融合通信产品项目的界面库①

于碧辉[1], 王 娇[1,2]

[1](中国科学院 沈阳计算技术研究所, 沈阳 110168)
[2](中国科学院大学, 北京 100049)

摘 要: 随着科技的进步, 融合通信受到越来越多的关注, 涌现出了众多的融合通信产品. 如今用户越来越重视产品的用户体验. 而传统界面开发模式存在界面与业务逻辑耦合度高的问题, 无法满足用户体验要求较高的融合通信产品的界面开发需求, 结合 DirectUI 技术设计并实现了一款 DirectUI 界面库, 可实现界面与业务逻辑彻底分离, 介绍了界面库的架构设计和各个模块的设计与实现过程, 详细阐述了界面描述语言的制订与消息机制的设计, 最后结合实例验证了界面库的特性.

关键词: DirectUI; 融合通信; MFC; 界面库; 消息机制

UI Library Based Convergence Communication System

YU Bi-Hui[1], WANG Jiao[1,2]

[1](Shenyang Institute of Computing Technology, Chinese Academy of Sciences, Shenyang 110168, China)
[2](University of Chinese Academy of Sciences, Beijing 100049, China)

Abstract: Along with the progress of the science and technology, more and more attentions were paid to Convergence Communication. Therefore, many products based Convergence Communication emerged. Nowadays, UX(User Experience) becomes more and more important. However, the traditional develop-mode of UI interface has the problem that UI is tightly coupled with business logic, which can't meet the demands of the ICT products' UI interface. A UI Library based on DirectUI is proposed, which decouples the UI interface and business logic. The architecture design and each module of the UI Library are described. What's more, the formulation of the user interface description language and the design of the message mechanism are detailed explained. The characteristics of the UI Library are proved by an ICT instance.

Key words: DirectUI; ICT; MFC; UI Library; message mechanism

随着 Internet 的飞速发展, 融合通信已成为业界关注的热点, 尤其企业客户对融合通信产品的需求更加迫切. 融合通信能将多种多样的通信方式整合在一起, 并且实现和企业业务流程的对接, 切合了企业的内在需求. 为了提高企业的工作效率, 改善企业的运行机制, 融合通信产品必将被企业大规模应用[1].

如今用户体验成为一款软件产品成败的关键[2]. 用户体验良好的融合通信产品也更易受到企业的青睐, 良好的界面可以提升用户体验. 而传统的界面开发模式如 Windows API 和 MFC, 存在友好美观界面开发难度大, 开发效率低, 界面与业务逻辑耦合度高, 无法适应目前不断改变的界面开发需求等问题[3].

DirectUI, 意为"Paint on parent dc directly", 即控件是直接绘制到父窗口上的, 没有窗口句柄[4]. 基于 DirectUI 的界面库可以方便地实现美观的界面效果, 提高开发效率, 减少开发成本[5], 完美地解决传统界面开发模式存在的问题. 目前, 腾讯、百度、360 等公司对自己的 DirectUI 界面库采取保密政策, 不对外公开. 专业的做 DirectUI 界面库的公司, 其授权使用费颇高, 并且不方便根据融合通信产品的界面需求进行深度定制[6].

针对融合通信产品界面开发需求, 本文结合 DirectUI

① 收稿时间:2015-12-16;收到修改稿时间:2016-01-11 [doi:10.15888/j.cnki.csa.005280]

技术设计并实现了一款界面库——QFuilib，使融合通信产品的界面开发更加简便.

1　概述

QFuilib 采用三层架构设计，基础层提供了一些与界面相关的基础类(CQuiPoint、CQuiSize、CQuiRect)和简单容器(CStdStringPtrMap、CStdPtrArray 等)，减少了对外部库的依赖. 核心层实现的是 QFuilib 的核心功能，包括消息管理、窗口管理、渲染引擎、资源管理和脚本引擎 5 个模块. 控件层调用核心层提供的功能实现所有的虚拟控件供用户使用. 控件层也分为两层: 核心控件层和基础控件层. 核心控件层提供控件基类，基础控件层的控件均派生自控件基类，包括容器控件和基本控件. QFuilib 的架构图如图 1 所示.

图 1　QFuilib 架构图

2　各模块的设计与实现

2.1　基础层

为了减少对外部的任何库的依赖，在 QFuilib 内部实现了一些基础类，如图 2 所示.

图 2　基础类

CQuiPonit 是对 Windows 结构 POINT 的封装，结构 POINT 表示屏幕上的一个二维点; CQuiSize 是对 Windows 结构 SIZE 的封装，结构 SIZE 表示一个矩形的长度和宽度; CQuiRect 是对 Windows 结构 RECT 的封装，结构 RECT 表示一块矩形区域，包含一个矩形的位置和尺寸.

CStdString 表示字符串数组，封装了字符串的操作，如字符串的大小写转换、连接、比较、替换查找等. CStdPtrArray 表示指针数组; CStdValArray 表示数据数组; CStdStringPtrMap 表示字符串 map，使用了 times33 哈希算法，它主要用于 QFuilib 中各种资源的管理.

tinyxml2 是一款由 C++编写的轻量级的 XML 解析器，简单易用小巧[7]. QFuilib 使用 tinyxml2 解析 xml 文件.

2.2　核心层

2.2.1　窗口管理模块

该模块的功能是解析配置文件，动态生成相应的虚拟窗口及控件实例. 包括界面描述语言的制订和配置文件的解析两部分.

① 界面描述语言的制订

图形用户界面是由各种各样的控件组合而成的，这些控件组成了一个对象树，树的根节点只有一个，代表了整个界面，树的叶子节点代表了基础控件，树的中间节点代表了容器控件[8]. XML 具有语言无关性、自描述性和良好的扩展性并且具有天然的树形结构等特点，所以，QFuilib 选取 XML 来描述图形用户界面. 一个界面的对象树结构可以用图 3 来表示.

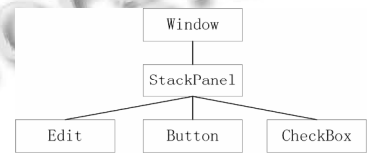

图 3　对象树

图形用户界面的控件一般具有一些基本属性，如名称、位置、高度和宽度等. 这些属性也需要使用 XML 来描述.

本文给出了一种用户界面描述语言[9]——QUIML. 其中根元素(Window)、容器元素(以 StackPanel 为例)、基本控件元素(以 Button 为例)的 DTD(文档类型声明)定义如下:

<!--Window 元素-->

<!ELEMENT Window (WrapPanel | StackPanel |TabPanel) + >

说明 Window 元素下可以包含至少任意一个容器元素.

```
<!--StackPanel 元素-->
<!ELEMENT StackPanel (Button | CheckBox | ComboBox | RadioBox | Edit | Label | ScrollBar |ProgressBar | SliderBar | SplitterBar | ListView | TreeView | LogoObj | PopupMenu | Calendar) * >
```

说明 StackPanel 元素可以包含零个或多个 Button、CheckBox、ComboBox、Edit 等基本控件元素. 将 StackPanel 元素替换为 WrapPanel 元素、TabPanel 元素即得到 WrapPanel 和 TabPanel 的 DTD 定义.

```
<!--Button 元素-->
<!ELEMENT Button EMPTY>
<!--公有属性-->
<!ATTLIST Button name CDATA #REQUIRED>
<!ATTLIST Button width CDATA #REQUIRED>
<!ATTLIST Button height CDATA #REQUIRED>
……
<!--Button 特有属性-->
<!ATTLIST Button normalimg CDATA #REQUIRED>
<!ATTLIST Button hotimg CDATA #REQUIRED>
<!ATTLIST Button pushedimg CDATA #REQUIRED>
<!ATTLIST Button focusedimg CDATA #REQUIRED>
<!ATTLIST Button disabledimg CDATA #REQUIRED>
……
```

每个元素除了具有公有的属性外, 还有自己特有的属性. 其他基本控件元素与 Button 元素定义格式类似.

使用 QUIML 描述界面的文档统称为配置文件. 每个配置文件都定义了一个窗口, 根结点必须是 Window, 表示窗口, 然后根结点内可以添加子结点. 各结点内可以添加属性. 配置文件的文件结构如下:

```
<?xml version=”1.0”?>
<Window>
  <[container]> <!--[container]容器控件标识 -->
    <[component]> <!--[component]基本控件标识-->
    </[component]>
  </[container]>
</Window>
```

② 配置文件的解析

QFuilib 使用 tinyxml2 解析器解析配置文件并创建对应的窗口及窗口上的各种控件. 设计的类如图 4 所示. 解析的步骤如下:

i. 加载配置文件. 调用 CXMLManager 类的 Load 方法将配置文件载入内存并在内存中生成 DOM 模型.

ii. 创建控件对象. 从 DOM 模型的根结点开始, 根据结点名字和属性, 创建对应的控件对象, 如 Button 元素对应生成 CQVButton 对象, 查看该结点有无子结点或兄弟结点, 若有则用同样的方法创建出它们对应的控件对象.

由于每个配置文件的根结点都是 Window, 表示窗口. 所以经过上述步骤即可创建出一个虚拟窗口.

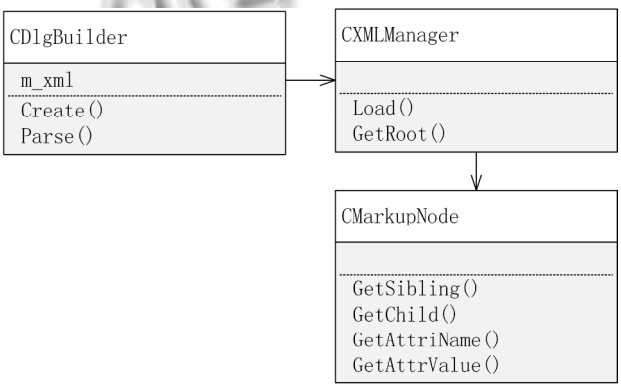

图 4 解析模块类图

2.2.2 消息管理模块

该模块负责处理消息. 程序中由::CreateWindow 创建出来的窗口为实窗口, 与之对应的是一个虚拟窗口, 是通过解析配置文件创建出来的, 无窗口句柄无法直接响应 Windows 消息, 通过窗口子类化技术[10]将消息派送到虚拟窗口的窗口过程函数进行处理. 消息流动过程如图 5 所示.

当一个 Windows 消息被派送到虚拟窗口的窗口过程函数时, 首先判断该消息是否与界面控件有关, 若与界面控件有关则获取此时鼠标的位置, 然后根据鼠标位置定位到基本控件对象, 然后调用 Event 机制对消息进行处理.

对于控件发送给逻辑程序的自定义控件消息通过 Notify 机制和自定义消息映射机制进行处理, 消息映射表如图 6 所示. 对于界面库不处理的消息则由系统默认处理函数处理.

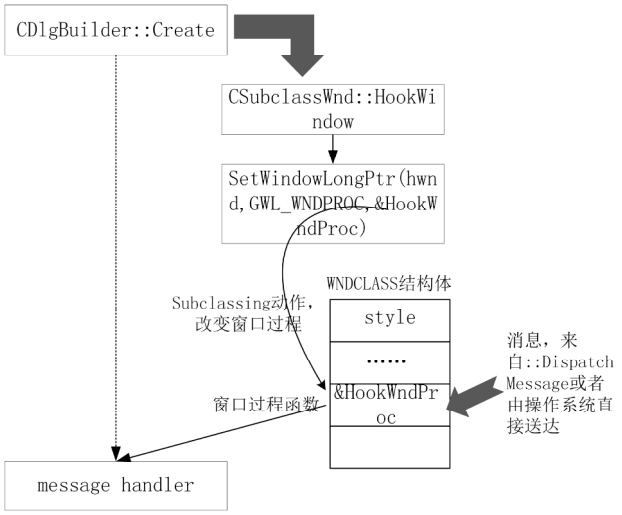

图 5　消息流动过程

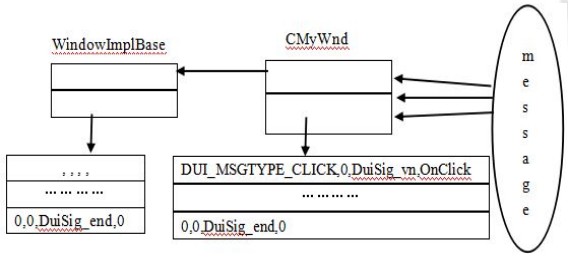

图 6　消息映射表

以用户左键单击按钮为例说明 Event 机制和 Notify 机制，流程图如图 7 所示.

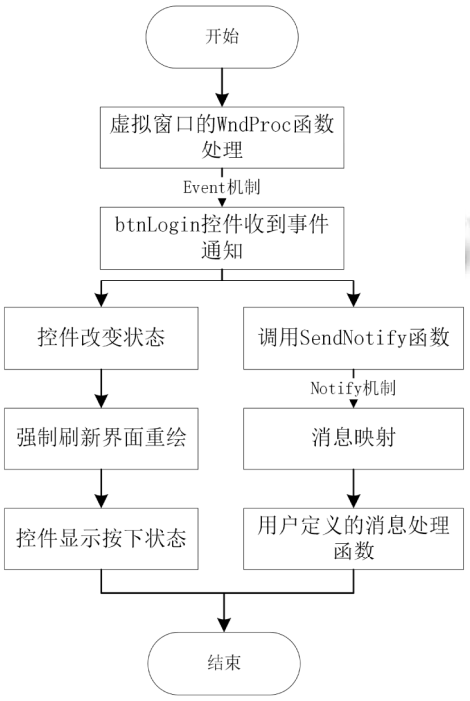

图 7　Event 机制和 Notify 机制流程图

2.2.3 渲染引擎模块

该模块的职责是进行渲染工作，将配置文件定义的界面显示出来，包括渲染图片，渲染区域和渲染文字.

为了对各种图像渲染引擎进行全面支持，设计了一个纯虚基类 IRenderDC，提供了绘制图片、区域、文字等的纯虚方法. 具体的渲染引擎，如 GDI 与 GDI+、OpenGL 等可直接继承 IRenderDC 并实现所有的纯虚方法. 类图如图 8 所示.

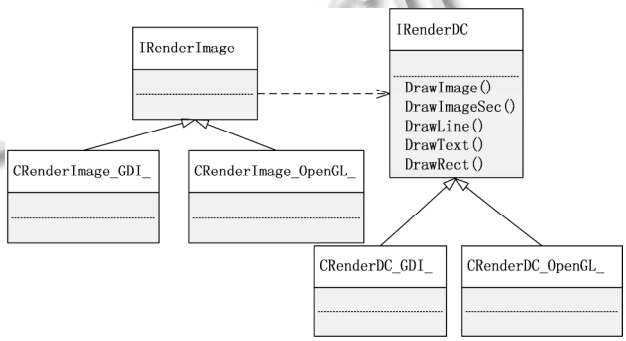

图 8　渲染引擎模块类图

2.2.4 脚本引擎模块

该模块负责提供能够使用脚本语言(Lua、javascript)来处理界面逻辑的功能.

通过自定义的 bindings-generator.py 脚本和 genbindings.py 脚本将 QFuilib 的函数注册进脚本环境里，这样就可在脚本文件中使用 QFuilib 的各种功能.

每个控件都有一个脚本对象名称(ScriptObjName)，通过该对象名称可以为该对象添加脚本代码. 示例如下:

```
//JavaScript 脚本——btnLogin.js
//绑定事件到登录按钮
btnLogin.AddEvent(WM_LBUTTONUP,”doLogin”,true)
//处理事件
function doLogin()
{
    //设置按下后按钮的信息
    btnLogin.SetText(“取消”);
    Invalidate();
}
```

在 QFuilib 中，设计了两个类来完成对于脚本文件的处理：CScriptEngine(脚本引擎)和 CScriptEngine Manager(脚本引擎管理器). 类图如图 9 所示.

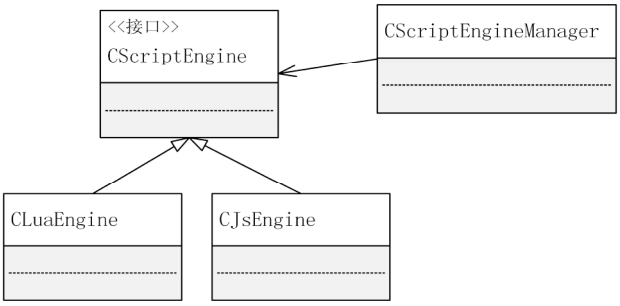

图 9 脚本引擎模块类图

程序中运用示例如下:

```
//取得 JavaScript 脚本引擎
CScriptEngine* pEngine = CJsEngine::GetInstance();
//设置脚本引擎管理器使用新创建的 JavaScript 脚本引擎
CScriptEngineManager::GetInstance()->setScriptEngine(pEngine);
//执行脚本文件
pEngine->runScript("btnLogin.js");
```

2.2.5 资源管理模块

该模块负责读取和管理应用程序引用的各种资源, 通过读取配置文件, 从文件中获取各种资源所需的信息, 再通过这些信息创建相关的真实资源, 并对这些资源进行统一管理. 为了实现最小的内存占用, 此模块采用了资源共享, 控件单实例等技术. 资源共享保证系统中不存在冗余的图片资源, 字体资源等.

2.3 控件层

核心控件层负责提供控件基类, 主要解决控件应该具有哪些属性和方法的问题. 基础控件层负责实现所有的控件, 分为两类: 容器控件和基本控件. 控件层的类图如图 10 所示.

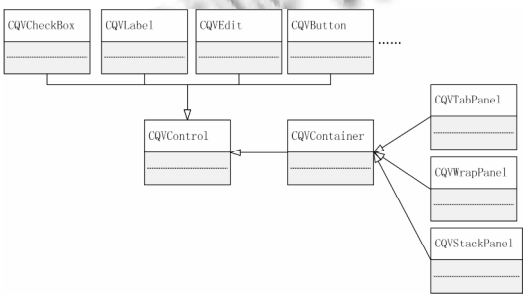

图 10 控件层类图

针对目前融合通信产品项目的界面开发需求, QFuilib 实现了 3 种容器控件: StackPanel(垂直布局容器控件)、WrapPanel(水平布局容器控件)、TabPanel(标签页布局容器控件); 实现了 15 种基本控件: Button(按钮控件)、Edit(编辑框控件)、ListView(列表控件)等. 其中部分控件根据融合通信产品的特点提供了相应的处理机制, 如 ListView 控件常常被用来实现融合通信产品中的企业通讯录, 如图 11 所示. 为避免出现数据量较大时出现内存、cpu 占用高和拖拽效率低的问题, QFuilib 的 ListView 控件提供了虚表机制, 使用定量的控件描述数量级很大的数据. 当 ListView 需要更新 Item 时, 抛送自定义 DUI_LV_UPDATEITEM 消息, 消息参数中包含需要更新的 Item 的数量和 Item 数组指针, 根据这些信息在消息处理函数中即可完成 ListView 的更新.

图 11 企业通讯录

3 实例及分析

QFuilib 已应用于自主研发的融合通信终端产品. 图 12 是融合通信终端产品的登录界面.

其部分配置文件如下:

```
<Window size="280,380" caption="0,0,0,30">
 <Font name="微软雅黑" size="12" bold="true"/>
 <StackPanel bkcolor="#fff0f2f3">
    ……
    <!--用户输入部分-->
    <StackPanel top="20">
      <!--用户名-->
      <WrapPanel left="30" childpadding="5">
       <Label padding="8,8,0,8" normalimg="user.png" hotimg="userHot.png"/>
```

```
        <Edit name=”user”/>
      </WrapPanel>
      ……
    </StackPanel>
    ……
    <!--登录按钮-->
    <WrapPanel top=”20”>
      <Button name=”btnLogin” left=”30” text=”登录”
     font=”0” align=”center” normalimg=”btnLogN.png”
    pushedimg=”btnLogP.png” hotimg=”btnLogH.png”/>
    </WrapPanel>
  </StackPanel>
</Window>
```

图 12　登录界面

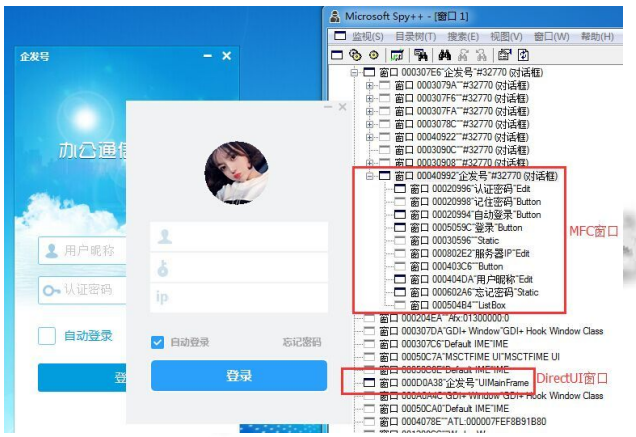

图 13　spy++结果

界面中有按钮、复选框、编辑框等控件，使用 Spy++查看窗口情况，结果如图 13 所示. 可发现整个程序只有一个 Windows 窗口，窗口句柄为 0x000d0a38. 验证了 DirectUI 的无句柄理念，控件都是虚拟控件. 而采用 MFC 的 Windows 程序中含有多个子窗口.

4　结语

为满足融合通信产品的界面开发需求，本文结合 DirectUI 技术设计并实现了一款界面库，并在融合通信项目中运用，验证了基于 DirectUI 的界面库具有灵活性高，能方便开发出友好美观的界面；安全性高，恶意程序无法获取窗口句柄，不易监控程序并产生危害. 本文给出的界面库还可以进一步完善，以满足更多的开发需求.

参考文献

1 武俊利.企业融合通信系统方案设计与实践[硕士学位论文].北京:北京邮电大学,2012.
2 张媛.互联网产品用户体验设计与评估研究[硕士学位论文].南京:南京航空航天大学,2014.
3 谭成基.DirectUI 框架开发与应用[硕士学位论文].广州:华南理工大学,2013.
4 逄晓东.基于 WTL 的 DirectUI 开发框架研究与发现[硕士学位论文].哈尔滨:哈尔滨工程大学,2013.
5 韦秋实.基于脚本语言的人机交互界面引擎的设计与实现[硕士学位论文].哈尔滨:哈尔滨工业大学,2012.
6 吴洙丁,冯运仿,邱阳,罗健,陈奇睿. DirectUI 技术的研究与界面库设计.福建电脑,2014(7):15–19.
7 王超,郑清.C++解析 XML 方法的研究和实现.电脑与电信,2012(5):66–67.
8 黄洪,林辉,王奔.一种图形用户界面的 XML 描述方法与工具开发.计算机应用与软件,2011,10(28):198–202.
9 杜一,邓昌智,田丰,任磊,戴国忠.一种可扩展的用户界面描述语言.软件学报,2013,24(5):1127–1142.
10 黄同.高级图形界面库设计方法.长春工程学院学报(自然科学版),2007,(1):55–58.