基于 ActiveX 和 XML 技术的 ATS 站场图设计与实现

张延龙，郭秀清
(同济大学 控制科学与工程系，上海 201804)

摘　要：ActiveX 控件和 XML 数据表是整个轨道交通列车监控(ATS)仿真系统的基础。针对 ATS 仿真系统开发的需求和现状并结合上海地铁 5 号线 ATS 仿真系统的实际应用，提出并设计了利用 VC2008 平台生成 ActiveX 控件进而生成站场图的方法。重点介绍了基于 XML 数据表的数据结构的设计与实现。

关键词：ActiveX；XML；站场图；数据结构
中图分类号：TP319　　**文献标识码**：A　　**文章编号**：1674-7720(2013)18-0010-04

The design and implementation of ATS station diagram based on ActiveX and XML

Zhang Yanlong, Guo Xiuqing
(Department of Control Science and Engineering, Tongji University, Shanghai 201804, China)

Abstract: ActiveX controls and XML database are the basement of Automatic Train Supervision (ATS) simulation system. According to the application of ATS simulation system on Shanghai metro line 5, this paper introduces the method of designing ActiveX controls on VC 2008 platform and the design of data processing module based on XML data table. The design of data structure based on XML is mainly introduced.

Key words: ActiveX; XML; station diagram; data structure

进入 21 世纪以来，随着中国经济的飞速发展和城市化进程的加快，城市轨道交通也进入大发展时期。我国已经成为世界最大的城市轨道交通市场。城市轨道交通的快速发展也带来了一个显著的问题，就是轨道交通设施一旦完工立即投入运行，根本没有时间允许对相关运营维护人员进行培训。此外，由于现场列车行车安全和时间等因素的限制，已经投入使用的列车控制系统等现场设备不可能用来教学培训。学员无法进行实际的练习。因此，怎样在保证安全的前提下使每一位学员系统、快速地掌握相关技术，就成为迫切需要解决的难题。鉴于此有必要开发一套完整的用于教学、培训的 ATS 仿真系统。

ActiveX 控件具有可扩展、可重用、易组合、语言无关等特点。将其应用在 ATS 仿真系统的设计与实现中可以大大减少重复劳动，缩短开发周期。可扩展标记语言 XML(Extensible Markup Language)是用于标记电子文件使其具有结构性的标记语言，可以用来标记数据，定义数据类型，是一种允许用户对自己的标记语言进行定义的源语言。XML 与其他数据表现形式最大的不同是：它极其简单。XML 的简单使其易于在任何应用程序中读写数据，这使 XML 很快成为数据交换的唯一公共语言。

1 ActiveX 控件的设计与实现

1.1 控件的设计

站场图是 ATS 仿真培训系统的基础，是 ATS 仿真系统可视控件的一部分。所有轨道设备的状态、进路生成状态、信号设备状态和列车运行状态都会在站场图上直接反应出来。通过对 ATS 仿真系统的分析，站场图的基本组成控件包括区段控件、道岔控件、信号机控件、站台控件、车次窗控件以及一些其他控件，如图 1 所示。

构建站场图控件的首要任务是设计控件的属性。站场图控件的属性可以分为两类：静态属性和动态属性。静态属性指的是在绘制站场图时可以修改的一些属性，站场图绘制完成后这些属性在 ATS 仿真培训系统运行时是不可改变的。动态属性指的是在绘制站场图时无需设置或仅需默认设置，在 ATS 仿真系统的运行过程中不断变化的属性[1]。例如，信号机控件的 ID、名称在 ATS 仿

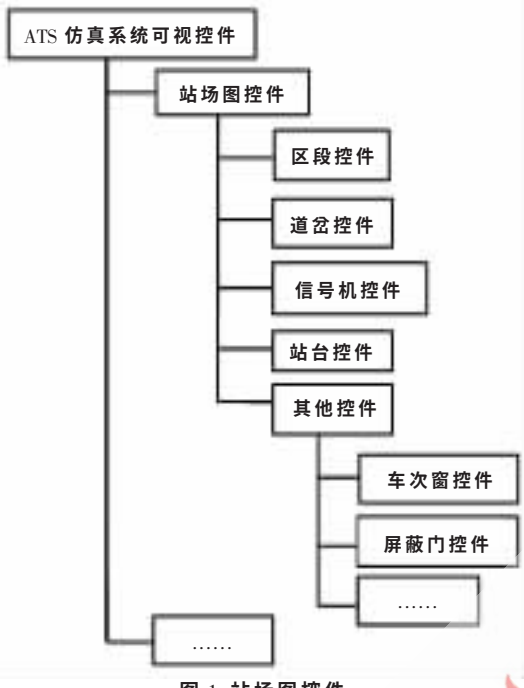

图 1 站场图控件

真系统运行过程中不会变化，这属于静态属性；信号机的背景颜色在程序运行过程中经常发生变化，这属于动态属性。

1.2 控件的实现

开发站场图控件使用的是 VC2008 开发平台，它是开发 ActiveX 控件的常用工具之一。VC2008 集成开发环境，使用了微软自己的类库 MFC，MFC 对开发 ActiveX 控件提供了全面的支持。MFC 对 ActiveX 控件的支持封装为 COlecontrol 类，站场图中各个控件均由此类派生[2]。本文以信号机为例说明 ActiveX 控件的设计与实现过程。信号机的静态属性有信号机名称、信号机位置类型和字体颜色，信号机的动态属性有信号机背景颜色、控件可见性和基座颜色。其主要代码如下：

```
class CSignalCtrl : public COleControl
{
… …
//静态属性
CString     m_SignalName;          //信号机名称
short       m_LocationType;        //信号机位置
COLORREF    m_FontColor;           //字体的颜色
//动态属性
COLORREF    m_SignalBackColor;     //信号机背景色
BOOL        m_Visible;             //控件可见性标志
COLORREF    m_VerticalColor;       //基座颜色
… …
}
```

在 VC++中设计 ActiveX 控件实际上就是对 OnCreate()、DoPropExchange()、OnDraw()等函数的处理。OnCreate()函数完成控件的创建以及控件结构、尺寸和字体等外形的设置。DoPropExchange() 函数负责的是状态永久性机制，利用这个函数可以把 ActiveX 控件的属性和内部信息保存到存储对象或者是流对象中。这个函数通常调用 PX_ family 函数来完成 OLE 控件的用户自定义属性操作。OnSize()函数用来调整控件显示窗口的大小和位置。OnDraw()函数利用指定图像在指定区域绘制 OLE 控件[3]。MFC 默认的控件边界形状是一个矩形。这个矩形可以由鼠标拖动而改变大小，所以在设计控件时应当根据默认矩形的上下左右边界计算出控件内部各点的相对坐标。这样在拖拽控件改变大小时控件内部线条按比例放大或缩小。

在 VC2008 编译环境中，信号机控件如图 2 所示。

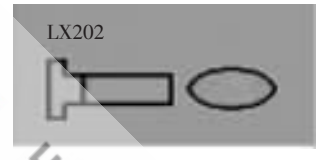

图 2 信号机控件

2 站场图的生成

2.1 控件注册

所有的 ActiveX 控件必须在注册之后才可以使用。Regsvr32 程序的作用就是注册 ActiveX 控件。将所有控件放入一个库中并在该库中建立一个名为 reg.dat 的 MS-DOS 批处理文件，在该文件中对控件进行注册。例如，对信号机进行注册的语句为：regsvr32 -s./Line5Signal.ocx。使用控件前运行该文件就可以将控件的所有信息(包括所在路径)都写入注册表中。

2.2 绘制站场图

ATS 仿真培训系统的站场图就是由一个个控件拼接而成的。按照轨道交通线路的实际情况，从控件库中选取合适的控件，然后配置控件的功能和数据，确定控件间的相互关系和通信方式，最终就可以生成一个站场图。本文按照上海市地铁 5 号线现场的实际情况分别绘制出 MV1、MV2、MV3 3 张站场图。考虑到项目接下来要在主视图中加载站场图，在为站场图所在的对话框添加类时要设置其基类为 CFormView。部分站场图如图 3 所示。

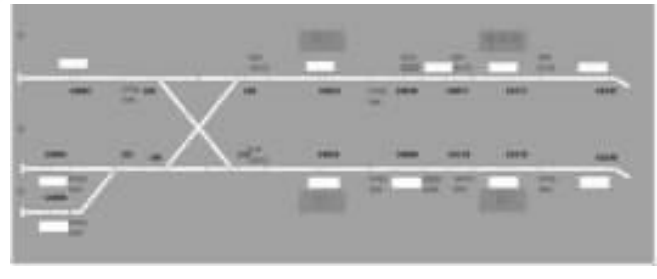

图 3 部分站场图

3 数据处理模块的设计

3.1 数据库的选择

可扩展标记语言 XML 是 W3C 组织于 1998 年 2 月发布的标准，是 Internet 环境中跨平台的、依赖于内容的技术，是当前处理结构化信息的有力工具。XML 文档结构严谨，层次分明，语义明确，具有良好的可读性、易编

写和易维护等特性，而且使得多媒体信息在不同的系统之间相互交流成为现实。

XML 可以充当小型数据库的功能。虽然 Access、Oracle 和 SQL Sever 等数据库都提供了强有力的数据存储和分析能力，例如数据索引、排序和查找等能力，但与这些数据库不同，XML 仅仅是存储数据。事实上 XML 与其他数据表现形式最大的不同是：它极其简单。这个优点使得 XML 与众不同。在 ATS 仿真系统中，由于站场元素间关系的复杂性和多样性，如果使用关系数据库，依据数据库设计范式而设计的数据结构会因为数据关系的复杂性而急剧膨胀，不利于存储一些结构松散和关系复杂的站场元素模型信息。另外，站场元素模型的数据量不大，此时 XML 简单的优点便发挥了用处，因此将站场元素模型信息以 XML 原始格式存储，方便灵活，没有版权，没有约束，也可以节约成本[4]。在站场图模块中，用 XML 描述了进路、区段、信号灯、道岔等站场元素的模型，记录它们的信息，在后面各个功能模块设计中都是基于这些数据的。

3.2 数据结构设计

因为静态数据包括基本信号点之间的逻辑关系，所以在静态数据模块中，不仅包含该信号点的基本信息(如类别、名称、长度等)，还应包含其他的信号点信息(如左边设备的名称、右边设备的名称)。因此在设计信号点基本数据时，每一个信号点都应包含完整的信息字段，例如区段字段，应该包括 ID、名称、左连接设备名称、右连接设备名称、占用标志和区段长度等。

XML 文件分为 Axle、Cross、Platform、Signal、Switch、TrainNumWnd、Timetable 和 Routes 这 8 个文件。其中前 6 个文件存储的是控件的位置信息，Routes 文件放置的是站场图中的进路信息，Timetable 是列车的时刻表信息。

3.2.1 信号机数据结构设计

以信号机为例，信号机主要包含 5 个字段：信号机 ID、信号机名称、接近区段名称、第一区段名称和所属集中站，其 XML 文件的结构如下：

```
<Signal>
    <ID>S1</ID>                    //信号机控件 ID
    <Name>X108</Name>              //信号机控件名称
    <JJQD>G0003</JJQD>             //信号机控件的接近区段
    <FirstQD>108</FirstQD>         //信号机控件的第一区段
    <JZZID>1</JZZID>               //信号机控件所属集中站
</Signal>
```

XML 数据库中每种类型的数据读进内存以后必须有相应的数据结构进行存储。同时该结构还要与界面上的具体控件相关联。为此以各个控件的原始类为基础添加上相应的位置和状态属性设计了控件的封装类。在 VC 程序中可以利用 CMarkup 类对 XML 文件进行解析，用解析出来的各个控件的信息构造相应的封装类，控件原始类在封装类中作为一个具有 public 属性的成员变量。在程序中数据的存储是以 C++ STL 中 vector 容器的形式存储。使用 vector 容器存储数据不但可以减少内存泄漏的危险而且可以借助于 vector 自身具有的属性方便地查找和设置元素。信号机控件的封装类如下：

```
class Signal
{
……
public:
    CLine5Signal *m_pSignalCtrl;        //绑定一个信号灯变量
public:
    CString m_ID;
    CString m_Name;
    CString m_JJQD;
    CString m_FirstQD;
    CString m_JzzID;
    BOOL m_SignalLock;                          //锁闭标志
    BOOL m_SignalARSFlag;          //单个信号的 ARS 功能
                                    开关标志
    BOOL m_SignalOpen;             //信号灯开放与否标志
    BOOL m_GuideFlag;              //信号灯引导标志
    BOOL m_DefaultMode;            //信号灯默认模式
    ……
public:                                         //故障标志
    BOOL m_fault1;                 //红灯主灯丝故障
    BOOL m_fault2;                 //红灯主副灯丝故障
    BOOL m_fault3;                 //红灯状态良好
    BOOL m_fault4;                 //绿灯主灯丝故障
    BOOL m_fault5;                 //绿灯主副灯丝故障
    BOOL m_fault6;                 //绿灯状态良好
    BOOL m_fault7;                 //无有效状态数据
    ……
    ……
}
```

信号机实体对象与界面控件的绑定是通过 DDX_Control()宏完成的。例如信号机控件 IDC_S1 与信号机实体对象的绑定通过 DDX_Control (pDX,IDC_S1, *pDoc->m_SignalPtrArray[0])完成。DDX_Control()是 MFC 中的宏，主要负责逻辑变量与界面控件的绑定操作。pDX 是指向 CdataExchange 对象的指针，IDC_S1 是控件 ID，*pDoc->m_SignalPtrArray[0]是内存中的信号机对象。

3.2.2 时刻表数据结构设计

时刻表实现了行车组织的工作计划编排。时刻表中包含一个运营日中列车运行的所有信息。时刻表是后续列车运行调整模块的核心，在生成站场图的部分只需知道它的基本数据结构即可，其数据结构包括：车次号、站台号、到达时间、出发时间，其 XML 文件结构如下所示：

```
<TimeTableLists>
    <Name>时刻表 1</Name>
    <TimeTable>
```

```
        <ID>10330Z</ID>                          //车次号
        <Record>
            <Platform>DCL2</Platform>            //站台号
            <Arrival>05:33:50</Arrival>          //到达时间
            <Departure>05:34:25</Departure>
                                                 //出发时间
        </Record>
        ……
    </TimeTable>
</TimeTableLists>
```

3.2.3 进路搜索原理

进路的生成主要是通过搜索信号机来完成的。具体过程如下：从这个信号机关联的轨道出发，沿着信号机的方向搜索轨道链，若遇到与斜股同向的道岔，则将该道岔放入一个搜索栈中，然后沿着直线方向继续搜索，直至找到反向的敌对信号机或下一车站的同向信号机；若此时搜索栈中仍有道岔，则取出道岔从斜股的方向按上面的描述搜索另一条进路，直到搜索栈中没有道岔对象。在搜索的同时即记录下相关控件对象信息。以图 2 中的信号机 X108 为例，一共有两条进路，一条进路是 X108-X106，另一条是 X108-X110。这两条进路在 XML 中按照如图 4 所示的数据结构表示。

(a) 进路一　　(b) 进路二

图 4 两条进路在 XML 中的表示

程序开始运行时先把保存在 XML 表中的控件封装类信息读到内存中，并以合理的数据结构存储起来，这样就不用频繁地读取数据库，能大大减少因读取数据库而占用的时间。程序从 XML 中读入控件的位置和连接信息，利用构造函数设置其状态信息。程序运行时也可以通过访问封装类的 public 成员变量设置其状态信息。结合时刻表和进路信息就可以使列车运行起来。

本文针对上海市地铁 5 号线 ATS 培训系统中的站场图进行设计，构建了用于拼接站场图的道岔和信号机等控件，并使用 XML 文件对控件位置和拼接信息进行存储。控件的划分与应用将站场图的绘制过程简化为简单的拼图操作，大大缩短了整个 ATS 仿真系统的开发周期。合理的数据结构极大地保证了程序的快速平稳运行。本文对于 ATS 仿真系统的后续研究有着铺垫作用。

参考文献

[1] 王野，郭秀清.基于组件技术的列车自动监控仿真系统开发平台[J].计算机应用，2007，2(z2)：286-288.

[2] 庄传平，陈永生. 可复用的列车自动监控仿真系统组件的设计与实现[J].城市轨道交通研究，2007(7)：31-33.

[3] 郭永瑞，孙明德.ActiveX 控件的编写和使用[J].计算机与信息技术，2007(22)：48-49.

[4] 李根，李彦明，刘成良.基于可扩展标记语言的故障模型表述[J]. 机械制造与自动化，2012，41(04)：115-117.

(收稿日期：2013-06-26)

作者简介：

张延龙，男，1988 年生，硕士研究生，主要研究方向：城市轨道列车运行仿真。

郭秀清，女，1965 年生，副教授，硕导，主要研究方向：过程控制与计算机控制。