

文章编号: 2096-1472(2016)-01-35-04

组件开发架构及在继电保护配置软件中的应用

贺 敏¹, 陈宏君²

(1. 国网电力科学研究院, 江苏 南京 211106;

2. 南京南瑞继电电气有限公司, 江苏 南京 211102)

摘 要: 传统继电保护装置的功能主要由开发人员直接编写程序, 开发工作量大, 维护成本高, 需要提高开发效率, 为此开发了基于组件化、图形化配置软件。本文介绍了该软件的组件开发架构, 架构由主框架、组件库、事件总线、主程序组成。主程序读取组件配置文件, 加载和创建相关组件对象, 组件对应的主窗体放置在主框架的界面容器内。组件之间需要协同配合时, 通过发送和响应预定义的事件实现。在继电保护配置软件中应用了该组件架构, 降低了模块间的耦合关系, 提高了软件可扩展能力和开发集成效率。

关键词: 组件; 主框架; 事件总线; 继电保护软件

中图分类号: TP31

文献标识码: A

Component Structure and Its Application in Configuration Software for Relay Protection

HE Min¹, CHEN Hongjun²

(1. State Grid Electric Power Research Institute, Nanjing 211106, China;

2. NR Electric Co., Ltd, Nanjing 211102, China)

Abstract: To solve the problem of massive work and high maintenance cost raised by writing source code manually in traditional development of relay protection device, the paper introduces a modular and graphical configuration software which adopts component architecture. The architecture is composed of mainframe, component library, event bus and master program. The master program reads component configuration file to load and create corresponding component objects while main window of each component will be displayed in interface container of the mainframe. The components cooperate with each other by sending and answering predefined events. Application of the component architecture in configuration software for relay protection reduces coupling relationship between various components, while increasing software expansion capability as well as developing and integrating efficiency.

Keywords: component; mainframe; explorer; event bus; software for relay protection

1 引言(Introduction)

在人机界面软件的开发维护周期内, 由于面向的领域较为复杂, 软件的复杂度和规模在不断延伸, 要求软件系统具有较强的演化能力^[1,2]。这一方面要求开发人员能快速适应改变, 减少软件维护的代价, 另一方面又需要一种通用的方法来实现满足特定的需求, 确保个性化的需求实现不影响已经成熟稳定的模块^[3-8]。人机界面软件存在交互频繁、基于鼠标和键盘操作无规律触发、运行周期长等特点, 用组件模式开发时, 面临组件数据和界面管理、组件间交互配合等难点。本文设计了一种轻量级、组件开发集成架构, 通过跨平台库开发, 面向不同权限的用户, 可配置加载不同组件, 并在继电保护装置配置软件中进行了应用。

2 基于组件技术的软件架构(Software architecture based on component technology)

2.1 组件概念与组件系统原理

组件(Component)是一个黑盒子, 是可独立发布的二进制单元。组件对外的接口主要是API功能和端口, 其中API是组

件对外提供的功能, 端口表示组件内部调用外部其他组件功能的函数。图1是一种组件模型^[1]:



图1 组件模型

Fig.1 Component model

嵌入式装置是一个典型的组件系统, 例如继电保护测控装置由若干块CPU/DSP/IO插件组成, 不同的装置将具有独立功能的集成电路插件按照设计要求进行组装, 通过总线进行高速通信和有效协同工作^[9]。类比于物理装置, 软组件系统由总线、接口、组件三部分组成。组件通过预先定制的接口连接到总线上, 总线是一个总控程序(宿主程序), 负责组件之间的通信和互操作; 接口负责组件启动、初始化、设置、注销等功能; 组件实际上是普通的程序, 可由界面、数据、函数组成, 可以完成某种功能, 编译为DLL。在对组件的接

口定义了统一的规范后，系统投入运行时，用户可以根据自己的需要通过接口集成到系统中，也可卸载，而框架程序不用修改，如果总控程序正在运行，修改某个组件时，不用停止运行，通过这种技术，增强了系统的灵活性和扩展性，延长了软件系统的生命周期。

基于组件开发设计的重点包括：(1)实现对组件的管理，宿主程序可以搜索、加载、卸载组件和维护组件的状态。将这些功能放到组件配置树(组件容器)模块实现。(2)接口的设计和识别，接口是不同组件之间、宿主程序与组件之间的通信基础。宿主程序只有识别接口，才能加载组件。(3)资源的互斥和同步，多个组件的并行运行可能对申请资源产生竞争。(4)组件的设计和生成，组件系统的功能由组件实现，在设计时需考虑与组件相关的界面显示、数据处理能够高效进行。

2.2 组件接口设计

本文采用导出函数作为组件提供服务的接口，通过函数指针方式实现对象的映射和接口的调用。定义组件基类 Component，编译形成comp.dll。组件有初始化、运行、停止等接口，有请求服务的回调函数，有发送组件状态的信号，其UML模型如图2所示。

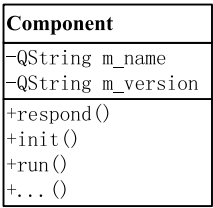


图2 组件的UML模型

Fig.2 Component UML model

其C++主要接口定义如下：

```
class COMP_DECLSPEC Component;  
public QObject {Q_OBJECT  
public:  
    Component ();  
    virtual~Component()=0;  
public://公共接口  
    virtual bool init(void*)=0;  
    virtual bool run()=0;  
    virtual bool stop()=0;  
    virtual void free()=0;  
signals://信号  
    void sigStarted(bool);  
    void sigFinished(int);  
    void sigRequire (void*,int);
```

```
public slots://回调函数  
    virtual void respond(void*,int);  
protected:  
    void* m_pdata;  
    QString m_name;  
    QString m_version;};
```

Component只是定义纯虚接口函数，具体的实现由各派生的组件细化。各子组件、顶层组件均继承自组件基类，继承的组件均须实现统一的组件构造接口：

```
extern"C"Component*  
newComponent();  
通过类似如下的实现方式，返回动态类型不同的组件实例。例如定值整定组件创建接口实现为：  
Component*newComponent(){  
    CSettingEditor*ptr=new  
    CSettingEditor();  
    return(Component*)ptr;}
```

2.3 组件配置树

一个软件系统的组件结构关系可用组合配置树 (Composition and Configuration Tree,CCT)来描述^[1]。形式化定义为一个三元组CCT=(N,E,R),其中：N表示树节点的集合；R是节点间语义关系的集合，R有组合关系和配置关系。R={Rcomp,Rconf}。配置关系通过包括OR(或选)、AND(并选)、Switch(多选一)、Multiplex(多选多)等关系。Rconf={AND,OR,Switch,Multiplex};E是节点之间的边的集合；E=Ecomp∪Econf，Ecomp表示组合关系边，Econf表示配置关系边。Econf=EAND∪EOR∪ESwitch∪EMultiplex以继电保护配置软件的代码生成子系统为例，其组合配置树如图3所示。

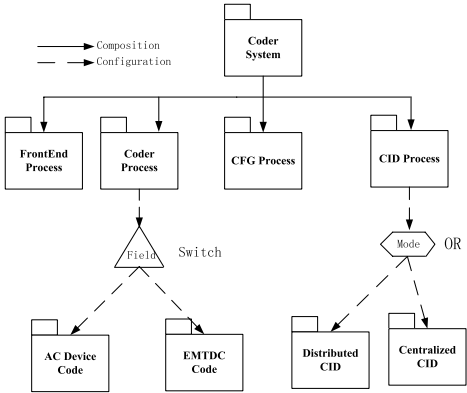


图3 组件配置树示例

Fig.3 Example of component configuration tree

在图3中，实线表示组合关系，虚线表示配置关系。

Coder系统可由多个主要组件构成,而其中部分组件也是由下层组件构成。三角形表示多选一配置关系,coder组件根据不同的应用领域,可选择对应交直流嵌入式保护装置code组件、EMTDC仿真code组件。六边形表示或选配置关系,CID的处理有直流装置分布式的配置建模组件和交流装置的集中式配置建模组件。可用XML文件描述该配置树关系,基于组件配置树实现软件系统的更新、维护,将修改的函数的影响尽量限定在某个组件的内部。

2.4 人机界面软件的组件架构

本文设计的人机界面软件集成架构包括:主程序、主框架、组件库、事件总线。

主程序是main.cpp文件编译后形成的可执行程序,仅包括一个main入口函数。在main函数中读取组件配置树文件,调用组件构造函数,形成组件对象实例,将组件对应的主窗体界面放入到主框架容器内。main函数还调用事件总线构造函数,将组件接口对象指针放入事件总线散列管理。

在main函数中创建主框架实例。主框架是一个通用的软件集成窗体,继承自QMainWindow,由菜单工具栏、左侧资源管理器、中间界面容器、右侧属性编辑器、底部输出栏五个区域构成,支持各顶层组件创建主框架的子菜单、图标、停靠栏内的标签窗体。在顶层主框架的界面容器内管理各顶层组件的主窗体界面,每个组件在主窗体容器内管理各自子界面。界面容器按照后入先出方式管理。主框架提供文件的新建、打开、关闭、保存操作入口,可显示、隐藏停靠栏。

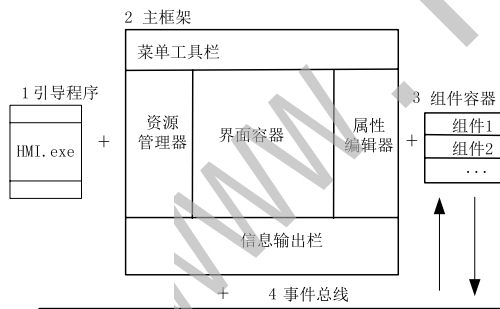


图4 人机界面软件组件架构

Fig.4 Human machine interface software component architecture

事件总线CEventBus是个全局对象类,各组件也可将事件发往循环处理队列中,总线周期定时触发后,将事件转发给目标组件。以保存文件为例,鼠标点击工具栏的保存按钮,工程管理组件响应该命令,调用发送接口,向所有组件群发保存:

```
CEvent*pevent=new CEvent
(getName(),"SaveData");
```

```
CEventBus::instance()->send("all",pevent);
delete pevent;
```

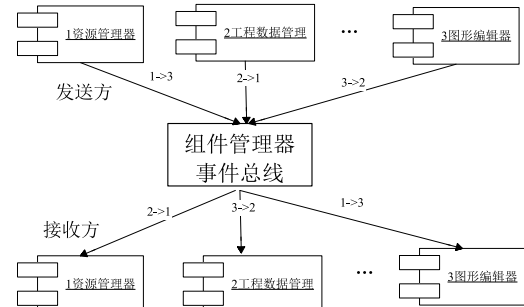


图5 事件发送机制

Fig.5 Event sending mechanism

如图5所示,事件总线串行发送,各组件接收到该命令后,依次进入活动状态,保存对应的数据。

3 继电保护配置软件设计(Design of relay protection configuration software)

随着IEC 61850智能变电站应用的深入,要求继电保护装置具备更灵活的可配置能力,基于手工编写代码和配置文件的开发方式,已经很难适应快速集成的需求^[10]。面向海外用户需要较强的可配置功能,例如根据现场应用增加减少开入开出插件、保护测控功能投入退出等,这都需要通过工具软件进行人机界面交互。

3.1 软件功能设计

继电保护配置软件是基于组件集成框架开发的,其输入为各类符号库,通过图形编辑实例化符号,并对符号进行编辑设置和连接输入输出关系,通过各个实例化的符号页面,生成变量库,变量库作为整个系统的数据中心,为各个应用服务配置提供数据源,对各类服务所需要的变量进行配置归类。各个应用服务的配置可以相对独立的加载和卸载,可以方便的扩充应用配置功能。主要功能如下:(1)装置硬件配置,从插件库中选择相关型号插件搭建装置,形成可编程的插件节点。(2)装置程序功能模块化配置,将继电保护程序进行模块化设计,形成可复用的元件符号库,在图形化页面中创建元件实例和进行数据连线,以可视化模块化的方式完成嵌入式装置程序组态。(3)装置HMI配置,从变量库中挑选信号,进行LCD层次菜单分组,包括模拟量分组、开关量分组、录波分组等功能。(4)装置定值分组和整定,包括定值的导入、导出、比较等功能。(5)装置系统选型配置,包括装置MOT选型、系统功能选择、保护测控功能模块投退。(6)在线调试功能。

3.2 软件组件划分

继电保护配置软件基于C++/Qt跨平台库开发,按照功能

可划分为：工程管理、离线配置、图形编辑、数据处理、在线调试五个部分，共分18个顶层组件：

(1)工程管理层：用资源管理(工作空间)组件展示，用工程数据管理组件维护数据响应操作。(2)离线配置层：包括基本信息配置、LCD层次菜单配置、录波等引用表配置、IEC61850配置浏览、LCD主画面编辑、集中式浏览、定值整定、系统功能配置共八个组件。(3)图形编辑层：包括可视化页面图形编辑器、符号库制作、符号库三个组件。其中图形编辑器可细分多种场景的子组件。(4)产物处理层：包括变量库、后端产物形成组件、驱动包处理(压缩、比较、升级)组件。(5)在线调试，包括可视化调试、Inspect瞬时录波等功能。

组件分层关系如图6所示。

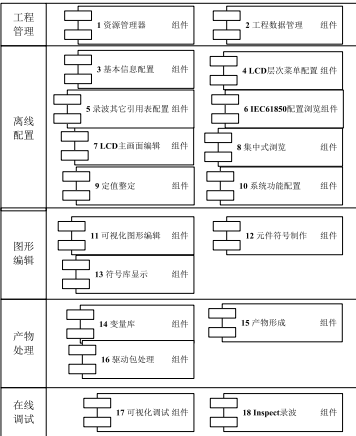


图6 配置软件结构划分图

Fig.6 Configuration software structure partition graph

以资源管理器为例，表1给出该组件的输入、输出、组件间交互设计要点。

表1 资源管理器组件

Tab.1 Explorer component

内容	描述
组件名	GuiExplorer.dll
功能概述	在左边停靠栏显示绘制工程及子节点；支持鼠标操作
输入	装置工程文件
输出	树形列表
交互	顶级节点绘制由工程数据管理组件创建；子节点由对应组件创建；鼠标操作时，在树形列表的响应函数中发送事件给总线，然后转发给所属的组件处理

例如：打开工程时只绘制装置节点，点击装置节点时，发送事件给工程管理组件，绘制装置子节点、加载显示变量库。

3.3 应用场景

基于组件化的软件架构，可根据不同用户角色(研发人员、工程人员、终端用户)，定制发布不同的组件库。图7是面

向研发人员的编辑场景，研发人员基于公共元件库，搭建图形化页面程序，之后在离线配置相关组件内完成HMI等人机接口配置。

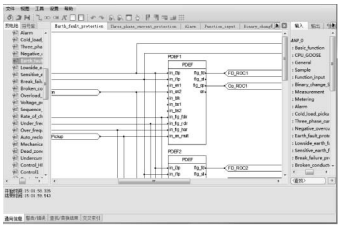


图7 配置软件应用场景

Fig.7 Configuration software application scenarios

4 结论(Conclusion)

本文介绍了一种人机界面软件的组件化框架，定义了组件的公共接口模型；设计了基于事件总线的交互方法。基于本文提出的组件开发架构在继电保护配置软件中进行了应用，该软件通过组件分层设计，将组件的交互协议和组件的内部实现细节分开，并支持部分升级，提高了软件开发集成效率。

参考文献(References)

[1] 陶传奇,等.基于模型的构件软件修改影响分析[J].软件学报,2013,24(5):942-960.

[2] 郁湧,刘永刚,侯江畔.一种基于构件的可信软件系统框架及其表示[J].软件工程师,2015,18(5):60-62.

[3] 谭海中.基于Java的Web通用DataTable组件设计[J].软件工程师,2015,18(7):22-23.

[4] 袁伟民,左春.基于样本程序的领域开发平台的研究与实践[J].计算机工程设计,2010,31(18):3979-3982.

[5] 张弛.软件组件接口扩展技术研究[J].微电子学与计算机,2007,24(8):35-41.

[6] 何鹏飞,等.组件技术在嵌入式系统中的应用[J].计算机系统应用,2014,23(6):220-223.

[7] 丁博,王怀民,史殿习.构造具备自适应能力的软件[J].软件学报,2013,9(24):1981-2000.

[8] 陈宏君,等.一种基于组件和脚本的可视化程序产物形成架构及应用[J].工业控制计算机,2013,26(12):1-5.

[9] 李响,等.新一代控制保护系统通用硬件平台设计与应用[J].电力系统自动化,2012,36(14):52-55.

[10] 陈宏君,等.新一代保护测控装置配套工具软件设计与应用[J].电力系统自动化,2013,37(20):92-96.

作者简介:

贺 敏(1972-),女,硕士,高级工程师.研究领域:电力系统及其自动化软件开发,新能源发电及控制.

陈宏君(1981-),男,硕士,高级工程师.研究领域:电力系统及其自动化软件开发.