Здравствуйте, я Завёрткин Михаил, и я представляю свой проект "Создание сервера и клиента для распределённых вычислений".

По данным исследовательской компании IDC объёмы сбора и хранения цифровой информации в мире неуклонно увеличиваются. А чем больше информации, тем больше требуется времени на её обработку. Следовательно, сейчас актуальность повышения эффективности вычислений велика как никогда.

Цель моего проекта - написать приложение для проведения распределённых вычислений. Из задач я выделил планирование структуры проекта, реализацию и отладку ошибок.

Распределённые вычисления - это способ решения трудоёмких вычислительных задач с использованием нескольких компьютеров. Например, платформа BOINC объединяет домашние компьютеры обычных пользователей. Так можно добиться вычислительных мощностей, близких к суперкомпьютерам, но с куда меньшими затратами. На слайде представлено сравнение вышеупомянутой платформы и китайского суперкомпьютера по ряду характеристик. Именно благодаря большей доступности мною был выбран данный способ повышения эффективности вычислений.

Несколько лет назад я уже писал аналогичное приложение, но на языке программирования PascalABC.NET. Это была программа в функциональном стиле, что сильно снижало её гибкость, из-за чего я пришёл к решению повторить функционал, но в виде, лучше отвечающем современным практикам программирования.

Сейчас мною разрабатывается новое приложение, ориентированное на гибкость кодовой базы. Она достигается путём: использования ООП, в частности интерфейсов и абстракций; применения инъекции зависимостей с помощью стандартного DI-контейнера для реализации инверсии контроля; использования структурных паттернов, таких как MVVM, фабрика. Внедрение логирования позволяет упростить процесс отладки.

Широкое применение абстракций в программе позволяет избавиться от деталей реализации каждого компонента, что даёт возможность легко их менять, не нарушая работоспособность программы. На слайде представлен абстрактный класс Shape и два потомка, которые реализуют его методы и содержат его поля. В результате мы можем пользоваться классом, не привязываясь к его конкретной реализации.

Инверсия контроля – подход, при котором класс не разрешает свои зависимости самостоятельно, а получает их извне. В сочетании с абстракциями это позволяет менять компоненты в одном месте, централизовано. Контейнеры инъекции зависимостей применяются для упрощения кода, потому что делать это вручную зачастую долго и сложно. На слайде поле dependency является зависимостью. Первый класс разрешает её самостоятельно, а второй получает извне.

Паттерны проектирования значительно упрощают разработку и коммуникацию между разработчиками. Проще применить готовый шаблон к своей задаче, нежели придумать его заново. И объяснить свою мысль с помощью общепринятой терминологии в разы легче. Сами паттерны подразделяются на три типа - порождающие, которые служат для создания объектов, структурные для организации кода, и поведенческие, помогающие реализовать логику приложения.

Для планирования и организации работы мною широко использовались возможности сервиса GitHub, например, проекты. Они позволяют группировать задачи по различным свойствам и организовывать их выполнение. Например, на слайде представлена сортировка по категории задачи - критическая, улучшение кода и новая возможность.

Моё приложение имеет клиент-серверную архитектуру, что означает наличие двух частей – сервера и клиента. Сервер координирует выполнение задач клиентами, которые используют вычислительные мощности локальных машин для расчётов.

После запуска сервер находится в состоянии прослушивания входящих подключений от клиентов. При их появлении запрос обслуживается, и ожидается следующий. Благодаря многопоточности можно обслуживать сразу множество клиентов. На слайде представлена диаграмма классов сервера.

Клиент построен с применением паттерна MVVM, что позволяет сделать представление и модель более слабосвязанными. Например можно заменить модель в пару действий. Пользовательский интерфейс разбит на несколько визуальных частей – список задач и подробная информация о выбранной задаче. После запуска пользователь может выбрать из списка желаемую задачу для выполнения.

На данный момент разработан гибкий рабочий прототип с большим потенциалом для улучшения. Моя программа может найти свою сферу применения как в любительских задачах, так и в более серьёзных исследовательских.