ASN.1/XML トランスレータ

5P－3

今村 剛*　丸山 宏*†
*日本アイ・ビー・エム(株) 東京基礎研究所
†東京工業大学 情報理工学研究科

1 はじめに

Abstract Syntax Notation One (ASN.1) [1] と Extensible Markup Language (XML) [2] は，共に構造化データ (の構造と内容) を記述するための言語である．特に XML は，データを Web 上で，目に見える形で交換・配布することを考慮して設計されている．また，プロセッサやエディタのような，フリーのツールが多いことも特徴である．従って，ASN.1 を XML に変換することができれば，ASN.1 の利便性は高まると言える．本稿では，そのような目的のもとに設計・実装した ASN.1/XML トランスレータについて述べる．

2 ASN.1 と XML

ASN.1 は CCITT (現在の ITU-T) と ISO により 1988 年に開発され，XML は W3C により 1998 年に開発された．

2.1 ASN.1 とは？

ASN.1 は，いくつかの基本的なデータ型と，それらを組み合わせて新しいデータ型を定義するためのデータ型を定義している．前者は単純型 (simple type) と呼ばれ，BOOLEAN, INTEGER, OCTET STRING などが含まれる．また，後者は構造型 (structured type) と呼ばれ，SEQUENCE, SET などが含まれる．図 1 に，データ型の定義例を示す．データ型 (の集合) は，ASN.1 抽象構文と呼ばれる．

```
Record ::= SEQUENCE {
   number Number,
   name   Name }
Number ::= INTEGER
Name ::= OCTET STRING
```

図 1: データ型の定義例

ASN.1 は，データを符号化するための規則も定義している．符号化規則はいくつかあり，Basic Encoding Rules (BER) [3] はそのうちの 1 つである．BER では，どのデータも，型 (type)，長さ (length)，値 (value) の組合せで符号化される．このため，TLV 符号化と呼ばれる．また，どの符号化データも，型によりデータ型を特定でき，長さによりデータの終わりを検出できるため，値を取り出して解読することができる．図 2 に，図 1 のデータ型への値の割当て例とその符号化を示す．符号化データは，ASN.1 転送構文と呼ばれる．

```
record Record ::= {       30 09
   number 12,                02 01 0C
   name   '1AA2FFFF' }       04 04 1AA2FFFF
```

図 2: 値の割当て例とその符号化

ASN.1/XML Translator
Takeshi Imamura (imamu@jp.ibm.com) and Hiroshi Maruyama
Tokyo Research Laboratory, IBM Japan, Ltd.
Dept. of Computer Science, Tokyo Institute of Technology

2.2 ASN.1 と XML の変換

冒頭で述べたように，ASN.1 と XML は共に構造化データを記述するための言語であることから，それらの変換はある程度は可能であると思われる．そして，ASN.1/XML トランスレータは，その変換を実現する．即ち，ASN.1 抽象構文を DTD に変換し，ASN.1 転送構文を DOM 木に変換する．また，その逆も行う．

3 ASN.1/XML トランスレータ

ASN.1 と XML を変換するトランスレータを設計・実装した．実装には，Java Development Kit (JDK) 1.1 [4] を使用した．

3.1 トランスレータの構成

図 3 に，トランスレータの構成を示す．トランスレータは，6 つのモジュールから構成される．

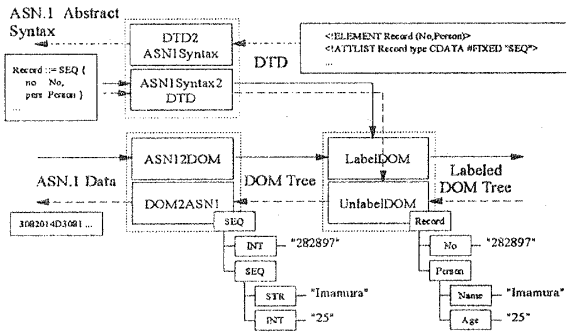

図 3: トランスレータの構成

3.2 各モジュールの実装

ASN.1 抽象構文と DTD の変換

ASN.1 抽象構文は，ある規則に従って DTD に変換される．表 1 に，その規則を示す．例えば，単純型に分類されるデータ型は，文字データとして扱われる．また，その名前は，逆変換のために，type 属性のデフォルト値として保持される．これらはほぼ 1:1 に対応するため，この変換とその逆は，一方を他方で単純に置き換えることで達成される．

ASN.1 抽象構文を解析する部分の実装には，Java Compiler Compiler (JavaCC) 0.8 [5] を使用した．

ASN.1 転送構文と DOM 木の変換

ASN.1 転送構文は，その構造を反映した DOM 木に変換される．その変換は，ASN.1 転送構文を先頭から走査し，TLV の解読と，それに対するノードの作成を繰り返すことで達成

表 1: ASN.1 抽象構文と DTD の変換規則

BOOLEAN などの単純型	`AAA ::= BOOLEAN`	`<!ELEMENT AAA (#PCDATA)>` `<!ATTLIST AAA type CDATA #FIXED "BOOLEAN">`

SEQUENCE と SET	`AAA ::= SEQUENCE {`
	`  b  BBB DEFAULT 10,`
	`  c  CCC }`
		`<!ELEMENT AAA (BBB?,CCC)>`
		`<!ATTLIST AAA type CDATA #FIXED "SEQUENCE">`
		`<!ATTLIST AAA default CDATA #FIXED "10,_">`

SEQUENCE OF と SET OF	`AAA ::= SET OF BBB`	`<!ELEMENT AAA (BBB*)>` `<!ATTLIST AAA type CDATA #FIXED "SET_OF">`
コメント	`-- This is a comment --`	`<!-- This is a comment -->`

される．TLV を解読する方法については，2.1 ですでに述べた．解読されたデータ型は要素として表現され，値はテキストとして表現される．図 4 に，図 2 の ASN.1 転送構文に対する DOM 木を示す．

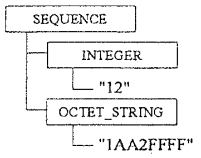

図 4: 図 2 の ASN.1 転送構文に対する DOM 木

逆変換は，以上の操作を逆に行うことで達成される．即ち，DOM 木を根から深さ優先で走査し，各ノードを TLV に符号化する．

TLV の符号化・解読には ASN.1 Runtime Encoding (ARE) [6] を，DOM 木の作成・操作には XML Parser for Java (XML4J) 2.0 [7] を使用した．

ラベルの付与・削除

DOM 木には，DTD に従って，より意味のあるラベルが付与される．その処理は，DOM 木に沿って内容モデルを再帰的に展開し，要素が確定したところから，その名前を書き換えることで達成される．内容モデルは，DOM 木の当該要素の名前が当該 type 属性のデフォルト値と一致する場合に，DOM 木が正当なものとなるように展開される．この再帰呼出しが停止するための条件は，以下の通りである．

- テキストに到達する，かつ
- 当該要素にテキストを挿入可能である．

基本的には，以上の方法でラベルが付与される．ただし，SET については，実際の要素が内容モデルに記述された順序で出現するとは限らないため，内容モデルに合うように要素を入れ換える操作も行う．図 5 に，図 1 の ASN.1 抽象構文から作成された DTD に従ってラベルが付与された，図 4 の DOM 木を示す．

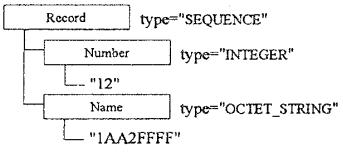

図 5: ラベルが付与された図 4 の DOM 木

ASN.1 抽象構文に曖昧さがあると，その DTD に従ったラベルの付与には複数の解が存在する可能性がある．現在のバージョンでは，その場合に，全ての解が返される．ASN.1 は，曖昧さを解消するために，データ型にタグを付与するメカニズム (タグ型) を提供している．この問題は，タグ型が実装されることで解消される．

逆変換は，DOM 木を根から走査し，各要素の名前を，その type 属性のデフォルト値で書き換えることで達成される．

4 実行例

図 6 に，実際の X.509 認証書に対するトランスレータの出力 (の一部) を示す．

```
<!ELEMENT Certificate (TBSCertificate,...)>
<!ATTLIST Certificate type ... "SEQUENCE">
<!ATTLIST Certificate default ... "_,_,_">
<!ELEMENT TBSCertificate (Version?,...)>
<!ATTLIST TBSCertificate type ... "SEQUENCE">
<!ATTLIST TBSCertificate default ... "0,...">
<!ELEMENT Version (#PCDATA)>
<!ATTLIST Version type CDATA #FIXED "INTEGER">
<!ELEMENT SerialNumber (#PCDATA)>
<!ATTLIST SerialNumber type ... "INTEGER">
...

<?xml version="1.0"?>
<!DOCTYPE Certificate SYSTEM "x509.dtd">
<Certificate>
  <TBSCertificate>
    <SerialNumber>1000</SerialNumber>
    <AlgorithmIdentifier>
      <Algorithm>1.2.840.113549.1.1.4</Algorithm>
      <Parameters>NULL</Parameters>
    </AlgorithmIdentifier>
    ...
```

図 6: X.509 認証書に対するトランスレータの出力

5 おわりに

本稿では，設計・実装した ASN.1/XML トランスレータについて述べた．また，その実行例も示した．今後の課題として，ASN.1 の全ての仕様を実装する，ASN.1 抽象構文を XML Schema に変換するなどが考えられる．

参考文献

[1] http://www.itu.int/itudoc/itu-t/rec/x/x200-499/x208.html

[2] http://www.w3.org/TR/REC-xml

[3] http://www.itu.int/itudoc/itu-t/rec/x/x200-499/x209.html

[4] http://java.sun.com/products/jdk/1.1/

[5] http://www.sun.com/suntest/products/JavaCC/

[6] http://assam.zurich.ibm.com/JavaCafe/

[7] http://www.alphaworks.ibm.com/formula/XML/

[8] 今村 剛. ASN.1/XML トランスレータの試作, 情報処理学会第 58 回全国大会講演論文集, vol. 3, pp. 259–260, 1999.