文章编号: 1003-5850(2008)05-0030-03

基于Qt/Embedded的GUI应用程序的实现

Implementation of GUI Application based on Qt/Embedded

杨中华 李 兵

(西华大学数学与计算机学院 成都 610039)

【摘 要】随着当前各种手持设备、无线设备及信息家电等嵌入式产品的迅猛发展,相应的嵌入式软硬件设计技术也在发生深刻的变化。如今越来越多的嵌入式终端需要一个图形化的人机接口面,GUI应用程序的设计是嵌入式系统设计的一个关键技术,能极大地提高人机交互的效率。介绍了嵌入式Linux GUI的发展概况,描述了Qt/Embedded的开发环境,并结合实例阐述了基于QT/Embedded的GUI应用程序的开发过程以及如何将应用程序添加到Qtopia。

【关键词】嵌入式Linux,Qt,Qt/Embedded,Qtopia,数据库SQLite

中图分类号: TP217 文献标识码: A

ABSTRACT With the rapid development of the handheld devices, wireless equipments and information appliances and other embedded products, corresponding embedded software and hardware design technologies develop greatly. Nowadays GUI is required by more and more embedded graphic terminals. The GUI application, as a key technology in the embedded system design, will be able to greatly improve the efficiency of human-computer interaction. This paper introduces the latest development of embedded Linux GUI and describes the GUI development environment of Qt/Embedded. Combining with the development experience, the procedure of GUI application development based on Qt/Embedded and how to add application into Qtopia are described in detail.

KEYWORDS embedded Linux, Qt, Qt/Embedded, Qtopia, database SQLite

1 嵌入式GUI的发展概况

普通Linux一般提供基于X-Window的KDE/GNOME作为图形用户界面,但对于嵌入式系统来说,这些都太庞大太臃肿,因此,一批基于嵌入式Linux的GUI系统就应用而生,目前发展比较成熟的主要包括以下几种: ①MiniGUI[1]是在Linux控制台上运行的,它是基于SVGALib和Linux Thread库的多窗口图形用户界面支持系统。MiniGUI采用了类Win32的API接口,实现了简化的类Windows 98风格的图形用户界面。但是API没有封装,不便移植。②Open GUI[2]采用LGPL条款发布。Open GUI比较适合基于X86平台的实时系统,基于汇编实现的内核并利用MMX指令提高运行速度。但可移植性稍差。③Microwindows/Nano X[3]是一个采用MPL条款的开放源码的项目。它的主要特色在于提供了类似X的客户/服务器体系结构,并提供了相对完善的图形功能。但它的图形引擎存在一些问题: 无任何硬件加速能力、低效算法和代码质量较差。④Qt/Embedded是跨平台C++图形用户界面工具包。因其面向对象、跨平台、界面设计更美观而得到广泛的应用。由于KDE等项目使用Qt作为支持库,所以由许多基于Qt的X Windows程序非常方便地移植到Qt/Embedded版本上。Qtopia是由TrollTech公司基于Qt/Embedded开发的第一个嵌入式窗口环境和应用程序,广泛应用于PDA、掌上设备和网络设施等。它包括全套的个人信息管理PM,如地址本、电脑、日程安排、MPEG、图像显示、浏览器等。

由于现在Qt/Embedded被广泛地应用于各种嵌入式产品和设备中,从消费电器(如智能手机、机顶盒)到工业控制设备(如医学成像设备、移动信息系统等),因此本文选择Qt/Embedded来开发GUI应用程序。

2 Qt/Embedded开发环境

Qt/Embedded开放了源代码,使开发人员可以在GPL许可协议下自由地使用它来开发嵌入式Linux应用系统。本设计使用了TrollTech公司的自由版: Qt/Embedded 2.3.7版本、Qtopia 1.7.0版本和Qt/X11 2.3.2版本。由于程序要运行在三星s3c2410的微处理器上,所以在Linux/X86上经过正确设置环境变量后要进行如下配置和编译: ./configure -xplatform linux-arm-g++ -no-xft -no-qvfb -depths 4, 8, 16, 32, 编译后将得到如下创建应用程序所需的工具包:

①designer是设计窗口组件的应用程序。使用它将最后生成file.ui文件和main.cpp文件。②uic是从XML文件生成代码的用户界面编译器。它的主要作

* 2007-11-29收到, 2008-04-03改回

** 杨中华, 女, 1979年生, 硕士研究生, 研究方向: 嵌入式系统及应用。

用是将file.ui文件生成file.h和file.cpp文件。③qmake工具是用来生成file.pro文件的;tmake是跨平台Makefile生成器,不管在PC环境还是交叉编译环境中都能生成相应的Makefile文件。

有了这些工具再加上编译器就可以进行应用程序的开发、编译和调试。

3 Qt/Embedded应用程序设计

我们实现的是一个具有娱乐与英汉电子词典功能的GUI系统,此系统的内核是采用2.4.18版的ARM-Linux内核,并在此基础上加以改写,运行在三星s3c2410微处理器上。本文重点介绍应用程序英汉电子词典的实现,词典主要有英汉互译、单词记忆、生词本、词库选择、帮助等五大功能。

Qt/Embedded在体系上是客户/服务器模型,任何一个Qt/Embedded程序都可作为系统中唯一的一个GUI Server存在。因此,Qt/Embedded应用程序执行过程如图1所示。

3.1 界面设计

主要介绍GUI系统中英汉电子词典的界面设计。为了用户操作简洁方便,本界面用Qt/Embedded的Tab设计了英汉互译、单词记忆、生词本、词库选择、帮助五大功能按纽。通过点击其中的按纽会出现各自的功能界面,以点击单词记忆按纽为例来说明使用Qt/Embedded设计的界面。在界面的上端设计了一个文本框用来显示文字内容,在文本框的下面设计了三个按纽用来选择单词,同时在界面中还设计了模式选择(其中包括手动选择和自动按时间设定选择)、记忆方记忆、正向检索和逆向记忆)、保存进度等式(其中包括顺序。具体如图2(实际界面缩小后抓取的图片)所示。

3.2 编程实现

①初始化。在Qt/Embedded应用程序中,首先是在main.cpp函数中创建QApplication类对象,QApplication类管理GUI应用程序的控制流和主要设置,其中包含主事件循环,来自窗口系统和其他资源的所用程序的初始化和结束,提供会话管理,并且处理大多数系统范围和应用程序范围的设置。

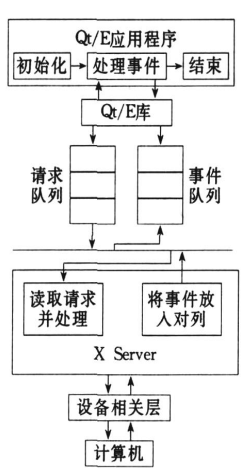

图1 应用程序的执行过程

②创建组件。在main.cpp中创建窗口对象main_form w,它是QDialog的继承类。在main_form构造函数中提供了典型的应用程序主窗口,另外在成员函数中加入了事件处理部分。在main_form.cpp中构造函数main_form。我们也可以使用工具designer来设计窗口组建的应用程序,使用它将最后生成file.ui文件和main.cpp文件。这样可以简化窗口设计。

③处理事件。Qt/Embedded的事件处理过程:当应用程序的main函数调用qApp->exec()时,应用程序进入Qt/Embedded的主循环事件,主循环事件从事件队列中取出本地窗口系统事件或其他的事件,把它们转换为QEvents,并使用函数QApplication::notify发送转化的事件给相应的对象QObiects。Qt/Embedded对象间的通讯是利用信号和槽机制,所有从QObject或其子类(例如Qwidget)派生的类都能够包含信号和槽。当对象改变其状态时,信号就由该对象发射(emit)出去,通知槽接收信号。这种机制真正地实现了封装的概念。在所有包含信号和槽的声明中都要加上Q-OBJECT。在信号和槽声明后,需要用QObject类的成员函数connect()将信号和槽连接起来,定义为:

connect (pushButtonWdRemPre, SIGNAL (clicked()), this, SLOT(slotSearchPre()));

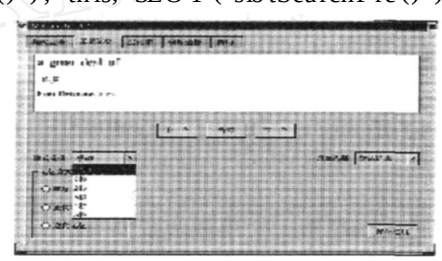

图2 界面图示

在实际应用中,Qt/Embedded要连接已经交叉编译好的sqlite数据库,连接好后通过界面来操纵数据库使其能实现这五个功能。这里只以点击单词记忆按钮功能中的上一个按钮为例来介绍槽函数的实现。具体实现过程如下:

```
void main_form::slotSearchPre()
{
  sqlite3 * db= NULL;
  int rc, i;
  int nrow = 0, ncolumn = 0; //数据表的行和列
  char * zErrMsg = 0, * * azResult;
    QString sql, search;
  rc = sqlite3_open("dict.db", &db);
    if(num>=2)
      num--;
    if( rc )
  {
    fprintf(stderr, "Can't open database: %s\n", sqlite3
_errmsg(db));
      sqlite3_close(db);
```

```
        exit(1);
        } //打开 sqlite
    sql.sprintf("SELECT Words, Phonetic, Explain, DIC_Name FROM Words,Dict_Type where Words.DIC_ID = Dict_Type.DIC_ID");
    //从数据表中查询
    sqlite3_get_table( db , sql , &azResult , &nrow , &ncolumn , &zErrMsg ); //返回查询结果指针
    qWarning("%d# %d# %d", nrow, ncolumn, num);
    textEditWordRMb-> clear(); //清屏
    for(i= 0; i< 4; i++)
    {
    if( (num > 0) && (num< = nrow) )
    {
    if(i= = 0)
    {
      QString str(azResult[num * ncolumn+ i]);
       str= str.replace(QRegExp("//"),"\n");
       textEditWordRMb-> append("< h2> "+ tr(str)+ "< /h2> ");
       } //显示选择的上个单词
       if(i= = 1)
       {
       if(strcmp(tr(azResult[num * ncolumn+ i]),"[]") != 0)
       {
    QString str(azResult[num * ncolumn+ i]);
       textEditWordRMb-> append("< font color= \"# ff0808\"> "+ tr(str)+ "< /font> ");
       } //显示选择的上个单词的音标
    }
    if(i= = 2)
    {
      QString str(azResult[num * ncolumn+ i]);
    str= str.replace(QRegExp("//"),"\n");
    textEditWordRMb-> append(tr(str));
    }//显示选择的上个单词的中文意思
    if(i= = 3)
    {
    QString str(azResult[num * ncolumn+ i]);
    textEditWordRMb-> append("From Dictionary: < font color= \"# 30bd26\"> "+ tr(str)+ "< /font> ");
          }//显示选择的上个单词的来源
       }
    }
    sqlite3_free_table( azResult );
    sqlite3_close(db);
    }
```

3.3 编译并添加应用程序到Qtopia

编译Qt/Embedded应用程序大致需要三步: a. 生成工程文件.pro; b. 生成Malefile文件; c. 使用Linux自带的make生成一个二进制的可执行文件, 具体编译过程在这里不再详述, 请参考在线帮助文档。下面介绍如何在我们已经安装好的Qtopia里添加我们编写的应用程序(Dictionary)。

①建立程序Dictionary的图标文件。制作一个32×32大小的PNG格式的图标文件, 将此文件存放在Qtopia/pic/inline目录下, 然后我们要用到qt-x11-free-3.3.3里的一个工具qembed, 将Qtopia/pics/inline下所有的图形文件转换成一个C语言的头文件, 此头文件包含了该目录下的图形文件的rgb信息。

②重新交叉编译qtopia(具体编译过程可参考在线文档[4])。

③建立应用启动器(.desktop)文件, 将其保存在$QPEDIR/apps/applications目录下, 具体可参考qtopia自带应用的.desktop文件。

④建立根文件系统。我们利用原有的qtopia.cramfs的根文件系统映象, 把我们新建的应用的相关文件添加到这个根文件系统中。首先我们要把qtopia.cramfs的根文件系统mount到我们工作机器上, 然后复制这个文件系统的内容到一个临时目录下, 这时我们可以看到根文件系统里的qpe安装目录, 接着把我们新建的应用的相关文件(包括启动器文件, 包含了图标的库文件libqte.so.*, 和应用程序的可执行文件)复制到qpe的对应的目录下。将生成的新的根文件系统烧写到s3c2410的FLASH根文件系统区。

⑤运行。重新运行qtopia, 就可以看到我们添加的应用图标。点击此图标, 就可以运行此应用程序。

4 结束语

嵌入式产品已经成为新的技术热点, 而嵌入式GUI是嵌入式Linux不可缺少的组成部分。Qt/Embedded和Qtopia在智能终端等嵌入式系统中得到广泛应用。结合应用程序实例介绍了基于Qt/Embedded的GUI应用程序开发过程, 并编译调试实现。Qt/Embedded是比较理想的GUI开发环境。

参考文献

[1] 任善全, 吕 强, 钱培德等. 一个基于Qt/Embedded的嵌入式Linux应用程序的实现[J]. 计算机应用与软件, 2006, 23(2): 105-107.

[2] 张 鹏, 张雪兰. 基于嵌入式Linux的GUI应用程序的实现[J]. 计算机应用, 2003, 23(4): 115-117.

[3] 王 策. 基于Linux的嵌入式系统开发. 计算机应用, 2002, 22(7): 54-55.

[4] Trolltech Online Reference Documentation. http://doc.trolltech.com, 2006-07-08.

[5] 齐 亮. C++ GUI Qt3编程[M]. 北京: 北京航空航天大学出版社, 2006.

[6] 倪继利. Qt及Linux操作系统窗口设计[M]. 北京: 电子工业出版社, 2006.