TTC 標準
TTC STANDARD

JT-X500
ディレクトリの基本アーキテクチャ

〔Overall Architecture of the Directory〕

第2版

1994年4月27日制定

社団法人
情報通信技術委員会

THE TELECOMMUNICATION TECHNOLOGY COMMITTEE

本書は、（社）情報通信技術委員会が著作権を保有しています。
内容の一部又は全部を（社）情報通信技術委員会の許諾を得ることなく複製、転載、改変、転用及びネットワーク上での送信、配布を行うことを禁止します。

＜参考＞

1．国際勧告等との関連

本標準は、１９８８年版X．５００シリーズ勧告に対する９３年版拡張として、１９９３年のＩＴＵ－Ｔ　ＳＧ７会合において勧告されたX．５００、X．５０１、X．５０９、X．５１１、X．５１８、X．５２０、X．５２１、X．５２５、及びX．４０２のＭＨＳにおけるディレクトリ利用とＭＨＳにおける名前付けに準拠したもので、主にX．５００、X．５０１及びX．４０２のディレクトリの利用に関する部分を規定したものである。

ただし、本標準は、上記勧告を部分的に記述したものであり、本標準に記述していない部分については、上記勧告を参照する必要がある。

2．上記国際勧告等に対する追加項目等

2.1　オプション選択項目

なし

2.2　ナショナルマター決定項目

なし

2.3　先行している項目

なし

2.4　追加した項目

日本語名の実現方法を付属資料Ｆとして追加した。

2.5　削除した項目

なし

2.6　その他

なし

表－A　参照勧告との対応（９２年版）

本標準	ＩＴＵ－Ｔ勧告	備考
1．概説	X.500(0 章) / X.501(0 章)	
2．適用範囲と分野	X.500(1 章) / X.501(1 章)	
3．参照勧告及び標準	X.500(2 章) / X.501(2 章)	
4．用語の定義	X.500(3 章) / X.501(3 章)	
5．略語	X.500(4 章) / X.501(4 章)	
6．概要	X.500(6 章)	
7．ディレクトリ情報ベースの概要	X.500(7 章)	
8．ディレクトリサービス	X.500(8 章)	
9．分散ディレクトリ	X.500(9 章)	
10．ディレクトリのアクセス制御	X.500(10章)	
11．ディレクトリの複製	X.500(11章)	
12．ディレクトリプロトコル	X.500(12章)	
13．ディレクトリモデル	X.501(6 章)	
14．ディレクトリ情報ベース	X.501(7 章)	
15．ディレクトリエントリ	X.501(8 章)	
16．名前	X.501(9 章)	
＜ディレクトリ管理権限モデル＞	X.501(10章)	
＜管理と運用情報のモデル＞	X.501(11章)	
17．ディレクトリスキーマ	X.501(12章)	
＜ディレクトリシステムスキーマ＞	X.501(13章)	
＜ディレクトリスキーマ管理＞	X.501(14章)	
18．セキュリティモデル	X.501(15章)	
＜基本アクセス制御＞	X.501(16章)	
＜ＤＳＡモデル＞	X.501(17章)	
＜知識＞	X.501(18章)	
＜ＤＳＡ情報モデルの基本要素＞	X.501(19章)	
＜ＤＳＡ情報の表現＞	X.501(20章)	
＜ＤＳＡ運用枠組みの概要＞	X.501(21章)	
＜運用結合＞	X.501(22章)	
＜運用結合の仕様と管理＞	X.501(23章)	
＜運用結合管理の操作＞	X.501(24章)	
19．ＭＨＳでの名前付け	X402(17 章)	
20．ＭＨＳでのディレクトリの利用	X402(20-24章)	
付Ａ：オブジェクト識別子の利用	X.501(AxA)	
付Ｂ：情報枠組みのASN.1 表記	X.501(AxB)	
付Ｃ：サブスキーマ管理のASN.1 表記	X.501(AxC)	
＜付：基本アクセス制御のASN.1 ＞	X.501(AxD)	
付Ｄ：ＤＳＡ運用属性型のASN.1 表記	X.501(AxE)	
＜付：運用結合管理操作のASN.1 表記＞	X.501(AxF)	
付Ｅ：日本語名の実現方法	－－－－－	ＴＴＣ独自規定
付Ｆ：ＭＨＳｱﾌﾟﾘｹｰｼｮﾝのｵﾌﾞｼﾞｪｸﾄ識別子定義	X402(Ax B)	
付１：ディレクトリの応用	X.500(AxA)	
付２：ツリーの構造	X.501(AxG)	
付３：名前付け基準	X.501(AxH)	
＜付：スキーマの様々な側面の例＞	X.501(AxJ)	
＜付：基本アクセス制御許可の概要＞	X.501(AxK)	
＜付：基本アクセス制御の例＞	X.501(AxL)	
＜付：ＤＳＥタイプの組み合わせ＞	X.501(AxM)	
＜付：知識のモデル化＞	X.501(AxN)	
付４：用語対照表	X.501(AxP)	

（注１）＜＞の部分については、本標準に記述していないため、対応する勧告の対応部分を参照する必要がある。

（注２）　Ax : Annex

4．改版の履歴

版　数	発　行　日	改　　版　　内　　容
第1版	昭和６３年１１月３０日	制　定
第2版	平成　６年　４月２７日	ディレクトリの９３年拡張を反映したディレクトリ勧告X．５００シリーズの改訂に伴う改版

5．工業所有権

本標準に関わる「工業所有権の実施の権利に係る確認書」の提出状況は、ＴＴＣホームページでご覧になれます。

目　　次

1．概　説

本標準はディレクトリの概念、モデル及び提供するサービスと機能を記述している。

(1) 本標準の６章から１２章ではディレクトリとＤＩＢの概念の紹介とモデル化を行い、提供するサービスと機能を概説する。このモデルは、ディレクトリが提供する抽象サービスの定義を行う場合、また、この抽象サービスを実現し、伝達するためのプロトコルの記述を行う場合に使用される。

(2) 本標準の１３章から１８章では、ディレクトリを異なる観点から見たモデルを示す。モデルには、全体（機能）モデル、管理機関モデル、一般的ディレクトリ情報モデル、一般的ＤＳＡ及びＤＳＡ情報モデル、運用上の枠組み及びセキュリティモデルがある。一般的ディレクトリ情報モデルは、ディレクトリが保持する情報の構造化法、たとえば、オブジェクトに関する情報をディレクトリのエントリとしてまとめる方法や、これらの情報から、オブジェクトの名前を得る方法を記述する。一般的ＤＳＡ及びＤＳＡ情報モデルと運用上の枠組みは、ディレクトリの分散をサポートする。

(3) 本標準の１９章と２０章では、ＭＨＳでの名前付け法とディレクトリの利用法を示す。

(4) 付属資料Ａでは、ディレクトリで使用するＡＳＮ．１オブジェクト識別子について要約する。

(5) 付属資料Ｂでは、情報枠組みに関する全ての定義を含むＡＳＮ．１モジュールを示す。

(6) 付属資料Ｃでは、サブスキーマ管理に関する全ての定義を含むＡＳＮ．１モジュールを示す。

(7) 付属資料Ｄでは、ＤＳＡ運用属性型に関する全ての定義を含むＡＳＮ．１モジュールを示す。

(8) 付属資料Ｅでは、運用結合管理操作に関する全ての定義を含むＡＳＮ．１モジュールを示す。

(9) 付属資料Ｆでは、日本語名の実現方法を示す。

(10) 付属資料Ｇでは、ＭＨＳアプリケーションのオブジェクト識別子定義モジュールについて記述する。

(11) 付録１では、ディレクトリの応用を示す。

(12) 付録２では、ツリー構造について要約する。

⑬ 付録3では、名前の設計の上で考慮すべきいくつかの基準を記述する。

⑭ 付録4では、用語対照表を記述する。

2．適用範囲と分野

2．1　ディレクトリの機能

ディレクトリはＯＳＩアプリケーション、ＯＳＩ管理、他のＯＳＩ層エンティティおよび通信サービスから要求されるディレクトリ機能を提供する。その機能の中で‘ユーザにとって親しみやすい名前付け’と呼ばれる機能は、人である利用者から引用に適した名前によってオブジェクトが参照されることを可能とする（しかしながら全てのオブジェクトがユーザに親しみやすい名前を必要とするわけではない）。「名前とアドレスのマッピング」と呼ばれる機能は、オブジェクトとその場所の動的な結合を可能とする。後者の機能により、例えばオブジェクトの追加、削除、場所の変更がＯＳＩネットワークの動作に影響を与えないという意味で、ＯＳＩネットワークは‘自己構築性’をもつことが可能となる。

2．2　ディレクトリと汎用データベースの関係

ディレクトリは汎用データベースシステムを目指すものではないが、そのようなシステム上に構築してもかまわない。例えば、通信用ディレクトリが主にそうであるように、更新よりもかなり高い頻度の‘問い合わせ’が発生すると考えられる。この場合、情報の更新間隔は、例えば、網側の要求よりも、むしろ、人や組織により依存すると予想される。また更新を全てのシステムに渡って一斉に実施する必要性はなく、同一の情報に関して利用可能である新旧の２つの世代の情報が存在するという過渡的な状態は容認できる。

2．3　ディレクトリの特徴

アクセス権の異なった場合や伝達することが出来ない情報の更新を行うとした場合を除き、ディレクトリへの問い合わせの結果が問い合わせ元の場所や身分によらないことはディレクトリの特徴である。この特徴のためディレクトリはある種の通信アプリケーション、例えばある種のルーティング機能には不適切である。

2．4　ディレクトリのモデル

本標準で定義するモデルは、ディレクトリの様々な局面を定義しているディレクトリ関連の他の標準に対して、概念的及び用語面での枠組みを示す。

機能モデルと管理モデルは、機能と管理の観点から、ディレクトリが分散する方法を定

義する。一般的ＤＳＡモデルとＤＳＡ情報モデル及び運用上の枠組みも、ディレクトリの分散のためのものである。

一般的ディレクトリ情報モデルは、ディレクトリユーザと管理ユーザの見地から、ＤＩＢの論理構造を記述する。これらのモデルにおいては、ディレクトリが集中型ではなく、分散型であるということを意識しなくてもよい。

一般的ディレクトリ情報モデルの限定化は、ディレクトリスキーマ管理を支援する。

(1) ディレクトリが提供するサービスは、情報枠組みの概念を使用して記述される。これにより、提供されるサービスは、ＤＩＢの物理的な分散に、独立したものになる。

(2) ディレクトリの分散操作は、上記のサービスを提供するように規定される。従って、言い替えれば、ＤＩＢが実際には複雑に分散していても、上記の論理構造を維持するように規定される。

(3) ディレクトリ全体のパフォーマンスを改善するための、ディレクトリの構成部分による複製能力については、ＩＴＵ－Ｔ勧告Ｘ．５２５にて定義されている。

セキュリティモデルは、アクセス制御メカニズムの観点から定義する。ＤＩＴの特定の部分にて機能するアクセス制御機構を定義する。それは、広範囲のアプリケーション及び使用方法に対して都合の良い、２つのフレキシブルで特徴的なものである。セキュリティモデルはディレクトリ情報へのアクセス制御にのみ関係し、情報を持つＤＳＡアプリケーションの実体へのアクセス制御には関係しない。

ＤＳＡモデルは、ディレクトリ構成要素の運用の仕様の枠組みを確立する。特に、

(1) ディレクトリ情報モデルは、それぞれがＤＳＡである１つあるいはそれ以上の構成要素として示される。

(2) ディレクトリ分散モデルは、ＤＩＢエントリやシャドウコピーが、ＤＳＡの間で分散して存在できる原理を示す。

(3) ＤＳＡ情報モデルは、ＤＳＡにて保存されるディレクトリ利用者情報および運用情報の構造を示す。

(4) ＤＳＡ間での協力形態の定義の手法を示す。

3．参照勧告及び標準

本標準で参照するＩＴＵ－Ｔ勧告及び標準を以下に示す。

Ｘ．５００（１９９３），ＩＳＯ／ＩＥＣ９５９４－１（１９９３）
　開放型システム間相互接続－ディレクトリ－概念、モデル及びサービスの概要
Ｘ．５０１（１９９３），ＩＳＯ／ＩＥＣ９５９４－２（１９９３）
　開放型システム間相互接続－ディレクトリ－モデル
Ｘ．５１１（１９９３），ＩＳＯ／ＩＥＣ９５９４－３（１９９３）
　開放型システム間相互接続－ディレクトリ－抽象サービスの定義
Ｘ．５１８（１９９３），ＩＳＯ／ＩＥＣ９５９４－４（１９９３）
　開放型システム間相互接続－ディレクトリ－分散処理
Ｘ．５１９（１９９３），ＩＳＯ／ＩＥＣ９５９４－５（１９９３）
　開放型システム間相互接続－ディレクトリ－プロトコル仕様
Ｘ．５２０（１９９３），ＩＳＯ／ＩＥＣ９５９４－６（１９９３）
　開放型システム間相互接続－ディレクトリ－代表的な属性型
Ｘ．５２１（１９９３），ＩＳＯ／ＩＥＣ９５９４－７（１９９３）
　開放型システム間相互接続－ディレクトリ－代表的なオブジェクトクラス
Ｘ．５０９（１９９３），ＩＳＯ／ＩＥＣ９５９４－８（１９９３）
　開放型システム間相互接続－ディレクトリ－認証の枠組み
Ｘ．５２５（１９９３），ＩＳＯ／ＩＥＣ９５９４－９（１９９３）
　開放型システム間相互接続－ディレクトリ－複製
Ｘ．２００（１９８８）
　開放型システム間相互接続－基本参照モデル
ＩＳＯ７４９８（１９８４）
　情報処理システム－開放型システム間相互接続－基本参照モデル
ＩＳＯ　７４９８－２
　情報処理システム－開放型システム間相互接続
　－基本参照モデル－セキュリティアーキテクチャ
Ｘ．２１９（１９９３），ＩＳＯ／ＩＥＣ９０７２－１（１９９３）
　遠隔操作－概念、モデル及び表記法
Ｘ．６８０（１９９３），ＩＳＯ／ＩＥＣ８８２４－１（１９９３）
　開放型システム間相互接続－抽象構文記法１(ASN.1)－基本記法の仕様
Ｘ．６８１（１９９３），ＩＳＯ／ＩＥＣ８８２４－２（１９９３）
　開放型システム間相互接続－抽象構文記法１(ASN.1)－情報オブジェクト仕様
Ｘ．６８２（１９９３），ＩＳＯ／ＩＥＣ８８２４－３（１９９３）
　開放型システム間相互接続－抽象構文記法１(ASN.1)－制約仕様
Ｘ．６８３（１９９３），ＩＳＯ／ＩＥＣ８８２４－４（１９９３）
　開放型システム間相互接続－抽象構文記法１(ASN.1)－ASN.1仕様のﾊﾟﾗﾒｰﾀ定義
ＩＳＯ／ＩＥＣ１０１８１－１（１９９３）
　開放型システム間相互接続－開放型システムおけるセキュリティの枠組み
　－Part1：概要
ＩＳＯ／ＩＥＣ１０１８１－２（１９９３）
　開放型システム間相互接続－開放型システムおけるセキュリティの枠組み
　－Part2：認証
ＩＳＯ／ＩＥＣ１０１８１－３（１９９３）
　開放型システム間相互接続－開放型システムおけるセキュリティの枠組み
　－Part3：アクセス制御

4．用語の定義

4．1　ＯＳＩ参照モデルの定義

この標準はX．２００で得られた概念に基づいており、その中で定義されている次の用語を使用する。

(1) 応用エンティティ（application-entity）

(2) 応用層（application layer）

(3) 応用プロセス（application-process）

(4) 応用プロトコルデータ単位（application protocol data unit）

(5) 応用サービス要素（application service element）

(6) アクセス制御（access control）

(7) 認証（authentication）

(8) セキュリティ方針（security pi）

4．2　ディレクトリモデルの定義

(1) アクセスポイント(Access Point)：

抽象サービスを獲得する点。

(2) アドミニストレーションディレクトリ管理領域（Administration Directory Management Domain, ADDMD）：

アドミニストレーションにより管理されるＤＭＤ。

（注）ここでは、アドミニストレーションは、公衆電気通信の主官庁、または、公衆電気通信サービスを提供する他の組織体を意味する。

(3) 管理機関（administrative authority）：

一つのディレクトリシステムエージェント内に保持される全エントリを制御する権利をもつエンティティ。

(4) ディレクトリ（the Directory）：

オブジェクトに関する全情報の格納場所、および、その情報へのアクセスであるディレクトリサービスをユーザに提供する主体。

(5) ディレクトリ管理と運用情報（Directory administration and operational）：

管理及び運用を目的としてディレクトリに使用される情報。

(6) ディレクトリ管理領域（Directory Management Domain）：

単一の組織により管理されている、ＤＳＡとＤＵＡの集合体。ＤＳＡのみで構成されることもある。

(7) ディレクトリシステムエージェント（Directory System Agent）：

ディレクトリの一部をなすＯＳＩ応用プロセス。

(8) ディレクトリユーザ（Directory user）：

ディレクトリのエンドユーザ。すなわち、ディレクトリにアクセスする人、または、エンティティ。

(9) ディレクトリユーザエージェント（Directory User Agent）：

ユーザがディレクトリにアクセスする際に、ユーザを代行するＯＳＩ応用プロセス。

(10) ディレクトリ利用者情報（Directory user information）：

利用者及びアプリケーションに有用な情報。

(11) ＤＩＴ領域（DIT Domain）：

（複数のＤＳＡで構成する）ＤＭＤにより保持されるグローバルなＤＩＴの一部。

(12) 領域管理組織（Domain Management Organization）：

ＤＭＤ（及びＤＩＴ領域に属しているもの）を管理する組織。

(13) 私設ディレクトリ管理領域（Private Directory Management Domain）：

アドミニストレーション以外の組織により管理されるＤＭＤ。

4．3　ディレクトリ情報ベースの定義

(1) 別名エントリ（alias entry）：

あるオブジェクトに別の名前を与えるために使用される情報を含む「別名」クラススのエントリ

(2) ディレクトリ情報ベース（Directory Information Base, DIB）：

ディレクトリ内のアクセスされる情報の集合で、ディレクトリの操作により、読出し、もしくは操作されるすべての情報を含む。

(3) ディレクトリ情報ツリー（Directory Information Tree, DIT）：

ツリー状のＤＩＢで、（ルートを除く）その節点がディレクトリのエントリとなる。

（注）ＤＩＴと言う用語は情報のツリー構造が問題となる状況でのみＤＩＢの代わりに使用される。

(4) ディレクトリエントリ、エントリ（Directory entry, entry）：

オブジェクトに関する情報を含むＤＩＢの一部分

(5) 直接上位（immediately superior）：

特定エントリもしくはオブジェクトに関連し（問題となっているコンテキストが明確である必要がある）、直接上位のエントリまたはオブジェクト。

(6) 直接上位エントリ（immediately superior entry）：

特定エントリに対し、その特定エントリが終了節点にあたるようなＤＩＴの枝の開始節点であるエントリ

(7) 直接上位オブジェクト（immediately superior object）：

特定のオブジェクトに対し、そのオブジェクトのエントリの直接上位になるエントリに対応するオブジェクト

(8)（対象となる）オブジェクト（object（of interest））：

ある「世界」のすべてのもの、一般に電気通信および情報処理の世界もしくはその一部の世界のもので、識別可能であり（名前で指定でき）、ＤＩＢに情報が保持されている。

(9) オブジェクトクラス（object class）：

一定の特性を共有するオブジェクト（もしくは想像されるオブジェクト）の群。

(10) オブジェクトエントリ（object entry）：

一つのオブジェクトに関するＤＩＢ内の主要な情報の集合のエントリであり、したがって、ＤＩＢ中のそのオブジェクトを代表するといえる。

(11) サブクラス（subclass）：

スーパクラスと関連。スーパクラスから導かれるオブジェクトクラス。そのサブクラスのメンバは、他のオブジェクトクラス（スーパクラス）のすべての特性を共有し、かつ、そのオブジェクトクラス（スーパクラス）のどのメンバも持たない付加特性を共有する。

(12) スーパクラス（superclass）：

サブクラスに関連。サブクラスが導かれるオブジェクトクラス。

(13) 上位（superior）：

（エントリまたはオブジェクトに適用し）、直接上位、もしくは、直接上位のその上位（再帰的）。

⑭ 下位（subordinate/inferior）：

上位の反対語。

⑮ サブツリー（subtree）：

ツリーの頂点に位置したエントリの集合。プレフィックスの「サブ」は、サブツリーのベース（またはルート）となる頂点は、常にＤＩＴのルートの下位であることを強調している。

⑯ サブツリー精緻化（subtree refinement）：

ＤＩＴの中において、明示的に特定されたエントリの部分集合。これは、サブツリーである必要はない。（つまり、エントリが位置付けられた頂点はツリーに接続されている必要はない。）

⑰ 直接スーパクラス（direct superclass）：

サブクラスに関連。サブクラスが直接的に導きだされるオブジェクトクラス。

4．4　ディレクトリエントリの定義

⑴ 属性（attribute）：

オブジェクトに関するある種の情報で、ＤＩＢの中でそのオブジェクトを記述するエントリの中に現われる。

⑵ 属性型（attribute type）：

属性の構成要素であり、その属性に与えられた情報クラスを示す。

⑶ 属性値（attribute value）：

属性型によって指示される情報クラスの実現値である。

⑷ 属性値提示（Attribute Value Assertion）：

エントリの値（主に識別値について）に関する提示であり、真、偽、あるいは不定である。

（注）属性値提示の例を記述するとき、“文字列１＝文字列２”という記法を使用している。この記法では“文字列１”は属性型の名前の略語であり、“文字列２”は適切な値をそのまま表現したものである。

例での属性型はＴＴＣ標準ＪＴ－Ｘ５２０で定義されている実際の型（例えば“Ｃ”は国名の略、“ＣＮ”は一般名の略）をもとにしているが、ディレクトリは通常使用されている属性型の意味を知らないことが多い。

(5) 識別値（distinguished value）：

エントリの属性値であり、エントリの相対識別名の中に現われる。

(6) 利用者属性(user attribute)：

利用者情報を表す属性。

(7) 属性階層(attribute hierarchy)：

より一般的な利用者属性型から詳細化された利用者属性型を導きだすことを可能にするような属性の側面。２つの属性型の定義（この定義はこれらの属性型に該当する属性にある種の振る舞いを要求する）は、この事により階層的になる。

(8) 属性サブタイプ(attribute subtype)：

属性型Ａが属性型Ｂより導きだされるか（この場合、ＡはＢの直接サブタイプとなる）、または属性型Ａが属性型Ｂのサブタイプである属性より導きだされた場合（この場合、ＡはＢの非直接サブタイプとなる）、属性型Ａは、属性型Ｂに関係あるといえる。

(9) 属性スーパタイプ(attribute supertype)：

属性型Ａが属性型Ｂより導きだされるか（この場合、ＢはＡの直接スーパタイプとなる）、または属性型Ａが属性型Ｂのサブタイプである属性より導きだされた場合（この場合、ＢはＡの非直接スーパタイプとなる）、属性型Ｂは、属性型Ａに関係あるといえる。

(10) 補助オブジェクトクラス(auxiliary object class)：

エントリまたは、エントリのクラスの記述的な内容であり、ＤＩＴの構造的な仕様に用いられないオブジェクトクラス。

(11) 集合属性(collective attribute)：

エントリ集合の各メンバに関してその利用者属性の値が全て等しい属性の集合。

(12) 直接属性参照(direct attribute reference)：

属性型の識別子を用いた一つまたはそれ以上の属性値の参照（ディレクトリまたはＤＳＡ抽象サービスの一つ）

(13) エントリ集合(entry collection)：

明示的に規定されたＤＩＴのサブツリーまたはサブツリー精緻化に属するエントリの集合。

(14) 間接属性参照(indirect attribute reference)：

属性型のスーパタイプの識別子を用いた一つまたはそれ以上の属性値の参照（ディレクトリまたはＤＳＡ抽象サービスの一つ）

⑮ 照合規則(matching rule)：

ディレクトリスキーマの一部となる規則。この規則により、そのエントリの属性値に関するある特定の記述（照合規則提示と言う）をする事により、エントリが選択できるようになる。

⑯ 照合規則提示(matching rule assertion)：

照合規則により定義された条件に一致した属性値のエントリが存在するか否かに関する命題。これには、真と偽と未定義がある。

⑰ 運用属性(operational attribute)：

運用及び／または管理情報を表す属性。

⑱ 構造オブジェクトクラス(structural object class)：

ＤＩＴの構造的仕様に用いられるオブジェクトクラス

⑲ エントリの構造オブジェクトクラス(Structural object class of entry)：

ある特定のエントリに関しては、単一の構造オブジェクトクラスがエントリに適用可能なＤＩＴ内規則とＤＩＴ構造規則を決定するために使用される。このオブジェクトクラスは、structuralObjectClass 運用属性により表示される。このオブジェクトクラスは、エントリの構造オブジェクトクラススーパクラス連鎖において最下位のオブジェクトクラスである。

4．5　名前の定義

(1) 別名（alias, alias name)：

ディレクトリ情報ツリー（ＤＩＴ）における１個以上の別名エントリにより定義されるオブジェクトに対する名前。

(2) 展開（dereferencing)：

別名を対応するオブジェクトの識別名に置換すること。

(3) 識別名（distinguished name)：

オブジェクトを表現する名前の中の１つを指し、オブジェクトエントリと上位のエントリの相対識別名から構成される名前。

(4) ディレクトリ名、名前（Directory name, name)：

すべてのオブジェクトから特定のオブジェクトを識別できるもの。この名前は曖昧ではない（１つのオブジェクトを指定する）が必ずしもユニーク（１つの名前）とは限らない。

(5) 提示名（purported name）：

構造は名前の形をしているが、必ずしも有効ではないもの。

(6) 命名機関（naming authority）：

名前の割当に責任を持つ機関。

(7) 相対識別名（Relative Distinguished Name, RDN）：

特定のエントリの識別値に関係する真の属性値提示の集合。

(8) 日本語名（Japanese name）：

Ｔ．６１文字列の日本語を使用する名前を示す。

(9) エントリ名（(entry) name）：

すべてのエントリから特定のエントリを識別できるもの。

4．6　ディレクトリスキーマの定義

(1) 属性構文(Attribute Syntax)：

属性の値を表現するために用いられるＡＳＮ．１のデータ型。

(2) ディレクトリスキーマ(Direcotry Schema)：

ＤＩＢを特徴づける規則及び制限（DIT構造規則、DIT内容規則、オブジェクトクラス、属性型，属性構文，照合規則）の集合。ディレクトリスキーマは、管理権のある管理領域（あるいは，サブスキーマの特別な部分）のエントリを管理する重なりのないサブスキーマの集合として、明確化される。ディレクトリスキーマは、ディレクトリ利用者情報のみに関係している。

(3) （ディレクトリ）サブスキーマ((Direcotry)Subschema)：

管理権のある管理領域（あるいは、サブスキーマの特別な部分）内にあるＤＩＢエントリを特徴づける規則及び制限（DIT構造規則、 DIT内容規則、オブジェクトクラス、属性型、属性構文、照合規則）の集合。

(4) ＤＩＴ内容規則(DIT Content Rule)：

特定の構造オブジェクトクラスあるいは別名オブジェクトクラスのエントリの内容を管理する規則。

(5) ＤＩＴ構造規則(DIT Structure Rule)：

許可される上位及び下位のエントリの関係を明確にすることによって、DITの構造を管理する規則。構造規則は、名前形成に関係し、それ故、上位の構造規則に対して、構造オブジェクトクラスあるいは別名オブジェクトクラスに関係する。これにより、名前形成によって識別される構造オブジェクトクラスあるいは別名オブジェクトクラスのエントリは、指示される上位の構造規則によって管理されるエントリの下位として、DIT内に存在することができる。

(6) （エントリの）管理構造規則(Governing Structure Rule (of an entry))：

特定のエントリに関して、そのエントリに適用される単独のDIT構造規則。

(7) 名前形成(Name Form)：

名前形成は、特定の構造オブジェクトクラスのエントリのために、許可されるRDNを明確にする。名前形成は、名前付けされたオブジェクトクラスと名前付けするために（そのRDNのために）用いれるべき１つあるいは複数の属性型を特定する。名前形成は、DIT構造規則で用いれる定義の基本的な一部である。

注）名前形成は登録されて、グローバルな範囲で有効である。それに対して、DIT構造規則は登録されていなくて、構造規則が関連する管理領域の範囲で有効である。

(8) 上位構造規則(Superior Structure Rule)：

特定のエントリに関して、そのエントリの上位を管理するDIT構造規則。

４．７　複製（レプリケーション：Replication ）の定義

複製に関する以下の用語については、ＩＴＵ－Ｔ勧告Ｘ．５２５における定義に基づき使用する。

(a) キャッシュコピー (cache copy)

(b) コンシューマリファレンス (consumer reference)

(c) エントリコピー (entry-copy)

(d) マスタＤＳＡ (master DSA)

(e) プライマリシャドウイング (primary shadowing)

(f) 複数エリア (replicated area)

(g) 複製 (replication)

(h) セカンダリシャドウイング (secondary shadowing)

(i) シャドウ消費者 (shadow consumer)

(j) シャドウ供給者 (shadow supplier)

(k) シャドウＤＳＡ特別エントリ (shadowed DSA-specific entry)

(l) シャドウイング (shadowing)

(m) サプライヤリファレンス (supplier reference)

5．略語

本標準で使用する用語について、その略語と英名を以下に示す。

ＡＣＤＦ	アクセス制御決定機能 (Access Control Decision Function)
ＡＣＩ	アクセス制御情報 (Access Control information)
ＡＣＩＡ	アクセス制御内部地域 (Access Control Inner Area)
ＡＣＳＡ	アクセス制御特定地域 (Access Control Specific Area)
ＡＤＤＭＤ	アドミニストレーションディレクトリ管理領域 (Administration Directory Management)
ＡＳＮ．１	抽象構文記法１ (Abstract Syntax Notation 1)
ＡＶＡ	属性値提示 (Attribute Value Assertion Domain)
ＢＥＲ	基本符号化規則 (Basic Encording Rule)
ＤＡＣＤ	ディレクトリアクセス制御領域 (Directory Access Control Domain)
ＤＡＰ	ディレクトリアクセスプロトコル (Directory Access Protocol)
ＤＩＢ	ディレクトリ情報ベース (Directory Information Base)
ＤＩＳＰ	ディレクトリ情報シャドウイングプロトコル (Directory Information Shadowing Protocol)
ＤＩＴ	ディレクトリ情報ツリー (Directory Information Tree)
ＤＭＤ	ディレクトリ管理領域 (Directory Management Domain)
ＤＭＯ	領域管理組織 (Domain Management Organization)
ＤＯＰ	ディレクトリ運用結合プロトコル (Directory Operation Binding Protocol)
ＤＳＡ	ディレクトリシステムエージェント (Directory System Agent)

ＤＳＥ	ＤＳＡ特別エントリ（DSA Specific Entry）
ＤＳＰ	ディレクトリシステムプロトコル（Directory System Protocol）
ＤＵＡ	ディレクトリユーザエージェント（Directory User Agent）
ＨＯＢ	階級組織運用結合（Hierarchical Operation Binding）
ＮＨＯＢ	不特定階級組織運用結合（Non-Specific HOB）
ＮＳＳＲ	不特定従属参照（Non Specific Subordinate Reference）
ＯＳＩ	開放型システム間相互接続（Open Systems Interconnection）
ＰＲＤＭＤ	私設ディレクトリ管理領域（Private Directory Management Domain）
ＰＳＡＰ	プレゼンテーションサービスアクセス点（Presentation Service Access Point）
ＲＤＮ	相対識別名（Relative Distinguished Name）
ＲＨＯＢ	関連階級組織運用結合（Relevant HOB）
ＳＤＳＥ	シャドウＤＳＥ（shadowed DSE）

６．概　要

６．１　ディレクトリの概要

「ディレクトリ」は、実存するオブジェクト群に関する情報の論理的なデータベースを保持するために協同動作する開放型システムの集合である。ディレクトリの「ユーザ」には人やコンピュータプログラムが含まれるが、これらのユーザは、許可されている場合に限り、情報全体もしくはその一部の読出しと更新を行なうことができる。ディレクトリユーザはディレクトリユーザエージェント（ＤＵＡ）と呼ばれる応用プロセスによってディレクトリにアクセスする。これらの概念を図６－１／ＪＴ－Ｘ５００に示す。

（注）本標準はディレクトリを単一のものとして示し、また多数のシステムとその要求を満たしている多数のアプリケーションによって構成されるひとつの論理的なディレクトリを単一で統合化された名前空間に創造することを意図している。これらのシステムはアプリケーションの必要性により相互動作を行うか否かを選択する。オブジェクトが相互に無関係なアプリケーションではこのような必要性が生じない場合もある。単一の名前空間は将来、その必要性が変更された場合に相互動作を容易にする。

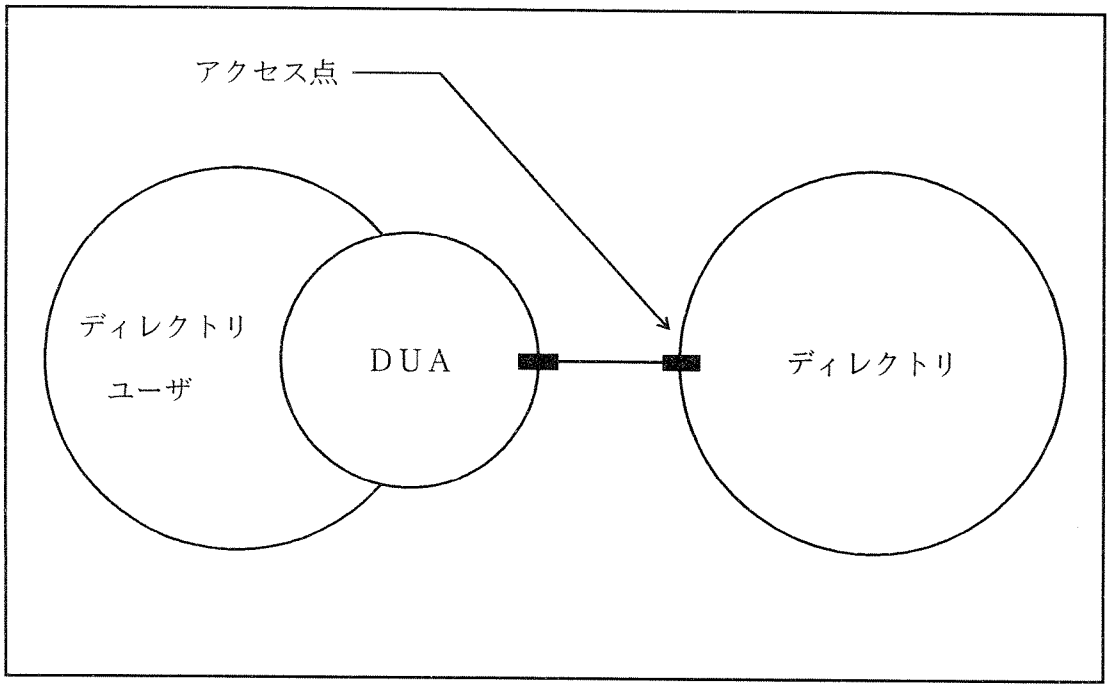


図６－１／ＪＴ－Ｘ５００　ディレクトリへのアクセス
(ITU-T X.500)

６．２　ディレクトリ情報ベース（ＤＩＢ）

ディレクトリが保持する情報を全体としてディレクトリ情報ベース（ＤＩＢ）と呼ぶ。本標準の７章にその構造を概説する。

６．３　ディレクトリの抽象サービス

ディレクトリは、ディレクトリの抽象サービスと呼ばれる適切に定義された一連のアクセス機能をユーザに提供する。このサービスは、本標準の８章に概説されるが、簡潔な情報の検索と更新の機能を提供する。この機能はローカルＤＵＡ機能とともにエンドユーザによって要求される機能を提供するためにつくられる。

６．４　分散ディレクトリのモデル

ディレクトリは広範囲に分散され、機能的、管理上でも分散されると予想される。９章にこのディレクトリに対応するモデルを示す。そこではある統合化された集合体を構築するために必要な多様な構成要素が、互いに協同動作するための枠組みを示している。

6．5　アクセス制御

様々な管理機関による、各機関で管理している情報に対するアクセスへの制御が、ディレクトリでは、行われる。アクセス制御については、１０章で概説する。

6．6　複製

ディレクトリが分散している時、性能および利用を高めるため、情報を複製することが望ましい。複製については、１１章で概説する。

6．7　応用プロセス間の協同動作

ディレクトリサービスを提供し利用するには、ディレクトリのユーザ（実際にはＤＵＡ）と機能的構成要素間の多様な協同動作を必要とする。多くの場合、これは様々な開放型システムの応用プロセス間の協同動作を必要とし、１２章に概説される協同動作を管理するための標準化された応用プロトコルが必要となる。

6．8　ディレクトリのアプリケーション

ディレクトリは多種多様なアプリケーションに適用できるように設計されている。適用されたアプリケーションは、どの様な種類のオブジェクトがディレクトリに含まれているか、どの様な種類のユーザが情報にアクセスするか、どの様な種類のアクセスを行なうかを管理する。適用されるアプリケーションは、例えば電子メールの配布リストの規定のように固有なものでもよいし、‘個人間通信ディレクトリ’のように汎用的なものであってもよい。ディレクトリは次のようなアプリケーション間の共通部分を提供する。

・一つのオブジェクトは、一つまたはそれ以上のアプリケーションに関連し得る：
　おそらく同一のオブジェクトに関する同じ情報が利用されるだろう。
　これをサポートするため、一連のアプリケーションに有用な多数のオブジェクトクラスと属性型を定義する。これらはＴＴＣ標準ＪＴ－Ｘ５２０で規定している。

・ディレクトリのある利用パターンはアプリケーションに関係なく共通であろう：
　これについては付録１で概説している。

7．ディレクトリ情報ベース（ＤＩＢ）の概要

（注）ＤＩＢはその構造を含め本標準の１４章で定義する。

7．1　ＤＩＢの構成

ＤＩＢはオブジェクトに関する情報によって構成されている。それらは（ディレクトリの）「エントリ」からなり、その各々のエントリはあるオブジェクトに関する情報の集まりである。各々のエントリは「属性」によりつくられており、各々の属性が１つの型と１つまたはそれ以上の値を持つ。個々のエントリ中に存在する属性の型はエントリが表わすオブジェクトの「クラス」に依存する。

7．2　ディレクトリ情報ツリー（ＤＩＴ）

ＤＩＢのエントリはツリーの形状に配置される。これがディレクトリ情報ツリー（ＤＩＴ）であり、その節点はエントリを表わす。ツリーの高い位置（ルートに近い位置）にあるエントリは国名や組織名を表す場合が多く、低い位置にあるエントリは人や応用プロセスを表す。

（注）ディレクトリ関連の標準で定義したサービスはツリー構造のＤＩＴにおいてのみ動作するが、（必要性によって生じる）他の構造を否定しない。

7．3　エントリの構成

各々のエントリは「識別名」を持ち、それによりエントリが明確かつ一義的に識別される。この識別名の特性は情報のツリー構造に由来している。あるエントリの識別名はその上位エントリの識別名とそのエントリから特に選択された属性値（識別値）とで構成される。

7．4　エントリの種類

ツリーのリーフにあたるエントリの一部は別名エントリであるが、他の全てのエントリはオブジェクトエントリである。別名エントリはオブジェクトエントリを指し、対応するオブジェクトの別の名前のための基盤となる。本標準では別名を利用して、日本語の指定を可能としている。

7．5　ディレクトリ・スキーマ

ディレクトリは、更新を行う場合、ＤＩＢを適切な形に保持するために一連の規則を用いる。これらの規則を「ディレクトリスキーマ」と呼び、エントリがそのオブジェクトクラスには誤った属性の型を持つこと、属性値がその属性型には誤った形式を持つこと、そしてエントリが誤ったクラスの下位エントリを持つことを防ぐ。

7．6　ＤＩＴの構成

図７－１／ＪＴ－Ｘ５００に上記のＤＩＴの概念とその構成を示す。

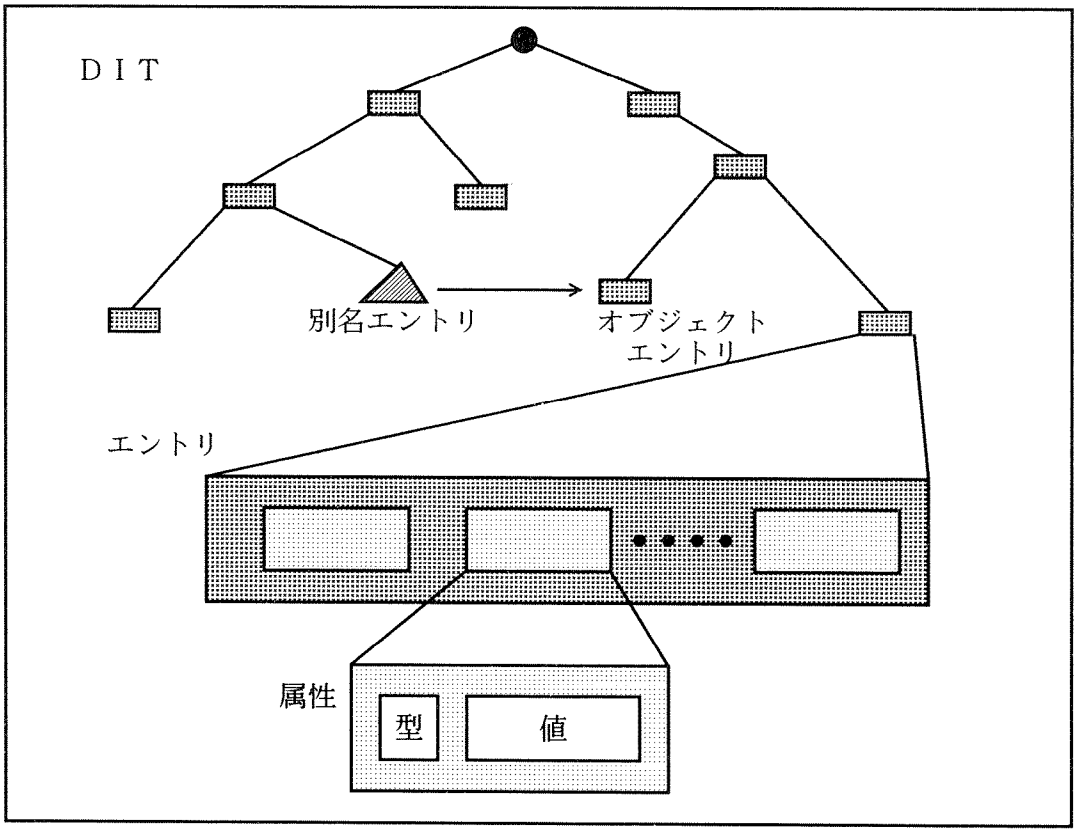


図７－１／ＪＴ－Ｘ５００　ＤＩＴとエントリの構造
(ITU-T X.500)

7．7　ＤＩＴの例

図７－２／ＪＴ－Ｘ５００に仮想的なＤＩＴの例を示す。ツリーには異なるオブジェクトを識別するためのいくつかの属性の型の例を示している。例えば名前：

{ C=JP, L=Tokyo, O=TTC, CN=Telex Terminal }

は、ある名前付けの属性として、地域名（=Locality)という地理属性をもつ応用エントリ

'Telex Terminal'を示す。同じ地理属性には次の名前：

{ C=JP, L=Tokyo, CN=Taro Suzuki }

も存在するが、これは居住者の Taro Suzuki を示す。

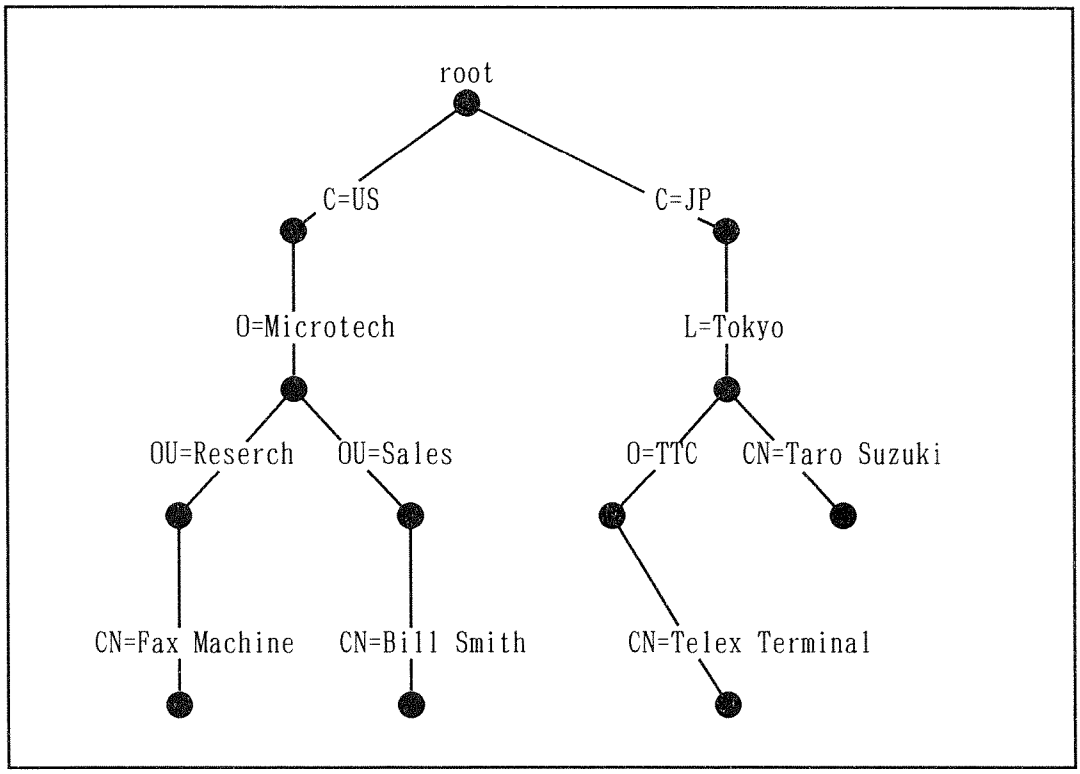


図７－２／ＪＴ－Ｘ５００　仮想的なディレクトリ情報ツリー
(ITU-T X.500)

（注）ここでは“文字列１＝文字列２”という記法を使用している。
“文字列１”は属性型の名前の略語であり（“Ｃ”は国名［country ］の略、“ＣＮ”は一般名［Common Name ］の略）、“文字列２”は属性値をそのまま表現したものである。

７．８　ディレクトリの管理機関

ＤＩＴの拡張と形状、ディレクトリスキーマの定義、そしてエントリ追加時の識別名の選択はいろいろな階層の管理機関の責任であり、それらの管理機関の相互関係がツリーの形に反映される。例えば、管理機関は責任範囲にある全てのエントリが重複しない識別名を持っていることを、それらの名前を表す属性型と値を注意深く管理することにより確認しなくてはならない。この責任は上位から下位の管理機関に委譲され、制御はスキーマによって行われている。

8．ディレクトリサービス

8．1　概　説

8.1.1　サービスの内容

この章では、ＤＵＡに代表されるような、ユーザに対してディレクトリが提供するサービスの概要を示している。ディレクトリが提供するサービスの全てはＤＵＡからの要求に応えるためのものである。要求には、8.3に示すようなディレクトリへの問い合わせ、8.4に示すような変更の要求がある。また、サービスに対する要求は、8.2に示すように制限される。ディレクトリは必ず各要求の結果を報告する。通常の結果の形式は要求に固有のものであり、要求の記述形式から明らかになる。ほとんどの異常結果はいくつかの要求に共通しており、8.5にその例を示す。

8.1.2　ローカル機能

現在のディレクトリ関連の標準では、全てのディレクトリの機能を規定していない。規定されていない機能については、標準化されるまでローカルな機能として提供されることが必要である。例えば、これらの機能には次の4項目がある。

- 任意のエントリの追加及び削除により、分散したディレクトリを作成する。
- アクセス制御の管理（つまり、特定のユーザが、特定の情報に対し、特定のアクセスをすることの許可または、許可を取り消すこと）
- ディレクトリスキーマの管理
- 知識情報の管理

8.1.3　ディレクトリの変更

ディレクトリはＤＩＢの変更を保証する。ディレクトリサービスの要求の結果であっても、他のある種の（ローカルな）手段であっても、結果としてＤＩＢはディレクトリスキーマのルールに従い続ける。

8.1.4　ユーザとの結合

ユーザとディレクトリは、ディレクトリへのアクセス点において結合付けられる。結合付ける時にユーザとディレクトリは、お互い識別することもできる。

8．2　サービスの制御機能

8.2.1　サービス制御

様々なサービス要求に対して適用できる多くの制御がある。これは主にディレクトリが越えてはならない資源の使用をユーザが定めることができるようにするためである。例えば時間、結果のサイズ、検索の範囲、相互動作モード、要求の優先順位に関する制御がある。

8.2.2　セキュリティパラメータ

各要求はディレクトリ情報を保護するためのセキュリティ機構を持つ情報を含んでいる。このような情報はユーザからの様々な種類の保護要求、例えば要求のディジタル署名、正しい当事者が署名を確認するのを助けるための情報を含んでいてもよい。

8.2.3　フィルタ

多数のエントリからの情報、または多数のエントリに関係している情報を含む結果を持つ要求にはフィルタを持たせるべきである。フィルタは１つのエントリが結果の一部として戻されるために満足しなければならない一つ以上の条件を表す。これは条件に合うエントリのセットを返させるものである。

8．3　ディレクトリ問い合わせ

8.3.1　読出し（Read）

読出し要求は特定のエントリを目的とし、そのエントリの属性のいくつかまたは全ての値を返させる。いくつかの属性のみが返されるとき、ＤＵＡは関係ある属性型のリストを提供する。

8.3.2　比較（Compare）

比較要求は特定のエントリの特定の属性を目的としている。ディレクトリは、与えられた値とエントリの属性の値が一致するか否かを検査する。

（注）例えばパスワードチェックであり、この場合、ディレクトリが持っているパスワードを読み出すことはできないが比較はできる。

8.3.3　リスト（List）

リスト要求によりディレクトリは、ＤＩＴの特定の名前を持つエントリの直接下位のものの一覧を返す。

8.3.4　検索（Search）

検索要求によりディレクトリは、いくつかのフィルタを満足するＤＩＴの、ある一部分の全エントリからの情報を返す。読出しと共に用いることにより各エントリから返された情報はそのエントリのいくつかまたは全ての属性から成る。

8.3.5　放棄（Abandon）

未完了の問い合わせ要求に対して適用される放棄要求は、ディレクトリに対し、要求発信者が、もはや、その実行されている要求に興味を持っていないという事を知らせる。ディレクトリは、例えば要求処理をやめたり、それまでに成し遂げられた結果を捨てたりする。

8．4　ディレクトリ変更

8.4.1　エントリ追加（Add Entry）

エントリ追加要求はＤＩＴに新しい末端エントリ（オブジェクトエントリまたは別名エントリ）を追加する。

8.4.2　エントリ削除（Remove Entry）

エントリ削除要求はＤＩＴから末端エントリを削除する。

（注）エントリ追加と共に、このサービスは現在リーフエントリだけに適用されるオペレーションであるが、将来一般的なケースにも強化されるであろう。

8.4.3　エントリ変更（Modify Entry）

エントリの変更要求では、ディレクトリは特定のエントリに対する一連の変更を実行する。全変更でも、そうでなくても、ＤＩＢはいつもスキーマと矛盾しない状態におかれている。変更は属性または属性値の追加、削除または置換えが可能である。

8.4.4　識別名の変更（Modify Distinguished Name）

識別名（ＤＮ）変更要求はＤＩＴのエントリ（オブジェクトエントリ又は別名エントリ）の相対識別名を、違う識別属性値の指定によって変更する。

あるいは、エントリをＤＩＴ内の新たな上位エントリのもとに移動する。エントリに下位エントリがある場合には、すべての下位エントリはこれに伴い名前を変更するか移動する。

8．5　他の結果

8.5.1　エラー

どのサービスも、例えばユーザが与えたパラメータに問題があることによって失敗することがあり、その場合エラーが報告される。問題を修正するための助けとなるように可能であれば、エラーとともに情報が戻される。ただし、一般的にはディレクトリで生じた最初のエラーだけが返される。さらに先に示したユーザが与えたパラメータの問題（特にエントリに対する無効名、または無効属性型）の他に、セキュリティ方針、スキーマ規則、サービス制御の違反によりエラーが生じる。

8.5.2　紹　介

サービスはＤＵＡが接している特定のアクセス点が要求実行にそぐわない場合（例えば要求に関係する情報がアクセス点から（論理的に）遠いことなどにより）失敗することがある。この場合、ディレクトリは、ＤＵＡが要求を出すことのできる代わりのアクセス点を提案する紹介を戻すことがある。

（注）ディレクトリとＤＵＡはそれぞれ、紹介を使用するか、要求を連鎖させるか（9.3.3(2)参照）を選択することができる。ＤＵＡはサービス制御を用いてその選択を表現する。ディレクトリはどちらのアプローチを使用するかを最終決定する。

9．分散ディレクトリ

9．1　機能モデル

ディレクトリの機能モデルを図９－１／ＪＴ－Ｘ５００に示す。

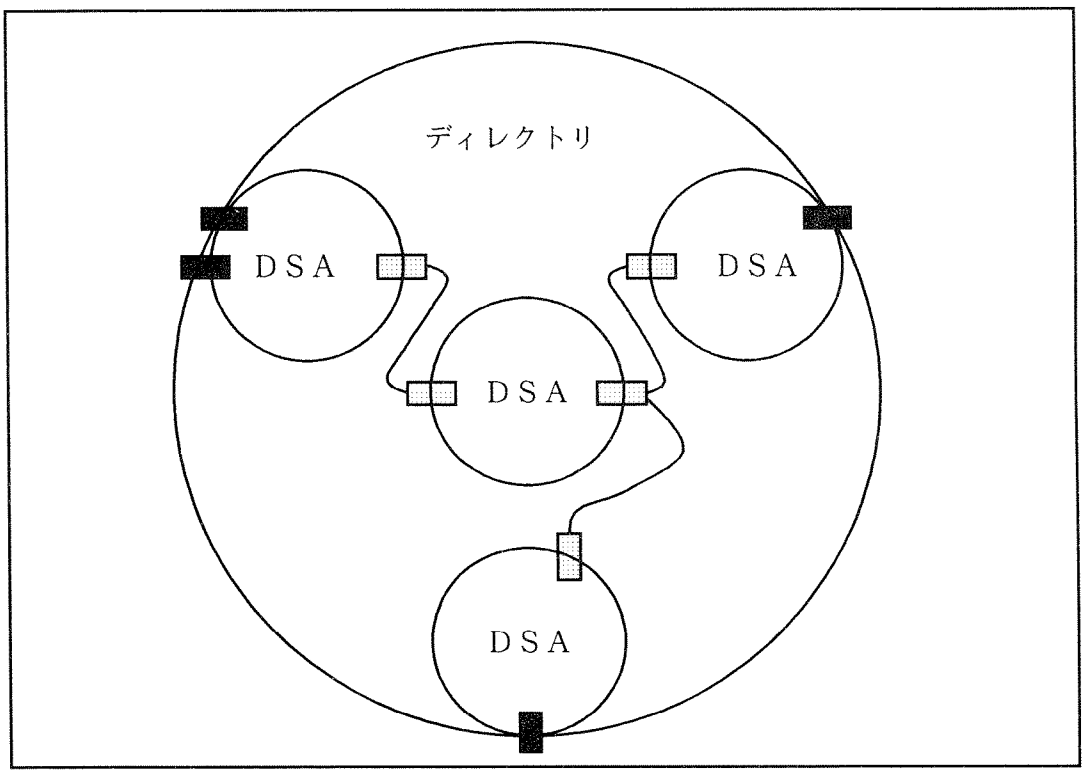


図９－１／ＪＴ－Ｘ５００　ディレクトリの機能モデル
(ITU-T X.500)

ディレクトリシステムエージェント（ＤＳＡ）は、ディレクトリの一部でありＯＳＩ応用プロセスである。その役割は、ＤＵＡ又は他のＤＳＡへのＤＩＢのアクセスを提供する。ＤＳＡは、そのＤＳＡ自身が持つデータベース（ローカルデータベース）に蓄積された情報を利用する事が可能であり、又、他のＤＳＡと共に要求を処理することもできる。更に、ＤＳＡは要求の処理を助ける他のＤＳＡにユーザを導くこともできる。ローカルデータベースの規定は実装時の事項である。

9．2　組織モデル

9.2.1　ディレクトリ管理領域（DMD）

一つの組織で管理される一つ以上のＤＳＡと０以上のＤＵＡの集合は、ＤＭＤ（ディレクトリ管理領域）を形成できる。この関連する組織は、ＤＭＤ内の機能要素間の通信を管理するために、ディレクトリ関連の標準を使わなくても良い。

9.2.2　ＤＳＡの動作

9.3において、ＤＳＡの動作についてのある観点について規定している。このことにより、あるDMD内のＤＳＡの集まりは、DMDを管理する組織のオプションとして、単一のＤＳＡとして動作しうる。

9.2.3　DMDの種類

DMDは、公共的通信機関によって運用されるか否かによって、ADDMD（アドミニストレーションディレクトリ管理領域）にもＰＲDMD（私設ディレクトリ管理領域）にもなり得る。

9．3　モデルの動作

9.3.1　ＤＵＡの動作

ＤＵＡは、一つ以上のＤＳＡと通信することで相互動作する。ＤＵＡは特定のＤＳＡと結びつく必要はない。ＤＵＡは要求を出す事でいろいろなＤＳＡと直接に相互動作が可能である。さまざまな管理上の理由から、ＤＵＡは要求の実行（例えば、あるディレクトリ情報を返信する）を必要とするＤＳＡと、常に直接に相互動作可能であるとは限らない。同様に、ＤＵＡは、単一のＤＳＡを通じてディレクトリにアクセスすることも可能である。このような理由により、ＤＳＡは互いに相互動作する必要がある。

9.3.2　ＤＳＡの動作

ＤＳＡは、ＤＵＡからの要求の実行や、必要な情報を持たない場合には、その情報の取得に関係している。ＤＳＡは、ＤＵＡに代わって他のＤＳＡと相互動作しその情報を取得する責任をもつ。

9.3.3　動作例

図９－２／ＪＴ－Ｘ５００～図９－６／ＪＴ－Ｘ５００に様々な要求処理の例を示し、以下で説明する。

(1)　図９－２／ＪＴ－Ｘ５００では、ＤＳＡは他のＤＳＡから紹介を受信すると、紹介で指定されたＤＳＡへその要求を伝達するか又は紹介を元のＤＵＡへ戻す。

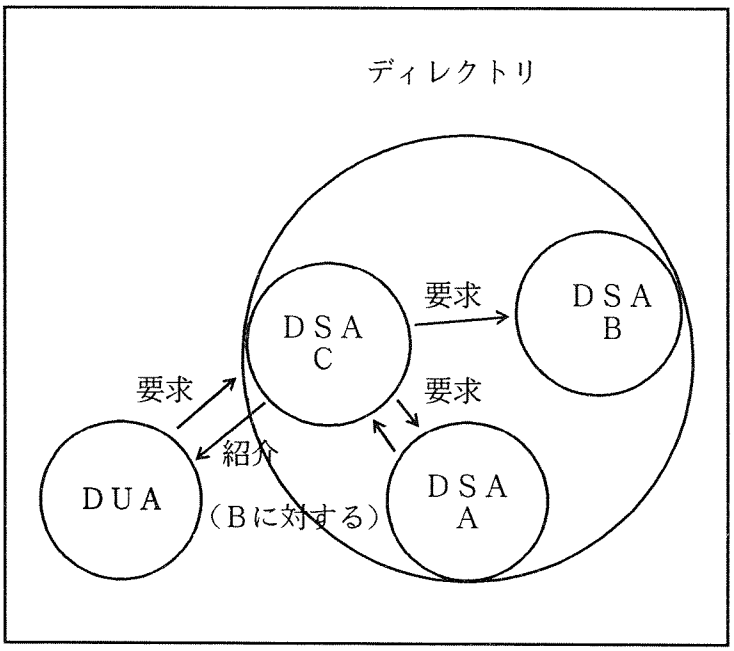


図９－２／ＪＴ－Ｘ５００　紹介(1)
(ITU-T X.500)

（注）もしもＤＳＡ．ＣがＤＵＡに対して紹介を返送すれば、（ＤＳＡ．Ｂに対する）要求は発生しない。同様にＤＳＡ．ＣがＤＳＡ．Ｂに要求を伝達すれば、ＤＵＡに対する紹介は返送されない。

図９－３／ＪＴ－Ｘ５００では、ＤＵＡは一つのＤＳＡから紹介を受信すると、紹介で指定されたＤＳＡへ直接その要求を伝達する。

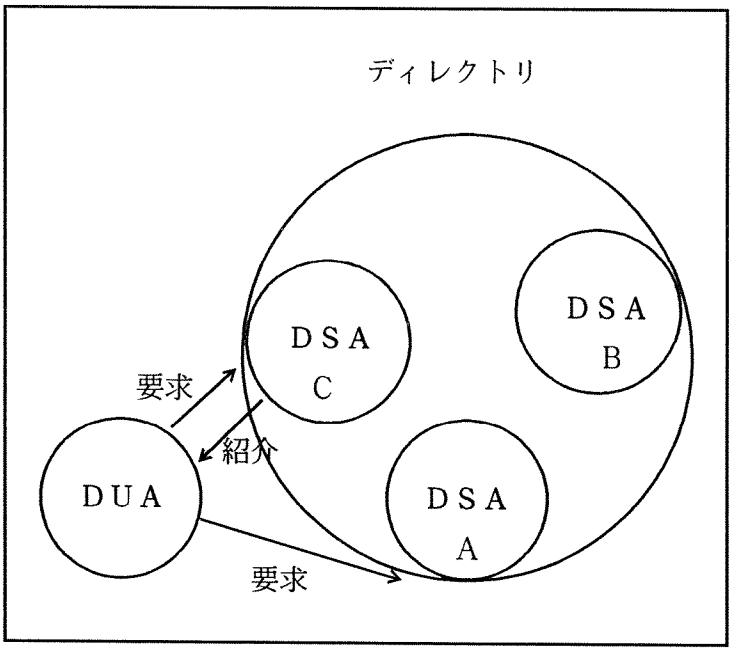


図９－３／ＪＴ－Ｘ５００　紹介(2)
(ITU-T X.500)

(2) 図９－４／ＪＴ－Ｘ５００ではＤＳＡ連鎖、すなわち応答を返送する前の、ＤＳＡ間における要求の経由を示す。

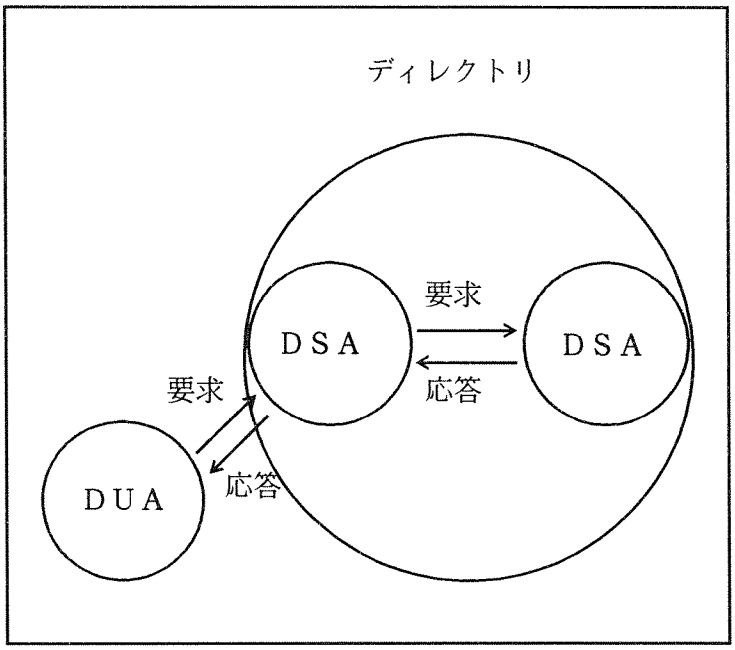


図９－４／ＪＴ－Ｘ５００　連　鎖
(ITU-T X.500)

(3) 図９－５／ＪＴ－Ｘ５００ではマルチ連鎖を示す。すなわち、ＤＵＡと連結しているＤＳＡは、ＤＵＡからの要求を二つ以上の他のＤＳＡに送ることで処理するが、この時、ＤＳＡへ送られる要求は同一である。

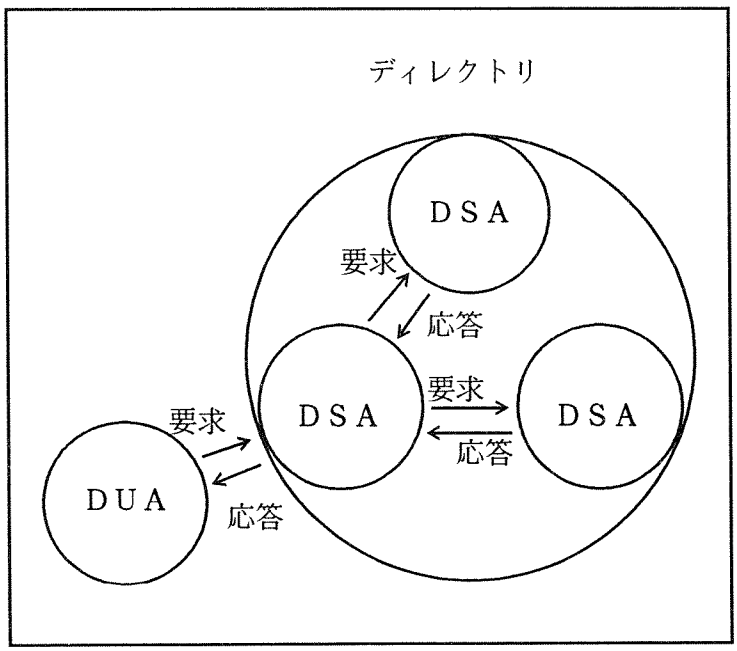


図９－５／ＪＴ－Ｘ５００　マルチ連鎖
(ITU-T X.500)

(4) どのアプローチもそれぞれ長所がある。例えば、図９－３／ＪＴ－Ｘ５００でのアプローチは、ローカルなＤＳＡの負荷を軽減するのが望ましい場合に利用される。別の環境下では、図９－６／ＪＴ－Ｘ５００に示すように、発信者の要求を満足するためにふさわしい相互動作モードの組合せによる複合アプローチが必要となる。

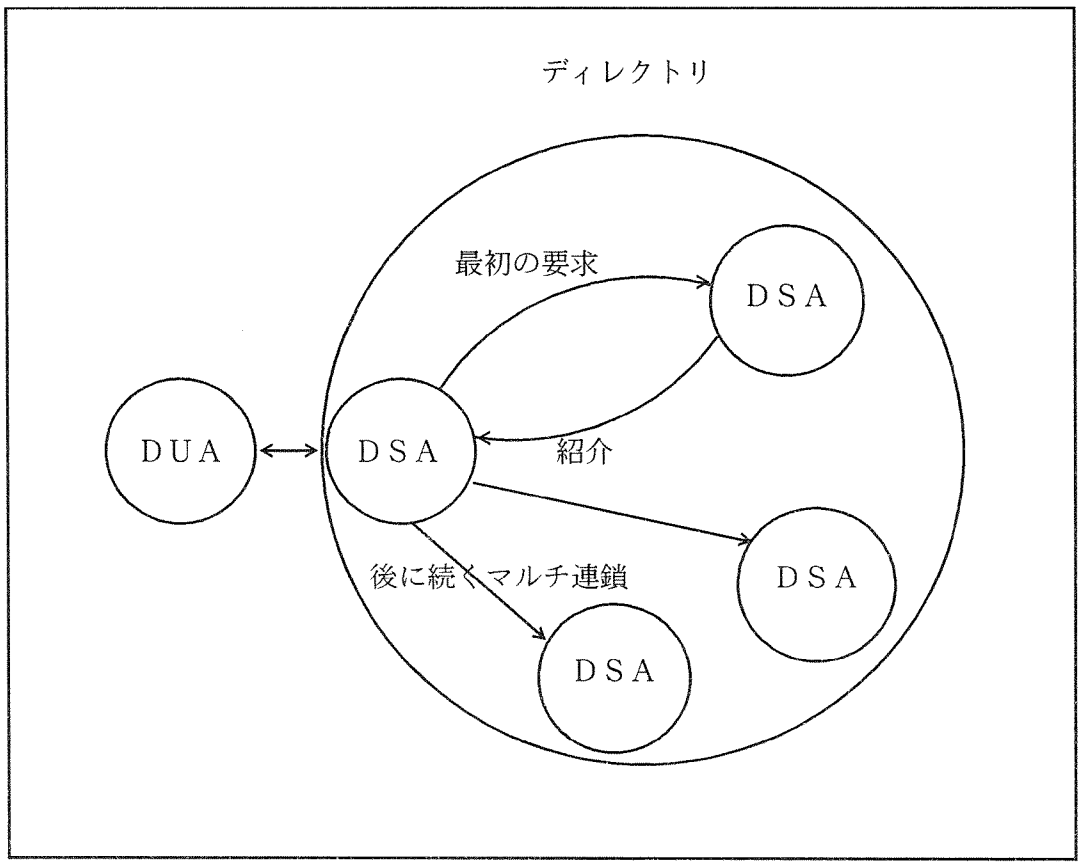


図９－６／ＪＴ－Ｘ５００　モード組合せによる複合アプローチ

１０．　ディレクトリのアクセス制御

（注）ディレクトリアクセス制御のモデルはＩＴＵ－Ｔ勧告　Ｘ．５２０／ＩＥＣ９５９４－２で定義されている。

ディレクトリ情報へのアクセスは、管理機関で管理されたセキュリティ方針によって限定されている。ディレクトリへのアクセスに影響するセキュリティ方針の２つの側面は、認証手順とアクセス制御機構である。

ディレクトリを支える認証の手順と機構は、ＤＳＡ、ディレクトリの利用者、アクセス点で受信された情報の発信源の身元を検証する方法及び、必要に応じてそれらを普及させる方法を含む。汎用的な認証手順はＩＴＵ－Ｔ勧告　Ｘ．５０９／ＩＥＣ９５９４－８で定義されている。

ディレクトリを支えるアクセス制御機構の定義は、アクセス制御情報を特定する方法、そのアクセス制御情報によって定義されているアクセス権を実施する方法、アクセス制御情報を維持する情報を含む。アクセス権の実施はＤＩＴ構造、ディレクトリ利用者情報、アクセス制御情報を含むディレクトリ運用情報に関係付けられたディレクトリ情報へのアクセス制御を包含する。

ＩＴＵ－Ｔ勧告　Ｘ．５０１／ＩＥＣ９５９４－２は、ディレクトリに対する基本アクセス制御として参照される一つの特定のアクセス制御機構（潜在的には多数のアクセス制御機構がある）を定義している。管理機関はセキュリティ方針を実施するに当り、このアクセス制御機構の全てまたは一部を用いても良いし、またその管理機関の裁量において独自のアクセス制御機構を自由に定義して良い。基本アクセス制御機構は、ＤＩＢ（潜在的には構造とアクセス制御情報を含む）内部のディレクトリ情報へのアクセスを制御する手段を提供する。情報へのアクセスを制御することは、その情報の無許可の検索、公表、改変の防止を可能とする。

ディレクトリのための基本アクセス制御モデルは、その全ての操作に対して、アクセス

制御の判断が起こり得る幾つかの要点を定義している。各アクセス制御の判断は次の点を含む：

（1）アクセスされつつあるディレクトリ内部の構成要素
（2）その操作を要求している利用者
（3）その操作の一部を完了するのに必要な特定の権利
（4）その項目へのアクセスを統括するセキュリティ方針

11　ディレクトリにおける複製

（注）ディレクトリの複製は、ＩＴＵ－Ｔ勧告　Ｘ．５２５及びＩＳＯ／ＩＥＣ９５９４－３で定義されている。

11．1　概要

ディレクトリの複製とは、ディレクトリエントリ情報のコピー、および、ある情報の新規登録と変更の責任を負うＤＳＡ以外のＤＳＡにより保持されるディレクトリ操作情報の存在を言う。オリジナル情報を含むＤＳＡをマスターＤＳＡと呼ぶ。

複製情報を全く用いないディレクトリシステムを作成することは可能である。

ディレクトリ情報の複製は、２つの一般的要求を満たすためにある。１つはディレクトリが提供する一般的サービス品質に関する要求で、もう１つはディレクトリシステムの管理に関する要求である。

ディレクトリエントリ情報の追加コピーを用意することは、以下の点において、ディレクトリが提供するサービスの改善に役立つ。

(a)　ディレクトリ情報を、それを欲する特別のディレクトリユーザに近付けることにより

、ディレクトリシステムの性能を改善する。

(b) ディレクトリ情報及びディレクトリ構成要素に冗長性を持たせることにより、個々の構成要素の障害がＤＩＴのある部分の情報への全アクセスを妨げることが避けられ、ディレクトリサービスの信頼性を改善する。

ディレクトリエントリ情報の追加コピーを用意することは、以下の点においてディレクトリシステムの管理に役立つ。

(a) ある操作情報（つまり知識）の広まりを容易にする。

(b) 他のディレクトリ構成要素により保持されている情報のコピーから、あるディレクトリの構成要素により保持されていた情報を再構成することにより、深刻なシステム障害からの復旧の機会を提供する。

１１．２　ディレクトリ複製の形式

ディレクトリの構成要素に保持される複製されたエントリ情報には、キャッシュコピーとシャドウ情報の２つの形式がある。

キャッシュコピーは、ディレクトリの構成要素がディレクトリ仕様には規定されない方法で獲得し利用する、エントリ情報のコピーである。

シャドウ情報は、ディレクトリの構成要素がＩＴＵ－Ｔ勧告X．５２５｜ＩＳＯ／ＩＥＣ９５９４－９に規定された方法で獲得し利用するディレクトリ情報のコピーである。

ディレクトリシステムエージェントは、他のＤＳＡから獲得した情報を、情報がもともと提供された際の合意や方針で認められている場合に限り、保持してよい。

このような情報を保持しているＤＳＡは、その情報に関係するアクセス制御方針と一致した場合のみ情報をＤＵＡに供給してもい。もしその情報上に読み出しアクセス制御がないことが知られていたら、読み出しが許可されたように供給してもよい。

コピー情報を有するＤＳＡは、コピー情報の更新要求、コピー情報を使用するべきでないことを示す要求を、すべて、情報を有するマスタＤＳＡに転送する。

コピー情報への質問に答える時、コピー情報を有するＤＳＡは、要求を満足するため、

コピーが使用されたことを示さなければならない。

シャドウ提供者である或るＤＳＡが、シャドウ消費者である他のＤＳＡに、ＤＩＴの合意された部分からの情報の提供を契約するための、シャドウイング合意を２つのＤＳＡに責任ある管理機関が締結してもよい。

シャドウ情報を獲得する際のシャドウイング合意により許可されたならば、シャドウ消費者は、他のＤＳＡとの合意でその情報のシャドウ供給者になってもよい。

更新が指示されたディレクトリの構成要素によって維持管理されているエントリ情報と不一致になり、ずっと不一致であるキャッシュエントリ情報に比べ、シャドウ消費者の有するシャドウエントリ情報は、シャドウ合意の一部として契約された計画に従い、シャドウ供給者の有する対応する情報（それ自身もシャドウエントリ情報であってよい）と一致させられる。

シャドウ消費者の有するエントリ情報を更新してもらうのに加え、運用情報（知識情報）もシャドウ供給者によりシャドウ消費者に提供されてもよい。

複製を提供するシャドウ合意では、複製される情報は、典型的には、以下の３つの要素からなる。

－ＤＩＴのサブツリー内からのエントリ情報の複製

－複製情報をすべて読みだすために必要なアクセス制御を含む関連運用情報

－下位の知識情報（も可）

複製された情報は、サブツリー内の完全な情報の部分集合を構成する。つまり、

－エントリの選択はオブジェクトクラスのある領域に合致するもののみを特定して、行われる－各エントリ内で、属性の仕様に従った属性の選択。

１１．３　ディレクトリ情報の複製と一貫性

ディレクトリにおける一貫性は、全定義属性のコピーが同一の時に成り立つ。ディレクトリ内にシャドウ情報として一時的な不一致が存在でき、またキャッシュ情報としての継続的な不一致が存在するため、時として一貫性は完全ではなくても成立する場合がある。

キャッシュされたエントリ情報は、更新の指図をうけたディレクトリ要素により保守さ

れたエントリ情報と不明瞭に矛盾して残ってもよい。対照的にシャドウ消費者により保持されたシャドウ情報は、シャドウ供給者により保持された相当する情報をシャドウイング合意の一部として契約されたスケジュールに従うことで合意が成立した。

個々のオブジェクトエントリの実例に含まれている情報が内部的に一貫性のあることが必須である。いくつかの複製の機構は、複製情報の内部的な一貫性とサービスの信頼性を維持するための機構を伴うであろう。エントリの内部的一貫性を保証するため、ディレクトリではスキーマ手順を定義する。

ＤＳＡを挟んで、ＤＩＴが分配されることを許す知識情報が正確であることもまた必須である。いくつかの複製の機構は、知識情報の正確さとサービスの信頼性を維持するための機構を伴うであろう。ＤＩＴの正確さを保証するため、ディレクトリではＤＳＡで必須とする最少知識情報をうまく処理するための手順を定義する。

ディレクトリ情報が複製される環境では、一貫性を確立するためのディレクトリは特定期間の制限をしない。複製情報の使用者は次の理由によりその情報に高い信頼を持てる。
－複製情報は内部的に一貫性がある。
－ＤＩＴに関連する情報は正確である。
－複製されたエントリは最終的にマスタＤＳＡ内のエントリと一致する。

１１．４　複製のビュー

本節では、ディレクトリ情報の複製の存在を以下の対象に対して見せるための明確な手段について述べる。

(a)　ディレクトリユーザ

(b)　管理ユーザ

(c)　ディレクトリの運用的構成要素（ＤＳＡs）

１１．４．１　ディレクトリユーザのビュー

ディレクトリの運用上の性質から、複製された情報は、一般的にその情報にとってマスタとなるＤＳＡによって保持される情報と一致するであろう。従って、一般的に、要求された情報、すなわちユーザに返される情報は、受け入れ可能な性質をもつであろう。そしてそれがコピーから送られてきた情報であるという事実は重要ではなくなるであろう。

ディレクトリユーザは、要求がエントリのコピー情報によって満足された場合、常にその通知を受ける。ユーザが厳密な要求を持っている場合、またはデータの矛盾を検出できる場合は、ユーザはマスタＤＳＡに保持される情報にアクセスを要求できるようなオプションを持つ。

従ってディレクトリユーザには、時々時代遅れの情報を受信する事を了解した上でより高度な性能と利便性を追求するか、または性能と利便性を犠牲にして情報の最高レベルの実時間性を追求するかといった選択肢が提供される。

１１．４．２　管理ユーザのビュー

管理ユーザはＤＳＡによって保持される情報とＤＳＡにより提供されるサービスに関する責任を負う。この管理機能を行うために、管理ユーザはＤＳＡのサービスをモニタし、制御し、最適化するためのツールを要求する。

複製をサポートするための標準化された（そしてローカルな）能力は、ＤＳＡにより提供されるサービスを最適化するために管理ユーザが用いる事ができる基本的なツールのひとつである。

１１．４．３　ＤＳＡのビュー

ＤＳＡは複製された情報とマスタにより保持される情報との間の違いを検出可能であるか、ＤＳＡは一般的に両者を同じように用いる。つまり、ユーザの質問要求に対して、最も便利である形になるように、いずれかの情報で要求を満足させる。

マスタと複製された情報の間の同等性には２つの例外がある。ＤＳＡはＤＩＢの修正要求と複製された情報不可と表示された質問要求に対してはエントリ情報のみ用いて要求を満足させる。

さらに、ローカルに保持される情報は部分的である事がわかるであろうから（１１．２参照）要求された情報をよりよく提供するために、ＤＳＡはエントリを他のエントリにパスしてもよい。

１１．５　複製とアクセス制御

アクセス制御モデルでは，アクセス制御情報がＤＩＴのある領域に対して特定されることを許可している．その領域は，ＤＳＡの境界に及ぶかもしれない．複数のＤＳＡが関係している場合，各ＤＳＡは適切なアクセス制御情報を保持する．

あるエントリが，別のＤＳＡに複製される時はいつも，アクセス制御情報も複製されなければならない．

１２．ディレクトリプロトコル

１２．１　プロトコルの種類

ディレクトリプロトコルを次に示す。

・ディレクトリアクセスプロトコル（ＤＡＰ）：
ＤＵＡとＤＳＡ間での要求とその結果の交換を定義する。

・ディレクトリサービスプロトコル（ＤＳＰ）：
２つのＤＳＡ間での要求とその結果の交換を定義する。

・ディレクトリ情報ジャドウイングプロトコル（ＤＩＳＰ）：
シャドウ合意された２つのＤＳＡ間での複製情報の交換を定義する。

・ディレクトリ運用結合プロトコル（ＤＯＰ）：
管理運用結合された２つのＤＳＡ間での管理情報の交換を定義する。

１２．２　プロトコルの構成

各プロトコルは、応用コンテキストによって定義され、プロトコル要素の集合である。例えば、ＤＡＰは、ディレクトリの照会と変更に関連するプロトコル要素から構成される。

１２．３　応用コンテキストの構成

各応用コンテキストは、応用サービス要素から構成される。応用サービス要素間の相互作用を構成し、サポートするために、応用サービス要素は、遠隔操作サービス（ＲＯＳ（ＩＴＵ－Ｔ勧告Ｘ．２１９｜ＩＳＯ／ＩＥＣ９０７２－１）を使用することが定義されている。このように、ＤＡＰとＤＳＰはＲＯＳの表記法を用いた遠隔操作とエラーからなる集合として定義される。

１３．　ディレクトリモデル

１３．１　定　義

本章で使用する以下の用語は、4.2で定義されている。

(1) アクセス点（Access Point）

(2) アドミニストレーションディレクトリ管理領域
（Administration Directory Management Domain）

(3) 管理機関（administrative authority）

(4) ディレクトリ（the Directory）

(5) ディレクトリ管理と運用情報
（Directory administrative and operational information）

(6) ディレクトリ管理領域（Directory Management Domain）

(7) ディレクトリシステムエージェント（Directory System Agent）

(8) ディレクトリユーザ（Directory user）

(9) ディレクトリユーザエージェント（Directory User Agent）

(10) ＤＩＴ領域（DIT Domain）

(11) 領域管理組織（Domain Management Organization）

⑿ ディレクトリユーザ情報（Directory user information）

⒀ 私設ディレクトリ管理領域（Private Directory Management Domain）

１３．２　ディレクトリとユーザ

ディレクトリは情報の格納場所である。この格納場所は、ディレクトリ情報ベース（ＤＩＢ）と総称される。ユーザに提供するディレクトリサービスは、これらの情報への様々な種類のアクセスに関係する。

ディレクトリで提供されるサービスは、ITU-T勧告X.511 | ISO/IEC 9594-3で定義されている。

ディレクトリユーザ（例えば、人、または、応用プロセス）は、ディレクトリにアクセスすることにより、ディレクトリサービスを享受する。より正確には、個々のユーザに代わって、ディレクトリユーザエージェント（ＤＵＡ）がディレクトリにアクセスし、ディレクトリと相互動作して、サービスを享受する。ディレクトリは、このようなアクセスのために、１つ以上のアクセス点を提供する。これらの概念を、図１３－１／ＪＴ－Ｘ５００に示す。

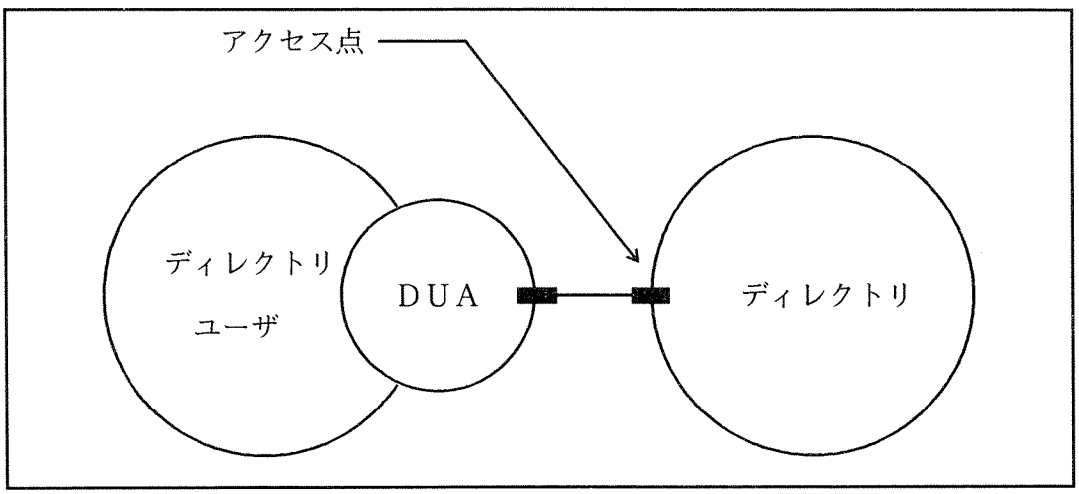


図１３－１／ＪＴ－Ｘ５００　ディレクトリへのアクセス
（ITU-T X.501）

ＤＵＡは応用プロセスとして実現される。ＤＵＡとディレクトリユーザは常に１対１に対応する。

ディレクトリは、ディレクトリシステムエージェント（ＤＳＡ）と総称される１つ以上の応用プロセスの集合として実現され、それらは１つ以上のアクセス点を提供する。

注）

1　ある開放型システムは、集約されたＤＵＡに実ユーザ（応用プロセス、人等）のための情報を取り出す機能を提供してもよい。これはディレクトリに透過である。

2　ＤＵＡ機能とＤＳＡは同じ開放型システムに共存でき、一つ以上のＤＵＡが応用エンティティとしてＯＳＩ環境内で明瞭にさせるかどうかは、実装上の選択である。

3　ＤＵＡは、ITU-T勧告及びISO/IEC 9594の範囲外でローカルな動作や構造を備えてもよい。例えば、ディレクトリユーザが人であるＤＵＡは、問い合わせを作成するものを援助したり、応答を解説するローカルな機能を提供してもよい。

１３．３　ディレクトリとＤＳＡ情報モデル

１３．３．１　一般的モデル

ディレクトリ情報は以下の様に分類される。

－　ユーザ又はユーザの代替者によりディレクトリに置かれ、ユーザ又はユーザの代替者によって、継続して管理されるユーザ情報。ITU-T X.501 3章でこれらの情報モデルを規定している。

－　多様な管理的及び運用的な要求に応じるディレクトリにより保持される管理及び運用情報。ITU-T X.501 5章でこれらの情報モデルを規定している。この章ではまた、ユーザ、管理及び運用情報モデル間の関係についての仕様を規定している。

異なる視点からＤＩＢの見方を表すこれらのモデルは、一般的なディレクトリ情報モデルとして参照される。

ディレクトリ情報モデルは、総括してディレクトリが情報を表現する方法を記述する。潜在的に共同するＤＳＡの集合としてのディレクトリの構成はこのモデルから抽象化される。一方、ＤＳＡ情報モデルは、ディレクトリを含むＤＳＡの集合が共同でディレクトリ情報モデルを実現するためにＤＳＡにより保持されなければならないＤＳＡとその情報に特に関係している。ＤＳＡ情報モデルはITU-T X.501 9章で規定している。

ＤＳＡ情報モデルは、ＤＳＡにより保持される情報とこれらの情報、ＤＩＢ及びＤＩＴ間の関係を記述している一般的なモデルである。

ＤＳＡ情報モデルにより表現される情報のいくつか（全てではない）は、ディレクトリ抽象サービスを経由してアクセスされる。それゆえ、これらのディレクトリ仕様に記述された全ての情報の管理は、ディレクトリ抽象サービスを経由する可能性はない。ＤＳＡ情報の管理は、初期はローカルマターとなるだろう。しかし偶発的にいくつかの一般的システム管理サービスが、ＤＳＡ情報モデル内で記述された情報の全てへのアクセスを備えることに従事させられるだろう。

１３．３．２　特定情報モデル

ディレクトリ全体やその要素、特定情報モデルの次の拡張に従い、ディレクトリやその要素の運用の特別な面での標準化のため情報モデルが要求される。

一般的なディレクトリ情報モデルは、以下にあげる特定の情報モデルの骨組を確立する。

- アクセス制御情報モデル
- サブスキーマ情報モデル
- 集合属性情報モデル

一般的なＤＳＡ情報モデルは、今後は以下にあげる特定情報モデルの骨組を確立する。

－　ＤＳＡの分散知識のモデル

－　ＤＳＡの写し知識のモデル

１３．４　ディレクトリ管理機関モデル

単一の組織により管理される、一つ以上のＤＳＡと０以上のＤＵＡの集合体は、ディレクトリ管理領域（ＤＭＤ）を形成する。

ＤＭＤ（を形成するＤＳＡ）によって保持されるグローバルなＤＩＴの一部は、ＤＩＴ領域として参照される。ＤＭＤとＤＩＴ領域は一対一に対応するものである。ＤＭＤという用語は、ディレクトリの機能要素の管理に関する場合に使用される。ＤＩＴ領域という用語は、ディレクトリ情報の管理への参照の場合に使用される。この用語に関する重要な点は、以下の２つである。

－ＤＩＴ領域は、ＤＩＴの一つ以上のばらばらになったサブツリーから構成される。
ＤＩＴ領域は、グローバルなＤＩＴのルートを含むべきではない。

－ＤＭＤという用語は、また両方の管理面が一緒に考察された場合も、一般的な用語として使用されてもよい。

ＤＭＤ（そしてＤＩＴ領域が属している。）を管理する組織は、領域管理組織（ＤＭＯ）として参照される。

（注）ＤＭＯはアドミニストレーション（すなわち、公衆電気通信の主官庁、または、公衆電気通信サービスを提供する他の組織体）でもかまわない。この場合には、ＤＭＤはアドミニストレーションＤＭＤ（ＡＤＤＭＤ）と呼ばれる。それ以外の場合に

は、DMDは私設DMD（ＰＲDMD）と呼ばれる。ＩＴＵ－Ｔ委員による私設ディレクトリシステムの為のサポート条項は、国際規定の構成の中に含まれることを認めるべきでない。このように記述された技術的可能性は、ディレクトリサービスを提供するアドミニストレーションによって、提供しなくてもよい。

私設ＤＭＤの内部運用及び構成は、ディレクトリ仕様の範囲内ではない。

図１３－２／ＪＴ－Ｘ５００は、DMO、DMD及びＤＩＴ領域間の関係を示している。

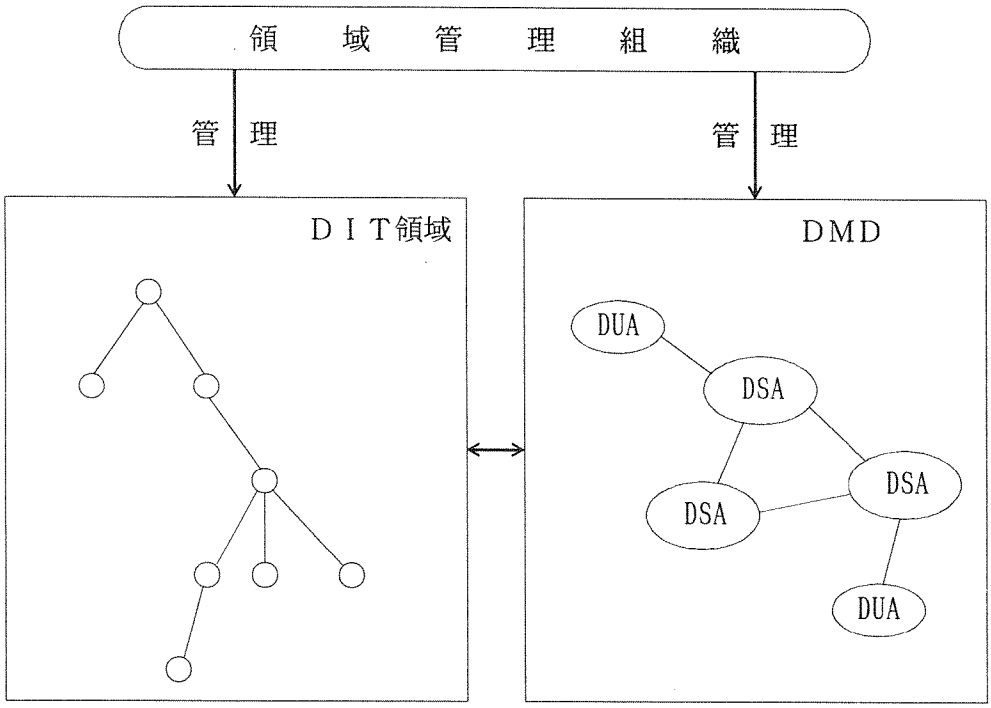


図13-2／ＪＴ－Ｘ５００－ﾃｨﾚｸﾄﾘ管理
（ITU-T X.500）

ＤＭＯによるＤＵＡの管理は、ＤＵＡに関するサービスの責任がＤＭＯにあることを意味する。例えば、ＤＭＤがＤＵＡの保守を行うことや所有権を持つことを意味する。ＤＭＯは、ＤＭＤ内に閉じたＤＵＡとＤＳＡ間の情報交換のために、ディレクトリ関連のディレクトリ仕様を利用しなくてもよい。

ディレクトリ管理の多方面に関するＤＭＯエージェントは、管理機関として参照される。

用語の administrative authority（小文字の場合）は、方針を実施するＤＭＯによって

管理機関内で付与された権限を任せられる。

（注）ディレクトリ管理機関モデルは、ITU-T X.501 4章に規定されている。

１４．ディレクトリ情報ベース

１４．１　定　義

本章で使用する以下の用語は4.3で定義されている。

(1) 別名エントリ（alias entry）

(2) ディレクトリ情報ベース（Directory Information Base, DIB）

(3) ディレクトリ情報ツリー（Directory Information Tree, DIT）

(4) ディレクトリエントリ、エントリ（Directory entry, entry）

(5) 直接上位（immediately superior）

(6) 直接上位エントリ（immediately superior entry）

(7) 直接上位オブジェクト（immediately superior object）

(8) （対象となる）オブジェクト（object of interest）

(9) オブジェクトクラス（object class）

(10) オブジェクトエントリ（object entry）

(11) サブクラス（subclass）

(12) スーパクラス（superclass）

(13) 上位（superior）

(14) 下位（subordinate/inferior）

(15) サブツリー（subtree）

(16) サブツリー精緻化（subtree refinement）

(17) 直接スーパクラス（direct superclass）

１４．２　オブジェクト

１４．２．１　オブジェクト

ディレクトリの目的は、ある「世界」に存在する‘対象となるオブジェクト’に関する

情報の保持とオブジェクトへのアクセスを実現することにある。オブジェクトは、その世界で識別可能な（名前付けられる）ものなら何でもよい。

（注１）「世界とは、一般に、電気通信や情報処理の世界もしくはその一部である。

（注２）ディレクトリのオブジェクトは必ずしも世界の実際の物と正確には対応しない。たとえば、実世界の人は、ディレクトリでは、ビジネス上の人と居住者の二つの異なったオブジェクトと見なされる。本標準では、その対応は規定しないが、ある応用分野ではディレクトリの利用者と提供者間の問題となる。

１４．２．２　オブジェクトクラス

オブジェクトクラスとは、ある特性を共有する識別可能なオブジェクト（または想像されるオブジェクト）群である。あるオブジェクトクラスは、他のオブジェクトクラスのサブクラスであることもある。その場合、サブクラスのメンバはまたスーパクラスのメンバである。サブクラスのサブクラス・・・というように、任意の深度まで考えられる。

１４．３　ディレクトリエントリ

１４．３．１　ディレクトリエントリ

ＤＩＢは（ディレクトリ）エントリより構成される。エントリとは情報の集合であり、かつそれには名前が付けられている。

エントリには以下の３種類がある。

－オブジェクトエントリ：ＤＩＢの中におけるあるオブジェクトに関しての情報の主要な集合である。あるオブジェクトに対応して、正確にひとつのオブジェクトエントリが存在する。オブジェクトエントリはオブジェクトを表わすものである。

－別名エントリ：オブジェクトエントリに対して別の名前を与えるために使用する。

－サブエントリ：ＤＩＢの中の情報の集合を表わし、ディレクトリの管理や運用上の要求を処理するために使用される。

１４．３．２　ディレクトリエントリの構成

ディレクトリエントリの構成を図１５－１／ＪＴ－Ｘ５００に示し、１５．２で解説する。

１４．３．３　オブジェクトクラス

各エントリは、そのエントリが属するオブジェクトクラス及びそのオブジェクトクラスのスーパクラスの表示を含む。

１４．３．４　管理エントリ

いくつかのオブジェクトエントリは、ディレクトリの管理の目的で特別に指定される。これらのエントリは、管理エントリと呼ばれる。ディレクトリユーザは通常はこの事を知らず、これらのエントリは他のオブジェクトエントリと同じ様に見える。

１４．４　ディレクトリ情報ツリー（ＤＩＴ）

潜在的に非常に大きいと考えられるＤＩＢの分散と管理に対する要求を満足し、かつ、オブジェクトを曖昧さなく指定でき（１６章参照）、エントリを見つけるためには、平面的な構造のエントリは、現実的でない。従って、エントリをディレクトリ情報ツリーと呼ぶツリー状に構成し、通常のオブジェクトにある階層的な関係（例えば、ある人がある課で働いており、その課がある組織に属し、その組織がある国に本社を置く。）を持たせるべきである。

（注）ツリー構造の概念と用語について、付録２に示す。

１４．４．１　ＤＩＴの構成

ＤＩＴの構成要素は、以下のように解釈される。

(1) 節点はエントリである。オブジェクトエントリは末端節点、もしくは、非末端節点であり、別名エントリは常に末端節点である。ルートはヌルオブジェクトエントリ((4)参照）と見なされる。

(2) 有向線分は、節点（従ってエントリ）間の関係を規定する。節点Ａから節点Ｂへの有向線分は、Ａのエントリ はＢのエントリの直接上位エントリ（直接上位）であることを意味し、逆に、Ｂのエントリは、Ａのエントリ直接下位エントリ（直接下位）であることを意味する。特定エントリの上位エントリ（上位）は、その直接上位とそのまた上位（再帰的に）からなる。ある特定エントリの下位エントリ（下位）は、その

直接下位とそのまた下位（再帰的に）からなる。

(3) あるエントリで示されるオブジェクトは、その下位に対する命名機関（１６章参照）と密接に関連する。

(4) ルートは、ＤＩＢの最高位の命名機関である。

１４．４．２　オブジェクトの上下関係

オブジェクト間の上位／下位の関係は、エントリ間の上位／下位の関係から導かれる。第一のオブジェクトのオブジェクトエントリが第二のオブジェクトの任意のエントリの直接上位である時だけ、第一のオブジェクトは第二のオブジェクトの直接上位である。直接下位オブジェクト、直接下位、上位及び下位の用語は（オブジェクトに適用する場合）、類似の意味を持つ。

オブジェクト間の可能な上位／下位の関係は、ＤＩＴ構造の定義（１７．６参照）で規定される。

１４．４．３　ディレクトリエントリの集合

ディレクトリは、ディレクトリエントリに関する情報に加えて、ディレクトリエントリの集合に関する付加情報を保持している。このような集合は（ＤＩＴ）のサブツリーか、または（真のツリー構造でないとき）サブツリー精緻化である。

１５．ディレクトリエントリ

１５．１　定　義

本章で使用する以下の用語は、4.4で定義されている。

(1) 属　性（attribute）

(2) 属性型（attribute type）

(3) 属性値（attribute value）

(4) 属性値提示（Attribute Value Assertion）

(5) 識別値（distinguished value）

(6) 利用者属性(user attribute)

(7) 属性階層(attribute hierarchy)

(8) 属性サブタイプ(attribute subtype)

(9) 属性スーパタイプ(attribute supertype)

(10) 補助オブジェクトクラス(auxiliary object class)

(11) 集合属性(collective attribute)

(12) 直接属性参照(direct attribute reference)

(13) エントリ集合(entry collection)

(14) 間接属性参照(indirect attribute reference)

(15) 照合規則(matching rule)

(16) 照合規則提示(matching rule assertion)

(17) 運用属性(operational attribute)

(18) 構造オブジェクトクラス(structural object class)

(19) エントリの構造オブジェクトクラス(structural object class of entry)

１５．２　全体構造

図１５－１／ＪＴ－Ｘ５００に示すようにエントリは属性の集合から成っている。

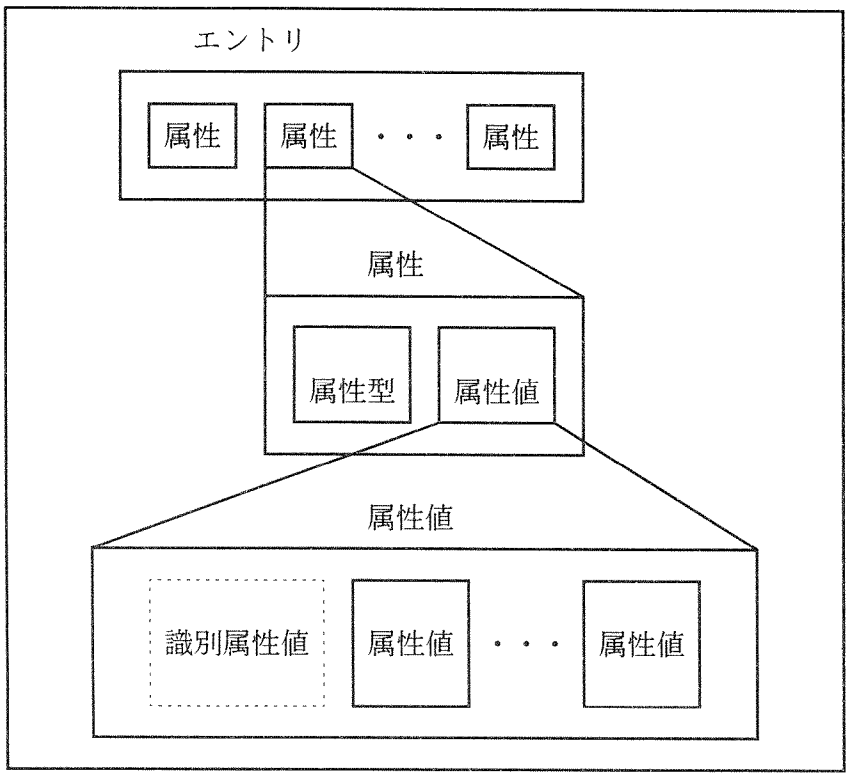


図１５－１／ＪＴ－Ｘ５００　エントリの構造
(ITU-T X.501)

(1)　各々の属性はそのエントリに対応するオブジェクトに関する情報を提供したり、ある特性について記述する。

（注）エントリの中に現われるかもしなれい属性の例としては、オブジェクトの個人名のような名前情報、電話番号のようなアドレス情報などがある。

(2)　属性は属性に与えられた情報クラスを示す属性型とそれに対応した属性値（エントリの中に現われているそのクラスの実現値）から成っている。

```
Attribute        ::= SEQUENCE {
    type     AttributeType( {SupportedAttributes} ),
    values   SET SIZE(1..MAX)OF AttributeValue( {SupportedAttributes}  {@type} ) }
```

（注）属性型と属性値については、15.4節と15.5節でそれぞれ記述される。

属性は単一値または複数値として指定される。ディレクトリは単一値属性がただ一つの値を持つことを保証するであろう。

１５．３　オブジェクトクラス

オブジェクトクラスは、ディレクトリ中において次のような多くの目的のために使用される。

・これらのオブジェクトに相当するオブジェクトとエントリの記述と分類

・必要であれば、ディレクトリ操作の制御にあてる

・ディレクトリＤＩＴ構造規則仕様に関連して、ＤＩＴ内におけるエントリの位置付けを調整する

・ＤＩＴの内容規則仕様に関連して、エントリに含まれる属性を調整する

・適切な管理機関により、特定の方向に関連付けられるエントリクラスを識別する

オブジェクトクラスは国際的に標準化される場合もあるし、私的機関や、国内の管理機関によって定義される場合もある。このことは、個々の機関が、オブジェクトクラスの定義とそれらの曖昧さのない識別に関して責任があることを意味している。これは、オブジェクトクラスを定義するときに、オブジェクト識別子のついた各々のオブジェクトクラスを識別することで実現される。このための記法は17.3.3節に記述されている。

（注）管理機関は、ディレクトリ仕様で定義され、記載されている有効なオブジェクトクラス以外のオブジェクトクラスを使用してもよい。管理機関は、例えばディレクトリ仕様で定義されているオブジェクトクラスを補うためのオブジェクトクラスを定義し、記載してもよい。

オブジェクトクラス（サブクラス）は、それ自身がより包括的なオブジェクトクラスから導かれるオブジェクトクラス（その直接スーパクラス）から導かれる。構造オブジェクトクラスでは、このプロセスは最も包括的なオブジェクトクラスであるtopの位置で停止する。あるオブジェクトクラスに関しての最も上位のオブジェクトクラスに到るまでのスーパクラスの順序集合はスーパクラス連鎖である。

オブジェクトクラスは、２つ以上の直接スーパクラスから導かれる（同じスーパクラス連鎖の一部ではないスーパクラス）。サブクラス化のこの特徴は、多重継承と呼ばれる。オブジェクトクラスの仕様は、属性が必須かオプションかを識別する。この仕様は、そのサブクラスにも適用される。サブクラスは、そのスーパクラスの付加属性仕様と必須属性仕様を継承していると言える。サブクラスの仕様は、スーパクラスの付加属性がサブクラスにおける必須属性であることを表示してもよい。

以下の３種のオブジェクトクラスがある。

・抽象オブジェクトクラス

・構造オブジェクトクラス

・補助オブジェクトクラス

各々のオブジェクトクラスは、必ずこれらの内の一つに属しており、残りのオブジェクトクラスについては、どの様な場合でもディレクトリの中で遭遇する。各々のオブジェクトクラスの定義は、それがどの種類のオブジェクトであるかを定義しなくてはならない。

すべてのエントリは、topオブジェクトクラスのメンバであり、また少なくとも一つの他のオブジェクトクラスのメンバとなるであろう。

１５．３．１　抽象オブジェクトクラス

抽象オブジェクトクラスは、オブジェクトクラス間での共通の特徴を備えた他のオブジェクトクラスを導きだすために使用される。エントリは、抽象オブジェクトクラスのみに属することはないであろう。

topは、全ての構造オブジェクトクラスのスーパクラスとして使用される抽象オブジェクトクラスである。

１５．３．２　構造オブジェクトクラス

ＤＩＴの構造仕様で使用するために定義するオブジェクトクラスを構造オブジェクトクラスと呼ぶ。構造オブジェクトクラスは、分かりやすいエントリのためのオブジェクトの名前の構造の定義に使用される。

オブジェクトまたは別名エントリは、最下位オブジェクトクラスとして単一の構造オブジェクトクラスを持つ一つの構造オブジェクトクラススーパクラス連鎖により明確に特徴付けられている。この構造オブジェクトクラスは、エントリの構造オブジェクトクラスとして参照される。

構造オブジェクトクラスは、関連するエントリに関係付けられる。

・構造オブジェクトクラスに適合するエントリは、オブジェクトクラスにより強制された実世界オブジェクトを表すであろう。

・ＤＩＴ構造規則は、構造オブジェクトクラスにのみ関係する。つまり、エントリの構造オブジェクトクラスは、ＤＩＴにおけるエントリの位置を定義するために使用される。

・エントリの構造オブジェクトクラスは、関連するＤＩＴ内容規則に加えて、エントリの内容を制御するために使用される。

エントリの構造オブジェクトクラスは変更すべきではない。

１５．３．３　補助オブジェクトクラス

ディレクトリで使用される仕様のアプリケーションは、多くの型のエントリの構造において使用される補助オブジェクトクラスを記述するのに便利であることが分かるであろう。例えばメッセージハンドリングシステムは、組織構成員や居住者のように構造オブジェクトクラスが変化しやすいエントリ型のための、必須または付加メッセージハンドリング属性の一まとまりを記述するためにＭＨＳＵｓｅｒ(ITU-T REC.X.402|ISO/IEC10021-2)補助クラスを使用する。

ある環境において、特定の、おそらく標準化されたクラスで許容されている属性のリストに加えたり、削除したりする事ができる必要がある。

この要求は、ローカル団体で維持されており、必要に応じて時々変化する、意味を持つ補助オブジェクトクラスの定義と、これを使用することにより実現できる。

この要求は、ＤＩＴの特定ポイントにおけるエントリに、属性を追加したり、削除したりすることをダイナミックに認めるために、ＤＩＴ内容規則定義のファシリティを使用することにより実現できる（17.3.3参照）。

補助オブジェクトクラスは、エントリかまたはエントリのクラスを記述するものである。

それゆえに、構造オブジェクトクラスのメンバとなることの他に、エントリは、一つ以上の補助オブジェクトクラスのメンバに自由になることが可能である。

エントリの補助オブジェクトクラスは、時とともに変化するであろう。

（注）この節で議論された要求条件を満たすために、１９８８年版のディレクトリ仕様で用いられた未登録オブジェクトクラスファシリティは、現在はＤＩＴ内容規則を用いるため使用されない。

１５．３．４　オブジェクトクラス定義と１９８８年版ディレクトリ仕様の関係

１９８８年版ディレクトリ仕様の用語を用いて定義されるオブジェクトクラスは、構造、補助、抽象のオブジェクトクラスの一つとしては分類されない。

１９８８年版ディレクトリ仕様の用語を用いて記述される別名オブジェクトクラスは、

抽象、補助、構造のいずれかのオブジェクトクラスとして定義されることが考えられる。そして、適宜サブスキーマに展開される。

１５．４　属性型

(1) いくつかの属性型は国際的に標準化されている。その他の属性型は、各国主官庁管理機関あるいは私的機関によって定義される。

このことは、いくつかに分かれた管理機関は、型の定義とその型の曖昧さのない識別に関して責任があることを意味している。

これは属性型の定義時にオブジェクト識別子によって各々の属性型を識別できるよう行なわれる。

１７．４．６節で定義されるＡＴＴＲＩＢＵＴＥ情報オブジェクトクラス記法を用いて、属性型は以下のように定義される。

```
AttributeType ::= ATTRIBUTE.&ID
```

(2) エントリの中のすべての属性は、属性型が異なるものでなければならない。

(3) ディレクトリはいくつかの属性型をディレクトリ自身の目的のために使用する。その属性型には以下のものがある。

(a) オブジェクトクラス（object class）

この型の属性はすべてのエントリに現われ、オブジェクトクラスとそのオブジェクトが属している上位クラスを示す。

(b) 別名付けられたエントリの名前（aliased entry name）

この型の属性はすべての別名エントリに現われ、その別名エントリが参照しているエントリの名前（16.5参照）を持っている。

これらの属性については、17.4.6で定義されている。

(4) オブジェクトまたは別名エントリの中に現れるであろう利用者属性の型は、そのエントリのためのＤＩＴ内容規則（17.7参照）だけでなく、指定されたオブジェクトクラス（一つとは限らない）に適用する規則にも照らして管理される。

サブエントリの中に現れるであろう属性の型はシステムスキーマの規則により管理される。

(5) ディレクトリ利用者には通常見えない特別な属性を含むエントリの存在が許されている。この属性は運用属性と呼ばれ、ディレクトリの運用管理上の要求を満たすために用いられる。

１５．５　属性値

(1) 属性を定義することは、属性の中の値が従わなければならない構文を定める（このことはデータ型を定めることでもある）ことを意味している。

１７．４．６節で定義されているＡＴＴＲＩＢＵＴＥ情報オブジェクトクラス記法を用いて、属性値は以下のように定義される。

AttributeValue :: ＝ ATTRIBUTE.&Type

(2) 属性の値の１つが識別値として指定されている場合、その属性値はエントリの相対識別名（16.3参照）の中に現われる。

１５．６　属性型の階層

属性型を定義する時、幾つかのより包括的な属性型の特性が、その定義の基本として選択的に用いられることが許されている。新しい属性型はdirect subtypeであり、それはより包括的な属性型supertypeから派生している。

属性階層は、利用者の必要とする情報の品質（詳細情報か、それとも概略情報か）に応じたＤＩＢへのアクセスを可能とする。これは、属性値へのアクセスが、その属性を特定する属性型識別子（その属性への直接参照）、あるいはその属性を指すより包括的な属性型識別子（その属性への間接参照）の何れを用いても実行できるようにすることにより実現される。

より特定的な属性は、より包括的な属性の下位に配置されるので、意味に関連性がある幾つかの属性は階層的に配置されることが想定できる。属性およびその値の探索と獲得は、より包括的な属性型を引用することにより、より容易になる。すなわち、詳細なフィルタ項目は、引用された属性型だけでなく、より特定的な属性型に対して評価される。

下位の特定的な属性型を探索結果の一部として返すために選択し、かつそれが利用可能なら、その属性型を返さなければならない。より包括的な属性型を探索結果の一部として返すために選択し、かつそれが利用可能なら、包括的な属性型及び特定的な属性型の両方を返さなければならない。属性値はその属性型の値として、常に返さなければならない。

ある属性の階層に属する属性型の値を含むエントリに対して、その属性型はそのエントリが属するオブジェクトクラスの定義の中にも明示的に含まれねばならない。なぜなら、ＤＩＴ内容規則がそれを許しているからである。

ある属性の階層の中の全ての属性型は、そのエントリの管理及びエントリの内容の利用者からの変更のために、一意かつ互いに関連のない属性型として扱われる。

ディレクトリのオブジェクトあるいは別名エントリに格納されている属性値は、正確に１つの属性型を持つ。その属性型は、その属性値がエントリに最初に加えられる時に示される。

１５．７　照合規則

１５．７．１　概要

ディレクトリのエントリが保持する属性値に関する提示に基づいて、エントリの集合をＤＩＢから選択できる能力は、ディレクトリにとって極めて重要である。

照合規則は、属性値に関するある特有の提示を行うことによりエントリを選択することを可能としている。

最も基本的な形式の提示は、属性値提示である。より複雑な提示は、照合規則提示を用いて行うことができる。照合規則提示は「真」、「偽」、「未定義」何れかの値を取る命題である。この命題は、照合規則に基づく基準を満足する属性値のエントリが存在するか否かに関する。

属性あるいは照合規則提示は、その提示に関係する照合規則に基づいて評価される。

照合規則は次の４つの仕様により定義される。

(1)　その規則が定めるべき属性構文の範囲

(2)　その規則が定めるべき照合に関する特定の形式

(3)　照合に関する各特定の形式を表記するために要求される構文

(4)　必要に応じて、属性構文の値から提示構文の値を導きだすための規則

（注）特定のアプリケーションを実現するために定義される照合規則に関して、制限はない。しかし、そのような照合規則は、ＤＵＡやＤＳＡにより一般的に実装されないかもしれない。可能な限り、本標準で定義される照合規則が、新しい照合規則の仕様に優先して用いられなければならない。

１つの照合規則と定められた照合形式の間に１対１の対応関係が見られることがある。

例えば、ディレクトリ抽象サービスは、エントリの中の属性の存在を検出するための存在照合規則を定めている。

１つの照合規則と定められた照合規則の間にｎ対ｎの対応関係が見られることがある。例えば、ディレクトリ抽象サービスは、２つの照合形式「大きい、あるいは等しい」、「小さい、あるいは等しい」を与える汎用的な順序規則を定めている。

１５．７．２　属性値提示

属性値提示（ＡＶＡ）は、「真」、「偽」、「未定義」何れかの値を取る命題であり、この判定は、その属性型に該当する照合規則に従って行われる。この照合規則は、ある特定の属性型の属性値に対するエントリの存在に関するものである。属性値提示は下記の表記のように、属性型と属性値を必要とする。

```
AttributeValueAssertion   ::=  SEQUENCE {
    type                          AttributeType(SupportedAttributes),
    assertion                     AssertionValue(SupportedAttributes {@type } ) }
  AssertionValue  ::=  ATTRIBUTE.&equality-match.&AssertionType
```

属性値提示の表記におけるassertionの構文は、その属性型のために定義された同値照合規則により決定されるため、その属性自身の構文とは異なるかもしれない。

属性値提示は下記のように判定される。

(a)　不定　以下のいずれかのとき

(1)　属性型が不明

(2)　その型の属性構文が同値照合規則を持たない。

(3)　提示された属性値が、その属性の同値照合規則の提示構文により示されるデータ型に一致しない。

（注）(2)、(3)は一般的に起こり得る「誤り属性値提示」である。しかし、(1)は局所的に起こるかもしれない。例えば、あるＤＳＡがその属性型を登録していないような場合に起こる。

(b)　真　　エントリがその型の属性を含んでいるとき、それらの値のうち一つがその値と一致する。

(c)　偽　　上記以外

１５．７．３　ビルトイン照合規則提示

関連する照合規則に関するいくつかの分類があることがディレクトリでは理解されている。その意味は一般的に理解され、またこれらは多くの異なる属性型の値に適用可能である。

(1)　存在

(2)　同値

(3)　サブストリング

(4)　順序

(5)　近似照合

これらの照合規則の分類に関係するある照合形式を提示するための構文はディレクトリ抽象サービスの中に構築されている次のものである。

(1)　存在規則に対するpresent構文

(2)　同値規則に対するequality構文

(3)　順序規則に対するGreaterOrEqual、LessOrEqual構文

(4)　サブストリング規則に対するinitial、any、final構文

(5)　近似照合の規則に対するapproximateMatch構文

present構文はあらゆる型のあらゆる属性に使用できる。存在照合は、ある特定の型に関する値の存在の検証を行う。

特定の同値、サブストリング、順序照合規則は、属性型が定義される時にその属性型と関連付けされるであろう。これらの特定の規則は、ディレクトリ抽象サービスに組み込まれた構文を用いて実行する同値、順序、サブストリング規則の提示を評価する際に用いられる。特別の規則が提供されていない場合には、これらの属性に関して実行される提示は定義されない。

近似照合構文は近似照合規則を支援するが、近似照合規則の定義はＤＳＡにとってはローカルマターである。

１５．７．４　照合規則条件

ディレクトリが首尾一貫して動作するために、ディレクトリ抽象サービスの中に構築されている構文と共に用いられねばならない照合規則に関して、ある制限を設けることが必要である。

提示の構文が、照合規則が適用される属性構文と異なるような同値照合規則の場合は、属性構文の値から提示構文の値を導き出すための規則が提供されるであろう。

名前付けに用いられる属性に関する同値照合規則は、他動的であり、かつ代用可能である。また、属性構文と等しい提示構文を持っている。

推移的な照合規則は、値ａが値ｂに照合し、値ｂが３番目の値ｃに照合すれば、その照合規則を用いて値ａは値ｃに必ず照合しなければならないという事実により特徴付けられる。

代用照合規則は、値ａが値ｂに照合するならば、値ｂは値ａに必ず照合しなければならないという事実により特徴付けられる。属性presentationAddressは、照合規則が共通ではない属性構文を定めている属性の例である。

特定の属性については、同値照合規則と順序照合規則が両方とももし存在するなら、少なくとも次の観点において両者は常に関連を持っていなければならない。すなわち、２つの値が順序関係を用いて等しいための必要十分条件は、その２つの値が等価関係を用いて等しいことである。さらに、その順序関係は秩序立っていなければならない。すなわち、全てのｘ、ｙ、ｚに対して、その順序関係によればｘがｙに先立ち、ｙがｚに先立つなら、ｘはｚに先立たなくてはならない。

（注）これらの要求は、順序関係が定義される時には、その定義が等価関係をも定義することを意味する。

特定の属性型については、同値照合規則とサブストリング照合規則が両方とももし存在するなら、少なくとも次の観点において両者は常に関連を持っていなければならない。等値性関係に従って照合する全ての値ｘとｙ、及びサブストリングの関係の全ての値ｚに対して、ｘに対する提示の評価結果は、ｙに対する提示の評価結果に等しくなければならない。すなわち、等価関係を用いて識別不能な２つの値はサブストリングの関係を用いても識別不能でなければならない。

１５．７．５　オブジェクト識別子と識別名の同値照合規則

属性値提示を評価するために用いられ、かつディレクトリ自身が知っており、ディレクトリそのものを運用するために用いる同値照合規則がある。それは次に示すものである。

(1) objectIdentifierMatch

この規則は、属性をObjectIdentifierSyntaxと照合するために用いられる。

(2) distinguishedNameMatch

この規則は、属性をDistinguishedNameSyntaxと照合するために用いられる。

１５．８　エントリ集合

１５．８．１　概要

オブジェクト及び別名エントリの集合は、ある特徴（例えば、集合の中の各々のエントリに対して同じ値を持つある複数の属性）を有するかもしれない。これは、対応するオブジェクトにおける共通の特徴あるいは共有される関係による。このようなエントリの分類は、エントリ集合と呼ばれる。

エントリ集合は、ＤＩＴの中の位置により関係付けられるオブジェクトと別名エントリを含むであろう。これらの集合は、サブツリーまたはサブツリー精緻化として規定される。

あるエントリは幾つかのエントリ集合に属するかもしれない。

１５．８．２　集合属性

利用者属性が、あるエントリ集合の複数のエントリに共有される場合には、それらは集合属性と呼ばれる。

同一の集合属性が、これらの集合の中の２つ以上の集合に独立に関係することも許されている。この場合、そのエントリの集合属性は複数の値を持つ。これゆえ、集合属性は常に複数の値を持つものとして規定されるであろう。

集合属性は、エントリ属性としてディレクトリへの質問操作を行う利用者に提示されるにもかかわらず、ディレクトリ情報モデルのエントリ属性とは相違して扱われる。この違いはディレクトリへの変更操作を行う利用者に明示される。なぜなら、集合属性は関連するサブエントリを経由して管理（すなわち変更）されねばならず、その集合属性の現れるエントリを経由して管理することはできないからである。

（注）これらの値の独立した格納場所は、ディレクトリへの質問操作を行う利用者には明示されない。

集合属性がエントリに現れるためには、その属性型の存在が、そのエントリを管理するＤＩＴ内容規則に従って許されねばならない。

エントリは特定の集合属性を除外するかもしれない。

１６．名　前

１６．１　定　義

本章で使用する以下の用語は4.５で定義されている。

(1) 別　名（alias, alias name）

(2) 別名展開（alias dereferencing）

(3) 識別名（distinguished name）

(4) ディレクトリ名、名前（Directory name, name）

(5) 提示名（purported name）

(6) 命名機関（naming authority）

(7) 相対識別名（Relative Distinguished Name）

(8) 日本語名（Japanese name）

(9) エントリ名（(entry) name）

１６．２　概　要

１６．２．１　名前の条件

「名前」はすべてのオブジェクトの中から特定のオブジェクトを識別できる構造を持つ。この名前は唯一のオブジェクトを示すため曖昧なものではいけない。

（ディレクトリ）名はエントリも識別する。このエントリはオブジェクトを表すオブジェクトエントリかまたはディレクトリがオブジェクトを表すエントリの位置を決定するのを支援するための情報を含む別名エントリかのいずれかである。

（注）このことより、オブジェクトの名前の集合は、オブジェクトの識別名とともに、
　　オブジェクトの別名の集合を含むことになる。

オブジェクトは、ディレクトリの中のエントリにより表されなくても識別名を割当られる。しかしこの名前は、そのオブジェクトエントリが持つであろう名前であり、それはディレクトリの中で特別の何かを表さなくてもよい。

１６．２．２　名前の構成

名前は構造的に、相対識別名を順に並べたものである。（16.3参照）

```
Name ::= CHOICE  {--現在は、ただ１つのみ可能-- RDNSequence}

  RDNSequence ::= [APPLICATION 1]
```

SEQUENCE OF RelativeDistinguishedName

DistinguishedName ::= RDNSequence

（注）上記以外の方法で構成された名前は将来拡張されるかもしなれい。

16．2．3　部分シーケンス

オブジェクトの中の部分シーケンスで示される名前もオブジェクトを示す名前である。ディレクトリ情報ツリーを上位から順に辿って行くことによりこのオブジェクトに対する名前を識別できる。

16．2．4　提示名

「提示名」は名前の形をしているが、必ずしも名前として有効ではない。

16．3　相対識別名

16．3．1　構　成

個々のオブジェクトとエントリは相異なった「相対識別名」を持つ。この相対識別名はエントリの中の識別名に関係する属性型と属性値の組の集合から構成される。

それぞれは、同値照合規則を用いて明確に識別されたエントリの属性値に一致する。

（注）同値照合規則が利用可能である。何故なら、同値照合規則の属性構文と提示構文が同一であるから。

```
RelativeDistinguishedName    ::= SET SIZE(1..MAX)OF AttributeTypeAndValue
AttributeTypeValue   ::=     SEQUENCE
   type                    (  {AttributeType } ),
   value                   (  {AttributeValue(  {SupportedAttributes }  {@type } ) }
```

この集合はエントリにおける識別値について1つの属性型と属性値の組を持つ。

属性型と属性値に関する2つ以上の組が同一の属性を参照することはない。

16．3．2　相対識別値の割当

同じ直接上位を持つ相対識別名は互いに異なる必要がある。関係機関は、責任をもって

識別値の適切な割当を行う。

（注）時々エントリは単一の識別値を持つ。（そしてＲＤＮは属性型と属性値に関する１つの組を持つ。）しかしながら、差を明確にするために付加的な値が使用されることがある。

１６．３．３　相対識別名の割当

オブジェクトクラスの性質に依存するが、どのような属性型の値も相対識別名の一部を構成することができる。

相対識別名の割当は、関係機関或いは主管庁間での取り決めの有無に関係無く行政上の責任である。本標準はそのような取り決め方法に関知するものではなく、いかに実行するかという点についても判断を与えるものではない。

相対識別名は必要ならば完全な置換により変更できる。

（注１）相対識別名は、長い期間で使用されるため注意深く変更すべきである。

（注２）葉でないエントリのＲＤＮを変更する事で、該当する下位のエントリのＤＮは自動的に変更される。

１６．４　識別名

あるオブジェクトの識別名は、そのオブジェクトとその上位にある全てのオブジェクトに対応するエントリの相対識別名の（降順の）並びとして定義される。オブジェクトとこのオブジェクトエントリは対応するため、オブジェクトを表わす識別名は、オブジェクトエントリの名前ともなる。

なお、日本語名は別名で使用するため、識別名としては用いない。

（注１）人間が取扱うオブジェクトに対する識別名はユーザにとっての親しみやすいことが望ましい。

（注２）ＩＴＵ－Ｔ勧告Ｘ．２００はプリミティブな名前の概念を定義している。識別名は、曖昧さがなく、ユニークであり内部構造を意識しなくて済むなどの理由により、オブジェクトに対するプリミティブな名前として用いられる。

（注３）オブジェクトエントリとその上位はつながるため、オブジェクトの識別名は別名エントリを含まない。

(1)　別名エントリは識別名も持つ。しかし、この名前はオブジェクトの識別名にはなりえ

ない。この区別が必要な場合は、「別名エントリの識別名」という用語を用いる。

一方、別名エントリを示す識別名はオブジェクトエントリの識別名として別名エントリおよび上位のエントリの相対識別名の並びとして定義される。

(2) ルートを示す識別名を定義することは便利であるが、それはオブジェクトの識別名にはならない。ルートを示す識別名は、ヌルシーケンスである。

(3) 相対識別名と識別名の概念を図１６－１／ＪＴ－Ｘ５００に示す。

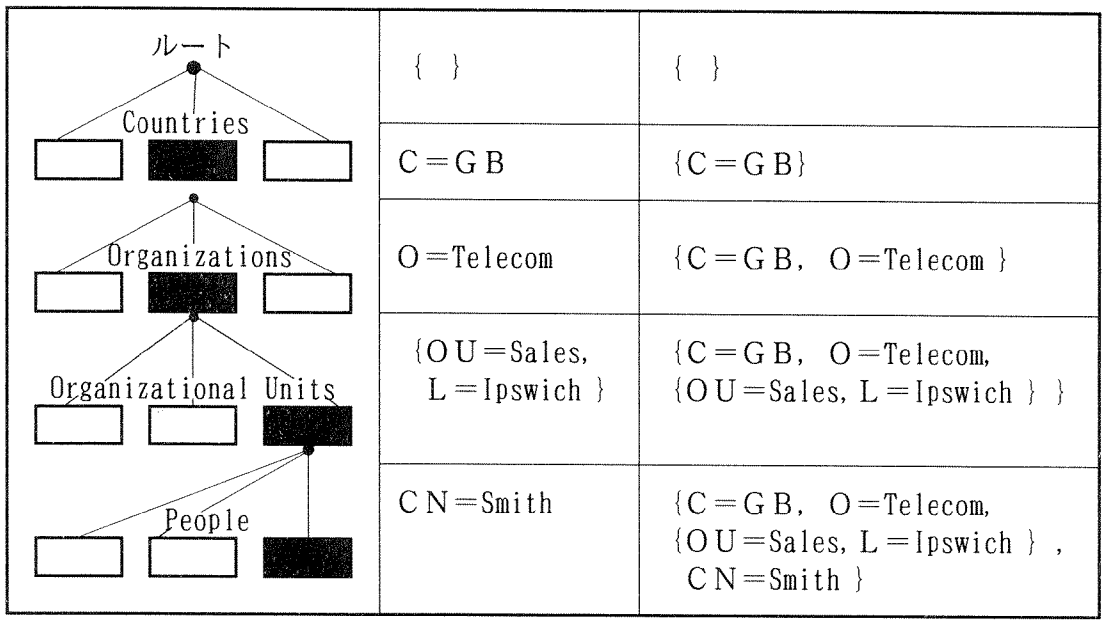


図１６－１／ＪＴ－Ｘ５００　識別名の定義
(ITU-T X.501)

１６．５　別　名

(1) オブジェクトの別名は、別名エントリの使用により提供されるオブジェクトまたはオブジェクトエントリに対する代わりの名前である。

(2) それぞれの別名エントリは、あるオブジェクトの名前を、aliasedEntryName属性の中に持っている。別名エントリの識別名は、この結果そのオブジェクトの名前ともなる。

（注）aliasedEntryNameの中の名前は、別名により指し示されていると言われている。それは、いずれかのエントリの識別名である必要はない。

(3) 別名のオブジェクト名への変換は、（別名）反対参照と呼び、該当するaliasedEntryName属性の値による別名のシステム的な置換（それは提示された名前の中に見つかる）を含んでいる。このプロセスは、一つ以上の別名エントリの調査を要求するであろう。

(4) ＤＩＴの中のある特別なエントリは、０かまたはそれ以上の別名を持つ。従って、いくつかの別名エントリが同一のエントリを指し示すこともありうる。別名エントリは末

端エントリでないエントリを指し示す。また他の別名エントリを指し示すこともある。

(5) 別名エントリは常に末端エントリであり、下位エントリを持たない。

(6) すべての別名エントリは、１７．３．３で定義されるalias オブジェクトクラスに属する。

(7) 日本語名を別名として、指定可能とする。付属資料Ｅに、日本語名の実現方法を示す。

１７　ディレクトリスキーマ

１７．１　定　義

本章で使用する以下の用語は，４．６節で定義される。

（１）属性構文

（２）ディレクトリスキーマ

（３）（ディレクトリ）サブスキーマ

（４）ＤＩＴ内容規則

（５）ＤＩＴ構造規則

（６）（エントリの）管理構造規則

（７）名前形成

（８）上位構造規則

１７．２　概要

ディレクトリスキーマは，次に関する定義と制限の集合である。すなわち、DITの構造、エントリの可能な名前付けの仕方、エントリが保持できる情報、その情報を表現するための属性、検索を容易にするための階層構造への体系、情報の修正、及び属性の値が属性値と照合規則の組において、照合させる方法である。

注）スキーマにより、ディレクトリシステムは、次のことが可能になる。

・間違ったオブジェクトクラスの下位エントリを作成しない。（例えば、”人”の下位として、”国”）

・エントリに、そのオブジェクトクラスとして、不適当な属性型を追加しない。（例えば、”人”に”通し番号”）

・属性型に定義されていない構文の属性値を追加しない。（例えば、”ビット列”に”印字可能文字列”など）

形式的に、ディレクトリスキーマは次のような集合からなる。

a) 名前形成定義：

構造オブジェクトクラスに対する基本的な名前付け関係を定義したもの。

b) DIT構造規則定義：

エントリが持っている名前とDIT内のエントリ間の関係を定義したもの。

c) DIT内容規則定義：

エントリの構造オブジェクトクラスによって、示されるエントリの属性以外で、許可され得る属性の仕様を拡張したもの。

d) オブジェクトクラス定義：

与えられたクラスにおいて、持つと予想される必須属性と持つことが許される選択属性の基本集合を定義し、17.3節のディレクトリ仕様で定義されたオブジェクトクラスの種類を示すもの。

e) 属性型定義：

属性を識別するオブジェクト識別子、属性構文、関連する照合規則、運用　　属性かどうか、集合属性かどうか、複数の値を持つことを許されているか　　どうか、属性型の継承があるかどうかを定義したもの。

f) 照合規則定義：

照合規則を定義したもの。

図17-1は、左側にスキーマとサブスキーマの定義と右側にDIT、ディレクトリエントリ、属性、属性値の関係を示したものである。

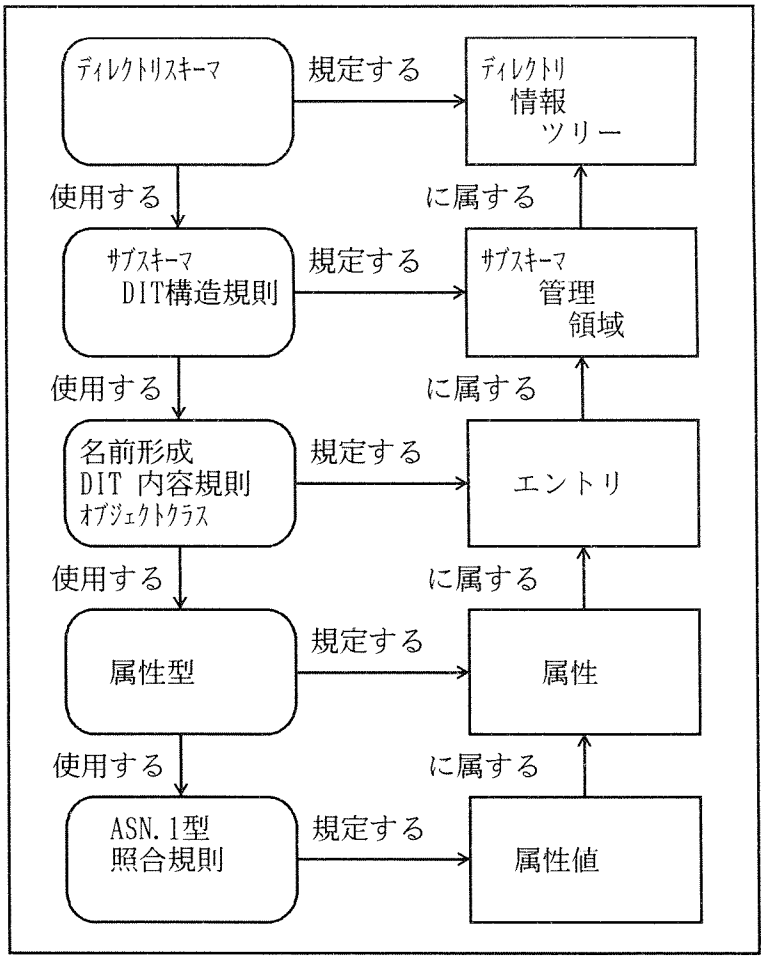


図１７－１／ＪＴ－Ｘ５００　ディレクトリスキーマの概要
（ITU-T X.501）

図17-1は、次のように解釈される。

・左側に垂直に並んだ項目は、スキーマの要素を表現している。

・右側に垂直に並んだ項目は、スキーマで定義される規則に対応するスキーマ項目の例を表現している。

・スキーマ項目間の関係は、”使用する”の関係によって示される。

・スキーマの別の側面の例の間の関係は、”規定する”の関係によって示される。

ディレクトリスキーマは、DIBが分散されるのと同様に分散されている。それは、管理権のある管理領域（サブスキーマの特定の部分）にあるエントリを各々管理する重なりのないサブスキーマの集合として明確になる。サブスキーマの管理機関は、サブスキーマを構成する規則と制限を確立する。

サブスキーマの管理機関は、ディレクトリ仕様（名前形成、オブジェクトクラス、属性

（型、照合規則））で定義されるグローバルな範囲を持つディレクトリスキーマの個々の要素を用いることを決定してよい。また、それはその環境へよりよく適応する要素を定義することを決めてよいし、または標準的かつ独自のスキーマ要素を用いて、中間的なアプローチを決めてよい。

サブスキーマの管理機関は、その範囲をサブスキーマ（DIT構造規則と内容規則）に制限するこのようなスキーマ要素を定義する。さらには、どの照合規則がどの属性型に適応するかをも詳細に規定する。

ディレクトリスキーマは、ディレクトリ利用者情報にのみ関係する。運用情報の仕様をサポートするものは、この章で定義する記法で与えられるが、ディレクトリ管理と運用情報の規定は、ディレクトリシステムスキーマの関係である。

注）ディレクトリシステムスキーマは、ITU-T勧告X.501を参照のこと。

１７．３　オブジェクトクラスの定義

オブジェクトクラスの定義は、以下のものからなる。

(a) 本クラスのスーパクラスの表示。

(b) オブジェクトクラスの種類の定義。

(c) すべてのスーパクラス及びそのオブジェクトクラスのエントリの必須属性型の表示。

(d) すべてのスーパクラス及びそのオブジェクトクラスのエントリの選択属性型の表示。

(e) オブジェクトクラスへのオブジェクト識別子の割当て。

注）集合属性は、オブジェクトクラス定義の属性型には現れない。

１７．３．１　サブクラス化

サブクラス化には次のような制限がある。

・抽象オブジェクトクラスしか他の抽象オブジェクトクラスのスーパクラスになれない。

すべての構造オブジェクトクラスをそのサブクラスに持つ特別なオブジェクトクラスがあり、それを"Top"と呼ぶ。"Top"は、抽象オブジェクトクラスである。

17．3．2　オブジェクトクラス属性

すべてのエントリは、そのエントリが属するオブジェクトクラスやスーパクラスを識別するために、属性型”objectClass”を持つ。この属性型の定義は、17.4.6節で与えられる。この属性は複数の値を持つ。

エントリの構造オブジェクトクラスとそのそれぞれのスーパクラスの値に対して、属性”objectClass”の値が存在する。

エントリの構造オブジェクトクラスは不変である。属性”objectClass”の初期値はエントリを作成する時、利用者によって与えれる。

補助オブジェクトクラスを用いるところでは、DIT内容規則によって許される補助オブジェクトクラスとそのスーパクラスに対する属性”objectClass”の値をエントリが持つ。許される補助オブジェクトクラスに対する1つの値が存在する場合、許される補助オブジェクトクラスのスーパクラスに対する複数の値も存在する。

属性”objectClass”が補助オブジェクトクラスに対するオブジェクト識別子を持つところでは、エントリは、常にそのオブジェクトクラスによって識別される必須の属性を持たなければならない。

注）1 「属性”objectClass”は全てのエントリに存在しなければならない」という必要条件は、Topの定義において反映される。

2　オブジェクトクラスがすべてのスーパクラスに属していると見なされるため、Topまでのスーパクラスのチェインのクラスは、属性”objectClass”の値（とフィルタによって一致したチェインの中のすべての値）によって表現される。

3 アクセス制御の制限は、属性”objectClass”の更新に対して、加えられる場合がある。

適用されるDIT内容規則とともに、ディレクトリは、DIBのすべてのエントリに対して定義されたオブジェクトクラスを強制する。エントリのDIT内容規則によってはっきりと許されないエントリのオブジェクトクラス定義に違反するようなエントリを変更しようとした場合は、失敗する。

注）特に、ディレクトリは、通常次のことを防止する。

a) エントリの構造オブジェクトクラスの定義が抜けていて、そのオブジェクトクラスのエントリに追加するエントリのDIT内容規則によって、許可されない属性型。

b) エントリのオブジェクトクラスに対して1つかそれ以上の必須属性がないエントリ。

c) エントリのオブジェクトクラスに対して必須属性を削除すること。

17．3．3 オブジェクトクラスの仕様

オブジェクトクラスは、次のＯＢＪＥＣＴ－ＣＬＡＳＳ情報オブジェクトクラスの値として、定義される。

```
OBJECT-CLASS              ::= CLASS {
    &Superclasses               OBJECT-CLASS OPTIONAL,
    &kind                       ObjectClassKind DEFAULT structural,
    &MandatoryAttributes        ATTRIBUTE OPTIONAL,
    &OptionalAttributes         ATTRIBUTE OPTIONAL,
    &id                         OBJECT INDENTFIER UNIQUE }
WITH SYNTAX  {
    [SUBCLASS OF                &Superclasses]
    [KIND                       &kind]
    [MUST CONTAIN               &Mandatory Attributes]
    [MAY CONTAIN                &OptionalAttributes]
    ID                          &id}
ObjectClassKind           ::= ENUMERATED  {
    abstract     (0),
    structural   (1),
    auxiliary    (2)}
```

この情報オブジェクトクラスを使用して定義されたオブジェクトクラスに対して、

(a) &superclassesは、その直接スーパクラスのオブジェクトクラスの集合である。

(b) &kindは、その種類である。

(c) &Mandatoryは、このクラスを持つエントリが持たなければならない属性の集合である。

(d) &Optionalは、このクラスを持つエントリが持つことを許されている属性の集合である。例外として、ある属性が、必須属性集合と選択属性集合にある場合、必須属性とする。

(e) &idは、それに割り当てられるオブジェクト識別子である。

明確に挙げたオブジェクトクラス（"top"と"alias"）を以下に定義する。

```
top OBJECT-CLASS        ::=     {
    KIND                abstract
    MUST CONTAIN         { objectClass  }
    ID                  id-oc-top }

alias OBJECT-CLASS      ::=     {
    SUBCLASS             { top  }
    MUST CONTAIN         { aliasedOjectName }
    ID                  id-oc-alias }
```

注）オブジェクトクラス”alias”は別名エントリのＲＤＮに対して適当な属性型を規定していない。管理機関は別名エントリのＲＤＮとして有用な属性型を定義した、”alias”のサブクラスを規定してもよい。

１７．４　属性型の定義

属性型の定義は、以下からなる。

(a) 属性型が既に定義されている属性のサブタイプであること、および、直接上位のスーパタイプの表示（しかし必ずしも表示する必要はない）。

(b) 属性型の属性構文の定義（しかし必ずしも表示する必要はない）。

(c) 属性型の照合規則（equality, ordering, substrings）の表示（しかし必ずしも表示する必要はない）。

(d) 属性値がただ一つの値をとりうるか、複数の値をとれるかの表示。

(e) 属性型が運用属性か利用者属性のいずれかの表示。

(f) 利用者属性か集合属性かの表示（しかし必ずしも表示する必要はない）。

(g) 運用属性が利用者に更新できないことの表示（しかし必ずしも表示する必要はない）。

１７．４．１　運用属性

運用属性には、利用者が直接制御できるものと、ディレクトリにより制御されるものとがある。後者の場合、利用者が属性値を更新できないことを、運用属性の定義に表示すべきである。

運用属性型の定義では、各属性型のアプリケーションを表示すべきである。ここで、アプリケーションは、以下のいずれかである。

－ディレクトリ運用属性（例、アクセス制御属性）

－ＤＳＡ共有運用属性（例、マスタ・アクセス・ポイント属性）

－ＤＳＡ専用運用属性（例、コピー状態属性）

１７．４．２　属性階層

属性階層は、利用者属性か運用属性のいずれかのみから成るべきである。すなわち、利用者属性は運用属性から導出されてはならず、運用属性は利用者属性から導出されてはいけない。

他の運用属性のサブタイプである運用属性は、スーパタイプと同じアプリケーションを持つべきである。

属性型が他の属性型のサブタイプでない場合、属性構文と照合規則（適用できれば）は属性型定義に示されるべきである。属性構文は直接ＡＳＮ．１データ型を定義する。

属性型が示された属性型のサブタイプである場合、その定義には属性構文を規定する必要はない。この場合、この属性構文は直接スーパタイプと同様である。属性構文が示されており、属性が直接上位のスーパタイプを持つ場合、示された構文はスーパタイプの構文と等しくなければならない。つまり、属性構文を満たすどの値も、スーパタイプの構文を満たさなければならない。

属性型が他の属性型のサブタイプの場合、スーパタイプに適用できる照合規則は、サブタイプの定義で拡張や変更がされていなければ、サブタイプにも適用できる。スーパタイプに定義された照合規則は、サブタイプを定義する時に取り除いてはならない。

１７．４．３　集合属性

運用属性は、集合属性として定義されてはならない。

利用者属性は、集合属性として定義してよい。つまり、”collectiveExclusions”属性の制限に従い、エントリ集合のエントリが同じ属性値をとるであろう。集合属性は複数の値をとれるべきである。

１７．４．４　属性構文

equality照合規則が属性型に対し定義されるなら、正しい属性構文がその属性型を持つ全ての属性に使用されることを、ディレクトリは保証すべきである。

１７．４．５　照合規則

equality,ordering,substrings照合規則が、属性型定義に示される（しかし必ずしも必要はない）。照合規則が、複数の異なる属性型の照合に適してよいなら、同じ照合規則が１つ以上の照合規則として用いてよい。

（注）この事は、照合規則の定義に反映されなければならない。

もしequality照合規則が示されていない場合、ディレクトリは以下を行う。

(a) ANY 型であるとして、この属性の値を取り扱う。すなわち、ディレクトリは属性構文として示されたデータ型や他の規則に、値が適合するかチェックしなくてもよい。

(b) 属性を名前付けに使用することを認めない。

(c) 複数の属性値から個々の値を追加したり削除したりを認めない。

(d) 属性値の比較を行わない。

(e) その様な属性型の値のAVAs評価を行わない。

もしequality照合規則が示されている場合、ディレクトリは以下を行う。

(a) この属性の値をその属性定義（もしくは、その属性のスーパタイプの属性定義）の&Typeフィールドで定義されたデータ型として扱う。

(b) この属性について属性値提示を評価するために、示されたequality照合規則を用いる。

(c) この属性型定義で規定された、適当なデータ型の与えられた値のみ照合する。

（注）属性型の構文とは異なる提示構文を用いるequality照合規則の属性にも等しく適用できる。

ordering照合規則が示されない場合、ディレクトリは、ディレクトリ抽象サービスによって与えられる構文を用いた、ordering照合規則の提示を未定義として扱う。

substrings照合規則が示されない場合、ディレクトリは、ディレクトリ抽象サービスによって与えられる構文を用いた、substrings照合規則の提示を未定義として扱う。

属性型の属性構文に適用した定義を、属性型の照合規則として示すのみとすべきである。

１７．４．６　属性の仕様

属性は、ATTRIBUTE情報オブジェクトクラスの値として定義される（しかし必ずしも使

用する必要はない）。

```
ATTRIBUTE          ::=  CLASS  {
    &derivation                ATTRIBUTE OPTIONAL,
    &Type                          OPTIONAL, --either &Type or &derivation required--
    &equality-match                MATCHING-RULE OPTIONAL,
    &ordering-match                MATCHING-RULE OPTIONAL,
    &substrings-match          MATCHING-RULE OPTIONAL,
    &single-valued                 BOOLEAN DEFAULE FALSE,
    &collective                    BOOLEAN DEFAULE FALSE,
    --operatinal extensions--
    &no-user-modification          BOOLEAN DEFAULE FALSE,
    &usage                         AttributeUsage DEFAULT userApplications,
    &id                            OBJECT IDENTIFIER UNIQUE}
WITH SYNTAX  {
    [SUBTYPE OF                    &supertype]
    [WITH SYNTAX                   &Type]
    [EQUALITY MATCHING RULE        &equality-match]
    [ORDERING MATCHING RULE    &ordering-match]
    [SUBSTRING MATCHING RULE   &substrings-match]
    [SINGLE VALUE                  &single-valued]
    [COLLECTIVE                    &collective]
    [NO USER MODIFICATION      &no-user-modification]
    [USAGE                         &usage]
    ID                             &id }
AttributeUsage     ::=  ENUMERATED {
    userApplications     (0),
    directoryOperation   (1),
    distributedOperation (2),
    dSAOperation         (3)}
```

この情報オブジェクトクラスを用いて定義された属性において

(a) &derivation は、もしあれば、そのスーパタイプである属性である。

(b) &Type はその属性構文である。これはＡＳＮ．１データ型であるべきある。

(c) &equality-match は、（もしあれば）そのequality照合規則である。

(d) &ordering-match は、（もしあれば）そのordering照合規則である。

(e) &substrings-match は、（もしあれば）そのsubstrings照合規則である。

(f) &single-valuedは、単数の時に真となり、複数の時に偽となる。

(g) &collective は、集合属性の時に真となり、集合属性でない時に偽である。

(h) &no-user-modification は、利用者による更新ができない運用属性の場合、真となる。

(i) &usageは属性の運用上の利用を示す。userApplicationsは利用者属性であることを示し、directoryOperation, distributedOperation, dSAOperationは、各々ディレクトリ運用属性、分散運用属性、ＤＳＡ運用属性であることを示す。

(j) &id は割り当てられたオブジェクト識別子である。

８８年版のディレクトリ仕様で定義された、ディレクトリで利用される属性を、以下に定義する。

```
objectClass ATTRIBUTE ::=  {
        WITH SYNTAX                 OBJECT IDENTIFIER
        EQUALITY MATCHING RULE      objectIdentifierMatch
        ID                          id-at-objectClass }

aliasedObjectName ATTRIBUTE ::=  {
        WITH SYNTAX                 DistinguishedName
        EQUALITY MATCHING RULE      distinguishedNameMatch
        SINGLE VALUE                TRUE
        ID                          id-at-aliasedobjectName }
```

（注）上記の定義で参照している照合規則は、17.5.1節で定義される。

１７．５　照合規則定義

照合規則の定義は以下からなる。

(a) 照合規則の提示構文の定義。

(b) 規則によりサポートされた異なるタイプの照合の規定。

(c) ＤＩＢの中に持たれている目的となる属性値に関する与えられた提示の評価に関する適切な規則の定義。

(d) 照合規則へのオブジェクト識別子の割当て。

照合規則は照合規則がequality照合規則である属性の属性値提示の評価のために使用されるべきである。属性値提示の中で使用される構文（すなわち、属性値提示の中の属性値構成要素）は照合規則の提示構文である。

照合規則は異なる属性構文を持った多くの異なる型の属性に適用できる。

照合規則の定義には照合規則の提示構文の規定と、照合を行うためにこの構文の値が用いられる方法が含まれている。この定義は照合規則が適用されるであろう属性構文の全ての仕様までは要求しない。異なるＡＳＮ．１構文を持った属性とともに用いられる照合規則の定義は、照合がいかに行われるかを規定する。

サブスキーマの仕様の中に含まれる、属性に対して定義された照合規則（そのほかにこれらの属性型の定義の中で使用される照合規則）の適用性は、サブスキーマ仕様を通して

ＩＴＵ－Ｔ勧告Ｘ．５０１で定義される運用属性matchingRuleUseで示される。

１７．５．１　照合規則の仕様

照合規則は、ＭＡＴＣＨＩＮＧ－ＲＵＬＥ情報オブジェクトクラスの値として定義される。

```
MATCHING RULE  ::= CLASS {
     &AssertionType          OPTIONAL
     &id                     OBJECT IDENTIFIER UNIQUE}
WITH SYNTAX       {
     [SYNTAX                 &AssertionType]
     ID                      &id }
```

この情報オブジェクトクラスを用いて定義される照合規則において

(a) &AssertionType はこの照合規則を用いた提示の為の構文である。これが省略された場合、提示構文は規則が適用される属性と同じ構文である。

(b) &id は割り当てられたオブジェクト識別子である。

ObjectIdentifierMatch照合規則は以下のように定義される。

```
objectIdentifierMatch MATCHING-RULE  ::= {
      SYNTAX         OBJECT IDENTIFIER
      ID             id-mr-objectIdentifierMatch }
```

与えられた型のオブジェクト識別子の値と目的の型のオブジェクト識別子の値は、両者が同じ数の必要な要素を持ち、かつ前者の必要な各要素が後者の各要素に一致する時、またその時のみ一致する。この照合規則はＡＳＮ．１型オブジェクト識別子の定義の中で固有なものである。ObjectIdentifierMatchはequality照合規則である。

DistinguishedNameMatchは以下のように定義される。

```
distinguishedNameMatch MATCHING-RULE  ::= {
      SYNTAX         DistinguishedName
      ID             id-mr-distinguishedNameMatch}
```

以下の全てが真の時に、またその時のみ、与えられた識別名の値は目的の識別名と一致する。

(a) ＲＤＮの数が同じである。

(b) 対応するＲＤＮが同じ数のＡＶＡを持つ。

(c) 対応するＡＶＡ（つまり同一属性型を持ち、対応するＲＤＮのＡＶＡ）は同値で一致する属性を持つ。（この照合において、属性値は同じ役割をする。すなわち、与えらた値と目的の値が識別名であり、全体が一致する。）DistinguishedNameMatchはequality照合規則である。

１７．６　ＤＩＴ構造定義

１７．６．１　概要

ディレクトリスキーマの基本的な側面は、ある特定のクラスのエントリがＤＩＴ上のどこに配置されているかということと、いかに名前付けられるべきかを規定することである。

－ＤＩＴの中のエントリの階層的関係（ＤＩＴ構造規則）

－エントリのＲＤＮを形成するために使用される属性または属性群（名前形成）

１７．６．２　名前形成の定義

名前形成の定義は以下からなる。

(a) 名前付けらてたオブジェクトクラスの規定

(b) この名前形成が適用されるオブジェクトクラスのエントリのためのＲＤＮとして使用されるべき必須属性の表示。

(c) この名前形成が適用されるオブジェクトクラスのエントリのためのＲＤＮとして使用してもよい選択属性（もしあれば）の表示。

(d) 名前形成に対するオブジェクト識別子の割り当て。

名前属性の異なる集合が与えられた構造オブジェクトクラスのエントリの為に必要とされる場合、名前形成は名前付けの為に使用されるべき各々の異なる属性集合に対して定義されなけらばならない。

構造オブジェクトクラスのみが名前形成に使用される。

ある特定の構造オブジェクトクラスのエントリがＤＩＢの一部に存在する場合、少なくとも１つの構造オブジェクトクラスに対する名前形成がスキーマの適切な部分に含まれるべきである。スキーマは必要であれば付加的な名前形成を含む。

ＲＤＮ属性（または属性群）は、それ自身の構造または別名オブジェクトクラス定義の中で規定する構造オブジェクトクラスの許可された属性のリストから選択される必要はない。

（注）名前属性は他の属性と同じようにＤＩＴ内容規則により管理される。

名前形成は、ＤＩＴを形成するするために必要な全体仕様の中の単なる基本要素である。その仕様とは与えられたＤＩＴの領域の名前付け方針を決定する管理機関または名前付け機関によって要求される。ＤＩＴ構造の仕様に関して残された点については17.6.5節で記述する。

１７．６．３　名前形成の仕様

名前形成はＮＡＭＥ－ＦＯＲＭ情報オブジェクトクラスの値として定義される。

```
NAME-FORM                  ::=  CLASS {
    &nameObjectClass            OBJECT-CLASS,
    &MandatoryAttributes        ATTRIBUTE,
    &OptionalAttributes         ATTRIBUTE OPTIONAL,
    &id                         OBJECT IDENTIFIER UNIQUE }
WITH SYNTAX  {
    NAMES                       &namedObjectClass
    WITH ATTRIBUTES &MandatoryAttributes
    [AND OPTIONALLY             &OptionalAttributes]
    ID                          &id}
```

この情報オブジェクトクラスを用いて定義される名前形成において

(a) &namedObjectClass は構造オブジェクトクラスの名前である。

(b) &MandatoryAttributesはエントリのＲＤＮに存在しなければならない属性の集合である。

(c) &OptionalAttributes はエントリのＲＤＮに存在してもよい属性の集合である。

(d) &id は割り当てられたオブジェクト識別子である。

必須または選択リストにある全ての属性は異なっているべきである。

１７．６．４　エントリの構造オブジェクトクラス

サブスキーマ仕様の中には、サブスキーマの中に現れる一つの構造オブジェクトクラスのスーパクラス連鎖当たりただ一つの構造オブジェクトクラスの名前形成を含むものもある。

サブスキーマ仕様の中には、サブスキーマの中に現れる一つの構造オブジェクトクラスのスーパクラス連鎖当たり一つ以上の構造オブジェクトクラスの名前形成を含んでもよいものもある。

いずれの場合でも、特定のエントリに関して、エントリの”objectClass”属性に現れる構造オブジェクトクラスのスーパクラス連鎖の最下位にある構造オブジェクトクラスのみがエントリに適用されるＤＩＴ内容規則やＤＩＴ構造規則を決定する。このオブジェクトクラスはエントリの構造オブジェクトクラスとして参照されるとともに、”structualObjectClass”運用属性によって示される。

１７．６．５　ＤＩＴ構造規則の定義

ＤＩＴ構造規則は、サブスキーマの適用範囲内のエントリの名前付けと配置を制御するためにディレクトリが使用する仕様であり、この仕様はサブスキーマ管理機関より提供される。各オブジェクトおよび別名エントリは単一のＤＩＴ構造規則に従う。ＤＩＴのサブツリーを管理するサブスキーマは常にサブツリーの中にいくつかの型のエントリが存在することを認めたいくつかのＤＩＴ構造規則を含むであろう。

ＤＩＴ構造規則の定義は以下からなる。

(a)　サブスキーマの適用範囲内でユニークな整数識別子。

(b)　ＤＩＴ構造規則により支配されるエントリの為の名前形成の表示。

(c)　許可される上位構造規則の集合（もし必要ならば）。

サブスキーマの為のＤＩＴ構造規則の集合はサブスキーマにより管理されるエントリの識別名の形式を規定する。

ＤＩＴ構造規則は、あるサブスキーマの中のエントリに、特定の名前形成に属することを許可する。エントリの識別名における最後のＲＤＮの形式はエントリを管理するＤＩＴ

構造規則の名前形成により決定する。

名前形成（のオブジェクトクラス）におけるnamedObjectClassはエントリの構造オブジェクトクラスに一致する。

ＤＩＴ構造規則はエントリに関係する名前形成により識別される構造オブジェクトクラスに属するエントリのみ許容すべきである。構造オブジェクトクラスのサブクラスに属するエントリには許容しない。

ある特定のエントリに関して、エントリを管理するＤＩＴ構造規則はエントリの支配構造規則と名付けられ、この規則はエントリの”governingStructureRule”属性を調べれば分かる。

ある特定のエントリに関して、エントリの上位を管理しているＤＩＴ構造規則は上位構造規則と名付けられる。

ＤＩＴ構造規則が、以下のように支配しているサブスキーマに存在した場合、エントリはＤＩＴ内の他の（上位の）エントリの下位としてのみ存在してよい。

－エントリの構造オブジェクトクラスに対する名前形成を提示し、かつ

－可能な上位構造規則としてエントリの上位構造規則を含むか、あるいは上位構造規則を規定しないかであり、この場合ではエントリはサブスキーマの保守管理ポイントでなければならない。

１７．６．６　ＤＩＴ構造規則の仕様

ＤＩＴ構造規則の抽象構文配下のＡＳＮ．１型で表現される。

```
DITStructureRule    ::=   SEQUENCE  {
     ruleIdentifier           RuleIdentifier,
                              --must be uniqe within the scope of the subschema
     nameForm                 NAME-FORM.&id,
     superiorStructureRules   SET OF RuleIdentifier OPTIONAL}
RuleIdentifier      ::=   INTEGER
```

17.6.5節で挙げた定義と上で定義したＡＳＮ．１型の様々な要素との関連性は以下の通りである。

(a) ruleIdentifierはサブスキーマ内でＤＩＴ構造規則をユニークに識別する。

(b) ＤＩＴ構造規則のnameFormはＤＩＴ構造規則により管理されるエントリに関する名前形成を規定する。

(c) superiorStructureRules はこの規則により管理されるエントリに対して、許される上位構造規則を特定する。省略された場合、このＤＩＴ構造規則は独立した管理ポイントに適用される。

ＳＴＲＵＣＴＵＲＥ－ＲＵＬＥ情報オブジェクトクラスはＤＩＴ構造規則の裏付けを促進するために提供される。

```
STRUCTURE-RULE        ::=   CLASS {
    &nameForm                       NAME-FORM,
    &SuperiorStructureRules         STRUCTURE-RULE OPTIONAL,
    &id                             RuleIdentifier UNIQUE  }
WITH SYNTAX  {
     [NAME FORM                       &nameForm]
     [SUPERIOR RULES                  &SuperiorStructureRules]
     ID                               &ruleIdentifier  }
```

１７．７　ＤＩＴ内容規則の定義

ＤＩＴ内容規則は、選択された補助オブジェクトクラスの集合、必須属性、選択属性、および除外された属性を特定することにより、特定の構造オブジェクトクラスのエントリに許容された内容を規定する。集合属性がそのエントリに許された場合、それをＤＩＴ内容規則に含めるべきである。

ＤＩＴ内容規則は以下からなる。

(a) この規則が適用される構造オブジェクトクラスの表示。

(b) この規則に管理されるエントリに対して許可される補助オブジェクトクラスの表示（しかし必ずしも必要ではない）。

(c) ＤＩＴ内容規則により管理されるエントリに対して必要とされる必須属性、さらに構造及び補助オブジェクトクラスによって必要とされる必須属性の表示（しかし必ずしも必要ではない）。

(d) ＤＩＴ内容規則により管理されるエントリに対して許可される選択属性、さらに構造及び補助オブジェクトクラスによって許可される選択属性の表示（しかし必ずしも必要ではない）。

(e) この規則に管理されたエントリから除外される、エントリの構造オブジェクトクラス、及び補助オブジェクトクラスの選択属性の表示（しかし必ずしも必要ではない）。

ある有効なサブスキーマ仕様に関して、各オブジェクトクラスには多くても一つのＤＩＴ内容規則が存在する。

ＤＩＴ内の各エントリは多くても一つのＤＩＴ内容規則により管理される。この規則はエントリの”structuralObjectClass”属性の値を調べることにより特定してよい。

構造オブジェクトクラスに関するＤＩＴ内容規則が存在しない場合、そのクラスのエントリは構造オブジェクトクラス定義により許容された属性のみ含むべきである。

エントリに関する構造オブジェクトクラスのスーパクラスのＤＩＴ内容規則は、そのエントリには適用されない。

ＤＩＴ内容規則は構造オブジェクトクラスと関係しているため、同じ構造オブジェクトクラスを持つ全てのエントリは、ＤＩＴの配置を支配するＤＩＴ構造規則とは無関係に、同じＤＩＴ内容規則を持つであろう。

ＤＩＴ内容規則に支配されるエントリは、ＤＩＴ構造規則の構造オブジェクトクラスに加えて、ＤＩＴ内容規則により特定される補助オブジェクトクラスの部分集合にも関係してよい。この関係はエントリの”objectClass”属性に影響する。

エントリの内容は、以下のようにその”objectClass”属性により示されるオブジェクトクラスと一貫性を持たなければならない。

－”objectClass”属性により示されたオブジェクトクラスの必須属性は常にエントリに存在すべきである。

－ＤＩＴ内容規則により示される補助オブジェクトクラスの（ＤＩＴ内容規則で付加的に選択または必須として示されない）選択属性は、”objectClass”属性がこれらの補助オブジェクトクラス属性を含む場合のみ、存在してよい。

構造または示された補助オブジェクトクラスと関係する必須属性はＤＩＴ内容規則で除外されるべきでない。

１７．７．１　ＤＩＴ内容規則の仕様

ＤＩＴ内容規則の抽象構文は以下のＡＳＮ．１型で表現される。

```
DITContentRule      ::=    SEQUENCE    {
     structuralObjectClass      OBJECT-CLASS.&ID,
     auxiliaries                SET OF OBJECT-CLASS.&id OPTIONAL,
```

```
    mandatory          [1]     SET OF ATTRIBUTE.&id OPTIONAL,
    optional           [2]     SET OF ATTRIBUTE.&id OPTIONAL,
    precluded          [3]     SET OF ATTRIBUTE.&id OPTIONAL  }
```

17.8節で挙げた定義と、上で定義したＡＳＮ．１型の様々な部分との関連性は以下の通りである。

(a) structualObjectClass はＤＩＴ内容規則が適用される構造オブジェクトクラスを特定する。

(b) auxiliaries はＤＩＴ内容規則が適用されるエントリに対して、許される補助オブジェクトクラスを特定する。

(c) mandatory はその構造及び補助オブジェクトクラスに従って含まなければならない属性に加えて、ＤＩＴ内容規則が適用されるエントリが含まなければならない利用者属性型を規定する。

(d) optional はその構造及び補助オブジェクトクラスに従って含んでもよい属性に加えて、ＤＩＴ内容規則が適用されるエントリが含んでもよい利用者属性型を規定する。

(e) precluded はＤＩＴ内容規則が適用されるエントリから除外される構造及び補助オブジェクトクラスの選択利用者属性を規定する。

ＣＯＮＴＥＮＴ－ＲＵＬＥ情報オブジェクトクラスはＤＩＴ内容規則の裏付けを促進するために提供される。

```
CONTENT-RULE      ::=  CLASS  {
    &structuralClass           OBJECT-CLASS.&id UNIQUE,
    &Auxiliaries               OBJECT-CLASS OPTIONAL,
    &Mandatory                 ATTRIBUTE OPTIONAL,
    &Optional                  ATTRIBUTE OPTIONAL,
    &Precluded                 ATTRIBUTE OPTIONAL }
WITH SYNTAX  {
    STRUCTURAL OBJECT CLASS    &structuralClass
    [AUXILIARY OBJECT CLASSES  &Auxiliaries]
    [MUST CONTAIN              &Mandatory]
    [MAY CONTAIN               &Optional]
    [MUST NOT CONTAIN          &Precluded]}
```

１８．１　用語

このディレクトリ仕様は、ITU-T X.200 | ISO/IEC 7498-2に定義してある用語を使用する。

－　アクセス制御
－　認証
－　セキュリティ方針

以下の用語は、このディレクトリ仕様に定義してある。

－　アクセス制御機構：　ディレクトリ情報へのアクセスの手段と潜在的にそれらのアクセス権が制御されてもよい。
－　保護項目：　アクセスが分割制御されるディレクトリ情報の要素。ディレクトリの保護項目は、エントリ、属性、属性値及び名前のいずれでもよい。

１８．２　セキュリティ方針

各管理機関の管理部分のＤＩＢへのアクセスを制御する環境にディレクトリは存在する。このようなアクセスは一般的に、セキュリティ方針を制御したアドミニストレーションに適合する。（ITU-T勧告 X.509 | ISO/IEC9594-8参照）

ディレクトリへのアクセスに影響するセキュリティ方針の二つの側面は、認証手続きとアクセス制御機構である。

１８．２．１　認証手続きと機構

ディレクトリにおいて認証の手続き及び機構は、必要な所で以下の２つを実証し伝達する方法を含んでいる。

－　ＤＳＡとディレクトリユーザの識別

－　アクセス点で受信した情報の発信者の識別

（注）管理機関は、管理ユーザの認証規定と非管理ユーザの認証規定を比較して違いを明記してもよい。

認証手続きの一般的な用法は、ITU-T勧告X.509 | ISO/IEC 9594-8で定義してある。そしてセキュリティ方針を強化する今回の修正で定義してあるアクセス制御機構と共に使用される。

（注）

1　将来のディレクトリ仕様の編集で他のアクセス制御機構を定義してもよい。

2　ローカルな管理方針は、ある他のＤＳＡ（すなわち、他のＤＭＤ内のＤＳＡ）で実施している認証が無視されることを明記してもよい。

一般的に、認証ＩＤ（すなわち、認証交換による認証としての人間であるユーザＩＤ）からアクセス制御ＩＤ（すなわち、ユーザを表現する付加的でユニークなユーザＩＤとともに識別するエントリの識別名）へ照合する機能があるだろう。この照合は、ディレクトリ仕様の範囲内に含まれない。しかしながら、特定のセキュリティ方針は、認証ＩＤとアクセス制御ＩＤが同じであることを述べてもよい。

１８．２．２　アクセス制御機構

ディレクトリでのアクセス制御機構の定義は、以下の方法を含む。

－　アクセス制御情報の指定

－　アクセス制御情報に定義された強化されたアクセス権

－　アクセス制御情報の保守

強化されたアクセス権は、以下のアクセス制御を適用する。

－　名前に関連したディレクトリ情報
－　ディレクトリユーザ情報
－　アクセス制御情報を含むディレクトリ運用情報

管理機関は、セキュリティ方針に実装される標準的なアクセス制御機構の全てまたは一部を利用するだろう。また、管理機関の判断で独自の機構を自由に定義してもよい。

しかしながら、管理機関はすべてあるいはいくつかのディレクトリ運用情報の保護のために分割した規定を明記してもよい。管理機関は通常のユーザに運用情報の保護の規定を検出する手段を提供することを要求されない。

（注）管理方針は、他の方法で適用するアクセス制御に関わらず特定の属性（すなわち、運用属性）へのいかなる形式のアクセスを認めても否定してもよい。

ＡＣＳＡ内のサブエントリ又はエントリは、もしこのようなＡＣＩが認められ、ＡＣＳＡに一致するaccessControlScheme属性値と一致すれば、ＡＣＩエントリを含むことは許可される。

ディレクトリはアクセス制御機構にaccessControlScheme運用属性の使用を通して識別されるＤＩＢの特定部分内に有効な手段を提供する。このような機構の範囲は、アクセス制御特定領域によって定義される。そしてそれは対応するセキュリティ管理機関に対する責任がある特定管理領域である。この属性は、管理点に相当する管理エントリ内にある。

アクセス制御指定点の管理エントリのみ、accessControlScheme属性を含むことが許される。

（注）もしこの運用属性が、あるエントリへのアクセスに関し存在しない場合、ディレクトリは、１９８８年版ＤＳＡに対するものとして振る舞わなければならない。（すなわち、アクセス制御の機構やその運用上の影響、結果、エラーの決定はロー

カルな問題である。）

```
accessControlScheme ATTRIBUTE :: =      {
      WITH SYNTAX              OBJECT IDENTIFIER
      EQUALITY MATCHING RULE   oidEqualityMatch
      SINGLE VALUED            TRUE
      USAGE                    directoryOperation
      ID                       id-aca-accessControlScheme  }
```

１９．ＭＨＳでの名前付け

この章ではユーザや配布リスト（ＤＬ）が一般にＭＨＳ、特にメッセージ転送においてどのように名前付けられるかについて規定している。さらに「Ｏ／Ｒ名」を定義し、ディレクトリ名の位置づけについて記述している。ディレクトリを使用せずにメッセージ発信や打診を行う場合、ユーザエージェント（ＵＡ）やメッセージ格納（ＭＳ）はメッセージ転送システム（ＭＴＳ）における受信者を識別する。一方、ＭＴＳはメッセージを配送する際、各々の受信側ＵＡやＭＳに対して発信者を通知する。

「Ｏ／Ｒ名」はこのような識別が可能なデータ構造を持つ。

１９．１　ディレクトリ名

ディレクトリ名はディレクトリの中で一つのオブジェクトの識別に使用されるＯ／Ｒ名の一構成要素である。この様な名前をディレクトリに対して示すことによりＭＨＳはユーザやＤＬに関するディレクトリエントリへアクセスすることができる。このエントリからＭＴＳはユーザやＤＬの「Ｏ／Ｒアドレス」等を得ることができる。

しかしながら、ユーザやＤＬはディレクトリに必ずしも登録されていない。

したがってすべてのユーザやＤＬがディレクトリ名を持つわけではない。

（注１）完全に相互接続された分散ディレクトリが運用される以前に、命名機関はユーザとＤＬにディレクトリ名を割当てても良い。

（注２）「Ｏ／Ｒアドレス」が基本的にＭＨＳの組織的もしくは物理的な構造に基づいて決められるのに対し、ディレクトリ名は必ずしもそうでなくともよい。したがって、一般にディレクトリ名は「Ｏ／Ｒアドレス」よりユーザに親しみやすく、変更が少ない。それ故ディレクトリ名はＭＴＳの範囲外でユーザとＤＬを

識別する基本的な手段となり、一方、「O／Rアドレス」は使用範囲をMTS内に限定することを意図している。

19．2　O／R名

ユーザやDLは一個以上の「O／R名」を持つ。このO／R名は発信者として指示されるユーザ、或いはメッセージや打診の受信者として指定されるユーザやDLに対する識別子である。O／R名はユーザやDLを区別し、また、MHSへのアクセス場所を識別することができる。

O／R名はディレクトリ名、「O／Rアドレス」のいずれか一方または両方から構成される。このディレクトリ名は有効であれば、1個以上の名前を持つユーザやDLを一意に識別可能であり、「O／Rアドレス」が存在する場合も同等以上に識別可能である。発信時、メッセージや打診の発信者であるUAやMSは個々のO／R名のいずれか一方または両方の要素をMTSに送信する。

もし「O／Rアドレス」がない場合、MTSはディレクトリにアクセスし、ディレクトリ名からO／Rアドレスを得る。ディレクトリ名がない場合、MTSは「O／Rアドレス」を使う。両方ある場合、MTSはまず「O／Rアドレス」を用いる。「O／Rアドレス」が廃棄されて無効ならば、「O／Rアドレス」が省略されている場合と同時にディレクトリ名を用いる。

配送時、MTSは個々のO／R名に「O／Rアドレス」と可能ならば、ディレクトリ名まで含めてメッセージ受信者あるいは通知関連メッセージもしくは打診の発信者へ渡す。ディレクトリ名は発信者が供給したときまたは、DLのメンバを展開しディレクトリ名が存在したときに通知される。

（注）メッセージ回送時或いはDL展開時、MTSはメッセージ配送に際し、UAやMSに発信者が供給していないO／R名を通知することがある。

20．MHSでのディレクトリの利用

20．1　概　要

本節では、ディレクトリが存在する場合の、MHSによるディレクトリのいくつかの利用例を記述する。ディレクトリが存在しない場合に、MHSが同一機能を実現する方法は、

ローカルな問題である。

本節では、以下について記述する。

(1) 認　証

(2) 名前の解読

(3) 配布リストの展開

(4) 機能の実装状況

２０．２　認　証

機能オブジェクト（ＭＴＡ／ＵＡ等）は、ディレクトリに格納された情報を使用して認証を行うことができる。

２０．３　名前の解読

機能オブジェクトは、ディレクトリを利用して、名前の解読を行うことができる。

ユーザまたは配布リスト（ＤＬ）のＯ／Ｒアドレスを得るために、機能オブジェクトはディレクトリにユーザまたはＤＬのディレクトリ名を提示し、そのオブジェクトのディレクトリエントリから以下の属性を要求する。

(1) ＭＨＳ　Ｏ／Ｒアドレス

(2) 優先配送方法

機能オブジェクトは、これを行うためにディレクトリに対し、自身の認証を行うこと、要求する情報へのアクセス権を持っていることが必要である。

２０．４　配布リスト展開

機能オブジェクトは、配布リストの発信許可が有ることを示した上でディレクトリを使用して、配布リストの展開を行うことができる。

そのディレクトリ名を処理することを目的として、ＤＬのメンバを得るために、機能オブジェクトはディレクトリにＤＬのディレクトリ名を提示し、その機能オブジェクトのディレクトリエントリの以下の属性を要求する。

(1) ＭＨＳ　ＤＬメンバ

(2) ＭＨＳ　ＤＬ発信許可

(3) 優先配送方法

機能オブジェクトは、これを行うためにディレクトリに対し、自身の認証を行うこと、要求する情報へのアクセス権を持っていることが必要である。

２０．５　機能の実装状況

機能オブジェクトは、ディレクトリを使用してユーザまたはＭＳの機能の実装状況を知ることができる。

以下のディレクトリ属性が、ＭＨＳにおけるユーザの重要な機能を表現する。

(1) ＭＨＳ　配布可能コンテント長

(2) ＭＨＳ　配布可能コンテントタイプ

(3) ＭＨＳ　配布可能ＥＩＴ

(4) 優先配送方法

以下のディレクトリ属性が、ＭＨＳにおけるＭＳの重要な機能を表現する。

(1) ＭＨＳ　実装自動動作

(2) ＭＨＳ　実装コンテントタイプ

(3) ＭＨＳ　実装オプション属性

ユーザまたはＭＳのある一つの機能の実装状況を知るために、機能オブジェクトはディレクトリにユーザまたはＭＳのディレクトリ名を提示し、機能オブジェクトのディレクトリエントリから機能に関連する属性を要求する。

機能オブジェクトは、これを行うためにディレクトリに対し、自身の認証を行うこと、要求する情報へのアクセス権を持っていることが必要である。

付属資料Ａ－オブジェクト識別子の用法

（ＪＴ－Ｘ５００に対する）

この付属資料は、本標準で割り当てられた全てのオブジェクト識別子（ただし、ＭＨＳに関するものは除く）が属する“オブジェクト識別子のサブツリー”の開始点を提供するものである。ここでは、下記のＡＳＮ．１モジュール“UsefulDefinitions"を用いて、その開始点を提供する。本モジュールにおいて、オブジェクト識別子のサブツリーにおける全ての非末端ノードに名前が割り当てられている。

```
UsefulDefinitions {joint-iso-ccitt ds(5) modules(1) usefulDefinitions(0) 2 }
DEFINITIONS ::=
-- EXPORTS ALL --
```

このモジュールで定義されている型と値は、ディレクトリ仕様の範囲内の他のＡＳＮ．１モジュールや、ディレクトリサービスにアクセスする他のアプリケーションに使用される。他のアプリケーションは独自の目的のため、このモジュールで定義された型と値を使用することができるが、このことによってディレクトリサービスの保守や改良に必要な機能拡張や修正に制限を与えることはない。

```
ID          ::=  OBJECT IDENTIFIER
ds          ::=  {joint-iso-ccitt ds(5)}


  --情報オブジェクトの種類 --

module                          ID   ::=   {ds 1}
serviceElement                  ID   ::=   {ds 2}
applicationContext              ID   ::=   {ds 3}
attributeType                   ID   ::=   {ds 4}
attirbuteSyntax                 ID   ::=   {ds 5}
objectClass                     ID   ::=   {ds 6}
--attributeSet                  ID   ::=   {ds 7}
algorithm                       ID   ::=   {ds 8}
--abstractSyntax                ID   ::=   {ds 9}
--object                        ID   ::=   {ds 10}
--port                          ID   ::=   {ds 11}
dsaOperationalAttribute         ID   ::=   {ds 12}
matchingRule                    ID   ::=   {ds 13}
knowledgeMatchingRule           ID   ::=   {ds 14}
nameForm                        ID   ::=   {ds 15}
group                           ID   ::=   {ds 16}
subentry                        ID   ::=   {ds 17}
operationalAttributeType        ID   ::=   {ds 18}
operationalBinding              ID   ::=   {ds 19}
schemaObjectClass               ID   ::=   {ds 20}
```

```
schemaOperationalAttribute      ID   ::=   {ds 21}
administrativeRoles             ID   ::=   {ds 23}
accessControlAttribute          ID   ::=   {ds 24}
rosObject                       ID   ::=   {ds 25}
contract                        ID   ::=   {ds 26}
package                         ID   ::=   {ds 27}
accessControlSchemes            ID   ::=   {ds 28}
```

--モジュール --

```
usefulDefinitions               ID   ::=   {module usefulDefinitions(0) 2 }
informationFramework            ID   ::=   {module informationFramework(1) 2}
directoryAbstractService        ID   ::=   {module directoryAbstractService(2) 2}
distributedOperations           ID   ::=   {module distributedOperations(3) 2 }
protocolObjectIdentifiers       ID   ::=   {module protocolObjectIdentifiers(4)  2}
selectedAttributeTypes          ID   ::=   {module selectedAttributeTypes(5) 2}
selectedObjectClasses           ID   ::=   {module selectedObjectClasses(6) 2 }
authenticationFramework         ID   ::=   {module authenticationFramework(7) 2 }
algorithmObjectIdentifiers      ID   ::=   {module algorithmObjectIdentifiers(8) 2}
directoryObjectIdentifiers      ID   ::=   {module directoryObjectIdentifiers(9) 2}
upperBounds                     ID   ::=   {module upperBounds(10) 2}
dap                             ID   ::=   {module dap(11) 2}
dsp                             ID   ::=   {module dsp(12) 2}
distributeDirectoryOIDs         ID   ::=   {module distributeDirectoryOIDs(13) 2}
directoryShadowOIDs             ID   ::=   {module directoryShadowOIDs(14) 2}
directoryShadowAbstractService  ID   ::=   {module directoryShadowAbstractService (15) 2
disp                            ID   ::=   {module disp (16) 2}
dop                             ID   ::=   {module dop(17) 2}
opBindingManagement             ID   ::=   {module opBindingManagement(18) 2}
opBindingOIDs                   ID   ::=   {module opBindingOIDs(19) 2}
hierarchicalOperationalBindings ID   ::=   {module hierarchicalOperationalBindings(20) 2

dsaOperationalAttributeTypes    ID   ::=   {module dsaOperationalAttributeTypes (22) 2}
schemaAdministration            ID   ::=   {module schemaAdministration (23) 2}
basisAccessControl              ID   ::=   {module basisAccessControl (24) 2}
operationalBindingOIDs          ID   ::=   {module operationalBindingOIDs (25) 2}
```

-- 同義語 --

```
id-oc                           ID   ::=   objectClass
id-at                           ID   ::=   attributeType
id-mr                           ID   ::=   matchingRule
id-nf                           ID   ::=   nameForm
id-sc                           ID   ::=   subentry
id-oa                           ID   ::=   operationalAttributeType
id-ob                           ID   ::=   operationalBinding
id-doa                          ID   ::=   dsaOperationalAttribute
id-kmr                          ID   ::=   knowledgeMatchingRule
id-soc                          ID   ::=   schemaObjectClass
id-soa                          ID   ::=   schemaOperationalAttribute
id-ar                           ID   ::=   administrativeRoles
id-aca                          ID   ::=   accessControlAttribute
id-ac                           ID   ::=   applicationContext
id-rosObject                    ID   ::=   rosObject
id-contract                     ID   ::=   contract
id-package                      ID   ::=   package
id-acScheme                     ID   ::=   accessControlSchemes
```

```
  --旧モジュールの識別子 --

--  usefulDefinitions                ID   ::=     {module 0}
--  informationFramework             ID   ::=     {module 1}
--  directoryAbstractService         ID   ::=     {module 2}
--  distributedOperations            ID   ::=     {module 3}
--  protocolObjectIdentifiers        ID   ::=     {module 4}
--  selectedAttributeTypes           ID   ::=     {module 5}
--  selectedObjectClasses            ID   ::=     {module 6}
--  authenticationFramework          ID   ::=     {module 7}
--  algorithmObjectIdentifiers       ID   ::=     {module 8}
--  directoryObjectIdentifiers       ID   ::=     {module 9}
--  upperBounds                      ID   ::=     {module 10 }
--  dap                              ID   ::=     {module 11 }
--  dsp                              ID   ::=     {module 12 }
--  distributedDirectoryObjectIdentifiers
                                     ID   ::=     {module 13 }


  --未使用モジュールの識別子 --

--  directoryShadowOIDs              ID   ::=     {module 14 }
--  directoryShadowAbstractService   ID   ::=     {module 15 }
--  disp                             ID   ::=     {module 16 }
--  dop                              ID   ::=     {module 17 }
--  opBindingManagement              ID   ::=     {module 18 }
--  opBindingOIDs                    ID   ::=     {module 19 }
--  hierarchicalOperationalBindings  ID   ::=     {module 20 }

--  dsaOperationalAttributeTypes     ID   ::=     {module 22 }
--  schemaAdministration             ID   ::=     {module 23 }
--  basicAccessControl               ID   ::=     {module 24 }
--  operationalBindingOIDs           ID   ::=     {module 25 }

END
```

付属資料Ｂ－ＡＳＮ．１の情報枠組み
（ＪＴ－Ｘ５００に対する）

この付属資料は、本標準に含まれるＡＳＮ．１の型、値、マクロ定義の要約を提供する。この定義はＡＳＮ．１のモジュール“InformationFramework" を形成する。

```
InformationFramework {joint-iso-ccitt ds(5) modules(1) informationFramework(1) 2}
DEFINITIONS ::=
-- EXPORTS全定義--
```

このモジュールで定義されている型と値は、ディレクトリ仕様の範囲内の他のＡＳＮ．１モジュールも、ディレクトリサービスにアクセスする他のアプリケーションに使用される。他のアプリケーションは独自の目的のため、このモジュールで定義された型と値を使用することができるが、このことによってディレクトリサービスの保守や改良に必要な機能拡張や修正に制限を与えることはない。

```
IMPORTS
  id-oc, id-at, id-mr, id-oa, id-sc, id-ar
    FROM UsefulDefinitions {joint-iso-ccitt-ds(5) modules(1) usefulIDefinitions(0) 2 }
--属性データ型--
Attribute               ::=   SEQUENCE {
    type        AttributeType (   {SupportedAttributes } ),
    values      SET SIZE(1..MAX) OF AttributeValue (  {SupportedAttributes }   {@type } ) }
AttributeType           ::= ATTRIBUTE.&id
AttributeValue          ::= ATTRIBUTE.&Type
AssertionValue          ::= ATTRIBUTE.&Equality-match.&AssertionType
AttributeTypeAndValue   ::= SEQUENCE {
    type        AttributeType (   {SupportedAttributes } ),
    value       AttributeValue(   {SupportedAttributes }   {@Type} )}
AttributeValueAssertion ::= SEQUENCE {
    type        AttributeType (   {SupportedAttributes } ),
    assertion   AsseriionValue(   {SupportedAttributes }   {@type} )}
```

おそらくプロファイルを標準化するためか、あるいはインプリメント適合申請を定義付けるためという理由により、以下の情報オブジェクトに関する定義は延期される。

この情報オブジェクトではAttribute のValue 要素、AttributeTypeAndValue のvalue 要素およびAttributeTypeAssertionのassertion 要素におけるテーブル制限を明確にすることが要求される。

```
--SupportedAttributes          ATTRIBUTE    ::=
--       { id-at-objectClass | id-at-aliasedEntryName   | ..}

--名前付けデータ型--
  Name                         ::=CHOICE {RDNSequence --現在は1つのみ有効--}
  RDNSequence                  ::=SEQUENCE OF RelativeDistinguishedName
  DistinguishedName            ::=RDNSequence
  RelativeDistinguishedName    ::=SET SIZE ( .. MAX }  OF AttributeTypeAndValue

--サブツリーデータ型--
  SubtreeSpecification         ::=SEQUENCE {
      base           [0]            LocalName DEFAULT  { } ,
      COMPONENTS OF                 ChopSpecification,
      specificationFilter  [4]      Refinement OPTIONAL }
      -- empty set specifies whole administrative area

  LocalName          ::=     RDNSequence
  ChopSpacification          ::= SEQUENCE  {
       specificExclusions  [1]    SET OF CHOICE {
                                      chopBefore   [0]     LocalName,
                                      chopAfter    [1]     LocalName } OPTIONAL,
       minimum             [2]    BaseDistance DEFAULT 0,
       maximum             [3]    BaseDistance OPTIONAL}
  BaseDistance       ::=    INTEGER (0.. MAX)
  Refinement         ::=     CHOICE  {
       item   [0]   OBJECT-CLASS.&id,
       and    [1]   SET OF Refinement,
       or     [2]   SET OF Refinement,
       not    [3]   Refinement}
--OBJECT-CLASS  情報オブジェクトクラス仕様--
  OBJECT-CLASS           ::=     CLASS {
       &Superclasses           OBJECT-CLASS OPTIONAL,
       &kind                   ObjectClassKind DEFAULT structural,
       &MandatoryAttributes    ATTRIBUTE OPTIONAL,
```

```
      &OptionalAttributes     ATTRIBUTE OPTIONAL,
      &id                     OBJECT IDENTIFIER UNIQUE }
  WTIH SYNTAX  {
      [ SUBCLASS OF           &Superclasses ]
      [ KIND                  &kind  ]
      [ MUST CONTAIN          &MandatoryAttributes ]
      [ MAY CONTAIN           &OptionalAttributes]
       ID                     &id}
  ObjectClassKind       ::=   ENUMERATED  {
      abstract      (0),
      structural    (1),
      auxiliary     (2)}
-- オブジェクトクラス --
  top   OBJECT-CLASS    ::=      {
      KIND            abstract
      MUST CONTAIN      { objectClass}
      ID                id-oc-top}
  alias  OBJECT-CLASS  ::=     {
      SUBCLASS OF       { top  }
      MUST CONTAIN      { aliasedEntryName }
      ID                id-oc-alias}


-- ATTRIBUTE  情報オブジェクトクラス仕様 --
ATTRIBUTE           ::=    CLASS {
       &derivation          ATTRIBUTE OPTIONAL,
       &Type                OPTIONAL,--&Type か&derivation のどちらかが要求さ
れる--
       &equality-match      MATCHINGRULE OPTIONAL,
       &ordering-match      MATCHINGRULE OPTIONAL,
       &substrings-match    MATCHINGRULE OPTIONAL,
       &single-valued       BOOLEAN DEFAULE FALSE,
       &collective          BOOLEAN DEFAULT FALSE,
       -- 運用拡張 --
       &no-user-modification  BOOLEAN DEFAULT FALSE,
       &usage               AttributeUsage DEFAULT userApplications,
       &id                  OBJECT IDENTIFIER UNIQUE }
WITH SYNTAX  {
       [ SUBTYPE OF            &supertype ]
       [ WITH SYNTAX           &Type]
```

```
        [ EQUALITY MATCHING RULE   &equality-match]
        [ ORDERING MATCHING RULE   &ordaring-match]
        [ SUBSTRINGS MATCHING RULE &substrings-match]
        [ SINGLE VALUE             &single-valued ]
        [ COLLECTIVE               &collective]
        [ NO USER MODIFICATION     &no-user-modification]
        [ USAGE                    &usage ]
        ID                         &id  }
AttributeUsage             ::=   ENUMERATED  {
         userApplications           (0),
         directortOperation         (1),
         distributedOperation       (2),
         dSAOperation               (3)}

 -- 属性--
objectClass         ATTRIBUTE          ::=     {
         WITH SYNTAX                     OBJECT IDENTIFIER
         EQUALITY MATCHING RULE          objectIdentifierMatch
         ID                              id-at-objectClass}
aliasedEntryName  ATTRIBUTE   ::=      {
         WITH SYNTAX                     DistinguishedName
         EQUALITY MATCHING RULE          distinguishedNameMatch
         SINGLE VALUE                    TRUE
         ID                              id-at-aliasedEntryName }
-- MATCHING-RULE  情報オブジェクトクラス仕様 --
MATCHING-RULE          ::=     CLASS {
         &AssertionType OPTIONAL,
         &id                    OBJECT IDENTIFIER UNIQUE }
WITH SYNTAX        {
         [ SYNTAX                        &AssertionType ]
         ID                              &id}

--  照合規則  --
objectIdentifierMatch          MATCHING-RULE     ::=     {
         SYNTAX      OBJECT IDENTIFIER
         ID          id-mr-objectIdentifierMatch}
distinguishedNameMatch  MATCHING-RULE     ::=      {
         SYNTAX      DistinguishedName
         ID          id-mr-distinguishedNameMatch }
```

```
--NAME-FORM 情報オブジェクトクラス仕様--
NAME-FORM               ::=     CLASS    {
          &namedObjectClass     OBJECT-CLASS,
          &MandatoryAttributes  ATTRIBUTE,
          &OptionalAttributes   ATTRIBUTE OPTIONAL,
          &id                   OBJECT IDENTIFIER UNIQUE}
WTIH SYNTAX  {
          NAMES                 &nameObjectClass
          WITH ATTRIBUTES       &MandataryAttributes
          [ AND OPTIONALLY      &OptionalAttributes ]
          ID                    &id }

--STRUCTURE-RULEクラスとDIT 構造規則データ型--
STRUCTURE-RULE        ::=       CLASS {
          &nameForm                   NAME-FORM,
          &SuperiorStructureRules     STRUCTURE-RULE OPTIONAL,
          &id                         RuleIdentifier UNIQUE}
WITH SYNTAX  {
          NAME FORM                   nameForm]
          [ SUPERIOR RULES            &SuperiorStructureRules ]
          ID                          &ruleIdentifier}
DITStructureRule      ::=     SEQUENCE  {
          ruleIdentifier              RuleIdentifier ,
                                 -- サブスキーマの範囲内では唯一でなくてはならない--
          nameForm                    NAME-FORM.&id,
          superiorStructureRules      SIT OF RuleIdentifier OPTIONAL }
RuleIdentifier        ::=     INTEGER

-- CONTENT-RULE クラスとDIT 内容規則データ型--
CONTENT-RULE     ::=     CLASS    {
          &structuralClass            OBJECT-CLASS.&id UNIQUE,
          &Auxiliaries                OBJECT-CLASS OPTIONAL,
          &Mandatory                  ATTRIBUTE OPTIONAL,
          &Optional                   ATTRIBUTE OPTIONAL,
          &Precluded                  ATTRIBUTE OPTIONAL }
WTIH SYNTAX  {
          STRUCTURAL OBJECT CLASS     &structuralClass
          [ AUXILIARY OBJECT CLASSES  &Auxiliaries ]
          [ MUST CONTAIN              &Mandatory ]
          [ MAY CONTAIN               &Optional]
```

```
          [ MUST NOT CONTAIN          &Pracluded ]}
DITContentRule     ::=     SEQUENCE  {
          structuralObjectClass      OBJECT-CLASS.&id,
          auxiliaries                SET OF OBJECT-CLASS.&id OPTIONAL,
          mandatory         [1]      SET OF ATTRIBUTE.&id OPTIONAL,
          optional          [2]      SET OF ATTRIBUTE.&id OPTIONAL,
          precluded         [3]      SET OF ATTRIBUTE.&id OPTIONAL  }

--システムスキーマ情報オブジェクト--

--サブオブジェクトクラス--
subentry                             OBJECT-CLASS  ::=   {
          SUBCLASS OF         { top  }
          KIND                structural
          MUST CONTAIN        { commonName  | subtreeSpecification  }
          ID                  id-sc-subentry }
accessControlSubentry                OBJECT-CLASS  ::=   {
          KIND                auxiliary
          ID                  id-sc-accessControlSubentry  }

collectiveAttributeSubentry          OBJECT-CLASS ::=    {
          KIND                auxiliary
          ID                  id-sc-collectiveAttributeSubentry  }

--属性--
createTimestamp           ATTRIBUTE   ::=      {
          WITH SYNTAX                 GeneraliZedTime
                        -- X.203/ISO 8824勧告の34.3(b) および(c) 項による--
          EQUALITY MATCHING RULE      generaliZedTimeMatch
          ORDERING MATCHING RULE      generaliZedTimeOrderingMatch
          SINGLE VALUE                TRUE
          NO USER MODIFICATION        TRUE
          USAGE                       directoryOperation
          ID                          id-oa-createTimestamp }
modifyTimestamp           ATTRIBUTE   ::=      {
          WITH SYNTAX                 GeneralizedTime
                        -- X.203/ISO 8824勧告の34.3(b) および(c) 項による--
          EQUALITY MATCHING RULE      generalizedTimeMatch
          ORDERING MATCHING RULE      generaIizedTimeOrderingMatch
          SINGLE VALUE                TRUE
```

```
        NO USER MODIFICATION        TRUE
        USAGE                       directoryOperation
        ID                          id-oa-modifyTimestamp }
creatorsName            ATTRIBUTE   ::=     {
        WITH SYNTAX                 DistinguishedName
        EQUALITY MATCHING RULE      distinguishedNameMatch
        SINGLE VALUE                TRUE
        NO USER MODIFICATION        TRUE
        USAGE                       directoryOperation
        ID                          id-oa-creatorsName  }
modifiersName           ATTRIBUTE   ::=     {
        WITH SYNTAX                 DistinguishedName
        EQUALITY MATCHING RULE      distinguishedNameMatch
        SINGLE VALUE                TRUE
        NO USER MODIFICATION        TRUE
        USAGE                       directoryOperation
        ID                          id-oa-modifiersName }
administrativeRole      ATTRIBUTE   ::=     {
        WITH SYNTAX                 OBJECT-CLASS.&id
        EQUALITY MATCHING RULE      objectIdentifierMatch
        USAGE                       directoryOperation
        ID                          id-oa-administrativeRole}
subtreeSpecification    ATTRIBUTE   ::=     {
        WITH SYNTAX                 SubtreeSpecification
        SINGLE VALUE                TRUE
        USAGE                       direcioryOperation
        ID                          id-oa-subtreeSpecification}
collectiveExclusions    ATTRIBUTE   ::=     {
        WITH SYNTAX                 OBJECT IDENTIFIER
        EQUALITY MATCHING RULE      objectIdentifierMatch
        USAGE                       directoryOperation
        ID                          id-oa-collectiveExclusions}

--オブジェクト識別子割当て--

--オブジェクトクラス--
id-oc-top                           OBJECT IDENTIFIER    ::=    {id-oc 0 }
id-oc-alias                         OBJECT IDENTIFIER    ::=    {id-oc 1 }

--属性--
```

```
id-at-objectClass                       OBJECT IDENTIFIER     ::=     {id-at 0 }
id-at-aliasedEntryName                  OBJECT IDENTIFIER     ::=     {id-at 1 }

--照会規則--
id-mr-objectIdentifierMatch             OBJECT IDENTIFIER     ::=     {id-mr 0 }
id-mr-distinguishedNameMatch            OBJECT IDENTIFIER     ::=     {id-mr 1 }

--運用属性--
id-oa-excludeAllCollectiveAttributes    OBJECT IDENTIFIER     ::=     {id-oa 0 }
id-oa-createTimestamp                   OBJECT IDENTIFIER     ::=     {id-oa 1 }
id-oa-madifyTimestamp                   OBJECT IDENTIFIER     ::=     {id-oa 2 }
id-oa-creatorName                       OBJECT IDENTIFIER     ::=     {id-oa 3 }
id-oa-modifiersName                     OBJECT IDENTIFIER     ::=     {id-oa 4 }
id-oa-administrativeRole                OBJECT IDENTIFIER     ::=     {id-oa 5 }
id-oa-subtreeSpecification              OBJECT IDENTIFIER     ::=     {id-oa 6 }
id-oa-collectiveExclusions              OBJECT IDENTIFIER     ::=     {id-oa 7 }

--サブエントリクラス--
id-sc-subentry                          OBJECT IDENTIFIER     ::=     {id-sc 0 }
id-sc-accessControlSubentry             OBJECT IDENTIFIER     ::=     {id-sc 1 }
id-sc-collectiveAttributeSubentry       OBJECT IDENTIFIER     ::=     {id-sc 2 }

--管理上の役割--
id-ar-autonomousArea                    OBJECT IDENTIFIER     ::=     {id-ar 1 }
id-ar-accessControlSpecificArea         OBJECT IDENTIFIER     ::=     {id-ar 2 }
id-ar-accessControlInnerArea            OBJECT IDENTIFIER     ::=     {id-ar 3 }
id-ar-subschemaAdminSpecificArea        OBJECT IDENTIFIER     ::=     {id-ar 4 }
id-ar-collectiveAttributeSpecificArea   OBJECT IDENTIFIER     ::=     {id-ar 5 }
id-ar-collectiveAttributeInnerArea      OBJECT IDENTIFIER     ::=     {id-ar 6 }

END
```

付属資料　C　　サブスキーマ管理のＡＳＮ．１表記

（ＪＴ－Ｘ５００に対する）

この付属資料は、ＡＳＮ．１モジュール"Schema Administration" の形態のサブスキーマ管理のASN.1 の型、値、情報オブジェクトクラスの定義を含む。

```
SchemaAdministration {joint-iso-ccitt ds(5) modules(1) schemaAdministration(23) 2 }
   DEFINITIONS ::=
   BEGIN

   -- EXPORTS ALL --
```

「本モジュールで定義された型および値は、ディレクトリ仕様の範囲内のASN.1 モジュールやディレクトリサービスにアクセスする他のアプリケーションにより引用される。他のアプリケーションは、各々の目的のため、本モジュールで定義された型および値を使用することが出来るが、これによって、ディレクトリサービスを保守したり、改良するために必要な機能拡張や修正に制約を与えるものではない。」

```
IMPORTS
  informationFramework, selectedAttributeTypes, upperBounds, id-soc, id-soa
   FROM UsefulDefinition {joint-iso-ccitt ds(5) modules(1) usefulDefinitions(0) 2 }
        OBJECT-CLASS, ATTRIBUTE, MATCHING-RULE, DITStructureRule, DIRContentRule,
        ObjectClassKind, AttributeUsage, subentry
                 FROM InformationFramework informationFramework
        DirectoryString  {}
                FROM SelectedAttributeTypes selectedAttributeTypes
        ub-schema
                FROM UpperBounds upperBounds  ;
--型--
   DITStructureRuIeDescription      ::=     SEQUENCE  {
          COMPONENTS OF          DITStructureRule,
          name                [1]      SET DirectoryString { ub-schema}  OPTIONAL,
          description                  DirectoryString   { ub-schema}  OPTIONAL,
          obsolete                     BOOLEAN DEFAULT FALSE}
   DITContentRuleDescription        ::=     SEQUENCE  {
          COMPONENTS OF          DITContentRuIe,
```

```
        name                [4]     SET OF DirectoryString  { ub-schema}  OPTIONAL,
        description                 DirectoryString { ub-schema} OPTIONAL,
        obsolete                    BOOLEAN DEFAULT FALSE}
MatchingRuleDescription         ::=     SEQUENCE  {
        identifier                  MATCHING-RULE.&id,
        name                        SET OF DirectryString { ub-schema}  OPTIONAL,
        description                 DirectoryString { ub-schema}  OPTIONAL,
        obsolete                    BOOLEAN DEFAULT FALSE,
        information         [0]     DirectoryString { ub-schema} }
                                    -- describes the ASN.1 syntax
AttributeTypeDescription        ::=     SEQUENCE {
        identifier                  ATTRIBUTE.&id,
        name                        SET OF DirectoryString  { ub-schema} OPTIONAL,
        description                 DirectoryString { ub-schema}  OPTIONAL,
        obsolete                    BOOLEAN DEFAULT FALSE,
        information         [0]     AttributeTypeInformation }


AttributeTypeInformation        ::=     SEQUENCE  {
        derivation          [0]   ATTRIBUTE.&id OPTIONAL,
        equalityMatch       [1]   MATCHING-RULE.&id OPTIONAL,
        orderingMatch       [2]   MATCHING-RULE.&id OPTIONAL,
        substringsMatch     [3]   MATCHING-RULE.&id OPTIONAL,
        attributeSyntax     [4]   DirectoryString  { ub-schema}  OPTIONAL,
        multi-valued        [5]   BOOLEAN DEFAULT TRUE,
        collective          [6]   BOOLEAN DEFAULT FALSE,
        userModifiable      [7]   BOOLEAN DEFAULT TRUE,
        application                AttributeUsage OPTIONAL}
ObjectClassDescription          ::=     SEQUENCE  {
        identifier                 OBJECT-CLASS.&id,
        name                       SET OF DirectoryString { ub-schema}  OPTIONAL,
        description                DirectoryString  { ub-schema}  OPTIONAL,
        obsolete                   BOOLEAN DEFAULT FALSE,
        information         [0]    ObjectClassInformation}
ObjectClassInformation    ::=   SEQUENCE {
        subclassOf                 SET OF OBJECT-CLASS.&id OPTIONAL,
        kind                       ObjectClassKind DEFAULT  {structural} ,
        mandatories         [3]    SET OF ATTRIBUTE.&id OPTIONAL,
        optionals           [4]    SET OF ATTRIBUTE.&id OPTIONAL }
NameFormDescription       ::=   SEQUENCE {
```

```
        identifier                  NAME-FORM.&id,
        name                        SET OF DirectoryString { ub-schema}  OPTIONAL,
        description                 DirectoryString  { ub-schema}  OPTIONAL,
        obsolate                    BOOLEAN DEFAULT FALSE,
        information          [0]    NameFormInformation}
NameFormInformation      ::=        SEQUENCE  {
        subordinate                 OBJECT-CLASS.&id,
        namingMandatories           SET OF ATTRIBUTE.&id,
        namingOptionals             SET OF ATTRIBUTE.&id OPTIONAL}
MatchingRuleUseDescription          ::=     SEQUENCE  {
        identifier                  MATCHING-RULE.&id,
        name                        SET OF DirectoryString  {ub-schema }  OPTIONAL,
        description                 DirectoryString { ub-schema}  OPTIONAL,
        obsolete                    BOOLEAN DEFAULT FALSE,
        information          [0]    SET OF ATTRIBUTE.&id }

-- オブジェクトクラス--
 subschema OBJECT-CLASS         ::=       {
        KIND                    auxiliary
        MAY CONTAIN {
                dITStructureRules |
                nameForms |
                dITContentRules |
                objectClasses |
                attributeTypes |
                matchingRules |
                matchingRuleUse }
        ID                          id-soc-subschema}

 --属性--
 dITStructureRules          ATTRIBUTE    ::=      {
        WITH SYNTAX                      DITStructureRuleDescription
        EQUALITY MATCHING RULE           integerFirstComponentMatch
        USAGE                            directoryOperation
        ID                               id-soa-dITStructureRule}


 dITContentRules            ATTRIBUTE    ::=       {
          WITH SYNTAX                    DITContentRuleDescription
          EQUALITY MATCHING RULE         objectIdentifierFirstComponentMatch
```

```
        USAGE                          directoryOperation
        ID                             id-soa-dITContentRules}
matchingRules            ATTRIBUTE     ::=      {
        WITH SYNTAX                    MatchingRuleDescription
        EQUALITY MATCHING RULE         objetIdentifierFirstComponentMatch
        USAGE                          directoryOperation
        ID                             id-soa-matchingRules}
attributeTypes           ATTRIBUTE     ::=      {
        WITH SYNTAX                    AttributeTypeDescription
        EQUALITY MATCHING RULE         objectIdentifierFirstComponentMatch
        USAGE                          directoryOperation
        ID                             id-soa-attributeTypes }
objectClasses            ATTRIBUTE   ::=      {
        WITH SYNTAX                    ObjectClassDescription
        EQUALITY MATCHING RULE         objectIdentifierFirstComponentMatch
        USAGE                          directoryOperation
        ID                             id-soa-objectClasses}
nameForms                ATTRIBUTE     ::=      {
         WITH SYNTAX                   NameFormDescription
         EQUALITY MATCHING RULE        objectIdentifierFirstComponentMatch
         USAGE                         directoryOperation
         ID                            id-soa-nameForms}
matchingRuleUse          ATTRIBUTE     ::=      {
         WITH SYNTAX                   MatchingRuIeUseDescription
         EQUALITY MATCHING RULE        objectIdentifierFirstComponentMatch
         USAGE                         directoryOperation
         ID                            id-soa-matchingRuleUse}
structuralObjectCIass    ATTRIBUTE     ::=      {
         WITH SYNTAX                   OBJECT IDENTIFIER
         EQUALITY MATCHING RULE        objectIdentifierMatch
         SINGLE VALUE                  TRUE
         NO USER MODIFICATION          TRUE
         USAGE                         directoryOperation
         ID                            id-soa-structuralObjectClass}
governingStructureRule   ATTRIBUTE     ::=      {
         WITH SYNTAX                   INTEGER
         EQUALITY MATCHING RULE        integerMatch
         SINGLE VALUE                  TRUE
         NO USER MODIFICATION          TRUE
         USAGE                         directoryOperation
```

```
        ID                              id-soa-governingStructureRule }

--オブジェクト識別子割付--

--スキーマオブジェクトクラス--
id-soc-subschema                OBJECT IDENTIFIER   ::=    {id-soc 1}

--スキーマ運用属性--
id-soa-dITStructureRule         OBJECT lDENTIFIER   ::=    {id-soa 1}
id-soa-dITContentRules          OBJECT IDENTIFIER   ::=    {id-soa 2}
id-soa-attributeSyntaxes        OBJECT IDENTIFIER   ::=    {id-soa 3}
id-soa-matchingRules            OBJECT IDENTlFIER   ::=    {id-soa 4}
id-soa-attributeTypes           OBJECT IDENTIFlER   ::=    {id-soa 5}
id-soa-objectGlasses            OBJECT IDENTIFIER   ::=    {id-soa 6}
id-soa-nameForms                OBJECT IDENTIFIER   ::=    {id-soa 7}
id-soa-matchingRuIrUse          OBJECT IDENTIFIER   ::=    {id-soa 8}
id-soa-structuralObjectClass    OBJECT IDENTIFIER   ::=    {id-soa 9}
id-soa-governingStructureRule   OBJEGT IDENTIFIER   ::=    {id-soa 10 }

END
```

付属資料D　　ＤＳＡ運用属性型のＡＳＮ．１表記

（ＪＴ－Ｘ５００に対する）

この付属資料はＡＳＮ．１モジュール"DSA Operational Attribute Type"の形態のディレクトリ仕様の全てのASN.1 の型および値定義を含む。

```
DSAOperationalAttributeTypes {joint-iso-ccitt ds(5) modules(1)
                                    dsaOperationalAttributes(22)version(2)}
DEFINITIONS ::=
BEGIN


--Exports all--
```

「本モジュールで定義された型および値は、ディレクトリ仕様の範囲内のASN.1 モジュールやディレクトリサービスにアクセスする他のアプリケーションにより引用される。他のアプリケーションは、各々の目的のため、本モジュールで定義された型および値を使用することが出来るが、これによって、ディレクトリサービスを保守したり、改良するために必要な機能拡張や修正に制約を与えるものではない。」

```
IMPORTS
  id-doa, id-kmr, informationFramework, distributedOperations
    FROM UsefulDefinitions {joint-iso-ccitt-ds {5) modules(1) usefulDefinitions(0) 2}
      ATTRIBUTE, MATCHING-RULE
        FROM InformationFramework informationFramework
        AccessPoint, MasterAndShadowAccessPoints
        FROM DistributedOperations distributedOperations

--データの型--
DSEType    ::=     BIT STRING   {
    root          (0),        --root DSE--
    glue          (1),        --represents knowledge of a name only --
    cp            (2),        --context prefix--
    entry         (3),        --object entry--
    alias         (4),        --alias entry --
    subr          (5),        --subordinate reference --
```

```
    nssr          (6),        --non-specific subordinate reference--
    supr          (7),        --superior reference--
    xr            (8),        --cross reference --
    admPoint      (9),        --administrative point--
    subentry      (10),       --subentry--
    shadow        (11),       --shadow copy --
    immSupr       (13),       --immediate superior reference--
    rhob          (14),       --rhod information--
    sa            (15) }      --subordinate reference to alias entry--

SupplierOrConsumer       ::=  SET  {
    COMPONENTS OF             AccessPoint,--supplier or consumer--
    agreementID               [3]   OperationalBindingID}
SupplierInformation      ::=  SET  {
    COMPONENTS OF             SupplierOrConsumer, --supplier--
    supplier-is-master        [4]   BOOLEAN DEFAULT TRUE,
    non-supplying-master      [5]   AccessPoint OPTIONAL}
ConsumerInformation      ::=  SupplierOrConsumer  --consumer--
SupplierAndConsumers     ::=  SET  {
    COMPONENTS OF   AccessPoint,  --supplier--
    consumers  [3]  SET OF AccessPoint}

--属性の型--
dseType                  ATTRIBUTE   ::=      {
        WITH SYNTAX                  DSEType
        EQUALITY MATCHING RULE       bitStringMatch
        SINGLE VALUE                 TRUE
        NO USER MODIFICATION         TRUE
        USAGE                        dSAOperation
        ID                           id-doa-dseType}
myAccessPoint            ATTRIBUTE   ::=      {
        WITH SYNTAX                  AccessPoint
        EQUALITY MATCHING RULE       accassPointMatch
        SINGLE VALUE                 TRUE
        NO USER MODIFICATION         TRUE
        USAGE                        dSAOperation
        ID                           id-doa-myAccessPoint}
superiorKnowledge        ATTRIBUTE   ::=      {
        WITH SYNTAX                  AccessPoint
        EQUALITY MATCHING RULE       accessPointMatch
```

```
        SINGLE VALUE                TRUE
        NO USER MODIFICATION        TRUE
        USAGE                       dSAOperation
        ID                          id-doa-superiorKnowledge}
specificKnowledge       ATTRIBUTE   ::=     {
        WITH SYNTAX                 MasterAndShadowAccessPoints
        EQUALITY MATCHING RULE      masterAndShadowAccessPointsMatch
        SINGLE VALUE                TRUE
        NO USER MODIFICATION        TRUE
        USAGE                       distributedOperation}
        ID                          id-doa-specificKnowledge}
nonSpecificKnowledge    ATTRIBUTE   ::=     {
        WITH SYNTAX                 MasterAndShadowAccessPoints
        EQUALITY MATCHING RULE      masterAndShadowAccessPointsMatch
        NO USER MODIFICATION        TRUE
        USAGE                       disiributedOperation}
        ID                          id-doa-nonSpecificKnowledga }
supplierKnowledge       ATTRIBUTE   ::=     {
        WITH SYNTAX                 SupplierInformation
        EQUALITY MATCHING RULE      supplierInformationMatch
        NO USER MODIFICATION        TRUE
        USAGE                       dSAOperation
        ID                          id-doa-supplierKnowledge}
consumerKnowledge       ATTRIBUTE   ::=     {
        WITH SYNTAX                 ConsumerInformation
        EQUALITY MATCHING RULE      consumerInformationMatch
        NO USER MODIFICATION        TRUE
        USAGE                       dSAOperation
        ID                          id-doa-consumerKnowledge}
secondaryShadows        ATTRIBUTE   ::=     {
        WITH SYNTAX                 SupplierAndConsumers
        EQUALITY MATCHING RULE      supplierAndConsumersMatch
        NO USER MODIFICATION        TRUE
        USAGE                       dSAOperation
        ID                          id-doa-secondaryShadows }

--照合規則--
accessPointMatch                    MATCHING-RULE   ::=     {
        SYNTAX      Name
        ID          id-kmr-accessPointMatch}
```

```
masterAndShadowAccessPointsMatch      MATCHING-RULE     ::=      {
        SYNTAX      SET OF Name
        ID          id-kmr-masterShadowMatch }
supplierOrConsumerInformationMatch    MATCHING-RULE     ::=      {
        SYNTAX      SET {ae-title [0] Name, agreement-identifier [2] INTEGER }
        ID          id-kmr-supplierConsumerMatch   }
supplierAndConsumersMatch             MATCHING-RULE     ::=      {
        ON          SupplierAndConsumers
        SYNTAX      Name
        ID          id-kmr-supplierConsumersMatch}

--オブジェクト識別子割付--

-- dsa運用属性--
id-doa-dseType                        OBJECT IDENTIFIER         ::=    {id-doa 0}
id-doa-myAccessPoint                  OBJECT IDENTIFIER         ::=    {id-doa 1}
id-doa-superiorKnowledge              OBJECT IDENTIFIER         ::=    {id-doa 2}
id-doa-specificKnowledge              OBJECT IDENTIFIER         ::=    {id-doa 3}
id-doa-nonSpecificKnowledge           OBJECT IDENTIFIER         ::=    {id-doa 4}
id-doa-supplierKnowledge              OBJECT IDENTIFIER         ::=    {id-doa 5}
id-doa-consumerKnowledge              OBJECT IDENTIFIER         ::=    {id-doa 6}
id-doa-secondaryShadows               OBJECT IDENTIFIER         ::=    {id-doa 7}

--知識照合規則--
id-kmr-accessPointMatch               OBJECT IDENTIFIER         ::=    {id-kmr 0}
id-kmr-masterShadowMatch              OBJECT IDENTIFIER         ::=    {id-kmr 1}
id-kmr-supplierConsummerMatch         OBJECT IDENTIFIER         ::=    {id-kmr 2}
id-kmr-supplierConsummersMatch        OBJECT IDENTIFIER         ::=    {id-kmr 3}

END
```

付属資料Ｅ－日本語名の実現方法
（ＪＴ－Ｘ５００に対する）

Ｅ．１　要求条件

ディレクトリシステムは、国際的な相互接続を想定している。したがって、日本国内のディレクトリシステムへの海外からのアクセス、または、逆の場合も考慮する必要がある。

また、日本語のディレクトリシステムでは日本語（T.61文字列）の利用も実現する必要がある。

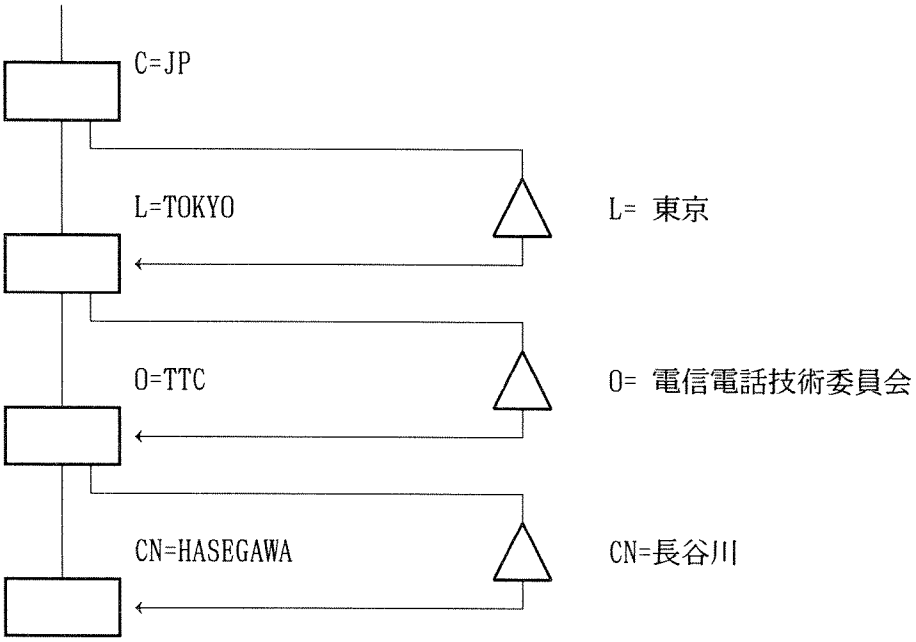
Ｅ．２　実現方法

国際接続性を考慮し、日本語をサポートしない端末からのアクセスを可能とするため、以下の方法がある。

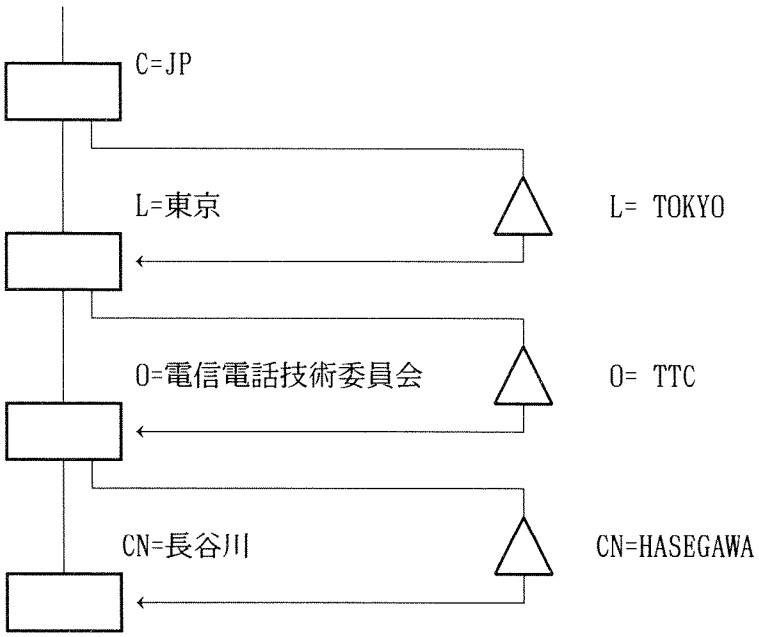
① 印字可能文字列をオブジェクトの識別名に用いる。また、日本語により、オブジェクトを識別する場合は、日本語を別名に規定して使用する。

② 日本語をオブジェクトの識別名に用いる。また、印字可能文字列を別名に規定して使用する。

付図Ｅ－１／ＪＴ－Ｘ５００に日本語名の指定例を示す。



（１）　日本語名の例（その１）



（２）　日本語名の例（その２）

凡例：　：オブジェクトエントリ　　：別名エントリ

付図Ｅ－１／ＪＴ－Ｘ５００　日本語名の例

付属資料Ｆ－ＭＨＳアプリケーションのオブジェクト識別子定義
（ＪＴ－Ｘ５００に対する）

この付属資料はＴＴＣ標準ＪＴ－Ｘ５２０付属資料ＥのＡＳＮ．１モジュールの中で引用されている各種のオブジェクト識別子の参照用に定義している。

ＡＳＮ．１を使用して定義している。ＭＨＳアプリケーションで割り付けるすべてのオブジェクト識別子はこの付属資料の中で割り付けられている。

この付属資料は、ＡＳＮ．１モジュール及びＭＨＳ自身へのオブジェクト識別子を除いてオブジェクト識別子を割り付けている。

```
MHSObjectIdentifiers  {joint-iso-ccitt
        mhs-motis(6) arch(5) modules(0) object-identifiers(0)}
DEFINITIONS IMPLICIT TAGS ::=
BEGIN
-- Prologue
-- Exports everything,
IMPORTS -- nothing -- ;
ID ::= OBJECT IDENTIFIER
--ＭＨＳ関連
id-mhs-protocols ID ::= {joint-iso-ccitt mhs-motis(6) protocols(0)}
                --ＭＨＳ応用コンテキスト
                --ＩＴＵ－Ｔ勧告　X.419参照
id-ipms  ID ::= {joint-iso-ccitt mhs-motis(6) imps (1)}
                --個人間メッセージ通信
                --ＩＴＵ－Ｔ勧告　X.420参照
id-asdc  ID ::= {joint-iso-ccitt mhs-motis(6) asdc (2)}
                --抽象サービス定義記法
                --ＩＴＵ－Ｔ勧告　X.407参照
```

```
id-mts   ID ::= {joint-iso-ccitt mhs-motis(6) mts  (3)}
                --メッセージ転送システム
                --ＩＴＵ－Ｔ勧告　X.411参照
id-ms    ID ::= {joint-iso-ccitt mhs-motis(6) ms   (4)}
                --メッセージ蓄積
                --ＩＴＵ－Ｔ勧告　X.413参照
id-arch  ID ::= {joint-iso-ccitt mhs-motis(6) arch (5)}
                --アーキテクチャ全般
                --ＩＴＵ－Ｔ勧告　X.402参照
id-group ID ::= {joint-iso-ccitt mhs-motis(6) group(6)}
                --予備
--カテゴリ
   id-mod   ID ::= {id-arch 0}   --モジュール（非規定）
   id-oc    ID ::= {id-arch 1}   --オブジェクトクラス
   id-at    ID ::= {id-arch 2}   --属性型
   id-as    ID ::= {id-arch 3}   --属性構文
--モジュール
   id-object-identifiers                 ID ::= {id-mod 0}   --非規定
   id-directory-objects-and-attributes   ID ::= {id-mod 0}   --非規定
--オブジェクトクラス
   id-oc-mhs-distribution-list           ID ::= {id-oc 0}
   id-oc-mhs-message-store               ID ::= {id-oc 1}
   id-oc-mhs-message-transfer-agent      ID ::= {id-oc 2}
   id-oc-mhs-user                        ID ::= {id-oc 3}
   id-oc-mhs-user-agent                  ID ::= {id-oc 4}
--属　性
   id-at-mhs-deliverable-content-length      ID ::= {id-at  0}
   id-at-mhs-deliverable-content-types       ID ::= {id-at  1}
   id-at-mhs-deliverable-eits                ID ::= {id-at  2}
   id-at-mhs-dl-members                      ID ::= {id-at  3}
```

```
   id-at-mhs-dl-submit-permissions          ID ::= {id-at  4}
   id-at-mhs-message-store-dr               ID ::= {id-at  5}
   id-at-mhs-or-addresses                   ID ::= {id-at  6}
   id-at-mhs-supported-automatic-actions    ID ::= {id-at  8}
   id-at-mhs-supported-content-types        ID ::= {id-at  9}
   id-at-mhs-supported-optional-attributes  ID ::= {id-at 10}
--属性構文
   id-as-mhs-dl-submit-permission       ID ::= {id-as 0}
   id-as-mhs-or-address                 ID ::= {id-as 1}
   id-as-mhs-or-name                    ID ::= {id-as 2}
END--of MHSObjectIdentifiers
```

付　録　1

ディレクトリの応用

1．ディレクトリ環境

（注）ここで、「ネットワーク」という用語は、一般的な意味で使用される。つまり、電気通信サービスに関連する接続されたシステムとプロセスの集合を表し、必ずしもＯＳＩのネットワーク層に関係するものだけに限るものではない。

ディレクトリは、以下の環境でサービスを提供する。

(1) 多くの電気通信ネットワークが大規模化し、絶えず変化している。

(a) 多様なオブジェクトが、単独またはグループ単位でネットワークに予告無く加入し或いは離脱する。

(b) オブジェクト（特にネットワークノード）の接続性は、そのオブジェクト間のパスの追加及び削除によって変化する。

(c) アドレス、可用性、物理的位置等のオブジェクトの性質は、いつでも変えることができる。

(2) ディレクトリ全体でみれば更新頻度は高いが、あるオブジェクトについての有効な期間は決して短くない。オブジェクトのアドレス、物理的位置、可用性などを更新する頻度より、使用する頻度のほうが高い。

(3) 一般に、現在の電気通信サービスに関係するオブジェクトは、数字やシンボルの列で識別される。しかし、これは割り当てや処理が簡単なように選ばれたもので、人間に使いやすいように選ばれたものではない。

2．ディレクトリサービスの特徴

ディレクトリ機能の必要性を、以下に示す。

(1) ネットワークユーザを、ネットワークの頻繁な変更から可能な限り切り離す。これは、ユーザとオブジェクト間を間接化することによって実現される。この間接化の一つとして、オブジェクトをアドレスでなく名前で参照する方法がある。ディレクトリは、このマッピングサービスに必要な機能を備える必要がある。

(2) ネットワークを、よりユーザに親しみ易く見せる。例えば、別名の使用やオブジェクトを属性から探し出す機能（3.5参照）等を備えることによって、ネットワーク情

報の検索および使用における負担を軽くする。

ディレクトリによって、ユーザはネットワークに関する種々の情報を得ることができる。またディレクトリは、その情報の保守、配布、機密保護を行う。

3．ディレクトリの使用形態

（注）ここでは、ディレクトリの検索だけについて述べる。ディレクトリの変更サービスは、ＤＩＢの保守のために使用時間外に使用するものと考える。

3.1　はじめに

ディレクトリサービスは、ＤＵＡが作成可能な要求とそのパラメタによって定義されている。しかし、アプリケーション設計者は、最終目的を指向してアプリケーションにおけるディレクトリに対する情報検索要求を考える。そこで、本節では、多くのアプリケーションに関連しそうなディレクトリサービスの高度な利用形態について述べる。

3.2　取り出し

ディレクトリに対する問い合わせで最も頻繁に使用されるのは、ディレクトリからの直接的な取り出しであろう。ＤＵＡはオブジェクトの識別名と属性型を与え、ディレクトリはその属性型の値を応答する。これは、要求した属性型があるアドレスに対応する典型的なディレクトリの機能を、一般化したものである。ＯＳＩのＰＳＡＰアドレス、メッセージ通信処理のＯ／Ｒアドレス、電話やテレックス番号など、種々のアドレスの属性型が標準化されている。

取り出しは、読出しサービスによって実現されるが、これは更に以下の様に一般化される。

・取り出しでは、別名など識別名以外のオブジェクトの名前も使用できる。

・１つの問い合わせで、複数の属性型の値を要求できる。極端な場合、エントリの全属性の値が応答されることがある。

3.3　ユーザに親しみやすい名前付け

オブジェクトに与える名前は、可能な限り人間に利用され覚え易くすることができる。

このような性質を持つ名前は、一般的に、ある目的に合わせて作られるのではなく、オブジェクトに固有な属性から作られる。オブジェクトの名前は、それを参照するすべてのアプリケーションに共通になるだろう。

3.4　ブラウジング

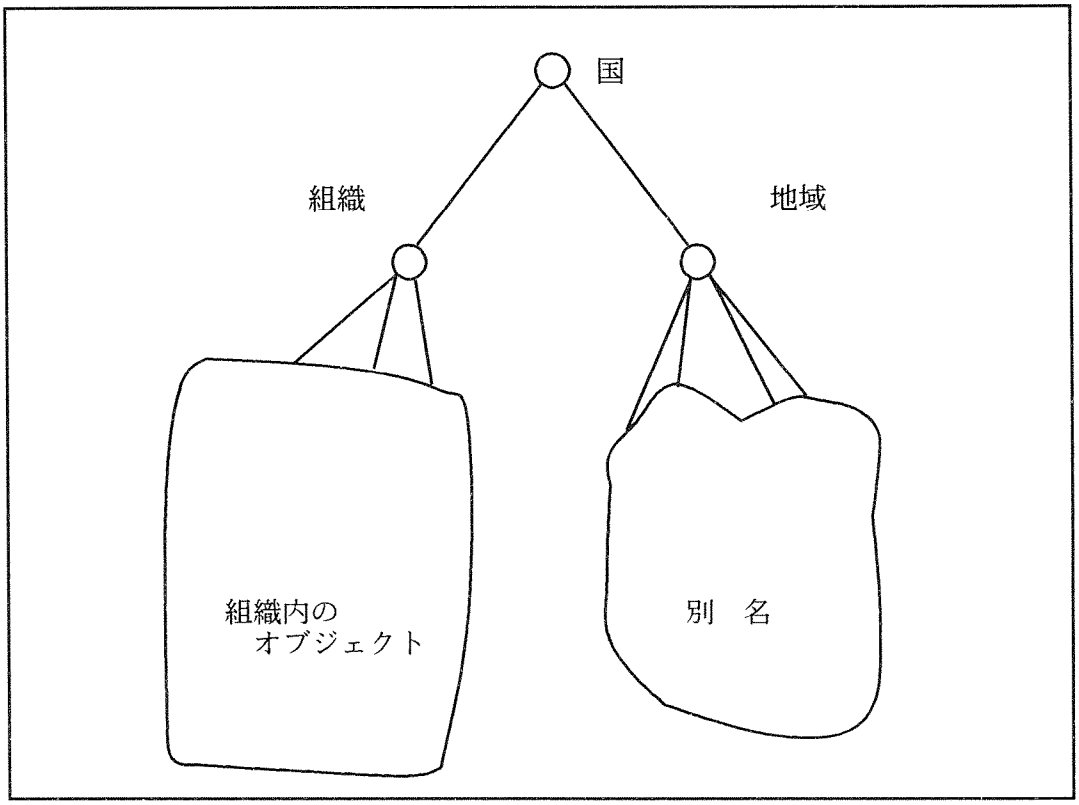
人間が直接ディレクトリを利用する場合、名前がユーザに親しみ易かろうと無かろうと、ユーザ（またはＤＵＡ）が目的のオブジェクトの名前を直接指定できないことがある。しかし、ユーザが目的のエントリを見たとき、おそらく自分が探しているエントリか否か、判断できるだろう。ブラウジングは、目的のエントリを探すために、人間のユーザがＤＩＢの中を探し回る機能を提供する。

ブラウジングは、リストサービスと検索サービスと読出しサービス（検索サービスは読出し機能に含まれるが）を組み合わせることで実現される。

3.5　職業別番号帳

職業別番号帳機能を実現するには、種々の方法が考えられる。最も簡単な方法は、値が分類（例えば、ＴＴＣ標準ＪＴ－Ｘ５２０に定義されている職種という属性型）に関係する属性に対し提示を使用する、フィルタリングに基づく方法である。この方法は、必要な属性が存在することを保証する以外には、ＤＩＴに特別な情報を格納する必要が無い。しかし、一般的に、フィルタリングは、フィルタされる全集合を作成する必要があるので、大きな母集団からの探索はコストがかかってしまう。

名前構造が職業別番号帳の探索用に特別に設計されたサブツリーに基づく、もう一つの方法がある。付録図１－３／ＪＴ－Ｘ５００に、別名エントリだけから成る職業別番号帳サブツリーの例を示す。実際の職業別番号帳サブツリーでは、オブジェクトのエントリと別名のエントリが混在するであろう。但し、ディレクトリの中に、あるオブジェクトのオブジェクトエントリが２つになってはならない。



付図１－３／ＪＴ－Ｘ５００　職業別番号帳へのアプローチ
（ITU-T X.500）

3.6　グループ

グループとはメンバの集合であり、メンバの追加削除によって変化する。そのメンバがオブジェクトであり、またグループも１つのオブジェクトである。ディレクトリに対し以下に示す要求を出すことができる。

・あるオブジェクトが、あるグループのメンバであるか否かを示す。

・あるグループのメンバをリストする。

グループは、グループのエントリが多値のメンバ属性（このような属性型はＴＴＣ標準ＪＴ－Ｘ５２０で定義されている）を持つことによってサポートされる。上記の２つの機能は、それぞれ比較と読出しで実現される。

もしアプリケーションにとって意味があれば、グループのメンバは、それ自体グループでもかまわない。しかし、これは、ＤＳＡが提供する非再帰的なサービスをＤＵＡが再帰的な照合サービスに拡張することによって実現されるべきである。

3.7　認　証

多くのアプリケーションは、ある動作を許可する前に、オブジェクトに対しユーザ本人である証明を必要とする。ディレクトリは、この認証の過程を提供する。（別の問題として、ディレクトリは、アクセス制御を提供するために、ユーザに認証を必要とする）。

「簡易認証」と呼ばれる簡単な認証の方法がある。あるサービスに対し自分自身の認証をしたいユーザのディレクトリエントリにユーザパスワードという属性を格納することに基づく方法である。あるサービス要求時に、ディレクトリは与えられた値が実際にユーザのパスワードであることを確認または否定する。これによって、ユーザは、各サービス毎に異なるパスワードを持たなくてすむ。簡易認証を使用する環境で、ローカルにパスワードを交換することが適当でない場合、ディレクトリは、再使用や誤用からパスワードを一方向関数によって保護するという付加的な機能を備える必要がある。

公開鍵暗号システムに基づく、より複雑で改竄を防ぐのに適した「厳密認証」と呼ばれる認証方法がある。ディレクトリは、ユーザの公開鍵の格納場所として使われる。

4．一般的アプリケーション

4.1　はじめに

ディレクトリは、ある電気通信サービスに特定せず多くの一般的なアプリケーションを暗黙的にサポートするかもしれない。このアプリケーションとして、個人間通信におけるディレクトリと、システム間通信（ＯＳＩ指向）におけるディレクトリの２つについて記述する。

（注）3.7で認証は、アクセスパターンとして述べたが、一般的なディレクトリアプリケーションとしても考えられる。

4.2　個人間通信

このアプリケーションの意図は、人あるいはその代行者に、他の人やグループ等と通信するための情報を提供することである。

人、組織役割、グループ名というオブジェクトのクラスのエントリが、このアプリケーションに関係している。おそらく直接的でないだろうが、国、組織、組織単位という他の多くのクラスも関係している。

属性型は、名前付けのためより、一般的には、アドレッシング属性が、考慮されている。ある人のエントリには、電話、電子メール、テレックス、ＩＳＤＮ、物理的配送手段（例えば、郵便システム）、ファクシミリ等を含む無限に伸びる可能性があるリストから、その人に届く各通信手段を選択し、それに対応するアドレスが格納されているであろう。電子メールなどの場合では、エントリは、ユーザ所有の機器が扱える情報のタイプ等ある付加的情報を格納している。もし認証がサポートされるならユーザパスワードと資格証明が必要になるだろう。

種々のオブジェクトクラスで使用されている名前付け方法は、ユーザに親しみ易くあるべきであり、別名を代わりの名前として格納するのに適し、名前の変更後の連続性を備えるのに適しているべきである。

このアプリケーションによって、取り出し、ユーザに親しみやすい名前付け、ブラウジング、職業別番号帳検索、グループのアクセス形態が明らかになるだろう。場合によって、認証もまた使用されるだろう。

4.3　（ＯＳＩ指向）システム間通信

ＯＳＩの参照モデルによれば、ＯＳＩは2つのディレクトリ機能を必要としている。1つは、応用層での操作で、応用タイトルからＰＳＡＰアドレスへのマッピングである。もう1つは、ネットワーク層での操作で、ＮＳＡＰアドレスからＳＮＰＡ(Subnetwork point of Attachment)アドレスへのマッピングである。

（注）ここでは、応用層のケースだけを扱う。

マッピングに必要な情報が他の方法から得られない場合、ディレクトリに問い合わせてマッピングが実行される。

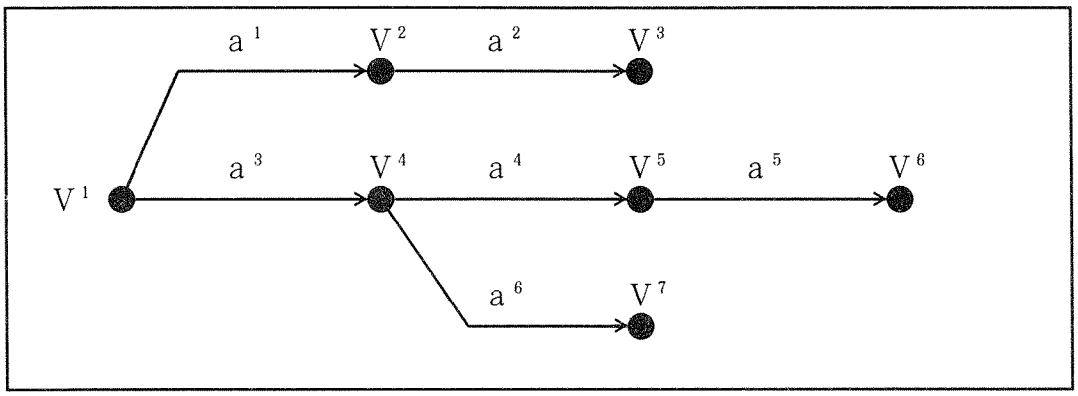
ユーザは、応用エンティティであり、着目しているオブジェクトクラスとそのサブクラスもまた応用エンティティである。

名前付けのために使用する以外の属性型で主に考慮されるのは、ＰＳＡＰアドレスである。他の属性型は、ディレクトリの機能の必要性の観点からでなく、応用エンティティタイプ、応用コンテキストのリスト、抽象構文などを、照合あるいは探索するためにある。認証に関係ある属性型も関係する。

明白になる主なアクセス形態は、取り出しであろう。

付　録　2

ツ　リ　ー　の　構　造



付図２－１／ＪＴ－Ｘ５００　ツリーの構造
(ITU-T X.501)

ツリーは「節点」と呼ばれる点および「枝」と呼ばれる有向線分の集合である。各枝ａは、ある節点Ｖからある節点Ｖ′へ向かう、と表現できる。例えば、上図のツリーは７つの節点（V^1 ～V^7）と６つの枝（a^1 ～a^6）を持つ。

ある枝ａが節点Ｖから節点Ｖ′へ向かう時、節点Ｖをその枝の「開始」節点、節点Ｖ′を「終了」節点と呼ぶ。例えば、V^2 とV^3 はそれぞれ枝a^2 の開始節点、終了節点である。開始節点を複数の枝が共有することはあるが、終了節点を複数の枝が共有してはならない。例えば、枝a^1 とa^3 は開始節点V^1 を共有しているが、どの２つの枝も終了節点を共有していない。

どの枝に対しても終了節点になっていない節点は、ツリーの「ルート」と呼ぶ。上図の例では、V^1 がルートである。

どの枝に対しても開始節点になっていない節点は、ツリーの「リーフ」と呼ぶ。上図の例では、V^3，V^6，V^7 がリーフである。

節点Ｖから節点Ｖ′への有向経路はa^1、a^2、・・・、a^n（$n \geq 1$）の集合である。ここにＶは枝a^1 の開始節点であり、Ｖ′は枝a^n の終了節点であり、かつ枝a^k の終了節点は枝a^{k+1} の開始節点（$1 \leq k \leq n$）であるものとする。例えば、節点V^1 からV^6 への有向経路は枝a^3、a^4、a^5 の集合である。「経路」という用語は、「ルート」から「ある節点」への方向性のある道筋を表すと理解されるべきである。

付　録　3

名　前　付　け　規　準

本標準の情報枠組みは非常に一般的であり、ＤＩＴ中のエントリと属性に関して任意の様々な形態を認めている。既に定義したように、名前はＤＩＴ上のパスに密接に関係しているので、名前においても任意の様々な形態が可能である。この節で名前付けについて検討されるべき規準を示す。この規準はＴＴＣ標準ＪＴ－Ｘ５２０に現れる標準の名前の形式を設計する時に使われるが、標準化された名前の形式が適用されないオブジェクトの名前付けもされるべきである。

現在のところ、利用者の親しみやすさという規準についてのみ着手されている。

（注）全ての名前が利用者に親しみやすい必要はない。

１．利用者への親しみやすさ

人間が直接扱わなければならない名前は親しみやすいべきである。利用者に親しみやすい名前は人間というユーザ側の観点に立つものであって、計算機側の観点に立つものではない。計算機が解釈しやすいことよりもむしろ、人が推論しやすく、思い出しやすく、かつ理解しやすいことにある。

利用者への親しみやすさの目標を次の二つの原則によって、より厳密に述べることができる：

- オブジェクト固有の情報にもとづいてそのオブジェクトの利用者に親しみやすい名前を人間が正しく推測できるべきである。例えば、ビジネス上の付き合いをごく普通にしていれば知り得る情報だけからその人の名前を思い出せるべきである。
- オブジェクトの名前が曖昧に指定されている時、ディレクトリはその名前がある特定のオブジェクトを識別していると結論するよりもむしろ曖昧であることを認識できるべきである。例えば、同じ苗字を持つ二人がいた時、苗字だけではどちらの人かを区別する情報に欠けると考えるべきである。

次の二次目標は上記の利用者への親しみやすさの目標から導かれる：

(1)　名前が自然な曖昧さを無理に取り除くべきではない。例えば、“佐藤”という姓を持つ二人がいた時、“W佐藤”とか“佐藤２”というような答えを要求すべきでない。むしろ名前をつける時の世間のしきたりに従って、それらのエンティティを区別する

利用者に親しみやすい方法を与えるべきである。例えば、姓に加えて名を加える。

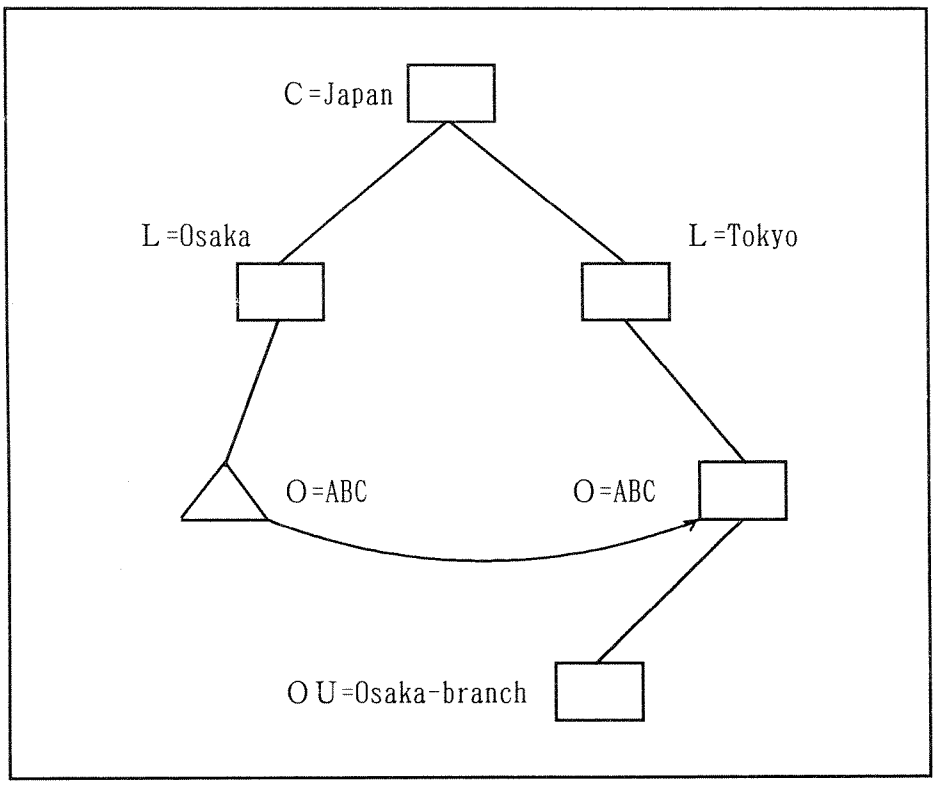
(2) 名前には通常の略語と綴りの通常の変形を認めるべきである。例えば、ある人が、Conway Steel株式会社に勤めていて、その会社が有名であったとしよう。この場合、“Conway Steel株式会社”、“Conway Steel㈱”、“Conway Steel”、および“ＣＳＣ”の何れも今調べている組織を識別するに十分でなければならない。

(3) 特定のエントリの探索を指示するのに、より利用者に親しみやすくしたり探索範囲を縮めるために別名を使うことができる。付図４－１／ＪＴ－Ｘ５００別名の例に示すように、大阪支店を｛C=Japan, L=Osaka, O=ABC, OU=Osaka-branch｝という名前で識別できる。

(4) 名前が複数の部分から成っているならば、その必須部分と任意部分の何れも比較的小さく、従ってまた思い出しやすいことが必要である。

(5) 名前が複数の部分から成り立っているならば、それらの部分の正確な出現順序は一般的に重要でないことが必要である。

(6) 利用者に親しみやすい名前に計算機アドレスが含まれるべきではない。



付図４－１／ＪＴ－Ｘ５００　別名の例
(ITU-T X.501)

付 録 4

用 語 対 照 表

【あ】

ｱﾄﾞﾐﾆｽﾄﾚｰｼｮﾝﾃﾞｨﾚｸﾄﾘ 管理領域	Administration Directory Management Domain
エントリ	entry
応用エンティティ	application-entity
応用サービス要素	application-service element
応用層	Application Layer
応用プロセス	application process
応用プロトコルデータユニット	application protocol data unit
オブジェクト	object
オブジェクトクラス	object class

【か】

下位オブジェクト	subordinate object
管理機関	Administrative Authority

【さ】

サブクラス	subclass
識別名	distinguished name
私設管理領域	Private Management Domain
私設ディレクトリ管理領域	Private Directory Management Domain
上位エントリ	superior entry
上位オブジェクト	superior object
属　性	attribute
属性型	attribute type
属性値	attribute value
スキーマ	schema
スーパクラス	superclass

【た】

直接上位エントリ	immediately superior entry

直接上位オブジェクト	immediately superior object
直接上位の	immediate superior
ツリー	tree
ＤＩＴ構造規則	DIT Structure Rule
ディレクトリ	the Directory
（ディレクトリ）エントリ	(Directory) entry
ディレクトリ管理領域	Directory Management Domain (DMD)
ディレクトリシステムエージェント	Directory System Agent (DSA)
ディレクトリ情報ツリー	Directory Information Tree (DIT)
ディレクトリスキーマ	Directory Schema
ディレクトリ情報ベース	Directory Information Base
（ディレクトリ）ユーザ	(Directory) user
ディレクトリユーザエージェント	Directory User Agent (DUA)

【な】

名　前	name

【は】

別　名	alias
別名エントリ	alias entry

【ら】

ルート	root