|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **Министерство образования и науки Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ Робототехники и комплексной автоматизации

КАФЕДРА Системы автоматизированного проектирования (РК-6)

**РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

***К КУРСОВОМУ ПРОЕКТУ***

***НА ТЕМУ:***

***Разработка программной реализация удалённого запуска графоориентированных решателей систем инженерного анализа.***

Студент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

(Группа) (Подпись, дата) (И.О.Фамилия)

Руководитель **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

(Подпись, дата) (И.О.Фамилия)

Консультант **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

(Подпись, дата) (И.О.Фамилия)

*2019 г.*

**Министерство образования и науки Российской Федерации**

**Федеральное государственное бюджетное образовательное учреждение**

**высшего образования**

**«Московский государственный технический университет имени Н.Э. Баумана**

**(национальный исследовательский университет)»**

**(МГТУ им. Н.Э. Баумана)**

УТВЕРЖДАЮ

Заведующий кафедрой \_\_\_\_\_\_\_\_\_\_

(Индекс)

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

(И.О.Фамилия)

« \_\_\_\_\_ » \_\_\_\_\_\_\_\_\_\_\_\_ 20\_\_\_ г.

**ЗАДАНИЕ**

**на выполнение курсового проекта**

по дисциплине:\_\_\_\_\_ Технологии интернет\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Студент группы \_\_\_РК6-72\_\_\_\_

\_\_\_\_\_\_Громов Иван Сергеевич\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(Фамилия, имя, отчество)

Тема курсовой работы:Разработка программной реализация удалённого запуска графоориентированных решателей систем инженерного анализа.

Направленность КР: (учебная, исследовательская, практическая, производственная, др.)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Исследовательская\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Источник тематики (кафедра, предприятие, НИР) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

График выполнения КР: 25% к \_\_\_ нед., 50% к \_\_\_ нед., 75% к \_\_ нед., 100% к \_\_\_ нед.

Техническое задание: а) Провести обзор литературы по теме: "Технологии и методы удалённого запуска процедур и функций на высокопроизводительных вычислительных системах". б) Создать тестовую функцию системы, с помощью которой можно будет осуществить запуск решателя, реализованного с использованием графоориентированного подхода. в) Разработать схему архитектуры подсистемы, обеспечивающей удалённый запуск графоориентированных решателей.

***Оформление курсовой работы:***

Расчетно-пояснительная записка на \_\_\_\_\_ листах формата А4.

Перечень графического (иллюстративного) материала (чертежи, плакаты, слайды и т.п.):

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

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

Дата выдачи задания « \_\_\_ » \_\_\_\_\_\_\_\_\_\_\_\_ 20\_\_ г.

**Руководитель курсовой работы**  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(Подпись, дата) (И.О.Фамилия)

**Студент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

(Подпись, дата) (И.О.Фамилия)

Примечание: Задание оформляется в двух экземплярах: один выдается студенту, второй хранится на кафедре.

#### **Содержание**

Сокращения и обозначения 5

Аннотация 6

ВВЕДЕНИЕ 7

1. ПОСТАНОВКА ЗАДАЧИ 9

2. АРХИТЕКТУРА ПРОГРАММНОЙ РЕАЛИЗАЦИИ 10

3. ПРИНЦИП РАБОТЫ ПРИЛОЖЕНИЯ 13

4. ТЕСТИРОВАНИЕ И ОТЛАДКА 17

5. ЗАКЛЮЧЕНИЕ 21

Список литературы: 22

#### **Сокращения и обозначения**

РВС GCD – Распределенная Вычислительная Система GCD.

GUI – графический пользовательский интерфейс (англ. Graphical User Interface).

aINI – формат представления исходных данных (англ. Advanced INI).

RPC – Удалённый запуск процедур (англ. Remote Procedure Call).

*CMake* – Кроссплатформенная система автоматизации сборки программного обеспечения из исходного кода;

*Boost* – набор библиотек для языка программирования *C*++.

#### **Аннотация**

При решении сложных ресурсоёмких задач зачастую вычислительных мощностей локального персонального компьютера уже не хватает и возникает потребность в использовании многопроцессорных вычислительных систем. Задача направлена на задействование разработанной в МГТУ им. Н.Э. Баумана специальной технологии построения программных реализаций сложных вычислительных методов, основанной на применении понятий теории графов. Технология позволяет систематизировать процесс разработки программных реализаций сложных вычислительных методов.

Тип работы: курсовой проект.

Тема работы (проект темы): Разработка программной реализация удалённого запуска графоориентированных решателей систем инженерного анализа.

Объект исследований: Решения в области реализации удаленного запуска процедур на распределенных вычислительных системах.

#### **ВВЕДЕНИЕ**

Необходимость реализации удалённого вызова вычислительных процессов возникла вместе с появлением распределённых вычислительных систем, когда мощности одной локальной машины перестало хватать для выполнения сложных вычислительных операций. Проблемы, связанные с реализацией данной технологии, заключаются в неоднородности сред выполнения различных частей программы, отладке механизма синхронизации этих частей и обеспечении отказоустойчивости распределённой системы в целом.

Известны различные варианты архитектур распределённых вычислительных систем, использующих механизмы удалённого запуска ресурсоёмких вычислительных процессов. В основном, все рассмотренные реализации основаны на применении уже готовых технологиях удалённого вызова процедур, таких как: GridMD [1], XML-RPC [2], собственных библиотек на C++, реализующих удалённый запуск процедур [3], COM и CORBA [6].

В процессе проведения обзора литературы был найден источник, в котором были описаны результаты вычислительного эксперимента с использованием различных реализаций удалённого запуска процедур и сравнение их производительности [4].

Первые реализации RPC были созданы в 70-x - 80-х годах 20 века с помощью C/C++. В общем случае механизм включает в себя подсистемы:

1. транспортную – механизм передачи данных по сети (TCP, UDP, реже HTTP);
2. сериализатор – механизм преобразования входных данных процедуры, вызываемой удалённо, в формат, пригодный для передачи по сети;
3. очередь потоков вызываемой стороны – механизм определения процедуры удалённого сервера, которая должна быть вызвана.

Позже механизм RPC был усовершенствован: а) взамен подсистемы сериализации была создана подсистема «маршалинга» - процесс более высокого уровня, который позволяет передавать по сети помимо данных их кодовую базу данных, а также ссылку на объект, не передавая сам объект. Для передачи данных стали использоваться современные структуры данных (JSON, XML).

Параллельно с усовершенствованиями технологии RPC были разработаны более прогрессивные реализации удалённого вызова процедур. Наиболее современные и перспективные из них представлены ниже:

1. Cap’n Proto – развитие технологии Protocol Buffers, основная особенность которой заключается в том, что результаты вызова функций немедленно возвращаются клиенту, однако, это целесообразно только для составных запросов, где результат одной части используется для выполнения других частей запроса [7].
2. Protocol Buffers – платформенно-независимый механизм Google для сериализации структурированных данных. Основная его особенность в том, что разработчик определяет формат данных для конкретной задачи, а механизм формирует код для работы с этими данными на всех популярных языках программирования.
3. gRPC – современная реализация устаревших RPC от Google. Основные принципы в данной реализации были сохранены, однако были использованы современные технологии для их реализации, что позволило получить высокую производительность и независимость от платформы и используемого языка программирования [8].
4. Finagle – RPC от компании Twitter. Основная особенность – высокая безопасность, отказоустойчивость, стабильная работа при больших нагрузках, что делает данный механизм удобным для использования в высоконагруженных системах.
5. Thrift – Механизм, разработанный компанией Apache. Основная особенность – возможность использования различных форматов передачи данных и протоколы связи, что делает его достаточно гибким и адаптируемым под различные решения [9].

#### **ПОСТАНОВКА ЗАДАЧИ**

В рамках данной курсовой работы требовалось.

1. Провести обзор литературы по теме: «Технологии и методы удалённого запуска процедур и функций на высокопроизводительных вычислительных системах».
2. Разработать архитектуры подсистемы, обеспечивающей удалённый запуск графоориентированных «решателей».
3. Реализовать программный механизм связи произвольной **функций системы определения исходных данных с конкретным «решателем»**.
4. Реализовать механизм определения графооринетированной модели решателя без участия пользователя.
5. Разработать формат файла исходных данных для тестовой функции запуска «решателя» на основе aINI [9].
6. Разработать тестовую функцию системы, с помощью которой можно будет осуществить запуск «решателя», реализованного с использованием графоориентированного подхода [11].

#### **АРХИТЕКТУРА ПРОГРАММНОЙ РЕАЛИЗАЦИИ**

На рисунке 1 представлена процедура удалённого запуска решателя в рамках РВС GCD. Она состоит из трёх компонентов:

1. Локальная клиентская машина пользователя, через которую осуществляется доступ к web-клиенту системы;
2. Удалённый сервер, на котором располагается web-клиент системы РВС GCD и вся его «логика».
3. Удалённая вычислительная система, на которой расположен сервер приложений РВС GCD и дополнительные его модули, в виде плагинов.



Рисунок 1 – Архитектура подсистемы удалённого запуска решателей в рамках РВС GCD

Web-клиент РВС GCD(comwpc) использует собственный генератор GUI, который на основе файла исходных данных в формате .slw создаёт GUI пользователя для ввода входных данных, таких как имя решателя и имя файла входных данных для него в формате .tsk. После чего преобразует входные данные в объект класса AnyMap, который содержит помимо ссылки на файл с входными данными в хранилище подобных файлов ссылку на ID «решателя» в Базе данных «решателей» (slvid). Затем эти данные «сериализуются» и отправляются в удалённую вычислительную систему.

Сервер приложений(comaps) «десериализует» принятые данные и обращается к плагину, отвечающему за запуск «решателей» с командой на запуск. После чего плагин проверяет входные данные, ищет необходимый «решатель» в Базе данных решателей. Если «решатель» найден, то запрашивает его функцию из хранилища библиотек функций решателей и переходит к стадии запуска «решателя». На этом этапе «решателю» на вход подаются файл исходных данных в формате .tsk, который был выбран пользователем, а также ID решателя в системе РВС GCD на основе которого определяется имя файла графовой модели этого «решателя» в формате .adot, который взят из хранилища графовых моделей решателей. После выполнения «решателя», результат его работы помещается в хранилище результатов работы решателей и работа плагина завершается.

В качестве инструментов подразумевается использование специально разработанного контейнера - словаря *AnyMap*, позволяющего хранить одновременно данные различного типа. Данный словарь будет хранить информацию, считываемую из файла исходных данных специального формата *aINI* . Формат позволяет интуитивно понятно задавать исходные данные различных типов в текстовом файле, что используется при выполнении инженерного анализа, вычислительных экспериментов в РВС *GCD* конечными пользователями системы. Инструментарий перевода содержимого *aINI*-файла в контейнер *AnyMap* также реализован в библиотеках РВС *GCD*.

Поскольку работа плагина планируется на сервере под ОС *Unix*, была произведена установка данного типа ОС на локальной машине. После этого производилась установка, отладка работы самого сервера (comaps) и вспомогательных библиотек, других средств, задействованных в реализации генератора (Boost; CMake; библиотек, разработанных на кафедре - comsdk, comfrm). В качестве системы сборки была использована система *CMake*, с помощью которой собраны почти все библиотеки РВС *GCD*. Были изучены основные механизмы работы *CMake* и успешно применены при сборке программы генератора и вспомогательных библиотек.

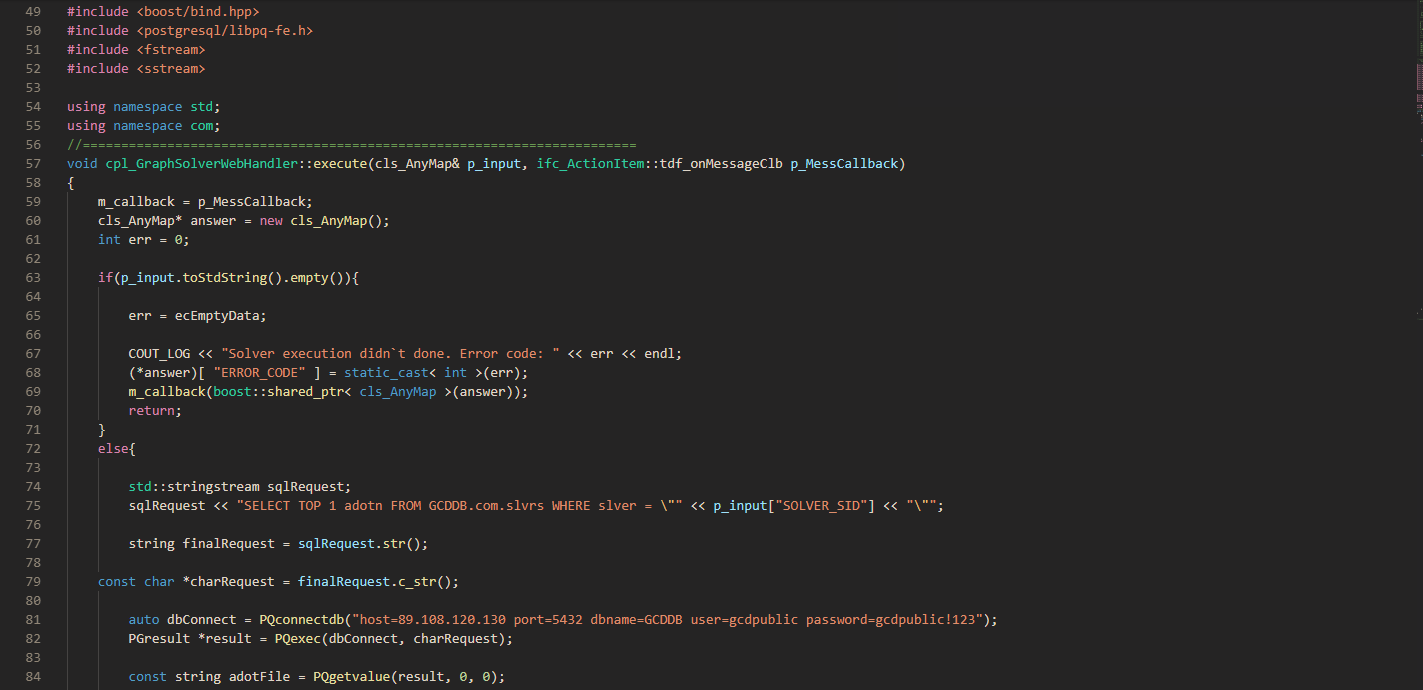
Само приложение было выполнено в формате модуля для сервера приложений РВС GCD, который регистрируется на нём при перекомпиляции приложения на этом сервере, что позволяет оптимизировать процесс его отладки и включения в систему и делает его легкодоступным с web-клиента РВС GCD.

#### **ПРИНЦИП РАБОТЫ ПРИЛОЖЕНИЯ**

В момент компиляции сервера приложений РВС GCD производится сканирование его директорий на наличие новых исполняемых модулей, после чего они регистрируются в приложении и могут быть использованы. Реализуемый плагин имеет название *comaps\_cpl\_GraphSolverWebHandler.* Исходный код плагина представлен на листинге 1. При его использовании вызывается метод *execute()*, который принимает на вход объект класса *AnyMap* со входными данными, а также сообщение для обратной связи с вызывающей его стороной. В объекте *AnyMap* находится имя выбранного «решателя», его Id в базе данных и имя файла с исходными данными, необходимыми для его выполнения. На стороне WEB-клиента вызов инициируется посредством кнопки «обработать», на странице функции запуска графоориентированных решателей с использованием WEB-клиента (функция GRPH\_SOLVER\_WEB) вычислительной подсистемы(GCDDBSFEA) Связывание плагина с функцией WEB-клинета происходит на уровне Базы данных РВС GCD. В таблицу функций WEB-клиента добавлено название функции на сервере приложений, которая обрабатывает данный запрос.

После вызова метода execute(), внутри него происходит проверка входных данных на их наличие и целостность (механизм проверки описан блок-схемой 1) и при их отсутствии возвращается ошибка, притом для каждой из возможных ошибок возвращается свой код, чтобы можно было определить место их возникновения. В случае корректных входных данных происходит запуск и выполнение «решателя». А также, в процессе выполнения происходит определения файла графовой модели выполняемого «решателя», на основе ID этого самого «решателя». Это делается с помощью базы данных РВС GCD. В таблице *com.slvrs* хранятся записи о всех существующих в системе решателях, поэтому, зная ID решателя (com.*slvrs.slver*) Мы легко можем узнать имя файла его графовой модели (*com.slvers.adotn*) .

Результат работы плагина, как положительный, так и отрицательный, сохраняются в специальном сообщении *p\_callback,* которое будет возвращено вызывающей стороне после выполнения.



Листинг 1. Исходный код плагина, реализующего удаленный запуск решателей.

Page-1

Блок-схема 1. Алгоритм работы плагина.

1. Тестирование и отладка

В рамках тестирования реализованного функционала был локально развернут сервер приложений РВС GCD и на нем была вызвана функция GRPH\_SOLVER\_WEB обработки тестового «решателя» под названием *testR*, данные для которого находятся в файле *testR.slw*, его структура представлена на рис.2.

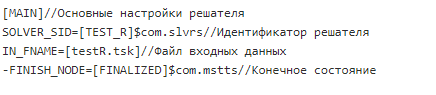
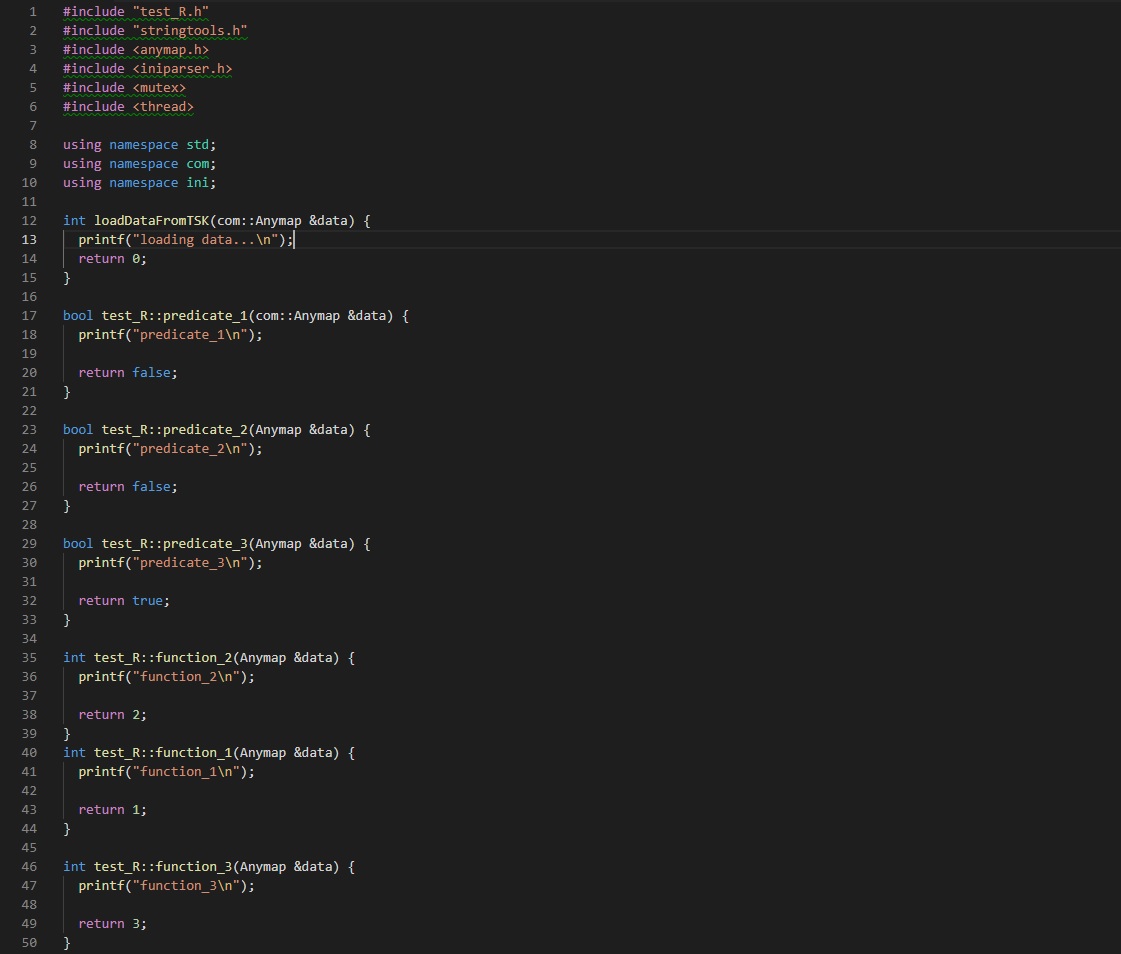


Рис.2 Структура файла *testR.slw*

На основе этого файла должен удалённо запускаться тестовый решатель TEST\_R, исходный код которого представлен в листинге 2.



Листинг 2. Исходный код решателя TEST\_R.

Этому решателю соответствует файл графовой модели *digraph TEST.аdot,* который представлен на рисунке 3. А также файл исходных данных, *testR.tsk*, который не используется конкретно в этом решателе, однако нужен для проверки доставки данных на сервер приложений.

Для тестирования реализованного плагина была написана функция для сервера приложений *test\_solverHandler* на языке С++, которая запускает выполнение плагина с произвольным объектом *AnyMap,* определяемым пользователем на входе. Исходный код этой функции представлен на листинге 3. Для запуска проверки необходимо задать в исходном коде объект *AnyMap* который будет предан плагину, а также в консоли указать IP сервера приложений (для локально развернутого это 192.168.1.100) а так же ID сервера приложений на конкретной машине (может быть любым).

Листинг 3. Исходный код функции проверки.

В результате тестирования были проверены различные варианты поведения плагина. Были протестированы два случая, в которых на вход плагину подавались некорректные данные, в обоих случаях он завершал свою работу, возвращая код ошибки. В третьем случае на вход были поданы корректные данные и плагин завершил свою работу корректно. Результаты работы тестирующей функции представлены на рисунках 3-5.

A screenshot of a cell phone

Description automatically generated

Рис. 3. Результат теста без входных данных.

A screenshot of a cell phone

Description automatically generated

Рис. 4. Результат теста с некорректными входными данными.

A screen shot of a computer

Description automatically generated

Рис. 5. Результат корректного завершения теста.

1. ЗАКЛЮЧЕНИЕ

Реализация возможности запускать «решатели» удалённо позволяет существенно упростить решение сложных вычислительных задач с использованием удалённых многопроцессорных вычислительных машин, имеющих большие мощности, чем домашние компьютеры. В рамках работы был произведён анализ источников литературы по теме «Разработка программной реализации удалённого запуска процедур», разработана программная реализация удалённого запуска процедур в РВС GCD, произведено тестирование работы программы, сделаны выводы.

#### Список литературы:

* + - 1. I.V. Morozov I. A. Valuev A. S. Automatic distributed workflow generation with GridMD library // [Computer Physics Communications](https://www.scopus.com/sourceid/13184?origin=resultslist" \o "Показать сведения о названии источника) - 182(9, - p. 2052-2058 - 2011.
      2. Tyng-Yeu Liang, Yu-Wei Chang, Hung-Fu Li. A CUDA programming toolkit on grids. //[International Journal of Grid and Utility Computing](https://www.scopus.com/sourceid/14100154713?origin=resultslist" \o "Показать сведения о названии источника) - 3(2-3) - p. 97-111 - 2012.
      3. Ahmed. L., Abhari A. A. multi-agent-based simulator for a transmission control protocol/internet protocol network. // [SIMULATION](https://www.scopus.com/sourceid/14452?origin=resultslist" \o "Показать сведения о названии источника) 90(5) - p. 511-521 - 2014
      4. Kiraly S., Szekely S. Analysing RPC and testing the performance of solutions. // [Informatica (Slovenia)](https://www.scopus.com/sourceid/25507?origin=resultslist" \o "Показать сведения о названии источника) 42(4) - с. 555-561 - 2018.
      5. Влацкая И. В., Сормов С. И. Управление и обработка информации в распределенных системах // Вестник ОГУ - №4 (110) - 2010.
      6. Вичугова А. А., Вичугов В. Н., Дмитриева Е. А., Цапко Г. П. Методы и средства интеграции инфорационных систем в рамках единого информационного пространства проектирования // Вестник науки Сибири №5. – 2012.
      7. Kenton. Is Cap’n Proto Secure? Cap`n Proto[электронный ресурс]. – Режим доступа: https://capnproto.org/faq.html#is-capn-proto-secure
      8. Grosu, P., Abdul Rehman, M., Anderson, E., Pai, V., & Miller, H. gRPC. Programming Models for Distributed Computation. Github. - Dist Prog Book[электронный ресурс] - Режим доступа: http://dist-prog-book.com/chapter/1/gRPC.html
      9. Thrift. Finagle-Quickstart. - Twitter[электронный ресурс]. - Режим доступа: https://twitter.github.io/finagle/guide/Quickstart.html
      10. Соколов А.П., Першин А.Ю. Формат данных Advanced INI (aINI) // Каркас системы – 2007-2017 – SA2 – 18 стр.
      11. Соколов А.П., Першин А.Ю., Щетинин В.Н., Сапелкин А.С. Реверсивная многомасштабная гомогенизация физико-механических характеристик гетерогенных периодических сред с использованием графоориентированного программного подхода – Композиты и наноструктуры. 2017, Т.9, № 3-4, с. 25-38.