ROSEを用いる応用層プログラムの自動生成における複数プロトコル処理

1G−6

長谷川 亨　加藤 聰彦　野村 眞吾　鈴木 健二
国際電信電話株式会社 研究所

1. はじめに

OSIの応用層では、MHSやディレクトリなど、多くのプロトコルが遠隔操作サービス要素(ROSE)[1]を用いる。このような応用層プロトコルでは、プロトコル要素(PDU)とサービスプリミティブの構造はASN.1を拡張したRO-記法により記述され、またその動作はROSEプロトコルに準拠している。筆者らは、RO-記法を形式的に処理することにより、ROSEを用いた応用層プロトコル用プログラムを自動生成する拡張ASN.1コンパイラの試作を行なっている[2]。

応用層においては、複数のプロトコルがROSEと組み合わせて使用されることが多い。例えば、MHSのメッセージストア・アクセスでは、P2プロトコルとP7プロトコルがROSEとともに使用され、ROSE PDUのユーザデータとしてP7 PDUが、さらにそのユーザデータとしてP2 PDUがそれぞれ運ばれる。従って、ROSEを用いる応用層プログラムの全体を自動生成するためには、組み合わせて使用される複数の応用層プロトコルを処理する必要がある。本稿では、拡張ASN.1コンパイラにおける複数プロトコルの処理方式について述べる。

2. 拡張ASN.1コンパイラの概要

拡張ASN.1コンパイラを用いて実装される応用層プログラムの構成を図1に示す。応用層プロトコルの手順は、あらかじめ準備された手順プログラムライブラリにより実行される。また、PDUの符号化/復号ルーチン群、ユーザプログラムとの間でやりとりする応用層プリミティブのデータ型、手順プログラムライブラリのための制御情報が、対象となる応用層プロトコル毎に拡張ASN.1コンパイラにより生成される。

手順プログラムライブラリは、ユーザプログラムから応用層プリミティブを受け取ると、制御情報をもとに対応する符号化ルーチンを呼び出し、プリミティブのパラメータから直接応用層PDUを作成する。次に、ROSE手順に従って応用層PDUにROSEのプロトコルヘッダを付加し、ACSEまたはプレゼンテーションプログラムに渡す。また、ACSEまたはプレゼンテーションプリミティブを受信した場合は、その逆の処理を行なう。

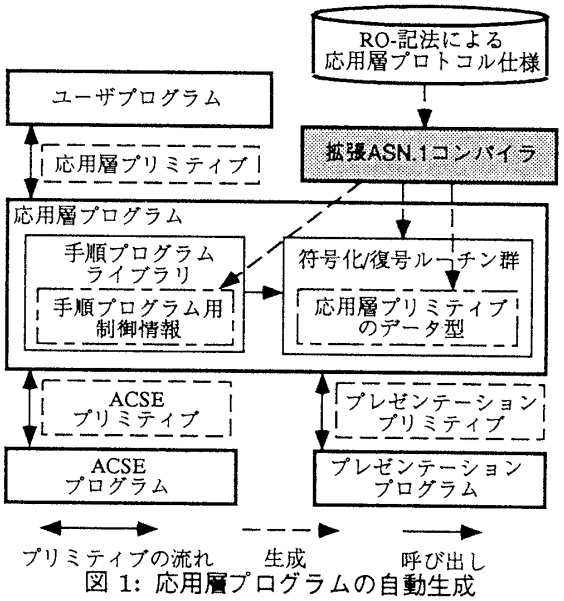

図1: 応用層プログラムの自動生成

3. 複数プロトコル処理の設計方針

拡張ASN.1コンパイラにおいて、ROSEを用いる複数のプロトコルをサポートする応用層プログラムを生成するために、以下の方針を立てた。

(1) MHSやCMIPなどのさまざまな応用層プロトコルに対処できるように、1つの応用層プロトコルのPDU内のANY型またはOCTET STRING型のパラメータを用いて、他の応用層プロトコルのPDUをユーザデータとして運ぶことができるようにする。

(2) 組み合わされる複数のプロトコルのPDUの構造を、RO-記法による1つの仕様として記述する。さらに、RO-記法の構文規則を拡張して、ユーザデータを運ぶパラメータと、運ばれるPDUの構造とのマッピングを規定できるようにする。

(3) 応用層プリミティブに、1つのROSE PDUで運ばれる複数プロトコルPDUのパラメータをすべて含むデータ型を持たせる。

(4) 1つのROSE PDUで運ばれる複数プロトコルPDUのパラメータを、一度に符号化/復号するルーチンを生成する。

"Automatic Implementation of Multiple Application Protocols over ROSE"
Toru HASEGAWA, Toshihiko KATO, Shingo NOMURA and Kenji SUZUKI
KDD R & D Laboratories

4. 複数プロトコル処理の実現方法

4.1 入力仕様

(1) ANY 型

ANY 型を用いて他プロトコルの PDU が運ばれる場合は、予約語 DEFINED BY により指定された、オブジェクト識別子型または整数型のパラメータによって、PDU のデータ型が識別される。そこで、図 2に示すように、予約語 DEFINED BY で識別用パラメータを指定した後に、運ばれる PDU の名前、そのデータ型、その PDU が選択される場合の識別用パラメータ値を列挙することとした。図 2の例では、オブジェクト識別子型の識別用パラメータ typeの値が { 2 6 3 3 0 } に設定された場合、PDU を運ぶパラメータ valueは、Upper1型となる。

```
LowerPDU  ::= SEQUENCE {
  type  OBJECT IDENTIFIER,
  value ANY DEFINED BY type
  (upper1  Upper1 { 2 6 3 3 0 },
   upper2  Upper2 { 2 6 3 3 1 } )  }
```

図 2: ANY 型の記述例

(2) OCTET STRING 型

OCTET STRING 型を用いた他プロトコルの PDU の転送は、MHS の P7プロトコルの PDU で、P2 プロトコルの PDU を運ぶ場合などに使用される。この場合、運ばれる PDU のデータ型は暗黙的に一つに決められている。そこで、OCTET STRING 型に対しては、以下のように、マッピングされる PDU のデータ型を仕様中で、予約語 TYPE の後に指定する方法を用いることとした。

```
UserData ::= OCTET STRING {TYPE=Upper}
```

4.2 応用層プリミティブのデータ型

(1) ANY 型

PDU を転送する ANY 型のパラメータに対しては、入力仕様で列挙された全ての PDU のデータ型を持つ共用体を対応させる。例えば、拡張 ASN.1 コンパイラは、図 2の記述から、図 3のデータ型を生成する。

```
struct Q_LowerPDU {
  OBJID  type;
  union {
    Upper1  upper1;
    Upper2  upper2; } value; };
```

図 3: プリミティブパラメータのデータ型の例

(2) OCTET STRING 型

転送する PDU のデータ型が記述された OCTET STRING 型のパラメータに対しては、指定されたデータ型を対応させる。

4.3 符号化/復号ルーチンの構成

(1) ANY 型

PDU を転送する ANY 型のパラメータに対して、拡張 ASN.1 コンパイラは、識別用パラメータの値を参照して PDU のデータ型を調べ、その PDU のデータ型に対応する符号化/復号を行なうルーチンを生成する。例えば、図 2の記述から生成される符号化ルーチンの構成を、図 4に示す。このルーチンは、まず SEQUENCE 型の識別子と長さを符号化してから (e_idlen)、識別用パラメータ typeを符号化する (e_objid)。次に、符号化する変数のオブジェクト識別子の値と、仕様に列挙されたオブジェクト識別子を、ルーチン objid_chkを呼び出して順に比較する。一致するものがあれば、対応する PDU のデータ型の符号化ルーチンを呼び出し、PDU を運ぶパラメータ valueを符号化する。

```
BOOLEAN  e_LowerPDU(p_var, edata, ... )
  LowerPDU  *p_var;  /* 符号化する変数  */
  PDU  *edata;  /* 符号化結果のオクテット列 */
  ...
{ /* 識別子と長さの符号化  */
  e_idlen(edata, ... );
  /* パラメータ type の符号化 */
  e_objid(&(*p_var).type, edata, ...);
  /* パラメータ value の符号化 */
    /* 一番目の候補 */
  if (objid_chk(&(*p_var).type, &upper1Id)
        == TRUE) {
    e_Upper1(&(*p_var).value.upper1,
             edata,...); }
    /* 二番目の候補 */
  else if (objid_chk(&(*p_var).type,
             &upper2Id) == TRUE) {
    e_Upper2(&(*p_var).value.upper2,
           edata, ...); }
  else { return (FALSE); }
  return (TRUE);
}
/* オブジェクト識別子の宣言 */
OBJID upper1Id = {{2, 6, 3, 3, 0 }, 5};
OBJID upper2Id = {{2, 6, 3, 3, 1 }, 5};
```

図 4: 複数 PDU を扱う符号化ルーチンの構成

(2) OCTET STRING 型

拡張 ASN.1 コンパイラは、OCTET STRING 型の識別子および長さの符号化/復号を行なった後に、指定された PDU のデータ型の符号化/復号ルーチンを用いて、コンテンツを符号化/復号するルーチンを生成する。

5. まとめ

本稿では、ROSE を用いる応用層プログラムを生成する拡張 ASN.1 コンパイラにおいて、複数プロトコルを処理する方式について述べた。これにより、応用層プログラムの自動生成をより完全にすることが可能となる。これまでに、本方式をサポートするコンパイラの実装を完了しており、MHS P2/P7 プロトコルの仕様から対応するプログラムを生成できることを確認している。今後、拡張 ASN.1 コンパイラおよび生成したプログラムの評価を進める予定である。最後に日頃御指導頂く KDD 研究所 浦野所長、眞家次長に感謝します。

参考文献

[1] CCITT Rec. X. 219, X.229, 1988
[2] 長谷川, 鈴木, 加藤, "ASN.1 コンパイラの拡張による OSI ROSE プロトコル・プログラムの自動生成," 信学研究会 IN92-103, Jan. 1992