

SCL 文件逐级自动更新算法设计与实现

陈宏君,冯亚东,熊 蕙,王国栋,叶 翔,文继锋

(南京南瑞继保电气有限公司,江苏 南京 211102)

摘 要:针对 SCL 文件前后变化导致变电站工程升级工作量较大的问题,文中设计了基于层次关键字逐级匹配的自动更新算法。该算法基于流文本方式快速解析两个版本的 SCL 文件,并以层次树型结构管理各文件节点数据。通过导入预先定义的节点匹配关键字设置信息,以旧版本文件数据为基准,在两个版本文件数据的各层结构内基于关键字进行匹配,标记节点的新增、删除或部分更新的标记,并将新版本文件差异内容更新到旧版本文件中,还进行逻辑节点内 DAI 短地址配置匹配处理和数据模板的刷新处理。基于该方法实现了变电站 SCL 文件自动对比、增量更新功能,方便了工程的升级维护。

关键词:SCL 文件;自动更新;逐级匹配;文件升级

中图分类号:TP301.6

文献标识码:A

文章编号:1673-629X(2016)03-0121-05

doi:10.3969/j.issn.1673-629X.2016.03.029

Design and Implementation of Hierarchical and Automatic Updating Algorithm for SCL File

CHEN Hong-jun, FENG Ya-dong, XIONG Hui, WANG Guo-dong, YE Xiang, WEN Ji-feng

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

Abstract: To solve the problem that massive updating workload is aroused by the change of SCL file in substations, it designs an automatic updating algorithm based on stepwise matching of hierarchical keywords in this paper. The algorithm implements fast analysis of two different version SCL files with streaming-text based approach, and manages data nodes of each SCL file with hierarchical tree structure. Firstly the algorithm imports pre-defined data nodes to match primary key information and considers the old version SCL file as a baseline. Then it compares the keyword of each node between the two version SCL files and marks every different node as added, deleted or partial updated. Finally the algorithm upgrades the old version SCL file node by node hierarchically with the marked distinctions, meanwhile implements the matching process of DOI short addresses within logic nodes and the refreshing of data templates. The algorithm achieves automatic comparison and incremental updating of substation SCL files and facilitates the engineering upgraded maintenance observably.

Key words: SCL file; automatic updating; hierarchical matching; file upgrading

1 概 述

SCL 文件是 IEC 61850 标准的重要内容,对该文件的配置、建模、修改是智能变电站建设的基础工作。已经在集成调试、现场投运中的装置,由于需求的变化或功能调整,可能需要升级。而升级前后的 SCD、ICD 等 SCL 文件通常有变化,例如当新增或删除信号后,可引起逻辑节点实例名和层次索引名的大幅变化,和正在使用的模型文件不兼容。此时集成配置软件需要重新关联虚端子、后台需更新画面中信号层次索引名,带来升级的工作量。SCL 文件采用 XML 层次结构描

述,不适合直接用文本对比后手动修改,需要分析其语义,提供自动比较、增量更新的方法,降低实际工程实施中的升级工作量。

文献[1]对当前信息模型的应用现状进行了总结,讨论了模型缺失、扩展、一致性测试 3 方面的问题。文献[2]提出了 IEC 61850 建模工具的概念,利用数据库存储模型描述等信息,通过提取相关信息,自动形成各种模型相关文件。文献[3]介绍了 SCL 解析器、数据模型的构建、配置模板库和相关界面的实现方法,侧重于配置工具的具体实现。文献[4]指出现有

收稿日期:2015-06-11

修回日期:2015-09-15

网络出版时间:2016-02-18

基金项目:国家“863”高技术发展计划项目(2015AA050101)

作者简介:陈宏君(1981-),男,高级工程师,硕士,研究方向为可视化编程软件和嵌入式软件研发。

网络出版地址: <http://www.cnki.net/kcms/detail/61.1450.TP.20160218.1630.036.html>

的智能变电站设计工具和系统配置工具已经无法满足用户日益增长的需求,提出了设计配置一体化的功能规范,在该文的项目管理子系统中简要介绍了版本管理模块,提出应支持基于语义的内容比较,不能简单列出不同版本的 XML 文本差异,限于篇幅,未具体阐述实现思路。文献[5]提及了 ICD 文件导入方法,该方法首先建立装置初始设计模板,重点用于对比虚端子,在第 1 次导入时需进行手动匹配。文献[6]开发了 SCL 比对工具,利用二维表结构,采用循环、排序、交叉对比的方法,可展示两个 SCL 文件的差异。上述文献均未涉及 SCL 文件自动更新的内容。

SCL 文件本质是 XML 格式的文本,文献[7]研究了大型 XML 解析技术,文献[8]研究了一种 XML 数据更新算法,利用 SAX 技术解析,在局部数据上利用 DOM 方法更新。文献[9]提出了支持 XML 文档更新的 DVLS 编码方法。但是 SCL 文件定义的相关节点有数据相关性,例如数据模板的变化,涉及到逻辑节点实例的更新,完全基于局部内容替换的方法,并不适合 SCL 文件的升级。为此文中提出一种处理思路:对于大部分节点,采用逐级关键字匹配,标记新增、删除、部分更新的标记,进行通用化的操作。对于数据模板、逻辑节点等关键数据,则在通用化算法的基础上,定义对应的处理规则并开发合适的算法程序,从而实现 SCL 文件的自动更新。

2 SCL 文件逐级更新算法

SCL 文件根节点名为 SCL,第 1 层节点包括 Header、SubStation、Communication、IED、DataTypeTemplates 共 5 个节点,在 IED 节点下包括 AccessPoint 节点,在 AccessPoint 节点下包括 Server 节点,在 Server 节点下包括 LDevcie 节点,其概要层次结构如图 1 所示^[10-12]。

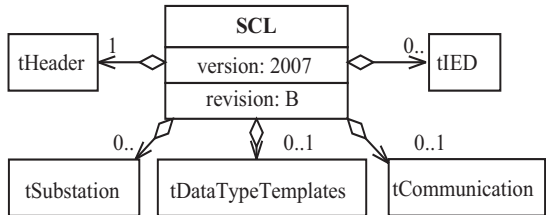


图 1 SCL 文件结构图

文献[10]详细定义了 SCL 相关的节点、属性、结构,文中不再具体介绍。本节重点介绍模型文件逐级自动更新算法的思路。

2.1 逐级自动更新原理

SCL 文件逐级自动更新算法的原理如图 2 所示。导入旧版本文件、新版本文件,以旧文件为蓝本,将新、旧文件的内容进行对比,根据不同层次的 XML 节点名称,按照预先定义的关键字,构建对应的查找匹配项,

标记节点的新增、删除、部分更新标记,在旧版本文件对应的内存数据中采取复制、删除、更新等操作,输出用于运行的文件。图 2 的原理也适用于后缀为 icd、cid 等 SCL 系列文件。

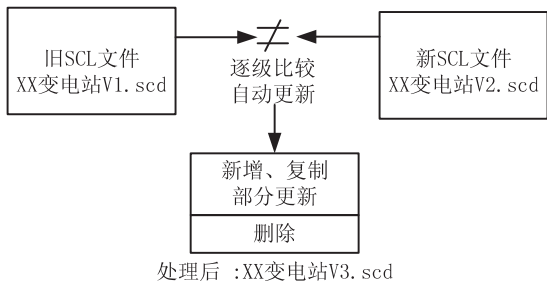


图 2 SCL 文件自动更新原理图

SCL 文件逐级自动更新算法步骤如图 3 所示。首先更新数据模板,再处理 IED 节点,采用 IED 的 name 属性值作为关键字匹配,在 name 属性相同的两个 IED 内,按照层次结构模型,进行分层更新,依次处理连接接入点 (AccessPoint)、逻辑设备 (LDevice)、逻辑节点 (LN)。之后更新变电站 (SubStation) 和通信节点 (Communication) 内容,对于文件头 (Header) 则将新文件内容直接替换到旧文件。

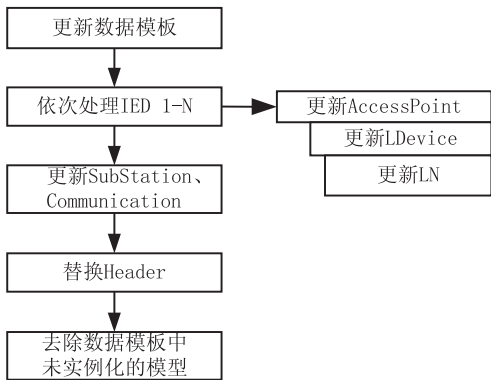


图 3 SCL 文件逐级自动更新步骤

在处理单个节点时,根据预定义的关键字,形成以关键字为主键的 Hash 表,关键字可以是本层节点、父层节点若干属性值拼接而成,则该节点的更新逻辑为 (数据模板除外):

(1) 对于关键字相同、在新旧文件中匹配的节点,则进入本层节点的部分更新处理环节,首先更新本层节点的属性值,之后递归调用更新函数,逐层进行子节点的更新;

(2) 当新版本文件的节点在旧版本文件内未找到时,将该节点标记为新增内容,并将新增内容复制到旧版本文件对应的内存数据中;

(3) 当旧版本文件的节点在新版本文件中未找到时,将该节点标记为删除内容,并从旧版本内存数据中删除。

表 1 给出了典型节点的匹配关键字。

表1 SCL节点匹配关键字

节点名	关键字
IED	name
AccessPoint	name
LDevice	inst
LN	prefix+lnType+inst
DAI	refName,sAddr
Dataset	name
LNodeType	id
DO	name+type
FCDA	ldInst+prefix+lnClass+lnInst+doName+daName+fc

在表1中, LN、FCDA的匹配关键字为多个属性值拼接而成。DAI有2个关键字, 其中refName表示层次索引名, 由DOI、SDI、DAI的name属性拼接而成, 其中sAddr是短地址信息。

每个DOI可设置一个主键DAI, 主键DAI通过INI文件可配置, INI文件按照DOType的CDC属性划分设置, 通常为DOI所管理的第1个DAI或第1个SDI下的DAI, 例如:

[CDC=INS]

stVal, ST, Key, dU

表示CDC为INS的各个实例化DOI, 其主键DAI为stVal(FC=ST), 提取stVal的sAddr属性值作为关键字, 同时dU的属性值取stVal对应的变量描述。

以站控层的AccessPoint节点下Server为例, 在图4中, 新文件有逻辑设备LD0、PROT、MEAS, 旧文件有逻辑设备LD0、PROT、CTRL, 通过inst关键字匹配后, 可得出MEAS为新增节点, CTRL为需删除节点, LD0、PROT为需进行逐层匹配更新的节点。

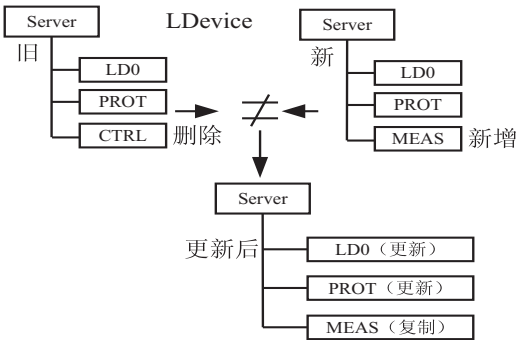


图4 逻辑设备更新示意图

SubNetwork的匹配关键字是name属性, 可用相同的原理处理变电站、通信设置信息等内容。

2.2 数据模板自动更新算法

随着《IEC 61850 工程继电保护应用模型》规范发布^[13]和新六统一入网测试的标准化建模, 数据模板的EnumType、DAType、DOType已经趋于统一和稳定, 各

个厂家或单个厂家在不同时期的数据模板的主要差异体现在LNodeType的定义上。本节重点阐述LNodeType的更新算法, 该算法也适用于数据模板内其他3种数据类的处理。对于新、旧版本的SCL文件中LNodeType, 以id为关键字构建Hash查找表, LNodeType的更新逻辑为:

(1) 对于id相同、在新旧文件中匹配的LNodeType节点, 则比较其成员DO的个数、DO的name+type的关键字是否相同, 如果不同, 则在新文件中重命名该id, 并刷新在新文件中对应的实例化LN的lnType值; 并置LNodeType为新增标记, 将其复制到旧文件的内存数据中;

(2) 当新版本文件的LNodeType在旧版本文件内未找到时, 则置为新增标记, 汇总该LNodeType中DO成员中使用的DOType、DAType、EnumType, 采用覆盖的策略, 将相关内容复制到旧文件内存数据中;

(3) 当旧版本文件的LNodeType在新版本文件内未找到时, 暂不置删除标记, 而是在逻辑节点更新处理结束后, 检测各个逻辑设备下是否有该类型的LN实例, 如果LNodeType在SCL中实例化个数为0, 则置删除标记。

2.3 逻辑节点自动更新算法

1个逻辑节点实例代表1个具体的功能, 是分解得到的最小功能单元。逻辑节点由若干公用数据类的派生实例组合而成。LN的匹配关键字在表1中定义为prefix+lnType+inst, 用lnType替换lnClass作为关键属性的原因是需相同类模板的实例才能匹配。LN的DOI配置内容是关键信息, 其主键DAI的sAddr属性填写装置的变量名或地址, 故对sAddr属性的更新是关键步骤。定义需要处理的DOI数据扩充类为CtDOIEx, 并命名新文件的LDevice实例为pnewLD, 旧文件匹配的实例为poldLD, 在LDevice内定义2个Hash表:

(1) 基于短地址QMultiHash<QString, CtDOIEx*>m_saddrHash, 用于汇总配置的变量;

(2) 基于DOI的主键层次索引名QHash<Qstring, CtDOIEx*>m_refHash, 用于汇总配置变量对应的层次索引名。

在逻辑设备范围内, 对各个逻辑节点的配置内容, 以sAddr为核心进行更新匹配, 其处理步骤为:

(1) 依次汇总新、旧文件中匹配的LDevice下各个LN的DOI配置信息, 形成pnewLD、poldLD的2个Hash表;

(2) 依次遍历poldLD->m_saddrHash的成员pex1, 以pex1的短地址在pnewLD->m_saddrHash查找, 如查找到, 则置保留标记; 若未查找到, 置删除标记;

(3) 依次遍历 `pnewLD->m_saddrHash` 的成员 `pex2`, 以 `pex2` 的层次索引名在 `poldLD->m_refHash` 查找, 若未查找到, 置新增标记;

(4) 汇总 `poldLD` 中置删除标记的 DOI, 在内存数据中清除该 DOI 配置, 并删除关联的数据集内 FCDA; 若单个 LN 的 DOI 都删除, 则删除该 LN;

(5) 汇总 `pnewLD` 中置新增标记的 DOI, 将该 LN 复制到 `poldLD` 内存数据中, 仅复制 LN 内为新增标记的 DOI 和关联的 FCDA, 并处理 LN 的实例号和更新数据模板。

上述处理还考虑了变量配置迁移到另一个 LN 的情况。图 5 是逻辑节点更新的示例。

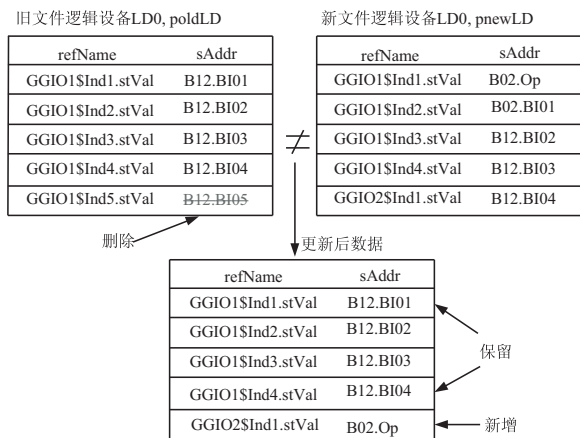


图 5 逻辑节点更新示意图

在图 5 中, 通过双向匹配, 旧文件中 B12.BI05 置删除标记, 新文件中的 B02.Op 置新增标记, B12.BI01 ~ B12.BI04 置保留标记, 其 `refName` 沿用旧文件的索引, 而 B02.Op 配置内容填充到新创建的 GGIO2.Ind1.stVal 的 `sAddr` 属性, 最大程度保留了旧文件的配置内容, 实现了增量更新。

在逻辑节点下, 还需处理数据集、报告块等内容更新, 这些节点均以 `name` 为关键字进行匹配。在数据集内, 单个 FCDA 以其所有属性值拼接后作为关键字进行匹配。

3 大型 SCL 文件读写实现

在实现第 2 节介绍的处理算法时, 需进行 XML 数据建模, 并基于该模型实现大型 SCL 读写操作等基础功能。

3.1 XML 节点类定义

XML 节点包括若干属性和若干子节点, 是个递归定义结构:

```
class SCL_DECLSPEC CElement{
public:
    CElement(QString& name);
    CElement(QString& name, CElement* parent);
```

```
virtual ~CElement();
public:
    QList<CAAttr*> m_attrList; //本层属性 QList<CElement*>
    m_childList; //子节点
protected:
    ushort m_classID; //类型标记
    CElement* m_pParent; //父节点
    QString m_strName; //本层 XML 节点名
    QString m_strValue; //本层 XML 设置值
};
```

CElement 提供属性和子节点查找、创建、删除、设置等接口。SCL 相关节点基于 CElement 派生, SCL 类设计参照 IEC 61850 标准发布的 SCL.xsd、SCL_IED.xsd 等文件的定义, 包括 CtDO、CtDOI、CtAnyLN、CtLN、CtLDevice、CtAccessPoint、CtIED 等数据类。

3.2 大型 SCL 文件读写处理

由于 SCL 文件更新是分层次进行, 并且基于 Hash 表查找, 处理的时间、空间复杂度较低, 主要的时间消耗在 SCL 文件的读写处理上 (据统计, 读写时间平均占用整个处理环节 65% 的时间)。文献 [3] 采用 Xerxes 开源库解析 SCL 文件, 采用基于 DOM 的树形结构读写文件。基于 DOM 的接口功能丰富, 但是读写速度慢。文献 [7] 研究了基于 SAX 方式解析大型 XML 文件, 解析速度较快, SAX 在碰到一个节点后, 会调用基类定义的回调虚函数, 应用层只能根据传入的参数实现不同处理, 代码实现复杂, 并且不支持增量更新写操作。QT4.5 版本提供了 QXmlStreamReader、QXmlStreamWriter 的接口, 是 SAX 方式的改进接口, 读写速度有 20% 的提升, 并支持分层读取。文中采用基于 QT 提供的流文本方式实现大型 SCL 文件读写。按照流文本方式扫描, 通过 `readNext()` 读取一个节点, 并对当前节点做相应处理。基于抽象数据结构 CElement 和对象工厂, 则可实现 SCL 文件的通用化读写, 核心代码仅为 150 行, 采用栈来临时管理创建的节点数据, 其读文件的算法思路为:

(1) 如果读到节点类型为 `StartElement`, 则根据节点名调用对象工厂创建派生实例, 如果栈为空, 是根节点, 否则弹出栈顶指针作为父节点, 并存储当前节点的属性列表; 新创建对象入栈;

(2) 如果读到节点类型为 `Characters`, 则处理 CDA-TA、TextNodes 类型的设置值;

(3) 如果读到节点类型为 `EndElement`, 则弹出栈顶元素;

(4) 读取到文件尾部, 结束。

SCL 读取函数的部分代码示例如下:

```
QStack<CElement*> qstack;
QXmlStreamReader reader(&file);
```



```
reader. setNamespaceProcessing( false );
while( ! reader. atEnd() ) {
reader. readNext();
if( reader. isStartElement() )
{
QString qname = \
( reader. qualifiedName(). toString() );
CElement * pbase = \
CtGlobal::creatByID( qname );
CElement * pParent = \
qstack. isEmpty()? 0:qstack. top();
if( pParent ) { parent->addElement( pbase ); }
else { m_pSCL=pbase; }
qstack. push( pbase );
... //读取和存储本节点属性
}
else if( reader. isCharacters() )
{ ...//读取节点值 }
else if( reader. isEndElement() )
{ if( ! qstack. isEmpty() ) qstack. pop(); }
}
```

基于流方式的 SCL 文件写操作,则是通过递归调用写节点的函数实现:

```
void writeNode( QXmlStreamWriter &qwr,
CElement * pNode ) {
foreach( CAttr * pattr, pNode->m_attrList )
{ qwr. writeAttribute( pattr->name, pattr->val ); }
if( pNode->isTextNode() )
{ qwr. writeCharacters( pNode->getValue() ); }
else if( pNode->isCdataNode() )
{ qwr. writeCDATA( pNode->getValue() ); }
foreach( CElement * pch, pNode->m_childList ) {
qwr. writeStartElement( pch->getName() );
writeNode( qwr, pch );
qwr. writeEndElement(); }
}
```

表 2 给出了不同大小文件的读写时间统计(测试环境为 4 G 内存、2.6 GHz 的台式机)。

表 2 SCL 文件读写耗时

文件名	文件大小/M	读耗时/s	写耗时/s
兰溪变.scd	25.378	3.006	1.958
未来城.scd	62.827	6.839	3.525
沙洲变.scd	135.529	11.715	7.847

4 结束语

文中提出一种 SCL 文件分层比较、自动更新算法,通过新、旧版本文件之间的逐级自动对比,设置相

关标记,实现复制、删除、部分更新等操作,能够实现模型文件的自动化升级,有效减少实际工程实施中的人工处理工作量,从而降低模型升级风险,提升数字化工程实施效率。该方法已经在新六统一入网测试中进行了应用,并通过了近两年的国内外数字化工程的实践检验。文中后续工作是支持 IEC 61850 Ed2 版本^[14]的 SCL 文件更新处理。

参考文献:

[1] 樊 陈,倪益民,窦仁晖,等.智能变电站信息模型的讨论[J].电力系统自动化,2012,36(13):15-19.

[2] 王丽华,张青山,张马龙,等.IEC 61850 建模工具的设计与实现[J].电力系统自动化,2008,32(4):73-76.

[3] 祁 忠,笃 竣,张志学,等.IEC61850SCL 配置工具的研究与实现[J].电力系统保护与控制,2009,37(7):76-81.

[4] 笃 竣,叶 翔,王长瑞,等.智能变电站设计配置一体化功能规范研究及工具开发[J].电力系统自动化,2014,38(20):85-89.

[5] 修黎明,高湛军,黄德斌,等.智能变电站二次系统设计方法研究[J].电力系统保护与控制,2012,40(22):124-128.

[6] 高 磊.IEC 61850 SCL 配置文件比对工具的研究与实现[J].电力系统自动化,2013,37(20):88-91.

[7] 张太彪,曾文华,陈志伟.大型 XML 文档解析技术的应用与研究[J].厦门大学学报:自然科学版,2009,48(3):338-341.

[8] 鲍培明,吉根林.XML 数据更新算法研究[J].计算机工程,2008,34(4):101-103.

[9] 付 鹏,蒋夏军,皮德常.一种新的支持 XML 文档更新的编码方法[J].计算机科学,2014,41(3):193-197.

[10] IEC/TC57. Communication networks and systems for power utility automation-part 6: configuration description language for communication in electrical substation related to IEDs [S]. [s. l.]:IEC/TC57,2003.

[11] IIEC/TC57. Communication networks and systems for power utility automation-part 7-3: basic communication structure-common data classes[S]. [s. l.]:IEC/TC57,2003.

[12] IIEC/TC57. Communication networks and systems for power utility automation-part 7-4: basic communication structure-compatible logical node classes and data object classes[S]. [s. l.]:IEC/TC57,2003.

[13] IIEC 61850 工程继电保护应用模型[S]. Q/GDW 1396-2012,2014.

[14] IIEC/TC57. Communication networks and systems for power utility automation-part 6: configuration description language for communication in electrical substation related to IEDs [S]. [s. l.]:IEC/TC57,2010.