マルチメディア通信と分散処理 79－14
（1996. 11. 15）

ASN.1 データベースのための ASN.1 符号化／復号処理系の設計と実装

小野 智弘　西山 智　堀内 浩規　小花 貞夫
国際電信電話株式会社

OSI ディレクトリ、OSI 管理や MHS 等の OSI の応用によっては、ASN.1 で定義されたデータ型を持つ情報をデータベースに格納する必要がある。筆者らは、これらの応用の開発を容易にするために、ASN.1 のデータを効率的に蓄積、検索できるデータベース (ASN1 データベース) を開発している。本稿では、ASN1 データベースのための ASN.1 符号化／復号処理系の設計と実装について述べる。ASN.1 処理系は、任意の ASN.1 定義を読み込み、運用中に動的なスキーマ変更が可能であり、速度の低下を抑えるために、従来の ASN.1 処理系に対して、1) コンパイラとインタプリタの併用機能、2) 部分符号化／復号機能、3)DER を用いた識別符号化機能、を補強している。

Design and Implementation of ASN.1 Translator for ASN.1 Database

Chihiro Ono, Satoshi Nishiyama, Hiroki Horiuchi and Sadao Obana
Kokusai Denshin Denwa Co.,LTD

As some OSI applications such as OSI Directory, OSI Management, MHS, need databases to store data defined by ASN.1(ASN.1 data), we are developing the database system which store ASN.1 data effectively (ASN.1 Database). This paper describes design and implementation of ASN.1 Translator for ASN.1 Database. The characteristics of this ASN.1 Translator are 1)Hybrid use of Compiler and Interpreter, 2)Introduction of partial encoding/decoding function, 3)Use of Distinguished Encoding Rules for data matching.

1　はじめに

抽象構文記法 1(ASN.1)[1][2] は、OSI(開放型システム間相互接続) 応用層で扱うプロトコルやデータ要素の情報を機種に依存することなく交換するための、データ型の記法と標準的な符号化規則を定めている。OSI ディレクトリにおける DSA(Directory System Agent)、OSI 管理における MIB(Management Information Base) や MHS(Message Handling Systems) におけるメッセージストア等の OSI の応用では、ASN.1 で定義されたデータ型を持つ情報 (以下、ASN.1 データと呼ぶ) をデータベースに格納する必要がある。

ASN.1 データを格納するために RDB や OODB 等の汎用 DBMS を直接用いる場合 [3][4] は、OSI 応用側で ASN.1 の定義と汎用の DBMS のスキーマ定義との間でデータ構造のマッピングを行なう必要があるため処理が複雑となる。また、汎用の DBMS のデータ格納構造、インデックス機構が ASN.1 のデータ構造に必ずしも適してはいないため、処理の高速化が困難である。

一方、OSI の応用毎に専用のデータベースを作成して使用する場合 [5][6] は、高速であるという利点がある一方で、動的に ASN.1 定義を変更することを想定していない、作成規模が大きくなるという問題があった。

そこで、ASN.1 定義をデータベースのスキーマとして、ASN.1 データを汎用的に格納、検索するためのデータベースの必要性が提案されている [7][8]。このうち、[7] では、ASN.1 の符号化／復号処理系としてコンパイラが生成した符号化／復号関数を使用して ASN.1 データの符号化／復号を行なう方式をとっている。このため、OSI 管理における管理オブジェクト (MO) や、OSI ディレクトリでのエントリ等などのように、データベース運用中に属性の追加、変更を行なう必要がある応用に使用するデータベースには適用できない。

これに対し、筆者らの提案する [8] の ASN.1 データベースでは、データベース運用中に動的に ASN.1 定義を変更可能とする、ASN.1 符号化／復号処理系 (以下、ASN.1 処理系と呼ぶ) を採用している。本稿では、ASN.1 データベースのための ASN.1 処理系の設計、実装について報告する。

2 ASN.1 データベースの概要

筆者らの提案する ASN.1 データベース [8][9][10] (以下、ASN.1 データベースと呼ぶ) は以下の特徴を持つ。

- 任意の ASN.1 定義をスキーマとして ASN.1 データを格納する。つまり、ASN.1 が DDL (データ定義言語) である。
- データベース運用中に動的にスキーマを変更することが可能。

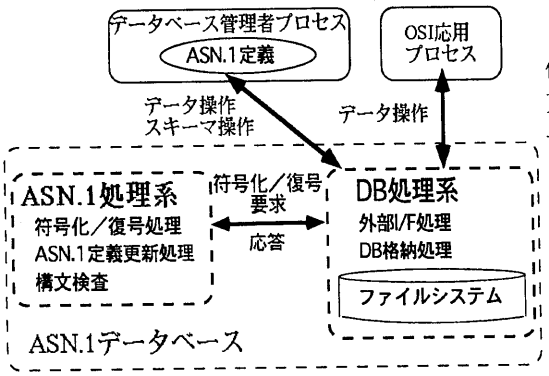

図 1: ASN.1 データベースの概念図

図 1に ASN.1 データベースの概念図を示す。1)ASN.1 データの符号化／復号処理、構文検査と ASN.1 定義の更新を行なう「ASN.1 処理系 (詳細は 3章)」、ならびに、2) それを利用して外部とのインタフェースを行なう機能、DB 格納処理を行なう機能を有する「DB 処理系」から構成される。DB 処理系は処理の高速化のために、以下の工夫を行なっている。

(a) 符号化バイト列でのデータ格納: データを通信路上の転送構文と同じ表現でファイルシステムに格納する。

(b) 格納要求単位でのデータ格納: データが格納された単位で検索される場合が多いことを考慮し、格納要求された ASN.1 データをそのまま格納する。つまり、構造体を構成する要素に分割しない。

(c) 一意識別符号化によるインデックス値の格納: BER[2] では一意に符号化できないため、インデックス値を一意に識別できる符号化列で格納する。

また、応用プロセスと ASN.1 データベース間の通信手順である DML (データ操作言語) も ASN.1 で記述し、Open, Fetch, Close, Modify, Insert, Delete の 6 つのデータ操作、並びに Select, Modify, Insert, Delete, CreateIndex, DeleteIndex の 6 つのスキーマ操作がある。

応用プロセスから ASN.1 データベースを利用する場合、応用プロセス側で応用操作 (OSI 管理の M-GET 操作等) と上記データベース操作の変換を行ない、ASN.1 データベースへ発行する。ASN.1 定義の変更は、データベース管理者プロセスから、変更すべき ASN.1 定義を入力することにより行なわれる。

3 ASN.1 処理系の基本設計

ASN.1 データベースのための ASN.1 処理系では、任意の ASN.1 定義を読み込み、動的にスキーマを変更可能とする機能を実現すると共に、それによる符号化／復号処理速度を最小限の低下に抑える必要がある。

従来からあるデータ転送のための ASN.1 の処理系 [11][12] は、データベースで用いる処理系としては、機能面で十分ではないため、以下の項目を補強した ASN.1 処理系を設計した。なお、ここでは、ASN.1 処理系は他の OSI 応用でも用いることのできるライブラリとして実現することとした。

(1) コンパイラとインタプリタの併用
(2) 部分符号化／復号
(3) 一意識別符号化

3.1 コンパイラとインタプリタの併用

一般に、OSI応用のASN.1定義は、運用中に動的に変更され得る部分(動的定義部)と、されない部分(静的定義部)に分けることができる。ASN.1データベースでは、図2のように、静的、動的定義部をそれぞれ以下の様に使い分ける。

[静的定義部]「応用プロセスに提供するデータベース操作プロトコル」、「スキーマ操作に用いるメタスキーマ定義(スキーマのためのスキーマ定義)」

[動的定義部] 格納するデータのASN.1定義

ASN.1処理系では、静的定義部は高速化のために、コンパイラを用いて実行前にあらかじめ符号化/復号関数を生成し、実行時にそれを用いて符号化/復号処理を行なう。一方、動的定義部は符号化/復号関数をあらかじめ固定できないため、インタプリタが実行中に動的に符号化/復号処理を行なう。

そのためにコンパイラ、インタプリタが共通に利用できる「ASN.1定義の内部表現」が重要であり、これについては4.2章で述べる。

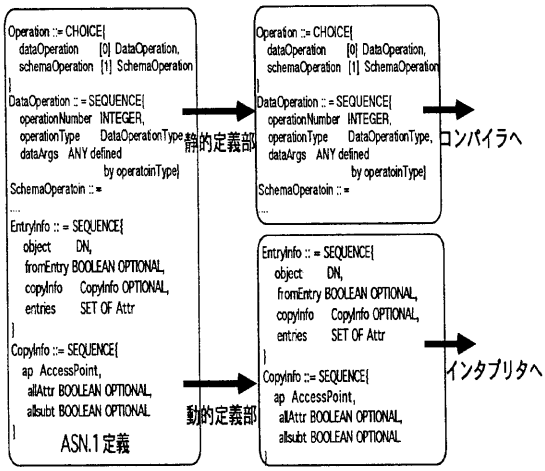

図 2: ASN.1 定義

3.2 部分符号化 / 復号

ASN.1データベースでは、格納されたデータは通信の上の転送構文(符号化列)と同じ表現であり、中身を判別するためには復号する必要がある。

例えば、下記のような条件に合致したデータを変更するModify操作では、対象となるデータの条件値と、データの該当箇所を比較する必要があり、この際にデータの一部の部分復号を行なう。また、古い値と新しい値を変更する際に、該当部分だけを復号して新しい値と入れ換えるために行なうために部分符号化を行なう。

```
DataModifyArgs::=SEQUENCE{
    scopeTarget    ScopeTarget,
    condition      ConditionGroup,
    modifyValueG   ModifyValueG}
```

[11]のように、データ転送のためのASN.1処理で用いられるヘッダ部分を対象とした部分復号のみでは、符号化列と復号された状態とが混在できる機能、切出しのみを行なう機能等、データベースで必要となる部分符号化/復号処理には適用できないため、新たに補強することとした。

これを実現するためには「ASN.1データの内部表現」が重要であり、これについては4.3章で述べる。

3.3 一意識別符号化

ASN.1データベースでは、検索の高速化のために、インデックス値と比較値とを符号化列での比較を行なう。このため、インデックス値、符号化列の双方を、符号化列が一意とならないBER[2]から、一意となるDER[2]に変換する機能を設けた。

4 ASN.1 処理系の実装

上記の設計方針に基づき、Solalis2.4上でC言語を用いてASN.1処理系を実装した。

4.1 ASN.1 処理系のソフトウエア構成

ASN.1処理系のソフトウエア構成は図3に示すように、コンパイラとインタプリタから構成される。インタプリタはASN.1定義処理部とデータ処理部から構成されるライブラリで、ASN.1データベース実行系に組み込まれる。

コンパイラはデータベース運用前にASN.1定義の静的定義部に該当する符号化/復号関数と、符号化/復号時に利用する「ASN.1定義の内部表現(4.2章参照)」を生成する。

インタプリタでは、1)DB処理系からのASN.1定義操作要求に対して、ASN.1定義処理部が構文解析を行ない、「ASN.1定義の内部表現」に対して追加、

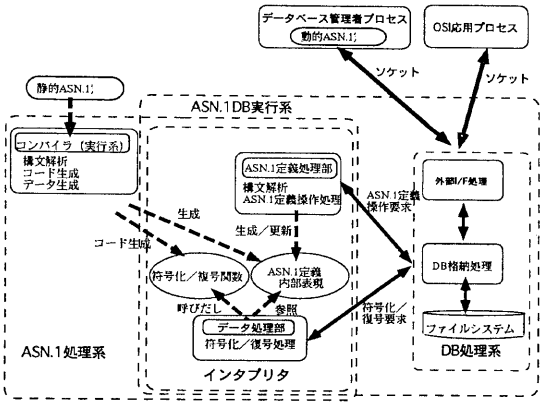

図 3: ASN.1 処理系のソフトウエア構成

更新を行う。また、2)DB 処理系からの符号化／復号要求に対して、データ処理部が「ASN.1 定義の内部表現」とコンパイラの生成した「符号化/復号関数」を用いてデータの符号化/復号処理を行う。この際、復号されたデータは「ASN.1 情報構造体 (4.3章参照)」を用いてデータベース内部で交換される。

4.2 ASN.1 定義の内部表現

ASN.1 データの符号化／復号時にコンパイラとインタプリタが共通に利用する ASN.1 定義の内部表現は、図 4に示すように、1) 個々の ASN.1 型参照名に対応する「ASN.1 型ノード」、2) 複数の要素を持つ ASN.1 構造型に対応し、各 ASN.1 型ノードを結び付ける「要素情報テーブル」、ならびに、3) 該当する「ASN.1 型ノード」を高速に導くための「検索テーブル」からなる。

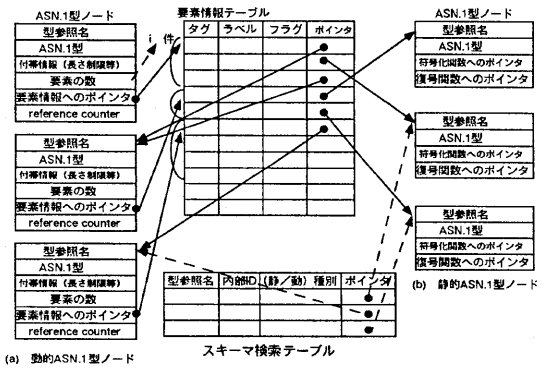

図 4: ASN.1 定義の内部表現

ASN.1 型ノードには、静的定義部用、動的定義部用にそれぞれ静的 ASN.1 型ノードと動的 ASN.1 型ノードがある。動的 ASN.1 型ノードは、型参照名、ASN.1 型、付帯情報 (ENUMERATED 型の定数値リストやサブタイプによる最大データ長制限等)、型を構成する要素の数、および要素情報 (タグ、付与されたラベル、OPTIONAL 指定の有無、静的定義/動的定義を識別するフラグ、その ASN.1 型ノードへのポインタ) のリストへのポインタからなる (図 4(a))。また、静的 ASN.1 型ノードは型参照名、ASN.1 型、生成された符号化関数および復号関数へのポインタからなる (図 4(b))。

4.3 ASN.1 データの内部表現

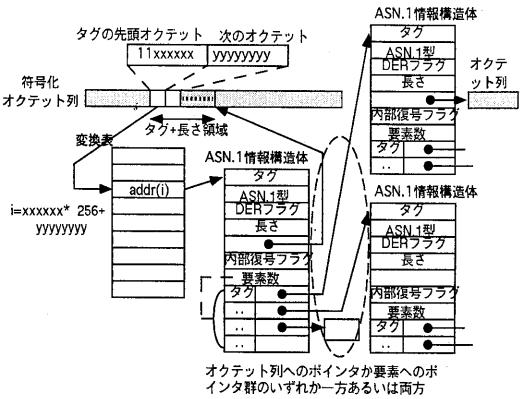

図 5: ASN.1 情報の内部表現

部分符号化/復号を可能とするために、ASN.1 データの内部表現には、1) 符号化列の表現と、2) 復号された表現 (ASN.1 情報構造体) が含まれる (図 5)。ASN.1 情報構造体は表 1に示す情報を持ち、ASN.1 型単位に ASN.1 データを保持する。ASN.1 定義の内部表現では、符号化されたオクテット列の一部が既に復号されていることを示すために、通常は使用されていない ASN.1 の私的タグを利用する。具体的にはタグの先頭 1 オクテットのうち私的タグを表現するための先頭 2 ビットを除く 6 ビットと、それに続く 1 オクテットを使用して ASN.1 情報構造体を識別する。

表 1: ASN.1 情報構造体の持つ情報

フィールド			項目内容
タグ			符号化後のタグ値 (注1)
ASN.1型			ASN.1型の種別(SET, INTEGER, Tagged等)を示す。
DERフラグ			既にDER符号化状態になっているか否かを示す。
符号化オクテット列の情報(注2)		長さ	オクテット列の長さ
		ポインタ	オクテット列へのポインタ
		内部復号フラグ	オクテット列内でさらに部分復号されているか否かのフラグ
復号された情報(注2)	要素数		その型に含まれる要素の数
	要素の情報	タグ	要素の符号化後のタグ値 (注1)。私的タグの値ならばオプショナルな値が省略されたことを示す
		ポインタ	要素のASN.1情報構造体へのポインタ、あるいは要素が基本形汎用型の場合はその情報を格納する領域へのポインタ

注1) 構造形/基本形を示すフラグはASN.1定義からは決定できないため除く
注2) 符号化オクテット列の情報と復号された情報の少なくとも一方が存在すること

4.4 データベース内部での処理の流れ

4.4.1 データ処理の流れ

OSI応用プロセスから、PDU(Protocol Data Unit)が入力されると、DB 処理系の外部 I／F 処理部が ASN.1 処理系を使用して（＝2 回の静的復号処理）PDU を復号して操作種別、操作パラメタを取得し、操作要求を DB 格納処理部へ渡す。

DB 格納処理部では、渡された操作パラメタを用いて、ASN.1 処理系を使用して DB 格納処理を行なう。(例えば、DataDelete 処理では、まず、ASN.1 処理系を呼んで、条件値を DER で表現し（条件値の個数回)、その条件値とインデックスの値とを比較する。ついで、合致したレコード件数分のデータを消去する。）結果を外部 I／F 処理部へ返す。

外部 I／F 処理部では、答を PDU に加工し（＝2 回の静的部分復号)、応用プロセスへ返す。

4.4.2 スキーマ処理の流れ

データベース管理者プロセスから、PDU が入力されると、DB 処理系の外部 I／F 処理部が ASN.1 処理系を使用して（＝2 回の静的復号処理）PDU を復号しながら操作種別、操作パラメタを取得し、操作要求を DB 格納処理部へ渡す。

DB 格納処理部では、ASN.1 定義処理部に対して ASN.1 定義を渡し、ASN.1 定義処理部にて ASN.1 定義の内部表現を更新する。DB 格納処理部では、また、永続性を実現するため、ファイルシステムに ASN.1 定義を符号化して格納する。

5 ASN.1 処理系の評価と考察

5.1 コンパイラとインタプリタの併用について

(1)ASN.1 処理系では、ASN.1 定義の静的定義部と動的定義部をそれぞれ、コンパイラとインタプリタで処理することによる高速化を図った。今回はオブジェクト定義を全て動的定義部として扱ったが、OSI 管理における system や log 等の共通に使用される管理オブジェクト等の、定義が変更されないことが保証されるオブジェクトがあれば、静的定義部として扱うことにより、さらに高速化を図ることが可能である。

(2)ASN.1 データベースでは、DB 内部の処理の簡便化を図るため、コンパイラとインタプリタが共通の内部表現を用いた。このため、コンパイラとインタプリタの処理速度の差は、直接関数を呼ぶか、参照する型を 特定するためのポインタをたどるか、によりコンパイラが高速となり、構造が複雑なデータになるほどその差は拡大する。一方、構造が単純なデータでは小規模なデータでは初期化等のオーバヘッドのため、速度が 80 %程になるなど、それほど効果は現れない。

5.2 部分復号機能の効果について

部分復号処理では、全体のバイト列のうちの復号部分が少ないほど速度は高速になり、また、該当箇所の切出しのみで済む場合は、さらに高速となる。そこで、全長 200 バイト、CHOICE と SEQUENCE からなる 3 段の入れ子構造の ASN.1 データを用いて、全体の符号化列中の復号バイト数の比率 (部分復号率) をパラメタとして、復号速度を測定した結果を図 6に示す。

「部分復号指定」で全てを復号する場合に比べて、部分復号率が小さくなるほど復号時間が短縮できる。一方、「全部復号指定」で復号を行なった場合、復号開始位置の特定処理が必要ないため、「部分復号指定」で全てを復号するのに比べて、85 %の所要時間で済む。図 6より、部分復号率が 75 %以下であれば、部分復号指定による方が高速となることがわかる。このことから、マルチメディアデータ等、大規模なデータを復号する際には、部分復号率が低くなるため、所要時間比がさらに小さくなると言える。

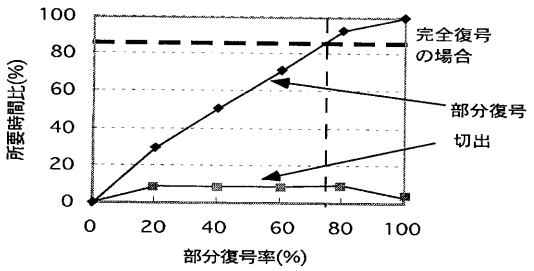

図 6: 部分復号率に応じた速度の変化

5.3 処理系の全体性能について

ASN.1 データベースの性能は頻繁に利用される ASN.1 処理系の性能に大きく依存している。ASN.1 データベースでの ASN.1 処理系の使用状況を表にすると以下のようになる。

表 2: ASN.1 処理系の使用状況

操作	全体	静的符号化	動的符号化	静的部分復号	動的部分復号	動的構文検査
DataOpen	4+(X+1)Y	2	Y	2	XY	0
DataFetch	4+Y+W	2	0	2	W+Y	0
DataDelete	4+(X+1)Y	2	Y	2	XY	0
DataInsert	6+I	2	1	2	I	1
DataModify	4+(2X+1)Y	2	XY	Y+2	XY	0

X=合致したレコード件数：Y=条件数
I=インデックスの種類：W=抽出フィールド数

例えば、DataOpen で各パラメータ (X,Y,I,W) を 1 とした場合、操作一回につき、ASN.1 処理系を 6 回呼んでいる。処理系を呼んだ回数の内訳は、4 回は静的定義、2 回は動的定義である。また、6 回とも部分復号だけで済んでおり、さらにそのうち 1 回は復号せずに、切出しだけで済んでいる。

従って、復号時間の合計は、5.1章、5.2章の結果を用いて、全て全部復号かつ動的定義で処理した場合の復号時間に比べて、18 %に抑えられていると推定できるため、本 ASN.1 データベースで採用している ASN.1 処理系で用いた方式が有効であると考えられる。

6 おわりに

本稿では、ASN.1 データベースための ASN.1 符号化／復号処理系の設計と実装を報告した。ASN.1 符号化／復号処理系は、任意の ASN.1 定義を読み込み、運用中の動的なスキーマ変更が可能であり、速度の低下を抑えるために、従来の ASN.1 符号化／復号処理系に対して、1) コンパイラとインタプリタの併用機能、2) 部分符号化／復号機能、3)DER を利用した一意識別符号化機能を補強したものである。最後に、日頃御指導頂く、KDD 研究所村上所長 に感謝します。

参考文献

[1] ITU-T 勧告 X.680:Specification of ASN.1, 1992

[2] ITU-T 勧告 X.690:Specification of Encoding Rules of ASN.1,1992

[3] 依田育生, 藤井伸朗 “伝送網オペレーションにおける管理情報ベース (MIB) の構成法”, 電子情報通信学会論文誌 Vol75 B-I, No8.pp517-527, 1992

[4] 桐葉佳明, 中井正一郎, 有馬啓伊子, 井原慈子, 栗山博, 長谷川聡, “管理情報ベース (MIB) の開発支援環境”, 情報処理学会第 8 6 回データベースシステム研究会報告, pp69-77, 1991

[5] 西山 智, 堀内 浩規, 横田 英俊, 小花 貞夫, 鈴木 健二, “OSI 管理情報ベース (MIB) 用データベースの設計と実装”, 情報処理学会第 9 5 回データベースシステム研究会報告, 1993

[6] 西山 智, 小花 貞夫, 堀内 浩規, 鈴木 健二 “拡張可能 DBMS 構築技法に基づく高速 OSI ディレクトリ用 DBMS の設計と評価”, 情報処理学会論文誌 Vol34, No6, 1993

[7] 春本 要, 仲秋 朗, 塚本 昌彦, 西尾章治郎, 宮原 秀夫, “抽象構文記法 1(ASN.1) に基づくデータベースシステム”, 情報処理学会第 9 7 回データベースシステム研究会報告, pp41-50, 1994

[8] 西山 智, 堀内 浩規, 小花 貞夫, 鈴木 健二, “ASN.1 データベースの実現方式に関する一考察”, 情報処理学会第 4 9 回全国大会, pp.4-255-256, 1994

[9] 西山 智, 堀内 浩規, 小野 智弘, 小花 貞夫, 鈴木 健二, “ASN.1 データベースのための高速な ASN.1 処理系の設計”, 情報処理学会第 5 0 回全国大会, pp.1-159-160, 1995

[10] 小野 智弘, 西山 智, 堀内 浩規, 小花 貞夫, “ASN.1 データベースにおけるスキーマ管理方式”, 情報処理学会第 5 3 回全国大会, pp.3-261-262, 1996

[11] 中川路 哲男, 宮内直人, 勝山光太郎, 水野 忠則, “OSI 抽象構文記法支援ソフトウエア APRICOT の開発と評価”, 電子情報通信学会論文誌 Vol.J73-D-I No.2 pp.225-234, 1990

[12] 長谷川亨, 野村真吾, 堀内浩規,”ASN.1 支援ツールの開発-コンパイラ及びエディタ-”, 情報処理学会第 3 9 回マルチメディアと分散処理研究会報告, pp.1-8, 1988