1. **Актуальность**

Общение – есть способ взаимодействия людей друг с другом, а с появлением всемирной паутины это общение вышло за рамки одного дома, города и даже одной страны.

У людей есть возможность общаться по всему миру с минимальной задержкой по доставке сообщений.

Отправить электронное письмо в другую страну дешевле, чем позвонить по телефону. Интернет также помог систематизировать методы ведения бизнеса, помогая предприятиям сократить расходы.

В то время как компании прошлого, возможно, не ценили эффективную коммуникацию, современное рабочее место требует, чтобы как руководство, так и персонал имели инструменты для эффективного взаимодействия внутри и снаружи. Это связано с тем, что эффективное общение важно для повышения производительности, поскольку оно напрямую влияет на поведение сотрудников и их работу. Интернет-общение играет важную роль в быстром и эффективном выполнении работы, что в конечном итоге повышает продуктивность. Плохое общение может отрицательно сказаться на производительности, поскольку сотрудники могут не получить адекватную информацию для выполнения порученной им работы.

Построить хорошие отношения между работниками по всему миру непросто. Командировки могут отрицательно сказаться на балансе жизни и работы. Члены команды могут быстро выгореть, если им приходится совершать командировки, которые лишают их возможности участвовать в важных мероприятиях с друзьями и семьей. Благодаря интернет-коммуникациям сотрудники могут легко общаться друг с другом в любое время из любой точки мира. Это позволяет сотрудникам, находящимся в разных частях мира, высказать свое мнение и высказать свои опасения. Таким образом, интернет-общение является важным бизнес-активом, особенно для компаний, вышедших на глобальные рынки.

1. **Особенности реализации и сборка проекта**

Затронем тему сборки с использованием фреймворка Qt. Помимо исходных файлов в проекте помещается файл СMakeLists.txt. Он необходим для вызова утилиты cmake и последующего создания make файлов. Он хранит в себе заранее предусмотренные инструкции, в которых прописывается название проекта, иерархия модулей, утилиты необходимые для сборки проекта [2].

Сравнивая систему сборки C++ с системой сборки Qt, видно, что система сборки C++ (серые прямоугольники) осталась неизменной (рисунок 2). Мы все еще пишем код на C++. Однако мы добавляем больше источников и заголовков (зеленые прямоугольники) (рисунок 2). Здесь задействованы три генератора кода:

Метаобъектная компиляция

Компилятор метаобъектов (moc) принимает все классы, начиная с макроса Q\_OBJECT, и генерирует исходный файл C++ moc\_\*.сpp. Этот файл содержит информацию о моделируемом классе, такую, ​​как имя класса, дерево наследования и т. д., а также реализацию сигналов. Это означает, что, когда вы посылаете сигнал, вы фактически вызываете функцию, сгенерированную moc.

Компиляция пользовательского интерфейса

Компилятор пользовательского интерфейса (uic) берет проекты из Designer и создает файлы заголовков. Эти файлы заголовков затем, как обычно, включаются в исходные файлы, что позволяет вызвать setupUi для создания экземпляра дизайна пользовательского интерфейса.

Компиляция ресурсов

Компилятор ресурсов Qt (rcc) - позволяет встраивать изображения, текстовые файлы и т.д. в исполняемый файл, но при этом иметь доступ к ним как к файлам.

Далее, после компиляции файлов в файлы \*.h и \*.cpp, относящихся к фреймворку Qt, берутся все файлы и производятся стандартные действия по сборке, свойственные для языка C++.

Препроцессинг

Препроцессор работает с одним исходным файлом C++ за раз, заменяя директивы #include содержимым соответствующих файлов (обычно это просто объявления), выполняя замену макросов (#define) и выбирая разные части текста в зависимости от #if, Директивы #ifdef и #ifndef.

Компиляция

Шаг компиляции выполняется на каждом выходе препроцессора. Компилятор анализирует чистый исходный код C ++ (теперь без каких-либо директив препроцессора) и преобразует его в код сборки, создавая фактический двоичный файл в некотором формате (ELF, COFF, a.out, ...). Этот объектный файл содержит скомпилированный код (в двоичной форме) символов, определенных во входных данных. Символы в объектных файлах называются по имени.

Созданные объектные файлы могут быть помещены в специальные архивы, называемые статическими библиотеками, для облегчения повторного использования в дальнейшем.

Компоновка

Компоновщик - это то, что производит окончательный вывод компиляции из объектных файлов, созданных компилятором. Этот вывод может быть либо статической или динамической библиотекой либо исполняемым файлом. Он связывает все объектные файлы, заменяя ссылки на неопределенные символы правильными адресами. Каждый из этих символов может быть определен в других объектных файлах или в библиотеках.

1. **Архитектура**
2. **Схема БД**