Draft EN 301 138 V1.1.1 (1998-01)

*European Standard (Telecommunications series)*

# Cordless Terminal Mobility (CTM); Network mobility management protocol; Phase 1 stage 3 specification

***European Telecommunications Standards Institute***

Reference

DEN/SPS-02028 (ajc00ico.PDF)

Keywords

CTM, DECT, mobility

***ETSI Secretariat***

Postal address

F-06921 Sophia Antipolis Cedex - FRANCE

Office address

650 Route des Lucioles - Sophia Antipolis
Valbonne - FRANCE
Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16
Siret N° 348 623 562 00017 - NAF 742 C
Association à but non lucratif enregistrée à la
Sous-Préfecture de Grasse (06) N° 7803/88

X.400

c= fr; a=atlas; p=etsi; s=secretariat

Internet

secretariat@etsi.fr
http://www.etsi.fr

## Contents

## Intellectual Property Rights

## Foreword

This European Standard (Telecommunications series) has been produced by ETSI Technical Committee Signalling Protocols and Switching (SPS), and is now submitted for the Public Enquiry phase of the ETSI standards Two-step Approval Procedure (TAP).

| **Proposed national transposition dates** | |
|---|---|
| Date of latest announcement of this EN (doa): | 3 months after ETSI publication |
| Date of latest publication of new National Standard or endorsement of this EN (dop/e): | 6 months after doa |
| Date of withdrawal of any conflicting National Standard (dow): | 6 months after doa |

## Introduction

This EN describes the application of core Intelligent Network Application Protocol (INAP) (see EN 301 140-1 [2] and ITU-T Recommendation Q.1228 [8]) for CTM phase 1 on the internetworking interfaces. Clause 4 describes the architecture of Cordless Terminal Mobility (CTM) based on Intelligent Network (IN) Capability Set 2 (CS2) architecture. In clause 5, the CTM information model is described. Clause 6 provides the addressing pattern used in CTM. In clause 7, the network interfaces and the application contexts are described. Clause 8 contains the description of the procedures used in the Service Data Function (SDF) and the Service Control Function (SCF). Clause 10 is the normative annex including the Abstract Syntax Notation No. 1 (ASN.1) module for CTM.

# 1 Scope

The present document specifies the application of core Intelligent Network Application Protocol (INAP) for the CTM phase 1 service and describes the internetworking interfaces.

The present document is applicable to CTM phase 1 as defined in DEN/NA-020039 [3]. It is mainly based on the information in core Intelligent Network Application Protocol (INAP), ITU-T Recommendation Q.1228 [8]. The CTM service relies on the IN architecture as described in ITU-T Recommendation Q.1224 [7].

# 2 References

References may be made to:

a) specific versions of publications (identified by date of publication, edition number, version number, etc.), in which case, subsequent revisions to the referenced document do not apply; or

b) all versions up to and including the identified version (identified by "up to and including" before the version identity); or

c) all versions subsequent to and including the identified version (identified by "onwards" following the version identity); or

d) publications without mention of a specific version, in which case the latest version applies.

A non-specific reference to an ETS shall also be taken to refer to later versions published as an EN with the same number.

## 2.1 Normative references

[1] ETS 300 287-1: "Integrated Services Digital Network (ISDN); Signalling System No. 7; Transaction Capabilities Application Part (TCAP) version 22; Part 1: Protocol specification [ITU-T Recommendations Q.771 to Q.775 (1993), modified] ".

[2] EN 301 140-1 (V1.1): "Intelligent Network (IN); Intelligent Network Capability Set 2 (CS2); Intelligent Network Application Protocol (INAP); Part 1: Protocol specification".

[3] DEN/NA-020039: "Network Aspects (NA), Cordless Terminal Mobility (CTM), Service Description Phase 1".

NOTE 1: Not available at the time of release of the present document for PE.

[4] ETR 060: "Signalling Protocols and Switching (SPS); Guidelines for using Abstract Syntax Notation One (ASN.1) in telecommunication application protocols".

[5] ITU-T Recommendation X.501 (1997) | ISO/IEC 9594-2 (1997): "Information technology - Open Systems Interconnection - The directory: Models".

[6] ITU-T Recommendation Q.1223 (1997): "General Recommendations on Telephone and Switching, Intelligent Network, Global Functional Plane for Intelligent Network Capability Set 2".

NOTE 1: Not yet published at the time of release of the present document for PE.

[7] ITU-T Recommendation Q.1224 (1997): "General Recommendations on Telephone and Switching, Intelligent Network, Distributed Functional Plane for Intelligent Network Capability Set 2".

NOTE 1: Not yet published at the time of release of the present document for PE.

[8] ITU-T Recommendation Q.1228 (1997): "General Recommendations on Telephone and Switching, Intelligent Network, Interface recommendations for Intelligent Network Capability Set 2".

NOTE 1: Not yet published at the time of release of the present document for PE.

[9] ITU-T Recommendation X.520 (1997) | ISO/IEC 9594-6 (1997): "Information technology - Open Systems Interconnection - The directory: Selected attribute types".

[10] ITU-T Recommendation X.521 (1997) | ISO/IEC 9594-7 (1997): " Information technology - Open Systems Interconnection - The directory: Selected object classes".

[11] ETR 090: "ETSI object identifier tree; Common domain; Intelligent Network (IN) domain".

[12] ETS 300 175-6: "Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI); Part 6: Identities and addressing".

[13] ITU-T Recommendation E.164: "The international public telecommunication numbering plan".

[14] ETS 300 009-1: "Integrated Services Digital Network (ISDN); Signalling System No.7; Signalling Connection Control Part (SCCP) (connectionless and connection-oriented class 2) to support international interconnection; Part 1: Protocol specification [ITU-T Recommendations Q.711 to Q.714 and Q.716 (1993), modified]".

[15] EN 301 144-1: "Integrated Services Digital Network (ISDN); Digital Subscriber Signalling System No. one (DSS1) protocol and Signalling System No.7 protocol; Signalling application for the mobility management service on the alpha interface; Part 1: Protocol specification ".

[16] ITU-T Recommendation E.212: "Identification plan for land mobile stations".

[17] ITU-T Recommendation E.214: "Structure of the land mobile global title for the signalling connection control part (SCCP)".

## 2.2 Informative references

[15] EG 201 096-1: "Intelligent Network (IN); Cordless Terminal Mobility (CTM); IN architecture and functionality for the support of CTM; Part 1: CTM phase 1 for single public network case".

[16] EG 201 096-2: "Intelligent Network (IN); Cordless Terminal Mobility (CTM); IN architecture and functionality for the support of CTM; Part 2: CTM interworking between public Intelligent Networks".

[17] EG 201 096-3: "Intelligent Network (IN); Cordless Terminal Mobility (CTM); IN architecture and functionality for the support of CTM; Part 3: CTM interworking between private networks and public Intelligent Network".

# 3 Abbreviations

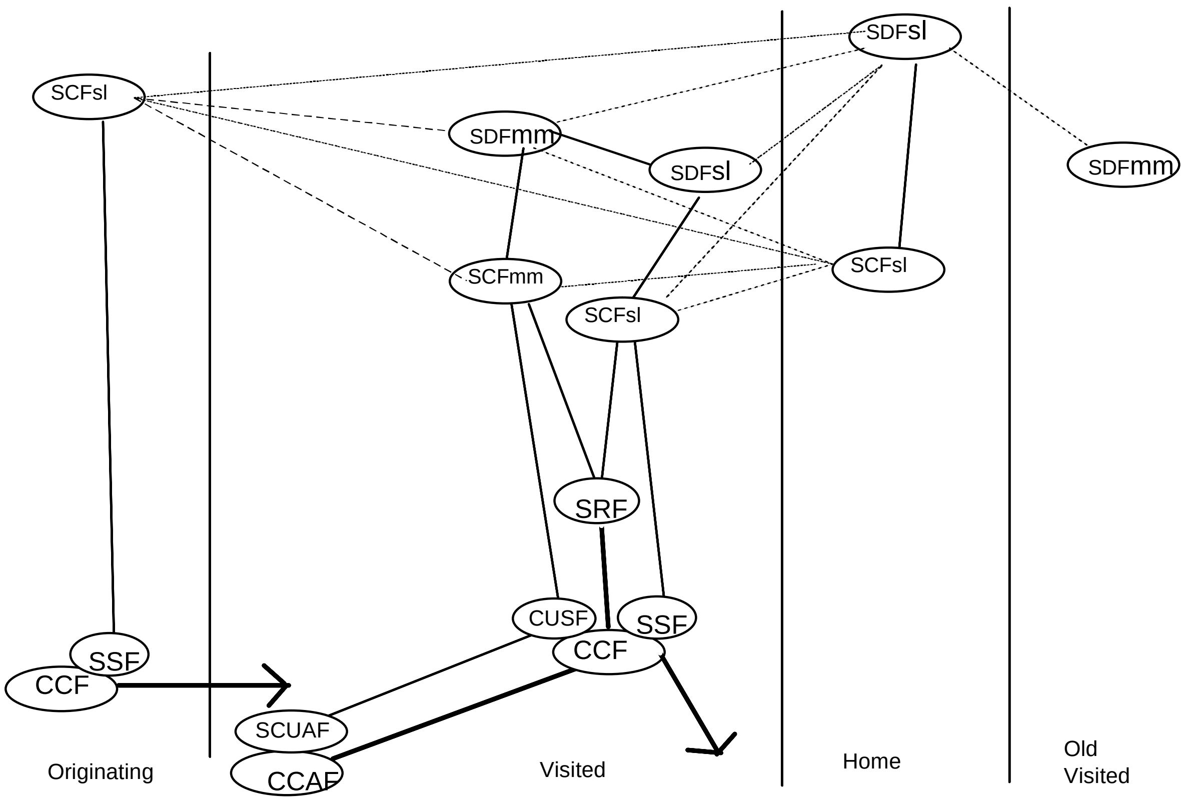
For the purposes of the present document, the following abbreviations apply:

| | |
|---|---|
| ASN.1 | Abstract Syntax Notation No. 1 |
| CLIR | Calling Line Identification Restriction |
| COLR | COnnected Line identification Restriction |
| CS2 | Capability Set 2 |
| CT2 | Second generation of Cordless Technology |
| CTM | Cordless Terminal Mobility |
| DECT | Digital Enhanced Cordless Telecommunications |
| DIT | Directory Information Tree |
| DN | Distinguished Name |
| DSA | Directory System Agent |
| DUA | Directory User Agent |
| FE | Functional Entity |
| FT | Fixed Termination |
| GT | Global Titles |
| GTAI | |

| | |
|---|---|
| IN | Intelligent Network |
| INAP | Intelligent Network Application Protocol |
| IP | Intelligent Peripheral |
| IPUI | International Portable User Identity |
| KS | DECT Session Key |
| PT | Portable radio Termination |
| RN | Re-routeing Number |
| RS | a value used to establish session keys in DECT |
| SCCP | Signalling Connection Control Part |
| SCF | Service Control Function |
| SDF | Service Data Function |
| SLP | Service Logic Program |
| SPC | Signalling Point Code |
| SRF | Specialized Resource Function |
| SSF | Service Switching Function |
| SSP | Service Switching Point |
| ST | |

## 4 Reference architecture

Figure 1 describes the reference architecture which is assumed by this EN for protocol specification purposes. Although this EN is primarily intended to support inter-network configurations, the procedures are specified in such a way that they can also be used in an intra-network configuration.



Legend

| | |
|---|---|
| CCAF | Call Control Agent Function |
| CCF | Call Control Function |
| CUSF | Call Unrelated Service Function |
| SCEF | Service Creation Environment Function |
| SCF | Service Control Function |
| SCUAF | Service Control User AgentFunction |
| SDF | Service Data Function |
| SSF | Service Switching Function |

| | |
|---|---|
| ──── | IN Service Control |
| ━━━━ | Bearer Connection Control |
| ........ | Internetworking Relationship |
| - - - - - | Internetworking Relationship based on mutual operators aggreement |

**Figure 1**

## 5 CTM information modelling

### 5.1 General

The stage 2 description defines two categories of Service Data Function (SDF): SDFsl and SDFmm. This distinction has no counterpart in the IN distributed functional plane. However SDFsl should be understood as a shorthand notation to refer to an SDF which contains CTM user profiles, while SDFmm should be understood as a shorthand notation to refer to an SDF which contains information about roaming users temporarily stored by a Service Control Function (SCF) as the result of location management procedures.

Two categories of SCF, SCFsl and SCFmm, are also defined. This distinction has no counterpart in the IN distributed functional plane. However SCFsl and SCFmm should be understood as a shorthand notation to refer to an SCF running a particular service logic. SCFsl refers to an SCF running a service logic dedicated to call handling and user profile management, while SCFmm refers to an SCF running a service logic dedicated to location management and mobility authentication.

When considering the distributed functional plane (see figure 2), an SDF may play the role of an SDFsl, an SDFmm or both. Similarly an SCF may play the role of an SCFsl, an SCFmm or both roles. In the physical plane each SDF may be implemented as a stand-alone equipment (SDP) or be collocated with an SCF. There is no restriction on the category of SDF and SCF which may be collocated.

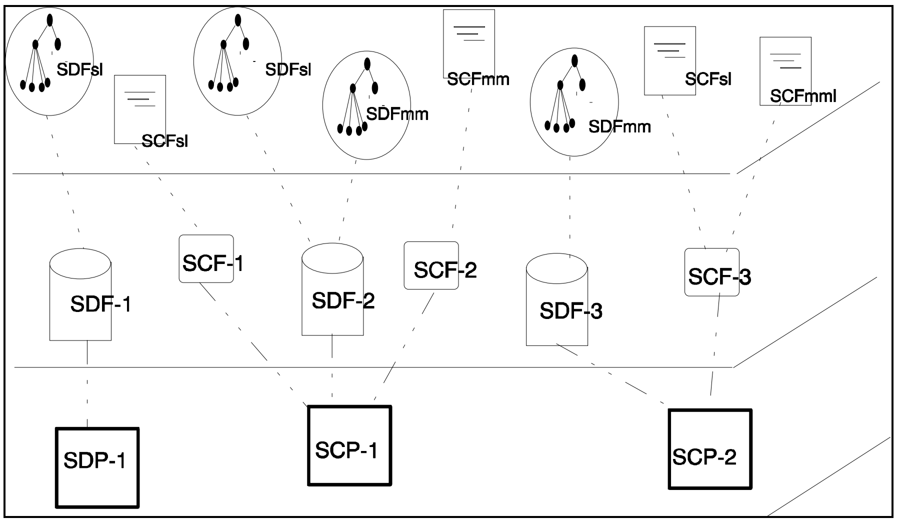


**Figure 2: Distributed functional plane**

## 5.2 Information base

The information held in the SDFs co-operating for the provision of the CTM service form the CTM information base. The CTM information base is a set of entries organized according to the CTM schema described in this subclause. The CTM schema conforms to the directory information model defined in ITU-T Recommendation X.501 [5].

The CTM schema identifies several object classes and attributes needed to fulfill the CTM service. It also specifies how entries are arranged into a tree and how each entry can be named. It is to be noted that this arrangement is only a way to get an external view of the data (dictionary of data), nothing is preconceived on the real database (structure of a record, link between them). It should also be noted that X.500 Recommendations do not place any constraints on the implementation. There exist X.500 products based on relational databases, object-oriented databases, tables, trees, etc.

It is simpler to use an X.500 product to run the data model, but it is not mandatory. A state machine which implements the required ASN.1 operations of the CTM procedures can also be used.

Figure 3 gives the inheritance relationships between the different object classes. All the object classes are subclasses of **top** which is an abstract class from which all the other classes are subclasses.

Each class used in the specification of the CTM schema is described in more detail in the following subclauses, except for those defined in the X.500 series of Recommendations (**top, alias, organization, organizationalUnit, country, person, applicationEntity**).

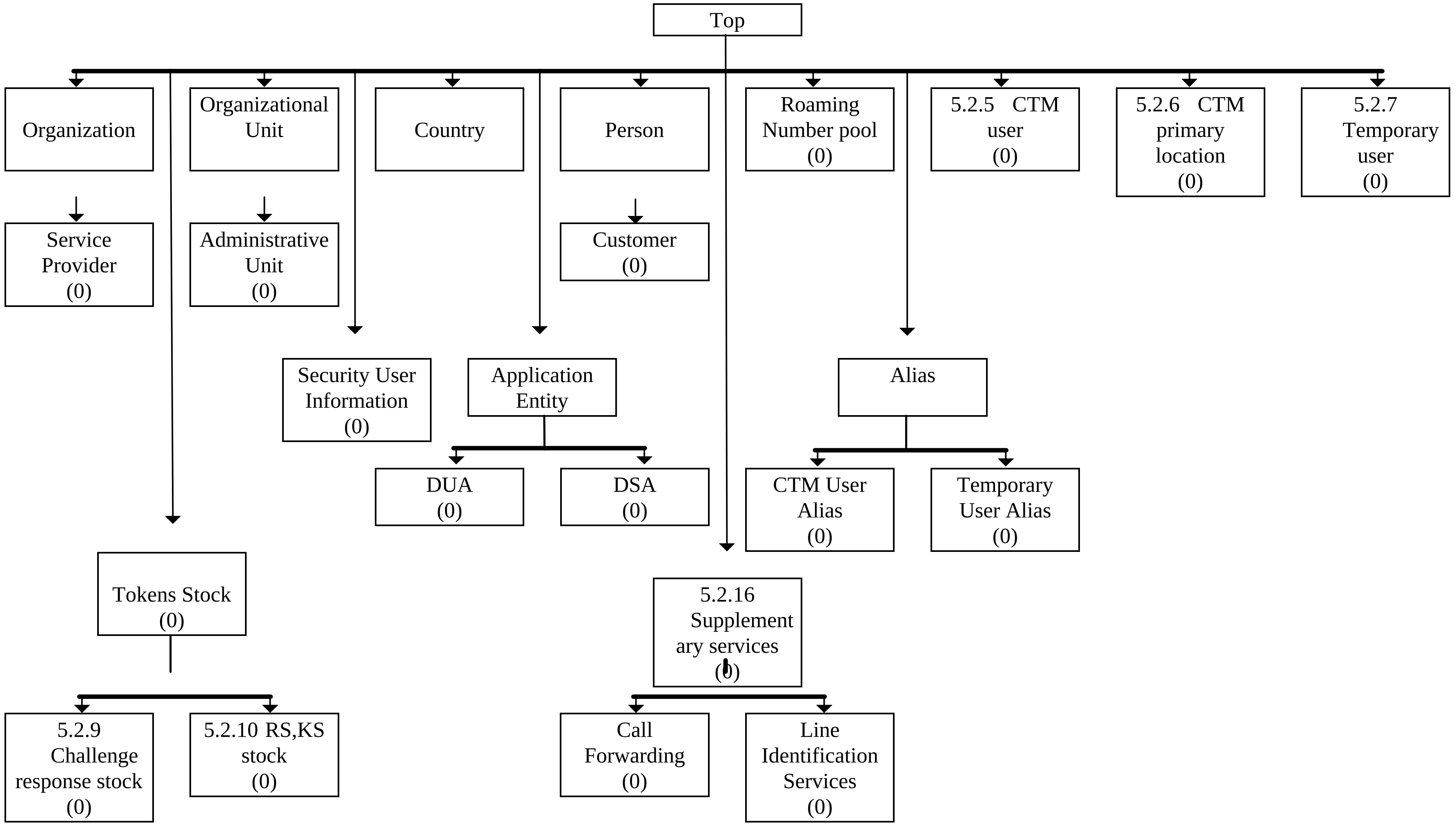


**Figure 3: Inheritance for the object classes**

### 5.2.1 Service provider

This OBJECT-CLASS defines a CTM service provider. It is a subclass of the **organization** OBJECT-CLASS. Besides the attribute of its superclass, the **serviceProvider** object class gives the code of the service provider (used in numbering) uniquely identifying the service provider and the list of service providers with whom the service provider has agreements (each service provider is identified by his code). The value of the service provider code is a Mobile Network Code (MNC) of the ITU-T Recommendation E.212 [16] numbering plan.

```
serviceProvider  OBJECT-CLASS ::= {
    SUBCLASS OF     {organization}
    MUST CONTAIN    {serviceProviderCode}
    MAY CONTAIN {partnersList}
    ID          id-oc-serviceProvider}
```

The **serviceProviderCode** uniquely identifies a provider.

```
serviceProviderCode  ATTRIBUTE ::= {
    WITH SYNTAX         NumericString (SIZE (1..ub-serviceProviderCode))
                            -- MNC in E.212
    EQUALITY MATCHING RULE  numericStringMatch
    SINGLE VALUE        TRUE
    ID              id-at-serviceProviderCode}
```

The max length of a **serviceProviderCode** is 3.

```
ub-serviceProviderCode  INTEGER ::= 3
```

The **partnersList** lists the providers which got an agreement with the provider indicated by the **serviceProviderCode**.

```
partnersList  ATTRIBUTE ::= {
    WITH SYNTAX         DistinguishedName
    EQUALITY MATCHING RULE  distinguishedNameMatch
    ID              id-at-partnersList}
```

### 5.2.2 Administrative unit

This object class is used to gather entries representing users that are administrated as a single logical entity. The main purpose of this object class is to be able to split the users over several DSAs without having to manage a larege number of knowledge references. It contains the attributes of its superclass **organizationalUnit** plus three attributes used for naming.

- The first one is used for naming administrative units which are the superior entry of entries of class **ctmUser**.
- the second one is used for naming administrative units which are the superior entry of entries of class **ctmUserAlias**.
- the third one is used for naming administrative units which are the superior entry of entries of class **temporaryUser**.

The distribution of the users over several administrative units is a implementation matter, however it is advisable to distribute users belonging to a same unit over a unique SDF and to avoid having entries representing CTM users and temporary users in the same administrative unit. But several administrative units could co-exist in the same SDF.

```
administrativeUnit  OBJECT-CLASS ::= {
    SUBCLASS OF {organizationalUnit}
    MAY CONTAIN -- A single attribute from the following three
            {ctmNumberPrefix|
            ctmIdentityPrefix |
            temporaryIdentityPrefix}
    ID      id-oc-administrativeUnit}

ctmNumberPrefix  ATTRIBUTE ::= {
    WITH SYNTAX         NumericString (SIZE (1..5))     -- E.164 prefix
    EQUALITY MATCHING RULE  numericStringMatch
    SINGLE VALUE        TRUE
    ID              id-at-ctmNumberPrefix}

ctmIdentityPrefix   ATTRIBUTE ::= {
    WITH SYNTAX         NumericString (SIZE (1..4))     -- E.212 prefix
    EQUALITY MATCHING RULE  numericStringMatch
    SINGLE VALUE        TRUE
    ID              id-at-ctmIdentityPrefix}
```

```
temporaryIdentityPrefix ATTRIBUTE ::= {
    WITH SYNTAX             NumericString (SIZE (1..4))
    EQUALITY MATCHING RULE      numericStringMatch
    SINGLE VALUE            TRUE
    ID                  id-at-temporaryIdentityPrefix}
```

### 5.2.3 Customer

This object class contains the information related to the organizational person that has subscribed to the service. This object class is not subject to standardization since it is not used in the provision of the service. For information only this object class could contain the list of users attached to the subscription and information provided at subscription time by the subscriber (e.g. address, account number, ...).

```
customer  OBJECT-CLASS ::= {
    SUBCLASS OF {person}
    MAY CONTAIN {member}
    ID          id-oc-customer}
```

### 5.2.4 Roaming number pool

This object class is used to represent information related to the roaming number procedures.

Roaming number related information is stored in some entries of class **administrativeUnit**. The **objectClass** attribute of such entries contains the OID value of the **roamingNumberPool** auxiliary class in addition to those of their structural object class. These entries are named using the **adminUnitNameform3** (see subclause 5.3.2).

```
roamingNumberPool OBJECT-CLASS ::={
KIND            auxiliary
MUST CONTAIN    {assignmentTable}
MAY CONTAIN         {maxtime|
                randomAssigned}
ID          id-oc-roamingNumberPool}
```

**assignmentTable** is a multi valued attribute which supports two contexts:

- **temporalcontext**;
- **assignmentContext**.

```
assignmentTable ATTRIBUTE ::= {
WITH SYNTAX     NumericString (SIZE(1..ub-international-isdn-number))
ID          id-at-assignmentTable}
```

The **maxtime** attribute is a measure of the time the reservation is maintained. It is used to create a suitable temporal context value to be associated with a selected value.

```
maxtime ATTRIBUTE ::= {
WITH SYNTAX     INTEGER
SINGLE VALUE    TRUE
ID      id-at-maxtime}
```

The **randomAssigned** attribute indicates that the values of the assignmentTable attribute have to be selected at random.

```
randomAssigned ATTRIBUTE ::= {
WITH SYNTAX BOOLEAN
SINGLE VALUE    TRUE
ID      id-at-randomAssigned}
```

### 5.2.5 CTM user

This object class gives the service information attached to one of the users in a subscription related to the CTM service. This information may differ from one user to another in the same subscription. It contains the user credit, the CTM Identity uniquely identifying the user, some service features that the user can use (i.e. included in the user's subscription).

```
ctmUser  OBJECT-CLASS ::= {
    MUST CONTAIN    {   ctmIdentity}
    MAY CONTAIN {   ctmNumber|
                    description|
                    radioSpecificEnvelop|
                    homeSpecificServices |
                    seeAlso|
                    serviceFeatures|
                    dectSessionSecurityInfo}
    ID          id-oc-ctmUser}
```

This attribute identifies the user by its radio identity.

```
ctmIdentity ATTRIBUTE ::= {
    WITH SYNTAX         NumericString (SIZE (1..ub-international-isdn-number))
    EQUALITY MATCHING RULE      numericStringMatch
    SUBSTRINGS MATCHING RULE    reversePrefixMatch
    SINGLE VALUE            TRUE
    ID                  id-at-ctmIdentity}
```

The following attribute provides the CTM number of the user. It could be used for a Calling Line identification.

```
ctmNumber  ATTRIBUTE ::= {
    WITH SYNTAX         NumericString (SIZE (1..ub-international-isdn-number))
    EQUALITY MATCHING RULE      numericStringMatch
    SUBSTRINGS MATCHING RULE    reversePrefixMatch
    SINGLE VALUE            TRUE
    ID                  id-at-ctmNumber}
```

The following attribute indicates to the SCF if the service to offer to the user is specific and which features are specific. It also contains the necessary information to determine the address of the home SCP in charge of this feature.

```
homeSpecificServices        ATTRIBUTE ::={
WITH SYNTAX         HomeSpecificMark
ID              id-at-homeSpecificServices}

HomeSpecificMark ::= SEQUENCE {
    serviceIdentifier       ServiceFeaturesIdentifier,
    scpAddress      EntityAddress OPTIONAL,
    ...}
```

This attribute permits the determination, for management purpose, of the privileges of a user on his profile.

```
serviceFeatures  ATTRIBUTE ::= {
    WITH SYNTAX     ServiceFeatures
    SINGLE VALUE    TRUE
    ID          id-at-serviceFeatures}

ServiceFeatures ::= BIT STRING {
    profileModification (0),
    profileInterrogation (1)}

ServiceFeaturesIdentifier   ::= CHOICE {
    local   INTEGER(0..127),
    global  OBJECT IDENTIFIER}
```

This attribute contains one or more couples (RS, KS) of the CTM user using a DECT terminal.

```
dectSessionSecurityInfo     ATTRIBUTE ::= {
    WITH SYNTAX     DectSessionSecurityInfo
    ID          id-at-dectSessionSecurityInfo}

DectSessionSecurityInfo ::= SEQUENCE {
    rs  [0] BIT STRING,
    ks  [1] BIT STRING}
```

**description** and **seeAlso** are predefined X.520 attributes [9], they permit the provision of complementary information on the CTM user (text or new attributes).

## 5.2.6 CTM primary location

This object class contains location information of the CTM user. It is used to facilitate the access to this information in case of multiprofile terminals. For each CTM user, an entry of the **ctmPrimaryLocation** object class needs to be created, subordinate to each entry of class **ctmUser**. This subordination is made to ease the evolution toward multi-profile terminals.

NOTE: Several CTM identities could be stored on the same terminal and the location information shared.

```
ctmPrimaryLocation OBJECT-CLASS ::= {
MUST CONTAIN    {locationCommonIdentifier}
MAY CONTAIN         -- at least one of the three following attributes
                {scpLocation |  -- could be used to address the SCFmm visited
                sdpLocation |   -- could be used to get a roaming number
                routingAddress} -- could be used to route the call in white ISUP
ID          id-oc-ctmPrimaryLocation}
```

This attribute contains a fixed value to name the entry which contains the location information.

```
locationCommonIdentifier        ATTRIBUTE ::=
    WITH SYNTAX             NumericString(SIZE(1)) -- to be fixed at value "1"
    EQUALITY MATCHING RULE      numericStringMatch
    SINGLE VALUE            TRUE
    ID                  id-at-locationCommonIdentifier}
```

The following attribute permits storage of the SCFmm address which controls the area where the CTM user is roaming.

```
scpLocation  ATTRIBUTE ::= {
    WITH SYNTAX     EntityAddress
    EQUALITY MATCHING RULE          octetStringMatch
    ID                      id-at-scpLocation}
```

The following attribute permits storage of the SDFmm address; it could be used to know in which SDFmm the user is previously roaming to delete the old data, but the direct use of the alias is more efficient.

```
sdpLocation  ATTRIBUTE ::= {
    WITH SYNTAX     EntityAddress
    EQUALITY MATCHING RULE          octetStringMatch
    ID                      id-at-sdpLocation}
```

The following attribute contains a routeing address where the CTM user is roaming. The choice of the value is specific to the visited domain: for example, it could be the first Fixed Termination (FT) address used in this area or a "triggering digits" address. Several roaming users could get the same value.

```
routingAddress  ATTRIBUTE ::= {
    WITH SYNTAX     NumericString (SIZE (1..ub-international-isdn-number))
    EQUALITY MATCHING RULE      numericStringMatch
    SUBSTRINGS MATCHING RULE    reversePrefixMatch
    ID                  id-at-routingAddress}
```

### 5.2.7 Temporary user

This object class contains the temporary information related to a roaming user. It is intended to be maintained close to the user's location.

```
temporaryUser  OBJECT-CLASS ::= {
    MUST CONTAIN    {localUserIdentifier|
                ctmIdentityList |
                temporaryLocation}
    MAY CONTAIN {
                radioSpecificEnvelop|
                dectSessionSecurityInfo|
                description|
                seeAlso}
    ID          id-oc-temporaryUser}
```

The following attribute contains a local user's identifier that is used as naming attribute of the entry.

NOTE: The value **localUserIdentifier** attribute is chosen during the first location procedure by the visited network. It may be based on one of the CTM identity of the user.

```
localUserIdentifier  ATTRIBUTE ::= {
    WITH SYNTAX             NumericString(SIZE(1..ub-international-isdn-number))
    EQUALITY MATCHING RULE      numericStringMatch
    SUBSTRINGS MATCHING RULE    reversePrefixMatch
    SINGLE VALUE            TRUE
    ID                  id-at-localUserIdentifier}
```

The following attribute contains the CTM identities attached to the **localUseridentifier**:

```
ctmIdentityList  ATTRIBUTE ::= {    -- this attribute is multivalued to take into account
                --  multi profile DECT terminals
    WITH SYNTAX         NumericString (SIZE (1..ub-international-isdn-number))
    EQUALITY MATCHING RULE      numericStringMatch
    SUBSTRINGS MATCHING RULE    reversePrefixMatch
    ID                  id-at-ctmIdentityList}
```

The following attribute contains a more accurate terminal location (e.g. the current FT address):

```
temporaryLocation ATTRIBUTE ::= {
    WITH SYNTAX         NumericString  (SIZE (1..ub-international-isdn-number))
    EQUALITY MATCHING RULE  numericStringMatch
    SUBSTRINGS MATCHING RULE    reversePrefixMatch
    SINGLE VALUE                        TRUE
    ID                                          id-at-temporaryLocation}
```

The following attribute contains a specific value to be provided at the radio interface to reconstruct the full radio identity of a terminal. This value is only carried in the fixed network, no analysis is done. In the case of DECT, the SCF could store/retrieve the information used to determine the IPUI from the CTMid.

```
radioSpecificEnvelop ATTRIBUTE ::={
WITH SYNTAX     BIT STRING
SINGLE VALUE    TRUE
ID      id-at-radioSpecificEnvelop}
```

## 5.2.8 Tokens stock

This object class is used to represent a set of information which is common to all disposable tokens (stock identifier, source, size of the set). Disposable tokens could be, for example, CTM phase1 authentication tokens pairs, GSM triplets.

```
tokensStock OBJECT-CLASS ::={
KIND            abstract
MUST CONTAIN        {stockId}
MAY CONTAIN         {source|
                            sizeOfRestocking}
ID          id-oc-tokensStock}
```

**stockId** is a mono-valued attribute of type **DT-Code** that is used as naming attribute.

```
stockId  ATTRIBUTE ::= {
    WITH SYNTAX         DT-Code
    EQUALITY MATCHING RULE  objectIdentifierMatch
    SINGLE VALUE        TRUE
    ID              id-at-stockId}

DT-Code ::= OBJECT IDENTIFIER
```

For CTM phase 1, only two instances of child entries have to be used. Thus two values of DT-CODE are provided to name them. The first stock contains different RS, KS values, the second would be a challenge response stock.

```
-- Disposable Stock id
id-dt-dect-stage1   DT-CODE ::= {id-dt authentication(1) dect(1) stage1(1)}
id-dt-challengeResponse     DT-CODE ::= {id-dt authentication(1) common(2)}
```

**source** is a mono-valued attribute of type choice.

```
source ATTRIBUTE ::={
WITH SYNTAX     SourceType
SINGLE VALUE    TRUE
ID      id-at-source}

SourceType ::=DistinguishedName
```

In the visited network, the **source** attribute will be used to store the distinguished name (DN) of the entry of class derived from **stockId**. In the home network, the attribute will contain the DN of an entry of class **securityUserInfo**; the **token** is generated using the method defined in the **fillMethodIdentifier** field of this entry of class **securityUserInfo**.

**sizeOfRestocking** is a mono-valued attribute which indicates how many tokens have to be requested or computed when the **tokens** attribute is empty.

```
sizeOfRestocking ATTRIBUTE ::= {
WITH SYNTAX             INTEGER
ORDERING MATCHING RULE      integerOrderingMatch
SINGLE VALUE                TRUE
ID                  id-at-sizeOfRestocking }
```

## 5.2.9 Challenge response stock

This object class is used to represent a set of challenge-response-cipherkey tokens. The challenge reponse parts would be [RAND,RES] for Second generation of Cordless Technology (CT2) and [RAND_RS,RES] for DECT. It inherits

**stockId**, **source** and **sizeOfRestocking** from the **tokenStock** object class. For each CTM user, an entry of the **ChallengeResponseStock** object class may be created, subordinate to each entry of class **ctmUser** or of class **TemporaryUser**.

```
challengeResponseStock OBJECT-CLASS ::= {
    SUBCLASS OF     {tokenStock}
    MUST CONTAIN    {challengeResponse}
    ID          id-oc-challengeResponseStock}
```

The following attribute could contain the precomputed set of [RAND_RS,RES,DCK] values of a CTM user to authenticate him and to cipher the radio link on DECT interface.

```
challengeResponse ATTRIBUTE ::= {
WITH SYNTAX         ThreePartsMessage
ID          id-at-challengeResponse}
```

### 5.2.10 RS,KS stock

This object class is used to represent a set of precomputed (RS,KS) in the case of authentication of a public user in a private network or in another public network. The challenge reponse pairs would be [RAND,RES] for CT2 and [RAND_RS,RES] for DECT. It inherits **stockId**, **source** and **sizeOfRestocking** from the **tokenStock** object class. For each CTM user, an entry of the **ChallengeResponseStock** object class may be created, subordinate to each entry of class **ctmUser** or of class **TemporaryUser**.

```
rsKsStock OBJECT-CLASS ::= {
    SUBCLASS OF     {tokenStock}
    MUST CONTAIN    {rsKs}
    ID          id-oc-rsKsStock}
```

The following attribute could contain the precomputed set of [RAND_RS,RES] values of a CTM user to authenticate him and the cipher key to be used.

```
rsKs ATTRIBUTE ::= {
WITH SYNTAX         TwoPartsMessage
ID          id-at-rsKs}
```

### 5.2.11 Security user information

The following object class could be used to store the necessary information about the CTM user security (parameters and policy). For each CTM user, an entry of the **securityUserInfo** object class may be created, subordinate to each entry of class **ctmUser**.

```
securityUserInfo OBJECT-CLASS ::={
MUST CONTAIN    {securityFacilityId|
                secretKey|
                identifierList}
MAY CONTAIN         {bindLevelIfOK|
                currentList|
                failureCounter|
                maxAttempts}
ID          id-oc-securityUserInfo }
```

**securityFacilityId** is an attribute to name the verification.

```
securityFacilityId ATTRIBUTE ::= {
    WITH SYNTAX             SF-CODE
    EQUALITY MATCHING RULE      objectIdentifierMatch
    SINGLE VALUE            TRUE
    ID                  id-at-securityFacilityId}

SF-CODE ::= OBJECT IDENTIFIER
```

The securityFacilityId could take different values for CTM phase 1:

- id-sf-pwd for management access to the database by password;
- id-sf-challengeResponse for standard access based on one pass challenge response authentication;
- id-sf-onAirSubscription to authenticate the access during on air subscription (the entry contains the same **identifierList** that the previous entry but **secretKey** is different.

```
-- Security Facility id
id-sf-pwd SF-CODE ::= {id-sf pwd(1)}
id-sf-challengeResponse SF-CODE ::=  {id-sf common (2)}
id-sf-onAirSubscription SF-CODE ::=  {id-sf subscription(3)}
```

**secretKey** is an attribute which contains the secret key (to be used by the cryptographic algorithm) of the user.

```
secretKey  ATTRIBUTE ::= {
    WITH SYNTAX     BIT STRING(SIZE(lb-secretKey..ub-secretKey))
    SINGLE VALUE    TRUE
    ID          id-at-secretKey}

lb-secretKey  INTEGER ::= 32
ub-secretKey  INTEGER ::= 128}
```

**identifierList** is an attribute which could contain four identifiers:

- **conformMethodIdentifier** identifies the method used to verify that some parts of the input message conform to some criteria such as size, value matching with an attribute, greater than a counter, included in a time window;

NOTE: It is common in security (e.g. UPT authentication by DTMF, Race Project Sesame) for the verifier to draw the challenge value to check that it is not a replay of a previous value. Two mechanisms can be used: a counter, a time based random.

- **fillMethodIdentifier** to identifies the method use to fill the input message (first part of a **twoPartMessage** or **ThreePartMessage**);

- **oneToOneAlgorithm** (resp. **oneToTwoAlgorithm**) identifies the cryptographic algorithm with one output (resp. two output). For DECT, if KS is the secret key, IN is the input and OUT the output, it would be OUT=output1of (A12(RS_size_in_bits first bits of IN,A11 (RAND_size_in_bits last bits of IN,KS))). (resp. (OUT1,OUT2)= (A12(RS_size_in_bits first bits of IN,A11(RAND_size_in_bits last bits of IN,KS))).

```
identifierList ATTRIBUTE ::= {
WITH SYNTAX SEQUENCE{
conformMethodIdentifier [1] MethodIdentifier, -- e.g. time window check
fillMethodIdentifier        [2] MethodIdentifier-- e.g. generate a random of required size,
oneToOneAlgorithm       [3] AlgorithmIdentifier
                    -- e.g. A11 and A12, output RES from RS,RAND
oneToTwoAlgorithm       [4] AlgorithmIdentifier }
                    -- e.g DECT algorithm output RES,SDK from RS,RAND
SINGLE VALUE            TRUE
ID                              id-at-identifierList}

-- AlgorithmIdentifier could be imported from Rec. X.509
AlgorithmIdentifier ::= SEQUENCE {
algorithm   ALGORITHM.&id ({SupportedAlgorithms}),
parameters  ALGORITHM.&Type({SupportedAlgorithms}{@algorithm}) OPTIONAL}

MethodIdentifier ::= SEQUENCE {
methodid            METHOD.&id ({SupportedMethods}),
inputAttributes     METHOD.&InputAttributes ({SupportedMethods}{@method}) OPTIONAL,
specific-Input      METHOD.&SpecificInput ({SupportedMethods}{@method}) OPTIONAL}
```

bindLevelIfOK is a mono-valued attribute which contains an **AuthenticationLevel**. It is to be used by the bind operation with the argument of the appropriate abstract-syntax to determine the level of privileges granted to the user. When this attribute is absent and a bind operation is invoked, the bind operation returns an error.

```
bindLevelIfOK ATTRIBUTE ::= {
WITH SYNTAX     AuthenticationLevel
SINGLE VALUE    TRUE
ID      id-at-bindLevelIfOK}
```

For some security facilities, it is useful to count the number of failures and if necessary to lock the facility when a threshold is reached. The two following attributes are used to store this information:

```
failureCounter ATTRIBUTE ::= {
WITH SYNTAX             INTEGER
ORDERING MATCHING RULE      integerOrderingMatch
SINGLE VALUE                TRUE
ID                  id-at-failureCounter}

maxAttempts ATTRIBUTE ::= {
WITH SYNTAX             INTEGER
ORDERING MATCHING RULE      integerOrderingMatch
SINGLE VALUE                TRUE
ID                  id-at-maxAttempts}
```

To check the non-replay of a challenge RAND drawn in another domain, it is necessary to maintain a list of the random already used for the valid period indicated by RS. The currentList attribute contains a list of RAND already played for the current period of time.

```
currentList  ATTRIBUTE ::= {
    WITH SYNTAX         BIT STRING,
    EQUALITY MATCHING RULE  bitStringMatch
    ID              id-at-currentList}
```

### 5.2.12 DUA

This object class describes a sub-entity of the SCF function which provides access to the SDF. This object class is only used in the CTM service to define the access rights of the SCF to the data of the service (i.e. it is necessary to associate to each SCF an entry in the DIT).

```
dUA OBJECT-CLASS ::= {
    SUBCLASS OF {applicationEntity}
    ID      id-oc-dUA}
```

### 5.2.13 DSA

This object class models the SDF function. The description of this object class is given in ITU-T Recommendation X.521 [10].

### 5.2.14 CTM user alias

This object class also describes the user profile. It is used to have another naming path for the entry representing the user when only the CTM Number is available in the SCF.

```
ctmUserAlias  OBJECT-CLASS ::= {
    SUBCLASS OF     {alias}
    MUST CONTAIN    {ctmNumber}
    ID          id-oc-ctmUserAlias}

ctmNumber  ATTRIBUTE ::= {
    WITH SYNTAX         NumericString (SIZE (1..ub-international-isdn-number))
    EQUALITY MATCHING RULE      numericStringMatch
    SUBSTRINGS MATCHING RULE    reversePrefixMatch
    SINGLE VALUE            TRUE
    ID                  id-at-ctmNumber}

ub-international-isdn-number  INTEGER ::= 15
```

### 5.2.15 Temporary user alias

For each CTM user, an entry of the **temporaryUserAlias** object class may be created, subordinate to each entry of class **ctmUser**. This entry contains a pointer to the entry of class **temporaryUser** which contains the precise location of the user. It could be used for example to erase data in the old visited database.

```
temporaryUserAlias  OBJECT-CLASS ::= {
    SUBCLASS OF     {alias}
    MUST CONTAIN    {registrationIdentifier}
    ID          id-oc-temporaryUserAlias}

registrationIdentifier  ATTRIBUTE ::= {
    WITH SYNTAX             INTEGER
    EQUALITY MATCHING RULE      integerMatch
    SINGLE VALUE            TRUE
    ID                  id-at-registrationIdentifier}
```

### 5.2.16 Supplementary services

This object class is used to represent a set of information which is common to all supplementary services (supplementary service code, status, etc.).

```
supplementaryService OBJECT-CLASS ::= {
KIND            abstract
    MUST CONTAIN    {supplServId |
                supplServiceStatus}
    MAY CONTAIN {description |
                name}
    ID          id-oc-supplementaryService}

supplServId ATTRIBUTE ::= {
    WITH SYNTAX         SS-CODE
    EQUALITY MATCHING RULE  objectIdentifierMatch
    SINGLE VALUE        TRUE
    ID              id-at-supplServId}

supplServiceStatus  ATTRIBUTE ::= {
    WITH SYNTAX     SupplServiceStatus
    SINGLE VALUE    TRUE
    ID          id-at-supplServiceStatus}

SupplServiceStatus ::= BIT STRING {
    activated (0),
    registered (1),
    provisioned (2)}

SS-CODE ::= OBJECT IDENTIFIER
-- Supplementary Service id --
-- if  for isdn Supplementary Services oids are already defined in Q730 series
-- they will be used instead

id-ss-clip SS-CODE ::= {id-ss li(1) 3}
id-ss-clir SS-CODE ::= {id-ss li(1) 4}
id-ss-colp SS-CODE ::= {id-ss li(1) 5}
id-ss-colr SS-CODE ::= {id-ss li(1) 6}
id-ss-mcid SS-CODE ::= {id-ss li(1) 7}
id-ss-cfb SS-CODE ::= {id-ss cf (2) 2}
id-ss-cfnr SS-CODE ::= {id-ss cf(2) 3}
id-ss-cfu SS-CODE ::= {id-ss cf(2) 4}

id-ss-cfnrc SS-CODE ::= {id-ss cf(2) 5}
```

### 5.2.17 Call forwarding

This object class is used to represent information on the call forwarding services (CFNRc). This information includes the forwarded-to number and the type of notifications as well as the status of the supplementary service and its identifier inherited from the **SupplementaryService** object class.

```
callForwarding OBJECT-CLASS ::= {
    SUBCLASS OF     {supplementaryService}
    MAY CONTAIN {forwardedToNumber |
typesOfNotifications}
    ID          id-oc-callForwarding}

forwardedToNumber  ATTRIBUTE ::= {
    WITH SYNTAX     NumericString (SIZE(1..ub-international-isdn-number))
    EQUALITY MATCHING RULE      numericStringMatch
    SUBSTRINGS MATCHING RULE    reversePrefixMatch
    ID                  id-at-forwardedToNumber}

typesOfNotifications  ATTRIBUTE ::= {
    WITH SYNTAX     TypesOfNotification
    SINGLE VALUE    TRUE
    ID          id-at-typesOfNotifications}

TypesOfNotification ::= BIT STRING {
    callingUserWithForwardedToNumber (1)}
```

### 5.2.18 Line identification service

This object class is used to represent information related to each calling line identification services ( CLIR, COLR). This includes the type of activation for the service as well as the status of the supplementary service and its identifier inherited from the **SupplementaryService** object class.

```
lineIdentificationServices OBJECT-CLASS ::= {
    SUBCLASS OF     {supplementaryService}
    MAY CONTAIN {perCallBasis}
    ID          id-oc-lineIdentificationServices}

perCallBasis ATTRIBUTE ::= {
WITH SYNTAX     BOOLEAN
EQUALITY MATHING RULE   booleanMatch
SINGLE VALUE        TRUE
ID          id-at-perCallBasis}
```

It is to be noted that the access interface does not transport the keypad in phase 1. Operators who wishes to offer this service for phase 1 could use prefix of dialled number but no "#" or "*" methods.

## 5.3 Structure of the CTM information model

The structure of the information tree for the CTM service is defined by the following name-forms and structure-rules.

Figure 4 gives an overview of that structure. The arrows on the figure are representing naming paths. The boxes are the different object classes used in the CTM information model. Their identifier are provided in bold letters at the top of the boxes. The dotted boxes represent some object classes that are not specific of the CTM information model but are defined in ITU-T Recommendation X.521 [10].

The lower part of the boxes (if any) contains the CTM specific attributes already mentioned in the OBJECT-CLASSes descriptions above. The naming attributes are underlined whereas the optional attributes are written in italics.

A bold arrow is drawn between each alias object class and the class of the aliased entries.

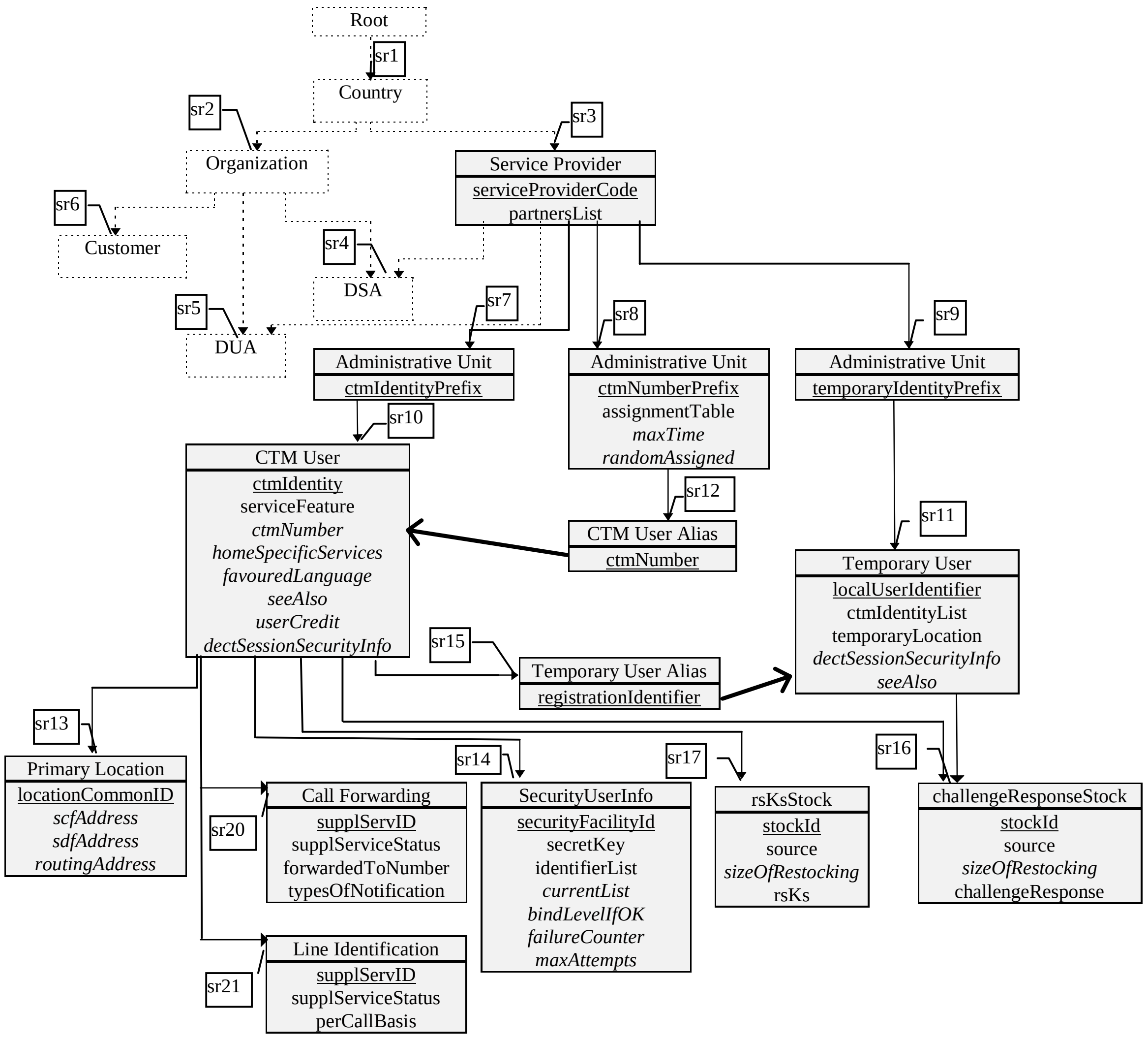


**Figure 4: Structure of the CTM information model**

### 5.3.1 Service provider

Entries of class **serviceProvider** are named using the value of the **serviceProviderCode** attribute. This value is the ITU-T Recommendation E.212 [16] MNC allocated to the service provider.

```
serviceProviderNameForm  NAME-FORM ::= {
    NAMES           serviceProvider
    WITH ATTRIBUTES     {serviceProviderCode}
    ID              id-nf-serviceProviderNameForm}
```

Such entries are subordinate to entries of class country which are immediate subordinates of the root entry.

```
sr1 STRUCTURE-RULE ::= {
    NAME FORM   countryNameForm
    ID      1}

sr3 STRUCTURE-RULE::= {
    NAME FORM       serviceProviderNameForm
    SUPERIOR RULES  {sr1}
    ID          3}
```

### 5.3.2 Administrative unit

Entries of class administrative unit are named using one of the following three forms.

The **adminUnitNameForm1** is used when the entry is intended to be the superior of entry of class **ctmUser**.

```
adminUnitNameForm1  NAME-FORM ::= {
    NAMES       administrativeUnit
    WITH ATTRIBUTES {ctmIdentityPrefix}
    ID          id-nf-adminUnitNameForm1}
```

The adminUnitNameForm2 is used when the entry is intended to be the superior of entry of class ctmUserAlias.

```
adminUnitNameForm2  NAME-FORM ::= {
    NAMES       administrativeUnit
    WITH ATTRIBUTES {ctmNumberPrefix}
    ID          id-nf-adminUnitNameForm2}
```

The adminUnitNameForm3 is used when the entry is intended to be the superior of entry of class temporaryUser.

```
adminUnitNameForm3  NAME-FORM ::= {
    NAMES       administrativeUnit
    WITH ATTRIBUTES {temporaryIdentityPrefix}
    ID          id-nf-adminUnitNameForm3}
```

Entries of class **administrativeUnit** are always subordinate to entries of class **serviceProvider**.

```
sr7 STRUCTURE-RULE::= {
    NAME FORM       adminUnitNameForm1
    SUPERIOR RULES  {sr3}
    ID          7}

sr8 STRUCTURE-RULE::= {
    NAME FORM       adminUnitNameForm2
    SUPERIOR RULES  {sr3}
    ID          8}

sr9 STRUCTURE-RULE ::= {
    NAME FORM       adminUnitNameForm3
    SUPERIOR RULES  {sr3}
    ID          9}
```

### 5.3.3 CTM user

Entries of class **ctmUser** are named using the **ctmIdentity**.

```
ctmUserNameForm  NAME-FORM ::= {
    NAMES       ctmUser
    WITH ATTRIBUTES {ctmIdentity}
    ID          id-nf-ctmUserNameForm}
```

They are subordinate to entries of class **administrativeUnit** which are named using a **ctmIdentityPrefix**.

```
sr10 STRUCTURE-RULE::= {
    NAME FORM       ctmUserNameForm
    SUPERIOR RULES  {sr7}
    ID          10}
```

### 5.3.4 Temporary user

Entries of class **temporaryUserAlias** are named using the **localUserIdentifier**.

```
temporaryUserNameForm  NAME-FORM ::= {
    NAMES           temporaryUser
    WITH ATTRIBUTES     {localUserIdentifier}
    ID              id-nf-temporaryUserNameForm}
```

They are subordinate to entries of class **administrativeUnit** which are named using a **localIdentityPrefix**.

```
sr11 STRUCTURE-RULE::= {
    NAME FORM       temporaryUserNameForm
    SUPERIOR RULES  {sr9}
    ID          11}
```

### 5.3.5 CTM user alias

Entries of class **ctmUserAlias** are named using the **ctmNumber**.

```
ctmUserAliasNameForm  NAME-FORM ::= {
    NAMES       ctmUserAlias
    WITH ATTRIBUTES {ctmNumber}
    ID          id-nf-ctmUserAliasNameForm}
```

They are subordinate to entries of class **administrativeUnit** which are named using a **ctmNumberPrefix**.

```
sr12 STRUCTURE-RULE::= {
    NAME FORM       ctmUserAliasNameForm
    SUPERIOR RULES  {sr8}
    ID          12}
```

### 5.3.6 Primary location

Entries of class **Primary Location** are named using the **locationCommonId** (constant 1).

```
primaryLocationNameForm NAME-FORM ::={
    NAMES       primaryLocation
    WITH ATTRIBUTES {locationCommonId}
    ID          id-nf-primaryLocationNameForm}
```

They are subordinate to entries of class **ctmUser**.

```
sr13 STRUCTURE-RULE::= {
    NAME FORM       primaryLocationNameForm
    SUPERIOR RULES  {sr10}
    ID          13}
```

### 5.3.7 Security user information

Entries of class securityUserInfo are named using securityFacilityId.

```
securityUserInfoNameForm NAME-FORM ::={
    NAMES       securityUserInfo
    WITH ATTRIBUTES {securityFacilityId}
    ID          id-nf-securityUserInfoNameForm}
```

They are subordinate to entries of class **ctmUser**.

```
sr14 STRUCTURE-RULE::= {
    NAME FORM       securityUserInfoNameForm
    SUPERIOR RULES  {sr10}
    ID          14}
```

### 5.3.8 Temporary user alias

Entries of class **temporaryUserAlias** are named using a **registrationIdentifier** attribute.

```
temporaryUserAliasNameForm  NAME-FORM ::= {
    NAMES           temporaryUserAlias
    WITH ATTRIBUTES     {registrationIdentifier}
    ID              id-nf-temporaryUserAliasNameForm}
```

They are subordinate to entries of class ctmUser.

```
sr15    STRUCTURE-RULE ::= {
    NAME FORM       temporaryUserAliasNameForm
    SUPERIOR RULES  {sr10}
    ID          15}
```

NOTE: The value of the naming attribute does not need to be known by the SCF. Access to the contents of such entries can be made by searching the subtree subordinate to their superior entry.

### 5.3.9 Challenge response stock

Entries of class **challengeResponseStock** are named using **StockId**.

```
challengeResponseStockNameForm NAME-FORM ::={
    NAMES       challengeResponseStock
    WITH ATTRIBUTES {stockId}
    ID          id-nf-challengeResponseStockNameForm}
```

They are subordinate to entries of class ctmUser.

```
sr16 STRUCTURE-RULE ::= {
    NAME FORM   challengeResponseStockNameForm
    SUPERIOR RULES  {sr10|sr11}
    ID          16}
```

### 5.3.10 RsKs stock

Entries of class **rsKsStock** are named using **StockId**.

```
rsKsStockNameForm NAME-FORM ::={
    NAMES       rsKsStock
    WITH ATTRIBUTES {stockId}
    ID          id-nf-rsKsStockNameForm}
```

They are subordinate to entries of class ctmUser.

```
sr17 STRUCTURE-RULE ::= {
    NAME FORM   rsKsStockNameForm
    SUPERIOR RULES  {sr10}
    ID          17}
```

### 5.3.11 Call forwarding service

Entries of class **callForwardingService** are named using the **supplementaryServiceCode**.

```
callForwardingNameForm  NAME-FORM ::= {
    NAMES       callForwarding
    WITH ATTRIBUTES {supplServId}
    ID          id-nf-callForwardingNameForm}
```

They are subordinate to entries of class **ctmUser**.

```
sr20 STRUCTURE-RULE::= {
    NAME FORM       callForwardingNameForm
    SUPERIOR RULES  {sr10}
    ID          20}
```

#### 5.3.12 Line identification service

Entries of class **lineIdentificationService** are named using the **supplementaryServiceCode**.

```
lineIdentificationServicesNameForm  NAME-FORM ::= {
    NAMES       lineIdentificationServices
    WITH ATTRIBUTES {supplServId}
    ID          id-nf-lineIdentificationServicesNameForm}
```

They are subordinate to entries of class **ctmUser**.

```
sr21 STRUCTURE-RULE::= {
    NAME FORM       lineIdentificationServicesNameForm
    SUPERIOR RULES  {sr10}
    ID          21}
```

## 5.4 Contexts and methods

### 5.4.1 Contexts

#### 5.4.1.1 List of required contexts

The following context types defined in ITU-T Recommendation X.520 [9] may be used to associate context values to attribute values in the CTM DIB:

- temporalContext.

```
temporalContext CONTEXT ::= {
    WITH SYNTAX     TimeSpecification
    ASSERTED AS TimeAssertion
    ID          id-avc-temporal }
```

The following context types defined in ITU-T Recommendation Q.1228 [8] subclause 7.2.3 may be used to associate context values to attribute values in the CTM DIB:

- assignmentContext.

Assuming that a set of available resources is modeled as a multi-valued attribute, the assignment of a particular user to a resource can be stored in the SDF by adding to the value representing the selected resource an attribute context value whose syntax is a DN (if the user is represented by the entry in the DIT) or a GenericNumber. For that purpose the "assignmentContext" is defined as follows:

```
-- This context associates a directory user name to a value.
assignmentContext CONTEXT ::={
WITH SYNTAX CHOICE {
    directoryUserName   DistinguishedName,
    genericNumber       GenericNumber}
ID          id-avc-assignmentContext}
```

#### 5.4.1.2 Context usage

The permissible context types that may be stored with an attribute are:

```
ctmDITContextUse1 DIT-CONTEXT-USE-RULE ::= {
ATTRIBUTE TYPE      id-at-localUserIdentifier
MANDATORY CONTEXTS  {temporalContext }}

ctmDITContextUse2 DIT-CONTEXT-USE-RULE::= {
ATTRIBUTE TYPE      id-at-temporaryLocation
MANDATORY CONTEXTS  {temporalContext }}

ctmDITContextUse3 DIT-CONTEXT-USE-RULE::= {
ATTRIBUTE TYPE      id-at-supplServiceStatus
MANDATORY CONTEXTS  {temporalContext }}

ctmDITContextUse4 DIT-CONTEXT-USE-RULE::= {
ATTRIBUTE TYPE      id-at-roamingNumberPool
MANDATORY CONTEXTS  {temporalContext | assignmentContext}}
```

```
ctmDITContextUse5 DIT-CONTEXT-USE-RULE::= {
ATTRIBUTE TYPE      id-at-primaryLocation
MANDATORY CONTEXTS  {temporalContext }}

ctmDITContextUse6 DIT-CONTEXT-USE-RULE::= {
ATTRIBUTE TYPE      id-at-registrationIdentifier
MANDATORY CONTEXTS  {temporalContext }}

ctmDITContextUse7 DIT-CONTEXT-USE-RULE::= {
ATTRIBUTE TYPE      id-at-ctmIdentity
MANDATORY CONTEXTS  {temporalContext }}

ctmDITContextUse8 DIT-CONTEXT-USE-RULE::= {
ATTRIBUTE TYPE      id-at-dectSessionSecurityInfo
MANDATORY CONTEXTS  {assignmentContext }}

ctmDITContextUse9 DIT-CONTEXT-USE-RULE::= {
ATTRIBUTE TYPE      id-at-currentList
MANDATORY CONTEXTS  {temporalContext }}
```

### 5.4.2 Methods

#### 5.4.2.1 CTM methods

##### 5.4.2.1.1 Reservation methods

Reservation methods are a couple of generic methods which permit to reserve a unique value in a pool. (one to assign, the other to release).

```
selectAndAssign METHOD ::={
SPECIFIC-INPUT  DistinguishedName
                    -- The DN of the user to which
                    -- the selected value is temporarily assigned
OUTPUT ATTRIBUTES   assignmentTable
BEHAVIOUR           "This method performs the following actions on the entry identified by the
execute argument:
1)  selects a value of the assignmentTable attribute which has no context associated with it or is
associated with an expired temporal context.
2)  adds an assignmentContextValue equal to the specific input to the selected value.
3)  adds a temporal context value so that the selected value becomes irrelevant after maxtime units
of time.
4) return the selected value without context values.
"
ID              id-mt-selectAndAssign}

findAndRelease METHOD ::={
INPUT ATTRIBUTE assignmentTable
SPECIFIC-OUTPUT DistinguishedName   -- The DN of the user to which
-- the selected value was
-- temporarily assigned
BEHAVIOUR   "This method performs the following actions on the entry identified by the execute
argument:
1) find the value of the assignmentTable
attribute which is equal to the one received in the
input-assertions element of the execute argument.
2) remove from the DIB all its associated context values.
3) if this value was associated with valid temporal
context values and an assignmentContext Value,
return the associated assignmentContext value (user DN).
ID  id-mt-findAndRelease}
```

##### 5.4.2.1.2 Provide disposable security authentication tokens

A DUA shall request a DSA to supply a stock of tokens by a method "provideTokens".

The role of DUA could be played by a SCF or by the local SDF to reprovision its stock.

The role of DSA could be played by the local SDF or by the home SDF.

```
provideTokens METHOD ::={
SPECIFIC-INPUT  INTEGER, -- how many tokens are requested (NOfRT)
                OBJECT IDENTIFIER   -- oid of the attribute (tokens)
SPECIFIC-OUTPUT     ATTRIBUTE --attribute selected as input (tokens)
BEHAVIOUR   "This method performs the following actions on the entry (thisEntry would be a variable
with the DN value of this entry) identified by the execute argument:
1)  if the attribute sizeOfRestocking doesn't exist in the entry, define a variable MAXNT with a
```

```
default value (e.g. 10) else MAXNT takes the value of the attribute sizeOfRestocking
2)  verify that NofRT is inferior or egal to MAXNT
(return an "execute error" if NofRT value is superior to MAXNT)
3)  read the attribute of the entry which has the selected oid and count the number of values (0 if
empty) and put the result in a variable N
(return "execute error if the attribute doesn't exist)
4)  read the source attribute in the entry
(return "execute error" error if source doesn't exist)
5)  if N is inferior to NOfRT and the DN of source indicates an entry of class or subclass
tokenStock :
5a) bind anonymous with the DSA which contains the entry defined by the address field of source
5b) execute the method provideTokens on the entry with MAXNT as value of the specific-input
5c) if none error is returned , modify the tokens attribute by adding the resulted values
    6)  if N is inferior to NOfRT and the DN of source indicates an entry of class or subclass
securityUserInformation
6a) execute the method defined by fillMethodIdentifier field value on the entry defined by the DN
with MAXNT as specific input
6b) if none error is returned ,modify the tokens attribute by adding the resulted values
    7)  read the tokens attribute
8)  define a variable "toBeReturned" with NofRT values of tokens attribute and a variable "toBeKept"
with remainder values
    9)  remove tokens attribute
    10) modify tokens attribute by adding the "toBeKept" values
    11) return the "toBeReturned" values
"
ID              id-mt-provideTokens}
```

In the case of CTM the temporary entry (of **objectClass challengeResponseStock**) will contain in the SDF visited the DN of the entry (of **objectClass challengeResponseStock**), and will contain in the SDF home the DN of the entry (of **objectClass securityUserInfo**), the entry **securityUserInfo** designated by the DN will contain in its **identifierList** attribute the value **id-mt-fillSecurityTokens** in the field **fillMethodIdentifier**.

```
fillSecurityTokens METHOD ::={
SPECIFIC-INPUT  INTEGER -- N number of value to be computed
SPECIFIC-OUTPUT SEQUENCE OF ThreePartMessage
BEHAVIOUR   "This method performs the following actions on the entry identified by the execute
argument, this entry shall be of object class (or subclass) genericSecurityUserInfo:
- read the secretKey attribute and oneToTwoAlgorithm attribute
- repeat N times
        - fill the first BIT STRING field with a random value
        - apply the oneToTwoAlgorithm cryptographic algorithms to compute
        the second and third field.
- return N ThreePartMessage values
"
ID      id-mt-fillSecurityTokens}
```

#### 5.4.2.1.3 Verification of user credentials

The METHOD to make a verification of the user credential against the information included in an entry of type **securityUserInfo**. This METHOD could be used during the bind or over a dialog in order to authenticate the user in the database. It could happen for example when the user changes its service data over a management access.

```
verifyCredentials METHOD ::={
SPECIFIC-INPUT  TwoPartMessage
-- see the definition of this type below
SPECIFIC-OUTPUT BOOLEAN
        -- to indicated the success of the verification
BEHAVIOUR   "This method performs the following actions on the entry identified by the execute
argument, this entry would be of class genericSecurityUserInfo:
1) if maxattempts is present, verify that failureCounter is inferior to its value
2)  read the value of identifierList attribute
(return "bad format entry" if failure)
3) if conformMethodIdentifier is NULL go to step 5)
4)  run conformMethodIdentifier method on TwopartMessage provided as specific input
(return a "badconformance" error if the execution fails or if the result is false)
5)  run the oneToOneAlgorithm on the messageData bit string to get an expected certificationCode bit
string
6)  return TRUE if the expected and provided certificationCode values matched,
7)  otherwise if failureCounter is present, increment it
8)  return FALSE
"
ID              id-mt-verifyCredentials}
```

In the case of a CTM user with a DECT terminal, the value of **conformMethodIdentifier** would be **id-mt-conformCredentials.**

```
ConformCredentials METHOD ::={
SPECIFIC-INPUT  TwoPartMessage
-- see the definition of this type below
SPECIFIC-OUTPUT BOOLEAN
        -- to indicated the success of the verification
BEHAVIOUR   "This method performs the following actions on the entry identified by the execute
argument, this entry would be of class genericSecurityUserInfo:
-   verify with an embedded conformance algorithm that messageData value of TwoPartMessage is no
replay (RAND is in the current time window and the associated RS is not in the list of the current
time windows currentList).
-   add RAND to time windows list currentList.
-   return TRUE if no replay,
-   otherwise return FALSE
"
ID              id-mt-dectConformCredentials}
```

#### 5.4.2.2 CTM method usage

Some object class shall be associated with supported method(s) by a method rule.

### 5.4.3 roamingNumberPool

The object class **roamingNumberPool** supports two methods (**findAndRelease, selectAndAssign**), this rule defines the link.

```
roamingNumberRule METHOD-USE-RULE ::= {
OBJECT CLASS TYPE id-oc-roamingNumberPool
MANDATORY METHODS {findAndRelease| selectAndAssign}}
```

### 5.4.4 challengeResponseStock

The object class **challengeResponseStock** supports the method **provideTokens**.

```
tokensRule METHOD-USE-RULE ::= {
OBJECT CLASS TYPE       id-oc-challengeResponseStock
MANDATORY METHODS   {provideTokens}}
```

### 5.4.5 SecurityUserInfo

The object class **SecurityUserInfo** supports the method **verifyCredentials**.

```
securityUserInfoRule METHOD-USE-RULE ::= {
OBJECT CLASS TYPE       id-oc-securityUserInfo
MANDATORY METHODS   {verifyCredentials| fillSecurityTokens|conformCrendentials}}
```

## 5.5 Mapping onto functional architecture

SDFsl refers to one or more subtrees (naming context), starting at an entry of class **administrativeUnit**, which contains collections of entries of class **ctmUser** and associated subordinated and alias entries. An SDFmm refers to one or more subtrees (naming context), starting at an entry of class **administrativeUnit**, which contains a collection of entries of class **temporaryUser**. Such entries are dynamically created and deleted by the SCFmm.

In addition to those associated with its role(s), an SDF may also include other kind of entries (e.g. for storing locally defined information) or sub-entries (e.g. for access control purposes).

Figure 5 illustrates a possible mapping of the CTM DIT onto several SDFs. SDF-A and SDF-B play the role of an SDFsl only, SDF-D plays the role of an SDFmm only while both roles are played by SDP-C.

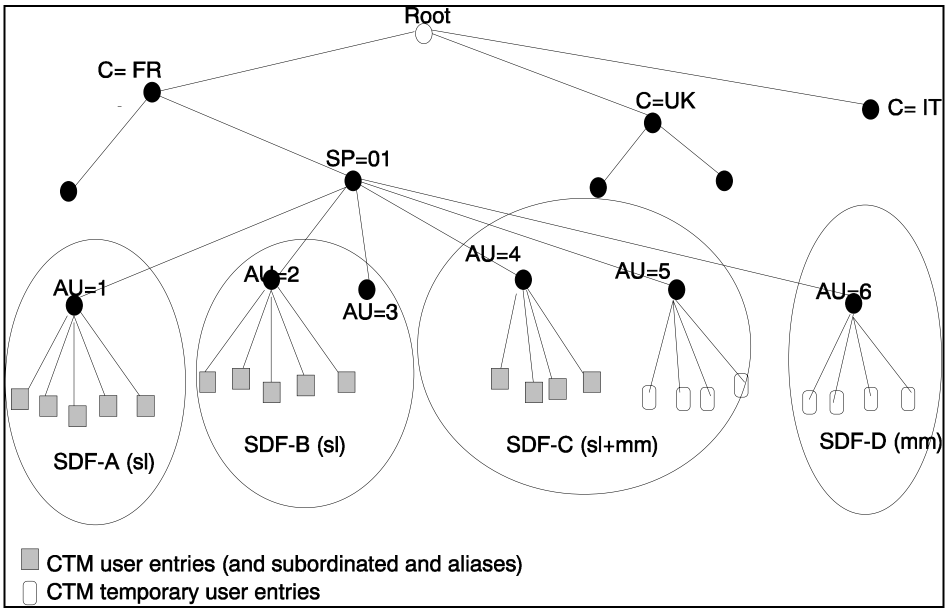


**Figure 5: Mapping of SDFsl/mm onto a DIT**

## 5.6 Naming

### 5.6.1 General

An SCF needs to be able to determine the Directory DN of the entry (or of an alias entry) representing this user and of the temporary entry created by the SCFmm when the terminal registers. Such DNs are used in the construction of bind credentials and operation arguments.

### 5.6.2 Access to the SDFsl

Depending on the information available to the service logic (CTM user identity or the CTM Number), the SCF determines the DN of the CTM user entry or of the alias entry.

#### 5.6.2.1 Use of the CTM identity

The SCF derives the DN of the entry of class **ctmUser** from the CTM Identity according to the mapping described in table 1.

**Table 1**

| RDN component | Value | Method |
|---|---|---|
| CountryName | IS 3166 country code | derived from CTM Identity MCC |
| Service Provider Code | E.212 MNC | taken from CTM Identity |
| CTM Identity Prefix | first two digits of E.212 MSIN | taken from CTM Identity |
| CTM Identity | trailing digits of E.212 MSIN | taken from CTM Identity |

#### 5.6.2.2 Use of the CTM number

The SCF derives the DN of the entry of class **ctmUserAlias** from the CTM Number according to the mapping described in table 2.

**Table 2**

| RDN component | Value | Method |
|---|---|---|
| CountryName | IS 3166 country code | derived from CTM Number CC |
| Service Provider Code | E.212 MNC | derived from CTM Number NDC |
| CTM Number Prefix | first two digits of CTM number | taken from CTM Number |
| CTM Identity | trailing digits of CTM number | taken from CTM Number |

### 5.6.3 Access to the SDFmm

The SCF derives the DN of the entry of class **temporaryUser** from the CTM Identity according to the mapping described in table 3.

**Table 3**

| RDN component | Value | Method |
|---|---|---|
| CountryName | Own IS 3166 country code | Taken from configuration |
| Service Provider Code | Own E.212 MNC | taken from configuration |
| Local Identity prefix | Locally defined prefix | derived from FT or CUSF address |
| Local User Identifier | All Digits of CTM Identity | CTM Identity |

## 5.7 Data distribution

The global CTM DIT is distributed over several DIT domains each of which is administered by a CTM Service Provider. Within one domain the DIT may be distributed over several SDFs.

Each SDF contains a fragment of the DIB which corresponds to one ore more subtrees of the DIT.

When distributed procedures cannot be use, the SCF needs to establish a dialogue with the SDF which holds the DIB fragment it needs to access. Section 6 defines the principles for addressing an SDF.

When distributed procedures can be used, the SCF establishes a dialogue with any SDF. This SDF is then responsible for chaining the requests up to the SDF(s) which holds the information involved by these requests.

### 5.7.1 Provider related information

Provider related information is held in entries of class **serviceProvider**. Since such entries do not need to be visible outside a domain their physical location does not need to be standardized. Typical implementation strategies are:

- within one domain, the serviceProvider's entry is held in a master SDF and in all other SDFs as an entry-copy. Such copies may be managed and maintained using the procedures defined in ITU-T Recommendation X.525 or by any other non standardized means;

- a single SDF within the domain holds the associated serviceProvider's entry.

If chaining procedures are not implemented, an SCF which needs to access information in this entry (e.g. to verify some agreement) needs to be able to bind to this SDF (i.e. this requires that its address be known as a configuration parameter by all the SCFs in that domain).

If chaining procedures are implemented, an SCF which need to access information in this entry (e.g. to verify some agreement) may bind to any SDF in its domain which will chain the request to SDF holding the provide related information.

### 5.7.2 CTM user related information

CTM User related information is held in entries subordinate to entries of class **administrativeUnit**.

Within each domain, one SDF shall contain a master copy of the subtrees representing the CTM User Profile (i.e. the subtrees starting at the entry of class **ctmUser** and **ctmUserAlias**).

If chaining procedures are not in use within that domain the two subtrees have to reside in the same SDF.

An SDF which primarily contains master copies of such subtrees is referred to as an SDFsl.

Temporary user/terminal information is stored in entries of class **temporaryUser** which are subordinate to entries of class **administrativeUnit**. An SDF which primarily contains entries of class **temporaryUser** is referred to as an SDFmm.

If distributed procedures are not in use within one domain, the SCCP routeing procedures should ensure that a dialogue related to one CTM user is always established with the physical entity where resides the SDF holding the profile information.

## 5.8 Data replication

Part of the entries representing a CTMUser may be replicated in other SDFs in the same domain or in other domains providing that a bilateral agreement exists. When the master SDF and the SDF holding a copy are not in the same domain, the copy shall be maintained and managed using the procedures defined for DISP subset in subclause 8.3 of ITU-T Recommendation Q.1228 [8] (i.e. using the Directory Information Shadowing Protocol).

In order to trigger the transfer of a (partial) copy of a CTM user entry to a local SDF, the SCF needs to modify the objectClass attribute of the CTM User entry. The precise modification to be performed is network specific or is part of the shadowing agreement between two networks.

The triggering could be done at different moment, for example during the location registration. The way to achieve this triggering is illustrated in the location registration procedure.

# 6 Addressing

## 6.1 General

Each network physical entity (e.g. SCP, SDP, SSP, IP, Portable radio Termination (PT)) shall be allocated addresses to open dialogs or continue dialog.

Addressing a functional entity (i.e. SCF or SDF) should be understood as addressing the physical entity which contains this Functional Entity (FE).

- The FEs may be at physically separate locations.
- Two or more FEs may be at the same location.
- The FE load could be shared between two or more different physically locations or a FE could have a back-up.
- Addressing of the correct FE shall be supported both in intra-domain or in inter-domain case.

Each functional entity for CTM shall be identified by a unique number. This forms the GTAI part of the global title. The unique number means this is a number that is not used by any other application.

The format and coding of address parameters carried by the SCCP shall comply with ETS 300 009-1 [14] with the following restrictions:

In case of inter-network addressing, the format and coding of address parameters carried by the SCCP across network boundaries shall comply with the following rules:

a) Called Party Address

   - SSN indicator = 1 (IN SSN always included);
   - Global title indicator = 0100 (Global title includes translation type, numbering plan, encoding scheme and nature of address indicator);
   - the translation type field will be coded "00000000" (Not used);

- Routeing indicator = 0 (Routeing on global title);

b) Calling Party Address

- SSN indicator = 1 (IN SSN always included);
- Point code indicator = 0;
- Global title indicator = 0100 (Global title includes translation type, numbering plan, encoding scheme and nature of address indicator);
- the translation type field will be coded "00000000" (Not used);
- Routeing indicator = 0 (Routeing on Global Title).

If a global title translation is required for obtaining routeing information, one of the numbering plans ITU-T Recommendation E.164 or ITU-T Recommendation E.214 [17] is applicable.

Either numbering plans E.164 or E.214 may be used in both the calling party address and the called party address of SCCP messages at any stage of a dialogue. The ITU-T Recommendation E.212 [16] numbering plan may only be used in SCCP primitives invoked at the originating node.

As stated in EN 301 140-1 [2], when load sharing or backup procedures are implemented, a responding application is responsible for providing an appropriate responding address in the first backward message it sends for a dialogue. This address shall uniquely identify the node where this application resides.

## 6.2 Addressing the SDFsl

There are several cases where the SDFsl has to be addressed:

### 6.2.1 During incoming call set-up

When a call is initiated, the SCF needs to retrieve user related information from the SDFsl.

#### 6.2.1.1 Intra-domain

If distributed procedures are in use, the SCF shall establish a dialog with a preferred SDF whose SCCP address is known as a configuration parameter. This preferred SDF is then responsible for chaining the request up to the SDF where the subscriber information is held. Addressing on the SDF-SDF interface is discussed in subclause 6.4.

If distributed procedures are not in use, the SCF shall establish a dialogue with the SDF which contains the subscriber profile (i.e. the entry of class ctmUser). The address provided to SCCP to reach this SDF is derived from the CTM Number dialed by the calling subscriber. The dialed number will be translated into either an SPC or a global title.

#### 6.2.1.2 Inter-domain

If there is a co-operation agreement for using distributed procedures between the two domains, the SCF shall establish a dialogue with a preferred SDF (in its own domain) whose SCCP address is known as a configuration parameter. This preferred SDF is then responsible for chaining the request up to the SDF where the subscriber information is held. Addressing on the SDF-SDF interface is discussed in subclause 6.4.

If there is no co-operation agreement for using distributed procedures between the two domains, the SCF shall provide SCCP with a Global Title derived from the CTM Number dialed by the calling subscriber. This Global Title should be such that the message is routed up to a node in the network where the subscriber profile resides. This node may be an SCCP Gateway which continues the translation in order to reach the SDF or an SDF acting as a gateway (if distributed procedures are in use in the home domain).

### 6.2.2 At location registration

When a CTM user registers for the first time in an area, the SCF has to initiate the modification of location information in the SDFsl.

The only data for addressing the SDFsl that the SCF has available is contained in the CTM identity, and addressing information for SCCP has to be derived from it. When continuing the established dialogue (as with any other dialogue), the SCF has to derive the routeing information towards the SDFsl from the Calling Party Address received with the first responding CONTINUE message until the dialogue terminating message is received. This means that the SCF has to be able to address the SDFsl based:

- on an E.214 mobile global title originally derived by the SDFmm from the CTMId; or
- on an E.164 SDF address; or
- in the case of intra-IN domain signalling, on an SPC.

If the SDFsl is in the same IN Domain as the SDFmm, local translation tables may exist to derive an SPC. In the inter-domain case, the Global title has to be derived from the CTM Id, using the principles contained in ITU-T Recommendation E.214 [17]. A summary of the translation from the CTM Id (ITU-T Recommendation E.212 [16]) to mobile global title (ITU-T Recommendation E.214 [17]) is shown below:

- E.212 mobile country code translates to E.164 Country Code;
- E.212 mobile network code translates to E.164 National Destination Code;
- E.212 Mobile Subscriber Identification Number (MSIN) is carried unchanged if within the E.164 number maximum length and terminated by the ST signal (15 digits + ST). If the mobile global title is more than 15 digits the number is truncated to 15 by deleting the least significant digits.

### 6.2.3 At outgoing call set-up

If replication of user profiles is not used, the SCFsl needs to establish a dialogue with the SDFsl. The procedures described for location updating are applicable.

## 6.3 The SDFmm

There are several cases when the SDFmm needs to be addressed:

### 6.3.1 Incoming call

When a call is initiated, the SCF may receive from the SDFsl the address of the SDFmm as location information.

The SCF provides this address information to SCCP for establishing a dialog with the SDFmm in order to retrieve more precise location information (i.e. the FT address).

### 6.3.2 Outgoing call

The SCFsl may initiate a dialogue with the SDFmm if a copy of a part of the entries representing the user profile is stored in the SDFmm.

The SCCP address of the SDFmm is known by the SCFsl as a configuration parameter or is derived from the FT Address.

### 6.3.3 Location registration/deregistration

If the concept of SDFmm is implemented, the SCFmm has to establish a dialogue with the SDFmm in order to determine whether the CTM user is already registered in its sub-domain.

The SCCP address of the SDFmm is known by the SCFmm as a configuration parameter or is derived from the FT Address.

## 6.4 Addressing by another SDF

An SDF may address another SDF if distributed procedures are in use. The address provided to SCCP is taken from the knowledge reference used to select the target SDF.

In order to enable the use of distributed procedures across SS7 signalling networks, the address field of knowledge references shall be structured as follows:

- the pSelector, sSelector and tSelector sub-fields shall be absent;
- the nAddresses subfield shall contain a value of type **EntityAddress** which identifies the terminating SDF.

## 6.5 Addressing of an SCF

### 6.5.1 Addressing by another SCF

In case home specific services need to be provided, the SCCP address (Global Titles (GT) field) of an assisting SCF is retrieved from the SDFsl by the controlling SCF. This information is used as a global title.

### 6.5.2 Addressing by an SDF

An SDF will only address an SCF in response to a dialogue initiated by the SCF. The address it uses shall be the one received in the Calling Party Address parameter of the SCCP message carrying the BEGIN message.

# 7 Network interfaces

## 7.1 SCF-SDF

The SCF-SDF interface shall conform to EN 301 140-1 [2].

A new application-context to be used over this interface is defined as follows to support a supplementary abstract syntax:

```
scf-sdf-ctm-context     APPLICATION-CONTEXT ::= {
    CONTRACT                scf-sdf-contract
    TERMINATION         basic
    ABSTRACT SYNTAXES       {dialogue-abstract-syntax |
                        challenge-response-abstract-syntax |
                        scf-sdf-as}
    APPLICATION CONTEXT NAME    id-ac-scfsdfmobile}
```

The abstract-syntax **challenge-response-abstract-syntax** is defined as follows. This abstract-syntax is used to convey challenge/response tokens in the argument of the **bind** operation with an external authentication.

```
challenge-response-abstract-syntax      ABSTRACT-SYNTAX ::={
SecurityFacilityCredentials
IDENTIFIED BY                   id-as-challenge-response}

id-as-challenge-response    OBJECT IDENTIFIER ::= {id-as challenge-response (1) version 1(1)}

SecurityFacilityCredentials ::= SEQUENCE {
secuCredArg SecurityCredentialsArg,
secuCredResult  SecurityCredentialsResult}
```

The arguments included the CTM number or CTM identity of the user, the security facility to be used (e.g. sf-dect or sf-subscription) to authenticate this user and verify the SCF assertions (e.g. no replay of the challenge), the challenge sent by the SCF to the terminal, the response of the terminal. When the optionally mutually authentication is required, the challenge send by the terminal could be included.

```
SecurityCredentialsArg ::= {
name            DistinguishedName,
securityFacilityId      SF-CODE,
challengeResponseUsr    TwoPartsMessage,
challengeNetwork        BIT STRING OPTIONAL}
```

If the bind succeed, the result would be the network response to be sent to the terminal to authenticate the network and the cipheringKey to be used for cipher procedure.

```
SecurityCredentialsResult ::= {
challengeResponseNetwork    ThreePartsMessage   OPTIONAL}
```

## 7.2 SDF-SDF

The SDF-SDF interface shall conform to EN 301 140-1 [2].

## 7.3 SCF-SCF

The SCF-SCF interface shall conform to EN 301 140-1 [2].

This EN describes the use of the SCF-SCF interface for offering home-specific services to moving subscribers. The use of this interface to provide an alternative realization (to that supported by the SCF-SDF interface) of mobility management procedures is outside the scope of this EN.

The following object identifier shall be used as an agreement ID by the SCFsl in the visited network when invoking the scfBind operation:

```
ctm-hss-agreement OBJECT IDENTIFIER ::= {itu identified-organization (4) etsi (9) inDomain (1) in-
ctm (3) interfaces (100) agreement (1)}
```

This interface may be used for requesting assistance from an SCF in the Home Network, when Home Specifc Services need to be provided for the user as indicated in the user profile (e.g. **homeSpecificServices** attribute) or in a mutual agreement.

Six values could be defined for the **homeSpecificServices** attribute:

```
-- local phase 1 service feature identifier
-- the following local identifiers indicate a home specific feature for incoming call
lid-hsf-ic1 ServiceFeaturesIdentifier ::= {local:ic-o1} --for the originating network during set-up
phase
lid-hsf-ic2 ServiceFeaturesIdentifier ::= {local:ic-v1} -- for a visited network during set-up phase
lid-hsf-ic3 ServiceFeaturesIdentifier ::= {local:ic-of} -- for an originating on setup failure
lid-hsf-ic4 ServiceFeaturesIdentifier ::= {local:ic-vf} -- for a visited on setup failure

-- the following local identifiers indicate a home specific feature for outgoing call
lid-hsf-og1 ServiceFeaturesIdentifier ::= {local:oc-v1} -- for the visited network during set-up
phase
lid-hsf-og3 ServiceFeaturesIdentifier ::= {local:oc-vf} -- for the visited network on set-up failure

ic-o1 INTEGER ::=  1
ic-v1 INTEGER ::= 2
ic-of INTEGER ::= 3
ic-vf INTEGER ::= 4
oc-v1 INTEGER ::= 11
oc-vf INTEGER ::= 12
```

The SCF could also use the six same integer values to fill the **RequestedType** field of the **HandlingInformationRequestArg** in the **HandlingInformationRequest** operation.

## 7.4 SCF-CUSF

The SCF-CUSF interface shall be conform to EN 301 140-1 [2].

NOTE: The actual contents of the USI information is defined in EN 301 144-1 [15].

## 7.5 SCF-SSF

The SCF to Service Switching Function (SSF) interface shall be conform to EN 301 140-1 [2].

## 7.6 SCF-SRF

The SCF to Specialized Resource Function (SRF) interface shall be conform to EN 301 140-1 [2].

# 8 SCF and SDF procedures

## 8.1 SDF procedures

### 8.1.1 "bind" procedure

The "bind" operation when the argument of the bind uses the **challenge-response-abstract-syntax** abstract syntax will proceed with the following steps:

- take the name to find the user entry, go down in the subentry (of object class **securityUserInfo**) indicated by the SF-CODE value;
- if **bindLevelIfOK** is absent, the entry could not be used to bind a user by its DN. When present, the value provides for the implemented access control the level of authentication (e.g. "simple", "strong");
- run the **verifyCredential** method on the entry;
- if the result is true, put the challenge sent by the terminal in the first bit string of a **ThreePartMessage**, run the **oneToTwoAlgorithm** algorithm on this **ThreePartMessage**;
- if no error, return the **ThreePartMessage**.

### 8.1.2 List of required methods

In addition to the procedures defined in EN 301 140-1 [2], the SDF shall support the following methods and associate them with some object classes.

Such methods are invoked by the SCF using the **execute** operation:

The following generic methods shall be supported:

- **findAndRelease**, **selectAndAssign** to reserve a unique value from a pool;
- **provideTokens** and **fillSecurityTokens** to distribute tokens to authenticate the visited user/terminal;
- **verifycredentials** and **conformCredentials** to verify security assertions to authenticate the user in the home domain.

## 8.2 SCF procedures

### 8.2.1 Overview

This subclause is an introduction to the CTM-specific procedures in the SCF.

The CTM-specific procedures described in this clause form the CTM-specific service logic program (SLP) needed in the SCF to handle the CTM procedures as described in ETR 090 [11], ETS 300 175-6 [12] and ITU-T Recommendation E.164 [13]. An SLP is normally implementation-dependent and should not be standardized in general. However this present document gives a global view of the CTM service and put in context the type of database requests that has to be standardized.

The arguments of the request could be operator specific and do not need to be standardized since the SCF-SDF interface is used.

The informative annex contains useful guidelines to define these arguments.

### 8.2.2 Authentication

In CTM stage2, three procedures could be used to authenticate a CTM user:

#### 8.2.2.1 Authentication in the SDFsl

This procedure is used to identify and authenticate a user in the home domain It initiates a dialogue between the SDFsl and the visited SCF on behalf of the user.

The dialog on this authenticated association is required for management access (modification of user profiles) it could be also required to offer mutual authentication during an On air subscription.

The arguments permit to indicate the user and the security facility to be used (type of authentication : passwd, ctm, subscription). The database contains the necessary data: keys and algorithms (dect, ct2).

A result indicate that the authentication succeed.

A bind error is returned in case of authentication failure.

#### 8.2.2.2 Authentication by DECT session key

##### 8.2.2.2.1 First authentication, no data stored in the SDFmm

This procedure is used to retrieve DECT session security information (a [KS,RS]) from the home database.

NOTE: For security reasons, this couple has to be changed when used in a multi-operator environment. Then an EXECUTE operation on the Security User Information entry identified by sf-dect-stage1 should be preferred with the id-mt-fillSecurityTokens.

Currently, two operations are possible but the first one is unsecured and requires huge data storage.

The first operation is a SEARCH operation in the CTM user entry; the SCF should indicate its identity in the **assignmentContext** context to retrieve its dedicated value.

The second operation is an EXECUTE operation on the "**RsKsStock**" entry associated to the user with the **provideTokens** METHOD.

The DECT security information parameters retrieved (if any) are stored for future use in the visited domain.

##### 8.2.2.2.2 Data stored in the SDFmm

To retrieve the data from the SDFmm, the operation used should be a SEARCH on the entry where it was stored previously.

#### 8.2.2.3 Authentication by set of n-uplet

This procedure is used by the SCFmm to retrieve a set of n-uplet from the SDFsl or a unique n-uplet from the SDFmm. The operation should be the execute operation on the challengeResponseStock associated to the user (request to the home SDF) or to the temporary user (local request to the visited SDF) with the **provideTokens** METHOD.

### 8.2.3 Location registration/deregistration

#### 8.2.3.1 Registration in the visited database

After the authentication of the roaming CTM user, three entries have to be created:

- two in the SDFmm to stored the mobility and authentication data of this user;

- one in the SDFsl to stored a copy or a knowledge reference to user profile data.

The choice to create the last entry during the location procedure is up to the visited network. An alternative could be to create this entry at the first incoming or outgoing call.

The SCFmm in the visited network triggers the creation of a copy by modifying one of the values of the object class attributes of the entry representing the CTM user. This assumes that in addition to that representing the entry's structural object class, the object class attribute includes the object identifier of an auxiliary object class representing the current visited domain. Such values shall be subordinate to the following root.

{itu identified-organization (4) etsi (9) inDomain (1) in-ctm (3) oc (6) shadowing (100)}

The last component of the object identifier shall be the E.164 number of the SDFsl acting as a shadow consumer.

Detailed guidance on how the replication areas shall be defined is provided in ITU-T Recommendation Q.1228 [8], Appendix II.

The visited network SCF could also store in the temporary user entry the **radioSpecificEnvelop** value.

##### 8.2.3.1.1 SDFmm entries creation

In order to add the temporary user entry, the SCF should use an "addentry" operation.

If the authentication procedure is based on triplet, the SCFmm should also create an authentication entry to store the set (minus one element) of triplet requested during the authentication (cf 0).

#### 8.2.3.2 Data updating in the home database

During this procedure, 4 operations should be launched until a generic mobility METHOD is defined to gather these operations in an atomic operation "execute".

- "removeEntry" to remove the old CTM primary Location entry;
- "addEntry" operation in the CTM primary Location entry (subordinate to the CTM User entry) to set the new location in the SDFsl;
- "removeEntry" operation on the Temporary User Alias entry to remove the Temporary User entry in the old visited SDFmm;
- "modifyEntry" operation on the Temporary User Alias entry with the **dontDereferenceAliases** bit set to True in the **serviceControls** value in order to update the alias with the new Temporary User entry value.

### 8.2.4 Provision of home specific services

By mutual agreement, two operators could decide to offer the home specific whatever the roaming user or it could be based on profile data of the roaming user.

The SCF should retrieve this data using the "search" operation.

When none home specific services are requested the supplementary service status and parameters have to be read. It is possible to have operations retrieving only one parameter and operations dealing with several services.

The parameters are retrieved only if the service is activated. By that means, this operation is used to check if a supplementary service is activated when a call attempt reports that this supplementary service should be invoked. The operation should be a search.

### 8.2.5 Incoming calls

#### 8.2.5.1 Phase before the call setup

If it can be ensured that white book ISUP is available all along the route to the visited network, the originating SCF retrieves a routeing address from the home SDFsl by a search operation. This address is used as a destination address in the Connect operation. The CTM number is passed back to the SSF as a generic number. In the visited network this destination address causes the SSF to triggers a dialogue with the local SCF. The generic number is used by the service logic to retrieve the CTMId from the SDFmm.

If the white book ISUP could not be used, the roaming number procedure has to be used. If the originating network and the visited network have a mutual agreement the SCF sent directly its request of roaming number to the visited SDFmm. Otherwise, it sends its request to the home SDFsl which chains it to the visited SDFmm.

The operation should be "execute" of a **selectAndAssign** METHOD with CTMid as parameter.

It should retrieve also the CTMid attribute and optionally the **homeSpecificServices** attribute to determine if the user subscribed to a network specific incoming feature if these parameters are not already fixed by a mutual agreement. The retrieval is done by a "search" operation.

#### 8.2.5.2 Routeing address or roaming number translation

The visited network received a call setup with the RN as called number, then it translates it back in CTMid, the operation should be "execute" of a **findAndRelease** METHOD which returns the parameter provided by the **selectAndAssign**.

If the Routeing Address was used, the home Network needs to translate the CTM number provided as Generic Number into the CTMid. The operation should be a "search" in the SDFsl local, which maintains a copy or a reference of the user profile.

#### 8.2.5.3 Paging of the terminal

The visited SCF retrieves, from the local SDFmm, the precise location information (FT address) using search operation. It could then page the terminal.

To authenticate the user, one of procedures described in subclause 8.2.2 can be used.

### 8.2.6 Outgoing calls

After the authentication done by the SCFmm the SCFsl look up if the user got a home specific mark. Then if the service is not home specific, it could retrieve the supplementary service data (e.g. line identification ...).

### 8.2.7 On air subscription for DECT terminals

This procedure is used to send the subscription information and set up the authentication key to the DECT terminal which has made a subscription request on the air interface. The procedure inside the application is an internal procedure in CTM phase 1. Only the access procedure needs to be standardized to provide the information of the terminal and be able to answer back the correct parameters.

It is up to the operator to choose its own procedure for its customers.

# Annex A (normative): ASN-1 module of the CTM information model

The following ASN.1 module defines the directory schema to be used for the provision of the CTM service.

```
CTM-DataModel   { etsi in-domain in-ctm modules(1) data-model(0) version1(0) }
-- this modules contains the ASN.1 Information Object Notation
-- for defining the contents of an SDF for the CTM service.

DEFINITIONS::=

BEGIN

--EXPORTS All; --

IMPORTS

    inDomainId FROM InDomainDefinitions {ccitt (0) identified-organization (4) etsi (0)
                      inDomain (1) inDomainDefinitions (0) version1 (1)}

    informationFramework
FROM UsefulDefinitions {joint-iso-ccitt ds (5) module (1) usefulDefinitions (0)}

    OBJECT-CLASS, ATTRIBUTE, NAME-FORM, STRUCTURE-RULE, top, alias,
    DistinguishedName,distinguishedNameMatch
FROM InformationFramework informationFramework

    country, organization, organizationalUnit, person,
    applicationEntity, dSA, countryNameForm, orgNameForm, dSANameForm
FROM SelectedObjectClasses {joint-iso-ccitt ds (5) module (1) selectedObjectClasses (6) 2}

    numericStringMatch, integerOrderingMatch,
    description, integerMatch, reversePrefixMatch, organizationalUnitName,
    commonName, seeAlso, name, member
FROM SelectedAttributeTypes {joint-iso-ccitt ds (5) module (1) selectedAttributeTypes (5) 2}

    userPassword
FROM AuthenticationFramework {joint-iso-ccitt ds (5) module (1) authenticationFramework (7) 2}

id-avc-temporal,temporalContext -- X520 am1
FROM SelectedAttributeTypes {joint-iso-ccitt ds(5) module(1) selectedAttributeTypes(5) 3}

METHOD, METHOD-USE-RULE
 FROM IN-CS2-SDF-InformationFramework
        { ccitt recommendation q 1228 module(0) sdfInformationFramework(9) version1(0) }
id-avc-AssignmentContext
FROM IN-CS2-object-identifiers
        { ccitt recommendation q 1228 modules(0) in-cs2-object-identifiers(17) version1(0) }

;

-- OBJECT CLASSES

administrativeUnit  OBJECT-CLASS ::= {
    SUBCLASS OF {organizationalUnit}
    MAY CONTAIN -- A single attribute from the following three
            {ctmNumberPrefix|
            ctmIdentityPrefix |
            temporaryIdentityPrefix}
    ID      id-oc-administrativeUnit}

callForwarding OBJECT-CLASS ::= {
    SUBCLASS OF     {supplementaryService}
    MAY CONTAIN {forwardedToNumber |
typesOfNotifications}
    ID          id-oc-callForwarding}

challengeResponseStock OBJECT-CLASS ::= {
    SUBCLASS OF     {tokenStock}
    MUST CONTAIN    {challengeResponse}
    ID          id-oc-challengeResponseStock}

ctmPrimaryLocation OBJECT-CLASS ::= {
MUST CONTAIN    {locationCommonIdentifier}
MAY CONTAIN         -- at least one of the three following attributes
                {scpLocation |  -- could be used to address the SCFmm visited
                sdpLocation |   -- could be used to get a roaming number
                routingAddress} -- could be used to route the call in white ISUP
ID          id-oc-ctmPrimaryLocation}
```

```
ctmUser  OBJECT-CLASS ::= {
    MUST CONTAIN    {   ctmIdentity}
    MAY CONTAIN {   ctmNumber|
                    description|
                    radioSpecificEnvelop|
                    homeSpecificServices |
                    seeAlso|
                    serviceFeatures|
                    dectSessionSecurityInfo}
    ID              id-oc-ctmUser}

ctmUserAlias  OBJECT-CLASS ::= {
    SUBCLASS OF     {alias}
    MUST CONTAIN    {ctmNumber}
    ID          id-oc-ctmUserAlias}

customer  OBJECT-CLASS ::= {
    SUBCLASS OF {person}
    MAY CONTAIN {member}
    ID          id-oc-customer}

dUA OBJECT-CLASS ::= {
    SUBCLASS OF {applicationEntity}
    ID      id-oc-dUA}

lineIdentificationServices OBJECT-CLASS ::= {
    SUBCLASS OF     {supplementaryService}
    MAY CONTAIN {perCallBasis}
    ID          id-oc-lineIdentificationServices}

roamingNumberPool OBJECT-CLASS ::={
KIND            auxiliary
MUST CONTAIN    {assignmentTable}
MAY CONTAIN         {maxtime|
                randomAssigned}
ID          id-oc-roamingNumberPool}

rsKsStock OBJECT-CLASS ::= {
    SUBCLASS OF     {tokenStock}
    MUST CONTAIN    {rsKs}
    ID          id-oc-rsKsStock}

securityUserInfo OBJECT-CLASS ::={
MUST CONTAIN    {securityFacilityId|
                secretKey|
                identifierList}
MAY CONTAIN         {bindLevelIfOK|
                currentList|
                failureCounter|
                maxAttempts}
ID          id-oc-securityUserInfo }

serviceProvider  OBJECT-CLASS ::= {
    SUBCLASS OF     {organization}
    MUST CONTAIN    {serviceProviderCode}
    MAY CONTAIN {partnersList}
    ID          id-oc-serviceProvider}

supplementaryService OBJECT-CLASS ::= {
KIND            abstract
    MUST CONTAIN    {supplServId |
                supplServiceStatus}
    MAY CONTAIN {description |
                name}
    ID          id-oc-supplementaryService}

temporaryUser  OBJECT-CLASS ::= {
    MUST CONTAIN    {localUserIdentifier|
                ctmIdentityList |
                temporaryLocation}
    MAY CONTAIN {
                radioSpecificEnvelop|
                dectSessionSecurityInfo|
                description|
                seeAlso}
    ID          id-oc-temporaryUser}

temporaryUserAlias  OBJECT-CLASS ::= {
    SUBCLASS OF     {alias}
    MUST CONTAIN    {registrationIdentifier}
    ID          id-oc-temporaryUserAlias}
```

```
tokensStock OBJECT-CLASS ::={
KIND            abstract
MUST CONTAIN        {stockId}
MAY CONTAIN         {source|
                            sizeOfRestocking}
ID          id-oc-tokensStock}

-- ATTRIBUTES
assignmentTable ATTRIBUTE ::= {
WITH SYNTAX     NumericString (SIZE(1..ub-international-isdn-number))
ID          id-at-assignmentTable}

bindLevelIfOK ATTRIBUTE ::= {
WITH SYNTAX     AuthenticationLevel
SINGLE VALUE    TRUE
ID      id-at-bindLevelIfOK}

challengeResponse ATTRIBUTE ::= {
WITH SYNTAX         ThreePartsMessage
ID          id-at-challengeResponse}

ctmIdentity ATTRIBUTE ::= {
    WITH SYNTAX         NumericString (SIZE (1..ub-international-isdn-number))
    EQUALITY MATCHING RULE      numericStringMatch
    SUBSTRINGS MATCHING RULE    reversePrefixMatch
    SINGLE VALUE            TRUE
    ID                  id-at-ctmIdentity}

ctmIdentityList  ATTRIBUTE ::= {    -- this attribute is multivalued to take into account
                    --  multi profile DECT terminals
    WITH SYNTAX         NumericString (SIZE (1..ub-international-isdn-number))
    EQUALITY MATCHING RULE      numericStringMatch
    SUBSTRINGS MATCHING RULE    reversePrefixMatch
    ID                  id-at-ctmIdentityList}

ctmIdentityPrefix   ATTRIBUTE ::= {
    WITH SYNTAX         NumericString (SIZE (1..4))     -- E.212 prefix
    EQUALITY MATCHING RULE  numericStringMatch
    SINGLE VALUE        TRUE
    ID              id-at-ctmIdentityPrefix}

ctmNumber  ATTRIBUTE ::= {
    WITH SYNTAX         NumericString (SIZE (1..ub-international-isdn-number))
    EQUALITY MATCHING RULE      numericStringMatch
    SUBSTRINGS MATCHING RULE    reversePrefixMatch
    SINGLE VALUE            TRUE
    ID                  id-at-ctmNumber}

ctmNumberPrefix  ATTRIBUTE ::= {
    WITH SYNTAX         NumericString (SIZE (1..5))     -- E.164 prefix
    EQUALITY MATCHING RULE  numericStringMatch
    SINGLE VALUE        TRUE
    ID              id-at-ctmNumberPrefix}

currentList  ATTRIBUTE ::= {
    WITH SYNTAX         BIT STRING,
    EQUALITY MATCHING RULE  bitStringMatch
    ID              id-at-currentList}

dectSessionSecurityInfo     ATTRIBUTE ::= {
    WITH SYNTAX     DectSessionSecurityInfo
    ID          id-at-dectSessionSecurityInfo}

failureCounter ATTRIBUTE ::= {
WITH SYNTAX             INTEGER
ORDERING MATCHING RULE      integerOrderingMatch
SINGLE VALUE                TRUE
ID                  id-at-failureCounter}

favoriteLanguage ATTRIBUTE ::= {
WITH SYNTAX             LanguageContextSyntax
EQUALITY MATCHING RULE      octetStringMatch
SINGLE VALUE                TRUE
ID                  id-at-favoriteLanguage}

forwardedToNumber  ATTRIBUTE ::= {
    WITH SYNTAX     NumericString (SIZE(1..ub-international-isdn-number))
    EQUALITY MATCHING RULE      numericStringMatch
    SUBSTRINGS MATCHING RULE    reversePrefixMatch
    ID                  id-at-forwardedToNumber}
```

```
homeSpecificServices        ATTRIBUTE ::={
WITH SYNTAX         HomeSpecificMark
ID              id-at-homeSpecificServices}

identifierList ATTRIBUTE ::= {
WITH SYNTAX SEQUENCE{
conformMethodIdentifier [1] MethodIdentifier, -- e.g. time window check
fillMethodIdentifier        [2] MethodIdentifier-- e.g. generate a random of required size,
oneToOneAlgorithm       [3] AlgorithmIdentifier
                    -- e.g. A11 and A12, output RES from RS,RAND
oneToTwoAlgorithm       [4] AlgorithmIdentifier }
                    -- e.g DECT algorithm output RES,SDK from RS,RAND
SINGLE VALUE            TRUE
ID                              id-at-identifierList}

localUserIdentifier  ATTRIBUTE ::= {
    WITH SYNTAX             NumericString(SIZE(1..ub-international-isdn-number))
    EQUALITY MATCHING RULE      numericStringMatch
    SUBSTRINGS MATCHING RULE    reversePrefixMatch
    SINGLE VALUE            TRUE
    ID                  id-at-localUserIdentifier}

locationCommonIdentifier        ATTRIBUTE ::=
    WITH SYNTAX             NumericString(SIZE(1)) -- to be fixed at value "1"
    EQUALITY MATCHING RULE      numericStringMatch
    SINGLE VALUE            TRUE
    ID                  id-at-locationCommonIdentifier}

maxAttempts ATTRIBUTE ::= {
WITH SYNTAX             INTEGER
ORDERING MATCHING RULE      integerOrderingMatch
SINGLE VALUE            TRUE
ID                  id-at-maxAttempts}

maxtime ATTRIBUTE ::= {
WITH SYNTAX     INTEGER
SINGLE VALUE    TRUE
ID      id-at-maxtime}

partnersList  ATTRIBUTE ::= {
    WITH SYNTAX         DistinguishedName
    EQUALITY MATCHING RULE  distinguishedNameMatch
    ID              id-at-partnersList}

perCallBasis ATTRIBUTE ::= {
WITH SYNTAX     BOOLEAN
EQUALITY MATHING RULE   booleanMatch
SINGLE VALUE        TRUE
ID          id-at-perCallBasis}

radioSpecificEnvelop ATTRIBUTE ::={
WITH SYNTAX     BIT STRING
SINGLE VALUE    TRUE
ID      id-at-radioSpecificEnvelop}

randomAssigned ATTRIBUTE ::= {
WITH SYNTAX BOOLEAN
SINGLE VALUE    TRUE
ID      id-at-randomAssigned}

registrationIdentifier  ATTRIBUTE ::= {
    WITH SYNTAX             INTEGER
    EQUALITY MATCHING RULE      integerMatch
    SINGLE VALUE            TRUE
    ID                  id-at-registrationIdentifier}

routingAddress  ATTRIBUTE ::= {
    WITH SYNTAX     NumericString (SIZE (1..ub-international-isdn-number))
    EQUALITY MATCHING RULE      numericStringMatch
    SUBSTRINGS MATCHING RULE    reversePrefixMatch
    ID                  id-at-routingAddress}

rsKs ATTRIBUTE ::= {
WITH SYNTAX         TwoPartsMessage
ID          id-at-rsKs}

scpLocation  ATTRIBUTE ::= {
    WITH SYNTAX     EntityAddress
    EQUALITY MATCHING RULE          octetStringMatch
    ID                      id-at-scpLocation}

sdpLocation  ATTRIBUTE ::= {
    WITH SYNTAX     EntityAddress
    EQUALITY MATCHING RULE          octetStringMatch
    ID                        id-at-sdpLocation}

secretKey  ATTRIBUTE ::= {
    WITH SYNTAX     BIT STRING(SIZE(lb-secretKey..ub-secretKey))
    SINGLE VALUE    TRUE
    ID          id-at-secretKey}

securityFacilityId ATTRIBUTE ::= {
    WITH SYNTAX             SF-CODE
    EQUALITY MATCHING RULE      objectIdentifierMatch
    SINGLE VALUE            TRUE
    ID                  id-at-securityFacilityId}

serviceFeatures  ATTRIBUTE ::= {
    WITH SYNTAX     ServiceFeatures
    SINGLE VALUE    TRUE
    ID          id-at-serviceFeatures}

serviceProviderCode  ATTRIBUTE ::= {
    WITH SYNTAX         NumericString (SIZE (1..ub-serviceProviderCode))
                            -- MNC in E.212
    EQUALITY MATCHING RULE  numericStringMatch
    SINGLE VALUE        TRUE
    ID              id-at-serviceProviderCode}

sizeOfRestocking ATTRIBUTE ::= {
WITH SYNTAX             INTEGER
ORDERING MATCHING RULE      integerOrderingMatch
SINGLE VALUE                TRUE
ID                  id-at-sizeOfRestocking }

source ATTRIBUTE ::={
WITH SYNTAX     SourceType
SINGLE VALUE    TRUE
ID      id-at-source}

stockId  ATTRIBUTE ::= {
    WITH SYNTAX         DT-Code
    EQUALITY MATCHING RULE  objectIdentifierMatch
    SINGLE VALUE        TRUE
    ID              id-at-stockId}

supplServiceStatus  ATTRIBUTE ::= {
    WITH SYNTAX     SupplServiceStatus
    SINGLE VALUE    TRUE
    ID          id-at-supplServiceStatus}

supplServId ATTRIBUTE ::= {
    WITH SYNTAX         SS-CODE
    EQUALITY MATCHING RULE  objectIdentifierMatch
    SINGLE VALUE        TRUE
    ID              id-at-supplServId}

temporaryIdentityPrefix ATTRIBUTE ::= {
    WITH SYNTAX             NumericString (SIZE (1..4))
    EQUALITY MATCHING RULE      numericStringMatch
    SINGLE VALUE            TRUE
    ID                  id-at-temporaryIdentityPrefix}

temporaryLocation ATTRIBUTE ::= {
    WITH SYNTAX         NumericString  (SIZE (1..ub-international-isdn-number))
    EQUALITY MATCHING RULE  numericStringMatch
    SUBSTRINGS MATCHING RULE    reversePrefixMatch
    SINGLE VALUE                        TRUE
    ID                                          id-at-temporaryLocation}

terminalID ATTRIBUTE ::= {
    WITH SYNTAX             BIT STRING(SIZE(36))
    EQUALITY MATCHING RULE      bitStringMatch
    SINGLE VALUE            TRUE
    ID                  id-at-terminalID}
typesOfNotifications  ATTRIBUTE ::= {
    WITH SYNTAX     TypesOfNotification
    SINGLE VALUE    TRUE
    ID          id-at-typesOfNotifications}

userCredit ATTRIBUTE ::= {
    WITH SYNTAX             INTEGER (1..ub-userCredit)
    ORDERING MATCHING RULE  integerOrderingMatch
    SINGLE VALUE            TRUE
    ID                  id-at-userCredit}
```

```
-- CONTEXTS

assignmentContext CONTEXT ::={
WITH SYNTAX CHOICE {
    directoryUserName   DistinguishedName,
    genericNumber       GenericNumber}
ID          id-avc-assignmentContext}

-- METHODS

ConformCredentials METHOD ::={
SPECIFIC-INPUT  TwoPartMessage
-- see the definition of this type below
SPECIFIC-OUTPUT BOOLEAN
        -- to indicated the success of the verification
BEHAVIOUR   "This method performs the following actions on the entry identified by the execute
argument, this entry would be of class genericSecurityUserInfo:
-   verify with an embedded conformance algorithm that messageData value of TwoPartMessage is no
replay (RAND is in the current time window and the associated RS is not in the list of the current
time windows currentList).
-   add RAND to time windows list currentList.
-   return TRUE if no replay,
-   otherwise return FALSE
"
ID              id-mt-dectConformCredentials}

fillSecurityTokens METHOD ::={
SPECIFIC-INPUT  INTEGER -- N number of value to be computed
SPECIFIC-OUTPUT SEQUENCE OF ThreePartMessage
BEHAVIOUR   "This method performs the following actions on the entry identified by the execute
argument, this entry shall be of object class (or subclass) genericSecurityUserInfo:
- read the secretKey attribute and oneToTwoAlgorithm attribute
- repeat N times
        - fill the first BIT STRING field with a random value
        - apply the oneToTwoAlgorithm cryptographic algorithms to compute
        the second and third field.
- return N ThreePartMessage values
"
ID      id-mt-fillSecurityTokens}

findAndRelease METHOD ::={
INPUT ATTRIBUTE assignmentTable
SPECIFIC-OUTPUT DistinguishedName   -- The DN of the user to which
-- the selected value was
-- temporarily assigned
BEHAVIOUR   "This method performs the following actions on the entry identified by the execute
argument:
1) find the value of the assignmentTable
attribute which is equal to the one received in the
input-assertions element of the execute argument.
2) remove from the DIB all its associated context values.
3) if this value was associated with valid temporal
context values and an assignmentContext Value,
return the associated assignmentContext value (user DN).
ID  id-mt-findAndRelease}

provideTokens METHOD ::={
SPECIFIC-INPUT  INTEGER, -- how many tokens are requested (NOfRT)
                OBJECT IDENTIFIER   -- oid of the attribute (tokens)
SPECIFIC-OUTPUT     ATTRIBUTE --attribute selected as input (tokens)
BEHAVIOUR   "This method performs the following actions on the entry (thisEntry would be a variable
with the DN value of this entry) identified by the execute argument:
1)  if the attribute sizeOfRestocking doesn't exist in the entry, define a variable MAXNT with a
default value (e.g. 10) else MAXNT takes the value of the attribute sizeOfRestocking
2)  verify that NofRT is inferior or egal to MAXNT
(return an "execute error" if NofRT value is superior to MAXNT)
3)  read the attribute of the entry which has the selected oid and count the number of values (0 if
empty) and put the result in a variable N
(return "execute error if the attribute doesn't exist)
4)  read the source attribute in the entry
(return "execute error" error if source doesn't exist)
5)  if N is inferior to NOfRT and the DN of source indicates an entry of class or subclass
tokenStock :
5a) bind anonymous with the DSA which contains the entry defined by the address field of source
5b) execute the method provideTokens on the entry with MAXNT as value of the specific-input
5c) if none error is returned , modify the tokens attribute by adding the resulted values
    6)  if N is inferior to NOfRT and the DN of source indicates an entry of class or subclass
securityUserInformation
6a) execute the method defined by fillMethodIdentifier field value on the entry defined by the DN
with MAXNT as specific input
6b) if none error is returned ,modify the tokens attribute by adding the resulted values
    7)  read the tokens attribute
8)  define a variable "toBeReturned" with NofRT values of tokens attribute and a variable "toBeKept"
```

```
with remainder values
    9)  remove tokens attribute
    10) modify tokens attribute by adding the "toBeKept" values
    11) return the "toBeReturned" values
"
ID              id-mt-provideTokens}

selectAndAssign METHOD ::={
SPECIFIC-INPUT  DistinguishedName
                    -- The DN of the user to which
                    -- the selected value is temporarily assigned
OUTPUT ATTRIBUTES   assignmentTable
BEHAVIOUR           "This method performs the following actions on the entry identified by the
execute argument:
1)  selects a value of the assignmentTable attribute which has no context associated with it or is
associated with an expired temporal context.
2)  adds an assignmentContextValue equal to the specific input to the selected value.
3)  adds a temporal context value so that the selected value becomes irrelevant after maxtime units
of time.
4) return the selected value without context values.
"
ID              id-mt-selectAndAssign}

verifyCredentials METHOD ::={
SPECIFIC-INPUT  TwoPartMessage
-- see the definition of this type below
SPECIFIC-OUTPUT BOOLEAN
        -- to indicated the success of the verification
BEHAVIOUR   "This method performs the following actions on the entry identified by the execute
argument, this entry would be of class genericSecurityUserInfo:
1) if maxattempts is present, verify that failureCounter is inferior to its value
2)  read the value of identifierList attribute
(return "bad format entry" if failure)
3) if conformMethodIdentifier is NULL go to step 5)
4)  run conformMethodIdentifier method on TwopartMessage provided as specific input
(return a "badconformance" error if the execution fails or if the result is false)
5)  run the oneToOneAlgorithm on the messageData bit string to get an expected certificationCode bit
string
6)  return TRUE if the expected and provided certificationCode values matched,
7)  otherwise if failureCounter is present, increment it
8)  return FALSE
"
ID              id-mt-verifyCredentials}

-- NAME FORMS
adminUnitNameForm1  NAME-FORM ::= {
    NAMES       administrativeUnit
    WITH ATTRIBUTES {ctmIdentityPrefix}
    ID          id-nf-adminUnitNameForm1}

adminUnitNameForm2  NAME-FORM ::= {
    NAMES       administrativeUnit
    WITH ATTRIBUTES {ctmNumberPrefix}
    ID          id-nf-adminUnitNameForm2}

adminUnitNameForm3  NAME-FORM ::= {
    NAMES       administrativeUnit
    WITH ATTRIBUTES {temporaryIdentityPrefix}
    ID          id-nf-adminUnitNameForm3}

callForwardingNameForm  NAME-FORM ::= {
    NAMES       callForwarding
    WITH ATTRIBUTES {supplServId}
    ID          id-nf-callForwardingNameForm}

challengeResponseStockNameForm NAME-FORM ::={
    NAMES       challengeResponseStock
    WITH ATTRIBUTES {stockId}
    ID          id-nf-challengeResponseStockNameForm}

ctmUserAliasNameForm  NAME-FORM ::= {
    NAMES       ctmUserAlias
    WITH ATTRIBUTES {ctmNumber}
    ID          id-nf-ctmUserAliasNameForm}

ctmUserNameForm  NAME-FORM ::= {
    NAMES       ctmUser
    WITH ATTRIBUTES {ctmIdentity}
    ID          id-nf-ctmUserNameForm}

customerNameForm  NAME-FORM ::= {
    NAMES       customer
    WITH ATTRIBUTES {commonName}
    ID          id-nf-customerNameForm}

dUANameForm  NAME-FORM ::= {
    NAMES       dUA
    WITH ATTRIBUTES {commonName}
    ID          id-nf-dUANameForm}

lineIdentificationServicesNameForm  NAME-FORM ::= {
    NAMES       lineIdentificationServices
    WITH ATTRIBUTES {supplServId}
    ID          id-nf-lineIdentificationServicesNameForm}

primaryLocationNameForm NAME-FORM ::={
    NAMES       primaryLocation
    WITH ATTRIBUTES {locationCommonId}
    ID          id-nf-primaryLocationNameForm}

rsKsStockNameForm NAME-FORM ::={
    NAMES       rsKsStock
    WITH ATTRIBUTES {stockId}
    ID          id-nf-rsKsStockNameForm}

securityUserInfoNameForm NAME-FORM ::={
    NAMES       securityUserInfo
    WITH ATTRIBUTES {securityFacilityId}
    ID          id-nf-securityUserInfoNameForm}

serviceProviderNameForm  NAME-FORM ::= {
    NAMES           serviceProvider
    WITH ATTRIBUTES     {serviceProviderCode}
    ID              id-nf-serviceProviderNameForm}

temporaryUserAliasNameForm  NAME-FORM ::= {
    NAMES           temporaryUserAlias
    WITH ATTRIBUTES     {registrationIdentifier}
    ID              id-nf-temporaryUserAliasNameForm}

temporaryUserNameForm  NAME-FORM ::= {
    NAMES           temporaryUser
    WITH ATTRIBUTES     {localUserIdentifier}
    ID              id-nf-temporaryUserNameForm}


-- STRUCTURE RULES
sr1 STRUCTURE-RULE ::= {
    NAME FORM   countryNameForm
    ID      1}

sr2 STRUCTURE-RULE::= {
    NAME FORM       orgNameForm
    SUPERIOR RULES  {sr1}
    ID          2}

sr3 STRUCTURE-RULE::= {
    NAME FORM       serviceProviderNameForm
    SUPERIOR RULES  {sr1}
    ID          3}

sr4 STRUCTURE-RULE::= {
    NAME FORM       dSANameForm
    SUPERIOR RULES  {sr2 | sr3}
    ID          4}

sr5 STRUCTURE-RULE::= {
    NAME FORM       dUANameForm
    SUPERIOR RULES  {sr2 | sr3}
    ID          5}

sr6 STRUCTURE-RULE::= {
    NAME FORM       customerNameForm
    SUPERIOR RULES  {sr2}
    ID          6}

sr7 STRUCTURE-RULE::= {
    NAME FORM       adminUnitNameForm1
    SUPERIOR RULES  {sr3}
    ID          7}

sr8 STRUCTURE-RULE::= {
    NAME FORM       adminUnitNameForm2
    SUPERIOR RULES  {sr3}
    ID          8}
```

```
sr9 STRUCTURE-RULE ::= {
    NAME FORM       adminUnitNameForm3
    SUPERIOR RULES  {sr3}
    ID          9}

sr10 STRUCTURE-RULE::= {
    NAME FORM       ctmUserNameForm
    SUPERIOR RULES  {sr7}
    ID          10}

sr11 STRUCTURE-RULE::= {
    NAME FORM       temporaryUserNameForm
    SUPERIOR RULES  {sr9}
    ID          11}

sr12 STRUCTURE-RULE::= {
    NAME FORM       ctmUserAliasNameForm
    SUPERIOR RULES  {sr8}
    ID          12}

sr13 STRUCTURE-RULE::= {
    NAME FORM       primaryLocationNameForm
    SUPERIOR RULES  {sr10}
    ID          13}

sr14 STRUCTURE-RULE::= {
    NAME FORM       securityUserInfoNameForm
    SUPERIOR RULES  {sr10}
    ID          14}

sr15    STRUCTURE-RULE ::= {
    NAME FORM       temporaryUserAliasNameForm
    SUPERIOR RULES  {sr10}
    ID          15}

sr16 STRUCTURE-RULE ::= {
    NAME FORM   challengeResponseStockNameForm
    SUPERIOR RULES  {sr10|sr11}
    ID          16}

sr17 STRUCTURE-RULE ::= {
    NAME FORM   rsKsStockNameForm
    SUPERIOR RULES  {sr10}
    ID          17}

sr20 STRUCTURE-RULE::= {
    NAME FORM       callForwardingNameForm
    SUPERIOR RULES  {sr10}
    ID          20}

sr21 STRUCTURE-RULE::= {
    NAME FORM       lineIdentificationServicesNameForm
    SUPERIOR RULES  {sr10}
    ID          21}

-- TYPES
-- AlgorithmIdentifier could be imported from Rec. X.509
AlgorithmIdentifier ::= SEQUENCE {
algorithm   ALGORITHM.&id ({SupportedAlgorithms}),
parameters  ALGORITHM.&Type({SupportedAlgorithms}{@algorithm}) OPTIONAL}

DectSessionSecurityInfo ::= SEQUENCE {
    rs  [0] BIT STRING,
    ks  [1] BIT STRING}

DT-Code ::= OBJECT IDENTIFIER

EntityAddress ::= OCTET STRING (SIZE(3..maxEntityAddressSize))
    -- Internal coding as follows:
    -- Octet 1 contains the Numbering plan and the encoding scheme encoded according
    -- to Q.713
    -- Octet 2 contains Nature of address encoded as in Q.713
    -- Subsequent octETS contain the actual GT digits encoded according the
    -- encoding scheme)

HomeSpecificMark ::= SEQUENCE {
    serviceIdentifier       ServiceFeaturesIdentifier,
    scpAddress      EntityAddress OPTIONAL,
    ...}

MethodIdentifier ::= SEQUENCE {
methodid            METHOD.&id ({SupportedMethods}),
inputAttributes     METHOD.&InputAttributes ({SupportedMethods}{@method}) OPTIONAL,
specific-Input      METHOD.&SpecificInput ({SupportedMethods}{@method}) OPTIONAL}

SecurityCredentialsArg ::= {
name            DistinguishedName,
securityFacilityId      SF-CODE,
challengeResponseUsr    TwoPartsMessage,
challengeNetwork        BIT STRING OPTIONAL}

SecurityCredentialsResult ::= {
challengeResponseNetwork    ThreePartsMessage   OPTIONAL}

SecurityFacilityCredentials ::= SEQUENCE {
secuCredArg SecurityCredentialsArg,
secuCredResult  SecurityCredentialsResult}

ServiceFeatures ::= BIT STRING {
    profileModification (0),
    profileInterrogation (1)}

ServiceFeaturesIdentifier   ::= CHOICE {
    local   INTEGER(0..127),
    global  OBJECT IDENTIFIER}

SF-CODE ::= OBJECT IDENTIFIER

SourceType ::=DistinguishedName

SS-CODE ::= OBJECT IDENTIFIER

SupplServiceStatus ::= BIT STRING {
    activated (0),
    registered (1),
    provisioned (2)}

TypesOfNotification ::= BIT STRING {
    callingUserWithForwardedToNumber (1)}

TwoPartsMessage ::= SEQUENCE {
messageData [1] BIT STRING,
certificationCode   [2] BIT STRING}

ThreePartsMessage ::= SEQUENCE {
challenge   [1] BIT STRING,
response    [2] BIT STRING,
cipherkey   [3] BIT STRING}

-- VALUES
challenge-response-abstract-syntax      ABSTRACT-SYNTAX ::={
SecurityFacilityCredentials
IDENTIFIED BY                   id-as-challenge-response}

ctmDITContextUse1 DIT-CONTEXT-USE-RULE ::= {
ATTRIBUTE TYPE      id-at-localUserIdentifier
MANDATORY CONTEXTS  {temporalContext }}

ctmDITContextUse2 DIT-CONTEXT-USE-RULE::= {
ATTRIBUTE TYPE      id-at-temporaryLocation
MANDATORY CONTEXTS  {temporalContext }}

ctmDITContextUse3 DIT-CONTEXT-USE-RULE::= {
ATTRIBUTE TYPE      id-at-supplServiceStatus
MANDATORY CONTEXTS  {temporalContext }}


ctmDITContextUse4 DIT-CONTEXT-USE-RULE::= {
ATTRIBUTE TYPE      id-at-roamingNumberPool
MANDATORY CONTEXTS  {temporalContext | assignmentContext}}

ctmDITContextUse5 DIT-CONTEXT-USE-RULE::= {
ATTRIBUTE TYPE      id-at-primaryLocation
MANDATORY CONTEXTS  {temporalContext }}

ctmDITContextUse6 DIT-CONTEXT-USE-RULE::= {
ATTRIBUTE TYPE      id-at-registrationIdentifier
MANDATORY CONTEXTS  {temporalContext }}

ctmDITContextUse7 DIT-CONTEXT-USE-RULE::= {
ATTRIBUTE TYPE      id-at-ctmIdentity
MANDATORY CONTEXTS  {temporalContext }}
```

```
ctmDITContextUse8 DIT-CONTEXT-USE-RULE::= {
ATTRIBUTE TYPE      id-at-dectSessionSecurityInfo
MANDATORY CONTEXTS  {assignmentContext }}

ctmDITContextUse9 DIT-CONTEXT-USE-RULE::= {
ATTRIBUTE TYPE      id-at-currentList
MANDATORY CONTEXTS  {temporalContext }}

lb-secretKey  INTEGER ::= 32

roamingNumberRule METHOD-USE-RULE ::= {
OBJECT CLASS TYPE id-oc-roamingNumberPool
MANDATORY METHODS {findAndRelease| selectAndAssign}}

scf-sdf-ctm-context     APPLICATION-CONTEXT ::= {
    CONTRACT                scf-sdf-contract
    TERMINATION         basic
    ABSTRACT SYNTAXES       {dialogue-abstract-syntax |
                        challenge-response-abstract-syntax |
                        scf-sdf-as}
    APPLICATION CONTEXT NAME    id-ac-scfsdfmobile}

securityUserInfoRule METHOD-USE-RULE ::= {
OBJECT CLASS TYPE       id-oc-securityUserInfo
MANDATORY METHODS   {verifyCredentials| fillSecurityTokens|conformCrendentials}}

tokensRule METHOD-USE-RULE ::= {
OBJECT CLASS TYPE       id-oc-challengeResponseStock
MANDATORY METHODS   {provideTokens}}

maxEntityAddressSize INTEGER ::= 10

ub-international-isdn-number  INTEGER ::= 15


ub-secretKey  INTEGER ::= 128

ub-serviceProviderCode  INTEGER ::= 3

SupportedAttributes ATTRIBUTE ::={
aliasedEntryName|
objectclass|
assignmentTable|
bindLevelIfOK|
ctmIdentity|
ctmIdentityList|
ctmIdentityPrefix|
ctmNumber|
ctmNumberPrefix|
currentList|
dectSessionSecurityInfo|
failureCounter|
favoriteLanguage|
forwardedToNumber|
identifierList|
homeSpecificServices|
localUserIdentifier|
locationCommonIdentifier|
maxtime|
partnersList|
perCallBasis|
radioSpecificEnvelop|
randomAssigned|
registrationIdentifier|
routingAddress|
rsKs|
scpLocation|
sdpLocation|
secretKey|
securityFacilityId|
serviceFeatures|
serviceProviderCode|
sizeOfRestocking|
source|
stockId|
subscriptionData|
supplServiceStatus|
supplServId|
temporaryIdentityPrefix|
temporaryLocation|
terminalID|
tokens|
typesOfNotifications|
userCredit , ...}

SupportedContexts CONTEXT ::= {
assignmentContext|
temporalContext|
basicServiceContext|
lineIdentityContext ,...}

SupportedMethods METHOD ::= {
conformCredentials|
fillSecurityTokens|
findAndRelease|
provideTokens|
selectAndAssign|
verifyCredentials,...}

-- object identifier assignments
id-ac   OBJECT IDENTIFIER ::={inDomainId in-ctm (3) ac (3)}
id-at   OBJECT IDENTIFIER ::={inDomainId in-ctm (3) at (4)}
id-as   OBJECT IDENTIFIER ::={inDomainId in-ctm (3) as (5)}
id-oc   OBJECT IDENTIFIER ::={inDomainId in-ctm (3) oc (6)}
id-nf   OBJECT IDENTIFIER ::={inDomainId in-ctm (3) nf (15)}

id-mt   OBJECT IDENTIFIER ::={inDomainId in-ctm (3) mt (40)}
id-ss OBJECT IDENTIFIER ::= { inDomainId in-ctm (3) ss (41)}

id-sf OBJECT IDENTIFIER ::= { inDomainId in-ctm (3) sf (50)}
id-dt OBJECT IDENTIFIER ::= { inDomainId in-ctm (3) dt (51)}

-- Supplementary Service id --
-- if  for isdn Supplementary Services oids are already defined in Q730 series
-- they will be used instead

id-ss-clip SS-CODE ::= {id-ss li(1) 3}
id-ss-clir SS-CODE ::= {id-ss li(1) 4}
id-ss-colp SS-CODE ::= {id-ss li(1) 5}
id-ss-colr SS-CODE ::= {id-ss li(1) 6}
id-ss-mcid SS-CODE ::= {id-ss li(1) 7}
id-ss-cfb SS-CODE ::= {id-ss cf (2) 2}
id-ss-cfnr SS-CODE ::= {id-ss cf(2) 3}
id-ss-cfu SS-CODE ::= {id-ss cf(2) 4}

id-ss-cfnrc SS-CODE ::= {id-ss cf(2) 5}

-- Security Facility id
id-sf-pwd SF-CODE ::= {id-sf pwd(1)}
id-sf-challengeResponse SF-CODE ::=  {id-sf common (2)}
id-sf-onAirSubscription SF-CODE ::=  {id-sf subscription(3)}

-- Disposable Stock id
id-dt-dect-stage1   DT-CODE ::= {id-dt authentication(1) dect(1) stage1(1)}
id-dt-challengeResponse     DT-CODE ::= {id-dt authentication(1) common(2)}

-- local phase 1 service feature identifier
-- the following local identifiers indicate a home specific feature for incoming call
lid-hsf-ic1 ServiceFeaturesIdentifier ::= {local:ic-o1} --for the originating network during set-up
phase
lid-hsf-ic2 ServiceFeaturesIdentifier ::= {local:ic-v1} -- for a visited network during set-up phase
lid-hsf-ic3 ServiceFeaturesIdentifier ::= {local:ic-of} -- for an originating on setup failure
lid-hsf-ic4 ServiceFeaturesIdentifier ::= {local:ic-vf} -- for a visited on setup failure

-- the following local identifiers indicate a home specific feature for outgoing call
lid-hsf-og1 ServiceFeaturesIdentifier ::= {local:oc-v1} -- for the visited network during set-up
phase
lid-hsf-og3 ServiceFeaturesIdentifier ::= {local:oc-vf} -- for the visited network on set-up failure

ic-o1 INTEGER ::=  1
ic-v1 INTEGER ::= 2
ic-of INTEGER ::= 3
ic-vf INTEGER ::= 4
oc-v1 INTEGER ::= 11
oc-vf INTEGER ::= 12

-- Application Context id
id-ac-scfsdfmobile OBJECT IDENTIFIER    ::= {id-ac 0}

-- attributes id
id-at-assignmentTable OBJECT IDENTIFIER ::= {id-at 20}
id-at-bindLevelIfOK OBJECT IDENTIFIER   ::= {id-at 23}
id-at-ctmIdentity OBJECT IDENTIFIER ::= {id-at 24}
id-at-ctmIdentityList OBJECT IDENTIFIER ::= {id-at 25}
id-at-ctmIdentityPrefix OBJECT IDENTIFIER   ::= {id-at 26}
```

```
id-at-ctmNumber OBJECT IDENTIFIER   ::= {id-at 27}
id-at-ctmNumberPrefix OBJECT IDENTIFIER ::= {id-at 28}
id-at-currentList OBJECT IDENTIFIER ::= {id-at 29}

id-at-dectSessionSecurityInfo OBJECT IDENTIFIER ::= {id-at 30}
id-at-failureCounter OBJECT IDENTIFIER  ::= {id-at 31}
id-at-favoriteLanguage OBJECT IDENTIFIER    ::= {id-at 32}
id-at-forwardedToNumber OBJECT IDENTIFIER   ::= {id-at 33}
id-at-identifierList OBJECT IDENTIFIER  ::= {id-at 34}
id-at-homeSpecificServices OBJECT IDENTIFIER    ::= {id-at 35}
id-at-localUserIdentifier OBJECT IDENTIFIER ::= {id-at 36}
id-at-locationCommonIdentifier OBJECT IDENTIFIER    ::= {id-at 37}
id-at-maxtime OBJECT IDENTIFIER ::= {id-at 38}

id-at-partnersList OBJECT IDENTIFIER    ::= {id-at 41}
id-at-perCallBasis OBJECT IDENTIFIER    ::= {id-at 42}
id-at-randomAssigned OBJECT IDENTIFIER  ::= {id-at 43}
id-at-registrationIdentifier OBJECT IDENTIFIER  ::= {id-at 44}
id-at-routingAddress OBJECT IDENTIFIER  ::= {id-at 45}
id-at-scpLocation OBJECT IDENTIFIER ::= {id-at 46}
id-at-sdpLocation OBJECT IDENTIFIER ::= {id-at 47}
id-at-secretKey OBJECT IDENTIFIER   ::= {id-at 48}
id-at-securityFacilityId OBJECT IDENTIFIER  ::= {id-at 49}

id-at-serviceFeatures OBJECT IDENTIFIER ::= {id-at 50}
id-at-serviceProviderCode OBJECT IDENTIFIER ::= {id-at 51}
id-at-sizeOfRestocking OBJECT IDENTIFIER    ::= {id-at 52}
id-at-source OBJECT IDENTIFIER  ::= {id-at 53}
id-at-stockId OBJECT IDENTIFIER ::= {id-at 54}
id-at-subscriptionData OBJECT IDENTIFIER    ::= {id-at 55}
id-at-supplServiceStatus OBJECT IDENTIFIER  ::= {id-at 56}
id-at-supplServId OBJECT IDENTIFIER ::= {id-at 57}
id-at-temporaryIdentityPrefix OBJECT IDENTIFIER ::= {id-at 58}
id-at-temporaryLocation OBJECT IDENTIFIER   ::= {id-at 59}

id-at-terminalID OBJECT IDENTIFIER  ::= {id-at 60}
id-at-tokens OBJECT IDENTIFIER  ::= {id-at 61}
id-at-typesOfNotifications OBJECT IDENTIFIER    ::= {id-at 62}
id-at-userCredit OBJECT IDENTIFIER  ::= {id-at 63}
id-at-rsKs OBJECT IDENTIFIER    ::= {id-at 85}
id-at-radioSpecificEnvelop OBJECT IDENTIFIER    ::= {id-at 86}

-- object classes id
id-oc-administrativeUnit OBJECT IDENTIFIER  ::= {id-oc 20}
id-oc-callForwarding OBJECT IDENTIFIER  ::= {id-oc 21}
id-oc-challengeResponseStock OBJECT IDENTIFIER  ::= {id-oc 22}
id-oc-ctmPrimaryLocation OBJECT IDENTIFIER  ::= {id-oc 23}
id-oc-ctmUser OBJECT IDENTIFIER ::= {id-oc 24}
id-oc-ctmUserAlias OBJECT IDENTIFIER    ::= {id-oc 25}
id-oc-customer OBJECT IDENTIFIER    ::= {id-oc 26}
id-oc-dUA OBJECT IDENTIFIER ::= {id-oc 27}
id-oc-lineIdentificationServices OBJECT IDENTIFIER  ::= {id-oc 28}

id-oc-outgoingRestriction OBJECT IDENTIFIER ::= {id-oc 30}
id-oc-roamingNumberPool OBJECT IDENTIFIER   ::= {id-oc 31}
id-oc-securityUserInfo OBJECT IDENTIFIER    ::= {id-oc 32}
id-oc-serviceProvider OBJECT IDENTIFIER ::= {id-oc 33}
id-oc-supplementaryService OBJECT IDENTIFIER    ::= {id-oc 34}
id-oc-temporaryUser OBJECT IDENTIFIER   ::= {id-oc 35}
id-oc-temporaryUserAlias  OBJECT IDENTIFIER ::= {id-oc 36}
id-oc-tokensStock OBJECT IDENTIFIER ::= {id-oc 37}
id-oc-rsKsStock OBJECT IDENTIFIER   ::= {id-oc 38}


--name forms id

id-nf-serviceProviderNameForm  OBJECT IDENTIFIER ::= {id-nf 20}
id-nf-customerNameForm  OBJECT IDENTIFIER ::= {id-nf 21}
id-nf-adminUnitNameForm1  OBJECT IDENTIFIER ::= {id-nf 22}
id-nf-adminUnitNameForm2  OBJECT IDENTIFIER ::= {id-nf 23}
id-nf-adminUnitNameForm3  OBJECT IDENTIFIER ::= {id-nf 24}
id-nf-ctmUserNameForm  OBJECT IDENTIFIER ::= {id-nf 25}
id-nf-temporaryUserNameForm  OBJECT IDENTIFIER ::= {id-nf 26}
id-nf-ctmUserAliasNameForm  OBJECT IDENTIFIER ::= {id-nf 27}
id-nf-callForwardingNameForm  OBJECT IDENTIFIER ::= {id-nf 28}
id-nf-lineIdentificationServicesNameForm  OBJECT IDENTIFIER ::= {id-nf 29}
id-nf-dUANameForm OBJECT IDENTIFIER ::= {id-nf 30}
id-nf-temporaryUserAliasNameForm OBJECT IDENTIFIER ::= {id-nf 31}
id-nf-securityUserInfoNameForm OBJECT IDENTIFIER ::= {id nf 32}
id-nf-primaryLocationNameForm OBJECT IDENTIFIER ::= {id nf 33}
id-nf-challengeResponseStockNameForm OBJECT IDENTIFIER ::= {id nf 34}
id-nf-rsKsStockNameForm OBJECT IDENTIFIER ::= {id nf 35}
```

```
-- context id

-- method id
id-mt-selectAndAssign OBJECT IDENTIFIER ::= {id-mt 1}
id-mt-findAndRelease OBJECT IDENTIFIER ::= {id-mt 2}
id-mt-provideTokens OBJECT IDENTIFIER ::= {id-mt 3}
id-mt-verifyCredentials OBJECT IDENTIFIER ::= {id-mt 4}
id-mt-fillSecurityTokens OBJECT IDENTIFIER ::= {id-mt 5}
id-mt-dectConformCredentials OBJECT IDENTIFIER ::= {id-mt 6}

-- abstract syntax id
id-as-challenge-response    OBJECT IDENTIFIER ::= {id-as challenge-response (1) version 1(1)}

-- agreement id
ctm-hss-agreement OBJECT IDENTIFIER ::= {itu identified-organization (4) etsi (9) inDomain (1)
                                                 in-ctm (3) interfaces (100) hss-agreement (1)}

shadowing-root OBJECT IDENTIFIER ::= {itu identified-organization (4) etsi (9) inDomain (1)
                                                 in-ctm (3) oc (6)  shadowing (100)}

END
```

# Annex B (informative): Guidance to add information model element

Some elements are not required for the CTM phase 1 information model. Nevertheless, some operators could wish to implement service features as a network specific extension. This annex provides some guidance to define these extensions in the phase 1 information model in order to ensure backward compatibility.

## B.1 Adding an attribute

To enhance an entry by adding a new specific attribute:

The attribute should be defined using the **ATTRIBUTE** class; it should be identified by an **OBJECT IDENTIFIER.**

The supported attributes list must be completed.

The enhanced entries which used this new attribute should be defined as follows:

```
enhancedEntryXXX OBJECT CLASS ::= {
SUBCLASS {entryXXX }
MAY CONTAIN {newattribute]
ID  id-oc-enhancedEntryXXX
```

## B.2 Adding object classes

The adding of a new entry should be done by defining its structure: the **OBJECT CLASS**, the new attributes that could be included.

The place of the entries that would use this object class: **NAME-FORM** and **STRUCTURE-RULE**.

For example, a new supplementary service could be defined:

```
CTM-newSupplementaryservice {xxx}
DEFINITIONS::=

BEGIN

--EXPORTS All; --

IMPORTS
SupportedAttributes,
supplementaryService,
FROM CTM-DataModel { etsi in-domain in-ctm modules(1) data-model(0) version1(0) }
;
ss-newone OBJECT-CLASS ::= {
    SUBCLASS OF         {supplementaryService}
    MUST CONTAIN {ss-attribute1}
    MAY CONTAIN {ss-attribute2}
    ID          id-oc-newone}

ss-attribute1 ATTRIBUTE ::= {
    WITH SYNTAX         SS-TYPE-1
    EQUALITY MATCHING RULE      eg-numericStringMatch
    SUBSTRINGS MATCHING RULE    eg-reversePrefixMatch
    ID                  id-at-ss-attribute1}

ss-attribute2 ATTRIBUTE ::= {
    WITH SYNTAX         SS-TYPE-2
    EQUALITY MATCHING RULE      eg-OctetStringMatch
    ID                  id-at-ss-attribute2}

ss-newoneNameForm NAME-FORM ::={
    NAMES       ss-newone
    WITH ATTRIBUTES {supplServId}
    ID          id-nf-ss-newone}

sr"nn" STRUCTURE-RULE::= {
    NAME FORM       ss-newoneNameForm
    SUPERIOR RULES  {sr10}
    ID      "nn"}

SupportedAttributes ATTRIBUTE ::= {CTM-DataModel.SupportedAttributes |
ss-attribute1|
ss-attribute2}
END
```

# B.3 Prototypes of SCF-SDF requests

## B.3.1 Preamble

In order to compile the ASN.1 modules, some values are fixed arbitrarily (e.g. a country code, a customer CTM identity). These values would be named with the prefix "a-". The ASN.1 module which includes these values is not provided in this document. The real values are provided by the SCF when it makes its request to the SDF.

To simplify, the arguments of the operation only Distinguished names based on CTM identity are described. In order, to use the CTM number only the Relatived Distinguished Names (a-ctmIdentityPrefix and a-ctmIdentity) have to be replaced by a-ctmNumberPrefix and a-ctmNumber.

Since the choice of **localUserIdentifier** attribute value is network specific (e.g. it could be equal to a part of the CTMid) it is also considered that the SCF translates it or search it before sending an operation on this entry.

## B.3.2 Authentication

### B.3.2.1 Authentication in the SDFsl

The operation used should be the BIND operation, the request could take the following argument:

```
bindAuthentication DirectoryBindArgument ::={
credentials externalProcedure {
    identification  syntax : id-as-challenge-response,
    data-value  notation: SecurityCredentialsArg: {
        name    {{ { type id-at-countryName, value PrintableString: a-countryName },
            { type id-at-serviceProviderCode, value NumericString: a-serviceProviderCode},
            { type id-at-ctmIdentityPrefix, value NumericString: a-cmtIdentityPrefix },
            { type id-at-ctmIdentity, value NumericString: a-ctmIdentity,
                valuesWithContext {{Context : a-currentTime}} },
        securityFacilityId      id-sf-challengeResponse
        challengeResponseUsr {
                messageData     a-challenge, -- RAND and RS
                certificationCode   a-response} -- RES
    }
} --authentication data of the terminal
```

The result will be if the authentication succeeds:

```
bindauthenticationResult DirectoryBindResult ::= {
    credentials externalProcedure: {
            identification syntax: id-as-challenge-response,
            data-value notation: NULL }}    -- or "a-cipheringKey"
                            -- if ciphering is agreed
```

#### B.3.2.1.1 Authentication by DECT session key

##### B.3.2.1.1.1 First authentication, no data stored in the SDFmm

The SEARCH operation could take the following argument:

```
{{baseObject{   rdnSequence{{
            { type id-at-countryName, value PrintableString: a-countryName },
            { type id-at-serviceProviderCode, value NumericString: a-serviceProviderCode},
            { type id-at-ctmIdentityPrefix, value NumericString: a-cmtIdentityPrefix },
            { type id-at-ctmIdentity, value NumericString: a-ctmIdentity,
                valuesWithContext {{Context : a-currentTime}} },
    selection   {attributes {select {id-at-dectSessionSecurityInfo}},
            contextSelection {selectedContexts {assignementContext :a-requestorProvider}
 }}
```

The EXECUTE operation could take the following argument:

```
{{baseObject{   rdnSequence{{
            { type id-at-countryName, value PrintableString: a-countryName },
            { type id-at-serviceProviderCode, value NumericString: a-serviceProviderCode},
            { type id-at-ctmIdentityPrefix, value NumericString: a-cmtIdentityPrefix },
            { type id-at-ctmIdentity, value NumericString: a-ctmIdentity,
                valuesWithContext {{Context : a-currentTime}} },
            { type id-at-stockId, value : id-dt-dect-stage1 } }},
    method-id       id-mt-provideTokens,
    specific-input  {{INTEGER:1},
            {OBJECT IDENTIFIER: id-at-rsKs}}}}
```

#### B.3.2.1.1.2 Data stored in the SDFmm

The operation used should be a search, the request could take the following argument:

```
{{baseObject{   rdnSequence{{
            { type id-at-countryName, value PrintableString: a-countryName },
            { type id-at-serviceProviderCode, value NumericString: a-serviceProviderCode},
            { type id-at-temporaryIdentityPrefix, value NumericString: a-temporaryIdPrefix
},
            { type id-at-localUserIdentifier, value NumericString: a-localUserIdentifier,
                valuesWithContext {{Context : a-currentTime}} },
    selection   {attributes {select { id-at-dectSessionSecurityInfo}} }}
```

### B.3.2.1.2 Authentication by set of n-uplet

The operation should be the execute operation; the request to the home SDFsl could take the following argument:

```
{{baseObject{   rdnSequence{{
            { type id-at-countryName, value PrintableString: a-countryName },
            { type id-at-serviceProviderCode, value NumericString: a-serviceProviderCode},
            { type id-at-ctmIdentityPrefix, value NumericString: a-cmtIdentityPrefix },
            { type id-at-ctmIdentity, value NumericString: a-ctmIdentity,
                valuesWithContext {{Context : a-currentTime}} },
            { type id-at-stockId, value : id-dt-challengeResponse } }},
    method-id       id-mt-provideTokens,
    specific-input  {{INTEGER:a-numberOfTokens},
            {OBJECT IDENTIFIER: id-at-challengeResponse}}}}
```

The request to the local SDFmm could take the following argument:

```
{{baseObject{   rdnSequence{{
            { type id-at-countryName, value PrintableString: a-countryName },
            { type id-at-serviceProviderCode, value NumericString: a-serviceProviderCode},
            { type id-at-temporaryIdentityPrefix, value NumericString: a-temporaryIdPrefix
},
            { type id-at-localUserIdentifier, value NumericString: a-localUserIdentifier,
                valuesWithContext {{Context : a-currentTime}} },
            { type id-at-stockId, value : id-dt-challengeResponse } }},
    method-id       id-mt-provideTokens,
    specific-input  {{INTEGER:1},
            {OBJECT IDENTIFIER: id-at-challengeResponse}}}}
```

The result would be:

```
{{method-id id-mt-provideTokens},
{specific-output    {type   id-at-ChallengeResponse, value a-setOfValues}}}
```

## B.3.2.2 Location registration/deregistration

### B.3.2.2.1 Registration in the visited database

#### B.3.2.2.1.1 SDFmm entries creation

The SCFmm should add the temporary user entry with the "addentry" operation, the request could take the following argument:

```
{{object rdnSequence{{
    { type id-at-countryName, value PrintableString: a-countryName },
    { type id-at-serviceProviderCode, value NumericString: a-serviceProviderCode},
    { type id-at-temporaryIdentityPrefix, value NumericString: a-temporaryIdPrefix },
    { type id-at-localUserIdentifier, value NumericString: a-localUserIdentifier,
        valuesWithContext {{Context : a-periodOfTime}} } }},
    entry   {{
    {type id-at-ctmIdentityList, values {{a-ctmIdentityList}},
-- {type id-at-dectSessionSecurityInfo, values {a-dectSessionSecurityInfo}}, -- -- optional
{type id-at-temporaryLocation, values {a-temporaryLocation}}}
}}}
```

If the authentication procedure is based on triplet, the SCFmm would also create an authentication entry to store the set (minus one element) of triplet requested during the authentication (cf 0). The addentry operation arguments could be:

```
{{object rdnSequence{{
    { type id-at-countryName, value PrintableString: a-countryName },
    { type id-at-serviceProviderCode, value NumericString: a-serviceProviderCode},
    { type id-at-temporaryIdentityPrefix, value NumericString: a-temporaryIdPrefix },
    { type id-at-localUserIdentifier, value NumericString: a-localUserIdentifier,
        valuesWithContext {{Context : a-periodOfTime }} },
{type id-at-stockId, value{ INTEGER : 1}}}},
    entry   {{
    {type id-at-source, values {a-sdfhomesource}},
-- {type id-at-sizeOfRestocking, values {a-setsize}}, -- -- optional
{type id-at-challengeResponse, values a-set-challengeResponse}}
}
```

### B.3.2.2.2 Data updating in the home database

During this procedure, 4 operations should be launched:

- "removeEntry" to remove the old CTM primary Location entry, the request could take the following argument:

```
{   object{ rdnSequence{{
            { type id-at-countryName, value PrintableString: a-countryName },
            { type id-at-serviceProviderCode, value NumericString: a-serviceProviderCode},
            { type id-at-ctmIdentityPrefix, value NumericString: a-cmtIdentityPrefix },
            { type id-at-ctmIdentity, value NumericString: a-ctmIdentity},
            { type id-at-locationCommonID, value : 0 } }}
```

- "addEntry" operation in the CTM primary Location entry (subordinate to the CTM User entry) to set the new location in the SDFsl, the request could take the following argument:

```
{   object{ rdnSequence{{
            { type id-at-countryName, value PrintableString: a-countryName },
            { type id-at-serviceProviderCode, value NumericString: a-serviceProviderCode},
            { type id-at-ctmIdentityPrefix, value NumericString: a-cmtIdentityPrefix },
            { type id-at-ctmIdentity, value NumericString: a-ctmIdentity},
            { type id-at-locationCommonID, value : 0 } ,
                valuesWithContext {{Context : a-periodTemporalContext1}} }},
    entry   {{
{ type  id-at-scpLocation,values  { NumericString: a-newScpLocation }},
{ type  id-at-sdpLocation,values    { NumericString: a-newSdpLocation }} ,
{ type  id-at-routingAddress,values { NumericString: a-newLocation }}}
```

- "removeEntry" operation on the Temporary User Alias entry to remove the Temporary User entry in the old visited SDFmm, the request could take the following argument:

```
{   object{ rdnSequence{{
            { type id-at-countryName, value PrintableString: a-countryName },
            { type id-at-serviceProviderCode, value NumericString: a-serviceProviderCode},
            { type id-at-ctmIdentityPrefix, value NumericString: a-cmtIdentityPrefix },
            { type id-at-ctmIdentity, value NumericString: a-ctmIdentity,
               valuesWithContext {{Context : a-currentTime}} },
            { type id-at-registrationIdentifier, value : a-registrationId } }}}
```

- "modifyEntry" operation on the Temporary User Alias entry with the ***dontDereferenceAliases*** bit set to True in the ***serviceControls*** value in order to update the alias with the new Temporary User entry value, the request could take the following argument:

```
{   object{ rdnSequence{{
        { type id-at-countryName, value PrintableString: a-countryName },
        { type id-at-serviceProviderCode, value NumericString: a-serviceProviderCode},
        { type id-at-ctmIdentityPrefix, value NumericString: a-cmtIdentityPrefix },
        { type id-at-ctmIdentity, value NumericString: a-ctmIdentity,
           valuesWithContext {{Context : a-currentTime}} },
        { type id-at-registrationIdentifier, value : a-registrationId }  }},
    changes {
        resetValues:    { type  id-at-aliasedEntryName},
        addValues:  { type  id-at-aliasedEntryName,
                values  { DistinguishedName: a-newTemporaryUser },
                valuesWithContext {{Context : a-periodTemporalContext1}} }}
```

## B.3.2.3 Provision of home specific services

The SCF should retrieve this data using the "search" operation, the request could take the following argument:

```
 {baseObject{   rdnSequence{{
        { type id-at-countryName, value PrintableString: a-countryName },
        { type id-at-serviceProviderCode, value NumericString: a-serviceProviderCode},
        { type id-at-ctmIdentityPrefix, value NumericString: a-cmtIdentityPrefix },
        { type id-at-ctmIdentity, value NumericString: a-ctmIdentity,
           valuesWithContext {{Context : a-currentTime}} },
    selection   {attributes {select { id-at-homeSpecificMark}}}}}
```

When non-home-specific services are requested, the supplementary service status and parameters have to be read . The operation should be a search, the request could take the following argument:

```
{   object{ rdnSequence{{
        { type id-at-countryName, value PrintableString: a-countryName },
        { type id-at-serviceProviderCode, value NumericString: a-serviceProviderCode},
        { type id-at-ctmIdentityPrefix, value NumericString: a-cmtIdentityPrefix },
        { type id-at-ctmIdentity, value NumericString: a-ctmIdentity,
           valuesWithContext {{Context : a-currentTime}} },
        { type id-at-supplServId, value : a-supplServiceId} }},
    filter{ item{   equality{   type        supplServiceStatus,
                    assertion   activated}}}}
```

## B.3.2.4 Incoming calls

If the originating SCF supports the "white ISUP", it should retrieve the location information from the home database by a "search" operation, the request could take the following argument:

```
{   object{ rdnSequence{{
        { type id-at-countryName, value PrintableString: a-countryName },
        { type id-at-serviceProviderCode, value NumericString: a-serviceProviderCode},
        { type id-at-ctmIdentityPrefix, value NumericString: a-cmtIdentityPrefix },
        { type id-at-ctmNumber, value NumericString: a-ctmNumber},
        { type id-at-locationCommonID, value : 0 },
           valuesWithContext {{Context : a-currentTime}}  }},
    selection   {attributes {allUserAttributes:NULL}}   -- routing address for ISUP ,
                                -- SCP and SDP address
```

It should also retrieve the CTMid attribute and optionally the **homeSpecificServices** attribute to determine if the user subscribed to a network specific incoming feature if these parameters are not already fixed by a mutual agreement. The retrieval is done by a "search" operation, the request could take the following argument:

```
{   object{ rdnSequence{{
        { type id-at-countryName, value PrintableString: a-countryName },
        { type id-at-serviceProviderCode, value NumericString: a-serviceProviderCode},
        { type id-at-ctmIdentityPrefix, value NumericString: a-cmtIdentityPrefix },
        { type id-at-ctmNumber, value NumericString: a-ctmNumber},
           valuesWithContext {{Context : a-currentTime}}  }},
    selection   {attributes {select:{{type id-at-ctmIdentity},
                {type id-at-homeSpecificServices}}}
```

To get a roaming number, an "execute" operation should be used; the request could take the following argument:

```
{{object{   rdnSequence{{
            { type id-at-countryName, value PrintableString: a-countryName },
            { type id-at-serviceProviderCode, value NumericString: a-serviceProviderCode},
            { type id-at-ctmIdentityPrefix, value NumericString: a-cmtIdentityPrefix }}}},
    method-id   id-mt-selectAndAssign,
    specific-input  {{rdnSequence {{
{ type id-at-countryName, value PrintableString: a-countryName },
            { type id-at-serviceProviderCode, value NumericString: a-serviceProviderCode},
            { type id-at-ctmIdentityPrefix, value NumericString: a-cmtIdentityPrefix },
            { type id-at-ctmIdentity, value NumericString: a-ctmIdentity}}}}}
 }}
```

The terminating network sends a call setup with the RN as called number, then the visited Network translates it back in CTMid, the operation should be "execute"; the request could take the following argument:

```
{{object{   rdnSequence{{
            { type id-at-countryName, value PrintableString: a-countryName },
            { type id-at-serviceProviderCode, value NumericString: a-serviceProviderCode},
            { type id-at-ctmIdentityPrefix, value NumericString: a-cmtIdentityPrefix }}}},
    method-id   id-mt-findAndRelease,
    input-assertions    {{type id-at-assignmentTable ,value a-RN}}
}}
```

If the Routeing Address is used, the operation should be a "search" in the local SDFsl; the request could take the following argument:

```
{   object{ rdnSequence{{
{ type id-at-countryName, value PrintableString: a-countryName },
            { type id-at-serviceProviderCode, value NumericString: a-serviceProviderCode},
            { type id-at-ctmNumberPrefix, value NumericString: a-cmtNumberPrefix },
            { type id-at-ctmNumber, value NumericString: a-ctmNumber,
                valuesWithContext {{Context : a-currentTime}} }},
    selection   {attributes {select { id-at-ctmIdentity}}}
```

Then the visited SCF should retrieve, from the local SDFmm, the location information (FT address) using search operation; the request could take the following argument:

```
{   object{ rdnSequence{{
            { type id-at-countryName, value PrintableString: a-countryName },
            { type id-at-serviceProviderCode, value NumericString: a-serviceProviderCode},
            { type id-at-temporaryIdentityPrefix, value NumericString: a-temporaryIdPrefix
}}},
    subset  onelevel,
    filter{ item{   equality{   type        ctmIdentityList,
                    assertion   a-ctmIdentity}}},
    selection   {attributes {select { localUserIdentifier, temporaryLocation }}} }
```

To authenticate the user, one of procedures described in subclause 8.2.2 is used.

## B.3.2.5 Outgoing calls

After the authentication done by the SCFmm, the SCFsl looks up whether the user got a home specific mark. Then if the service is not home specific, it could retrieve the supplementary service data (outgoing restrictions, line identification, notification of waiting messages, etc.).

## B.3.2.6 CTM procedure module

ASN.1 description of the operations samples that could be used over the SCF-SDF interface.

```
CTM-DataOperationTemplate
-- this modules contains the ASN.1 Information Object Notation
-- to illustrate the request of an SCF toward a SDF for the CTM service.

DEFINITIONS::=

BEGIN

--EXPORTS All; --
```

```
IMPORTS

    DirectoryBindArgument,
    DirectoryBindResult,
    EntryInformationSelection,
    Filter,
    ModifyEntryArgument,
    SearchArgument
FROM DirectoryAbstractService {joint-iso-ccitt ds(5) module(1) directoryAbstractService(2) 2}

    ExecuteResult,ExecuteArgument
FROM IN-CS2-SCF-SDF-Operations
            {ccitt recommendation q 1228 module(0) scf-sdf-operations(11) version1(0) }
    emptyUnbind
FROM Remote-Operations-Useful-Definitions {joint-iso-ccitt remote-operations(4) useful-
definitions(7) version1(0)}
    DistinguishedName,
    Name,
    id-at-objectClass
FROM InformationFramework {joint-iso-ccitt ds(5) module(1) informationFramework(1) 2}

    id-at-countryName
FROM SelectedAttributeTypes {joint-iso-ccitt ds(5) module(1) selectedAttributeTypes(5) 2}

    id-as-challenge-response,
id-at-assignmentTable,
id-at-authorizedDestinations,
id-at-barredDestinations,
id-at-bindLevelIfOK,
id-at-ctmIdentity,
id-at-ctmIdentityList,
id-at-ctmIdentityPrefix,
id-at-ctmNumber,
id-at-ctmNumberPrefix,
id-at-currentList,
id-at-dectSessionSecurityInfo,
id-at-failureCounter,
id-at-favoriteLanguage,
id-at-forwardedToNumber,
id-at-identifierList,
id-at-homeSpecificServices,
id-at-localUserIdentifier,
id-at-locationCommonIdentifier,
id-at-maxtime,
id-at-messageWaitingIndication,
id-at-noReplyConditionTimer,
id-at-partnersList,
id-at-perCallBasis,
id-at-randomAssigned,
id-at-registrationIdentifier,
id-at-routingAddress,
id-at-scpLocation,
id-at-sdpLocation,
id-at-secretKey,
id-at-securityFacilityId,
id-at-serviceFeatures,
id-at-serviceProviderCode,
id-at-sizeOfRestocking,
id-at-source,
id-at-stockId,
id-at-subscriptionData,
id-at-supplServiceStatus,
id-at-supplServId,
id-at-temporaryIdentityPrefix,
id-at-temporaryLocation,
id-at-terminalID,
id-at-tokens,
id-at-typesOfNotifications,
id-at-userCredit,
    id-sf-dect,
    id-ss-cc,
    id-ss-cfb,
    id-ss-cfnrc,
    id-ss-or,
    id-ss-vm,
    SecurityCredentialsArg,
    SecurityCredentialsResult,
    SS-CODE
 FROM DataModel
 a-countryName,a-serviceProviderCode, a-ctmIdentityPrefix
FROM SCFfilledvalues -- operator specific values to compile the asn1 modules
                            -- the value during the processing of the service would be filled
                            -- by the SCFs

;

bindAuthentication DirectoryBindArgument ::={
credentials externalProcedure {
    identification  syntax : id-as-challenge-response,
    data-value  notation: SecurityCredentialsArg: {
        name    {{ { type id-at-countryName, value PrintableString: a-countryName },
            { type id-at-serviceProviderCode, value NumericString: a-serviceProviderCode},
            { type id-at-ctmIdentityPrefix, value NumericString: a-cmtIdentityPrefix },
            { type id-at-ctmIdentity, value NumericString: a-ctmIdentity,
                valuesWithContext {{Context : a-currentTime}} },
        securityFacilityId      id-sf-challengeResponse
        challengeResponseUsr {
                messageData     a-challenge, -- RAND and RS
                certificationCode   a-response} -- RES
    }
} --authentication data of the terminal

bindauthenticationResult DirectoryBindResult ::= {
    credentials externalProcedure: {
            identification syntax: id-as-challenge-response,
            data-value notation: NULL }}    -- or "a-cipheringKey"
                            -- if ciphering is agreed

searchHomeRsKs SearchArgument ::= {
{{baseObject{   rdnSequence{{
            { type id-at-countryName, value PrintableString: a-countryName },
            { type id-at-serviceProviderCode, value NumericString: a-serviceProviderCode},
            { type id-at-ctmIdentityPrefix, value NumericString: a-cmtIdentityPrefix },
            { type id-at-ctmIdentity, value NumericString: a-ctmIdentity,
                valuesWithContext {{Context : a-currentTime}} },
    selection   {attributes {select {id-at-dectSessionSecurityInfo}},
            contextSelection {selectedContexts {assignementContext :a-requestorProvider}
 }}}

searchLocalRsKs SearchArgument ::={
{{baseObject{   rdnSequence{{
            { type id-at-countryName, value PrintableString: a-countryName },
            { type id-at-serviceProviderCode, value NumericString: a-serviceProviderCode},
            { type id-at-temporaryIdentityPrefix, value NumericString: a-temporaryIdPrefix
},
            { type id-at-localUserIdentifier, value NumericString: a-localUserIdentifier,
                valuesWithContext {{Context : a-currentTime}} },
    selection   {attributes {select { id-at-dectSessionSecurityInfo}} }}}

executeGetRsKsTokens ExecuteArgument ::= {
{{baseObject{   rdnSequence{{
            { type id-at-countryName, value PrintableString: a-countryName },
            { type id-at-serviceProviderCode, value NumericString: a-serviceProviderCode},
            { type id-at-ctmIdentityPrefix, value NumericString: a-cmtIdentityPrefix },
            { type id-at-ctmIdentity, value NumericString: a-ctmIdentity,
                valuesWithContext {{Context : a-currentTime}} },
            { type id-at-stockId, value : id-dt-dect-stage1 } }},
    method-id       id-mt-provideTokens,
    specific-input  {{INTEGER:1},
            {OBJECT IDENTIFIER: id-at-rsKs}}}}}

executeGetAuthTokens ExecuteArgument ::= {
{{baseObject{   rdnSequence{{
            { type id-at-countryName, value PrintableString: a-countryName },
            { type id-at-serviceProviderCode, value NumericString: a-serviceProviderCode},
            { type id-at-ctmIdentityPrefix, value NumericString: a-cmtIdentityPrefix },
            { type id-at-ctmIdentity, value NumericString: a-ctmIdentity,
                valuesWithContext {{Context : a-currentTime}} },
            { type id-at-stockId, value : id-dt-challengeResponse } }},
    method-id       id-mt-provideTokens,
    specific-input  {{INTEGER:a-numberOfTokens},
            {OBJECT IDENTIFIER: id-at-challengeResponse}}}}}

executeGetAuthOneToken ExecuteArgument ::= {
{{baseObject{   rdnSequence{{
            { type id-at-countryName, value PrintableString: a-countryName },
            { type id-at-serviceProviderCode, value NumericString: a-serviceProviderCode},
            { type id-at-temporaryIdentityPrefix, value NumericString: a-temporaryIdPrefix
},
            { type id-at-localUserIdentifier, value NumericString: a-localUserIdentifier,
                valuesWithContext {{Context : a-currentTime}} },
            { type id-at-stockId, value : id-dt-challengeResponse } }},
    method-id       id-mt-provideTokens,
    specific-input  {{INTEGER:1},
            {OBJECT IDENTIFIER: id-at-challengeResponse}}}}}

addRoamingUserEntry AddEntryArgument ::= {
{{object rdnSequence{{
    { type id-at-countryName, value PrintableString: a-countryName },
    { type id-at-serviceProviderCode, value NumericString: a-serviceProviderCode},
    { type id-at-temporaryIdentityPrefix, value NumericString: a-temporaryIdPrefix },
    { type id-at-localUserIdentifier, value NumericString: a-localUserIdentifier,
        valuesWithContext {{Context : a-periodOfTime}} } }},
    entry   {{
    {type id-at-ctmIdentityList, values {{a-ctmIdentityList}},
-- {type id-at-dectSessionSecurityInfo, values {a-dectSessionSecurityInfo}}, -- -- optional
{type id-at-temporaryLocation, values {a-temporaryLocation}}}
}}}}

addAuthenticationEntry AddEntryArgument ::= {
{{object rdnSequence{{
    { type id-at-countryName, value PrintableString: a-countryName },
    { type id-at-serviceProviderCode, value NumericString: a-serviceProviderCode},
    { type id-at-temporaryIdentityPrefix, value NumericString: a-temporaryIdPrefix },
    { type id-at-localUserIdentifier, value NumericString: a-localUserIdentifier,
        valuesWithContext {{Context : a-periodOfTime }} },
{type id-at-stockId, value{ INTEGER : 1}}}},
    entry   {{
    {type id-at-source, values {a-sdfhomesource}},
-- {type id-at-sizeOfRestocking, values {a-setsize}}, -- -- optional
{type id-at-challengeResponse, values a-set-challengeResponse}}
}}

removePrimaryLocation RemoveEntryArgument ::= {
{   object{ rdnSequence{{
            { type id-at-countryName, value PrintableString: a-countryName },
            { type id-at-serviceProviderCode, value NumericString: a-serviceProviderCode},
            { type id-at-ctmIdentityPrefix, value NumericString: a-cmtIdentityPrefix },
            { type id-at-ctmIdentity, value NumericString: a-ctmIdentity},
            { type id-at-locationCommonID, value : 0 } }}}

addPrimaryLocation AddEntryArgument ::= {
{   object{ rdnSequence{{
            { type id-at-countryName, value PrintableString: a-countryName },
            { type id-at-serviceProviderCode, value NumericString: a-serviceProviderCode},
            { type id-at-ctmIdentityPrefix, value NumericString: a-cmtIdentityPrefix },
            { type id-at-ctmIdentity, value NumericString: a-ctmIdentity},
            { type id-at-locationCommonID, value : 0 } ,
                valuesWithContext {{Context : a-periodTemporalContext1}} }},
    entry   {{
{ type  id-at-scpLocation,values  { NumericString: a-newScpLocation }},
{ type  id-at-sdpLocation,values    { NumericString: a-newSdpLocation }} ,
{ type  id-at-routingAddress,values { NumericString: a-newLocation }}}
}

removeTemporaryUserAlias RemoveEntryArgument ::= {
{   object{ rdnSequence{{
            { type id-at-countryName, value PrintableString: a-countryName },
            { type id-at-serviceProviderCode, value NumericString: a-serviceProviderCode},
            { type id-at-ctmIdentityPrefix, value NumericString: a-cmtIdentityPrefix },
            { type id-at-ctmIdentity, value NumericString: a-ctmIdentity,
                valuesWithContext {{Context : a-currentTime}} },
            { type id-at-registrationIdentifier, value : a-registrationId } }}}}

modifyTemporaryUserAlias ModifyEntry ::= {
{   object{ rdnSequence{{
            { type id-at-countryName, value PrintableString: a-countryName },
            { type id-at-serviceProviderCode, value NumericString: a-serviceProviderCode},
            { type id-at-ctmIdentityPrefix, value NumericString: a-cmtIdentityPrefix },
            { type id-at-ctmIdentity, value NumericString: a-ctmIdentity,
                valuesWithContext {{Context : a-currentTime}} },
            { type id-at-registrationIdentifier, value : a-registrationId }  }},
    changes {
        resetValues:    { type  id-at-aliasedEntryName},
        addValues:  { type  id-at-aliasedEntryName,
                values  { DistinguishedName: a-newTemporaryUser },
                valuesWithContext {{Context : a-periodTemporalContext1}} }}}

searchHomeSpecificMark SearchArgument ::= {
 {baseObject{   rdnSequence{{
            { type id-at-countryName, value PrintableString: a-countryName },
            { type id-at-serviceProviderCode, value NumericString: a-serviceProviderCode},
            { type id-at-ctmIdentityPrefix, value NumericString: a-cmtIdentityPrefix },
            { type id-at-ctmIdentity, value NumericString: a-ctmIdentity,
                valuesWithContext {{Context : a-currentTime}} },
    selection   {attributes {select { id-at-homeSpecificMark}}}}}
}

searchSSStatus SearchArgument ::= {
{   object{ rdnSequence{{
            { type id-at-countryName, value PrintableString: a-countryName },
            { type id-at-serviceProviderCode, value NumericString: a-serviceProviderCode},
            { type id-at-ctmIdentityPrefix, value NumericString: a-cmtIdentityPrefix },
            { type id-at-ctmIdentity, value NumericString: a-ctmIdentity,
                valuesWithContext {{Context : a-currentTime}} },
            { type id-at-supplServId, value : a-supplServiceId} }},
    filter{ item{   equality{   type        supplServiceStatus,
                    assertion   activated}}}}}

searchRoutingAddress ::= {
{   object{ rdnSequence{{
            { type id-at-countryName, value PrintableString: a-countryName },
            { type id-at-serviceProviderCode, value NumericString: a-serviceProviderCode},
            { type id-at-ctmIdentityPrefix, value NumericString: a-cmtIdentityPrefix },
            { type id-at-ctmNumber, value NumericString: a-ctmNumber},
            { type id-at-locationCommonID, value : 0 },
                valuesWithContext {{Context : a-currentTime}}  }},
    selection   {attributes {allUserAttributes:NULL}}   -- routing address for ISUP ,
                                -- SCP and SDP address}

executeGetRoamingNumber ExecuteArgument ::= {
{{object{   rdnSequence{{
            { type id-at-countryName, value PrintableString: a-countryName },
            { type id-at-serviceProviderCode, value NumericString: a-serviceProviderCode},
            { type id-at-ctmIdentityPrefix, value NumericString: a-cmtIdentityPrefix }}}},
    method-id   id-mt-selectAndAssign,
    specific-input  {{rdnSequence {{
{ type id-at-countryName, value PrintableString: a-countryName },
            { type id-at-serviceProviderCode, value NumericString: a-serviceProviderCode},
            { type id-at-ctmIdentityPrefix, value NumericString: a-cmtIdentityPrefix },
            { type id-at-ctmIdentity, value NumericString: a-ctmIdentity}}}}}
 }}}

executeReleaseRoamingNumber ExecuteArgument ::= {
{{object{   rdnSequence{{
            { type id-at-countryName, value PrintableString: a-countryName },
            { type id-at-serviceProviderCode, value NumericString: a-serviceProviderCode},
            { type id-at-ctmIdentityPrefix, value NumericString: a-cmtIdentityPrefix }}}},
    method-id   id-mt-findAndRelease,
    input-assertions    {{type id-at-assignmentTable ,value a-RN}}
}}}

searchHomeSpecificMark SearchArgument ::= {
{   object{ rdnSequence{{
            { type id-at-countryName, value PrintableString: a-countryName },
            { type id-at-serviceProviderCode, value NumericString: a-serviceProviderCode},
            { type id-at-ctmIdentityPrefix, value NumericString: a-cmtIdentityPrefix },
            { type id-at-ctmNumber, value NumericString: a-ctmNumber},
                valuesWithContext {{Context : a-currentTime}}  }},
    selection   {attributes {select:{{type id-at-ctmIdentity},
                {type id-at-homeSpecificServices}}} }

searchCTMid SearchArgument ::= {
{   object{ rdnSequence{{
{ type id-at-countryName, value PrintableString: a-countryName },
            { type id-at-serviceProviderCode, value NumericString: a-serviceProviderCode},
            { type id-at-ctmNumberPrefix, value NumericString: a-cmtNumberPrefix },
            { type id-at-ctmNumber, value NumericString: a-ctmNumber,
                valuesWithContext {{Context : a-currentTime}} }},
    selection   {attributes {select { id-at-ctmIdentity}}}}

searchFTaddress SearchArgument ::= {
{   object{ rdnSequence{{
            { type id-at-countryName, value PrintableString: a-countryName },
            { type id-at-serviceProviderCode, value NumericString: a-serviceProviderCode},
            { type id-at-temporaryIdentityPrefix, value NumericString: a-temporaryIdPrefix
}}},
    subset  onelevel,
    filter{ item{   equality{   type        ctmIdentityList,
                    assertion   a-ctmIdentity}}},
    selection   {attributes {select { localUserIdentifier, temporaryLocation }}} }}

searchSubscriptionData SearchArgument ::= {
{{object{   rdnSequence{{
        { type id-at-countryName, value PrintableString: a-countryName },
        { type id-at-serviceProviderCode, value NumericString: a-serviceProviderCode},
        { type id-at-terminalID, value BitString: a-terminalID}}}},
    subset  baseObject,
    selection   {attributes {select {ctmIdentity,subscriptionData }}}

searchCTMNumber SearchArgument ::= {
{{baseObject{   rdnSequence{{
            { type id-at-countryName, value PrintableString: a-countryName },
            { type id-at-serviceProviderCode, value NumericString: a-serviceProviderCode},
            { type id-at-ctmIdentityPrefix, value NumericString: a-cmtIdentityPrefix },
            { type id-at-ctmIdentity, value NumericString: a-ctmIdentity,
                valuesWithContext {{Context : a-currentTime}} },
    subset      baseObject,
    selection   {attributes {select {ctmNumber}}
 }}

bindauthenticationResult DirectoryBindResult ::= {
    credentials externalProcedure: {
                identification syntax: id-as-challenge-response,
                data-value notation: {cipheringKey a-ciphKey,
challengeResponseNetwork a-res2 } --to be sent to the terminal
    }}

END
```

## History

| Document history | | | | |
|---|---|---|---|---|
| V1.1.1 | January 1998 | Public Enquiry | PE 9820: | 1998-01-16 to 1998-05-15 |
| | | | | |
| | | | | |
| | | | | |
| | | | | |