リレーショナルアプローチによるOSIディレクトリのDIB（ディレクトリ情報ベース）の実装と評価†

小花貞夫†† 西山智†† 鈴木健二††

多様化，高度化する通信ネットワーク環境で効率よく通信するためには，接続番号（アドレス）や通信能力など通信相手に関する情報をあらかじめ取得することが必要である．このような情報を提供するためには，各種のディレクトリが存在可能であるが，将来の利便性や相互接続性を考慮した場合，OSI（開放型システム間相互接続）ディレクトリが注目されている．このOSIディレクトリの実装では，ディレクトリ情報のデータベースであるDIBを，電子電話帳，FTAMやMHSなどのネームサーバとして適用可能なように汎用的に実現することが課題となっている．とりわけ，普及度の高い既存のリレーショナル型DBMS(RDB)を用いて効率的にDIBを作成することが重要な課題である．本論文では，既存のRDBが提供する機能を活用して，汎用的なDIBを効率的に実現するための手法を論じた．まず，DIBとRDBの対応付けでは，RDBへのアクセス回数を減らすテーブル構造の設計，ASN.1データに対する一意のデータ表現形式，ディレクトリスキーマ情報のプログラムからの分離が重要であることを示した．ついで，これらの対応付けに基づいたDIBソフトウェアの実装概要を報告した．最後に，実証実験等を通して，DIB機能の汎用性，処理効率，ソフトウェア規模，データ規模などを評価・考察し，機能や応答時間の面で実用レベルのDIBを，RDBを用いて効率的に開発できることを示した．

1. まえがき

多様化，高度化するネットワーク環境で効率よく通信するためには，接続番号（アドレス）や通信能力など通信相手に関する情報が必要となる．国際電信電話諮問委員会（CCITT）および国際標準化機構（ISO）では，通信相手に関する情報を提供し，通信利用者や各種通信システムを支援するOSIディレクトリの勧告／標準化[1]を行った．現在，OSIディレクトリの電信電話帳，FTAMやMHS用のネームサーバとしての適用が検討されており，システムの早急な実現と普及が望まれている．

OSIディレクトリの標準では，ディレクトリ情報の抽象的なモデルであるディレクトリ情報ベース（DIB）とそれにアクセスするためのプロトコルを規定している．DIBは，ディレクトリ情報のデータベースであり，電子電話帳，FTAMやMHSのネームサーバなどOSIディレクトリの適用形態により，扱う情報の種類や構造が異なる．このため，OSIディレクトリを実装する際には，DIBを，さまざまな適用形態に対応可能なように，汎用的に実現することが重要な課題である．

汎用的なDIBの実現例としては，これまでに，オブジェクト指向言語を用いてDIB専用のデータベースシステムを新たに作成したもの[2]や，OSの標準ファイルシステムが提供する階層的なファイル管理機能を活用したもの[3]などが報告されている．しかしながら，既存の汎用データベース管理システム（DBMS），とりわけ，普及度が高く，しかも完成度の高い既存のリレーショナル型DBMS（RDB）[4]が提供するデータの効率的な格納，管理，検索機能やコミットメント，排他制御等の機能を有効活用してDIBを効率的に構築する方法は，これまで論じられていない．

本論文では，汎用的なDIBを既存のRDBを用いて効率的に実現する手法を論じる．まず，DIBとRDBにおけるデータ構造，データ表現形式，ディレクトリスキーマおよび操作の対応付けについて論じる．ついで，それらの対応付けに基づいたDIBソフトウェアの実装概要を報告する．最後に，実証実験等を通して，DIB機能の汎用性，処理効率，ソフトウェア規模，データ規模などを評価・考察し，機能や応答時間の面で実用レベルのDIBを，RDBを用いて効率的に開発できることを示す．

2. OSIディレクトリのDIBの概要

以下に，CCITTとISOで勧告／標準化[1]されたOSIディレクトリの概要を述べる．

（1） OSIディレクトリのモデル

OSIディレクトリは，通信の対象となる“物”（オブジェクトと呼ぶ）に関する情報のデータベースと，

† A Relational Approach to an Implementation of DIB (Directory Information Base) for OSI Directory by SADAO OBANA, SATOSHI NISHIYAMA and KENJI SUZUKI (KDD R&D Laboratories).

†† 国際電信電話(株)研究所

ディレクトリ利用者エージェント (DUA) と呼ばれる応用プロセスからこれらの情報にアクセス (読出し，探索，変更など) する機能を規定している．OSIディレクトリが提供するデータベースをディレクトリ情報ベース (DIB) と呼ぶ．また，DIBは，ディレクトリシステムエージェント (DSA) と呼ばれる互いに協調動作する複数の応用プロセスに分散して格納・管理される．

(2) DIBの情報構造

DIBはディレクトリ情報木 (DIT) と呼ばれる木構造で表現される (図1)．木の節はエントリを，また木の枝はエントリ間の従属関係を表す．エントリは，基本的に，ひとつのオブジェクト (例えば，人，システム，応用プロセス，ファイル，組織，国など) を表し，それに関する属性情報 (例えば，人名，住所，電話番号，プレゼンテーションアドレス，O/Rアドレスなど) を持つ．すべてのエントリは識別名 (DN) により一意に識別される．エントリの識別名は，エントリの直接上位のエントリの識別名とそれに対する相対識別名 (RDN) の組により再帰的に表される．DITは，ディレクトリスキーマと呼ばれるDIBの構造上の制限の規定 (DIT構造，オブジェクトクラス，属性タイプおよび属性シンタックスの規定) に従って構成される．このディレクトリスキーマは，OSIディレクトリの適用形態に依存して定義される．また，複数のDSAが協調動作を行うために，各DSAは，DIBのどの部分がどのDSAに収容されているかなどに関する内部知識を持つ必要があり，知識木 (KT) と呼ばれる木構造で表現される．これらの知識には，自DSA内のエントリを指す内部リファレンス，直接下位のエントリが存在するDSAを指す下位リファレンスなど5つの知識がある．

(3) ディレクトリ操作

DITに対し，既知の識別名からエントリの属性情報を読み出すRead操作，特定オブジェクトの直接下位に位置するエントリの一覧をとるList操作，特定の条件に合致するエントリを検索し，その属性情報を得るSearch操作，エントリの属性情報を更新するModifyEntry操作など9種類のディレクトリ操作が規定されている．

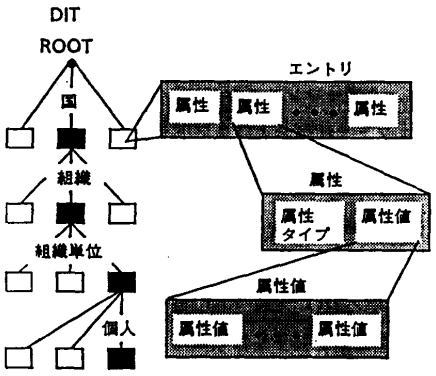

図1 ディレクトリ情報木
Fig. 1 Directory Information Tree (DIT).

3. 既存RDBを用いた汎用DIBの構築方法

OSIディレクトリのさまざまな適用形態で必要となる情報を扱える汎用的なDIBの実現手法としては，これまでに，オブジェクト指向言語を用いてDIB専用のデータベースシステムを新たに作成する手法が論じられている[2]．しかしながら，既存の汎用データベース管理システム (DBMS) が既に提供する機能を有効活用することにより，DIBを効率的に構築する方法は，これまで論じられていない．DBMSの中でも，とりわけリレーショナル型DBMS (RDB) は，汎用パッケージが広く普及しており，DBMSパッケージとしての完成度も高く，データの効率的な格納・管理機能，処理の高速化機能，コミットメントを実現するトランザクション機能，複数ユーザの同時サポート機能，排他制御機能ならびにデータ保守のための各種ユーティリティなどを完備したものが多い．このため，RDBパッケージが提供するこれらの機能を利用することにより，ディレクトリ情報の変更に関する一連の処理のコミットメントや障害時の回復制御，複数利用者からの同時検索や更新操作の効率的な排他制御などの複雑な処理をRDBにまかせることができ，DIB用に新たに専用データベースソフトウェアを作成する場合に比べて，ソフトウェアの開発を容易とすることが期待できる．

汎用的なDIBをRDBを用いて実装する場合，DIBとRDBにおける，1)データ構造の対応付け，2)データ表現形式の対応付け，3)ディレクトリスキーマの対応付けおよび4) 操作の対応付けが重要となる．以下にそれぞれの対応付けについて詳細に検討する．

3.1 データ構造の対応付け

DIBとRDBにおいては，データ構造上，以下に示す特徴的な相違がある．

- DIBにおけるDITおよびKTは木構造で表現され，しかも，ディレクトリスキーマで規定される制

約の範囲内において，木の深さには制限がない．これに対して RDB では，データはフラットなテーブル構造で表現される．

- DIB では，エントリの情報を構成する属性の値は，各属性に対して複数個持てる．これに対して RDB では，各行（タプル）の各属性（テーブルのカラムに対応）に，単一の値しか持てない．

DIB と RDB におけるこのようなデータ構造上の相違を解決するため，以下に示すデータ構造の対応付けを行うこととする．また，OSI ディレクトリでは，DIB に対する操作のほとんどが検索であり，更新頻度はきわめて少ないことを想定しているため，ここでは，更新時間よりも検索時間の短縮を優先したデータ構造とする．

図2に，DIB に対応する RDB のテーブル構造を示し，また，**表1**に DIB と RDB のテーブル要素との対応を示す．

DIT と一部の KT（内部リファレンスおよび下位

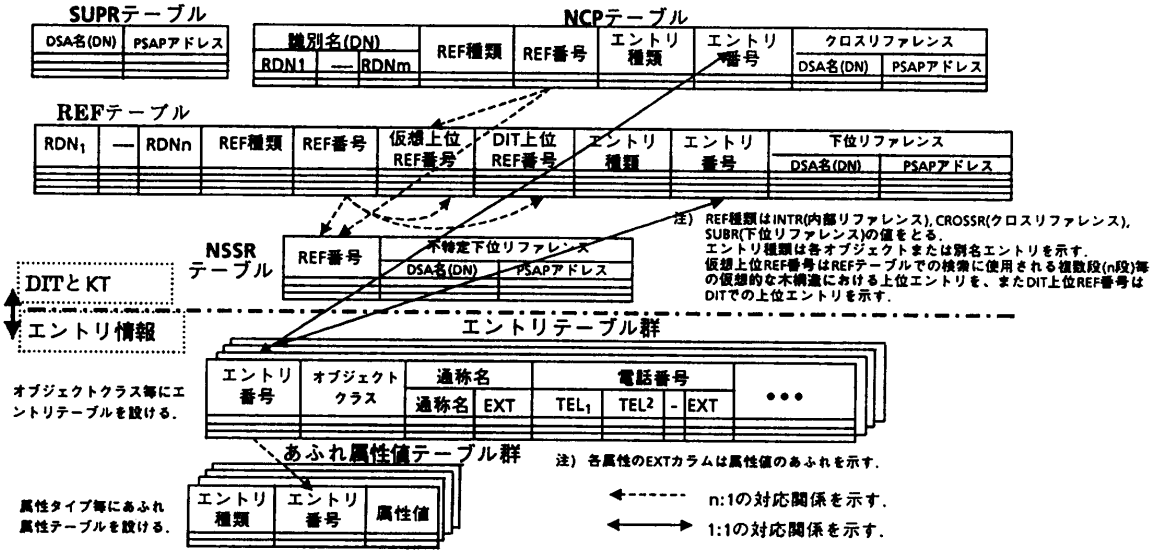

図 2 ディレクトリ情報のテーブルと対応関係
Fig. 2 Directory Information Tables and their relationships.

表 1 DIB と RDB におけるデータ構造の対応付け
Table 1 Mapping on data structure between DIB and RDB.

DIB の要素		RDB のテーブル要素
KT	クロス リファレンス	NCP テーブルのカラム（クロスリファレンス） 注）タプルは，ネーミングコンテクストごと
	上位リファレンス	SUPR テーブルのカラム（上位リファレンス） 注）タプルは，上位 DSA ごと
	不特定下位 リファレンス	NSSR テーブルのカラム（不特定下位リファレンス） 注）タプルは，リファレンスごと
	内部リファレンス	REF テーブルのカラム（エントリ種類およびエントリ番号）
	下位リファレンス	REF テーブルのカラム（下位リファレンス）
DIT	節 （エントリ）	REF テーブルのタプル
	枝 （エントリの従属関係）	REF テーブルのカラム （仮想上位 REF 番号および DIT 上位 REF 番号）
エントリ情報（属性）		オブジェクトクラスごとのエントリテーブル 注）タプルは，エントリに対応 属性タイプごとのあふれ属性値テーブル 注）タプルは，エントリ＋属性値に対応

リファレンス）は，REF (Reference) テーブルに格納する．REF テーブルの各行（タプル）は，DIT におけるエントリに対応する．REF テーブルは，データ操作言語（DML）を用いた一回の RDB アクセスで DIT の複数段（n 段）のエントリをまとめて検索可能とし，DML の発行回数を減らすため，相対識別名 (RDN) に対応する複数 (n) のカラムを設け，各 RDN はそれぞれのカラムに対応付ける．つまり，1段目から n 段目までのエントリに対応するタプルの RDN の各カラムには，それぞれの RDN を収容する．RDN の数が n に満たないエントリの場合には，その RDN のカラムは NULL とする．$n+1$ 段目から $2n$ 段目までのエントリに対応するタプルは，n 段目のエントリに対応するタプルのリファレンス番号 (REF 番号) を仮想上位 REF 番号として保持し，また，RDN のカラムには $n+1$ 段目以後の RDN を保持する．REF テーブルにおける RDN のカラム数 (n) は，OSI ディレクトリの適用形態で使用する RDN 数の平均値により決定する．これにより，通常一回の DML の発行により，目的のエントリに到達できる．また，複数 DSA 間の協調動作で必要となる DIT 部分木と DSA との対応関係（ネーミングコンテクスト）とクロスリファレンスは NCP (Naming Context Prefix) テーブルに，上位リファレンスおよび不特定下位リファレンスは，それぞれ，SUPR (Superior Reference) テーブルおよび NSSR (Non-Specific Subordinate Reference) テーブルに格納する．

エントリ情報は，オブジェクトクラスごとに異なる属性を持つためオブジェクトクラスごとに別のエントリテーブルを，また，あふれ属性値テーブルを各属性タイプごとに設ける．通常一回の DML でエントリに関する情報の検索を完了するように，複数の属性値が許される属性に対しては，エントリテーブルに複数カラムをあらかじめ設ける．このカラム数は，OSI ディレクトリの適用形態で必要となる属性値の数から決定する．属性値の数があらかじめエントリテーブルに確保したカラム数を超える場合，あふれ属性値テーブルに収容する．あふれ属性値テーブルの使用の有無は，拡張指示カラム (EXT) により示す．

3.2 データ表現形式の対応付け

DIB では，エントリにおける属性値のデータ型が ASN. 1[5] により規定されているが，ASN. 1 の符号化規則に従って符号化すると，符号化した値が必ずしも一意に定まらない．このため ASN. 1 で符号化されたデータをそのまま RDB のテーブルに格納すると，完全一致や部分一致などの比較演算を RDB の機能で実行させることができない．そこで，以下のように，RDB における属性値の表現形式が一意となるようにする．

（1） 属性値が ASN. 1 の基本型（文字列など）の場合

RDB の文字列型のデータとして格納する．ただし，大文字，小文字の区別を行わない CaseIgnore の文字列の場合は，表現の一意性を保つため，大文字あるいは小文字のどちらかに変換し格納する必要がある．

（2） 属性値が ASN. 1 の構造型 (SET など) の場合

属性に対し値の比較方法が定義される場合には，図 3 に示すフラットなデータ表現形式により正規化し，RDB の文字列型のデータとして格納する．特に相対識別名 (RDN) のように ASN. 1 の集合型 (SET) で規定されるものは，SET の構成要素に，属性タイプのオブジェクト ID 順などの順序規則を設ける工夫が必要である．比較方法が定義されない属性値の場合は，ASN. 1 で符号化したものをそのまま格納できる．

3.3 ディレクトリスキーマの対応付け

DIB に対する操作(ディレクトリ操作)を，3.1 節および 3.2 節でそれぞれ述べた RDB のテーブルおよびデータ表現形式に対する DML に変換するためには，DIB のデータ構造やデータ表現形式を規定するディレクトリスキーマと RDB で実際に使用されるデータ構造やデータ表現形式との対応付けに関する知識（ここでは，ディレクトリスキーマ情報と呼ぶ）が必要となる．

オブジェクトクラスや属性タイプの追加などによるディレクトリスキーマの変更に対して柔軟に対応できる汎用性の高い DIB を実現するため，ディレクトリ

識別名のDIBにおけるデータ表現形式(ASN.1)

```
DistinguishedName ::=          SEQUENCE OF RDN
RDN ::=     SET OF AttributeValueAssertion
AttributeValueAssertion ::=    SEQUENCE { AttrType,
                               AttrValue }
```

識別名のRDBにおけるデータ表現形式

```
(6:"JP",
 7:"SAITAMA",
 10:"KDD",
 (7:"KAMIFUKUOKA",11:"R & D LABS"),
 11:"OSI SYSTEM GROUP",
 3:"TARO KOKUSAI")
```

注)数字は属性タイプのオブジェクトIDを示す．

図 3 データ表現形式の対応例
Fig. 3 Example of mapping on data representation between DIB and RDB.

スキーマ情報をプログラムから分離し，図4に示すディレクトリスキーマ関連のテーブルとして管理することとする．ディレクトリスキーマ関連のテーブルには，DIT 構造定義テーブル，オブジェクトクラス定義テーブルおよび属性タイプ／シンタックス定義テーブルが含まれる．このうちDIT構造定義テーブルはDIT構造定義を，オブジェクトクラス定義テーブルは，オブジェクトクラス定義およびテーブル／カラムとの対応を，属性タイプ／シンタックス定義テーブルは，属性シンタックス定義とそのローカルな格納形式（長さを含む）などをそれぞれ管理する．

また，テーブル構造の管理は，RDB が提供するテーブルの定義／拡張機能を活用することにより行う．

3.4 操作の対応付け

ディレクトリ操作を実行するための名前解読および操作実行における操作の対応付けを以下に述べる．

（1） 名 前 解 読

名前解読では，まず，NCP テーブルを参照してネーミングコンテクストが自 DSA に存在するか否かを判定する．検索したタプルのリファレンス種類（REF 種類）がクロスリファレンス（CROSSR）の場合は，クロスリファレンスのカラムの値を用いて当該 DSA に操作を依頼する．また，NCP テーブルにネーミングコンテクストが存在しない場合には，SUPR テーブルの上位リファレンスを用いて当該 DSA に操作を依頼する．

上記でネーミングコンテクストが自 DSA に存在する場合，未検索の RDN をキーに REF テーブルを参照して操作対象エントリのタプルを検索する．この際，n 段（REF テーブルにおける RDN のカラム数）の RDN と n 段ごとの上位の REF 番号（仮想上位 REF 番号）をキーとして検索を行う．また，n 段以上 RDN がある場合には，n 段ごとに複数回にわけて検索を行う．検索したタプルの REF 種類が下位リファレンス（SUBR）の場合，下位リファレンスのカラムの値を用いて当該 DSA に操作を依頼する．また，REF テーブルに該当タプルが存在しない場合には，NSSR テーブルを参照し DSA 名および PSAP アドレスのカラムの値が示す不特定下位リファレンスを用いて当該 DSA に操作を依頼する．検索したタプルの REF 種類が内部リファレンス（INTR）の場合には，次に述べる操作実行を行う．

（2） 操 作 実 行

操作実行では，名前解読で得た REF テーブルのタプルにおけるエントリ種類をもとにエントリテーブルを決定し，エントリ番号をキーとして，各ディレクトリ操作ごとに異なる手順で値の読み出し・比較・更新等を行う．

- Read 操作，Compare 操作：対象タプルの値の読み出し（Read 操作の場合）や比較（Compare 操作の場合）を行う．属性にあふれ値がある場合，あふれ属性値テーブルの読み出しや比較も行う．
- List 操作：エントリテーブルは用いず，REF テーブルのみを用いて指定されたエントリの直接下位のエントリに対応するタプルの RDN カラムの値を得る．
- Search 操作：エントリテーブルのタプルを読み出

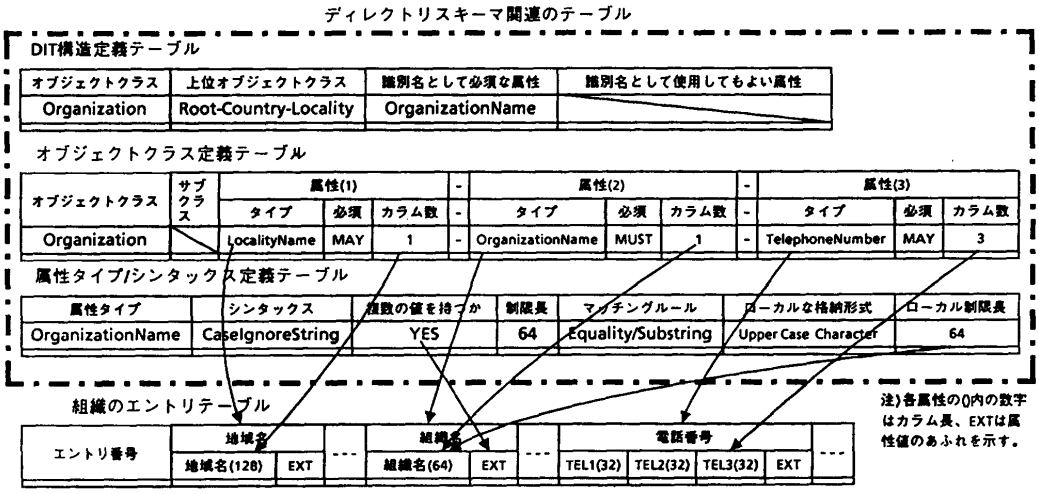

図 4 ディレクトリスキーマ関連のテーブルとエントリテーブルとの関係
Fig. 4 Relationship among Directory Schema related tables and Entry Tables.

し，フィルタで指定された探索条件を満たすか否かを判定する．属性にあふれ値がある場合は，あふれ属性値テーブルも読み出す．また，DIT部分木に含まれる複数エントリが探索範囲となる場合，REFテーブルを用いて下位エントリのタプルを検索し，上記操作を適用する．

● 更新操作 (AddEntry, RemoveEntry, ModifyEntry および ModifyRDN 操作)：エントリテーブルおよびあふれ属性値テーブルの更新（タプルの追加／削除，カラムの値変更など）を行う．ModifyEntry 操作では，属性値の削除（カラムの値削除）に伴い，あふれ属性値テーブルからエントリテーブルへの値の詰め替えも必要となる場合がある．また，他の3つの更新操作によりエントリの追加・削除・名前変更が行われた場合には，KTを構成するREFテーブルの内部リファレンスも更新する．

上記操作でREFテーブルを参照した結果，REF種類が下位リファレンスや不特定下位リファレンスであるタプルを検索した場合，他DSAに操作を依頼する．

4. DIBの実装概要

3章で述べた対応付けに基づいて，DIBを含むOSIディレクトリのソフトウェアをVAX (OS: VMS) 上に実装した[6),7)]．本ソフトウェアは，**図5**に示すDUA（ディレクトリ利用者エージェント）およびDSA（ディレクトリシステムエージェント）のソフトウェアからなる．DIBを実現するRDBには，RDBの標準的な機能を提供する代表的なパッケージのひとつであるORACLE[8)]を使用し，また，プログラムはC言語で記述した．実装したソフトウェアの機能を**表2**に示す．

DSAのソフトウェアは，さらにDSAプロセスとDIBプロセスから構成した．DSAプロセスは，DAP（ディレクトリアクセスプロトコル）およびDSP（ディレクトリシステムプロトコル）のプロトコル処理，ならびにROS（遠隔操作サービス），ACSE（アソシエーション制御サービス要素）以下のOSI通信プロトコル処理を行う．ACSE以下のプロトコル処理ソフトウェアは既に作成済みのもの[9)]を用いた．また，DSAプロセスは，複数アソシエーションの管理や他DSAとの協調動作（チェイン，マルチキャストなど）の制御を行う．

DIBプロセスは，名前解読および操作実行を行う．各種テーブルへのアクセスは，RDBのDML (ORACLEでは，SQL[10)]) 文を生成し，RDBに発行することで実現した．また，同時に複数のディレクトリ利用者からのディレクトリ操作要求を実行できるように，アソシエーションごとに独立のDIBプロセスを割当て，RDBに対してそれぞれログインし，RDBの複

表2 ソフトウェアの主な仕様
Table 2 Software specification.

オブジェクトクラス	国，地域，組織，組織部署，住人，デバイス，応用プロセス等標準で推奨された15種類（追加定義可）
属性タイプ	国名，地域名，通称名，電話番号，TLX番号，ISDNアドレス，郵便住所，郵便番号，肩書，PSAPアドレス等標準で推奨された36種類（追加定義可）
ディレクトリ操作	(読み出し) Read, Compare, Abandon (探　索) List, Search (更　新) AddEntry, RemoveEntry ModifyEntry, ModifyRDN
DSAの協調動作機能	チェイン，マルチキャスト リフェラル
プロトコル	DAP（ディレクトリアクセスプロトコル） DSP（ディレクトリシステムプロトコル） ROSプロトコル （プレゼンテーションマッピング）

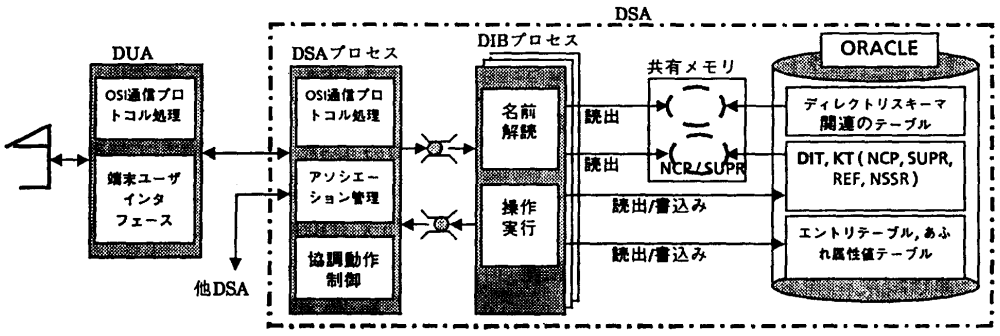

図5 OSIディレクトリのソフトウェア構成
Fig. 5 Software structure of OSI Directory.

数ユーザサポート機能を利用することとした．DIB プロセスはアソシエーション確立時に生成することとした．

今回の実装では，DIB プロセスの処理効率を向上させるため以下の工夫を行った．

- NCP, SUPR やディレクトリスキーマ関連のテーブルは，規模が小さくしかも変更が少ないため，システム起動時に RDB から主記憶（共有メモリ）にロードし，常駐させた．
- RDB が提供するインデックス機能を用い，属性値やリファレンス番号などカラム内容を条件検索するカラムに対しインデックスファイルを生成した．

また，ModifyEntry 等の更新操作で，ひとつの操作で複数のテーブルにまたがって更新を行う場合には，処理の一貫性（すべて更新するか，あるいは全く行わないか）を保証するために，RDB が提供するコミットメント機能を用いた．複数のユーザからの同一データに対する更新要求の競合などアクセスの排他制御についても，RDB が提供する機能を用いて実現した．複数 DSA にまたがる更新については，一貫性を保つためのメカニズムが未だ標準で明確でないため，今回，ひとつの DSA 内に閉じた更新操作のみ可能とした．

5. 評価および考察

5.1 DIB の機能について

（1） DIB の汎用性

RDB を用い3章で述べた4つの対応付けを行うことによる DIB の汎用性を，ディレクトリスキーマ変更に対する柔軟性の観点から考察する．

- DIT 構造，オブジェクトクラス，属性タイプの新規（または追加）定義に対する柔軟性：DIT 構造，オブジェクトクラスおよび属性タイプの定義に対しては，それぞれ，DIT 定義テーブル，オブジェクトクラス定義テーブル，属性タイプ／シンタックス定義テーブル内の値の追加とエントリ情報のテーブル，すなわちエントリテーブルおよびあふれ属性値テーブルの定義を行う．DIT 定義テーブルなどへの値の追加やエントリ情報のテーブル定義は，RDB が既に提供するデータ操作ユーティリティやテーブル定義機能を用いることで行える．また，REF テーブルなど DIT や KT に関するテーブルについては，変更の必要はない．
- 属性シンタックスの新規（または追加）定義に対する柔軟性：今回実装した OSI ディレクトリ標準で推奨された属性シンタックスを用いる場合には，属性タイプ／シンタックス定義テーブルの値変更のみでよい．それ以外の新たな属性シンタックスを用いる場合には，属性タイプ／シンタックス定義テーブルの値追加に加え，そのASN. 1 データとテーブルカラムにおけるデータ表現形式との変換を行うプログラムを若干追加する必要がある．

以上，OSI ディレクトリのさまざまな適用形態にあわせてディレクトリスキーマを変更する場合でも，プログラムに対する変更を極少化でき，汎用性の高い DIB を実現できたと考えられる．

（2） DIB 機能の制約

一般に RDB のパッケージでは，ひとつのテーブルに定義できる最大カラム数や各カラムに収容できる最大データ長に制限があるため，各エントリが持てる属性数や“記述”，“所有者”など1K バイトを超えるような長いデータ長を想定している一部の属性値のデータ長に制約が生じることとなる．しかし，電子電話帳への適用においては，そのような制約内で十分対応でき，実用上問題はない．

5.2 処理効率について

実装したソフトウェアの処理効率を評価するため，以下の実験を行った．ここでは，REF テーブルにおける RDN のカラム数を5とし，1回の REF テーブルへのアクセスで，DIT の5段のエントリを一括検索できるようにした．また，DIB の内容には，電子電話帳の基本的なオブジェクト（国，組織，組織単位，組織人）と属性（通称名，電話番号）を想定したデータを用いた．

（1） ネットワーク層の計算機内部折返しによる実験

VAX 8700 上に，トランスポート層以上のプロセスを1つの DUA と2つの DSA ごとにそれぞれ別に用意し，ネットワーク層を VAX 8700 内部で折り返した．DUA から DSA に以下の操作を発行した．

- 1つの DSA で処理を完了し，DSA の協調動作なし
- 2つの DSA が協調動作して処理を実行する

（2） X. 25 網経由による実験

μVAX (DUA 側) および VAX 8700 (DSA 側) 上に，トランスポート層以上のプロセスをそれぞれ用意し，X. 25 網 (9.6 Kbps) を経由して，DUA から DSA に対し協調動作なしの操作を発行した．

DUA が操作を発行してから応答を受信するまでの

時間を表3に示す．ひとつのDSAで処理が完了する場合，VAX 8700の内部折返し試験では，DIBのエントリ件数10,000件（組織人のエントリ件数），DUAとDSAの間でアソシエーションがあらかじめ確立されている状態で，Read操作が1.1秒（内訳は，DIBプロセス処理0.6秒，OSI通信処理0.5秒），Search操作（DIT 1レベルで5件のエントリが対象）が3.1秒（内訳は，DIBプロセス処理2.6秒，OSI通信処理0.5秒）という値を得た．内部折返し試験は，ほぼLANなどの高速通信環境を擬似した場合と考えられるが，使用した計算機（VAX 8700）が6 MIPSと特に処理能力の高い計算機ではないことや，同一計算機上にDUAと2つのDSAを疑似的に実現し，通常とは異なる負荷の環境下で評価したことを考慮すると，測定結果は，ソフトウェアが十分実用できることを示していると考えられる．またネットワーク層以下にX. 25網を使用した場合においても，パケット交換機等の伝送・交換遅延があるが，実用できることを確認できた．

一方，複数のDSAで協調動作を行う場合，DSA間でアソシエーションを確立する必要があるが，操作要求時にアソシエーションを確立すると，そのオーバヘッドが生じる（内部折返しで0.7秒，またX. 25網経由で4秒）．したがって頻繁にアクセスするDSA間では，常時アソシエーションを確立しておくことが肝要である．

また，Search操作では，DIT部分木のオブジェクトをひとつずつ探索することとなるため，処理時間が探索対象となるオブジェクトの数に大きく依存する．このためSearch操作を出す場合には，探索範囲を十分絞り込む必要がある．

さらに，内部折返し試験により，複数ユーザから同時に操作要求を行った場合の応答時間を測定した．ここでは，同時に4ユーザからの操作要求を実行させた．ReadやSearchなどの検索系の操作では，単一ユーザの場合に比べて1.2倍から1.5倍の応答時間となった．これは，ユーザの増加に伴って，対応するDUAプロセスやDIBプロセスの数が増加し，プロセス切り替えなどのオーバヘッドが生じたためと考えられる．

表3 応答時間測定結果
Table 3 Response time evaluation.

操作＼実験形態	内部折返し（単位：秒）注1)	X.25網経由（単位：秒）注1)，注2)
アソシエーション確立	0.7	4
協調動作なし		
Read 注3)	1.1(0.6) 注7)	2.6
Search 注4)	3.1(2.6) 注7)	4.5
Compare	1.3	
List 注5)	1.7	
AddEntry	2.5	—
RemoveEntry	1.9	
ModifyEntry 注6)	3.9	
ModifyRDN	2.3	
協調動作あり		
Read 注3)	3.0 注8)	
Search 注4)	4.8 注8)	—

注1) プレゼンテーション・カーネル，セションBCS（全二重），トランスポート・クラス0，TPDU長512バイト．
注2) X.25，9.6Kbps，パケット長128バイト．
注3) 1属性抽出．
注4) 1レベル，5件ヒット，1属性抽出．
注5) 5件ヒット．
注6) 1属性追加．
注7) 括弧内はDIBプロセスの処理時間を示す．
注8) DSA間のアソシエーション確立時間含む．
注9) エントリ件数約10,000件．

5.3 データの規模について

本アプローチによるデータ規模について評価するため，電子電話帳のオブジェクトと属性を用いてファイル容量を測定した．10,000件のオブジェクトエントリがある場合で，約4.3Mバイトのデータ量となった．これには，オブジェクトエントリを構成する属性情報の実データ（約2.0Mバイト）に加えて，RDBとして蓄積するためのオーバヘッド（約2.0Mバイト）および検索処理を効率化するインデックスファイルを用いたことによるオーバヘッド（約0.3Mバイト）等が含まれる．実データ量と比較して，本アプローチによるファイル容量のオーバヘッドはそれほど大きくないと考えられる．

5.4 ソフトウェアの規模について

DSAのソフトウェア規模は，DIBプロセスがC言語で21Kステップ，DSAプロセス（既にFTAM等で作成済みのACSEおよびプレゼンテーション層以下のOSI通信プロトコル処理の部分を含まず）が19.6Kステップ，また，ASN. 1の符号化と復号で上記プロセスから共通に使用される部分が34Kステップであった．これらのソフトウェアは，表2に示したすべての機能を実現しており，特に，標準で推奨された36種類の属性タイプすべてを扱え，また，属性シンタックスの追加を伴わない属性タイプの追加をプ

ログラム変更なしに行えること，ならびに DSA 間の協調動作を可能としていることなどを考慮すると，新たに DIB 専用のデータベースシステムを構築する場合[2]に比べて，小規模なソフトウェアで実現できたと言える．

5.5 今後の課題

（1） OSI ディレクトリ標準の問題点

相対識別名 (RDN) で用いられる通称名 (Common Name) などの属性では，T. 61 文字列が使用可能で，漢字など各国語文字を使用できるが，実装する文字集合など T. 61 の実装基準が明確でない．このため，国際間の相互接続に支障をきたすこととなり，T. 61 の実装基準の明確化が必要である．

また，標準では，オブジェクトへの識別名の割当て方法を特に規定していない．このため，地理的区分と DIT 部分木が必ずしも一致せず，電子電話帳など利用者がイエローページ的検索を行う場合，目的のオブジェクトエントリに到達するまでに，試行錯誤的な操作を強いられる可能性がある．そこで，地理的区分に従って名前付けする規則を設けるなど OSI ディレクトリの運用方法の制約が必要である．

（2） 同時アクセス時における効率的な処理方式

4 ユーザからの同時アクセスの応答時間を測定し，今回のソフトウェア構成でも実用レベルの応答性を達成できることを確認した．さらに多くのユーザからの同時アクセスに対するシステムの振る舞いについては，プロセス構成やプロセス間通信方式などといったソフトウェアの実装方法の視点からも検討する必要がある．

6. むすび

本論文では，一般に広く普及している既存のリレーショナル型 DBMS (RDB) を用いることにより OSI ディレクトリのディレクトリ情報ベース (DIB) を効率的に実現する手法を論じた．具体的には，DIB と RDB における 4 種類の対応付けを示し，ここでは，RDB へのアクセス回数を減らせるテーブル構造の設計，ASN. 1 データに対する一意のデータ表現形式，ディレクトリスキーマ情報のプログラムからの分離が重要であることを示した．また，これらの対応付けに基づいて実装した OSI ディレクトリ用 DIB のソフトウェアを用いて，DIB 機能の汎用性，処理効率，ソフトウェア規模，データ規模などの観点より評価・考察し，機能や応答時間の面で実用レベルの DIB を，既存の RDB を用いることにより効率的に開発できることを示した．

本論文で示した手法は，DIB に類似したデータ構造や操作を持つ，ネットワーク管理 (OSI 管理) 用の管理情報ベース (MIB) の構築にも有効であり，その応用範囲は広いと考えられる．

謝辞 日頃ご指導いただく国際電信電話(株)研究所小野所長，浦野次長に感謝します．

参考文献

1) CCITT 勧告 X. 500, X. 501, X. 509, X. 511, X. 518, X. 519, X. 520, X. 521, The Directory (1988).
ISO/IEC 9594-1～8: Information Processing Systems—Open Systems Interconnection—The Directory (1988).
2) 中川路，勝山，宮内，玉田，水野：OSI ディレクトリシステムにおける DIB（ディレクトリ情報ベース）のオブジェクト指向による実現，情報処理学会論文誌，Vol. 32, No. 3, pp. 304-313 (1991).
3) Robbins, C. J., Kille, S. E. and Turland, A.: The ISO Development Environment: User's Manual, Vol. 5: QUIPU (1989).
4) Codd, E. F.: A Relational Model of Data for Large Shared Data Bank, *Comm. ACM*, Vol. 13, No. 6, pp. 377-387 (1970).
5) ISO 8824, 8825: Information Processing Systems—Open Systems Interconnection—Specification of Abstract Syntax Notation One (ASN. 1) (1987)./Specification of Basic Encoding Rules for Abstract Syntax Notation One (ASN. 1) (1987).
6) 小花，西山：OSI ディレクトリ・システムの実装と評価，情報処理学会マルチメディア通信と分散処理研究会資料，42-11, pp. 81-88 (1989).
7) 西山，小花：リレーショナル型データベースを用いた OSI ディレクトリ情報ベース (DIB) の実現と評価，情報処理学会マルチメディア通信と分散処理研究会資料，42-12, pp. 89-96 (1989).
8) ORACLE RDBMS, Release Notes, ORACLE Part No. 30001-V5. 1 (1986).
9) 小花，加藤，鈴木：OSI プレゼンテーション，ACSE, FTAM プロトコルの実装と評価，情報処理学会論文誌，Vol. 30, No. 7, pp. 741-753 (1989).
10) ISO 9075: Information Processing Systems—Database Language SQL (1989).

(平成 2 年 11 月 19 日受付)
(平成 3 年 9 月 12 日採録)

小花 貞夫（正会員）

昭和28年生．昭和51年慶応義塾大学工学部電気工学科卒業．昭和53年同大学院修士課程修了．同年国際電信電話(株)入社．現在，同社研究所OSI通信グループ主任研究員．この間，パケット交換方式，ネットワークアーキテクチャ，OSIプロトコルの実装，データベース，国際ビデオテックス通信，分散処理技術の研究に従事．電子情報通信学会会員．

西山 智（正会員）

昭和36年生．昭和59年東京大学工学部電気工学科卒業．同年国際電信電話(株)入社．平成2年から3年まで米国テキサス大学オースチン校に留学．現在，同社研究所OSI通信グループに勤務．この間，データベース，ネットワークアーキテクチャ，国際ビデオテックス通信，分散処理技術およびエキスパートシステムの研究に従事．

鈴木 健二（正会員）

昭和20年生．昭和44年早稲田大学理工学部電気通信学科卒業．昭和44年から45年までオランダのフィリップス国際工科大学に招待留学．昭和51年早稲田大学大学院博士課程修了．工学博士．同年国際電信電話(株)入社．現在，同社研究所OSI通信グループリーダ．この間，磁気記録，パケット交換方式，ネットワークアーキテクチャ，通信処理の研究に従事．昭和62年度前島賞受賞．電子情報通信学会，IEEE各会員．