TTC 標準
TTC STANDARD

JT-Q1218
インテリジェントネットワーク
網間インタフェース

〔Inter-network Interface for Intelligent Network〕

第3版

1996年4月24日制定

社団法人
情報通信技術委員会

THE TELECOMMUNICATION TECHNOLOGY COMMITTEE

＜参考＞

１．国際勧告との関係

本標準は、1995 年に改版された ITU-T 勧告 Q.1214、Q.1215、Q.1218、Q.1290 をベースとし、1996 年１月／２月の ITU-T SG11 会合における勧告草案 Q.1224、Q11228 の審議結果に基づいて、IN 構造網間のインタフェースを定めたものである。ITU-T における勧告草案 Q.1224、Q.1228 の審議は、IN 能力セット 2（CS-2）に関するものであり、本 TTC 標準において規定する IN 能力セットは、ITU-T IN CS-2 に部分的に準拠していることを考慮し、本文中は「CS-1（改）」と呼称する。

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

無し

２．２　ナショナルマター項目

無し

２．３　その他

(1)　本標準は上記 ITU-T 勧告に対して、以下の項目を削除している。

(a) Q.1214：4.2 節　SSF/CCF モデル
(b) Q.1214：4.3 節　SRF モデル
(c) Q.1214：5 章　　サービス非依存ビルディングブロックのステージ２記述
(d) Q.1214：6.4 節　SCF－SSF 相互関係
(e) Q.1214：6.5 節　SCF－SRF 相互関係
(f) Q.1214：6.7 節　情報フローと関連する SIBs のまとめ
(g) Q.1214：参考 A　コールセグメント間の通信
(h) Q.1214：参考 B　SSF/CCF 相互関係シナリオ
(i) Q.1214：付属資料 I　能力セット１に対して今後の検討課題とされた
　　分散機能プレーンの側面
(j) Q.1215：5.3.1 節　SCP-SSP インタフェース
(k) Q.1215：5.3.2 節　AD-SSP インタフェース
(l) Q.1215：5.3.3 節　IP-SSP インタフェース
(m)Q.1215：5.3.4 節　SN-SSP インタフェース
(n) Q.1215：5.3.5 節　SCP-IP インタフェース
(o) Q.1215：5.3.6 節　AD-IP インタフェース
(p) Q.1215：5.3.8 節　ユーザインタフェース
(q) Q.1218：2.1 節　SSF/SCF, SCF/SRF, SSF/SRF インタフェース
(r) Q.1218：3.1.1 節　SSF アプリケーションエンティティ手順
(s) Q.1218：3.1.2.4 節　SCF 管理エンティティ部分の状態遷移図
(t) Q.1218：3.1.2.5.1 節　状態１″空″
(u) Q.1218：3.1.2.5.2 節　状態２″ SSF 指示準備中″
(v) Q.1218：3.1.2.5.3 節　状態３″リソースへのルーティング″
(w)Q.1218：3.1.2.5.4 節　状態１″ユーザ相互動作″
(x) Q.1218：3.1.3 節　SRF アプリケーションエンティティ手順

(y) Q.1218：参考資料Ｉ　能力セット１に対して今後の検討課題とされた
ネットワークインタフェースの側面

上記項目を削除した理由は、インテリジェントネットワークの網内のモデル化もしくはインタフェースに関する規定であり、網間を規定する TTC 標準の範囲外としたためである。

また、上記以外にも、網間に関わらない部分の記述を削除してある。

２．４　原勧告と章立ての構成比較表

上記国際勧告との章立ての構成の相違を下表に示す。

分散機能プレーン編（Q.1214 および草案 Q.1224 対応）

ＴＴＣ標準	ＩＴＵ－Ｔ勧告	備考
１章　概要	１章　概要	
２章　範囲	２章　範囲	部分削除
３章　網間分散機能モデル	３章　分散機能モデル	部分削除
４章　機能エンティティモデル	４章　機能エンティティモデル	部分削除
－－－	５章　サービス非依存ビルディングブロックのステージ２記述	
５章　FE 間の相互関係	６章　FE 間の相互関係	部分削除 部分追加
－－－	ANNEX A、B	
－－－	APPENDIX I II	

物理プレーンアーキテクチャ編（Q.1215 対応）

ＴＴＣ標準	ＩＴＵ－Ｔ勧告	備考
１章　概要	１章　概要	
２章　要求条件と仮定	２章　要求条件と仮定	
３章　物理エンティティ	３章　物理エンティティ	部分削除
４章　マッピングの要求条件	４章　マッピングの要求条件	
５章　分散機能プレーンの物理プレーンへのマッピング	６章　分散機能プレーンの物理プレーンへのマッピング	部分削除

インタフェース編（Q.1218 および草案 Q.1228 対応）

ＴＴＣ標準	ＩＴＵ－Ｔ勧告	備考
概要	概要	
０章　序論	０章　序論	部分削除
１章　SACF／MACF 規則	１章　SACF／MACF 規則	
２章　IN 能力セット１（改）のアプリケーションプロトコルの抽象構文	２章　アプリケーションプロトコルのアブストラクトシンタックス	部分削除 部分追加
３章　手順	３章　手順	部分削除 部分追加
－－－	Annex A	
付属資料 A　IN の定義で用いられる用語集	－－－	Q.1290 より
付録Ｉ　サービスデータのモデル化	APPENDIX I	
－－－	APPENDIX II	
付録 II　認証の枠組み	－－－	部分追加
付録 III　ISPT のためのオブジェクトモデリング	－－－	
付録 IV　SCSM 及び SDSM プロセスの記述		

３．改版の履歴

版　数	制　定　日	改　　版　　内　　容
第１版	１９９４年１１月２４日	制　定
第２版	１９９５年１１月２８日	ITU-T 勧告 Q.1214、Q.1215、Q.1218、Q.1290 が改版されたことに伴い内容を充実させた。
第３版	１９９６年　４月２４日	ISPT をベースとした PHS ローミングを実現するための IN 網間インタフェースの拡張 特に、SDF／SDF インタフェースの追加（注）

（注）本標準は ITU 勧告草案 Q.1224, Q.1228 をベースに、以下の項目を追加している。

(a) 第１編　５.５節　　ＳＤＦ／ＳＤＦ相互関係

(b) 第３編　２.２節　　ＳＤＦ－ＳＤＦインタフェース

(c) 第３編　付録 III　　ISPT のためのオブジェクトモデリング

４．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。

５．その他

(1) 参照している勧告、標準等

ＴＴＣ標準：　JT-Q701(2 版)、
JT-Q711(1 版)、JT-Q712(1 版)、JT-Q713(2 版)、JT-Q714(1 版)、
JT-Q762(6 版)、JT-Q763(6 版)、
JT-Q771(1 版)、JT-Q772(1 版)、JT-Q773(1 版)、JT-Q774(1 版)、
JT-Q932(2 版)
JT-X500(2 版)

ＩＴＵ－Ｔ勧告：I.130(1993)
Q.775(1993)
Q.1200(1993)、Q.1201(1993)、Q.1204(1993)、Q.1205(1993)、
Q.1208(1993)、Q.1211(1993)　Q.1290(1995)、
X.25(1993)、
X.200(1993)、
X.500(1993)、X.501(1993)、X.509(1993)、X.511(1993)、　X.518(1993)、
X.519(1993)、　X.525(1993)、
X.680(1994)、X.681(1994)、X.682(1994)、X.683(1994)、
X.690(1994)、X.880(1994)

ＩＳＯ標準：　IS 9545

目　　次

第１編　インテリジェントネットワーク網間ＣＳ－１（改）の分散機能プレーンアーキテクチャ編

第２編　インテリジェントネットワーク網間ＣＳ－１（改）の物理プレーンアーキテクチャ編

第３編　インテリジェントネットワーク網間ＣＳ－１（改）のインタフェース編

第１編　インテリジェントネットワーク網間ＣＳ－１（改）の分散機能プレーンアーキテクチャ編

１．概要

分散機能プレーン（Distributed Functional Plane: DFP）の一般的な側面は、Q.1204 の 1 章に含まれている。

２．能力セット１（改）でのＩＮ分散機能プレーンの範囲

ＩＮの能力セット 1（改）（Capability Set 1 -rev.: CS-1-rev.）のＩＮ－ＤＦＰアーキテクチャの範囲は、網間の能力を要求するサービスによってもたらされ、発展可能な網技術の組み込まれた基本能力によって制約される。望まれたＣＳ－１（改）サービスをサポートするのに必要な機能の範囲は以下を提供する機能を含む。

- 網間でのサービス制御

これらの側面の範囲は、以下に述べられる。

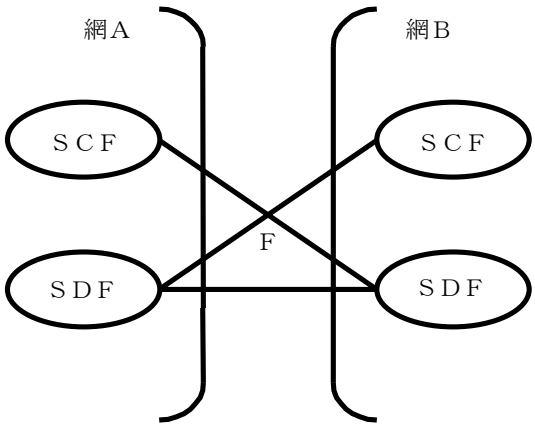
2.1　網間でのサービス制御

網間での情報の送信や受信のための網相互動作は、サービス制御機能から起動され、サービスデータ機能によりサポートされる。このサービス制御の起動が、各網でどのように実現されるかについては、本標準の対象外であり、各網毎の実現方法によりサポートされる。

３．能力セット１（改）のための網間分散機能モデル

3.1　図の説明

図１－３－１／ＪＴ－Ｑ１２１８はＣＳ－１（改）のためのＩＮ－ＤＦＰモデルを示す。
この図は、網間のＣＳ－１（改）に適用できる機能エンティティと相互関係を描いている。
この図は、網間のＩＮアーキテクチャを規定するために設けられ、ＩＴＵ－Ｔ勧告Ｑ．１２０４の２章で述べられた一般的なＩＮ－ＤＦＰモデルのサブセットである。
機能エンティティ、相互関係、図の一般的な説明は、ＩＴＵ－Ｔ勧告Ｑ．１２０４の２．１章に含まれている。



凡例
　ＳＣＦ　サービス制御機能（Service Control Function）
　ＳＤＦ　サービスデータ機能（Service Data Function）

両網の関係は、双方向にＳＣＦ－ＳＤＦの関係を持つことが必須では無い。

図１－３－１／ＪＴ－Ｑ１２１８
ＣＳ－１（改）の網間相互動作に関わるＩＮ分散機能プレーンモデル

3.2　ＩＮ機能モデル

第１編３．１章で述べたように、ＣＳ－１（改）の網間のためのＩＮ－ＤＦＰは一般的なＩＮ－ＤＦＰのサブセットである。特に、

－　ＳＣＦ、ＳＤＦ機能エンティティだけが含まれている。
－　図に示したようにＩＮサービスの実行に関連する相互関係だけが述べられている。
－　各機能エンティティのサービス管理と監督の側面が含まれている。しかしながら、ＣＳ－１（改）に特化して述べているわけではない。機能エンティティと対応したサービス管理機能の各種監督のインプリメンテーションを制約するためのＣＳ－１（改）での試みがされているのではない。

3.3　ＩＮサービスの実行に関連する機能エンティティの定義

ＳＣ機能（ＳＣＦ）

ＳＣＦは、ＩＮ提供かつ／またはカスタムサービスの要求の処理において、網間の相互動作を要求される場合に、呼／サービス論理インスタンスを処理するために要求された情報（サービスまたはユーザデータ）を得るために他の機能エンティティと相互動作する。

a) サービスデータ機能（Service Data Function：SDF）機能エンティティとインタフェースを持ち相互動作する。
b) それは、ＩＮ提供サービスへの要求を取り扱うのに必要な論理と処理能力を有する。

ＳＤ機能（ＳＤＦ）

ＳＤＦは、ＩＮ提供サービスの実行の中で網間の相互動作が要求される場合に、ＳＣＦやＳＤＦによる実時間アクセスのためのカスタマと網のデータを含む。それは、要求に応じてＳＣＦやＳＤＦとインタフェースを持ち相互動作する。

注：ＳＤＦは、ＩＮ提供サービスの提供または操作に直接関連するデータを含んでいる。したがって、それはクレジット情報のような第３者によって提供されたデータを含む必要はないが、これらのデータへのアクセスを提供するだろう。

４．機能エンティティ呼／サービス論理処理モデル

4.1　概要

IN 呼／サービス論理処理は、ＳＣＦでのサービス処理、ＳＤＦでのデータの利用をそれぞれ含む。この章では、網間での相互動作の機能レベルのモデリングという形で、この処理を述べる。網内の呼とコネクションの処理や各種リソースの利用については本仕様の対象外であり、各網において実現される。

－　サービス処理モデリングは、サービスおよびベンダ／インプリメンテーション非依存な高いレベルの抽象化レベルを提供する。

－　サービス処理のモデリングは、ＳＣＦにとってアクセス可能なＳＤＦの動作とリソースの抽象化と同様にこのサービス論理実行をサポートするために必要なＳＣＦの動作とリソースの抽象化を提供する。

モデリングは、ＳＣＦ、ＳＤＦの動作とリソースの外部からの見え方を提供するだけであるので、このモデリングは機能エンティティモデル要素と一対一のマッピングとして製品の中に機能エンティティのインプリメントをベンダの義務として含んでいない。

この章のモデリングは、３章／Q.1204 で記述されたモデリングの目的、仮定、アーキテクチャに基づいている。また、CS-1 に適用可能である限りにおいて、その付録で示されたツールを利用している。

4.2　サービス制御機能（ＳＣＦ）モデル

4.2.1　概要

ＳＣＦのモデルは図 1-4-1/JT-Q1218(ITU-T Q.1214)に示されている。このモデルの目的は、ＳＣＦに関する網間の相互動作の枠組みを提供することである。

サービス制御機能（ＳＣＦ）の網間相互動作に関わる機能は、サービス論理処理プログラム（SLP）の形で提供されるサービス論理の実行に基づく他網内データのアクセスであり、それ故、機能エンティティアクセス管理も含んでいる。

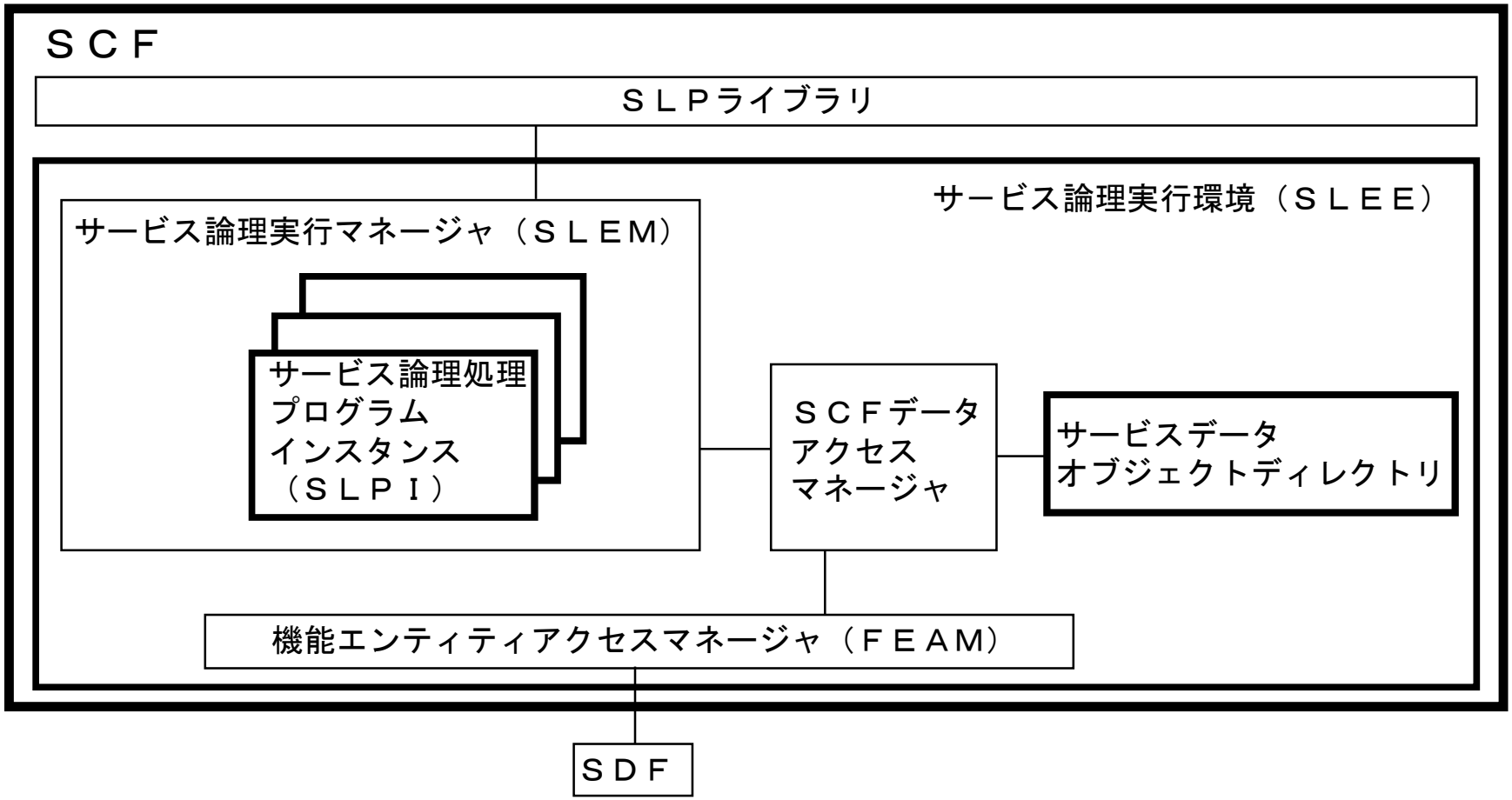


図１－４－１/JT－Q1218(ITU-T Q.1214)　ＳＣＦモデル

4.2.2　ＳＣＦ構成要素

4.2.2.1　概要

上で定義された機能性を実現するために、ＳＣＦモデルが図 1-4-1/JT-Q1218(ITU-T Q.1214)に示されている。この図はＳＣＦの概念モデルを示しており、ＳＣＦの実際のインプリメンテーションを示す意図はないことに注意すべきである。

ＳＣＦプラットフォームは、適切なサービス処理を提供するために、サービス論理処理プログラム(SLP)が走行するサービス論理実行環境（SLEE）を提供する。SLP は SLEE によって起動されるサービスアプリケーションプログラムであり、SLEE の制御下でサービス処理を実現するのに用いられる。複数の SLP の同時起動、実行も SLEE で管理される。

図 1-4-1/JT-Q1218(ITU-T Q.1214)で示された各エンティティは後続の項で記述する。

4.2.2.2　サービス論理実行マネージャ(SLEM)

4.2.2.2.1　概要

SLEM はサービス論理実行動作全体を処理し、制御する機能である。SLEM はサービス論理処理プログラムインスタンス（SLPI）を実現し、これに必要な機能を含む。SLPI 実行を支援するために、SLEM はＳＣＦデータアクセスマネージャ及び機能エンティティアクセスマネージャとも相互動作する。これらの側面に加えて SLEM には次のような機能が必要となる:

－　SLPI を実行し、SLPI に対応した一時的な（SLPI 状態情報のように SLPI の存在期間の間だけ持続する情報）を保持する。

－　ＳＣＦデータアクセスマネージャ（以下の第１編 4.2.2.3 章参照。）を介してＳＤＦへの SLPI アクセスを管理する。

4.2.2.2.2　サービス論理処理プログラムインスタンス（SLPI）

サービス論理処理プログラム（SLP）は、SLEE により起動されサービス処理を実現するために用いられるサービスアプリケーションプログラムである。それは実行されたときサービス実行のフローを制御する論理的構成概念と、サービス実行に必要なネットワークリソースとデータにアクセスするためのステートメントを含んでいる。SLP が選択され、起動されると、それはサービス論理処理プログラムインスタンス（SLPI）と呼ばれる。SLP に対比してそれに対応する SLPI は、サービス実行フローをアクティブに制御する動的なエンティティである。

4.2.2.3　ＳＣＦデータアクセスマネージャ

4.2.2.3.1　概要

ＳＣＦデータアクセスマネージャは、ＳＣＦ内で共有する持続的な情報（例えば、SLPI の存在期間を越えて持続する情報。）の蓄積、管理、アクセスを提供するために必要な機能を提供する。ＳＣＦデータアクセスマネージャはＳＤＦ内のリモート情報にアクセスするのに必要な機能も提供する。ＳＣＦデータアクセスマネージャはこれらの機能を SLPI に提供するために、SLEM と相互動作する。

図 1-4-1/JT-Q1218(ITU-T Q.1214)はＳＣＦデータとして扱われる一つの構造を示す。これらは、次のものを含んでいる:

－　サービスデータオブジェクトディレクトリ

これについては、次の項で記述する。

4.2.2.3.2　サービスデータオブジェクトディレクトリ(SDOD)

図 1-4-1/JT-Q1218(ITU-T Q.1214)はサービスデータオブジェクトディレクトリを示している。それは特定の網間で可視なデータオブジェクトにアクセスするための適切なＳＤＦを指し示すための方法を提供する。

SLEM はＳＤＦのサービスデータオブジェクトにアクセスするため、ＳＣＦデータアクセスマネージャと相互動作する。ＳＣＦデータアクセスマネージャは、SLEM（とその SLPI）に対してトランスペアラントな方法で、ネットワーク間のサービスデータオブジェクトを指し示すために、サービスデータオブジェクトディレクトリを利用する。このようにして、SLEM（とその SLPI）はネットワークのサービスデータオブジェクトをグローバルに一様にみることができる。

4.2.2.4　機能エンティティアクセスマネージャ（FEAM）

機能エンティティアクセスマネージャは、ＳＣＦデータアクセスマネージャがメッセージを介して他の機能エンティティと情報を交換するために必要な機能を提供する。このメッセージ処理機能は次のようであるべきである:

－　SLPI に対してトランスペアレントである。
－　確実なメッセージ転送を提供する。
－　シーケンシャルメッセージ配布を保証する。
－　要求/応答メッセージペアが関連付けられるものであることを許容する。
－　複数のメッセージが相互に関連付けられるものであることを許容する。
－　OSI 構造及び原則に従う。

4.3 サービスデータ機能（ＳＤＦ）モデル

4.3.1 概要

ＳＤＦのモデルは図 1-4-2/JT-Q1218(ITU-T Q.1214)に示されている。このモデルの目的は、ＳＤＦに関するサービスデータ機能の枠組みを提供することである。

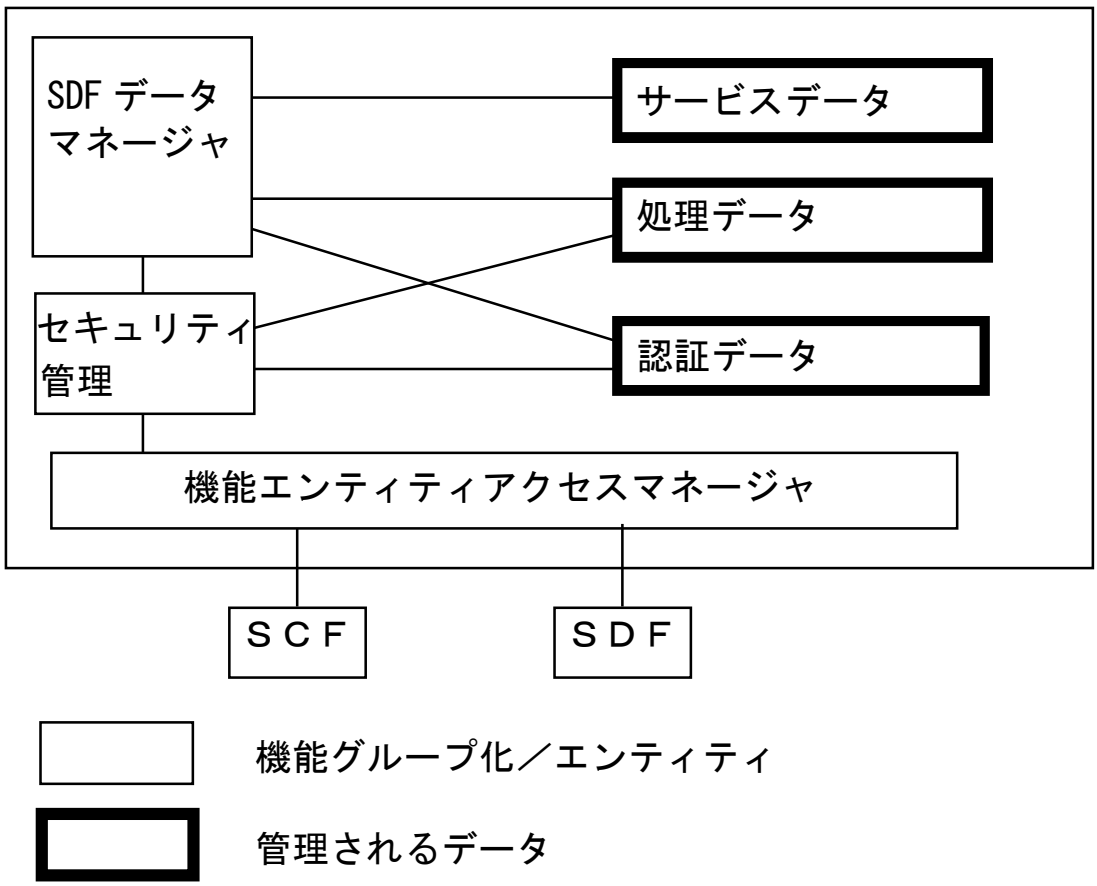


図 1-4-2/JT-Q1218 (ITU-T Q.1214)
SDF モデル

4.3.2 章では詳細なＳＤＦアーキテクチャを記述し、4.3.3 章ではＳＤＦで扱われるデータ種別を明らかにし、かつ分類する。

ＳＤＦはサービス論理処理プログラム（SLP）に関係するデータ、また、SLP インスタンス（SLPI）の実行においてアクセスされるデータを持ち、これを管理する。

4.3.2　ＳＤＦ構成要素

4.3.2.1　概要

上記の定義された機能を実現するために、ＳＤＦモデルが図 1-4-2/JT-Q1218　(ITU-T Q.1214)に示されている。図 1-4-2/JT-Q1218(ITU-T Q.1214)に示した各機能エンティティは、後続の項において記述する。これは特定のインプリメンテーションを意味するわけではない。

4.3.2.2　ＳＤＦデータマネージャ

ＳＤＦデータマネージャはＳＤＦ内の情報を蓄積、管理し、これにアクセスするために必要な機能を提供する。

ＳＤＦデータマネージャは、ＳＣＦからアクセスされるデータモデルを提供するとともに、データモデルで規定された情報構造に対する検索、更新、比較等の処理を実現し、ＳＤＦ内に格納されている情報の管理と物理的構成との対応付けを行う。

例えば、もしデータがデータベースとして物理的に構成されていれば、ＳＤＦデータマネージャは SQL のようなデータベースアクセス言語も処理する。

通常のデータアクセス機能に加えて、以下に示す具体的な機能を提供する。

ＳＤＦ内に格納される情報（属性）をＳＣＦから検索する場合、ＳＤＦマネージャの保有する知識情報（応用に依存）により、その属性値の全体、または一部を検索結果として返す機能を提供する。また、 既に検索結果として返答した属性値については、「使用済みの属性値」として管理する機能を保有する。更に、未使用の属性値の数がある一定値（例えば、残数＝０）に達すると、その属性値の補充を要求する機能も保有する。

4.3.2.3　機能エンティティアクセスマネージャ

機能エンティティアクセスマネージャは、ＳＤＦデータマネージャが他の機能エンティティ、ＳＣＦ、ＳＤＦ、ＳＭＦとメッセージを介して情報を交換するのに必要な機能を提供する。

このメッセージ処理機能は次のようであるべきである:

－　確実なメッセージ転送を提供する。
－　シーケンシャルメッセージ配布を保証する。
－　メッセージの要求/応答ペアを関連付けられるものであることを許容する。
－　複数のメッセージを相互に関連付けられるものであることを許容する。
－　OSI 構造と原則に従う。

ここでは、ネットワーク内のデータの分散がＳＣＦに対して完全に不可視化されているので、機能エンティティアクセスマネージャは他のＳＤＦにアクセスする。

4.3.2.4　セキュリティ管理

セキュリティ管理は、たとえば、認証の無いユーザへのアクセス拒否のように、ＳＤＦ内にある異なったデータ種別に対する安全なアクセスを提供する。

この機能は次のようであるべきである：

－　ＳＣＦのアクセス権をチェックする。
－　提供された情報によりユーザを認証する。
－　特定のユーザのために認証の試みが失敗した回数を数える。（この機能の実行をＳＤＦで実現できるかどうかは今後の検討課題）
－　データへのアクセスを妨げる。
－　ユーザのアクセス権を指定する。
－　ユーザの要求の間アクセス権を記憶する。
－　特別なデータへアクセスするユーザの権利を制御する。
－　ＳＤＦデータマネージャの要求により、特定のユーザの秘密情報（例えば、暗号鍵）を用いてそのユーザの認証のための情報を生成する。

4.3.3　ＳＤＦで扱われるデータ種別

サービス処理に必要なデータは以下の種別に分類できる。

1）認証データ　－　これらのデータはＳＣＦを通してデータベースにアクセスするユーザの認証に使われる。
　例、ＰＩＮコード、認証失敗のカウンタの値。
　認証データの集合は通常アクセス権のレベルと組み合わされる。
2）処理データ　－　これらのデータはＳＬＰＩには必要無い、しかし、ＳＤＦ自身によって処理と管理の目的のために使われる。
　例、オブジェクトクラスへの参照、アクセス制御データ。
3）サービスデータ　－　これらのデータはサービスの供給のために使われる。
　例、加入者プロフィール、サービス供給者の同意事項。
　これらのデータは必要ならば、いくつかのサービスによって使用することができる。

５．ＦＥ間の相互関係

5.1　概要

この章では、網間に関わる機能エンティティ（FE）間の情報フロー（IF）を記述している。この内容は、情報フローから JT-Q1218 第３編に定義された抽象構文と一貫した形式での、FE から FE へのインタフェースへのマッピングを提供する。

関連する ITU-T 勧告 Q.1204 の 4.1 章は、FE 間の相互関係のアーキテクチャの側面を記述している。

5.2　相互関係

CS-1 の網間の相互動作では、情報フローは次のような相互関係に対して定義される:

SCF-SDF（F）

括弧内の文字は ITU-T 勧告 Q.1211 で定義された機能インタフェースに対応する。

各相互関係に対して次のような情報が与えられている:

i)　相互関係が設定、終結し得る条件。

ii)　相互関係に関与する二つのエンティティ間の、アルファベット順の、情報フロー

各情報フローに関して次のような事柄がリストアップされている。

a.　情報フローの名前

b.　関係している FE 相互関係（例、SCF から SDF、あるいは、SDF から SCF）

c.　情報フローにおける各情報要素（IE）の名前。各 IE についてそれが各 IF に含めなければならない必須（M）のものであるか、あるいは IE が省略される状況があることを示すオプショナル（O）なものであるかが指定される。もしその IE がオプショナルなものであれば、オプショナルであるという条件下の厳密な状況とデフォルト値が与えられる。

d.　各 IE の記述。信号プロトコルにおけるパラメータへの IE のマッピングが、　JT-Q1218 第３編の 2 章で与えられている。

e.　適切であれば、この IF と対応する FE モデルとの間のマッピング。当該 IF が送信、あるいは受信される前（前－条件）と後（後－条件）に伴われる条件の形でこれは記述される。

上記の c.d.e.について、標準で記述するのに適切でない IF については、他の勧告や本標準の他の部分等に対する参照を明記している。

5.3　ＦＥ間の情報フロー

二つの FE 間の情報フローは、要求/応答ペアもしくは要求だけの何れかからなる。情報フローは、物理プレーンにある対応する物理エンティティ間の信号メッセージと一対一にマッピングしていないかもしれないことに注意。

必要があれば、SCF は FE 間の複数の情報フローの協調をとる。ある情報フローの順序関係が述べられている。

二つの FE 間の IF の完全なセットは、これらの FE 間の相互関係を定義する。

必要なところには、他の情報フローの影響を取り消すための特定の情報フローが定められている。

この章において、エラー条件に関連している IF は記述されていないことに注意。

5.4 ＳＣＦ－ＳＤＦ相互関係

5.4.1 概要

SCF が SDF 内に含まれるあるデータを検索または修正する要求をするとき、その SCF の要求時に SCF と SDF 間に相互関係が確立される。相互関係は SDF によって終結される。

SDF に関連する IF は、サポートされるサービスに依存してある程度の処理に対応付けられている。この処理とは、データ操作に関わるもので呼制御に関わるものではない。

SCF はデータの論理的な視え方のみを知っている。IF はデータの物理的な構成またはどのようにデータを蓄積するかについては意味しない。特に、データが複製される事実は SCF にはわからない。

ディレクトリアクセスのプロトコルが本版での物理プレーンに導入されたため、IF はそれらに合わせている。また、多くの IE は X.511 における同等のパラメータに置き換えられている。但しそれらの IE のいくつかは本版の物理プレーンには使用されていない。

5.4.2 SCF-SDF 間の情報フロー

5.4.2.1 認証(Authenticate)

a. FE 相互関係：SCF から SDF

b. 概要

この IF は、SCF-SDF 間に認証された相互関係の確立を要求するために末端のユーザに代わって用いられる。

c. 情報要素

認証された相互関係の Id(Authorized Relationship Id)　　(M)

認証情報(Authentication Information)　(M)

d. IE 記述

認証された相互関係の Id は、オペレーションが適用され得るサービス論理とデータベースとの間に確立される認証された相互関係を識別する。

認証情報は、要求された種別の認証を実行するのに必要な情報を含む。必要な情報はない。認証情報(Authentication Information)は X.511 の 8.1 節における DirectoryBindArgument と同じである。

e. FE モデルへのマッピング

SCF は認証の実行を SDF に要求するために、この情報フローを SDF に送信する。

5.4.2.2 認証結果(Authenticate Result)

a. FE 相互関係：SDF から SCF

b. 概要

この IF は、SCF-SDF 間の認証された相互関係の確立を確認するために末端のユーザに代わって用いられる。

c. 情報要素

認証された相互関係の Id(Authorized Relationship Id)　　(M)

認証情報(Authentication Information)　(O)

d. IE 記述

認証された相互関係の Id については、以前に定義されたものとする。

認証情報は、認証 IF で定義されたものとする。但しここでは当該情報は返却される。

e. FE モデルへのマッピング

SDF は、認証の結果を SCF に通知し、および／または、相互の認証を実行するために、この情報フローを SCF に送信する。

5.4.2.3　認証された相互関係終了(End Authenticated Relationship)

a. FE 相互関係：SCF から SDF

b. 概要

この IF は、SCF-SDF 間の認証された相互関係を終了するために、末端のユーザに代わって SCF により発行される。

c. 情報要素

認証された相互関係の Id(Authorized Relationship Id)　　(M)

d. IE 記述

認証された相互関係の Id については、以前に定義されたものとする。

e. FE モデルへのマッピング

SCF は以前に確立された SCF-SDF 間の認証された相互関係を終了するために、末端のユーザに代わってこの情報フローを SDF に送信する。

5.4.2.4　探索(Search)

a. FE 相互関係：SCF から SDF

b. 概要

探索操作は DIT 中から関連するエントリを探し、そのエントリから選択された情報を読み出すのに用いられる。（例えば、フリーフォン番号の翻訳）

c. 情報要素

認証された相互関係の Id(Authorized Relationship Id)　　(M)

基底オブジェクト(baseObject)　　(M)

部分集合(subset)　(M)

選択(selection)　　(M)

フィルタ(filter)　(O)

一致した値のみの指定(matched Values Only)　　(O)

d. IE 記述

認証された Relationship ID は、オペレーションが適用され得るサービス論理とデータベースとの間に確立される認証された Relationship を識別する。IN 網間能力セット 1 の物理プレーンでは、これは TCAP トランザクション識別子にマッピングされる。

基底オブジェクト（Base Object)は、要求された情報が存在し得る DIB 内の特定のエントリを識別する。基底オブジェクト（Base Object)は、X.511　10.2 節における base Object と同じである。

部分集合（Subset)は、検索がどのレベルで行われるべきかを示す。即ち、基底オブジェクトのみであるか、あるいは基底オブジェクトのうち直接に従属する部分であるか、あるいは基底オブジェクトと全ての従属部分であるかを示す。部分集合（Subset)は X.511　10.2 節における Subset と同じである。

選択（Selection）は、エントリからどのような情報が要求されるかを示す。選択は X.511　7.6 節における EntryInformationSelection と同じである。

フィルタ（Filter)は、基底オブジェクト（Base Object)と部分集合（Subset)により特定される検索空間より関係のないエントリを除外する。フィルタ（Filter)は、X.511　7.8 節における Filter と同じである。

一致した値のみ出力の指定（Matched Values Only)は、ある属性値（attribute values)が返却されたエントリ情報から省略されることを示す。

一致した値のみ出力の指定（Matched Values Only)は、X.511　10.2 節における matched Values Only と同じである。

Search IF によって問い合わせ可能な情報を以下に示す。

- ルーチング　アドレス
- Offnet/Onnet 表示（VPN における着アドレスか否か）
- 期待される authorization 結果
- 期待される確認（verification)結果
- 期待されるスクリーニング結果

注意：ここでいう確認とスクリーニングは認証（Authenticate)IF を使用する場合、認証と関係付けるべきではない。

基底オブジェクト（Base Object),部分集合（Subset),フィルタ（Filter)選択（Selection)となり得る候補の例を以下に示す。

- Called Number
- Pin + ID
- Callig Party Number
- ユーザからの入力（ダイヤル数字）
- 発端末能力
- 発／着　サービスプロファイル
- スクリーニングされたリスト ID + スクリーニングされた情報 ID
- ベアラリソース種別（このキーは、他のキーと結合される）
- 等

初期 DPIF の中の情報要素は、全て基底オブジェクト（Base Object),部分集合（Subset),選択（Selection),フィルタ（Filter)の組を決定するための候補である。この IF の IE の正確な構造及びとりうる値は、サービス仕様に依存する。

e. FE モデルへのマッピング

SCF は、サービスデータオブジェクトを読む（名前が完全にわからないオブジェクトの探索と、与えられたあるクライテリアに従ったオブジェクトの比較）ために、この情報フローを SDF に送信する。

5.4.2.5　探索結果(SearchResult)

a. FE 相互関係：SDF から SCF

b. 概要

この IF は探索(Search)IF に対する応答である。物理プレーンにおいてこの IF は探索(Search)の結果応答(RESULT)に対応付けられることに注意。

c. 情報要素

認証された相互関係の Id(Authorized Relationship Id)　　(M)

探索情報(searchinfo)　　(M)

d. IE 記述

認証された相互関係の Id については、以前に定義されたものとする。

探索情報（Search info)は、探索（Search)IF を使用する時に要求される。これはデータあるいは単なる比較結果を含み得る。

e. FE モデルへのマッピング

SDF は、指定されたサービスデータオブジェクトを探索し、読み出した結果、あるいは比較の結果を返すために、この情報フローを SCF に送信する。

5.4.2.6　エントリ更新(ModifyEntry)

a. FE 相互関係：SCF から SDF

b. 概要

エントリ更新操作は、単一のエントリに対して変更を行うために用いられる。

但し、あるデータへの同時アクセスのような問題は、IF によっては解決されない。

c. 情報要素

認証された相互関係の Id(Authorized Relationship Id)　　(M)

オブジェクト(object)　　(M)

変更順(changes)　(M)

選択(selection)　　(MO)

d. IE 記述

認証された相互関係の Id については、以前に定義されたものとする。

オブジェクト（Object)は、対象とするあるひとつの特定エントリを示す。オブジェクト（Object)は、X.511 7.6 節の Object と同じである。

変更順（Changes)は、その命令の中で適用される更新の順序を示す。更新には属性（Attribute)または値（Values)の追加／削除、値（Values)の変更、値（Value)のデフォルトへの値設定の 6 種類がある。

変更順（Changes）は、X.511 11.3 節に定義される Entry Modification の順序である。（但し、デフォルトへの値設定を除く）

選択（Selection)は、更新を行う場合にオブジェクトに含まれるあるデータを要求するために使用される。選択（Selection)は X.511 7.6 節の Entry Information Selection と同じである。

e. FE モデルへのマッピング

SCF は、サービスデータオブジェクトを変更するために、この情報フローを SDF に送信する。

5.4.2.7　エントリ更新結果(ModifyEntryResult)

a. FE 相互関係：SDF から SCF

b. 概要

この IF はエントリ更新(ModifyEntry)IF に対する応答である。物理プレーンにおいてこの IF はエントリ更新(Modify Entry)の結果応答(RESULT)に対応付けられることに注意。

c. 情報要素

認証された相互関係の Id(Authorized Relationship Id)　　(M)

情報(information)　(O)

d. IE 記述

認証された相互関係の Id については、以前に定義されたものとする。

情報は、要求されたオペレーションの結果を指す。例えば成功または失敗（理由付き）。（情報が空でない場合、それは更新結果を含む。即ち、変更されたデータ、あるいは先記のエントリ更新（Modify Entry)IF 時に要求されたある情報を含むこともある。）

e. FE モデルへのマッピング

SDF は、指定されたサービスデータオブジェクトへの変更結果を返すために、この情報フローを SCF に送信する。

5.4.2.8　エントリ追加(AddEntry)

a. FE 相互関係：SCF から SDF

b. 概要

エントリ追加操作は、葉エントリを SDF の DIT に追加するのに用いられる。

c. 情報要素

認証された相互関係の Id(Authorized Relationship Id)　　(M)

オブジェクト(object)　　(M)

エントリ(entry)　(M)

d. IE 記述

認証された相互関係の Id については、以前に定義されたものとする。

エントリは、生成される属性情報（attribute Information)(の組）を含む。エントリ（Entry)は X.511 11.1 節の entry と同じである。

e. FE モデルへのマッピング

SCF は、SDF の DIT に葉エントリを追加するために、この情報フローを SDF に送信する。

5.4.2.9　エントリ追加結果(AddEntryResult)

a. FE 相互関係：SDF から SCF

b. 概要

この IF はエントリ追加(AddEntry)IF に対する応答である。物理プレーンにおいてこの IF はエントリ追加(AddEntry)の結果応答(RESULT)に対応付けられることに注意。

c. 情報要素

認証された相互関係の Id(Authorized Relationship Id)　　(M)

d. IE 記述

認証された相互関係の Id については、以前に定義されたものとする。

e. FE モデルへのマッピング

SDF は、SDF の DIT への葉エントリ追加の結果を与えるために、この情報フローを SCF に送信する。

5.4.2.10　エントリ削除(RemoveEntry)

a. FE 相互関係：SCF から SDF

b. 概要

エントリ削除操作は、SDF の DIT から葉エントリを削除するのに用いられる。

c. 情報要素

認証された相互関係の Id(Authorized Relationship Id)　　(M)

オブジェクト(object)　　(M)

d. IE 記述

以前に定義されたものとする。

e. FE モデルへのマッピング

SCF は、SDF の DIT から葉エントリを削除するために、この情報フローを SDF に送信する。

5.4.2.11　エントリ削除結果(RemoveEntryResult)

a. FE 相互関係：SDF から SCF

b. 概要

この IF はエントリ削除(RemoveEntry)IF に対する応答である。物理プレーンにおいてこの IF はエントリ削除(RemoveEntry)の結果応答(RESULT)に対応付けられることに注意。

c. 情報要素

認証された相互関係の Id(Authorized Relationship Id)　　(M)

d. IE 記述

以前に定義されたものとする。

e. FE モデルへのマッピング

SDF は、SDF の DIT からの葉エントリ削除の結果を与えるために、この情報フローを SCF に送信する。

5.5　ＳＤＦ－ＳＤＦ相互関係

5.5.1　概要

ＳＤＦ－ＳＤＦ相互関係には２つの目的がある。ＳＤＦ－ＳＤＦ相互関係はデータの位置に関する透過性とデータ分散の効率の双方をサポートすべきである。

これらの２つのＳＤＦ－ＳＤＦ相互関係の使用法は、２種類のオペレーションに対応する。まず第一に、データの分散を隠蔽する際に、あるＳＤＦに送られたオペレーションは、関連するデータを保持する他のＳＤＦにＳＤＦ－ＳＤＦインタフェースを用いて転送され得る。この場合、ＳＤＦ－ＳＤＦインタフェースを用いた手順はＳＣＦ-ＳＤＦ手順の連鎖のために用いられる。ＳＤＦ－ＳＤＦ相互関係がデータ分散を管理し、かつデータ処理の総体的な効率を高めるために用いられる場合には、ＳＤＦ－ＳＤＦインタフェースはあるＳＤＦから別のＳＤＦにデータを転送するために用いられる。この場合、これらの手順は転送およびコピー手順である。

ＳＤＦ－ＳＤＦ相互関係は，呼処理中のみならず、実際に呼が存在しない場合（非呼関連）のサービスのための情報フローをサポートする。大抵の場合、非呼関連の動作は、ターミナルモビリティ、パーソナルモビリティのための登録、認証、暗号化、及びハンドオーバ手順をサポートするためのものである。

ＳＤＦ－ＳＤＦ相互関係は、網間接続をサポートする相互関係の集合の一部である。本相互関係はネットワーク特有構造を隠蔽し、かつ他公衆網からのアクセスセキュリティを確保しながら、ネットワークへの相互接続ポイントを提供する。

ＳＤＦ－ＳＤＦ相互関係は，以下の２つの目的のために用いられる。どちらの目的も網内・網間の場合に適用しうる。

ａ）　ＳＤＦ間の保証されたデータの取得：　この場合、ＳＣＦによりサービスデータアクセスを完結するために、ＳＤＦは自身が保持していないデータを本相互関係を用いて要求する。本相互関係における重要な点は、ネットワークの独立性（isolation）と相互接続をサポートするためのセキュリティが必要なことである。
ｂ）　ＳＤＦ間の保証されたサービスデータの複製：　この場合、ＳＤＦは別のＳＤＦによって管理されているサービスデータの一部を複製する。本相互関係における重要な点は、一旦データが複製された場合には、そのサービスデータに対するオリジナルセキュリティ及びアクセス情報を保持しておくこと、およびコピーが不要になった場合にはデータのコピーを削除することを保証することである。

5.5.2　ＳＤＦ－ＳＤＦ間の情報フロー

5.5.2.1　認証（Authenticate）

ａ．　ＦＥ相互関係：ＳＤＦからＳＤＦ

ｂ．　概要

この手順は一つのＳＤＦ－ＳＤＦ相互関係に含まれる２つのＳＤＦの識別と認証のために用いられる。この手順はＳＤＦ－ＳＤＦ間インタフェース上のどのような手順にも先立って用いられ、データベース間のアクセス制御を実施するために用いられる。

ｃ．　情報要素

認証情報（Authentication Information）　　(M)

ｄ．　ＩＥ記述

認証情報は要求された種類の認証を実行するために必要な情報を含む。情報が必要でない場合もありうる。認証情報はＸ．５１１の８．１節における Directory Bind Argument と同じである。

ｅ．　ＦＥモデルへのマッピング

起動ＳＤＦは認証を実行するために相互動作するＳＤＦに対してこの情報フローを送信する。

5.5.2.2　認証結果（Authenticate Result）

ａ．　ＦＥ相互関係：ＳＤＦからＳＤＦ

ｂ．　概要

この手順は相互動作ＳＤＦが認証結果を確認するために用いられる。

ｃ．　情報要素

認証情報（Authentication Information)　　(M)

ｄ．　ＩＥ記述

認証情報は以前に定義されたとおり、認証手順の結果を提供するために用いられる情報を含む。

ｅ．　ＦＥモデルへのマッピング

応答ＳＤＦは認証を実行するために相互動作しているＳＤＦにこの情報フローを送信する。

5.5.2.3　連鎖要求（Chaining Request）

ａ．　ＦＥ相互関係：ＳＤＦからＳＤＦ

ｂ．　概要

この手順は起動ＳＤＦから応答ＳＤＦに対して、起動ＳＤＦでは実行できなかったデータベース要求を転送するために用いられる手順である。応答ＳＤＦはデータベース要求を実行するために必要なデータを保持するＳＤＦとして起動ＳＤＦから認識される。起動ＳＤＦのデータベースからの指示が応答ＳＤＦへの連鎖手順の送信の契機となる。

ｃ．　情報要素

連鎖引数（Chained Argument）　　(M)

セキュリティパラメータ（Security Parameters）　　(M)

ｄ．　ＩＥ記述

連鎖引数は、ＳＣＦ-ＳＤＦインタフェースのデータベースオペレーションの引数を含む。すなわち、転送されたオペレーションの引数である。

セキュリティパラメータはＸ．５１１に定義されているようにディレクトリオペレーションに関連する様々なセキュリティフィーチャのオペレーションを決定する。

ｅ．　ＦＥモデルへのマッピング

起動ＳＤＦは連鎖オペレーションを実行するために相互動作するＳＤＦにこの情報フローを送信する。これにはＳＤＦ間で実行された認証の方法に関する表示が含まれる。

5.5.2.4　連鎖結果（Chaining Result）

ａ．　ＦＥ相互関係：ＳＤＦからＳＤＦ

ｂ．　概要

この手順は応答ＳＤＦから起動ＳＤＦに対して起動ＳＤＦで実行できなかったデータベース要求の実行要求に対する結果を送るために用いられる。

ｃ．　情報要素

連鎖された結果（Chained Result）　　(M)

セキュリティパラメータ（Security Parameters）　　(M)

ｄ．　ＩＥ記述

連鎖された結果は連鎖要求の結果を与える。

セキュリティパラメータはＸ．５１１に定義されているようにディレクトリオペレーションに関連する様々なセキュリティフィーチャのオペレーションを決定する。これにはＳＤＦ間で実行された認証の方法に関する表示が含まれる。

ｅ．　ＦＥモデルへのマッピング

応答ＳＤＦは連鎖オペレーションを実行するために相互動作する起動ＳＤＦにこの情報フローを送信する。

5.5.2.5　コピー要求（Copy Request）

ａ．　ＦＥ相互関係：ＳＤＦからＳＤＦ

ｂ．　概要

この手順は応答ＳＤＦに含まれているデータを起動ＳＤＦにコピーするために用いられる。データのコピー（もしくはコピーの一部）は定期的に更新されるか、もしくは全く更新されなくてもよい。コピーのメンテナンスはコピー更新手順を用いて行われる。コピー手順は、網イベントあるいはサービスイベントにより要求される。この手順は、起動ＳＤＦ内にマスタコピーの位置についての情報を必要とする。

ｃ．　情報要素

複製領域（Replication Area）	(M)
保守対象部分（Maintained Part）	(M)
更新モード（Update Mode）	(M)
更新方法（Update Strategy）	(M)
マスタ（Master）	(M)

ｄ．　ＩＥ記述

複製領域は応答ＳＤＦのデータベースにおいて起動ＳＤＦのデータベースに複製されるべき部分を示す。

保守対象部分は定期的にメインテインされるべきコピーの部分を示す。

更新モードはコピーの更新時期（変更時あるいは周期的）と更新の主導権をだれが持つかを表示する。

更新方法はすべてのコピーを返送すべきか、あるいは変更部分のみを返送するかを表示する。

マスタはマスタの名称を与える。本情報要素はコピー元を認識するために用いられる。

ｅ．　ＦＥモデルへのマッピング

起動ＳＤＦはコピーを要求するために応答ＳＤＦに対してこの情報フローを送信する。

5.5.2.6　コピー結果（Copy Result）

ａ．　ＦＥ相互関係：ＳＤＦからＳＤＦ

ｂ．　概要

本手順は応答ＳＤＦに含まれるコピーデータを起動ＳＤＦに提供するために用いられる。

ｃ．　情報要素

複製データ（Replicated Data）　　(M)

ｄ．　ＩＥ記述

複製データは起動網に置かれるべきコピーを含む。

ｅ．　ＦＥモデルへのマッピング

応答ＳＤＦはコピーを提供するために本情報フローを起動ＳＤＦに送信する。

5.5.2.7　認証終了（End Authenticate）

ａ．　ＦＥ相互関係：ＳＤＦからＳＤＦ

ｂ．　概要

本情報フローは２つのＳＤＦ間の認証された相互関係を終了するために、ＳＤＦによって発行される。

ｃ．　情報要素

なし

ｄ．　ＩＥ記述

なし

ｅ．　ＦＥモデルへのマッピング

起動ＳＤＦは以前の認証された相互関係を終了するためにＳＤＦに対して本情報フローを送信する。

5.5.2.8　コピー更新（Update Copy）

ａ．　ＦＥ相互関係：ＳＤＦからＳＤＦ

ｂ．　概要

この情報フローは、選択された更新モードがコピーの更新が送信されるべきであることを示している（例えば、応答する網のコピーの変更）場合、コピーが提供されているＳＤＦに含まれるコピーを保守するために用いられる。送信されるべき情報のフォーマットはコピー要求手順の更新方法情報要素に含まれる表示にしたがう。

ｃ．　情報要素

リフレッシュされる情報（Refreshed Information）　　(M)

ｄ．　ＩＥ記述

リフレッシュされる情報はコピー要求手順の更新方法（update strategy）情報要素で指定されたフォーマットにしたがったコピーの更新分を含む。

ｅ．　ＦＥモデルへのマッピング

起動ＳＤＦはコピーを更新するために、前もってコピーが提供されているＳＤＦに対して本情報フローを送信する。

5.5.2.9 コピー更新結果（Update Copy Result）

ａ．　ＦＥ相互関係：ＳＤＦからＳＤＦ

ｂ．　概要

本情報フローはコピー更新（Update Copy）の結果を確認する。

ｃ．　情報要素

なし

ｄ．　ＩＥ記述

なし

ｅ．　ＦＥモデルへのマッピング

コピー更新が提供されているＳＤＦはコピー更新の結果を確認するために本情報フローを送信する。

第２編　インテリジェントネットワーク網間ＣＳ－１（改）の物理プレーンアーキテクチャ編

１．概要

本編は能力セット１（改）のＩＮ網間アーキテクチャの物理プレーンについて述べる。一般的なＩＮ物理プレーンの情報は勧告Ｑ．１２０５に含まれている。

ＩＮ概念モデルの物理プレーンは異なる物理エンティティとこれらエンティティ間のインタフェースを示している。

物理プレーンアーキテクチャはＩＮ概念モデルと整合していなければならない。ＩＮ概念モデルは以下の主要目的を満足するためにＩＮアーキテクチャの設計に用いることが可能なツールである：

－　サービス実現からの独立性

－　網実現からの独立性

－　ベンダと技術からの独立性

Ｉ．１３０の３ステージサービス記述方法が、物理プレーンアーキテクチャの開発に用いられるかもしれない（これはノードの機能仕様とノード間プロトコルの詳細記述を含む）。

２．要求条件と仮定

2.1　要求条件

物理プレーンの重要な要求条件は以下のとおりである：

－　能力セット１（改）分散機能プレーンの機能エンティティは能力セット１物理エンティティにマッピング可能である。

－　１つもしくはそれ以上の機能エンティティが同一の物理エンティティにマッピングされるかも知れない。

－　１つの機能エンティティは２つの物理エンティティ間に分けることは出来ない（即ち、機能エンティティは１つの物理エンティティに完全にマッピングされる）。

－　機能エンティティの重複したインスタンスが、同一物理エンティティではない異なる物理エンティティにマッピング可能である。

－　物理エンティティは物理アーキテクチャを構成するためにグループ化され得る。

－　物理エンティティは標準インタフェースを提供するかもしれない。

－　ベンダは機能エンティティのマッピングと標準インタフェースに基づく物理エンティティの開発が可能でなければならない。

－　ベンダは利用している技術と利用可能となった新技術をサポート可能でなければならない。

2.2　仮定

物理プレーンアーキテクチャの開発のために以下の仮定が成される：

－　ＩＮ概念モデルがＩＮ物理アーキテクチャの開発のためのツールとして用いられる。

－　物理エンティティの開発のために既存及び新技術を用いることが出来る。

－　分散機能プレーンの機能エンティティと物理プレーンの標準インタフェースの仕様は網、ベンダ、サービスに非依存に出来るであろう。

－　能力セット１（改）に関してサービスを提供するために、有意な数のインタフェースが特定されるであろう。サービス生成とＯＡＭ機能は含まれないであろう。

３．物理エンティティ

本章はＩＮ能力セット１（改）を実現するＰＥの選択が述べられる。能力セット１（改）を実現するための他の任意のＩＮＰＥの適用を除外もしくは拒絶することを意図しているものではない。

a) サービス制御局（ＳＣＰ）

ＳＣＰは、ＩＮベースの提供サービスの実現に用いられるサービス論理プログラム（ＳＬＰ）とデータを含む。

サービスの信頼性を向上させＳＣＰ間の負荷分散を可能とするために、複数のＳＣＰ が同一のＳＬＰとデータを含むことが可能である。機能的には１つのＳＣＰはサービス制御機能（ＳＣＦ）とサービスデータ機能（ＳＤＦ）を含む。

b) サービスデータ局（ＳＤＰ）

ＳＤＰは、サービスの実行中にアクセスされるカスタマと網のデータを含む。機能的には、
ＳＤＰはＳＤＦを含む。

４．マッピングの要求条件

- 第２編２．１章に上げられている物理プレーンアーキテクチャの要求条件を満足する必要がある。
- 機能エンティティは、能力セット１（改）ベンチマークとして上げられたサービスを提供する形で、物理エンティティにマッピングされる必要がある。
- 機能エンティティの物理エンティティへのマッピングは既存ＰＥへの効率的なインプリメントを許容しなければならない。
- 機能エンティティの物理エンティティへのマッピングはサービス非依存なインタフェースを介して網機能間の標準通信を許容しなければならない。

５．分散機能プレーンの物理プレーンへのマッピング

5.1 機能エンティティの物理エンティティへのマッピング

本章は、能力セット１（改）のための、機能エンティティの物理エンティティへのマッピングを提供し、ＰＥ間の参照点を記述する。その中で、能力セット１（改）のための、適切な機能の分散と標準化に適した機能インタフェースが強調される。本章で述べられるＰＥは例示のみを目的としており、能力セット１（改）のための機能の唯一可能なマッピングを意図しているものではない。

本章は、いくつかのＰＥから構成される柔軟な物理アーキテクチャを記述する。各ＰＥは１つもしくはそれ以上の機能エンティティを含む。これはそのＩＮ機能性を規定するものである。図２－５－１／ＪＴ－Ｑ１２１８に示されている物理アーキテクチャに含まれるＰＥはサービス制御ポイント（ＳＣＰ）とサービスデータポイント（ＳＤＰ）である。

機能エンティティの物理エンティティへのマッピングの典型的なシナリオは表２－５－１／ＪＴ－Ｑ１２１８に示されている。

物理エンティティ(PEs)：

SCP　　サービス制御ポイント

SDP　　サービスデータポイント

機能エンティティ(FEs)：

SCF　　サービス制御機能

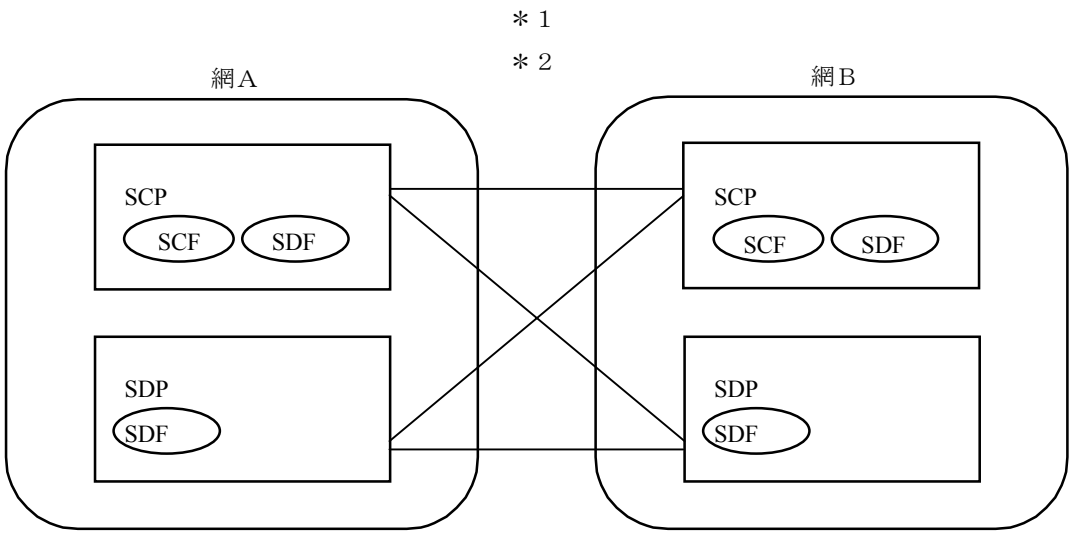
SDF　　サービスデータ機能

インタフェース：

SCP-SDP

SCP-SDP

SCP-SDP



信号 －－－－－－　　*1：各網内は SS No.7 網で転送されることも有り得る

*2：網間は共通の SS No.7 網を設けることも有り得る

なお、両網の関係は、双方向に SCP-SDP の関係を持つことが必須ではない。

図2-5-1/JT-Q1218　物理アーキテクチャのシナリオ

表２－５－１／ＪＴ－Ｑ１２１８

ＦＥのＰＥへのマッピングの典型的なシナリオ

	ＳＣＦ	ＳＤＦ
ＳＣＰ	Ｃ	Ｃ
ＳＤＰ	－	Ｃ

Ｃ： 必須

－: 許容されない

表が、表に示されていない物理エンティティに帰着するという意味での他の任意の機能エンティティの組み合わせを許容しないということを意図するものではない。

上記のマッピングは、図２－５－１／ＪＴ－Ｑ１２１８に示されている。各ＰＥは何らかのマッピングされる機能エンティティを持つ。図の点線は、ＩＮベースの提供サービスのためのアプリケーションレイヤメッセージを転送可能な信号パスを示す。

5.2 ＦＥ－ＦＥ相互関係のＰＥ－ＰＥ相互関係へのマッピング

能力セット１（改）に含まれるＦＥ－ＦＥインタフェースはＳＣＦ－ＳＤＦ及びＳＤＦ－ＳＤＦである。ＰＥ－ＰＥインタフェースへのマッピングは表２－５－２／ＪＴ－Ｑ１２１８に提供されている。

表２－５－２／ＪＴ－Ｑ１２１８

ＦＥのＦＥ相互関係からＰＥ－ＰＥ相互関係

ＦＥ－ＦＥ	ＰＥ－ＰＥ
ＳＣＦ－ＳＤＦ	ＳＣＰ－ＳＤＰ
ＳＤＦ－ＳＤＦ	ＳＤＰ－ＳＤＰ ＳＣＰ－ＳＣＰ ＳＣＰ－ＳＤＰ

本表は、能力セット１（改）標準に包含されるかもしれない全ての可能なＰＥ－ＰＥ相互関係の網羅的な一覧を意味していない。

5.3　下位プロトコルプラットフォームの選択

本章は物理アーキテクチャの要素間の能力セット１（改）のためのインタフェース候補について記述する。インタフェースは：

－　ＳＣＰ－ＳＤＰ
　　ＳＣＰ－ＳＣＰ
　　ＳＤＰ－ＳＤＰ

既存下位レイヤプロトコルがＩＮベースの提供サービスから要求されるアプリケーションレイヤメッセージを転送するための候補インタフェースにたいして提案される。即ち、能力セット１（改）のための標準化作業の焦点はアプリケーションレイヤプロトコルに当てられる。

以下の章は、インタフェース上で用いられるためのいくつかの提案されたプロトコルを示す。

5.3.1　ＳＣＰ－ＳＤＰ／ＳＣＰ－ＳＣＰ／ＳＤＰ－ＳＤＰインタフェース

ＳＣＰとＳＤＰ間のインタフェースとして提案された下位プロトコルプラットフォームはＳＳ　Ｎｏ．７の信号接続制御部（ＳＣＣＰ）/メッセージ転送部（ＭＴＰ）におけるトランザクション機能応用部（ＴＣＡＰ）である。網外のＳＤＰ（例えばクレジットカード会社のクレジットカード検証データベース）に関しては、網内におかれ、ＳＳ　Ｎｏ．７　ＴＣＡＰを公衆もしくは私設データ転送プロトコル（例えばＸ．２５）に翻訳する相互接続ユニットが用いられ得る。

第３編　インテリジェントネットワーク網間ＣＳ－１（改）のインタフェース編

０．序論

この標準は、能力セット１をベースにした網間提供サービスのサポートのために必要とされる網間ＩＮＡＰ（網間インテリジェントネットワークアプリケーションプロトコル）を定義する。

それは、ＩＮ機能モデルで定義されたように、次の２つの機能エンティティ（ＦＥ）の間（ＳＣＦ－ＳＤＦ間）及び同一機能エンティティの間（ＳＤＦ－ＳＤＦ間のみ）の相互動作をサポートする。（注：以下２つのＦＥ名称は網間ＩＮＡＰにおいて変更される可能性がある。例えば、ＳＣＦ (local)，ＳＤＦ (remote) など。）

－サービス制御機能（ＳＣＦ）
－サービスデータ機能（ＳＤＦ）

0.1　定義方法論

プロトコルの定義は、以下の３つの章に分けられる。

－プロトコルのためのＳＡＣＦ／ＭＡＣＦ規則の定義（第３編１章）
－エンティティ間で転送されるオペレーションの定義（第３編２章）
－各エンティティで行われる動作の定義（第３編３章）

ＳＡＣＦ／ＭＡＣＦ規則は、文章記述で定義される。オペレーション定義は、抽象構文表記（ＡＳＮ．１，X．６８０～X．６８３勧告参照）で行われており、動作は、状態遷移図で定義される。オペレーションの受信において実行される動作におけるそれ以上の手引きは、ＪＴ－Ｑ１２１８の第３編２章とＪＴ－Ｑ１２１８第１編の５章の対応する情報フローから得られる（情報フローとオペレーションの間の相互関係は、第３編０．５章参照）。

網間ＩＮＡＰは、ＲＯＳＥユーザプロトコル（X．２１９／２２９参照）である。ＲＯＳＥプロトコルは、ＴＣＡＰ（ＪＴ－Ｑ７７１－７７４及びＩＴＵ－Ｔ勧告Ｑ．７７５参照）とＤＳＳ１（ＪＴ－Ｑ９３２）のコンポーネントサブレイヤに含まれる。現在、ＲＯＳＥＡＰＤＵ（アプリケーションプロトコルデータユニット）は、Ｎｏ．７共通線信号方式のトランザクションサブレイヤメッセージとＪＴ－Ｑ９３１の登録（REGISTER）、ファシリティ（FACILITY）及びＤＳＳ１の呼制御メッセージで運ばれる。他のサポートするプロトコルは、後で追加されるかもしれない。

網間ＩＮＡＰ（１つのＲＯＳＥユーザとして）とＲＯＳＥプロトコルは、ＡＳＮ．１を使って規定されている。現在、ＰＤＵ（Protocol Data Unit）に符号化する唯一の標準化された方法は、基本符号化規則（X．２０９参照）である。

0.2　物理シナリオ例

プロトコルは、機能エンティティの物理エンティティ（ＰＥ）への任意のマッピングをサポートする。どのようにＦＥを可能なもっとも有利な条件で同じ場所に配置するかの決定は、網オペレータや装置製造者に任されている。なぜならば、その条件は製造者及び網オペレータで変化するかもしれないからである。それゆえ、プロトコルは、最大限に分散することを想定（例えば、ＦＥにつき１つのＰＥ）して、定義される。

この章で描かれる図は、Ｎｏ．７共通線信号網環境で、網間ＩＮＡＰがどのようにサポートされるかを示している。これは、網間ＩＮＡＰをサポートするために、Ｎｏ．７共通線信号方式だけが網間プロトコルとして使われるということを意味しているわけではない。

遠方に配置されたＳＣＦとＳＤＦ及びＳＤＦとＳＤＦの間のインタフェースは、コネクションレスＳＣＣＰとＭＴＰ（図３－０－１／ＪＴ－Ｑ１２１８（ＩＴＵ－Ｔ　Ｑ．１２１８）及び図３－０－２／ＪＴ－Ｑ１２１８参照）のサービスを順次用いるＴＣＡＰを使用するＩＮＡＰになるであろう。ＳＤＦは、他の種別の網をアクセスするための他のプロトコルとの相互動作に対して責任がある。

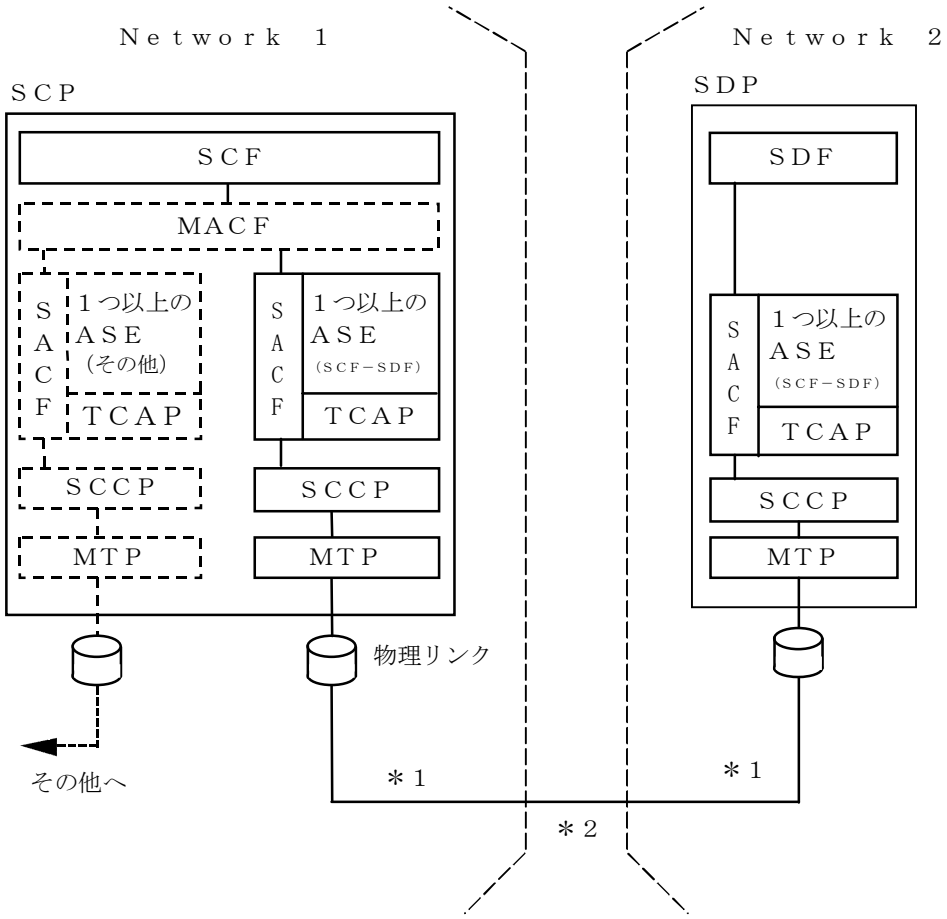
以降の図においてＴＣＡＰが現われる場合には、それは、１つの対話とトランザクションに関連したＴＣＡＰ機能を表わすものと理解されるべきである。（１つのＴＣＡＰエンティティではなく）

メッセージの長さのためにＳＣＦ－ＳＤＦインタフェース上及びＳＤＦ－ＳＤＦインタフェース上（必要なら他のインタフェース上でも）で、ＩＮＡＰメッセージのセグメンテーションとリアセンブリが要求される場合には、ＪＴ－Ｑ７１４で規定されている、ＳＣＣＰコネクションレスメッセージのためのセグメンテーションとリアセンブリ手順が用いられるべきである。

0.3　網間ＩＮＡＰプロトコルアーキテクチャ

この章で使われる用語の多くは、ＩＳＯ　ＩＳ－９５４５で定義されるＯＳＩアプリケーションレイヤ構造に基づいている。

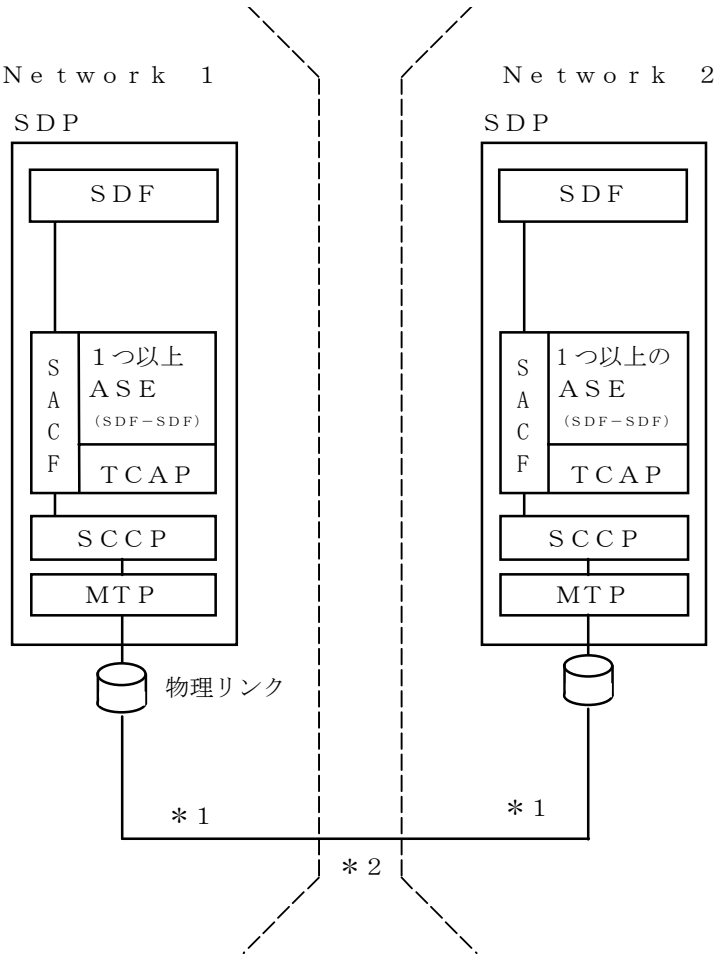
網間ＩＮＡＰプロトコルアーキテクチャは、図３－０－３／ＪＴ－Ｑ１２１８（ＩＴＵ－ＴＱ．１２１８）で示すように描かれる。



＊１　各網内はＳＳＮｏ．７網で転送されることもあり得る
＊２　網間は共通のＳＳＮｏ．７網を設けることもあり得る

注：　ＳＣＰにおけるその他のＡＳＥ（ｓ）の部分は、ＳＣＦにおけるＭＡＣＦの必要性を示すために参考として記述（点線で示されている）されており、本標準の規定外である。

図３－０－１／ＪＴ－Ｑ１２１８（ＩＴＵ－Ｔ　Ｑ．１２１８）
ＳＣＰとＳＤＰの間にインターフェース（網間ＩＮＡＰ用）

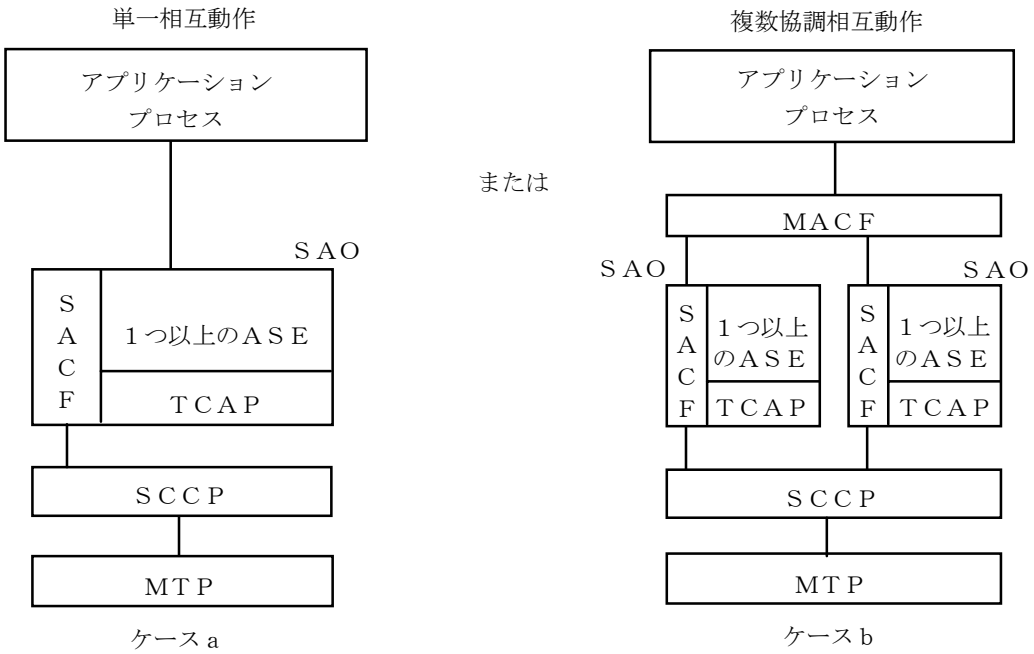


＊１　各網内はＳＳＮｏ．７網で転送されることもあり得る
＊２　網間は共通のＳＳＮｏ．７網を設けることもあり得る

図３－０－２／ＪＴ－Ｑ１２１８
ＳＤＰとＳＤＰの間のインターフェース（網間ＩＮＡＰ用）

図３－０－１／ＪＴ－Ｑ１２１８（ＩＴＵ－Ｔ　Ｑ．１２１８）及び図３－０－２／ＪＴ－Ｑ１２１８の説明：

ＳＡＣＦ　：　単一アソシエーション制御機能
ＭＡＣＦ　：　複数アソシエーション制御機能
ＳＡＯ　：　単一アソシエーションオブジェクト
ＡＳＥ　：　アプリケーションサービス要素
網間ＩＮＡＰ　：　インテリジェントネットワークアプリケーションプロトコル
（注：網間ＩＮＡＰは、すべてのＩＮ　ＡＳＥの内ＳＣＦ－ＳＤＦ関連及びＳＤＦ－ＳＤＦ関連のＡＳＥの仕様を集めたものである。）



注：本標準では単一相互動作のみを取り扱う

図３－０－３／ＪＴ－Ｑ１２１８（ＩＴＵ－Ｔ　Ｑ．１２１８）
網間ＩＮＡＰプロトコルアーキテクチャ

網間ＩＮＡＰに関連する１つの物理エンティティは、他の物理エンティティと１つの相互動作をもつ（ケースａ）かあるいは、複数の協調された相互動作をもつ（ケースｂ）。

ケースａにおいて、ＳＡＣＦは、使用しているＡＳＥ間の協調機能を提供する。それは、（受け取ったプリミティブの命令を基に）ＡＳＥによってサポートされるオペレーションの順序づけを含む。 ＳＡＯは、１対のＰＥの間の単一の相互動作上で用いられ、ＳＡＣＦと１組のＡＳＥで表現される。

ケースｂにおいて、ＭＡＣＦは、遠隔のＰＥにある１つのＳＡＯと相互動作をする複数のＳＡＯの間の協調機能を提供する。

各ＡＳＥは、１つあるいはそれ以上のオペレーションをサポートする。各オペレーションの記述は、関係するＦＥモデル化の動作（ＪＴ－Ｑ１２１８の第１編と第３編の３章参照）に結びつけられている。各オペレーションは、図３－０－４／ＪＴ－Ｑ１２１８（ＩＴＵ－ＴＱ．１２１８）で記述されるオペレーションマクロを用いて規定される。（ＳＣＦ－ＳＤＦインタフェースについてはこの限りではない。この場合、クラス表記を用いて記述される。）

（ＪＴ－Ｑ７７ｘシリーズ勧告で定義されたように）アプリケーションコンテキストネゴシエーションメカニズムの使用は、通信している２つのエンティティが、それぞれの能力が何であるのか、またインタフェース上で必要とされる能力は何であるかを正確に識別することを許容する。これは、能力セットを通じた発展を許容するために使用されるべきである。

特定のアプリケーションコンテキストの表示が、通信している１対のＦＥによってサポートされなければ、前もってコンテキストを定めておくための何らかのメカニズムが、サポートされなければならない。

本標準では、網間ＩＮＡＰにおけるダイアログコントロールと、アプリケーションコンテキストネゴシエーション方法を規定する。（アプリケーションコンテキストネゴシエーションを含め）ＳＣＦ－ＳＤＦ間及びＳＤＦ－ＳＤＦ間の相互関係のコントロールがＴＣＡＰにおいてどのように実現されるかは本標準第３編の２章において規定される。

0.3.1　Ｎｏ．７共通線信号網での網間ＩＮＡＰ 信号輻輳制御

信号輻輳制御のための網間ＩＮＡＰ手順は、ＩＴＵ－Ｔ勧告Ｑ．７６７，§Ｄ．２，１１で規定されているＩＳＤＮユーザパート信号輻輳手順と同様の手順が用いられる可能性がある。すなわち、ＳＣＣＰから”信号局輻輳”の情報を含むＮ－ＰＣＳＴＡＴＥ指示プリミティブを受信すると、網間ＩＮＡＰは影響する方向に対して、数段階でトラヒックの負荷を減ずるという手順である。

上記の手順は、影響する方向に対して、ＭＴＰ局コードアドレシングを用いている場合のトラヒックのみに適用される。

0.4　網間ＩＮＡＰアドレッシング

ＳＣＣＰグローバルタイトルとＭＴＰ信号局コードアドレッシング（ＪＴ－Ｑ７１ｘとＪＴ－Ｑ７０ｘ参照）は、それが、どの網にあるにもかかわらず、ＰＤＵがそれらの物理的な宛先（すなわち、正しい信号局コード）に到達することを保証する。

局内では、網間ＩＮＡＰに割り当てられるＳＳＮ（サブシステム番号）は、網オペレータあるいは開発者の選択による。

上記にかかわらず、ＳＣＣＰによってサポートされるアドレッシング機構が使用される。

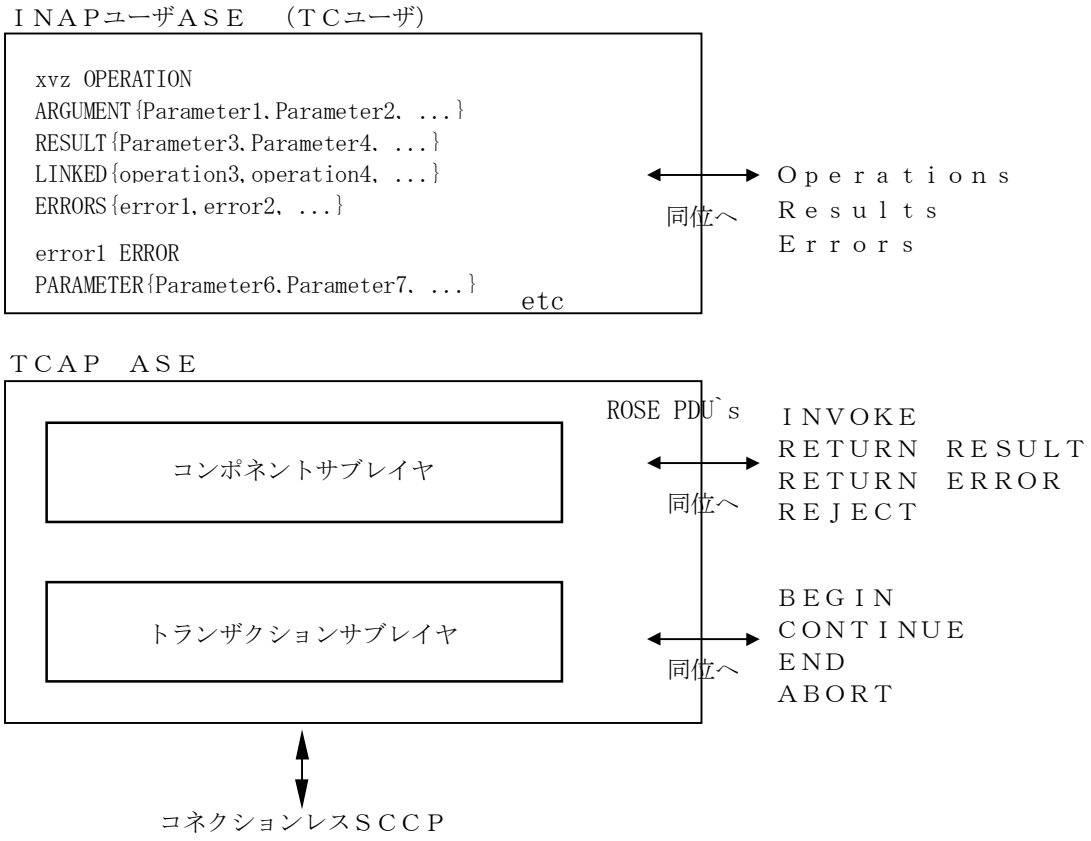


図３－０－４／ＪＴ－Ｑ１２１８（ＩＴＵ－Ｔ　Ｑ．１２１８）

オペレーション記述（ＴＣＡＰの場合）

0.5　ＪＴ－Ｑ１２１８の第１編と第３編の相互関係

以下は、第３編２章以降で対象とする網間情報フローのリストである。これらは、表示されているものを除いて、オペレーションと１対１にマッピングする。

ＪＴ－Ｑ１２１８第１編参照	情報フロー	オペレーション
0.5.4.2.1	認証 （Authenticate）	ディレクトリ結合 （Bind）
0.5.4.2.2	認証結果 （Authenticate Result）	ディレクトリ結合の結果応答 （BindのReturn Result）
0.5.4.2.3	認証された相互関係終了 （End Authenticated Relationship）	結合解放 （Unbind）
0.5.4.2.4	探索 （Search）	探索 （Search）
0.5.4.2.5	探索結果 （Search Result）	探索の結果応答 （SearchのReturn Result）
0.5.4.2.6	エントリ更新 （Modify Entry）	エントリ更新 （ModifyEntry）
0.5.4.2.7	エントリ更新結果 （Modify Entry Result）	エントリ更新の結果応答 （ModifyEntryのReturn Result）
0.5.4.2.8	エントリ追加 （Add Entry）	エントリ追加 （AddEntry）
0.5.4.2.9	エントリ追加結果 （Add Entry Result）	エントリ追加の結果応答 （AddEntryのReturn Result）
0.5.4.2.10	エントリ削除 （Remove Entry）	エントリ削除 （Remove Entry）
0.5.4.2.11	エントリ削除結果 （Remove Entry Result）	エントリ削除の結果応答 （Remove EntryのReturn Result）

0.5.5.2.1	認証 （Authenticate）	ＤＳＡシャドウ結合 （DSA Shadow Bind） ＤＳＡ結合 （DSA Bind） オペレーショナル結合設定 （Establish Operational Binding）
0.5.5.2.2	認証結果 （Authenticate Result）	ＤＳＡシャドウ結合の結果応答 （DSA Shadow BindのReturn result） ＤＳＡ結合の結果応答 （DSA BindのReturn result） オペレーショナル結合設定の結果応答 （Establish Operational Bindingの Return result）
0.5.5.2.3	連鎖要求 （Chaining Request）	連鎖オペレーション （IN-Chained(Operation:operation)）
0.5.5.2.4	連鎖結果 （Chaining Result）	連鎖オペレーションの結果応答 （IN-Chained(Operation:operation) のReturn Result）
0.5.5.2.5	コピー要求 （Copy Request）	シャドウ更新要求 （Request Shadow Update）
0.5.5.2.6	コピー結果 （Copy Result）	シャドウ更新 （Update Shadow）
0.5.5.2.7	認証終了 （End Authenticate）	ＤＳＡシャドウ結合解放 （DSA Shadow Unbind） ＤＳＡ結合解放 （DSA Unbind） オペレーショナル結合終了 （Terminate Operational Binding）
0.5.5.2.8	コピー更新 （Update Copy）	シャドウ更新調整 （Coordinate Shadow Unbind） シャドウ更新 （Update Shadow）
0.5.5.2.9	コピー更新結果 （Update Copy Result）	シャドウ更新の結果応答 （Update ShadowのReturn result）

0.6　網ＩＮＡＰに用いられたコンパティビリティメカニズム

ＳＣＦ－ＳＤＦインタフェースについてのメカニズムは、２．１章に記述する。

ＳＤＦ－ＳＤＦインタフェースについてのメカニズムは、２．２章に記述する。

１．ＳＡＣＦ／ＭＡＣＦ規則

1.1　ＴＣＡＰ　ＡＣの反映

ＴＣＡＰアプリケーションコンテキストネゴシエーション規則は、申し込まれたＡＣが、受け付けられれば、最初の逆方向メッセージに反映されるということを要求する。

もしそのＡＣが受け付けられなくて、ＴＣユーザが対話を続けようとしなければ、新しい対話を始めるのに使うことができる代わりのＡＣが起動者に提供されるかもしれない。

ＴＣＡＰ　ＡＣネゴシエーションは、ＳＣＦインタフェースにのみ適用される。

ＴＣＡＰ　ＡＣネゴシエーションメカニズムのより詳細な記述については、ＪＴ－Ｑ７７ｘを参照すること。

ディレクトリ結合（Bind）、ディレクトリ結合結果の結果応答（Bind のリターンリザルト）のＴＣＡＰ上へのマッピング方法は本標準の第３編２．１章の規定による。

1.2　オペレーションの直列／並列実行

一般にＴＣＡＰではオペレーションの直列・並列実行を許している。網間ＩＮＡＰで利用するＸ．５００アプリケーションプロトコルのオペレーションでも、Bindを除き複数のオペレーションを同一のＴＣＡＰメッセージで送信して直列に実行する点について問題はない（ただしオペレーションが意図された通りに実行されるか否かの保証については別個の問題である）。網間ＩＮＡＰの規定として、複数オペレーションは直列実行する。

２．ＩＮ能力セット１のアプリケーションプロトコルの抽象構文

2.1　ＳＣＦ－ＳＤＦインタフェース

2.1.1　ＩＮ Ｘ．５００ ＤＡＰサブセットへの導入

2.1.1.1　Ｘ．５００の概念とＩＮとの整合

ＳＣＦ－ＳＤＦ間インタフェースおよびＳＤＦの内容を規定するためにＸ．５００シリーズの勧告を用いる。Ｘ．５００勧告の考え方の多くは直接ＩＮの環境でも使用できるが、ディレクトリで導入された考え方を正しく理解するために、用語レベルの整合が必要であり、本項はこれを目的としている。そこで、ＩＮであいまいな用語に絞って記述する。

ＳＣＦの構成において、サービスデータ管理部がＳＤＦとのインタラクションを行なうが、これがディレクトリユーザエージェント（ＤＵＡ）の概念に対応する。ユーザ対応にＳＣＦがＳＤＦとのアソシエーションを設定する際、ＤＵＡのインスタンスがＳＬＰＩ内で生成される。またそのインスタンスは、アソシエーションが終了するときに消滅する。

ＳＤＦはデータベースへの要求に応えるエンティティである。この機能エンティティはディレクトリシステムエージェント（ＤＳＡ）に対応する。ＳＣＦ－ＳＤＦ間のアソシエーションが設定されると、ＤＳＡのインスタンスがアソシエーションの存続期間中生成される。

ディレクトリはＤＳＡ（ＳＤＦ）の集合である。この集合は特定サービスあるいは各種のサービスで使用できる。ディレクトリの概念はＩＮにおけるデータベースシステムのそれと同等のものである。

ディレクトリはまたデータの倉庫と見ることができる。ＩＮのサービスでは様々なデータアクセスをユーザに提供する。情報はエントリという型で構成される。エントリは識別（あるいは名前で特定）できる情報の集まりである。エントリがオブジェクトを表わす（すなわち、オブジェクトに関する主要な情報を含む）場合は、オブジェクトエントリと呼ばれる。

オブジェクトとは、識別（あるいは名前で特定）でき、かつデータベース内に情報を保持することが必要なものである。典型的な例はユーザである。オブジェクトは複数のエントリによって表現できる。オブジェクトを表現するために用いる個々の情報は属性である。これはエントリと関連する。

ＩＮ環境では、ＤＳＡに含まれるデータの管理と運用はサービス提供者が行なう。したがって、サービス提供者が管理者の役割を担う。これは、Ｘ．５００で言う管理上の権威者に相当する。サービス提供者はセキュリティ手順（認証およびアクセス制御）を実行する。

2.1.1.2　Ｘ．５００の限定されたサブセットの利用

Ｘ．５００勧告の目的は、ディレクトリサービスを提供することであって、ＳＣＦ－ＳＤＦ間インタフェースの記述ではない。Ｘ．５００の機能はＩＴＵ－Ｔ　ＩＮ ＣＳ－１で必要とする機能よりも広い。本項では、ＣＳ－１（改）の範囲においてインプリする際に考慮しサポートすべきディレクトリ抽象サービスの側面を示す。また、サポートしていないパラメータを受信した場合の対応についても触れる。様々なパラメータの状態を表す手段としてプロファイリングを用いる。

一つのメッセージで送信するパラメータ数をできるだけ少なくすることは、信号トラヒック量および処理時間の点から重要である。したがって、信号送信時に必須でないパラメータは削除する。信号を受信した際、削除されたパラメータは受信側のエンティティによって解釈されるべきである。これによって、1993年版のディレクトリの記述に従って将来プロファイルの拡張が許容される。

便利かつ明確のため、このプロファイルはＡＳＮ．１サブタイピングファシリティーズを用いて定義するが、プロトコルの仕様を記述しているのではない。これは単純にインプリにおいて送信すべきでないパラメータを示している。ただし、ディレクトリ抽象サービスの元の定義に適合している値をデコードできる受信側のエンティティのふるまいを変えるものではない。そうではあるが、サブタイピングによって除外された要素は無視されるべきである。

2.1.1.3 前提条件

ＩＮ ＣＳ－１（改）のためのディレクトリ抽象サービスのプロファイルを設計するために、いくつかの前提条件を用いる。これらの前提条件は 2.1.2.1 項から参照される。

・前提条件 1： ＩＴＵ－Ｔ ＩＮ ＣＳ－１で用いるディレクトリ抽象サービスは 1993 年版である。1988 年版でのみ用いられているパラメータは無視される。将来の能力セットで必要になるかもしれない機能は、サポートされていなくても考慮だけはしておくべきである。

・前提条件 2：ＩＮにおけるエリアスエントリは、オブジェクトに対して別の名称を提供する単なる手段であり、したがって必要な場合は参照されるべきである。

・前提条件 3：ＳＣＦ－ＳＤＦ間のオペレーションは破棄できない。たとえオペレーションが非常に多くの時間を要したとしても、タイマが満了するため、破棄する必要がない。

2.1.2 ＩＮ Ｘ．５００ ＤＡＰサブセット

2.1.2.1 ＩＮで用いるためのＸ．５１１の見直し

2.1.2.1.1 情報タイプおよび共通プロシージャ

2.1.2.1.1.1 Common arguments

```
IN-CommonArguments ::= CommonArguments (
        WITH COMPONENTS {
            serviceControls (IN-ServiceControls),
            securityParameters          ,
            requestor                   ,
            operationProgress           ,
            aliasedRDNs                 ABSENT,
            criticalExtensions          ,
            referenceType               ,
            entryOnly                   ,
            exclusions                  ,
            nameResolveOnMaster         })
```

CommonArguments パラメータの各要素の状態は以下のようになる。

・serviceControls－（2.1.2.1.1.3 項を参照）

・securityParameters－ディレクトリオペレーションに関するセキュリティの特徴（すなわちオペレーションの署名）を制御する。

・requestor－特定のオペレーションの発信元の名前を提供する。主に署名付きの要求に使用される。

・operationProgress, referenceType, entryOnly, exclusions, nameResolveOnMaster－ＤＵＡがＤＳＡからの継続参照の上で動作するときに使用される。

・aliasedRDNs－コンパチビリティのためだけに 1993 年版で現れている。したがって、1993 年版のディレクトリのインプリでは常に削除すべきである（前提条件 1）。

・criticalExtensions－1988 年版によるインプリに対して、1993 年版で利用可能ないくつかの拡張の重要性を示すために用いる。ＣＳ－１（改）では 1993 年版が使用されるため（前提条件 1）、本要素は送られるべきではない。しかしながら、ＩＮの環境においては以下の拡張がサポートされるべきである。

subentries, matchedValuesOnly, useAliasOnUpdate

2.1.2.1.1.2 Common results

```
IN-CommonResults ::= CommonResults (
        WITH COMPONENTS {
            securityParameters          ,
            performer                   ,
            aliasDereferenced           })
```

CommonResults パラメータの各要素の状態は以下のようになる。

・securityParameters－

・performer－特定のオペレーションを実行側の名前を提供する。主に署名付きの結果に使用される。

・aliasDereferenced－オペレーションの処理中にエリアスに遭遇し、参照される場合に TRUE に設定する。

2.1.2.1.1.3 Service controls

```
IN-ServiceControls ::= ServiceControls (
    WITH COMPONENTS {
      options                   ,
    priority                    ,
    timeLimit                   ABSENT,
    sizeLimit                   ABSENT,
    scopeOfReferral             ,
    attributeSizeLimit          ABSENT})
```

ServiceControls パラメータの各要素の状態は以下のようになる。

・option－いくつかの意味を含む。

・dontDereferenceAliases は、注目するオブジェクトがエリアスの場合にのみ含まれる。（例、エリアスの削除）

・subentries は、アドレスする必要のあるものがエントリがサブエントリかに応じて設定される。

・priority－サービスを提供する優先度を規定するために用いる。本パラメータは、ＤＳＡ内の輻輳を管理するために使用できる。

・timeLimit－要求を実行するための最大時間を示す。これはＴＣＡＰのオペレーションタイマと冗長であるため、必要ではない。

・sizeLimit および attributeSizeLimit－オブジェクトおよび属性に関して、結果上のサイズ限界を設定する。一般的な要求（要求元がＤＳＡの構造を知らない）の場合には有用であるが、ＩＮの場合にはこのような制限は適用可能とは思えない。

・scopeOfReferral－照会に関する範囲を示す。

2.1.2.1.1.4　Entry information selection

```
IN-EntryInformationSelection ::= EntryInformationSelection (
        WITH COMPONENTS {
            attributes              ,
            infoTypes               (attributeTypesAndValues),
            extraAttributes         })
```

EntryInformationSelection パラメータの各要素の状態は以下のようになる。

・attributes－参照系のサービスで返送される属性を表す。allUserAttributes のオプションは必要以上のトラヒックを生むために、サービス規定者には使用を避けるように忠告はするが、許されている。なるべく要求する属性の名前を指定する select のオプションを使用すべきである。

・infoTypes－属性の種別と値を返送すべきか種別のみで良いかを規定する。ＩＮでは、サービスは主にその処理に関連する属性値を意識するため、本要素はデフォルト値を与えることで存在すべきではない。

・extraAttributes － attributes と同様の使い方であり、違いは allUserAttributes のオプションが allOperationalAttributes に置き換わるのみである。

2.1.2.1.1.5　Entry information

```
IN-EntryInformation ::= EntryInformation (
        WITH COMPONENTS {
            name                PRESENT,
            fromEntry           (TRUE),
            information         (WITH COMPONENTS {
                                attributeType               ABSENT,
                                attribute                   PRESENT}) OPTIONAL,
            incompleteEntry })
```

EntryInformation パラメータの各要素の状態は以下のようになる。

・name－返送すべきエントリの名前を示し、必須要素である。

・fromEntry－コピーを返送するのかエントリ自身を返送するのかを表す。ＩＴＵ－Ｔ　ＩＮ　ＣＳ－１ではコピーメカニズムを使用しないため（前提条件 3）、デフォルト値のみが使用される。

・information－返送すべき関連情報を含む。infoTypes 要素で規定した選択（2.1.2.1.1.4 参照）を考慮すると、attribute のオプションのみが使用される。

・incompleteEntry－返送された結果が、例えばアクセス権の制約などで完全ではない場合に示す。このようなパラメータをサポートすることは必要である。

2.1.2.1.2　Bind オペレーション

```
in-DirectoryBind      OPERATION ::= {
    ARGUMENT            IN-DirectoryBindArgument
    RESULT              IN-DirectoryBindResult
    ERRORS              {in-DirectoryBindError}}
```

DirectoryBind および DirectoryUnbind のオペレーションは、ディレクトリをアクセスする特定期間の初めと終わりに使用される。

2.1.2.1.2.1 Bind arguments および results

```
IN-DirectoryBindArgument ::= DirectoryBindArgument
        (WITH COMPONENTS {
            credentials             (IN-Credentials) OPTIONAL,
            versions                ({v1})})
IN-Credentials ::= Credentials
        (WITH COMPONENTS {
            simple                  ,
            strong                  ,
            externalProcedure           })
IN-DirectoryBindResult ::=IN-DirectoryBindArgument
```

DirectoryBindArgument パラメータの各要素の状態は以下のようになる。

・credentials－ユーザの身元を確立するために使用される。認証を要求するサービスでは必要である。

・versions－使用するディレクトリサービスのバージョンを表す。ＩＮ ＣＳ－１（改）では一つのバージョンのみがサポートされるため、このパラメータは送る必要がない。サポートされていない v1 を受信したときは、サービスエラーを返送すべきである。

上述と同様のことが DirectoryBindResult にも適用される。

2.1.2.1.2.2 Bind errors

```
in-DirectoryBindError       ERROR ::= {
    PARAMETER               IN-DirectoryBindErrorParameter}
IN-DirectoryBindErrorParameter ::= DirectoryBindErrorParameter
    (WITH COMPONENTS {
        versions                ({v1}),
        error               (WITH COMPONENTS {
                                securityError       (SecurityProblem (1｜2｜7)),
                                serviceError    (ServiceProblem (2))})})
```

（注） DirectoryBindError は実際にはＸ．５１１(1993)で定義されていない。

DirectoryBindError は、ディレクトリ抽象サービスで定義されているものである。受信した場合、Bind のエラーを解釈するために、あらゆる可能なエラーをサポートすべきである。

2.1.2.1.3 Search オペレーション

```
in-Search   OPERATION ::= {
    ARGUMENT            IN-SearchArgument
    RESULT              IN-SearchResult
    ERRORS              {nameError｜in-ServiceError｜securityError｜attributeError｜referral}
    CODE                id-opcode-in-search}
```

search オペレーションは、所望のエントリを求めてＤＩＴを検索するために用いる。

2.1.2.1.3.1 Search arguments および errors

```
IN-SearchArgument ::= SearchArgument
        (WITH COMPONENTS {
            baseObject          PRESENT,
            subset              ,
            filter              ,
            searchAliases       (TRUE),
            selection           (IN-EntryInformationSelection),
            pagedResults        ABSENT,
            matchedValuesOnly   ,
            extendedFilter      ABSENT})
```

SearchArgument の各パラメータの状態は以下のようになる。

・baseObject－検索を行うオブジェクトエントリを示し、必須パラメータである。

・subset－検索オペレーションの対象となるＤＩＴの部分を表す。このパラメータは必要であるが、wholeSubtree を使用することは、オペレーションタイマが満了しないようにするためにも避けるべきである。

・filter－検索空間を限定するために使用する。コンパチビリティの理由で extendedFilter パラメータがディレクトリの 1993 年版で追加されたが、送られるべきではない。filter パラメータのみを送るべきである。

・searchAliases－（ベースオブジェクトを除く）検索空間で遭遇したエリアスを考慮すべきか否かを示す。ＩＮでは検索する際、エリアスは常に参照されるため、本パラメータはデフォルト値でのみ使用される。

・selection－エントリ中のどの情報（例、タイプと値）が要求されているかを表す。（2.1.2.1.1.4 参照）

・pagedResults－ページ毎の結果を要求するために使用する。本パラメータは検索オペレーションの結果をページフォーマットで表すために使用する。結果はＳＣＦが処理するため、本情報はＣＳ－１（改）では必要ない。

・matchedValuesOnly－フィルタを満足する値のみを返送するために使用する。これはＩＮ ＣＳ－１（改）でも必要かもしれない。

エラーに関しては、abandoned はサポートされない。

2.1.2.1.3.2 Search results

```
IN-SearchResult ::= SearchResult
        (WITH COMPONENTS {
            searchInfo              (WITH COMPONENTS {
                                        name                    ,
                                        entries                 (WITH COMPONENT (IN-EntryInformation)),
                                        partialOutcomeQualifier (PartialOutcomeQualifier
                                            (WITH COMPONENTS {
                                            limitProblem                ,
                                            unexplored                  ,
                                            unavailableCriticalExtensions ,
                                            unknownErrors               ,
                                            queryReference              ABSENT}))}),
            uncorrelatedSearchInfo          })
```

SearchResult の各パラメータの状態は以下のようになる。

・name－返送されたエントリの名前が baseObject の名前と異なるときに、そのエントリの名前を表す。検索オペレーションの結果、基点オブジェクトと異なるエントリの時に必要となる。

・entries－フィルタを満足するエントリを含む（2.1.2.1.3.1 参照）。

・partialOutcomeQualifier－検索オペレーションが完全に終了しなかったときに存在する。これには、検索オペレーションが終了しなかった理由，オペレーションが止まった箇所に関する情報が含まれる。

・limitProblem－どんな（運用上の）制約にぶつかったかを表す。このような表示を運ぶことは必要である。

・unexplored－検索オペレーションが止まった箇所を表し、オペレーションの状態（考慮されたオブジェクトおよび進行状況）を示す。オペレーションを再度送信する場合はこのようなパラメータは必要であるが、これは照会と等価となる。

・unavailableCriticalExtensions－critical 拡張で利用できないものがあることを表す。

・unknownErrors－他のＳＤＦから未知のエラーを受信したことに相当する。これは連携メカニズム使用することを意味する。

・queryReference－ページを付与した結果が要求される場合に使用するものであるため、必要ではない。

・uncorrelatedSearchInfo－署名無しにできない署名付きの結果を含む。

2.1.2.1.4　AddEntry オペレーション

```
in-AddEntry     OPERATION ::= {
        ARGUMENT        IN-AddEntryArgument
        RESULT          AddEntryResult
        ERRORS          {nameError｜in-ServiceError｜securityError｜attributeError｜updateError｜referral}
        CODE            id-opcode-in-addEntry}
```

addEntry オペレーションは、ＤＩＴにリーフエントリを追加するために用いる。

2.1.2.1.4.1　AddEntry arguments, results および errors

```
IN-AddEntryArgument ::= AddEntryArgument
        (WITH COMPONENTS {
            object          PRESENT,
            entry           PRESENT,
            targetSystem    })
```

AddEntryArgument の各パラメータの状態は以下のようになる。

・object－追加が行われるエントリを特定するものであり、必須パラメータである。

・entry－追加するエントリの内容を記述するものであり、必須パラメータである。

・targetSystem－エントリを追加するＳＤＦの名前を含む。

2.1.2.1.5 RemoveEntry オペレーション

```
in-RemoveEntry      OPERATION ::= {
        ARGUMENT            IN-RemoveEntryArgument
        RESULT              RemoveEntryResult
        ERRORS              {nameError | in-ServiceError | securityError | updateError | referral}
        CODE                id-opcode-in-removeEntry}
```

removeEntry オペレーションは、ＤＩＴからリーフエントリを削除するために用いる。

2.1.2.1.5.1 RemoveEntry arguments, results および errors

IN-RemoveEntryArgument ::= RemoveEntryArgument

パラメータは、削除すべきオブジェクトの名前のみであり、必須である。オペレーションのアーギュメントは、ディレクトリに記述されているものと同一である。

2.1.2.1.6 ModifyEntry オペレーション

```
in-ModifyEntry      OPERATION ::= {
        ARGUMENT        IN-ModifyEntryArgument
        RESULT          IN-ModifyEntryResult
        ERRORS          {nameError | in-ServiceError | securityError | attributeError | updateError | referral}
        CODE            id-opcode-in-modifyEntry}
```

modifyEntry オペレーションは、一つのエントリに対して一連の変更を行うために用いる。

2.1.2.1.6.1 ModifyEntry arguments および errors

```
IN-ModifyEntryArgument ::= ModifyEntryArgument
        (WITH COMPONENTS {
            object          PRESENT,
            changes         PRESENT,
            selection       (IN-EntryInformationSelection)})
```

パラメータは、変更すべきオブジェクトの名前，変更のリスト，返送すべき属性と値を規定する selection である。最初の二つは必須であり、最後のは 2.1.2.1.1.4 項に対応する。changes パラメータで利用可能なすべての変更はＩＮ ＣＳ－１（改）で有用であるため、オペレーションのアーギュメントは、selection パラメータを除きディレクトリに記述されているものと同一である。

2.1.2.1.6.2 ModifyEntry results

```
IN-ModifyEntryResult ::= ModifyEntryResult
        (WITH COMPONENTS {
            null            ,
            information     Information
                            (WITH COMPONENTS {
                                entry       (IN-EntryInformation)})})
```

modifyEntry オペレーションによって参照すべき情報がない場合は、null 結果が返送される。それ以外の場合は、information 結果の entry 要素によって情報が返送される。ＩＮ ＣＳ－１（改）では本要素は 2.1.2.1.1.5 項で規定される。

2.1.2.1.7 Errors

ディレクトリで定義されている優先順位ルールが適用される。abandoned, abandonFailed エラーは、ＣＳ－１（改）ではサポートされない（前提条件 1 および 4）ため、考慮しない。

2.1.2.1.7.1 Attribute エラー

attributeError で表されるあらゆる属性の問題はＣＳ－１（改）で使用可能であるため、本エラーはディレクトリからそのまま取り込み、修正はしない。

2.1.2.1.7.2 Name エラー

nameError で表されるあらゆる名前の問題はＣＳ－１（改）で使用可能であるため、本エラーはディレクトリからそのまま取り込み、修正はしない。

2.1.2.1.7.3 Security エラー

securityError で表されるあらゆるセキュリティの問題はＣＳ－１（改）で使用可能であるため、本エラーはディレクトリから直接取り込み、修正はしない。

2.1.2.1.7.4 Service エラー

```
in-ServiceError        ERROR ::= {
        PARAMETER              IN-ServiceErrorParameter
        CODE                   id-errcode-in-serviceError}
IN-ServiceErrorParameter ::= ServiceErrorParameter
        (WITH COMPONENTS {
                problem(ServiceProblem (1｜2｜3｜4｜5｜6｜8｜9｜10｜11｜12))})
```

serviceError はサービスの提供に関する問題を報告する。これは発生した問題の種別を表す problem パラメータのみを含む。起こり得る問題の中で、ページを付与した結果の使用に関する invalidQueryReference は送るべきではない。

2.1.2.1.7.5 Update エラー

updateError で表されるあらゆる更新上の問題はＣＳ－１（改）ですべて使用可能である。従って、本エラーは修正することなく利用されるべきで、ディレクトリから直接ハンドリングされる。

2.1.2.1.7.6 Referral

referral エラーは、別のＳＤＦへ問い合わせをリダイレクトするための情報を示す。本エラーは修正することなく利用されるべきで、ディレクトリから直接ハンドリングされる。

2.1.2.2　ディレクトリアクセスプロトコルのサブセット

2.1.2.2.1　プロトコルの概略

2.1.2.2.1.1　遠隔オペレーションの仕様およびＳＳ７での実現

ＩＴＵ－Ｔ勧告Ｘ．８８０｜ＩＳＯ／ＩＥＣ　９０７２－１は、ディレクトリの仕様で定義されている様々なディレクトリプロトコルのようなＲＯＳベースのアプリケーションプロトコルを規定する際に有用である情報オブジェクトクラスを定義している。これらのクラスは、後述の項で使用される。ＩＴＵ－Ｔ勧告Ｘ．８８０｜ＩＳＯ／ＩＥＣ　９０７２－１で提供される規定技術は、オブジェクト間の一般的なプロトコルを定義するために使用される。ＳＳ７のアプリケーションレイヤプロトコルとして実現する場合、ＩＴＵ－Ｔ勧告Ｘ．８８０｜ＩＳＯ／ＩＥＣ　９０７２－１の概念は、ＩＴＵ－Ｔ勧告Ｑ．７７５のＳＳ７の概念に対応する。本勧告で定義されるＳＳ７のアプリケーションレイヤプロトコルは、ペアのアプリケーションプロセス間の通信を提供するプロトコルである。ＳＳ７の環境では、ペアのアプリケーションエンティティ（ＡＥ）間のトランザクション能力（ＴＣ）を用いた通信として表せる。ＡＥの機能は、アプリケーションサービス要素（ＡＳＥ）の集合によって表せる。ＡＥ間のインタラクションは、ＡＳＥによって提供されるサービスを使用するという形で記述される。ディレクトリのＡＳＥで提供されるすべてのサービスは、一つのＡＥ内に含まれる。ＴＣのコンポーネントハンドラ（ＣＨＡ）は、オペレーションの要求／応答型をサポートする。ディレクトリのＡＳＥは、ディレクトリのオペレーションパッケージの抽象構文標記からＴＣによって提供されるサービスへのマッピング機能を提供する。ＴＣのダイアログハンドラ（ＤＨＡ）は、ペアＡＥ間の「対話」というアプリケーションアソシエーションの設定および解放をサポートする。ＤＵＡとＤＳＡ間の対話はＤＵＡからのみ設定される。

2.1.2.2.1.2　ディレクトリのＲＯＳオブジェクトおよびコントラクト

ＩＴＵ－Ｔ勧告Ｘ．５１１・ＩＳＯ／ＩＥＣ　９５９４－３は、ユーザがディレクトリサービスにアクセスするためのアクセスポイントであるディレクトリとＤＵＡ間の抽象サービスを定義している。2.1.2 項はこの抽象サービスをＩＮで利用した場合のサブセットを定義している。ＲＯＳオブジェクトの dua クラスは、このクラスのインスタンスでありコントラクト dapContract の起動側であるＤＵＡを規定している。このコントラクトは、ディレクトリの仕様ではディレクトリ抽象サービスと言われ、2.1.2.2.1.3 項でＲＯＳベースの情報オブジェクトとして規定される。

```
dua                 ROS-OBJECT-CLASS ::=
{
        INITIATES           {dapContract}
        ID                      id-rosObject-dua
}
```

ＲＯＳオブジェクトの directory クラスは、ディレクトリ抽象サービスの提供側を記述する。この提供側は dapContract の応答側である。

```
directory           ROS-OBJECT-CLASS ::=
{
        RESPONDS        {dapContract}
        ID                      id-rosObject-directory
}
```

ディレクトリはさらに、ＤＵＡに対して特定のアクセスポイントをサポートするＤＳＡによって表されるものとしてモデル化できる。インテリジェントネットワークにおいては、各ＤＳＡはディレクトリに対するアクセスポイントとなる可能性がある。directory オブジェクトはＤＳＡの集まり（それぞれはＳＤＦ内に存在）として表現される。directory を構成する各ＤＳＡは dap-dsa クラスのインスタンスである。dap-dsa オブジェクトは dapContract における応答側の役割を担う。

```
dap-dsa              ROS-OBJECT-CLASS ::=
{
        RESPONDS        {dapContract}
        ID                      id-rosObject-dapDSA
}
```

本標準の将来の版では、様々な目的を達成するために、ＤＳＡ間でお互いに連携動作することも可能となるであろう。

2.1.2.2.1.3　ＤＡＰコントラクトおよびパッケージ

dapContract はコントラクトというクラスの情報オブジェクトとして定義される。

```
dapContract             CONTRACT ::=
{
        CONNECTION              dapConnectionPackage
        INITIATOR CONSUMER OF   {searchPackage｜modifyPackage}
        ID                              id-contract-dap
}
```

ＤＵＡとＤＳＡが異なるＩＮの物理エンティティにあるときは、このアソシエーションコントラクトは、ＩＮディレクトリアクセスプロトコル（ＤＡＰ）と呼ばれるＳＳ７のアプリケーションレイヤプロトコルとして実現される。本プロトコルのＳＳ７のアプリケーションコンテキストという意味での定義は、2.1.2.2.2.2 に示す。dapContract は、dapConnectionPackage というコネクションパッケージ，および 2 つのオペレーションパッケージ searchPackage, modifyPackage から成る。dapConnectionPackage というコネクションパッケージは、コネクションパッケージクラスの情報オブジェクトとして定義される。このコネクションパッケージの結合および結合解放のオペレーション、すなわち directoryBind および directoryUnbind は、ＩＴＵ－Ｔ勧告Ｘ．５１１｜ＩＳＯ／ＩＥＣ　９５９４－３で定義されている。

```
dapConnectionPackage        CONNECTION-PACKAGE ::=
{
        BIND                    directoryBind
        UNBIND                  directoryUnbind
        ID                      id-package-dapConnection
}
```

オペレーションパッケージ searchPackage および modifyPackage は、オペレーションパッケージクラスの情報オブジェクトとして定義される。これらのオペレーションパッケージのオペレーションは、ＩＴＵ－Ｔ勧告Ｘ．５１１｜ＩＳＯ／ＩＥＣ　９５９４－３で定義されている。ＩＴＵ－Ｔ勧告Ｘ．５１１｜ＩＳＯ／ＩＥＣ　９５９４－３では、さらにディレクトリへのアクセスをサポートする別のオペレーションを定義しているが、インテリジェントネットワークにおいては使用しない。

```
searchPackage                   OPERATION-PACKAGE ::=
{
        CONSUMER INVOKES        {search}
        ID                              id-package-search
}
modifyPackage                   OPERATION-PACKAGE ::=
{
                                CONSUMER INVOKES        {addEntry | removeEntry | modifyEntry}
                                ID                                      id-package-modify
}
```

注）これらのパッケージがＡＳＥとして実現される場合は、この規定で定義されるアプリケーションコンテキストの構築のために使用されるものであって、個々のＡＳＥあるいはＡＳＥの組み合わせに対するコンフォーマンスの要求を許容する意図はない。

ＤＵＡは dapContract の起動側であるため、コントラクトのオペレーションパッケージの消費者の役割を担う。これはこのコントラクトおよびＳＳ７での実現において、ＤＵＡのみがオペレーションを発行できることを意味する。

2.1.2.2.1.4　下位サービスの利用

ＤＡＰプロトコルは以下に記すような下位サービスを利用する。

2.1.2.2.1.4.1　コンポーネントハンドリングサービスの利用

ディレクトリＡＳＥは、アプリケーションプロセスによって使用される TC-L-REJECT および TC-L-CANCEL 以外のＴＣコンポーネントハンドリングサービスを使用する。TC-L-REJECT-表示および TC-L-CANCEL-表示を受信すると、アプリケーションプロセスは対話を破棄する（すなわち、TC-U-ABORT-要求プリミティブを送信する）。TC-U-CANCEL サービスは使用しない。

2.1.2.2.1.4.2　ダイアログハンドリングサービスの利用

ダイアログハンドリングサービスは、ディレクトリ結合およびディレクトリ結合解放のオペレーションをサポートするため、およびディレクトリパッケージに含まれるオペレーションに関する APDU の送信を起動するために使用される。アプリケーションプロセスの制御下で TC-BEGIN および TC-CONTINUE を適当に使用することによって、コンポーネントのグループ化が実行される。TC-END は結合解放手順をサポートするためだけに用いられる（すなわち、コンポーネントを送信するためには使用しない）。空きの TC-CONTINUE 要求プリミティブを受信した場合、ＳＤＦはそのプリミティブを無視すべきである。データベース要求で TC-END 要求を受信した場合、ＳＤＦはそのデータベース要求を実行せず、要求された TC-END サービスを結合解放手順とみなすべきである。その場合、対話は終了する（2.1.2.2.3.1.2 項参照）。ＳＤＦ内で TC-U-REJECT 表示を受信した場合、このプリミティブは無視されるべきである。ＳＤＦ内で TC-R-REJECT 表示を受信した場合、その対話は TC-U-ABORT 要求で解放すべきである。拒否の条件がＳＤＦ内で検出された場合は、TC-U-REJECT 要求に続いて TC-CONTINUE 要求を送信すべきである。プリアレンジド終了手順は使用しない。アプリケーションプロセスは、TC-P-ABORT および TC-NOTICE の単独のユーザである。対話上で TC-U-ABORT-表示あるいは TC-P-ABORT-表示を受信すると、すべての要求処理を終

了する。要求のあった変更が生じたかどうかを確認するのは、アプリケーションプロセスの役割である。また、正しいＳＤＦへメッセージをルーティングするために、下位のＳＣＣＰで使用できる着アドレスをTC-BEGIN-要求プリミティブ内に設定することもアプリケーションプロセスの役割である。

2.1.2.2.2　ディレクトリプロトコル抽象構文

2.1.2.2.2.1　抽象構文

今回のディレクトリアクセスプロトコルの版は、3 つの抽象構文のサポートを要求する。

a) ＳＣＦおよびＳＤＦ間の対話を確立するために必要であり、ＩＴＵ－Ｔ勧告Ｑ．７７３で規定されているＴＣ対話制御プロトコルデータ単位の抽象構文である dialogue-abstract-syntax。

b) ディレクトリ結合およびディレクトリ結合解放オペレーションの起動、およびその結果通知のためのプロトコルデータ単位を送信するための抽象構文。

c) 2.1.2.2.1.3 項で規定されているオペレーションパッケージに含まれるオペレーションの起動、およびその結果通知のためのプロトコルデータ単位を送信するための抽象構文。

2 番目の抽象構文の値が得られるＡＳＮ．１タイプは、ＩＴＵ－Ｔ勧告Ｘ．８８０で定義されているパラメータ化されたタイプ、Bind{}および Unbind{}を用いて規定される。

最後の抽象構文の値が得られるＡＳＮ．１タイプは、ＩＴＵ－Ｔ勧告Ｑ．７７３で定義されているパラメータ化されたタイプ、TCAPMessage{}を用いて規定される。これらすべての抽象構文は、ＩＴＵ－Ｔ勧告Ｑ．７７３にリストアップされている制約を付けた基本ＡＳＮ．１コード化規則にしたがって（最低限）コード化される。

2.1.2.2.2.1.1　ＤＡＰ抽象構文

2.1.2.2.1.3 項で規定したオペレーションパッケージを実現する複数のディレクトリＡＳＥは、一つの抽象構文 directoryOperationsAbstractSyntax を共有する。これは抽象構文クラスの情報オブジェクトとして規定される。

```
inDirectoryOperationsAbstractSyntax          ABSTRACT-SYNTAX ::= {
        BasicDAP-PDUs
        IDENTIFIED BY          id-as-directoryOperationsAS}
BasicDAP-PDUs ::= TCAPMessage {{DAP-Invokable},{DAP-Returnable}}
DAP-Invokable      OPERATION ::= {search｜addEntry｜removeEntry｜modifyEntry}
DAP-Returnable     OPERATION ::= {search｜addEntry｜removeEntry｜modifyEntry}
```

2.1.2.2.1.3 項で規定したコネクションパッケージは、別の抽象構文 directoryBindingAbstractSyntax を用いて実現される。これは抽象構文クラスの情報オブジェクトとして規定される。

```
inDirectoryBindingAbstractSyntax          ABSTRACT-SYNTAX ::= {
        DAPBinding-PDUs
        IDENTIFIED BY          id-as-directoryBindingAS}
DAPBinding-PDUs ::= CHOICE
{
        bind                   Bind {directoryBind},
        unbind                 Unbind {directoryUnbind}
}
```

2.1.2.2.2.2　ディレクトリアプリケーションコンテキスト

2.1.2.2.2.2.1　ディレクトリアクセスアプリケーションコンテキスト

dapContract は iNdirectoryAccessAC として実現される。このアプリケーションコンテキストは、アプリケーションコンテキストクラスの情報オブジェクトとして規定される。

```
iNdirectoryAccessAC       APPLICATION-CONTEXT ::=
{
        CONTRACT                        dapContract
        DIALOGUE MODE                   structured
        TERMINATION                 basic
        ABSTRACT SYNTAXES           {dialogue-abstract-syntax｜
                                        inDirectoryOperationsAbstractSyntax｜
                                        inDirectoryBindingAbstractSyntax}
        APPLICATION CONTEXT NAME    id-ac-directoryAccessAC
}
```

2.1.2.2.2.3　オペレーションコード

本勧告で定義されているパッケージに含まれるオペレーションは、ＩＴＵ－Ｔ勧告Ｘ．５１９で規定されており、オペレーションコードの割当もＩＴＵ－Ｔ勧告Ｘ．５１９からインポートされる。

2.1.2.2.2.4　エラーコード

本勧告で定義されているパッケージに含まれるエラーは、ＩＴＵ－Ｔ勧告Ｘ．５１９で規定されており、エラーコードの割当もＩＴＵ－Ｔ勧告Ｘ．５１９からインポートされる。

2.1.2.2.2.5　バージョンおよび拡張ルール

ディレクトリは分散する可能性があるため、2 つ以上のディレクトリアプリケーションエンティティが連携動作して要求に応えることもある。複数のディレクトリＡＥは、プロトコルバージョンが異なるかどうかは分からないが、異なるディレクトリサービス仕様の版に適合してインプリされているかもしれない。バージョンの値は、直接結合関係にある 2 つのディレクトリＡＥ間で共通の最も高い値に決められる。

2.1.2.2.2.5.1　バージョンの決定

ＤＡＰを利用したアソシエーション、すなわち結合を受信した際、取り決めを行ったバージョンは、ＤＵＡとそれに接続されたＤＳＡ間で交換されるプロトコルのポイントトゥポイントの側面にのみ影響を与える。当該アソシエーション上の後続の要求あるいは応答は、決められたバージョンには制約を受けない。

（注）ＤＡＰにおけるポイントトゥポイントの側面で、異なるプロトコルバージョンによって現在表されるものはない。

2.1.2.2.2.5.2　ＤＵＡ側

2.1.2.2.2.5.2.1　ＤＵＡ側での要求および応答処理

ＤＵＡが要求を行う際、サポートしている仕様の内最も高い版を用いることができる。要求の中の一つ以上の要素がクリティカルな場合、criticalExtensions パラメータ内に拡張の数を示す。

（注）CHOICE, ENUMERATED, （ENUMERATED として使用する）INTEGER 型の中で置き換えられた拡張の情報が、以前の仕様の版に従ってインプリされているＤＳＡで固有なオペレーションに対して本質的である場合は、拡張をクリティカルに指定することが推奨される。

応答を処理する際ＤＵＡは、

a) bit string 内で未知のビット割当はすべて無視する。

b) ENUMERATED あるいは（ENUMERATED として使用する）INTEGER 型の中の未知の名前付きの数値は、それが SET あるいは SEQUENCE のオプショナル要素である場合、すべて無視する。

c) SETs の中の未知のすべての要素，SEQUENCE の最後にある未知のすべての要素，SET あるいは SEQUENCE のオプショナル要素である CHOICEs の中の未知のすべての要素，は無視する。

（注）インプリによってはディレクトリ PDU の追加要素をローカルオプションとして無視しても良い。特に、未知の名前付きの数値，SET あるいは SEQUENCE の必須要素内の未知の CHOICE は、オペレーションを無効にすることなく無視できる。そのような要素の具体化は今後の課題である。

d) 未知の属性型および属性値を受信することを、プロトコル違反とは考えない。

e) 未知の属性型および属性値をオプションでユーザに通知する。

2.1.2.2.2.5.2.2　ＤＵＡ側でのエラーハンドリングのための拡張ルール

既知のエラータイプの中の未知の問題およびパラメータを処理する際、ＤＵＡは、

a) 未知の問題およびパラメータを受信することを、プロトコル違反とは考えない。（すなわち、TC-U-REJECT の発行あるいは対話のアボートは行わない。）

b) 追加エラー情報をオプションでユーザに通知する。

未知のエラータイプを処理する際、ＤＵＡは、

a) 未知のエラータイプを受信することを、プロトコル違反とは考えない。（すなわち、TC-U-REJECT の発行あるいはアプリケーションアソシエーションのアボートは行わない。）

b)エラーをオプションでユーザに通知する。

2.1.2.2.2.5.3　ＤＳＡ側での要求処理

オペレーションを実行するＤＳＡが、意味不明の criticalExtensions を検出した場合には、serviceError として unavailableCriticalExtension を返送する。

（注）一つ以上の 0 値を持つ criticalExtensions を受信した場合は、その値に対応する拡張が存在しないかあるいはクリティカルではないことを表す。criticalExtensions の中に 0 値が存在することは、APDU の対応する拡張が存在するかしないかとしては表せない。

一方、ＤＵＡからの要求を処理する際ＤＳＡは、

a) bit string 内で未知のビット割当はすべて無視する。

b) ENUMERATED あるいは（ENUMERATED として使用する）INTEGER 型の中の未知の名前付きの数値は、それが SET あるいは SEQUENCE のオプショナル要素である場合、すべて無視する。

c) SETs の中の未知のすべての要素，SEQUENCE の最後にある未知のすべての要素，SET あるいは SEQUENCE のオプショナル要素である CHOICEs の中の未知のすべての要素，は無視する。

（注）インプリによってはディレクトリ PDU の追加要素をローカルオプションとして無視しても良い。特に、未知の名前付きの数値，SET あるいは SEQUENCE の必須要素内の未知の CHOICE は、オペレーションを無効にすることなく無視できる。そのような要素の具体化は今後の課題である。

2.1.2.2.3　使用するサービスへのマッピング

本項はＤＡＰからＴＣサービスへのマッピングを定義する。

2.1.2.2.3.1　対話サービスへのマッピング

本項はディレクトリ結合およびディレクトリ結合解放サービスから、ＩＴＵ－Ｔ勧告Ｑ．７７１で定義されているＴＣ対話ハンドリングサービスへのマッピングを定義する。

2.1.2.2.3.1.1　結合

ディレクトリ結合サービスは以下のようにＴＣサービスに対応する。

(a) TC-BEGIN サービスはディレクトリ結合オペレーションを起動するために使用する。

(b) TC-CONTINUE サービスはディレクトリ結合オペレーションの成功を報告するために使用する。

(c) TC-U-ABORT サービスはディレクトリ結合オペレーションの失敗を報告するために使用する。

これらのサービスのパラメータの使い方を以下で記す。

2.1.2.2.3.1.1.1　TC-BEGIN へのマッピング

2.1.2.2.3.1.1.1.1　サービス品質（Quality of Service）

ＩＴＵ－Ｔ勧告Ｑ．７７１で規定されているように用い、特別な制約はつけない。

2.1.2.2.3.1.1.1.2　着アドレス（Destination Address）

このパラメータは、ＤＳＡの役を担うＳＤＦのアドレスを含む。

2.1.2.2.3.1.1.1.3　アプリケーションコンテキスト名（Application-Context-Name）

iNdirectoryAccessAC オブジェクトのアプリケーションコンテキスト名フィールドの値をとる。

2.1.2.2.3.1.1.1.4　発アドレス（Originating Address）

ＤＵＡが存在するＳＣＦのアドレスを含む。

2.1.2.2.3.1.1.1.5　対話 id（Dialogue Id）

ＩＴＵ－Ｔ勧告Ｑ．７７１で規定されているように用いる。

2.1.2.2.3.1.1.1.6　ユーザ情報（User Information）

このパラメータは、DirectoryBindArgument 型の値を含む。

2.1.2.2.3.1.1.1.7　コンポーネントプレゼント（Component Present）

ＩＴＵ－Ｔ勧告Ｑ．７７１で規定されているように用いる。

2.1.2.2.3.1.1.2　TC-CONTINUE へのマッピング

2.1.2.2.3.1.1.2.1　サービス品質（Quality of Service）

ＩＴＵ－Ｔ勧告Ｑ．７７１で規定されているように用い、特別な制約はつけない。

2.1.2.2.3.1.1.2.2　発アドレス（Originating Address）

ＩＴＵ－Ｔ勧告Ｑ．７７１で規定されているように用いる。存在する場合は、ＤＳＡの役を担うＳＤＦのアドレスを含む。

2.1.2.2.3.1.1.2.3　アプリケーションコンテキスト名（Application-Context-Name）

ＩＴＵ－Ｔ勧告Ｑ．７７１で規定されているように用いる。

2.1.2.2.3.1.1.2.4　対話 id（Dialogue Id）

ＩＴＵ－Ｔ勧告Ｑ．７７１で規定されているように用いる。

2.1.2.2.3.1.1.2.5　ユーザ情報（User Information）

このパラメータは、DirectoryBindResult 型の値を含む。

2.1.2.2.3.1.1.2.6　コンポーネントプレゼント（Component Present）

ＩＴＵ－Ｔ勧告Ｑ．７７１で規定されているように用いる。

2.1.2.2.3.1.1.3　TC-U-ABORT へのマッピング

2.1.2.2.3.1.1.3.1　サービス品質（Quality of Service）

ＩＴＵ－Ｔ勧告Ｑ．７７１で規定されているように用い、特別な制約はつけない。

2.1.2.2.3.1.1.3.2　対話 id（Dialogue Id）

ＩＴＵ－Ｔ勧告Ｑ．７７１で規定されているように用いる。

2.1.2.2.3.1.1.3.3　アボート理由（Abort Reason）

ＩＴＵ－Ｔ勧告Ｑ．７７１で規定されているように用いる。

2.1.2.2.3.1.1.3.4　アプリケーションコンテキスト名（Application-Context-Name）

ＩＴＵ－Ｔ勧告Ｑ．７７１で規定されているように用いる。ＳＤＦが、受信したアプリケーションコンテキスト名をサポートしていないという理由で対話を拒否した場合は、iNdirectoryAccessAC オブジェクトのアプリケーションコンテキスト名フィールドの値をとる。

2.1.2.2.3.1.1.3.5　ユーザ情報（User Information）

アボート理由が「対話拒否」の場合、このパラメータは DirectoryBindError 型の値を含む。それ以外の場合は存在しない。

2.1.2.2.3.1.2　結合解放

ディレクトリ結合解放サービスは TC-END サービスに対応する。TC-END サービスのパラメータの使い方を以下に記す。

2.1.2.2.3.1.2.1　サービス品質（Quality of Service）

ＩＴＵ－Ｔ勧告Ｑ．７７１で規定されているように用い、特別な制約はつけない。

2.1.2.2.3.1.2.2　アプリケーションコンテキスト名（Application-Context-Name）

この段階では本パラメータは使用しない。

2.1.2.2.3.1.2.3　対話 id（Dialogue Id）

ＩＴＵ－Ｔ勧告Ｑ．７７１で規定されているように用いる。

2.1.2.2.3.1.2.4　ユーザ情報（User Information）

このパラメータは空である。

2.1.2.2.3.1.2.5　コンポーネントプレゼント（Component Present）

ＩＴＵ－Ｔ勧告Ｑ．７７１で規定されているように用いる。

2.1.2.2.3.2　コンポーネントハンドリングサービスへのマッピング

ディレクトリＡＳＥサービスはＴＣコンポーネントハンドリングサービスに対応する。オペレーションとエラーのＴＣサービスへのマッピングは、ＩＴＵ－Ｔ勧告Ｑ．７７４に規定されている。TC-INVOKE-要求プリミティブのタイムアウトパラメータは、以下の表にしたがって設定される。

表１　ＤＡＰオペレーションのＴＣタイマ値

オペレーション	タイムアウト
探索（search）	中
エントリ変更（modifyEntry）	中
エントリ追加（addEntry）	中
エントリ削除（removeEntry）	中

2.1.2.2.4　コンフォーマンス

本項は、本規定に対するコンフォーマンスの要求条件を定義する。

2.1.2.2.4.1　ＳＣＦによるコンフォーマンス

本規定に対するコンフォーマンスを要求するＳＣＦのインプリは、2.1.2.2.4.1.1 項から 2.1.2.2.4.1.3 項で規定された要求条件を満足する必要がある。

2.1.2.2.4.1.1　ステートメントの要求条件

以下が記述される。

a) コンフォーマンスが要求され、ＳＣＦが送信可能な iNdirectoryAccessAC アプリケーションコンテキストのオペレーション。

b) コンフォーマンスが要求されるセキュリティレベル（無し，簡易，厳密）。

c) コンフォーマンスが要求され、ＳＣＦが起動可能なＩＴＵ－Ｔ勧告Ｘ．５１１｜ＩＳＯ／ＩＥＣ ９５９４－３の 7.3.1 項の表に列挙されている拡張。

2.1.2.2.4.1.2　静的な要求条件

ＳＣＦは、

a) 2.1.2.2.2 項で抽象構文によって定義される iNdirectoryAccessAC アプリケーションコンテキストをサポートする能力をもつ。

b) 2.1.2.2.4.1.1 項 c)にあるコンフォーマンスが要求された拡張に対して適合する。

2.1.2.2.4.1.3　動的な要求条件

ＳＣＦは、

a) 2.1.2.2.3 項で定義されている「使用するサービスへのマッピング」に適合する。

b) 2.1.2.2.2.5.2 項で定義されている拡張手順の規則に適合する。

2.1.2.2.4.2　ＳＤＦによるコンフォーマンス

本規定に対するコンフォーマンスを要求するＳＤＦのインプリは、2.1.2.2.4.2.1 項から 2.1.2.2.4.2.3 項で規定された要求条件を満足する必要がある。

2.1.2.2.4.2.1　ステートメントの要求条件

以下が記述される。

a) コンフォーマンスが要求されるアプリケーションコンテキスト。本勧告の現在の版では、iNdirectoryAccessAC アプリケーションコンテキストに対するコンフォーマンスのみが要求される。

（注）アプリケーションコンテキストはここに記述するもの以外は分割されない。特に特定のオペレーションに対するコンフォーマンスは要求されない。

b) コンフォーマンスが要求されるセキュリティレベル（無し，簡易，厳密）。

c) コンフォーマンスが要求される属性型。また、構文 DirectoryString に基づく属性に対して、UNIVERSAL STRING 選択のためのコンフォーマンスが要求されるか否か。

d) コンフォーマンスが要求されるオブジェクトクラス。

e) コンフォーマンスが要求され、ＳＤＦが応答可能なＩＴＵ－Ｔ勧告Ｘ．５１１｜ＩＳＯ／ＩＥＣ ９５９４－３の 7.3.1 項の表に列挙されている拡張。

f) ＩＴＵ－Ｔ勧告Ｘ．５０１｜ＩＳＯ／ＩＥＣ ９５９４－２の 8.8 項，およびＩＴＵ－Ｔ勧告Ｘ．５１１｜ＩＳＯ／ＩＥＣ ９５９４－３の 7.6, 7.8.2, 9.2.2 項で定義されているコレクティブ属性に対して、コンフォーマンスが要求されるか否か。

g) ＩＴＵ－Ｔ勧告Ｘ．５１１｜ＩＳＯ／ＩＥＣ ９５９４－３の 7.6, 7.8.2, 9.2.2 項で定義されているハイアラキカル属性に対して、コンフォーマンスが要求されるか否か。

h) コンフォーマンスが要求される、ＩＴＵ－Ｔ勧告Ｘ．５０１｜ＩＳＯ／ＩＥＣ ９５９４－２で定義されているオペレーショナル属性型および他のオペレーショナル属性型。

i) ＩＴＵ－Ｔ勧告Ｘ．５１１｜ＩＳＯ／ＩＥＣ ９５９４－３の 7.7.1 項で定義されている別名の返送に対して、コンフォーマンスが要求されるか否か。

j) ＩＴＵ－Ｔ勧告Ｘ．５１１｜ＩＳＯ／ＩＥＣ ９５９４－３の 7.7.6 項で記述されているように、返送されたエントリ情報が完全であることを示すためのコンフォーマンスが要求されるか否か。

k) ＩＴＵ－Ｔ勧告Ｘ．５１１｜ＩＳＯ／ＩＥＣ ９５９４－３の 11.3.2 項で記述されているように、補助オブジェクトクラスを特定して、値の追加および／あるいは削除を行うために、オブジェクトクラスの属性を変更するためのコンフォーマンスが要求されるか否か。

l) 基本アクセス制御に対してコンフォーマンスが要求されるか否か。

m) 簡易化アクセス制御に対してコンフォーマンスが要求されるか否か。

n) コンフォーマンスが要求されるネームバインディング。

o) ＩＴＵ－Ｔ勧告Ｘ．５０１｜ＩＳＯ／ＩＥＣ ９５９４－２で定義されているように、ＳＤＦがコレクティブ属性の管理ができるか否か。

p) コンテキストに対してコンフォーマンスが要求されるか否か。

2.1.2.2.4.2.2　静的な要求条件

ＳＤＦは、

a) 2.1.2.2.2 項で抽象構文によって定義されている、コンフォーマンスが要求されるアプリケーションコンテキストをサポートする能力をもつ。

b) ＩＴＵ－Ｔ勧告Ｘ．５０１｜ＩＳＯ／ＩＥＣ ９５９４－２で抽象構文によって定義されている情報の枠組みをサポートする能力をもつ。

c) 抽象構文によって定義されている、コンフォーマンスが要求される属性型をサポートする能力をもつ。

d) 抽象構文によって定義されている、コンフォーマンスが要求されるオブジェクトクラスをサポートする能力をもつ。

e) 2.1.2.2.4.2.1 項でコンフォーマンスが要求される拡張に適合する。

f) コレクティブ属性に対してコンフォーマンスが要求されるのであれば、ＩＴＵ－Ｔ勧告Ｘ．５１１｜ＩＳＯ／ＩＥＣ ９５９４－３の 7.6, 7.8.2, 9.2.2 項で定義されている関連の手順を実行する能力をもつ。

g) ハイアラキカル属性に対してコンフォーマンスが要求されるのであれば、ＩＴＵ－Ｔ勧告Ｘ．５１１｜ＩＳＯ／ＩＥＣ ９５９４－３の 7.6, 7.8.2, 9.2.2 項で定義されている関連の手順を実行する能力をもつ。

h) コンフォーマンスが要求されるオペレーショナル属性型をサポートする能力をもつ。

i) 基本アクセス制御に対してコンフォーマンスが要求されるのであれば、基本アクセス制御の定義に適合するＡＣＩアイテムを保持する能力をもつ。

j) 簡易化アクセス制御に対してコンフォーマンスが要求されるのであれば、簡易化アクセス制御の定義に適合するＡＣＩアイテムを保持する能力をもつ。

2.1.2.2.4.2.3　動的な要求条件

ＳＤＦは、

a) 2.1.2.2.3 項で定義されている「使用するサービスへのマッピング」に適合する。

b) 2.1.2.2.2.5.3 項で定義されている拡張手順の規則に適合する。

c) 基本アクセス制御に対してコンフォーマンスが要求されるのであれば、基本アクセス制御の手順に従ってＳＤＦ内の情報を保護する能力をもつ。

d) 簡易化アクセス制御に対してコンフォーマンスが要求されるのであれば、簡易化アクセス制御の手順に従ってＳＤＦ内の情報を保護する能力をもつ。

2.1.2.3　Ｘ．５０１プロファイル

勧告Ｘ．５０１は、ディレクトリによって提供されるサービスをサポートするために必要な一般的な情報モデルを示している。ＩＮに適用する場合、一般的な情報モデルは勧告Ｘ．５０１の 1 項から 7 項に適合すべきである。ただし、その一部にはサポートされる必要がないものがある。例えば、使用に関してローカルに決められるＤＩＴ内容規則である。

本勧告の範囲外のものもある。ＩＮ ＣＳ－１（改）では考慮されていない事項である。したがって、勧告Ｘ．５０１における以下のものは適用されない。

－16.2.3 項の節 f), h), i)

－16.2.4 項の節 a)で、比較オペレーションは使用せず、代わりに探索オペレーションを使用する。したがって、Compare 許可は FilterMatch 許可に置き換わる。

2.1.2.4　ＩＴＵ－Ｔ　ＩＮ　ＣＳ－１をサポートするためのＸ．５００の拡張

2.1.2.4.1　Entry Information Selection の拡張

この EntryInformationSelection の拡張仕様は、コンテキストに関係するすべての属性に使用される contextAssertions 要素を含んでいる。ここでは拡張分のみを記述し、その他の部分は 1993 年版のＸ．５００シリーズ勧告で規定されているものを使用する。

```
EntryInformationSelection ::= SET {
    attributes CHOICE {
            allUserAttributes     [0]    NULL,
            select                [1]    SET OF AttributeType
    -- 空のセットは属性が要求されないことを意味する-- } DEFAULT allUserAttributes : NULL,
    infoTypes [2]     INTEGER {
            attributeTypesOnly (0),
            attributeTypesAndValues (1)
            } DEFAULT attributeTypesAndValues,
    extraAttributes CHOICE {
                allOperationalAttributes    [3]    NULL,
                select                      [4]    SET OF AttributeType
                } OPTIONAL,
    contextAssertions     SET OF TypeAndContextAssertion OPTIONAL,
    returnContexts        BOOLEAN          DEFAULT FALSE }
TypeAndContextAssertion ::= SEQUENCE {
    type                  AttributeType,
    contextAssertion      ContextAssertion }
```

contextAssertions 要素は、type がリストされ関連する contextAssertion を満足しないような属性値は返送されるエントリ情報から削除すべきであることを規定するために用いる。しかしながら、いずれの属性値も contextAssertions を満足しないならば、fallback が真であるコンテキストをもついかなる値も返送する。

returnContexts 要素は、ディレクトリに対して属性値をそのコンテキストとともに可能であれば返送することを要求するためのものである。これが指定されていない場合は、エントリ情報を返送する前にすべてのコンテキストは取り除かれる。

2.1.2.4.2　Modify Entry Argument および Modify Entry Result の拡張

ModifyEntryArgument の拡張には、新たな selection パラメータ，EntryModification に対する alterValues の追加があり、ModifyEntryResult の拡張には、オプションの information パラメータの追加がある。ここでは拡張部分のみを記述し、その他の部分は 1993 年版のＸ．５００シリーズ勧告で規定されているものを使用する。

```
ModifyEntryArgument             ::=   OPTIONALLY-SIGNED {SET {
    object                      [0]   Name,
    changes                     [1]   SEQUENCE OF EntryModification,
    selection                   [2]   EntryInformationSelection   OPTIONAL,
    COMPONENTS OF                     CommonArguments}}
ModifyEntryResult               ::=   CHOICE {
    null        NULL,
    information                       OPTIONALLY-SIGNED SET {
        entry                   [0]   EntryInformation   OPTIONAL,
        COMPONENTS OF                 CommonResults}}
EntryModification ::= CHOICE {
    addAttribute                [0]   Attribute,
    removeAttribute             [1]   AttributeType,
    addValues                   [2]   Attribute,
    removeValues                [3]   Attribute,
    alterValues                 [4]   AttributeTypeAndValues,
    resetValue                  [5]   AttributeType}
```

変更 alterValues は、アーギュメント内の属性型によって属性を特定し、当該属性のすべての値に加えるべき値を規定する。これは属性が INTEGER あるいは REAL の場合にのみ与えられるものであり、その他の場合は AttributeError になる。

変更 resetValue は、その型によって属性を特定し、当該属性において fallback が偽である属性値コンテキストを持つすべての値を削除する。

selection パラメータは、オペレーション結果中に情報を返送するか否かを制御し、返送すべき属性および値を特定する EntryInformationSelection を規定する。

要求が成功した場合は結果が返送される。もしオペレーションのアーギュメントで selection を規定しなければ null 結果が返送される。そうでなければ、要求された属性型および値を含む情報を持つエントリ結果が返送される。

追加すべき属性あるいは属性値は、コンテキスト有りで規定されることもあればコンテキストなしで規定されることもある。コンテキスト有りで追加することは、格納されているディレクトリ情報にいかにコンテキストを追加するかである。属性値を削除する場合、コンテキストは与えられないが、存在しているいかなるコンテキストも属性値とともに削除される。既存の属性値に関するコンテキストを変更する場合は、まず属性値を削除し、次に新たなコンテキストリストを付けて置き換えを行う。

2.1.2.4.3 結合オペレーションのセキュリティエラーの拡張

SecurityProblem の仕様を拡張して blockedCredentials の問題を追加し、結合要求失敗時の応答で用いる。ここでは拡張部分のみを記述し、その他の部分は 1993 年版のＸ．５００シリーズ勧告で規定されているものを使用する。

```
SecurityProblem          ::=    INTEGER {
        inappropriateAuthentication (1),
        invalidCredentials (2),
        insufficientAccessRights (3),
        invalidSignature (4),
        protectionRequired (5),
        noInformation (6),
        blockedCredentials (7)}
```

セキュリティの理由（例、不当なパスワードが連続して何度も使用された）により資格証明をブロックする場合に、blockedCredentials を使用する。このエラーを返送することの判断は、ＤＳＡで実施されているセキュリティの考え方によって支配される。

blockedCredentials エラーは結合エラーに含まれる。

2.1.2.4.4　属性コンテキストの仕様

以下のツールは 1993 年版Ｘ．５００勧告にあるツールへの拡張を含み、ＩＮ ＣＳ－１（改）のＳＣＦ－ＳＤＦ間インタフェースの要求条件をサポートするために必要となる情報モデル中の属性コンテキストを規定するために用いる。ここでは拡張部分のみを記述し、その他の部分は 1993 年版のＸ．５００シリーズ勧告で規定されているものを使用する。

エントリ，属性およびコンテキストの関係を図 3-2-1/ＪＴ－Ｑ１２１８に示す。またコンテキストの定義を以下に示す。

・コンテキスト…属性値に付与できる特性であり、値の適用性あるいは有効性を決定するために用いる情報を規定する。

・コンテキスト型…コンテキスト内の要素であり、該コンテキストの型あるいは目的を表す。

・コンテキストリスト…ある属性に属するコンテキストの集合。

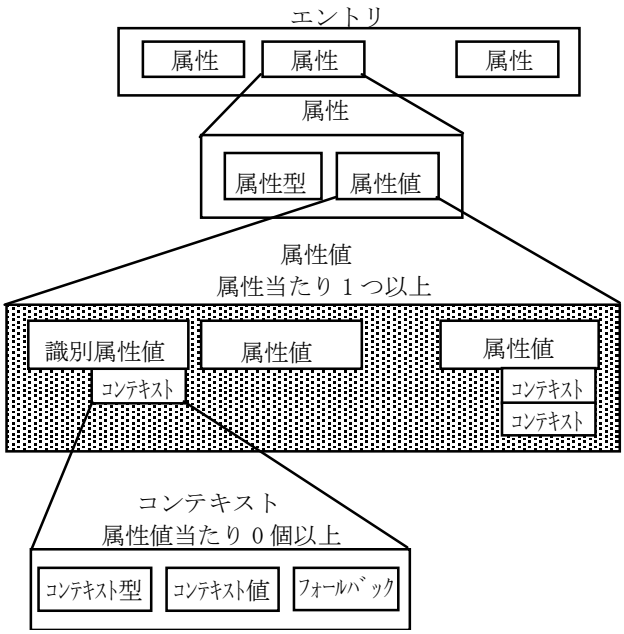


図 3-2-1/ＪＴ－Ｑ１２１８　エントリの構造

この属性の拡張仕様には、新たな valuesWithContext 要素が追加され、これはあるコンテキストに関連するあらゆる属性値に対して使用される。

```
Attribute                    ::=    SEQUENCE {
    type        ATTRIBUTE.&id ({SupportedAttributes}),
    values      SET SIZE (1..MAX) OF ATTRIBUTE.&Type ({SupportedAttributes}{@type}),
    valuesWithContextSET SIZE (1..MAX) OF SEQUENCE {
                value           ATTRIBUTE.&Type ({SupportedAttributes}{@type}) OPTIONAL,
                contextList     SET SIZE (1..MAX) OF Context } OPTIONAL}
```

（注）属性値は 0 個以上の関連するコンテキストをもつ可能性がある。しかしながら、ディレクトリシステムの 1988 年版と 1993 年版のコンパチビリティの理由から、プロトコル上はすべての属性はコンテキストをもたない値を少なくとも一つはもつとして表現する必要がある。ある属性のすべての値にコンテキストがある場合は、属性をコード化するＤＵＡもしくはＤＳＡが、属性の表現を強制的に変更する。すなわち、ある値（あるいは同一のコンテキストリストをもつ複数の値）を選択し、コンテキストリストを付けずに values フィールドに設定する。次に valuesWithContext フィールドには、value フィールドを空きにし、削除したコンテキストリストを contextList フィールドに設定する。これは、value フィールドを contextValues から削除すべき場合のみである。

属性は値が一つでも複数でもよい。値が一つの属性に対してアクセスがあった場合には、常に values フィールド内の値が一つであることをディレクトリは保証する。値が関連するコンテキストリストをもち、そのリストがすべて異なる場合には、追加の値が提供される。そのようなコンテキストは相互に排他であることが期待されるため、値が一つの属性では常に一つの値のみが有効である。ディレクトリはこの排他性を強要することはないが、値が一つの属性の値が一つであることを保証するために、ディレクトリ情報を返送するときはいつも任意の値を必要に応じて削除する。

属性値に関連したコンテキストの集合あるいはリストは、その属性値の適用性あるいは有効性を判断するために用いる付加情報を提供する。例えば、コンテキストリストは属性値の言語，時間的あるいは地理的有効性などを表す。

各コンテキストは属性と同様に、型フィールド，その型によって構文が決まる値フィールド，フォールバックフラグからなる。本項でさらに定義する CONTEXT 情報オブジェクトクラスの記法を用いて、コンテキストは以下のように定義される。

```
Context ::= SEQUENCE     {
    contextType             CONTEXT.&id ({SupportedContexts}),
    contextValues           SET SIZE(1..MAX) OF CONTEXT.&Type ({SupportedContexts}{@type}),
    fallback                BOOLEAN DEFAULT FALSE}
```

contextType はオブジェクト識別子であり、CONTEXT 情報オブジェクトクラスを用いて規定する。これは Context によって表されるコンテキスト情報の種類を規定する。

contextValues はコンテキスト情報の実際の値である。これは関連する属性値が適用可能あるいは有効である条件を規定する。

fallback は、参照条件を与えたときにそのコンテキストが無効の場合であっても、関連する属性値をディレクトリが返送すべきかどうかを規定する。もし有効な値がなかった場合は、ディレクトリは fallback が真である属性の値を返送する。

```
AttributeValueAssertion ::= SEQUENCE {
    type                ATTRIBUTE.&id ({SupportedAttributes}),
    assertion           ATTRIBUTE.&equality-match.&AssertionType
                                ({SupportedAttributes}{@type}),
    contextAssertion    ContextAssertion    OPTIONAL }
ContextAssertion ::= Context
```

AttributeValueAssertion 内の contextAssertion はオプショナルである。contextAssertion が指定された場合は、アサーションは指定されたコンテキスト内に存在する属性値に対してのみ有効となる。

contextAssertion が提供されない場合は、ＤＳＡは下記に示すように、エントリを制御する contextAssertion サブエントリ内のデフォルト contextAssertion を捜す。そのようなサブエントリがない場合は、ＤＳＡはローカルで定義したデフォルトを適用しても良い。そのようなデフォルトはローカルなパラメータを反映する。例えば、ＤＳＡを展開した場所の言語あるいは位置，さらには現在時刻などであるが、それが応答する各ＤＵＡに応じて変更することもできる。

contextAssertion が指定されずデフォルトも利用できない場合は、アサーションはあらゆるコンテキスト内にある属性のあらゆる値に対して有効となる。

相対識別名を含む値は関連するコンテキスト情報をもつかもしれないが、コンテキストは相対識別名には現れない。エントリはただ一つの相対識別名をもち、これはエントリ情報を参照するときに使用するコンテキストアサーションとは独立である。

コンテキスト型の定義には以下のものが含まれる。

－コンテキストの構文を規定する

－コンテキストの意味を定義する

－コンテキスト型にオブジェクト識別子を割り当てる

型のオブジェクト識別子が合致し、値の各要素もＡＳＮ．１にしたがって合致した場合、2 つのコンテキストは合致する。

コンテキストは CONTEXT 情報オブジェクトクラスを用いて定義される。

```
CONTEXT ::= CLASS {
    &Type,
    &id                 OBJECT IDENTIFIER UNIQUE }
WITH SYNTAX {
    WITH SYNTAX     &Type,
    ID              &id }
```

コンテキストを定義する際、その仕様に含まれるのは、コンテキストの意味の記述と、コンテキストを評価すべき条件がオペレーション中に指定されていない場合にＤＳＡが提供するデフォルト値の性質である。

コンテキストアサーションのデフォルト値を提供するサブエントリは、以下のように定義される。

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

このオブジェクトクラスのサブエントリは、contextAssertionDefault 属性を含む。

```
contextAssertionDefault    ATTRIBUTE ::= {
    WITH SYNTAX                ContextAssertion
    EQUALITY MATCHING RULE     objectIdentifierFirstComponentMatch
    USAGE directoryOperation
    ID                         id-oa-contextAssertionDefault }
```

コンテキストが評価され、ユーザからコンテキストアサーションが提供されない場合はいつも、ディレクトリはアクセスされるエントリを制御するコンテキストアサーションサブエントリ内のこの属性の値に等しいデフォルトアサーションを提供する。そのような属性が存在しない場合は、ディレクトリはローカル規則に従ってデフォルトを提供する。

contexts 属性は、サブスキーマ内で使用されるコンテキスト型を規定する。

```
contexts    ATTRIBUTE ::= {
WITH SYNTAX                    ContextDescription
EQUALITY MATCHING RULE         objectIdentifierFirstComponentMatch
USAGE                          directoryOperation
ID                             id-soa-attributeValueContext }
ContextDescription ::= SEQUENCE {
    identifier                 CONTEXT.&id,
    name                       SET OF DirectoryString {ub-schema} OPTIONAL,
    description                DirectoryString {ub-schema} OPTIONAL,
    obsolete                   BOOLEAN DEFAULT FALSE,
    information    [0]         ContextInformation }
```

contexts 属性の値の identifier 要素は、コンテキスト型を特定するオブジェクト識別子である。

contexts 属性は複数の値をとり得るものであり、各値が一つのコンテキストを記述する。

```
ContextInformation ::= SEQUENCE {
      syntax        DirectoryString {ub-schema} OPTIONAL }
```

syntax 要素は、対応する情報オブジェクトクラスで導入された対応する記法と同じ意味をもつ。

属性のデフォルト値を扱うためには、ディレクトリ内の overrideContext コンテキストを使用することを薦める。このコンテキストの定義は以下のようになる。

```
overrideContext    CONTEXT ::= {
    WITH SYNTAX        NULL,
    ID                 id-c-override }
```

これにより属性のデフォルト値は overrideContext コンテキストと関連し、フォールバックフラグは真に設定する。他のすべての属性値（すなわち非デフォルト値）は、コンテキストがない（すなわち関連するコンテキストをもたない）かもしくはフォールバックを偽に設定したフォールバックをもつかのいずれかである。エントリ属性のデフォルト値は、addEntry オペレーションを用いてエントリを生成するときに割り当てることができる。addEntry オペレーションの要求は、属性のデフォルト値のみもしくはデフォルトおよび非デフォルト値の両方を割り当てることができる。

2.1.2.4.5　アクセス制御の拡張

ＩＮ　ＣＳ－１（改）のＳＣＦ－ＳＤＦ間インタフェースの要求条件をサポートするために、1993 年版のＸ．５００のアクセス制御情報（ＡＣＩ）の仕様に対して以下の拡張が必要となる。ここでは拡張部分のみを記述し、その他の部分は 1993 年版のＸ．５００シリーズ勧告で規定されているものを使用する。

ProtectedItems の定義は以下のように拡張される。

```
ProtectedItems ::= SEQUENCE {
    entry                          [0]    NULL OPTIONAL,
    allUserAttributeTypes          [1]    NULL OPTIONAL,
    attributeType                  [2]    SET OF AttributeType OPTIONAL,
    allAttributeValues             [3]    SET OF AttributeType OPTIONAL,
    allUserAttributeTypesAndValues [4]    NULL OPTIONAL,
    attributeValue                 [5]    SET OF AttributeTypeAndValue OPTIONAL,
    selfValue                      [6]    SET OF AttributeType OPTIONAL,
    rangeOfValues                  [7]    Filter OPTIONAL,
    maxValueCount                  [8]    SET OF MaxValueCount OPTIONAL,
    maxImmSub                      [9]    INTEGER OPTIONAL,
    restrictedBy                   [10]   SET OF RestrictedValue OPTIONAL}
MaxValueCount ::= SEQUENCE {
    type                AttributeType,
    maxCount            INTEGER}
RestrictedValue ::= SEQUENCE {
    type                AttributeType,
    valuesIn            AttributeType}
```

・rangeOfValues－指定されたフィルタに対し正しく評価するための値である。

・maxValueCount－属性あるいは属性値を追加する場合は、該属性に対して本アイテムで特定した最大許容属性値数の制限に従う。

・maxImmSub－エントリに対してそれに従属するものを追加する場合は、本アイテムで特定した最大許容従属数の制限に従う。エントリが他のＤＳＡにその従属を持っている場合は、この制限はマスタエントリが存在するＤＳＡの従属数に適用される。

・restrictedBy－属性 type に対して許容される値が属性 valuesIn にあることを意味する。どちらの属性も、同一エントリ内にあり、同一の構文であることが必要である。

なお、アイテム rangeOfValues は、属性値を陽に規定するものとして扱える。

2.1.2.4.6　拡張テーブルの拡張

ＩＮに必要な拡張を考慮すると、クリティカル拡張を含むテーブルの変更分は以下のようになる。

拡張	ID	適用オペレーション	重要性	定義
userContexts	13	Search, AddEntry, RemoveEntry	non-critical	
reverseMatch	16	Search	non-critical	
selectionOnModify	18	ModifyEntry	non-critical	
extendedEntryMods	19	ModifyEntry	critical	

2.1.2.4.7 Matching Rule Assertion の拡張

Matching Rule Assertion は、逆マッチングを容易にするために以下のように拡張する。

```
Matching Rule Assertion ::= SEQUENCE {
    matchingRule    [1]    SET SIZE (1..MAX) OF MATCHING-RULE.&id,
    type            [2]    AttributeType OPTIONAL,
    matchValue      [3]    MATCHING-RULE.&AssertionType (CONSTRAINED
BY {
            -- matchValue は、matchingRule で指定された MATHING-RULE 情報オブジェ
            -- クトの内の一つの&AssertionType フィールドで規定された型の値でなけれ
            -- ばならない。  --}),
    dnAttributes    [4]    BOOLEAN DEFAULT FALSE,
    reverseMatch    [5]    BOOLEAN DEFAULT FALSE}
```

もし reverseMatch が真であれば、比較する複数の要素（すなわち、目的の値とディレクトリ内のある属性の値）は比較のために逆転される。これは例えば、蓄積されている値（電話番号の市外局番）が目的の値（電話番号）の最初の部分文字と一致することを求めて、ディレクトリ内の値と目的の値とを比較するために用いることができる。

2.1.2.5 ＩＮ ＣＳ－１（改）のためのディレクトリ抽象サービスのＡＳＮ．１プロファイル

2.1.2.5.1 拡張したＸ．５００シリーズのＡＳＮ．１モジュール

以下で記述するＡＳＮ．１モジュールには、ＩＮをサポートするために必要となるディレクトリ仕様への変更がすべて含まれている。モジュールは変更された型を用いているため、それらの変更によって影響される定義もまた含まれる。1993 年版に対する修正はボールド体で示す。

2.1.2.5.1.1 Ｘ．５０１勧告の拡張

本モジュールは、ＩＮの要求を満たすためにＸ．５０１勧告に対してなされた拡張を含む。

```
ExtendedInformationFramework
{ccitt recommendation q 1218 modules(0) informationFramework(11) version1(0)}
DEFINITIONS
BEGIN

--EXPORTS ALL--
--型および値は、ディレクトリ抽象サービスおよびディレクトリアクセスプロトコルのＩＮプロファイルを
定義するＡＳＮ．１モジュールの中で使用するためにエクスポートされる。

IMPORTS
    id-oa,id-soa,id-c,AttributeTypeAndValue,ATTRIBUTE,OBJECT-CLASS,AttributeType,AttributeValue
    FROM InformationFramework {joint-iso-ccitt ds(5) module(1) informationFramework(1) 2}
    DirectoryString{}, objectIdentifierFirstComponentMatch
    FROM SelectedAttributeTypes {joint-iso-ccitt ds(5) module(1) selectedAttributeTypes(5) 2} Filter
    FROM DirectoryAbstractService {joint-iso-ccitt ds(5) module(1) directoryAbstractService(2) 2}
    ub-schema
    FROM UpperBounds {joint-iso-ccitt ds(5) module(1) upperBounds(10) 2};
```

```
Attribute ::= SEQUENCE {
        type            AttributeType({SupportedAttributes}),
        values          SET SIZE (1..MAX) OF AttributeValue({SupportedAttributes}{@type}),
        valuesWithContext       SET SIZE (1..MAX) OF SEQUENCE {
        value                   ATTRIBUTE.&TYPE({SupportedAttributes}{@type}) OPTIONAL,
        contextList             SET SIZE (1..MAX) OF Context} OPTIONAL}
Context :: = SEQUENCE {
        contextType             CONTEXT.&id ({SupportedContexts}),
        contextValues           SET SIZE (1..MAX) OF
                                        CONTEXT.&Type ({SupportedContexts}{@type}),
        fallback                BOOLEAN DEFAULT FALSE }
AttributeValueAssertion :: = SEQUENCE {
        type                    AttributeType(SupportedAttributes),
        assertion               ATTRIBUTE.&equality-match.&AssertionType ({SupportedAttributes}{@type}),
        contextAssertion        ContextAssertion OPTIONAL}
        ContextAssertion :: = Context
SupportedAttributes ATTRIBUTE :: = {}...
        CONTEXT :: = CLASS {
                &Type,
                &id             OBJECT IDENTIFIER UNIQUE}
        WITH SYNTAX {
                WITH SYNTAX     &Type,
                ID              &id}
        contextAssertionSubentry OBJECT-CLASS ::= {
                KIND            auxiliary
                ID              id-sc-contextAssertionSubentry}
        id-sc-contextAssertionSubentry ::= {id-sc}
        contextAssertionDefault ATTRIBUTE :: = {
                WITH SYNTAX                     ContextAssertion
                EQUALITY MATCHING RULE          objectIdentifierFirstComponentMatch
                USAGE                           directoryOperation
                ID                                      id-oa-contextAssertionDefault}
        id-oa-contextAssertionDefault :: = {id-oa}
        contexts ATTRIBUTE :: = {
        WITH SYNTAX                             ContextDescription
        EQUALITY MATCHING RULE                  objectIdentifierFirstComponentMatch
        USAGE                                   directoryOperation
        ID                                      id-soa-attributeValueContext}
        id-soa-attributeValueContext :: = {id-soa}
        ContextDescription ::= SEQUENCE {
                identifier              CONTEXT.&id,
                name                    SET OF DirectoryString {ub-schema} OPTIONAL,
```

```
        description             DirectoryString {ub-schema} OPTIONAL,
        obsolete                BOOLEAN DEFAULT FALSE,
        information     [0]     ContextInformation}
    ContextInformation ::= SEQUENCE {
        syntax          DirectoryString {ub-schema} OPTIONAL}
    overrideContext CONTEXT :: = {
        WITH SYNTAX     NULL,
        ID                      id-c-override}
    id-c-override ::= {id-c}
    ProtectedItems ::= SEQUENCE {
        entry                           [0]     NULL OPTIONAL,
        allUserAttributeTypes           [1]     NULL OPTIONAL,
        attributeType                   [2]     SET OF AttributeType OPTIONAL,
        allAttributeValues              [3]     SET OF AttributeType OPTIONAL,
        allUserAttributeTypesAndValues  [4]     NULL OPTIONAL,
        attributeValue                  [5]     SET OF AttributeTypeAndValue OPTIONAL,
        selfValue                       [6]     SET OF AttributeType OPTIONAL,
        rangeOfValues                   [7]     Filter OPTIONAL,
        maxValueCount                   [8]     SET OF MaxValueCount OPTIONAL,
        maxImmSub                       [9]     INTEGER OPTIONAL,
        restrictedBy                    [10]    SET OF RestrictedValue OPTIONAL}
    MaxValueCount ::= SEQUENCE{
        type            AttributeType,
        maxCount        INTEGER}
    RestrictedValue ::= SEQUENCE{
        type            AttributeType,
        valuesIn        AttributeType}

END
```

2.1.2.5.1.2　Ｘ．５１１勧告の拡張

本モジュールは、ＩＮの要求を満たすためにＸ．５１１勧告に対してなされた拡張を含む。

```
ExtendedDirectoryAbstractService
{ccitt recommendation q 1218 modules(0) abstractService(14) version1(0)}

DEFINITIONS

BEGIN
--EXPORTS ALL--
--型および値は、ディレクトリ抽象サービスおよびディレクトリアクセスプロトコルのＩＮプロファイルを
定義するＡＳＮ．１モジュールの中で使用するためにエクスポートされる。

IMPORTS
        informationFramework, distributedOperations dap
            FROM UsefulDefinitions {joint-iso-ccitt ds(5) module(1) usefulDefinitions(0) 2}
        AccessPoint
            FROM DistributedOperations distributedOperations
        id-opcode-read, id-opcode-compare, id-opcode-abandon, id-opcode-list, id-opcode-search,
        id-opcode-addEntry, id-opcode-removeEntry, id-opcode-modifyEntry, id-opcode-modifyDN,
        id-errcode-abandoned, id-errcode-abandonFailed, id-errcode-attributeError,
        id-errcode-nameError, id-errcode-referral, id-errcode-securityError, id-errcode-serviceError,
        id-errcode-updateError
            FROM DirectoryAccessProtocol dap
        OPERATION, ERROR
            FROM            Remote-Operations-Information-Objects {joint-iso-ccitt remote-operations(4)
                            informationObjects(5) version1(0)}
        MATCHING-RULE, Name, AttributeValue, AttributeType, AttributeTypeAndValue
            FROM InformationFramework {joint-iso-ccitt ds(5) module(1) informationFramework(1) 2}
        AttributeValueContext, AttributeValueAssertion, Attribute
            FROM            ExtendedInformationFramework
                            {ccitt recommendation q 1218 modules(0) informationFramework(11) version1(0)}
        OPTIONALLY-SIGNED, CommonArguments, CommonResults, Versions, ServiceProblem,
        attributeError, nameError, serviceError, referral, abandoned, updateError, PageResultsRequest,
        PartialOutcomeQualifier, DirectoryBindResult, DirectoryBindArgument
            FROM DirectoryAbstractService {joint-iso-ccitt ds(5) module(1) directoryAbstractService(2) 2}
        ContextAssertion
            FROM ExtendedInformationFramework
                            {ccitt recommendation q 1218 modules(0) informationFramework(11) version1(0)}
;
```

```
EntryInformationSelection ::= SET {
        attributes          CHOICE {
            allUserAttributes           [0]   NULL,
            select                      [1]   SET OF AttributeType
-- 空のセットは、要求される属性がないことを意味する。-- } DEFAULT allUserAttributes : NULL,
        infoTypes                       [2]   INTEGER {attributeTypesOnly (0),
                                                attributeTypesAndValues (1)
                                                } DEFAULT attributeTypesAndValues,
        extraAttributes     CHOICE {
            allOperationalAttributes    [3]   NULL,
            select                      [4]   SET OF AttributeType
                                              } OPTIONAL,
        contextAssertions               SET OF TypeAndContextAssertion OPTIONAL,
        returnContexts                  BOOLEAN DEFAULT FALSE}
TypeAndContextAssertion ::= SEQUENCE {
        type                            AttributeType,
        contextAssertion                ContextAssertion}
EntryInformation :: = SEQUENCE {
        name            Name,
        fromEntry       BOOLEAN DEFAULT TRUE,
        information     SET OF CHOICE {
                            attributeType                   AttributeType,
                            attribute                       Attribute} OPTIONAL,
        incompleteEntry             [3]   BOOLEAN DEFAULT FALSE
-- 1988 年版システムにはない。-- }
Filter ::= CHOICE {
        item                [0]   FilterItem,
        and                 [1]   SET OF Filter,
        or                  [2]   SET OF Filter,
        not                 [3]   Filter}
FilterItem :: = CHOICE {
        equality            [0]   AttributeValueAssertion,
substrings                  [1]   SEQUENCE {
            type            AttributeType ({SupportedAttributeTypes}),
            strings         SEQUENCE OF CHOICE {
                                initial     [0]   AttributeValue({SupportedAttributes}{@type}),
                                any         [1]   AttributeValue ({SupportedAttributes}{@type}),
                                final       [2]   AttributeValue ({SupportedAttributes}{@type})}},
        greaterOrEqual      [2]   AttributeValueAssertion,
        lessOrEqual         [3]   AttributeValueAssertion,
        present             [4]   AttributeType,
        approximateMatch    [5]   AttributeValueAssertion,
```

```
        extensibleMatch         [6]   MatchingRuleAssertion}
MatchingRuleAssertion :: = SEQUENCE {
        matchingRule            [1]   SET SIZE (1..MAX) OF MATCHING-RULE.&id,
        type                    [2]   AttributeType OPTIONAL,
        matchValue              [3]   MATCHING-RULE.&AssertionType (CONSTRAINED BY {
            --  matchValue は、matchingRule によって特定された MATCHING-RULE 情報オブジェ
            --  クトの中の一つの&AssertionType フィールドによって規定される型の値である必要
            --  がある。--}),
        dnAttributes            [4]   BOOLEAN DEFAULT FALSE,
        reverseMatch            [5]   BOOLEAN DEFAULT FALSE}
directoryBind OPERATION :: = {
        ARGUMENT        DirectoryBindArgument
        RESULT          DirectoryBindResult
        ERROR           DirectoryBindError}
directoryBindError ERROR :: = {
        PARAMETER       SET {
        versions                [0]     Versions DEFAULT {v1},
        error                   CHOICE {
                                serviceError    [1]     ServiceProblem,
                                securityError           [2]     SecurityProblem}}}
search OPERATION :: = {
        ARGUMENT                SearchArgument
        RESULTSearchResult
        ERRORS{attributeError｜nameError｜serviceError｜referral｜abandoned｜securityError}
        CODE                    id-opcode-search}
SearchArgument :: = OPTIONALLY-SIGNED {SET {
        baseObject              [0]     Name,
        subset                  [1]     INTEGER {
                                          baseObject(0),
                                          oneLevel(1), wholeSubtree(2)}DEFAULT baseObject,
        filter                  [2]     Filter DEFAULT and : { },
        searchAliases           [3]     BOOLEAN DEFAULT TRUE,
        selection               [4]     EntryInformationSelection DEFAULT { },
        pagedResults            [5]     PagedResultsRequest OPTIONAL,
        matchedValuesOnly       [6]     BOOLEAN DEFAULT FALSE,
        extendedFilter          [7]     Filter OPTIONAL,
        COMPONENTS OF CommonArguments}}
SearchResult ::= OPTIONALLY-SIGNED {CHOICE {
        SearchInfo      SET {
            name                    Name OPTIONAL,
            entries                 [0]   SET OF EntryInforamtion,
            partialOutcomeQualifier         [2]   PartialOutcomeQualifier OPTIONAL,
            COMPONENTS OF CommonResults},
```

```
        uncorrelatedSearchInfo      [0]    SET OF SearchResult}}
addEntry OPERATION ::= {
        ARGUMENT                    AddEntryArgument
        RESULTAddEntryResult
        ERRORS{attributeError | nameError | serviceError | referral | securityError | updateError}
        CODE    id-opcode-addEntry}
AddEntryArgument :: = OPTIONALLY-SIGNED {SET {
        object                  [0]    Name,
        entry                   [1]    SET OF Attribute,
        targetSystem            [2]    AccessPoint OPTIONAL,
        COMPONENTS OF CommonArguments}}
AddEntryResult ::= NULL
removeEntry OPERATION :: = {
        ARGUMENT                    RemoveEntryArgument
        RESULTRemoveEntryResult
        ERRORS{nameError | serviceError | referral | securityError | updateError}
        CODE                        id-opcode-removeEntry}
RemoveEntryArgument :: = OPTIONALLY-SIGNED {SET {
        object    [0]           Name,
        COMPONENTS OF CommonArguments}}
RemoveEntryResult :: = NULL
modifyEntry OPERATION :: = {
        ARGUMENT                    ModifyEntryArgument
        RESULTModifyEntryResult
        ERRORS{attributeError | nameError | serviceError | referral | securityError | updateError}
        CODE          id-opcode-modifyEntry}
ModifyEntryArgument ::= OPTIONALLY-SIGNED {SET {
        Object[0]    Name,


--EXPORTS ALL--
        changes                     [1]    SEQUENCE OF EntryModification,
        selection                   [2]    EntryInformationSelection OPTIONAL,
        COMPONENTS OF CommonArguments}}
ModifyEntryResult ::= CHOICE {
        null                    NULL,
        information             OPTIONALLY-SIGNED {SET {
          entry                             [0]                 EntryInformation OPTIONAL,
          COMPONENTS OF CommonResults}}}
EntryModification :: = CHOICE {
        addAttribute                    [0]    Attribute,
        removeAttribute                 [1]    AttributeType,
        addValues                       [2]      Attribute,
        removeValues                    [3]      Attribute,
```

```
        alterValues                     [4]     AttributeTypeAndValue,
        resetValue                      [5]     AttributeType}
securityError ERROR :: = {
        PARAMETER           SET {
                            problem             [0]   SecurityProblem}
        CODE   id-errcode-securityError}
SecurityProblem :: = INTEGER {
        inappropriateAuthentication (1),
        invalidCredentials (2),
        insufficientAccessRights (3),
        invalidSignature (4),
        protectionRequired (5),
        noInformation (6),
        blockedCredentials (7)}

END
```

2.1.2.5.2 ＩＮ Ｘ．５００プロファイルのＡＳＮ．１モジュール

本項では、ディレクトリ抽象サービスの全ＡＳＮ．１プロファイルを含む。

```
IN-DirectoryAbstractService{ccitt recommendation q 1218 modules(0) abstractService(15)version1(0)}

DEFINITIONS :: =

BEGIN

IMPORTS
        attributeError, ServiceControls, EntryInformation, NameErrorParameter, DirectoryBindArgument,
        Credentials, SearchArgument, SearchResult, SearchInfo, PartialOutcomeQualifier, AddEntryArgument,
        RemoveEntryArgument, ModifyEntryArgument, ServiceProblem, UpdateProblem, directoryBindError,
        AddEntryResult, BindErrorParameter, ModifyEntryResult, RemoveEntryResult, SecurityErrorParameter,
        ServiceErrorParameter, AttributeErrorParameter, UpdateErrorParameter
            FROM DirectoryAbstractService
                {joint-iso-ccitt ds(5) module(1) directoryAbstractService(2) 2}
        EntryInformationSelection, SecurityProblem
            FROM ExtendedDirectoryAbstractService
                {ccitt recommendation q 1218 modules(0) abstractService(14) version1(0)}
        Code, OPERATION, ERROR
            FROM Remote-Operations-Information-Objects
                {joint-iso-ccitt remote-operations(4) informationObjects(5) version1(0)};
```

```
IN-ServiceControls :: = ServiceControls
        (WITH COMPONENTS {
            options                 ,
            priority                ,
            timeLimit               ABSENT,
            sizeLimit               ABSENT,
            scopeOfReferral         ,
            attributeSizeLimit      ABSENT})
IN-EntryInformationSelection :: = EntryInformationSelection
        (WITH COMPONENTS {
            attributes              ,
            infoTypes               (attributeTypesAndValues),
            extraAttributes         ,
            contextAssertion        ,
            returnContexts})
IN-EntryInformation :: = EntryInformation
        (WITH COMPONENTS {
            name                    PRESENT,
            fromEntry               (TRUE),
            information             (WITH COMPONENTS {
                                    attributeType           ABSENT,
                                    attribute               PRESENT}),
            incompleteEntry})
in-DirectoryBind OPERATION :: = {
        ARGUMENT            IN-DirectoryBindArgument
        RESULT              IN-DirectoryBindResult
        ERRORS              {in-DirectoryBindError}}
IN-DirectoryBindArgument ::= DirectoryBindArgument
        (WITH COMPONENTS {
            credentials             (IN-Credentials) OPTIONAL,
            versions            ({v1})})
IN-Credentials :: = Credentials
        (WITH COMPONENTS {
            simple                  ,
            strong                  ,
            externalProcedure       })
IN-DirectoryBindResult ::= IN-DirectoryBindArgument
in-DirectoryBindError ERROR ::= {
        PARAMETER           IN-DirectoryBindErrorParameter}
IN-DirectoryBindErrorParameter ::= DirectoryBindErrorParameter
        (WITH COMPONENTS {
            versions            ABSENT,
```

```
        error                   (WITH COMPONENTS {
                                        securityError       (SecurityProblem (1 | 2 | 7)),
                                        serviceError    (ServiceProblem (2))})})
in-Search OPERATION ::= {
    ARGUMENT        IN-SearchArgument
    RESULTIN-SearchResult
    ERRORS{nameError | in-ServiceError | securityError | attributeError | referral}
    CODE            id-opcode-in-search}
IN-SearchArgument ::= SearchArgument
    (WITH COMPONENTS {
        baseObject              PRESENT,
        subset                  ,
        filter                  ,
        searchAliases           (TRUE),
        selection               (IN-EntryInformationSelection),
        pagedResults            ABSENT,
        matchedValuesOnly       ,
        extendedFilter          ABSENT,
        serviceControls         (IN-ServiceControls),
        securityParameters      ,
        requestor               ,
        operationProgress       ,
        aliasedRDNs             ABSENT,
        criticalExtensions      ,
        referenceType           ,
        entryOnly               ,
        exclusions              ,
        nameResolveOnMaster     })
  IN-SearchResult ::= SearchResult
    (WITH COMPONENTS {
        searchInfo          (WITH COMPONENTS {
            name    ,
            entries                     (WITH COMPONENT (IN-EntryInformation)),
            partialOutcomeQualifier PartialOutcomeQualifier
                    (WITH COMPONENTS {
                        limitProblem                ,
                        unexplored                  ,
                        unavailableCriticalExtensions   ,
                        unknownErros                ,
                        queryReference              ABSENT}),
            securityParameters      ,
            performer               ,
            aliasDereferenced       }),
```

```
        uncorrelatedSearchInfo              })
in-AddEntry OPERATION ::= {
        ARGUMENT        IN-AddEntryArgument
        RESULTAddEntryResult
        ERRORS{nameError | in-ServiceError | securityError | attributeError | updateError | referral}
        CODE    id-opcode-in-addEntry}
IN-AddEntryArgument ::= AddEntryArgument
        (WITH COMPONENTS {
                object                      PRESENT,
                entry                       PRESENT,
                targetSystem                ,
                serviceControls             (IN-ServiceControls),
                securityParameters          ,
                requestor                   ,
                operationProgress           ,
                aliasedRDNs                 ABSENT,
                criticalExtensions          ,
                referenceType               ,
                entryOnly                   ,
                exclusions                  ,
                nameResolveOnMaster         })
in-RemoveEntry OPERATION ::= {
                ARGUMENT                    IN-RemoveEntryArgument
                RESULT                      RemoveEntryResult
                ERRORS                      {nameError | in-ServiceError | securityError | updateError | referral}
                CODE                        id-opcode-in-removeEntry}
IN-RemoveEntryArgument ::= RemoveEntryArgument
        (WITH COMPONENTS {
                object                      PRESENT,
                serviceControls             (IN-ServiceControls),
                securityParameters          ,
                requestor                   ,
                operationProgress           ,
                aliasedRDNs                 ABSENT,
                criticalExtensions          ,
                referenceType               ,
                entryOnly                   ,
                exclusions                  ,
                nameResolveOnMaster         })
in-ModifyEntry OPERATION ::= {
                ARGUMENT            IN-ModifyEntryArgument
                RESULT                      ModifyEntryResult
                ERRORS    {nameError | in-ServiceError | securityError | attributeError | updateError | referral}
```

```
        CODE                    id-opcode-in-modifyEntry}
IN-ModifyEntryArgument ::= ModifyEntryArgument
    (WITH COMPONENTS {
        object                  PRESENT,
        changes                 PRESENT,
        selection               (IN-EntryInformationSelection),
        serviceControls         (IN-ServiceControls),
        securityParameters      ,
        requestor               ,
        operationProgress       ,
        aliasedRDNs             ABSENT,
        criticalExtensions      ,
        referenceType           ,
        entryOnly               ,
        exclusions              ,
        nameResolveOnMaster     })
IN-ModifyEntryResult ::= ModifyEntryResult
    (WITH COMPONENTS {
        null            ,
        information         Information
            (WITH COMPONENTS {
                entry               (IN-EntryInformation),
                securityParameters  ,
                performer           ,
                aliasDereferenced   })})
in-ServiceError ERROR ::= {
        PARAMETER           IN-ServiceErrorParameter
        CODE                id-errcode-in-serviceError
IN-ServiceErrorParameter ::= ServiceErrorParameter
        (WITH COMPONENTS {
            problem     (ServiceProblem (1 | 2 | 3 | 4 | 5 | 6 | 8 | 9 | 10 | 11 | 12))})
id-errcode-in-serviceError      Code ::= local:3
id-opcode-in-modifyEntry        Code ::= local:8
id-opcode-in-addEntry           Code ::= local:6
id-opcode-in-removeEntry        Code ::= local:7
id-opcode-in-search             Code ::= local:5

END
```

2.1.2.5.3　ＤＡＰのＡＳＮ．１モジュール

本項では、DirectoryAccessProtocol というＡＳＮ．１モジュールの形で、このディレクトリの仕様に含まれるすべてのＡＳＮ．１の型および値を記述している。さらにまた、ProtocolObjectIdentifiers というＡＳＮ．１モジュールの形で、このディレクトリの仕様に割り当てられるＡＳＮ．１のオブジェクト識別子も記述している。

```
INDirectoryAccessProtocol {ccitt recommendation q 1218 modules(0) indap(12) version1(0)}

DEFINITIONS::=

BEGIN

-- EXPORTS ALL --
-- 本モジュールで定義した型および値は、ディレクトリの仕様に含まれる他のＡＳＮ．１モジュールある
-- いは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で使用できるようにエキスポートす
-- る。他のアプリケーションではその目的のためだけに使用するかもしれないが、このことは、ディ
-- レクトリサービスを維持・向上するために必要な拡張あるいは修正を制限することにはならない。

IMPORTS
        directoryAbstractService
            FROM UsefulDefinitions {joint-iso-ccitt ds(5) module(1) usefulDefinitions(0) 2}
        ROS-OBJECT-CLASS, CONTRACT, OPERATION-PACKAGE, CONNECTION-PACKAGE,
        Code, OPERATION
            FROM Remote-Operations-Information-Objects
                {joint-iso-ccitt remote-operations(4) InformationObjects(5) version1(0)}
        Bind{}, Unbind{}, InvokeID
            FROM Remote-Operations-Generic-ROS-PDUs
                {joint-iso-ccitt remote-operations(4) generic-ROS-PDUs(6) version1(0)}
        TCAPMessage{}
            FROM TCAPMessages
                {ccitt recommendation q 773 modules(2) messages(1) version3(3)}
        APPLICATION-CONTEXT
            FROM TC-Notation-Extension
                {ccitt recommendation q 775 modules(2) notation-extensions(4) version1(1)}
        dialogue-as-id
            FROM DialoguePDUs
                {ccitt recommendation q 773 modules(2) dialoguePDUs(2) version1(1)}
        id-ac-directoryAccessAC, id-rosObject-dua, id-rosObject-directory, id-rosObject-dapDSA,
        id-contract-dap, id-package-dapConnection, id-package-search, id-package-modify,
        id-as-directoryOperationsAS, id-as-directoryBindingAS
            FROM SDFProtocolObjectIdentifiers
                {ccitt recommendation q 1218 modules(0) sdfProtocolObjectIdentifiers(10) version1(0)}
        directoryUnbind FROM DirectoryAbstractService directoryAbstractService ;
        directoryBind, search, addEntry, removeEntry, modifyEntry
```

```
            FROM ExtendedDirectoryAbstractService
                {ccitt recommendation q 1218 modules(0) informationFramework(11) version1(0)}
;

-- application contexts --
inDirectoryAccessAC          APPLICATION-CONTEXT        ::=
{
        CONTRACT                    dapContract
        DIALOGUE MODE               structured
        TERMINATION             basic
        ABSTRACT SYNTAXES   {dialogue-abstract-syntax | inDirectoryOperationsAbstractSyntax |
                                                        inDirectoryBindingAbstractSyntax}
        APPLICATION CONTEXT NAME id-ac-directoryAccessAC
}

-- ROS-objects --
dua             ROS-OBJECT-CLASS            ::=
{
        INITIATES           {dapContract}
        ID                      id-rosObject-dua
}
directory       ROS-OBJECT-CLASS            ::=
{
        RESPONDS            {dapContract}
        ID                      id-rosObject-directory
}
dap-dsa         ROS-OBJECT-CLASS            ::=
{
        RESPONDS            {dapContract}
        ID                      id-rosObject-dapDSA
}

-- contracts --
dapContract     CONTRACT    ::=
{
        CONNECTION                  dapConnectionPackage
        INITIATOR CONSUMER OF   {searchPackage・modifyPackage}
        ID                              id-contract-dap
}
```

```
-- connection package --
dapConnectionPackage        CONNECTION-PACKAGE        ::=
{
        BIND                        directoryBind
        UNBIND              directoryUnbind
        ID                          id-package-dapConnection
}

-- search package --
searchPackage               OPERATION-PACKAGE         ::=
{
        CONSUMER INVOKES    {search}
        ID                          id-package-search
}

-- modify package --
modifyPackage               OPERATION-PACKAGE         ::=
{
        CONSUMER INVOKES    {addEntry | removeEntry | modifyEntry}
        ID                          id-package-modify
}

-- abstract-syntaxes --
inDirectoryOperationsAbstractSyntax    ABSTRACT-SYNTAX        ::=    {
    BasicDAP-PDUs
    IDENTIFIED BY       id-as-directoryOperationsAS}
BasicDAP-PDUs       ::=     TCAPMessage {{DAP-Invokable}, {DAP-Returnable}}
DAP-Invokable       OPERATION       ::=     {search | addEntry | removeEntry | modifyEntry}
DAP-Returnable      OPERATION       ::=     {search | addEntry | removeEntry | modifyEntry}
inDirectoryBindingAbstractSyntax            ABSTRACT-SYNTAX        ::=    {
    DAPBinding-PDUs
    IDENTIFIED BY               id-as-directoryBindingAS}
DAPBinding-PDUs     ::=     CHOICE
{
    bind                        Bind    {directoryBind},
    unbind                      Unbind      {directoryUnbind}
}


END
```

```
SDFProtocolObjectIdentifiers
        {ccitt recommendation q1218 module(0) sdfProtocolObjectIdentifiers(10) version1(0)}

DEFINITIONS::=
BEGIN

-- EXPORTS ALL --

IMPORTS

-- useful definitions --
in-ds                   OBJECT IDENTIFIER       ::=
            {ccitt recommendation q 1218 sdf-objects(10)}
id-rosObject            OBJECT IDENTIFIER       ::=     {in-ds 25}
id-contract                     OBJECT IDENTIFIER       ::=     {in-ds 26}
id-package                      OBJECT IDENTIFIER       ::=     {in-ds 27}
id-ac                   OBJECT IDENTIFIER       ::=     {in-ds 3}
id-as                   OBJECT IDENTIFIER       ::=     {in-ds 5}

-- ROS Objects --
id-rosObject-dua        OBJECT IDENTIFIER       ::=     {id-rosObject 1}
id-rosObject-directory          OBJECT IDENTIFIER       ::=     {id-rosObject 2}
id-rosObject-dapDSA             OBJECT IDENTIFIER       ::=     {id-rosObject 3}

-- contracts --
id-contract-dap         OBJECT IDENTIFIER       ::=     {id-contract 1}

-- packages --
id-package-search       OBJECT IDENTIFIER       ::=         {id-package 2}
id-package-modify       OBJECT IDENTIFIER       ::=         {id-package 3}
id-package-dapConnection        OBJECT IDENTIFIER       ::=         {id-package 10}

-- application contexts --
id-ac-directoryAccessAC     OBJECT IDENTIFIER       ::=     {id-ac 1}

-- abstract syntaxes --
id-as-directoryOperationsAS             OBJECT IDENTIFIER       ::=     {id-as 1}
id-as-directoryBindingAS    OBJECT IDENTIFIER       ::=     {id-as 2}

END
```

2.2　ＳＤＦ－ＳＤＦインタフェース

2.2.1　ＩＮ X.500 DOP, DSP, DISP サブセットへの導入

SDF-SDF インタフェースの目的は、1 つのＳＤＦから別のＳＤＦにサービスプロファイルのコピーの転送を可能とし、データベース間のコピーを管理することである。X.500 の機能はＣＳ－１（改）の要求条件を満たすために必要な機能よりも広い。本項では、DSP および DISP において、何を考慮しサポートすべきであるか、何を無視すべきであるかということを示す。様々なパラメータの状態を表す手段としてプロファイリングを用いる。

一つのメッセージで送信するパラメータ数をできるだけ少なくすることは、信号トラヒック量および処理時間を削減するために重要である。したがって、信号送信時に必須でないパラメータは削除する。信号を受信した際、削除されたパラメータは受信側のエンティティによって扱われるべきではないが、解釈は行われるべきである。これによって、1993 年版のディレクトリの記述に従って将来プロファイルの拡張が許容される。

便利かつ明確のため、このプロファイルは ASN.1 サブタイピングファシリティーズを用いて定義するが、これらの定義はプロトコルの仕様を記述しているのではない。これは単純にインプリメントにおいて送信すべきでないパラメータを示している。ただし、DSP および DISP の本来の定義に適合している値をデコードできる受信側のエンティティのふるまいを変えるものではない。それでもなお、サブタイピングによって除外された要素は解釈されるべきであるが、扱われるべきではない。

2.2.2　前提条件

ＩＮ　ＣＳ－１（改）のための DOP, DSP, DISP を設計するために、いくつかの前提条件を用いる。

・前提条件 1：データの転送に関係する網オペレータ間の合意は、オフライン（例、管理オペレーション）で定義される。establishOperationalBinding オペレーションは合意を活性化するためだけに用いる。
・前提条件 2：合意はオンラインのオペレーションでは変更できない。
・前提条件 3：terminateOperationalBinding オペレーションは、網オペレータ間の合意を終了するために用いる。これは、シャドウ消費側にあるコピーをもはや維持管理しないことを意味する。それは使用すべきではなく、削除すべきである。しかしながら、二つの網間で今後のアソシエーション確立のために合意が要求されることもあるため、情報は保持すべきである。
・前提条件 4：シャドウの更新は、マスタコピーを管理しているシャドウ供給側から起動される。したがって、コピーの更新はシャドウされたコピーではなく、常にマスタコピーに対してなされる。更新の要求は、連鎖オペレーションを用いてマスタコピーに送信される。コピーは変化が生じる度に更新される。
・前提条件 5：DSA 間では直接的な参照のみが用いられる。したがって、オペレーションの連鎖は一度に限られる。一度連鎖してもオペレーションが実行できない場合は、リフェラルが返送されるべきである。
・前提条件 6：コピーのコピーを作成することは可能ではない。コピーを得るためにはマスタコピーを参照するべきである。

・前提条件 7 : シャドウするメカニズムは、特定の DAP オペレーションあるいは管理オペレーションによって起動される。管理オペレーションについては今後の検討課題である。

・前提条件 8 : シャドウの合意が終了する時間はサービスの種別に依存する。ただし、ほとんどの場合はコピーの数に基づく。DIT のある一部に対してコピーの数が最大値に達すると、最も古いコピーを削除し、その合意を非活性化する必要がある。コピーの最大値は 1 となることもある。

・前提条件 9 : SDF-SDF 間のオペレーションを廃棄することはできない。オペレーションの実行に時間がかかった場合は、タイマが満了するため、廃棄する必要はない。

2.2.3 シャドウの合意の規定

シャドウの合意は以下のように規定される。

```
IN-ShadowingAgreementInfo ::= ShadowingAgreementInfo
        (WITH COMPONENTS {
                shadowSubject           ,
                updateMode              ,
                master                  ABSENT,
                secondaryShadows        ABSENT})
```

shadowSubject は、シャドウすべきサブツリー，エントリおよび属性を特定し、UnitOfReplication により表される。UnitOfReplication の要素は ITU-T 勧告 X.525 の 9.2 項で定義されている。

updateMode は、シャドウ領域の更新をいつ実施するようにスケジュールするかを特定する。UpdateMode の要素は ITU-T 勧告 X.525 の 9.3 項で定義されている。

master は、マスタ領域を含む DSA のアクセスポイントを含む。DSA は既にこの情報を認識しているため、ＩＮでは不要である。

secondaryShadows は、二次的なシャドウ情報がシャドウ供給側に後に供給されることを許容する。二次的なシャドウはＩＮの観点では無視する（前提条件 5）ため、本要素は含まれるべきではない。

2.2.4 ＩＮ X.500 DISP サブセット

2.2.4.1 DSA シャドウ結合

DSA シャドウ結合（dSAShadowBind）オペレーションは、シャドウを供給する期間の最初に用いる。

```
dSAShadowBind ::= in-DirectoryBind
```

ＩＮ ＣＳ－１（改）では、2.1.2.1.2 項に規定されているように dSAShadowBind オペレーションを用いる。

2.2.4.2 DSA シャドウ結合解放

DSA シャドウ結合解放（dSAShadowUnbind）オペレーションは、シャドウを提供する期間の最後に用いる。

```
dSAShadowUnbind    OPERATION ::= directoryUnbind
```

ＩＮ ＣＳ－１（改）では、ITU-T 勧告 X.525 の 7.4 項に規定されているように dSAShadowUnbind オペレーションを用いる。

2.2.4.3 シャドウ更新調整

シャドウ更新調整（inCoordinateShadowUpdate）オペレーションは、シャドウ供給側が意図している更新の送信に関連するシャドウの合意を指定するために用いる。

```
inCoordinateShadowUpdate    OPERATION ::= {
        ARGUMENT            IN-CoordinateShadowUpdateArgument
        RESULT              IN-CoordinateShadowUpdateResult
        ERRORS              {shadowError}
        CODE                id-opcode-coordinateShadowUpdate}

IN-CoordinateShadowUpdateArgument ::= CoordinateShadowUpdateArgument (
        WITH COMPONENTS {
                agreementID             ,
                lastUpdate              ,
                updateStrategy          (standard:{total|incremental}),
                securityParameters})

IN-CoordinateShadowUpdateResult ::= NULL
```

各パラメータの意味を以下に規定する。

a) agreementID は、シャドウの合意を特定する。

b) lastUpdate は、当該の合意に対して最後の更新が送信された時刻に対するシャドウ供給側の認識を示し、シャドウ供給側のＳＤＦによって提供された時刻である。このパラメータは、特定のシャドウの合意に対する最初のシャドウ更新調整あるいはシャドウ更新要求オペレーションの場合にのみ省略してもよい。

c) updateStrategy は、シャドウ供給側がこの更新の際に用いようとしている更新の方法を特定する。ＩＮ ＣＳ－１（改）においては、全体更新あるいは差分更新を使用すべきである。“変更無し”の選択肢を用いることはない。

d) securityParameters は、ITU-T 勧告 X.511|ISO/IEC 9594-3 の 7.10 項で規定されている。

2.2.4.4　シャドウ更新

シャドウ更新（inUpdateShadow）オペレーションは、シャドウ供給側がシャドウ消費側に対して複製領域の更新を送信するために用いる。このオペレーションが起動される前に、シャドウ更新調整あるいはシャドウ更新要求オペレーションが、指定されたシャドウの合意に対して正常に完了していなければならない。

```
inUpdateShadow    OPERATION ::= {
        ARGUMENT                    IN-UpdateShadowArgument
        RESULT                      IN-UpdateShadowResult
        ERRORS                      {shadowError}
        CODE                        id-opcode-updateShadow}

IN-UpdateShadowArgument ::= UpdateShadowArgument (
        WITH COMPONENTS {
                agreementID         ,
                updateTime          ,
                updateWindow        ,
                updatedInfo         (IN-RefreshInformation),
                securityParameters})

IN-UpdateShadowResult ::= NULL
```

各パラメータの意味を以下に規定する。

a) agreementID は、確立されているシャドウの合意を特定する。

b) updateTime は、シャドウ供給側が供給する。この時刻は、次のシャドウ更新調整あるいはシャドウ更新要求オペレーションの際に、シャドウ供給側とシャドウ消費側がシャドウされた情報に関する共通認識を保証するために用いる。

c) updateWindow は、それが存在する場合には、シャドウ供給側が更新を送信することを予定している次のウィンドウを特定する。

d) updatedInfo は、シャドウ消費側がシャドウされた情報を更新するために必要な情報を提供する。このパラメータで送信される情報の意味は、シャドウ消費側が供給された変更を反映することに行き着く。

e) securityParameters は、ITU-T 勧告 X.511|ISO/IEC 9594-3 の 7.10 項で規定されている。

```
IN-RefreshInformation ::= RefreshInformation (
        WITH COMPONENTS {
                noRefresh           ,
                total               ,
                incrementsl         ,
                otherStrategy       ABSENT})
```

各パラメータの意味を以下に規定する。

a) noRefresh は、以前の要求から現時点までの間にシャドウされた情報に対して変更がなかったことを示している。これはシャドウ更新オペレーションが、シャドウの合意（更新モード）に規定されている特定の時間毎に送信されなければならないという場合で、かつ変更が実際には起きていない場合に用いられる。
b) total は、シャドウされた情報の新しいインスタンスを提供する。しかしながら、信号量を削減するためには、差分更新の方法を用いることが好ましい。
c) incremental は、シャドウされた情報を完全に置き換えるのではなく、最も最近に要求したシャドウ更新調整（あるいはシャドウ更新要求）の中の lastUpdate と現在のシャドウ更新の要求（あるいはシャドウ更新要求の応答）の中の updateTime の間にシャドウされた情報に生じた変更だけを提供する。
d) otherStrategy は、ディレクトリの仕様の範囲外のメカニズムを用いて更新を送信する能力を提供する。ただしＩＮ ＣＳ－１（改）では、全体変更あるいは差分変更の方法を使用すべきである。

2.2.4.5 シャドウ更新要求

シャドウ更新要求（inRequestShadowUpdate）オペレーションは、シャドウ消費側がシャドウ供給側からの更新を要求するために用いる。

```
inRequestShadowUpdate    OPERATION ::= {
        ARGUMENT                IN-RequestShadowUpdateArgument
        RESULT                  IN-RequestShadowUpdateResult
        ERRORS                  {shadowError}
        CODE                    id-opcode-requestShadowUpdate}

IN-RequestShadowUpdateArgument ::= RequestShadowUpdateArgument (
        WITH COMPONENTS {
                agreementID             ,
                lastUpdate              ,
                requestedStrategy       (standard:{incremental|total}),
                securityParameters})

IN-RequestShadowUpdateResult ::= NULL
```

各パラメータの意味を以下に規定する。

a) agreementID は、シャドウの合意を特定する。
b) lastUpdate は、最も最近に成功した更新においてシャドウ供給側が提供する時刻である。このパラメータは、特定のシャドウの合意に対する最初のシャドウ更新調整あるいはシャドウ更新要求オペレーションの場合にのみ省略してもよい。
c) requestedStrategy は、シャドウ消費側が要求する更新の種別を特定する。シャドウ消費側は、シャドウ供給側からの差分更新あるいは全体更新のいずれかを要求してよい。
d) securityParameters は、ITU-T 勧告 X.511|ISO/IEC 9594-3 の 7.10 項で規定されている。

2.2.5 ＩＮ X.500 DS サブセット

2.2.5.1 情報タイプおよび共通プロシージャ

2.2.5.1.1 連鎖引き数（Chaining Arguments）

連鎖引き数は、全体の処理の中の一部をきちんと実行するのに必要な情報を DSA に転送するために、各連鎖オペレーションの中に存在する。

```
IN-ChainingArguments ::= ChainingArguments (
    WITH COMPONENTS {
        originator              ,
        targetObject            ,
        operationProgress       ,
        traceInformation        ABSENT,
        aliasDereferenced       ABSENT,
        aliasedRDNs             ABSENT,
        returnCrossRefs         ABSENT,
        referenceType           ,
        info                    ABSENT,
        timeLimit               ABSENT,
        securityParameters      ,
        entryOnly               ABSENT,
        uniqueIdentifier        ,
        authenticationLevel     ,
        exclusions              ABSENT,
        excludeShadows          ABSENT,
        nameResolveOnMaster     ABSENT})
```

各要素の意味を以下に規定する。

a) originator は、セキュリティパラメータの中で規定されていない場合に、要求元の名前を伝える。CommonArguments の中に requester が存在する場合は、この引数は省略してもよい。

b) targetObject は、ルーティングされるディレクトリエントリを持つオブジェクトの名前を伝える。このオブジェクトの役割は関連する特定のオペレーションに依存し、このオブジェクトのエントリが操作される場合もあり、また要求あるいはサブ要求に対して、複数のオブジェクトを含むベースオブジェクトとなることもある（例、ChainedModify）。この要素は、連鎖オペレーション内のオブジェクトまたはベースオブジェクトパラメータと同一の値をもつときに限り省略することができる。

c) operationProgress は、オペレーションの実行状況を DSA に通知するために使用される。したがって、全体のパフォーマンスの中で果たすべき役割を持つ。ＩＮ ＣＳ－１（改）では、直接的な知識参照が前提となっているが、このパラメータは適用可能と思われる。というのは、オペレーションが連鎖されているＳＤＦでは、連鎖オペレーションの dsaReferral エラーを継続参照を用いて応答することが可能であるからである。

d) traceInformation は、連鎖の際に DSA 間でループすることを避けるために使用される。DSA は他の DSA にオペレーションを連鎖する前に、新しい要素を追跡情報に追加する。オペレーションの実行を要求された DSA は、追跡情報を調べることによって、オペレーションがループになっていないことをチェックする。ＩＮ ＣＳ－１（改）では、直接的な知識参照が前提であるため、このパラメータは非適用と考えられる。

e) aliasDereferenced は、分散された名前解決の際に一つ以上の別名エントリに遭遇および展開したかどうかを示すために用いる論理値である。ＩＮの別名エントリは、オブジェクトに対して別の名称を提供するだけの手段であるため、必要な場合は展開されるべきであり、したがってこの指標は必要ない。

f) aliasedRDNs は、targetObject 名の中の何個の RDN が、一つ（あるいはそれ以上）の別名エントリの aliasedEntryName 属性から生成されるかを示す。別名エントリに遭遇し展開されると、整数値が設定される。ＩＮの別名エントリは、オブジェクトに対して別の名称を提供するだけの手段であるため、必要な場合は展開されるべきであり、したがってこの指標は必要ない。

g) returnCrossRefs は、分散オペレーションの実行中に使用された知識参照が、結果あるいはリフェラルとともにクロス参照として起動元 DSA に返送される必要があるかどうかを示すために使用される論理値である。ＩＮ ＣＳ－１（改）では、直接的な知識参照が前提であるため、このパラメータは非適用と考えられる。

h) referenceType は、オペレーションの実行を要求された DSA に対して、その要求をそこへ中継するために用いられた知識の種別を示す。これによって DSA は、要求側が保持する知識の中にあるエラーを検出することができる。そのようなエラーが検出された場合は、invalidReference 問題を付与した ServiceError によって示される。ReferenceType の詳細は 2.2.5.1.3 項に記述する。

I) infor は、一つの要求を処理する複数の DSA 間で DMD(Directory Management Domain)特有の情報を伝えるために使用される。ＣＳ－１（改）では管理プロトコルは対象外であるため、このパラメータは非適用と考えられる。

j) timeLimit は、存在する場合は、オペレーションが完了すべき期限を示す。これは TCAP のオペレーションタイマと冗長であるため、不要である。

k) securityParameters は、ITU-T 勧告 X.511|ISO/IEC 9594-3 で規定されている。

l) entryOnly は、元のオペレーションが探索（Search）であり、subset を oneLevel に設定し、別名エントリが baseObject の直接の従属として遭遇した場合に、真に設定される。targetObject 名の名前解決を実行し成功した DSA は、その名前の付いたエントリに対してのみオブジェクトの評価を実行する。ＣＳ－１（改）で連鎖されるオペレーションはエントリ更新（ModifyEntry）のみであるため、このパラメータは不要である。

m) uniqueIdentifier は、起動元の名前（起動元とは要求を中継したＳＤＦ）を確認することが要求される際にオプションで提供される。uniqueIdentifier は ITU-T 勧告 X.501|ISO/IEC 9594-2 で規定されている。

n) authenticationLevel は、ＳＤＦ間で実行された認証の方法を示すことが要求される際にオプションで提供される。authenticationLevel は ITU-T 勧告 X.501|ISO/IEC 9594-2 で規定されている。

o) exclusions は、探索（Search）オペレーションに対してのみ意味があり、存在する場合には、targetObject に従属するエントリのどのサブツリーを探索（Search）の結果から除外するかということを示す。ＣＳ－１（改）で連鎖されるオペレーションはエントリ更新（ModifyEntry）のみであるため、このパラメータは不要である。

p) excludeShadows は、探索（Search）および一覧（List）オペレーションに対してのみ意味があり、エントリのコピーではなくエントリに対して探索を適用すべきであることを示す。このオプションの要素は、重複した結果を受信することを避けるための一つの方法として、DSA が使用することができる。ＣＳ－１（改）では、直接的な知識参照が前提であるため、このパラメータは非適用と考えられる。

q) nameResolveOnMaster は、名前解決においてのみ意味があり、NSSRs(non-specific knowledge references)に遭遇したときにのみ設定される。真に設定された場合は、後続の名前解決、すなわち nextRDNToBeResolved からの残りの RDN のマッチングは、エントリのコピーの情報は用いず、その RDN によって指定されるエントリに対するマスタ DSA においてなされるべきであることを示す。ＣＳ－１（改）では、直接的な知識参照が前提であるため、このパラメータは非適用と考えられる。

2.2.5.1.2 連鎖結果（Chaining Results）

連鎖結果は、各オペレーションの結果の中に存在し、オペレーションを起動した DSA に対するフィードバックを提供する。

```
IN-ChainingResults ::= ChainingResults (
    WITH COMPONENTS {
        info                    ABSENT,
        crossReferences         ABSENT,
        securityParameters      ,
        alreadySearched         ABSENT})
```

各要素の意味を以下に規定する。

a) infor は、一つの要求を処理する複数の DSA 間で DMD 特有の情報を伝えるために使用される。ＣＳ－１（改）では管理プロトコルは対象外であるため、このパラメータは非適用と考えられる。

b) crossReferences は、対応する要求の returnCrossRefs 要素が真に設定されていなければ、連鎖結果内には存在しない。ＣＳ－１（改）では、直接的な知識参照が前提であるため、このパラメータは非適用と考えられる。

c) securityParametes は、ITU-T 勧告 X.511|ISO/IEC 9594-3 で規定されている。これが存在しないことは、セキュリティパラメータのセットが存在しないことと等価と考えられる。

d) alreadySearched は、存在する場合は、targetObject の直下に従属するどの RDN が、連鎖された探索（ChainedSearch）オペレーションの一部として処理され、したがって後続のサブ要求では除外されるべきであるか、ということを示す。ＣＳ－１（改）で連鎖されるオペレーションはエントリ更新（ModifyEntry）のみであるため、このパラメータは不要である。

2.2.5.1.3　参照型（Reference Type）

参照型の値は、ITU-T 勧告 X.501|ISO/IEC 9594-2 で規定されている種々の参照の内の一つを示す。

```
IN-ReferenceType ::= ReferenceType (1|2|4|5|6|7|8)
```

ＩＮ　ＣＳ－１（改）では直接参照が前提であるため、値 3（クロス参照）は非適用である。

2.2.5.1.4　アクセス点（Access Point）情報

アクセス点には 3 つの型がある。

a) アクセス点の値は、ディレクトリ、特に DSA へのアクセスが行われる特定の点を識別する。アクセス点は、関係する DSA の Name およびその DSA への No.7 信号方式で使用される PresentationAddress からなる。

```
IN-AccessPoint ::= AccessPoint (
        WITH COMPONENTS {
                ae-title                ,
                address                 ,
                protocolInformation     ABSENT})
```

address には、No.7 信号方式における DSA のネットワークアドレスが含まれる。

b) MasterOrShadowAccessPoint の値は、ディレクトリへのアクセス点を識別する。アクセス点の category、すなわち master あるいは shadow は、名称コンテキストを示すか共通的に利用可能な複製領域を示すかに依存する。

```
IN-MasterOrShadowAccessPoint ::= MasterOrShadowAccessPoint (
        WITH COMPONENTS {
                COMPONENTS OF IN-AccessPoint,
                category})
```

c) MasterAndShadowAccessPoints の値は、ディレクトリへのアクセス点の集合、すなわち関連する DSA の集合を識別する。これらのアクセス点は、エントリ情報を持つ DSA に対して、それぞれが、共通の名称コンテキストあるいは値が nonSpecific Knowledge 属性の値であるときに一つの DSA を主とする名称コンテキストの共通集合から、参照するという性質を共有する。MasterAndShadowAccessPoints の値は、含まれる各アクセス点の値の category を示す。名称コンテキストのマスタ DSA のアクセス点は、その集合に含まれる必要はない。

```
IN-MasterAndShadowAccessPoints ::= SET OF MasterOrShadowAccessPoint
```

AccessPointInformation の値は、ディレクトリへの一つ以上のアクセス点を特定する。

```
IN-AccessPointInformation ::= AccessPointInformation (
        WITH COMPONENTS {
                COMPONENTS OF IN-MasterOrShadowAccessPoint,
                additionalPoints        SET OF IN-MasterOrShadowAccessPoint    OPTIONAL})
```

2.2.5.1.5 継続参照（Continuation Reference）

継続参照は、オペレーションの全体あるいは一部の実行を、一つ以上の異なった DSA でどのように継続できるかを示す。一般的には、関係する DSA が要求自体を伝達できない、あるいは伝達したくないときに、リフェラルとして返送される。

```
IN-ContinuationReference ::= ContinuationReference (
        WITH COMPONENTS {
                targetObject            ,
                aliasedRDNs             ABSENT,
                operationProgress       ,
                rdnsResolved            ABSENT,
                referenceType           (IN-ReferenceType),
                accessPoints            SET OF (IN-AccessPoint),
                entryOnly               ABSENT,
                exclusions              ,
                returnToDUA             ,
                nameResolveOnMaster     ABSENT})
```

各要素の意味を以下に規定する。

a) targetObject 名は、オペレーションを継続するときに使用することが提案される名前を示す。例えば、別名が展開されるかあるいは探索（Search）における基底オブジェクトが特定された場合には、入力要求で受信した targetObject 名と異なる場合がある。

b) aliasedRDNs は、目的のオブジェクト名の中の（もしあれば）いくつの RDN が、別名を展開することによって生成されたかを示す。ＩＮの別名エントリは、オブジェクトに対して別の名称を提供するだけの手段であるため、必要な場合は展開されるべきであり、したがってこの指標は必要ない。

c) operationProgress は、継続参照を受信した DSA あるいは DUA が、それに従いたい場合に、完了した名称解決の内、指定された DSA においてオペレーションの継続実行を支配する名称解決の量を示す。

d) rdnsResolved の値（名前の中のいくつかの RDN が、完全な名称解決の対象ではないが、クロス参照からは正しいと仮定された場合にのみ、存在する必要がある）は、内部参照のみを利用して、いくつの RDN が実際に解決したかを示す。 ＩＮ ＣＳ－１（改）では、直接的な知識参照が前提であるため、このパラメータは非適用と考えられる。

e) referenceType は、この継続を生成する際に利用された知識の種別を示す。

f) accessPoints は、この継続を実現するために参照すべきアクセス点を示す。不特定の従属参照が関係している場合にのみ、二つ以上の AccessPointInformation 項目が存在する可能性がある。

g) entryOnly は、元のオペレーションが探索（Search）であり、subset を oneLevel に設定し、別名エントリが baseObject の直接の従属として遭遇した場合に、真に設定される。targetObject 名の名前解決を実行し成功した DSA は､その名前の付いたエントリに対してのみオブジェクトの評価を実行する。ＩＮの別名エントリは、オブジェクトに対して別の名称を提供するだけの手段であるため、必要な場合は展開されるべきであり、したがってこの指標は必要ない。

h) exclusions は、受信した DSA では調査すべきではない従属の名称コンテキストの集合を示す。

i) returnToDUA は、継続参照を生成した DSA が、途中の DSA を介して情報を返送したくなく（例えば、セキュリティの理由）、むしろ起動元 DUA と該 DSA 間で DAP オペレーションによって直接情報を利用可能としたいことを示す場合に、オプションで提供される。returnToDUA が真に設定されている場合は、referenceType は self に設定され得る。この情報は、ＩＮにおいても網オペレータ間で確立したシャドウの合意をサポートするために使用可能である（例、在圏ＳＤＦからホームＳＤＦへの更新が、アクセス制御の制約から失敗することもあり得る）。

j) nameResolveOnMaster は、継続参照を生成する DSA が NSSRs に遭遇したときにオプションで提供される。ＩＮ ＣＳ－１（改）では、直接的な知識参照が前提であるため、このパラメータは非適用と考えられる。

2.2.5.2 DSA 結合（DSA Bind）

DSA 結合オペレーションは、ディレクトリサービスを提供する二つの DSA 間で協調する期間を開始するために用いる。

DSABind ::= in-DirectoryBind

ＩＮ ＣＳ－１（改）では、2.1.2.1.2 項に規定されているように dSABind オペレーションを用いる。

2.2.5.3 DSA 結合解放（DSA Unbind）

DSA 結合解放オペレーションは、ディレクトリサービスを提供する二つの DSA 間で協調する期間を終了するために用いる。

DSAUnbind ::= DirectoryUnbind

2.2.5.4 連鎖オペレーション（Chained operations）

DUA からオペレーションを受信した DSA は、そのオペレーションの連鎖の形式を構築し、別の DSA に伝達することが可能である。ＩＮ ＣＳ－１（改）では、連鎖形式のオペレーションを受信した DSA は他の DSA へそれを連鎖することは許容されない。

連鎖形式のオペレーションを起動する DSA は、そのオペレーションの引数に署名することがオプションで可能である。そのように要求された場合、オペレーションを実行する DSA はオペレーションの結果に署名してもよい。

連鎖形式のオペレーションは、パラメータ化された種別の IN-chained{}を用いて規定される。

```
IN-chained {OPERATION:operation} OPERATION ::= {
    ARGUMENT            OPTIONALLY-SIGNED {SET {
            IN-chainedArgument              (IN-ChainingArguments),
            argument                [0]     operation.&ArgumentType}}
    RESULT              OPTIONALLY-SIGNED {SET {
            IN-chainedResult                ABSENT,
            result                  [0]     operation.&ResultType}}
    ERRORS                  {operation.&Errors EXCEPT(referral|dsaReferral)}
    CODE                    operation.&code}
```

a) IN-chainedArgument は、ChainingArguments の値であり、元々DUA が提供した引数に加えて、オペレーションを実行する DSA が実行するために必要な情報を含む。
b) argument は、operation.&Argument の値であり、元々DUA が提供した引数からなる。

要求が成功した場合、オペレーション結果には以下の要素を含む。

a) IN-chainedResult は、IN-ChainingResults の値であり、起動元 DUA に提供されるべき情報に加えて、連鎖における前の DSA に必要とされる情報を含む。ＩＮ ＣＳ－１（改）では、2 回以上の連鎖はない前提であるため、このパラメータの必要性はない。
b) result は、operation.&Result の値であり、オペレーションの実行側から返送され、起動元 DUA へ返送されるべき結果で構成される。この情報は、ITU-T 勧告 X.511|ISO/IEC 9594-3 で規定されている。

要求が失敗した場合には、operation.&Errors 集合の中のエラーの一つが返送される。ただし、 dsaReferral が referral の代わりに返送される。

2.2.5.5 連鎖エラー（Chained errors）

dsaReferral エラーは、DSA が何らかの理由でオペレーションを他の DSA に連鎖して実行を継続することを好まない場合に生成される。ＩＮ ＣＳ－１（改）では、DSA は他の DSA から受信したオペレーションを連鎖することはない。

```
IN-dsaReferral    ERROR ::= {
    PARAMETER           SET {
            reference               [0]     IN-ContinuationReference,
            contextPrefix                   ABSENT}
CODE                        id-errcode-dsaReferral}
```

各パラメータの意味を以下に規定する。

a) IN-ContinuationReference は、起動側が要求をさらに他の DSA に伝達するために必要な情報を含む。

b) 該当オペレーションに対する ChainingArguments の returnCrossRefs 要素が、真の値であり、リフェラルが従属またはクロス参照に基づいている場合は、context Prefix パラメータがオプションで含まれることもある。DSA の管理権限者は、どの知識参照をこの方法で返送するかを決定する（例えば、その DSA の秘密情報の場合がある）。ＩＮ ＣＳ－１（改）では、直接的な知識参照が前提であるため、このパラメータは非適用と考えられる。

2.2.6 プロトコルの概略

2.2.6.1 ROS オブジェクトおよびコントラクト

分散した DIB が存在する際に、ディレクトリ抽象サービスを提供するために一般的に必要となる DSA 間の相互動作は、dspContract として定義される。このコントラクトに加わっている DSA は、dsp-dsa クラスの ROS オブジェクトとして定義される。本標準においては、このコントラクトは DSA 抽象サービスと称する。

```
dsp-dsa    ROS-OBJECT-CLASS ::= {
        BOTH                {dspContract}
        ID                  id-rosObject-dspDSA}
```

シャドウ抽象サービスは、シャドウ供給側 DSA とシャドウ消費側 DSA 間の情報のシャドウイングを規定する。このサービスは二つの形式で表され、したがって二つの異なるコントラクトとして規定される。それらは ROS ベースの情報オブジェクトとして 2.2.6.3 項で規定される。

shadowConsumerContract は、シャドウ消費側、すなわち initiating-consumer-dsa クラスの ROS オブジェクトがコントラクトを起動するサービスの形式を表す。responding- supplier-dsa クラスの ROS オブジェクトが、このコントラクトにおいて応答する。

```
initiating-consumer-dsa    ROS-OBJECT-CLASS ::= {
        INITIATES           {shadowConsumerContract}
        ID                  id-rosObject-initiatingConsumerDSA}

responding-supplier-dsa    ROS-OBJECT-CLASS ::= {
        RESPONDS            {shadowConsumerContract}
        ID                  id-rosObject-respondingSupplierDSA}
```

shadowSupplierContract は、シャドウ供給側、すなわち initiating-supplier-dsa クラスの ROS オブジェクトがコントラクトを起動するサービスの形式を表す。responding- consumer-dsa クラスの ROS オブジェクトが、このコントラクトにおいて応答する。

```
initiating-supplier-dsa    ROS-OBJECT-CLASS ::= {
        INITIATES              {shadowSupplierContract}
        ID                     id-rosObject-initiatingSupplierDSA}

responding-consumer-dsa    ROS-OBJECT-CLASS ::= {
        RESPONDS               {shadowSupplierContract}
        ID                     id-rosObject-respondingConsumerDSA}
```

操作上の結合の集合を管理するための DSA 間の相互動作は、dopContract として定義される。

```
dop-dsa    ROS-OBJECT-CLASS ::= {
        BOTH                   {dopContract}
        ID                     id-rosObject-dopDSA}
```

このコントラクトに加わっている DSA は、dop-dsa クラスの ROS オブジェクトとして定義される。このコントラクトは ROS ベースの情報オブジェクトとして xxx 項で規定される。

2.2.6.2 DSP コントラクトおよびパッケージ

dspContract は、CONTRACT クラスの情報オブジェクトとして定義される。

```
dspContract    CONTRACT ::= {
        CONNECTION                      dspConnectionPackage
        INITIATOR CONSUMER OF           {chainedModifyPackage}
        ID                              id-contract-dsp}
```

異なるオープンシステムに存在する一組の DSA が相互動作するとき、このアソシエーションコントラクトは、No.7 信号方式のアプリケーションレイヤプロトコルとして実現され、ＩＮディレクトリシステムプロトコル（DSP）と称する。No.7 信号方式のアプリケーションという意味でのこのプロトコルの定義は、2.2.7.3 項に示す。

dspContract は、コネクションパッケージ dspConnectionPackage およびオペレーションパッケージ chainedModifyPackage からなる。

コネクションパッケージ dspConnectionPackage は、CONNECTION-PACKAGE クラスの情報オブジェクトとして定義される。これはコネクションパッケージ dapConnectionPackage と同一である。

```
dspConnectionPackage    CONNECTION-PACKAGE ::= {
        BIND                   dSABind
        UNBIND                 dSAUnbind
        ID                     id-package-dspConnection}
```

オペレーションパッケージ chainedModifyPackage は、OPERATION-PACKAGE クラスの情報オブジェクトとして定義される。このパッケージのオペレーションは、ITU-T 勧告 X.518 で規定されている。

```
chainedModifyPackage    OPERATION-PACKAGE ::= {
    OPERATIONS          {chainedAddEntry|chainedRemoveEntry|chainedModifyEntry}
    ID                  id-package-chainedModify}
```

dspContract においては、いずれの DSA が起動側になり、コントラクトのオペレーションを発行してもよい。

2.2.6.3 DISP コントラクトおよびパッケージ

shadowConsumerContract および shadowSupplierContract は、CONTRACT クラスの情報オブジェクトとして定義される。

```
shadowConsumerContract    CONTRACT ::= {
    CONNECTION                      dispConnectionPackage
    INITIATOR CONSUMER OF           {shadowConsumerPackage}
    ID                              id-contract-shadowConsumer}

shadowSupplierContract    CONTRACT ::= {
    CONNECTION                      dispConnectionPackage
    INITIATOR CONSUMER OF           {shadowSupplierPackage}
    ID                              id-contract-shadowSupplier}
```

ディレクトリ情報シャドウイングプロトコル（DISP）と称するシャドウ抽象サービスの二つの形式に対する No.7 信号方式での実現方法は、No.7 信号方式のアプリケーションという意味で 2.2.7.4 項に示す。

shadowConsumerContract および shadowSupplierContract は、共通のコネクションパッケージ dispConnectionPackage およびそれぞれに対応するオペレーションパッケージ shadowConsumerPackage あるいは shadowSupplierPackage からなる。

コネクションパッケージ dispConnectionPackage は、CONNECTION-PACKAGE クラスの情報オブジェクトとして定義される。これはコネクションパッケージ dapConnection Package と同一である。

```
dispConnectionPackage    CONNECTION-PACKAGE ::= {
    BIND                 dSAShadowBind
    UNBIND               dSAShadowUnbind
    ID                   id-package-dispConnection}
```

オペレーションパッケージ shadowConsumerPackage および shadowSupplierPackage は、OPERATION-PACKAGE クラスの情報オブジェクトとして定義される。これらのパッケージのオペレーションは、ITU-T 勧告 X.525 で規定されている。

```
shadowConsumerPackage    OPERATION-PACKAGE ::= {
    CONSUMER INVOKES                {requestShadowUpdate}
    SUPPLIER INVOKES                {updateShadow}
    ID                              id-package-shadowConsumer}

shadowSupplierPackage    OPERATION-PACKAGE ::= {
    SUPPLIER INVOKES                {coordinateShadowUpdate|updateShadow}
    ID                              id-package-shadowSupplier}
```

シャドウ消費側は、shadowConsumerContract の起動側であるため、shadowConsumer Package の消費側の役割を前提とする。すなわち、シャドウ消費側が requestShadowUpdate オペレーションを発行し、シャドウ供給側が updateShadow オペレーションを発行することを意味する。

シャドウ供給側は、shadowSupplierContract の起動側であるため、shadowSupplier Package の供給側の役割を前提とする。すなわち、シャドウ供給側がそのコントラクトのオペレーションを発行することを意味する。

2.2.7 プロトコル抽象構文

2.2.7.1 DSP 抽象構文

2.2.6.2 項で規定されているオペレーションパッケージを実現するディレクトリ ASE は、一つの抽象構文 inDirectorySystemAbstractSyntax を共有する。これは、ABSTRACT-SYNTAX クラスの情報オブジェクトとして定義される。

```
inDirectorySystemAbstractSyntax    ABSTRACT-SYNTAX ::= {
    BasicDSP-PDUs
    IDENTIFIED BY          id-as-directorySystemAS}

BasicDSP-PDUs ::= TCAPMessage {{DSP-Invokable}, {DSP-Returnable}}

DSP-Invokable    OPERATION ::= {chainedAddEntry | chainedRemoveEntry | chainedModifyEntry}

DSP-Returnable    OPERATION ::= {chainedAddEntry | chainedRemoveEntry | chainedModifyEntry}
```

2.2.6.2 項で規定されたコネクションパッケージの実現については、別の抽象構文である inDirectoryDSABinding AbstractSyntax を使用する。これは、ABSTRACT-SYNTAX クラスの情報オブジェクトとして定義される。

```
inDirectoryDSABindingAbstractSyntax    ABSTRACT-SYNTAX ::= {
        DSABinding-PDUs
        IDENTIFIED BY          id-as-directoryDSABindingAS}

DSABinding-PUDs ::= CHOICE {
        bind                   Bind {dSABind},
        unbind                 Unbind {dSAUnbind}}
```

2.2.7.2 DISP 抽象構文

2.2.6.3 項で規定されているオペレーションパッケージを実現するディレクトリ ASE は、抽象構文 inDirectoryShadowAbstractSyntax を共有する。この抽象構文は、ABSTRACT- SYNTAX クラスの情報オブジェクトとして定義される。

```
inDirectoryShadowAbstractSyntax    ABSTRACT-SYNTAX ::= {
        BasicDISP-PDUs
        IDENTIFIED BY          id-as-directoryShadowAS}

BasicDISP-PDUs ::= TCAPMessage {{DISP-Invokable}, {DISP-Returnable}}

DISP-Invokable OPERATION ::= {requestShadowUpdate|updateShadow|coordinateShadowUpdate}

DISP-Returnable OPERATION ::= {requestShadowUpdate|updateShadow|coordinateShadowUpdate}
```

同じくコネクションパッケージの実現については、別の抽象構文である inDirectoryDSAShadowBinding AbstractSyntax を使用する。これは、ABSTRACT-SYNTAX クラスの情報オブジェクトとして定義される。

```
inDirectoryDSAShadowBindingAbstractSyntax    ABSTRACT-SYNTAX ::= {
        DISPBinding-PDUs
        IDENTIFIED BY          id-as-dsaShadowBindingAS}

DISPBinding-PDUs ::= CHOICE {
        bind                   Bind {dSAShadowBind},
        unbind                 Unbind {dSAShadowUnbind}}
```

2.2.7.3 ディレクトリシステムアプリケーションコンテキスト

dspContract は、inDirectorySystemAC として実現される。このアプリケーションコンテキストは、APPLICATION-CONTEXT クラスの情報オブジェクトとして定義される。

```
inDirectorySystemAC    APPLICATION-CONTEXT ::= {
    CONTRACT                        dspContract
    DIALOGUE MODE                   structured
    TERMINATION                     basic
    ABSTRACT SYNTAXES               {dialogue-abstract-syntax|
                                        inDirectorySystemAbstractSyntax|
                                        inDirectoryDSABindingAbstractSyntax}
    APPLICATION CONTEXT NAME        id-ac-directorySystemAC}
```

2.2.7.4　ディレクトリシャドウアプリケーションコンテキスト

shadowSupplierContract は、shadowSupplierInitiatedAC として定義される。このアプリケーションコンテキストは、APPLICATION-CONTEXT クラスの情報オブジェクトとして定義される。

```
shadowSupplierInitiatedAC    APPLICATION-CONTEXT ::= {
    CONTRACT                        shadowSupplierContract
    DIALOGUE MODE                   structured
    TERMINATION                     basic
    ABSTRACT SYNTAXES               {dialogue-abstract-syntax|
                                        inDirectoryShadowAbstractSyntax|
                                        inDirectoryDSAShadowBindingAbstractSyntax}
    APPLICATION CONTEXT NAME        id-ac-inShadowSupplierInitiatedAC}
```

shadowConsumerContract は、shadowConsumerInitiatedAC として定義される。このアプリケーションコンテキストは、APPLICATION-CONTEXT クラスの情報オブジェクトとして定義される。

```
shadowConsumerInitiatedAC    APPLICATION-CONTEXT ::= {
    CONTRACT                        shadowConsumerContract
    DIALOGUE MODE                   structured
    TERMINATION                     basic
    ABSTRACT SYNTAXES               {dialogue-abstract-syntax|
                                        inDirectoryShadowAbstractSyntax|
                                        inDirectoryDSAShadowBindingAbstractSyntax}
    APPLICATION CONTEXT NAME        id-ac-inShadowConsumerInitiatedAC}
```

2.2.8　サービスへのマッピング

DSP および DISP は、TC サービスにマッピングできる。本項では、DSABind, DSAUnbind, DSAShadowBind, DSAShadowUnbind から、TTC 標準 JT-Q771 で規定されている TC ダイアログハンドリングサービスへのマッピングを定義する。

DirectoryBind サービスは、以下のように TC サービスへマッピングされる。

a) DSAShadowBind および DSABind オペレーションを起動するために、TC-BEGIN サービスが使用される。

b) DSAShadowBind および DSABind オペレーションの成功を通知するために、TC- CONTINUE サービスが使用される。

c) DSAShadowBind および DSABind オペレーションの失敗を通知するために、TC-U- ABORT サービスが使用される。

DSABind および DSAShadowBind サービスに関して、パラメータの TC サービスへのマッピングは以下のようになる。

- TC-BEGIN サービスのサービス品質（Quality of Service）パラメータは、TTC 標準 JT-Q771 で規定されているように使用し、特別な制約はつけない。
- TC-BEGIN サービスの着アドレス（Destination Address）パラメータは、応答側 DSA の役割をするＳＤＦのアドレスを含む。
- TC-BEGIN サービスのアプリケーションコンテキスト名（Application-Context-Name）パラメータは、inDirectorySystemAC, shadowSupplierInitiatedAC あるいは shadowConsumerInitiatedAC オブジェクトのアプリケーションコンテキスト名フィールドの値をとる。
- TC-BEGIN サービスの発アドレス（Originating Address）パラメータは、要求を送信したＳＤＦのアドレスを含む。
- TC-BEGIN サービスの対話 ID（Dialogue ID）パラメータは、TTC 標準 JT-Q771 で規定されているように使用する。
- TC-BEGIN サービスのユーザ情報（User Information）パラメータは、DirectoryBind Argument 型の値を含む。
- TC-BEGIN サービスのコンポーネントプレゼント（Component Present）パラメータは、TTC 標準 JT-Q771 で規定されているように使用する。
- TC-CONTINUE サービスのサービス品質（Quality of Service）パラメータは、TTC 標準 JT-Q771 で規定されているように使用し、特別な制約はつけない。
- TC-CONTINUE サービスの発アドレス（Originating Address）パラメータは、TTC 標準 JT-Q771 で規定されているように使用する。存在する場合は、応答側 DSA の役割をするＳＤＦのアドレスを含む。
- TC-CONTINUE サービスのアプリケーションコンテキスト名（Application-Context-Name）パラメータは、TTC 標準 JT-Q771 で規定されているように使用する。
- TC-CONTINUE サービスの対話 ID（Dialogue ID）パラメータは、TTC 標準 JT-Q771 で規定されているように使用する。認証されたリレーションシップ ID は、TCAP 対話 ID にマッピングされる。
- TC-CONTINUE サービスのユーザ情報（User Information）パラメータは、Directory BindResult 型の値を含む。
- TC-CONTINUE サービスのコンポーネントプレゼント（Component Present）パラメータは、TTC 標準 JT-Q771 で規定されているように使用する。
- TC-U-ABORT サービスのサービス品質（Quality of Service）パラメータは、TTC 標準 JT-Q771 で規定されているように使用し、特別な制約はつけない。
- TC-U-ABORT サービスの対話 ID（Dialogue ID）パラメータは、TTC 標準 JT-Q771 で規定されているように使用する。認証されたリレーションシップ ID は、TCAP 対話 ID にマッピングされる。
- TC-U-ABORT サービスのアボート理由（Abort Reason）パラメータは、TTC 標準 JT-Q771 で規定されているように使用する。

- TC-U-ABORT サービスのアプリケーションコンテキスト名（Application-Context-Name）パラメータは、TTC 標準 JT-Q771 で規定されているように使用する。ＳＤＦが、受信したアプリケーションコンテキスト名をサポートしていないという理由で対話を拒否する場合は、このパラメータは、inDirectorySystemAC, shadowSupplier InitiatedAC あるいは shadowConsumerInitiatedAC オブジェクトのアプリケーションコンテキスト名フィールドの値をとる。
- アボート理由パラメータが「対話拒否」の値をとる場合、TC-U-ABORT サービスのユーザ情報（User Information）パラメータは、DirectoryBindError 型の値を含む。それ以外の場合には存在しない。

DSAUnbind および DSAShadowUnbind サービスは、TC-END サービスにマッピングされる。TC-END サービスのパラメータの使用法を以下に示す。

- サービス品質（Quality of Service）パラメータは、TTC 標準 JT-Q771 で規定されているように使用し、特別な制約はつけない。
- アプリケーションコンテキスト名（Application-Context-Name）パラメータは、この段階では使用しない。
- 対話 ID（Dialogue ID）パラメータは、TTC 標準 JT-Q771 で規定されているように使用する。
- ユーザ情報（User Information）パラメータは、空である。
- コンポーネントプレゼント（Component Present）パラメータは、TTC 標準 JT-Q771 で規定されているように使用する。

ディレクトリ ASE サービスは、TC コンポーネントハンドリングサービスにマッピングされる。オペレーションおよびエラーの TC サービスへのマッピングは、 TTC 標準 JT-Q774 で規定されている。TC-INVOKE-要求プリミティブのタイムアウトパラメータは、以下の表にしたがって設定される。

表 1　DSP/DISP オペレーションの TC タイマ値

オペレーション	タイムアウト
連鎖エントリ追加（chainedAddEntry）	中
連鎖エントリ削除（chainedRemoveEntry）	中
連鎖エントリ更新（chainedModifyEntry）	中
連鎖探索（chainedSearch)	中
シャドウ更新（updateShadow)	中
シャドウ更新調整（coordinateShadowUpdate）	中
シャドウ更新要求（requestShsdowUpdate）	中

注）タイムアウトの“中”は、60 秒以下のタイムアウト値を示す。

2.2.9 コンフォーマンス

ＳＤＦのコンフォーマンスに関して、既に記述されているリストに加えて、以下の記述を追加すべきである。

2.2.9.1 ＳＤＦによるコンフォーマンス

2.2.9.1.1 ステートメントの要求条件

以下が記述されるべきである。

a) コンフォーマンスが要求されるアプリケーションコンテキスト。本標準の現在の版では、inDirectorySystemAC アプリケーションコンテキストに対するコンフォーマンスが必要である。
注）アプリケーションコンテキストは、ここに記述されているものを除いて分割すべきではない。特に、コンフォーマンスは特定のオペレーションに対して要求されるべきではない。
b) コンフォーマンスが inDirectorySystemAC アプリケーションコンテキストに対して要求される場合、ITU-T 勧告 X.518 で規定されているオペレーションの連鎖モードがサポートされているか否か。
c) コンフォーマンスが要求されるセキュリティレベル（無し，簡易，厳密）。
d) コンフォーマンスが要求される属性型。さらに、構文 DirectoryString に基づく属性に対して、UNIVERSAL STRING 選択のためのコンフォーマンスが要求されるか否か。
e) コンフォーマンスが要求されるオブジェクトクラス。
f) ITU-T 勧告 X.501 の 8.8 項および ITU-T 勧告 X.511 の 7.6, 7.8.2, 9.2.2 項で規定されているコレクティブ属性に対して、コンフォーマンスが要求されるか否か。
g) ITU-T 勧告 X.511 の 7.6, 7.8.2, 9.2.2 項で規定されているハイアラキカル属性に対して、コンフォーマンスが要求されるか否か。
h) コンフォーマンスが要求される、ITU-T 勧告 X.501 で規定されているオペレーショナル属性型および他のオペレーショナル属性型。
i) ITU-T 勧告 X.511 の 7.7.1 項で規定されている別名の返送に対して、コンフォーマンスが要求されるか否か。
j) ITU-T 勧告 X.511 の 7.7.6 項で規定されている、返送されたエントリ情報が完全であることを示すためのコンフォーマンスが要求されるか否か。
k) ITU-T 勧告 X.511 の 11.3.2 項で規定されている、補助オブジェクトクラスを識別する値を追加および／あるいは削除するために、オブジェクトクラスの属性を変更するためのコンフォーマンスが要求されるか否か。
l) 基本アクセス制御に対してコンフォーマンスが要求されるか否か。
m) 簡易アクセス制御に対してコンフォーマンスが要求されるか否か。
n) コンフォーマンスが要求される名前バインディング。
o) ITU-T 勧告 X.501 で規定されているコレクティブ属性をＳＤＦが管理できるか否か。
p) 属性コンテキストに対してコンフォーマンスが要求されるか否か。

2.2.9.1.2　静的な要求条件

ＳＤＦは、

a) 2.2.7 項で抽象構文によって定義されており、コンフォーマンスが要求されるアプリケーションコンテキストをサポートする能力をもつ。
b) ITU-T 勧告 X.501 で抽象構文によって定義されている情報の枠組みをサポートする能力をもつ。
c) ITU-T 勧告 X.518 で定義されている最小の知識の要求条件に適合する。
d) 抽象構文によって定義されており、コンフォーマンスが要求される属性型をサポートする能力をもつ。
e) 抽象構文によって定義されており、コンフォーマンスが要求されるオブジェクトクラスをサポートする能力をもつ。
f) 前項においてコンフォーマンスが要求されている拡張に適合する。
g) コレクティブ属性に対してコンフォーマンスが要求されるのであれば､ITU-T 勧告 X.511 の 7.6, 7.8.2, 9.2.2 項で定義されている関連の手順を実行する能力をもつ。
h) ハイアラキカル属性に対してコンフォーマンスが要求されるのであれば、ITU-T 勧告 X.511 の 7.6, 7.8.2, 9.2.2 項で定義されている関連の手順を実行する能力をもつ。
i) コンフォーマンスが要求されるオペレーショナル属性型をサポートする能力をもつ。
j) 基本アクセス制御に対してコンフォーマンスが要求されるのであれば、基本アクセス制御の定義に適合する ACI アイテムを保持する能力をもつ。
k) 簡易アクセス制御に対してコンフォーマンスが要求されるのであれば、簡易アクセス制御の定義に適合する ACI アイテムを保持する能力をもつ。

2.2.9.1.3　動的な要求条件

ＳＤＦは、

a) 2.2.8 項で定義されている「使用するサービスへのマッピング」に適合する。
b) ITU-T 勧告 X.518 で定義されている、リフェラルに関するディレクトリの分散オペレーションに対する手順に適合する。
c) directorySystemAC アプリケーションコンテキストに対してコンフォーマンスが要求されるのであれば、ITU-T 勧告 X.518 で定義されている相互動作のリフェラルモードに適合する。
d) 相互動作の連鎖モードに対してコンフォーマンスが要求されるのであれば、 ITU-T 勧告 X.518 で定義されている相互動作の連鎖モードに適合する。
注）この場合にのみ、DSA が directorySystemAC のオペレーションを発行可能であることが必要である。
e) xxx 項で定義されている拡張手順の規則に適合する。
f) 基本アクセス制御に対してコンフォーマンスが要求されるのであれば､基本アクセス制御手順に従ってＳＤＦ内の情報を保護する能力をもつ。
g) 簡易アクセス制御に対してコンフォーマンスが要求されるのであれば､簡易アクセス制御手順に従ってＳＤＦ内の情報を保護する能力をもつ。

2.2.9.2　シャドウ供給側によるコンフォーマンス

シャドウ供給側の役割において、本ディレクトリ仕様に対するコンフォーマンスを要求するＳＤＦのインプリメンテーションは、以下に規定する要求条件を満たす必要がある。

2.2.9.2.1　ステートメントの要求条件

以下が記述されるべきである。

a) シャドウ供給側としてコンフォーマンスが要求されるアプリケーションコンテキスト、すなわち inShadowSupplierInitiatedAC。
b) コンフォーマンスが要求されるセキュリティレベル（無し，簡易，厳密）。
c) UnitOfReplication がどの程度までサポートされるか、特に、以下のオプションのフィーチャの内いずれがサポートされるか、
    - ObjectClass に関するエントリのフィルタリング
    - AttributeSelection による属性の選択／除外
    - 複製領域における従属知識の包含
    - 従属知識に加えて拡張知識の包含

2.2.9.2.2　静的な要求条件

ＳＤＦは、

a) 2.2.7 項で抽象構文によって定義されており、コンフォーマンスが要求されるアプリケーションコンテキストをサポートする能力をもつ。
b) modifyTimestamp および createTimestamp オペレーショナル属性のサポートを提供する。

2.2.9.2.3　動的な要求条件

ＳＤＦは、

a) 2.2.8 項で定義されている「使用するサービスへのマッピング」に適合する。
b) DISP に関係する ITU-T 勧告 X.525|ISO/IEC 9594-9 の手順に適合する。

2.2.9.3　シャドウ消費側によるコンフォーマンス

シャドウ消費側として、本ディレクトリ仕様に対するコンフォーマンスを要求するＳＤＦのインプリメンテーションは、以下に規定する要求条件を満たす必要がある。

2.2.9.3.1　ステートメントの要求条件

以下が記述されるべきである。

a) シャドウ消費側としてコンフォーマンスが要求されるアプリケーションコンテキスト、すなわち inShadowConsumerInitiatedAC。
b) コンフォーマンスが要求されるセキュリティレベル（無し，簡易，厳密）。
c) ＳＤＦが重複する複製単位のシャドウイングをサポートするか否か。

2.2.9.3.2　静的な要求条件

ＳＤＦは、

a) 2.2.7 項で抽象構文によって定義されており、コンフォーマンスが要求されるアプリケーションコンテキストをサポートする能力をもつ。
b) 重複する複製単位がサポートされる場合、modifyTimestamp および create Timestamp オペレーショナル属性のサポートを提供する。
c) copyShallDo サービス制御のサポートを提供する。

2.2.9.3.3　動的な要求条件

ＳＤＦは、

a) 2.2.8 項で定義されている「使用するサービスへのマッピング」に適合する。
b) DISP に関係する ITU-T 勧告 X.525|ISO/IEC 9594-9 の手順に適合する。

2.2.10　X.500 勧告のプロファイル

2.2.10.1　X.501 プロファイル

シャドウイングをサポートするためには、ITU-T 勧告 X.501 の 8 および 9 章の考慮が必要となる。そこには、DIT を複数のＳＤＦ上にどのように分散可能であるかが記述されており、また、知識参照および一つのデータベースを複数のデータベースに分割するために必要となるすべての情報の記述も含まれている。DOP で使用されるオペレーショナルバインディングメカニズムも定義されている。

2.2.10.2　X.518 プロファイル

1 章から 4 章、および 5 章の 19.1 項までがインポートされるべきである。

2.2.10.3　X.525 プロファイル

ITU-T 勧告 X.525 からは、6.3.2, 8, 10.2, 11.2 項以外はインポートされるべきである。

３．　手順

3.1　手順とエンティティの定義

3.1.1　ＳＣＦ応用エンティティ手順

3.1.1.1　概要

この章は、網間ＩＮＡＰにおける SCF-SDF インタフェースに関連したＳＣＦ応用エンティティ（AE）手順の定義を提供する。その手順は No.7 共通線信号方式(SS#7）の利用を前提としている。別の信号方式も用いられうる。

更に、他の能力はＳＣＰ，ＡＤもしくはＳＮにおいてインプリメント依存な方法でサポートされるかもしれない。

TTC 標準 JT-Q700、JT-Q771、および ITU-T 勧告 Q.1400 の中で定義されたアーキテクチャに従って、ＡＥは、ＴＣＡＰ（トランザクション機能応用部）及び一つ以上のＴＣユーザと呼ばれる一つ以上のＡＳＥを含む。　以下の章は、TTC 標準 JT-Q771 で規定されたプリミティブを用いているＴＣＡＰとインタフェースする、ＴＣユーザＡＳＥとＳＡＣＦ＆ＭＡＣＦ規則を定義する。

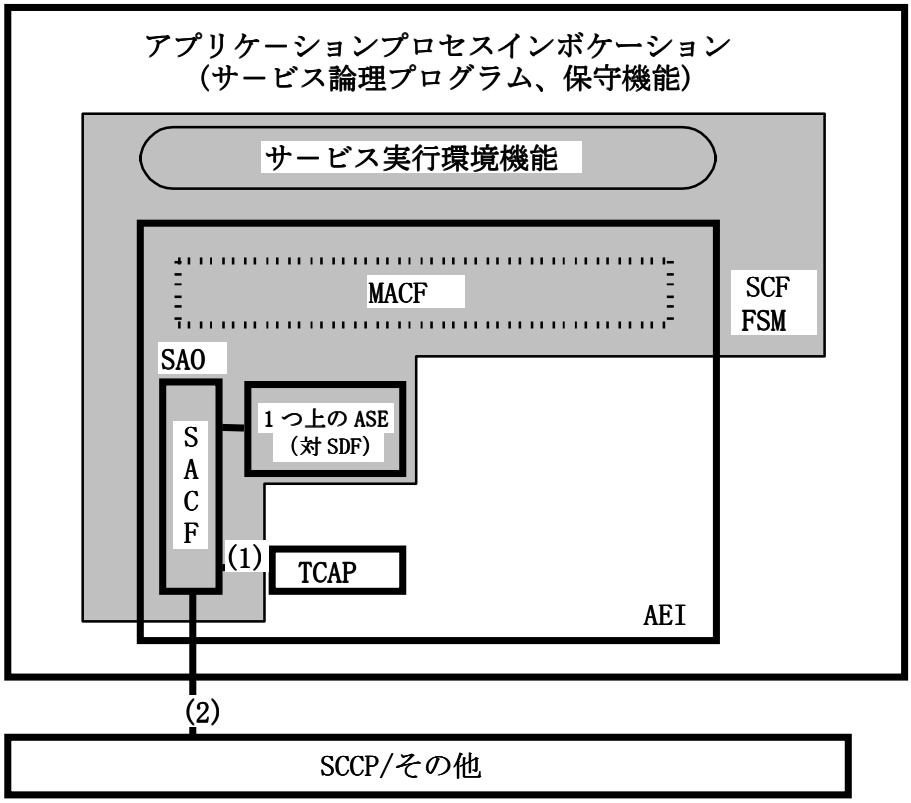
その手順は、定義されたアプリケーションレイヤ構造をサポートしている、他のメッセージ形式の信号方式で同等に用いられるかもしれない。このテキストは、サービス論理プログラムに対し何らの制約を規定することを意図しているわけではない。

以下に規定されるアプリケーションエンティティの手順の解釈がＴＣＡＰサービスを利用するためのルールと詳細手順と異なる場合には、本標準の 3 編 2.1 章および 3.3 章に従うものとする。

3.1.1.2　モデルとインタフェース

AE-SCF の機能モデルを図 3-3-1/JT-Q1218（ITU-T Q.1218）に示す。ＡＳＥはＳＤＦと通信するためにサポートプロトコルレイヤにインタフェースし、サービス論理プログラム及び保守機能とインタフェースする。この標準の範囲は、図 3-3-1/JT-Q1218（ITU-T Q.1218）において影を付けた部分に限定される。

図 3-3-1/JT-Q1218（ITU-T Q.1218）に示されたインタフェースは、TTC 標準 JT-Q771 の中で規定された TC ユーザＡＳＥのプリミティブ（インタフェース(1)）を用いる。インテリジェントネットワークアプリケーションプロトコル（INAP）のオペレーションとパラメータは、この標準の第 3 編 2 章で定義されている。



(1) TC プリミティブ
(2) N プリミティブ

AEI:Application Entity Invocation（アプリケーションエンティティインボケーション）
SCF:Service Control Function（サービス制御機能）
FSM:Finite State Model（有限状態モデル）
MACF:Multiple Association Control Function（複数アソシエーション制御機能）
SACF:Single Association Control Function（単一アソシエーション制御機能）
SAO:Single Association Object（単一アソシエーションオブジェクト）

SCF FSM はいくつかの有限状態機械を含む事に注意

注：本標準では MACF については対象外である

図 3-3-1/JT-Q1218（ITU-T Q.1218）SCF AE の機能モデル

3.1.1.3 SCF FSM と SLP/保守機能間の相互関係

SCF FSM とサービス論理プログラム/保守機能間のプリミティブインタフェースは、内部インタフェースであり、本標準での対象ではない。

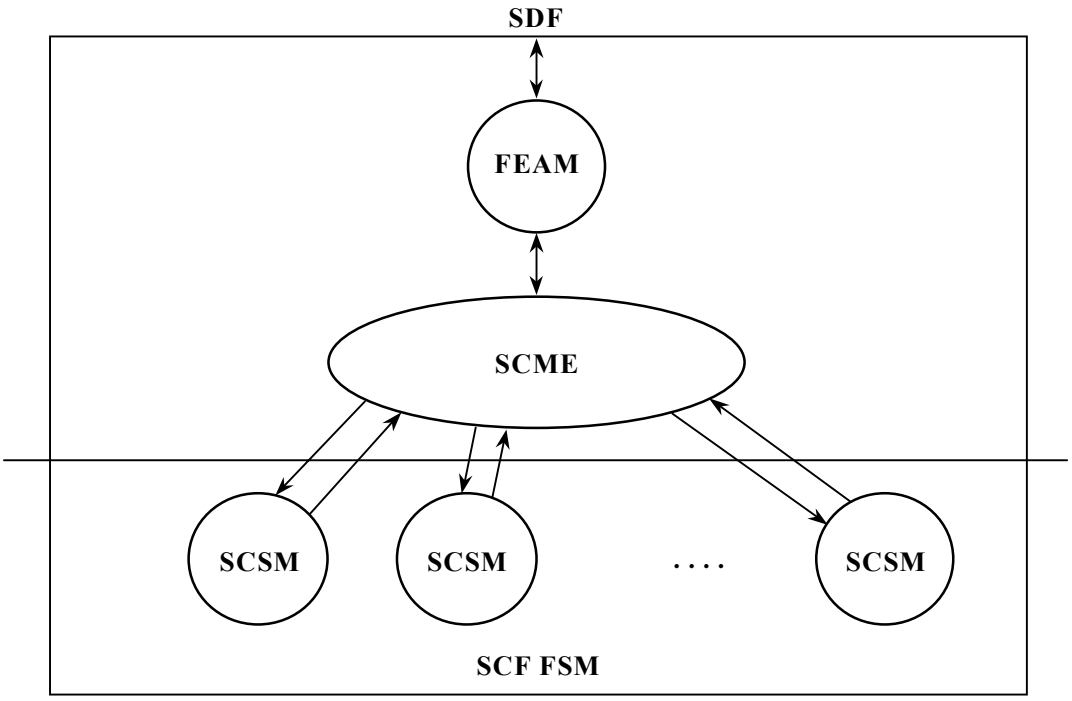
サービス論理プログラムと SCF FSM 間の相互関係は、（エンドユーザにより起動された呼とＩＮサービス論理により起動された呼の両方場合について、）次のように記述されるかもしれない。

－ ＩＮ呼処理要求により、ＳＣＦ呼状態モデル（SCSM）のインスタンスが生成され、対応するサービス論理プログラムが起動される。

－　　呼の起動がサービス論理から要求されると、SCSM のインスタンスが生成される。

どちらの場合においても、SCF FSM は要求に応じて SDF FSM との相互動作を処理し、必要に応じてイベントをサービス論理プログラムに通知する。

ＳＣＦから受信されたオペレーションの実行に関連した管理機能は、SCF 管理エンティティ（SCME）によって実行される。SCME は SCME 制御と複数の SCME FSM インスタンスにより構成される。SCME 制御は、それぞれ異なった複数の SCF 呼状態モデル（SCSM）及び機能エンティティアクセスマネージャ（FEAM）とインタフェースする。図 3-3-2/JT-Q1218（ITU-T Q.1218）は SCF FSM 構造を示す。



SCME: SCF 管理エンティティ　　FEAM: 機能エンティティアクセスマネージャ
SCSM: SCF 呼状態モデル

図 3-3-2/JT-Q1218（ITU-T Q.1218）SCF FSM　構造

以下のテキストは、エンティティの機能的能力よりむしろオペレーションの正確な順序を規定することを主要目的として、ＳＣＦと他の機能エンティティ間のインタフェースの手順の側面を系統的に記述している。従って、このテキストはＳＣＦの機能的能力のサブセットだけを記述している。

SCSM はサービス論理のためにＳＤＦとの対話を維持する。

複数の要求が並行して、非同期的にＳＣＦにより実行されており、このことは SCF FSM オブジェクトの生成、起動、維持のタスクを実行するエンティティが一つ必要であることを意味する。このエンティティは SCF 管理エンティティ制御（SCME 制御）と呼ばれる。上記のタスクに加えて、SCME は SCSM の全てのインスタンスのためにＳＤＦとの対話を維持する。特に SCME 制御は:

1. 別のＦＥからの入力メッセージを解釈し、それらを対応する SCSM イベントに翻訳する。
2. SCSM 出力を別のＦＥへの対応するメッセージに翻訳する。
3. （呼処理と）非同期的な全ての動作を実行する。（そのような動作の一つはフロー制御である。）
4. ＳＣＦと他のＦＥ間の持続的相互動作をサポートする。

最後に、機能エンティティアクセスマネージャ（FEAM）は SCME から低レベルのインタフェース機能を軽減する。FEAM の機能は次のようなものを含んでいる:

1. ＳＤＦとのインタフェースを設定し、維持すること。
2. ＳＤＦから受信されたメッセージを SCME に引き渡すこと（必要な時には待ち合わせること）。
3. SCME から受信したメッセージを形式化し、（必要な時には）待ち合わせ、ＳＤＦに送信すること。

3.1.1.4 SDF 関連状態

ＳＤＦとの相互動作は、ＳＣＦの任意の状態から可能である。以下の項では一つのＳＣＦと一つのＳＤＦの間の関連状態が規定されている。もし、ＳＣＦが他のＳＤＦへアクセスする必要があれば、新しい FSM が生成される。

以下では、状態とイベントに番号が付与されている。図 3-3-3/JT-Q1218（ITU-T Q.1218）を参照のこと

（注：SCF-SDF 間にディレクトリアクセスプロトコルが導入されたため、1993 年版 ITU-T Q.1218 の記述と本節の記述は一部異なっている。異常時の取扱については第 3 編 2.1 章の記述に従う）

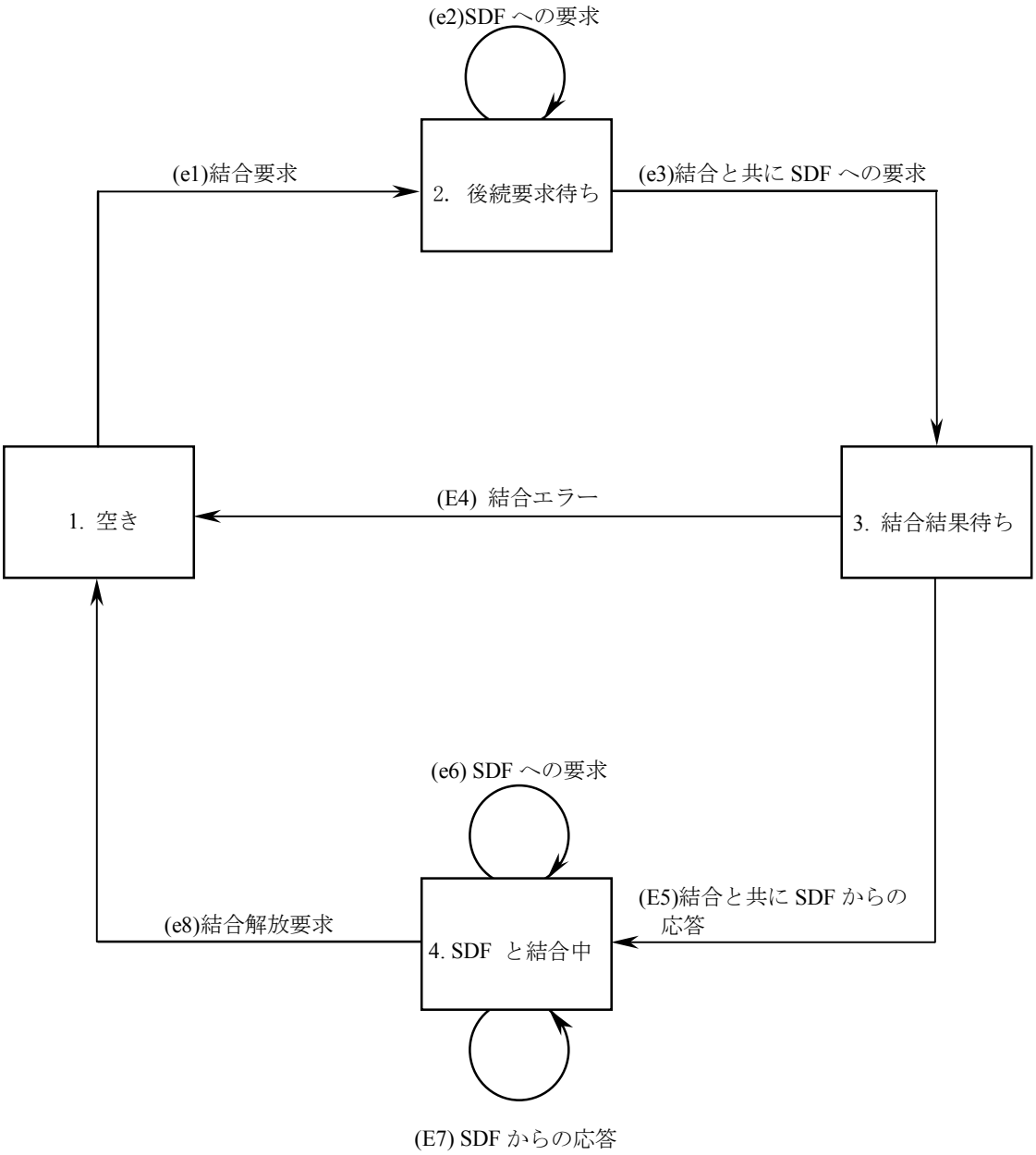


図 3-3-3/JT-Q1218（ITU-T Q.1218） SCSM FSM （SDF **関連部**）

3.1.1.4.1 状態 1：”空き（Idle）”

この状態は、ＳＤＦに対する要求が発行されるより前の任意の状態を表すサブ状態である。以下のイベントがこの状態において考慮される。

・（e1）結合要求(Bind_Request):これは内部イベントであり、データへのアクセスを開始するためにＳＤＦへの結合をサービス論理が必要とすることにより生ずる。このイベントは状態 2, 後続要求待ち (Wait for subsequent requests)への遷移を生ずる。サービス論理から結合の要求を受信することで本遷移は生じる。

3.1.1.4.2　状態　2：”後続要求待ち(Wait for subsequent requests)”

この状態において、Bind オペレーション（DirectoryBind）と共に（同一メッセージで）送られるべき引き続くオペレーションが存在するかも知れない。以下 2 つのイベントがこの状態において考慮される。

・（e2）SDF への要求（Request_to_SDF）:これは内部イベントであり、DirectoryBind に引き続くオペレーションを受信することにより生ずる。オペレーションはデリミタの受信（またはタイマ満了）までバッファリングされる。SCSM は同一状態に留まる。
・（e3）結合と共に SDF への要求（Request_to_SDF_with_Bind）:これは内部イベントであり、デリミタの受信により引き起こされる。デリミタは送信されるべき最終オペレーションの受信を示している。いったんデリミタが受信されると、DirectoryBind オペレーションのアーギュメントと、もしあれば、その他のオペレーションのアーギュメントを含むメッセージがＳＤＦへ送信される。このイベントは本状態から状態 3、結合結果待ち（Wait for Bind result）への遷移を生じる。ＳＤＦへ以下のオペレーションが発行される：
ディレクトリ結合（DirectoryBind）またはディレクトリ結合（DirectoryBind）と本状態においてサービス論理から受け取った要求に対応する一つのディレクトリアクセスオペレーション（探索：Search、エントリ更新：ModifyEntry、エントリ追加：AddEentry、エントリ削除：RemoveEntry の何れか）

3.1.1.4.3　状態 3：”結合結果待ち（Wait for Bind result）”

この状態において、ＳＣＦはＳＤＦからの応答を待っている。２つのイベントがこの状態において考慮される。

・（E5）結合と共に SDF からの応答(Response_from_SDF_with_Bind):これは外部 イベントであり、以前にＳＤＦに対して発行されたディレクトリ結合（DirectoryBind）オペレーションに対する成功応答、またはディレクトリ結合（DirectoryBind）オペレーションと他の（一つの）オペレーションに対する応答の受信により生ずる。このイベントは本状態から状態 4,SDF と結合中(SDF　Bound)への遷移を生ずる。本イベントはＳＤＦから以下の応答を受信することにより生じる：
ディレクトリ結合（DirectoryBind）の結果応答、またはディレクトリ結合（DirectoryBind）の結果応答と一つのディレクトリアクセスオペレーション（探索：Search、エントリ更新：ModifyEntry、エントリ追加：AddEentry、エントリ削除：RemoveEntry の何れか）の結果応答またはエラー応答

・（E4）ディレクトリ結合エラー(Bind_Error):これは外部イベントであり、以前にＳＤＦに対して発行されたディレクトリ結合（DirectoryBind）オペレーションに対するエラー応答の受信により生じる。このイベントは本状態から状態 1,空き(Idle)への遷移を生ずる。本イベントはＳＤＦから以下の応答を受信することにより生じる：ディレクトリ結合（DirectoryBind）のエラー応答

3.1.1.4.4　状態　4：”SDF と結合中(SDF Bound)”

この状態において、ＳＣＦはＳＤＦとの間に認証済みアクセスを確立している。そして、サービス論理からＳＤＦへの（データ更新、取得の複数の）または、以前にＳＤＦへ発行されたオペレーションに対する応答を待っている。３つのイベントがこの状態において考慮される。

・（e6）SDF への要求(Request_to_SDF):これは内部イベントであり、ＳＤＦ内のデータへアクセスすることをサービス論理が必要とする場合に生ずる。SCSM は同一状態に留まる。ＳＣＦはＳＤＦへ以下のオペレーションを発行する：
一つのディレクトリアクセスオペレーション（探索：Search、エントリ更新：ModifyEntry、エントリ追加：AddEntry、エントリ削除：RemoveEntry の何れか）

・（E7）SDF からの応答(Response_from_SDF):これは外部イベントであり、以前にＳＤＦに対して発行された複数のオペレーションに対する応答の受信により生ずる。SCSM は同一状態に留まる。本イベントはＳＤＦから以下の応答を受信することにより生じる：
一つのディレクトリアクセスオペレーション（探索：Search、エントリ更新：ModifyEntry、エントリ追加：AddEntry、エントリ削除：RemoveEntry の何れか）の結果応答またはエラー応答

・（e8）結合解放要求(Unbind_Request):これは内部イベントであり、ＳＤＦとの間の認証済みアクセスをサービス論理が解放する必要の生じた時に生じる。このイベントは本状態から状態　1,空き（Idle）への遷移を生ずる。ＳＣＦはＳＤＦへ以下のオペ レーションを発行する：
ディレクトリ結合解放（Unbind）

3.1.2　SDF 応用エンティティ手順

3.1.2.1　概要

この章は、網間ＩＮＡＰにおける SDF-SCF および SDF-SDF インタフェースに関連した SDF アプリケーションエンティティ（AE）手順の定義を提供する。その手順は No.7 共通線信号方式(SS#7）の利用を前提としている。

他の能力はＳＣＰ，ＳＤＰ，ＳＳＰ，ＡＤ,もしくはＳＮにおいてインプリメント依存な方法でサポートされるかもしれない。

TTC 標準 JT-Q700、JT-Q771、および ITU 勧告 Q.1400 の中で定義されたアーキテクチャに従って、ＡＥは、ＴＣＡＰ（トランザクション機能応用部）及び一つ以上のＴＣユーザと呼ばれるＡＳＥを含み、これらはディレクトリに基づいて構成される（X.500 シリーズ勧告）。以下の章は、TTC 標準 JT-Q771 で規定されたプリミティブを用いているＴＣＡＰとインタフェースする、ＴＣユーザＡＳＥとＳＡＣＦ＆ＭＡＣＦ規則を定義する。

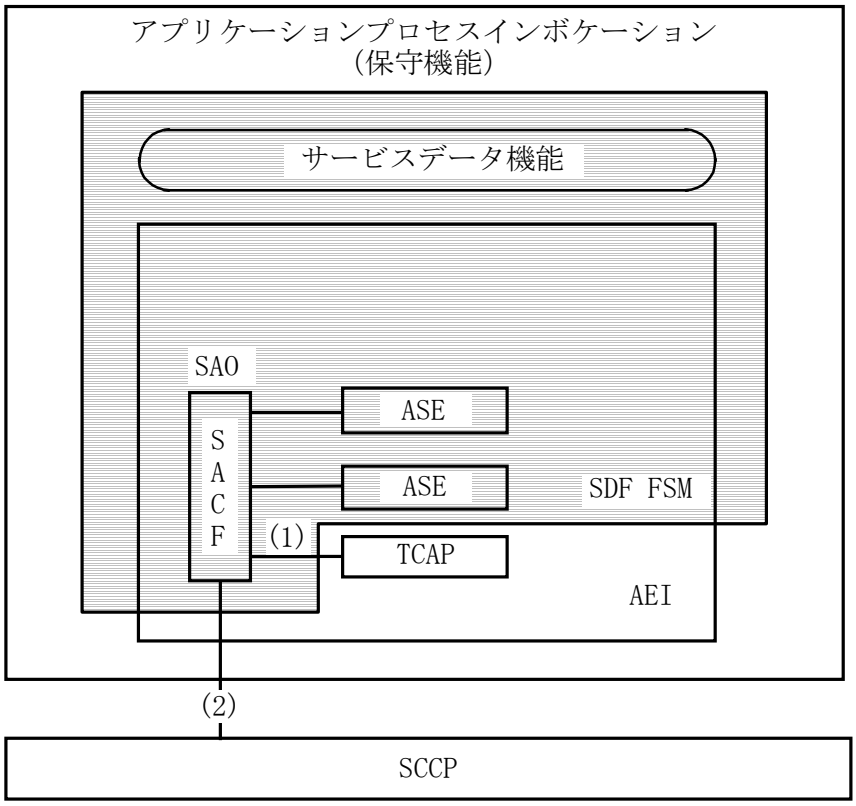
その手順は、定義されたアプリケーションレイヤ構造をサポートしている、他の信号メッセージ転送方式で同等に用いられるかもしれない。

本章で定義されるアプリケーションエンティティ手順が第 3 編 2.1 章と異なる場合には、2.1 章で定義されるルールに従うこととする。

3.1.2.2 モデルとインタフェース

AE-SDF の機能モデルを図 3-3-4/JT-Q1218（ITU-T Q.1218）に示す。ＡＳＥはＳＣＦおよび他のＳＤＦと通信するためにＴＣＡＰとインタフェースし、保守機能とインタフェースする。この標準の範囲は、図 3-3-4/JT-Q1218（ITU-T Q.1218）において影をつけた部分に限定される。

図 3-3-4/JT-Q1218（ITU-T Q.1218）に示されたインタフェースは、TTC 標準 JT-Q771 の中で規定されたＴＣユーザＡＳＥのプリミティブ（インタフェース(1)）を用いる。網間インテリジェントネットワークアプリケーションプロトコル（網間ＩＮＡＰ）のオペレーションとパラメータは、この標準の第 3 編 2 章で定義されている。



(1) TCプリミティブ
(2) Nプリミティブ

AEI:Application Entity Invocation（アプリケーションエンティティインボケーション）
SDF:Service Data Functions（サービスデータ機能）
FSM:Finite Status Model（有限状態モデル）
SACF:Single Association Control Function（単一アソシエーション制御機能）
SAO:Single Associoation Object（単一アソシエーションオブジェクト）

SDF FSM はいくつかの有限状態機械を含むことに注意

図 3-3-4/JT-Q1218（ITU-T Q.1218）　SDF AE の関連モデル

以下のいずれかの事象が生じた際に、SDF FSM のインスタンスが生成される。

(1) ＩＮの呼に関するあるいは呼と関連しない要求をＳＣＦから受信した場合

(2) ＳＤＦが別のＳＤＦに連鎖したオペレーションを発行あるいは別のＳＤＦから連鎖したオペレーションを受信した場合

(3) ＳＤＦが別のＳＤＦにシャドウイングオペレーションを発行あるいは別のＳＤＦからシャドウイングオペレーションを受信した場合

SDF FSM は、SCF FSM および別の SDF FSM との相互動作を処理する。

3.1.2.3 SDF FSM の構成

SDF FSM の構成を図 3-3-5/JT-Q1218 に示す。

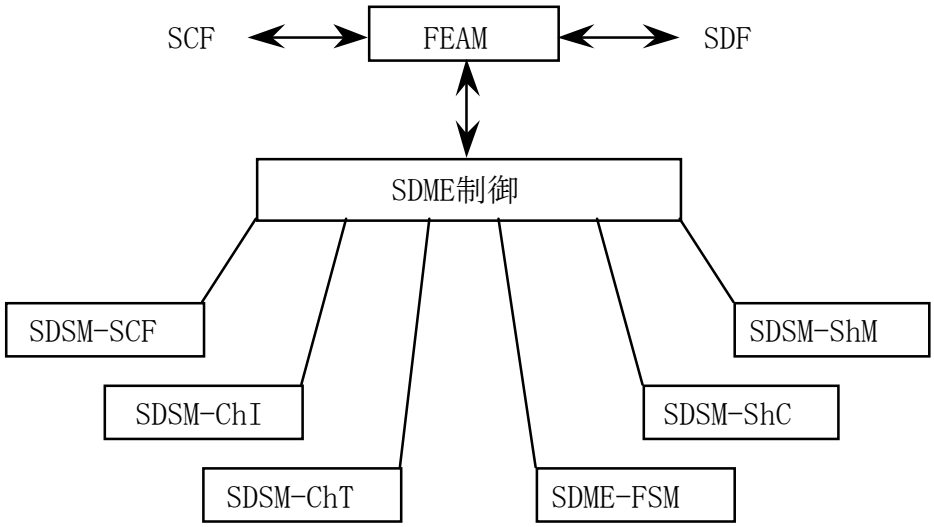


図3-3-5/JT-Q1218　SDFインタフェース

SDF FSM（SDSM-SCF）は SCF FSM との相互動作を処理する。SDSM-ChI および SDSM-ChT は連鎖の起動および終端に関するＳＤＦ間の相互動作を処理する。SDSM-ShM および SDSM-ShC はシャドウイングのマスタ制御およびコピー制御に関するＳＤＦ間の相互動作を処理する。SDME-FSM はＳＤＦとＳＤＦ管理機能間の相互動作を処理する。

ＳＣＦあるいは協調しているＳＤＦから受信したオペレーションの実行に関する管理機能は、SDF 管理エンティティ（SDME）により実行される。SDME は SDME 制御および SDME FSM の複数のインスタンスからなる。SDME 制御は、機能エンティティアクセスマネージャ（FEAM）のみならず、異なる SDF FSM（例、SDSM-SCF）および SDME-FSM ともそれぞれインタフェースを持つ。

FEAM は以下を含む下位レベルのインタフェース保守機能を提供する。

1) ＳＣＦおよび協調しているＳＤＦとのインタフェースの設定および保守
2) ＳＣＦおよび協調しているＳＤＦから受信したメッセージの SDME 制御に対する転送および（必要であれば）キューイング
3) SDME 制御から受信したメッセージの編集，（必要であれば）キューイング，およびＳＣＦおよび協調しているＳＤＦへの送信

SDME 制御は、すべての SDF FSM（例、SDSM-SCF, SDSM-ChI）のインスタンスのために、ＳＣＦおよび協調しているＳＤＦとのアソシエーションを保守する。SDF FSM のこれらのインスタンスは、ＳＤＦに関連するイベントの生起にしたがって同時にあるいは非同期に起こる。これはすなわち、SDF FSM の生成, 起動および保守を実行するためには唯一のエンティティが必要であることを意味する。特に、SDME 制御は以下の機能を実行する。

1) 他のＦＥからの入力メッセージを解釈し、対応する SDF FSM イベントに変換する。
2) SDF FSM からの出力を他のＦＥへの対応するメッセージに変換する。
3) ＳＤＦ内の管理および監視機能に関する非同期の動作を検出し、SDME-FSM のインスタンスを生成する。例えば、網事業者間でのシャドウイング手順に関する管理による起動である。この場合、SDME 制御は管理に関するオペレーションを処理するために SDME-FSM のインスタンスを生成する。

3.1.2.4　ＳＤＦ状態遷移モデル

3.1.2.4.1　ＳＣＦ関連の状態に対するＳＤＦ状態遷移モデル

網間ＩＮＡＰにおけるＳＣＦとの相互動作に関しては、ＳＤＦは、ディレクトリ結合（Bind）手順後のＳＣＦからの全ての要求に応答する。図 3-3-6/JT-Q1218（ITU-T Q.1218）に有限状態モデル（FSM）を示す。

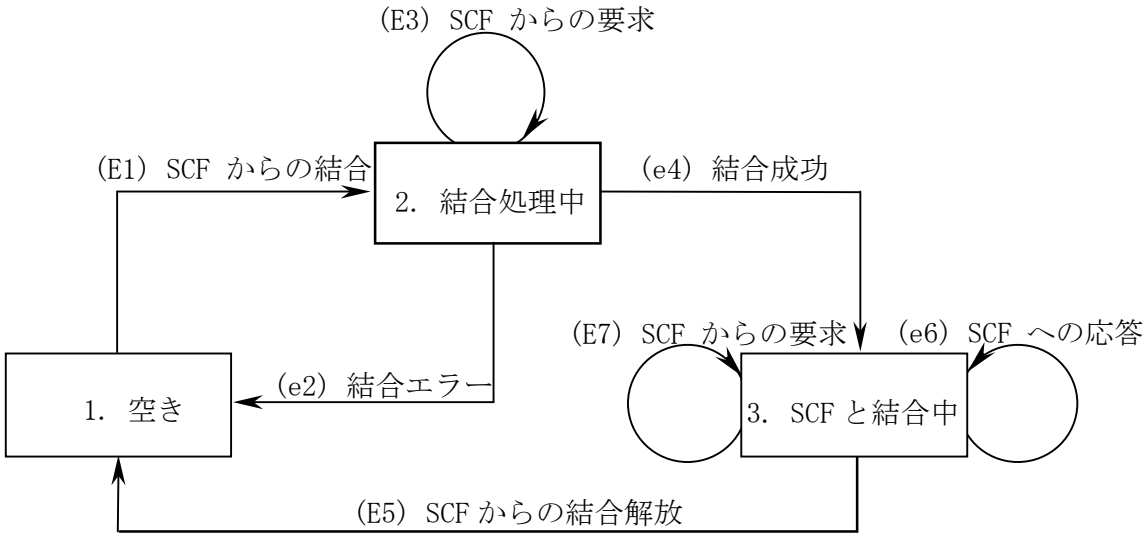


図 3-3-6/JT-Q1218（ITU-T Q.1218）SDSM FSM

各状態は以下の項毎に論じられている。

一つ以上の状態に適用可能な一般的な規則は以下のようである。

任意の状態において、ＳＣＦとの間の対話が終了されると SDF FSM は、呼に割り当てられたどのリソースも解放されていることを保証した後で、空き(Idle)状態に戻る。

3.1.2.4.1.1　状態 1：”空き（Idle）”

この状態において許容されるただ一つのイベントは：

・（E1）SCF からの結合(Bind_From_SCF)である。これは外部イベントであり、ＳＣＦからのディレクトリ結合（DirectoryBind）オペレーションの受信により生ずる。このイベントは本状態から状態 2, 結合処理中(Bind Pending)への遷移を生ずる。

3.1.2.4.1.2　状態 2：”結合処理中(Bind Pending)”

この状態においては、結合要求がＳＣＦから受信されている。受信したＳＤＦは結合の設定の動作と同時にＳＣＦへのアクセスコントロールを実行する（例えば、アクセスに関する認証）。しかし、ディレクトリ結合（DirectoryBind）オペレーションがダミーということもあり得る。この時、アクセスに関する認証は不要である。本状態では 3 つのイベントが生じ得る：
（注：イベントの番号順に整列を行ったため、以下で第 1 版と項目の順序が異なっている部分がある。）

・（e2）結合エラー(Bind_Error):これは内部イベントであり、以前にＳＤＦに対して発行されたディレクトリ結合（DirectoryBind）オペレーションが失敗したことにより生じる。 このイベントは状態 1, 空き（Idle）への遷移を生ずる。ディレクトリ結合（DirectoryBind）オペレーションに対するエラーが起動元ＳＣＦへ返される。

・（E3）SCF からの要求(Request_from_SCF):これは外部イベントであり、ディレクトリ結合（DirectoryBind）オペレーションの結果が決定する前にオペレーションを受信することにより生じる。SDSM は同一状態に留まる。

・（e4）結合成功(Bind_Successful):これは内部イベントであり、以前にＳＤＦに対して発行されたディレクトリ結合（DirectoryBind）オペレーションが正常に完了したことにより生じる。このイベントは状態 3, SCF と結合中(SCF Bound)への遷移を生ずる。

3.1.2.4.1.3　状態 3：”SCF と結合中(SCF Bound)”

この状態において、ＳＤＦに対するＳＣＦのアクセスは認証されており、ＳＣＦからのオペレーションが受け付けられている。ＳＣＦからの要求待ちの他に、この状態でＳＤＦは以前に発行されたオペレーションへの応答を送信することができる。本状態では３つのイベントが生じ得る：

・（E5）SCF からの結合解放(Unbind_from_SCF):これは外部イベントであり、ＳＣＦからディレクトリ結合解放（Directory Unbind）オペレーションを受信することにより生じる。SCF-SDF 間アソシエーションは終了され、全ての関連するリソースは解放される。このイベントは本状態から状態 1, 空き（Idle）への遷移を生ずる。

・（e6）SCF への応答(Responce_to_SCF):これは内部イベントであり、ＳＣＦから発行されたオペレーションの実行が完了する，あるいはＳＣＦに対するリフェラルエラーを生成することにより生じる。この時、応答またはリフェラルがＳＣＦへ返される。SDSM は同一状態に留まる。

・（E7）SCF からの要求(Request_from_SCF):これは外部イベントであり、ＳＣＦからＳＤＦに対しての要求を受信することにより生じる。SDSM は同一状態に留まる。

3.1.2.4.2　ＳＤＦ関連の状態に対するＳＤＦ状態遷移モデル

他のＳＤＦとの相互動作に関しては、ＳＤＦはシャドウイングおよび連鎖オペレーションを実行する。シャドウイングのための SDF/SDF 相互動作に関する状態を 3.1.2.4.2.1 項に示す。また、連鎖のための SDF/SDF 相互動作に関する状態を 3.1.2.4.2.2 項に示す。

3.1.2.4.2.1　シャドウイングに関するＳＤＦ状態遷移モデル

シャドウイング手順に関しては、ＳＤＦはコピー供給側にもコピー消費側にもなり得る。さらに、シャドウイング手順はコピー供給側だけでなく、コピー消費側からも起動され得る。したがって、以下に示すように全部で 4 種類の FSM が存在する。

4 つの SDF FSM をまとめて 1 つの複雑な FSM を定義することもできるが、ＳＤＦの異なる役割を明確に示すため、4 つの FSM を別々に記述する方が良い。

以下の FSM では、DSA シャドウ結合(DSAShadowBind)オペレーションと他の DISP オペレーションを一つの TC メッセージで一緒に送信する可能性を考慮している。

【シャドウ供給側起動による供給側状態モデル】

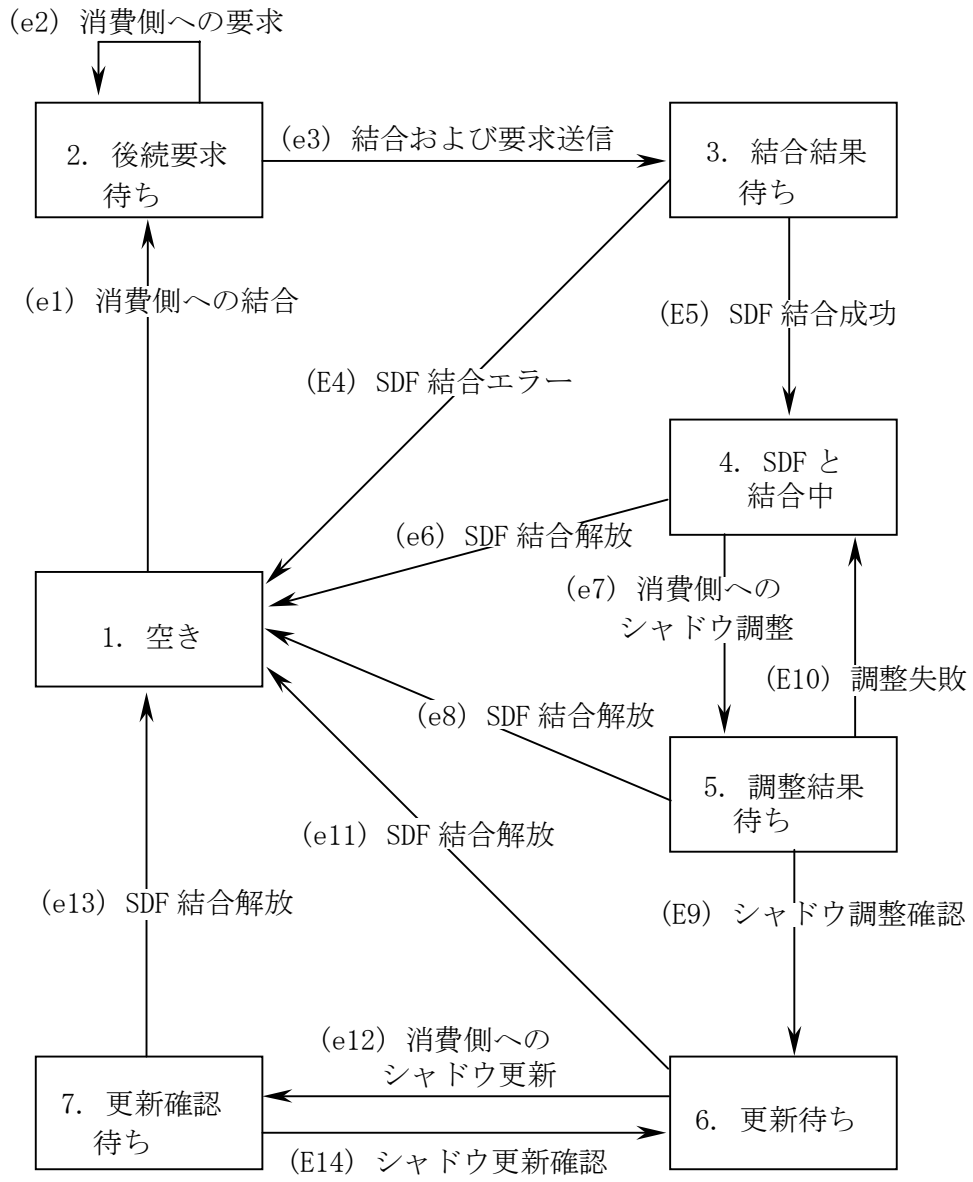


図 3-3-7/JT-Q1218　シャドウ供給側起動による供給側状態モデル

状態 1 : 空き (Idle)

この状態で許容される唯一のイベントは以下である。

・ (e1) 消費側への結合(Bind_to_Consumer):これは内部イベントであり、DSA シャドウ結合(DSAShadowBind)オペレーションを送信することにより生じる。これにより状態 2, 後続要求待ち(Wait for Subsequent Requests)に遷移する。

状態 2 : 後続要求待ち (Wait for Subsequent Requests)

この状態では、DSA シャドウ結合(DSAShadowBind)オペレーションと同一メッセージで消費側に送信するオペレーションを待っている。以下の 2 つのイベントが考えられる。

・（e2）消費側への要求(Request_to_Consumer):これは内部イベントであり、オペレーションを受信することにより生じる。オペレーションはデリミタ（あるいはタイマ満了）を受信するまでバッファリングされる。これにより同一の状態に留まる。

・（e3）結合および要求送信(Send_Bind_with_Requests):これは内部イベントであり、送信すべき最終のオペレーションを受信したことを示すデリミタを受信することにより生じる。デリミタが受信されると、DSA シャドウ結合(DSAShadowBind)オペレーションと（もしあれば）他のオペレーションを含むメッセージが消費側ＳＤＦに対して送信される。これにより状態 3, 結合結果待ち(Wait for Bind Results)に遷移する。

状態 3 : 結合結果待ち（Wait for Bind Result）

この状態では、シャドウ消費側からの DSA シャドウ結合(DSAShadowBind)オペレーションの結果を待っている。以下の 2 つのイベントが考えられる。

・（E4）SDF 結合エラー(SDF_Bind_Error):これは外部イベントであり、消費側に発行した DSA シャドウ結合(DSAShadowBind)オペレーションが失敗したことにより生じる。DSA シャドウ結合エラーが返送される。これにより状態 1, 空き(Idle)に遷移する。

・（E5）SDF 結合成功(SDF_Bind_Success):これは外部イベントであり、消費側に発行した DSA シャドウ結合(DSAShadowBind)オペレーションが成功したことにより生じる。これにより状態 4, SDF と結合中(SDF Bound)に遷移する。

状態 4 : SDF と結合中（SDF Bound）

この状態では、ＳＤＦは消費側への“認証されたアソシエーション”を確立し、シャドウ更新調整(CoordinateShadowUpdate)オペレーションを送信する準備が完了している。以下の 2 つのイベントが考えられる。

・（e6）SDF 結合解放(SDF_Unbind):これは内部イベントであり、ＳＤＦ間に確立した“認証されたアソシエーション”を終了する必要性により生じる（例、ユーザの解放手順）。これにより状態 1, 空き(Idle)に遷移する。

・（e7）消費側へのシャドウ調整(Shadow_Coordinate_to_Consumer):これは内部イベントであり、消費側に対して後に更新するためにシャドウ更新の調整の要求を送信することにより生じる。これにより状態 5, 調整結果待ち(Wait for Coordination Result)に遷移する。

状態 5 : 調整結果待ち（Wait for Coordination Result）

この状態では、供給側ＳＤＦがシャドウ更新調整(CoordinateShadowUpdate)要求を送信し、消費側ＳＤＦからの結果を待っている。以下の 3 つのイベントが考えられる。

・（e8）SDF 結合解放(SDF_Unbind):これは内部イベントであり、ＳＤＦ間に確立した“認証されたアソシエーション”を終了する必要性により生じる（例、ユーザの解放手順）。これにより状態 1, 空き(Idle)に遷移する。

・（E9）シャドウ調整確認(Shadow_Coordinate_Confirmed):これは外部イベントであり、発行したシャドウ更新調整(CoordinateShadowUpdate)オペレーションに対する応答を受信することにより生じる。これにより状態 6, 更新待ち(Wait for Update)に遷移する。

・（E10）調整失敗(Coordinate_Failure):これは外部イベントであり、発行したシャドウ更新調整(CoordinateShadowUpdate)オペレーションに対するエラーを受信することにより生じる。これにより状態 4, SDF と結合中(SDF Bound)に遷移する。

状態 6 : 更新待ち(Wait for Update)

この状態では、供給側ＳＤＦが、発行したシャドウ更新調整要求に対する確認を受信し、消費側にシャドウ更新要求を送信する準備が完了している。以下の 2 つのイベントが考えられる。

・（e11）SDF 結合解放(SDF_Unbind):これは内部イベントであり、ＳＤＦ間に確立した“認証されたアソシエーション”を終了する必要性により生じる（例、ユーザの解放手順）。これにより状態 1, 空き(Idle)に遷移する。

・（e12）消費側へのシャドウ更新(Shadow_Update_to_Consumer):これは内部イベントであり、シャドウ更新する要求を消費側ＳＤＦへ送信することにより生じる。これにより状態 7, 更新確認待ち（Wait for Update Confirmation)に遷移する。

状態 7 : 更新確認待ち（Wait for Update Confirmation）

この状態では、ＳＤＦはシャドウ更新(UpdateShadow)要求を送信し、消費側からの結果を待っている。以下の 2 つのイベントが考えられる。

・（e13）SDF 結合解放(SDF_Unbind):これは内部イベントであり、ＳＤＦ間に確立した“認証されたアソシエーション”を終了する必要性により生じる（例、ユーザの解放手順）。これにより状態 1, 空き(Idle)に遷移する。

・（E14）シャドウ更新確認(Shadow_Update_Confirmed):これは外部イベントであり、発行したシャドウ更新(UpdateShadow)オペレーションに対する応答を受信することにより生じる。これにより状態 6, 更新待ち(Wait_for_Update)に遷移する。

【シャドウ消費側起動による供給側状態モデル】

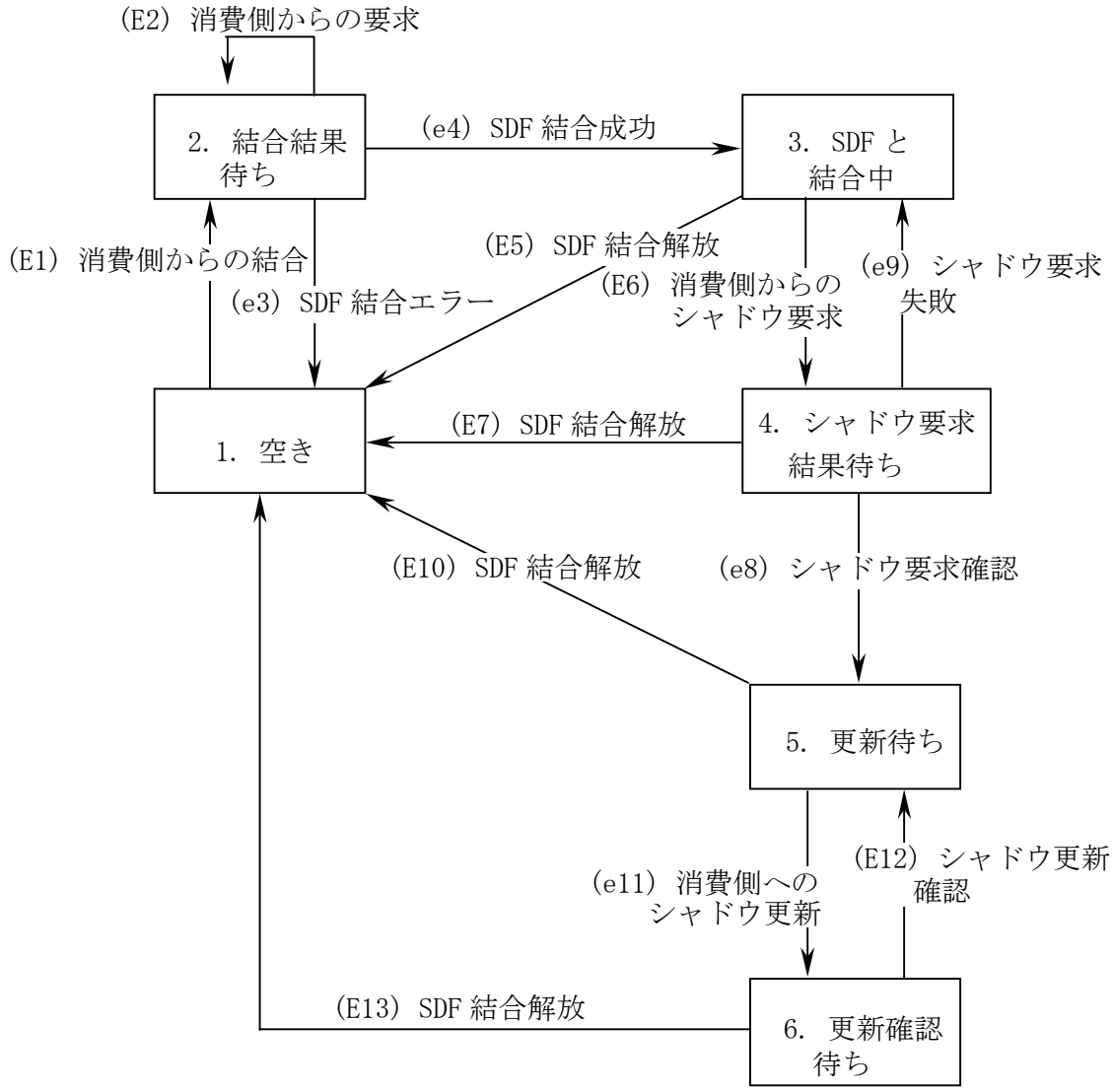


図 3-3-8/JT-Q1218　シャドウ消費側起動による供給側状態モデル

状態 1 : 空き（Idle）

この状態で許容される唯一のイベントは以下である。

・（E1）消費側からの結合(Bind_from_Consumer):これは外部イベントであり、DSA シャドウ結合(DSAShadowBind)オペレーションを受信することにより生じる。これにより状態 2, 結合結果待ち(Wait for Bind Result)に遷移する。

状態 2 : 結合結果待ち（Wait for Bind Result）

この状態では、 DSA シャドウ結合(DSAShadowBind)オペレーションを実行している。以下の 3 つのイベントが考えられる。

・（E2）消費側からの要求(Request_from_Consumer):これは外部イベントであり、DSA シャドウ結合(DSAShadowBind)オペレーションの結果が判明する前にオペレーションを受信することにより生じる。これにより同一の状態に留まる。

・（e3）SDF 結合エラー(SDF_Bind_Error):これは内部イベントであり、消費側から発行された DSA シャドウ結合(DSAShadowBind)オペレーションが失敗したことにより生じる。DSA シャドウ結合エラーが返送される。これにより状態 1, 空き(Idle)に遷移する。

・（e4）SDF 結合成功(SDF_Bind_Success):これは内部イベントであり、消費側から発行された DSA シャドウ結合(DSAShadowBind)オペレーションが成功したことにより生じる。これにより状態 3, SDF と結合中(SDF Bound)に遷移する。

状態 3 : SDF と結合中（SDF Bound）

この状態では、供給側ＳＤＦは消費側ＳＤＦからのシャドウ更新要求(RequestShadow Update)オペレーションを待っている。以下の 2 つのイベントが考えられる。

・（E5）SDF 結合解放(SDF_Unbind):これは外部イベントであり、ＳＤＦ間に確立した“認証されたアソシエーション”を終了する必要性により生じる（例、ユーザの解放手順）。これにより状態 1, 空き(Idle)に遷移する。

・（E6）消費側からのシャドウ要求(Request_for_Shadow_from_Consumer):これは外部イベントであり、消費側からのシャドウ更新要求(RequestShadowUpdate)オペレーションを受信することにより生じる。これにより状態 4, シャドウ要求結果待ち(Wait for RequestShadow Result)に遷移する。

状態 4 : シャドウ要求結果待ち（Wait for RequestShadow Result）

この状態では、供給側ＳＤＦは消費側ＳＤＦからのシャドウ更新要求(RequestShadow Update)オペレーションを受信している。以下の 3 つのイベントが考えられる。

・（E7）SDF 結合解放(SDF_Unbind):これは外部イベントであり、ＳＤＦ間に確立した“認証されたアソシエーション”を終了する必要性により生じる（例、ユーザの解放手順）。これにより状態 1, 空き(Idle)に遷移する。
・（e8）シャドウ要求確認(Shadow_Request_Confirmed):これは内部イベントであり、受信したシャドウ更新要求(RequestShadowUpdate)オペレーションに対する応答を送信することにより生じる。これにより状態 5, 更新待ち(Wait for Update)に遷移する。

・（e9）シャドウ要求失敗(Failed_Shadow_Request):これは内部イベントであり、受信したシャドウ更新要求(RequestShadowUpdate)オペレーションに対するエラーを送信することにより生じる。これにより状態 3, SDF と結合中(SDF Bound)に遷移する。

状態 5 : 更新待ち(Wait for Update)

この状態では、供給側ＳＤＦがすでに受信したシャドウ更新要求(RequestShadowUpdate)オペレーションに対する応答を送信し、消費側に対してシャドウ更新(UpdateShadow)オペレーションを送信する準備が完了している。以下の 2 つのイベントが考えられる。

・（E10）SDF 結合解放(SDF_Unbind):これは外部イベントであり、ＳＤＦ間に確立した“認証されたアソシエーション”を終了する必要性により生じる（例、ユーザの解放手順）。これにより状態 1, 空き(Idle)に遷移する。

・（e11）消費側へのシャドウ更新(Shadow_Update_to_Consumer):これは内部イベントであり、シャドウ更新の要求を消費側ＳＤＦへ送信することにより生じる。これにより状態 6, 更新確認待ち(Wait_for_Update_ Confirmation)に遷移する。

状態 6 : 更新確認待ち（Wait for Update Confirmation）

この状態では、供給側ＳＤＦはシャドウ更新(UpdateShadow)要求を送信し、消費側ＳＤＦからの応答を待っている。以下の 2 つのイベントが考えられる。

・（E12）シャドウ更新確認(Shadow_Update_Confirmed):これは外部イベントであり、発行したシャドウ更新(UpdateShadow)オペレーションに対する応答を受信することにより生じる。これにより状態 5, 更新待ち(Wait for Update)に遷移する。

・（E13）SDF 結合解放(SDF_Unbind):これは外部イベントであり、ＳＤＦ間に確立した“認証されたアソシエーション”を終了する必要性により生じる（例、ユーザの解放手順）。これにより状態 1, 空き(Idle)に遷移する。

【シャドウ供給側起動による消費側状態モデル】

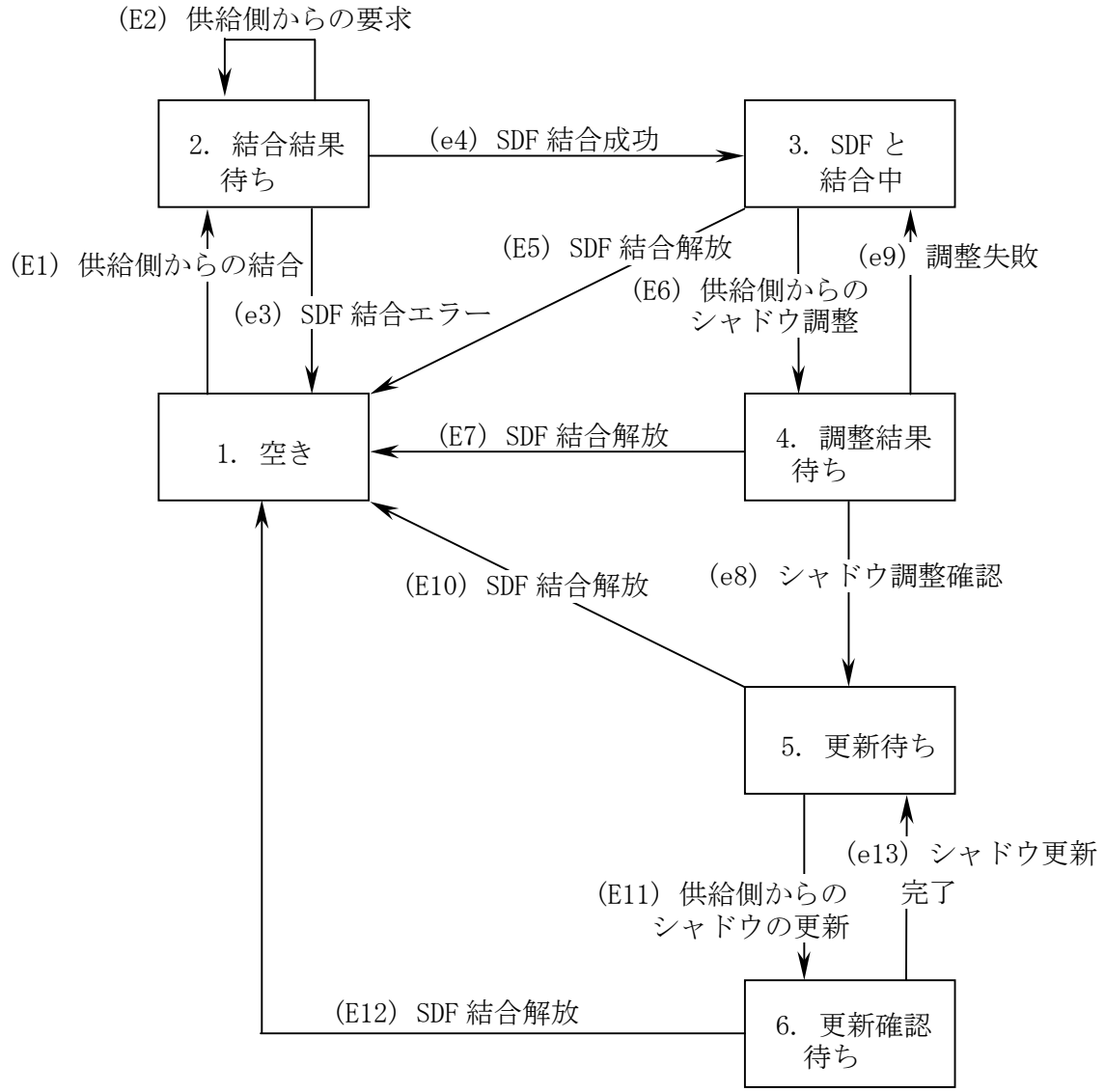


図 3-3-9/JT-Q1218　シャドウ供給側起動による消費側状態モデル

状態 1 : 空き（Idle）

この状態で許容される唯一のイベントは以下である。

・（E1）供給側からの結合(Bind_from_Supplier):これは外部イベントであり、DSA シャドウ結合(DSAShadowBind)オペレーションを受信することにより生じる。これにより状態 2, 結合結果待ち(Wait for Bind Result)に遷移する。

状態 2 : 結合結果待ち（Wait for Bind Result）

この状態では、消費側ＳＤＦは DSA シャドウ結合(DSAShadowBind)オペレーションを受信し、応答しようとしている。以下の 3 つのイベントが考えられる。

・（E2）供給側からの要求(Request_from_Supplier):これは外部イベントであり、DSA シャドウ結合(DSAShadowBind)オペレーションの結果が判明する前にオペレーションを受信することにより生じる。これにより同一の状態に留まる。

・（e3）SDF 結合エラー(SDF_Bind_Error):これは内部イベントであり、供給側から発行された DSA シャドウ結合(DSAShadowBind)オペレーションが失敗したことにより生じる。DSA シャドウ結合エラーが返送される。これにより状態 1, 空き(Idle)に遷移する。

・（e4）SDF 結合成功(SDF_Bind_Success):これは内部イベントであり、供給側から発行された DSA シャドウ結合(DSAShadowBind)オペレーションが成功したことにより生じる。これにより状態 3, SDF と結合中(SDF Bound)に遷移する。

状態 3 : SDF と結合中（SDF Bound）

この状態では、消費側ＳＤＦは供給側ＳＤＦからのシャドウ更新調整(Coordinate ShadowUpdate)オペレーションを待っている。以下の 2 つのイベントが考えられる。

・（E5）SDF 結合解放(SDF_Unbind):これは外部イベントであり、ＳＤＦ間に確立した“認証されたアソシエーション”を終了する必要性により生じる（例、ユーザの解放手順）。これにより状態 1, 空き(Idle)に遷移する。

・（E6）供給側からのシャドウ調整(Shadow_Coordinate_from_Supplier):これは外部イベントであり、供給側からのシャドウ更新調整(CoordinateShadowUpdate)オペレーションを受信することにより生じる。これにより状態 4, 調整結果待ち(Wait for Coordination Result)に遷移する。

状態 4 : 調整結果待ち（Wait for Coordination Result）

この状態では、消費側ＳＤＦが供給側ＳＤＦからのシャドウ更新調整(Coordinate ShadowUpdate)オペレーションを受信し、それを処理している。以下の 3 つのイベントが考えられる。

・（E7）SDF 結合解放(SDF_Unbind):これは外部イベントであり、ＳＤＦ間に確立した“認証されたアソシエーション”を終了する必要性により生じる（例、ユーザの解放手順）。これにより状態 1, 空き(Idle)に遷移する。

・（e8）シャドウ調整確認(Shadow_Coordinate_ Confirmed):これは内部イベントであり、供給側ＳＤＦから受信したシャドウ更新調整(CoordinateShadow Update)オペレーションが成功したことにより生じる。これにより状態 5, 更新待ち(Wait for Update)に遷移する。

・（e9）調整失敗(Coordinate_Failure):これは内部イベントであり、受信したシャドウ更新調整(CoordinateShadowUpdate)オペレーションに対するエラーを送信することにより生じる。これにより状態 3, SDF と結合中(SDF Bound)に遷移する。

状態 5 : 更新待ち（Wait for Update）

この状態では、消費側ＳＤＦはシャドウ更新(UpdateShadow)オペレーションを待っている。以下の 2 つのイベントが考えられる。

・（E10）SDF 結合解放(SDF_Unbind):これは外部イベントであり、ＳＤＦ間に確立した“認証されたアソシエーション”を終了する必要性により生じる（例、ユーザの解放手順）。これにより状態 1, 空き(Idle)に遷移する。

・（E11）供給側からのシャドウの更新(Update_Shadow_from_Supplier):これは外部イベントであり、供給側ＳＤＦからのシャドウ更新(UpdateShadow)オペレーションを受信することにより生じる。これにより状態 6, 更新確認待ち(Wait for Update Confirmation)に遷移する。

状態 6 : 更新確認待ち(Wait for Update Confirmation)

この状態では、消費側ＳＤＦが供給側ＳＤＦからのシャドウ更新(UpdateShadow)オペレーションを受信し、コピーの更新を処理している。以下の 2 つのイベントが考えられる。

・（E12）SDF 結合解放(SDF_Unbind):これは外部イベントであり、ＳＤＦ間に確立した“認証されたアソシエーション”を終了する必要性により生じる（例、ユーザの解放手順）。これにより状態 1, 空き(Idle)に遷移する。

・（e13）シャドウ更新完了(Shadow_Updated):これは内部イベントであり、シャドウ更新(UpdateShadow)オペレーションを正常に完了し、応答を送信することにより生じる。これにより状態 5, 更新待ち(Wait for Update)に遷移する。

【シャドウ消費側起動による消費側状態モデル】

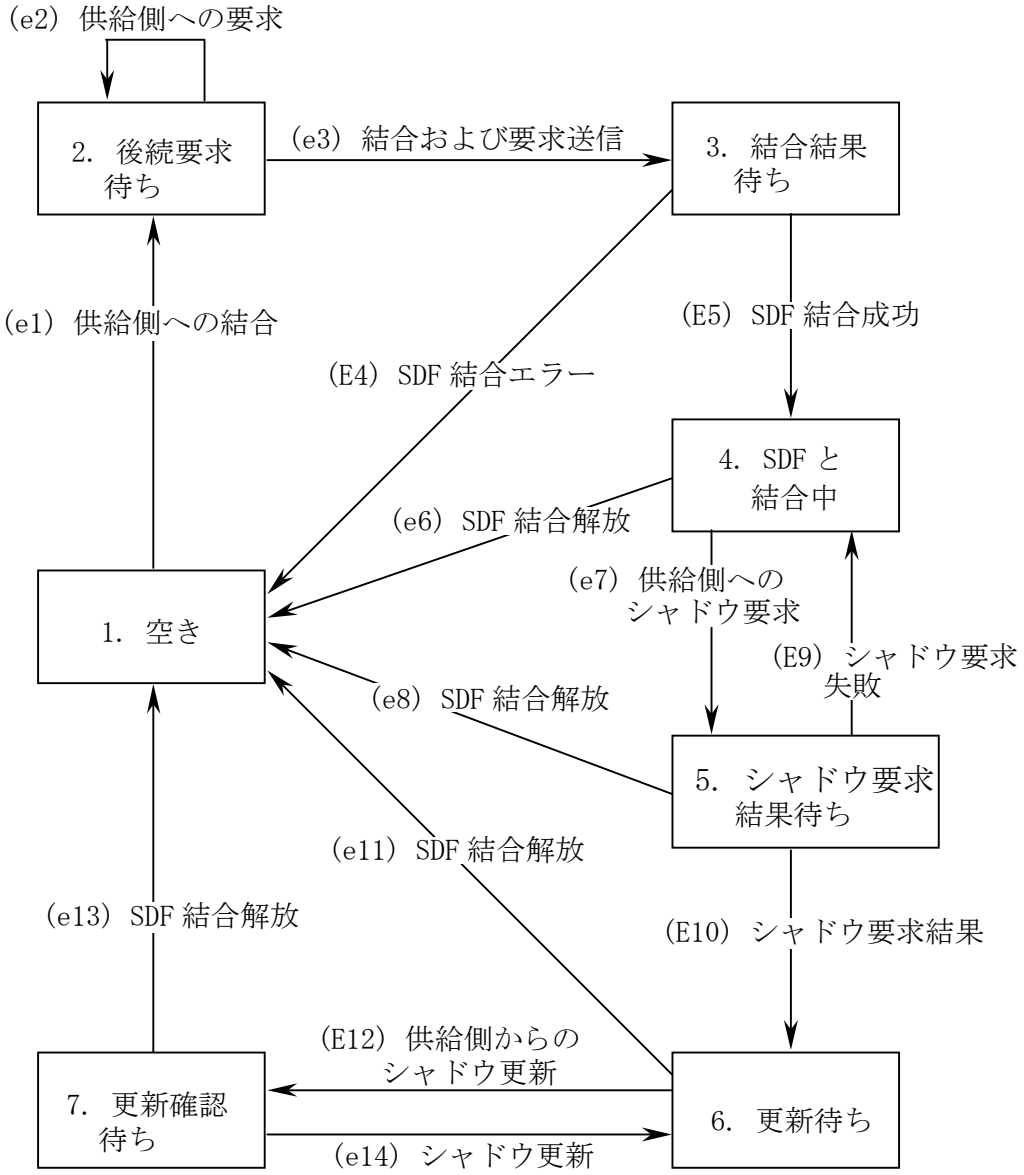


図 3-3-10/JT-Q1218　シャドウ消費側起動による消費側状態モデル

状態 1 : 空き（Idle）

この状態で許容される唯一のイベントは以下である。

・(e1)供給側への結合(Bind_to_Supplier):これは内部イベントであり、DSA シャドウ結合(DSAShadowBind)オペレーションを送信することにより生じる。これにより状態 2, 後続要求待ち(Wait for Subsequent Requests)に遷移する。

状態 2 : 後続要求待ち(Wait for Subsequent Requests)

この状態では、DSA シャドウ結合(DSAShadowBind)オペレーションと同一メッセージで供給側に送信するオペレーションを待っている。以下の 2 つのイベントが考えられる。

・（e2）供給側への要求(Request_to_Supplier):これは内部イベントであり、オペレーションを受信することにより生じる。オペレーションはデリミタ（あるいはタイマ満了）を受信するまでバッファリングされる。これにより同一の状態に留まる。

・（e3）結合および要求送信(Send_Bind_with_Requests):これは内部イベントであり、送信すべき最終のオペレーションを受信したことを示すデリミタを受信することにより生じる。デリミタが受信されると、DSA シャドウ結合(DSAShadowBind)オペレーションと（もしあれば）他のオペレーションを含むメッセージが供給側ＳＤＦに対して送信される。これにより状態 3, 結合結果待ち(Wait for Bind Results)に遷移する。

状態 3 : 結合結果待ち（Wait for Bind Result）

この状態では、 供給側ＳＤＦからの DSA シャドウ結合(DSAShadowBind)の結果を待っている。以下の 2 つのイベントが考えられる。

・（E4）SDF 結合エラー(SDF_Bind_Error):これは外部イベントであり、供給側へ発行された DSA シャドウ結合(DSAShadowBind)オペレーションが失敗したことにより生じる。DSA シャドウ結合エラーが返送される。これにより状態 1, 空き(Idle)に遷移する。

・（E5）SDF 結合成功(SDF_Bind_Success):これは外部イベントであり、供給側へ発行された DSA シャドウ結合(DSAShadowBind)オペレーションが成功したことにより生じる。これにより状態 4, SDF と結合中(SDF Bound)に遷移する。

状態 4 : SDF と結合中（SDF Bound）

この状態では、消費側ＳＤＦは供給側ＳＤＦに対してシャドウ更新要求(Request ShadowUpdate)オペレーションを送信する準備をしている。以下の 2 つのイベントが考えられる。

・（e6）SDF 結合解放(SDF_Unbind):これは内部イベントであり、ＳＤＦ間に確立した“認証されたアソシエーション”を終了する必要性により生じる（例、ユーザの解放手順）。これにより状態 1, 空き(Idle)に遷移する。

・（e7）供給側へのシャドウ要求(Shadow_Request_to_Supplier):これは内部イベントであり、供給側に対してシャドウ更新要求(RequestShadowUpdate)オペレーションを送信することにより生じる。これにより状態 5, シャドウ要求結果待ち(Wait for RequestShadow Result)に遷移する。

状態 5 : シャドウ要求結果待ち（Wait for RequestShadow Result）

この状態では、消費側ＳＤＦはシャドウ更新要求(RequestShadowUpdate)オペレーションを送信し、供給側ＳＤＦからの応答を待っている。以下の 3 つのイベントが考えられる。

・（e8）SDF 結合解放(SDF_Unbind):これは内部イベントであり、ＳＤＦ間に確立した“認証されたアソシエーション”を終了する必要性により生じる（例、ユーザの解放手順）。これにより状態 1, 空き(Idle)に遷移する。

・（E9）シャドウ要求失敗(Failed_Shadow_Request):これは外部イベントであり、発行したシャドウ更新要求(RequestShadowUpdate)オペレーションに対するエラーを受信することにより生じる。これにより状態 4, SDF と結合中(SDF Bound)に遷移する。

・（E10）シャドウ要求結果(Request_Shadow_Result):これは外部イベントであり、発行したシャドウ更新要求(RequestShadowUpdate)オペレーションに対する応答を受信することにより生じる。これにより状態 6, 更新確認待ち(Wait for Update Confirmation)に遷移する。

状態 6 : 更新待ち(Wait for Update)

この状態では、消費側ＳＤＦがシャドウ更新要求(RequestShadowUpdate)の結果を受信し、供給側ＳＤＦからのシャドウ更新(UpdateShadow)オペレーションを待っている。以下の 2 つのイベントが考えられる。

・（e11）SDF 結合解放(SDF_Unbind):これは内部イベントであり、ＳＤＦ間に確立した“認証されたアソシエーション”を終了する必要性により生じる（例、ユーザの解放手順）。これにより状態 1, 空き(Idle)に遷移する。

・（E12）供給側からのシャドウ更新(Shadow_Update_from_Supplier):これは外部イベントであり、供給側ＳＤＦから発行されたシャドウ更新(UpdateShadow)オペレーションを受信することにより生じる。これにより状態 7, 更新確認待ち(Wait for Update Confirmation)に遷移する。

状態 7 : 更新確認待ち（Wait for Update Confirmation）

この状態では、消費側ＳＤＦは供給側ＳＤＦからのシャドウ更新(UpdateShadow)オペレーションを受信し、コピーの更新を処理している。以下の 2 つのイベントが考えられる。

・（e13）SDF 結合解放(SDF_Unbind):これは内部イベントであり、ＳＤＦ間に確立した“認証されたアソシエーション”を終了する必要性により生じる（例、ユーザの解放手順）。これにより状態 1, 空き(Idle)に遷移する。

・（e14）シャドウ更新完了(Shadow_Updated):これは内部イベントであり、シャドウ更新(UpdateShadow)オペレーションを正常に完了し、応答を送信することにより生じる。これにより状態 6, 更新待ち(Wait for Update)に遷移する。

3.1.2.4.2.2　連鎖に関するＳＤＦ状態遷移モデル

連鎖手順に関しては、ＳＤＦは連鎖の起動側にも終端側にもなり得る。したがって、以下に示すように 2 つの FSM が存在する。

以下の FSM では、DSA 結合(DSABind)オペレーションと他の DSP オペレーションを一つの TC メッセージで一緒に送信する可能性を考慮している。

【連鎖起動に関するＳＤＦ状態遷移モデル（SDSM-ChI）】

ＳＤＦが連鎖の起動側になった場合に、他ＳＤＦとの相互動作を表すための有限状態機械を図 3-3-11/JT-Q1218 に示す。

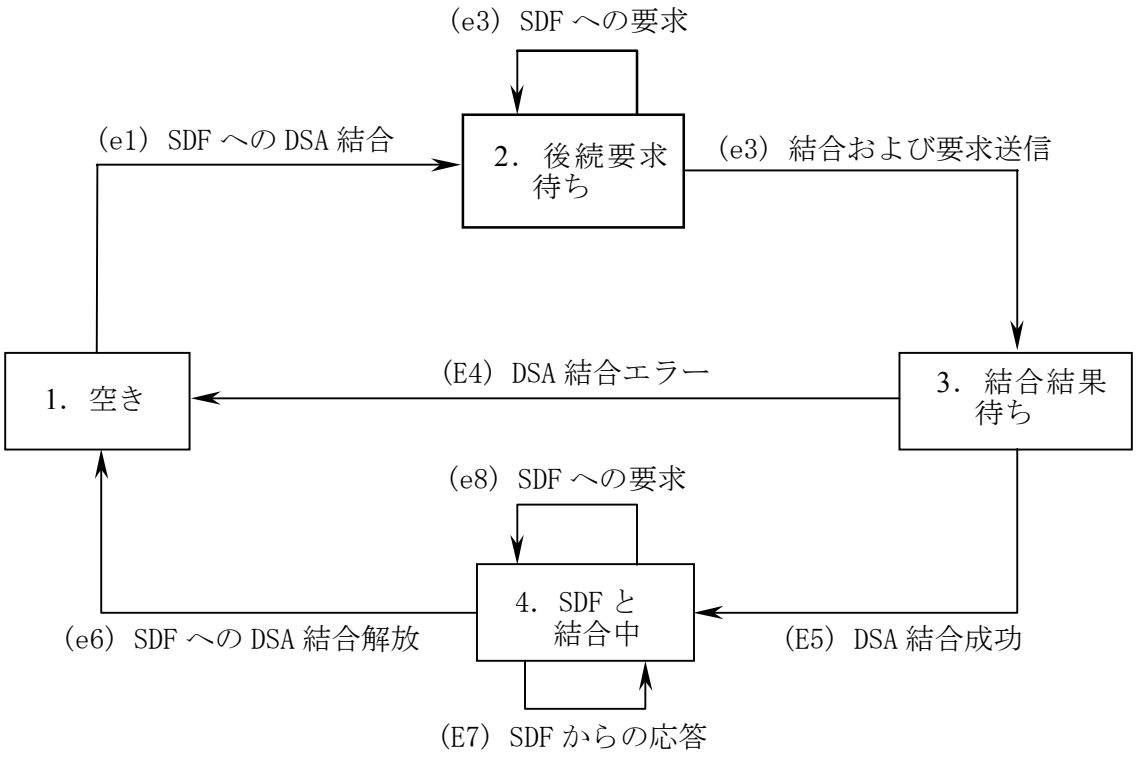


図 3-3-11/JT-Q1218　SDF-SDF 連鎖起動側の有限状態機械（SDSM-Ch Ⅰ）

状態 1：空き（Idle）

この状態で許容される唯一のイベントは以下である。

・（e1）SDF への DSA 結合(DSABind_to_SDF):これは内部イベントであり、SDSM-ChT に対して DSA 結合(DSABind)オペレーションを送信することにより生じる。これにより状態 2, 後続要求待ち(Wait for Subsequent Requests)に遷移する。

状態 2 : 後続要求待ち(Wait for Subsequent Requests)

この状態では、DSA 結合(DSABind)オペレーションと同一メッセージで SDSM-ChT に送信するオペレーションを待っている。以下の 2 つのイベントが考えられる。

・（e2）SDF への要求(Request_to_SDF):これは内部イベントであり、オペレーションを受信することにより生じる。オペレーションはデリミタ（あるいはタイマ満了）を受信するまでバッファリングされる。これにより同一の状態に留まる。
・（e3）結合および要求送信(Send_Bind_with_Requests):これは内部イベントであり、送信すべき最終のオペレーションを受信したことを示すデリミタを受信することにより生じる。デリミタが受信されると、DSA 結合(DSABind)オペレーションと（もしあれば）他のオペレーションを含むメッセージが SDSM-ChT に対して送信される。これにより状態 3, 結合結果待ち(Wait for Bind Results)に遷移する。

状態 3 : 結合結果待ち(Wait for Bind Results)

この状態では、DSA 結合(DSABind)要求が SDSM-ChT に対して送信されており、SDSM-ChT では DSA 結合(DSABind)オペレーションに関するＳＤＦアクセス制御手順（例、アクセス認証）を実行している。以下の 2 つのイベントが考えられる。

・（E4）DSA 結合エラー(DSABind_Error):これは外部イベントであり、SDSM-ChT に送信した DSA 結合(DSABind)オペレーションが失敗したことにより生じる。これにより状態 1, 空き(Idle)に遷移する。

・（E5）DSA 結合成功(DSABind_Successful): これは外部イベントであり、SDSM-ChT に送信した DSA 結合(DSABind)オペレーションに対する確認を受信することにより生じる。これにより状態 4, SDF と結合中(SDF Bound)に遷移する。

状態 4 : SDF と結合中（SDF Bound）

この状態では、SDSM-ChI から SDSM-ChT へのアクセスが認証され、SDSM-ChT への連鎖オペレーションが送信可能となり、連鎖オペレーションに対する SDSM-ChT からの結果が受信される。ＣＳ－１（改）では、直接的な知識参照の使用を前提とする。したがって、SDSM-ChT が連鎖要求をさらに連鎖することはない。以下の 3 つのイベントが考えられる。

・（e6）SDF への DSA 結合解放(DSAUnbind_to_SDF):これは内部イベントであり、SDSM-ChT に対して DSA 結合解放(DSAUnbind)オペレーションを送信することにより生じる。SDF-SDF 間のアソシエーションが終了し、関連するすべてのリソースが解放される。これにより状態 1, 空き(Idle)に遷移する。

・（E7）SDF からの応答(Response_from_SDF):これは外部イベントであり、SDSM-ChI により送信されたオペレーションに対する結果を受信する、あるいは SDSM-ChT からのリフェラルを受信することにより生じる。これにより同一の状態に留まる。

・（e8）SDF への要求(Request_to_SDF):これは内部イベントであり、SDSM-ChT に対して連鎖オペレーションを送信することにより生じる。これにより同一の状態に留まる。

【連鎖終端に関するＳＤＦ状態遷移モデル（SDSM-ChT）】

ＳＤＦが連鎖の終端側になった場合に、他ＳＤＦとの相互動作を表すための有限状態機械を図 3-3-12/JT-Q1218 に示す。

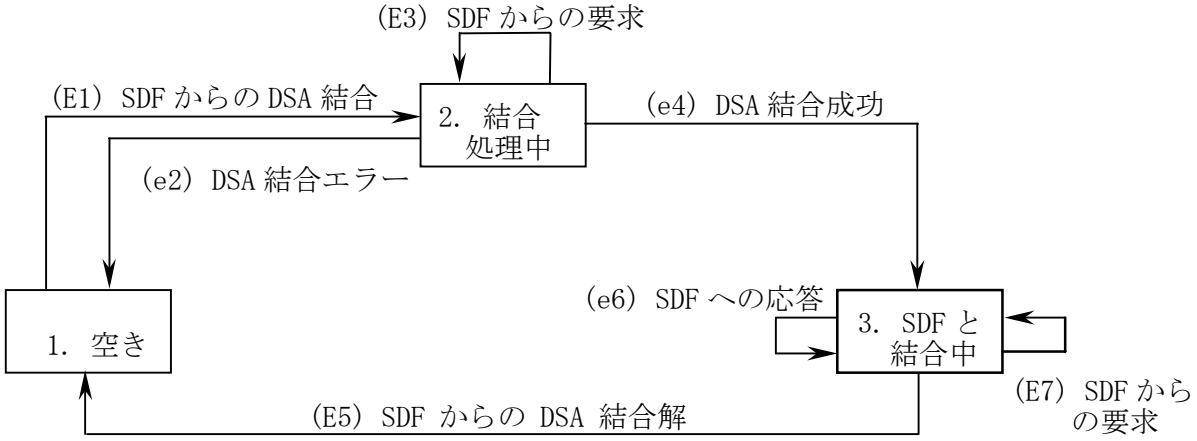


図 3-3-12/JT-Q1218　SDF-SDF 連鎖終端側の有限状態機械（SDSM-ChT）

状態 1 : 空き（Idle）

この状態で許容される唯一のイベントは以下である。

・（E1）SDF からの DSA 結合(DSABind_from_SDF):これは外部イベントであり、SDSM-ChI からの DSA 結合(DSABind)オペレーションを受信することにより生じる。これにより状態 2, 結合処理中(Bind Pending)に遷移する。

状態 2 : 結合処理中（Bind Pending）

この状態では、DSA 結合(DSABind)要求を SDSM-ChI から受信しており、SDSM-ChT では DSA 結合(DSABind)オペレーションに関するＳＤＦアクセス制御手順（例、アクセス認証）を実行している。以下の 2 つのイベントが考えられる。

・（e2）DSA 結合エラー(DSABind_Error):これは内部イベントであり、SDSM-ChT に送信された DSA 結合(DSABind)オペレーションが失敗したことにより生じる。これにより状態 1, 空き(Idle)に遷移し、結合エラーが SDSM-ChI に返送される。

・（E3）SDF からの要求(Request_from_SDF):これは外部イベントであり、DSA 結合(DSABind)オペレーションの結果が判明する前にオペレーションを受信することにより生じる。これにより同一の状態に留まる。

・（e4）DSA 結合成功(DSABind_Successful): これは内部イベントであり、SDSM-ChT に送信された DSA 結合(DSABind)オペレーションが成功することにより生じる。これにより状態 3, SDF と結合中(SDF Bound)に遷移する。

状態 3 : SDF と結合中（SDF Bound）

この状態では、SDSM-ChI から SDSM-ChT へのアクセスが認証され、SDSM-ChI から送信される連鎖オペレーションが許容される。SDSM-ChI からの要求を待つことに加え、SDSM-ChT は受信したオペレーションに対する応答を本状態で送信できる。ＣＳ－１（改）では、直接的な知識参照の使用を前提とする。したがって、SDSM-ChT が連鎖要求をさらに連鎖することはない。以下の 3 つのイベントが考えられる。

・（E5）SDF からの DSA 結合解放(DSAUnbind_from_SDF):これは外部イベントであり、SDSM-ChI からの DSA 結合解放(DSAUnbind)オペレーションを受信することにより生じる。SDF-SDF 間のアソシエーションが終了し、関連するすべてのリソースが解放される。これにより状態 1, 空き(Idle)に遷移する。

・（e6）SDF への応答(Response_to_SDF):これは内部イベントであり、SDSM-ChI から受信したオペレーションが完了する、あるいは SDSM-ChI へのリフェラルを生成することにより生じる。応答あるいはリフェラルが SDSM-ChI に送信される。これにより同一の状態に留まる。

・（E7）SDF からの要求(Request_from_SDF):これは外部イベントであり、SDSM-ChI からの要求を受信することにより生じる。これにより同一の状態に留まる。

3.2 エラー手順

本節は、ＩＮＡＰに対する一般的なエラー手順を定義する。

3.2.1 オペレーション関連エラー手順

以下の節では、オペレーションに関連するエラーに対する一般的なエラー手順を定義する。エラーはオペレーションエラーとして第３編2章に定義される。

個々のオペレーションに対する固有の手順を有するエラーは第３編3.3章に関連するオペレーションの詳細手順とともに示される。

3.2.1.1 属性エラー(AttributeError)

3.2.1.1.1 一般記述

3.2.1.1.1.1 エラー記述

このエラーは、属性関連の問題を報告するためにＳＤＦからＳＣＦあるいはＳＤＦに送信される。属性エラーが送信される条件は、ITU-T勧告X.511(1993)12.4章に定義されている。

3.2.1.1.1.2 アーギュメント記述

属性エラーのパラメータ及び問題コードは、ITU-T勧告X.511(1993)12.4章に定義されている。

3.2.1.1.2　ＳＣＦ-->ＳＤＦオペレーション

探索(Search)

エントリ更新(ModifyEntry)

エントリ追加(AddEntry)

起動側エンティティ(ＳＣＦ)での手順

A) オペレーション送信

前条件：SCSM 状態　　4　SDF と結合中　あるいは　2　後続要求待ち

後条件：SCSM 状態　　4　SDF と結合中　あるいは　2　後続要求待ち

B) エラー受信

前条件：SCSM 状態　　4　SDF と結合中

後条件：SCSM 状態　　4　SDF と結合中

エラー手順はサービス論理に依存する。ＳＣＦが要求を変更することが可能であれば、別のＳＤＦへの問い合わせを行うことが可能である。そうでなければサービス処理は終了すべきである。

応答側エンティティ(ＳＤＦ)での手順

A) オペレーション受信

前条件：SDSM 状態　　3　SCF と結合中

後条件：SDSM 状態　　3　SCF と結合中

B) エラー返送

前条件：SDSM 状態　　3　SCF と結合中

後条件：SDSM 状態　　3　SCF と結合中

ＳＤＦはオペレーションを実行できないため、ＳＣＦに属性エラーを返送する。エラーを返送した後、行われるエラー手順はない。

3.2.1.1.3　ＳＤＦ-->ＳＤＦオペレーション

連鎖探索(ChainedSearch)

連鎖エントリ更新(ChainedModifyEntry)

連鎖エントリ追加(ChainedAddEntry)

起動側エンティティ(ＳＤＦ)での手順

A) オペレーション送信

前条件：SDSM-ChI状態　　4　SDFと結合中　あるいは　2　後続要求待ち

後条件：SDSM-ChI状態　　4　SDFと結合中　あるいは　2　後続要求待ち

B) エラー受信

前条件：SDSM-ChI状態　　4　SDFと結合中

後条件：SDSM-ChI状態　　4　SDFと結合中

本エラーはＳＣＦへ通知される。

応答側エンティティ(ＳＤＦ)での手順

A) オペレーション受信

前条件：SDSM-ChT状態　3　SDFと結合中

後条件：SDSM-ChT状態　3　SDFと結合中

B) エラー返送

前条件：SDSM-ChT状態　3　SDFと結合中

後条件：SDSM-ChT状態　3　SDFと結合中

ＳＤＦはオペレーションを実行できないため、ＳＤＦに属性エラーを返送する。エラーを返送した後、行われるエラー手順はない。

3.2.1.2　ネームエラー(NameError)

3.2.1.2.1　一般記述

3.2.1.2.1.1　エラー記述

このエラーは、オブジェクトのネームに関連した問題を報告するためにＳＤＦからＳＣＦあるいはＳＤＦに送信される。ネームエラーが送信される条件は、ITU-T勧告X.511(1993)12.5章に定義されている。

3.2.1.2.1.2　アーギュメント記述

ネームエラーのパラメータ及び問題コードは、ITU-T勧告X.511(1993)12.5章に定義されている。

3.2.1.2.2　ＳＣＦ-->ＳＤＦオペレーション

探索(Search)

エントリ更新(ModifyEntry)

エントリ追加(AddEntry)

エントリ削除(RemoveEntry)

起動側エンティティ(ＳＣＦ)での手順

A) オペレーション送信

前条件：SCSM 状態　2　後続要求待ち　あるいは　4　SDF と結合中

後条件：SCSM 状態　2　後続要求待ち　あるいは　4　SDF と結合中

B) エラー受信

前条件：SCSM 状態　4　SDF と結合中

後条件：SCSM 状態　4　SDF と結合中

エラー手順はサービス論理に依存する。もし、ＳＣＦが要求を変更することが可能であれば、別のＳＤＦへの問い合わせを行うことが可能である。そうでなければサービス処理は終了すべきである。

応答側エンティティ(ＳＤＦ)での手順

A) オペレーション受信

前条件：SDSM 状態　3　SCF と結合中

後条件：SDSM 状態　3　SCF と結合中

B) エラー返送

前条件：SDSM 状態　3　SCF と結合中

後条件：SDSM 状態　3　SCF と結合中

ＳＤＦはオペレーションを実行できないため、ＳＣＦにネームエラーを返送する。エラーを返送した後、行われるエラー手順はない。

3.2.1.2.3　ＳＤＦ-->ＳＤＦオペレーション

連鎖探索(ChainedSearch)

連鎖エントリ更新(ChainedModifyEntry)

連鎖エントリ追加(ChainedAddEntry)

連鎖エントリ削除(ChainedRemoveEntry)

起動側エンティティ(ＳＤＦ)での手順

A) オペレーション送信

前条件：SDSM-ChI状態　　4　SDFと結合中　あるいは　2　後続要求待ち

後条件：SDSM-ChI状態　　4　SDFと結合中　あるいは　2　後続要求待ち

B) エラー受信

前条件：SDSM-ChI状態　　4　SDFと結合中

後条件：SDSM-ChI状態　　4　SDFと結合中

本エラーはＳＣＦへ通知される。

応答側エンティティ(SDF)での手順

A) オペレーション受信

前条件：SDSM-ChT状態　　3　SDFと結合中

後条件：SDSM-ChT状態　　3　SDFと結合中

B) エラー返送

前条件：SDSM-ChT状態　　3　SDFと結合中

後条件：SDSM-ChT状態　　3　SDFと結合中

ＳＤＦはオペレーションを実行できないため、ＳＣＦにネームエラーを返送する。エラーを返送した後、行われるエラー手順はない。

3.2.1.3　リフェラル(Referral)

3.2.1.3.1　一般記述

3.2.1.3.1.1　エラー記述

ITU-T勧告X.511 12.6 Referral参照。

3.2.1.3.1.2　アーギュメント記述

ITU-T勧告X.511 12.6 ReferralにおけるASN.1記述参照。

3.2.1.3.2　ＳＣＦ-->ＳＤＦオペレーション

探索(Search)

エントリ更新(ModifyEntry)

エントリ追加(AddEntry)

エントリ削除(RemoveEntry)

起動側エンティティ(ＳＣＦ)での手順

A) オペレーション送信

前条件：SCSM状態　2　後続要求待ち　あるいは　4　SDFと結合中

後条件：SCSM状態　2　後続要求待ち　あるいは　4　SDFと結合中

B) エラー受信

前条件：SCSM状態　4　SDFと結合中

後条件：SCSM状態　4　SDFと結合中

本エラー受信後、ＳＣＦはエラーにより表示された新たなデータベースへアクセスし、サービス論理を継続可能である。

応答側エンティティ(ＳＤＦ)での手順

A) オペレーション受信

前条件：SDSM状態　3　SCFと結合中

後条件：SDSM状態　3　SCFと結合中

B) エラー返送

前条件：SDSM状態　3　SCFと結合中

後条件：SDSM状態　3　SCFと結合中

エラーを返送した後、行われるエラー手順はない。

3.2.1.4　セキュリティエラー(SecurityError)

3.2.1.4.1　一般記述

3.2.1.4.1.1　エラー記述

このエラーは、セキュリティの理由でオペレーションを実行した場合の問題を報告するためにＳＤＦからＳＣＦあるいはＳＤＦに送信される。セキュリティエラーが送信される条件は、ITU-T勧告X.511(1993)12.7章に定義されている。

3.2.1.4.1.2　アーギュメント記述

セキュリティエラーのパラメータ及び問題コードは、ITU-T勧告X.511(1993)12.7章に定義されている。

3.2.1.4.2　ＳＣＦ-->ＳＤＦオペレーション

ディレクトリ結合（Bind）

探索(Search)

エントリ更新(ModifyEntry)

エントリ追加(AddEntry)

エントリ削除(RemoveEntry)

起動側エンティティ(ＳＣＦ)での手順

A) オペレーション送信

前条件：SCSM 状態　1　空（Bind 時）

あるいは　　2　後続要求待ち（Bind 以外のオペレーション時）

あるいは　　4　SDF と結合中（Bind 以外のオペレーション時）

後条件：SCSM 状態　2　後続要求待ち（全ての可能なオペレーション時）

あるいは　　4　SDF と結合中（Bind 以外のオペレーション時）

B) エラー受信

前条件：SCSM 状態　3　 結合結果待ち　（Bind 時）

あるいは　　4　SDF と結合中（Bind 以外のオペレーション時）

後条件：SCSM 状態　1 空（Bind 時）

あるいは　　4　SDF と結合中（Bind 以外のオペレーション時）

エラー手順はサービス論理に非依存である。サービス処理は終了すべきである。

応答側エンティティ(ＳＤＦ)での手順

A) オペレーション受信

前条件：SDSM 状態　1 空（Bind 時）

あるいは　　3　SCF と結合中（Bind 以外のオペレーション時）

後条件：SDSM 状態　2　結合処理中（Bind 時）

あるいは　　3　SCF と結合中（Bind 以外のオペレーション時）

B) エラー返送

前条件：SDSM 状態　2　結合処理中（Bind 時）

あるいは　　3　SCF と結合中（Bind 以外のオペレーション時）

後条件：SDSM 状態　1　空（Bind 時）

あるいは　　3　SCF と結合中（Bind 以外のオペレーション時）

ＳＤＦはオペレーションを実行できないため、ＳＣＦにセキュリティエラーを返送する。エラーを返送した後、行われるエラー手順はない。

3.2.1.4.3　ＳＤＦ-->ＳＤＦオペレーション

DSA結合（DSABind）

連鎖探索(ChainedSearch)

連鎖エントリ更新(ChainedModifyEntry)

連鎖エントリ追加(ChainedAddEntry)

連鎖エントリ削除(ChainedRemoveEntry)

DSAシャドウ結合(DSAShadowBind)

起動側エンティティ(ＳＤＦ)での手順

A) オペレーション送信

前条件：SDSM-ChI 状態　　4　SDF と結合中　あるいは

2　後続要求待ち（連鎖オペレーションの場合）

SDSM-ChI 状態　　1　空き（DSA 結合オペレーションの場合）

SDSM-ShM 状態　　1　空き（シャドウ供給側起動 DSA シャドウ結合の場合）

SDSM-ShC 状態　　1　空き（シャドウ消費側起動 DSA シャドウ結合の場合）

後条件：SDSM-ChI 状態　4　SDF と結合中（連鎖オペレーションの場合）
SDSM-ChI 状態　2　後続要求待ち
（DSA 結合オペレーションおよび連鎖オペレーションの場合）
SDSM-ShM 状態　2　後続要求待ち
（シャドウ供給側起動 DSA シャドウ結合の場合）
SDSM-ShC 状態　2　後続要求待ち
（シャドウ消費側起動 DSA シャドウ結合の場合）

B) エラー受信

前条件：SDSM-ChI 状態　4　SDF と結合中（連鎖オペレーションの場合）
SDSM-ChI 状態　3　結合結果待ち（DSA 結合オペレーションの場合）
SDSM-ChM 状態　3　結合結果待ち（シャドウ供給側起動 DSA シャドウ結合の場合）
SDSM-ShC 状態　3　結合結果待ち（シャドウ消費側起動 DSA シャドウ結合の場合）
後条件：SDSM-ChI 状態　4　SDF と結合中（連鎖オペレーションの場合）
SDSM-ChI 状態　1　空き（DSA 結合オペレーションの場合）
SDSM-ChM 状態　1　空き（シャドウ供給側起動 DSA シャドウ結合の場合）
SDSM-ChC 状態　1　空き（シャドウ消費側起動 DSA シャドウ結合の場合）

連鎖オペレーションの場合、本エラーはＳＣＦへ通知される。

応答側エンティティ(ＳＤＦ)での手順

A) オペレーション受信

前条件：SDSM-ChT 状態　3　SDF と結合中（連鎖オペレーションの場合）
SDSM-ChT 状態　1　空き（DSA 結合オペレーションの場合）
SDSM-ShC 状態　1　空き（シャドウ供給側起動 DSA シャドウ結合の場合）
SDSM-ShM 状態　1　空き（シャドウ消費側起動 DSA シャドウ結合の場合）
後条件：SDSM-ChT 状態　3　SDF と結合中（連鎖オペレーションの場合）
SDSM-ChT 状態　2　結合処理中（DSA 結合オペレーションの場合）
SDSM-ShC 状態　2　結合結果待ち（シャドウ供給側起動 DSA シャドウ結合の場合）
SDSM-ShM 状態　2　結合結果待ち（シャドウ消費側起動 DSA シャドウ結合の場合）

B) エラー返送

前条件：SDSM-ChT 状態　3　SDF と結合中（連鎖オペレーションの場合）
SDSM-ChT 状態　2　結合処理中（DSA 結合オペレーションの場合）
SDSM-ShC 状態　2　結合結果待ち（シャドウ供給側起動 DSA シャドウ結合の場合）
SDSM-ShM 状態　2　結合結果待ち（シャドウ消費側起動 DSA シャドウ結合の場合）
後条件：SDSM-ChT 状態　3　SDF と結合中（連鎖オペレーションの場合）
SDSM-ChT 状態　1　空き（DSA 結合オペレーションの場合）
SDSM-ShC 状態　1　空き（シャドウ供給側起動 DSA シャドウ結合の場合）
SDSM-ShM 状態　1　空き（シャドウ消費側起動 DSA シャドウ結合の場合）

ＳＤＦはオペレーションを実行できないため、ＳＤＦにセキュリティエラーを返送する。エラーを返送した後、行われるエラー手順はない。

3.2.1.5　サービスエラー(ServiceError)

3.2.1.5.1　一般記述

3.2.1.5.1.1　エラー記述

このエラーは、サービスの用意に関連した問題を報告するためにＳＤＦからＳＣＦあるいはＳＤＦに送信される。サービスエラーが送信される条件は、ITU-T勧告X.511(1993)12.8章に定義されている。

3.2.1.5.1.2　アーギュメント記述

サービスエラーのパラメータ及び問題コードは、ITU-T勧告X.511(1993)12.8章に定義されている。

3.2.1.5.2　ＳＣＦ-->ＳＤＦオペレーション

ディレクトリ結合(Bind)

探索(Search)

エントリ更新(ModifyEntry)

エントリ追加(AddEntry)

エントリ削除(RemoveEntry)

起動側エンティティ(ＳＣＦ)での手順

A) オペレーション送信

前条件：SCSM 状態　1　空（Bind 時）
　　あるいは　2　後続要求待ち（Bind 以外のオペレーション時）
　　あるいは　4　SDF と結合中（Bind 以外のオペレーション時）
後条件：SCSM 状態　2　後続要求待ち（全ての可能なオペレーション時）
　　あるいは　4　SDF と結合中（Bind 以外のオペレーション時）

B) エラー受信

前条件：SCSM 状態　3　結合結果待ち（Bind 時）
　　あるいは　4　SDF と結合中（Bind 以外のオペレーション時）
後条件：SCSM 状態　1　空　（Bind 時）
　　あるいは　4　SDF と結合中（Bind 以外のオペレーション時）

エラー手順はサービス論理に依存する。もし、代替のＳＤＦがあれば、別の問い合わせを行うことが可能である。そうでなければサービス処理は終了すべきである。

応答側エンティティ(ＳＤＦ)での手順

A) オペレーション受信

前条件：SDSM 状態　1　空　（Bind 時）
　　あるいは　3　SCF と結合中（Bind 以外のオペレーション時）
後条件：SDSM 状態　2　結合処理中（Bind 時）
　　あるいは　3　SCF と結合中（Bind 以外のオペレーション時）

B) エラー返送

前条件：SDSM 状態　2　結合処理中（Bind 時）
　　あるいは　3　SCF と結合中（Bind 以外のオペレーション時）
後条件：SDSM 状態　1　空　（Bind 時）
　　あるいは　3　SCF と結合中（Bind 以外のオペレーション時）

ＳＤＦはオペレーションを実行できないため、ＳＣＦにサービスエラーを返送する。エラーを返送した後、行われるエラー手順はない。

3.2.1.5.3　ＳＤＦ-->ＳＤＦオペレーション

DSA結合（DSABind）

連鎖探索(ChainedSearch)

連鎖エントリ更新(ChainedModifyEntry)

連鎖エントリ追加(ChainedAddEntry)

連鎖エントリ削除(ChainedRemoveEntry)

DSAシャドウ結合(DSAShadowBind)

起動側エンティティ(ＳＤＦ)での手順

A) オペレーション送信

前条件：	SDSM-ChI状態	4	SDFと結合中
	あるいは	2	後続要求待ち（連鎖オペレーションの場合）
	SDSM-ChI状態	1	空き（DSA結合オペレーションの場合）
	SDSM-ShM状態	1	空き（シャドウ供給側起動DSAシャドウ結合の場合）
	SDSM-ShC状態	1	空き（シャドウ消費側起動DSAシャドウ結合の場合）
後条件：	SDSM-ChI状態	4	SDFと結合中（連鎖オペレーションの場合）
	SDSM-ChI状態	2	後続要求待ち（DSA結合オペレーションおよび連鎖オペレーションの場合）
	SDSM-ShM状態	2	後続要求待ち（シャドウ供給側起動DSAシャドウ結合の場合）
	SDSM-ShC状態	2	後続要求待ち（シャドウ消費側起動DSAシャドウ結合の場合）

B) エラー受信

前条件：	SDSM-ChI状態	4	SDFと結合中（連鎖オペレーションの場合）
	SDSM-ChI状態	3	結合結果待ち（DSA結合オペレーションの場合）
	SDSM-ChM状態	3	結合結果待ち（シャドウ供給側起動DSAシャドウ結合の場合）
	SDSM-ShC状態	3	結合結果待ち（シャドウ消費側起動DSAシャドウ結合の場合）
後条件：	SDSM-ChI状態	4	SDFと結合中（連鎖オペレーションの場合）
	SDSM-ChI状	1	空き（DSA結合オペレーションの場合）
	SDSM-ChM状態	1	空き（シャドウ供給側起動DSAシャドウ結合の場合）
	SDSM-ChC状態	1	空き（シャドウ消費側起動DSAシャドウ結合の場合）

連鎖オペレーションの場合、本エラーはＳＣＦへ通知される。

応答側エンティティ(ＳＤＦ)での手順

A) オペレーション受信

前条件：	SDSM-ChT状態	3	SDFと結合中（連鎖オペレーションの場合）
	SDSM-ChT状態	1	空き（DSA結合オペレーションの場合）
	SDSM-ShC状態	1	空き（シャドウ供給側起動DSAシャドウ結合の場合）
	SDSM-ShM状態	1	空き（シャドウ消費側起動DSAシャドウ結合の場合）
後条件：	SDSM-ChT状態	3	SDFと結合中（連鎖オペレーションの場合）
	SDSM-ChT状態	2	結合処理中（DSA結合オペレーションの場合）
	SDSM-ShC状態	2	結合結果待ち（シャドウ供給側起動DSAシャドウ結合の場合）
	SDSM-ShM状態	2	結合結果待ち（シャドウ消費側起動DSAシャドウ結合の場合）

B) エラー返送

前条件：	SDSM-ChT状態	3	SDFと結合中（連鎖オペレーションの場合）
	SDSM-ChT状態	2	結合処理中（DSA結合オペレーションの場合）
	SDSM-ShC状態	2	結合結果待ち（シャドウ供給側起動DSAシャドウ結合の場合）
	SDSM-ShM状態	2	結合結果待ち（シャドウ消費側起動DSAシャドウ結合の場合）
後条件：	SDSM-ChT状態	3	SDFと結合中（連鎖オペレーションの場合）
	SDSM-ChT状態	1	空き（DSA結合オペレーションの場合）
	SDSM-ShC状態	1	空き（シャドウ供給側起動DSAシャドウ結合の場合）
	SDSM-ShM状態	1	空き（シャドウ消費側起動DSAシャドウ結合の場合）

ＳＤＦはオペレーションを実行できないため、ＳＤＦにサービスエラーを返送する。エラーを返送した後、行われるエラー手順はない。

3.2.1.6 更新エラー(UpdateError)

3.2.1.6.1 一般記述

3.2.1.6.1.1 エラー記述

このエラーは、ＳＤＦ中での情報の追加、削除、あるいは更新に関連した問題を報告するためにＳＤＦからＳＣＦあるいはＳＤＦに送信される。サービスエラーが送信される条件は、ITU-T勧告X.511(1993)12.9章に定義されている。

3.2.1.6.1.2 アーギュメント記述

更新エラーのパラメータ及び問題コードは、ITU-T勧告X.511(1993)12.9章に定義されている。

3.2.1.6.2 ＳＣＦ-->ＳＤＦオペレーション

エントリ更新(ModifyEntry)

エントリ追加(AddEntry)

エントリ削除(RemoveEntry)

起動側エンティティ(ＳＣＦ)での手順

A) オペレーション送信

前条件：SCSM 状態　4　SDF と結合中

あるいは　2　後続要求待ち

後条件：SCSM 状態　4　SDF と結合中

あるいは　2　後続要求待ち

B) エラー受信

前条件：SCSM 状態　4　SDF と結合中

後条件：SCSM 状態　4　SDF と結合中

エラー手順はサービス論理に依存する。

応答側エンティティ(ＳＤＦ)での手順

A) オペレーション受信

前条件：SDSM 状態　3　SCF と結合中

後条件：SDSM 状態　3　SCF と結合中

B) エラー返送

前条件：SDSM 状態　3　SCF と結合中

後条件：SDSM 状態　3　SCF と結合中

ＳＤＦはオペレーションを実行できないため、ＳＣＦに更新エラーを返送する。エラーを返送した後、行われるエラー手順はない。

3.2.1.6.3　ＳＤＦ-->ＳＤＦオペレーション

連鎖エントリ更新(ChainedModifyEntry)

連鎖エントリ追加(ChainedAddEntry)

連鎖エントリ削除(ChainedRemoveEntry)

起動側エンティティ(ＳＤＦ)での手順

A) オペレーション送信

前条件：SDSM-ChI状態　4　SDFと結合中　あるいは　2　後続要求待ち

後条件：SDSM-ChI状態　4　SDFと結合中　あるいは　2　後続要求待ち

B) エラー受信

前条件：SDSM-ChI状態　4　SDFと結合中

後条件：SDSM-ChI状態　4　SDFと結合中

本エラーはＳＣＦへ通知される。

応答側エンティティ(ＳＤＦ)での手順

A) オペレーション受信

前条件：SDSM-ChT状態　3　SDFと結合中

後条件：SDSM-ChT状態　3　SDFと結合中

B) エラー返送

前条件：SDSM-ChT状態　　3　SDFと結合中

後条件：SDSM-ChT状態　　3　SDFと結合中

ＳＤＦはオペレーションを実行できないため、ＳＣＦに更新エラーを返送する。エラーを返送した後、行われるエラー手順はない。

3.2.1.7　DSA リフェラル (DSAReferral)

3.2.1.7.1　一般記述

3.2.1.7.1.1　エラー記述

このエラーは､オペレーションを別のDSAに連鎖して実行することをDSAが望まない場合に生成される。

DSAリフェラルが返送される条件は、ITU-T勧告X.518(1993)8.3章に定義されている。

3.2.1.7.1.2　アーギュメント記述

DSAReferralのパラメータ及び問題コードは、ITU-T勧告X.518(1993)13.2章に定義されている。

3.2.1.7.2　ＳＤＦ-->ＳＤＦオペレーション

連鎖探索(ChainedSearch)

連鎖エントリ更新(ChainedModifyEntry)

連鎖エントリ追加(ChainedAddEntry)

連鎖エントリ削除(ChainedRemoveEntry)

起動側エンティティ(ＳＤＦ)での手順

A) オペレーション送信

前条件：SDSM-ChI状態　　4　SDFと結合中　あるいは　2　後続要求待ち

後条件：SDSM-ChI状態　　4　SDFと結合中　あるいは　2　後続要求待ち

B) エラー受信

前条件：SDSM-ChI 状態　　4　SDFと結合中

後条件：SDSM-ChI 状態　　4　SDFと結合中

本エラー受信後、ＳＤＦはエラーにより表示された新たなデータベースへアクセスし、オペレーションを継続することが可能である。

応答側エンティティ(ＳＤＦ)での手順

A) オペレーション受信

前条件：SDSM-ChT状態　　3　SDFと結合中

後条件：SDSM-ChT状態　　3　SDFと結合中

B) エラー返送

前条件：SDSM-ChT 状態　　3　SDFと結合中

後条件：SDSM-ChT 状態　　3　SDFと結合中

エラーを返送した後、行われるエラー手順はない。

3.2.1.8　シャドウエラー(ShadowError)

3.2.1.8.1　一般記述

3.2.1.8.1.1　エラー記述

このエラーは、シャドウイングオペレーションに適用される。シャドウエラーが返送される条件は、ITU-T勧告X.525(1993)12章に定義されている。

3.2.1.8.1.2　アーギュメント記述

シャドウエラーのパラメータ及び問題コードは、ITU-T勧告X.525(1993)11.3.3章に定義されている。

3.2.1.8.2　ＳＤＦ-->ＳＤＦオペレーション

シャドウ更新調整(coordinateShadowUpdate)

シャドウ更新要求(requestShadowUpdate)

シャドウ更新(updateShadow)

マスタエンティティ(ＳＤＦ)での手順

A-1) オペレーション送信

前条件：

SDSM-ShM（シャドウ供給側起動による供給側状態モデル）

4　SDFと結合中（coordinateShadowUpdateの場合）あるいは

6　更新待ち（updateShadowの場合）

SDSM-ShM（シャドウ消費側起動による供給側状態モデル）

5　更新待ち（updateShadowの場合）

後条件：

SCSM-ShM（シャドウ供給側起動による供給側状態モデル）

5　調整結果待ち（coordinateShadowUpdateの場合）あるいは

7　更新確認待ち（updateShadowの場合）

SDSM-ShM（シャドウ消費側起動による供給側状態モデル）

6　更新確認待ち（updateShadowの場合）

B-1) エラー受信

前条件：

SDSM-ShM（シャドウ供給側起動による供給側状態モデル）

5　調整結果待ち（coordinateShadowUpdateの場合）あるいは

7　更新確認待ち（updateShadowの場合）

SDSM-ShM（シャドウ消費側起動による供給側状態モデル）

6　更新確認待ち（updateShadowの場合）

後条件：

SDSM-ShM（シャドウ消費側起動による供給側状態モデル）

4　SDFと結合中（coordinateShadowUpdateの場合）あるいは

6　更新待ち（updateShadowの場合）

SDSM-ShM（シャドウ消費側起動による供給側状態モデル）

5　更新待ち（updateShadowの場合）

A-2) オペレーション受信

前条件：

SDSM-ShM（シャドウ消費側起動による供給側状態モデル）

3　SDFと結合中（requestShadowUpdateの場合）

後条件：

SDSM-ShM（シャドウ消費側起動による供給側状態モデル）

4　シャドウ要求結果待ち（requestShadowUpdateの場合）

B-2) エラー返送

前条件：

SDSM-ShM（シャドウ消費側起動による供給側状態モデル）

4　シャドウ要求結果待ち（requestShadowUpdateの場合）

後条件：

SDSM-ShM（シャドウ消費側起動による供給側状態モデル）

3　SDFと結合中（requestShadowUpdateの場合）

本エラーの受信/返送後、行われるエラー手順はない。

消費側エンティティ(ＳＤＦ)での手順

A-1) オペレーション送信

前条件：

SDSM-ShC（シャドウ消費側起動による消費側状態モデル）

4　SDFと結合中（requestShadowUpdateの場合）

後条件：

SDSM-ShC（シャドウ消費側起動による消費側状態モデル）

5　シャドウ要求結果待ち（requestShadowUpdateの場合）

B-1) エラー受信

前条件：

SDSM-ShC（シャドウ消費側起動による消費側状態モデル）

5　シャドウ要求結果待ち（requestShadowUpdateの場合）

後条件：

SDSM-ShC（シャドウ消費側起動による消費側状態モデル）

4　SDFと結合中（requestShadowUpdateの場合）

A-2) オペレーション受信

前条件：

SDSM-ShC（シャドウ供給側起動による消費側状態モデル）

3　SDFと結合中（coordinateShadowUpdateの場合）あるいは

5　更新待ち（updateShadowの場合）

SDSM-ShC（シャドウ消費側起動による消費側状態モデル）

6　更新待ち(updateShadowの場合）

後条件：

SCSM-ShC（シャドウ供給側起動による消費側状態モデル）

4　調整結果待ち（coordinateShadowUpdateの場合）あるいは

6　更新確認待ち（updateShadowの場合）

SDSM-ShC（シャドウ消費側起動による消費側状態モデル）

7　更新確認待ち（updateShadowの場合）

B-2) エラー返送

前条件：

SDSM-ShC（シャドウ供給側起動による消費側状態モデル）

4　調整結果待ち（coordinateShadowUpdateの場合）あるいは

6　更新確認待ち（updateShadowの場合）

SDSM-ShC（シャドウ消費側起動による消費側状態モデル）

7　更新確認待ち（updateShadowの場合）

後条件：

SDSM-ShC（シャドウ供給側起動による消費側状態モデル）

3　SDFと結合中（coordinateShadowUpdateの場合）あるいは

5　更新待ち（updateShadowの場合）

SDSM-ShC（シャドウ消費側起動による消費側状態モデル）

6　更新待ち（updateShadowの場合）

本エラーの受信/返送後、行われるエラー手順はない。

3.3 オペレーション手順の詳細

3.3.1 ディレクトリ結合手順

3.3.1.1 概要

X.500 の‘ディレクトリ結合(DirectoryBind)’オペレーションは、ＳＣＦがエンドユーザに代わってＳＣＦとＳＤＦ間の認証されたアソシエーションを確立するために使用される。エンドユーザの認証情報がある場合にはこれがそれを伝える。ディレクトリ結合オペレーションの完全な記述については ITU-T 勧告 X.511 の 8.1 節を参照。

3.3.1.1.1 パラメータ

ITU-T 勧告 X.511 の 8.1.2 及び 8.1.3 節を参照。

3.3.1.2 起動側エンティティ(ＳＣＦ)

3.3.1.2.1 通常手順

SCF 前条件：

(1) SCSM:“空き”

SCF 後条件：

(1) SCSM:“SDF と結合中”(ディレクトリ結合が成功した場合)

(2) SCSM:“空き”(ディレクトリ結合が失敗した場合)

SCSM が“空き”状態にあり、ＳＤＦへの問い合わせがサービス論理において必要である場合、内部イベントが生じる。このイベントは(e1)結合要求と呼ばれ、“後続要求待ち”状態への遷移を引き起こし、他のオペレーションは待たされる。アプリケーションプロセスがデリミタによって結合の送信を指示しない限り、SCSM は“後続要求待ち”状態に留まり、オペレーションは送信されない。デリミタが受信されると(e3)結合と共に SDF への要求という内部イベントを経て“結合結果待ち”状態へ遷移する。オペレーションはＳＤＦに送信される。SCSM はＳＤＦからの応答を待つ。ディレクトリ結合オペレーションが成功した場合は、以前にＳＤＦに対して発行されたディレクトリ結合に対する応答の受信（(E5)結合と共に SDF からの応答）により、ＳＣＦの状態は“SDF と結合中”という状態へ遷移する。その他の場合は、SCSM はエラーの受信（(E4)結合エラー）により“空き”状態へ戻る。

3.3.1.2.2 エラー手順

オペレーションに関連するエラーの一般的な扱いは ITU-T 勧告 X.511 の 8.1.4 節に､オペレーションエラーを通知するためのＴＣＡＰサービスは第３編 2.1 章に記述されている。

3.3.1.3 応答側エンティティ(ＳＤＦ)

3.3.1.3.1 通常手順

SDF 前条件：

(1) SDSM:“空き”

SDF 後条件：

(1) SDSM:“SCF と結合中”(ディレクトリ結合が成功した場合)

(2) SDSM:“空き”(ディレクトリ結合が失敗した場合)

ＳＤＦは最初“空き”状態にある。ＳＣＦからの‘ディレクトリ結合’オペレーションの受信により引き起こされる(E1)SCF からの結合という外部イベントの受信後、“結合処理中”状態へ遷移する。ＳＤＦは Bind argument の内容に従ってディレクトリ結合オペレーションを実行する。ＳＤＦが‘ディレクトリ結合’オペレーションを完了させると、結果応答あるいはエラーがＳＣＦに対して返される。ＳＤＦはディレクトリ結合が失敗した場合には“空き”状態に戻り、成功した場合には“SCF と結合中”状態へ遷移する。ディレクトリ結合要求が成功した場合、返送される結果応答はディレクトリ結合結果(DirectoryBindResult)中の資格証明(Credentials)から構成される。これらの資格証明によりユーザはディレクトリを識別することができる。これにより(ディレクトリサービスを直接提供する)ＤＳＡの識別情報をＤＵＡに伝達することができる。資格証明は、ユーザによって提供されたのと同じ形式(即ち名前とパスワード)でなければならない。

尚、認証手法には簡易認証と厳密認証があり、簡易認証は更にユーザ識別名認証、単純パスワード認証、秘匿パスワード認証に分類される。簡易認証においてどの認証手法が適用されるかは、ＳＣＦから送信されるパラメータの種別により決定される。

3.3.1.3.2　エラー手順

オペレーションに関連するエラーの一般的な扱いは ITU-T 勧告 X.511 の 8.1.4 節に､オペレーションエラーを通知するためのＴＣＡＰサービスは第３編 2.1 章に記述されている。

3.3.2　ディレクトリ結合解放手順

3.3.2.1　概要

X.500 の‘ディレクトリ結合開放(Unbind)’オペレーションは、ＳＣＦがエンドユーザに代わってＳＣＦとＳＤＦ間の認証済みアクセスを終了させるために使用される。

結合解放オペレーションの完全な記述については ITU-T 勧告 X.511 の 8.2 節を参照。

3.3.2.1.1　パラメータ

なし。

3.3.2.2　起動側エンティティ(ＳＣＦ)

3.3.2.2.1　通常手順

SCF 前条件：

(1) SCSM:“SDF と結合中”

SCF 後条件：

(1) SCSM:“空き”

SCSM はＳＤＦディレクトリに対して既にディレクトリ結合オペレーションを開始しており、“SDF と結合中”状態にある。サービス論理が認証済みのＳＤＦアクセスが終了すべきことを決定する。SCSM はディレクトリ結合解放オペレーション（(e8)結合解放要求）を発行し、“空き”状態への遷移が生ずる。

3.3.2.2.2　エラー手順

‘ディレクトリ結合解放’オペレーションは、オペレーションに関連するエラーを持たない。

3.3.2.3　応答側エンティティ(ＳＤＦ)

3.3.2.3.1　通常手順

SDF 前条件：

(1) SDSM:“SCF と結合中”

SDF 後条件：

(1) SDSM:“空き”

結合オペレーションが既に発行され、SDSM は“SCF と結合中”状態でＳＣＦからの要求を待っているか、またはオペレーションを実行している。結合解放オペレーションを受信すると、(E5)SCF からの結合解放により“空き”状態に遷移する。

3.3.2.3.2　エラー手順

‘ディレクトリ結合解放’オペレーションは、オペレーションに関連するエラーを持たない。

3.3.3　探索手順

3.3.3.1　概要

X.500 の‘探索(Search)’オペレーションは、関心のあるエントリを探すためにＳＤＦ内のＤＩＴの一部を検索し、それらのエントリから選択された情報を返送するために用いられる。探索オペレーションの完全な記述については ITU-T 勧告 X.511 の 10.2 節を参照。

3.3.3.1.1　パラメータ

ITU-T 勧告 X.511 の 10.2.2 及び 10.2.3 節を参照。

3.3.3.2　起動側エンティティ(ＳＣＦ)

3.3.3.2.1　通常手順

SCF 前条件：

(1) SCSM:“SDF と結合中”あるいは“後続要求待ち”

SCF 後条件：

(1) SCSM:“SDF と結合中”

SCSM が“後続要求待ち”状態にあり、ＳＤＦに管理されている情報の検索あるいは参照がサービス論理において必要である場合、内部イベント（(e2)SDF への要求）が生じる。アプリケーションプロセスが、デリミタ（またはタイマの満了）によってオペレーションの送信を指示しない限り、SCSM は“後続要求待ち”状態に留まりオペレーションは送信されない。オペレーションは Bind argument を含んだメッセージによってＳＤＦに送信される。SCSM はＳＤＦからの応答を待つ。以前にＳＤＦに対して発行されたディレクトリ結合オペレーションに対する応答の受信((E5)結合と共に SDF からの応答あるいは(E4)結合エラー)を受けて、ＳＣＦは“SDF と結合中”または“空き”状態へ遷移する。SCSM が“空き”状態に遷移した場合、探索オペレーションは破棄される。“SDF と結合中”状態において探索オペレーションの応答（(E7)SDF からの応答）を受けるとＳＣＦは同一の状態（“SDF と結合中”）へ遷移する。その応答は、探索オペレーションの結果、あるいはエラーであるかも知れない。

SCSM が“SDF と結合中”状態にあり、ＳＤＦに管理されている情報の検索あるいは参照がサービス論理において必要である場合、内部イベントが生じる。このイベントは(e6)SDF への要求と呼ばれ、全く同じ“SDF と結合中”状態への遷移を引き起こし、SCSM はＳＤＦからの応答を待つ。以前にＳＤＦに対して発行された探索オペレーションに対する応答の受信((E7)SDF からの応答)を受けて、ＳＣＦは全く同じ“SDF と結合中”状態へ遷移する。ＳＤＦからの応答は、探索オペレーションの結果、あるいはエラーであるかもしれない。

3.3.3.2.2　エラー手順

オペレーションに関連するエラーの一般的な扱いは ITU-T 勧告 X.511 の 10.2.4 節及び 10.2.5 節に、オペレーションエラーを通知するためのＴＣＡＰサービスは第３編 2.1 章に記述されている。

3.3.3.3 応答側エンティティ(ＳＤＦ)

3.3.3.3.1 通常手順

SDF 前条件：

(1) SDSM:“SCF と結合中”あるいは“結合処理中”

SDF 後条件：

(1) SDSM:“SCF と結合中”

ＳＤＦは“結合処理中”状態にあり、ＳＣＦからの‘探索(Search)’オペレーションの受信により引き起こされる(E3)SCF からの要求という外部イベントが生ずる。ＳＤＦはディレクトリ結合オペレーションの実行が成功するまでオペレーションを処理せず、同一状態に留まる。

ＳＤＦは“SCF と結合中”状態にあり、ＳＣＦからの‘探索(Search)’オペレーションの受信により引き起こされる(E7)SCF からの要求という外部イベントが生ずる。ＳＤＦはオペレーションが実行されるのを待つ。

イベント(E7)を受信すると、オペレーションのパラメータの規定に従ってデータの参照を行う前に、ＳＤＦは以下の動作を行う。

-要求に関するオブジェクトが存在することの検証

-要求はユーザに代わって実行されるが、ユーザがオペレーションの実行に際して辿られるオブジェクト及び属性にアクセスするための十分なアクセス権を有しているかどうかの検証

-オブジェクト中にオペレーションが実行されるべき属性が存在するかどうかの検証

上述の動作が完全に行われた後、ＳＤＦは参照規範を満足する全ての属性をＳＣＦに返送する。結果応答の送信は(e6)SCF への応答というイベントに対応する。

3.3.3.3.2 エラー手順

オペレーションに関連するエラーの一般的な扱いは ITU-T 勧告 X.511 の 10.2.4 節及び 10.2.5 節に、オペレーションエラーを通知するためのＴＣＡＰサービスは第３編 2.1 章に記述されている。

3.3.4 エントリ更新手順

3.3.4.1 概要

X.500 の‘エントリ更新(ModifyEntry)’オペレーションはデータオブジェクトに対して、一つあるいは複数の変更を施すことをＳＤＦに要求するために使用される。変更はオブジェクトを構成する属性とその値に対して行われる。実行される変更種別は、ＳＣＦが指定するオペレーションアーギュメント中で与えられる。変更は、オブジェクトを識別するために使用される属性(即ちオブジェクト名)に対して行われることはない。エントリ更新オペレーションの完全な記述については ITU-T 勧告 X.511 の 11.3 節を参照。

3.3.4.1.1 パラメータ

ITU-T 勧告 X.511 の 11.3.2 及び 11.3.3.3 節を参照。

3.3.4.2 起動側エンティティ(ＳＣＦ)

3.3.4.2.1 通常手順

SCF 前条件：

(1) SCSM:“SDF と結合中”あるいは“後続要求待ち”

SCF 後条件：

(1) SCSM:“SDF と結合中”

SCSM が“後続要求待ち”状態にあり、ＳＤＦに管理されている情報の更新がサービス論理において必要である場合、内部イベント（(e2)SDF への要求）が生じる。アプリケーションプロセスが、デリミタ（またはタイマの満了）によってオペレーションの送信を指示しない限り、SCSM は“後続要求待ち”状態に留まりオペレーションは送信されない。オペレーションは Bind argument を含んだメッセージによってＳＤＦに送信される。SCSM はＳＤＦからの応答を待つ。以前にＳＤＦに対して発行されたディレクトリ結合オペレーションに対する応答の受信((E5)結合と共に SDF からの応答あるいは(E4)結合エラー)を受けて、ＳＣＦは“SDF と結合中”または“空き”状態へ遷移する。SCSM が“空き”状態に遷移した場合、エントリ更新オペレーションは破棄される。“SDF と結合中”状態においてエントリ更新オペレーションの応答（(E7)SDF からの応答）を受けるとＳＣＦは同一の状態（“SDF と結合中”）へ遷移する。その応答は、エントリ更新オペレーションの結果、あるいはエラーであるかも知れない。

SCSM が“SDF と結合中”状態にあり、ＳＤＦに管理されている情報の更新がサービス論理において必要である場合、内部イベントが生じる。このイベントは(e6)SDF への要求と呼ばれ、全く同じ“SDF と結合中”状態への遷移を引き起こし、SCSM はＳＤＦからの応答を待つ。以前にＳＤＦに対して発行されたエントリ更新オペレーションに対する応答の受信((E7)SDF からの応答)を受けて、ＳＣＦは全く同じ“SDF と結合中”状態へ遷移する。ＳＤＦからの応答は、エントリ更新オペレーションの結果、あるいはエラーであるかも知れない。

3.3.4.2.2　エラー手順

オペレーションに関連するエラーの一般的な扱いは ITU-T 勧告 X.511 の 11.3.4 節及び 11.3.5 節に、オペレーションエラーを通知するためのＴＣＡＰサービスは第３編 2.1 章に記述されている。

3.3.4.3　応答側エンティティ(ＳＤＦ)

3.3.4.3.1　通常手順

SDF 前条件：

(1) SDSM:“SCF と結合中”あるいは“結合処理中”

SDF 後条件：

(1) SDSM:“SCF と結合中”

ＳＤＦは“結合処理中”状態にあり、ＳＣＦからの‘エントリ更新(ModifyEntry)’オペレーションの受信により引き起こされる(E3)SCF からの要求という外部イベントが生ずる。ＳＤＦは結合オペレーションの実行が成功するまでオペレーションを処理せず、同一状態に留まる。

ＳＤＦは“SCF と結合中”状態にありＳＣＦからの‘エントリ更新(ModifyEntry)’オペレーションの受信により引き起こされる(E7)SCF からの要求という外部イベントが生ずる。ＳＤＦはオペレーションが実行されるのを待つ。

イベント(E7)を受信すると、オペレーションのパラメータに規定される属性の更新を行う前にＳＤＦは以下の動作を行う。

-要求に関するオブジェクトが存在することの検証

-要求はユーザに代わって実行されるが、ユーザがオペレーションに含まれる個々の変更をオブジェクトに対して行うための十分なアクセス権を有しているかどうかの検証

-オブジェクト中にオペレーションが実行されるべき属性あるいは値が存在するかどうかの検証

-要求された変更がオブジェクトの属するオブジェクトクラスあるいは属性の構文と矛盾しないことの検証

上述の動作がオペレーションで要求される全ての変更に対して完全に行われた後、同一の属性に対する全ての変更が‘変更順(changes)’パラメータに与えられた順序に従って行われる。結果応答がＳＣＦに返される。結果応答の送信は(e6)SCF への応答というイベントに対応する。

3.3.4.3.2　エラー手順

オペレーションに関連するエラーの一般的な扱いは ITU-T 勧告 X.511 の 11.3.4 節及び 11.3.5 節に、オペレーションエラーを通知するためのＴＣＡＰサービスは第３編 2.1 章に記述されている。

3.3.5　エントリ追加手順

3.3.5.1　概要

X.500 の‘エントリ追加(AddEntry)’オペレーションはＤＩＴへの葉エントリ（オブジェクトエントリあるいはエリアスエントリ）の追加をＳＤＦに要求するために用いられる。エントリ追加オペレーションの完全な記述については ITU-T 勧告 X.511 の 11.1 節を参照。

3.3.5.1.1　パラメータ

ITU-T 勧告 X.511 の 11.1.1 及び 11.1.2 節を参照。

3.3.5.2　起動側エンティティ(ＳＣＦ)

3.3.5.2.1　通常手順

SCF 前条件：

(1) SCSM:“SDF と結合中”あるいは“後続要求待ち”

SCF 後条件：

(1) SCSM:“SDF と結合中”

SCSM が“後続要求待ち”状態にあり、ＳＤＦにおけるエントリの追加がサービス論理において必要である場合、内部イベント（(e2)SDF への要求）が生じる。アプリケーションプロセスが、デリミタ（またはタイマの満了）によってオペレーションの送信を指示しない限り、SCSM は“後続要求待ち”状態に留まりオペレーションは送信されない。オペレーションは Bind argument を含んだメッセージによってＳＤＦに送信される。SCSM はＳＤＦからの応答を待つ。以前にＳＤＦに対して発行されたディレクトリ結合オペレーションに対する応答の受信((E5)結合と共に SDF からの応答あるいは(E4)結合エラー)を受けて、ＳＣＦは“SDF と結合中”または“空き”状態へ遷移する。SCSM が“空き”状態に遷移した場合、エントリ追加オペレーションは破棄される。“SDF と結合中”状態においてエントリ追加オペレーションの応答（(E7)SDF からの応答）を受けるとＳＣＦは同一の状態（“SDF と結合中”）へ遷移する。その応答は、エントリ追加オペレーションの結果、あるいはエラーであるかも知れない。

SCSM が“SDF と結合中”状態にあり、ＳＤＦにおけるエントリの追加がサービス論理において必要である場合、内部イベントが生じる。このイベントは(e6)SDF への要求と呼ばれ、全く同じ“SDF と結合中”状態への遷移を引き起こし、SCSM はＳＤＦからの応答を待つ。以前にＳＤＦに対して発行されたエントリ追加オペレーションに対する応答の受信((E7)SDF からの応答)を受けて、ＳＣＦは全く同じ“SDF と結合中”状態へ遷移する。ＳＤＦからの応答は、エントリ追加の結果、あるいはエラーであるかも知れない。

3.3.5.2.2　エラー手順

オペレーションに関連するエラーの一般的な扱いは ITU-T 勧告 X.511 の 11.1.4 節及び 11.1.5 節に、オペレーションエラーを通知するためのＴＣＡＰサービスは第３編 2.1 章に記述されている。

3.3.5.3　応答側エンティティ(ＳＤＦ)

3.3.5.3.1　通常手順

SDF 前条件：

(1) SDSM:“結合処理中”あるいは“SCF と結合中”

SDF 後条件：

(1) SDSM:“SCF と結合中”

ＳＤＦは“結合処理中”状態にあり、ＳＣＦからの‘エントリ追加(AddEntry)’オペレーションの受信により引き起こされる(E3)SCF からの要求という外部イベントが生ずる。ＳＤＦは結合オペレーションの実行が成功するまでオペレーションを処理せず、同一状態に留まる。

ＳＤＦは“SCF と結合中”状態にあり、ＳＣＦからの‘エントリ追加(AddEntry)’オペレーションの受信により引き起こされる(E7)SCF からの要求という外部イベントが生ずる。ＳＤＦはオペレーションが実行されるのを待つ。

イベント(E7)を受信すると新たなエントリを追加する前に、ＳＤＦは以下の動作を行う。

-エントリ追加が行われる上位オブジェクトがＳＤＦに存在することの検証

-SDF に追加するエントリが未だ存在しないことの検証

-エントリとその構成要素（属性及び値）を追加するアクセス権が十分であることの検証

-エントリがディレクトリスキーマに従っていることの検証

上述の動作が完全に行われた後、エントリがＳＤＦのデータベースに追加される。ＳＣＦに対して空の結果が返送される。結果応答の送信は(e6)SCF への応答というイベントに対応する。

3.3.5.3.2　エラー手順

オペレーションに関連するエラーの一般的な扱いは ITU-T 勧告 X.511 の 11.1.4 節及び 11.1.5 節に、オペレーションエラーを通知するためのＴＣＡＰサービスは第 3 編 2.1 章に記述されている。

3.3.6　エントリ削除手順

3.3.6.1　概要

X.500 の‘エントリ削除(removeEntry)’オペレーションはＤＩＴからの葉エントリ（オブジェクトエントリあるいはエリアスエントリ）の削除をＳＤＦに要求するために用いられる。エントリ削除オペレーションの完全な記述については ITU-T 勧告 X.511 の 11.2 節を参照。

3.3.6.1.1　パラメータ

ITU-T 勧告 X.511 の 11.2.2 及び 11.2.3 節を参照。

3.3.6.2　起動側エンティティ(ＳＣＦ)

3.3.6.2.1　通常手順

SCF 前条件：

(1) SCSM:“SDF と結合中”あるいは“後続要求待ち”

SCF 後条件：

(1) SCSM:“SDF と結合中”

SCSM が“後続要求待ち”状態にあり、ＳＤＦからのエントリの削除がサービス論理において必要である場合、内部イベント（(e2)SDF への要求）が生じる。アプリケーションプロセスが、デリミタ（またはタイマの満了）によってオペレーションの送信を指示しない限り、SCSM は“後続要求待ち”状態に留まりオペレーションは送信されない。オペレーションは Bind argument を含んだメッセージによってＳＤＦに送信される。SCSM はＳＤＦからの応答を待つ。以前にＳＤＦに対して発行されたディレクトリ結合オペレーションに対する応答の受信((E5)結合と共に SDF からの応答あるいは(E4)結合エラー)を受けて、ＳＣＦは“SDF と結合中”または“空き”状態へ遷移する。SCSM が“空き”状態に遷移した場合、エントリ削除オペレーションは破棄される。“SDF と結合中”状態においてエントリ削除オペレーションの応答（(E7)SDF からの応答）を受けるとＳＣＦは同一の状態（“SDF と結合中”）へ遷移する。その応答は、エントリ削除オペレーションの結果、あるいはエラーであるかも知れない。

SCSM が“SDF と結合中”状態にあり、ＳＤＦからのエントリの削除がサービス論理において必要である場合、内部イベントが生じる。このイベントは(e6)SDF への要求と呼ばれ、全く同じ“SDF と結合中”状態への遷移を引き起こし、SCSM はＳＤＦからの応答を待つ。以前にＳＤＦに対して発行されたエントリ削除オペレーションに対する応答の受信((E7)SDF からの応答)を受けて、ＳＣＦは全く同じ“SDF と結合中”状態へ遷移する。ＳＤＦからの応答は、エントリ削除オペレーションの結果、あるいはエラーであるかも知れない。

3.3.6.2.2　エラー手順

オペレーションに関連するエラーの一般的な扱いは ITU-T 勧告 X.511 の 11.2.4 節及び 11.2.5 節に、オペレーションエラーを通知するためのＴＣＡＰサービスは第３編 2.1 章に記述されている。

3.3.6.3　応答側エンティティ(ＳＤＦ)

3.3.6.3.1　通常手順

SDF 前条件：

(1) SDSM:“結合処理中”あるいは“SCF と結合中”

SDF 後条件：

(1) SDSM:“SCF と結合中”

ＳＤＦは“結合処理中”状態にあり、ＳＣＦからの‘エントリ削除(removeEntry)’オペレーションの受信により引き起こされる(E3)SCF からの要求という外部イベントが生ずる。ＳＤＦは結合オペレーションの実行が成功するまでオペレーションを処理せず、同一状態に留まる。

ＳＤＦが“SCF と結合中”状態にありＳＣＦからの‘エントリ削除(removeEntry)’オペレーションの受信により引き起こされる(E7)SCF からの要求という外部イベントが生ずる。ＳＤＦはオペレーションが実行されるのを待つ。

イベント(E7)を受信すると、エントリを削除する前に、ＳＤＦは以下の動作を行う。

-削除されるエントリが存在しかつ葉エントリであることの検証

-エントリを削除するためのアクセス権が十分であることの検証

上述の動作が完全に行われた後、ＳＤＦのデータベースからエントリが削除される。ＳＣＦに対して空の結果が返送される。結果応答の送信は(e6)SCF への応答というイベントに対応する。

3.3.6.3.2　エラー手順

オペレーションに関連するエラーの一般的な扱いは ITU-T 勧告 X.511 の 11.2.4 節及び 11.2.5 節に、オペレーションエラーを通知するためのＴＣＡＰサービスは第３編 2.1 章に記述されている。

3.3.7　連鎖オペレーション手順

3.3.7.1　概要

X.500 の‘連鎖(chained)’オペレーションは、最初のＳＤＦが別のＳＤＦを起動し、それ自身では実行不可能なオペレーションを行うために用いられる。‘ディレクトリ結合’、‘ディレクトリ結合解放’を除く SCF/SDF インタフェースで定義された個々のオペレーションに対応する‘連鎖(chained)’オペレーションが、オペレーションを実行する上で協調するＳＤＦ間を利用するための SDF 抽象サービスにおいて定義される。‘連鎖(chained)’オペレーションの完全な記述については ITU-T 勧告 X.518 の 12.1 節を参照。

別のＳＤＦをアクセスする期間の初めと終わりに、ＤＳＡ結合及びＤＳＡ結合解放オペレーションが協調するＳＤＦにより使用される。‘ＤＳＡ結合(DSABind)’及び‘ＤＳＡ結合解放(DSAUnbind)’オペレーションの完全な記述については ITU-T 勧告 X.518 の 11 節を参照。

3.3.7.1.1　パラメータ

‘連鎖(chained)’オペレーションに関しては ITU-T 勧告 X.518 の 10.3 節を参照。

‘ＤＳＡ結合(DSABind)’オペレーションのパラメータは、第３編 2.1.2.1.2 に規定される‘in-DirectoryBind’オペレーションと同様である。‘ＤＳＡ結合解放(DSAUnbind)’オペレーションはパラメータを持たない。

3.3.7.1.2　エラー手順

オペレーションに関連するエラーの一般的な扱いは ITU-T 勧告 X.518 の 13 節に、オペレーションエラーを通知するためのＴＣＡＰサービスは第３編 2.2.6 章に記述されている。

3.3.7.2　ＤＳＡ結合

3.3.7.2.1　起動側エンティティ(ＳＤＦ)

SDF 前条件：

(1) SDSM-ChI:“空き”

SDF 後条件：

(1) SDSM-ChI:“SDF と結合中”(ＤＳＡ結合が成功した場合)

(2) SDSM-ChI:“空き”(ＤＳＡ結合が失敗した場合)

SDSM-ChI が“空き”状態にあり、別のＳＤＦへの問い合わせが必要である場合、内部イベントが生じる。このイベントは(e1)SDF への DSA 結合と呼ばれ、“後続要求待ち”状態への遷移を引き起こし、他のオペレーションは待たされる。アプリケーションプロセスがデリミタによって結合の送信を指示しない限り、SDSM-ChI は“後続要求待ち”状態に留まり、オペレーションは送信されない。デリミタが受信されると(e3)結合および要求送信という内部イベントを経て“結合結果待ち”状態へ遷移する。オペレーションは他のＳＤＦ(終端側ＳＤＦ)に送信される。SDSM-ChI はＳＤＦからの応答を待つ。‘ＤＳＡ結合’オペレーションが成功した場合は、以前にＳＤＦに対して発行された‘ＤＳＡ結合’に対する応答の受信（(E5)DSA 結合成功）により、ＳＤＦの状態は“SDF と結合中”という状態へ遷移する。その他の場合は、SDSM-ChI はエラーの受信（(E4)DSA 結合エラー）により“空き”状態へ戻る。

3.3.7.2.2　応答側エンティティ(ＳＤＦ)

SDF 前条件：

(1) SDSM-ChT:“空き”

SDF 後条件：

(1) SDSM-ChT:“SDF と結合中”(ＤＳＡ結合が成功した場合)

(2) SDSM-ChT:“空き”(ＤＳＡ結合が失敗した場合)

ＳＤＦは最初“空き”状態にある。ＳＤＦ(起動側ＳＤＦ)からの‘ＤＳＡ結合’オペレーションの受信により引き起こされる(E1)SDF からの DSA 結合という外部イベントの受信後、“結合処理中”状態へ遷移する。ＳＤＦは dSABind argument の内容に従って‘ＤＳＡ結合’オペレーションを実行する。ＳＤＦが‘ＤＳＡ結合’オペレーションを完了させると、結果応答あるいはエラーが起動側ＳＤＦに対して返される。ＳＤＦはＤＳＡ結合が失敗した場合には“空き”状態に戻り、成功した場合には“SDF と結合中”状態へ遷移する。

3.3.7.3　ＤＳＡ結合解放

3.3.7.3.1　起動側エンティティ(ＳＤＦ)

SDF 前条件：

(1) SDSM-ChI:“SDF と結合中”

SDF 後条件：

(1) SDSM-ChI:“空き”

SDSM-ChI は他のＳＤＦに対して既に‘ＤＳＡ結合オペレーション’を開始しており、“SDF と結合中”状態にある。ＳＤＦは協調するＳＤＦへの認証済みのアクセスが終了すべきことを決定すると‘ＤＳＡ結合解放’オペレーション((e6)SDF への DSA 結合解放)を発行し、SDSM-ChI は“空き”状態に遷移する。

3.3.7.3.2　応答側エンティティ(ＳＤＦ)

SDF 前条件：

(1) SDSM-ChT:“SDF と結合中”

SDF 後条件：

(1) SDSM-ChT:“空き”

‘ＤＳＡ結合’オペレーションが既に発行され、SDSM-ChT は“SDF と結合中”状態でＳＤＦからの要求を待っているか、またはオペレーションを実行している。‘ＤＳＡ結合解放’オペレーションを受信すると、(E5)SDF からの DSA 結合解放により“空き”状態に遷移する。

3.3.7.4　連鎖オペレーション

3.3.7.4.1　起動側エンティティ(ＳＤＦ)

SDF 前条件：

(1) SDSM-ChI:“SDF と結合中”あるいは“後続要求待ち”

SDF 後条件：

(1) SDSM-ChI:“SDF と結合中”

SDSM-ChI が“後続要求待ち”状態にあり、ＳＣＦから受信したオペレーションを連鎖することが必要である場合、内部イベント（(e2)SDF への要求）が生じる。アプリケーションプロセスが、デリミタ（またはタイマの満了）によってオペレーションの送信を指示しない限り、SDSM-ChI は“後続要求待ち”状態に留まりオペレーションは送信されない。オペレーションは dSABind argument を含んだメッセージによって終端側ＳＤＦに送信される。SDSM-ChI は終端側ＳＤＦからの応答を待つ。以前に終端側ＳＤＦに対して発行されたＤＳＡ結合オペレーションに対する応答の受信((E5)DSA 結合成功あるいは(E4)DSA 結合エラー)を受けて、起動側ＳＤＦは“SDF と結合中”または“空き”状態へ遷移する。SDSM-ChI が“空き”状態に遷移した場合、連鎖オペレーションは破棄される。“SDF と結合中”状態において連鎖オペレーションの応答（(E7)SDF からの応答）を受けると起動側ＳＤＦは同一の状態（“SDF と結合中”）へ遷移する。その応答は、連鎖オペレーションの結果、あるいはエラーであるかも知れない。

SDSM-ChI が“SDF と結合中”状態にあり、 ＳＣＦから受信したオペレーションを連鎖することが必要である場合、内部イベントが生じる。このイベントは(e8)SDF への要求と呼ばれ、全く同じ“SDF と結合中”状態への遷移を引き起こし、SDSM-ChI は終端側ＳＤＦからの応答を待つ。以前に終端側ＳＤＦに対して発行された連鎖オペレーションに対する応答の受信((E7)SDF からの応答)を受けて、SDSM-ChI は全く同じ“SDF と結合中”状態へ遷移する。終端側ＳＤＦからの応答は、連鎖オペレーションの結果、あるいはエラーであるかもしれない。

3.3.7.4.2　応答側エンティティ(ＳＤＦ)

SDF 前条件：

(1) SDSM-ChT:“SDF と結合中”あるいは“結合処理中”

SDF 後条件：

(1) SDSM-ChT:“SDF と結合中”

ＳＤＦは“結合処理中”状態にあり、起動側ＳＤＦからの‘連鎖(Chained)’オペレーションの受信により引き起こされる(E3)SDF からの要求という外部イベントが生ずる。終端側ＳＤＦはＤＳＡ結合オペレーションの実行が成功するまでオペレーションを処理せず、同一状態に留まる。

ＳＤＦは“SDF と結合中”状態にあり、起動側ＳＤＦからの‘連鎖(Chained)’オペレーションの受信により引き起こされる(E7)SDF からの要求という外部イベントが生ずる。終端側ＳＤＦはオペレーションが実行されるのを待つ。

オペレーションのパラメータの規定に従ってオペレーションを実行した後、 終端側ＳＤＦは (e6)SDF への応答によって結果応答を送信する。

3.3.8　シャドウオペレーション手順

3.3.8.1　概要

X.500 の‘シャドウイング(shadowing)’オペレーションにより、２つのＳＤＦ間で情報がコピーされる。シャドウイングオペレーションはコピーされた情報を維持するためにも用いられる。２つのＳＤＦ間の個々のシャドウイング合意において、一方のＳＤＦはコピーされた情報の供給側となり、他方のＳＤＦは消費側となる。

DSA シャドウ結合(DSAShadowBind)オペレーションと DSA シャドウ結合解放(DSAShadowUnbind)オペレーションは、コピーを供給するある期間のはじめと終わりに、協調して動作するＳＤＦによって使用される。シャドウ更新調整(coordinateShadowUpdate)オペレーションは、シャドウの供給側が更新を意図する対象のシャドウイング合意を示すために使用される。シャドウ更新要求(requestShadowUpdate)オペレーションは、シャドウの消費側がシャドウの供給側からの更新を要求するために使用される。シャドウ更新(updateShadow)オペレーションは、シャドウの供給側がコピーされるデータをシャドウの消費側に送信するために起動される。このオペレーションより前に、シャドウ更新調整あるいはシャドウ更新要求のオペレーションが行われなければならない。‘シャドウイング(shadowing)’オペレーションの完全な記述については ITU-T 勧告 X.525 を参照。

3.3.8.1.1　パラメータ

シャドウ結合オペレーションのパラメータは第３編 2.1.2 章に規定されている in-DirectoryBind オペレーションのパラメータと同一である。シャドウ結合解放オペレーションはパラメータを持たない。

シャドウ更新調整オペレーションについては、ITU-T 勧告 X.525 の 11.1 節を参照。

シャドウ更新要求オペレーションについては、ITU-T 勧告 X.525 の 11.2 節を参照。

シャドウ更新オペレーションについては、ITU-T 勧告 X.525 の 11.3 節を参照。

3.3.8.1.2　エラー手順

シャドウイングオペレーションに関連するエラーの一般的な扱いは ITU-T 勧告 X.525 の 13 節に、オペレーションエラーを通知するためのＴＣＡＰサービスは第３編 2.2.4 章に記述されている。

3.3.8.2　シャドウ結合

3.3.8.2.1　供給側起動によるシャドウ結合

3.3.8.2.1.1　マスタエンティティ(ＳＤＦ)

SDF 前条件：

（１）SDSM-ShM:“空き”

SDF 後条件：

（１）SDSM-ShM:“SDF と結合中”(シャドウ結合が成功した場合)

（２）SDSM-ShM:“空き”(シャドウ結合が失敗した場合)

SDSM-ShM が“空き”状態にあり、コピーの供給が必要である場合、内部イベントが生じる。このイベントは(e1)消費側への結合と呼ばれ、“後続要求待ち”状態への遷移を引き起こし、他のオペレーションは待たされる。アプリケーションプロセスがデリミタによって結合の送信を指示しない限り、SDSM-ShM は“後続要求待ち”状態に留まり、オペレーションは送信されない。デリミタが受信されると(e3)結合および要求送信という内部イベントを経て“結合結果待ち”状態へ遷移する。オペレーションは他のＳＤＦ(消費側ＳＤＦ)に送信される。SDSM-ShM は消費側ＳＤＦからの応答を待つ。‘シャドウ結合’オペレーションが成功した場合は、以前に消費側ＳＤＦに対して発行された‘シャドウ結合’に対する応答の受信（(E5)SDF 結合成功）により、ＳＤＦの状態は“SDF と結合中”という状態へ遷移する。その他の場合は、SDSM-ShM はエラーの受信（(E4)SDF 結合エラー）により“空き”状態へ戻る。

3.3.8.2.1.2　消費側エンティティ(ＳＤＦ)

SDF 前条件：

（1）SDSM-ShC:“空き”

SDF 後条件：

（1）SDSM-ShC:“SDF と結合中”(シャドウ結合が成功した場合)

（2）SDSM-ShC:“空き”(シャドウ結合が失敗した場合)

ＳＤＦは最初“空き”状態にある。ＳＤＦ(供給側ＳＤＦ)からの‘シャドウ結合’オペレーションの受信により引き起こされる(E1)供給側からの結合という外部イベントの受信後、“結合結果待ち”状態へ遷移する。ＳＤＦは dsaShadowBind argument の内容に従って‘シャドウ結合’オペレーションを実行する。ＳＤＦが‘シャドウ結合’オペレーションを完了させると、結果応答あるいはエラーが供給側ＳＤＦに対して返される。ＳＤＦは‘シャドウ結合’が失敗した場合には“空き”状態に戻り、成功した場合には“SDF と結合中”状態へ遷移する。

3.3.8.2.2　消費側起動によるシャドウ結合

3.3.8.2.2.1　マスタエンティティ(ＳＤＦ)

SDF 前条件：

（1）SDSM-ShM:“空き”

SDF 後条件：

（1）SDSM-ShM:“SDF と結合中”(シャドウ結合が成功した場合)

（2）SDSM-ShM:“空き”(シャドウ結合が失敗した場合)

ＳＤＦは最初“空き”状態にある。ＳＤＦ(消費側ＳＤＦ)からの‘シャドウ結合’オペレーションの受信により引き起こされる(E1)消費側からの結合という外部イベントの受信後、“結合結果待ち”状態へ遷移する。ＳＤＦは dsaShadowBind argument の内容に従って‘シャドウ結合’オペレーションを実行する。ＳＤＦが‘シャドウ結合’オペレーションを完了させると、結果応答あるいはエラーが消費側ＳＤＦに対して返される。ＳＤＦは‘シャドウ結合’が失敗した場合には“空き”状態に戻り、成功した場合には“SDF と結合中”状態へ遷移する。

3.3.8.2.2.2　消費側エンティティ(ＳＤＦ)

SDF 前条件：

（1）SDSM-ShC:“空き”

SDF 後条件：

（1）SDSM-ShM:“SDF と結合中”(シャドウ結合が成功した場合)

（2）SDSM-ShM:“空き”(シャドウ結合が失敗した場合)

SDSM-ShC が“空き”状態にあり、更新の要求が必要である場合、内部イベントが生じる。このイベントは(e1)供給側への結合と呼ばれ、“後続要求待ち”状態への遷移を引き起こし、他のオペレーションは待たされる。アプリケーションプロセスがデリミタによって結合の送信を指示しない限り、SDSM-ShC は“後続要求待ち”状態に留まり、オペレーションは送信されない。デリミタが受信されると(e3)結合および要求送信という内部イベントを経て“結合結果待ち”状態へ遷移する。オペレーションは他のＳＤＦ(供給側ＳＤＦ)に送信される。SDSM-ShC は供給側ＳＤＦからの応答を待つ。‘シャドウ結合’オペレーションが成功した場合は、以前に供給側ＳＤＦに対して発行された‘シャドウ結合’に対する応答の受信（(E5)SDF 結合成功）により、ＳＤＦの状態は“SDF と結合中”という状態へ遷移する。その他の場合は、SDSM-ShC はエラーの受信（(E4)SDF 結合エラー）により“空き”状態へ戻る。

3.3.8.3　シャドウ結合解放

3.3.8.3.1　供給側起動によるシャドウ結合解放

3.3.8.3.1.1　マスタエンティティ(ＳＤＦ)

SDF 前条件：

（1）SDSM-ShM:“SDF と結合中”

（2）SDSM-ShM:“調整結果待ち”

（3）SDSM-ShM:“更新待ち”

（4）SDSM-ShM:“更新確認待ち”

SDF 後条件：

（1）SDSM-ShM:“空き”

SDSM-ShM は消費側ＳＤＦに対して既に‘シャドウ結合’オペレーションを開始しており、“SDF と結合中”、“調整結果待ち”、“更新待ち”あるいは“更新確認待ち”のいずれかの状態にある。ＳＤＦはＳＤＦ間に確立された認証済みのアクセスが終了すべきことを決定すると（例、ユーザの解放手順）、‘シャドウ結合解放’オペレーション((e6)、(e8)、(e11)あるいは(e13)SDF 結合解放)を発行し、SDSM-ShM は“空き”状態に遷移する。

3.3.8.3.1.2　消費側エンティティ(ＳＤＦ)

SDF 前条件：

（1）SDSM-ShC:“SDF と結合中”

（2）SDSM-ShC:“調整結果待ち”

（3）SDSM-ShC:“更新待ち”

（4）SDSM-ShC:“更新確認待ち”

SDF 後条件：

（1）SDSM-ShC:“空き”

‘シャドウ結合’オペレーションが既に発行され、SDSM-ShC は“SDF と結合中”、“調整結果待ち”、“更新待ち”あるいは“更新確認待ち”のいずれかの状態で供給側ＳＤＦからの要求を待っているか、またはオペレーションを実行している。‘シャドウ結合解放’オペレーションを受信すると、SDF 結合解放((E5)、(E7)、(E10)あるいは(E12))により“空き”状態に遷移する。

3.3.8.3.2　消費側起動によるシャドウ結合解放

3.3.8.3.2.1　マスタエンティティ(ＳＤＦ)

SDF 前条件：

（1）SDSM-ShM:“SDF と結合中”

（2）SDSM-ShM:“シャドウ要求結果待ち”

（3）SDSM-ShM:“更新待ち”

（4）SDSM-ShM:“更新確認待ち”

SDF 後条件：

（1）SDSM-ShM:“空き”

‘シャドウ結合’オペレーションが既に発行され、SDSM-ShM は“SDF と結合中”、“シャドウ要求結果待ち”、“更新待ち”あるいは“更新確認待ち”のいずれかの状態で消費側ＳＤＦからの要求/応答を待っているか、またはオペレーションを実行している。‘シャドウ結合解放’オペレーションを受信すると、SDF 結合解放((E5)、(E7)、(E10)あるいは(E13))により“空き”状態に遷移する。

3.3.8.3.2.2　消費側エンティティ(ＳＤＦ)

SDF 前条件：

（1）SDSM-ShM:“SDF と結合中”

（2）SDSM-ShM:“シャドウ要求結果待ち”

（3）SDSM-ShM:“更新待ち”

（4）SDSM-ShM:“更新確認待ち”

SDF 後条件：

（1）SDSM-ShM:“空き”

SDSM-ShC は供給側ＳＤＦに対して既に‘シャドウ結合’オペレーションを開始しており、“SDF と結合中”、“シャドウ要求結果待ち”、“更新待ち”あるいは“更新確認待ち”のいずれかの状態にある。ＳＤＦはＳＤＦ間に確立された認証済みのアクセスが終了すべきことを決定すると（例、ユーザの解放手順）、‘シャドウ結合解放’オペレーション((e6)、(e8)、(e11)あるいは(e13)SDF 結合解放)を発行し、SDSM-ShC は“空き”状態に遷移する。

3.3.8.4　シャドウ更新調整

3.3.8.4.1　マスタエンティティ(ＳＤＦ)

SDF 前条件：

（1）SDSM-ShM:“SDF と結合中”

SDF 後条件：

（1）SDSM-ShM:“更新待ち”（シャドウ更新調整が成功した場合）

（2）SDSM-ShM:“SDF と結合中”（シャドウ更新調整が失敗した場合）

SDSM-ShM が“SDF と結合中”状態にあり、シャドウの調整が必要である場合、内部イベントが生じる。このイベントは(e7)消費側へのシャドウ調整と呼ばれ、“調整結果待ち”状態への遷移を引き起こし、消費側ＳＤＦへオペレーションが送信される。SDSM-ShM は消費側ＳＤＦからの応答を待つ。‘シャドウ更新調整’オペレーションが成功した場合は、以前に消費側ＳＤＦに対して発行された‘シャドウ更新調整’に対する応答の受信（(E9)シャドウ調整確認）により、ＳＤＦの状態は“更新待ち”という状態へ遷移する。その他の場合は、SDSM-ShM はエラーの受信（(E10)調整失敗）により“SDF と結合中”状態へ戻る。

3.3.8.4.2　消費側エンティティ(ＳＤＦ)

SDF 前条件：

（1）SDSM-ShC:“SDF と結合中”

SDF 後条件：

（1）SDSM-ShC:“更新待ち”（シャドウ更新調整が成功した場合）

（2）SDSM-ShC:“SDF と結合中”（シャドウ更新調整が失敗した場合）

ＳＤＦは最初“SDF と結合中”状態にある。ＳＤＦ(供給側ＳＤＦ)からの‘シャドウ更新調整’オペレーションの受信により引き起こされる(E6)供給側からのシャドウ調整という外部イベントの受信後、“調整結果待ち”状態へ遷移する。ＳＤＦは coordinateShadowUpdate argument の内容に従って‘シャドウ更新調整’オペレーションを実行する。ＳＤＦが‘シャドウ更新調整’オペレーションを完了させると、結果応答あるいはエラーが供給側ＳＤＦに対して返される。ＳＤＦは‘シャドウ更新調整’が失敗した場合には“SDF と結合中”状態に戻り、成功した場合には“更新待ち”状態へ遷移する。

3.3.8.5　シャドウ更新要求

3.3.8.5.1　マスタエンティティ(ＳＤＦ)

SDF 前条件：

（1）SDSM-ShM:“SDF と結合中”

SDF 後条件：

（1）SDSM-ShM:“更新待ち”（シャドウ更新要求が成功した場合）

（2）SDSM-ShM:“SDF と結合中”（シャドウ更新要求が失敗した場合）

ＳＤＦは最初“SDF と結合中”状態にある。ＳＤＦ(消費側ＳＤＦ)からの‘シャドウ更新要求’オペレーションの受信により引き起こされる(E6)消費側からのシャドウ要求という外部イベントの受信後、“シャドウ要求結果待ち”状態へ遷移する。ＳＤＦは requestShadowUpdate argument の内容に従って‘シャドウ更新要求’オペレーションを実行する。ＳＤＦが‘シャドウ更新要求’オペレーションを完了させると、結果応答あるいはエラーが消費側ＳＤＦに対して返される。ＳＤＦは‘シャドウ更新要求’が失敗した場合には“SDF と結合中”状態に戻り、成功した場合には“更新待ち”状態へ遷移する。

3.3.8.5.2　消費側エンティティ(ＳＤＦ)

SDF 前条件：

（1）SDSM-ShC:“SDF と結合中”

SDF 後条件：

（1）SDSM-ShC:“更新待ち”（シャドウ更新要求が成功した場合）

（2）SDSM-ShC:“SDF と結合中”（シャドウ更新要求が失敗した場合）

SDSM-ShC が“SDF と結合中”状態にあり、シャドウの更新の要求が必要である場合、内部イベントが生じる。このイベントは(e7)供給側へのシャドウ要求と呼ばれ、“シャドウ要求結果待ち”状態への遷移を引き起こし、供給側ＳＤＦへオペレーションが送信される。SDSM-ShC は供給側ＳＤＦからの応答を待つ。‘シャドウ更新要求’オペレーションが成功した場合は、以前に供給側ＳＤＦに対して発行された‘シャドウ更新要求’に対する応答の受信（(E10)シャドウ要求結果）により、ＳＤＦの状態は“更新待ち”という状態へ遷移する。その他の場合は、SDSM-ShC はエラーの受信（(E9)シャドウ要求失敗）により“SDF と結合中”状態へ戻る。

3.3.8.6　シャドウ更新

3.3.8.6.1　供給側起動によるシャドウ更新

3.3.8.6.1.1　供給側エンティティ(ＳＤＦ)

SDF 前条件：

（1）SDSM-ShM:“更新待ち”

SDF 後条件：

（1）SDSM-ShM:“更新待ち”

SDSM-ShM が“更新待ち”状態にあり、シャドウの更新が必要である場合、内部イベントが生じる。このイベントは(e12)消費側へのシャドウ更新と呼ばれ、“更新確認待ち”状態への遷移を引き起こし、消費側ＳＤＦへオペレーションが送信される。SDSM-ShM は消費側ＳＤＦからの応答を待つ。以前に消費側ＳＤＦに対して発行された‘シャドウ更新’に対する応答の受信（(E14)シャドウ更新確認）により、ＳＤＦの状態は“更新待ち”という状態に遷移する。消費側ＳＤＦからの応答は、シャドウ更新オペレーションの結果、あるいはエラーであるかもしれない。

3.3.8.6.1.2　消費側エンティティ(ＳＤＦ)

SDF 前条件：

（1）SDSM-ShC:“更新待ち”

SDF 後条件：

（1）SDSM-ShC:“更新待ち”

ＳＤＦは最初“更新待ち”状態にある。ＳＤＦ(供給側ＳＤＦ)からの‘シャドウ更新’オペレーションの受信により引き起こされる(E11)供給側からのシャドウの更新という外部イベントの受信後、“更新確認待ち”状態へ遷移する。ＳＤＦは updateShadow argument の内容に従って‘シャドウ更新’オペレーションを実行する。ＳＤＦが‘シャドウ更新’オペレーションを完了させると、結果応答あるいはエラーが供給側ＳＤＦに対して返される。ＳＤＦは“更新待ち”状態へ戻る。

3.3.8.6.2　消費側起動によるシャドウ更新

3.3.8.6.2.1　供給側エンティティ(ＳＤＦ)

SDF 前条件：

（1）SDSM-ShM:“更新待ち”

SDF 後条件：

（1）SDSM-ShM:“更新待ち”

SDSM-ShM が“更新待ち”状態にあり、シャドウの更新が必要である場合、内部イベントが生じる。このイベントは(e10)消費側へのシャドウ更新と呼ばれ、“更新確認待ち”状態への遷移を引き起こし、消費側ＳＤＦへオペレーションが送信される。SDSM-ShM は消費側ＳＤＦからの応答を待つ。以前に消費側ＳＤＦに対して発行された‘シャドウ更新’に対する応答の受信（(E12)シャドウ更新確認）により、ＳＤＦの状態は“更新待ち”という状態に遷移する。消費側ＳＤＦからの応答は、シャドウ更新オペレーションの結果、あるいはエラーであるかもしれない。

3.3.8.6.2.2　消費側エンティティ(ＳＤＦ)

SDF 前条件：

（1）SDSM-ShC:“更新待ち”

SDF 後条件：

（1）SDSM-ShC:“更新待ち”

ＳＤＦは最初“更新待ち”状態にある。ＳＤＦ(供給側ＳＤＦ)からの‘シャドウ更新’オペレーションの受信により引き起こされる(E12)供給側からのシャドウ更新という外部イベントの受信後、“更新確認待ち”状態へ遷移する。ＳＤＦは updateShadow argument の内容に従って‘シャドウ更新’オペレーションを実行する。ＳＤＦが‘シャドウ更新’オペレーションを完了させると、結果応答あるいはエラーが供給側ＳＤＦに対して返される。ＳＤＦは“更新待ち”状態へ戻る。

付属資料Ａ　INの定義で用いられる用語

アソシエーション：ある機能を実行するうえで動作するエンティティ間の論理的な関連。

アプリケーションエンティティ（ＡＥ）：システムに依存しないアプリケーション動作であり、アプリケーションエージェントによってアプリケーションサービスとして利用できる。例えば、１つのアプリケーションプロセスの通信面の一部或はすべてを担うアプリケーションサービス要素（ＡＳＥ）の一式がこれにあたる。

アプリケーションコンテキスト（ＡＣ）：１つのアプリケーションコンテキストはある特定の通信インスタンスに使われる機能を規定する。

アプリケーションサービス要素（ＡＳＥ）：例えばアプリケーションエンティティ間などのアプリケーション通信を補助する統合機能の集合。

アプリケーションプロトコルデータ単位（ＡＰＤＵ）：アプリケーションプロトコルで指定するアプリケーションデータの単位で、アプリケーションプロトコルの制御情報とアプリケーションプロトコルのユーザデータから成る。

インテリジェントネットワーク（ＩＮ）：顧客管理下のものも含めた新しい能力やサービスを導入するための設備を柔軟に提供することができるテレコミュニケーションネットワークのアーキテクチャ。

インテリジェントネットワークアプリケーションプロトコル（ＩＮＡＰ）：（ＯＳＩモデルの応用層にあたる）第７層に含まれるインテリジェントネットワークのアプリケーションのためのプロトコル。

ＩＮ概念モデル（ＩＮＣＭ）：インテリジェントネットワークアーキテクチャを定義するための立案モデル。

エンティティ：部品やデバイス、サブシステム、機能ユニット、装置、システムで、個々に存在しうるもの。ＩＳＤＮ用語では、ユーザ端末やデジタル交換機などのある特定のシステムやサブシステムを指し示す時に用いられる。また、あるところの特定のシステムの機能の組み合わせで、例えばユーザ端末で信号システムのもつ第２層機能を参照するのに使われる。

オブジェクト：属性と機能において適切なレベルで抽象化され記述されたあるエンティティに固有なコンポーネント。

管理機能：エンティティ管理のために用いられる一式のプロセス（例、オペレーション継続、統括、保守とプロビジョニングを網羅するデータベース管理機能）。

機能エンティティ：与えられたところのある機能一式を構成するエンティティ。
機能エンティティ［アプリケーションを提供する通信サービスにおける］：ある１ヶ所で機能を提供するサービスの組み合わせで、サービスを提供する上で必要となるすべての機能のサブセット。

結合：認証のための Association Control 中に用いられるメカニズム。ＩＴＵ－Ｔ勧告Ｘ．５００を参照。

結合解放：認証のために Application Control 制御の最中に用いられるメカニズム。ＩＴＵ－Ｔ勧告Ｘ．５００を参照。

コール：２者以上のユーザやサービスの間で確立する１つ以上のコネクションを使用する、或は使用するかもしれない。

サービス生成：補助サービスを提供する能力により、設計から開発、検証まで行う活動。

サービス制御機能（ＳＣＦ）：インテリジェントネットワークにより提供されるサービスにおいて、機能エンティティを制御するためのサービス論理の応用。

サービス制御局（ＳＣＰ）：サービス制御機能を実装するインテリジェントネットワーク内のエンティティ。

サービスデータ機能（ＳＤＦ）：サービスデータのテンプレートに準じてサービスデータ管理を行う機能一式。

サービスデータ局（ＳＤＰ）：サービスデータ機能を実装する物理エンティティ。

サービス論理（ＳＬ）：ある特定のサービスを提供する為に用いられる一連のプロセス／機能。

サービス論理処理（ＳＬＰ）：プログラムサービス論理を含むソフトウエアプログラム。

サービス論理処理プログラムインスタンス（ＳＬＰＩ）：特定の発呼／サービス実施試行に対するサービス或いはサービスフィーチャを提供する特定のサービス論理プログラムの起動と応用。

識別名（ＤＮ）：ＴＴＣ標準ＪＴ－Ｘ５００（４．５章） を参照。

情報フロー：一対の機能エンティティ間の通信で行われるやりとり。

属性：ＴＴＣ標準ＪＴ－Ｘ５００（４．４章）を参照。

ダイアログ：会話や情報交換。

単一アソシエーションオブジェクト（ＳＡＯ）：単一の Application Association を通じて相手と通信するために必要となる機能。

単一アソシエーション制御機能（ＳＡＣＦ）：単一の Application Association を通じて相手と通信するために用いられるＡＳＥの利用を制御する規則と規制。

ディレクトリ：ＴＴＣ標準ＪＴ－Ｘ５００（４．２章）を参照。

ディレクトリアクセスプロトコル（ＤＡＰ）：ＩＴＵ－Ｔ勧告Ｘ．５００（１２章）を参照。

ディレクトリエントリ（ＤＥ）：ＴＴＣ標準ＪＴ－Ｘ５００（４．３章）を参照。

ディレクトリシステムプロトコル（ＤＳＰ）： ＩＴＵ－Ｔ勧告Ｘ．５００（１２章）を参照。

ディレクトリユーザエージェント：ＴＴＣ標準ＪＴ－Ｘ５００（４．２章） を参照。

ディレクトリ情報木（ＤＩＴ）：ＴＴＣ標準ＪＴ－Ｘ５００（４．３章） を参照。

トランザクション機能応用部（ＴＣＡＰ）：コンポーネントサブレイヤーとトランザクションサブレイヤーで構成されるＯＳＩモデルの第７層における応用層のサービスとプロトコル（ITU-T Q.771－4）。

能力セット（ＣＳ）：標準化活動が目的としているインテリジェントネットワークの能力の組み合わせで、ある期間内に勧告を標準化することを目指している。

物理プレーン：機能エンティティを実現する要素とそれら要素間インターフェースを含むインテリジェントネットワーク概念モデル上のプレーンの１つ。

分散機能プレーン（ＤＦＰ）：機能エンティティとその関係を含むインテリジェントネットワーク概念モデル上のプレーンの１つ。

付録Ⅰ：　サービスデータモデリング

この付録は，サービスデータモデルとその2.2.2.3項で定義されている情報モデルへのマッピングについての必要な段階規則とその例について概説する。

これらの規則と例は単純なメカニズムを提供しており、本付録のⅠ.1.1項とⅠ.2.1項で示される。

付加的なツール（例えば属性コンテキスト仕様）はデータモデリングに解を与え得るかもしれないが、本付録においては、これらのツールについては言及しない。

Ⅰ.1.　サービスデータモデリングの必要性の整理

SCF-SDF間のインタフェースに対してX.511のオペレーションを用いるとともに、SDF内のサービスデータを表わすためのデータモデルとして、X.501の情報モデルを採用することで、INのサービスデータをどのようにしてこの情報モデルに対応させるべきかという問題が明らかになった。

サービスデータからSDF内で使用するX.500の情報モデルへのマッピングが不適当であると、SCF-SDF間で転送しなければならない情報の量が劇的に増加してしまうこともありえる（場合によっては、TCAPリンクの能力を超えてしまうことさえある）。マッピングの選択は高トラヒックに加えて、SCF内の機能エンティティ動作（FEA）によって実施すべきデータ処理の量にも影響を与え、SCFのFEAのインプリメンテーションをサービスデータに依存させてしまうことまでありえる。

本項は、SCF-SDF間インタフェースのトラヒックを最小限にし、サービス非依存にデータを扱うための、サービスデータに適用できる単純なマッピング規則を概観する。

Ⅰ.1.1.　一般的なモデル化の条件

INサービスデータをX.500の情報モデル上で構築する際には、一般的な要求条件が3つある。

これらのうちどの要求も同じ重みを持つ。

・要求1　SCF-SDF間のトラヒックを最小限にする

SCF-SDF間インタフェースは恐らく、IN環境のインタフェースでは最も激しく使用されるものの1つであろう。SCF内の各SLPIは、SDF内のデータに複数回アクセスするかもしれない。処理能力の点から見て、データ転送を最小限に保つことが重要である。

・要求2　SCFで行なうデータ処理を最小限にする

SCF内のデータ処理は、JT-Q1218の第1編で規定されている情報フローを処理するFEAによってなされる。このFEAは実行しているサービスには非依存であることが前提となっている。もし転送されるデータの情報の粒度が悪いと、サービス依存の処理がSCF内のFEAによって実行されるかも知れない。したがって、FEAのサービス非依存性を維持するために、INサービスデータに対するX.500の情報モデルの設計に注意が必要である。

・要求3　X.500のデータ管理におけるデータの分散の効果を考慮する

CS1システムでは、SDFのインプリメンテーションは非分散であることが期待されている。しかしながら、X.500のシステムは本質的に分散データ管理システムであることに注意が必要である。もしDITの構成を設計する際にデータの分散を考慮しないと、結果として生じるディレクトリは、複数のディレクトリサービスエージェント（DSA）にまたがってディレクトリの情報にアクセスする際に必要な「知識」の管理という意味では管理不能となる。集中化したディレクトリのインプリメンテーションでは、分散を考慮する必要がないとは言えるが、処理能力の要求条件からはディレクトリを多数のローカルなDSA上に分割する可能性もあり、その際には各DSA内で分散の知識を管理する必要がある。

Ⅰ.1.2　サービスのモデル化－必要な情報の定義－

サービスのモデル化は Q.1203 で規定されている。これには以下の段階が含まれる。

(1) サービスを構成するサービスフィーチャを特定する。
(2) 必要なサービスフィーチャをサポートするために必要なデータを特定する。
(3) サービスによって使用されるサービスフィーチャ間で共用するデータを特定する。
(4) サービスデータに必要なオペレーションおよび処理を特定する。
(5) サービスデータに対するアクセス制御の要求条件を特定する。

これらの段階は本ドキュメントには含まれない。サービスデータを2.2項で定義されたＳＤＦ情報モデルにマッピングするのに必要な付加的な段階のみが含まれる。

Ⅰ.2.　サービスデータモデルを構築するためのガイドライン

INのサービスデータモデルをサービスフィーチャのデータという意味で定義したら、それをX.500の情報モデルにマッピングする必要がある。これは以下の段階によって実現できる。

(1) 属性（ATTRIBUTE）の選択
(2) オブジェクトクラス（OBJECT-CLASS）の選択
(3) サービスのディレクトリ情報木（DIT）の設計
(4) 属性およびオブジェクトクラスへのアクセス制御の割当

Ⅰ.2.1.　属性の選択

属性の選択は、SCFから要求されるオペレーションおよび処理に影響される。選択を援助するために以下の規則が提供される。

・規則1

属性は、SCFによって処理するためにSCF-SDF間インタフェース上で転送されるINサービスデータの認識可能な最小単位である。属性の定義の中では、構造化されたデータ型は避ける必要がある。

・規則2

ディレクトリ内の属性でINサービスから参照されるすべての属性は、唯一の値を持つことが好ましい。

・規則3

もしサービスデータが複数のカラムからなる情報テーブルを含む場合は（付図I-1/JT-Q1218(ITU-T Q.1218)参照）、テーブルの各カラムは属性の定義をもつ必要がある。

付図I-1A/JT-Q1218(ITU-T Q.1218) （時間によるルーティング）では、3つの属性（開始時刻，終了時刻，着アドレス）が定義されている。また付図I-1B/JT-Q1218(ITU-T Q.1218) （短縮番号）では、2つの属性（短縮番号，着アドレス）が定義されている。

すべての属性を特定した後で同一の定義があった場合（例、付図I-1/JT-Q1218(ITU-T Q.1218)の着アドレス）に、属性の合計数を減らすことも可能である。

開始時刻	終了時刻	着アドレス
01:00	09:00	204 23456
09:01	15:00	204 12345
15:01	23:00	204 23456

A: 時間によるルーティングテーブル

短縮番号	着アドレス
01	204 23456
02	204 12345
03	204 23456

B: 短縮番号テーブル

付図 I-1/JT-Q1218(ITU-T Q.1218)　サービスデータの複数カラムテーブル

Ⅰ.2.1.1. 属性型の定義

属性型は ASN.1 のデータ定義によって定義される。

```
thisAttribute ATTRIBUTE ::= {
        . .
        WITH SYNTAX thisAttributeASN1DataType,
        . .
        }
```

属性は別の属性のサブタイプとしても定義でき、この場合は親の属性の定義に適合しなければならない。

```
thisAttribute ATTRIBUTE ::= {
        . .
        SUBTYPE OF another Attribute,
        . .
        }
```

これは属性の定義を再利用する便利な方法であるが、要求1に対するインパクトも考えられる。すなわち、特定の属性型に対するX.500の探索オペレーションは、要求された属性値だけでなく、要求された属性のサブタイプである属性の値をも返送するからである。

特定の属性に対する探索では、いかなる親の属性値も返送しないということには注意が必要である。 属性は集合的な属性(Collective Attribute)としても定義され得る。これは、値が多くのディレクトリエントリで共有され得ることを示唆している。

```
collectiveAttribute ATTRIBUTE ::= {
        . .
        COLLECTIVE                  TRUE,
        . .
        }
```

既定義の属性の例は、X.520の２章に見られる。

Ｉ.2.1.2. 許容された値(Permitted Values)の定義

許容された値を属性に定義するために２つの方法がある。

(1) 値を特定の値のセットに制限する。

(2) 値を数値の範囲に制限する。

最初のケースでは２つの追加属性が要求される。

(a) 現在の属性に対して許容された値を保持するための属性

(b) ２つの属性をリンクするアクセス制御属性。この属性は、AC 項目のシンタックスと、以下に示すProtectedItem フィールドを含む必要がある。

```
ProtectedItem ::= SEQUENCE {
        . .
        restrictedBy      [10]        SET OF RestrictedValue OPTIONAL,
        . .
        }
```

制限された値(Restricted Value)は現在の属性の属性識別子を含み、その属性は許容された値を含む。制限された値(Restricted Value)はアクセス制御属性の一部であるから、制限された値(Restricted Value)を変更する能力は、アクセス制御属性中に含まれる現在のエントリの全てのアクセス制御パラメータを変更する能力を示している。

２番目のケースでは１つの付加的な属性のみが要求される。

(c) 現在の属性の値の範囲を保持するアクセス制御属性。この属性は ACIItem のシンタクスを持ち、以下に示す ProtectedItem フィールドを含む必要がある。

```
ProtectedItem ::= SEQUENCE {
        . .
        rangeOfValues     [7]          Filter OPTIONAL,
        . .
        }
```

フィルタ(Filter)は属性の値の範囲の仕様を含む。値の範囲の一部を変更するためには、フィルタを置き換える必要があるということには注意が必要である。また、フィルタの値はアクセス制御属性の一部であるから、フィルタの値を変更する能力は、アクセス制御属性に含まれる現在のエントリの全てのアクセス制御パラメータを変更する能力を示している。

Ｉ.2.1.3. デフォルト値（Default Values）の定義

属性型（attribute type)のデフォルト値は、実行時に、フォールバックフィールドを真(TRUE)に設定したコンテキストを持つ属性値として規程することにより設定される。

コンテキストのフォールバックが真(TRUE)に設定された属性値は、他の属性値で基準（criteria）に適合するものがない場合に限り返送される。

Ｉ.2.1.4. 最大属性数（MAX Attribute Values）の定義

属性が持つ値の数を制限することは、以下に示すProtectedItemを含むアクセス制御フィールドを規定することによってなされる。

```
ProtectedItem ::= SEQUENCE{
        . .
        maxValueCount     [8]          SET OF MaxValueCount OPTIONAL,
        . .
        }
```

最大値カウンタ(MaxValueCount)は属性Idと属性に蓄積される最大数を含んでいる。最大値カウンタはアクセス制御属性の一部であるから、最大値カウンタの値を変更する能力は、アクセス制御属性の現在のエントリの全てのアクセス制御パラメータを変更する能力を示している。

Ｉ.2.1.5. 属性寿命（Attribute Lifetimes）の定義

属性寿命（Attribute Lifetimes)は属性値に一時的コンテキスト(TemporalContext)を与えることにより設定される（拡張時）。

Ⅰ.2.1.6. 照合規則（ MATCHING-RULES）の定義

照合規則は、フィルタの探索や名前照合に使用される外部の値に対して属性値がどのようにテストされるかを定義している。１つの属性定義は３つの異なる型の照合規則を持つ。

等価(EQUALITY)　　テスト用（値＝＝属性値）
順序(ORDERING)　　テスト用（値＜属性値）
部分文字列(SUBSTRING)　部分文字列の照合テスト用

this Attribute ATTRIBUTE ::= {
. .
EQUALITY MATCHING RULE equalityMatchingRuleName,
ORDERING MATCHING RULE orderingMatching RuleName,
SUBSTRINGS MATCHING RULE substringsMatchingRuleName,
. .
}

既定義の照合規則の例は、X.520 の 3 節に見られる。

Ⅰ.2.1.7. 属性のオブジェクト識別子の割当

定義された各属性には唯一のオブジェクト識別子が割り当てられる。このオブジェクト識別子の形式は以下のようになる。

```
-- INオブジェクト識別子プレフィクス
in-oi       OBJECT IDENTIFIER ::=       {ccitt recommendation q 1218 ...}

--  サービス属性の定義
attributeX      OBJECT IDENTIFIER ::=    {in-oi ServiceID attributeType(4) AttributeID}

--  サービスidは属性が属するサービス／サービスフィーチャを定義する
--  属性idはサービス内の属性のidを定義する: これはサービス設計者によって割り当てられる.
```

Ⅰ.2.2. オブジェクトクラスの選択

サービスデータモデル内で用いられる属性が特定されたならば、それらを特定のオブジェクトクラスの要素に纏める（グループ化する）ことができる。

Ⅰ.2.2.1. オブジェクトクラスの定義

オブジェクトクラスは、ディレクトリのエントリの規定を行なう。それは複数の属性を1つのオブジェクトにまとめたものである。DIT内のディレクトリエントリの各インスタンスは、唯一の実体を持つ（仕様についてはX.501の6.2.2.2参照）。オブジェクトクラスの定義を援助するために、以下の規則が用いられる。

・規則 4

共通の唯一の実体を共用する複数のサービスデータの値は、1つのディレクトリエントリ内の属性として纏めるべきである。（例、あるサービスに対するユーザデータ）

・規則 5

複数カラムのテーブルの一部である複数のサービスデータの値（規則3参照）は、1つのオブジェクトクラスの定義に纏めるべきである。

オブジェクトクラスは、別のオブジェクトクラスのサブクラスとして規定できる。この場合、そのオブジェクトクラスは親のオブジェクトクラスのすべての属性および性質を含む。これはC++のインヘリタンスに似ている。

Ⅰ.2.2.2. RDN（名前形式）の選択

各オブジェクトクラスは、可能なRDNの属性として指定された多数の属性を持つことが可能である。これは名前形式（NAME-FORM）の仕様を用いてなされる。ディレクトリエントリが生成されるとき、（エントリ追加オペレーションで定義された識別名を通して）いずれかの属性にアクティブなRDNが指定される必要がある。オブジェクトが生成された後では、アクティブなRDNのみが、アクセスオペレーション（探索，エントリ変更，エントリ削除等）を用いてオブジェクトの位置を特定するための名前の照合手順で使用できる。したがって、同一のオブジェクトクラスの2つのオブジェクトが、同一の属性値を持つが異なるアクティブなRDNを用いて生成された場合、DIT内では2つの異なるオブジェクトとして扱われる。もし同一のRDNが2つめのオブジェクトに対して使われた場合は、オブジェクト生成エラーが生じるはずである。

Ⅰ.2.2.3. オブジェクトクラスのオブジェクト識別子の割当

定義された各オブジェクトクラスには唯一のオブジェクト識別子が割り当てられる。このオブジェクト識別子の形式は以下のようになる。

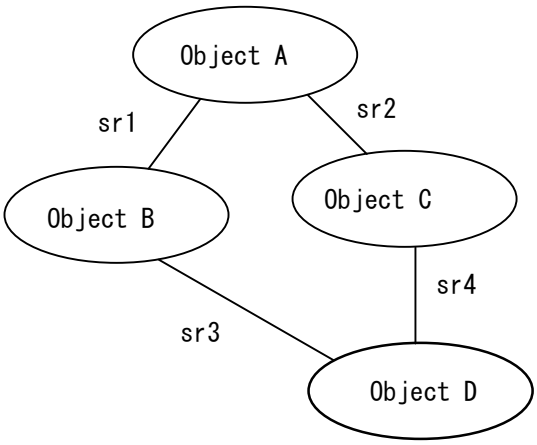
```
-- INオブジェクト識別子プレフィクス
    in-oi       OBJECT IDENTIFIER ::=       {ccitt recommendation q 1218 ...}

-- サービス属性の定義
  objectClassX OBJECT IDENTIFIER ::=   {in-oi ServiceID objectClass(6) ObjectClassID}

-- サービスidは属性が属するサービス／サービスフィーチャを定義する
-- オブジェクトクラスidはサービス内のオブジェクトクラスのidを定義する: これはサービス設計者によって割り当てられる
```

Ⅰ.2.3. DIT の定義

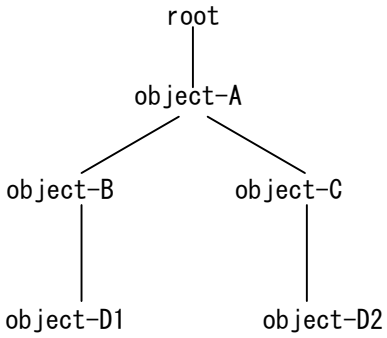
IN サービスデータモデルのための DIT の構築方法を示す。

Ⅰ.2.3.1. オブジェクトハイアラーキ（STRUCTURE-RULES）の定義

DITはデータオブジェクトのハイアラーキを定義する。オブジェクトの各識別名は、そのオブジェクトのRDNとDIT内でそれより上位にあるオブジェクトのRDNとの連結から成る。DIT内のオブジェクトのハイアラーキは構造規則（STRUCTURE-RULES）を用いて定義される。一組のオブジェクトクラスと構造規則によってどのようにDITを構成するかを付図I-2/JT-Q1218(ITU-T Q.1218)に示す。



Ａ：構造規則設計



Ｂ：ＤＩＴ構成

付図 I-2/JT-Q1218(ITU-T Q.1218)：ＤＩＴへの構成規則のマッピング

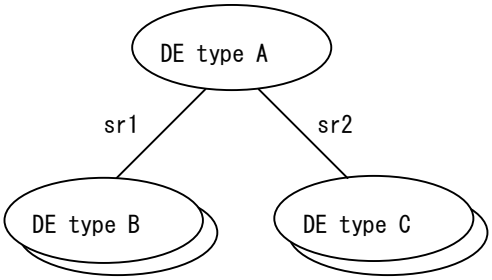
以下の規則が、一組のオブジェクトクラスの仕様から DIT を構築するために用いることができる。

・規則 6

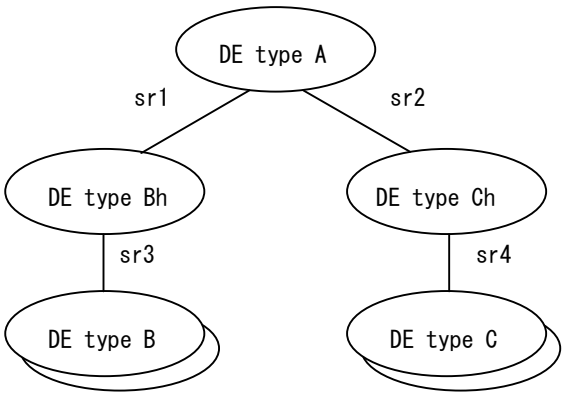
複数のサービスデータの値が共通の構造を持っているが、複数の値の集合を持つかもしれない（すなわち、時刻の翻訳テーブルにおける行，サービスにおけるユーザ固有のデータ）場合、別のディレクトリエントリの下位のディレクトリエントリ内に配置すべきである（すなわち、ディレクトリエントリ間の関係を定義する構造規則が存在する。）

・規則 7

複数のデータエントリの組みが特定のデータエントリの配下に存在する場合、各々の組みは、特定のデータエントリの下位にある単一のデータエントリのさらに下位にあるものとして配置することが推奨される（付図I-3/JT-Q1218(ITU-T Q.1218)参照）。これによって、X.500のオペレーションを用いて組みになっているデータエントリをより効率的に探索できるであろう。



Ａ：設計後



Ｂ：推奨する実現法

付図 I-33-4-8/JT-Q1218(ITU-T Q.1218)：複数の組の推奨モデル構成

Ⅰ.2.3.2. もうひとつの命名パス（Aliases）の定義

オブジェクトが生成されると、その識別名が決まる。もしそのオブジェクトに対して別の命名パスが要求された場合は、別名オブジェクトがDIT内で定義される必要がある。この別名オブジェクトは、それ自身のRDN属性を持ち、参照されるべきオブジェクトへのポインタも含んでいる。別名オブジェクトは、それ以前に定義されている別名(alias)オブジェクトクラスのサブタイプであるオブジェクトクラスである。

Ⅰ.2.3.3. サブエントリの定義

エントリが持つであろうサブエントリの数を制限することは、以下に示すシンタクスを持つProtectedItemフィールドを含むエントリ中のアクセス制御属性で規定することによってなされる。

```
ProtectedItem     ::= SEQUENCE {
                  . .
                  maxImmSub       [8]             INTEGER OPTIONAL,
                  . .
                  }
```

maxImmSubの値は、現在のエントリの下に蓄積されるサブエントリの最大数を含む。この制限は現在のDSAに対してのみ適用されることに注意が必要である。maxImmSubの値はアクセス制御属性の一部であるから、maxImmSubの値を変更する能力は、アクセス制御属性に含まれる現在のエントリの全てのアクセス制御パラメータを変更する能力を示している。

Ⅰ.3. サービス例（UPT ライクなサービス）

この節では、マッピング規則を用いて、UPT サービスの単純なサブセットのための X.500 DIT を実現する。

Ⅰ.3.1. サービス例の内容記述

ここで扱うサービス例は以下に示すサービスフィーチャを持つ。

（１） 入接続（Incall)
（２） 入接続登録（Incall Registration)
（３） 入接続スクリーニング（Incall Screening)
（４） 出接続（Outcall)
（５） 出接続（位置によるスクリーニング） (Outcall Screening by location)

これらのフィーチャについては、以下にサービス設計段階（Ⅰ.1.2参照）後に使用できるようになるデータ情報とともに記述してある。それぞれのサービスフィーチャについての情報は以下に示すコラムを持つテーブルの形で表現されている。

データ	：データ item の名前
記述	：データ item 型の定義(ASN.1 記述)
基本性(Cardinality)	：データ item についてどれほどの異なるバージョンが起こり得るかを定義する。（例えば、各々のサービスユーザに対して１つ）
フィーチャオペレーション	：データ item を操作するのにどのようなオペレーション（SCF-SDF インタフェース）が使用されるかを定義する。
アクセス制御	：異なるレベルのサービスオペレータでどのようなレベルのアクセス の制御が要求されるかを定義する。

注意：この例は完全なサービスを意図したものではない。SDF情報モデルへのマッピングがどのようになされるかを示すのに十分なだけである。

Ｉ.3.1.1. サービスフィーチャ：入接続

通常の入接処理は以下に示すルーチングアルゴリズムを実行する。

（１） ユーザがアクティブな登録位置を持っていればその登録位置に呼を送る。

（２） アクティブ time-of-day 位置がアクティブであれば、time-of-day 位置に方路をとる。

（３） ユーザがデフォールトの位置を持っていればデフォールトの位置に方路をとる。

（４） ”ユーザは使用できない”というアナウンスを行うとともに、呼を拒否する。

全ての定義された位置は、ユーザにとっては、既定義の位置の組みの１つであるものとする。

サービスフィーチャ情報

データ	定義	基本性	フィーチャ オペレーション	セキュリティ アクセス制御
Registered Locations	SET OF Location	one per service user	none	service: R user management: R/W user: R

Time Of Day Location	SEQUENCE {
	StartTime SEQUENCE{
	Day
	DayOfWeek,
	Time TimeOfDay },
	StopTime SEQUENCE{
	day
	DayOfWeek,
	Time TimeOfDay },
	Location Location}
		many per service user	search	service: R
				user management: R/W
				user: R

Current Location	Location	one per user	search	service: R user R/W
Current Location Expiry	TimeOfDay	one per user	none	service: R user R/W
Default Expiry	TimeOfDay	one per user	modifyEntry	service: R user management: R/W user: R
Default Location	Location	one per user	search	service: R user management: R/W user: R

Ｉ.3.1.2. サービスフィーチャ：入接続登録

本サービスでは、ユーザは現在の位置を登録できる。現在位置は現在の端末(CLIにより識別）でも、定義した位置でもよい。ユーザは現在位置での満了時間をオプションで設定可能である。満了時間の設定がない場合、デフォールト満了時間が使用される。現在位置は、既定義の位置の組みのうちのいずれかでなければならない。

サービスフィーチャ情報

データ	定義	基本性	フィーチャオペレーション	セキュリティアクセス制御	
Registered Locations	SET OF Location	one per service user	none	Service: User management: user:	R R/W R
Current Location	Location	one per user	modifyEntry	Service: User	R R/W
Current Location Expiry	TimeOfDay	one per user	modifyEntry	Service: User	R R/W
Default Expiry	TimeOfDay	one per user	none	Service: User management: user:	R R/W R

Ｉ.3.1.3. サービスフィーチャ：入接続スクリーニング

本サービスは、入接続呼が特定の位置からのものである場合に拒否することを許容する。スクリーニングに使用可能な位置の数には制限がある。

サービスフィーチャ情報

データ	定義	基本性	フィーチャオペレーション	セキュリティアクセス制御	
Incall Screen List	SET OF Location SIZE(1..MaxlistSize)	one per service user	search	Service: User management: user:	R R/W R
MaxListSize	INTEGER	one per service user	none	Service: User	R R/W

Ｉ.3.1.4. サービスフィーチャ：出接続

本サービスでは、ユーザが端末の位置から呼を発生させることを可能にする。この位置はそのユーザに対して許容されたうちの一つでなければならない。

その呼に対しての課金はそのサービスのユーザに対してなされる。端末の所持者（オーナ）に対してではない。

サービスフィーチャ情報

データ	定義	基本性	フィーチャ オペレーション	セキュリティ アクセス制御
Registered Locations	SET OF Location	one per service user	Search	Service: R User management: R/W user: R
User Id	ServiceNumber	one per service user	match/bind	Service mgt.: R/W Service: R user mgt.: R user: R
User Password	Password	one per service user	match/bind	Service: R user mgt.: R/W user: R

Ｉ.3.1.5. サービスフィーチャ：出接続（位置によるスクリーニング）

本サービスフィーチャは、特定の位置への出接続の進行を制限する。規定される位置は、ユーザの使用するそれぞれの位置により異なる。

サービスフィーチャ情報

データ	定義	基本性	フィーチャ オペレーション	セキュリティ アクセス制御

Outcall Screen List	Set Of {
	SEQUENCE{
	stop SET OF
	Location,
	location Location
	}}
		one per service user	Search	User management: R/W
				user: R

Ｉ.3.1.6. 一般的なサービス情報

サービスフィーチャはインタラクションのために以下に示す関係を持つ。

- 入接続のスクリーニングは入接続の前になされる。
- 出接続は位置による出接続スクリーニングの前になされる。

付加的なサービス情報

以下に示す付加的なサービスパラメータが提供される。

データ	定義	基本性	フィーチャ オペレーション	セキュリティ アクセス制御
Service Key	ServiceId	one per service	match/bind	Service mgt.: R/W Service: R user mgt.: R user: R
User Name	String	one per service user	match	user mgt.: R/W user: R

Ｉ.3.2. サービスデータモデリングの例

これまででサービスデータ情報が定義された。従って、このデータのＩ.2節で定義した段階を用い、情報モデルへのマッピングに着手できる。

Ｉ.3.2.1. 属性の選択

最初のステップはサービスデータに要求される属性の定義を決定することである。属性識別子を定義する目的のため、以下に示すオブジェクト識別子が用いられる。

exampleAttribute OBJECT IDENTIFIER ::={in0oi exampleService(1)attributeType(4)}

Data: Registered Locations

登録位置（Registered Location)はその位置においてユーザが入接続の呼を受ける位置を蓄積する。

Registered Locations	SET OF Location	one per service user	match	service: user mgt.: user:	R R/W R

この例での目的のために、位置の定義は電話番号と等しいと考えてもよい。そうすることで、X.520(電話番号）で既定義の属性の位置の型（Location Type)を用いるサービス属性についてSUBTYPE定義を用いることが可能となる。従って、登録位置属性は以下のようになる。

```
regLocations
        SUBTYPE OF telephoneNumber,
        COLLECTIVE TRUE,
        ID              in-at-regLocations}
```

この属性は集合的な属性(Collective Attribute)として定義される。何故ならば、その値はDIT(Ｉ.3.3.2参照）中の多くのエントリで共有されるためである。

Data: Time of Day Location

Time of Day Locationは実際は５コラム(startTime.day,startTime.time,stopTime.day, stopTime.time,location)からなる翻訳テーブルである。

Time Of Day Location	SEQUENCE {
	StartTime SEQUENCE{
	day
	DayOfWeek,
	time TimeOfDay },
	stopTime SEQUENCE{
	day
	DayOfWeek,
	time TimeOfDay },
	location Location}
		many per service user	search	service:
				user management:
				user:
					R
					R/W
					R

規則３によると、５つの異なる属性が、各々のコラムに対して一つ、定義される。

startDay

WITH SYNTAX daysOfWeek -from proposed X.520 Temporal Context
SINGLE VALUE TRUE
ID in-at-start-Day}

startTime

WITH SYNTAX DayTime-from proposed X.520 Temporal Context
EQUALITY MATCHING RULE numericStringMatch -from X.520
ORDERING MATCHING RULE numericStringOrderingMatch -from X.520
SUBSTRINGS MATCHING RULE numricStringSubstringsMatch -from X.520
SINGLE VALUE TRUE
ID in-at-startTime}

stopDay

WITH SYNTAX DaysOfWeek -from proposed X.520 Temporal Context
SINGLE VALUE TRUE
ID in-at-stopDay}

stopTime

WITH SYNTAX DayTime-from proposed X.520 Temporal Context
EQUALITY MATCHING RULE numericStringMatch -from X.520
ORDERING MATCHING RULE numericStringMatch -from X.520
SUBSTRINGS MATCHING RULE numricStringSubstringsMatch -from X.520
SINGLE VALUE TRUE
ID in-at stopTime}

location

SUBTYPE OF telephoneNumber
SINGLE VALUE TRUE
ID in-at-location}

この属性は、登録位置（regLocations)属性に含まれるように値を規制するためには付加的なセットアップを必要とする。これは、後に、登録位置と位置（Location)の両方を含むエントリの中のアクセス制御属性によってなされる。

Data: Current Location

現在位置(Current Location)はユーザにとっての登録位置を定義する。本データは寿命（lifetime)を持つ。

Current Location	Location	one per user	match	service: user:	R R/W

従って現在位置（Current Location)属性の定義は以下に示すようになる。

current Location ATTIRUBTE::={

SUBTYPE OF telephoneNumber

SINGLE VALUE TRUE

ID in-at-cur-location}

この属性は、登録位置（regLocations)属性に含まれるように値を規制するためには付加的なセットアップを必要とする。これは、後に、登録位置と現在位置（currentLocation)の両方を含むエントリの中のアクセス制御属性によってなされる。

本属性の寿命（lifetime)はendTimeフィールドがCurrent Location ExpiryかDefault Expiryかの値に設定された一時的なコンテキストを与えることにより規定される。どちらのendTimeを使用するかの選択は属性値を設定する機能の責である。

Data: Current Location Expiry

現在位置満了（Current Location Expiry）データは現在位置データの満了時間を含む。

Current Location Expiry	TimeOfDay	one per user	none	service: user:	R R/W

Current Expiryデータは currentLocation属性の寿命を設定するためのみに使用されるため、このデータはディレクトリ中に独立した属性として蓄積されることはない。そのかわり、現在位置（currentLocation)の値に付随した一時的コンテキスト値（TemporalContxt Value)として存在する。これは、ユーザからオプションとして提供された値であることに注意を要する。もしもこの値が提供されないのならば、代わりにDefault Expiryが用いられる。この選択は、属性の管理を行うSCFのFEAによってなされる。

Data: Default Expiry

デフォールト満了（Default Expiry）データは、現在位置の満了データ（current location expiry data)に対してのデフォールトの満了時間を持つ。

Default Expiry	TimeOfDay	one per user	none	Service: User management: user:	R R/W R/W

Default Expiryデータは currentLocation属性の寿命を設定するためのデフォールト値としてのみに使用されるため、このデータはディレクトリ中に独立した属性として蓄積されることはない。そのかわり、現在位置（currentLocation)の値に付随した一時的コンテキスト値（TemporalContxt Value)として存在する。これは、ユーザがCurrent Location Expiryの値を提供しない場合にのみ使用されることに注意を要する。この選択は、属性の管理を行うSCFのFEAによってなされる。

Data: Default Location

デフォールト位置（Default Location）は他の全てのルーチングオプションに失敗した場合に使用される位置を定義する。

Default Location	Location	one per user	search	Service: User management: user:	R R/W R

従って、デフォールト位置（Default Location）属性の定義は以下のようになる。

defaultLocation
SUBTYPE OF telephoneNumber
SINGLE VALUE TRUE
ID in-at-def-location}

この属性は、登録位置（regLocations)属性に含まれるように値を規制するためには付加的なセットアップを必要とする。これは、後に、登録位置とデフォールト位置（Default Location）の両方を含むエントリの中のアクセス制御属性によってなされる。

Data: Incall Screen List

入接続スクリーンリスト（Incall Screen List）は、その位置からのユーザへの入接続を拒否する位置のリスト（規定された最大長さを持つ）を定義する。

Incall Screen List	SET OF Location SIZE 1..MaxListSIZE	one per service user	search	service: user management: user:	R R/W R

入接続スクリーンリスト（Incall Screen List）は、２つの方法で考えられる（１つのコラムを持つテーブルとして、あるいは複数の値を持つ属性として）。本ケースでは、オペレーションは特定の値が存在するかを調べるのみであり、値を取り出すことはしないため、複数の値を持つ属性という方法が使用される。

従って、入接続スクリーンリスト（Incall Screen List）の定義は以下のようになる。

incallScreenList
SUBTYPE OF telephoneNumber,
ID in-at-incall-screen}

この属性は、属性中に蓄積され得る値の数を規制するためには付加的なセットアップを必要とする。これは、後に、登録位置と位置（Location) の両方を含むエントリの中のアクセス 制御属性によってなされる。

Data: MaxListSize

最大リスト長（MaxListSize）データは、入接続スクリーンリスト（Incall Screen List）の最大長を保持する。

MaxListSize	INTEGER	one per service user	none	service: service mgt.:	R R/W

最大リスト長（MaxListSize）データは入接続スクリーンリスト（Incall Screen List）属性の最大属性値数としてのみに使用されるため、ディレクトリ中に独立した属性として蓄積されることはない。そのかわり、エントリ中にincallScreenListを含むアクセス制御属性のProtected Itemのmax ValueCountとして現れる。

Data: User Id

ユーザId(User Id)はユーザを識別する数を定義する。サービスプレフィクス（例えば014)とともに使用されると位置(Location）を表す。

User Id	ServiceNumber	one per service user	match/bind	service mgt.: service: user mgt.: user:	R/W R R/W R

ユーザId(User Id)属性の定義は以下のようになる。

userId

WITH SYNTAX	NumericStrin	
EQUALITY MATCHING RULE	numericStringMatch	-fromX.520
ORDERING MATCHING RULE	numericStringOrderingMatch	-fromX.520
SUBSTRING MATCHING RULE	numericStringSubstringsMatch	-fromX.520
SINGLE VALUE	TRUE	
ID	in-at-user-id}	

Data: User Password

ユーザパスワード（User Password）はユーザのPIN番号を保持する。本データは登録と出接続情報を検証するために使用される。

User Password	Password	one per service user	match/bind	service: user mgt.: user:	R R/W R

本属性は結合（Bind）オペレーションで使用されるため、ユーザパスワードはX.509 の6.3.節で定義されているuserPassword属性定義を用いる必要がある。

Data: Outcall Screen List

出接続スクリーンリスト（Outcall Screen List）は、ユーザの持つそれぞれの登録位置に対して許容しない位置のリストを提供する。本データは２つのコラム（regLication, stopList）を持つテーブルである。

Outcall Screen List	SET OF{
	SEQUENCE{
	stopList SET OF
	Location,
	Location Location
	}}
		one per service user	search	user mgt.:
				user:
					R/W
					R

出接続スクリーンリスト（Outcall Screen List）に対して２つの属性が定義される。

outcallScreenLocation

SUBTYPE OF	telephoneNumeber
SINGLE VALUE	TRUE
ID	in-at-outcall-location}

この属性は、登録位置（regLocations)属性に含まれるように値を規制するためには付加的なセットアップを必要とする。これは、後に、登録位置と出接続スクリーン位置（Outcall Screen Location）の両方を含むエントリの中のアクセス制御属性によってなされる。

outcallScreenStopList

SUBTYPE OF	telephoneNumber
ID	in-at-outcall-stoplist}

Data: Service Key

サービスキー（ Service Key）データは使用されるサービスを識別する。

Service Key	ServiceId	one per service	match/bind	service mgt.: service: user mgt.: user:	R/W R R R

サービスキー（ Service Key）属性の定義は以下のようになる。

serviceKey

WITH SYNTAX	Object Identifier
SINGLE VALUE	TRUE
ID	in-at-service-key}

Data: User Name

ユーザ名（User Name）データはユーザの名前を保持する。

User Name	String	one per service user	match	user mgt.: user:	R R

ユーザ名（User Name）属性の定義は以下のようになる。

```
userName ATTRIBUTE::={
        SUBTYPE OF      name,                   -from X.520
        ID              in-at-user-name}
```

Ｉ.3.2.2. オブジェクトクラスの選択

属性が選択されると、それらの属性はオブジェクトクラスに組み入れられる。規則５を用いて、以下に示す属性がグループ化される（[ ]中の属性は参照属性を示す）。

- startDay, startTime, stopDay, stopTime, location, [regLocations]
- currentLocation, defailtLocation, incallScreenList, userPassword, userName, [regLocations]

属性はオブジェクトクラス中で必須あるいはオプショナルになり得る。サービス論理の記述から、currentLocation属性とdefaultLocation属性はオプショナルであると決定できる。また、１つのテーブルの一部分にグループ化される属性は必須の属性でなければならない。他の属性も同様に必須であるように作られるべきである。

要求されるオブジェクトクラスの定義を以下に示す。

```
serviceClass OBJECT-CLASS ::= {
        MUST CONTAIN            serviceKey
        ID                      in-oc-service}

serviceUserClass OBJECT-CLASS ::= {
        MUST CONTAIN        userId,userPassword,userName,incallScreenList,regLocations
        MAY CONTAIN         current Location, defaultLocation
        ID                  in-oc-service-user}

timeOfDayLocation OBJECT-CLASS ::= {
        MUST CONTAIN        startDay, startTime, stopDay, stopTime,location,regLocation
        ID                  in-oc-time-location}

outcallScreenList OBJECT-CLASS ::= {
        MUST CONTAIN        outcallScreenLocation, outcallScreenStopList,regLocations
        ID                  in-oc-outcall-screen}
```

属性アクセス制御に使用されるアクセス制御属性の定義は DITの設計に委ねられている（Ｉ.3.2.4.参照）。

オブジェクトクラスが定義されると、オブジェクトに対してのネーミング属性（naming attribute）が指定されなければならない。このために、このオブジェクトの発生に対して唯一の値を持つ必須の属性の一つを識別することが要求される。

定義したオブジェクトの考察から以下に示すネーミング属性が使用され得る。

ServiceClass	serviceKey
serviceUserClass	userId
timeOfDayLocation	?(nothing useable)
outcallScreenList	outcallScreenLocation

ネーミングに関して問題のあるのはtimeOfDayLocation オブジェクトクラスのみである。この問題に対して２つの方法がある。１つはネーミングを可能とするために唯一の属性（例えばtableRowIndex）を定義することである。これはオペレーションの性能に影響を与えず、ここでは考慮していないが、管理オペレーションにおいて有用である。

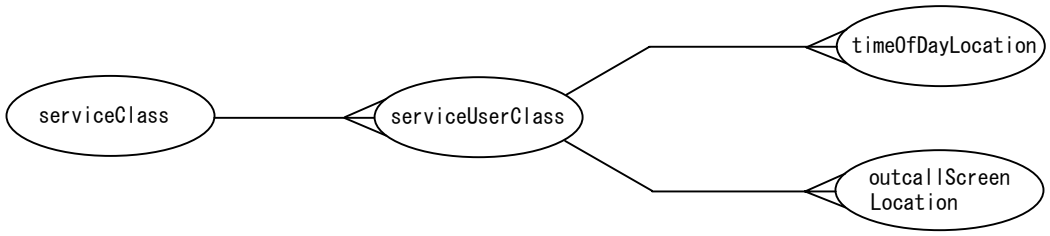
２つめの解は StartDay,StartTime属性を結合して混成属性（composite attribute）を作ることである。こうすれば、その値は唯一となる。本方式はMATCHING規則の再定義と付随処理を含むであろう。

単純化のために最初の解を選択する。要求される名前の形式を以下に示す。

```
tableRowIndex ATTRIBUTE::={
        WITH SYNTAX     Integer
        SINGLE VALUE    TRUE
        ID              in-at-maxListValues}

timeOfDayLocation OBJECT-CLASS::= {
        MUST CONTAIN    tableRowIndex, startDay, startTime, stopDay,
                        stopTime, location, regLocations
        ID              in-oc-time-location}

serviceName NAME-FORM::= {
          NAMES                 serviceClass
          WITH ATRRIBUTES       serviceKey
          ID                    in-nf-service-name }

serviceUserName NAME-FORM ::=
          NAMES                 serviceUserClass
          WITH ATTRIBUTES       userId
          ID                    in-nf-service-user-name}

timeOfDayLocationName-Form ::=
          NAMES                 timeOfDayLocation
          WITH ATTRIBUTES       tableRowIndex
          ID                    in-nf tod-location-name}

outcallScreenName NAME-FORM ::=
          NAMES                 outcallScreenList
          WITH ATTRIBUTES       outcallScreenLocation
          ID                    in-nf-outcall-screen-name}
```

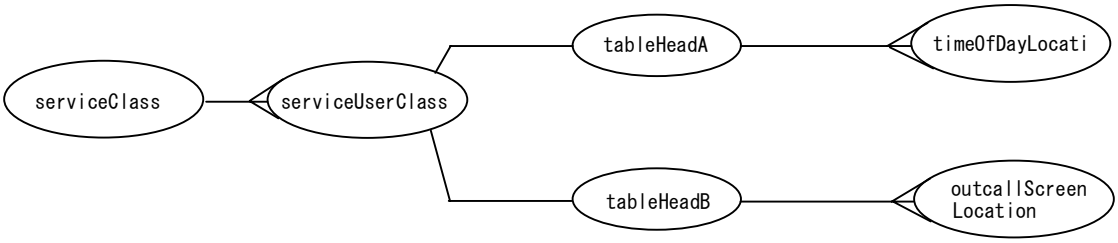
Ｉ.3.2.3. オブジェクトクラスの選択

この時点でDITの設計が可能となる。基本性（cardinality）の情報から、次に示すオブジェクトクラス関係が導出される



付図 I-4/JT-Q1218(ITU-T Q.1218)：オブジェクトクラス関係

serviceUserClassはその配下に２つのエントリセットを持つため、規則７が適用され、その結果以下に示すように特別なオブジェクトのレベルが追加される。



付図 I-5/JT-Q1218(ITU-T Q.1218)：規則７適用後のオブジェクトクラス関係

従って追加の追加のオブジェクトクラスと属性の定義が必要となる。これを以下に示す。

tableHeadIdentifier

WITH SYNTAX	Integer
SINGLE VALUE	TRUE
ID	oi-gen-table-head}

tableHead OBJECT-CLASS::= {

MUST CONTAIN	tableHeadIdentifier
ID	in-oc-tableHead}

tableHeadA OBJECT-CLASS ::=

MUST CONTAIN	tableHeadIdentifer
ID	in-oc-tableHeadA}

tableHeadNameA NAME-FORM ::=

NAMES	tableHeadA
WITH ATTRIBUTEs	tableHeadIdentifier
ID	in-nf-tableHeadNameA}

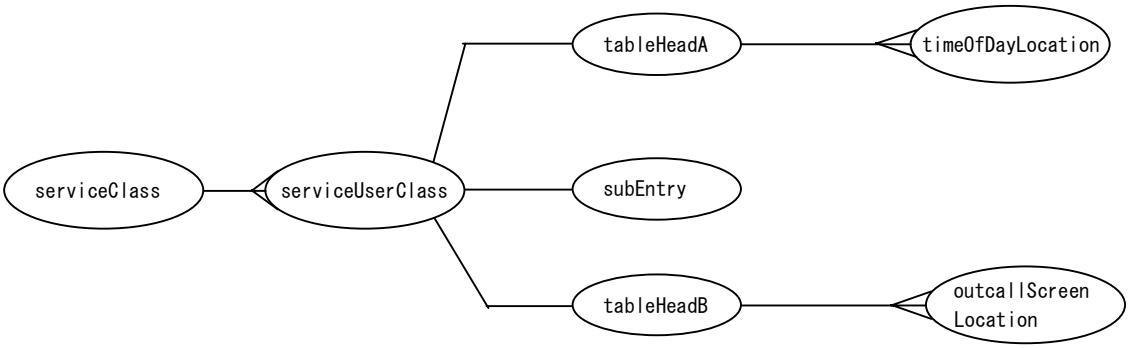
```
tableHeadB OBJECT-CLASS ::= {
        MUST CONTAIN        tableHeadIdentifier
        ID                  in-oc-tableHeadB}

tableHeadNameB NAME-FORM ::=
        NAMES               tableHeadB
        WITH ATTRIBUTEs     tableHeadIdentifier
        ID                  in-nf-tableHeadNameB }
```

従って、DIT構成を定義するのに要求される構成規則は以下に示すものとなる。

```
sr-root STRUCTURE-RULE ::=
        NAME FORM           serviceClassName
        ID                  sr0 }

sr-user STRUCTURE-RULE ::=
        NAME FORM           serviceClassName
        SUPERIOR RULES sr0
        ID                  sr1}

sr-tableA STRUCTURE-RULE ::= { NAME FORM
        tableHeadNameA
        SUPERIOR RULES sr1
        ID                  sr2}

sr-tod-location STRUCTURE-RULE ::=
        NAME FORM           timeOfDayLocationName
        SUPERIOR RULES sr2
        ID                  sr3}

sr-tableB STRUCTURE-RULE ::= { NAME FORM
        tableHeadNameB
        SUPERIOR RULES sr1
        ID                  sr4}

sr-outcall-screen STRUCTURE-RULE ::=
        NAME FORM           outcallScreenListName
        SUPERIOR RULES sr4
        ID                  sr5}
```

subEntry のクラスオブジェクトのエントリを作ることにより、集合的な属性が適用されるsub-treeの仕様を保持することになる。regLocations collective attributeの場合には、subEntry はserviceUserClassオブジェクトクラスのエントリへ従属する。

regLocation 属性を変更する場合には、subEntryの識別名（Distinguished Name ）が与えられることを必要とする。



付図 I-6/JT-Q1218(ITU-T Q.1218)：sub-entry 位置を示すディレクトリツリー

Ｉ.3.2.4. アクセス制御の割り当て

DITが完成したので次に要求される段階はDITの中のそれぞれのエントリのアクセス制御属性の値を定義することである。１つのエントリのアクセス制御属性（エントリＡＣＩ，X.501,16.5.2参照）はユーザが該属性値にどのようにアクセスするのかを制御するACIItemの型の値を含む。次に示す表はDITの中で定義されているそれぞれの属性に要求されるアクセス制御を定義している。

Object	Attribute	UserClasses defined by X.500 Management					Special
		User-X	AllUser	User Mgt	Service	service Mgt	Protected Items
ServiceClass	serviceKey	+R+F	+F	+R+F	+R+F	+R+A+D+F	
ServiceUserClass	userId	+R+F	+F	+R+F	+R+F	+R+A+D+F	
	userPassword	+R+F+C		+R+A+D+F	+R+F	+R+F	
	userName	+R+F+C		+R+A+D+F	+R+F	+R+F	
	incallScreenList	+R+F		+R+A+D+F(*)	+R+F	+R+F	maxValueCount
	currentLocation	+R+A+D+F(*)		+R+F	+R+F	+R+F	restrictedBy
	defaultLocation	+R+A+D+F(*)		+R+F	+R+F	+R+F	restrictedBy
TableHeadA	tableHeadIdentifier	+R+F		+R+F	+R+F	+R+A+D+F	
TimeOfdayLocation	tableRowIndex	+R+F		+R+A+D+F	+R+F	+R+F	
	startDay	+R+F		+R+A+D+F	+R+F	+R+F	
	startTime	+R+F		+R+A+D+F	+R+F	+R+F	
	stopDay	+R+F		+R+A+D+F	+R+F	+R+F	
	stopTime	+R+F		+R+A+D+F	+R+F	-R+F	
	location	+R+F		+R+A+D+F(*)	+R+F	+R+F	restrictedBy
TableHeadB	tableHeadIdentifier	+R+F		+R+F	+R+F	+R+A+D+F	
OutcallScreenList	outcallScreenLocation	+R+F		+R+A+D+F(*)	+R+F	+R+F	restrictedBy
	outcallScreenStopTime	+R+F		+R+A+D+F	+R+F	+R+F	
SetviceUserSubentry	regLocations	+R+F		+R+A+D+F	+R+F	+R+F	

+	grant permission	A	add attribute to entry	M	modify attribute
−	deny permissions	R	read access	C	compare attribute
*	special protected Item	D	remove access	F	filter match attribute

UserClassesの下のそれぞれのコラムは entry ACI 属性のACIItem となり得る値を示している。実際、値は組み合わされて一つのACIItemになり得る。そのACIItemは同じUserClassesの組に対して同じaccess permissions を持つprotected items の組を組み合わせる。それぞれのACIItemは唯一の識別子フィールドを持ち、その識別子フィールドはアクセス制御属性それ自体の個々の値へのアクセス制御へ適用するのに使用可能である。このことは異なる”ユーザ”が、アクセス制御属性の他の値を変更できなくても、異なるアクセス制御値（許容された値（permitted value）あるいはサイズの制限（sizelimits）を含むであろう）を変更することを可能とする。

permissions が（＊）で示された箇所は、permitted valueあるいはvalue count limit controlsを扱う為には、最後のコラム（Special Protected Items）で規定された型のSpecial Protected ItemsがACIItem に含まれる必要があることを示している。

Ｉ.4. コンテキストの定義

この項ではINサービスにおいて有用であることが理解できるであろうコンテキスト型を定義する。

Ｉ.4.1. 数値インデクス属性コンテキスト

数値インデクス属性コンテキスト（Numerical Index Attribute Value Context)は数値インデクスを属性値と関連付ける。

NumericalIndexAVC		ATTRIBUTE-VALUE-
	SYNTAX	INTEGER
	Id	in-avc-numercallIndex}

数値インデクス属性コンテキスト（Numerical Index Attribute Value Context)を使用する例は、”短縮から拡張番号へ”の変換テーブルである。そこでは、インデクス（コンテキスト値）は短縮番号であり、属性値は拡張番号である。

同様の文字列インデクス属性値コンテキスト型（String Index Attribute Value Context type）も同様に定義されることに注意のこと。

付録Ⅱ　認証フレームワーク

Ⅱ.1. はじめに

本認証フレームワークでは、ネットワークサービスを提供するサービス制御機能(SCF)が、ユーザの正当性の検証を行うため、ユーザの認証処理を実行可能な SDF にアクセスし認証を行う通信環境を想定する。SDF は、ユーザの認証に必要となる情報をユーザ毎に蓄積保有すると共に、認証を実行するための計算処理機能を具備する。SCF は、ユーザから得た認証情報を SDF に転送することにより SDF に対し認証の依頼を行う。SDF は転送された認証情報に基づき認証のための計算処理を実行し、認証結果を SCF へ返答する。

本標準は、SCF と SDF の間で提供する認証手法(メカニズム)を規定するもので、勧告 X.509(ディレクトリ認証フレームワーク)をベースとする。本認証手法は、多様な応用への適用を考慮し、パスワードを用いた簡易な認証から公開鍵暗号系を用いる強度の認証に至るまで幅広い認証手法を包含する。但し、本標準の認証手法において利用する計算処理アルゴリズム(すなわち暗号アルゴリズム)については、本標準の対象外とする。

Ⅱ.2. 認証手法の概要

本標準では、簡易認証、および厳密認証を規定する。

Ⅱ.2.1 簡易認証

ユーザ識別名、パスワードを用いた簡易な認証手法であり、以下の認証手法を規定する。

(1)ユーザ識別名認証

ユーザの識別名(Distingushed Name：DN)を SCF から SDF へ転送し、SDF ではその識別名が正当に登録されているかを判定する。登録されている場合は認証成功とする。

(2)単純パスワード認証

ユーザの識別名(DN)、およびパスワード(PW)を SCF から SDF へ転送し、SDF ではその識別名とパスワードのペアが正当に登録されているかを判定する。登録されている場合は認証成功とする。

(3)秘匿パスワード認証

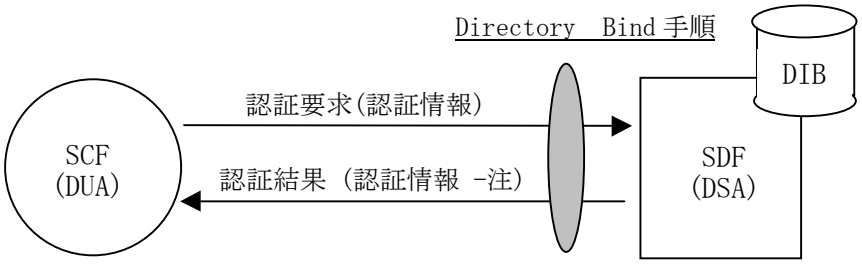
秘匿パスワード認証は、回線上を流れるパスワード情報を秘匿するために、SCF においてパスワード(PWa)の秘匿処理を行い、その処理結果(秘匿パスワード:PWsa)を SDF に送信するもので、パスワード情報そのものが回線上に露呈しないことを特徴とする。SDF では、識別名 A から A のパスワード(PWa)を検索し、それを用いて SCF と同等の秘匿処理を行うことにより秘匿パスワード PWsx を生成する。SDF における認証の検証としては、PWsa と PWsx を比較することにより実行する。PWsa と PWsx が等しい場合は認証成功とする。

Ⅱ.2.2 厳密認証

ユーザの識別名(DN)、およびユーザの秘密鍵で暗号処理された認証情報を SCF から SDF へ転送する。SDF では、登録されている識別名とユーザの公開鍵(上記秘密鍵とのペア)を検索し、その公開鍵を用いて転送された認証情報を復号し、ユーザの正当性を確認する。本手法においては、公開鍵暗号技術の利用が必須であり、SDF においては正当なユーザの公開鍵の取得が必要となる。

Ⅱ.3. 認証手順

本標準においては、アソシエーション確立時に認証を実行する手順を規定する。図 - 1 で示す通り、SCF が SDF に対して認証情報を送信し、SDF が認証処理を実行してその結果を返送する手順が基本となる。本手順は、ITU-T 勧告 X.511 で規定されるディレクトリ結合(Directory Bind)手順を用いる。



注 ： 認証結果と同時に SCF が SDF を認証するための認証情報を運ぶことも可能とする。

図 -付録 2-1 認証手順(アソシエーション確立時)

認証手順のためには、以下の手順要素を用いる。尚、それぞれの手順要素の ASN.1 記法については、ITU-T 勧告 X.511 を参照する。

(1)ディレクトリ結合(Directory Bind)

本手順要素は、SCF から SDF に発行されるもので、以下のパラメータの設定により使用される認証手法の要求が SDF に対してなされる。

●資格証明(Credentials) ： 簡易資格証明(Simple Credentials), 厳密資格証明(Strong Credentials) により構成され、認証が必要となる場合はどちらかを選択する。認証を行わない場合は、本パラメータは使用しない。

●簡易資格証明(Simple Credentials) ： 本パラメータが選択されると簡易認証を使用する。本パラメータは、ユーザ識別名(DN)、パスワード(PW)、日時情報(time1 & time2)、乱数(random1 & random2)により構成され、ユーザ識別名(DN)のみが設定された場合は、SDF ではユーザ識別名認証が実行され、ユーザ識別名(DN)、パスワード(PW)が設定された場合は、単純パスワード認証が実行される。また、ユーザ識別名(DN)、パスワード(PW)、日時情報(time1 & time2)、乱数(random1 & random2)が設定された場合は、SDF では秘匿パスワード認証が実行される。尚、秘匿パスワード認証においては、日時情報と乱数の設定が 1 ペアと 2 ペアの場合があり、それぞれそこで使用される関数が異なる(第 3 編参照)。

●厳密資格証明(Strong Credentials) ： 本パラメータが選択されると厳密認証を使用する。厳密認証の中の一方向認証と双方向認証のいずれを使用するかは予め取決めが必要である(第 3 編参照)。

(2)ディレクトリ結合結果(DirectoryBind Result)

本手順要素は、SDF から SCF に発行されるもので、ディレクトリ結合(Directory Bind)の応答として認証に成功した場合にのみ使用される。簡易認証、および厳密認証(一方向認証)において認証に成功したことを返答する場合は、ヴァージョン(Versions) のみを返答する。また、厳密認証(双方向認証)において認証に成功したことを返答する場合は、ヴァージョン(Versions) に加えて資格証明(Credentials) も含めて返答する。

(3)ディレクトリ結合エラー(Directory Bind Error)

本手順要素は、SDF から SCF に発行されるもので、ディレクトリ結合(Directory Bind)の応答として認証に失敗した場合にのみ使用される。本手順のパラメータの使用方法については、第 3 編参照。

Ⅱ.4. 認証手法の詳細

Ⅱ.4.1 ユーザ識別名認証

SCF に接続されるユーザ(DN=A)を SDF がその識別名をもって認証する場合の詳細メカニズムを以下に規定する。

(a)SCF は、ユーザの識別子(A)を SDF に送る(ディレクトリ結合(Directory Bind))。

(b)SDF は、SCF から得た A をキーとして A が正当なユーザであるか否かを検索する。

(c)A が正当な識別名である場合は、認証成功を SCF に返送し(ディレクトリ結合結果(Directory Bind Result))、A が不正であれば認証失敗を返送する(ディレクトリ結合エラー(Directory Bind Error))。

Ⅱ.4.2 単純パスワード認証

SCF に接続されるユーザ(DN=A)を SDF が単純パスワード(暗号処理のないパスワード)を用いて認証する場合の詳細メカニズムを以下に規定する。

(a)SCF は、ユーザの識別子(A)とパスワード(PWa)を平文(暗号なし)で SDF に送る(ディレクトリ結合(Directory Bind))。

(b)SDF は、SCF から得た A をキーとして DIB(ディレクトリ情報ベース)に蓄えられた A の PWx を検索する。

(c)識別名 A から検索した PWx と SCF から送信された PWa を比較検証し、その結果を SCF に返す。PWx=PWa であれば認証成功を SCF に返送し(ディレクトリ結合結果(Directory Bind Result))、PWx≠PWa であれば認証失敗を返送する(ディレクトリ結合エラー(Directory Bind Error))。

Ⅱ.4.3 秘匿パスワード認証

SCF に接続されるユーザ(DN=A)を SDF が秘匿パスワードを用いて認証する場合の詳細メカニズムを以下に規定する。

(1)基本メカニズム

SCF と SDF に全く同じ秘匿関数を保有する。SCF では、秘匿関数への入力情報として、ユーザの識別名(A)、パスワード(PWa)、乱数、時間情報を用いる。SDF へ送信する情報としては、秘匿関数の出力情報(秘匿パスワード)、および PWa 以外の秘匿関数への入力情報である。SDF では、受信したユーザの識別名(A)からパスワード PWa を検索し、SCF より送信された秘匿関数への入力情報に PWa を加えて SDF の秘匿関数へ入力する。すなわち、SCF と全く同等の秘匿関数処理を実行する。SDF での秘匿関数の出力と、SCF より転送された秘匿パスワードを比較し、等しい場合はユーザが認証できたこととする。図 - 2 に基本メカニズムを示す。

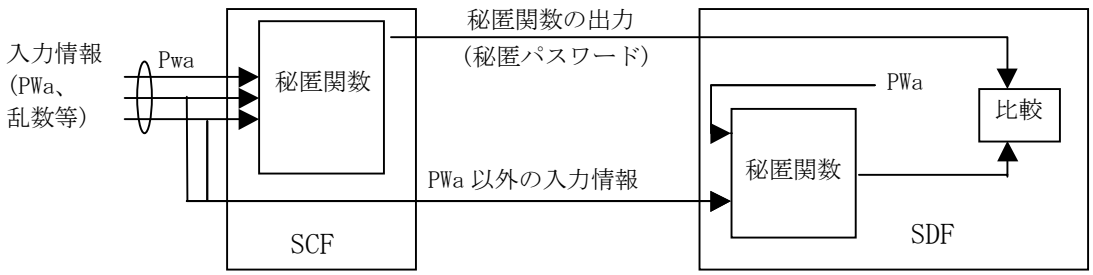


図 –付録 2- 2 秘匿パスワード認証基本メカニズム

(2)秘匿関数

本認証手法を実現する秘匿関数には、以下の 2 種類の関数を規定する。

関数 1　：関数 1 (f1)の入力情報は、以下のものである。

- ▼ ユーザの識別名(A)
- ▼ パスワード(PWa)
- ▼ 時間 1 (T1a)
- ▼ 乱数 1 (Q1a)

従って、出力としては、f1 (A, PWa, T1a, Q1a) と記述できる。

関数 1&2 ：上記関数 1(f1)の他に、関数 2(f2)を連鎖させる。関数 1(f1)への入力情報は上記の通りである。関数 2(f2)への入力情報は、以下のものである。

- ▼ f1 出力情報 ： f1 (A, PWa, T1a, Q1a)
- ▼ 時間 2 (T2a)
- ▼ 乱数 2 (Q2a)

従って、関数 1&2 の出力としては、

f2 (f1 (A, PWa, T1a, Q1a) T2a, Q2a) と記述できる。

図 - 付録 2-3 に秘匿関数の処理概要を示す。

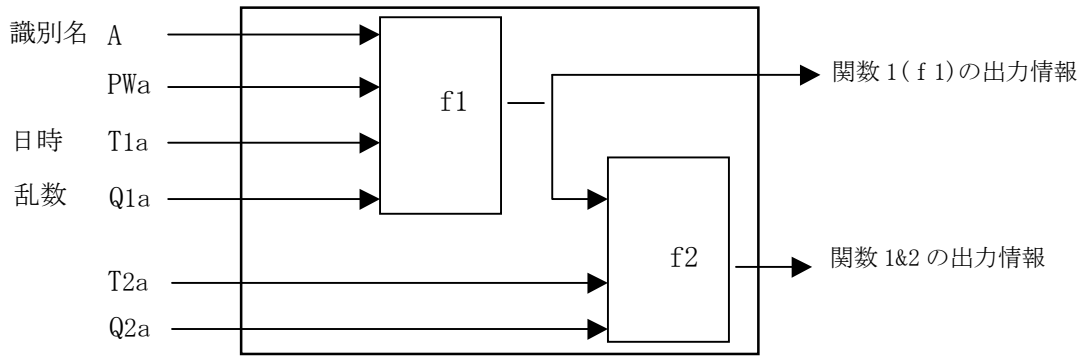


図 -付録 2- 3　　秘匿関数の処理概要

(3)認証メカニズム

前提として、SCF と SDF は、予め秘匿パスワード認証を行うための環境設定を行う必要がある。環境設定としては、関数 1、または関数 1&2 の選択、およびそれぞれの関数の詳細動作規定を行う必要がある。尚、本標準においては、それぞれの秘匿関数の詳細動作の規定は標準の対象外とする。

[関数 1 を用いた秘匿パスワード認証]

(a)SCF は、ユーザの識別子(A)とパスワード(PWa)を得る。さらに、認証を要求する時間(T1a)を獲得し、乱数(Q1a)を生成する。但し、時間と乱数の利用についてはオプションである。

(b)SCF では、f1 に上記情報を入力し、その出力 f1 (A, PWa, T1a, Q1a) を得る。

(c)SCF は、出力情報 f1 (A, PWa, T1a, Q1a) と、ユーザの識別子(A)、時間(T1a)、乱数(Q1a)を SDF に送信する(ディレクトリ結合(Directory Bind))。

(d)SDF では、送信されたユーザの識別子(A)よりユーザのパスワード PWa を検索し、SCF より送信されたユーザの識別子(A)、時間(T1a)、乱数(Q1a)と合わせて SDF の f1 に入力する。

(e)SDF の f1 の出力と、SCF より送信された f1 (A, PWa, T1a, Q1a) を比較し、その結果を SCF に返す。等しい場合は認証成功を SCF に返送し(ディレクトリ結合結果(Directory Bind Result))、等しくない場合は認証失敗を返送する(ディレクトリ結合エラー(Directory Bind Error))。

[関数 1&2 を用いた秘匿パスワード認証]

(a)’SCF は、上記①、②の処理後、f2 の処理を実行する時間(T2a)を獲得し、その時点で乱数(Q2a)を生成する。すなわち、時間 T1a≠T2a、および乱数 Q1a≠Q2a が望ましい。但し、時間と乱数の利用についてはオプションである。

(b)’SCF では、f2 に f1 (A, PWa, T1a, Q1a)、時間(T2a)、および乱数(Q2a)を入力し、その出力 f2 (f1 (A, PWa, T1a, Q1a) T2a, Q2a) を得る。

(c)’SCF は、出力情報 f2 (f1 (A, PWa, T1a, Q1a) T2a, Q2a) と、ユーザの識別子(A)、時間(T1a、T2a)、乱数(Q1a, Q2a)を SDF に送信する(ディレクトリ結合(Directory Bind))。

(d)’SDF では、送信されたユーザの識別子(A)よりユーザのパスワード PWa を検索し、SCF より送信されたユーザの識別子(A)、時間(T1a)、乱数(Q1a)と合わせて SDF の f1 に入力し、その出力 f1 (A, PWa, T1a, Q1a)を得る。その後、SCF より送信された時間(T2a)、乱数(Q2a)と、SDF で計算した f1 (A, PWa, T1a, Q1a) を f2 に入力し、その出力 f2 (f1 (A, PWa, T1a, Q1a) T2a, Q2a) を得る。

(e)’SDF の f2 の出力と、SCF より送信された f2 (f1 (A, PWa, T1a, Q1a) を比較し、その結果を SCF に返す。等しい場合は認証成功を SCF に返送し(ディレクトリ結合結果(Directory Bind Result))、等しくない場合は認証失敗を返送する(ディレクトリ結合エラー(Directory Bind Error))。

Ⅱ.4.4 厳密認証

SCF に接続されるユーザ(DN=A)を SDF が公開鍵暗号を用いて認証する場合の詳細メカニズムを以下に示す。認証手法としては、以下の 2 通りを規定する。

(1)一方向認証 (One - Way 認証)

SCF から SDF に対して公開鍵暗号を用いて作成した認証情報を送信し、SDF が SCF(実際は SCF に接続されるユーザ(DN=A)を)を認証する。すなわち、認証される側が SCF(ここでは、A とする)で、認証の検証を行うものを SDF とする。尚、正式な証明付きのユーザ A の公開鍵 Ap を得るためには、信頼できる資格証明センタ(CA)にアクセスすることが必要となる。

(a) ユーザ A は、自分の公開鍵証明の要求を資格証明センタ(CA)に対して行なう。この際、パラメタとしてユーザ A の識別子 A を設定する。

(b) CA は、ユーザ A の公開鍵を資格証明センタ(CA)の秘密鍵で復号化した公開鍵証明 Cert A をユーザ A に返す。Cert A は以下の情報である(ITU-T 勧告 X.509 参照)。

Cred A=CA ‖ AI ‖ A ‖ Ap ‖ TA ‖ V ‖ SN

hA = h (Cred A)

Cert A = Cred A ‖ CAs[hA]

用語 ： Cred A ： ユーザ A を識別するための資格証明情報、

CA : 資格証明センタ(CA)の識別子、

AI : CA で用いる認証アルゴリズム識別子、

Ap ： ユーザ A の公開鍵、

TA : ユーザ A が使用できる有効期限、

V: バージョン、

SN : 通番、

h (X) ： 情報 X をハッシュ処理する関数 h、

CAs [X] : CA の秘密鍵で情報 X を処理(復号)、

‖ ： 連結

上記手順(a)(b)は、正当なユーザ A の公開鍵 Ap を CA に保証してもらうために行う手続きであり、事前に本処理がなされている場合は、本手順を省略することができる。但し、公開鍵 Ap の有効期限 TA が切れた場合は、再度本手順を行う必要がある。

(c)　ユーザ A は認証情報(TokenAB)を作成し、ユーザ A の資格証明(Cert A)とともに SDF に送る(ディレクトリ結合(Directory Bind))。ここで認証情報は、日時情報、乱数及び SDF の識別子(B)を A の秘密鍵で復号化したものである。TokenAB は以下の情報である。

TOKEN AB=B ‖ tA ‖ rA ‖ As (h(B ‖ tA ‖ rA))

用語 ： B ： SDF の識別子、

As ： ユーザ A の秘密鍵、

As [X] : ユーザ A の秘密鍵で情報 X を処理(復号)、

tA : TOKEN AB 発行時の日時情報、

rA : ユーザ A が生成した乱数

(d)　SDF では、資格証明(Cert A)からユーザ A の公開鍵 Ap と取りだし、これを用いて、認証情報を暗号化して、日時情報、乱数、識別子の正当性を検証する。検証方法としては、以下の手順に従う。

- 識別子 A の正当性を確認する。
- 日時情報 tA と現在の日時情報を比較し、大きな隔たりのないことを確認する。
- Ap を用いて Ap(As (h(B ‖ tA ‖ rA)))=h(B ‖ tA ‖ rA)を得る。
- (B ‖ tA ‖ rA)をハッシュ関数 h を用いて処理し、h(B ‖ tA ‖ rA)’を得る。
- 上記 h(B ‖ tA ‖ rA)と h(B ‖ tA ‖ rA)’を比較し、等しい場合は検証成功とする。

認証の検証が成功裏に終了した場合は、認証成功を SCF に返送し(ディレクトリ結合結果(Directory Bind Result))、失敗した場合は認証失敗を返送する(ディレクトリ結合エラー(Directory Bind Error))。

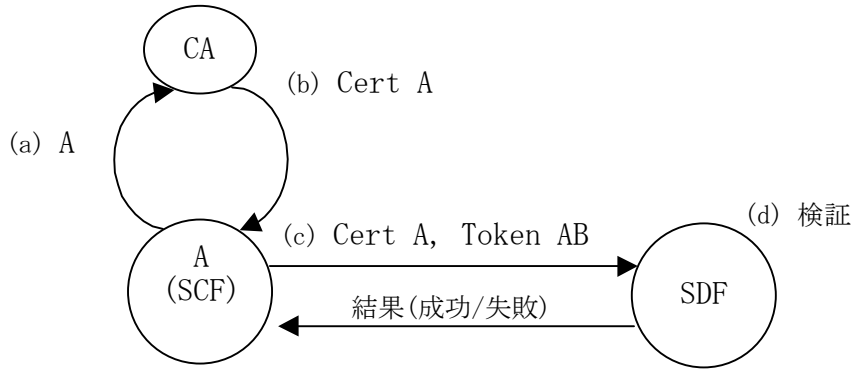


図　付録２-4　　厳密認証　一方向認証

(2)　　双方向認証 (Two - Way 認証)

双方向認証では、SCF から SDF に対する認証処理については、上記一方向認証と全く同じであるが、ここでは逆方向の認証をも可能とする。すなわち、SCF が SDF を認証することが可能である。以下に双方向認証のメカニズムを示す。

(a) - (b)　一方向認証と同じメカニズム

(c)　　一方向認証と同じメカニズム

(d)　SDF における検証処理も一方向認証と同じである。ここでは、さらに SCF が SDF を認証するために、SDF は認証情報(Token BA)を作成し、SCF に送信する。TokenBA は以下の情報である。

TOKEN BA=A ‖ tB ‖ rA ‖ rB ‖ eKsB (h(A ‖ tB ‖ rA ‖ rB))

用語　：　tB :　Token BA 発行時に SDF が生成する日時情報、

rB : B が生成した乱数

(e)　SCF　では、SDF　の公開鍵　Bp　を用いて、認証情報を暗号化して、日時情報、乱数、識別子の正当性を検証し、これらがすべて正しい場合に、SCF　は　SDF　の正当性を認証する。正しい場合は認証成功とし、その後の通信を継続し、正しくない場合は認証失敗として、通信をアボート(強制終了)する。

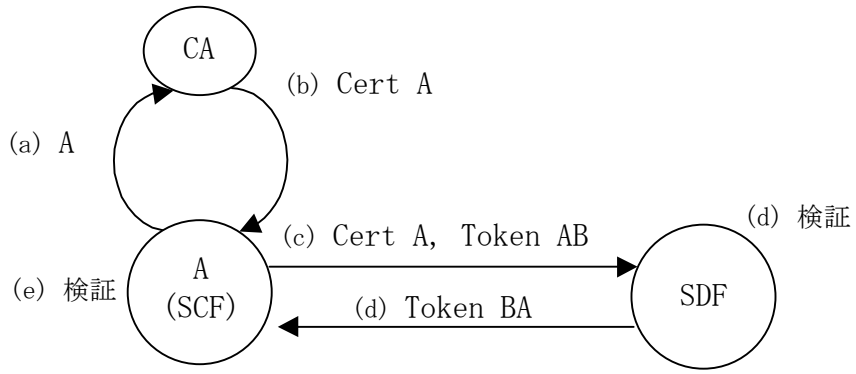


図　付録２－５　　厳密認証　双方向認証

付録　Ⅲ　ISPT のためのオブジェクトモデリング

この付録では、Inter-Network Service Profile Transfer (ISPT)において利用可能であるオブジェクトモデル化の手法を示す。

Ⅲ.1　前提

Ⅲ.1.1　移動体通信サービスと加入者識別子

Ⅲ.1.1.1　移動体加入者固有識別子

同じサービス提供者に属する個々の加入者を識別するため、移動体通信サービス加入者は、数値表現される固有の識別子を持たなければならない。この加入者識別子は、ローカル（一つのサービス提供者）において意味をなす。つまり個々のサービス提供者は、他のサービス提供者において付与される固有の加入者識別子を知る必要はない。

識別子の最大長については互いサービス提供者間で合意すべきであるが、その最大値を超えない範囲にでは付けられる加入者識別子の長さは、ローカルな決め事である。

Ⅲ.1.1.2.　移動体通信サービス提供者プリフィックス

個々の移動体通信サービス提供者は、固有の数値識別子(プリフィックス)を持たなければならない。サービス提供者のプリフィックスは、広域的（協力関係にある複数のサービス提供者間）に意味をなす。つまり協力関係にある個々のサービス提供者は、互いに他の全てのサービス提供者達のプリフィックスを理解しなければならない。

プリフィックスの最大長の設定については互いに合意すべきである。

個々のローカル加入者固有の識別子につけるプリフィックスとして、そのサービス提供者の識別子を用いることにより、全ての移動体通信サービス加入者は、固有かつ明確に識別できる。

Ⅲ.1.1.3.　移動体通信サービス提供者サブ-プリフィックス

ある移動体通信サービス提供者が、複数の DSA を用いて加入者のサービスデータを管理する例のように、加入者を論理的に分類する場合には、サービス提供者のサブ-プリフィックスを用いることができる。サービス提供者のサブ-プリフィックスは、ローカル（一つのサービス提供者）において意味をなす。つまり、協力関係にある個々の移動体通信サービス提供者は、互いに他の全てのサービス提供者のサブ-プリフィックスを理解する必要はない。

識別子の最大長については互いサービス提供者間で合意すべきであるが、その最大値を超えない範囲にでは付けられる加入者識別子の長さは、ローカルな決め事である。

サブ-プリフィックスの付加的な利用方法は、将来的な検討課題である。この利用方法が将来的に必要であると想定しているわけではないが、これを定義し、実装することは困難なことではなかろう。

Ⅲ.1.1.4. 符号化

ローミングしている固有の加入者識別子、サービス提供者のプリフィックスおよびサブ-プリフィックスを在圏網に知らせる手段は、本付録で扱う範疇外である。本付録は、ユーザ固有の識別子や、サービス提供者のプリフィックス、サブ-プリフィックスを個々に識別するために、オブジェクト識別子、長さや値のようないくつかの符号化タイプについて言及することを目的とする。

Ⅲ.1.2. プロファイルエントリの露呈

あるサービス提供者に属する加入者エントリの全リストが、その他のサービス提供者にて利用可能とすべきであると要求することは、望ましいものではない。ローミングしているユーザのプロファイルのみ、利用可能とすべきである。

Ⅲ.2 DIT スキーマ

Ⅲ.2.1 X.500 DIT

図1におけるX.500 DITは、本付録で推奨するDIT構造を示す。

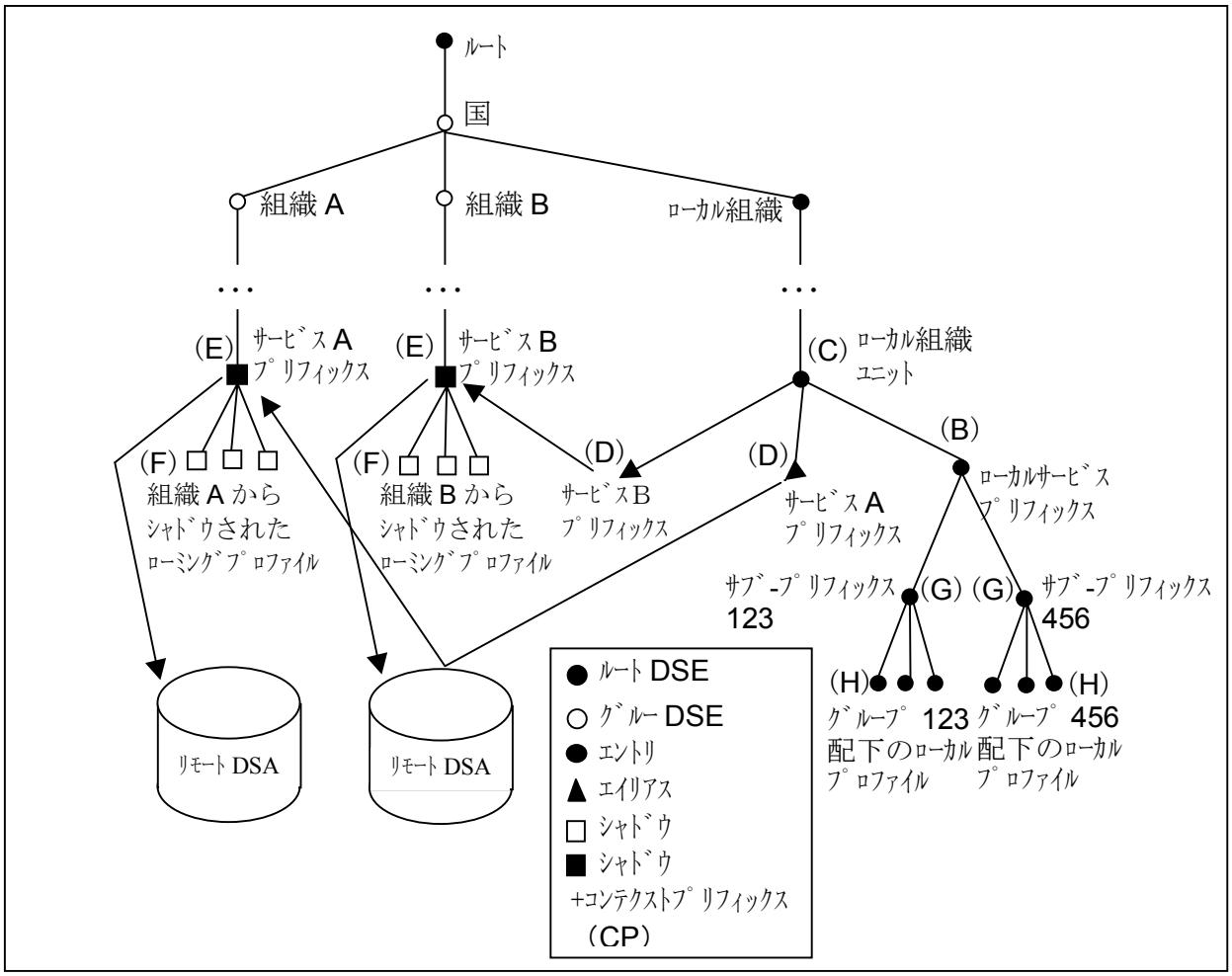


図　付録3-1: 移動体通信サービス提供者用ローカルDITの例

Ⅲ.2.1.1　ローカルプロファイルの格納場所

ひとつのサービス提供者のプリフィックスは、サービス提供者のローカル DIT 中において、配下に全てのローカル加入者プロファイルを格納するエントリノード（図 1 にて(B)とラベルされたノードを参照）を命名するために用いられる。サービス提供者プリフィックスエントリは、値が inMobilityService プリフィックスであるオブジェクトクラスを含む。

ユーザプロファイルは、更にサービス提供者のサブ-プリフィックスにより分類することができる。仮にサブ-プリフィックスを使用する場合は、付加的な Relative Distinguished Name (RDN)と、ユーザ固有の識別子とサービス提供者のプリフィックス間の DIT におけるノード（(G)とラベルされたノード）により実現される。サービス提供者サブ-プリフィックスエントリは、もしそれが用いられる場合、値が inMobilityServiceSub プリフィックスであるオブジェクトクラスを含む。サブ-プリフィックスエントリは、サービス提供者サブ-プリフィックス識別子を用いて命名される。

ローカルプロファイルエントリ（(H)とラベルされたノード）は、値が inMobilityUserProfile であるオブジェクトクラスを含む。ローカル加入者エントリ群は、加入者の固有の移動体通信サービス識別子を用いて命名される。

Ⅲ.2.1.2　遠隔の加入者プロファイルの格納場所

ひとつのサービス提供者の加入者プロファイル群は、その他の協力関係にあるサービス提供者に利用可能なものでなければならない。そこで、遠隔の DSA において格納されるプロファイル群の上位エントリ（(E)とラベルされたノード）群は、ローカル DSA 上にシャドウされなければならない。このシャドウ DSE エントリが不完全な下位知識を持つ場合に限り、在圏網は、continuation reference 情報をこのエントリが格納する supplierKnowledge 属性から生成することができる。 この continuation reference 情報は、実際のプロファイル群が存在するローミングユーザのホーム DSA を指し示す。

Ⅲ.2.1.3　ローミング時の加入者プロファイルの格納場所

ローミング加入者（locally visiting subscribers)は、前述のノード（(E)とラベルされたノード）の配下にシャドウされる、加入者プロファイルエントリ（(F)とラベルされたノード）を持つ。個々のプロファイルエントリは、加入者が属するホームのサービス提供者にそれぞれ対応している。

Ⅲ.2.1.4　メッセージフローの削減

ユーザが別の網にローミングする場合、在圏網は、ローミングしているユーザのホーム網におけるユーザプロファイルを更新しなければならない。これは、 ユーザのプロファイルが在圏網に転送されることによるものである。

ユーザのプロフィルを更新するために、在圏網はローミングしているユーザの DN を確定しなければならない。メッセージフローを削減するため、DAP における複数の Search 操作を用いることなく、この DN は解析できる。加入者識別子とディレクトリ番号の DN へのマッピングは、固定長番号計画の使用によって実現できる。

前章の前提により、ローミングしているユーザ固有の識別子とプリフィックス（必要であればサブ-プリフィックスも）は、在圏網において認識できる。これらの識別子は、RDN の組み合わせにより、ローミングしているユーザの DN として用いることができる。仮の DN は、ホーム網のプリフィックスエントリに対する上位ノード（ノード(B)）の DN に、これらの RDN 群を追加することにより生成可能である。

協力関係にあるそれぞれのサービス提供者に対するエイリアスエントリ群（(D)とラベルされたノード）は、ローカルサービスに関する加入者のプロファイル群を配下に格納するノード（ノード(B)）に隣接して格納される。これらのエイリアスエントリもまた、サービス提供者プリフィックス番号を用いて命名できる。これらエイリアスエントリは、エイリアスエントリに対応するローカル DIT 中のノード（(E)とラベルされたノード）を指し示すために用いられる。

DAP オペレーションで使用される場合、ローミングユーザの仮の DN は、ローミングユーザのホーム網のプリフィックスのシャドウである DIT におけるノードを指し示すために展開される。このシャドウ DSE エントリが不完全な下位知識を持つ場合に限り、在圏網は、continuation reference 情報をこのエントリが格納する supplierKnowledge 属性から生成することができる。continuation reference 情報は、実際のプロファイルが存在するローミングユーザのホーム DSA を指し示すものである。すなわち、この操作はローミングしているユーザのホーム DSA にその後 chain されるものである。

Ⅲ.2.2 オブジェクトクラス

Ⅲ.2.2.1 inMobilityUserProfile

inMobilityUserProfile オブジェクトクラスは、格納プロファイル情報として定義されている。次に示す ASN.1 定義は、inMobilityUserProfile object クラスを表現する。

```
inMobilityUserProfile OBJECT-CLASS      ::=         {
        SUBCLASS OF             { top}
        MUST CONTAIN            { inMobilityID |
                                inMobilityPIN |
                                  <other mandatory attributes}
        MAY CONTAIN             { <optional attributes>}
        ID                              Id-oc-inMobilityUserProfile}
```

inMobilityID 属性は、distinguished 属性である。

このタイプのエントリ群は、inMobilityServiceProvider タイプ、若しくは inMobilitySubscriberGroup タイプの配下のみに配置されるものとする。

Ⅲ.2.2.2 inMobilityServiceProvider

このオブジェクトクラスは、配下に inMobilityUserProfile タイプと inMobilitySubscriberGroup タイプのオブジェクトを格納可能とするノードを定義するために設定される。次に示す ASN.1 定義は、inMobilityServiceProvider オブジェクトクラスを表現する。

```
inMobilityServiceProvider               OBJECT-CLASS    ::=     {
        SUBCLASS OF                     { top}
        MUST CONTAIN                    { inMobilityPrefix | <other mandatory attributes>}
        MAY CONTAIN                     { <optional attributes>}
        ID                                      Id-oc-inMobilityServiceProvider}
```

The inMobility プリフィックス属性は、distinguished 属性である。

このタイプの属性は、遠隔の DSA に格納されている、プロファイルエントリ若しくはサブ-プリフィックス（使用されている場合）を指し示すための、non-specific subordinate reference (NSSR)を含むものである。

このオブジェクトクラス配下のエントリ群は、inMobilityUserProfile オブジェクトクラス、若しくは inMobilitySubscriberGroup オブジェクトクラスタイプでなければならない。

Ⅲ.2.2.3 inMobilitySubscriberGroup

このオブジェクトクラスは、配下に inMobilityUserProfile タイプのオブジェクトを格納可能とするノードを定義するために設定される。次に示す ASN.1 定義は、inMobilitySubscriberGroup オブジェクトクラスを表現する。

```
inMobilitySubscriberGroup       OBJECT-CLASS    ::=         {
        SUBCLASS OF             { top}
        MUST CONTAIN            { inMobilitySubPrefix | <other mandatory attributes>}
        MAY CONTAIN             { <optional attributes>}
        ID                              Id-oc-inMobilitySubscriberGroup}
```

The inMobilitySub プリフィックス属性は、the distinguished 属性である。

このオブジェクトクラス配下のエントリ群は、inMobilityUserProfile オブジェクトクラスタイプでなければならない。

Ⅲ.2.3 Attribute Types

Ⅲ.2.3.1 inMobilityID

この属性は、ある特定のサービス提供者若しくはサービス加入者グループに属する、移動体通信サービス加入のユーザ（mobility ユーザ）を個々に識別するために用いられる。

inMobilityID のための ASN.1 定義を次に示す。

```
inMobilityID                        ATTRIBUTE ::= {
        WITH SYNTAX                 Digits (SIZE(lb-inMobilityID..ub-inMobilityID))
        EQUALITY MATCHING RULE      octetStringMatch
        ID                                  id-at-inMobilityID}
```

Ⅲ.2.3.2 inMobilityPIN

この属性は、mobility ユーザの PIN number を格納するために用いられる。

```
inMobilityPIN               ATTRIBUTE ::= {
        WITH SYNTAX         userPassword (SIZE lbinMobilityPIN..ubinMobilityPIN)
        ID                  id-at-inMobilityPIN}
```

Ⅲ.2.3.3 inMobilityPrefix

この属性は、複数の移動体通信サービス提供者間を個々に識別する。

```
inMobilityPrefix                    ATTRIBUTE ::= {
        WITH SYNTAX                 Digits (SIZE(lb-inMobilityPrefix..ub-inMobilityPrefix))
        EQUALITY MATCHING RULE      octetStringMatch
        ID                                  id-at-inMobilityPrefix}
```

Ⅲ.2.3.4 inMobilitySub プリフィックス

この属性は、複数の mobility 加入者を分類するために用いる。

```
inMobilitySubPrefix                 ATTRIBUTE ::= {
        WITH SYNTAX                 Digits (SIZE(lb-inMobilitySubPrefix.. ub-
                                            inMobilitySubPrefix))
        EQUALITY MATCHING RULE      octetStringMatch
        ID                                  id-at-inMobilitySubPrefix}
```

Ⅲ.2.4　DIT 構造定義

Ⅲ.2.4.1　Name Forms

name form は、ある特定オブジェクトクラスの RDN として用いられるべき属性を特定する。

Ⅲ.2.4.1.1.　inMobilityUserProfileNameForm

次に示す name form 定義は、inMobilityID が、inMobilityUserProfile オブジェクトクラスの distinguished 属性として許可された属性であることを示す。

```
inMobilityUserProfileNameForm     NAME-FORM ::= {
        WITH ATTRIBUTES       inMobilityID
        ID                              id-nf-inMobilityUserProfileNameForm}
```

Ⅲ.2.4.1.2.　inMobilityServiceProviderNameForm

次に示す name form 定義は、inMobilityPrefix が、inMobilityServiceProvider オブジェクトクラスの distinguished 属性として許可された属性であることを示す。

```
inMobilityServiceProviderNameForm         NAME-FORM ::= {
        NAMES                     inMobilityServiceProvider
        WITH ATTRIBUTES       inMobilityPrefix
        ID                              id-nf-inMobilityServiceProviderNameForm}
```

Ⅲ.2.4.1.3.　inMobilitySubscriberGroup

次に示す name form 定義は、inMobilitySub プリフィックスが、inMobilitySubscriberGroup オブジェクトクラスの distinguished 属性として許可された属性であることを示す。

```
inMobilitySubscriberGroupNameForm     NAME-FORM ::= {
        NAMES                     inMobilitySubscriberGroup
        WITH ATTRIBUTES       inMobilitySubPrefix
        ID                              id-nf-inMobilitySubscriberGroupNameForm}
```

Ⅲ.2.4.2　構造化規則

構造化規則は、DIT において許可される下位と上位のエントリ群を指定する。次の図 2 に示す構造化規則は、移動体通信サービスに求められる構造化規則を定義するにあたり、その基準として用いることができる。

```
sr1       STRUCTURE-RULE ::= {
                NAME-FORM          countryNameForm
                ID                          1}

sr2       STRUCTURE-RULE ::= {
                NAME-FORM          orgNameForm
                SUPERIOR RULES     sr1
                ID                          1}
                ...
```

sr<n>　　STRUCTURE-RULE ::= {

NAME-FORM　　inMobilityServiceProviderNameForm

SUPERIOR RULES sr<n-1>

ID　　<n>}

sr<n+1>STRUCTURE-RULE ::= {

NAME-FORM　　inMobilitySubscriberGroupNameForm

SUPERIOR RULES sr<n>

ID　　<n+1>}

sr<n+2>STRUCTURE-RULE ::= {

NAME-FORM　　inMobilityUserProfileNameForm

SUPERIOR RULES　　sr<n>, sr<n+1>

ID　　<n+2>}

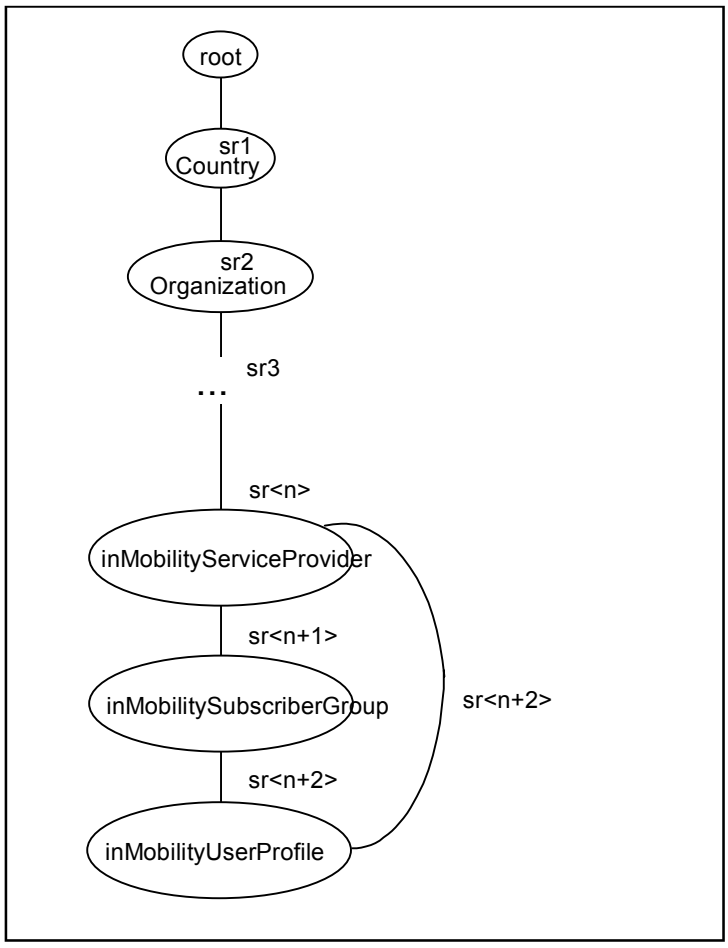


図　付録 3-2:構造化規則

III.2.4.3　オブジェクト識別子の割り当て

次に示すオブジェクト識別子の割り当ては、X.500.において移動体用オブジェクトを識別するために用いることができる。

id-at-inMobilityID	OBJECT IDENTIFIER ::= {id-at-inMobility 0}
id-at-inMobilityPIN	OBJECT IDENTIFIER ::= {id-at-inMobility 1}
id-at-inMobilityPrefix	OBJECT IDENTIFIER ::= {id-at-inMobility 2}
id-at-inMobilitySubPrefix	OBJECT IDENTIFIER ::= {id-at-inMobility 3}
id-oc-inMobilityUserProfile	OBJECT IDENTIFIER ::= {id-oc-inMobility 0}
id-oc-inMobilityServiceProvider	OBJECT IDENTIFIER ::= {id-oc-inMobility 1}
id-oc-inMobilitySubscriberGroup	OBJECT IDENTIFIER ::= {id-oc-inMobility 2}
id-nf-inMobilityUserProfileNameForm	OBJECT IDENTIFIER ::= {id-nf-inMobility 0}
id-nf-inMobilityServiceProviderNameForm	OBJECT IDENTIFIER ::= {id-nf-inMobility 1}
id-nf-inMobilitySubscriberGroupNameForm	OBJECT IDENTIFIER ::= {id-nf-inMobility 2}

付録Ⅳ　SCSM 及び SDSM プロセスの記述

１．SCSM プロセスの記述

SCF 側のアプリケーションプロセスがエンドユーザに代わって SCF と SDF の間の結合を生成する場合、SCSM プロセスのインスタンスがアプリケーションプロセスによって生成される。結合を行う中で、認証とバージョンの折衝という二つのプロセスが生ずる。認証はユーザとディレクトリに関係がある。標準は認証手法として以下を提供する。

・パスワードの利用
・保護されたパスワードの利用

SCSM プロセスのインスタンスが生成されるのは、SCSM が"空き"状態にある時である。

1.1　状態１:"空き"

1.1.1　通常手順

SCSM プロセスのインスタンスが"空き"状態にあり、アプリケーションプロセスが SDF への問い合わせを必要とする場合、結合起動プリミティブが SCF アプリケーションプロセスによって送信される。結合のアーギュメントが蓄積され、タイマ T1 が始動される。このイベントにより"後続要求待ち"状態への遷移が生じ、他のイベントは待たされる。タイマ T1 は"後続要求待ち"状態を監視するために始動され、アプリケーションプロセスとの通信が中断された場合にはSCSMプロセスが削除されることを保証する。TCの対話処理サービスは、ディレクトリ結合オペレーションをサポートし、対応する SDF プロセスのインスタンスに対して関連する APDU を送信する契機を与えるために使用される。

1.1.2　例外手順

特になし。

1.2　状態２:"後続要求待ち"

1.2.1　通常手順

本状態において、結合オペレーションと共に（同一のメッセージで）SDF に対して送信されるべき後続オペレーションが予想される。通常、以下のイベントが本状態において想定される。

・オペレーション起動プリミティブがアプリケーションプロセスから受信されるとコンポーネント処理プリミティブ、TC-INVOKE 要求が TC に送信され、プロセスのインスタンスは"後続要求待ち"状態に戻る。このオペレーションは結合のアーギュメントを含むメッセージで SDF に送信される。SCSM プロセスのインスタンスは、アプリケーションプロセスからのデリミタプリミティブによる応答を待つ。TC-INVOKE 要求プリミティブによって伝えられるべきオペレーションは、探索、エントリ更新、エントリ追加、あるいはエントリ削除である。

・デリミタプリミティブがアプリケーションプロセスから受信されるとSCSMプロセスのインスタンスは、タイマT1を停止させタイマT2を始動し、対話処理プリミティブTC-BEGIN要求をTCに送信する。TC-BEGIN要求プリミティブがTCによって受信されると、結合のアーギュメント及び存在する場合には他オペレーションのアーギュメントを含んだメッセージがSDFに送信される。このイベントによってSCSMプロセスのインスタンスは、状態３"結合結果待ち"へ遷移する。SCSMプロセスのインスタンスはSDFからの応答を待つ。タイマT2は、"結合結果待ち"状態を監視するために始動され、対応するSDFのアプリケーションプロセスとの通信が中断された場合にはSCSMプロセスが削除されることを保証する。

1.2.2　例外手順

・タイマT1が満了しSCSMプロセスのインスタンスが"後続要求待ち"にある時、"対話終結"プリミティブが、蓄積された状態で転送を待っていたディレクトリ起動オペレーションを破棄するため、TCのコンポーネント処理に対して送信される。"プロセスエラー"プリミティブがSCFアプリケーションのプロセスに送信され、SCSMプロセスは終結される。

・SCFアプリケーションのプロセスは、対応するSDFとの通信の中止を要求する場合、"AP_U_アボート"プリミティブをSCSMプロセスのインスタンスに対して送信する。SCSMプロセスのインスタンスは、蓄積された状態で転送を待っていたディレクトリ起動オペレーションを破棄するため、"対話終結"プリミティブをTCのコンポーネント処理に対して送信する。SCSMプロセスは終結される。

1.3　状態３:"結合結果待ち"

1.3.1　通常手順

本状態において、SCFはSDFからの応答を待っている。TCからのTC-CONTINUE指示プリミティブによって、以前にSDFに発行した結合オペレーションに対する応答を受信すると、タイマT2は停止され結合応答プリミティブがアプリケーションプロセスに送信される。SCFの"SDFと結合中"状態への遷移が生ずる。

1.3.2　例外手順

・タイマT2が満了し、SCSMプロセスのインスタンスが"結合結果待ち"状態にある時、対応するSDFプロセスとの間に開始されている対話を中止する為に、"TC-U-ABORT"要求プリミティブがTCの対話処理に送られる。"プロセスエラー"プリミティブがSCFアプリケーションのプロセスに送信され、SCSMプロセスは終結される。

・SCFアプリケーションのプロセスは、対応するSDFとの通信の中止を要求する場合、"AP_U_アボート"プリミティブをSCSMプロセスのインスタンスに対して送信する。SCSMプロセスのインスタンスは、対応するSDFプロセスとの間に開始されている対話を中止する為に、TCの対話処理に"TC-U-ABORT"要求プリミティブを送信し、SCSMプロセスは終結される。

・TC-U-ABORT指示プリミティブをTCから受信し、対応するSDFプロセスからのディレクトリ結合エラーオペレーションが存在するとき、このオペレーションは"結合エラー"プリミティブによってアプリケーションプロセスに渡され、SCSMプロセスは終結される。

・TC-U-ABORT指示プリミティブをTCから受信し、対応するSDFプロセスからのディレクトリ結合オペレーションが存在しないとき、"U_アボート"プリミティブがアプリケーションプロセスに渡され、SCSMプロセスは終結される。

1.4　状態 4:"SDF と結合中"

1.4.1　通常手順

本状態において、SCF は SDF に対して認証済みアクセスを行っており、サービス論理からの SDF に対する要求、あるいは以前に SDF に発行されたオペレーションに対する応答を待っている。

・オペレーション起動プリミティブをアプリケーションプロセスから受信すると、コンポーネント処理プリミティブ、TC-INVOKE 要求が TC に対して送信され、プロセスのインスタンスは"SDF と結合中"状態に戻る。SCSM プロセスのインスタンスはアプリケーションプロセスからのデリミタプリミティブを待つ。TC-INVOKE 要求プリミティブによって伝えられるオペレーションは、探索、エントリ更新、エントリ追加、あるいはエントリ削除である。

・"デリミタ"プリミティブがアプリケーションプロセスから受信されると、リモート SDF のアプリケーションプロセスのインスタンスに対して受信したオペレーションを転送する為、TC-CONTINUE 要求プリミティブが TC に送信される。

・以前に SDF に発行されたオペレーションに対する応答の受信が成功したことによって TC から TC-RESULT 指示プリミティブが受信されると、SCSM プロセスのインスタンスはアプリケーションプロセスにオペレーションのリターン結果を送信し、同じ状態に戻る。

・SCF のアプリケーションプロセスから起動されたオペレーションが、対応する SDF 側において失敗したことを示す TC-U-ERROR 指示プリミティブが TC から受信されると、SCSM プロセスはアプリケーションプロセスにオペレーションのリターンエラーを送信し、同じ状態に戻る。

・TC のコンポーネントサブレイヤが不当なオペレーションを受信したことを示す TC-L-REJECT 指示プリミティブが受信されると、SCSM プロセスのインスタンスはアプリケーションプロセスにオペレーション拒否を送信し、同じ状態に戻る。

・オペレーションがリモートのコンポーネントサブレイヤによって拒否されたことを示す TC-R-REJECT 指示プリミティブが TC から受信されると、SCSM プロセスのインスタンスはアプリケーションプロセスにオペレーション拒否を送信し、同じ状態に戻る。

・オペレーションがリモートのアプリケーションプロセスのインスタンスによって拒否されたことを示す TC-U-REJECT 指示プリミティブが TC から受信されると SCSM プロセスのインスタンスはアプリケーションプロセスにオペレーション拒否を送信し、同じ状態に戻る。

・タイムアウト条件によりオペレーションの起動がローカルに終結されたことを示す TC-L-CANCEL 指示プリミティブが TC から受信されると SCSM プロセスのインスタンスはアプリケーションプロセスに通知を行い、同じ状態に戻る。

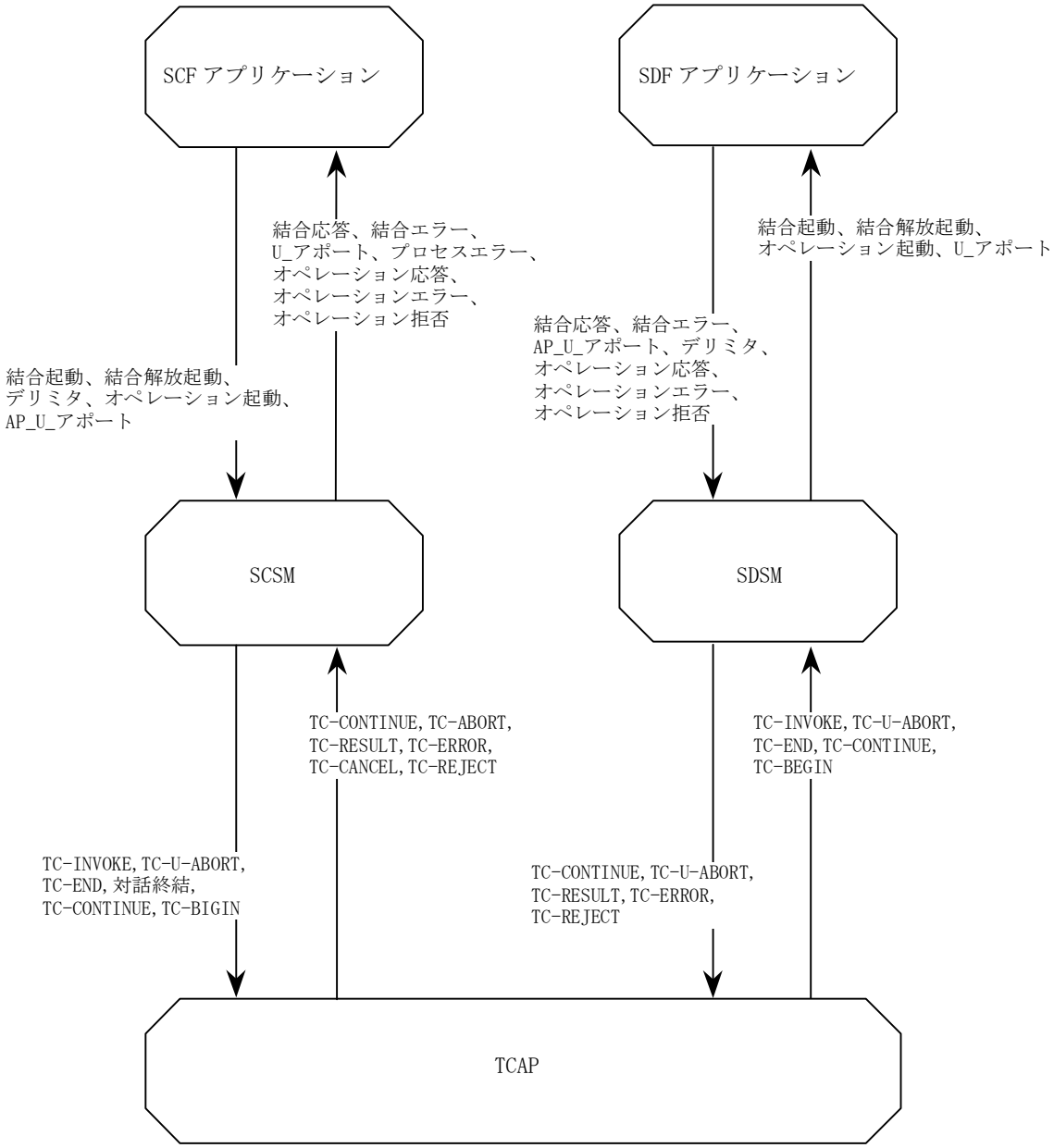
・SDF に対する認証済みアクセスを終結させる必要が生じたことによって、SCF のアプリケーションプロセスから結合解放起動プリミティブが受信されると、リモート SDF のアプリケーションプロセスとの対話を終了するために、TC-END 要求プリミティブが TC に送信される。SCSM プロセスのインスタンスは終結される。

1.4.2 例外手順

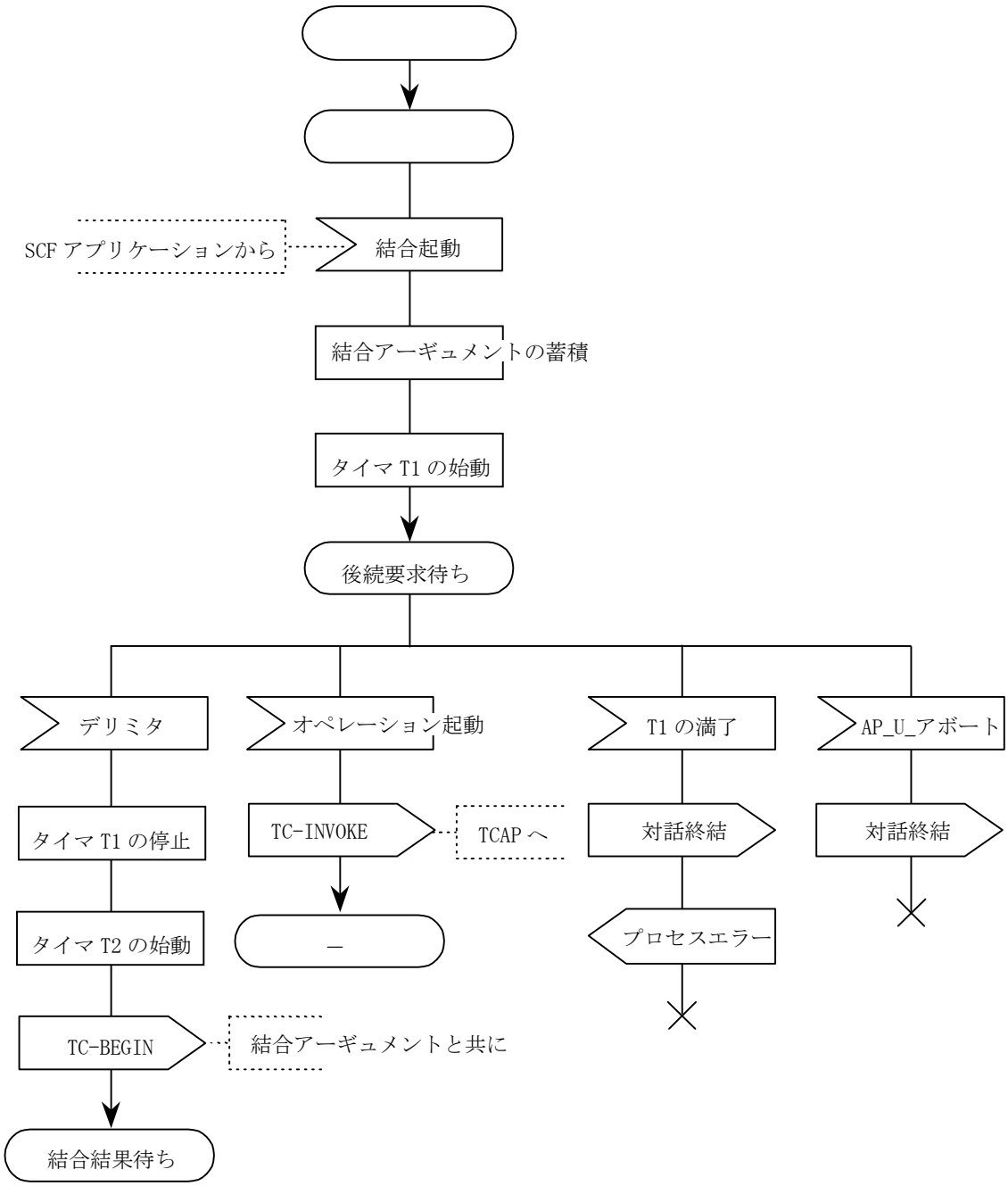
・SCF のアプリケーションプロセスは、対応する SDF との通信の中止を要求する場合、"AP_U_アボート"プリミティブを SCSM プロセスのインスタンスに対して送信する。SCSM プロセスのインスタンスは、対応する SDF プロセスとの間に開始された対話を中止するため、"TC-U-ABORT"要求プリミティブを TC の対話処理に対して送信する。SCSM プロセスは終結される。

・TC-U-ABORT 指示プリミティブが TC から受信されると、"U_アボート"プリミティブがアプリケーションプロセスに対して渡され、SCSM プロセスは終結される。

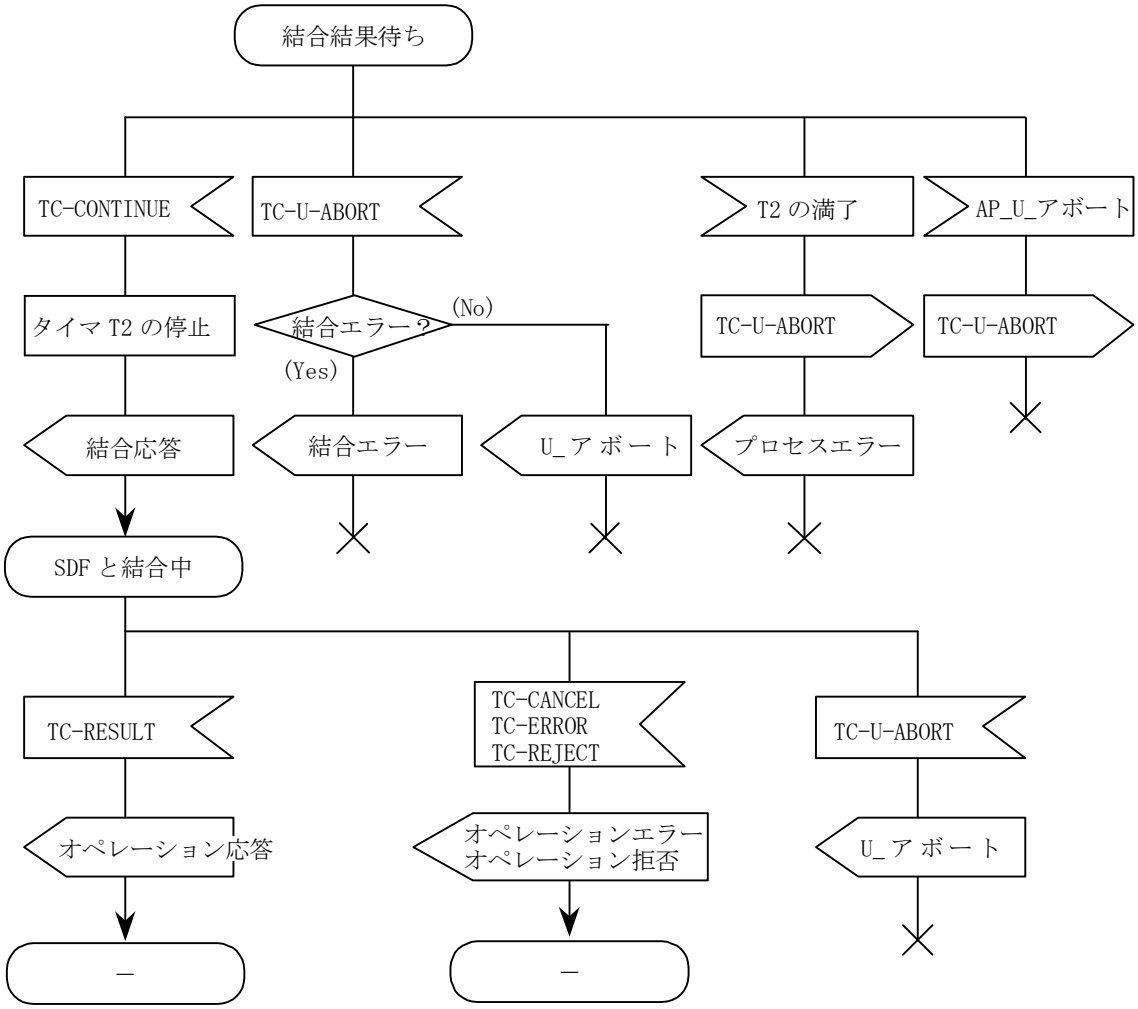
SCF、SDF ブロック



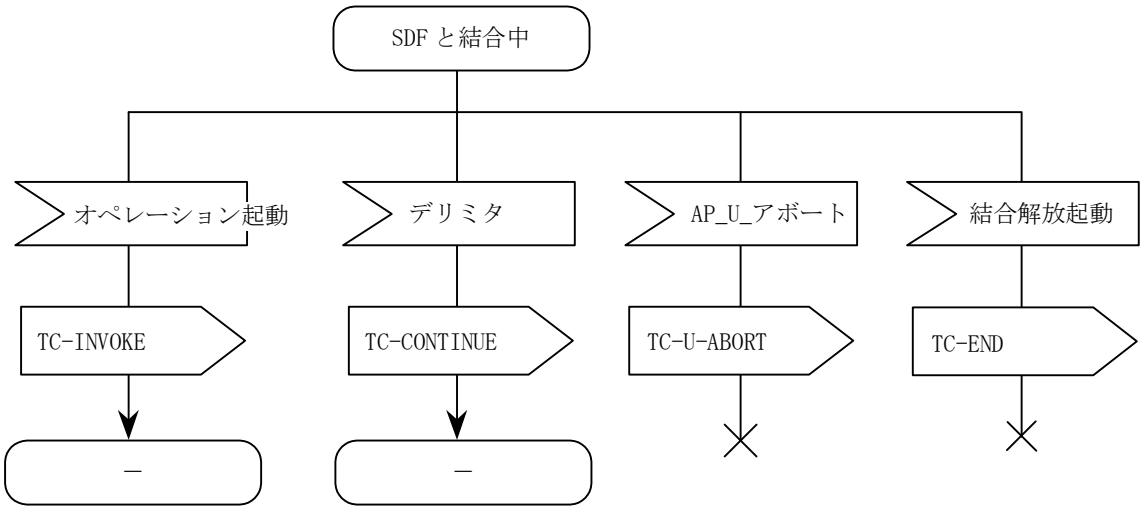
SCSM プロセス(1/3)


SCF アプリケーションから
結合起動
結合アーギュメントの蓄積
タイマ T1 の始動
後続要求待ち
デリミタ
オペレーション起動
T1 の満了
AP_U_アボート
タイマ T1 の停止
TC-INVOKE
TCAP へ
対話終結
対話終結
タイマ T2 の始動
－
プロセスエラー
TC-BEGIN
結合アーギュメントと共に
結合結果待ち

SCSM プロセス(2/3)


結合結果待ち
TC-CONTINUE
TC-U-ABORT
T2 の満了
AP_U_アボート
タイマ T2 の停止
結合エラー？
(No)
(Yes)
TC-U-ABORT
TC-U-ABORT
結合応答
結合エラー
U_アボート
プロセスエラー
SDF と結合中
TC-RESULT
TC-CANCEL
TC-ERROR
TC-REJECT
TC-U-ABORT
オペレーション応答
オペレーションエラー
オペレーション拒否
U_アボート
－
－

SCSM プロセス(3/3)


SDF と結合中
オペレーション起動
デリミタ
AP_U_アボート
結合解放起動
TC-INVOKE
TC-CONTINUE
TC-U-ABORT
TC-END
－
－

２．SDSM プロセスの記述

SDF ディレクトリ抽象サービスは、エンドユーザに代わっての SCF からの要求により SDF が実行できる多くのオペレーションの形で定義される。これらのオペレーションを起動する前に、SCF はいくつかのアクセスポイントで結合オペレーションを実行する。実行後、SCF はオペレーションを起動する。それぞれの起動は、実行されるべきオペレーションを特定し、実際の要求を定義するアーギュメントを運ぶ。

2.1 状態１：”空き”

2.1.1 通常手順

TC の対話処理サービスがディレクトリ結合オペレーションを含む TC-BEGIN 指示を送出する場合､SDSM プロセスのインスタンスが生成され、”結合起動”プリミティブがアプリケーションプロセスに送出されるとともに、タイマーT3 が始動される。このイベントにより、状態は”結合処理中”に遷移する。タイマ T3 は”結合処理中”状態を監視するために、始動され、アプリケーションプロセスとの通信が中断された場合には SDSM プロセスが削除されることを保証する。

2.1.2 例外手順

特になし。

2.2 状態２”結合処理中”

2.2.1 通常手順

本状態では、結合要求が SCF から受信されている。SDF は結合の処理と同時に SCF へのアクセスコントロールを実行する（例えば、アクセスに関する認証）。しかし、ディレクトリ結合オペレーションがダミーということも有り得る。この時、アクセスに関する認証は不要である。本状態では、以下の２つのイベントが想定される。

・アプリケーションプロセスから結合オペレーションの結果が送られる前に、オペレーションの受信による TC-INVOKE 指示コンポーネント処理プリミティブを受信すると、”オペレーション起動”オペレーション指示プリミティブがアプリケーションプロセスに送信され、SDSM プロセスは”結合処理中”状態に戻る。

・結合処理が成功すると、SDF アプリケーションプロセスは SDSM のプロセスのインスタンスに、ディレクトリ結合の結果応答を含む”結合応答”プリミティブを送信する。”結合応答”プリミティブを受信すると、SDSM プロセスのインスタンスはタイマ T3 を停止し、アプリケーションプロセスから”デリミタ”プリミティブが受信されるまで情報をローカルに蓄積し、”SCF と結合中”状態に遷移する。

2.2.2 例外手順

・結合要求が失敗した場合、アプリケーションプロセスは、ディレクトリ結合エラーを含む”結合エラー”プリミティブを SDSM プロセスのインスタンスに送信する。SDSM プロセスのインスタンスは、対応する SCF のプロセスとの間で開始した対話を中止するために、TC の対話処理に”TC_U_ABORT”要求を送信し、SDSM プロセスは終結される。

・タイマ T3 が満了し、SDSM プロセスのインスタンスが”結合処理中”状態である場合、対応する SCF のプロセスとの間で開始した対話を中止するために、”TC_U_ABORT”要求プリミティブが TC の対話処理に、”プロセスエラー”プリミティブが SDF のアプリケーションプロセスに送信され、SDSM プロセスは終結される。

・SDF のアプリケーションプロセスが対応する SCF のプロセスとの間の通信の中止を要求する場合、“AP_U_アボート”プリミティブを SDSM プロセスのインスタンスに対して送信する。SDSM プロセスのインスタンスは、対応する SCF プロセスとの間に開始されている対話を中止するために、”TC_U_ABORT”要求プリミティブを TC の対話処理に送信し、SDSM プロセスは終結される。

2.3 状態３：”SCF と結合中”

2.3.1 通常手順

この状態において、SDF に対する SCF のアクセスは認証されており、SCF からのオペレーションが受け付けられる。SCF からの要求待ちの他に、この状態で SDF は以前に発行されたオペレーションへの応答を送信することができる。本状態では、以下のイベントが想定される。

・アプリケーションプロセスから、オペレーションが正しく実行されたことを示すオペレーション応答プリミティブを受信した場合、TC にコンポーネント処理プリミティブである”TC_RESULT”要求が送信され、プロセスのインスタンスは”SCF と結合中”状態に戻る。SDSM プロセスのインスタンスは、TC 対話処理エンティティに対して TC_CONTINUE 要求プリミティブを送信するために、アプリケーションプロセスからのデリミタプリミティブを待つ。これにより、TC はこの結果応答を対応する SCSM プロセスのインスタンスに転送することが可能となる。TC_RESULT 要求プリミティブによって伝えられるべき結果応答は、以前に受信した探索、エントリ更新、エントリ追加、あるいはエントリ削除オペレーションからのもので有り得る。

・アプリケーションプロセスから、オペレーションが正しく実行されなかったことを示すオペレーションエラープリミティブを受信した場合、TC にコンポーネント処理プリミティブである”TC_ERROR”要求が送信され、プロセスのインスタンスは”SCF と結合中”状態に戻る。SDSM プロセスのインスタンスは、TC 対話処理エンティティに対して TC_CONTINUE 要求プリミティブを送信するために、アプリケーションプロセスからのデリミタプリミティブを待つ。これにより、TC はこの応答エラーを対応する SCSM プロセスのインスタンスに転送することが可能となる。TC_ERROR 要求プリミティブによって伝えられるべき応答エラーは、以前に受信した探索、エントリ更新、エントリ追加、あるいはエントリ削除オペレーションからのもので有り得る。

・アプリケーションプロセスからデリミタプリミティブを受信すると、SDSM プロセスのインスタンスは TC に対話処理プリミティブの TC_CONTINUE 要求を送信する。いったん TC が TC_CONTINUE 要求プリミティブを受信すると、結合の結果応答が、SCFに送信される。本イベントにより、SDSMプロセスのインスタンスは”SCFと結合中”状態に戻る。

・SCF からのディレクトリ結合解放オペレーションを受信したことにより、TC の対話処理から TC_END 指示プリミティブを受信すると、SDSM プロセスのインスタンスは、アプリケーションプロセスに”結合解放起動”プリミティブを送信し、SDSM プロセスのインスタンスを終結させる。SCF-SDF のアソシエーションは終了し、関連する全てのリソースは解放される。

・TC から TC_INVOKE 指示コンポーネント処理プリミティブを受信すると、アプリケーションプロセスに”オペレーション起動”プリミティブが送信され、SDSM プロセスのインスタンスは”SCF と結合中”状態に戻る。

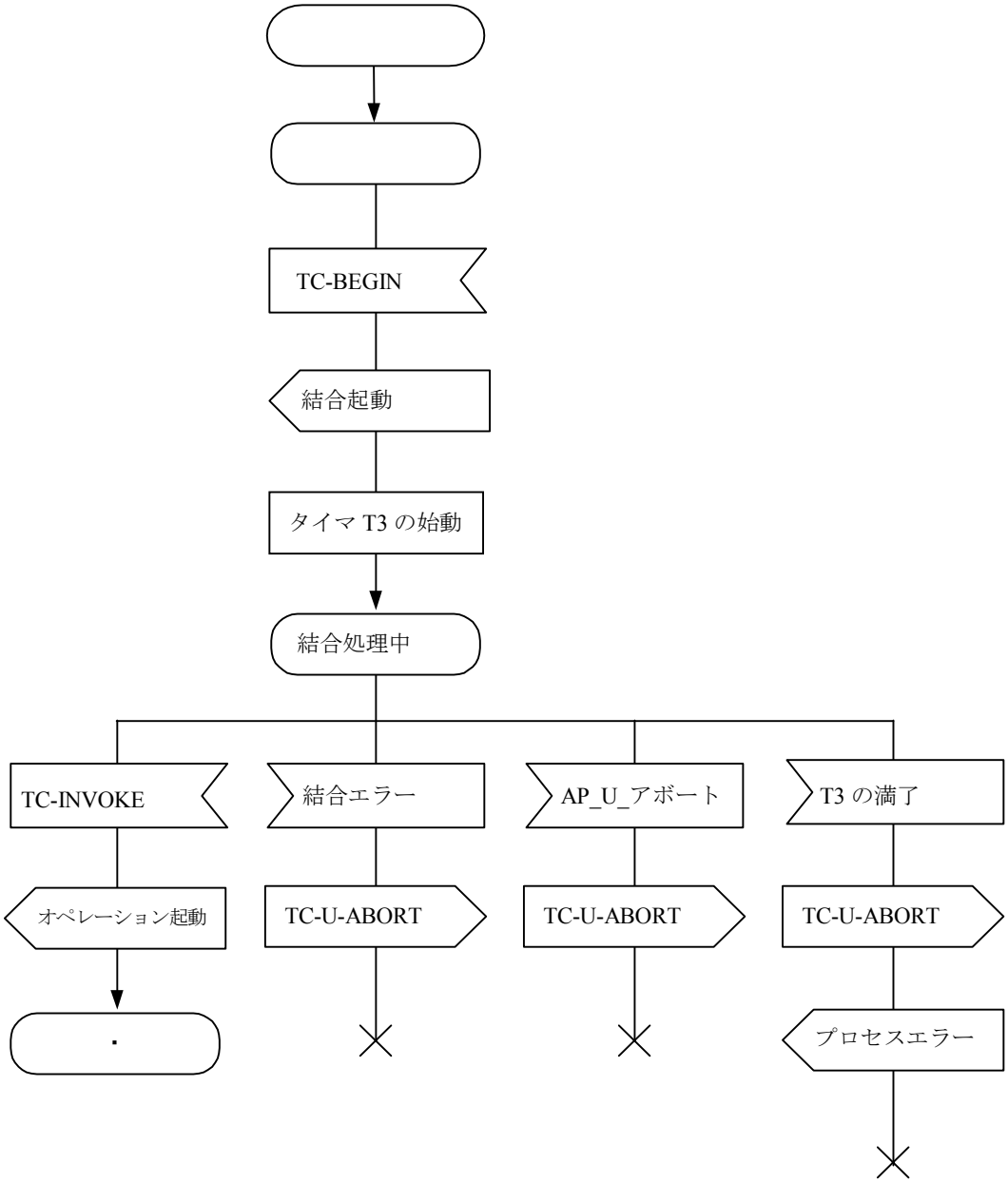
2.3.2 例外手順

・アプリケーションプロセスから以前に受信したオペレーションが拒否されたことを示すオペレーション拒否プリミティブを受信すると、コンポーネント処理プリミティブの TC_REJECT 要求が TC に送信され、プロセスのインスタンスは”SCF と結合中”状態に戻る。SDSM プロセスのインスタンスは、TC 対話処理エンティティに TC_CONTINUE 要求プリミティブを送信するためにアプリケーションプロセスからのデリミタプリミティブ応答を待つ。これにより、TC はこの拒否を対応する SCSM プロセスのインスタンスに転送することが可能となる。TC_RELECT 要求プリミティブによって伝えられるべき拒否は、以前に受信した探索、エントリ更新、エントリ追加、あるいはエントリ削除オペレーションからのもので有り得る。

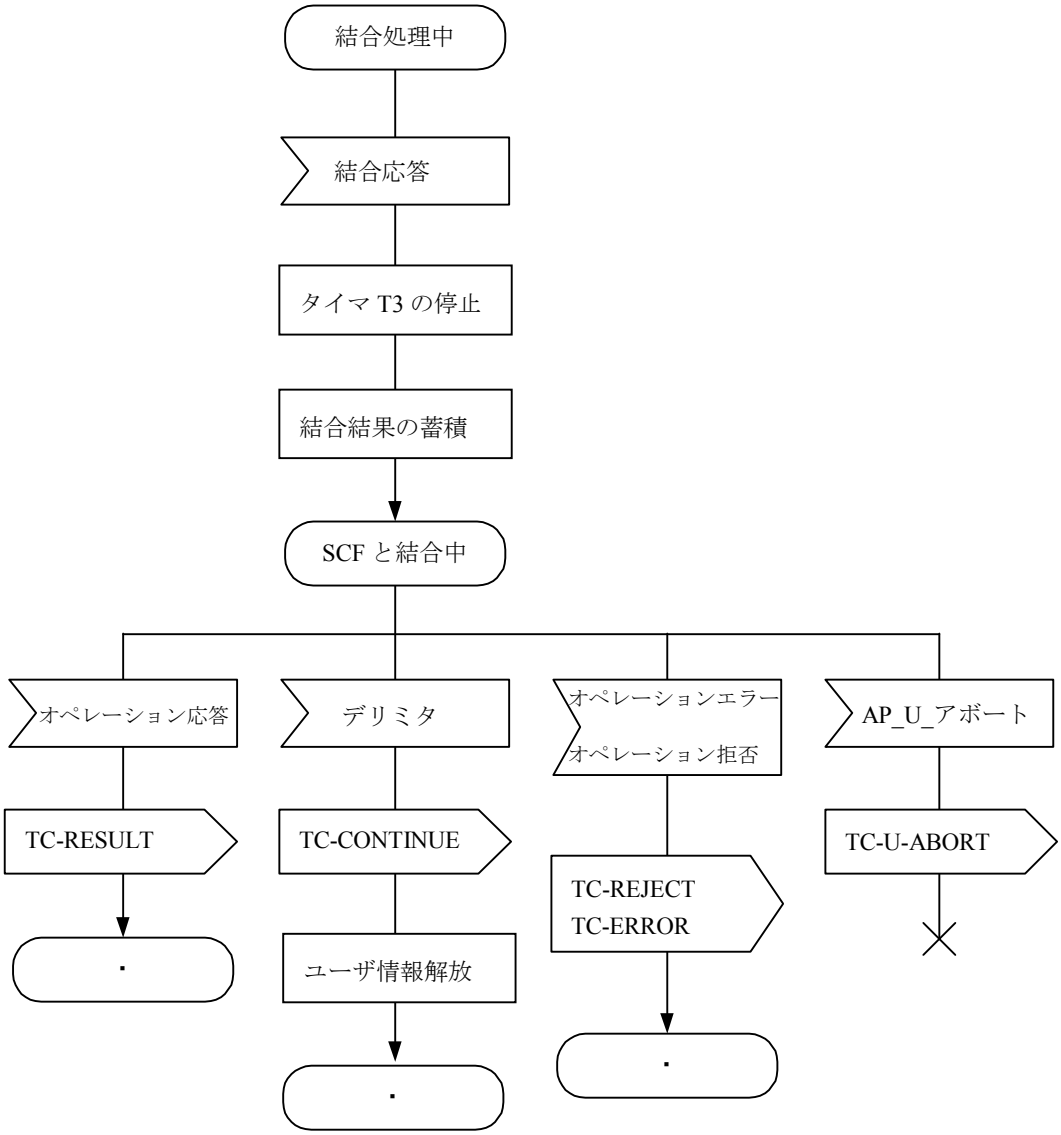
・TC から TC_U_ABORT 指示プリミティブを受信すると、”U_アボート”プリミティブがアプリケーションプロセスに渡され、SDSM プロセスは終結する。

・SDF のアプリケーションプロセスが対応する SCF のプロセスとの間の通信の中止を要求する場合、”AP_U_アボート”プリミティブを SDSM プロセスのインスタンスに対して送信する。SDSM プロセスのインスタンスは、対応する SCF プロセスとの間に開始されている対話を中止するために、”TC_U_ABORT”要求プリミティブを TC の対話処理に送信し、SDSM プロセスは終結される。

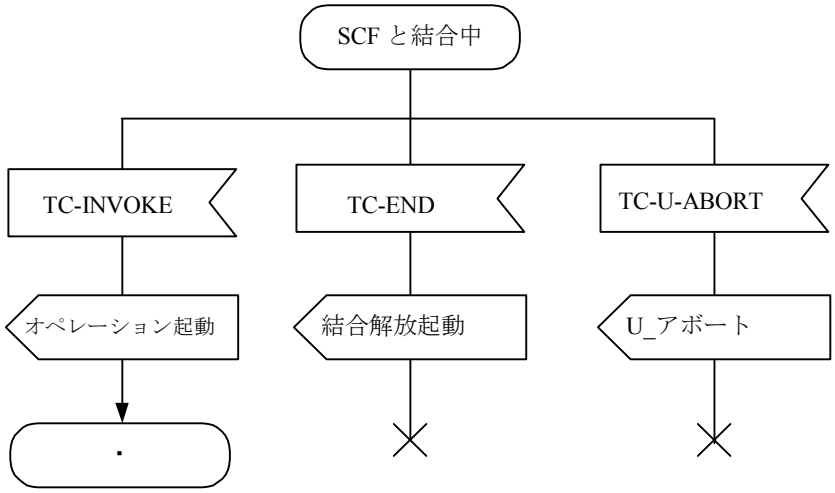
SDSM プロセス(1/3)


TC-BEGIN
結合起動
タイマ T3 の始動
結合処理中
TC-INVOKE
結合エラー
AP_U_アボート
T3 の満了
オペレーション起動
TC-U-ABORT
TC-U-ABORT
TC-U-ABORT
プロセスエラー

SDSM プロセス(2/3)


結合処理中
結合応答
タイマ T3 の停止
結合結果の蓄積
SCF と結合中
オペレーション応答
デリミタ
オペレーションエラー
オペレーション拒否
AP_U_アボート
TC-RESULT
TC-CONTINUE
TC-REJECT
TC-ERROR
TC-U-ABORT
ユーザ情報解放

SDSM プロセス(3/3)



第３版作成協力者（１９９６年１月３１日現在）

第一部門委員会　　（敬称略）

部門委員長	川口　憲一	国際電信電話（株）
副部門委員長	庄司　滋彦	日本電信電話（株）
副部門委員長	林　和行	（株）日立製作所
	橘　薫	第二電電（株）
	稲葉　安男	東京通信ネットワーク（株）
	薗田　宏	沖電気工業（株）
	山口　健二	日本電気（株）
	遠藤　一美	富士通（株）
	中尾　康二	国際電信電話（株）
	大西　邦宏	日本電信電話（株）
	星野　隆資	日本電信電話（株）
	関口　幹夫	日本無線（株）
	岡田　忠信	日本電信電話（株）
	久保　征英	富士通（株）
	中野　栄	三菱電機（株）
	北見　憲一	日本電信電話（株）
	松下　正彦	日本電信電話（株）
	益田　淳	国際電信電話（株）
	中島　昭久	ＮＴＴ移動通信網（株）

（敬称略）

第一部門委員会　第一専門委員会

専門委員長	遠藤　一美	富士通（株）
副専門委員長	中尾　康二	国際電信電話（株）
副専門委員長	大西　邦宏	日本電信電話（株）
	泊　哲郎	国際ﾃﾞｼﾞﾀﾙ通信（株）
	藤田　増之	国際電信電話（株）
	竹原　啓五	第二電電（株）
	佐口　雅広	東京通信ネットワーク（株）
	清水　悟	日本高速通信（株）
	柳下　健二	日本国際通信（株）
	吉村　隆之	日本テレコム（株）
	大羽　巧	日本電信電話（株）
	大貫　雅史	ＮＴＴ移動通信網（株）
	宮北　弘	（株）東京デジタルホン
	西田　護	（株）四国情報通信ネットワーク
	堀　智尚	中部テレコミュニケーション（株）
	榎本　一夫	日本移動通信（株）
	懸樋　恒久	大阪メディアポート（株）
	近　義起	DDI東京ﾎﾟｹｯﾄ電話（株）
	渡邉　恭行	（株）アステル東京
	山田　博	（株）インテック
	後藤　雅徳	沖電気工業（株）
	田村　慶章	（株）東芝
	岩本　真人	日本ﾃﾞｨｼﾞﾀﾙ ｲｸｲｯﾌﾟﾒﾝﾄ （株）
	山口　健二	日本電気（株）
	境　穣	日本無線（株）
	上岡　貞雄	日本モトローラ（株）
	岡崎　稔	ﾉｰｻﾞﾝﾃﾚｺﾑｼﾞｬﾊﾟﾝ（株）
	新保　勲	（株）日立製作所
	坪井　洋治	富士通（株）
	大塚　晃	三菱電機（株）
	住田　正臣	日本エリクソン（株）
	浜田　啓嗣	日本情報通信ｺﾝｻﾙﾃｨﾝｸﾞ（株）
特別専門委員	松本　弘行	国際電信電話（株）
特別専門委員	幕田　和彦	日本高速通信（株）
ＴＴＣ事務局	小森　秀夫	

（ＪＴ－Ｑ１２１８　検討グループ）　　（敬称略）

リーダ	中尾　康二	国際電信電話（株）
特別専門委員	栢森　夏樹	国際ﾃﾞｼﾞﾀﾙ通信（株）
特別専門委員	山口　明	国際電信電話（株）
特別専門委員	酒井　清一郎	国際電信電話（株）
委員	竹原　啓五	第二電電（株）
委員	佐口　雅広	東京通信ネットワーク（株）
特別専門委員	幕田　和彦	日本高速通信（株）
特別専門委員	中里　浩二	日本テレコム（株）
特別専門委員	吉見　正信	日本電信電話（株）
特別専門委員	丸山　康夫	ＮＴＴ移動通信網（株）
委員	西田　護	（株）四国情報通信ネットワーク
委員	堀　智尚	中部テレコミュニケーション（株）
委員	榎本　一夫	日本移動通信（株）
委員	近　義起	DDI東京ﾎﾟｹｯﾄ電話（株）
特別専門委員	小林　稔和	沖電気工業（株）
委員	田村　慶章	（株）東芝
特別専門委員	山本　卓	日本ﾃﾞｨｼﾞﾀﾙ ｲｸｲｯﾌﾟﾒﾝﾄ （株）
委員	山口　健二	日本電気（株）
特別専門委員	柴　博昭	日本電気（株）
委員	境　穣	日本無線（株）
特別専門委員	佐久間　哲雄	（株）日立製作所
特別専門委員	野村　一郎	富士通（株）
特別専門委員	森　陽子	富士通（株）
特別専門委員	木下　裕介	三菱電機（株）
委員	住田　正臣	日本エリクソン（株）