第 35 卷 第 7 期 计 算 机 工 程 2009 年 4 月
Vol.35 No.7 Computer Engineering April 2009
·开发研究与设计技术· 文章编号：1000—3428(2009)07—0253—03 文献标识码：A 中图分类号：TP391.41

基于 JGraph 的变电站综合自动化图形组态系统

陈道达，曹冬林，李绍滋
(厦门大学智能科学与技术系，厦门 361005)

摘 要： JGraph 是一套使用 Java 开发的、兼容 Swing 的开源图形组件，具有丰富的图形操作接口和良好的可扩展性。该文基于 JGraph 实现一个跨平台的变电站综合自动化图形组态系统。与现有图形组态系统相比，该系统具有更好的自动处理能力，能实现电力系统设备的自动检测和接线图的自动生成，简化传统变电站图形系统在接线图设计方面的工作。

关键词： JGraph 组件；图形组态；跨平台；变电站综合自动化

Graph Configuration System for Substation Integrated Automation Based on JGraph

CHEN Dao-da, CAO Dong-lin, LI Shao-zi
(Department of Cognitive Science and Technology, Xiamen University, Xiamen 361005)

【Abstract】 JGraph is an open-source and Swing compatible graph component written in Java. JGraph provides a range of graph API and good extensibility. This paper implements a cross-platform graph configuration system of substation integrated automation based on JGraph, which is more automatic than the existing system. The feature of higher automation is proved by the function of automatic equipment recognition and automatic diagram design. It simplifies the diagram design of the tradition substation graph system.

【Key words】 JGraph; graph configuration; cross-platform; substation integrated automation

1 概述

近年来，图形组态技术被越来越多地应用于电力系统，国内外各大主要电力厂商，如西门子、ABB、南瑞等，相继开发了自己的图形组态系统。一些厂商完全自行开发底层的图形模块，另一些则基于 AutoCAD 等现有图形平台进行二次开发[1-2]。自行开发底层图形模块的代码编写量极大，其开发周期长且维护成本高。基于 AutoCAD 图形库的二次开发只能开发基于 Windows 平台的图形系统。而现代图形组态系统必须实现良好的跨平台性——能在 Windows 下正常运行的系统无须进行过多移植和修改，即可放到 Linux 或 Unix 下正常运行。目前国内组态系统的自动化程度通常不高，例如，设备相关的接线图依靠人工绘制，其工作繁重且容易出错，极大影响了整体效率。

针对上述 2 个问题，本文提出以 JGraph 作为基本框架，设计并实现一个跨平台图形组态系统。

2 JGraph的特点及其基本结构

2.1 JGraph的特点

JGraph 是兼容 Swing、基于 MVC(Model-View-Controller)体系结构的图形组件。Swing 是以 Java 为桌面应用程序开发的一套轻量级 GUI 组件。它为 Java 的跨平台特性提供了很好的支持，没有本地代码，不受操作系统的影响。Swing 实现了真正的跨平台应用，能提供本地窗口系统不支持的其他特性。JGraph 具有以下特点：(1)完全兼容 Swing，可以被结合到任何 Swing 的应用程序中。(2)简单高效的设计，JGraph 中应用了许多经典设计模式，如创建图元对象时使用的原型模式以及创建 UI 时使用的工厂方法模式。(3)运行效率高、实时性强。(4)纯 Java 语言编写，接口丰富、可扩展性强。

2.2 JGraph的逻辑结构

JGraph 组件是基于 MVC 设计模式的，其中，JGraph 继承于 JComponent，在整个应用程序中起中间桥梁的作用，关联 GraphModel, GraphLayoutCache 和 GraphUI；GraphLayoutCache 对应 MVC 模式中的 V，在前一个版本的 JGraph 中，该类称为 GraphView，它控制着所有图元的视图；GraphModel 对应 MVC 模式中的 M，用于存储应用程序逻辑数据；GraphUI 类是 JGraph 中的 User interface Control handling(UI)，负责用户鼠标、键盘的输入事件处理和消息转发[3]。JGraph 的逻辑结构如图 1 所示。

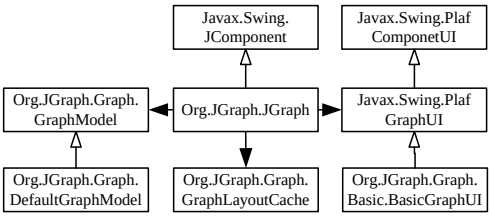

图 1 JGraph 的逻辑结构

2.3 JGraph的数据组织结构

JGraph 图元之间的关系基于数据结构中树的概念。JGraph 中的所有图元都是 DefaultGraphCell 的对象，DefaultGraphCell 继承于 Java.swing.tree.DefaultMutableTreeNode。因此，每个图元都是树的一个节点，图元之间的组合关系就是

基金项目： 福建省重点科技基金资助项目(2006H0037)
作者简介： 陈道达(1984－)，男，硕士，主研方向：电力系统自动化，自然语言处理；曹冬林，讲师、硕士；李绍滋，教授、博士生导师
收稿日期： 2008-11-08 **E-mail：** szlig@xmu.edu.cn

树的父子关系。当几个图元需要进行组合操作时，只要为这些图元创建一个共同的图元作为其父图元，并添加到原来的树中即可。在 DefaultGraphModel 中有一个 List 类型的变量 roots，roots 中存放所有没有父亲节点的图元，通过 roots 可以访问这颗树中的每个节点。

3 系统的设计与实现

3.1 系统整体结构

基于 JGraph 的变电站综合自动化图形组态系统整体结构如图 2 所示。

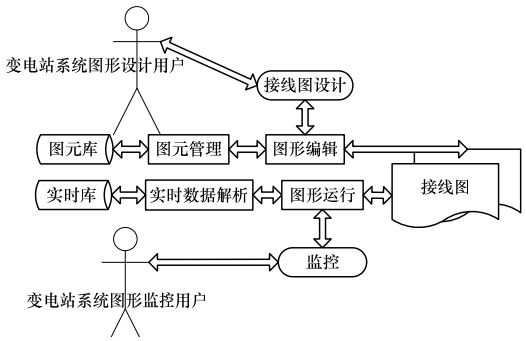

图 2 基于 JGraph 的变电站综合自动化图形组态系统整体结构

变电站综合自动化图形组态系统包括 2 个子系统：图形编辑子系统和图形运行监控子系统。在系统功能和系统结构上，图形编辑和图形监控是 2 个完全独立的模块，因此，上述划分可以使系统结构更清晰，减少系统的耦合性，并有利于不同职责分工的系统操作人员更好地使用系统。

在图形编辑子系统中，电力用户可以根据系统提供的图元库进行接线图的设计。接线图的设计包括：

(1)绘制基本的设备符号。在图形编辑过程中，支持更改、回退、重做、拷贝、粘贴、拖拽等操作；支持图元的组合、解组、对齐、均匀分布、相同大小等排列操作；支持图形的平移、缩放、导航(全局缩小)、漫游、鹰眼(局部放大)、旋转等功能。

(2)图元的动态属性设置。动态属性是指图元在监控状态下的动态效果。当设备数据满足一定条件时，会触发相应动态属性，使其显示在监控画面中。

(3)图元与设备对应关系设置。要求能编辑图元对应的设备，实现图元与设备数据的一致。

(4)接线图的自动生成。现代电力图形组态系统要求能在几秒内生成一个小规模站点的接线图。

图形运行监控子系统的主要功能是让用户可以通过监控软件监视并控制远端电力设备，具体如下：

(1)监视功能。用户可以实时观察图形的显示与动态变化、动态着色、潮流等。监控画面应反映厂站的实时运行状况，包括电气量实时值、设备运行状态、潮流方向等。

(2)控制功能，即用户对远端电力设备的实时控制。支持通过画面进行控制、挂牌、人工置数等各种操作，具体包括遥控、遥调、批量遥控、保护压板投退、保护定值整定、信号复归、序列控制、取反、挂接地线、拉闸、检修等。

(3)提供各种报警功能。

3.2 基本绘图功能设计

JGraph 中每个图元都关联一个图元视图类(CellView)，用于负责图元的显示。在图元视图类中，3 个最重要的内嵌对象是 Editor, Renderer, Handler，分别负责图元的文字编辑、显示状态以及移动和缩放功能的管理。图元的形状、大小、边框和颜色则于图元绘制显示前，在 Renderer 中进行设置。图 3 描述了系统绘图功能管理。

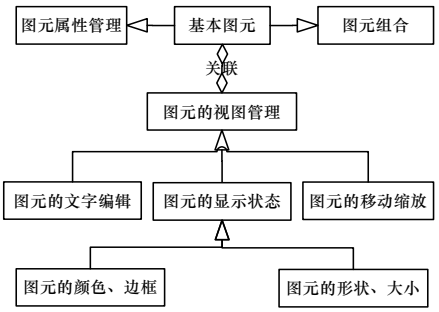

图 3 系统绘图功能管理

JGraph 提供了一些基本图形符号，如直线、折线、矩形、圆形等，但无法满足变电站图形组态系统的需求，例如没有扇形。因此，需要自定义更多的基本图形符号。有了基本图元后，可以利用图元的组合功能，绘制各种能表示电力系统设备的图形符号，并能将各种设备的图形符号保存在不同类别的图元库中。上述图元库以 XML 文件形式保存，下次使用时，可以直接从图元库中将各种图形符号拖到绘图面板上。

每个图元(基本图元或组合后的图元)都有很多属性，如颜色、边框、位置、大小等。属性的定义可以在 GraphCosntants 和 JGraphpadGraphCosntants 类中看到。例如，要实现旋转功能，就要为每个图元添加一个角度属性，并为每个图元添加一个在运行态中需要用到的动态属性。上述属性都保存到 GraphCosntants 的一个哈希表中，并通过 GraphCosntants 的静态函数 set 和 get 来访问。

3.3 图形与设备数据的关联

数据是应用软件分析的基础。电力系统应用软件需要的数据包括设备参数、拓扑结构、运行状态、动态特性等。在变电站系统的图形组态系统中，还需要图形数据。图形数据和实际设备的数据须采用合适的方式存储，并选择恰当的方式关联[4]。在本系统中，图形数据和设备数据分开存放。图形数据包括图元的各种静态和动态属性。静态属性是指图形的编辑态，如图元的颜色、图元的大小和位置以及组合关系等。动态属性是指图元将在图形运行态下显示的动态状态，包括闪烁、移动缩放、镜像旋转、颜色改变等。上述每个属性都作为一个成员变量保存在图元类中，并最终序列化到一个 XML 文件中。设备数据随系统的启动保存在实时库中，图形系统通过 socket 与实时库进行交互，进而通过共同的设备 ID 把图形数据和设备数据关联起来。该关联的好处是避免把所有数据都作为图元的属性进行保存，减少了图元属性的数量。设备数据只是作为更改图形显示状态的一种触发条件，与实际的图形显示函数不相关。

3.4 设备的自动识别设计

设备的自动识别是指发现有新设备添加到厂站时，图形组态系统能自动识别设备，并在监控画面中生成该设备的图形模型。自动设备识别是接线图自动生成的基础，只有在自动找到设备的前提下，才能绘制相应的接线图。

设备自动识别过程如图 4 所示。设备联网后自动发送数据到运行实时库的中心服务器，汇报自己的型号和状态后，中心服务器注册该设备。中心服务器通过套接字向图形组态系统发送通知，图形组态系统根据收到的设备型号和设备状

态在监控画面中添加设备的图形模型。

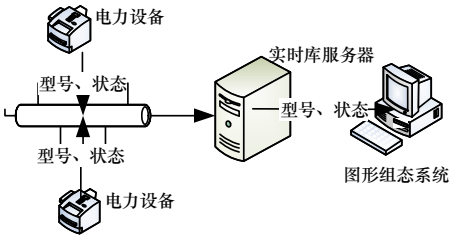

图 4　设备自动识别过程

3.5　接线图的自动生成设计

接线图的自动生成是在建站初期，依据站点的设备对应关系生成一个粗略的接线图，以减轻接线图设计的工作量。主要生成 4 种图：主接线图，监控图，网络连接图和装置组接线图。接线图自动生成基本流程如图 5 所示。

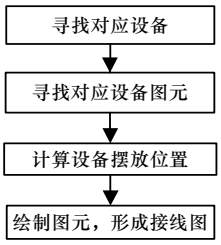

图 5　接线图自动生成基本流程

接线图自动生成基本流程包括 4 个部分：

(1)寻找对应设备。当用户指定生成某种接线图后，程序将自动从数据库中找到目前已注册的设备，并形成相关接线图的设备表。

(2)寻找对应设备图元。确定设备表后，需要从图元库中找到每个设备对应的图元。每个设备 d 是一个三元组(t, v, l)，其中，t 是设备的基本名称，如“开关”；v 是设备对应的电压等级；l 是设备所在段号。本文设备图元搜索算法如下：

1)获取设备 d 的三元组(t, v, l)信息。

2)搜索图元表中包含三元组(t, v, l)的图元，若存在该图元，则转 6)。

3)搜索图元表中包含三元组(t, v)的图元，若存在该图元，则转 6)。

4)搜索图元表中包含三元组(t)的图元，若存在该图元，则转 6)。

5)若中间出现错误，则输出错误信息并结束。

6)输出图元并结束。

(3)计算设备摆放位置。确定图元后，依据电力设备的排列规则，计算每个设备图元在图中的具体位置。每个设备对应的图元 g 是一个四元组(t, s, p, c)，其中，t 是设备图元名；s 是图元大小；p 是设备的位置信息；c 是设备的连接类型，描述与哪种类型的母线相连，有高、中、低 3 种值。需要计算每个设备图元的 p 属性值，其算法步骤如下：

1)获取设备表及其对应的图元表。

2)依据设备的 v 和 l 信息对设备进行排序。排序方式为按 v 进行排序，若 v 相同，则按 l 进行排序。对每个设备 d 进行步骤 3)~步骤 6)所述的操作。

3)获取设备 d 的三元组(t, v, l)信息及其对应图元 g 的四元组(t, s, p, c)信息。

4)搜索适应于该设备类型的电力排布规则。若没有对应规则，则输出错误信息。若只有一条规则，则按该规则进行排布。若有一条以上规则，则取使排布位置最大的规则。即 $p = \max(p_i)$，其中，p_i 是每条规则计算出的排布位置。

5)依据图元 g 的 c 信息找到对应的连接母线。

6)依据母线位置和步骤 4)所述排布规则得到的位置信息 p 对图元 g 进行相应的平移和旋转。

7)若所有设备计算完毕，则算法结束。否则转步骤 3)。

(4)绘制图元，形成接线图。补充图中缺失的设备连接线并按接线图的默认格式形成最终的接线图。

4　图形组态系统的运行效果与实时性

4.1　系统的运行效果

本系统主要应用在变电站综合自动化系统中，其良好的跨平台性和高度自动化给用户带来了很大方便。图 6、图 7 分别是图形组态系统编辑态界面和监控态界面，可以看出，其界面很友好。

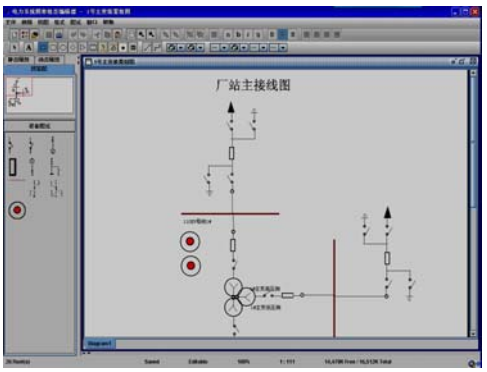

图 6　图形组态系统编辑态界面

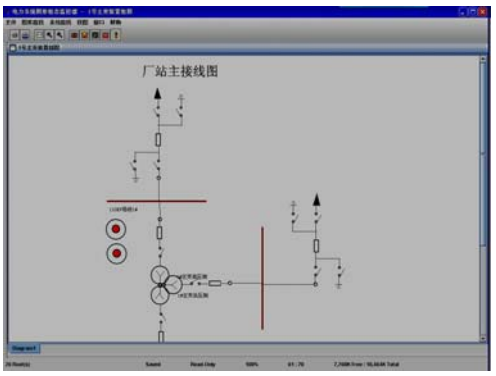

图 7　图形组态系统监控态界面

4.2　系统的实时性

多年来，实时性要求高的图形系统基本都使用 C, C++进行开发，因为它们在处理图形时的运算速度和效率较高。随着 Java 技术的不断发展，包括 JDK 的不断更新、虚拟机技术的不断进步以及 Java2D, Java3D 相继推出，目前，Java 技术在图形图像处理方面已具有较高性能。使用本系统进行实验，实验平台为 OS Fedora Core 2.4，CPU 为 P4 2.8 GHz，内存为 1 GB。实时性测试结果如表 1 所示。

表 1　实时性测试结果

图元个数	刷新时间/s
300	0.030
600	0.050
900	0.075
1 500	0.150

由表 1 可知，当 1 500 个图元需要改变状态重画刷新时，利用 JGraph 编写的程序只要 0.15 s 就能完成。其速度很快，

（下转第 276 页）