第25卷第5期 华东交通大学学报 Vol. 25 No. 5
2008年10月 Journal of East China Jiaotong University Oct. ,2008

文章编号:1005-0523(2008)05-0071-06

采用面向对象的ASN.1 BER编译器的研究

丁青锋

(华东交通大学 电气与电子工程学院,江西 南昌 330013)

摘要:传统ASN.1编译器有占用空间较大、运行效率不高的矛盾以及升级困难等缺点. 采用面向对象的C++语言实现的编解码系统,利用一种新的、更为简单有效的方式来标识变电站通信报文的信息内容以及类型特征,从而大大提高编解码效率和占用更小的空间.

关 键 词:抽象描述文法(ASN.1);BER;编解码

中图分类号:TP311;TP393 文献标识码:A

1 ASN.1基本编码规则

ASN.1中定义了1套简单的基本数据类型和1组构造应用类型的规则以及给这些类型赋值的机制. 另外,为了在解码时能区别结构数据类型的可选项成员,这时需要采用派生的方法改变其中1个数据成员的类型标识,称之为派生类型. 派生数据类型标识的方法有两种:显式派生法和隐式派生法,Tag定义中默认为显式模式[1].

ASN.1的基本编码规则BER采用的编码结构应由下列次序的4种成分组成:标识符字段、长度字段、内容字段和内容结束字段,但是除非长度字节的值需要出现内容结束字段,否则该内容结束字段不应出现,即T-L-V结构[2].

2 编解码方案

2.1 传统编译器的特点

传统的ASN.1编解码器的实现方案主要有两种. 一种是扫描类型定义表法,在程序初始化时,将相应的ASN.1描述文件转换成一张"类型定义表",该表记录了所有的数据类型的定义方法以及相互之间的关系,当有数据要求进行编码时,就在对该类型定义表进行扫描的同时进行编码,扫描完成之后,编码也就完成了. 当接收到数据需进行解码时,同样会对该表进行解码;另外一种方法是用专用的ASN.1编译器对ASN.1描述文件进行编译,对所定义的所有数据类型都生成相应的编码和解码函数,目前现有ASN.1的第三方编译器如ASN1C、SNACC等. 当有数据要求进行编码或解码时,便按照其数据类型调用相应的编码或解码函数[3].

采用传统的编译器的优点是生成的源码层次结构非常清晰,可读性较好. 但是传统方案在性能上的缺点也是很明显的.

(1) 效率与空间问题[4]

收稿日期:2008-07-18

基金项目:华东交通大学校立科研基金资助(07DQ02)

作者简介:丁青锋(1980-),男,安徽安庆人,助教,硕士,研究方向为变电站自动化系统、交通信息工程及控制等.

第一种方法只需编写生成类型定义表和驱动基本类型编解码库的程序,其对空间的要求相对很小,但是由于每次进行编解码时都需要扫描类型定义表,所以运行的效率相对较低;相对第一种方法,方法二由于省去了扫描过程而直接调用编解码函数,所以效率相对较高,但是由于要对每一数据类型提供编解码函数,因此,会占用较大的代码空间,对内存空间要求较高.

从上面的分析可以看出,传统的 ASN.1 编解码方案不能兼顾效率与所占空间的大小问题,这也是本文需要解决的问题.

(2) 不利于升级和维护[5,6]

由于编译器通常是对应于特定的操作系统和硬件,其生成的接口代码文件也是与特定的操作系统和硬件有关,这样就很大的限制了编解码器的使用平台. 另外,变电站通信报文的 ASN.1 描述文件有变化,都会使编译的程序源码发生很大的改变,与之相关的其他模块需要重新测试,这是软件编写中不希望看到的.

2.2 编解码方案

采用传统编译器的方法会带来诸多弊端,如效率与空间的问题以及代码升级难等. 在使用 ASN1C 编译器对用 ASN.1 抽象语法描述的变电站通信报文进行编译时,生成的文件包括头文件(.H)和源文件(.Cpp). 其中头文件是通信报文的 ASN.1 描述文件所映射的 C++结构. 源文件是对 ASN.1 描述文件中定义的每种类型的编解码函数,这些函数与 ASN.1 描述文件相对应,该文件包含有大量的冗余信息,如函数的调用过多的嵌套,且每个函数的入口参数重复,当 ASN.1 描述文件有变化时,将涉及大量源代码的修改,与之相关的模块又必须重新调试. 显然这种方式是大型程序编写中所忌讳的.

因此,在设计变电站通信报文的 ASN.1 编解码模块时,希望 ASN.1 编解码模块的核心部分相对稳定,该部分不随 ASN.1 描述文件的改变而改变[7]. 模块的核心部分程序包括主编解码函数、ASN.1 实时库(ASN.1 Runtime Library)以及其他辅助函数等,这些函数共同完成对报文数据的编码解码工作. 这一部分是相对稳定的,当 ASN.1 描述文件有所变化时,只需要修改信息文件就可以了,而核心的编解码模块部分是不需要修改的. 信息文件主要完成编译器的头文件部分功能,用来对消息各项元素赋值,另外还包括描述文件各种类型的定义信息,这些信息主要作为编解码模块的参数输入.

ASN.1 编解码器读入信息文件后,主编解码函数根据其中的定义信息判断编解码元素的数据类型,调用 ASN.1 实时库中对应的数据类型编解码函数对其进行编码或者解码,然后将编码/解码结果送到应用程序[8,9].

3 编解码系统的主模块设计实现

由于本项目所针对的 MMS PDU 仅仅涉及到 ASN.1 BER 编码规则中的一小部分子集,所以下面以 BER 的编解码的实现,说明 ASN.1 编解码的主模块的设计思想.

本文采用的编程环境是 Visual Studio.NET,采用面向对象技术实现 ASN.1 BER 的编解码系统. 首先通过提取 ASN.1 各种数据类型的公共特性,定义一个基类 CValue. 在基类中,定义各个基本类型所共有的特性,如标识类 Tag_Class 以及基本数据类型 OriginalTag 等,还有 Tag 编解码函数 BEncodeDistinctTag()、BEncodeOriginalTag()和 BERDecodeTag()以及各种内容、长度编解码函数. 然后是各个具体的数据类型所对应的子类的定义,而在子类中定义了该类型所对应内容、长度字段的编解码函数,是基类中内容、长度编解码函数的重载.

3.1 基类的公共函数算法及实现

在 ASN.1 BER 规则中,对每种数据类型的编码分为对标识符 TAG 的编码、对长度 LENGTH 的编码、对内容 VALUE 的编码. 对标识符 TAG、长度 LENGTH、内容 VALUE 的编码/解码,在基类中定义了以下几个函数:

显式派生类型编码函数 BEREncode();隐式派生类型编码函数 BERImplicitEncode();显式派生类型解码函数 BERDecode();隐式派生类型解码函数 BERImplicitDecode();派生类数据类型标识编码函数 BEREncodeDistinctTag();通用类数据类型标识编码函数 BEREncodeOriginalTag();数据类型标识解码函数 BERDecodeTag();长度字段编码函数 BEREncodeFixLength();长度字段解码函数 BERDecodeFixLength();对输入基

本数据类型的结构判断函数 Jud _ Tag( ) ; 对输入类型标识的结构判断函数 IsConstruct( ) .

在上述函数中,函数 Jud _ Tag( ) 、IsConstruct( ) 无返回值; 其余函数的返回值类型为 encode _ buf* 结构类型,表示函数编解码结束后的比特流的指针.

1) 数据类型标识编解码函数

数据类型标识分为基本类数据类型标识和派生类数据类型标识,类型标识字段的编码函数为 BEncode-OriginalTag(pOutBuf,asn _ elem,k) 和 BEREncodeDistinctTag(pOutBuf,asn _ elem,k) . 其中参数 asn _ elem 为待编码类型的通用信息结构. 标识字段的解码由函数 BERDecodeTag(pInBuf,&asn _ elem) 来完成,解码所得类型标识信息由参数通用信息结构 asn _ elem 返回,为接下来的内容字段的解码函数提供参数.

(1) 通用类数据类型标识编码函数算法及实现

函数 BEncodeOriginalTag (pOutBuf,asn _ elem,k) 完成对通用类的基本类数据类型标识 TAG 的编码. 通用类数据类型标识函数的具体算法为: 首先是对 TAG 的合法性进行判断,输入的 TAG 值应为通用类型; 然后通过调用函数 Jud _ Tag (asn _ elem,&tag _ construct) 判断 TAG 值为简单类型还是结构类型; 最后对 TAG 值进行编码即可.

(2) 派生类数据类型标识编码函数算法及实现

派生类数据类型标识编码算法是通过判断输入的 TAG 值,根据不同的范围确定数据类型标识编码的长度. 在经过 ASN. 1 编码后 TAG 值并非只是 1 个字节的八位位组.

在派生类数据类型标识编码函数 BEREncodeDistinctTag( ) 中对类型标识是否为结构类型的判断通过调用基类函数 IsConstruct(asn _ elem,&tag _ construct) 完成: 当为结构类型时置布尔类型参数 tag _ construct 为 TRUE,否则为 FLASE.

(3) 数据类型标识解码函数算法及实现

函数 BERDecodeTag(InBuf,&asn _ elem) 完成各种通用类数据类型和派生类数据类型的类型标识的解码. 其中参数 asn _ elem 返回类型标识信息,包括标识类、类型、是否含有关键字 IMPLICIT 等信息,为内容字段解码函数提供类型参数.

数据类型标识的解码函数算法首先通过判断待解码数据流的首个字节高两位的值,确定标识类是否为通用类; 若为通用类则通过 switch 语句判断是何种基本数据类型,若为派生类,置相应的 TAG _ CLASS 值,另外还需确定是否包含 IMPLICIT 关键字,最后确定其具体的基本数据类型.

2) BER 长度编解码函数

BER 长度编码/解码函数分别为 BEREncodeFixLength(pOutBuf,len,&k) 、BERDecode _ FixLength(InBuf,&len) . 其中函数 BEREncodeFixLength(pOutBuf,len,&k) 主要是辅助各种类型长度编码函数完成长度编码,其中参数 len 为待编码的内容字节长度,参数 k 表示该长度字段所占用的字节数; 而各种类型没有专门的长度解码函数,都是通用调用基类长度解码函数 BERDecodeFixLength (InBuf,&len) 完成长度解码,其中由参数 len 返回待解码内容字段长度.

(1) 长度字段的编码函数算法

长度字节编码函数 BEREncodeFixLength(pOutBuf,len,&k) 算法分为单字节和多字节两种情况. 当内容字段所占的字节数小于或者等于 127 时,长度编码采用单字节,其中 bit8 = 0,位 7 至位 1 为 7 位无符号二进制数表示内容字节数; 当内容字段所占的字节数大于 127 时采用多字节,长度字段的初始字节编码如下: bit8 = 1,bit7 到 bit1 的值表示长度字段后继字节数,接下来的是后继字节表示内容字段实际所占的字节数.

(2) 长度字段的解码函数

长度解码函数 BERDecodeFixLength(InBuf,&len) 完成各种类型的数据内容长度解码. 其算法是先判断首字节的 bit8 位是否为 0: 如果为 0,则为单字节数,将低 7 位的无符号数返回给长度值参数 len; 若为 1,得出低 7 位数的值 L,再返回接下来的 L 个字节值给长度值参数 len.

3) BER 编解码函数实现

在 MMS PDU 的 ASN. 1 描述文件中,类型标识 TAG 定义常常出现嵌套派生以及关键字 IMPLICIT 的情况. 这使其编解码变得更加复杂,为此,本文在基类中设计了显式和隐式两类派生类型编解码函数. 而在各个

数据类型子类中定义了基本数据类型的长度、内容字段的编解码函数,以此构成 ASN. 1 实时库(ASN. 1 Runtime Library) .

(1) 显式派生类型数据编码/解码函数算法

显式派生类型编码由函数 BEREncode( ) 来完成. 编码函数的算法是首先需要判断数据类型标识是否为通用类型,如果是通用类基本类型数据标识,则异常退出. 如果数据类型标识是派生数据类型,则调用派生类标识编码函数 BEREncodeDistinctTag(pOutBuf,asn _ elem) ,然后还需要进一步判断参数 asn _ elem 是否带有隐式类型关键字 IMPLICIT: 如果没有 IMPLICIT 关键字,则调用长度编码函数 BEREncodeLength(pOutBuf, length,lack _ bits,Asn _ elem,Elem _ value) ,然后调用类型标识、长度和内容字段编码函数; 若有 IMPLICIT 关键字则直接调用长度和内容字节编码函数即可,完成对长度和内容字节的编码,而不需要对类型标识进行编码.

显式派生类型解码函数 BERDecode( ) 实现是与编码函数很类似的过程,首先是通过对其类型标识比特流的判断,得出其标识类、TAG 的值以及是否包含 IMPLICIT 关键字等标识信息. 解码函数将解码后的类型信息和数据内容等信息返回给信息文件的通用信息结构 General _ Infor 结构和赋值结构 Message _ Str 结构.

(2) 隐式派生类型编码/解码函数的算法

隐式派生类型编码函数算法与显式编码函数算法最大的不同在于对标记 IMPLICIT 后的 TAG 是否编码,其中隐式派生类型编码函数不需要对其紧跟的 TAG 字段进行编码. 隐式派生类型编码/解码函数分别为 BERImplicitEncode( ) 和 BERImplicit Decode( ) .

4) 其他函数说明

为完成对 MMS PDU 的编解码,本文还编写了一些辅助函数,这些函数与编解码函数共同完成报文的编解码工作.

3.2 子类中基本类型编解码函数

在 ASN. 1 BER 规则中,对每个数据类型的编码分为对标识符 TAG 的编码、对长度 LENGTH 的编码和对内容 VALUE 的编码. 而对长度和内容的编码,因类型不同其编码函数也会各不相同,对解码函数也同样如此.

如果派生类中重定义的成员函数与基类中同名函数的参数表不完全相同,则称为派生类重载了基类的成员函数,这与同一个类中函数重载的情况是一样,系统会根据调用时实参的不同调用不同的函数. 本文采用派生类重载方法,在不同的类型子类中重载长度和内容编码/解码函数,其中长度字段解码都是通过调用基类中 BERDecodeFixLength( ) 这一长度解码函数. 这样,系统在编码/解码时会根据类型不同调用相应类型的编解码函数. 在基类中,定义了如下 3 个编解码函数: 内容字段编码函数、内容字段解码函数、长度字段编码函数.

在 ASN. 1 的各个数据类型子类中定义对内容和长度编解码进行具体实现的成员函数,这些基本数据类型的编码/解码函数就构成了 ASN. 1 的实时库(ASN. 1 Runtime Library) . 表 1 对基类、REAL 和 SEQUENCE 类型的子类中的编解码函数进行比较.

表 1 各种类中的编解码函数比较

类	基类 CValue	子类 CRealValue	子类 CSEQUENCEValue
BEREncodeContent( )	空函数	对实数类型值编码	对 SEQUENCE 中各个域依次编码
BERDecodeContent( )	空函数	对实数类型值解码	对 SEQUENCE 中各个域依次解码
BEREncodeLength( )	空函数	对实数类型值的长度编码	对 SEQUENCE 中各个域长度之和编码
BERDecodeLength( )	实函数	调用基类解码函数对实数类型值的长度解码	调用基类解码函数对 SEQUENCE 中各个域长度之和解码

在这些函数中,主题部分为依据 ASN. 1 基本编码规范(X. 690) 对各个类型值编解码的实现. 在对内容

和长度进行编解码时,根据不同的数据类型执行相应子类的成员函数即实现各种数据类型的内容和长度字段的编解码. 其中长度字段解码函数通过调用基类长度字段解码函数从待解码数据流中得出长度值赋给参数 len 即可,此参数为内容字段解码函数的输入参数,内容解码函数将解码结果返回到 Message _ Str 结构和 General _ Str 结构中.

1) 简单类型编解码函数

简单类型的编码算法分为对标识符 TAG 的编码、对长度 LENGTH 的编码和对内容 VALUE 的编码,其中对类型标识 TAG 的编码是通过调用基类的基本类型标识编码函数 BEREncodeOriginalTag( ) 实现.

2) 复合类型编解码函数

对于复合类型编码,是将低级的编码结果提交给上一级的结构体. 对于上一级的结构体来说, "子"结构体就是其内容,所以对于这一级的结构体来说它的长度值就是所有这些返回的长度值、标识字段以及长度字段的总和. 因此需要将本结构体整体编码后的长度值返回给其"父"结构体,这样从底层一级级返回整体编码的长度值就可以方便的填写 LENGTH 值了.

4　ASN. 1 编解码模块的功能测试与性能分析

4. 1　ASN. 1 编解码模块的功能测试

为了完成对 ASN. 1 中的基本数据类型的编解码函数功能进行测试,通过编写相应的调试程序,调用 ASN. 1 的各种数据类型的编解码函数,分析其编解码后的结果. 分析得出: 编码结果符合 ASN. 1 编码规范,解码结果和原 PDU 数据符合. 另外,还对各种数据类型的编解码函数做了极限测试.

4. 2　ASN. 1 编解码模块的性能分析

本文的编解码模块与传统编译器最大的不同是采用信息文件为主编解码函数提供类型参数,而编译器是为每个类型生成不同的编解码函数. 信息文件相对与编译器生成的编解码函数来说,代码量大大减少. 而且由于变电站通信报文中很多元素的定义是相同的,因此,各个报文可以公用某些元素的通用信息 Message _ Str 结构,这也大大节省了代码空间. 而两者的 ASN. 1 基本数据类型构成的 ASN. 1 实时库文件的代码大小基本相同. 经过多次实验证明: 采用本文的编解码模块所产生的代码只相当于编译器产生的代码量的 1/10 左右.

另外,对通用信息 Message _ Str 结构采用的是链表形式,这样整个报文所有元素可以采用递归访问方式. 在编解码时,只需通过链表指针的移动,完成整个报文的编解码工作,这相对编译器的多层函数调用要简单得多.

5　结论

主要探讨 ASN. 1 定义的数据类型以及采用 ASN. 1 基本编码规则的各种数据类型具体编码、解码算法实现,在基本数据类型编解码算法的基础上构成了编解码系统的实时库,因为采用面向对象的方法,节省了系统的代码量. 同时采用链表结构大大提高了对结构类型数据的嵌套访问的速度.

参考文献:

[1] 李　婷. 智能化 DER 编码系统的设计与实现 [D]. 北京: 北京化工大学,2000.

[2] GB/T 16263. 1 - 2006/ISO/IEC 8825 - 1: 2002, Information Technology ASN. 1 Encoding Rules [S].

[3] GB/T 16263. 1 - 2006/ISO/IEC 8824 - 1: 2002, Information Technology Abstract Systax Notation One (ASN. 1) : Specification of Basic Notation [S].

[4] 袁作月,黄本雄,苏应飙. 嵌入式 VOIP 系统中 ASN. 1 编解码器的实现 [J]. 计算机工程与应用,2002,8: 178 - 180.

[5] Neufeld G W, Yang Y L. The design and implementation of an ASN - complier [J]. IEEE Transaction of Software Engineering, 1990, 16(10) : 31 - 34.

[6] ZHANG Hai – bin, Ai Bo. The implement of the ASN. 1 _ C + + compiler and its application in IN [J]. High Technology Letters, 1998, 12(2): 4 – 6.
[7] 江明德. 面向对象程序设计 [M]. 北京: 电子工业出版社, 1994. 45 – 69.
[8] 吕 谦. 现代电信协议 ASN. 1 编解码实现机制的研究 [D]. 武汉: 华中科技大学, 2002.
[9] Mary Shaw, David Garlan. Software architecture [M]. Carnegie Mellon University: Prentice Hall International Inc., 1998. 41 – 42.

A Research of the Compiler for ASN. 1 BER Based on Object – oriented

DING Qing – feng

(School of Electrical and Electronic Engineering, East China Jiaotong University, Nanchang 330013, China)

Abstract: The tranditional ASN. 1 complier has such defects as bulky occupation, operating inefficiency and upgrading difficulty. The thesis adopts an encoding – decoding system based on object – oriented C + + by means of a more effective method to mark the content and the type characteristic of the substation communication, which can improve its operating efficiency and require relatively smaller occupation.

Key words: ASN. 1(Abstract Syntax Notation 1); basic encode rule; encode/decode

(责任编辑: 刘棉玲)

(上接第 53 页)

AMPS/AA Superabsorbent Resin Synthesized by Inverse Suspension Polymerization

YANG Xiao – min, LIU Jian – ping, XIONG Le – yan, QIN Tian – tian

(School of Basic Sciences, East China Jiaotong University, Nanchang 330013, China)

Abstract: Inverse suspension copolymerization of acrylate acid (AA)/acrylamidomethylpropane sulfonic acid (AMPS) is studied by using methylene bisacrylamide as the cross linking agent, potassium persulfate as the initiator and the span 60 as the stabilization agent. The optimum conditions from orthogonal experiment are as follows: n_{AMPS} : n_{AA} = 1.8: 1, neutralization degree of acrylate acid 70 %, wt (initiator) = 0.1 %, wt (cross – linking agent) = 0.05 %. Under the optimum conditions, the ability of absorbing water by the resin increases by 990 and that of absorbing NaCl (0.9 %) solution increases by 123g/g.

Key words: superabsorbent resin; acrylamidomethylpropane sulfonic acid; inverse suspension polymerization

(责任编辑: 李 萍)