KNIHOVNA QT4, PROSTŘEDÍ QTCREATOR A MOŽNOSTI JEJICH VYUŽITÍ

Petr Fiala, Michal Rost, Vladimír Španihel, Miroslav Virius
České vysoké učení technické v Praze, Fakulta jaderná a fyzikálně inženýrská

ABSTRAKT:

Příspěvek nás stručně seznámí s frameworkem Qt4, zejména s objektovým modelem této knihovny, a na příkladu ukáže způsob komunikace mezi objekty v rámci aplikace založené na Qt4. Tento příspěvek představuje přípravnou studii projektu využití této knihovny při vytváření speciálních aplikací pro řízení přístrojů používaných v lékařství; shrnuje základní informace o prostředí QtCreator a hodnotí jeho využitelnost pro tvorbu předpokládaného typu aplikací.

ABSTRACT:

This paper introduces Qt4 development framework. Basic facts about Qt4 library and its object model are explained. It is described how features like signals and slots or qmake work. Furthermore this paper introduces QtCreator development tool, summarizes its features and discuss it's usability for application development.

KLÍČOVÁ SLOVA:

Knihovna Qt, programovací jazyk C++, QtCreator

ÚVOD

Na schopnosti současných programovacích jazyků jsou kladeny nemalé požadavky. Množí se názor, že jazyky, které nejsou čistě objektové, neumožňují reflexi a pohodlné vytváření GUI, jsou odsouzeny k zániku. Stále je však mnoho pracovišť zejména z oblasti fyzikálního či lékařského výzkumu a praxe, kde se používá jazyk C++.

Při tvorbě aplikací, které slouží k ovládání lékařských zařízení, jsme se museli vyrovnat s mnoha požadavky, jako je efektivní komunikace v reálném čase nebo nezávislost použitých knihoven na operačním systému. Framework Qt4 představuje jedno z možných řešení, jak jazyk C++ alespoň částečně rozšířit o možnosti současných jazyků, jakými jsou Java nebo C#.

1. KNIHOVNA QT4

1.1 Základní charakteristika

Qt je na platformě nezávislý framework rozšiřující jazyk C++ o možnosti vytváření GUI, práci s řetězci, práci s vlákny, práci s kontejnerovými datovými typy nebo parsování XML. Qt tak do jisté míry nahrazuje standardní knihovnu jazyka C++ (STL), či hojně využívané multimediální knihovny, jako je například knihovna SDL.

Celý framework byl původně vytvořen společností Trolltech. Od roku 2008 je za jeho vývoj zodpovědná společnost Nokia. Aktuální verze Qt 4.6 je pro nekomerční využití distribuována pod licencí LGPL.

Podrobné informace o knihovně Qt lze najít např. v [4] a [8].

1.2 Objektový model

Framework Qt4 je plně založen na objektech. Obsahuje propracovaný objektový model *Meta Object System* (MOS), který umožňuje rozšířit stávající objektový model jazyka C++ o reflexi a s ní spojené výhody.

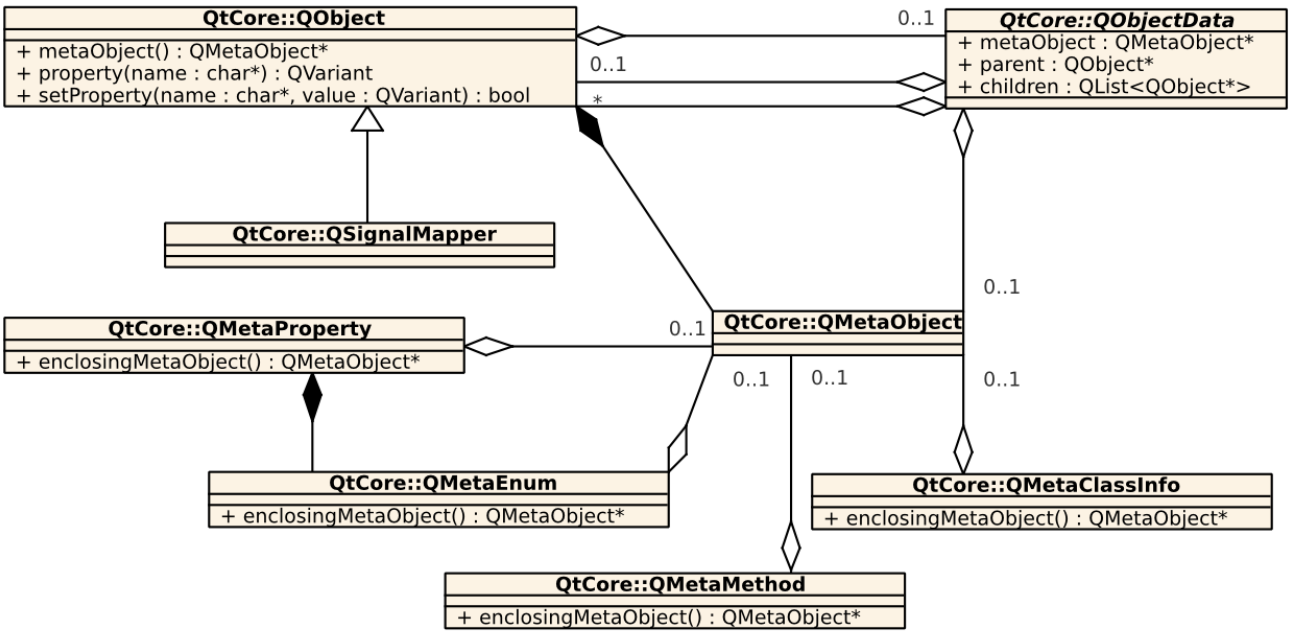
1.2.1 Meta Object System

Celý MOS je postaven na třech základních pilířích: datovém typu `QObject`, makru `Q_OBJECT` a preprocesoru Meta Object Compiler (MOC).

`Object` je základní třídou knihovny Qt4. Objekty, které chtějí využívat MOS, musí být od této třídy odvozeny. Dědění od třídy `QObject` s sebou nese několik omezení. Prvním významným omezením je, že potomek `QObject` nemůže mít veřejný kopírovací konstruktor. Druhým omezením je nutnost uvést makro `Q_OBJECT` v deklaraci potomka.

Použití makra `Q_OBJECT` v soukromé sekci deklarace třídy odvozené od `QObject` zpřístupní pro deklarovanou třídu reflexi.

Preprocesor MOC slouží k vygenerování metadat pro třídy, které jsou potomky `QObject`. Preprocesor prohledá zdrojový kód Qt4-aplikace a nalezne třídy, které v deklaraci obsahují makro `Q_OBJECT`. Pro tyto třídy MOC vygeneruje C++ zdrojový soubor obsahující metadata a odvolá se na něj direktivou `#include` z místa deklarace uvedené třídy. MOC je nutné pro zvolený projekt spustit před zahájením vlastního překladu projektu (o to se lze postarat v souboru makefile).



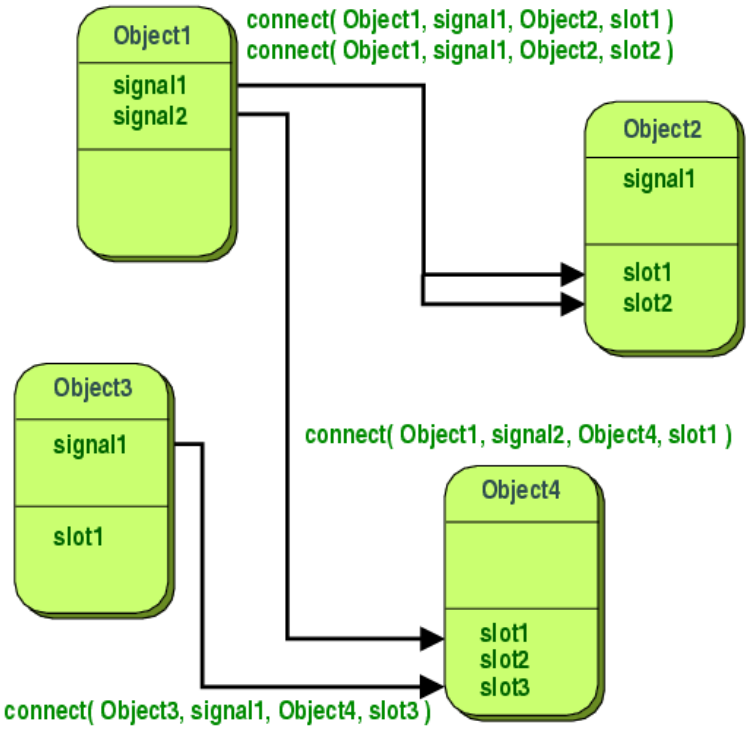
Obrázek 1: Základní třídy QtCore

1.2.2 Výhody plynoucí z použití MOS, signály a sloty

Třída využívající MOS poskytuje programátorovi metadata. Programátor má přístup ke jménům datových složek a metod používané třídy a k jejich počtu, dále k názvu třídy a dalším souhrnným informacím. Lze jednoduše programově zjistit, zda je vybraná instance odvozená od vybrané třídy.

Unikátním řešením pro zprostředkování komunikace mezi objekty jsou tzv. *signály* a *sloty*. *Signály* jsou události vysílané objekty a *sloty* jsou metody představující reakce na signály. Signály a sloty mohou být popsány v rámci deklarace třídy využívající MOS.

Poznamenejme, že jde o implementaci standardního návrhového vzoru *pozorovatel* (Observer).



Obrázek 2: Znázornění použití signálů a slotů (převzato z [4])

Každý signál může být pro konkrétní instanci třídy, v níž je popsán, spojen s odpovídajícím slotem libovolné instance libovolné třídy, ve které je tento slot popsán. Slot odpovídá signálu, shoduje-li se s ním v parametrech. Mezi signály a sloty je vztah *m*:*n*, jeden signál může být připojen k více slotům a k jednomu slotu může být připojeno více signálů.

V závislosti na svém stavu může objekt vyvolat signál a předat mu požadované parametry. Vyvolání signálu má za následek volání jemu přiřazených slotů a předání potřebných parametrů. Samotné sloty mohou být volány také programátorem, jako obyčejné metody.

Signály a sloty jsou hojně využívány v rámci Qt4 GUI. Jednotlivé elementy GUI mají předdefinovány celou řadu signálů a slotů, jejichž prostřednictvím lze zajistit reakce na uživatelské vstupy.

Příklad: deklarace třídy, obsahující signál a slot

```
#include <QObject>

class Counter : public QObject {
Q_OBJECT
public:
Counter() {
  value = 0;
}
int getValue() const {
  return value;
}
public slots:
  void setValue(int value);
signals:
  void valueChanged(int newValue);
private:
  int value;
};
```

Definice slotu

```
void Counter::setValue(int value) {
  if (this->value != value) {
    this->value = value;
    emit valueChanged(value);
  }
}
```

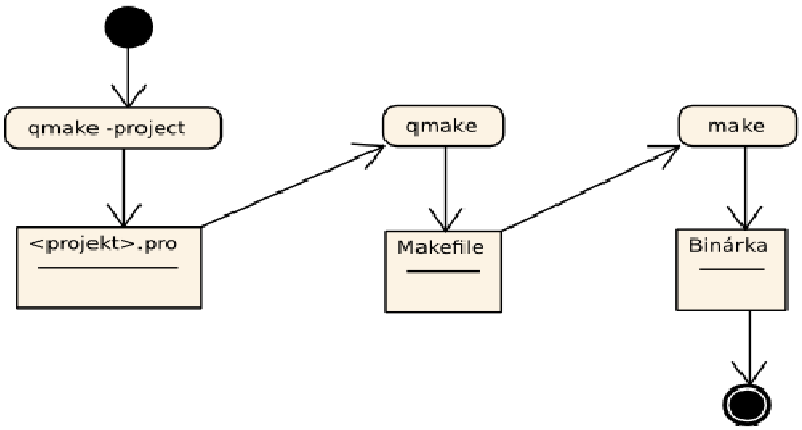
Použití signálů a slotů

```
Counter a, b;
QObject::connect(&a, SIGNAL(valueChanged(int)), &b,
SLOT(setValue(int)));
a.setValue(12); // a.getValue() == 12, b.getValue() == 12
b.setValue(48); // a.getValue() == 12, b.getValue() == 48
```

1.3 Qmake

Qmake [4] je nástroj, který umožňuje zjednodušit proces vytváření souborů makefile pro libovolné platformy, podobně jako známější cmake slouží k automatickému generování souboru makefile na základě uživatelem popsaného jednoduchého konfiguračního souboru *.pro.

V konfiguračním souboru mohou být zadány cesty k zdrojovým souborům projektu a cesty k používaným knihovnám a jejich hlavičkovým souborům pro jednotlivé platformy. V závislosti na používané platformě může být pak pomocí `qmake` vygenerován požadovaný makefile. Hlavní výhodou `qmake` oproti cmake je zabudovaná podpora pro Qt4 a automatické generování instrukcí pro MOC do makefile.



Obrázek 3: Překlad programu s využitím qmake

2. VÝVOJOVÉ PROSTŘEDÍ QTCREATOR

2.1. Základní informace, klady a zápory

QtCreator je vývojové prostředí vyvíjené společností Nokia a je určené pro programování v jazyce C++ s využitím frameworku Qt4 [2][4]. QtCreator je plně modulární a jeho funkčnost je možné rozšířit rozličnými zásuvnými moduly. Ve výchozím stavu v sobě integruje přehledný editor kódu, ladicí nástroje a SVN, GIT klienty pro správu verzí.

Velkou předností tohoto prostředí je možnost využití cmake nebo `qmake` konfiguračního souboru jako projektového souboru pro vyvíjenou aplikaci, což má za následek že projekt jako celek není vázán pouze na jedno vývojové prostředí a zvyšuje se tím jeho přenositelnost. Do prostředí QtCreator je také plně integrován grafický návrhář okenních aplikací QtDesigner.

V současné verzi QtCreatoru výrazně chybí konfigurovatelný nástroj pro automatické formátování kódu. Chybí jednoduchý refaktorovací nástroj pro C++, který by uměl alespoň automaticky generovat settery a gettery.

Prostředí QtCreator je šířeno i se svými zdrojovými kódy. Analýza těchto kódu může pomoci zkušenějšímu programátorovi, který začíná pracovat s Qt4, začít tento framework účinně využívat. Jedním z důvodů je propracovaný objektový model tohoto prostředí, který je celý založen na návrhových vzorech [5].

2.2. Podpora jednotkových testů

Zajímavým prvkem v prostředí QtCreator je zabudovaná podpora jednotkových testů. QtCreator má integrovaný nástroj QTestLib framework pro testování Qt4 aplikací a knihoven. S pomocí tohoto nástroje lze psát jak běžné jednotkové testy tak základní testy GUI, založené na

simulování uživatelských vstupů pomocí myši a klávesnice. Pro vytváření složitějších GUI testů jsou k dispozici komerční nástroje, např. *Squish for Qt* od firmy Froglogic nebo *KD Executor* od firmy KDAB. QTestLib rovněž obsahuje podporu benchmarků. Navíc jsou výstupy produkované QTestLib frameworkem snadno interpretovatelné v prostředích Visual Studio a Kdevelop, díky čemuž není programátor používající testy nucen vyvíjet Qt4 aplikace pouze v Qt creatoru.

Test se vytvoří tak, že se napíše třída využívající MOS a do ní se přidá jeden nebo několik soukromých slotů. Až na několik výjimek je každý soukromý slot v testu považován za testovací funkci. Spuštění všech testovacích funkcí v testu může být provedeno pomocí `QTest::qExec()`.

Pokud je k sestavení používán `qmake`, stačí přidat do projektového souboru řádek `QT += testlib`. V ostatních případech je potřeba správně nastavit cestu k hlavičkovým soborům QTestLib frameworku.

3. VYUŽITÍ QT4 PŘI TVORBĚ MEDICÍNSKÝCH APLIKACÍ

Na obslužný software je pro manipulaci s lékařskými přístroji, pro sběr a vyhodnocování naměřených dat a pro jejich následné zobrazení jsou kladeny následující požadavky:

- Schopnost komunikovat se zařízením v reálném čase
- Schopnost rozšiřovat funkčnost aplikace pomocí pluginů
- Schopnost fungovat pod operačními systémy Linux a Windows
- Poskytovat uživateli přehledné GUI
- Zobrazovat 3D vizualizaci používaného zařízení.

Požadavek, aby obslužný software byl schopen efektivní komunikace v reálném čase, vedl jednoznačně k volbě programovacího jazyka C++.

Poslední dva uvedené požadavky však vyžadují využití specializovaných knihoven. Aby byla aplikace schopná fungovat pod libovolným operačním systémem, museli jsme se omezit pouze na knihovny, které jsou pro požadované operační systémy dostupné. Odpadla tak například možnost využití knihovny MFC pro tvorbu GUI nebo knihovny DirectX [7] pro tvorbu 3D vizualizací.

Pro tvorbu vizualizací byla vybrána knihovna OpenGL [6] jakožto jediná možná na operačním systému nezávislá alternativa. K tvorbě GUI lze v C++ využít dvou rozsáhlých knihoven: již zmiňované knihovny Qt4, nebo knihovny GTK+ [3]. Po analýze možností obou knihoven byla dána přednost knihovně Qt4, díky jejím následujícím výhodám:

- Nativní podpora OpenGL
- Využívá principů návrhového vzoru MVC a umožňuje důsledně oddělit data od jejich grafické reprezentace (více o návrhovém vzoru MVC v [5], více o MVC v Qt4 v [8])
- Je k dispozici obsáhlejší a ucelenější dokumentace
- Využití `qmake` výrazně zvyšuje přenositelnost zdrojového kódu i s požadovanými závislostmi
- Na rozdíl od knihovny GTK+, která je pouze grafickým toolkitem, je Qt4 framework, s propracovaným objektovým modelem, umožňujícím využít při programování například již zmiňovanou reflexi

ZÁVĚR

Z uvedených skutečností vyplývá, že pro programování aplikací pro řízení lékařských přístrojů v jazyce C++ představuje vývojový framework Qt4 nejlepší volbu z uvedených možností.

PODĚKOVÁNÍ

Práce na tomto příspěvku byla podporována z grantů MŠMT LA08015 a SGS 10/094.

LITERATURA

[1] Dirk L.; Mejzlík P.; Virius M. *Jazyky C a C++ podle normy ANSI/ISO*. Praha: Grada Publishing 1999. ISBN 80-7169-631-5

[2] Future Publishing Limited. *Qt Creator*. TuxRadar [online]. 8. 5. 2009, [cit. 2010–03–18]. Dostupný z WWW: <http://www.tuxradar.com/content/qt-creator>.

[3] GTK+ Team. *GTK+ Documentation.* The GTK+ Project [online]. 2008, [cit. 2010-03-19].

Dostupný z WWW: <http://www.gtk.org/documentation.html>

[4] Nokia Corporation. *Qt Reference Documentation*. [online]. 15. 2. 2010, [cit. 2010–03–17]. Dostupný z WWW: <http://doc.trolltech.com/4.6/>.

[5] Pecinovský R. *Návrhové vzory*. Brno: Computer Press 2007. ISBN 978-80-251-1582-4

[6] Rost M. *Grafická knihovna OpenGL.* Bakalářská práce. Praha: ČVUT 2009

[7] Španihel V. *Uživatelská příručka ke knihovně DirectX*. Bakalářská práce. Praha: ČVUT 2009

[8] Vaněk P.; Watzke, D. *Grafické programy v Qt4*. Abc linuxu [online]. 26. 1. 2010, [cit. 2010–03–17]. Dostupný z WWW: <http://www.abclinuxu.cz/serialy/qt-4-psani-grafickych-programu>.