УДК 377+378

РАЗРАБОТКА КРОССПЛАТФОРМЕННЫХ ПРИЛОЖЕНИЙ НА C++ В СРЕДЕ ПРОГРАММИРОВАНИЯ Qt

Н.Л. Боброва
Институт информационных технологий БГУИР, г. Минск, Республика Беларусь
natasha.bobrowa@gmail.com

Qt is a powerful cross-platform application development library - desktop, network and mobile devices. This article describes how to take full advantage of the most valuable of the new API, introduced in the latest version. The focus is on approaches with the greatest efficiency and flexibility, but it does not create additional difficulties.

Qt - кросс-платформенный инструментарий для разработки программного обеспечения. Этот инструментарий создан компанией Trolltech и в данный момент принадлежит компании Nokia. Qt – это совокупность кроссплатформенной библиотеки классов, реализованной на языке C++, и ряда дополнительных инструментальных средств, включающих Meta Object Compiler (MOC) – объектный предкомпилятор, User Interface Compiler (UIC) – компилятор пользовательских интерфейсов, qmake – средство управления сборкой проектов. Поддерживаются операционные системы MS Windows, Linux, MacOS, а также встраиваемые операционные системы Embedded Linux, Windows CE, Symbian. Наиболее известными примерами разработки на Qt являются: программа-коммуникатор Skype, медиа-плеер VLC, Google Earth, графический интерфейс пользователя KDE, применяемый в ОС Linux. В состав Qt входят следующие группы классов:

- классы, обеспечивающие разработку оконного графического интерфейса пользователя, включая все основные управляющие примитивы;
- классы, реализующие работу с потоками, объектами синхронизации процессов/потоков;
- классы для работы с 2-х и 3-х мерной графикой,
- классы, реализующие поддержку некоторых графических форматов хранения;
- классы для работы с XML.

В настоящее время существует две не полностью совместимые ветви версий Qt – 5.x и 4.x. При этом ветвь 4.x сохраняется для поддержки старых программ, а разработка новых рекомендована с использованием 5.x. Qt 5.x и 4.x поставляются в составе современных Linux-дистрибутивов, обеспечивая возможность разработки в интегрированных средах KDevelop, Eclipse и пр. Для ОС Windows имеются средства, позволяющие интегрировать Qt в среду разработки: uic, moc – компиляторы и QtDesigner. При этом возможна интеграция Qt 5.x в MS Visual Studio 2013 и Qt 4.x — в MS Visual Studio 2010/2008. OpenSource Qt для Windows может быть интегрирована в IDE Eclipse с подключенным компилятором mingw-gcc, а также использоваться совместно с кроссплатформенной IDE QtCreator. Библиотеки для использования могут быть откомпилированы любым компилятором C++, имеющимся в ОС, например для Windows - MS Visual C++, Borland C++, mingw-gcc. В библиотеке реализовано автоматическое удаление объектов, являющихся элементами графического интерфейса пользователя. Механизм реализован следующим образом: любой подобный объект Qt является потомком QObject, в состав которого входят средства хранения и позиционирования списка потомков, т.е. объектов, при создании которых этот объект указан как parent. При использовании Qt совершенно естественным является переопределение классов Qt средствами C++, что существенно упрощает код в приложениях, требующих однотипной реализации нестандартных элементов, например

создание класса кнопки виртуальной клавиатуры с изменяемой надписью/рисунком на основе стандартного класса QButton. В Qt может быть использовано кроссплатформенное средство управления сборкой проектов qmake, посредством которого из .pro-файлов генерируются файлы makefile для конкретной платформы с конкретными компиляторами и компоновщиками[1].

Формы с использованием классов Qt могут создаваться вручную или с использованием специального пакета QtDesigner. При создании форм вручную программист кодирует текст программы, включая по мере необходимости вызовы объектов классов Qt. При использовании QtDesigner программист графически компонует внешний вид и связи сигналов и слотов формы, а компилятор интерфейса UIC формирует из полученного описания формы код на языке C++, обеспечивающий создание этой формы. Qt расширяет синтаксис описания классов C++ специальными средствами, обработка которых возложена на MOC. MOC обрабатывает исходный текст программы, подставляя вместо специфических конструкций реализацию заказанных свойств на C++. Соответственно на выходе MOC получается исходный код C++. Компиляция и сборка программы осуществляется компилятором C++ и компоновщиком, доступными в рамках платформы, где осуществляется сборка (см. рисунок 1)[2].

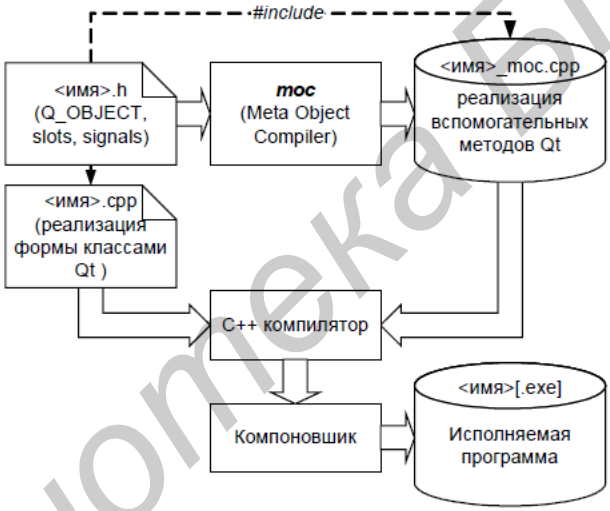

Рисунок 1 – Схема сборки приложения, реализованного вручную

Ключевым механизмом взаимодействия объектов в Qt являются *сигналы* и *слоты*. Каждый объект, интегрированный в систему управления Qt, т.е. описанный как Q_OBJECT, может иметь типизированные слоты, обеспечивающие прием и обработку типизированных сигналов от других объектов, и собственные сигналы, прием которых могут осуществлять другие объекты. Связь между сигналами и слотами конкретных объектов устанавливается посредством функции **connect**(…). Декларация сигналов и слотов осуществляется в теле класса с помощью ключевых слов signals и slots, воспринимаемых компилятором moc. Если необходимо предотвратить использование указанных ключевых слов, встречающихся в других библиотеках, то вместо них используют ключевые слова Q_SIGNALS, Q_SLOTS. По правилам Qt один слот может принимать несколько сигналов, а один сигнал транслироваться на несколько слотов. Причем во взаимодействии участвуют не классы, а конкретные объекты, поэтому схема передачи сигналов к слотам может быть в любой момент динамически изменена[3].

При необходимости быстрого получения результата, проведения экспериментов по размещению объектов, общей оценки интерфейса возможно использование

специального редактора интерфейсов QtDesigner. QtDesigner не накладывает никаких ограничений на средства разработки, поскольку интерфейс, созданный им, в конечном счете будет преобразован компилятором uic в код программы на языке C++, обеспечивающий создание именно этого интерфейса (см. рисунок 2).

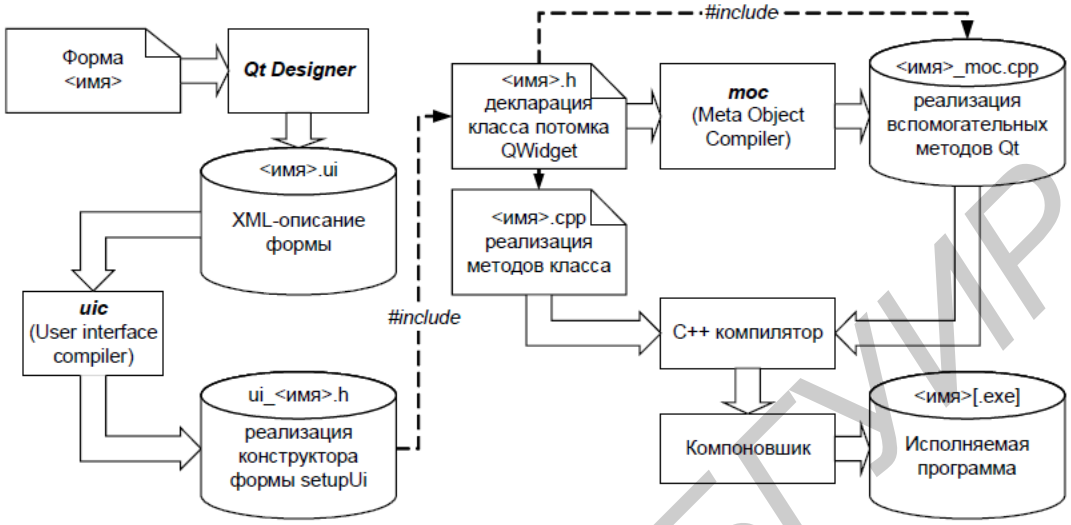

Рисунок 2 – Схема сборки приложения с формами, сделанными в QtDesigner

Это позволяет также использовать QtDesigner для обучения принципам программирования Qt и размещения элементов в форме, т.к. результирующий код является доступным и использует те же классы Qt, которые необходимы при ручной разработке. При использовании QtDesigner описание поведения, т.е. слотов, осуществляется программистом в отдельном файле. В Qt5.x такой файл имеет суффикс *.ui.h,* а в Qt4.x – .h. Сформированный QtDesigner файл .ui представляет собой XML-описание диалога. По созданному .ui – описанию user interface compiler (uic) генерирует код программы на языке C++, где создание диалога осуществляется классами Qt. В Qt3.x формируется класс-потомок QWidget, в Qt4.x формируется самостоятельный код, обеспечивающий создание формы по вызову метода setupUi(). Диалоги, созданные в QtDesigner также могут подключаться в программу динамически посредством класса QFormBuilder или QWidgetFactory::create("form.ui") в Qt5.x без необходимости генерации и компиляции кода их создания на C++.

Литература:

1. Шлее М. - Qt4. Профессиональное программирование на C++/ Шлее М. – Л.: Наука, 2013. – 770 с.

2. Бланшет Ж., Саммерфилд М. - QT 4: программирование GUI на C++/ Бланшет Ж., Саммерфилд М. М. — М. : ALT Linux, 2015. — 948 с. : ил.

3. А. В.Чеботарев Библиотека Qt 4. Программирование прикладных приложений в среде Linux / А. В.Чеботарев – Л.: Наука, 2013. – 821 с.