铁路计算机应用
RAILWAY COMPUTER APPLICATION

基于XSD的配置工具数据处理安全平台

黄　健，李　倩，李方晴，张奕男

XSD based data processing security platform for configuration tool

HUANG Jian, LI Qian, LI Fangqing, and ZHANG Yinan

引用本文:
黄健, 李倩, 李方晴, 等. 基于XSD的配置工具数据处理安全平台[J]. 铁路计算机应用, 2023, 32(7): 62-67.
HUANG Jian, LI Qian, LI Fangqing, et al. XSD based data processing security platform for configuration tool[J]. Railway Computer Application, 2023, 32(7): 62-67.

在线阅读 View online: http://tljsjyy.xml-journal.net/2023/I7/62

文章编号：1005-8451（2023）07-0062-06

基于 XSD 的配置工具数据处理安全平台

黄　健，李　倩，李方晴，张奕男
（卡斯柯信号有限公司　平台技术中心，上海　200071）

摘　要：为城市轨道交通信号系统设备的用户配置工具开发了可通用的、基于可扩展标记语言模式定义（XSD，XML Schema Definition）的配置工具数据处理安全平台，该平台通过类生成模块与通用数据处理模块的相互配合，完成信号系统配置数据的输入/输出处理，用户配置工具可方便地进行数据的读/写操作，提高用户配置工具的开发效率。

关键词：轨道交通；安全平台设备；配置工具；可扩展标记语言模式定义（XSD）；通用数据处理方法

中图分类号：U231.7 : TP39　**文献标识码：**A　**DOI：**10.3969/j.issn.1005-8451.2023.07.12

XSD based data processing security platform for configuration tool

HUANG Jian, LI Qian, LI Fangqing, ZHANG Yinan
(Platform Technology Center, CASCO Signal Co. Ltd., Shanghai　200071, China)

Abstract: This paper developed a universal and extensible XML Schema Definition (XSD) based data processing security platform for user configuration tools of urban rail transit signal system equipment. The platform completed the input/output processing of signal system configuration data through the cooperation of class generation module and general data processing module, made it convenient for user configuration tools to read/write data and improved the development efficiency of user configuration tools.

Keywords: rail transit; safety platform device; configuration tool; XML Schema Definition (XSD); general data processing method

城市轨道交通信号系统通常包含多个执行不同功能的子系统/设备，以基于通信的列车运行控制系统为例，它包括车载列车自动保护子系统、区域控制器、轨旁设备等[1]。需要将车站设备配置数据烧录到系统/设备硬件存储板卡中，才能使这些子系统/设备正常运行。目前，信号系统的各个子系统/设备一般都有配套的工具用来生成配置数据，如区域控制器数据配置工具、轨旁安全平台配置工具等[2-3]。丰富的配置工具一定程度上提升了配置数据的效率，但存在以下问题。

（1）有的配置工具采用直接读取解析特定版本输入文件的模式[4]，一旦输入数据量增大，解析难度及代码量都随之增加。当每轮迭代输入结构也发生变动时，需要花费大量时间修改数据的解析，影响配置数据的进度。

（2）配置工具输出的数据文件未定义通用模板，导致每份输出文件都需要搭建对应代码框架才能实现相应结构的文件输出功能[5-6]，代码耦合度高，难以复用，同时，后期维护也相对困难。

（3）不同的配置工具，其输入/输出处理代码仅适用于专有的数据配置项目，难以通用。

为此，本文借助可扩展标记语言（XML），开发了可通用的、基于可扩展标记语言模式定义（XSD，XML Schema Definition）的配置工具数据处理安全平台，该平台也称为通用数据文件分析与可视化（CAVA，Common Data File Analytics and Visualization Architecture）平台。CAVA 平台将用户配置工具需要处理的输入/输出文件中结构和字段信息映射配置到对应的 XSD 文件[7-9]，并通过类生成模块将这一系列 XSD 文件解析，生成包含元数据的 Class 文件，将其打包为可供用户配置工具调用的 Jar 包；用户配置工具调用包含输入/输出结构信息的

收稿日期：2022-09-21
作者简介：黄　健，工程师；李　倩，高级工程师。

Jar 包和 CAVA 平台的通用数据处理模块后，即可方便地进行数据的读/写操作，提高配置工具的开发效率。

1　平台组成

CAVA 平台使用 JAVA 语言开发，包含 2 大模块，分别为类生成模块和通用数据处理模块，2 个模块之间独立运行。CAVA 平台组成如图 1 所示。

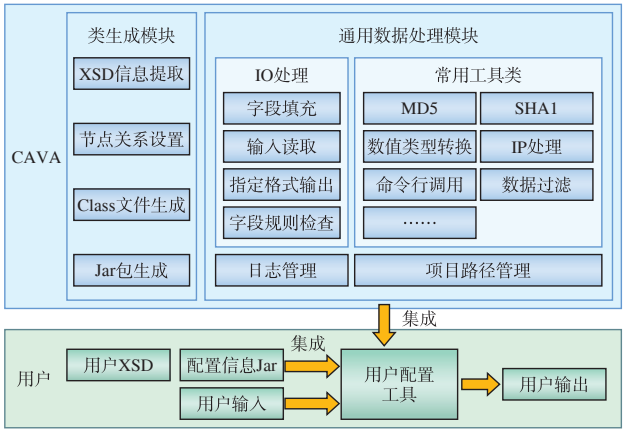

图1　CAVA 平台组成

1.1　通用数据处理模块

该模块主要实现输入文件读取、字段的填充与规则检查、输出指定格式文件，其他如常用工具类、日志管理和项目路径管理则提供了处理输入/输出过程中常用的功能接口。

本文将以 INI 格式协议文件为例，简述该协议配置文件输入/输出处理。

图 2 为 INI 格式文件的一般结构，由 [LOCAL_NET_1] 作为起始行，下面会有一组或多组如 IP = 10.2.2.5 的键值对，每组键值对占一行，它们构成了一个通常由多个节点组成的节点文件。

文件中仅包含一组结构与 [LOCAL_NET_1] 相同的节点，称之为复杂类型单节点；包含多组结构如 [REMOTE_NET_2] 的节点，称之为复杂类型列表，它们内部每一行键值对类型均为简单类型单节点。[REMOTE_NET_2] 中出现的一组或多组带有_0 或_1 后缀键值对称之为子序列，其实质是一个嵌套在 [REMOTE_NET_2] 中的复杂类型列表。正式的输入

```
[LOCAL_NET_1]
IP = 10.2.2.5
MASK = 255.255.0.0
PORT = 60013

[REMOTE_NET_2]
PERIPHERAL_ID = 1027
EQU_TYPE = 1
IP_0 = 10.3.2.5
PORT_0 = 60000
IP_1 = 10.3.2.6
PORT_1 = 60001
...
```

图2　INI 格式文件结构

输出处理流程，分为以下 3 小节。

1.1.1　读取输入文件

INI 格式文件读取接口函数如下：

iniNode iniNode = new iniNode（参数 1，参数 2，参数 3…）

该接口接收多个传递参数，前 3 个主要的参数分别为 INI 文件的绝对路径、区分文件类型的前缀、文件中的节点结构查找表（Map），创建成功返回的 iniNode 数据对象内包含一个将协议文件内的节点按照一定的规则匹配并分类填充存储的 Map。

1.1.2　字段的填充与规则检查

得到协议文件数据对象 iniNode 后，使用类生成模块生成的 Jar 包中对应该协议文件的接口，Jar 包中 INI 格式协议文件的接口调用如下：

Cprotocols protocol= new Cprotocols（iniNode）

该接口以文件读取接口返回的对象作为传递参数，接口创建后会调用 fill 接口依次处理协议文件对象中的子节点，处理顺序类似于深度优先算法的搜索模式。

协议文件的 Cprotocols 接口填充流程示意如图 3 所示，以填充协议文件子节点复杂类型列表 PRO_list 为例，需要经过以下 3 个步骤。

（1）调用 PRO_list1 对象的 fill 接口，依次处理内部两个简单类型单节点填充。

（2）network 为 PRO_list 内部嵌套的复杂类型列表，从 iniNode 数据对象中获取当前处理的 PRO_list1 节点下的 network 数据列表，依次遍历 network 数据列表并调用 network 的 fill 接口填充，填

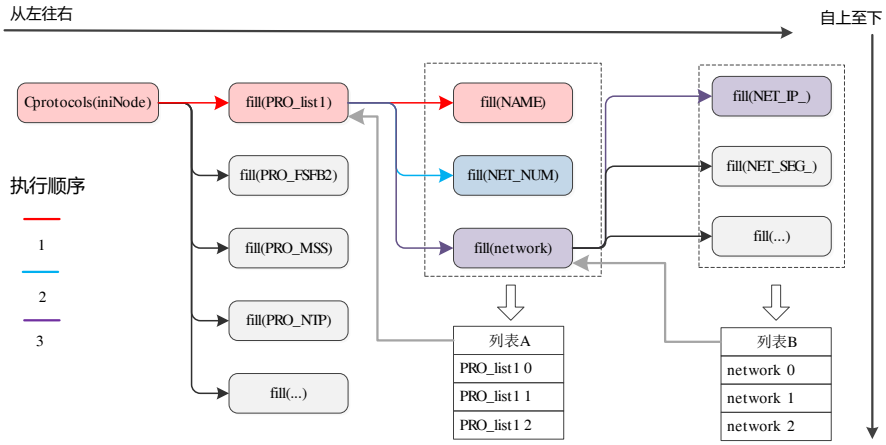

图3 协议文件节点填充示意

充后的每组 network 对象都会放到列表 B 中。最后列表 B 作为返回值赋值给 PRO_list1 对象的 network 属性。

（3）上述两步完成了一个 PRO_list1 节点填充，依次遍历剩余的 PRO_list1 节点填充，逐个按组放到列表 A，最后列表 A 作为返回值赋值协议文件 PRO_list1 属性。

余下的 PRO_FSFB2、PRO_MSS 等复杂类型列表填充流程与 PRO_list1 一致，依次遍历协议文件所有子节点填充，最终得到的数据对象 protocol 包含该协议输入文件中所有字段信息。返回的协议文件数据对象 protocol 数据结构，如图 4 所示，该结构层次清晰并且和 XSD 定义的结构一一对应，与 iniNode 接口返回的 Map 数据对象相比，不需要通过多层循环遍历获取特定数据，直接根据字段名索引获取即可。

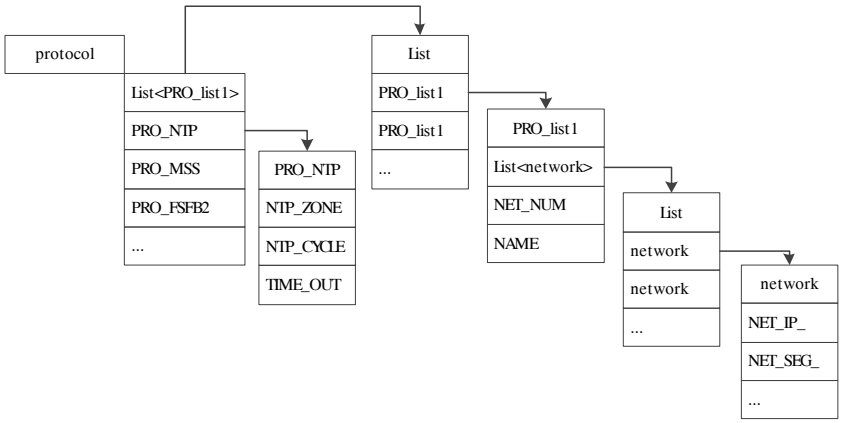

图4 protocol 对象内字段索引结构

1.1.3 输出指定格式文件

输出指定格式文件实际上就是读取的逆向处理。以生成 INI 格式为例，调用 IniWriter 类中 writerToFile 接口，输入生成文件路径，以及填充了字段信息的数据对象等参数。接口会将输入的数据对象中数据按照当前要生成文件的格式补充并以字节流形式写入指定位置的 INI 格式文件。

1.2 类生成模块

该模块用于将描述用户输入文件结构的 XSD 文件转换，生成带有输入文件结构信息的 Jar 包。

1.2.1 XSD 文件的转换

类生成模块处理流程如图 5 所示，该模块先从存储了 XSD 文件路径的配置文件中读取一份 XSD 文件数据，再解析当前文件中的节点信息。

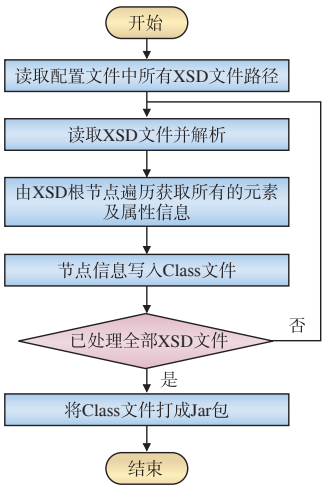

图5　类生成模块运行流程

1.2.2　节点关系设置

节点信息解析流程如图 6 所示，从根节点下一层的 PRO_list1 元素开始，将该元素下的子节点信息获取存储并设置父子关系后再向下递归至节点不包含子节点为止，返回根节点后就得到一个包含了当前 XSD 配置文件从根节点到叶子节点的所有字段信息且具有严格父子节点关系的数据对象。

上述数据对象会再经历一轮递归，此轮遍历顺序与节点的解析相同，为的是提取节点的信息赋值至新构建的字段对象中，同时，根据元素属性值出现次数及是否包含子节点，将所有节点划分为 4 种类型的字段对象。

（1）包含子节点且最大出现次数大于 1 的归为复杂类型列表；

（2）包含子节点且出现次数等于 1 的归为复杂类型单节点；

（3）不包含子节点且最大出现次数大于 1 的归为简单类型列表；

（4）不包含子节点且最大出现次数等于 1 的归为简单类型单节点。

本轮递归完成后，元素节点和属性节点内的信息都将被提取并赋值到字段对象相应的属性参数中，字段对象类中包含了元素数据类型、范围大小、出现次数、可选性、默认值、是否包含子节点、子节点信息及区分元素类型等若干个属性参数。

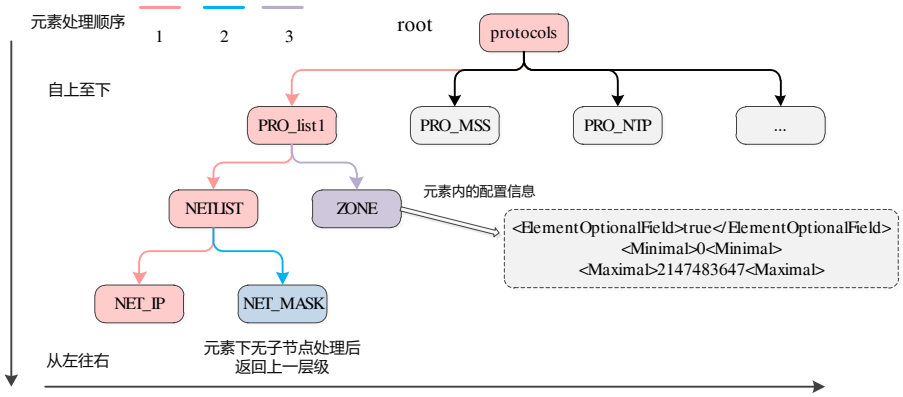

图6　节点信息解析流程

1.2.3　Class 文件生成

转换为字段对象后，包含子节点的 Field 对象会生成包含子节点信息的独立 Class 文件，文件数目取决于配置文件中复杂类型列表和复杂类型单节点字段对象数目总和。每个字段对象的属性信息将被提取，并以注解的形式写入到 Class 文件中该变量字段定义的上方，CAVA 平台类生成模块提供的自定义注解类型包括@AnnoFieldDesc（包含字段类型名称、默认值、原始类型等）、@AnnoFieldRangeCheck（字段数值范围）、@AnnoDecimalDesc（字段数值精度）、@AnnoListCountCheck（最大及最小出现次数）、@AnnoFileIdOptionalDesc（字段是否可选）等。

1.2.4　Jar 包生成

将所有的 XSD 文件按照上述流程处理完毕后得到的 Class 文件打包为 Jar 包，并导入到相应的配置工具中，即可创建与实际文件中结构对应的实体类

对象实例。

1.3 CAVA 平台与用户配置工具的关系

CAVA 平台的类生成模块将映射了输入/输出文件结构信息的一个或多个 XSD 文件作为输入，读取、解析、生成 Class 文件后打包为可供用户配置工具调用的 Jar 包；CAVA 平台的通用数据处理模块以 Jar 包形式与类生成模块生成的包含字段配置信息的 Jar 包一并引入到用户配置工具中，用户配置工具调用通用数据处理模块提供的各类接口函数，对 INI、XML、CSV、PAR、BIN 格式的输入/输出文件进行处理。

2 平台功能

2.1 XSD 配置文件转换

读取用于描述输入文件结构的 XSD 文件，提取字段信息后转换为 Class 文件，生成带有输入文件结构信息的 Jar 包。

2.2 多格式文件处理

结合带有配置信息的 Jar 包，可以处理包括 INI、XML、CSV、PAR、BIN 等多种格式的输入文件。以 Jar 包内对应文件结构为模板，将输入文件按照模板结构提取字段信息，并填充生成带有结构信息的数据对象。填充过程支持字段合法性检查及字段数值的类型转换，数据对象通过接口函数可以生成 INI、XML、CSV、PAR 和 BIN 格式的输出文件。

2.3 作为通用开发工具

（1）提供了日志管理功能，CAVA 平台的用户不需要自行构建日志功能模块，通过简单配置即可使用。

（2）提供了常用的包括 MD5 和 SHA1 计算、数值类型转换、IP 处理、命令行工具、数据过滤等开发常用函数接口。

（3）提供了项目路径管理功能，可以准确地提供当前项目的主路径。

3 关键技术

3.1 XSD 文件的配置与信息提取

采用 XSD 语言描述不同格式文件的结构，根据需求对输入/输出文件内字段进行定义，将字段配置为复杂类型元素或简单类型元素，得到对应的 XSD 配置文件[10]。

类生成模块读取 XSD 配置文件中存储的配置信息，将配置信息中的元素类型和属性类型存储在树形结构的对象中；从对象的根节点开始递归遍历，获取所有元素类型和属性类型信息，设置节点之间的父子关系；再次对该对象递归遍历，判断遍历得到的元素类型是否为复杂类型元素，每个复杂类型元素均单独生成一个 Class 文件；将生成的所有 Class 文件汇总，并打包生成包含字段配置信息的 Jar 包。

3.2 多格式文件输入/输出处理

把类生成模块生成的包含字段配置信息的 Jar 包和通用数据数据处理模块 Jar 包导入至用户配置工具中，根据用户配置工具实际输入文件的格式，调用通用数据处理模块 Jar 包中对应格式的文件读取接口函数，得到一个包含原始输入文件数据的对象 A；调用实际输入文件在包含字段配置信息的 Jar 包中对应的填充接口函数，接口函数根据字段名到对象 A 中查找对应的数值，并将该数值填充到对象 B 中的相应字段。填充字段对象时会判断当前元素是否包含限制条件或转换条件，若存在则作相应的检查或转换，直至完成所有字段的填充，得到一个与输入文件结构一致的数据对象 B。

最终得到的数据对象 B 可调用通用数据处理模块 Jar 包内的与项目输出文件格式相匹配的内部接口函数，生成指定格式的输出文件。

4 结束语

本文为城市轨道交通信号系统设备的用户配置工具开发了可通用的、基于 XSD 的配置工具数据处理安全平台，该平台已在卡斯柯信号有限公司城市轨道交通信号系统的多个用户配置工具开发项目中得到应用。开发人员只需要使用该平台配置好当前用户配置工具需要处理的 XSD 文件，无论用户配置工具的输入/输出文件字段发生更新还是需要生成其

他格式的文件，都只需要修改相应的 XSD 文件或者调用相应接口函数即可，相比之前每次需求变动都要调整用户配置工具的文件解析代码，显著提升了用户配置工具的开发效率。

参考文献

[1] 李　飞，柴慧君. 浅谈城市轨道交通信号系统数据配置 [J]. 铁道通信信号，2018，54（10）：75-77.

[2] 于　超，郑生全，石文静. 城市轨道交通CBTC系统互联互通方案研究 [J]. 铁道通信信号，2010，46（1）：44-47.

[3] 朱莹莹，王英波. 信号设备数据配置工具在CBTC中的研究与实现 [J]. 铁路通信信号工程技术，2019，16（9）：61-64.

[4] 张雯君，王天鸣. 基于全电子平台数据配置工具的研究与实现 [J]. 铁路通信信号工程技术，2021，18（6）：15-19.

[5] 余云飞. 铁路信号数据一体化配置系统设计及实现 [J]. 铁路通信信号工程技术，2017，14（6）：27-30，35.

[6] 陈倩佳，卢佩玲. 列控工程数据自动审核的研究与实现 [J]. 铁路计算机应用，2015，24（3）：6-9.

[7] 张路平，于高荣. XML在列控系统配置中的应用 [J]. 铁路通信信号工程技术，2018，15（1）：16-19.

[8] 张雪洁，王志坚，许　峰，等. 基于XML的领域异构数据库间的数据转换 [J]. 计算机与现代化，2004（5）：71-74.

[9] 鉴保瑞，宋余庆，陈健美，等. 一种基于关系的XML文档模型映射方法 [J]. 计算机应用研究，2011，28（12）：4621-4624.

[10] 杨　辉，徐国平，田绪俊. 安全关键系统配置数据生成技术研究与实践 [J]. 铁道通信信号，2019，55（5）：56-58.

责任编辑　徐侃春