**Федеральное государственное автономное образовательное учреждение   
высшего образования**

**КАЗАНСКИЙ (ПРИВОЛЖСКИЙ) ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ**

**ВЫСШАЯ ШКОЛА ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ И   
ИНФОРМАЦИОННЫХ СИСТЕМ**

Направление подготовки: 09.03.03 Прикладная информатика

Профиль: Прикладная информатика в экономике

**ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА**

**Разработка клиент-серверного фреймворка для высоконагруженных вычислений на базе Linux и MPI**

**Работа завершена:**

«\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_2017 г.

Студент группы 11-308 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_И.А. Габидуллин

**Работа допущена к защите:**

Научный руководитель

«\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_2017 г. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_А.С. Тощев

Директор Высшей школы ИТИС

«\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_2017 г. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ А.Ф. Хасьянов

Казань – 2017 г.

Содержание

[Введение 3](#_Toc479865662)

[1. Обзорно-аналитическая часть 4](#_Toc479865663)

[2. Технологическая часть 5](#_Toc479865664)

[2.1 Язык программирования 5](#_Toc479865665)

[2.2 Средства разработки 5](#_Toc479865666)

[2.3 Средства проектирования 6](#_Toc479865667)

[2.4 Разработка серверной части 6](#_Toc479865668)

[2.5 Разработка клиентской части 6](#_Toc479865669)

[3. Проектирование и разработка фреймворка 7](#_Toc479865670)

[3.1 Проектирование фреймворка 7](#_Toc479865671)

[3.2 Разработка серверной части 7](#_Toc479865672)

[3.2.1 Разработка MPI сервиса 7](#_Toc479865673)

[3.2.2 Разработка сервера управления 7](#_Toc479865674)

[3.3 Разработка клиентской части 7](#_Toc479865675)

[3.3.1 Разработка консольного приложения 7](#_Toc479865676)

[3.3.2 Разработка веб-клиента 7](#_Toc479865677)

[3.4 Взаимодействие клиента с сервером 7](#_Toc479865678)

[Заключение 8](#_Toc479865679)

[Список использованных источников 9](#_Toc479865680)

[Приложение 10](#_Toc479865681)

# Введение

В мире существует множество задач, которые можно разделить на мелкие подзадачи и решать их параллельно. Особенно часто это встречается в науке. Разнообразные математические преобразования и вычисления. И в мире, в котором даже на мобильных телефонах по несколько ядер у процессора, необходимо использовать возможности параллельных вычислений.

Параллельные вычисления уже давно стали важной частью Computer Science и активно используются на практике. Во многих научных организациях, например, в Казанском Федеральном Университете существует кластер электронно-вычислительных машин для выполнения параллельных вычислений. Но не существует удобного способа управлять вычислениями: разворачивать программный код, запускать, отслеживать, получать результаты или данные о скорости работы. Сейчас всё это происходит копированием файлов и запуском определенного скрипта, что, по моему мнению, не очень удобно и отвлекает от основной работы.

Исходя из вышеизложенного, было принято решение создать удобный фреймворк для работы с параллельными вычислениями. Данный фреймворк должен позволять гибко настраивать взаимодействия с сервером параллельных вычислений. Единожды развернув и настроив фреймворк пользователь получает удобный и гибкий способ управления своими вычислениями.

Таким образом, данная работа несет в себе практическую пользу в научной сфере.

# Обзорно-аналитическая часть

# Технологическая часть

Важной частью в разработке какого-либо проекта является выбор инструментов и технологий, которые будут использоваться. Выбирать следует исходя из целей и задач, так как неверно выбранная технологическая база в дальнейшем может помешать разработке. Или в худшем случае придется начать работу сначала, используя уже подходящие технологии. В данной главе будет рассмотрен выбор тех или иных инструментов, которые будут использоваться на проекте.

## Язык программирования

В качестве языка разработки был выбран C# 7. Данный язык программирования подходит для разработки разнообразных видов приложений, как приложений под мобильную платформу, так и разнообразных бизнес приложений, либо высоконагруженных систем. В каждой новой версии C# добавляются новые удобные возможности для разработчика и становится всё больше возможностей сделать код более структурированным и читабельным. Например, в новой версии добавились такие возможности как интерполяция строк и локальные функции.

## Средства разработки

Очевидно, что если в разработке используется свежая версия языка программирования, то и среду разработку надо выбирать самую последнюю. В данном случае выбор пал на Visual Studio 2017 Community.

Данная среда позволяет быстро и качественно писать код, так как в ней существует большое количество подсказок, которые могут помочь разработчику быстрее и качественнее создавать программный код. В ней удобно следить за исполнением кода. И она предоставляет все метрики проекта, которые важны в разработке. Также данная среда поддерживает большое количество языков программирования, а значит программисту не надо осваивать большое количество сред разработки, а достаточно лишь изучить одну.

При помощи Visual Studio можно легко собирать проекты, а также разворачивать, либо настраивать развёртывание при необходимости.

Для тестирования веб-сервера использовался плагин для веб-браузера Opera – RestMan. Благодаря интуитивному и удобному интерфейсу, плагином легко пользоваться.

Данный плагин позволяет отправлять REST веб запросы любого вида к указанному веб серверу и получать ответы. Ответы отображаются в удобном для анализа виде.

## Средства проектирования

В данном параграфе описывается при помощи какого программного обеспечения будет происходить проектирование данной работы, то есть где будут составляться всевозможные UML диаграммы…

## Разработка серверной части

Так как языком разработки был выбран C#, то выбор фреймворка ограничивается разными видами .NET Framework. В свою очередь .NET Framework имеет множество версий, между которыми можно выбирать:

1. .NET Framework – классический Windows фреймворк. Является старейшей платформой среди .NET платформ. Поддерживает большое количество типов приложений (Windows приложение, веб-приложение, мобильное приложение). Также платформа имеет большие возможности взаимодействия с системой. Но данная платформа поддерживает лишь семейство операционных систем Microsoft Windows. Поэтому данный фреймворк не подходит для данного проекта.
2. Mono – некоммерческий кроссплатформенный (Linux, MacOS, Windows) фреймворк. Является перенесенной под другие операционные системы .NET Framework. Но с некоторыми недоработками;
3. .NET Core – является разработанной компанией Microsoft кроссплатформенным фреймворком. В данной платформе используются наработки Mono. Несмотря на то, что является самой свежей платформой, благодаря поддержке Microsoft, развивается очень быстро.

Таким образом, стоит выбор между .NET Core и Mono. Автор предпочел .NET Core. Выбранная платформа поддерживает все нужные для проекта возможности, позволяет создать кроссплатформенный код. Также с данной платформой легче начать работу, так как Visual Studio 2017 сразу после установки поддерживает разработку под эту платформу.

## Разработка клиентской части

Предполагается то, что клиентская часть можно написать, используя любую платформу, так как взаимодействие с серверной частью происходит через протокол HTTPS (Web API).

Либо разрабатывается консольное приложение, либо веб приложение…

# Проектирование и разработка фреймворка

## Проектирование фреймворка

## Разработка серверной части

### Разработка MPI сервиса

### Разработка сервера управления

## Разработка клиентской части

### Разработка консольного приложения

### Разработка веб-клиента

## 3.4 Взаимодействие клиента с сервером

# Заключение

# Список использованных источников

# Приложение