TTC技術レポート
Technical Report

TR-DTS03016

プロトコルフレームワーク定義；一般（メタプロトコル）

Protocol Framework Definition;
General(meta-protocol)

第１版

2002 **年** 9 **月** 5 **日制定**

社団法人
情報通信技術委員会

THE TELECOMMUNICATION TECHNOLOGY COMMITTEE

TTC Telecommunication Technology Committee

目 次

<要約>

1　技術書作成の経緯

本技術書は、TIPHON サービスにおけるプロトコルフレームワークを定義しており、ETSI 標準 DTS03016 （TS101 882 V.1.1.1 (2002-05)）に準拠している。

2　原標準との差分

本技術書は原標準の紹介を目的としているため、原標準との差分はない。

3　改版履歴

版数	発行日	改版内容
第 1 版	2002 年 9 月 5 日	制定

4 参照している勧告および標準

ETSI 標準: TS 101 314、TR 101 877、TS 101 878、TS 101 315、TR 101 301、TR 101 835、TR 102 008、TR 101 311、TS 101 883、TS 101 885、TS 101 520、TS 101 521、TS 101 522、TS 101 804

ISO/IEC　: ISO/IEC 9646-7

5 技術レポート作成部門

第四部門委員会　第五専門委員会　サブワーキンググループ２

Intellectual Property Rights

IPRs essential or potentially essential to the present document may have been declared to ETSI. The information pertaining to these essential IPRs, if any, is publicly available for **ETSI members and non-members**, and can be found in SR 000 314: *"Intellectual Property Rights (IPRs); Essential, or potentially Essential, IPRs notified to ETSI in respect of ETSI standards"*, which is available **free of charge** from the ETSI Secretariat. Latest updates are available on the ETSI Web server (http://www.etsi.org/ipr/homw.asp)

Pursuant to the ETSI IPR Policy, no investigation, including IPR searches, has been carried out by ETSI. No guarantee can be given as to the existence of other IPRs not referenced in SR 000 314 (or the updates on the ETSI Web server) which are, or may be, or may become, essential to the present document.

はじめに（Foreword）

この技術規定（TS）は ETSI プロジェクト Telecommunications and Internet Protocol Harmonization Over Networks（TIPHON）により作成された。

導入（Introduction）

本ドキュメントは、TR 101 835 [7]に記述されている TIPHON 開発過程手順 D（stepD）の TIPHON リリース 3（TR 101 301[6]）におけるの成果の一つである。

現在進められている TIPHON の標準化アプローチは、PSTN、ISDN そして GSM 向けに過去において適用された手法を起点にしている。また、装置およびサービスの設計における技術刷新の促進のため、より広い検討スコープを許容するものである。さらに、異なる技術に基づく網を含む、複数網の相互接続を介したサービス提供を容易にするため、適切な標準仕様を提供するものである。本ドキュメントは、サービス能力の初期コアセットを提示するものである。ここで言うサービス能力とは、より先進的なサービスの継続的な開発を促す一方で、既存の PSTN サービスと安全にインターワークするであろう、TIPHON 網上のサービスをプロバイダが提供することを可能とするために要求されると考えられるものである。

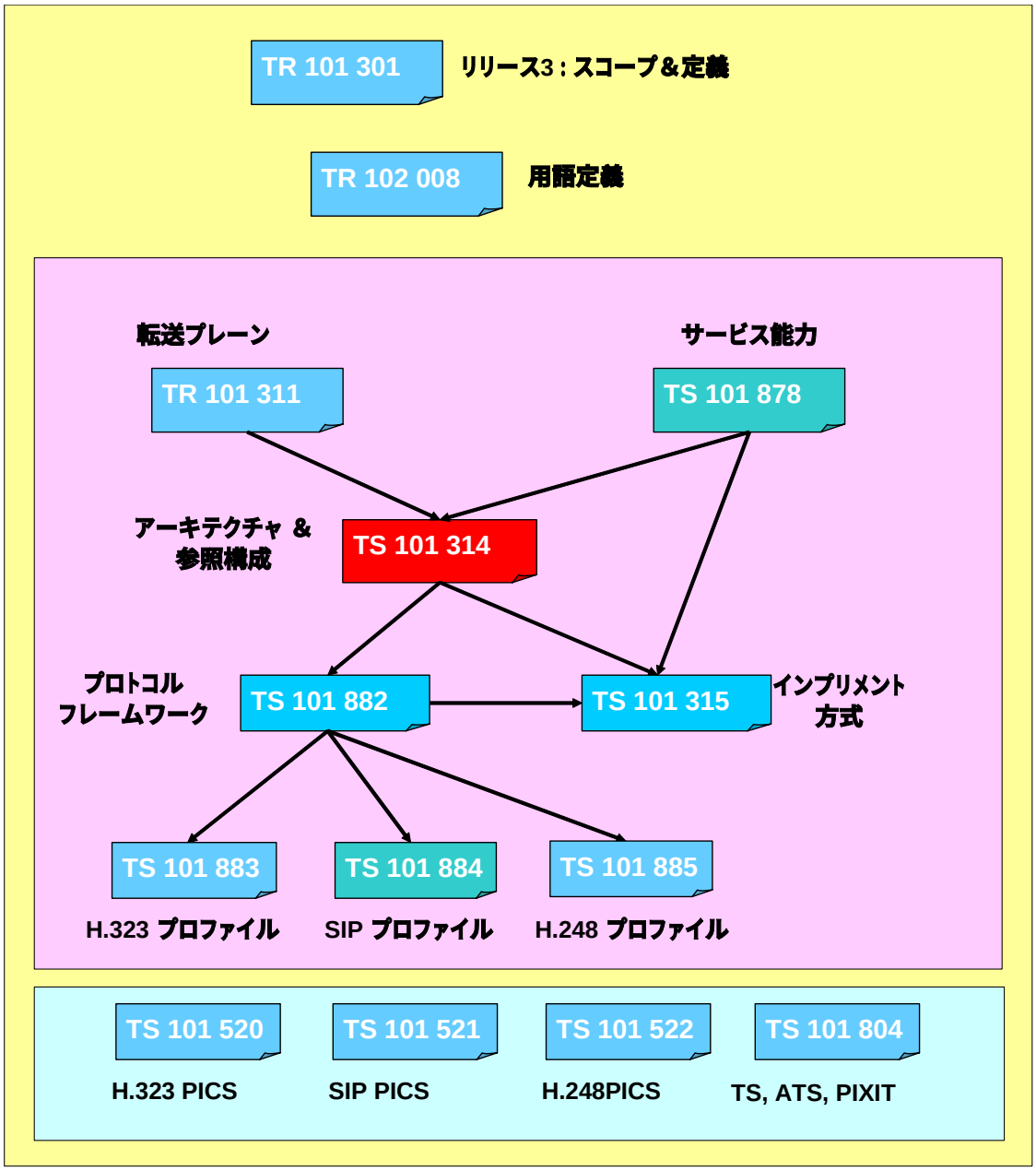


図 1　その他の TIPHON リリース 3 ドキュメントとの関係

- TR 101 311[9]は、転送プレーン上の必要条件を提供する。
- TS 101 878[3]は、シンプルコールのために TIPHON リリース 3 で使用されるサービス能力を定義する。
- TS 101 882（本ドキュメント）は、本書で定義されるシンプルコールサービス能力を実装するために、TIPHON リリース 3 アーキテクチャに基づいたプロトコルフレームワークを提供する。
- TS 101 315[5]は、TS 101 878[3]で定義されるサービス能力を実現するため、メタプロトコルの使用方法を示す実装方式である。
- TS 101 883[10]は、ITU-T H.323 プロファイルのためのプロトコルマッピングを提供する。
- TS 101 884 は、SIP プロファイルのためのプロトコルマッピングを提供する。
- TS 101 885[11]は、ITU-T H.248 プロファイルのためのプロトコルマッピングを提供する。
- TS 101 314[1]は、TIPHON リリース 3 のためのアーキテクチャと参照構成を提供する。

1 適用範囲（Scope）

本ドキュメントは、TS 101 878[3]に記述された能力群の実装に必要な TIPHON アーキテクチャ（TS 101 314[1]）に定義された参照ポイント群のための複数プロトコルフレームワークを定義するものである。なお、ここでいう実装とは、相互運用されるプロトコル群（H.323, SIP, H.248）を用いたフレームワークに準拠した実装等をさす。

プロトコルフレームワークは、構文（Syntax）と作用（behaviour）の両面から記述される一連のメタプロトコルの形態をとる。ここでいう構文とは ASN.１（Abstract Syntax Notation 1：参考文献参照）をさし、作用とは MSCs（Message Sequence Charts：参考文献 ITU-T 勧告 Z.120 参照）と文章に付加される簡単な SDL（Specification and Description Language：参考文献 ITU-T 勧告 Z.120 参照）図をさす。メタプロトコルは、インボーク、制御、そしてメタプロトコル経過レポートのために高位または低位のレイヤにより用いられるサービスプリミティブと、同位エンティティとの通信に用いられる M-PDU(Meta Protocol Data Unit)の両方を提示する。

本書における今次ヴァージョンのメタプロトコルは、TS 101 314 [1]の R、C、N そして T において定義される参照点用に記述されるものである。

本ドキュメントは、TIPHON リリース 3 のサポートに必要なプロトコル群へ適用が可能である。

要求条件の状態を表示する記述（例：制限指示または禁止、許可または能力、可能性など）においては、TS 101 878[3]に記述された能力を提供するために使用されるプロトコル群（H.323, SIP, H.248）の個々に閉じた要求条件の特質を修正する場合もある。

2 参照文献（References）

以下のドキュメントは、本テキスト内での参照を通じて、このドキュメントの条項を構成する条項を含む。

- 参照文献には、確定したもの（出版日、編集番号、ヴァージョン番号、等により識別された）、未確定のものがある。
- 確定した参照文献には、その後のヴァージョンは適用されない。
- 未確定の参照文献では、最新ヴァージョンが適用される。

[1] ETSI TS 101 314: "Telecommunications and Internet Protocol Harmonization Over Networks (TIPHON) Release 3; Abstract Architecture and Reference Points Definition; Network Architecture and Reference Points".

[2] ETSI TR 101 877: "Telecommunications and Internet Protocol Harmonization Over Networks (TIPHON); Requirements Definition Study; Scope and Requirements for a Simple call".

[3] ETSI TS 101 878: "Telecommunications and Internet Protocol Harmonization Over Networks (TIPHON) Release 3; Service Capability Definition; Service Capabilities for a simple call".

[4] ISO/IEC 9646-7: "Information technology - Open Systems Interconnection - Conformance testing methodology and framework - Part 7: Implementation Conformance Statements".

[5] ETSI TS 101 315: "Telecommunications and Internet protocol Harmonization Over Networks (TIPHON) Release 3; Functional Entities, Information Flow and Reference Point Definitions; Guidelines for application of TIPHON functional architecture to inter-domain services".

[6] ETSI TR 101 301: "Telecommunications and Internet Protocol Harmonization Over Networks (TIPHON) Release 3; Release Definition; TIPHON Release 3 Definition".

[7] ETSI TR 101 835: "Telecommunications and Internet Protocol Harmonization over Networks (TIPHON); Project method definition".

[8] ETSI TR 102 008: "Telecommunications and Internet Protocol Harmonization Over Networks (TIPHON) Release 3; Terms and Definitions".

[9] ETSI TR 101 311: "Telecommunications and Internet Protocol Harmonization Over Networks (TIPHON) Release 3; Service Independent requirements definition; Transport Plane".

[10] ETSI TS 101 883: "Telecommunications and Internet Protocol Harmonization Over Networks (TIPHON) Release 3; Technology Mapping; Implementation of TIPHON architecture using H.323".

[11] ETSI TS 101 885: "Telecommunications and Internet Protocol Harmonization Over Networks (TIPHON) Release 3; Technology Mapping; Technology Mapping of TIPHON reference point N to H.248/MEGACO protocol".

[12] ETSI TS 101 520: "Telecommunications and Internet Protocol Harmonization Over Networks (TIPHON); Telecommunications and Internet Protocol Harmonization Over Networks (TIPHON); Support of ITU-T Recommendation H.323".

[13] ETSI TS 101 521: "Telecommunications and Internet Protocol Harmonization Over Networks (TIPHON); Protocol Implementation Conformance Statement (PICS) proforma for the support of call signalling protocols and media stream packetization for packet-based multimedia communication systems; Support of ITU-T Recommendation H.225.0".

[14] ETSI TS 101 522: "Telecommunications and Internet Protocol Harmonization Over Networks (TIPHON); Protocol Implementation Conformance Statement (PICS) proforma for the support of control protocol for multimedia communication; Support of ITU-T Recommendation H.245".

[15] ETSI TS 101 804: "Telecommunications and Internet Protocol Harmonization Over Networks (TIPHON) Release 3; Technology compliance specifications".

3 定義、シンボルと略語（Definitions, symbols and abbreviations）

3.1 定義

本ドキュメントにおいては、TR 101 877 および TS 101 878 で与えられる定義が適用される。

3.2 略語

本ドキュメントにおいては、次の略語が適用される。

API	Application Programming Interface
ASN.1	Abstract Syntax Notation no. 1
BC	Bearer Control
CC	Call Control
CCUA	Call Control User Agent
FE	Functional Entity
FG	Functional Grouping
GoS	Grade of Service
IP	Internet Protocol
IPTN	IP Telephony Network
ISDN	Integrated Services Digital Network
MC	Media Control
M-PMU	Meta Protocol Message Unit
MSC	Message Sequence Chart

PCM	Pulse Code Moduration
PDU	Protocol Data Unit
PSTN	Public Switched Telephony Network
QoS	Quality of Service
SAP	Service Access Point
SC	Service Control
SCN	Switched Circuit Network
SDL	Specification and Description Language
SL	Service Layer
SNCC	Serving Network Call Control
TCC-SAP	TIPHON Call Control SAP
TLL-SAP	TIPHON Lower Layer SAP
TNCC	Transit Network Call Control
TRL	TIPHON Resource Location
TR-SAP	TIPHON Registration SAP
TT-SAP	TIPHON Transport SAP
URI	Uniform Resource Identifier

4　導入（Introduction）

本ドキュメントの Annex（補遺）は、TS 101 314[1]で定義されている参照ポイントに適用されるメタプロトコル表記に関する有用な情報を含むものである。 それぞれの Annex は完結したものであり、複数のメタプロトコル間の相互作用については、更なる記述が TS 101 315[5]においてなされている。

Annex A (normative):参照点 R におけるメタプロトコル

TIPHON ネットワークでは、アタッチメントのレジストレーションポイント（Registration point of Attachment (RpoA)）が提供されなければならない。そのため、RpoA が提供されないネットワークは、TIPHON 機能が有効でないとみなされる。

例： TIPHON ネットワークが IP ベースに基づくものであり、DHCP により提供される端末（ホスト）設定を有する場合、この場合の DHCP 手順は、ネットワークトランスポートドメインのユーザが先行的な承認なしに起動する"anonymous"サービス（例：緊急サービス、情報サービス）はもちろんのこと、端末の IP アドレス、DNS パラメータ、そして RpoA を端末に付与する。

A.1 概要

レジストレーションメタプロトコル処理は、入退出が厳しく管理されているドメイン群において、ユーザ（レジストレーション者）が、サービスを起動するための権限を探索して入手することを可能とする。提供されるべきサービスアプリケーションは、ユーザプロファイルの中で保持される情報により、部分的に決定されなければならない。

レジストレーション（registration）における中核的な要素の関連を図２に示す。

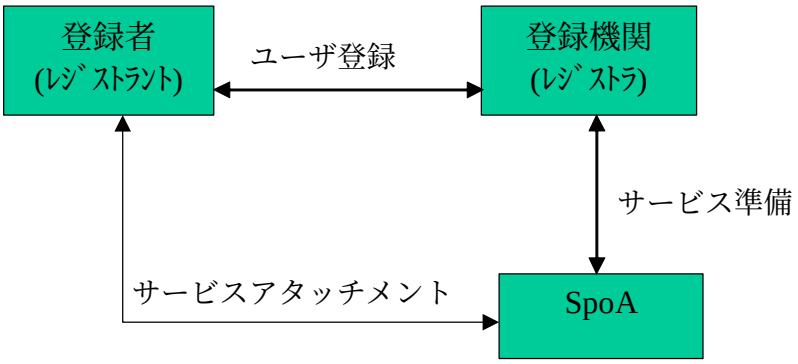


図2　レジストレーションにおける構成要素の関係

レジストレーションサービスは、ユーザに対し、ユーザのホームと訪問先（visited）ドメインの両方においてサービスを受けることを可能とする。レジストラントがレジストレーションするレジストラは、サービスアプリケーション群のプロファイルをそのレジストラントのために維持しなけれらばならない。レジストラは、レジストラントが位置している、または、サービスアプリケーション群を制御している構成要素が位置しているドメインのサービスアプリケーション群に対するアクセスを許可することができる。

レジストレーションは、端末申し込みにおいて提供される間接的なサービスであってもよい。そのような場合、この章で記述されるメタプロトコルは適用されるべきでなない。

注:　サービスアタッチメントは、PSTN もしくは PABX に対する電話セットのアタッチメントと比較することが可能である。

レジストレーションメタプロトコルは以下の 2 段階手順を有する。：

- レジストラントは、レジストラにレジストレーションしレジストレーションに成功した場合、サービスアプリケーション群のプロファイルへのアクセス権を得る；
- レジストレーションに成功した場合、レジストラは、アプリケーションサーバからのサービス要求時に、レジストラントによって使用されるチケットの形態を有した、それぞれのアプリケーションサービス用の信任状を提供する。

レジストラントが利用可能な各サービスアプリケーションは、異なるサービスプロバイダにより提供されることが可能である。これらのサービスプロバイダたちは、異なるドメインに存在し、さらに、自身のために異なる SpoA を提供することが可能である。信任状であるチケットは、レジストラントが適切な SpoA 群を介して目的のサービスアプリケーションの利用を許可されていることを提示しなければならない。同チケッ

トはさらに、一定の期間そのレジストレーションが有効であることを証明する必要がある。
ひとつのセッションにおける最初の試行時、レジストレーションモード（registration mode ）は、RegistrationMode element (A.5 章参照)を"InitialRegistration"にセットすることにより識別される。周期的な再レジストレーションは、随時発生させることが可能であり、さらに、RegistrationMode element (A.5 章参照)を"LocationUpdate"にセットすることによりこの章で記述されるプロトコルに従ってなされなければならない。
端末は、サービスの生成または受信を可能とする前に、レジストレーションされ承認される必要がある。しかしながら、このような“ サービス起動に先立ったレジストレーション（"Register before service invocation" ）”は、あるコールタイプ（例：緊急呼）のために無効とされることが可能である。
各レジストラントは、ホームレジストラのドメインが識別される（ことを許容す）べき、ユニークなレジストレーション ID を有する。

A.1.1 Reachable (利用可) 表示

レジストレーションサービスの成功裏完了により、レジストラント（ユーザ）は、ホームレジストラに対して自分がサービスに対して利用可（Reachable）であり、サービスにアタッチ（attach）していることを提示する。
レジストラントが、ユーザプロファイル中のいかなるレジストレーションされたサービスアプリケーションに対する利用をもはや望まない場合、レジストラは SpoA を介した通信により、対象のサービスアプリケーションに対して明示的にデッタチ（Detach）しなければならない。
レジストラが、サービスアプリケーションまたはサービスアプリケーションセットに対し、ユーザがもはや利用可とすべきでないと判断した場合（例：プリペイドカードアカウントの無効化）、レジストラは、ユーザに対して通知するため、明示的にレジストレーション解除プロトコルを開始しなければならない。

A.1.2 レジストレーションサービス定義

レジストレーションサービスは、ひとつのネットワークもしくはネットワークセット（ネットワークの一部が異なる管理制御下である場合）を横断して提供されるべきである。レジストレーションサービスは、TIPHON Registration Service Access Point (TR-SAP)において示されるプリミティブ（表１）を用いることにより起動される。

表１　TR-SAPに現れるレジストレーションプリミティブ

プリミティブ	備考	能力 (NOTE参照)
TR_RegistrationRequest_req	ユーザプレーンがレジストレーションを起動することを許可する	
TR_RegistrationRequest_conf	ユーザプレーンに対しユーザ起動のレジストレーション結果を与える	
TR_DeRegistrationRequest_req	ユーザプレーンがレジストレーション解除を起動することを許可する	
TR_DeRegistrationRequest_conf	ユーザプレーンに対しユーザ起動のレジストレーション解除結果を与える	
TR_RegistrationStatus_ind	ユーザプレーンに対しレジストラからの通知情報を報告する	
NOTE:　TS 101 878 [3]における能力定義参照		

表2　レジストレーションプリミティブにおけるパラメータ

プリミティブ	パラメータ		
	Request	Confirm	Indication
TR_RegistrationRequest	UserID, [TerminalID] [List of ServiceApplications]	RegistrationResult	-
TR_DeRegistrationRequest	UserID, [List of ServiceApplications]	DeRegistrationResult	-
TR_RegistrationStatus	-	-	TBD

レジストレーションプリミティブにおけるパラメータ値を以下に示す。

RegistrationResult =

Success {list of ServiceApplications};

FailureReason {list of ServiceApplications};

No response from Registrar;

Protocol timer expired;

…

DeRegistrationResult =

Success;

No Active Registrations;

…

ServiceApplication =

Service#1;

Service#2;

…

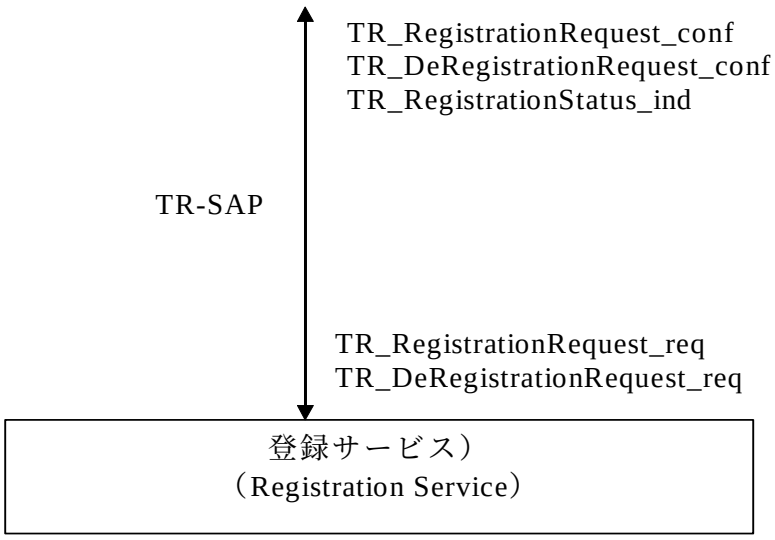


図3　プリミティブにより識別されるレジストレーションサービスにより提供されるサービス群

A.1.3 チケット内容と処理

サービスの利用承認チケットは以下の処理を行う。

- ユーザの確認 (レジストラントとしての確認)、

- レジストラの確認(チケット発行者としての確認)、
- サービスアプリケーション（群）の確認、
- それぞれのサービスアプリケーション（群）のサービスプロバイダの確認、
- ユーザに提供されるべきサービスアプリケーション（群）期間の提示、
- オプション処理として、サービスプロバイダ（群）へ暗号化認証手段を提供する。

チケットにおけるユーザIDは、サービスアプリケーションにより要求されるIDとする。一人のユーザは、自分が選択したサービスアプリケーション（群）に依存した、多くの異なるユーザIDを保有することが可能である。一人のユーザがひとつ以上の複数のユーザIDを保有する場合、個々のチケットがそれぞれのユーザIDのために提供されるべきであり、その場合のチケットは、それらユーザIDのためのサービスアプリケーション群のリストを含む必要がある。

チケットの ASN.1 定義を以下に示す。

```
TicketType ::= SEQUENCE
{
    registrantId        VisibleString,
    registrarId         VisibleString,
    serviceCredential   SET OF ServiceCredentialType,
    cryptoDigest        DigestType OPTIONAL
}

ServiceCredentialType ::= SEQUENCE
{
    serviceAppId        ServiceApplicationType,
    spoA                SpoAType, -- shall be in the user's terminal addressing realm
    startTime           GeneralizedTime,
    stopTime            GeneralizedTime, -- Shall be greater than StartTime
    cryptoDigest        DigestType OPTIONAL
}
```

A.1.3.1 **認証**

レジストレーションサービスは、ひとつの強力な埋め込み（間接的な）認証サービスを含むことが可能である。認証が提供される場合、レジストレーション要求は、レイストラントの信任状（群）をひとつの認証トークンの形態で含む必要がある。

認証されるべき構成要素は以下の通り。

- 動的に規定されたパラメータ群を有する端末群、
- 動的に規定されたパラメータ群を有するネットワーク内の機能要素群、
- トランスポートアタッチメントにおける動的ポイントを有する端末群、
- サービスアタッチメントにおける動的ポイントを有する端末群、
- トランスポートアタッチメントにおける動的ポイントを有するネットワーク内の機能要素群、
- サービスアタッチメントにおける動的ポイントを有するネットワーク内の機能要素群。

強力な認証が使用されない場合は、例え認証が成功した場合でも、その認証結果は認証されなかったものとして扱われるべきである。提供されるべきサービスアプリケーション（群）は、ユーザプロファイルの情報により、部分的に、決定されるものとする。認証の成功に基づき、ユーザに与えられる結果は、これらサービスへの到達（利用）の仕方を秘匿性を維持した状態で提示される。 (例：アプリケーションサーバが、認証された要求元からの要求を認知する場合など)

A.2 **サービスアプリケーションのためのレジストレーション**

レジストレーション成功は、レジストラントに対し、サービスアプリケーションへのアクセスを許可する。サービスアプリケーション利用の資格を与えるもの（ユーザサービスプロファイル）のローカルコピーは、レジストラの端末において管理されることが可能である。アクセスが許可されている個々のサービスアプリケーションをサポートするための端末の能力群は、端末のため、ローカルに保守される。単機能端末におけるサービスプロファイルは、（サーバなどにより一括して保守されることで）間接的に扱われることが可能である。

レジストレーションを要求する場合、そのレジストラントは、レジストラ自身がサービスを受けることを希望し自身の端末がそのサービスのサポートが可能である、アクセス資格が与えられたサービスの中から、サービスアプリケーション群を提示しなければならない。

A.3 レジストラの発見

レジストラの ID とアドレスは、サービス利用準備時において、ユーザ（またはユーザ端末）に対して提供されることが可能である。またレジストラの ID とアドレスは、DHCP などの設定プロトコルの使用や、レジストラとサービスプロファイル内においてレジストレーション ID 要素として保守された情報により、提供されることも可能である。

レジストレーション ID は次のような形態で提示される。

```
RegIdType ::= SEQUENCE
{
    registrarId         VisibleString,
    registrarLoc        TRL,
    registrantId        VisibleString
}
```

下記の通り、TRL 要素タイプは TIPHON リソースロケーションを提示する。

```
TRL ::= SEQUENCE
{
    protocolID      VisibleString,
    nameorAddress       VisibleString,
    port                INTEGER OPTIONAL
}
```

A.4 機能を構成する要素

サービスアプリケーション群は、常時ホーム環境において実行されるものとする。ここでいう、ホーム環境とは、実際的な場合または仮想的な場合が考えられる。

表3　レジストレーション機能の要素

ID	名称
FE1	レジストラント
FE2 (note 2 参照)	RpoA により示されるレジストラ (note 1 参照)
FE3 (note 2 参照)	Service point of Attachment (SpoA)により示されるアプリケーションサーバ
NOTE 1:　現在レジストレーション中の端末群のレジストラを含む (レジストレーションID (regID) へのキー) NOTE 2: FE2 とFE3は分散配置される機能要素として実装されることが可能である。	

A.4.1 レジストラント

レジストラントはレジストレーション対象となる論理的な要素であり、端末（例：電話送受信機）または、より直接的にユーザ（例：信任状を提供している人）に関係した形態をとることが可能である。レジストレーションを要求する場合レジストラントは、レジストレーションされるべきサービスアプリケーション群のリストをレジストラに対して提示する。

A.4.2 レジストラ

レジストラは、レジストラントのユーザプロファイルを保守する。

個々のレジストレーションにおけるローカルアプリケーションサービス群への結合（binding）を最適化するため、レジストラは、RpoA に対する SpoA の履歴に関する保守も行う。レジストラは、ユーザプロファイルへのレジストレーション要求を検証し、レジストラントの RopA にローカルなサービスプロバイダを識別する。SopA が RopA に対してローカルに識別される場合、同 SopA は、ホームサービス環境として振舞う。RopA に対してローカルに識別される SopA がない場合、その SopA は、より遠方であるものの適切なサーバのひとつが提供される。

適切なサービスレジストラが識別されると、そのレジストラは、レジストラントの認証要求へコンタクトする。認証結果は、チケットの形態を用いてレジストラントへ送付され、SopA を識別する。

A.4.3 SpoA

レジストレーションにおいて、SopA は、ユーザのサービスへのアタッチメントを保守する。

A.5 情報フロー

表 4 及び以降の節に記述される情報フローはレジストレーションサービス M-PDUs を含む。

これらの M-PDUs の ASN.1 宣言は A.8 節にて与えられる。

表4　レジストレーションM-PDUs

M-PDU名称 (注1, 2を参照)	方向	要素	M/O/C
U_RegistrationRequest	FE1 to FE2	RegId RegistrationMode RpoA ServiceApplicationId AuthenticationToken	M M M M O
D_RegistrationResponse	FE2 to FE1	RegId ServiceCredentials	M M
D_RegistrationReject	FE2 to FE1	RegId ServiceRejectReason	M M
D_RegistrationPending	FE2 to FE1	RegId	M
U_DeRegistrationRequest	FE1 to FE2	RegId	M
D_DeRegistrationResponse	FE2 to FE1	RegId RegistrationRemovedFlag	M M
U_SpoAServiceAttachRequest	FE1 to FE3	RegId ServiceRequestTicket	M M
D_SpoAServiceAttachResponse	FE3 to FE1	RegId ServiceOfferTicket	M M
D_SpoAServiceAttachReject	FE3 to FE1	RegId ServiceRejectReason	M M
U_SpoAServiceDetachRequest	FE1 to FE3	RegId ServiceOfferTicket	M M
D_SpoAServiceDetachResponse	FE3 to FE1	RegId ServiceDetachedFlag	M M
D_SpoAClientAttachNotify	FE2 to FE3	RegistrarID RegistantID ServiceAppId RegistrantAuthTicket	M M M M
U_SpoAClientNotifyResponse	FE3 to FE2	RegistrarID ClientAcceptedFlag SpoAAuthTicket	M M C
D_SpoAClientDetachNotify	FE2 to FE3	RegistrarID RegistantID ServiceAppId RegistrantAuthTicket	M M M M
U_SpoAClientDetachConfirm	FE3 to FE2	RegistrarID ClientDetachedFlag	M M
注 1: "U_" **で始まる**M-PDUs**は端末で生成され、ネットワーク方向にのみ(即ちネットワークへの上り方向に)向かう** M-PDUs**であることを示す。** **注**2: "D_"**で始まる**M-PDUs**はネットワークで生成され、端末方向にのみ(即ちネットワークからの下り方向に)向かう**M-PDUs**であることを示す。**			

A.5.1 U_RegistrationRequest

The U_RegistrationRequest メッセージは端末サービス(FE1)によるレジストラ(FE2)への要求に使用され、ユーザのプロファイルのオープン、各サービスがレジストレーションされている SpoA (FE3)の特定、及び有効性の確認に使用される。

Direction: FE1 to FE2;

Response to: none;

Response expected: D_RegistrationPending or D_RegistrationResponse or D_RegistrationReject.

U_RegistrationRequest のパラメータは M-PDURegistrationRequestType のタイプであり、以下に宣言される ASN.1 フォーマットに従う。

```
M-PDURegistrationRequestType ::= SEQUENCE
{
    regID               RegIdType,
    registrationMode    RegistrationModeType DEFAULT initialRegistration,
    rpoA                RpoAType,
    serviceAppId        SET OF ServiceApplicationType,
    authToken           AuthenticationTokenType OPTIONAL
}

AuthenticationTokenType ::= TokenType

RegistrationModeType ::= ENUMERATED
{
    initialRegistration (0),
    locationUpdate (1)
}
```

A.5.2 D_RegistrationConfirm

The D_RegistrationConfirm メッセージは FE2 により使用され、FE1 からのレジストレーション要求を受け入れたことを明確に示すものである。

Direction: FE2 to FE1;

Response to: U_RegistrationRequest;

Response expected: none.

D_RegistrationConfirm のパラメータは M-PDURegistrationResponseType のタイプであり、以下に宣言される ASN.1 フォーマットに従う。

```
M-PDURegistrationResponseType ::= SEQUENCE
{
    regID               RegIdType,
    serviceCredential   SET OF TicketType
}
```

A.5.3 D_RegistrationReject

D_RegistationReject メッセージは FE2 により使用され、FE1 からのレジストレーション要求を拒絶したことを明確に示すものである。

Direction: FE2 to FE1;

Response to: U_RegistrationRequest;

Response expected: none.

D_RegistrationReject のパラメータは M-PDURegistrationRejectType のタイプであり、以下に宣言される ASN.1 フォーマットに従う。

```
M-PDURegistrationRejectType ::= SEQUENCE
{
    regID               RegIdType,
    serviceRejectReason SET OF ServiceRejectReasonType
}

ServiceRejectReasonType ::= SEQUENCE
{
```

```
    serviceAppId        ServiceApplicationType, -- Enumerated list defined by operator
    rejectReason        TIPHONErrorType     -- Enumerated list defined by the present document
}

TIPHONErrorType ::= SEQUENCE
{
    reason              ENUMERATED {badThingWrong (0)},
    diagnostic          ENUMERATED {somethingWrong(0)} OPTIONAL,
    freeText            VisibleString OPTIONAL -- This should inform the user behaviour
}
```

A.5.4 D_RegistrationPending

D_RegistrationPending メッセージは FE2 により使用され、レジストレーションプロセスが継続することを示すものである。

Direction: FE2 to FE1;

Response to: U_RegistrationRequest;

Response expected: none.

D_RegistrationPending パラメータは M-PDURegistrationPendingType タイプであり、以下に宣言される ASN.1 フォーマットに従う。

```
M-PDURegistrationPendingType ::= SEQUENCE
{
    regID           RegIdType
}
```

A.5.5 U_DeRegistrationRequest

U_DeRegistrationRequest メッセージは FE1 により使用され、全てのサービスアタッチメントのクリアを要求するものである。

Direction: FE1 to FE2;

Response to: none;

Response expected: D_DeRegistrationResponse.

U_DeRegistrationRequest のパラメータは M-PDUDeRegistrationRequestType のタイプであり、以下に宣言される ASN.1 フォーマットに従う。

```
M-PDUDeRegistrationRequestType ::= SEQUENCE
{
    regID           RegIdType
}
```

A.5.6 D_DeRegistrationResponse

D_DeRegistationResponse メッセージは FE2 により使用され、レジストレーションの解除の承諾及び FE2 により維持される全てのサービスアタッチメントのクリアを明確に確認するものである。

Direction: FE2 to FE1;

Response to: U_DeRegistrationRequest;

Response expected: none.

D_DeRegistrationResponse のパラメータは M-PDUDeRegistrationResponseType のタイプであり、以下に宣言される ASN.1 フォーマットに従う。

```
M-PDUDeRegistrationResponseType ::= SEQUENCE
{
    regID                   RegIdType,
    registrationRemovedFlag BOOLEAN
}
```

A.5.7 U_SpoAServiceAttachRequest

U_SpoAServiceAttachRequest メッセージは FE1 により使用され、FE2 からの D_RegistrationResponse を受信した特定のサービス提供クレデンシャルに対する FE3 (SpoA)へのアタッチメントを要求するものである。

Direction: FE1 to FE3;

Response to: D_RegistrationResponse (from FE2);

Response expected: D_SpoAServiceAttachResponse.

U_SpoAServiceAttachRequest のパラメータは M-PDUSpoAServiceAttachRequestType のタイプであり、以下に宣言される ASN.1 フォーマットに従う。

```
M-PDUSpoAServiceAttachRequestType ::= SEQUENCE
{
    regID                   RegIdType,
    serviceRequestTicket    TicketType
}
```

A.5.8 D_SpoAServiceAttachResponse

D_SpoAServiceAttachResponse メッセージは FE3 により使用され、サービス要求の承諾を示すものである。

Direction: FE3 to FE1;

Response to: U_SpoAServiceAttachRequest;

Response expected: none.

D_SpoAServiceAttachResponse は M-PDUSpoAServiceAttachResponseType のタイプであり、以下に宣言される ASN.1 フォーマットに従う。要求された各々のサービスに対する SpoA が TicketType で返送される点に注意が必要である。

```
M-PDUSpoAServiceAttachResponseType ::= SEQUENCE
{
    regID               RegIdType,
    serviceOfferTicket  TicketType
}
```

A.5.9 D_SpoAServiceAttachReject

D_SpoAServiceAttachReject メッセージは FE3 により使用され、サービスアタッチ要求を明確に拒絶するものである。

Direction: FE3 to FE1;

Response to: U_SpoAServiceAttachRequest;

Response expected: none.

D_SpoAServiceAttachReject は M-PDUSpoAServiceAttachRejectType のタイプであり、以下に宣言される ASN.1 フォーマットに従う。

```
M-PDUSpoAServiceAttachRejectType ::= SEQUENCE
{
    regID               RegIdType,
    serviceRejectReason ServiceRejectReasonType -- Enumerated list defined by the present
document
}
```

A.5.10 U_SpoAServiceDetachRequest

U_SpoAServiceDetachRequest メッセージは FE1 により使用され、SpoA からのディタッチを明確に示すものである。

Direction: FE1 to FE3;

Response to: none;

Response expected: D_SpoAServiceDetachResponse.

U_SpoAServiceDetachRequest は M-PDUSpoAServiceDetachRequestType のタイプであり、以下に宣言される ASN.1 フォーマットに従う。

```
M-PDUSpoAServiceDetachRequestType ::= SEQUENCE
{
    regID               RegIdType,
    serviceOfferTicket  TicketType
```

```
}
```

A.5.11 D_SpoAServiceDetachResponse

D_SpoAServiceDetachResponse メッセージは FE3 により使用され、サービスディタッチリクエストの承諾を示すものである。

Direction: FE3 to FE1;

Response to: U_SpoAServiceAttachRequest;

Response expected: none.

D_SpoAServiceDetachResponse は M-PDUSpoAServiceDetachResponseType のタイプであり、以下に宣言される ASN.1 フォーマットに従う。

```
M-PDUSpoAServiceDetachResponseType ::= SEQUENCE
{
    regID               RegIdType,
    serviceDetachedFlag BOOLEAN
}
```

A.5.12 D_SpoAClientAttachNotify

D_SpoAClientAttachNotify メッセージは FE2 により使用され、クライアントがサービスへのアタッチを意図しており、また、このアタッチメントが FE3 により許可されたものであることを FE3 に伝えるものである。

Direction: FE2 to FE3;

Response to: U_RegistrationRequest (from FE1);

Response expected: U_SpoAClientNotifyResponse.

D_SpoAClientAttachNotify のパラメータは M-PDUSpoAClientAttachNotifyType のタイプであり、以下に宣言される ASN.1 フォーマットに従う。

```
M-PDUSpoAClientAttachNotifyType ::= SEQUENCE
{
    registrarID             RegIdType,
    registantID             RegIdType,
    serviceAppId            ServiceApplicationType,
    registrantAuthTicket    TicketType
}
```

A.5.13 U_SpoAClientNotifyResponse

U_SpoAClientNotifyResponse メッセージは FE3 により使用され、FE2 から要求されたクライアント(FE1)へのサービス提供への承諾または拒絶を明確に示すものである。

Direction: FE3 to FE2;

Response to: D_SpoAClientAttachNotify;

Response expected: none.

U_SpoAClientNotifyResponse は M-PDUSpoAClientNotifyResponseType のタイプであり、以下に宣言される ASN.1 フォーマットに従う。

```
M-PDUSpoAClientNotifyResponseType ::= SEQUENCE
{
    registrarID         RegIdType,
    clientAcceptedFlag  BOOLEAN,
    spoAAuthTicket      TicketType OPTIONAL -- Conditional on value of flag
}
```

A.5.14 D_SpoAClientDetachNotify

D_SpoAClientDetachNotify メッセージは FE2 により使用され、クライアントがレジストラを通じて全てのサービスからのレジストレーションの解除を要求しており、また、クライアントの SpoA のサービスの使用に関する承認の解除を SpoA が承認したことを FE3 に通知するものである。

Direction: FE2 to FE3;

Response to: U_DeRegistrationRequest (from FE1);

Response expected: U_SpoAClientDetachResponse.

D_SpoAClientDetachNotify は M-PDUSpoAClientDetachNotifyType のタイプであり、以下に宣言される ASN.1 フォーマットに従う。

```
M-PDUSpoAClientDetachNotifyType ::= SEQUENCE
{
    registrarID             RegIdType,
    registantID             RegIdType,
    serviceAppId            ServiceApplicationType,
    registrantAuthTicket    TicketType
}
```

A.5.15 U_SpoAClientDetachResponse

U_SpoAClientDetachResponse メッセージは FE3 により使用され、クライアント(FE1)によるサービスからの離脱を示すものである。

Direction: FE3 to FE2;

Response to: D_SpoAClientDetachNotify;

Response expected: none.

U_SpoAClientDetachResponse は M-PDUSpoAClientDetachResponseType のタイプで示され、以下に宣言される ASN.1 フォーマットに従う。

```
M-PDUSpoAClientDetachResponseType ::= SEQUENCE
{
    registrarID             RegIdType,
    clientDetachedFlag      BOOLEAN
}
```

A.6 処理に関する記述

A.6.1 レジストレーション

A.6.1.1 レジストラント

レジストレーションはレジストラントが U_RegistrationRequest M-PDU を送信することにより、その要求と適合するサービスアプリケーションを示すレジストラが呼び出される。同時に、レジストラントはタイマ TR001 を起動する。呼び出し結果は以下のどれかの状況とならなければならない。

- レジスタしているユーザからの要求;
- 予め定義されている、レジストレーションの定期的なリフレッシュ保証期間の満了;
- TpoA の変化.

レジストレーションのモードは U_RegistrationRequest M-PDU の RegistrationMode 要素の適切な設定により識別されなけらばならない。

もしタイマ TR001 が満了した場合、FE1 は TR_RegistrationRequest_conf プリミティブに失敗理由として"No response from Registrar"を設定してユーザアプリケーションに対しレジストレーションの失敗を通知しなければならない。

FE1 は効率的な方法でネットワークを操作できるよう、適切な段階を踏んでユーザアプリケーションからの要求を結合しなければならない。他のレジストレーションが既に完了している後に、一度にユーザアプリケーションが要求を出した場合、新しい呼がシングルチケットに保持可能なデータを利用して要求できるよう、FE1 はユーザアプリケーションにより要求される全てのサービスを再レジストレーションしなければならない。

FE2 から D_RegistrationPending M-PDU を受信した場合、FE1 はタイマ TR001 をリセットし再スタートしなければならない。

FE2 から D_RegistrationResponse M-PDU を受信した場合、FE1 はクレデンシャルを受信した各サービスのアタッチメントを準備しなければならない。各サービスに対し、FE1 はそのサービスクレデンシャルのコンテンツに基づき、U_SpoAServiceAttachRequest M-PDU を準備しなければならない。FE1 はタイマ TR001 をキャンセルしなければならない。

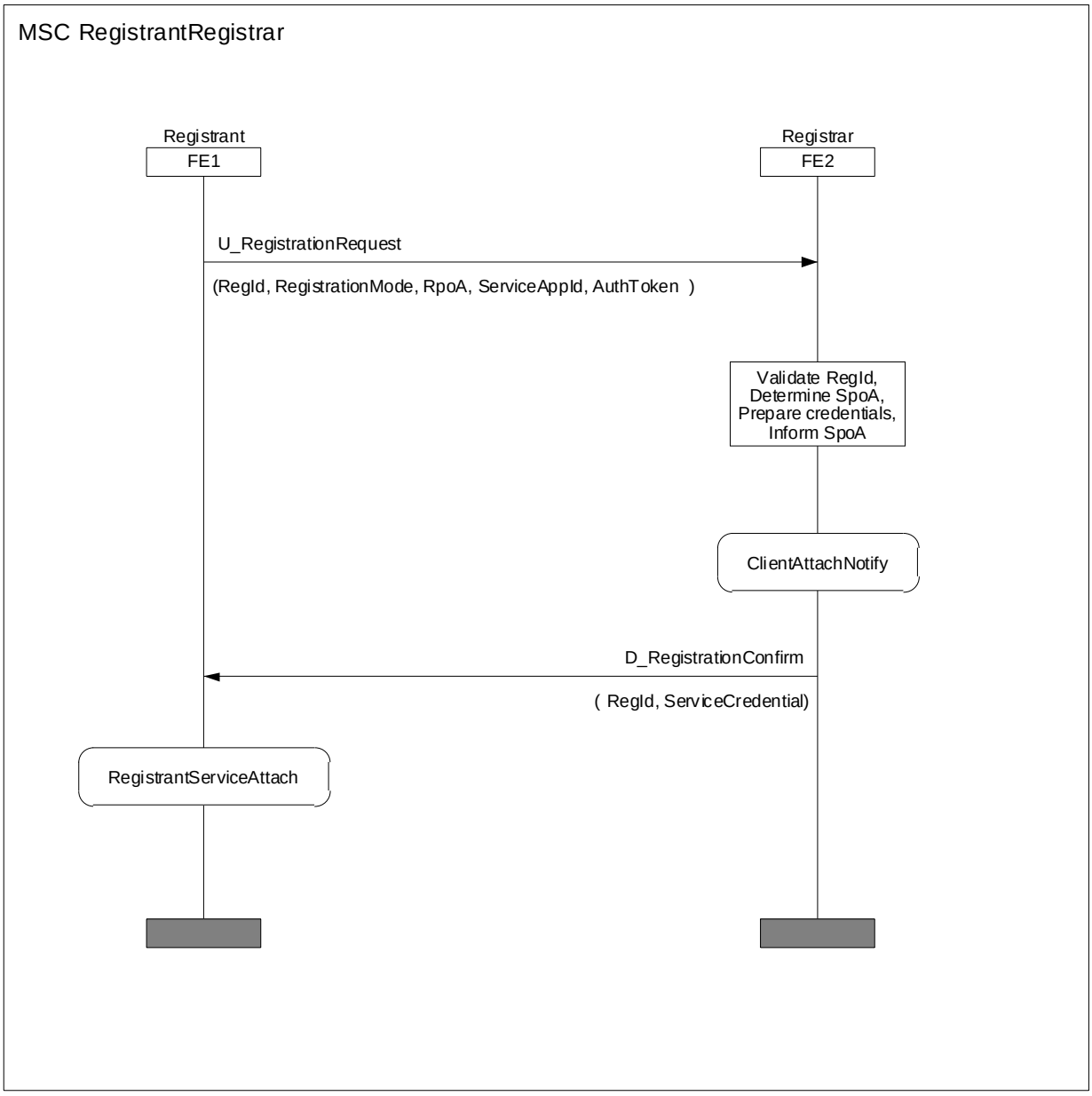


図4　レジストレーションメタプロトコルMSC (レジストラントからレジストラ)

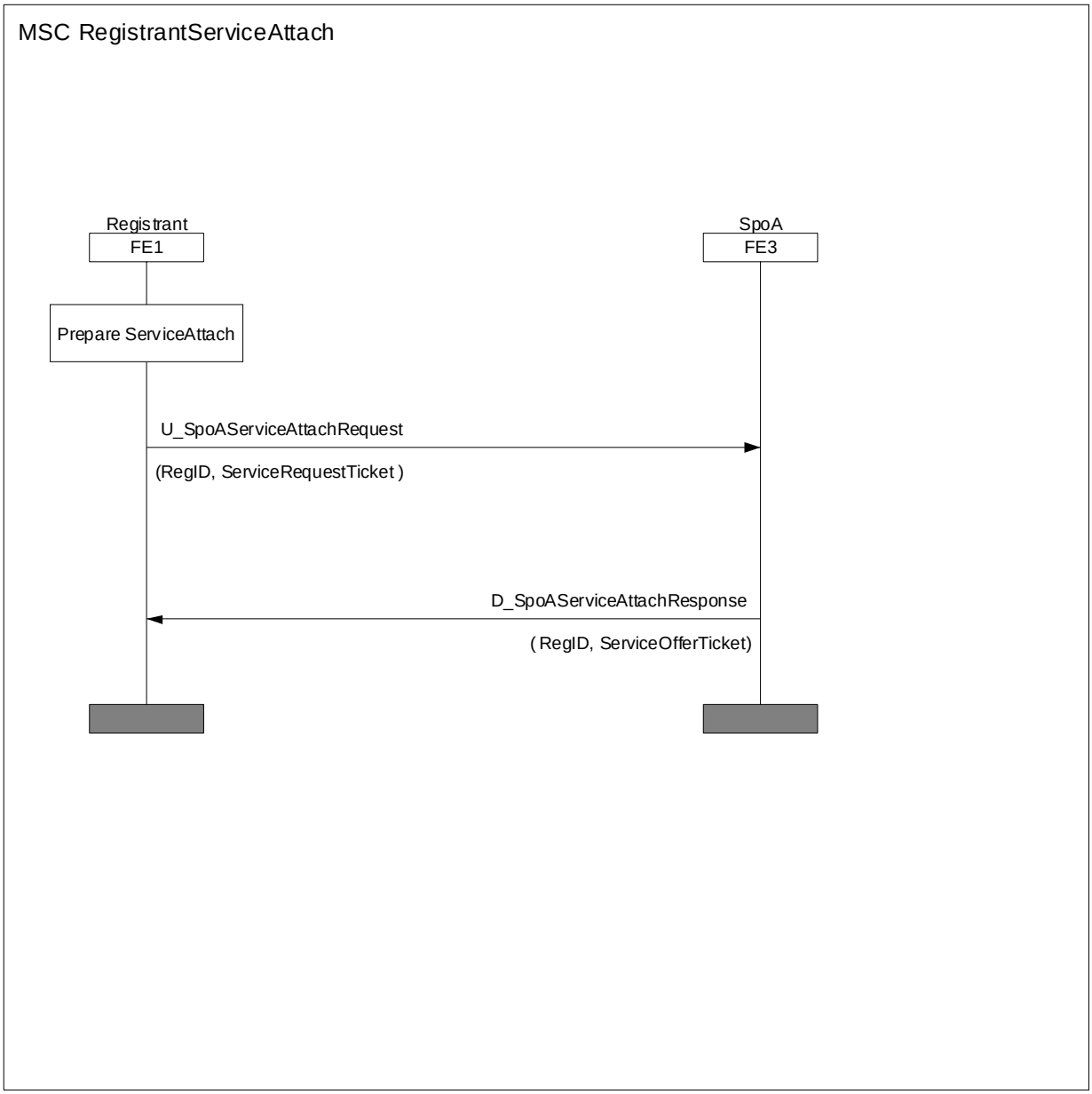


図5　レジストレーションメタプロトコルMSC (レジストラントからSpoA)

A.6.1.2 **レジストラ**

FE1 から U_RegistrationRequest を受信した場合、レジストラ(FE2)は以下のタスクを実行しなければならない。

FE1(レジストラント)の確認

- 提供されたRegIdがローカルにメンテナンスされているユーザプロファイルのリストにあるかを確認しなければならない。もし提供されたRegIdがユーザプロファイルが存在しない場合、レジストラは単一のServiceRejectReason 要素に"Identity not known"を、そしてServiceApplicationIdにNULLを設定して、FE1に対してD_RegistrationRejectを送信しなければならない。もしプロファイルが見つかった場合は、正常な状態に回復して次のタスクが実行されなければならない。

サービスを要求された各 FE2(レジストラ)は以下のタスクを実行しなければならない。

要求されたサービスの確認

- ユーザプロファイルに要求されたサービスが存在するかどうか決定する必要がある。もし要求されたサービスがユーザプロファイルに存在せず、かつ、明示的な承認が必要とされている場合、FE2はD_RegistrationRejectのreject reason 要素に"Invalid or Unknown service"を設定してレジストレーション要求を拒絶しなければならない。もしサービスが見つかった場合、次のタスクを実行しなければならない。

SpoAの特定

- FE2は提供されるTpoAに適切な場所に存在するSpoAを発見しようとしなければならない。この動作はFE2が各々のサービスのための各々のSpoAのTpoAのレコードをメンテナンスするために必要である。

SpoAへアタッチする意図の通知

- FE2はFE1がサービスを提供できるよう、レジストラント、レジストラ、サービス、及び要求の検証を行えるようにするためのチケットを含むD_SpoAClientAttachNotify M-PDUをFE3(SpoA)に送信することで要求しなければならない。

FE3の応答の待機

U_SpoAClientNotifyResponse を受信した場合、FE2 は以下のタスクを実行しなければならない。

失敗した場合

- FE3から失敗の応答を受信して、他の手段が利用可能な場合、FE2はレジストラント、レジストラ、サービス、及び要求の検証を行えるようにするためのチケットを含むD_SpoAClientAttachNotify M-PDUを新たなFE3に送信しなければならない。

成功した場合

- 成功し、全てのSpoAsの最終的な失敗を受信した場合、レジストラはレジストラントに送信するためのサービスクレデンシャル(チケット)を構成し、D_RegistrationResponse M-PDUを送信しなければならない。

注: FE3 のリトライ回数はその FE1s のために FE2 によって確立されたタイマ TR001 の値によって通知されなければならない。

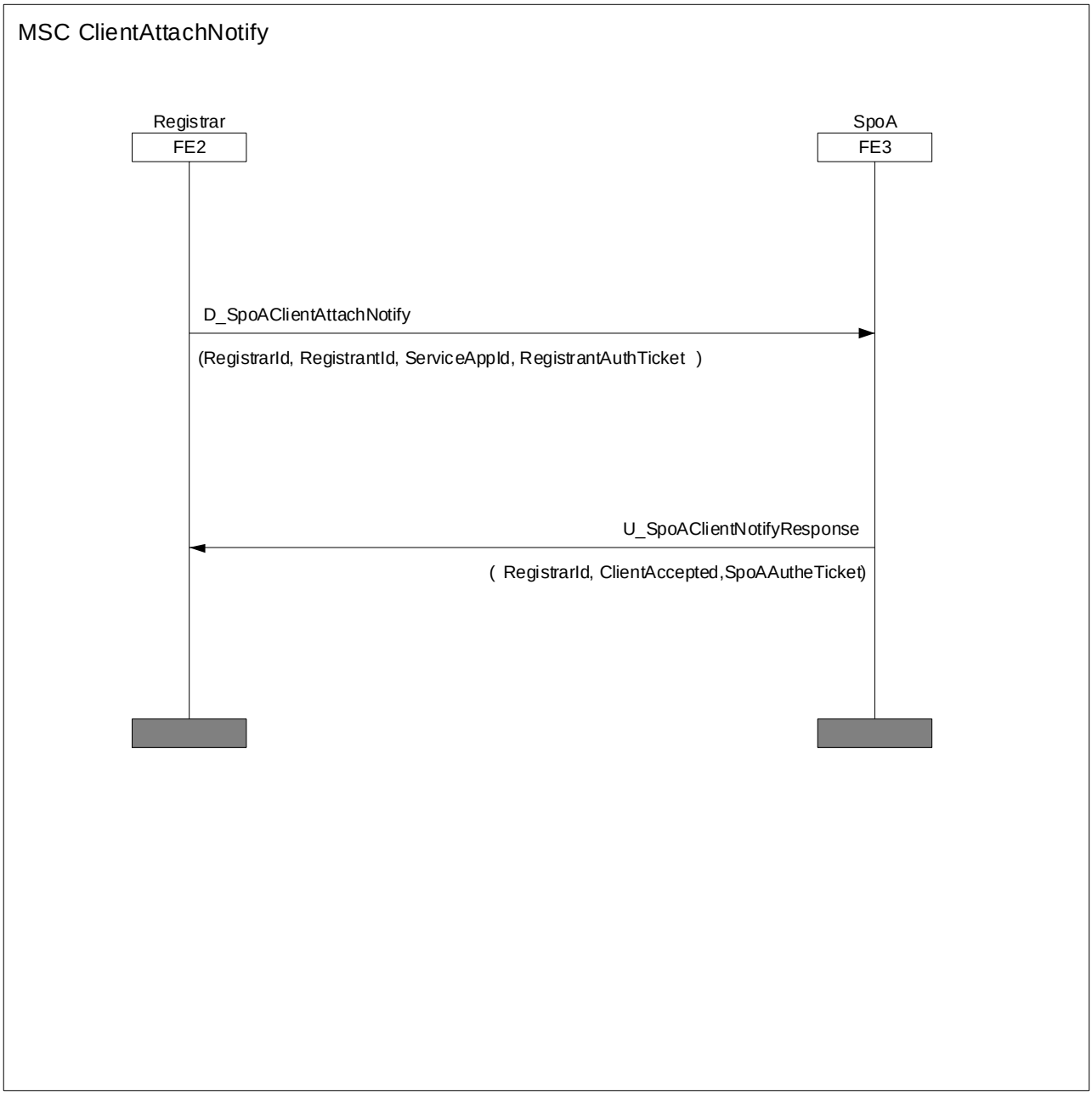


図6　レジストラからSpoAへのプロトコルレジストレーションのMSC

A.6.1.3　SpoA

FE2(レジストラ)からの D_SpoAClientAttachNotify を受信した FE3(SpoA)はそれが呼び出されたサービスに対するクライアントからのいかなる要求も満たす十分なリソースがあるか確認しなければならない。もし十分なリソースがあれば FE3 は FE2 に対し、サービスを有効とする承認チケットを付けて U_SpoAClientNotifyConfirm を返送する。

レジストラントから U_SpoAServiceAttachRequest M-PDU を受信した SpoA はチケットを検証し、もし正しければ D_SpoAServiceAttachConfirm M-PDU により確認を行い、サービスを提供しなければならない。もし健勝に失敗した場合、SpoA は D_SpoAServiceAttachReject M-PDU を用いてクライアントに通知しなければならない。

A.6.2 **レジストレーションの解除**

レジストレーションの解除は以下 2 つの形式を取りうる:

1) レジストラを通じた、集中的な全サービスのレジストレーションの解除
2) SpoAを通じた、局所的な単一のサービスのレジストレーションの解除

これらの形式は本節及びそれ以降の節に示す。

A.6.2.1 **レジストラント**

レジストレーションの解除を行うには、ユーザアプリケーションは TR_SAP を経由して端末(FE1)のレジストレーションエンティティに対して TR_DeRegistrationRequest_req プリミティブを転送しなければならない。TR_DeRegistrationRequest_req はレジストレーションの解除が行われるサービスを示さなければならない。

もし FE1 がアクティブなサービスアタッチメントを持っていない場合、ユーザアプリケーションに対して"No-Active-Service-Attachments"のタイプのエラーを示す TR_DeRegistrationRequest_conf プリミティブにより応答しなければならない。

もし全てのサービスのレジストレーションの解除を行う場合、FE1 はレジストラエンティティ(FE2)に対し U_DeRegistrationRequest M-PDU を送信し、タイマ TR002 を起動しなければならない。D_DeRegistrationResponse を受信した場合、FE1 はクリアにしてタイマ TR002 を停止しなければならない。

もし、全てではないが、一つまたはそれ以上のサービスのレジストレーションの解除を行う場合、FE1 は適切な SpoA(FE3)に対し U_SpoAServiceDetachRequest M-PDU を送信し、タイマ TR002 を起動しなければならない。もし一つまたはそれ以上の SpoA が含まれる場合、FE1 はレジストレーションの解除を要求されたサービスに応じて U_SpoAServiceDetachRequest M-PDUs を送信しなければならない。FE3 から D_SpoAServiceDetachConfirm を受信した時点でクリアにしてタイマ TR002 を停止しなければならない。

A.6.2.2 **レジストラ**

FE1 からの U_DeRegistrationRequest を受信した FE2 は、チケットが発行された各々の FE3 に対して D_SpoAClientDetachNotify M-PDU を送信しなければならない。

各々の FE3 から全ての U_SpoAClientDetachConfirm M-PDUs を受信した時点で、FE3 は FE1 に対し D_DeRegistrationConfirm を返送しなければならない。

A.6.2.3 SpoA

U_SpoAServiceDetachRequest を受信した時点で、FE3 は提供された regID に対して、受信したサービス提供チケット(ServiceOfferTicket)のコンテンツを検証しなければならない。もし有効であれば受信した regID に対して予約された全てのリソースを削除し、FE1 に対して ServiceDetached エレメントを TRUE に設定した D_SpoAServiceDetachResponse M-PDU を返送しなければならない。

FE2 から D_SpoAClientDetachNotify を受信した FE3 は、RegistrantId、 RegistrarId 及び ServiceAppId の組に対してレジストラント承認チケット(RegistrantAuthTicket)のコンテンツを検証しなければならない。もし有効であれば FE3 は受信した RegistrantId に対して予約された全てのリソースを削除し、ClientDetached エレメントを TRUE に設定した U_SpoAClientDetachResponse M-PDU を返送しなければならない。

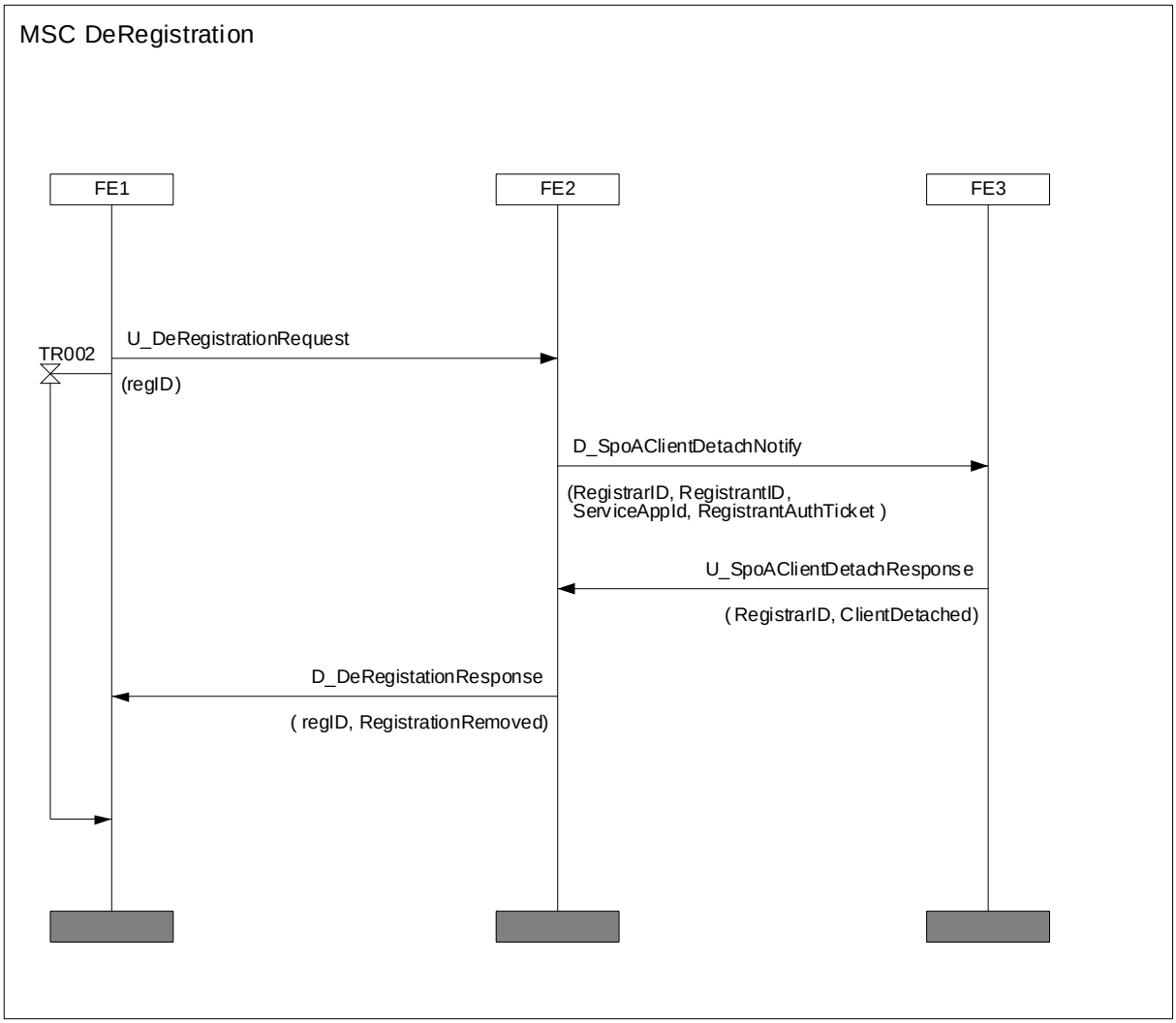


図7　ユーザ起動による全サービスのレジストレーション解除

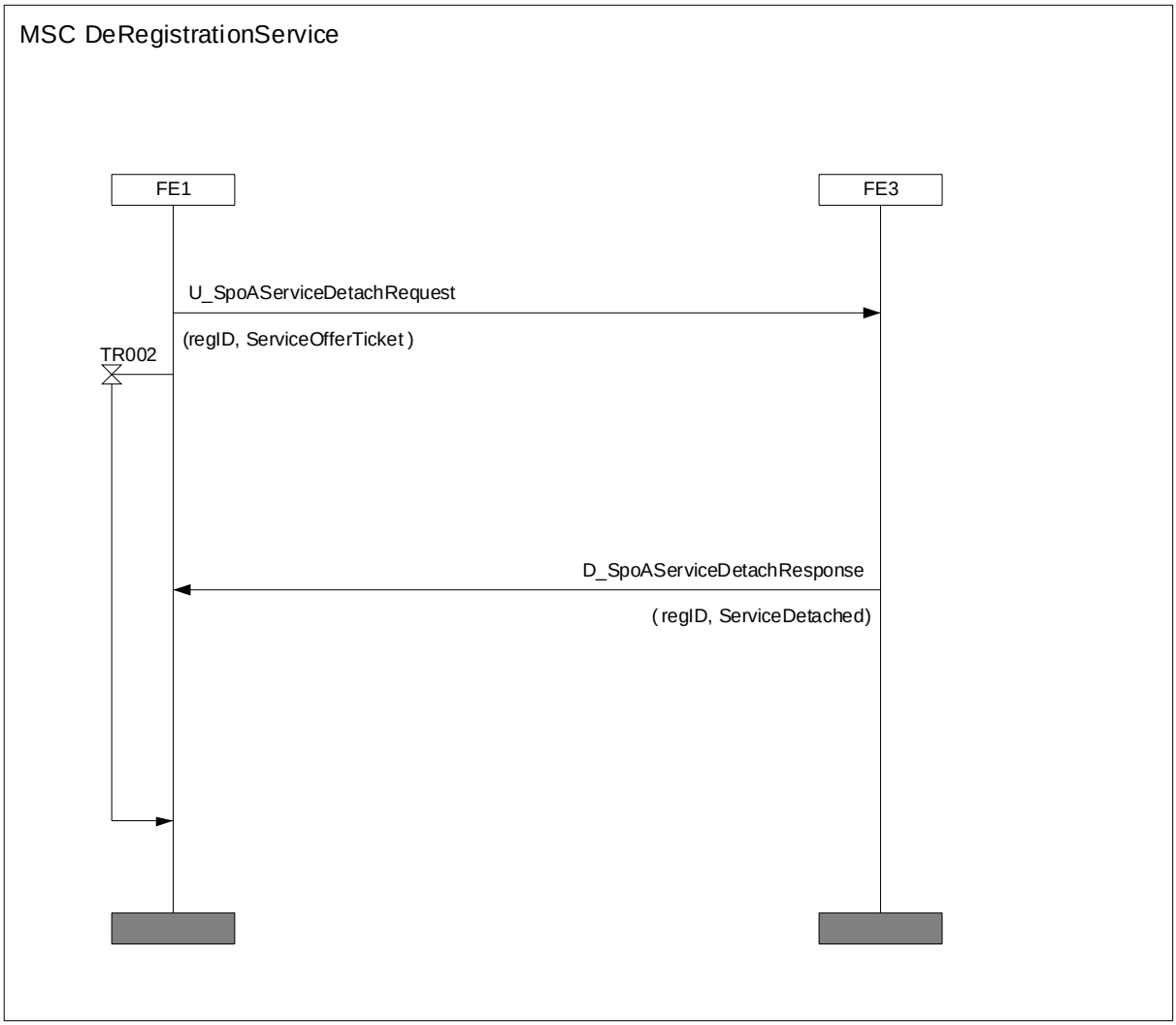


図8　ユーザ起動による個々のサービスのレジストレーション解除

A.7 **タイマ**

A.7.1 TR001, **レジストレーションタイマ**

本タイマのデフォルト値は 30 s でなければならない。

A.7.2 TR002, **レジストレーション解除タイマ**

本タイマのデフォルト値は 30 s でなければならない。

A.8 **データ定義** (ASN.1)

本節では参照点 R に見られる情報フローにおけるデータ定義を示す。

注:　OSS シンタックスチェッカを使用してチェックした場合、この ASN.1 はエラーを含んでいないが、いくつかの M-PDUs が同一形式の構造(the same form of structure)にエンコードされる旨の警告が出る(例えば U_SpoAServiceAttachRequest と D_SpoAServiceAttachResponse は共に RegIDType と TicketType のシーケンス型を含んでいる。)。このことについて、本節では特にエラーとしては扱わず、単なる情報として扱う。

```
TIPHONr3-RefPtR DEFINITIONS ::=

BEGIN

-- Start of M-PDU definitions

M-PDURegistrationPendingType ::= SEQUENCE
```

```
{
    regID           RegIdType
}

M-PDUDeRegistrationRequestType ::= SEQUENCE
{
    regID           RegIdType
}

M-PDUDeRegistrationResponseType ::= SEQUENCE
{
    regID                   RegIdType,
    registrationRemovedFlag BOOLEAN
}

M-PDURegistrationRequestType ::= SEQUENCE
{
    regID               RegIdType,
    registrationMode    RegistrationModeType DEFAULT initialRegistration,
    rpoA                RpoAType,
    serviceAppId        SET OF ServiceApplicationType,
    authToken           AuthenticationTokenType OPTIONAL
}

M-PDURegistrationResponseType ::= SEQUENCE
{
    regID               RegIdType,
    serviceCredential   SET OF TicketType
}

M-PDURegistrationRejectType ::= SEQUENCE
{
    regID               RegIdType,
    serviceRejectReason SET OF ServiceRejectReasonType
}

M-PDUSpoAServiceAttachRequestType ::= SEQUENCE
{
    regID                   RegIdType,
    serviceRequestTicket    TicketType
}

M-PDUSpoAServiceAttachResponseType ::= SEQUENCE
{
    regID               RegIdType,
    serviceOfferTicket  TicketType
}

M-PDUSpoAServiceAttachRejectType ::= SEQUENCE
{
    regID               RegIdType,
    serviceRejectReason ServiceRejectReasonType -- Enumerated list defined by the present
document
}

M-PDUSpoAServiceDetachRequestType ::= SEQUENCE
{
    regID               RegIdType,
    serviceOfferTicket  TicketType
}

M-PDUSpoAServiceDetachResponseType ::= SEQUENCE
{
    regID               RegIdType,
    serviceDetachedFlag BOOLEAN
}

M-PDUSpoAClientAttachNotifyType ::= SEQUENCE
{
    registrarID             RegIdType,
    registantID             RegIdType,
    serviceAppId            ServiceApplicationType,
    registrantAuthTicket    TicketType
}

M-PDUSpoAClientNotifyResponseType ::= SEQUENCE
{
    registrarID         RegIdType,
    clientAcceptedFlag  BOOLEAN,
    spoAAuthTicket      TicketType OPTIONAL -- Conditional on value of flag
```

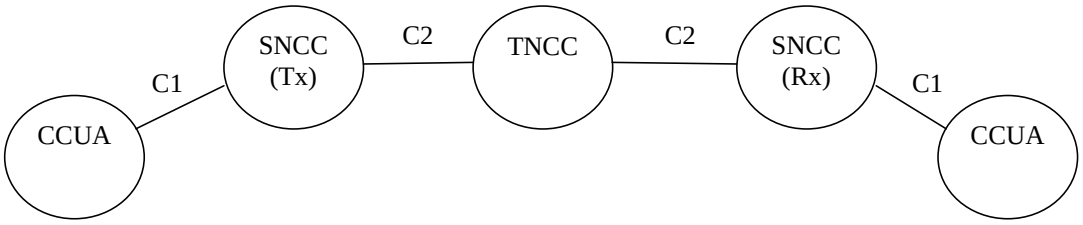
```
}

M-PDUSpoAClientDetachNotifyType ::= SEQUENCE
{
    registrarID             RegIdType,
    registantID             RegIdType,
    serviceAppId            ServiceApplicationType,
    registrantAuthTicket    TicketType
}

M-PDUSpoAClientDetachResponseType ::= SEQUENCE
{
    registrarID         RegIdType,
    clientDetachedFlag  BOOLEAN
}

-- End of M-PDU definitions

-- Start of element definitions

TicketType ::= SEQUENCE
{
    registrantId        VisibleString,
    registrarId         VisibleString,
    serviceCredential   SET OF ServiceCredentialType,
    cryptoDigest        DigestType OPTIONAL
}

ServiceCredentialType ::= SEQUENCE
{
    serviceAppId    ServiceApplicationType,
    spoA            SpoAType, -- shall be in the user's terminal addressing realm
    startTime       GeneralizedTime,
    stopTime        GeneralizedTime, -- Shall be greater than StartTime
    cryptoDigest    DigestType OPTIONAL
}

RegIdType ::= SEQUENCE
{
    registrarId     VisibleString,
    registrarLoc    TRL,
    registrantId    VisibleString
}

TRL ::= SEQUENCE
{
    protocolID      VisibleString,
    nameorAddress       VisibleString,
    port                INTEGER OPTIONAL
}

AuthenticationTokenType ::= TokenType

RegistrationModeType ::= ENUMERATED
{
    initialRegistration (0),
    locationUpdate (1)
}

ServiceRejectReasonType ::= SEQUENCE
{
    serviceAppId        ServiceApplicationType, -- Enumerated list defined by operator
    rejectReason        TIPHONErrorType     -- Enumerated list defined by the present document
}

TIPHONErrorType ::= SEQUENCE
{
    source      ENUMERATED
                {
                    callControl,
                    bearerControl,
                    mediaControl,
                    transportControl,
                    …
                },
    severity    ENUMERATED
                {
                    fatalError,
                    warning,
                    information
```

```
                },
    reason      ENUMERATED
                {
                    invalid,
                    not_Supported,
                    unavailable,
                    does_not_exist,
                    insufficient_resources,
                    …
                },
    diagnostic  TIPHONErrorDiagnosticType OPTIONAL,
    freeText    VisibleString OPTIONAL,
    embeddedError   TIPHONErrorType OPTIONAL
}

SpoAType ::= VisibleString

ServiceApplicationType ::= VisibleString

DigestType ::= VisibleString

TokenType ::= VisibleString

RpoAType ::= VisibleString

-- End of element definitions

END
```

Annex B (normative):**参照点 C におけるメタプロトコル**

B.1 **概要**

TS 101 314[1]内で定義される参照点 C は、同等の呼制御エンティティ間そして同等のベアラ制御エンティティ間に存在する。したがってこの参照点で見えるメタプロトコルは、呼制御エンティティとベアラ制御エンティティのみに関係している。



CCUA = Call Control User Agent
SNCC = Serving Network Call Control
TNCC = Transit Network Call Control

図9　呼制御の参照モデル

TIPHON の呼制御は、例えば呼設定をコミットする前に通信するパーティ間で既知の終端間のベアラサービスが認められるといった、終端から終端へのパス中の各機能グループ内で呼をサポートするベアラ能力が確立しなければならない、断定されたプロトコルである。

注釈:　終端間のベアラサービスは、複数のテクノロジにわたって実現される場合がある。これは、メディア制御（annex C 参照）の影響か、メディア中継間の内部接続機能エンティティの影響かを暗示する。

参照点 C 上の任意のプロトコルの発動に先立って、適切なサービスプロバイダの独自性と位置は、メタプロトコルのレジストレーションを通して（それゆえ受け取ったサービスチケットの中に含まれる（annex A 参照））か又は前もって手配することのどちらかにより、親ターミナルが提供される場合がある。

呼制御サービスはネットワークを介して提示された、ピアリング・サービスである。呼制御サービスは、

表 1 で表される TIPHON 呼制御サービス・アクセスポイント（TCC-SAP）上で可視の、プリミティブの使用によりターミナルで起動されるものとする。

表5　TCC-SAP上で可視の呼制御プリミティブ

Primitive	Short description	Capability (see note)
TCC_CallSetup_req	Used to invoke a call setup	Basic Call Control (see [3], clause 8.1), Carrier Selection (see [3], clause 8.11).
TCC_CallSetup_conf	Used by the CC service to inform the user plane of the success of a call setup attempt	Basic Call Control (see [3], clause 8.1).
TCC_CallSetup_ind	Used by the CC service to inform the user plane of an incoming call setup attempt	Basic Call Control (see [3], clause 8.1).
TCC_CallSetup_resp	Used by the user plane to direct the CC service how to deal with the incoming call setup attempt	Basic Call Control (see [3], clause 8.1).
TCC_CallClear_req/conf	Used by the user plane to invoke a call cleardown	Basic Call Control (see [3], clause 8.1).
TCC_CallClear_ind/resp	Used to indicate to the user plane that the other party has invoked a call cleardown	Basic Call Control (see [3], clause 8.1).
TCC_CallModify_req/conf	Used by the user plane to modify data relating to call in progress, or to add data to a signalling session in progress	Overlap sending of called party identifier information (see [3], clause 8.24).
TCC_CallModify_ind/resp	Used to indicate to the user plane that the call in progress is to be modified, or to request data from the user plane to a signalling session in progress	Overlap sending of called party identifier information (see [3], clause 8.24).
TCC_SetLocalProfile_req/conf	Allows the user plane to set parameters for the **local** call processing. This includes the ability to set Anonymous Call rejection behaviour, and to set calling party identification restrictions	Anonymous Call Rejection (see [3], clause 8.7), Calling User Identity Generation (see [3], clause 8.4).
TCC_CallReport_ind	Used to indicate to the user plane that reports relating to the processing of the call have been received. These may be used to trigger tones in a user's headset (e.g. ringing tone)	Basic Call Control (see [3], clause 8.1).
NOTE:　The capability cross references to the capability definition in TS 101 878 [3].		

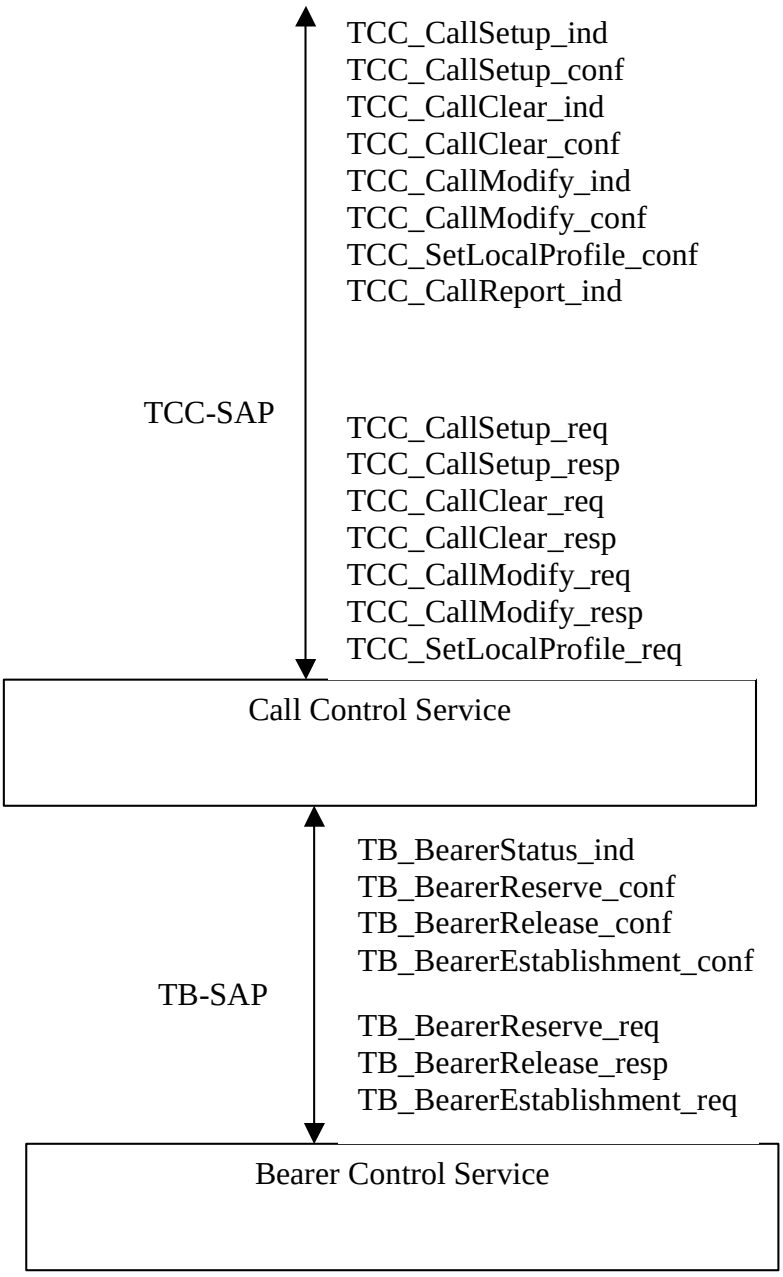
ベアラ制御と確立の機能をサポートするには、表 2 に表されるプリミティブは、TIPHON ベアラ・サービス・アクセスポイント（TB-SAP）上で可視である。

注釈 1:　TB-SAP は現ドキュメントのこの版内のアプリケーション層からは不可視であるが、呼制御エンティティからのみアクセスされる。

注釈 2:　TB-SAP は TS 101 314 [1]内にマップされていない。

表6　TB-SAP上で可視のベアラ制御プリミティブ

Primitive	Short description	Capability (see note)
TB_BearerReserve_req	Allows the upper layer to reserve bearer capability	
TB_BearerReserve_conf	Indicates to the upper layer the result of a bearer reservation request	
TB_BearerRelease_req	Allows the upper layer to delete a bearer reservation	
TB_BearerRelease_conf	Indicates to the upper layer the result of a bearer release request	
TB_BearerStatus_ind	Indicates to the upper layer the current status of the bearer	
TB_BearerEstablishment_req	Used by the upper layer applications to request establishment of previously reserved bearer resources	
TB_BearerEstablishment_conf	Confirms the establishment	
NOTE:　The capability cross references to the capability definition in TS 101 878 [3].		



注釈 1: TCC-SAP**は、**[1]**中で定義された**SC1**参照点を包含する。**
注釈 2: TB-SAP**は**[1]**中で定義されていない。**

図10　呼とベアラ制御サービスを有効にするよう提案するプリミティブ

呼制御メタプロトコルは、ターミナルとネットワーク固有の呼制御間の相互作用のための参照点 C1([1]を参照)、およびネットワーク固有の呼制御エンティティ間の相互作用のための参照点 C2 上に存在する。
ベアラ制御メタプロトコルは、ターミナルとネットワーク固有のベアラ制御間の相互作用のための参照点 C1([1]を参照)、およびネットワーク固有のベアラ制御エンティティ間の相互作用のための参照点 C2 上に存在する。

B.2 機能を構成する要素

呼制御機能エレメントは呼状態を維持し、ベアラ制御機能を使用して呼のメディアフェーズ用資源を割り付けて解放するのと同様に、呼を設定し解放するサポートするメタプロトコルを使用するものとする。

表7　呼制御機能エレメント

Identity	Name
CCUA-O	Call control agent at originating terminal
SNCC-O	Call control agent at SpoA of originating terminal
TNCC	Call control agent in an intermediate network
SNCC-T	Call control agent at SpoA of terminating terminal
CCUA-T	Call control agent at terminating terminal

ベアラ制御機能エレメントはベアラ状態を維持し、ベアラを設定し解放するサポートするメタプロトコル・サービスを利用するものとする。

表8　ベアラ制御機能エレメント

Identity	Name
BCUA-O	Bearer control agent at originating terminal
SNBC-O	Bearer control agent at SpoA of originating terminal
TNBC	Bearer control agent in an intermediate network
SNBC-T	Bearer control agent at SpoA of terminating terminal
BCUA-T	Bearer control agent at terminating terminal

B.3 情報フロー

表 6 に記述された情報の流れは呼制御 M-PDU を含む。これらの M-PDU の正式の ASN.1 宣言が与えられる。呼制御 M-PDU は、参照点 C1[1](終端からネットワークまで)、および C2[1](ネットワーク間(それらは異なる管理下にあるかもしれない))に存在する。

表9　呼制御 M-PDUs

M-PDU name (see notes 1 and 2)	Parameters	M/O/C
U_CallRequest	CallID CallingPartyRestriction CallingPartyID CalledPartyID CallPriority operatorSelection (see note 3) ServiceOfferTicket	M M C M M O O
D_CallReject	CallId CallRejectReason	M M
D_CallReport	CallId ReportReason ReportParameters	M M C
D_CallConnect	CallId	M
U_CCAdditionalDigits	CallID AdditionalDigits	M M
D_CallRequest	CallID CallingPartyID CallingPartyRestriction CalledPartyID CallPriority	M C M M M
U_CallAlert	CallId	M
U_CallConnect	CallId	M

M-PDU name (see notes 1 and 2)	Parameters	M/O/C
NW_CallRequest	CallID CallingPartyID CallingPartyRestriction CalledPartyID CallPriority	M C M M M
NW_CallReport	CallId ReportReason ReportParameters	M M C
NW_CallConnect	CallId	M
NW_CallReject	CallId CallRejectReason	M M
NOTE 1: M-PDUs prefixed by "U_" indicate M-PDUs generated by a terminal and have direction only towards the network (i.e. Up to the network). NOTE 2: M-PDUs prefixed by "D_" indicate M-PDUs generated by the network in the direction only of the terminal (i.e. Down from the network). NOTE 3: operatorSelection may be used to select a preferred carrier or service provider. NOTE 4: The bearer descriptor set contains an unordered description of all bearers able to maintain the call service.		
Encoding of parameter presence in M-PDU: M = Mandatory; C = Conditional; O = Optional		

B.3.1 U_CallRequest

U_CallRequest メッセージは、呼制御(FE6)が呼を確立することを SpoA に要求するターミナル装置(FE5)によって使用されるものとする。

Direction: FE5 to FE6;

Response to: TCC_CallSetup_req;

Response expected: D_CallConnect or D_CallReject or D_CallReport.

注釈 1: ドキュメントのこの版においては、通常、U_CallRequest が U_BearerRequest と同時に送信される。

U_CallRequest のパラメータは、以下で ASN.1 で正式に宣言される型 M-PDUCallRequestType であるものとする。

```
M-PDUCallRequestType ::= SEQUENCE
{
    callId                  CallIdType,
    callingPartyRestriction IdentityRestrictionType,
    callingPartyId          TIPHONPartyIdType OPTIONAL,
    calledPartyId           TIPHONPartyIdType,
    callpriority            TIPHONCallPriorityType DEFAULT normal,
    operatorSelection       OperatorSelectionType OPTIONAL,
    nWRoutingNumber         SET OF RoutingNumberType OPTIONAL,
    nWLocationData          SET OF LocationData OPTIONAL,
    serviceOfferTicket      TicketType OPTIONAL
}
```

注釈 2: nWRoutingNumber と nWLocationData エレメントはネットワーク機能エンティティにのみ使用され、また U_CallRequest では意味を持っていない。

Where:

```
TIPHONPartyIdType ::= CHOICE
{   e164        E164Type,
    url         VisibleString,
    displayName VisibleString
}

IdentityRestrictionType ::= ENUMERATED
{
    identityAvailable (0),
    identityUnavailable (1)
}
```

```
TIPHONCallPriorityType ::= ENUMERATED
{
    normal (0),
    emergency (1),
    authorizedETS (2)
}
```

B.3.2 D_CallReject

FE5 による呼出しの試みを拒絶するために、D_CallReject メッセージは発 SpoA(FE6)によって使用されるものとする。

Direction: FE6 to FE5;

Response to: U_CallRequest;

Response expected: none.

D_CallReject のパラメータは型 M-PDUCallRejectType とする。

```
M-PDUCallRejectType ::= SEQUENCE
{
    callId              CallIdType,
    callRejectReason    TIPHONErrorType
}
```

B.3.3 D_CallReport

ターミナル装置(FE5)に呼び出し試みの進展を報告するために、D_CallReport メッセージは発 SpoA(FE6)によって使用されるものとする。

Direction: FE6 to FE5;

Response to: U_CallRequest or U_CCAdditionalDigits;

Response expected: none.

D_CallReport のパラメータは型 M-PDUCallReportType とする。

```
M-PDUCallReportType ::= SEQUENCE
{
    callId          CallIdType,
    report          TIPHONReportType,
    reportParams    VisbleString OPTIONAL
}
```

Where:

```
TIPHONReportType ::= ENUMERATED
{
    addressComplete (0),
    addressIncomplete (1),
    callProceeding (2),
    callAlerting (3)
}
```

B.3.4 D_CallConnect

呼セット・アップ・フェーズが完了しており、メディア・フェーズに入ることができることを示すために、D_CallConnect メッセージが発 SpoA(FE6)によって使用されるものとする。

Direction: FE6 to FE5;

Response to: U_CallRequest (indirect response);

Response expected: none.

D_CallConnect のパラメータは型 M-PDUCallConnectType とする。

```
M-PDUCallConnectType ::= SEQUENCE
{
    callId      CallIdType
}
```

NOTE: For this edition of the present document D_CallConnect is normally sent concurrently with U_BearerConnect.

B.3.5 U_CCAdditionalDigits

U_CCAdditionalDigits メッセージは、呼確立の初期フェーズを終了させる FE6 に、追加のアドレス数字を提示するターミナル装置(FE5)によって使用されるものとする。

Direction: FE5 to FE6;

Response to: D_CallReport (Incomplete address);

Response expected: D_CallReport.

以下でデータタイプが ASN.1 で正式に宣言される場合、U_CCAdditionalDigits のパラメータは型 M-PDUCCAdditionalDigitsType であるものとする。

```
M-PDUCCAdditionalDigitsType ::= SEQUENCE
{
    callId              CallIdType,
    additionalDigits    SEQUENCE OF DialledDigitType
}
```

Where

```
DialledDigitType ::= VisibleString (SIZE (1)) (FROM ("1234567890#*"))
```

B.3.6 D_CallRequest

ターミナル装置(FE9)への呼を開始するために、D_CallRequest メッセージは終端 SpoA(FE8)によって使用されるものとする。

Direction: FE8 to FE9;

Response to: none;

Response expected: U_CallAlert.

D_CallRequest のパラメータは型 M-PDUCallRequestType とする。

NOTE: For this edition of the present document D_CallRequest is normally sent concurrently with D_BearerRequest.

B.3.7 U_CallAlert

U_CallAlert メッセージは、FE8 への入呼を受理する意図を示すターミナル装置(FE9)によって使用されるものとする。

Direction: FE9 to FE8;

Response to: D_CallRequest;

Response expected: none.

U_CallAlert のパラメータは、以下で ASN.1 で正式に宣言される型 M-PDUCallAlertType とする。

```
M-PDUCallAlertType ::= SEQUENCE
{
    CallId                  CallIdType
}
```

B.3.8 U_CallConnect

U_CallConnect メッセージは、ベアラとメディアの予約が呼のためになされたことを FE8 に示す端末装置(FE9)によって使用されるものとする。

Direction: FE9 to FE8;

Response to: D_CallRequest;

Response expected: D_CallConnectAck.

U_CallConnect のパラメータは、以下で ASN.1 で正式に宣言される型 M-PDUCallConnectType とする。

```
M-PDUCallConnectType ::= SEQUENCE
{
    CallId                  CallIdType
}
```

B.3.9 Void

B.3.10 NW_CallRequest

NW_CallRequest メッセージは、呼を確立すよう FE7(あるいは FE8)に要求するネットワーク装置(FE6)によって使用されるものとする。

Direction: FEx to FEx;

Response to: U_CallRequest (from FE5);

Response expected: NW_CallReport or NW_CallConnect.

NW_CallRequest のパラメータは、型 M-PDUCallRequestType とする。

注釈: 現ドキュメントのこの版においては、通常 NW_CallRequest が NW_BearerRequest と同時に送信される。

B.3.11 NW_CallReport

NW_CallReport メッセージは、呼の経過について報告するネットワーク装置間で使用される。

Direction: FEx to FEx;

Response to: NW_CallRequest;

Response expected: NW_CallConnect or none (see note).

NW_CallReport のパラメータは型 M-PDUCallReportType とする。

注釈: 報告が再ルーティングするのを支持する呼中継の拒否を示すべきである場合、送信されているエンティティは応答を期待しない。

B.3.12 NW_CallConnect

NW_CallConnect メッセージは、呼を確立するよう FE7(あるいは FE8)に要求するネットワーク装置(FE6)によって使用されるものとする。

Direction: FEx to FEx;

Response to: NW_CallRequest;

Response expected: none.

NW_CallConnect のパラメータは型 M-PDUCallConnectType とする。

注釈: 現ドキュメントのこの版については、通常 NW_CallConnect が NW_BearerConnect と同時に送信される。

B.3.13 U_BearerRequest

U_BearerRequest メッセージは、関連する呼のためのベアラを確立することを FE16 に要求する端末装置(FE15)によって使用されるものとする。

Direction: FE15 to FE16;

Response to: none;

Response expected: D_BearerConnect.

U_BearerRequest のパラメータは、以下で ASN.1 で正式に宣言される型 M-PDUBearerRequestType とする。

```
M-PDUBearerRequestType ::= SEQUENCE
{
    bearerId        BearerIdType,
    uplinkBearer    SET OF BearerDescriptorType,
    downlinkBearer  SET OF BearerDescriptorType
}
```

BearerDescriptor が、**QoSServiceClass**(TIPHON サービス・クラス)、およびその **QoSServiceClass** を満足する多数の方法を提供する **FlowDescriptors** のリストを組み合わせるところである。

```
BearerDescriptorType ::= SEQUENCE
{
    serviceClass    TIPHONServiceClassType,
```

```
    flowDescriptor  SET OF FlowDescriptorType
}
```

ServiceClass は TIPHON QoS サービス・クラスを表している。

```
TIPHONServiceClassType ::= ENUMERATED
{
    bestEffort,         -- R value greater than 50 (not guaranteed)
    narrowbandAccep表,  -- R value greater than 50 (guaranteed)
    narrowbandMedium,       -- R value greater than 70 (guaranteed)
    narrowbandHigh,         -- R value greater than 80 (guaranteed)
    wideband                -- R value not defined (guaranteed)
}
```

FlowDescriptor は、メディア・フローを説明し、コーデック、端末遅延およびトランスポート・ディスクリプタに関する情報を含んでいる。

```
FlowDescriptorType ::= SEQUENCE
{
    codecDescriptor     CodecDescriptorType,
    delayBudget         INTEGER, -- In milliseconds
    framesPerPacket     INTEGER,
    transportDescriptor TransportDescriptorType
}
```

TransportDescriptor はフローのトランスポートについての適切な情報をすべて供給する。それは次の情報を含んでいる:

- **maximum gross bit rate:**パケット化と組み立てのオーバーヘッドを含むCODECの最大ビットレート;
- **delay budget:**フローのための許される(残りの)遅延;
- **packet rate:**このパラメータは、適切なバッファー数を決めるのにトランスポートにヒントを与える;
- **packet delay variation:**ネットワーク・オプションに依存する、遅延の中で許されるか達成されるかのどちらかの変動(jitter);
- **packet loss:**ネットワーク・オプションに依存する、許されるか達成されるかのどちらかのトランスポート内のパケットロス;
- **originator transport address:** 送信元のIPアドレスとポート; そして
- **destination transport address:** 受信先のIPアドレスとポート。

```
TransportDescriptorType ::= SEQUENCE
{
    maxCodecGrossBitRate    INTEGER, -- In bits per second
    remainderDelayBudget    INTEGER, -- In milliseconds
    packetRate              INTEGER, -- In packets per second
    packetDelayVariation    INTEGER, -- In milliseconds
    packetLoss              INTEGER, -- In percent
    originatorMpoA          MpoAType,
    destinationMpoA         MpoAType
}
```

注釈: 現ドキュメントのこの版については、通常 U_BearerRequest は U_CallRequest と同時に送信される。

B.3.14 D_BearerConnect

関連する呼のためのベアラを FE15 に示すために、D_BearerConnect メッセージはネットワーク(FE16)によって使用されるものとする。

Direction: FE16 to FE15;

Response to: U_BearerRequest;

Response expected: none.

D_BearerConnect のパラメータは、１つのフロー・ディスクリプタだけが各方向に現われるべきである、型 M-PDUBearerConnectType であるものとする。

```
M-PDUBearerConnectType ::= SEQUENCE
{
    bearerId        BearerIdType,
    uplinkBearer    BearerSelectionType,
    downlinkBearer  BearerSelectionType
}
```

Where:

```
BearerSelectionType ::= SEQUENCE
```

```
{
    serviceClass    TIPHONServiceClassType,
    flowDescriptor  FlowDescriptorType
}
```

B.3.15 NW_BearerRequest

NW_BearerRequest メッセージは、関連する呼のためのベアラを確立するネットワーク機能的な要素(FE16、FE17、FE18)間で使用されるものとする。

Direction: FE16 [to FE17] to FE18;

Response to: U_BearerRequest (from FE15);

Response expected: NW_BearerConnect.

NW_BearerRequest のパラメータは型 M-PDUBearerRequestType とする。

B.3.16 NW_BearerConnect

NW_BearerConnect メッセージは、関連する呼のためのベアラを確立するネットワーク機能的な要素(FE16、FE17、FE18)間で使用されるものとする。

Direction: FE18 [to FE17] to FE16;

Response to: NW_BearerRequest;

Response expected: none.

NW_BearerConnect のパラメータは、1 つのフロー・ディスクリプタだけが各方向に現われるべきである、型 M-PDUBearerConnectType であるものとする。

B.3.17 NW_CallReject

NW_CallReject メッセージは、呼の確立を拒絶するネットワーク装置によって使用されるものとする。

Direction: FE7 to FE6;

Response to: NW_CallRequest;

Response expected: none.

NW_CallReject のパラメータは型 M-PDUCallRejectType とする。

B.4 処理に関する記述

B.4.1 呼設定

呼制御の動作は、有限状態機械および状態間の認められた推移に基いて決定される。TCC の状態は表 6 に定義される。

注釈: ここに定義された状態は、メタプロトコル用のみにあり、候補となるプロトコルによる従順な写像において見えてはならない。従順な写像はステートフルかもステートレスかもしれない。

表10 端末呼制御状態

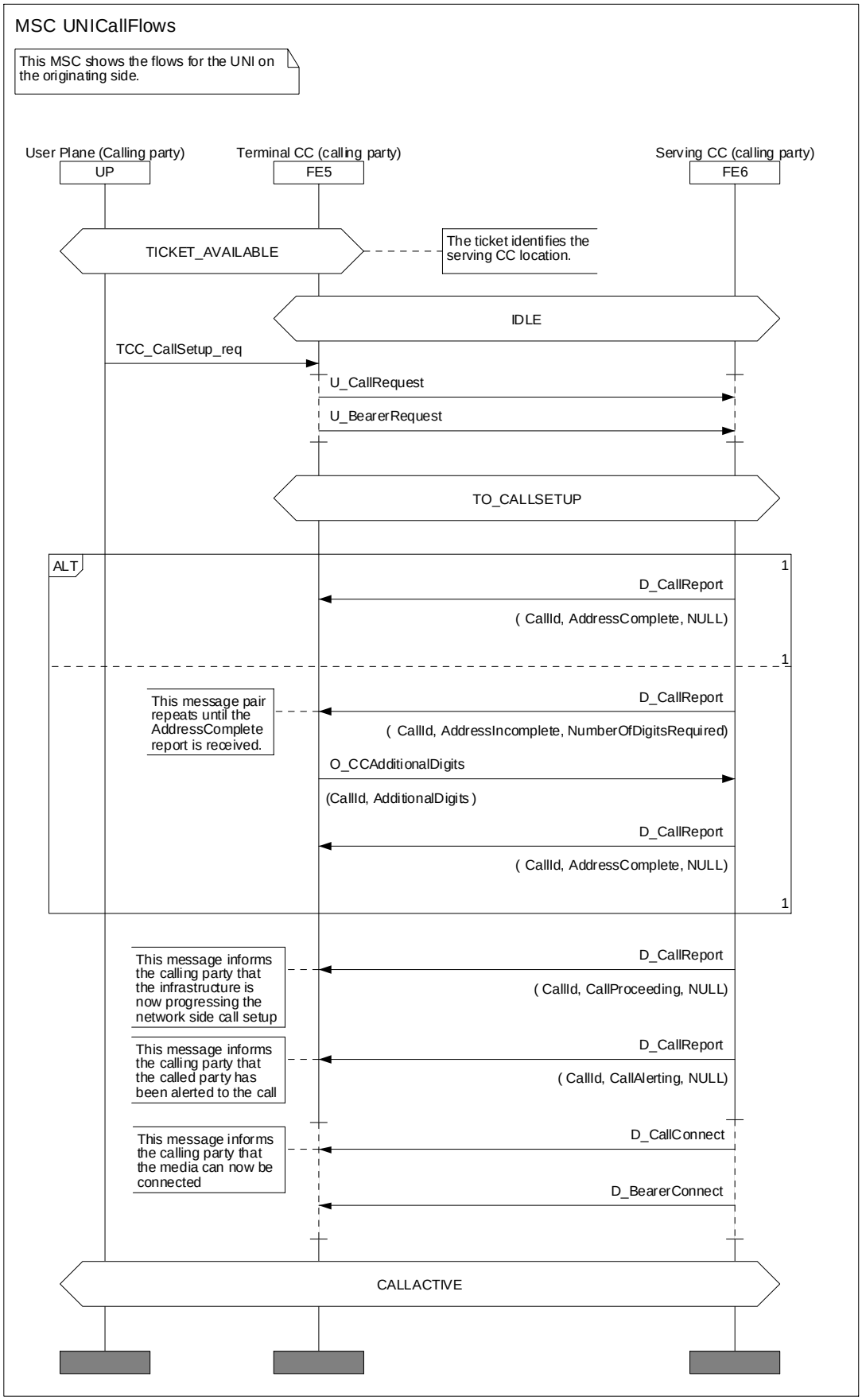
State	Summary description
IDLE	The resting state for the call control entity. This state is reached after successful registration and on opening of the protocol.
CALLACTIVE	The media state of a call.
TO_CALLSETUP	The state during which terminal originated calls are established.
TT_CALLSETUP	The state during which terminal terminated calls are established.
CALLDISCONNECT	The intermediate state whilst awaiting confirmation of call clearance before returning to IDLE state.

B.4.1.1 発信 (端末から発する、端末の動作)

この節中で与えられたテキスト、メッセージ・シーケンス・チャート(MSC)、および SDL は、呼呼制御エンティティ(FE6 への FE5)にかなう端末の標準の動作を示す。

呼設定を始めるために、ユーザ・アプリケーションは、TCC_SAP を越えて呼制御(CC)エンティティ(FE5)に

対して TCC_CallSetup_req プリミティブを転送すべきである。TCC_CallSetup_req は、IDLE 状態にある CC エンティティによって扱われるものとする。
FE5 は、TTC_CallSetup_req を対応する U_CallRequest M-PDU に変換すべきであり、TT_TransportFrame_req プリミティブ(annex D を参照)をサービス特定の SpoA(レジストレーションで識別された)に設定して、送信すべきである。その後、FE5 は TO_CALLSETUP 状態に入るものとし、タイマ TC001 を開始すべきである。
サーバー呼制御エンティティ(FE6)はチケットを有効にすべきである。チケットのどの要素も無効の場合、FE6 は D_CallReject M-PDU を使用して、FE5 に知らせるべきであり、また FE5 は IDLE 状態に戻るべきである。
FE6 は着呼パーティー識別子を有効にすべきである。着呼パーティー識別子が完全な場合、FE6 は、報告理由集合に AddressComplete を、および追加の報告パラメータ・フィールド集合に NULL 設定した D_CallReport M-PDU を使用して、FE5 に知らせるべきである。しかしながら着呼パーティー識別子が不完全な場合、FE6 は、報告理由集合に AddressIncomplete を、不明な数字の個数を示すために追加の報告パラメータ・フィールド集合を設定した D_CallReport M-PDU を使用して、FE5 に知らせるべきである。FE5 は U_CCAdditionalDigits M-PDU で応答すべきである。FE6 が、着呼パーティー識別子が完全であり、報告理由集合に AddressComplete を設定し追加の報告パラメータ・フィールド集合に NULL を設定した D_CallReport M-PDU を使用して、FE5 に知らせることを決めるまで、このやり取りは継続すべきである。
FE6 は、報告理由集合を CallProceeding に、追加の報告書パラメータ・フィールド・集合を NULL に設定した D_CallReport M-PDU を使用して通知することで、以後に発呼リクエストを処理する意図を、FE5 に示すべきである。この時点で CallId フィールドは、この呼のために FE6 によって決めるべきであり、またこの呼のための今後の M-PDU はすべて、この CallId の値を使用すべきである。
端末側(FE8/FE9)から通知される応答においては、報告理由集合に CallAlerting を、追加の報告書パラメータ・フィールド集合に NULL を備えた D_CallReport M-PDU を使用して、FE6 は FE5 に通知すべきである。この点では FE6 が FE5 に対し、呼の中で使用されるべきベアラ特性を示すことができる。
端末側(FE8/FE9)から通知される応答においては、D_CallConnect M-PDU を使用して、FE6 が FE5 に通知するものとする。その後、FE5 は、FE6 によって示されたベアラを確立すべきであり、FE5 はタイマ TC001 をクリアし、CALLACTIVE 状態に遷移すべきである。



NOTE: D_CallConnect is only sent after bearer and media have been established.

図11　UNIでの端末からの発呼の初期フロー(C1)

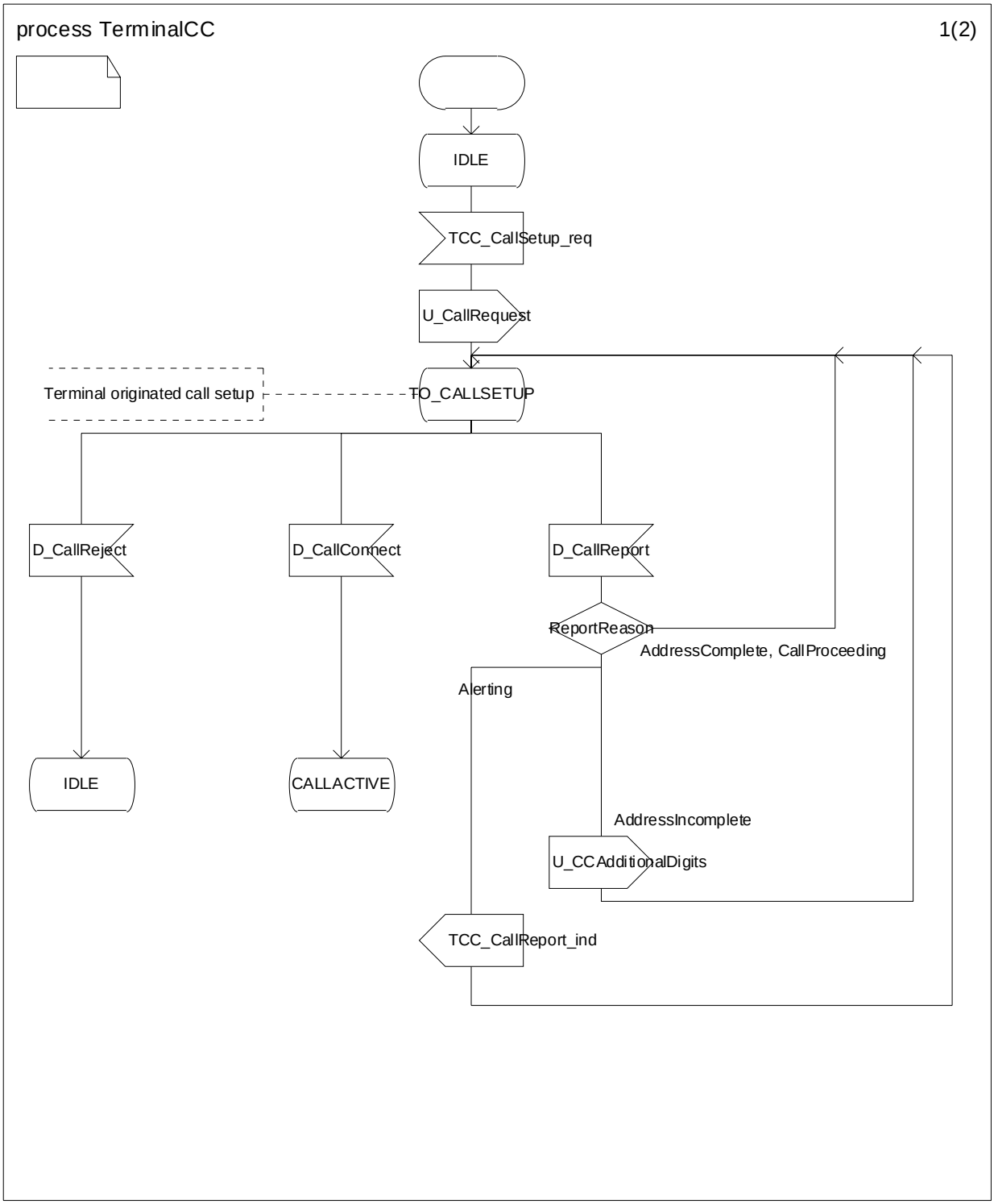


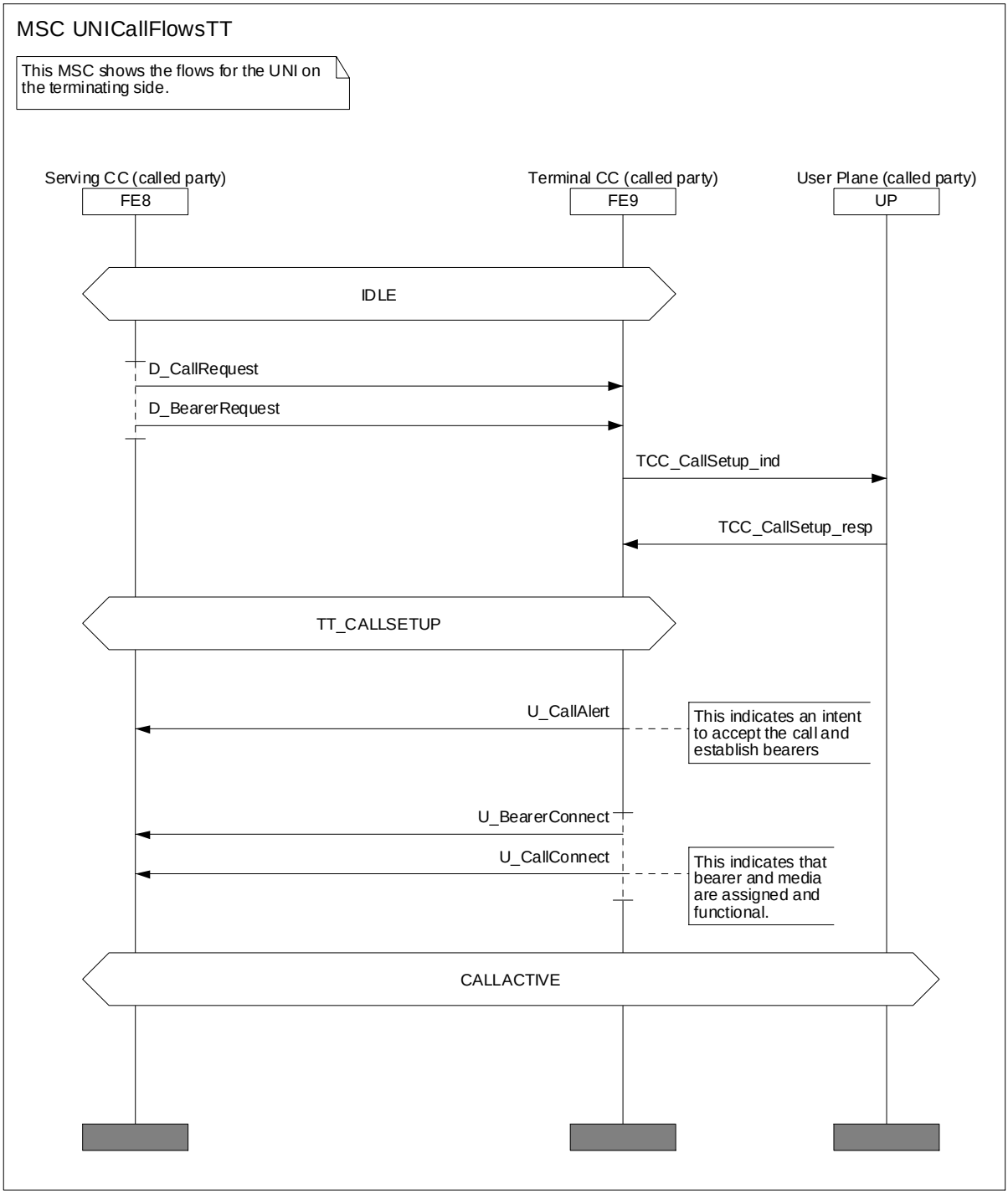
図12　端末からの発呼の場合のUNI状態遷移モデル

B.4.1.2　着信(端末での終端、端末の動作)

端末の呼制御エンティティ(FE9)への着信要求の到着通知は、D_CallRequest M-PDU の受信により行われるべきである。この通知は、FE9 へ呼の受け入れ或いは拒否示す TCC_CallSetup_resp プリミティブを使用して、アプリケーションが応答べき TCC_CallSetup_ind プリミティブを使用して、アプリケーション層に配達されるべきである。D_CallRequest を受信するとすぐに、FE9 は TT_CALLSETUP 状態に入り、タイマ TC002 を開始すべきである。

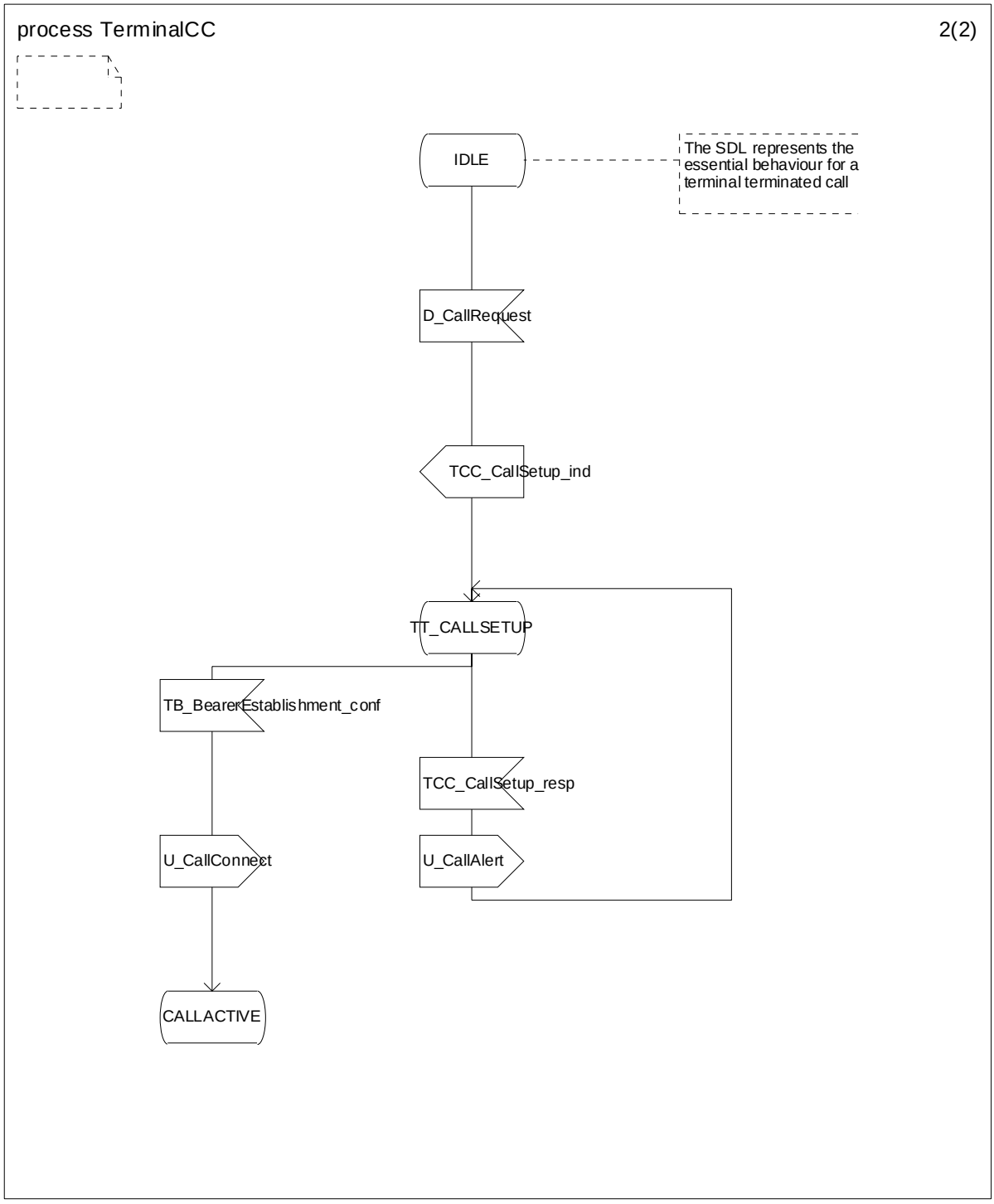
D_CallRequest M-PDU は、呼に関連した全ての後続信号の中で使用される CallId の値を含むべきである。

D_CallRequest M-PDU は、FE8 により呼に割り当てられたベアラ特性を含み、これらを TB_BearerReserve_req プリミティブを使用してベアラ制御エンティティへ渡すべきである。
要求された呼のためのベアラとメディアの資源が作られたことを示す TB_BearerReserve_conf プリミティブの受信において、利用可能な FE9 は、FE8 に U_CallConnect M-PDU を送信し、タイマ TC002 をクリアし、CALLACTIVE 状態に入るべきである。



NOTE: U_CallConnect is only sent after bearer and media have been established.

図13　着呼される端末のUNIフロー



NOTE: The TB_BearerReserve_req is shown to indicate that the bearer and media have been established and thus to allow the active phase of the call to be entered.

図14　着呼される端末のためのUNI状態遷移モデル

B.4.1.3 **ネットワークの動作**

FE6 は FE5 から U_CallRequest M-PDU を受信して TO_CALLSETUP 状態に移行する。

U_CallRequest における ServiceApplication 要素は，要求された QoS とサービスクラスを FE6 に示す。要求されたサービスアプリケーションに対して発呼を試みる端末の認証は，検証される。サービスアプリケーションが認証された場合、呼設定処理を継続する。サービスアプリケーションが認証されない場合，呼設定は拒否され、FE6 は「認証されていないサービス (Service not authorized)」と設定された CallRejectReason を添え

て D_CallReject を返し、FE6 は IDLE 状態に移行する。
FE6 は CalledParty 識別子の正当性を確認しルーチング･アドレス (チェーンにおける次の SpoA の識別子) を決定することによって、次の呼制御サーバへの経路を決定する。「不完全な着端末識別子(Incomplete Called Party Identity)」という理由で着端末識別子の正当性が確認できない場合、FE6 は「アドレス不完全(Address Incomplete)」と設定された ReportReason を添えて D_CallReport を送信することによって追加の数字列を要求する。「未知の着端末識別子 (Unknown Called Party Identity)」という理由で着端末識別子の正当性が確認できない場合、FE6 は「不明な着端末識別子 (Called Party not recognized)」と設定された CallRejectReason を添えて D_CallReject を返し、IDLE 状態に移行する。
着端末アドレスの正当性が確認された場合、FE6 は次のシグナリング要素 (FE7 あるいは FE8) への経路を決定し、次の SpoA (すなわち，FE7 あるいは FE8) へ NW_CallRequest を送信する。

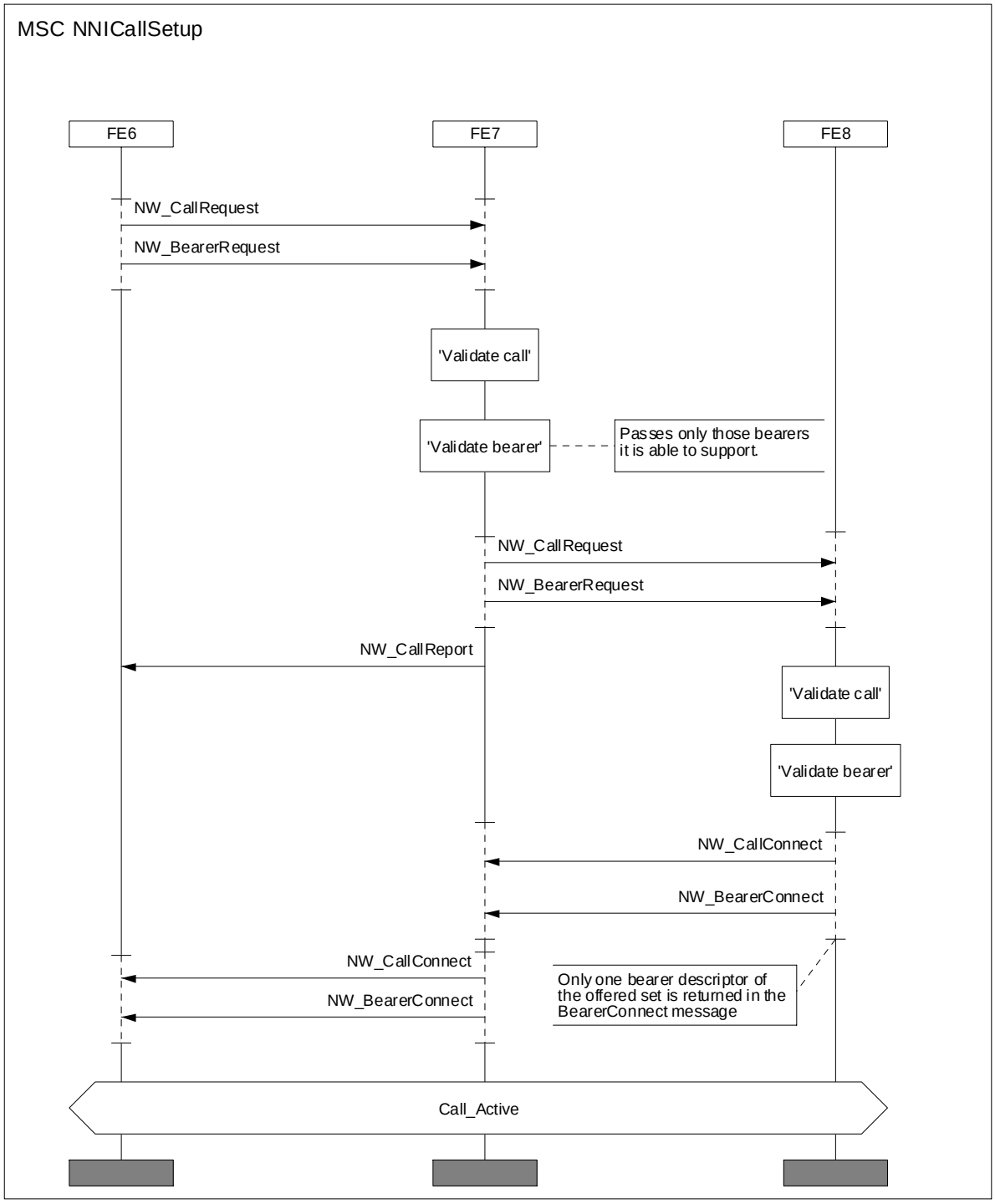


図15　呼設定のためのNNI情報フロー

B.4.2 **呼の解放**

B.4.2.1 **端末の動作**

呼は、発端末あるいは着端末のどちらによっても解放されうる。呼パス (メディアとシグナリング) 中のそれぞれのエンティティは呼解放の結果として IDLE 状態に戻る。

注 1:明快さのために、本記述の残りの部分は CC エンティティの解放に際して FE5 を参照するだろう。.

注 2:本節の動作は、非緊急呼だけに適用する。

解放を行う端末は、TCC_SAP を経由して TCC_CallClear_req プリミティブを FE5 に発行することによって、呼解放フェイズを開始する。FE5 は CALLDISCONNECT 状態に移行し、FE6 に U_CallClear M-PDU を送信し、

WAITRELEASE サブ状態に移行する。

FE6 から D_RELEASE M-PDU を受信した場合、FE5 はアプリケーションに TCC-SAP 経由で TCC_CallClear_conf プリミティブを送信し、CallId を解放し、TB-SAP を経由する TB_BearerRelease_req プリミティブを用いてベアラの解放を開始する。TB-SAP 経由で TB_BearerRelease_conf を受信した場合、FE5 は IDLE 状態に遷移する。

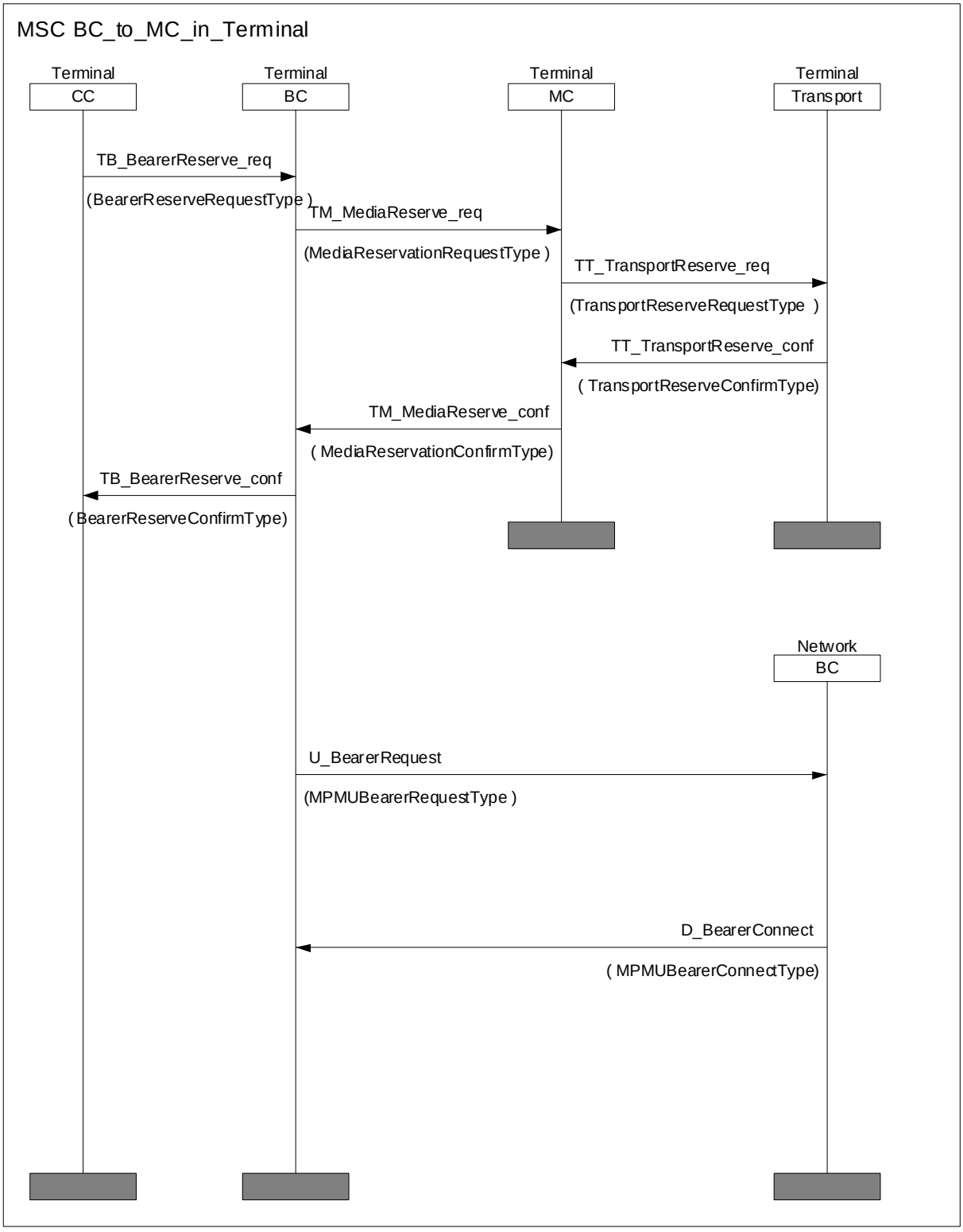


図16　Bearer pre-selection

B.5 タイマ

B.5.1 TC001，呼制御の開始，呼設定タイマ

このタイマのデフォルト値は、30 秒とする。

B.5.2 TC002，呼制御の終了，呼設定タイマ

このタイマのデフォルト値は、30 秒とする。

B.6 データ定義 (ASN.1)

本節は参照点 C において可視である情報フローについての定義を含む。

注: OSS文法チェッカを用いてチェックした場合、このASN.1は誤りを含んではいないが、いくつかのM-PDUは同じ構造形式に符合化すること、あるいは構造中に二重タグが現れることを警告する。本節では、この件は誤りではなく情報として扱う。

```
TIPHONr3-RefPtC DEFINITIONS ::=

BEGIN

--
-- Start of M-PDU data definitions
--

M-PDUCallRequestType ::= SEQUENCE
{
    callId                  CallIdType,
    callingPartyRestriction IdentityRestrictionType,    callingPartyId
    TIPHONPartyIdType OPTIONAL,
    calledPartyId           TIPHONPartyIdType,
    callpriority            TIPHONCallPriorityType DEFAULT normal,
    nwData                  SET OF VisibleString OPTIONAL,
    operatorSelection       OperatorSelectionType OPTIONAL,
    nWLocationData          SET OF LocationData OPTIONAL,
    nWRoutingNumber         SET OF RoutingNumberType OPTIONAL,
    serviceOfferTicket      TicketType OPTIONAL
}

M-PDUCallRejectType ::= SEQUENCE
{
    callId              CallIdType,
    callRejectReason    TIPHONErrorType
}

M-PDUCallReportType ::= SEQUENCE
{
    callId          CallIdType,
    report          TIPHONReportType,
    reportParams    VisibleString OPTIONAL
}

M-PDUCallConnectType ::= SEQUENCE
{
    callId      CallIdType
}

M-PDUCCAdditionalDigitsType ::= SEQUENCE
{
    callId              CallIdType,
    additionalDigits    SEQUENCE OF DialledDigitType
}

M-PDUCallAlertType ::= SEQUENCE
{
    callId      CallIdType
}

M-PDUBearerRequestType ::= SEQUENCE
{
    bearerId        BearerIdType,
    uplinkBearer    SET OF BearerDescriptorType,
    downlinkBearer  SET OF BearerDescriptorType
}
```

```
M-PDUBearerConnectType ::= SEQUENCE
{
    bearerId        BearerIdType,
    uplinkBearer    BearerSelectionType,
    downlinkBearer  BearerSelectionType
}

--
-- End of M-PDU data definitions
--

--
-- Start of data element definitions
--

Addr32 ::= SEQUENCE (SIZE(4)) OF Byte

Addr128 ::= SEQUENCE (SIZE(16)) OF Byte

BearerDescriptorType ::= SEQUENCE
{
    serviceClass    TIPHONServiceClassType,
    flowDescriptor  SET OF FlowDescriptorType
}

BearerIdType ::= INTEGER

BearerSelectionType ::= SEQUENCE
{
    serviceClass    TIPHONServiceClassType,
    flowDescriptor  FlowDescriptorType
}

Byte ::= INTEGER (0..255)

CallIdType ::= INTEGER

CodecDescriptorType ::= VisibleString

DialledDigitType ::= VisibleString (SIZE (1)) (FROM ("1234567890#*"))

DigestType ::= VisibleString

E164Type ::= SEQUENCE
{
    natureOfAddress     NatureOfAddressType,
    screeningIndicator  ENUMERATED
                        {
                            userProvided,
                            notVerifiedUserProvided,
                            verifiedAndPassedNetworkProvided
                        } OPTIONAL,
    digits              SEQUENCE OF TelephoneDigitType
}

NatureOfAddressType ::= ENUMERATED
{
    nationalSubscriberNumber,
    nationalUnknown,
    nationalSignificantNumber,
    internationalNumber,
    nationalNetworkSpecificNumber,
    nationalSignificantRoutingNumber,
    …
}


FlowDescriptorType ::= SEQUENCE
{
    codecDescriptor     CodecDescriptorType,
    delayBudget         INTEGER, -- In milliseconds
    framesPerPacket     INTEGER,
    transportDescriptor SET OF TransportDescriptorType
}

IdentityRestrictionType ::= ENUMERATED
{
    identityAvailable,
    identityUnavailable
}
```

```
IPv4Type ::= SEQUENCE
{
    iPv4address     Addr32,
    ipv4protocol    MpoAProtocolType,
    ipv4port        INTEGER (0..65535)
}

IPv6Type ::= SEQUENCE
{
    iPv6address     Addr128,
    ipv6protocol    MpoAProtocolType,
    ipv6port        INTEGER (0..65535)
}

LocationData ::= VisibleString

MpoAProtocolType ::= ENUMERATED
{
    uDP,
    rTP,
    rTCP_plus_RTCP,
    rTCP
}

MpoAType ::= CHOICE
{
    iPv4address     IPv4Type,
    iPv6address     IPv6Type,
    scn             INTEGER  -- SCN
}

NetworkFGIdType ::= VisibleString

NetworkIdType ::= VisibleString

OperatorSelectionType ::= SEQUENCE OF TelephoneDigitType

RoutingNumberType ::= CHOICE
{
    toSCN   RoutingNumberToSCNType,
    toIP    RoutingNumberToIPType
}

RoutingNumberToSCNType ::= CHOICE
{
    recipientNetworkID      NetworkIDType,
    pointOfInterconnection  MpoAType,
    recipientExchange       SomeTypeC
}

RoutingNumberToIPType ::= CHOICE
{
    serviceProviderIdentity SpoAType,
    pointOfInterconnection  MpoAType,
    recipientNetworkFG      NetworkFGIdType
}

ServiceApplicationType ::= VisibleString

ServiceCredentialType ::= SEQUENCE
{
    serviceAppId    ServiceApplicationType,
    spoA            SpoAType, -- shall be in the user's terminal addressing realm
    startTime       GeneralizedTime,
    stopTime        GeneralizedTime, -- Shall be greater than StartTime
    cryptoDigest    DigestType OPTIONAL
}

SpoAType ::= VisibleString

TelephoneDigitType ::= NumericString (SIZE (1)) (FROM ("1234567890"))

TicketType ::= SEQUENCE
{
    registrantId        VisibleString,
    registrarId         VisibleString,
    serviceCredential   SET OF ServiceCredentialType,
    cryptoDigest        DigestType OPTIONAL
}
```

```
TIPHONCallPriorityType ::= ENUMERATED
{
    normal (0),
    emergency (1),
    authorizedETS (2)
}

TIPHONErrorType ::= SEQUENCE
{
    source      ENUMERATED
                {
                    callControl,
                    bearerControl,
                    mediaControl,
                    transportControl,
                    …
                },
    severity    ENUMERATED
                {
                    fatalError,
                    warning,
                    information
                },
    reason      ENUMERATED
                {
                    invalid,
                    not_Supported,
                    unavailable,
                    does_not_exist,
                    insufficient_resources,
                    …
                },
    dataElementInError  ASN1ElementId OPTIONAL,
    diagnostic  TIPHONErrorDiagnosticType OPTIONAL,
    freeText    VisibleString OPTIONAL,
    embeddedError   TIPHONErrorType OPTIONAL
}

TIPHONPartyIdType ::= CHOICE
{   e164        E164Type,
    url         VisibleString,
    displayName VisibleString
}

TIPHONReportType ::= ENUMERATED
{
    addressComplete (0),
    addressIncomplete (1),
    callProceeding (2),
    callAlerting (3),
    nwCallLegDenialNoData,
    nwCallLegDenialReroutingData, -- Add Rerouting data in reportParams element
}

TIPHONServiceClassType ::= ENUMERATED
{
    bestEffort ,            -- R value greater than 50 (not guaranteed)
    narrowbandAccep表,  -- R value greater than 50 (guaranteed)
    narrowbandMedium,       -- R value greater than 70 (guaranteed)
    narrowbandHigh,         -- R value greater than 80 (guaranteed)
    wideband                -- R value not defined (guaranteed)
}

UserInputType ::= CHOICE
{
    userDialledDigits   SEQUENCE OF DialledDigitType,
    userEnteredString   VisibleString
}

TransportDescriptorType ::= SEQUENCE
{
    maxCodecGrossBitRate    INTEGER, -- In bits per second
    remainderDelayBudget    INTEGER, -- In milliseconds
    packetRate              INTEGER, -- In packets per second
    packetDelayVariation    INTEGER, -- In milliseconds
    packetLoss              INTEGER, -- In percent
    originatorMpoA          MpoAType,
    destinationMpoA         MpoAType
```

```
}

--
-- End of data element definitions
--

END
```

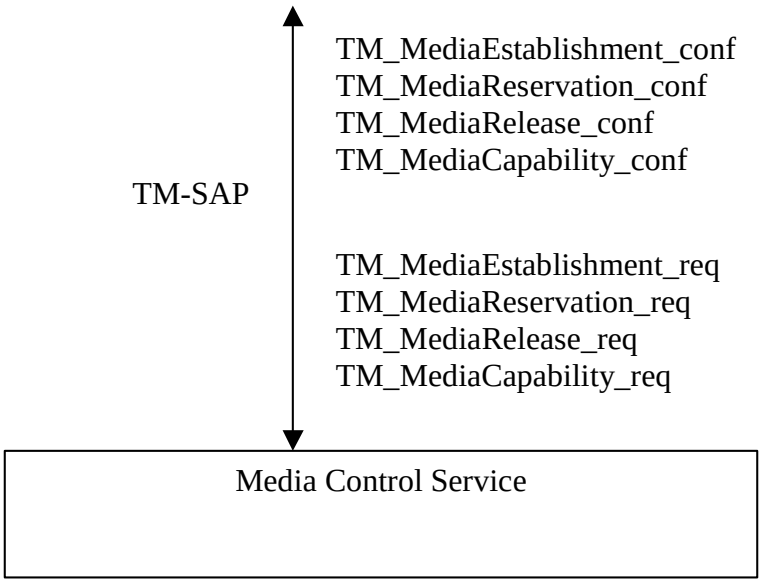
Annex C (normative):参照点 N におけるメタプロトコル

C.1 メディア・コントロール・サービス

メディア･コントロールは、メディア･ストリーム作成のためにリソースの予約や配分を制御する。(例えば、ソフト・コーデックの処理能力の予約のため、あるいはパス・ハード・コーデックにスイッチするためである)

表11　TM-SAPにおいて可視となるメディア・コントロール・プリミティブ

プリミティブ	概要	能力 (注を参照)
TM_MediaEstablishment_req	あらかじめ予約済みのメディア資源の確立を要求するために、アプリケーションによって使用される。	
TM_MediaEstablishment_conf	確立を確認する。	
TM_MediaReservation_req	現在利用可能なメディア・オプションの１つを予約するようメディア・コントローラに要求する。	
TM_MediaReservation_conf	予約を確認する。	
TM_MediaCapability_req	現在利用可能なメディア・オプションの１つについて報告するようメディア・コントローラに要求する。	
TM_MediaCapability_conf	現在利用可能なメディア・オプションを上位レイヤに提供する。	
TM_MediaRelease_req	現在確立されているメディア・フローを解放するようメディア・コントローラに要求する。	
TM_MediaRelease_conf	以前に確立されたメディア・フローが解放されたことを上位レイヤに対して確認する。	
注:　能力は、TS 101 878 [3] における能力定義を相互参照する。		



注:　TM-SAPは [1] において定義された参照点Nを取り囲んでいる。

図17　プリミティブによって識別されるメディア・コントロール・サービスによって提示されたサービス

メディアコントロール(MC)は、呼とベアラの両方をサポートするために要求されたメディアエレメントを確立する。MC は、呼制御メタプロトコルによって識別される QoS クラスに従って QoS 制御されたトランスポート能力を確立するために用いられる。

MC は次のことを行う。

- メディア状態の維持.
- メディアエレメントの設定と解放.

MC は図 18 および 19 に示すような有限状態マシンによって記述される。各状態遷移における MC の動作は記述される。

注: 各予約は、ひとつのメディアリソースと関連するトラフィックリソースだけに適用する。もし複数のメディア予約が要求されたなら、このインタフェースに対する複数の呼び出しが要求される。

表12　メディア・コントロール・プリミティブのパラメータ

プリミティブ	パラメータ	
	要求	確認
TM_MediaReservation	RequestedMediaHandle RxFlowDescriptor TxFlowDescriptor	RequestedMediaHandle Status ReservedMediaHandle ReservedRxFlowDescriptor ReservedTxFlowDescriptor
TM_MediaEstablishment	ReservedMediaHandle	ReservedMediaHandle Status EstablishedMediaHandle
TM_MediaRelease	EstablishedMediaHandle	EstablishedMediaHandle Status
TM_MediaCapability	TBD	TBD

ここで,

- xxMediaHandle は MediaHandleType 型とする。(C.2 節を参照)
- xxFlowDescriptor は FlowDescriptorType 型とする。(C.2 節を参照)
- 状態は、MediaStatusType 型とする。(C.2 節を参照)

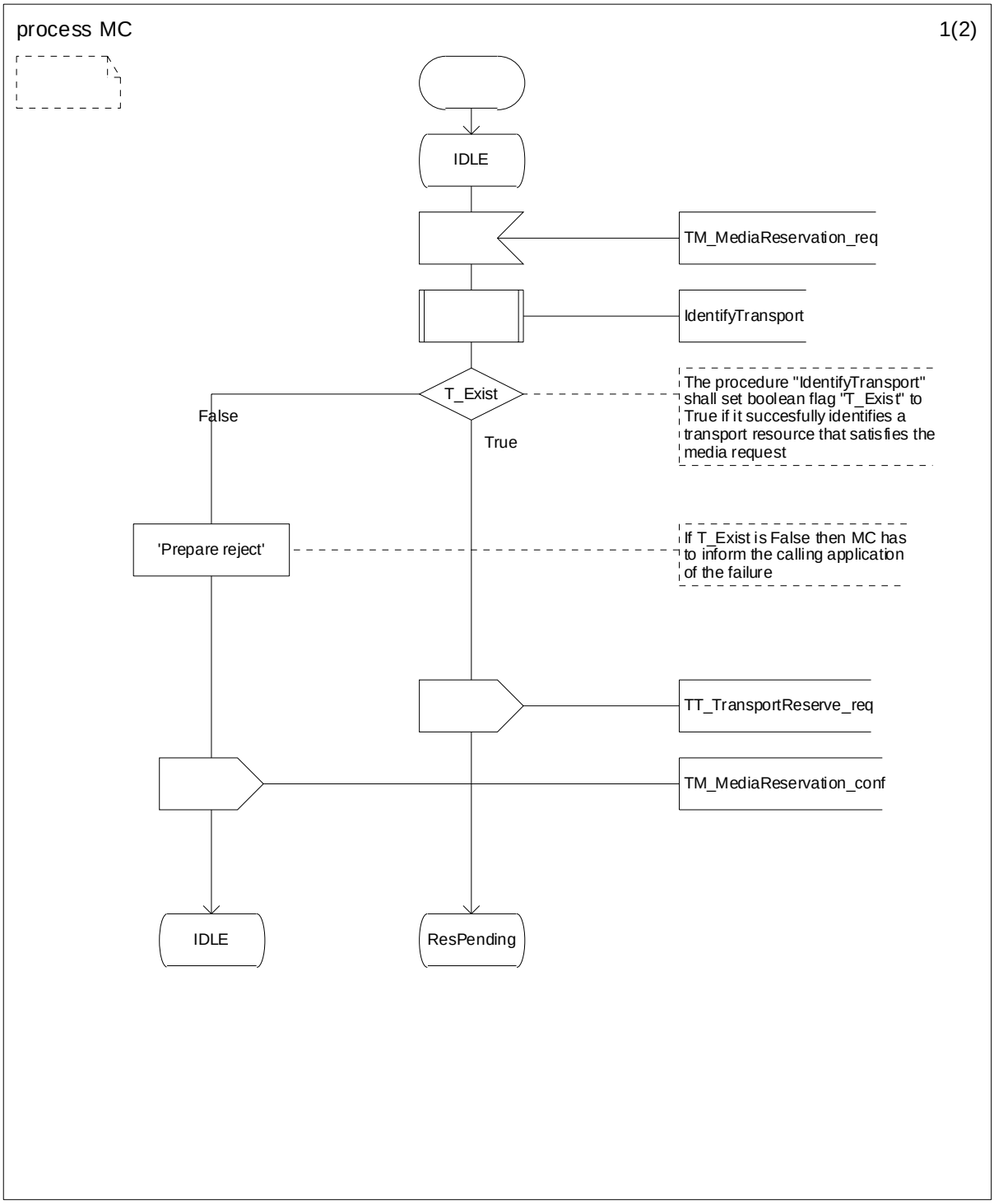


図18　MCプロセスの状態遷移ダイアグラム (page 1 of 2)

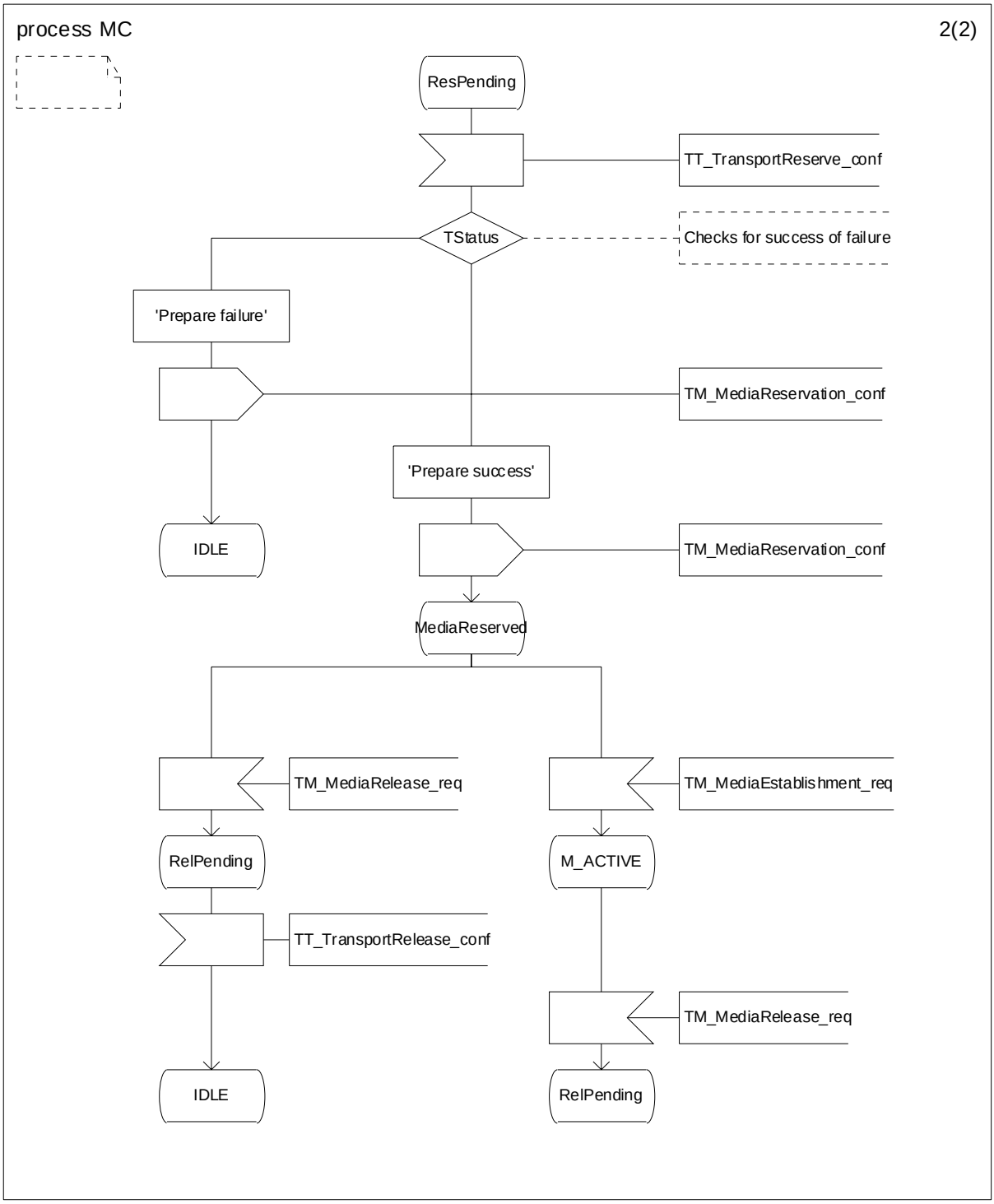


図19　MCプロセスの状態遷移ダイアグラム (page 2 of 2)

表13　MCの状態

状態	遷移メッセージ	次の状態	動作に関する記述
IDLE	TM_MediaReservation_req	ResPending	Clause C.1.1
ResPending	TT_TransportReservation_conf (success) TT_TransportReservation_conf (failure)	MediaReserved IDLE	Clause C.1.2
MediaReserved	TM_MediaRelease_req TM_MediaEstablishment_req	RelPending M_ACTIVE	Clause C.1.3
RelPending	TT_TransportRelease_conf	IDLE	Clause C.1.4
M_ACTIVE	TM_MediaRelease_req	RelPending	Clause C.1.5
注:　　結果として状態変更とならないメッセージは、この表に掲載していない。			

C.1.1 IDLE 状態の動作

IDLE 状態において、メディアの予約およびメディアのアクティベーションは実施しない。

TM_MediaReservation_req メッセージを受信した場合，MC は TM_MediaReservation_req において明らかにされたコーデックに対するトランスポート要求を確認する IdentifyTransport プロシージャを起動し、TT_SAP を経由してトランスポートプレーンに送信される TT_TransportReserve_req に結果を納めて ResPending 状態へ移行する。IdentifyTransport プロシージャがトランスポート資源セットの確認に失敗した場合、MC は「失敗、確認されたトランスポート資源なし (Failure, no transport resource identified)」に設定された状態を添えて TM_MediaReservation_conf を準備し、IDLE 状態に戻る。

C.1.2 ResPending 状態の動作

ResPending 状態では、MC は TT_TransportReserve_conf を待っている。TT_TransportReserve_conf を受信した場合、MC は状態要素の値を調べる。受信した状態値が失敗を示している場合、MC は「失敗、トランスポート資源は利用不能 (Failure, transport resource unavailable)」に設定された状態を備えた TM_MediaReservation_conf を用意する。受信した状態値が成功を示している場合、MC は「成功 (Success)」に設定された状態を備えた TM_MediaReservation_conf を準備し、メディア記述子を返し、MediaReserved 状態に移行する。

C.1.3 MediaReserved 状態の動作

MediaReserved 状態において、メディアは確立され使用状態にされるか、または解放されてよい。TM_MediaRelease_req を受信した場合、この状態に移行する前になされた全予約をトランスポート・コントローラに解放させるために、MC は TT_TransportRelease_req を送信する。TM_MediaEstablishment_req を受信した場合、MC は M_ACTIVE 状態に移行する。

C.1.4 RelPending 状態の動作

RelPending 状態において、トランスポート・プレーンにおけるリソースが解放されていることを示している TT_TransportRelease_conf プリミティブだけに MC は応答し、IDLE 状態に移行する。

C.1.5 M_ACTIV 状態における動作

M_ACTIVE 状態において、メディア・パケットによって搬送されたメディアは割当てられたメディア資源を通して渡される。TM_MediaRelease_req を受信した場合、MC はトランスポート・コントローラに対して TT_TransportRelease_req コマンドを発行し、RelPending 状態へ移行する。

C.2 データ定義 (ASN.1)

本節は，参照点 N において可視となる情報フローについてのデータ定義を含んでいる.

```
TIPHONr3-RefPtN DEFINITIONS ::=

BEGIN

-- Start of primitive definitions

MediaReservationRequestType ::= SEQUENCE
{
    invokingControllerReference MediaHandleType,
    rxFlowDescriptor        SET OF FlowDescriptorType,
```

```
    txFlowDescriptor        SET OF FlowDescriptorType
}

MediaReservationConfirmType ::= SEQUENCE
{
    invokingControllerReference MediaHandleType,
    status                  MediaStatusType,
    mediaDescriptor         SET OF MediaDescriptorWithHandle
}

MediaEstablishementRequestType ::= SEQUENCE
{
    mcReservedMediaReference        MediaHandleType
}

MediaEstablishmentConfirmType ::= SEQUENCE
{
    mcReservedMediaReference        MediaHandleType,
    status                  MediaStatusType,
    mcEstablishedMediaReference MediaHandleType OPTIONAL
}

MediaReleaseRequestType ::= SEQUENCE
{
    mcEstablishedMediaReference MediaHandleType
}

MediaReleaseConfirmType ::= SEQUENCE
{
    mcEstablishedMediaReference MediaHandleType,
    status                  MediaStatusType
}

-- End of primitive definitions

-- Start of element definitions

MediaDescriptorWithHandle ::= SEQUENCE
{
    mcReservedMediaReference            MediaHandleType,
    reservedrxFlowDescriptor    FlowDescriptorType,
    reservedtxFlowDescriptor    FlowDescriptorType
}

MediaStatusType ::= TIPHONErrorType

MediaHandleType ::= VisibleString

FlowDescriptorType ::= SEQUENCE
{
    codecDescriptor         CodecDescriptorType,
    framesPerPacket         INTEGER,
    frameRate               INTEGER,
    transportDescriptor     SET OF TransportDescriptorType
}

TransportDescriptorType ::= SEQUENCE
{
    genericQoSDescriptor    GenericQoSDescriptorType,
    specificQoSDescriptor   SpecificQoSDescriptorType OPTIONAL,
    originatorMpoA          MpoAType,
    destinationMpoA         MpoAType
}

SpecificQoSDescriptorType ::= VisibleString

GenericQoSDescriptorType ::= SEQUENCE
{
    delayBudget             INTEGER, -- In milliseconds
    maxPacketSize           INTEGER, -- In bits
    packetRate              INTEGER, -- In packets per second
    packetDelayVariation    INTEGER, -- In milliseconds
    packetLoss              INTEGER - In percent
}

CodecDescriptorType ::= SEQUENCE
{
    codecID                     ENUMERATED {g711, ...},
    silenceSuppressionEnabled   BOOLEAN,
    codecSpecificParameters VisibleString
```

```
}

TIPHONErrorType ::= SEQUENCE
{
    source      ENUMERATED
                {
                    callControl,
                    bearerControl,
                    mediaControl,
                    transportControl,
                    …
                },
    severity    ENUMERATED
                {
                    fatalError,
                    warning,
                    information
                },
    reason      ENUMERATED
                {
                    invalid,
                    not_Supported,
                    unavailable,
                    does_not_exist,
                    insufficient_resources,
                    …
                },
    diagnostic  TIPHONErrorDiagnosticType OPTIONAL,
    freeText    VisibleString OPTIONAL,
    embeddedError   TIPHONErrorType OPTIONAL
}

Byte ::= INTEGER (0..255)

IPv4Type ::= SEQUENCE
{
    iPv4address     Addr32,
    ipv4protocol    MpoAProtocolType,
    ipv4port        INTEGER (0..65535)
}

IPv6Type ::= SEQUENCE
{
    iPv6address     Addr128,
    ipv6protocol    MpoAProtocolType,
    ipv6port        INTEGER (0..65535)
}

MpoAType ::= CHOICE
{
    iPv4address     IPv4Type,
    iPv6address     IPv6Type,
    scn             INTEGER  -- SCN
}

MpoAProtocolType ::= ENUMERATED
{
    uDP,
    rTP,
    rTCP_plus_RTCP,
    rTCP
}

Addr32 ::= SEQUENCE (SIZE(4)) OF Byte

Addr128 ::= SEQUENCE (SIZE(16)) OF Byte

-- End of element definitions

END
```

Annex D (normative):**参照点** T **におけるメタプロトコル**

D.0 **序論**

トランスポート層はアプリケーション層に対して次のサービスを提供する。 (TIPHON Transport Service

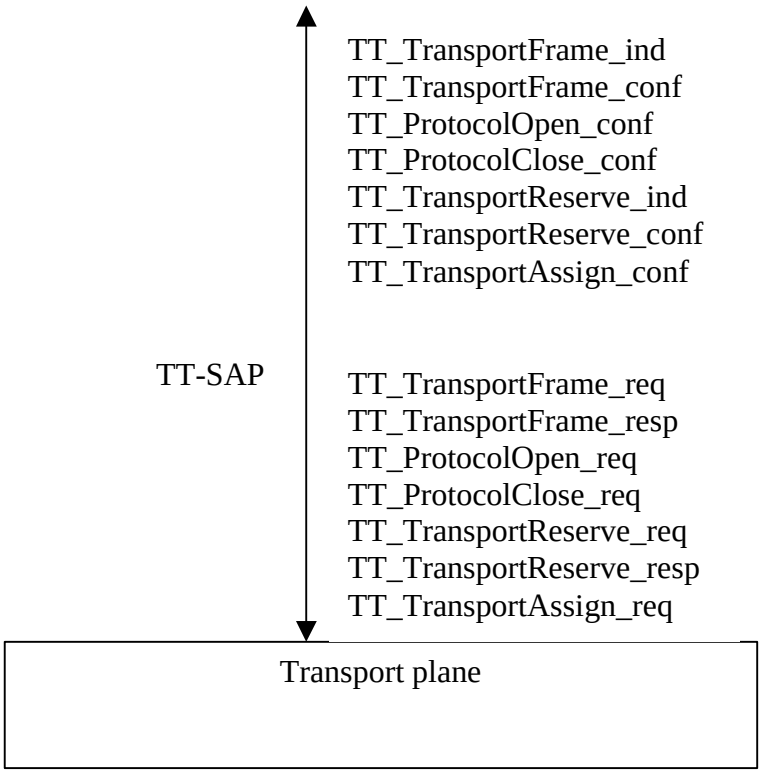
Access Point (TT_SAP)にて認識できる。):

- フレームの転送(通常はCCもしくは レジストレーションからM-PDUへ).
- 特定転送能力の予約.

これらのサービスは表 14 に示される primitive のセットを通して設定される。

表14　TT-SAPにて認識できるTransport primitives

Primitive	備考
TT_TransportFrame_req/conf	データフレームを伝送するための transport plane への応答
TT_TransportFrame_ind/resp	データフレームを受信したことの Transport plane から application plane への表示
TT_TransportReserve_req/conf	transport capability を予約するための transport plane に対する要求. QoS transport establishment にて使用される
TT_TransportReserve_ind/resp	Transport capability を受信したことの transport plane から application plane への表示
TT_TransportAssign_req/conf	transport reservation から transport assignment への移動
TT_TransportRelease_req/conf	以前の予約された transport capability を解放するための transport plane への要求
TT_ProtocolOpen_req/conf	特定のプロトコルの為の transport plane から application plane へのパスを開くための要求 .
TT_ProtocolClose_req/conf	特定のプロトコルの為の transport plane から application plane へのパスを閉じるための要求 .
TT_TransportCapability_req/conf	transport plane の現在能力を要求するための application plane の許可
TT_TransportCapability_ind/resp	application plane にその現在能力を表示するための transport plane の許可



注:　　TT-SAPは[1]で定義されるreference point Tを含む

図20　primitivesによって定義されたTrasnport planeによって提供されるサービス

表15　transport primitives (req/conf)のパラメータ

Primitive	パラメータ	
	Request	Confirm
TT_TransportReserve	TargetTransportDescriptor EndPointId	ReservedTransportHandle ReservedTransportDescriptor
TT_TransportAssign	ReservedTransportHandle	AssignedTransportHandle
TT_TransportFrame	TransportHandle EndPointId DataField	ResponseDataField
TT_TransportRelease	AssignedTransportHandle	-
TT_ProtocolOpen	ProtocolId	ProtocolOpenResult
TT_ProtocolClose	ProtocolId	ProtocolCloseResult
注: TransportHandle は成功したservationで返される。TransportHandleの予め定義された値はnon-QoS transportのために使用される		

表16　transport primitives (ind/resp)のパラメータ

Primitive	パラメータ	
	Indication	Response
TT_TransportFrame	DataField ConfirmationFlag ResponseToEndPointId	ResponseDataField

Primitives は D.2 項の ASN.1 にて完全に定義されている。

D.1 Transport control のステートマシーン

Transport Control (TC)のための transport resources を制御する QoS 割付派有限のステートマシーンによって図 21 に示される。TC の各状態遷移での動作を記述する。

QoS controlled transport が要求されないシグナリングパケットの伝送時にはこのステートマシーンを使用する必要はない。

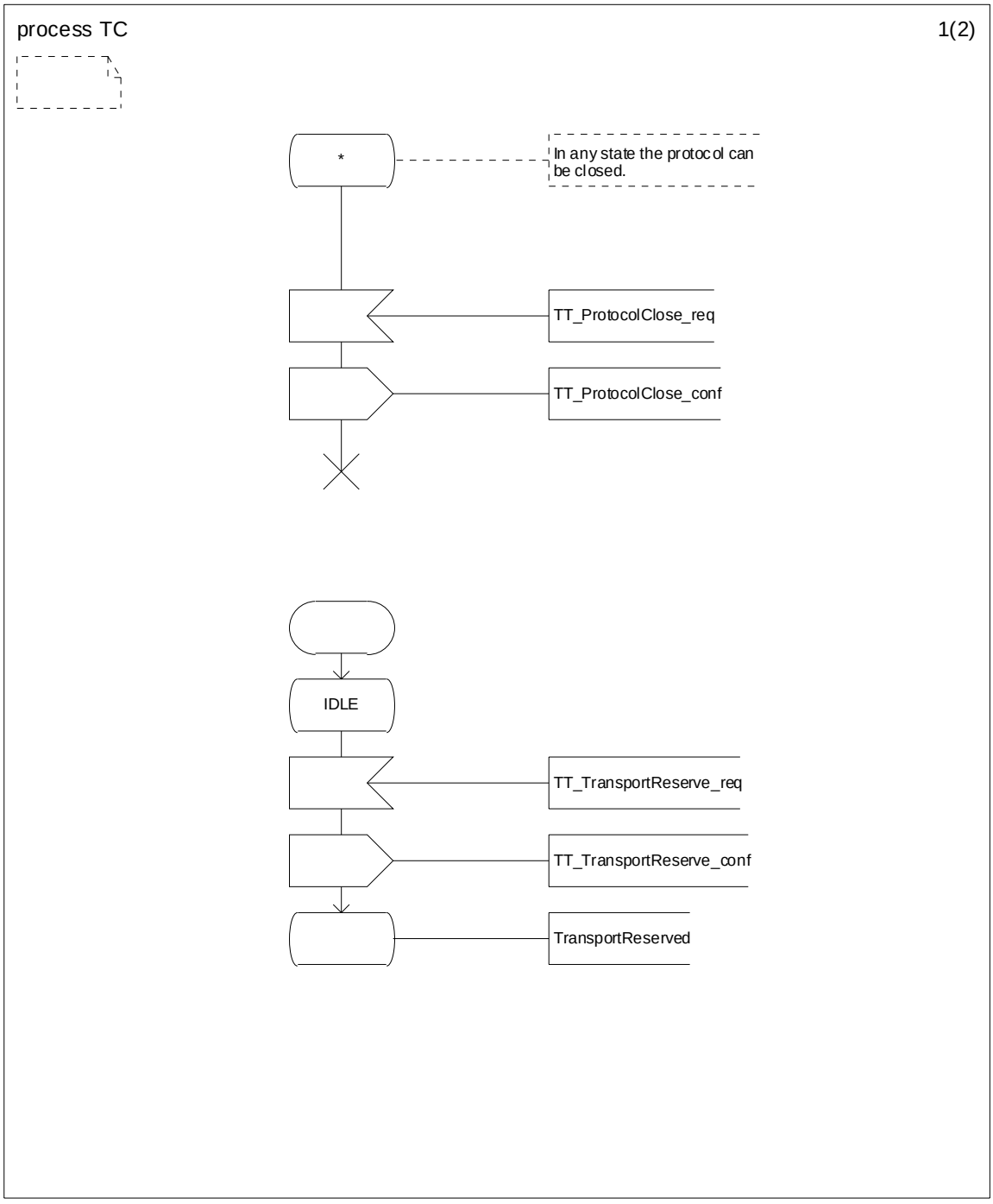


図21　TransportControlの状態遷移

表17　TCの状態

状態	Transition message	Next state	Description of behaviour
IDLE	TT_TransportReserve_req	TransportReserved	Clause D.1.1
TransportReserved	TT_TransportAssign_req TT_TransportRelease_req	TransportActive IDLE	Clause D.1.2
TransportActive	TT_TransportRelease_req	IDLE	Clause D.1.3
NOTE:　状態の変化のないﾒｯｾｰｼﾞは本表には表示されていない。			

D.1.1 IDLE 状態の動作

プロトコルがオープンして TT_ProtocolOpen_req/conf が交換された時に IDLE 状態に入る。 IDLE 状態は各

プロトコルに適用される。　IDLE 状態の時は、トラヒックを有効にするための QoS の transport resource は予約されない。そしてトラヒックを有効にするための QoS の transport resource は活動しないことは有効であるべきである。

TT_TransportReserve_req を受信次第 TC process は応答に対する resources 割り当てを予約するべきであり、TT_TransportReserve_conf primitive を使用する予約を確認して TransportReserved 状態に移動する。

ステートマシーンは TT_ProtocolClose_req/conf 交換を使用したプロトコルが終了した時に終了されるべきである。

D.1.2 TransportReserved 状態の動作

TT_TransportAssign_req を受信次第、 TC process は resource 確認の予約を確認するべきであり、TT_TransportAssign_conf primitive を使用した割り当ての確認から TransportActive 状態に移動する。

D.1.3 TransportActive 状態の動作

TransportActive 状態になった時は、 QoS のデータパケットは転送されるべきである。

データフレームを A (application plane の中)から B (application plane の中)に送出するための図 22 のメッセージシーケンスチャートは primitives の相互作用を示す。

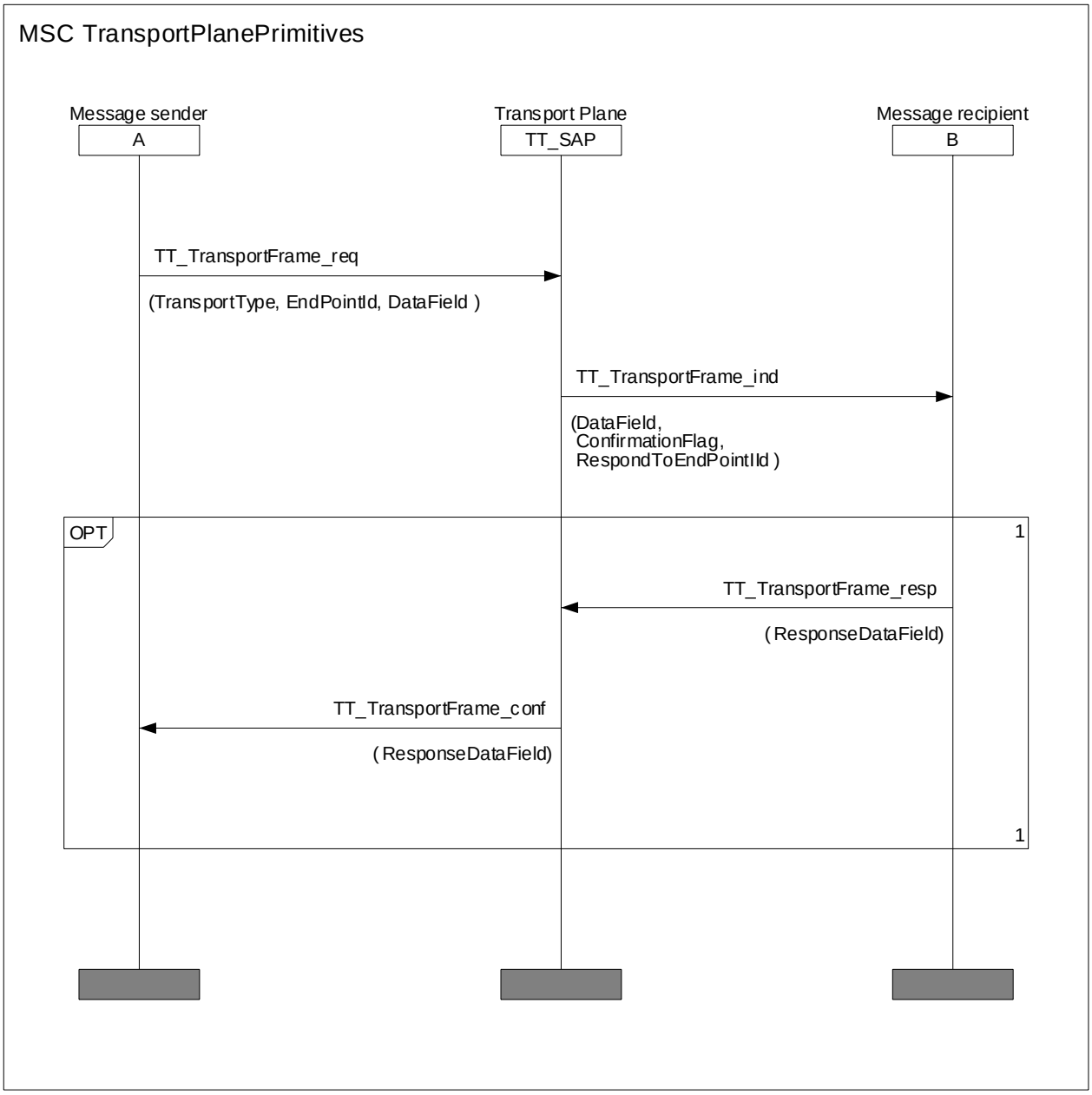


図22　application planeからtransport planeへのフレーム転送のMSC

図 22 において、応答と確認フローはオプションとして示している。オプションが true にセットされたときはデータ転送を確認し、false のときはそれ以外となる。

図 24 は転送予約応答のための類似した MSC を示す。典型的にはこれは呼の acitive phase の時に使用される会話のための QoS 制御チャネルを準備するための call setup 中に使用される。IP メディアケースの中では経路は単一方向であり B-party は類似した予約が必要となる。

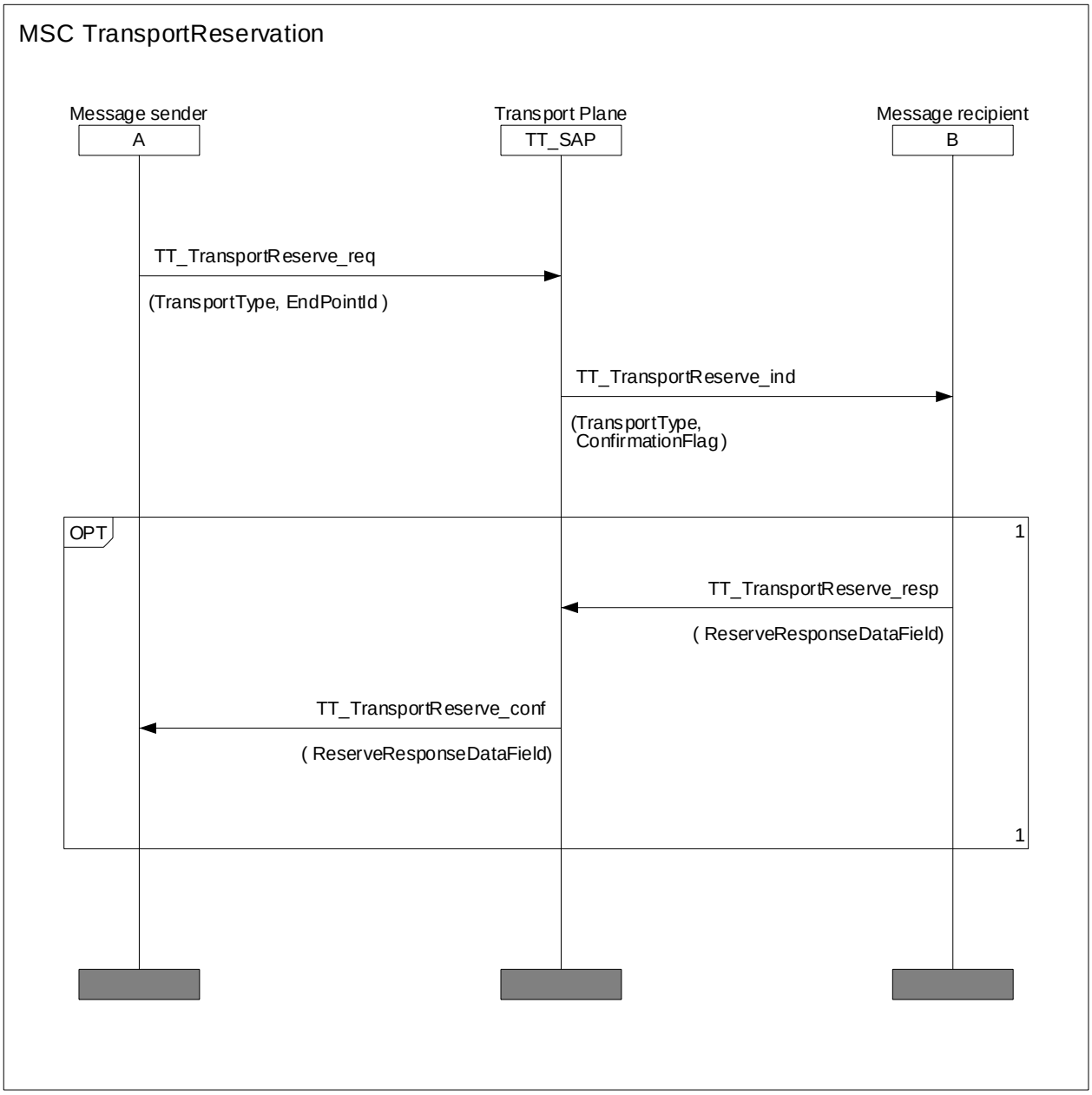


図23　application planeからtransport planeへの転送予約のMSC

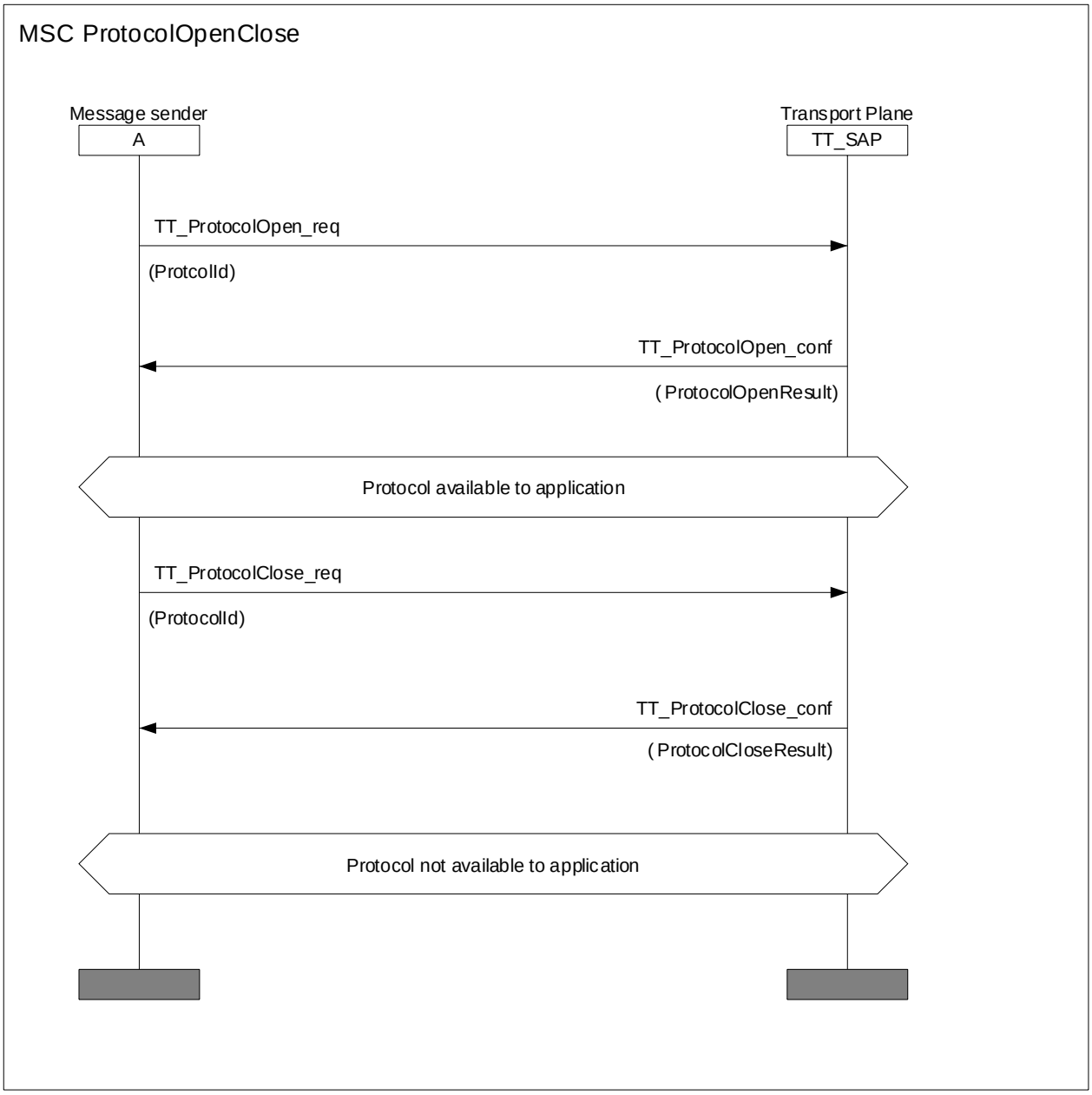


図24　application planeがtransport planeにプロトコルのopenまたはcloseを要求するMSC

D.2 Data **定義** (ASN.1)

本項目は reference point T で見える情報フローの為の data 定義の情報を包含する。

```
TIPHONr3-RefPtT DEFINITIONS ::=

BEGIN

TransportReserveRequestType ::= SET OF TransportDescriptorType

TransportReserveConfirmType ::= SET OF TDTWithHandleType

TDTWithHandleType ::= SEQUENCE
{
    handle      TransportHandleType,
    transport   TransportDescriptorType
}

TransportHandleType ::= VisibleString

TransportAssignType ::= SET OF TransportHandleType
```

```
TIPHONErrorType ::= SEQUENCE
{
    source      ENUMERATED
                {
                    callControl,
                    bearerControl,
                    mediaControl,
                    transportControl,
                    …
                },
    severity    ENUMERATED
                {
                    fatalError,
                    warning,
                    information
                },
    reason      ENUMERATED
                {
                    invalid,
                    not_Supported,
                    unavailable,
                    does_not_exist,
                    insufficient_resources,
                    …
                },
    diagnostic  TIPHONErrorDiagnosticType OPTIONAL,
    freeText    VisibleString OPTIONAL,
    embeddedError   TIPHONErrorType OPTIONAL
}

END
```

Annex E (normative):PICS 様式の表紙

現在の文書の本文に関して、著作権に関する規制条項があるにも関わらず、その文書のユーザーは 付属資料の PICS 様式を、意図した目的に使用できるよう自由に複製できる、さらにはその完成した PICS 様式を自由に出版できる、ということを ETSI は承諾している。

PICS（Protocol Implementation Conformance Statement）

E.1 PICS 様式完成の手引き（Guidance for completing the PICS proforma）

E.1.1 目的と構造（Purposes and structure）

PICS 様式の表紙内容はさらに細かい次の情報カテゴリーの項目に分けられる。

- PICS 様式を完成させるための手引き
- 実装の認証
- プロトコルの認証
- 様式の全体記述

E.1.2 略語と協定（Abbreviations and conventions）

本付属資料中の PICS 様式 は表になった情報から成り立つ。

Item column

item column は番号を含む。その番号は表中の item を認証する。

Item description column

item description column は各々それぞれの item のフリーテキストで記述される。(例えば parameters, timers, etc.).

ぞれは暗に < item description >は実装によってサポートされるか？を意味している。

Status column

次の表記法は ISO/IEC 9646-7 にて定義される。また status column で使用される。

M or m	必須である（mandatory） –能力がサポートされることが要求される。
O or o	オプション（optional） – 能力がサポートもしくは未サポートでも良い。
N/A or n/a	あてはまらず（not applicable） -与えられた状況において能力を使用するのは不可能。
X or x	禁止(prohibited) (除く) – 与えられた状況で能力を使わないことが要求される。
O.i or o.i	条件付きのオプション(qualified optional) – 相互に両立しない。または選択によるオプション. "i" は整数であり、関連したオプションアイテムのユニークなグループであることを確認する。そしてすぐに次の表で定義される論理の選択である。
Ci.j or ci.j	条件付き (conditional)- 要求された能力の ("m", "o", "x" or "n/a") は、他のオプションのサポートや条件付アイテムに依存する。 "i"は整数であり表を区別する 。そして "j" は整数であり連続的に表の中で割り当てられる。Ci.j はすぐに次の表で定義されるユニークな条件付状態表現で構成される。
I or i	不適切（irrelevant） (範囲外) – 能力の要求は参照の使用の範囲外である。 供給者からは応答無し。

Reference column

reference column は reference ドキュメント中の referring clause を参照する。ただし他の点で明確に述べられている場所を除く。

Support column

support column は実装のの供給者に記述されるべきである。次の通常の表記法は ISO/IEC 9646-7 で定義され、support column で使用される。

Y or y	実装によってサポートされる。
N or n	実装によってサポートされない。
N/A or n/a	回答は要求されない。(状態が n/a の時だけ許可,直ちにもしくは条件付き状態での評価後).

もしこの PICS 様式がシステムの中で複数のプロファイルサポートを記述するために完了したならば、能力

が１つのプロファイルの為であり他のプロファイルはサポートしないという回答ができる必要がある。そのようなケースの場合、供給者は「？」の前に記述する（例えば？３）条件付き表式にユニークなリファレンスに入るべきである。この表式は表の最終行に提供されるコメントの空欄に与えられる。それは SCS のなかで定義される述語を使用する。そして各々がシングルプロファイルをもしプロファイルが使用されていえるならば参照して TRUE の値を得る。

例: ?3: IF prof1 THEN Y ELSE N.

表の最終行に提供されるスペースに回答としてのコメントを提供することは可能である。

注: ISO/IEC 9646-7 にて記述される，受信 PDU をサポートしており、PDU の全ての有効なパラメータを文法的関係に説明する能力を要求する。有効なパラメータを説明する能力が無い間に PDU をサポートする事は適合しない。PDU におけるパラメータをサポートすることはパラメータの記号論をサポートすることを意味する。

Values allowed column

values allowed column は type, the list, the range, or the length of values を含む。次の表記法が使用される。:

Range of values:	< min value > .. < max value >:	Example: 5 .. 20.
List of values:	< value1 >, < value2 >, ..., < valueN >:	Example: 2, 4, 6, 8, 9; Example: ‘1101'B, ‘1011'B, ‘1111'B; Example: ‘0A'H, ‘34'H, ‘2F'H.
List of named values:	< name1 >(< val1 >), < name2 >(< val2 >), ..., < nameN >(< valN >:	Example: reject(1), accept(2).
Length:	size (< min size > .. < max size >):	Example: size (1 .. 8).

Values supported column

values supported column は実装供給者に記述されるべきである。この column 中では,実装によってサポートされる値や値の範囲が示されるべきである。

References to items

PICS 様式の中の各々のアイテムの回答は (support column の中の回答) ユニークなリファレンスが例えば条件式表式のなかに存在または使用される。それらは表の識別名として定義され、表のアイテム番号の後に続く斜線文字「/」の後に続く。もし表の中にサポートする column が 1 つ以上ある時は、columns は個々に(a,b 等)の文字によって識別される。

例 1：A.5/4 は付属資料 A の表 5 のアイテム 4 に対応する。

例 2：A.6/3b は付属資料 A の表 6 のアイテム 3 の（第二サポート column の中）に対応する。

Prerequisite line

Prerequisite line は次の形式をとる: Prerequisite: < predicate >.

項の後や表のヘッダーの前の prerequisite line は、項全体もしくは 述語が FALSE で、表への要求が完全ではない表全体を示す。.

E.1.3 PICS 様式を完成させるための手引き

実装の供給者は PICS 様式の提供された各々の空欄 を完成させるべきである。特に、E.1.2 項に記述される表記法を用いてサポートもしくは supported column 欄が提供されていたら明確な回答が入力されるべきである。

もし必要ならば、供給者は表の最終スペースにコメントを追加、あるいは別紙を提供してもよい。

より詳細な説明書は PICS 様式の違う項目の最初に与えられている。

E.2 実装の認証

テストにおける実装の認証(ITU)と(the System Under Test (SUT))に属するシステムは、ヴァージョンナンバーと configuration options に関する可能な限り詳細な情報を提供するために記述されるべきである。

製品供給情報とクライアント情報はそれらが異なる場合は両方記入されるべきである。

PICS によって提供された情報について回答出来る人は contact person として名前を提供するべきである。

E.2.1 日付の記述

.....

E.2.2 Implementation Under Test (IUT)の認証

IUT name:

.....

.....

IUT version:

.....

E.2.3 System Under Test (SUT) の認証

SUT name:

.....

.....

Hardware configuration:

.....

.....

.....

Operating system:

.....

E.2.4 Product supplier（製品供給者）

名前(Name):

.....

住所(Address):

.....

.....

.....

電話番号(Telephone number):

.....

ファクシミリ番号(Facsimile number):

.....

E-mail アドレス(E-mail address):

.....

付加情報(Additional information):

.....

.....

.....

E.2.5 Client (製品供給者ではない人)

名前(Name):

.....

住所(Address):

.....

.....

.....

電話番号(Telephone number):

.....

ファクシミリ番号(Facsimile number):

.....

E-mail アドレス(E-mail address):

.....

付加情報(Additional information):

.....

.....

E.2.6 PICS **連絡担当者**

(連絡を取りたい人、PICS の内容に関して質問がある人)

名前(Name):

.....

電話番号(Telephone number):

.....

ファクシミリ番号(Facsimile number):

.....

E-mail アドレス(E-mail address):

.....

付加情報(Additional Information)

.....

.....

Annex F (Informative):参考文献

ITU-T Recommendation X.680: "Information technology - Abstract Syntax Notation One (ASN.1): Specification of basic notation".

ITU-T Recommendation Z.100: "Specification and description language (SDL)".

ITU-T Recommendation Z.120: "Message Sequence Chart (MSC)".

ETSI TS 101 884: "Telecommunications and Internet Protocol Harmonization Over Networks (TIPHON) Release 3; Technology Mapping; Implementation of TIPHON architecture using SIP".

履歴

Document history		
V1.1.1	May 2002	Publication