システムソフトウェアと
オペレーティング・システム 64−26
マルチメディア通信と分散処理 65−26
（1994. 5. 20）

OODBMSを用いたASN.1データベースの実現

仲秋 朗† 春本 要† 齋藤 淳* 塚本昌彦** 西尾章治郎†

†大阪大学工学部情報システム工学科 *奈良先端科学技術大学院大学情報科学研究科
**シャープ（株）技術本部

抽象構文記法1(ASN.1)は，マシン間で曖昧性なくデータ交換を行なうためのデータ構造記述言語であり，通信をはじめ，マルチメディア，ゲノムデータなど様々なアプリケーション分野で用いられている．本稿では，このようなアプリケーション間で交換されるASN.1データを蓄積するためのデータベースシステムの設計および実装について論ずる．特にASN.1のデータ構造記述能力は，従来データベースシステムが取り扱ってきたデータモデルと比べると強力で柔軟であるため，データ型の変換やアプリケーションプログラムが複雑になるなどの問題が生じる．本稿では，システムをオブジェクト指向データベース管理システム (Object-Oriented DataBase Management System: OODBMS) 上で実現し，これらの問題の解決を試みる．

Design and Implementation of ASN.1 Database using OODBMS

Akira NAKAAKI† Kaname HARUMOTO† Jun SAITOH*
Masahiko TSUKAMOTO** Shojiro NISHIO†

†Department of Information Systems Engineering, Faculty of Engineering, Osaka Univ.
*Graduate School of Information Science, Nara Institute of Science and Technology
**Information Technology Research Laboratories, SHARP Corp.

ASN.1 (Abstract Syntax Notation One) is a standard language to describe data structures exchanged among systems. It is designed for excluding the ambiguity of data representation in the heterogeneous computer architectures and used in a wide variety of areas such as computer networks, multimedia databases, and genome databases. In this paper, we discuss the design and implementation issues of a new database system, called an ASN.1 database, which can handle ASN.1 data exchanged among such applications. Since the type definition capability of ASN.1 is much stronger than those of the conventional data models, there are several problems for their implementation such as the complexity of data translation and application programming. We describe how we solved these problems using an OODBMS (Object-Oriented DataBase Management System: OODBMS) as a kernel system in our implementation.

1 はじめに

抽象構文記法1(Abstract Syntax Notation One: ASN.1) は，国際標準化機構 (ISO) によって定められたデータ構造記述言語であり，言語仕様 [5] および交換するデータを符号化するための規則 [6] から成る．ASN.1 は，アーキテクチャの異なるマシン間において曖昧さのないデータ交換を可能にすることを目的とし，あらゆるアプリケーション間で交換されるデータの構造を表現できるように強力なデータ構造記述能力を有している．このような特性から ASN.1 は多岐の分野で利用されており，交換する電子文書のデータ構造を定めた ODA (Open Document Architecture) や，メッセージ交換のサービスを提供する MHS (Message Handling System) などの OSI アプリケーション，TCP/IP でも SNMP (Simple Network Management Protocol) のネットワーク管理情報などが ASN.1 で記述されている．また，通信以外のマルチメディア [2, 13] やゲノムデータ (遺伝子情報) [15] の分野でも ASN.1 が利用されている．

ASN.1 の利用分野は年々拡大しており，様々なアプリケーションの開発に伴ってアプリケーション間で交換されるデータを蓄積して有効利用したいという要求が出ている．それに対しこれまでにも，ODA 文書を格納するデータベースの構築 [8, 9] や管理情報ベース (Management Information Base: MIB) 用データベースの構築 [7, 10], OSI ディレクトリ用データベースの構築 [11] に関する研究が行なわれてきた．これらのシステムが個々のアプリケーションに固有のデータベース構成によりパフォーマンスの向上を目指しているのに対し，著者らはむしろアプリケーション開発効率の向上を目指し，ASN.1 でデータ型を記述するデータベースシステム ASN.1/DB を提案した [4]．特に，ASN.1 のデータ構造記述能力は，従来データベースシステムが取り扱ってきたデータモデルと比べると強力で柔軟性があるため，データ型の変換やアプリケーションプログラミングが複雑になるなどの問題が生じる．本稿では，システムを商用オブジェクト指向データベース管理システム (Object-Oriented Database Management System: OODBMS) ObjectStore†[12] で実現し，これらの問題の解決を試みる．

```
PersonalRecords DEFINITIONS ::=
BEGIN
Person ::= SEQUENCE {
        name        OCTET STRING,
        age         INTEGER,
        belongTo    BelongTo
}
BelongTo ::= CHOICE {
        univ    [0] University,
        comp    [1] Company
}
University ::= SEQUENCE {...}
Company ::= SEQUENCE {...}
END
```

図 1: ASN.1 によるモジュール定義の例

以下では，2 章で ASN.1/DB の概要を述べ，3 章で ASN.1/DB の構成と実装について述べる．最後に，4 章でまとめと今後の課題について述べる．

2 ASN.1/DB の概要

2.1 ASN.1 の概要

ASN.1 は，アーキテクチャの異なるマシン間でも曖昧さのないデータ交換を行なえるように，ネットワーク上におけるデータ表現の共通形式を定めたものであり，交換されるあらゆるデータのデータ構造を記述する能力を提供している．ネットワークの管理情報やアプリケーションで扱うデータの構造など，共通の目的に関係している ASN.1 記述の集合をモジュールと呼び，データ構造はモジュール単位で定められる．図1にモジュール定義例を示す．この例は，人とその所属に関するデータの構造を定義するものであり，Person 型，BelongTo 型，Uni-

†ObjectStore は Object Design, Inc の登録商標である．

versity 型，Company 型をもつPersonalRecordsという名のモジュールである．Person 型は，name，age，belongTo の 3 つのフィールドから成る SEQUENCE 型である．

2.2 ASN.1/DB の必要性

ASN.1 の適用分野の拡大に伴い，アプリケーション間で交換するデータを蓄積利用したいという要求が増大している．これに対しこれまでにも，MHS や OSI ディレクトリなど特定のアプリケーション専用のデータベースシステムの構築に関する研究が行なわれている．これらのシステムがパフォーマンスの向上を目指しているのに対し，本稿では，あらゆる ASN.1 データを扱えるデータベースプログラミング言語，および，ASN.1 データを格納するための統一されたプラットフォームを提供する汎用的なデータベースは，効率の良いアプリケーション開発環境を提供すると考える．そこで，以下の特徴をもつデータベースシステム ASN.1/DB を構築する．

- ASN.1 で記述されたデータ構造をもつデータを格納することができる．
- ASN.1 のデータ型をサ　ートするデータベースプログラミング言語を提供する．

2.3 ASN.1/DB の全体像

ASN.1 データを格納するための ASN.1/DB の全体像を図 2に示す．ASN.1 を利用しているあらゆるアプリケーションが簡単にデータベースを利用できるように，アプリケーションのデータ構造定義をそのままデータベーススキーマ定義とする．また，効率の良いアプリケーション開発を可能にするために，ASN.1 のデータ型をサ　ートするデータベースプログラミング言語を用いてソースコードの記述を行なう．

3 ASN.1/DB の設計および実装

ASN.1 の多くのアプリケーションにおいて，ASN.1 によるデータ構造定義は複雑な入れ子構造になっている．そこで，このようなデータ構造の

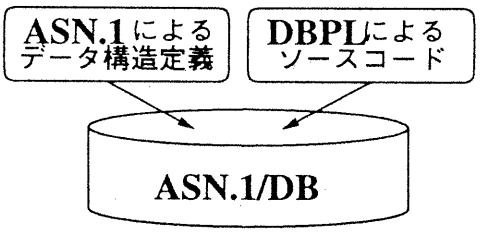

図 2: ASN.1/DB の全体像

表現能力をもつ OODBMS を利用して ASN.1/DB を実現する．

3.1 全体構成

C++言語に基づく OODBMS である ObjectStore を利用して ASN.1/DB を設計し，実装した (図 3参照)．ASN.1/DB の構成要素として以下のものが挙げられる．

ASN.1 コンパイラ： ASN.1 で定義されたモジュールを，3.2節で述べる ASN.1 データ操作のためのメンバ関数等をもつ C++言語のクラス定義に変換するコンパイラ．C++言語のクラス定義を含むヘッダファイル (.h)，および，データの符号化，復号化，表示等のためのメンバ関数定義 (.C) を生成する．符号化/復号化のための関数をもつ C++言語のクラス定義に変換するコンパイラ snacc[14] を改変することで実現した．

言語処理系： ASN.1 データを操作するために，プログラミング言語で記述されたプログラムを C++言語のソースコードに変換するコンパイラ．

3.2 ASN.1 コンパイラの機能

ASN.1 コンパイラは，ASN.1 によるデータ構造定義を C++言語のクラス定義に変換し，また，ASN.1 データを操作するための関数をクラスのメンバ関数として定義する．以下，どのような変換を行なうかについて述べる．

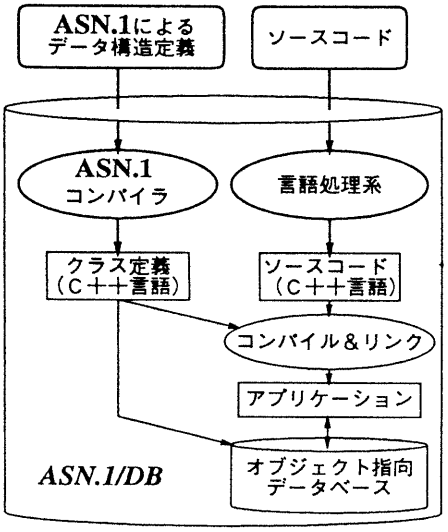

図 3: ASN.1/DB の全体構成

3.2.1 データ構造の変換

ASN.1 におけるデータ型は，ASN.1 コンパイラによって次のように変換される．

単純型： システムが提供する基本クラス．

構造型およびメタ型： 新たなクラスとして定義．

ここでは，データ型変換の問題点の一つとして CHOICE 型がどのようなクラス定義に変換されるかを説明する．図 1に示されるデータ構造定義のうち，CHOICE 型である BelongTo 型は University 型か Company 型を任意に選択できる型であるが，C++言語はこれに対応するデータ型を有しておらず別のデータ構造に変換する必要がある．実現方法には次の二通りが考えられている [8, 9]．

1. 共用体部 (C++言語の union) と型の選択を識別するタグフィールドから成るクラスによって実現する．共用体部のデータ型が変化するたびにタグの値をプログラマが更新しなければならないという問題がある．
2. CHOICE 型の各要素の上位クラスを定義することで，各要素をそのクラスのデータとして扱う．データ内容を参照するには各要素の型に明示的に型変換するか，メンバ関数を用いなければならないといった問題をもつ．

ASN.1/DB では，構造の簡明さ，クラス定義の自動生成の容易さといった点から 1 の方法を用いた．

3.2.2 オブジェクト集合 extent の定義

OODBMS に格納されたオブジェクトを取り出すにはオブジェクト識別子が必要であり，OODBMS では，プログラムがこの識別子情報を得るために extent と呼ばれる集合を導入している [1]．extent はそれが定義されたクラスの全てのインスタンスの集合であり，クラス A で extent を定義すると，A :: *extent* はクラス A の全てのインスタンスの識別子を保持する集合となる．新たにデータベースに格納されたオブジェクトは全てこの集合 extent にに挿入され，データベースからオブジェクトを削除すると extent からも削除される．よって，データベースへの問合せはこの集合への操作に置き換えられる．ASN.1 コンパイラは，この集合 extent を各クラスで定義し，挿入および削除の操作をそれぞれ生成関数，消去関数内において定義する．

3.2.3 属性の定義

属性を埋め込み型として定義した場合，埋め込まれたオブジェクトはオブジェクト識別子をもたないため，問合せの対象とすることができない．そのため，例えば，OCTET STRING 型である name が Person に埋め込まれているとき，`select x.name from x <- Person...` といった問合せができない (一般に集合は，データ本体ではなくオブジェクト識別子を保持する)．そこで，基本型，構造型に関係なく属性を全て他の型への インタとして定義することにより，属性の値の集合を扱うようなデータの部分的な操作を可能にする．

3.2.4 メンバ関数の定義

ASN.1 コンパイラは，ASN.1 データの操作記述をより容易にするためにいくつかのメンバ関数を定義する．以下にその一部を列挙する．

等価性判定関数 Equiv()：データベースへの問合せではオブジェクト値の等価性が重要になる．そこで，同じ型の他のオブジェクトとの値等価性を判定する関数を提供する．

寿命変換関数 PersistentCopy(), TransientCopy()：オブジェクトには，一時的オブジェクトとデータベースに格納されている永続的オブジェクトがあり，アプリケーションによってはこの“寿命”を変換したい場合がある．この変換を値の等しいオブジェクトを生成することで実現する．

集合要素判定関数 contains()：問合せにとって重要な機能であり，引数に与えられたオブジェクトが集合に含まれるかどうかを調べる．

3.3 データベースプログラミング言語

今回の実装では C++言語をプログラミング言語とする OODBMS を利用しているので，ASN.1/DB に格納されたデータ操作の記述に C++言語を使用することができる．しかし，ASN.1 のデータ構造から変換された C++言語の構造は一般に複雑であり，そのためこの構造をもつデータを直接操作するための記述も複雑になる．そこで ASN.1 データ操作の記述をより容易にするための言語として，LL (Light Language for ASN.1/DB) および ASN.1/PL を提供する．

3.3.1 LL

LL は，C++言語をベースに ASN.1 データ操作において特に複雑な問合せと集合操作を拡張した言語であり，構造化問合せ記述 (select-from-where 式) および集合操作構文 (foreach 文) を提供する．LL は，その他の処理は全て C++言語で記述することを前提とする．

構造化問合せ記述 select-from-where 式：データベース内のデータの操作は集合 extent の操作に置換されるので，この集合内のデータ操作を容易にするための構造化問合せ記述を提供する．問合せ記述内では，ASN.1 のデータ構造をドット記述で操作することができるものとする．図 4は，図 1の Person 型のデータに対し所属が大学である人の名前と年齢の組を求める記述の例である．“isa” は CHOICE 型がどの型をとっているかを識別するための演算子であり，図 4においては Person 型である x の属性 belongTo が University 型であれば真を，それ以外の場合には偽を返す．

```
SetOfPerson*  persons;
students =
   select x.name, x.age
   from   x <- persons
   where  x.belongTo isa University;
```

図 4: LL のプログラム例

集合操作構文 foreach： foreach 文は集合型から要素を順に取り出して与えられた命令を要素に対して実行する．foreach 式において，要素が代入される繰り返し子の型を明示的に宣言する必要はない．

3.3.2 ASN.1/PL

LL は，拡張構文を C++言語に“埋め込む”ため十分な静的型検査を実現できないという問題点をもつ．一般に，データベースシステムがもつデータ型とプログラミング言語がもつデータ型が必ずしも一致しないため，プログラミング言語の静的型検査機構を十分に利用できず様々な型誤りを引き起こすという問題がある．このような問題を解決するための言語 ASN.1/PL を ASN.1/DB のもう一つのデータベースプログラミング言語として提供する．ASN.1/PL に関しては文献 [4] で論じられている．

3.4 言語処理系

ASN.1 データに対する操作のための記述を，OODBMS のデータに対する操作に変換するための処理系を提供する．LL および ASN.1/PL の処理系は以下の方針に基づいて変換する．

- 問合せ式は，問合せ内で使用される変数を引数とし結果を返り値とする関数を定義し，その関数の呼び出しに置き換える．
- 問合せ結果が新たな型をもつ場合それに対応するクラス定義を生成する．
- foreach 文を for ループに置換し繰り返し子の型宣言を生成する．

4 おわりに

本稿では，ASN.1 データを格納するためのデータベースシステム ASN.1/DB の OODBMS 上での実現について論じ，ASN.1 コンパイラおよび言語処理系について述べた．そして，実装上問題となった点を明らかにし，その解決方法について述べた．ASN.1/DB の実現により，蓄積された ASN.1 データを扱うアプリケーションの開発が容易になった．

今回の実装では既存の OODBMS を利用して ASN.1/DB を実装し，物理記憶構造やアクセス制御，並行処理制御は既存のものをそのまま用いた．今後は，ASN.1 特有の性質を考慮した物理記憶構造などをもつデータベースシステムの開発を行なっていく予定である．

謝辞

本稿をまとめるにあたり，有益な御助言を頂いたシャープ株式会社 田中理恵子氏ならびに奈良先端科学技術大学院大学 山口英 助教授に感謝致します．

参考文献

[1] Cattell, R.G.G., the Object Database Standard: ODMG-93, Morgan Kaufmann Publishers, 1993.

[2] 半田晶彦, 山岸靖明, 中川透, マルチメディア/ハイパーメディア符号化標準 (MHEG) の動向, 電子情報通信学会技術研究報告 DE92-42, 1992.

[3] 春本要, 抽象構文記法 1(ASN.1) に基づくデータベースシステム, 大阪大学基礎工学部情報工学科修士学位論文, 1994.

[4] 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 97 回データベースシステム研究会報告, 1994.

[5] ISO 8824, Information Processing Systems — Open Systems Interconnection — Specification of Abstract Syntax Notation One (ASN.1), 1st edition, 1987.

[6] ISO 8825, Information Processing Systems — Open Systems Interconnection — Specification of Basic Encoding Rules for Abstract Syntax Notation One (ASN.1), 1st edition, 1987.

[7] 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 86 回データベースシステム研究会報告, 1991.

[8] 増永良文, マルチメディア文書の標準データベース格納構造の考察, 情報処理学会第 93 回データベースシステム研究会報告, 1993.

[9] 増永良文, ODA に基づくマルチメディア文書データベースの格納方式, 情報処理学会第 47 回全国大会, 1993.

[10] 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 95 回データベースシステム研究会報告, 1993.

[11] 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 Vol.34, No.6, 1993.

[12] Object Design, Inc., ObjectStore リリース 2.0 リファレンス マニュアル, 1993.

[13] Price, R., MHEG: An Introduction to the Future International Standard for Hypermedia Object Interchange, Proceedings of ACM Multimedia93, 1993.

[14] Sample, M., Snacc 1.1: A High Performance ASN.1 to C/C++ Compiler, *snacc reference manual*, 1993.

[15] 高木利久, 遺伝子データベースの現状と研究開発動向, 情報処理学会アドバンスドデータベースシステムシンポジウム予稿集, 1992.