Структура ВКР

1. Общая часть – проблема, известные пути её решения, наша идея её решения.
   1. Проблема – это АСНИ в области научных исследований и, в частности, исследования стандартов мобильной связи.
      1. Обработка потоков данных, их трансформация, формирование выходных потоков
      2. Проработка возможности формирования контрольных точек для сохранения и возобновления процессов обработки потоков данных
      3. Сохранение параметров, индикаторов и результатов преобразований

Проблема автоматизации и повышения эффективности научных исследований существовала с самого начала их появления. В настоящее время существует множество решений этой проблемы. Одним из них является использование в научных исследованиях автоматизированной системы научных исследований (сокращённо АСНИ), реже используются термины САНИ – система автоматизации научных исследований и САЭ – система автоматизации эксперимента.

Автоматизированная система научных исследований – это программно-аппаратный комплекс, использующий вычислительную технику в своей основе и предназначенный для проведения исследований практически в любой сфере науки. В основе АСНИ лежит идея моделирования объектов и их отношений из реальной жизни и оперирование этими моделями для симуляции изучаемых процессов. Научные исследования в большинстве своём предполагают произведение точных и алгоритмически сложных вычислений. Производимые старым способом, то есть вручную, они занимают много времени и усилий исследователя. Автоматизированные системы повышают эффективность научных исследований за счёт своих вычислительных мощностей, ускоряя проведение вычислений во много раз.

Сложно придумать сферу, где не пригодилась бы АСНИ. Такие системы можно применять в экономике, физике, биологии, геологии, медицине, промышленности и так далее. В частности, АСНИ могут эффективно применяться в области исследования стандартов мобильной связи.

Мобильная связь – это способ связи с беспроводным доступом к каналу связи, то есть без использования кабеля. Вместо этого, в основе мобильной связи лежит принцип радиосвязи – способ передачи сообщений на расстоянии посредством радиоволн. Благодаря ему можно производить соединение между перемещающимися в пространстве абонентами, из-за чего связь и называется мобильной. Исследования в области мобильной связи предполагают большие и сложные эксперименты, симулирующие работу системы с множеством различных объектов, что предполагает обработку объёмных потоков данных, вычисления с множеством параметров и применением различных алгоритмов обработки на одних и тех же этапах для сравнения их эффективности, формирование, хранение и анализ выходных потоков и так далее. Для проведения исследований такого масштаба удобно использовать автоматизированную систему научных исследований.

<как будто бы место для твоего текста>

Наш проект начался именно с такой проблемы. Было предложено разработать систему для помощи с разработкой сценариев экспериментов, исследующих возможности нового стандарта мобильной связи. Наше собственное решение представляет собой автоматизированную систему научных исследований, применение которой выходит за рамки данной сферы, что означает, что её можно применять не только в области мобильной связи, но и для любых других исследований. Это возможно, так как наша система позволяет:

1. Заниматься гибко настраиваемой обработкой входных потоков данных, путём написания собственного или использования уже написанного сценария из взаимозаменяемых собственноручно закодированных или встроенных в систему стадий.
2. Трансформировать данные из входных потоков в процессе проведения эксперимента и формировать выходные потоки.
3. Формировать систему из контрольных точек для получения возможности сохранения, остановки и возобновления процессов обработки потоков данных по необходимости и желанию исследователя.
4. Сохранять параметры, индикаторы и результаты преобразований в базу данных, эффективно использовать операции обращения к данным в ней (поиск, удаление и добавление), а также делать выгрузку данных из базы в удобных для исследователя форматах.
5. Анализировать полученные и сохранённые данные, например, для сравнения эффективности разных алгоритмов, применённых на одной и той же стадии; выявление проблем и новых исследовательских вопросов, возникающих при взаимодействии определённых объектов или сценариев их поведения и другое.

В общем и целом, подводя итог вышесказанному в пунктах, система предназначена для организации (создании моделей объектов участников эксперимента, каналов связи и сценариев их взаимодействия), проведения экспериментов и анализа их результатов, а также реализации сопутствующих сервисов, в частности, для моделирования работы мобильной сети, а вообще – в любой мыслимой научной области.

* 1. Пути решения проблемы – в настоящее время это решается чисто разработкой симуляционных систем. Какие есть АСНИ и что они могут.
  2. Наш подход и наша архитектура.
     1. Графический pipeline для формирования этапов и алгоритмов обработки данных
     2. Система хранения данных, необходимых для дальнейшего анализа и сравнения результатов расчётов (преобразований)
     3. Аналитическая система для собственно сравнительного анализа результатов различных экспериментов
  3. Описание платформы разработки и используемых систем.

1. Подзадача: её описание, роль в общей системе и общая идея реализации.
   1. Характеристика
   2. Требования (функциональные и нефункциональные)
   3. Идея и подробности имплементации
   4. Небольшие примеры (если это имеет смысл)
2. Реальный пример использования всей системы
3. Заключение
   1. Итоги разработки
   2. Направления дальнейшего развития и дальнейших разработок

**Описание платформы разработки и используемых систем**

Компиляторы и средства сборки:

В качестве компилятора был выбран “*Microsoft Visual C++*”, поставляемый вместе с бесплатной интегрированной средой разработки “*Mircosoft Visual Studio Express 2017*”. Данная версия компилятора и поставляемая с ним стандартная библиотека шаблонов (STL (Standard Template Library)) поддерживает стандарт C++17, который было решено использовать, как самый новый из тех, что могут быть доступны при использовании.

Помимо интегрированной среды разработки “*Microsoft Visual Studio*”, для прототипирования GUI была также использована IDE “*Qt Creator*”, ключевая особенность которого состоит в наличии редакторе форм, позволяющим добавлять в приложение элементы графического интерфейса без редактирования кода.

Для сборки проекта было принято решение использовать “*CMake*” – кроссплатформенное программное средство для автоматической сборки программы из исходного кода. Решение принято, исходя из распространенности “*CMake*” в современных проектах на языках C и C++, а также из намерения введения будущей поддержки кроссплатформенности для запуска АСНИ на различных операционных системах и аппаратных платформах.

Система управления базами данных (СУБД):

Для хранения данных, генерирующихся в ходе проведения эксперимента, было решено использовать “*PostgreSQL*” – кроссплатформенная объектно-реляционная система управления базами данных с открытым исходным кодом.

Используемые библиотеки:

1. “*Qt 5.12.12*” – набор библиотек для разработки кроссплатформенного программного обеспечения на языке программирования C++. Из поставляемого набора были использованы библиотеки “*Qt5 Core*”, ”Qt*5 GUI*”, “*Qt5 Charts*” , “*Qt5 Widgets*”.

|  |  |
| --- | --- |
| Название библиотеки | Содержание библиотеки |
| Qt5 Core | Набор классов, напрямую не связанных с GUI и используемых во всех остальных библиотеках Qt. Содержит метаобъекты для слотов и сигналов, контейнеры, например, QVector, QLine, QPoint, QMap, QRect. В библиотеку также включена событийная система QEvent. |
| Qt5 GUI | Низкоуровневые базовые классы, используемые для разработки компонентов графического интерфейса. В рамках проекта библиотека используется виджетами для рисования: QPixmap, QPainter. |
| Qt5 Charts | Компоненты графического интерфейса, используемые для построения и отображения графиков на основе статических или изменяющихся в процессе исполнения данных. Используется для отрисовки графиков в дашборде. |
| Qt5 Widgets | Содержит все компоненты оконного приложения, а также QApplication, который представляет из себя само оконное приложение, фактически, являющееся виджетом. |

1. “Google Test” – самый распространенный фреймворк с открытым исходным кодом для модульного тестирования на языке C++. Предоставляет следующие возможности:
   1. Создание одиночных тестов (тест-кейсов). Тест-кейс определяется синтаксисом “TEST(GroupName, TestName)”, где “*GroupName*” – название тестовой группы, “ *TestName* ” – наименование тест-кейса;
   2. Группировка тестов – тесты могут быть объединены в единые наборы по выбранному программистом критерия. Самым распространенным способом объединения является группировка тестов по классу, над которым проводится тестирование;
   3. Создание тестовых классов. Тестовый класс представляет из себя единую конфигурацию объектов, которая может быть повторно использована в тестах или наборах тестов. Тест-кейс, использующий тестовый класс должен определяться через макрос “TEST\_F(TestFixtureClassName, TestName)”, где “*TestFixtureClassName*” – имя того класса, который подразумевается использовать, а “TestName” – имя тест-кейса;
   4. Набор макросов, используемых для проверки утверждений. Проверки делятся на 2 типа: “Assertions” и “Expectations” и выражаются в коде через макросы, представляющие из себя регулярное выражение “ASSERT\_\*” или “EXPECT\_\*” соответственно. Одним из множества примеров таких макросов является “ASSERT\_EQ(val1, val2)” и “EXPECT\_EQ(val1, val2)”. Он проверяет, что val1 и val2 равны между собой. Различием между “ASSERT\_\*” и “EXPECT\_\*” является поведение тестовой функции после исполнения кода макроса. В случае неисполнения утверждения в “ASSERT\_\*”, тестовая функция завершит свое исполнение, а тест-кейс будет помечен, как проваленный. Если ожидание в функции “EXPECT\_\*” будет провалено, тогда тестовая функция продолжит исполнение, но при окончании тест также будет завершен с ошибкой;
   5. Создание mock-классов и имитация поведения интерфейсных функций. Mock-классы используются для имитирования поведения составных частей тестируемого класса. Они необходимы для тестирования класса при различных сценариях поведения его составных частей, связанных с ним отношениями композиции и агрегации.   
      Mock-класс определяется, как наследник базового класса, в котором все интерфейсные методы, поведение которых должно быть имитировано, заносятся под макрос “MOCK\_METHOD”.
2. “*libpq*” – библиотека с открытым исходным кодом для взаимодействия с СУБД “*PostgreSQL*”. Предоставляет набор функций, написанных на языке C и используемых для формирования запросов и транзакций к СУБД. Важно отметить, что существует еще две распространенные библиотеки для работы с СУБД “*PostreSQL*”: “*libpq++*” и “*libpqxx*”, которые предоставляют интерфейс на языке C++. “libpq” был выбран, так как в сравнении с библиотеками на C++ имеет полную и подробную документацию.