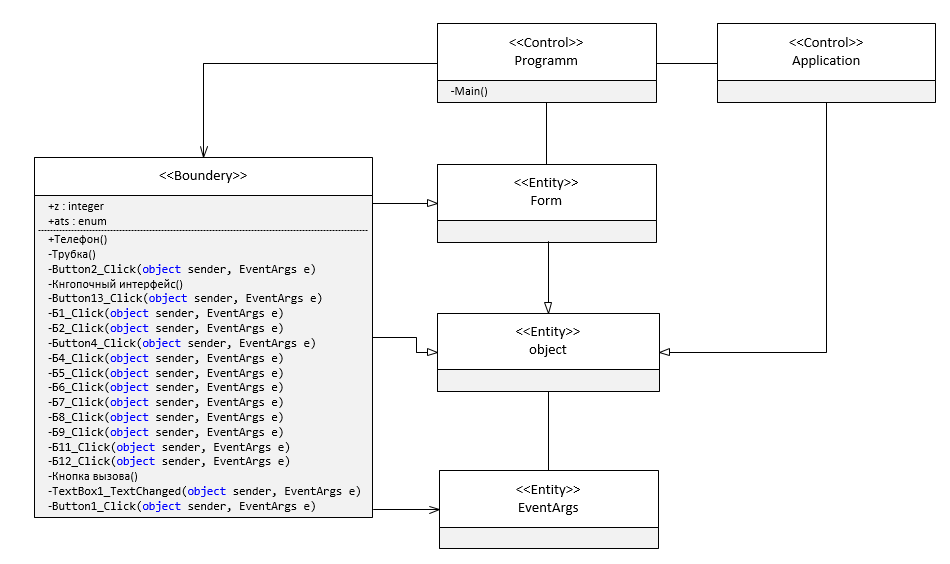


Рисунок 9 – UML-диаграмма классов системы «Телефон»

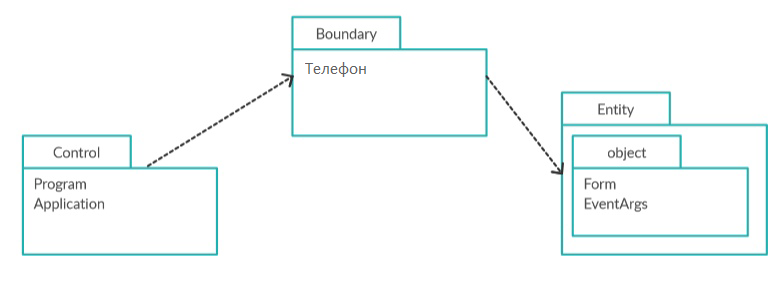


Рисунок 10 – UML-диаграмма пакетов системы «Телефон»

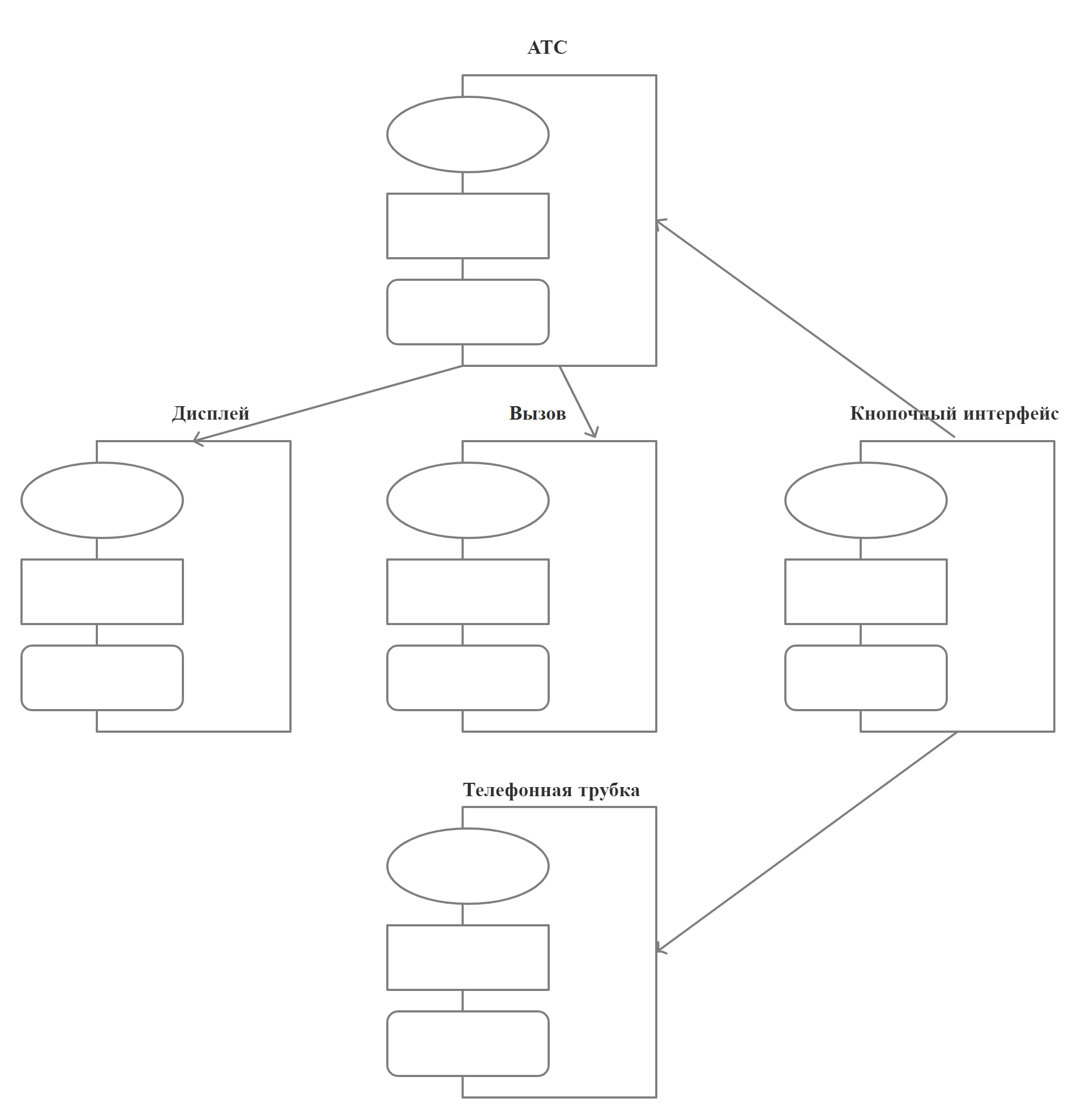


Рисунок 11 – UML-диаграмма компонентов системы «Телефон»

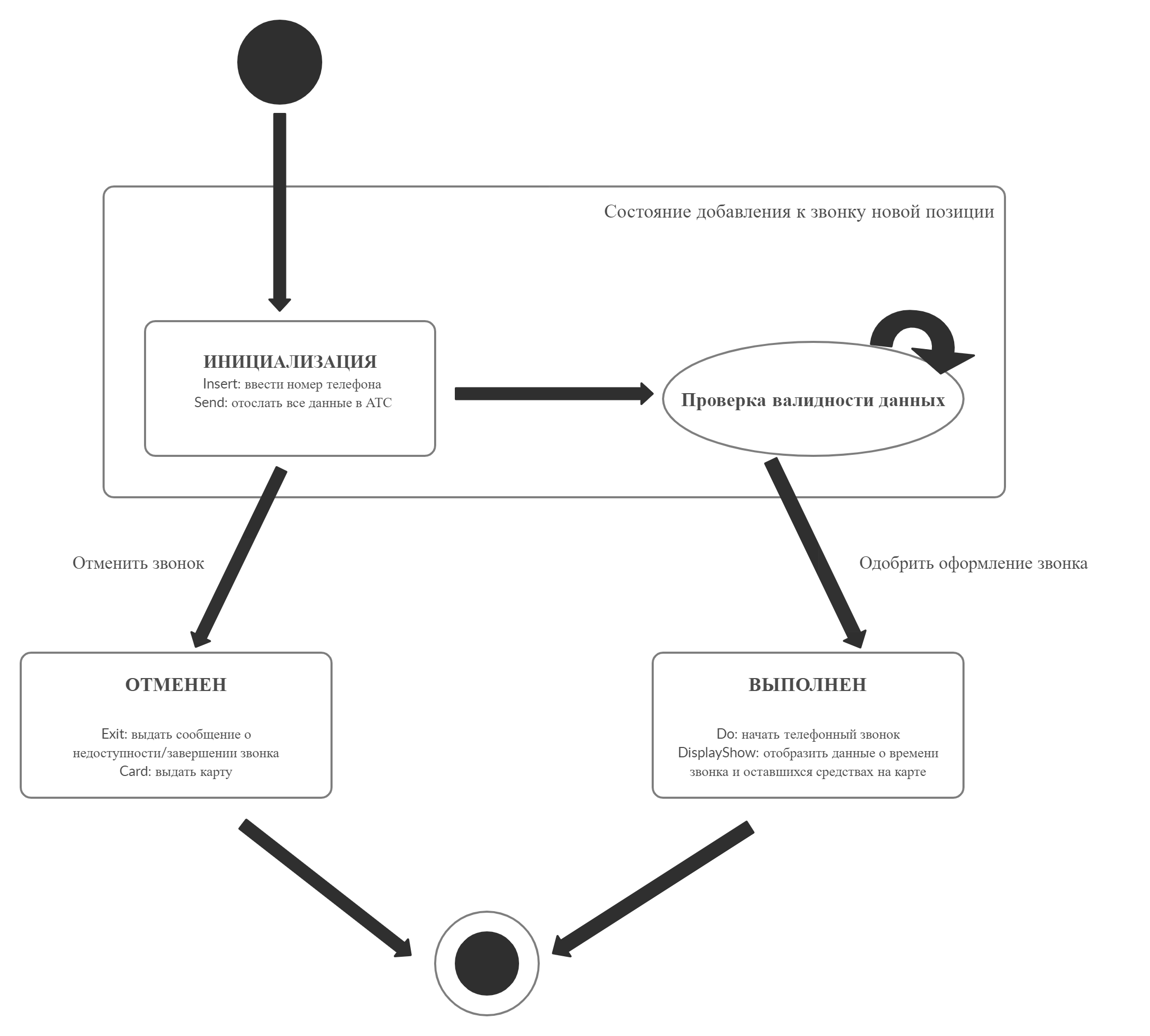


Рисунок 12 – UML-диаграмма состояний системы «Телефон»

# EPC

Событийная цепочка процессов (EPC-диаграмма, англ. event-driven process chain) — «тип блок-схемы, используемой для бизнес-моделирования. EPC может быть использована для настройки системы планирования ресурсов предприятия (ERP), и для улучшений бизнес-процессов.

Организации используют EPC-диаграммы для планирования потоков работ бизнес-процессов. Существует ряд инструментов для создания EPC-диаграмм, некоторые из этих средств поддерживают инструмент независимый формат обмена данными EPC — язык разметки EPML. EPC-диаграммы используют символы нескольких видов, чтобы показать структуру потока управления (последовательность решений, функции, события и другие элементы) бизнес-процесса.

EPC-метод был разработан Августом-Вильгельмом Шеером в рамках работ над созданием ARIS в начале 1990-х годов. Используется многими организациями для моделирования, анализа и реорганизации бизнес-процессов.»

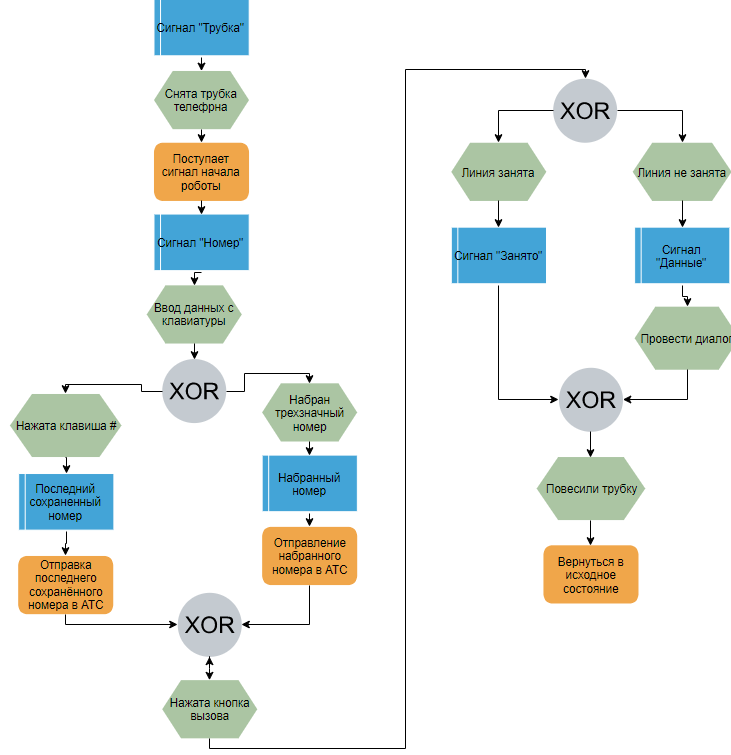


Рисунок 13 – EPC-диаграмма системы «Телефон»

# BPMN

Бизнес-процесс — это многократно повторяющаяся, логически связанная последовательность действий, направленная на создание ценности и формирование результата. И для таких процессов существует нотация, которая представляет собой описание графических элементов, используемых для построения схемы протекания рассматриваемого процесса.

BPMN – это язык моделирования бизнес-процессов, который является промежуточным звеном между формализацией/визуализацией и воплощением бизнес-процесса.

Такая схема нужна, чтобы выстроить в соответствии с ней бизнес процесс и понятно регламентировать его для всех участников. Моделирование BPMN позволяет впоследствии провести автоматизацию бизнес-процессов в соответствии с имеющейся схемой.»

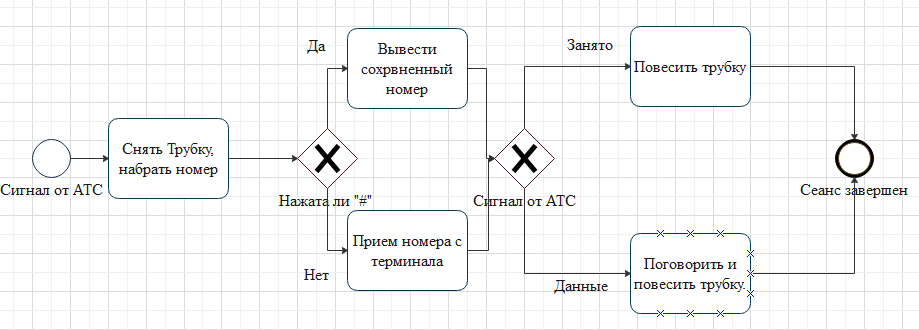


Рисунок 14 – Диаграмма BPMN

# FURPS+

Классификация требований к системе FURPS+ была разработана Робертом Грэйди (Robert Grady) из Hewlett-Packard и предложена в 1992 году.

F – Functionality, функциональность. Включает в себя функциональные возможности системы, которыми могут выступать возможность автоматизированного выполнения функций основных бизнес-процессов, автоматизация отдельных функций по информационной поддержке работы с партнерами, автоматизированная обработка и анализ управленческой информации.

U – Usability, удобство использования. Включает в себя: интуитивная понятность, эстетическая привлекательность, доступность и удобство в получении справочной информации.

R – Reliability, надежность. Включает в себя: режим доступности системы, точность проводимых расчетов, отказоустойчивость, пригодность ИС к восстановлению данных и информации.

P – Performance, производительность. Включает в себя: пропускная способность, время отклика системы, масштабируемость.

S – Supportability, поддерживаемость. Включает в себя: пригодность к проведению ремонтных работ, простота установки, адаптируемость к условиям конкретной среды, совместимость с программно-аппаратным обеспечением и другими ИС.

+ необходимо помнить о таких возможных ограничениях, как:

* ограничения проектирования, design
* ограничения разработки, implementation
* ограничения на интерфейсы, interface
* физические ограничения, physical

Полученная для данного проекта FURPS+ выглядит так:

F – отсутствует;

U – интуитивная понятность;

R – режим доступности 24/7, защита от прослушивания, большая дальность связи;

P – долговечные аккумуляторы, моментальный отклик;

S – простая установка и настройка, не требующая профессиональных знаний;

+ - никаких ограничений

# Результаты машинного тестирования программы

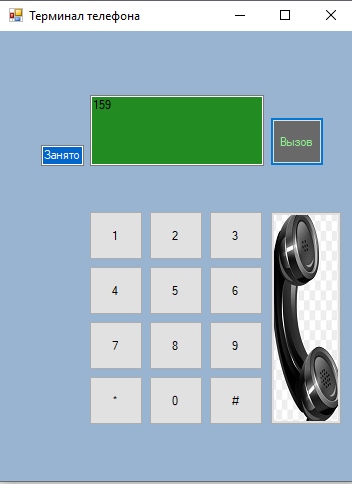


Рисунок 15 – Совершенный звонок (Занято)

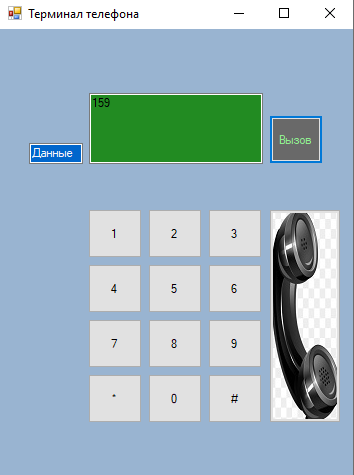


Рисунок 16 – Совершенный звонок (Данные)

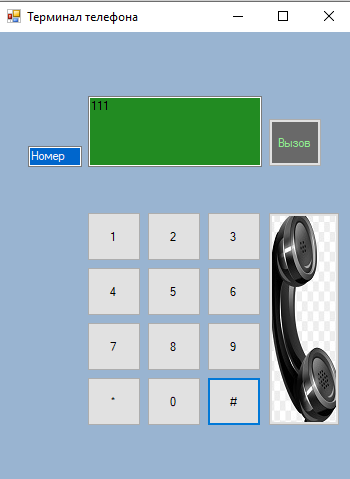


Рисунок 17 – Работа кнопки “#”

1. **Системные требования**

Таблица 1 – Системные требования программы

|  |  |
| --- | --- |
| Процессор | 2.5 ГГц |
| Оперативная память | 150 Мб |
| Монитор | 1920 x 1080 |
| Свободное место на носителе | 15 Мб |
| Устройства взаимодействия с пользователем | Клавиатура и мышь |
| Программное обеспечение | Visual Studio 2019 года последней версии |

1. **Руководство пользователя**

Запуск программы можно осуществить следующим способом: открыть исполняемый файл.

Запуск программы при помощи исполняемого файла:

* 1. Находим исполняемый файл.
  2. Запускаем исполняемый файл с форматом .exe.

Теперь перед нами интерфейс программы: кнопка телефонной трубки, панель для набора номера, кнопка вызова, кнопка сброса при ошибках “\*”, информационное окно для набора номера, информационное окно сигналов.

Для начала работы пользователь должен снять трубку, об этом гласит надпись «Трубка» на сигнальном окне. Далее надо набрать номер вызываемого абонента, при этом номер содержит только 3 цифры, если нажата клавиша «#», то воспроизводиться сохраненный номер. После нажатия кнопки «Вызов» набранный номер отправляется в АТС. Далее в сигнальном окне появиться надпись: «Данные», звонок прошел и после «разговора» стоит опять нажать кнопку трубки. Если на экране горит надпись: «Занято», то абонент разговаривает, повесьте трубку и перезвоните позже. Если программа выдала ошибку или произошел сбой нажмите кнопку «\*».

**Заключение**

В результате выполнения данного курсового проекта была спроектирована система «Мини-АТС» на языке высокого уровня C#, которая была соединена с системой «Телефон».

При построении диаграмм использовались основные правила и принципы моделирования, включающие графическое представление объектов и связей между ними, иерархическое построение, а также названия, отражающие назначение той или иной сущности, или взаимодействия.

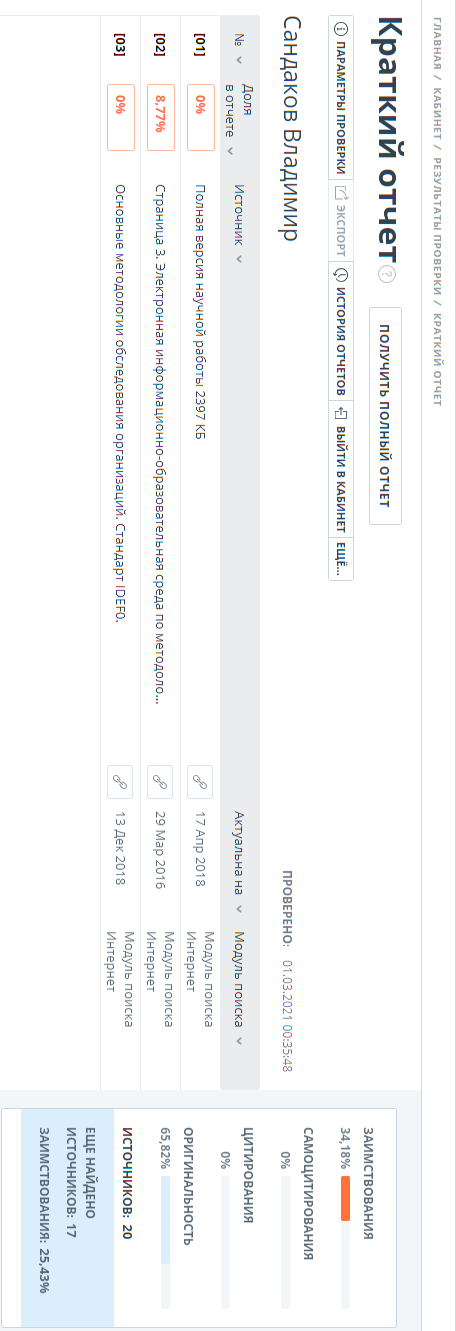
Благодаря детальному разбору проекта при помощи диаграмм проектирования, полученных в процессе разработки, а также тестированию, были выявлены и устранены ошибки в проекте. Конечный результат полученный в данном проекте не идеален, но может быть доработан во время его эксплуатации и при дальнейшей поддержке проекта.

Во время выполнения работы, разработка «Телефон» несла в себе новые знания о работе с языком высокого уровня С#. Новая программа объединяет в себе две разные программы, которые взаимодействуют между собой и не конфликтуют.

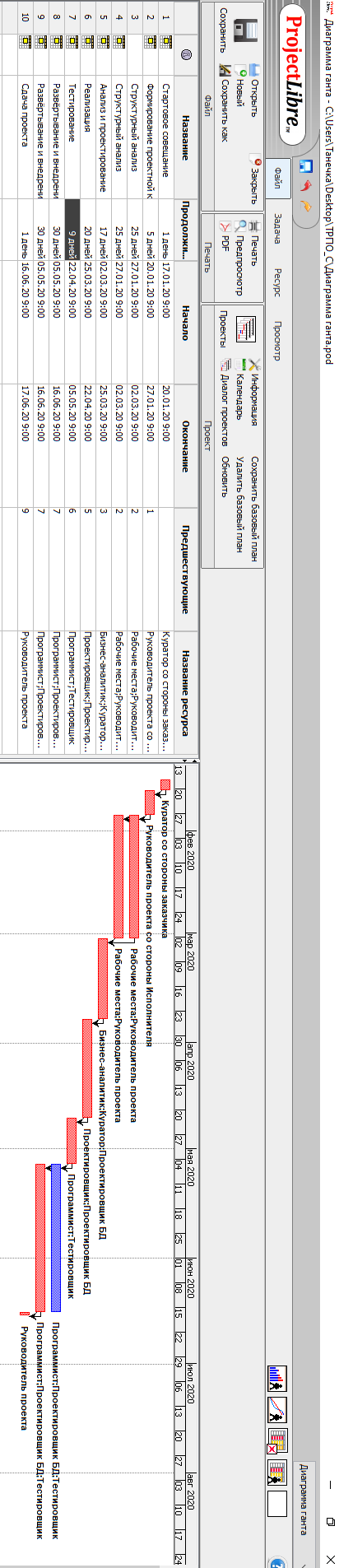
**Список использованных источников**

1. Ларман, Крэг. Применение UML 2.0 и шаблонов проектирования. Введение в объектно-ориентированный анализ, проектирование и итеративную разработку / КрэгЛарман. - Москва: Гостехиздат, 2017. - 736 c.
2. Роберт А. Максимчук. UML для простых смертных / Роберт А. Максимчук, Эрик Дж. Нейбург. - Москва: СИНТЕГ, 2014. - 272 c.
3. Йордон, Эдвард. Объектно-ориентированный анализ и проектирование систем / Эдвард Йордон , Карл Аргила. - М.: ЛОРИ, 2014. - 264 c.
4. SoloLearn – C# Tutorial. [Электронный ресурс]: - Режим доступа: <https://www.sololearn.com/Course/CSharp/> (Дата обращения 13.03.2020).
5. Википедия. [Электронный ресурс]: - Режим доступа: [https://ru.wikipedia.org](https://ru.wikipedia.org/wiki/Тестовые_функции_для_оптимизации) (Дата обращения 17.09.2019).
6. GitHub – yarajtf/intercom. [Электронный ресурс]: - Режим доступа: <https://github.com/yarajtf/intercom> (Дата обращения 06.05.2020).
7. Comindware – Нотация BPMN 2.0 [Электронный ресурс]: - Режим доступа: <https://comindware.com/ru/blog-нотация-bpmn-2-0-элементы-и-описание/> (Дата обращения 28.02.2020)
8. SysAna– Требования к системе: классификация FURPS+ [Электронный ресурс]: - Режим доступа: <https://sysana.wordpress.com/2010/09/16/furps/> (Дата обращения 03.03.2020)

# Приложение А – Проверка на антиплагиат



# Приложение Б – Диаграмма Ганта



# Приложение В – Листинг программы

Таблица 2 – Характеристика переменных к заданию

|  |  |  |  |
| --- | --- | --- | --- |
| Имя переменной | Смысл переменной | Назначение переменной | Ограничения |
| z | Счетчик | Вспомогательная | Интовая |
| f00 | Переменная вывода информации | Вспомогательная | Строковая |

Код программы:

namespace Телефон

{

public partial class Form1 : Form

{

int z;

public Form1()

{

InitializeComponent();

}

enum ats : int

{

Занято, Данные

}

Random rnd = new Random();

private void Button13\_Click(object sender, EventArgs e)

{

string a0 = Convert.ToString(0);

textBox1.Text = textBox1.Text + a0;

z++;

}

private void Б1\_Click(object sender, EventArgs e)

{

string a1 = Convert.ToString(1);

textBox1.Text = textBox1.Text + a1;

z++;

}

private void Б2\_Click\_1(object sender, EventArgs e)

{

string a2 = Convert.ToString(2);

textBox1.Text = textBox1.Text + a2;

z++;

}

private void Button4\_Click(object sender, EventArgs e)

{

string a3 = Convert.ToString(3);

textBox1.Text = textBox1.Text + a3;

z++;

}

private void Б4\_Click\_1(object sender, EventArgs e)

{

string a4 = Convert.ToString(4);

textBox1.Text = textBox1.Text + a4;

z++;

}

private void Б5\_Click\_1(object sender, EventArgs e)

{

string a5 = Convert.ToString(5);

textBox1.Text = textBox1.Text + a5;

z++;

}

private void Б6\_Click\_1(object sender, EventArgs e)

{

string a6 = Convert.ToString(6);

textBox1.Text = textBox1.Text + a6;

z++;

}

private void Б7\_Click\_1(object sender, EventArgs e)

{

string a7 = Convert.ToString(7);

textBox1.Text = textBox1.Text + a7;

z++;

}

private void Б8\_Click\_1(object sender, EventArgs e)

{

string a8 = Convert.ToString(8);

textBox1.Text = textBox1.Text + a8;

z++;

}

private void Б9\_Click\_1(object sender, EventArgs e)

{

string a9 = Convert.ToString(9);

textBox1.Text = textBox1.Text + a9;

z++;

}

private void Б11\_Click\_1(object sender, EventArgs e)

{

string a11 = Convert.ToString('\*');

System.Windows.Forms.Application.Restart();

System.Environment.Exit(1);

}

private void Б12\_Click\_1(object sender, EventArgs e)

{

string a12 = Convert.ToString("111");

textBox1.Text = textBox1.Text + a12;

z=3;

}

private void TextBox1\_TextChanged(object sender, EventArgs e)

{

textBox1.MaxLength = 3;

}

private void TextBox2\_TextChanged(object sender, EventArgs e)

{

}

private void Button1\_Click(object sender, EventArgs e)

{

if (z == 3)

{

textBox2.Text = "Вызов";

}

if (z != 3)

{

textBox2.Text = "Ошибка";

}

if (Convert.ToString(textBox2.Text)== "Вызов")

{

ats f00 = (ats)rnd.Next(2);

textBox2.Text = f00.ToString();

}

}

private void Form1\_Load(object sender, EventArgs e)

{

}

private void Button2\_Click(object sender, EventArgs e)

{

if (Convert.ToString(textBox2.Text) == "Трубка")

{ textBox2.Text = "Номер"; }

if ( (Convert.ToString(textBox2.Text) == "Данные") || (Convert.ToString(textBox2.Text) == "Занято"))

{

System.Windows.Forms.Application.Restart();

System.Environment.Exit(1);

}

}

}

}