嵌入式 Linux 操作系统下 GUI 解决方案对比分析

在嵌入式系统领域，有不少 GUI 系统，如 QNX Photon MicroGUI 等，可是具体到嵌入式 Linux 领域又有哪些可供选择的 GUI 系统呢？

在嵌入式环境底下，GUI 系统的整体构架跟 PC Desktop 相去不远，例如绘图函数库、字型、事件处理等也都是嵌入式 GUI 系统所要面临的。但是嵌入式系统本身由于体积小、资源少的特点，所以在整体设计上必须较为严谨，必须考虑的条件更多，有时很像又回到了 Dos 下编制程序的年代，对于软件所占的存储量有时可以说是锱铢必较。

Unix 环境下的图形视窗标准为 X Window System（以下简称 X 标准），Linux 是类 Unix 系统，所以顶层运行的 GUI 系统是兼容 X 标准的 XFree86 系统。X 标准大致可以划分 X Server、Graphic Library（底层绘图函数库）、Toolkits、Window Manager、Internationalization(I18N)等几大部分(详细内容见链接)。

笔者认为，虽然 X 架构不错，但却不怎么适用于嵌入式环境，因为实际工作起来实在太过于庞大，因此许多嵌入式 Linux GUI 系统会把上述几点合并，甚至全部绑到一起，当然这样同时也会失去很多弹性与扩展功能，但为了适应于嵌入式系统，这也是一个解决问题的方法。本文下面就介绍一下现存的主流嵌入式 Linux 下 GUI 解决方案。

主流解决方案介绍

Qt/Embedded

Qt 是 Trolltech 这家商业公司所开发的一个跨平台 FrameWork 环境，在 X 环境下可以看作是一套功能完整的用户界面工具包，它采用类似 C++的语法，并且具备物件导向功能。跨平台的特性可以让使用 Qt 编写的软件，在 Microsoft Windows 95/98/2000、Microsoft Windows NT、MacOS X、Linux、Solaris、HP-UX、Tru64 (Digital UNIX)、Irix、FreeBSD、BSD/OS、SCO、AIX 等许多平台上执行。虽然是商业公司的产品，但是 Qt 走的却是开源路线，并遵循同样的游戏规则，提供免费下载，全部都是开放源代码，非商业用途亦采用 GPL 的版权宣告，著名的 Open Source “KDE” 项目便是采用 Qt 所开发的。

Trolltech 也针对嵌入式环境推出了“Qt/Embedded”产品。与桌面版本不同，Qt/Embedded 已经直接取代掉 X Server 及 X Library 等角色，所有的功能全部整合在一起。

Qt/Embedded 同样具有跨平台的特点，省掉了不少移植软件的功夫，这样的概念和 Java 十分接近。同时它还采用模块化设计，其最大的好处是有弹性，Qt/Embedded 号称最小可以缩到 800Kb 左右，最多可以长到 3Mb(for Intel x86)，这样的弹性也让 Qt/Embedded 更适合在嵌入式环境下生存。

另外，Trolltech 公司还推出了针对 PDA 软件的整体解决方案 QPE（Qt Plamtop Environment）。它从底层的 GUI 系统、Window Manger、Soft Keyboard 到上层的 PIM、浏览器、多媒体等，全部都考虑进去了。其主界面如图 1 所示 ，内部包括地址簿、计算器、世界时间、时间设置、记事本、终端、文件浏览器、帮助、媒体播放器、图像浏览器、文本编辑器等。

相对其他 GUI 来说， Qt/Embedded 应该说是肥美型的产品，功能丰富，但能消化它的东西只能是高端产品，32MB 内存是运行它的最小要求。因此，如果开发的产品不是高端的信息终端类产品，不应优先考虑 Qt/Embedded。

GtkFB 自从 Qt 推出了嵌入式版本之后，虽然 GTK＋并非商业公司所发展，但也加紧脚步推出了 GtkFB 方案，其宗旨就是要为嵌入式系统推出一套基于 GTK＋的 GUI 解决方案。与 Qt/Embedded 类似，GtkFB 也跳过 X 层直接与 FrameBuffer 沟通，因此也具有 Qt/Embedded 的几项优点，不过由于不是商业软件，在发展的速度上较为缓慢。

Microwindows

Microwindows Open Source Project 成立的宗旨在于针对体积小的装置，建立一套先进的视窗环境，在 Linux 桌面上通过交叉编译可以很容易地制作出 micro-windows 的程序。MicroWindows 能够在没有任何操作系统或其他图形系统的支持下运行，它能对裸显示设备进行直接操作。这样，MicroWindows 就显得十分小巧，便于移植到各种硬件和软件系统上。

然而 MicroWindows 的免费版本进展一直很慢，几乎处于停顿状态，而且至今为止，国内没有任何一家专业对 MicroWindows 提供全面技术支持、服务和担保的公司。

MiniGUI 是中国人做得比较好的自由软件之一，它是在 Linux 控制台上运行的多窗口图形操作系统，可以在以 Linux 为基础的应用平台上提供一个简单可行的 MiniGUI 支持系统。“小”是 MiniGUI 的特色，MiniGUI 可以应用在电视机顶盒、实时控制系统、掌上电脑等诸多场合。由于这是由中国人自己开发的 GUI 系统，所以 MiniGUI 对于中文的支持最好。它支持 GB2312 与 BIG5 字元集，其他字元集也可以轻松加入。

Pure X 架构

Tiny X Server 是 XFree86 Project 的一部分，由 Keith Pachard 先生所发展，而他本身就是 XFree86 专案的核心成员之一。一般的 X Server 都太过于庞大，因此 Keith Packard 就以 XFree86 为基础，精简了不少东西而成 Tiny X Server，它的体积可以小到几百 Kb 而已，非常适合应用于嵌入式环境。

以纯 X Window System 搭配 Tiny X Server 架构来说，最大的优点就是弹性与开发速度，因为与桌面的 X 架构相同，因此相对于很多以 Qt、GTK+、FLTK 等开发的软件可以很容易地移植上来。

虽然移植方便，但是却有体积大的缺点，由于很多软件本来是针对桌面环境开发的，因此无形之中具备了桌面环境中很多复杂的功能。因此“调校”变成采用此架构最大的课题，有时候重新改写都可能比调校所需的时间还短。

OpenGUI

OpenGUI 在 Linux 系统上已经存在很长时间了。最初的名字叫 FastGL，只支持 256 色的线性显存模式，但目前也支持其他显示模式，并且支持多种操作系统平台，比如 MS-DOS、QNX 和 Linux 等，不过目前只支持 x86 硬件平台。OpenGUI 分为三层：最低层是由汇编语言编写的快速图形引擎；中间层提供了图形绘制 API，包括线条、矩形、圆弧等，并且兼容 Borland 的 BGIAPI；第三层用 C++编写，提供了完整的 GUI 对象库。 OpenGUI 采用 LGPL 条款发布。OpenGUI 比较适合于基于 x86 平台的实时系统，跨平台的可移植性较差，目前发展较慢。

一点建议

综合上述 GUI 解决方案各方面的性能，归结起来有在国内有四种 GUI 较为适用，笔者对其做出推荐。

1. OpenGUI

由于基于汇编实现内核，并利用 MMX 指令进行了优化，OpenGUI 运行速度非常快。它支持 32 位的机器，能够在多种操作系统下运行，主要用来在这些系统中开发图形应用程序和游戏。由于历史悠久，OpenGUI 非常稳定，但是由于其内核用汇编语言实现，其内部使用的是私有的 API，所以其可移植性较差，可配置性也较差。

2. Qt/Embedded

这个版本的主要特点是可移植性较好。因为 Qt 是 KDE 等项目使用的 GUI 支持库，所以许多基于 Qt 的 X Window 程序可以非常方便地移植到 Qt/Embedded 版本上。因此，自从 Qt/Embedded 以 GPL 条款发布以来，就有大量的嵌入式 Linux 开发商转到了 Qt/Embedded 系统上，如韩国的 Mizi 公司。但是，由于它是基于 C++类库的，所以和其他 GUI 相比系统消耗资源较大。因此说 Qt/Embedded 是肥美型的产品，功能丰富，一般用于手持式高端信息产品。

3. MiniGUI 和 Micro-Windows 的比较

MiniGUI 和 MicroWindows 均为自由软件，但这两个系统的技术路线却有所不同。MiniGUI 的策略是建立在比较成熟的图形引擎之上，比如 Svgalib 和 LibGGI，开发的重点在于窗口系统、图形接口之上。而 MicroWindows 的开发重点则在底层的图形引擎之上，所以可以对裸显示器直接操作，而窗口系统和图形接口方面的功能还稍有欠缺。比如说，MiniGUI 有一套用来支持多字符集和多编码的函数接口，可以支持各种的字符集，包括 GB、BIG5、UNI-CODE 等，而 MicroWindows 在多字符支持上尚没有统一接口。

X Window System 的分层架构

● X Server

X Window System 架构上有一项特点是别的 GUI 系统所没有的，这个特点就是 Client/Server 架构，注意这里和一般我们所熟知的某某服务器（Server 端）跟 PC 端（Client 端）相连接的情形是不同的。惟一类似的是 X Window System 本身也是采用网路架构设计。具体而简单一点的说明就是，X Client 可以看作我们在 X 上执行的软件，X Server 则是负责显示及传递使用者输入事件（包括键盘、鼠标等硬件装置的输入）。

● Graphic Library

我们可以把一幅图案想象成有成千上万个细微小点所组成，这种小点的单位通常为 pixel，在同一平方单位里这些小点数越高图案就越清晰、画质就越好，也就是说分辨率或解析率高。事实上我们要设计的视窗当然不可能是这样一点一点地画上去的，这样太过浪费时间，基于这种观念我们就设计出高阶一点的函数来帮助我们解决这个繁琐的步骤，例如各类视窗编程里用到的画点、画线、画矩形、画圆形、画不规则形、上色等函数。透过这些函数是的程序设计者不用去管画一条线要几个点以及如何让显示器显示等林林总总低阶的工作，我们称绘图相关的一组函数库为 GUI 的基本 Graphic Library。

● Toolkits

有了点、线、面的函数之后，虽然已经去除了大半的重复无聊工作，但是就开发视窗程序来说，还是显得非常没有效率，怎么办呢？只有继续将构成视窗的抽象元件，如按钮、卷轴、组合框等各类控件抽离出来，重新定义一组更高阶的函数库，在配合上联系的语法函数就成立 Toolkits 这类的东西，目前流行的有 QT、GTK+等。

● Window Manager

有了 Toolkits，我们可以很轻松地建立视窗模块（X Client），但是每个视窗模块只负责自己模块内的事务，那么不同视窗间的沟通、协调，例如视窗的切换、放大、缩小等，就没有模块管理了，于是视窗管理员（Window Manager）就应运而生了。

● Internationalization

国际化通常是我们东方语系国家的人比较关心的议题，但是很多软件一开始都由西方国家所主导开发，因此这点常常受到忽略，这个问题牵扯的层面很多，上从语言的显示、输入、中止语言习惯，下到文字位元的处理，完整的解决是必须从头到脚彻底配合才能达成，只处理一半都只能说是一个蹩脚的系统。

随着东方国家使用 GUN/Linux 的人口越来越多，I18N（i-eighteen-letters-n 的缩写）也日益受到重视，目前底层 libc 部分已经有完整的支持，剩下来便是 GUI 系统的问题，由于处理双位元所耗的资源较大，西方国家主导的系统很多情况下，经过一些取舍，I18N 就被牺牲掉了，整体而言 Embedded Linux GUI 系统在 I18N 的程度通常都没有 PC 端的好，只有在需求时才会使用。