INTERNATIONAL TELECOMMUNICATION UNION

ITU-T

TELECOMMUNICATION
STANDARDIZATION SECTOR
OF ITU

Q.1248.4

(07/2001)

SERIES Q: SWITCHING AND SIGNALLING

Intelligent Network

Interface Recommendation for Intelligent Network Capability Set 4: SCF-SDF interface

ITU-T Recommendation Q.1248.4

ITU-T Q-SERIES RECOMMENDATIONS

SWITCHING AND SIGNALLING

SIGNALLING IN THE INTERNATIONAL MANUAL SERVICE	Q.1–Q.3
INTERNATIONAL AUTOMATIC AND SEMI-AUTOMATIC WORKING	Q.4–Q.59
FUNCTIONS AND INFORMATION FLOWS FOR SERVICES IN THE ISDN	Q.60–Q.99
CLAUSES APPLICABLE TO ITU-T STANDARD SYSTEMS	Q.100–Q.119
SPECIFICATIONS OF SIGNALLING SYSTEMS No. 4 AND No. 5	Q.120–Q.249
SPECIFICATIONS OF SIGNALLING SYSTEM No. 6	Q.250–Q.309
SPECIFICATIONS OF SIGNALLING SYSTEM R1	Q.310–Q.399
SPECIFICATIONS OF SIGNALLING SYSTEM R2	Q.400–Q.499
DIGITAL EXCHANGES	Q.500–Q.599
INTERWORKING OF SIGNALLING SYSTEMS	Q.600–Q.699
SPECIFICATIONS OF SIGNALLING SYSTEM No. 7	Q.700–Q.799
Q3 INTERFACE	Q.800–Q.849
DIGITAL SUBSCRIBER SIGNALLING SYSTEM No. 1	Q.850–Q.999
PUBLIC LAND MOBILE NETWORK	Q.1000–Q.1099
INTERWORKING WITH SATELLITE MOBILE SYSTEMS	Q.1100–Q.1199
INTELLIGENT NETWORK	**Q.1200–Q.1699**
SIGNALLING REQUIREMENTS AND PROTOCOLS FOR IMT-2000	Q.1700–Q.1799
SPECIFICATIONS OF SIGNALLING RELATED TO BEARER INDEPENDENT CALL CONTROL (BICC)	Q.1900–Q.1999
BROADBAND ISDN	Q.2000–Q.2999

For further details, please refer to the list of ITU-T Recommendations.

ITU-T Recommendation Q.1248.4

Interface Recommendation for Intelligent Network Capability Set 4: SCF-SDF interface

Summary

The Q.1248.x series of ITU-T Recommendations defines the Intelligent Network (IN) Application Protocol (INAP) for IN Capability Set 4 (IN CS-4), the INAP for IN CS-4 based upon IN CS-3 Q.1238 specification (2000), and the general rules for INAP provided in ITU-T Q.1208, and is consistent with the scope of IN CS-4 as defined in ITU-T Q.1241.

Within the Q.124x series of ITU-T Recommendations, the Q.1248.x series describes the protocol realizing the Q.1241 Distributed Functional Plane in a service and vendor implementation independent manner, as constrained by the capabilities of the embedded base of network technology. This provides the flexibility to allocate distributed functionality into multiple physical network configurations and to evolve IN from IN CS-4 to some future CS-N.

This Recommendation belongs to the Q.1248.x series of ITU-T Recommendations for IN Capability Set 4. It covers the SCF-SDF interface including the description of the aspects of the SCF and SDF Functional Entities which are relevant to this interface.

This Recommendation includes an electronic attachment containing Annex A, ASN.1 definitions.

Source

ITU-T Recommendation Q.1248.4 was prepared by ITU-T Study Group 11 (2001-2004) and approved under the WTSA Resolution 1 procedure on 13 July 2001.

Keywords

Intelligent Network (IN), Intelligent Network Application Protocol (INAP), IN Capability Set 4 (IN CS-4), Service Control Function (SCF), Service Data Function (SDF).

FOREWORD

The International Telecommunication Union (ITU) is the United Nations specialized agency in the field of telecommunications. The ITU Telecommunication Standardization Sector (ITU-T) is a permanent organ of ITU. ITU-T is responsible for studying technical, operating and tariff questions and issuing Recommendations on them with a view to standardizing telecommunications on a worldwide basis.

The World Telecommunication Standardization Assembly (WTSA), which meets every four years, establishes the topics for study by the ITU-T study groups which, in turn, produce Recommendations on these topics.

The approval of ITU-T Recommendations is covered by the procedure laid down in WTSA Resolution 1.

In some areas of information technology which fall within ITU-T's purview, the necessary standards are prepared on a collaborative basis with ISO and IEC.

NOTE

In this Recommendation, the expression "Administration" is used for conciseness to indicate both a telecommunication administration and a recognized operating agency.

INTELLECTUAL PROPERTY RIGHTS

ITU draws attention to the possibility that the practice or implementation of this Recommendation may involve the use of a claimed Intellectual Property Right. ITU takes no position concerning the evidence, validity or applicability of claimed Intellectual Property Rights, whether asserted by ITU members or others outside of the Recommendation development process.

As of the date of approval of this Recommendation, ITU had not received notice of intellectual property, protected by patents, which may be required to implement this Recommendation. However, implementors are cautioned that this may not represent the latest information and are therefore strongly urged to consult the TSB patent database.

© ITU 2002

All rights reserved. No part of this publication may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm, without permission in writing from ITU.

CONTENTS

		Page
1	Scope	1
2	References	1
3	Abbreviations and acronyms	1
4	Introduction	1
4.1	SCF-SDF relationship	1
4.2	Introduction to the reuse of X.500 for SDF interfaces	1
	4.2.1 Alignment between the X.500 concepts and the IN	1
	4.2.2 Use of a limited subset of X.500	2
4.3	Versions and the rules for extensibility	3
	4.3.1 Version negotiation	3
	4.3.2 SCF side	3
	4.3.3 Request processing at the SDF side	4
5	Service Control Function (SCF) Model	4
6	Service Data Function (SDF) Model	4
6.1	General	4
6.2	SDF components	5
	6.2.1 General	5
	6.2.2 SDF data manager	5
	6.2.3 Functional entity access manager	5
	6.2.4 Security manager	6
6.3	Data types handled by SDF	6
7	FSM for SCF	6
7.1	Overview	6
7.2	SCME-FSM	7
	7.2.1 State 1: "Idle"	8
	7.2.2 State 2: "Bind Pending"	8
	7.2.3 State 3: "Active"	8
7.3	SDF-related states (SCSM-SDF)	9
	7.3.1 State 1: "Idle"	9
	7.3.2 State 2: "Wait for subsequent Requests"	9
	7.3.3 State 3: "Wait for Bind Result"	10
	7.3.4 State 4: "SDF Bound"	10
8	FSM for SDF	10
8.1	Overview	10
8.2	SDME-FSM	12

Page

8.2.1 State 1: "Idle" 12
8.2.2 State 2: "Wait for TFC" 12
8.2.3 State 3: "Wait for BindResult" 13
8.2.4 State 4: "Active" 13
8.3 SDF state transition model for SCF-related states 13
8.3.1 State 1: "Idle" 14
8.3.2 State 2: "Bind Pending" 14
8.3.3 State 3: "SCF Bound" 14
9 Detailed operations procedure 15
9.1 IN-Execute procedure 15
9.1.1 General description 15
9.1.2 Parameters 15
9.1.3 Invoking entity (SCF) 16
9.1.4 Responding entity (SDF) 17
9.2 IN-AddEntry procedure 18
9.2.1 General description 18
9.2.2 Parameters 18
9.2.3 Invoking entity (SCF) 19
9.2.4 Responding entity (SDF) 19
9.3 IN-DirectoryBind procedure 20
9.3.1 General description 20
9.3.2 Parameters 20
9.3.3 Invoking Entity (SCF) 20
9.3.4 Responding Entity (SDF) 21
9.4 IN-ModifyEntry procedure 21
9.4.1 General description 21
9.4.2 Parameters 21
9.4.3 Invoking entity (SCF) 22
9.4.4 Responding entity (SDF) 22
9.5 IN-RemoveEntry procedure 23
9.5.1 General description 23
9.5.2 Parameters 23
9.5.3 Invoking entity (SCF) 23
9.5.4 Responding entity (SDF) 24
9.6 IN-Search procedure 24
9.6.1 General description 24
9.6.2 Parameters 24
9.6.3 Invoking entity (SCF) 25
9.6.4 Responding entity (SDF) 26

Page

9.7 IN-Unbind procedure 26
9.7.1 General description 26
9.7.2 Invoking entity (SCF in Directory Access) 26
9.7.3 Responding entity (SDF in Directory Access) 27
9.7.4 Invoking entity (SDF in traffic flow control) 27
9.7.5 Responding Entity (SCF in traffic flow control) 27
9.8 TFCBind procedure 28
9.8.1 General description 28
9.8.2 Invoking entity (SDF) 28
9.8.3 Responding entity (SCF) 28
9.9 TrafficFlowControl procedure 29
9.9.1 General description 29
9.9.2 Parameters 29
9.9.3 Invoking entity (SDF) 30
9.9.4 Responding entity (SCF) 30
9.10 Operation timer values 30
10 Parameters 31
10.1 BaseObject 31
10.2 Changes 31
10.3 CheckOverspecified 31
10.4 CommonResults 31
10.5 ControlType 31
10.6 Credentials 31
10.7 Duration 32
10.8 Entry 32
10.9 Filter 32
10.10 IN-CommonArguments 32
10.11 Information 33
10.12 Input-assertions 33
10.13 MatchedValuesOnly 34
10.14 Method-id 34
10.15 Null 34
10.16 Object 34
10.17 Output-assertions 34
10.18 SearchAliases 34
10.19 SearchInfo 34
10.20 Security 35

Page

10.21 Selection 35
10.22 Specific-input 36
10.23 Specific-output 36
10.24 Subset 36
10.25 TargetSystem 36
10.26 TfcCriteria 36
10.27 UncorrelatedSearchInfo 37
10.28 Versions 37
10.29 WasChained 37

11 Errors 37
11.1 Operation-related error procedures 37
11.1.1 AttributeError 38
11.1.2 ExecutionError 38
11.1.3 IN-DirectoryBindError 39
11.1.4 IN-ServiceError 40
11.1.5 NameError 40
11.1.6 Referral 41
11.1.7 SecurityError 41
11.1.8 TFCBindError 42
11.1.9 UpdateError 42

12 Service assumed from TCAP 43
12.1 Normal procedures 43
12.1.1 SCSM/SDSM-related messages 43
12.1.2 SCME/SDME-related messages 43
12.2 Abnormal procedures 44
12.3 Dialogue handling 44
12.3.1 Dialogue establishment 44
12.3.2 Dialogue continuation 44
12.3.3 Dialogue termination 44
12.3.4 User abort 44
12.3.5 Provider abort 44
12.3.6 Mapping to TC dialogue primitives 44
12.4 Component handling 47
12.4.1 Procedures for INAP operations 47
12.4.2 Mapping to TC component parameters 47

Annex A – ASN.1 definitions (electronic attachment) 47

ITU-T Recommendation Q.1248.4

Interface Recommendation for Intelligent Network Capability Set 4: SCF-SDF interface[1]

1 Scope

This Recommendation belongs to the Q.1248.x series of ITU-T Recommendations for IN Capability Set 4. It specifies the protocol on the SCF-SDF interface and provides a description of the aspects of the SCF and SDF Functional Entities which are involved in the realization of this interface.

2 References

All ITU-T Recommendations and other references are identified in ITU-T Q.1248.1.

3 Abbreviations and acronyms

All abbreviations and acronyms used in this text are defined in ITU-T Q.1248.1.

4 Introduction

4.1 SCF-SDF relationship

This SCF-SDF relationship is used for messages between an SCF and an SDF in the public network. The relationship supports access to service data, which can be for intra-network or inter-network purposes.

The SCF-SDF relationship supports messages for services when no call is actually taking place (call unrelated) as well as during call processing. For the most part, the call unrelated actions are to support the registration, authentication, encryption and handover procedures for terminal and personal mobility.

The SCF-SDF relationship is also one of a limited set of relationships to support inter-networking. As such it provides a point of interconnection to the network, effectively hiding the specific structure of the network and providing access security to the network from other public networks.

The SCF-SDF relationship is used for various purposes. Each of the purposes can apply to the intra-network or inter-network cases. Specifically, the relationship is used for the Secured Data Acquisition from an SDF by an SCF. In this case, an SCF requires data to complete its processing of service logic. This relationship is the same as the SCF-SDF relationship as defined in IN CS-1 and CS-2 and uses the same operations. It differs from the IN CS-1 and CS-2 relationship in the provision of additional methods for supporting verification of user/terminal credential.

4.2 Introduction to the reuse of X.500 for SDF interfaces

4.2.1 Alignment between the X.500 concepts and the IN

The X.500 series of ITU-T Recommendations are used to specify the SCF-SDF interface and the contents of the SDF. Most of the concepts of the X.500 series are directly used in the IN environment; however, some alignments need to be done at the terminology level to ensure that the

[1] This Recommendation includes an electronic attachment containing Annex A, ASN.1 definitions.

concepts introduced in the Directory are correctly understood. The purpose of this subclause is to provide this alignment. It therefore only concentrates on the terms that are ambiguous in the IN environment.

When looking at the structure of the SCF, the Service Data Management is the part of the SCF responsible for the interactions with the SDF. It can be mapped onto the concept of Directory User Agent (DUA). When an SCF on behalf of a user wants to set up an association with an SDF, an instance of a DUA is created in the SLPI. It is killed when the association is ended.

The SDF is the entity responsible for answering the database requests. This functional entity can be mapped onto the Directory System Agent (DSA). When an association is set up between an SCF and an SDF, an instance of a DSA is created for the length of the association.

The Directory is a collection of DSAs/SDFs. This set can be used for a specific service or for a variety of services. The notion of Directory is equivalent to the concept of database systems in IN.

The Directory can also be seen as a repository of data. IN services provide various kinds of data access to users. The information is organized into entries. An entry is a collection of information that can be identified (or named). When it represents an object (i.e. contains primary information about an object), it is called an object entry.

Objects are anything which are identifiable (can be named) and which are of interest to hold information on in the database. A typical example of an object is a user. Objects can be described by several entries. Each individual information that is used to describe an object is an attribute. They are associated to entries.

In the IN environment, the service provider is responsible for the management and the administration of the data contained in a DSA. Therefore, the service provider plays the administrator role. He is the administrative authority in X.500 terminology. The service provider enforces the security procedure (authentication and access control).

4.2.2 Use of a limited subset of X.500

The primary purpose of the X.500 series of ITU-T Recommendations is to provide a directory service and not the description of the SCF-SDF interface as SG 11 wants to use them. The X.500 functionality covers more than the functionality needed for this capability set. The Operation Procedure, Parameters, and ASN.1 definitions try to indicate which aspects of the Directory Abstract Service should be considered and supported by implementors within the scope of this IN capability set. It also mentions the attitude to adopt when a non-supported parameter is received.

Profiling is used as a means to present the status of the different parameters. For convenience and clarity, this profile is defined using ASN.1 subtyping facilities; however, these definitions do not form a protocol specification. The ASN.1 module is described in subclause 12.5 for information. This simply indicates which parameters should not be sent for an implementation. It does not change the behaviour of the receiving entity which shall still be capable of decoding values which conform to the original definition of the Directory Abstract Service. Nevertheless, elements that are excluded by subtyping should be ignored.

It is important to mention that the number of parameters carried in a message should be minimized, because each of them is associated to a load in the signalling traffic and to some processing time. This is the reason why the parameters are removed unless they are absolutely necessary when they are sent. On reception, removed parameters should not be treated but should be understood by the receiving entity. This allows the extensions of the profile in the future according to its actual description in the third edition of the Directory.

Several assumptions were used to design the Directory Abstract Service profile. They are as follows:

Assumption 1: The version of the Directory Abstract Service used for this capability set is the third edition. The parameters only used for the 1988 version shall be ignored. Functionalities that might be needed in future Capabilities Sets should be at least considered if not supported.

Assumption 2: The alias entries in IN are just a means to provide an alternative name for an object and therefore should be dereferenced when needed.

Assumption 3: An SCF-SDF operation cannot be abandoned. If an operation takes too much time, its timer expires and there is no need to abandon it.

4.3 Versions and the rules for extensibility

The Directory may be distributed and more than two Directory Application Entities may interoperate to service a request. The Directory AEs may be implemented conforming to different editions of the Directory specification of the Directory service which may or may not be represented by different protocol version numbers. The version number is negotiated to the highest common version number between two directly binding Directory AEs.

The rules in ITU-T Q.1248.1 are replaced with the rules in this clause.

4.3.1 Version negotiation

When accepting an association, i.e. binding, utilizing the DAP, the version negotiated shall only affect the point-to-point aspects of the protocol exchanged between the SCF and the SDF to which it is connected. Subsequent requests or responses on the dialogue shall be constrained by the version negotiated.

4.3.2 SCF side

4.3.2.1 Request and response processing at the SCF side

The SCF may initiate requests using the highest edition of the specification of that request it supports. If one or more elements of the request are critical, it shall indicate the extension number(s) in the critical Extensions parameter.

NOTE 1 – If the information the extension replaced in a CHOICE, ENUMERATED or INTEGER (used as ENUMERATED) type would be essential for proper operation in an SDF implemented according to an earlier edition of the Specification, it is recommended that the extension be marked critical.

When processing a response, an SCF shall:

a) ignore all unknown bit name assignments within a bit string; and

b) ignore all unknown named numbers in an ENUMERATED type or INTEGER type that is being used in the enumerated style, provided the number occurs as an optional element of a SET or SEQUENCE; and

c) ignore all unknown elements in SETs, at the end of SEQUENCEs, or in CHOICEs where the CHOICE is itself an optional element of a SET or SEQUENCE.

   NOTE 2 – Implementations may as a local option ignore certain additional elements in a Directory PDU. In particular, some unknown named numbers and unknown CHOICEs in mandatory elements of SETs and SEQUENCEs can be ignored without invalidating the operation. The identification of such elements is for further study.

d) not consider the receipt of unknown attribute types and attribute values as a protocol violation; and

e) optionally report the unknown attribute types and attribute values to the user.

4.3.2.2 Extensibility rules for error handling at the SCF side

When processing a known error type with unknown indicated problems and parameters, an SCF shall:

a) not consider the receipt of unknown indicated problems and parameters as a protocol violation (i.e. it shall not issue a TC-U-REJECT or abort the dialogue); and

b) optionally report the additional error information to the user.

When processing an unknown error type, an SCF shall:

a) not consider the receipt of unknown error type as a protocol violation (i.e. it shall not issue a TC-U-REJECT or abort the application association); and

b) optionally report the error to the user.

4.3.3 Request processing at the SDF side

If any SDF performing an operation detects an element **criticalExtensions** whose semantic is unknown, it shall return an **unavailableCriticalExtension** indication as an **IN-ServiceError**.

NOTE 1 – If a **criticalExtensions** string with one or more zero values is received, this indicates either that the extensions corresponding to the values are not present or are not critical. The presence of a zero value in a **criticalExtensions** string shall not be inferred as either the presence or absence of the corresponding extension in the APDU.

Otherwise, when processing a request from an SCF, an SDF shall:

a) ignore all unknown bit name assignments within a bit string; and

b) ignore all unknown named numbers in an ENUMERATED type or INTEGER type that is being used in the enumerated style, provided the number occurs as an optional element of a SET or SEQUENCE; and

c) ignore all unknown elements in SETs, at the end of SEQUENCEs, or in CHOICEs where the CHOICE is itself an optional element of a SET or SEQUENCE.

NOTE 2 – Implementations may as a local option ignore certain additional elements in a Directory PDU. In particular, some unknown named numbers and unknown CHOICEs in mandatory elements of SETs and SEQUENCEs can be ignored without invalidating the operation. The identification of such elements is not specified in this capability set.

5 Service Control Function (SCF) Model

Generic SCF model is described in ITU-T Q.1248.1.

6 Service Data Function (SDF) Model

6.1 General

An FE model of the SDF is shown in Figure 1. The purpose of this model is to provide a framework for service data functionality subjects with respect to the SDF. Clause 6.2 describes the detailed SDF architecture, and clause 6.3 clarifies and classifies data types which are handled by the SDF.

The SDF contains and manages the data which are related to service logic processing programs (SLPs) and accessed in the execution of the SLP instances (SLPIs). Therefore, data such as SLP selection data and SCF service data object directory which are accessed before the execution of an SLPI, are not included in the SDF handling data.

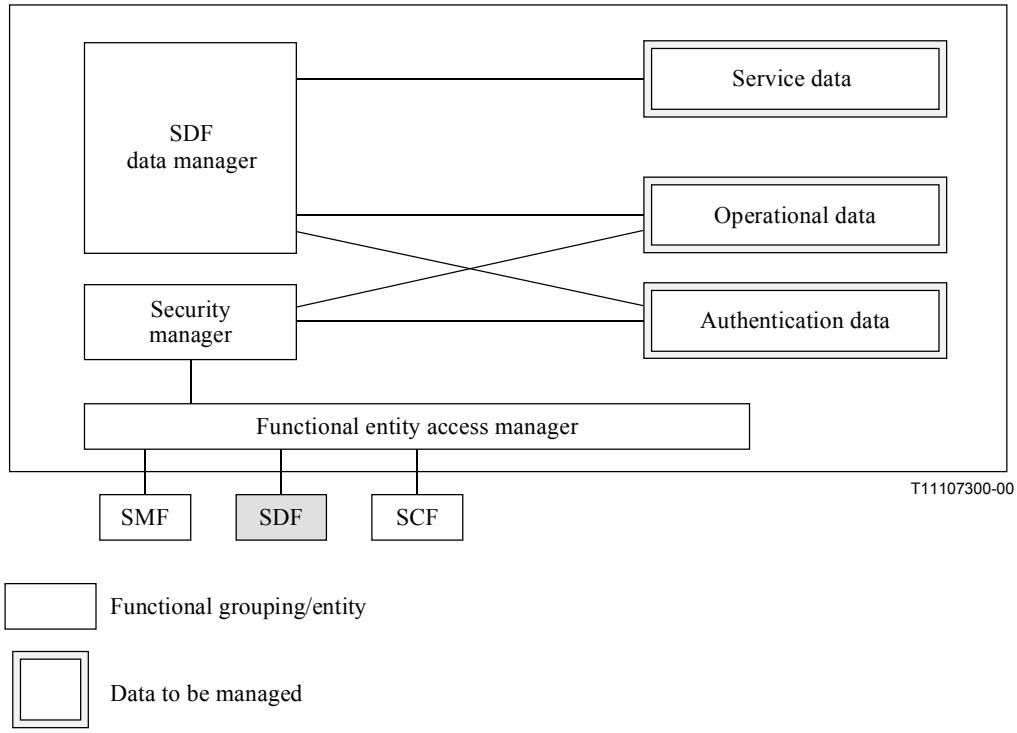

Figure 1/Q.1248.4 – SDF FE model

6.2 SDF components

6.2.1 General

To realize the above defined functionality, the recommended SDF model is shown in Figure 1. Each of the functional entities shown in Figure 1 will be described in the successive clauses. This is not meant to imply any specific implementation.

6.2.2 SDF data manager

The SDF data manager provides the functionality needed for storing, managing and accessing information in the SDF. If, for example, the data are physically structured as a database, the SDF data manager may also handle database accessing language such as an SQL.

6.2.3 Functional entity access manager

The functional entity access manager provides the functionality needed by the SDF data manager to exchange information with other functional entities, i.e. SCF, SDF and SMF, via messages. This message handling functionality should:

– provide reliable message transfer;
– ensure sequential message delivery;
– allow message request/response pairs to be correlated;
– allow multiple messages to be associated with each other; and
– comply with OSI structures and principles.

Here, the functional entity access manager may access other SDFs, because the data distribution in the network can completely be transparent to the SCF.

6.2.4 Security manager

The security manager provides secure access to the different types of data held in the SDF, for example, denied access to the data for unauthenticated users. This functionality should:

– check the access rights of the SCF;

– authenticate users with provided information;

– count the failed authentication attempts for a given user (whether the implementation of this function can be realized in the SDF has not been investigated in this capability set);

– block data access;

– assign user's access rights;

– memorize user's access rights during his request;

– control user's right to access specific data.

The security manager also provides the functionality needed for providing secure access to the functions of the SDF by other Functional Entities (SCF, SDF). This functionality should:

– generate security parameters for outgoing messages to provide data confidentiality, data origin authentication, data integrity for a specific interface;

– verify security parameters for incoming messages to provide data confidentiality, data origin authentication, data integrity for a specific interface. If an incoming message has an invalid security parameter, then the message should be rejected and the invalid access attempt logged. The level of security applied to an interface is determined by which of the security parameter features are active. The selection of active features is performed through interaction with the SMF.

6.3 Data types handled by SDF

The data which are handled by the SDF can be classified into the following types:

1) Authentication data – These data are used to authenticate a user that accesses the database through an SCF, e.g. a PIN code, the value of a counter for failed authentication. The set of authentication data used is associated to a level of access rights.

2) Operational data – These data are not needed by the SLPIs, but are used by the SDF itself for operational and administrative purposes, e.g. references to an object class, access control data.

3) Service data – These data are used for the provision of a service, e.g. a subscriber profile, service provider agreements. These data can be used by several services if necessary.

7 FSM for SCF

7.1 Overview

The general SCF FSM structure is depicted in Figure 2. The function of Service Control Function (SCF) is the execution of service logic, which is provided in the form of Service Logic Processing Programs (SLPs). The SCF also includes the SLP execution supporting functions, such as service logic selection/interaction management, functional entity access management, security management, and SLP provisioning management. One of the FSM managed by SCSM is the SDF FSM which handles interaction with SDF. SCME-FSM includes the TFC (Traffic Control) FSM with SDF.

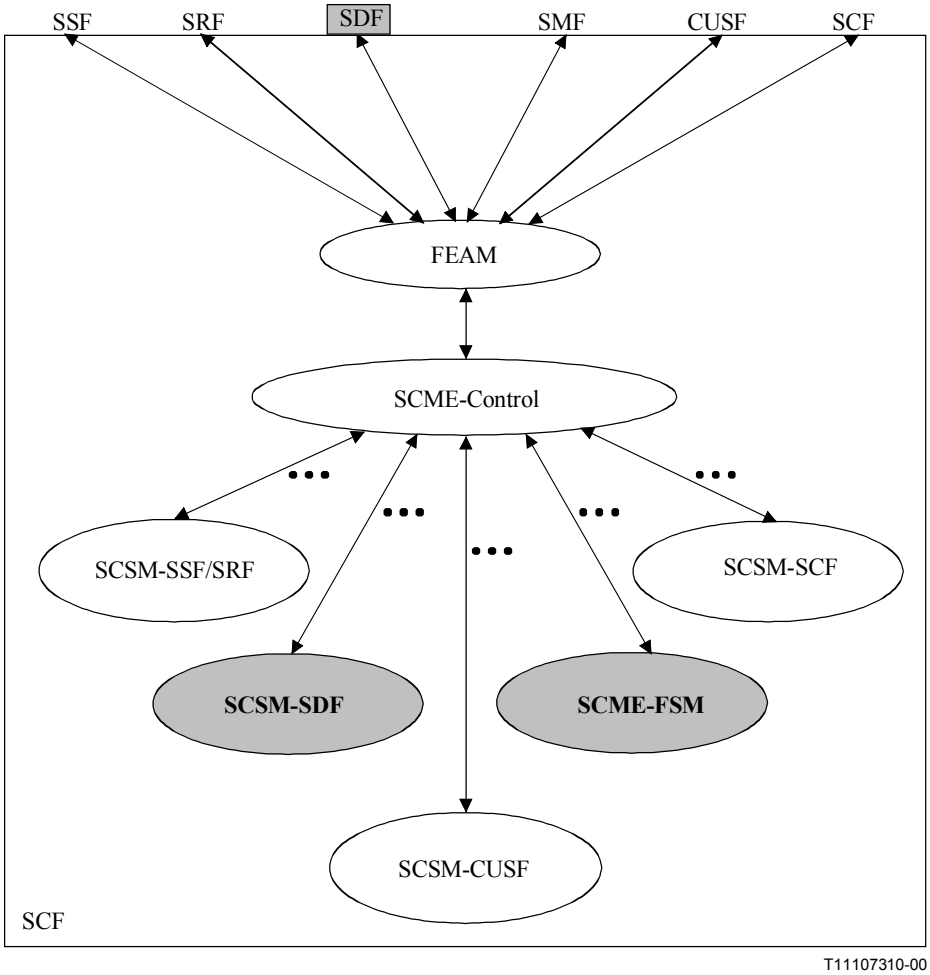

Figure 2/Q.1248.4 – SCF interfaces

7.2 SCME-FSM

The SCME handles a Traffic Flow Control (TFC) related operations in SCF-SDF relationship.

The TFC Respondor finite state model (FSM) which is located in SCME-FSM is depicted in Figure 3.

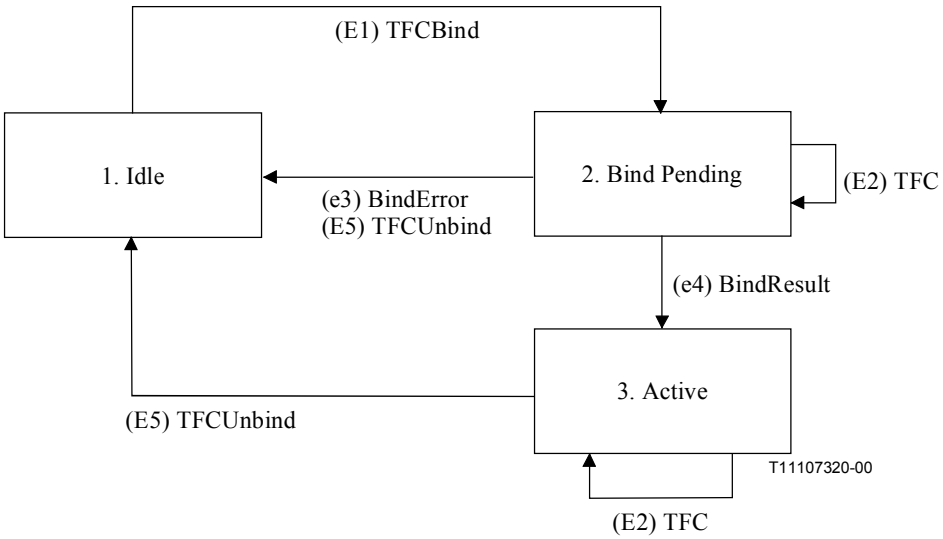

Figure 3/Q.1248.4 – State model for TFC responder

Each state is discussed in one of the following subclauses.

General rules applicable to more than one state are as follows:

In any state, if the dialogue with the Responder is terminated, then the TFC Responder FSM returns to **Idle** state after ensuring that any resources allocated to the dialogue have been deallocated.

7.2.1 State 1: "Idle"

This state is the initial state for the FSM after it has been created by the FE message management. The only event accepted in this state is:

– (E1) TFCBind: This event causes the FSM to notify the Application that a TFC association has been requested by a remote TFC Initiator FSM and causes a transition out of this state to State 2, **Bind Pending**.

7.2.2 State 2: "Bind Pending"

The following events are considered in this state:

– (E2) TFC: This event causes the FSM to save the event to be processed after a transition to another state occurs. The FSM remains in the same state;

– (e3) BindError: This event causes the FSM to construct a **tfcBindError** error message and sends it to the remote TFC Initiator FSM and causes a transition out of this state to State 1, **Idle**;

– (e4) BindResult: This event causes the FSM to construct a **tfcBindResult** message and sends it to the remote TFC Initiator FSM and causes a transition out of this state to State 3, **Active**;

– (E5) TFCUnbind: This event causes the FSM to notify the Application that the association has been terminated by an **IN-Unbind** operation from the remote TFC Initiator FSM and causes a transition out of this state to State 1, **Idle**.

7.2.3 State 3: "Active"

The following events are considered in this state:

– (E2) TFC: This event causes the FSM to notify the Application of the need to process a new traffic flow control request from the remote TFC Initiator FSM. The FSM remains in the same state; and

– (E5) TFCUnbind: This event causes the FSM to notify the Application that the association has been terminated by an **IN-Unbind** operation from the remote TFC Initiator FSM and causes a transition out of this state to State 1, **Idle**.

7.3 SDF-related states (SCSM-SDF)

The interaction with the SDF is possible from any state of the SCF. In the following subclauses, the SDF-related states are specified. The model describes the relationship of one SCF with one SDF. If an SCF needs to open the new dialogue, a new FSM should be instantiated.

In what follows, the states and events are enumerated independent of the rest of the SCSM; the discussion is accompanied by Figure 4.

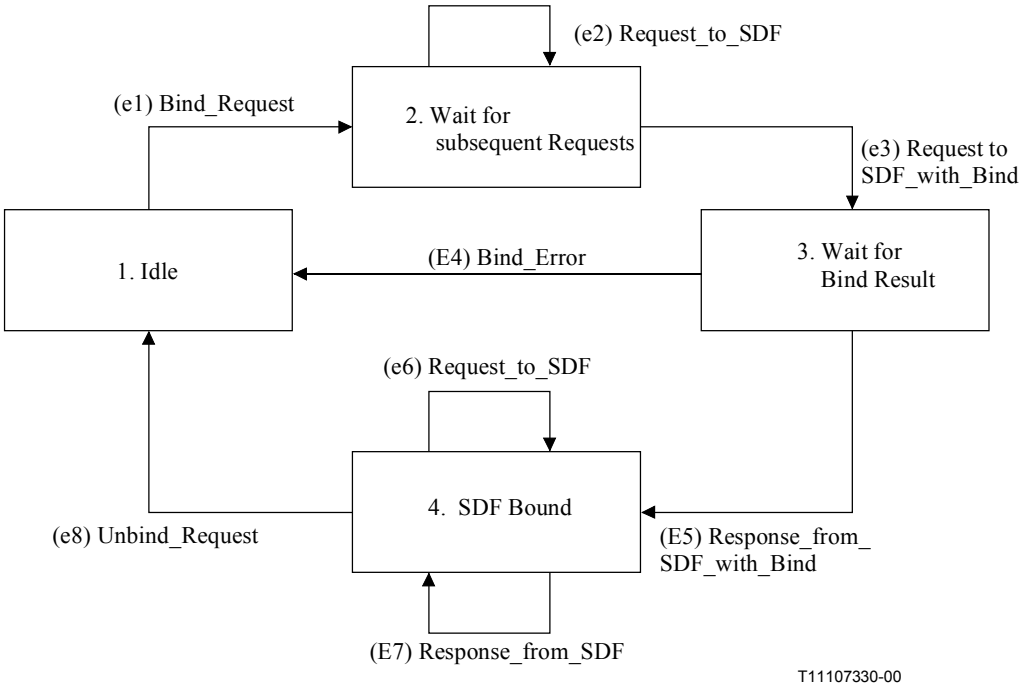

Figure 4/Q.1248.4 – The SDF-related states

7.3.1 State 1: "Idle"

The following event is considered in this state:

– (e1) Bind_Request: This is an internal event, caused by the need of the service logic to create an association with an SDF in order to begin accessing data. This event causes a transition to State 2, **Wait for subsequent Requests**.

7.3.2 State 2: "Wait for subsequent Requests"

In this state, subsequent operations to be sent with the **IN-DirectoryBind** operation (in the same message) to the SDF are expected. The following two events are considered in this state:

– (e2) Request_to_SDF: This is an internal event caused by the reception of an operation. The operation is buffered until the reception of a delimiter (or a timer expiration). The SCSM remains in the same state; and

– (e3) Request_to_SDF_with_Bind: This is an internal event caused by the reception of a delimiter, that indicates the reception of the last operation to be sent. Once the delimiter is received, a message containing the argument of the **IN-DirectoryBind** operation and other operations' arguments, if any, is sent to the SDF. This event causes a transition out of this state to State 3, **Wait for Bind Result**.

7.3.3 State 3: "Wait for Bind Result"

In this state, the SCF is waiting for the response from the SDF. Two events are considered in this state:

– (E4) Bind_Error: This is an external event, caused by the reception of an error to the **IN-DirectoryBind** operation previously issued to the SDF. This event causes a transition out of this state to State 1, **Idle**.

– (E5) Response_from_SDF_with_Bind: This is an external event, caused by the reception of the **IN-DirectoryBind** result combined with the responses to other operations previously issued to the SDF (if any). This event causes a transition to State 4, **SDF Bound**.

7.3.4 State 4: "SDF Bound"

In this state, the SCF has established an access to the SDF, and is waiting for requests to the SDF from the service logic or is waiting for responses to the operations previously issued to the SDF. Three events are considered in this state:

– (e6) Request_to_SDF: This is an internal event, caused by the need of the service logic to access data in the SDF. The SCSM remains in the same state;

– (E7) Response_from_SDF: This is an external event, caused by the reception of responses to the operations previously issued to the SDF. The SCSM remains in the same state; and

– (e8) Unbind_Request: This is an internal event, caused by the need of the service logic to terminate the access to the SDF and by sending an **IN-Unbind** operation. This event causes a transition to State 1, **Idle**.

8 FSM for SDF

8.1 Overview

The structure of the SDF FSMs is illustrated in Figure 5.

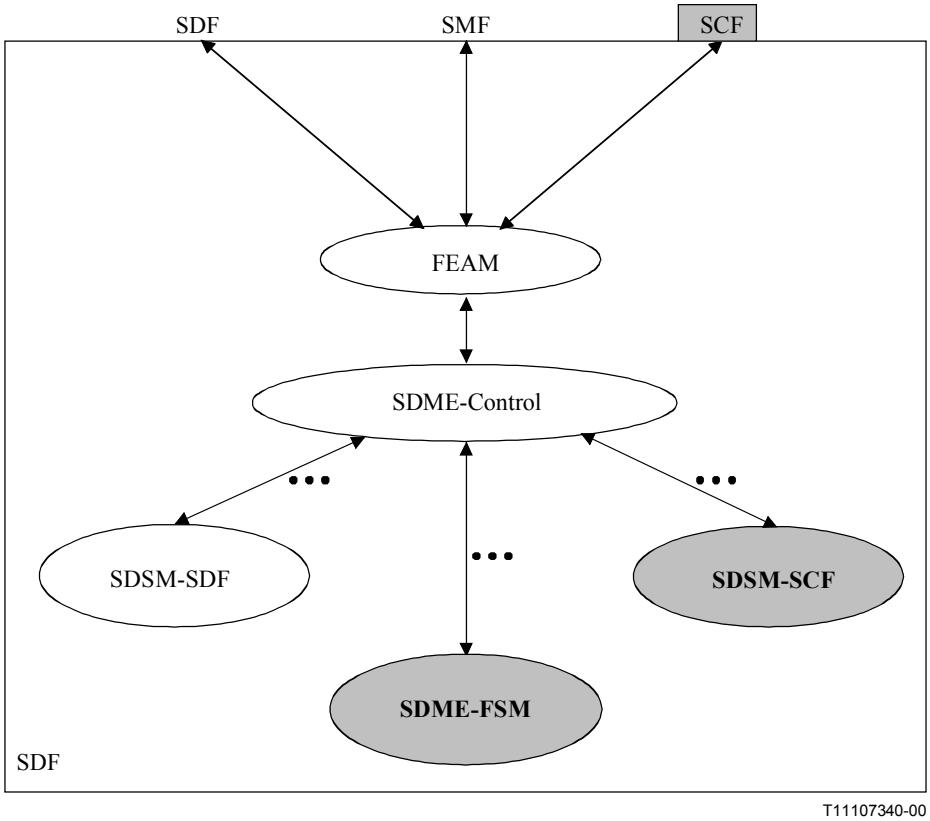

Figure 5/Q.1248.4 – SDF interfaces

The SDF FSM (SDSM-SCF) handles the interaction with the SCF FSM. The SDME-FSM handles the interaction between the SDF and the SDF management functions.

The management functions related to the execution of operations received from the SCF or cooperating SDF are executed by the SDF Management Entity (SDME). The SDME is comprised of an SDME control, traffic control (TFC), and several instances of SDME FSMs. The SDME control interfaces the different SDF FSMs (e.g. SDSM-SCF) and SDME-FSMs respectively as well as the Functional Entity Access Manager (FEAM).

The FEAM provides the low-level interface maintenance functions.

The SDME-Control maintains the associations with the SCF and cooperating SDFs on behalf of all instances of the SDF FSMs (e.g. SDSM-SCF, SDSM-ChI). These instances of the SDF FSMs occur concurrently and asynchronously as SDF-related events occur, which explains the need for a single entity that performs the task of creation, invocation and maintenance of the SDF FSMs. In particular, the SDME-Control performs the following tasks:

1) interprets the input messages from other FEs and translates them into corresponding SDF FSM events;

2) translates the SDF FSM outputs into corresponding messages to other FEs;

3) captures asynchronous activities related to management or supervisory functions in the SDF and creates an instance of an SDME-FSM: for example, management invocation of a shadowing procedure between network operators. In this case, the SDME-Control will create an instance of the SDME-FSM to handle this management-related operation.

8.2 SDME-FSM

The SDME handles a Traffic Flow Control (TFC) related operations in SCF-SDF relationship.

The TFC Initiator finite state model (FSM) which is located in SDME-FSM is depicted in Figure 6.

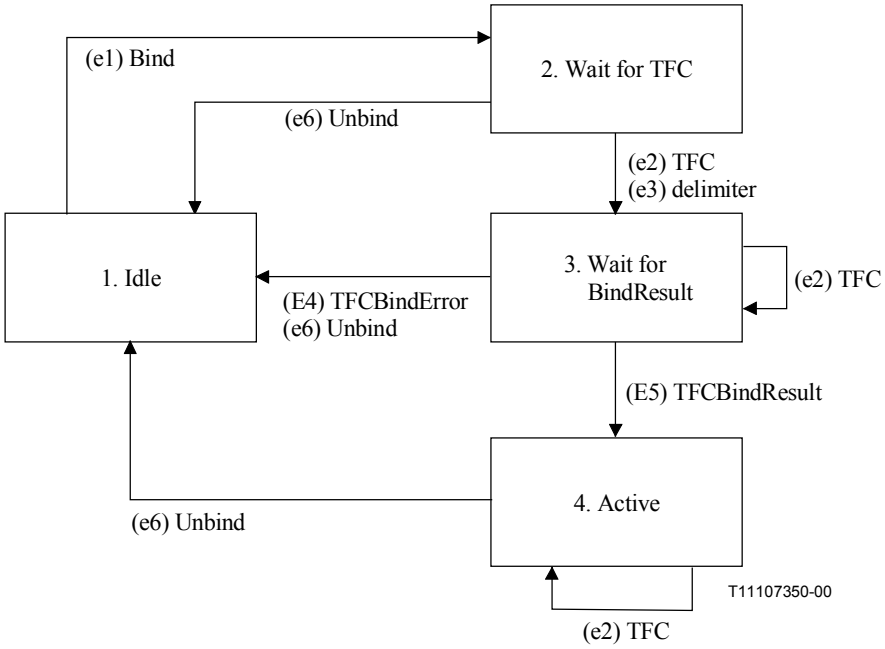

Figure 6/Q.1248.4 – State model for TFC initiator

Each state is discussed in one of the following subclauses.

General rules applicable to more than one state are as follows:

In any state, if the dialogue with the Respondor is terminated, then the TFC Initiator FSM returns to **Idle** state after ensuring that any resources allocated to the dialogue have been deallocated.

8.2.1 State 1: "Idle"

This state is the initial state for the FSM after it has been created by the FE message management. The only event accepted in this state is:

– (e1) Bind: On receipt of this event the FSM constructs a **TFCBind** operation. This event causes a transition out of this state to State 2, **Wait for TFC**.

8.2.2 State 2: "Wait for TFC"

In this state, a Bind request has been received from the Application and a **TFCBind** operation has been constructed, but not sent. Three events are considered in this state:

– (e2) TFC: This event causes a **TrafficFlowControl** operation to be constructed. The **TFCBind** and **TrafficFlowControl** operations are sent to the remote TFC Respondor FSM and the event causes a transition out of this state to State 3, **Wait for BindResult**;

– (e3) delimiter: This event causes the constructed **TFCBind** operation to be sent to the remote TFC Respondor FSM and the event causes a transition out of this state to State 3, **Wait for BindResult**; and

– (e6) Unbind: This event causes the constructed **TFCBind** operation to be discarded and causes a transition out of this state to State 1, **Idle**.

8.2.3 State 3: "Wait for BindResult"

The following events are considered in this state:

– (e2) TFC: This event causes the FSM to save the event to be processed after a transition to another state occurs. The FSM remains in the same state;

– (E4) TFCBindError: This event causes the FSM to notify the Application that the association establishment failed and causes a transition out of this state to State 1, **Idle**;

– (E5) TFCBindResult: This event causes the FSM to notify the Application that the association establishment succeeded and transitions out of this state to State 4, **Active**; and

– (e6) Unbind: This event causes the FSM to construct and send an **IN-Unbind** to the remote TFC Respondor FSM and causes a transition out of this state to State 1, **Idle**.

8.2.4 State 4: "Active"

The following events are considered in this state:

– (e2) TFC: This event causes a **TrafficFlowControl** operation to be constructed and sent to the remote TFC Respondor FSM. The FSM remains in the same state; and

– (e6) Unbind: This event causes the FSM to construct and send an **IN-Unbind** to the remote TFC Respondor FSM and causes a transition out of this state to State 1, **Idle**.

8.3 SDF state transition model for SCF-related states

The SDF's job relating to interactions with the SCF is to (synchronously) respond to every request from the SCF after the Bind procedure. The respective finite state model (FSM) is depicted in Figure 7.

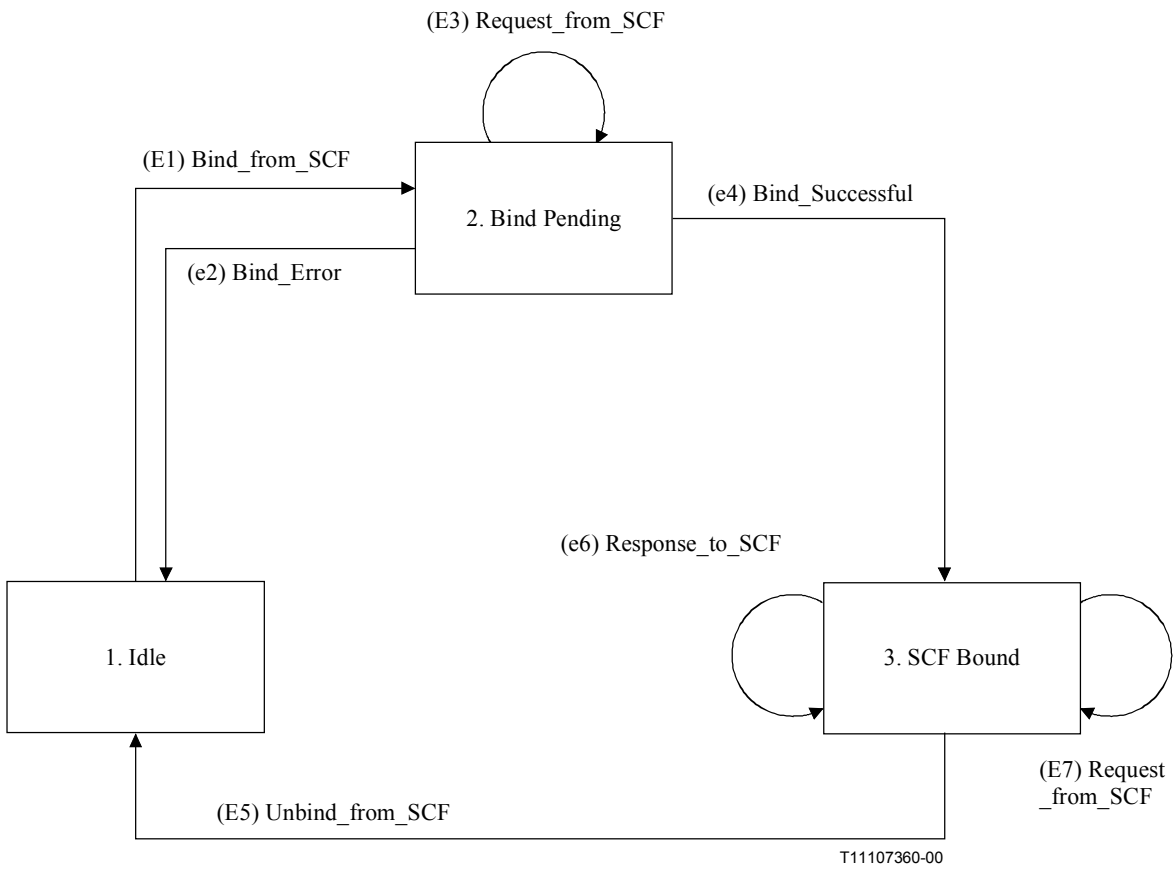

Figure 7/Q.1248.4 – The SDF FSM

Each state is discussed in one of the following subclauses.

General rules applicable to more than one state are as follows:

In any state, if the dialogue with the SCF is terminated, then the SDF FSM returns to **Idle** state after ensuring that any resources allocated to the dialogue have been deallocated.

8.3.1 State 1: "Idle"

The only event accepted in this state is:

– (E1) Bind_from_SCF: This is an external event caused by the reception of **IN-DirectoryBind** operation from the SCF. This event causes a transition out of this state to State 2, **Bind Pending**.

8.3.2 State 2: "Bind Pending"

In this state, a Bind request has been received from the SCF. The SDF is performing the SCF access control procedures behind the **IN-DirectoryBind** operation (e.g. access authentication). There may also be a case such that the **IN-DirectoryBind** operation is a dummy one. Then, the access authentication is not required. Three events are considered in this state:

– (e2) Bind_Error: This is an internal event, caused by the failure of the **IN-DirectoryBind** operation previously issued to the SDF. This event causes a transition out of this state to State 1, **Idle** and an **IN-DirectoryBindError** error is returned to the invoking SCF;

– (E3) Request_from_SCF: This is an external event, caused by the reception of operations before the result from the **IN-DirectoryBind** operation is determined.

  It involves one of the following operations:

  - **IN-Search**;
  - **IN-AddEntry**;
  - **IN-RemoveEntry**;
  - **IN-ModifyEntry**; and
  - **IN-Execute**.

  The operations are stored and the SDSM remains in the same state. When a transition occurs to another state, the operations are re-examined as if they had occurred in that state; and

– (e4) Bind_Successful: This is an internal event, caused by the successful completion of the **IN-DirectoryBind** operation previously issued to the SDF. This event causes a transition out of this state to State 3, **SCF Bound**.

8.3.3 State 3: "SCF Bound"

In this state, the access of the SCF to the SDF was authorized and operations coming from the SCF are accepted. Besides waiting for requests from the SCF, the SDF can send in that state responses to previously issued operations. Three events are considered in this state:

– (E5) Unbind_from_SCF: This is an external event, caused by the reception of the **IN-Unbind** operation from the SCF. The SCF-SDF association is ended and all associated resources are released. This event causes a transition out of this state to State 1, **Idle**;

– (e6) Response_to_SCF: This is an internal event, caused either by the completion of the operations previously issued by the SCF or by generation of a referral error to the SCF. Responses or referrals are sent to the SCF. The SDSM remains in the same state; and

– (E7) Request_from_SCF: This is an external event, caused by the reception of a request from the SCF to the SDF.

It involves one of the following operations:

- **IN-Search**;
- **IN-AddEntry**;
- **IN-RemoveEntry**;
- **IN-ModifyEntry**; and
- **IN-Execute**.

The SDSM remains in the same state.

9 Detailed operations procedure

In 9.1.2, "Parameters", some parameters or subparameters are in parentheses and *italic*. These are used in X.500-related specification and not used in this IN capability set.

9.1 IN-Execute procedure

9.1.1 General description

This operation performs a sequence of execution steps, according to a predefined method, using input information and returns result information. Each step is either a DAP operation (that could be an IN-Execute operation), the execution of an algorithm or a decision test.

This operation is used to execute a method on an Entry of the SDF resident DIT and to return selected information determined from the result of the method. A method is used to encapsulate a sequence of complex SDF operations into a single SCF-SDF operation. It does this by associating a data access script with the entry method in the SDF. This script can involve one or more of any operations that can be invoked on the SCF-SDF interface, along with additional logic required to make subsequent operations. This additional logic may include decision-making and data manipulation. The script may use part of the input value parameter as a parameter of any of the internal operations invoked by the script.

The SCF continues to provide service-specific logic and to command call control functions in the SSF.

The parameters of the individual DAP operations are taken from the input parameters and the results of previous operations and/or the output of the algorithms associated with the method. The output parameters are taken from the results of the individual operations. The IN-Execute operation is considered to be an atomic operation.

9.1.2 Parameters

9.1.2.1 Argument parameters

The operation argument consists of the following parameters. These parameters are defined in clause 10.

– object:

  This parameter identifies the entry in the DIT from/on which the method is to be executed;

– method-id;

– input-assertions:

  - type;
  - values;

- valuesWithContext:
  - value;
  - contextList;
    - context;

– specific-input;

– in-CommonArguments.

9.1.2.2 Result parameters

The operation result consists of the following parameters. These parameters are defined in clause 10.

– method-id;

– output-assertions:
  - type;
  - values;
  - valuesWithContext:
    - value;
    - contextList:
      - context;

– specific-output;

– commonResults.

9.1.3 Invoking entity (SCF)

9.1.3.1 Normal procedure

SCF Precondition:

1) SCSM: "SDF Bound" or "Wait for Subsequent Requests".

SCF Postcondition:

1) SCSM: "SDF Bound".

When the SCSM is in the state "Wait for Subsequent Requests", this operation causes a transition to the same state. The operation is not sent until the application process indicates with a delimiter. When the delimiter is received, the operation is sent to the SDF in a message containing an IN-DirectoryBind argument. The SCSM waits for the response from the SDF. The response to the operation is received with the Bind result. If the Bind is successful, the SCSM transits to the state "SDF Bound" and the reception of the response (return result, return error, etc.) transits to the same state. If the Bind is not successful, the SCSM transits to the state "Idle" and the response is discarded.

When the SCSM is in the state "SDF Bound", this operation causes a transition to the same state and the SCSM waits for the response from the SDF. The reception of the response causes a transition to the same state.

9.1.3.2 Error handling

Generic error handling for the operation-related errors is described in clause 11 and in ITU-T X.511. The TCAP services that are used for reporting operating errors are described in clause 12.

9.1.4 Responding entity (SDF)

9.1.4.1 Normal procedure

SDF Precondition:

1) SDSM: "SCF Bound" or "Bind Pending".

SDF Postcondition:

1) SDSM: "SCF Bound".

When this operation is received in the state "Bind Pending", the SDF does not proceed to the operation until an IN-DirectoryBind operation has been successfully executed. It remains in the same state.

When this operation is received in the state "SCF Bound", the SDF waits for sending back the response of the operation.

Before retrieving the data as specified in the operation parameters, the SDF takes the following actions:

– the SDF verifies that the object accessed by the request exists;
– the SDF verifies that the method referenced in the operation exists in the object, and that the argument is of the correct type;
– the SDF verifies that the user on behalf of whom the request is performed has sufficient access rights to execute the method on the entry.

If these actions are successfully executed, the SDF executes the data access script associated with the entry method in the SDF. Before each internal operation initiated by the script is performed, the SDF takes the following actions:

– the SDF may verify that the user on behalf of whom the request is performed has sufficient access rights to perform the operation;
– the SDF may verify that object, attributes or methods on which an operation should be performed exists in the DIT.

If all of the specified actions indicated above are successfully executed, the SDF returns the result of the script to the SCF.

9.1.4.2 Error handling

Generic error handling for the operation related errors is described in clause 11 and in ITU-T X.511.

If basic access control is in effect for the entry on which the method is being executed, the following sequence of access controls applies.

– ExecuteMethod permission is required to the entry on which the method is being executed. If permission is not granted, the operation fails in accordance with ITU-T X.511. In this case a SecurityError with problem insufficientAccessRights or noInformation shall be returned.
– Before each internal operation initiated by a data access script associated with entry method is performed, basic access control is applied as if the operation had been invoked on the SCF-SDF interface. If any of these internal operations fails, the entire operation fails and an executionError with problem executionFailure shall be returned.

If the operation fails due to failure of any other data access logic, an executionError with problem executionFailure shall be returned.

If the operation fails due to an incorrect type or value of the input-value, an executionError with problem missingInputValues shall be returned.

If a request is made to execute a method on a non-existent entry, a NameError with problem noSuchObject shall be returned.

The TCAP services that are used for reporting operating errors are described in clause 12.

9.2 IN-AddEntry procedure

9.2.1 General description

The IN-AddEntry procedure uses the X.511 AddEntry operation.

This is used to request the SDF to add a leaf entry (either an object entry or an alias entry) in the DIT. For a full description of the AddEntry operation, refer to ITU-T X.511.

9.2.2 Parameters

9.2.2.1 Argument parameters

The operation argument consists of the following parameters. These parameters are defined in clause 10.

- object;
- entry;
- targetSystem:

  If this parameter is present, the targetSystem bit in the criticalExtensions subparameter in in-CommonArguments parameter shall be set, indicating that this extension is critical.
- in-CommonArguments:
  - serviceControls:
    - options;
    - priority;
    - (– *timeLimit*);
    - (– *sizeLimit*);
    - (– *scopeOfReferral*);
    - (– *attributeSizeLimit*);
    - manageDSAITPlaneRef;
  - securityParameters;
  - requestor;
  - operationProgress;
  - (• *aliasedRDNs*);
  - criticalExtensions:

    If the "targetSystem" parameter is present, the **targetSystem** bit shall be set, indicating that this extension is critical.
  - referenceType;
  - entryOnly;
  - exclusions;
  - nameResolveOnMaster;
  - operationContexts.

9.2.2.2 Result parameters

The operation result consists of the following alternatives. These parameters are defined in clause 10.

– null; or

– information (commonResults):

  • securityParameters;

  • performer;

  • aliasDereferenced.

9.2.3 Invoking entity (SCF)

9.2.3.1 Normal procedure

Refer to the Normal procedure of Invoking entity (SCF) in IN-Execute operation procedure.

9.2.3.2 Error handling

Generic error handling for the operation related errors is described in clause 11 and in ITU-T X.511, and the TCAP services that are used for reporting operating errors are described in clause 12.

9.2.4 Responding entity (SDF)

9.2.4.1 Normal procedure

SDF Precondition:

1) SDSM: "SCF Bound" or "Bind Pending".

SDF Postcondition:

1) SDSM: "SCF Bound".

When this operation is received in the state "Bind Pending", the SDF does not proceed to the operation until an IN-DirectoryBind operation has been successfully executed. It remains in the same state.

When this operation is received in the state "SCF Bound", the SDF waits for sending back the response of the operation.

Before adding the new entry item, the SDF takes the following actions:

– verify that the superior object to which the entry should be added exists in the SDF;

– verify that the entry does not already exist in the SDF;

– verify that the access rights to add the entry and each of its components (attributes and values) are sufficient;

– verify that the entry conforms to the Directory schema.

After the specified actions indicated above are successfully executed, the entry is added into the SDF database. A null result is returned to the SCF.

9.2.4.2 Error handling

Generic error handling for the operation-related errors is described in clause 11 and in ITU-T X.511, and the TCAP services that are used for reporting operating errors are described in clause 12.

9.3 IN-DirectoryBind procedure

9.3.1 General description

The IN-DirectoryBind procedure uses the X.511 DirectoryBind operation.

This IN-DirectoryBind is used by the SCF to create an association between an SCF and an SDF on behalf of the end user. It carries the authentication information of the end user if any. For a full description of the DirectoryBind operation, refer to ITU-T X.511.

9.3.2 Parameters

9.3.2.1 Argument parameters

The operation argument consists of the following parameters. These parameters are defined in clause 10.

– credentials:
  - simple; or
  - strong; or
  - externalProcedure; or
  - spkm;
– versions.

9.3.2.2 Result parameters

The operation result consists of the following parameters. These parameters are defined in clause 10.

– credentials:
  - simple; or
  - strong; or
  - externalProcedure; or
  - spkm;
– versions.

9.3.3 Invoking Entity (SCF)

9.3.3.1 Normal procedure

SCF Precondition:

1) SCSM: "Idle".

SCF Postconditions:

1) SCSM: "SDF Bound" in case of success.
2) SCSM: "Idle" in case of failure.

When the SCSM is in the state "Idle", the SLPI can request to send an IN-DirectoryBind operation. This causes a transition to the state "Wait for Subsequent Requests" and the other operations are awaited. The SCSM remains in the state "Wait for Subsequent Requests" and the operation is not sent until the application process indicates with a delimiter. The reception of the delimiter causes a transition to the state "Wait for Bind Result". The IN-DirectoryBind operation and the subsequent operations are sent to the SDF. The SCSM waits for the response from the SDF. The reception of the response to the IN-DirectoryBind operation causes a transition to the state "SDF Bound" if the result of the operation is positive. Otherwise the reception of an error moves back the SCSM to the state "Idle".

9.3.3.2 Error handling

Generic error handling for the operation-related errors is described in clause 11 and in ITU-T X.511, and the TCAP services that are used for reporting operating errors are described in clause 12.

9.3.4 Responding Entity (SDF)

9.3.4.1 Normal procedure

SDF Precondition:

1) SDSM: "Idle".

SDF Postconditions:

1) SDSM: "SCF Bound" (success).
2) SDSM: "Idle" (failure).

When the SDSM is in the state "Idle", an "IN-DirectoryBind" operation from the SCF is received, and a transition to state "Bind Pending" occurs. The SDF performs the operation according to the contents of the argument. Once the SDF has completed the operation, the result or error indication is returned to the SCF. The SDF returns to the state "Idle" if the Bind fails or to the state "SCF Bound" if the Bind is successful. Should the Bind request succeed, the result returned may consist of credentials of the DirectoryBindResult. These credentials allow the user to establish the identity of the Directory. They allow information identifying the SDF (that is directly providing the Directory service) to be conveyed to the SCF. The credentials are of the same form as those supplied by the user.

9.3.4.2 Error handling

Generic error handling for the operation-related errors is described in clause 11 and in ITU-T X.511, and the TCAP services that are used for reporting operating errors are described in clause 12.

9.4 IN-ModifyEntry procedure

9.4.1 General description

The IN-ModifyEntry procedure uses the X.511 ModifyEntry operation.

This is used to request the SDF to make one or several modifications to a data object. The modifications concern the attributes and their values of which the object is composed. The type of modifications to perform is given in the operation argument provided by the SCF. The modifications do not concern the values of the attributes used to identify the object (i.e. the object name). For a full description of the ModifyEntry operation, refer to ITU-T X.511.

9.4.2 Parameters

9.4.2.1 Argument parameters

The operation argument consists of the following parameters. These parameters are defined in clause 10.

– object;
– changes;
– selection (in-EntryInformationSelection):

  The selection parameter specifies some attributes and values to be returned.

  - attributes;
  - infoTypes;
  - extraAttributes;

- contextSelection;
- returnContexts;

– in-CommonArguments.

9.4.2.2 Result parameters

The operation result consists of the following parameters. These parameters are defined in clause 10.

– null:

  If no information is to be retrieved by the operation, the null result is returned. This means no result parameters; or

– information:

  - entry (in-EntryInformation):

    If selection was specified in the IN-ModifyEntry operation, this entry subparameter shall be returned.

    – name;
    – fromEntry;
    – information:
      (• *attributeType or*);
      • attribute;
    – incompleteEntry;
    – partialName;

  - commonResults.

9.4.3 Invoking entity (SCF)

9.4.3.1 Normal procedure

Refer to the Normal procedure of Invoking entity (SCF) in IN-Execute operation procedure.

9.4.3.2 Error handling

Generic error handling for the operation-related errors is described in clause 11 and in ITU-T X.511, and the TCAP services that are used for reporting operating errors are described in clause 12.

9.4.4 Responding entity (SDF)

9.4.4.1 Normal procedure

SDF Precondition:

1) SDSM: "SCF Bound" or "Bind Pending".

SDF Postcondition:

1) SDSM: "SCF Bound".

When this operation is received in the state "Bind Pending", the SDF does not proceed to the operation until an IN-DirectoryBind operation has been successfully executed. It remains in the same state.

When this operation is received in the state "SCF Bound", the SDF waits for sending back the response of the operation.

Before updating the different attributes specified in the operation parameters, the SDF shall take the following actions:

– verify that the object accessed by the request exists;
– verify that the user on behalf of whom the request is performed has sufficient access rights to modify the object for each elementary modification contained in the operation;
– verify that each attribute or value on which an operation should be performed exists in the object;
– verify that the proposed modifications are compatible with the object class of the object or with the syntax of the attribute.

After the specified actions indicated above are successfully executed for all the modifications requested in the operation, all the modifications for a same attribute take place in the order given in the "changes" parameter. A result is returned to the SCF.

9.4.4.2 Error handling

Generic error handling for the operation-related errors is described in clause 11 and in ITU-T X.511, and the TCAP services that are used for reporting operating errors are described in clause 12.

9.5 IN-RemoveEntry procedure

9.5.1 General description

The IN-RemoveEntry procedure uses the X.511 RemoveEntry operation.

This is used to request the SDF to remove a leaf entry (either an object entry or an alias entry) from the DIT. For a full description of the RemoveEntry operation, refer to ITU-T X.511.

9.5.2 Parameters

9.5.2.1 Argument parameters

The operation argument consists of the following parameters. These parameters are defined in clause 10.

– object;
– in-CommonArguments.

9.5.2.2 Result parameters

The operation result consists of the following parameters. These parameters are defined in clause 10. As for information parameter in this operation, refer to "CommonResults" parameter definition.

– null; or
– information (commonResults).

9.5.3 Invoking entity (SCF)

9.5.3.1 Normal procedure

Refer to the Normal procedure of Invoking entity (SCF) in IN-Execute operation procedure.

9.5.3.2 Error handling

Generic error handling for the operation-related errors is described in clause 11 and in ITU-T X.511, and the TCAP services that are used for reporting operating errors are described in clause 12.

9.5.4 Responding entity (SDF)

9.5.4.1 Normal procedure

SDF Precondition:

1) SDSM: "SCF Bound" or "Bind Pending".

SDF Postcondition:

1) SDSM: "SCF Bound".

When this operation is received in the state "Bind Pending", the SDF does not proceed to the operation until an IN-DirectoryBind operation has been successfully executed. It remains in the same state.

When this operation is received in the state "SCF Bound", the SDF waits for sending back the response of the operation.

Before removing the entry item, the SDF takes the following actions:

– verify that the object to be removed exists and is a leaf entry;

– verify that the access rights to remove the entry are sufficient.

After the specified actions indicated above are successfully executed, the entry is removed from the SDF database. A null result is returned to the SCF.

9.5.4.2 Error handling

Generic error handling for the operation-related errors is described in clause 11 and in ITU-T X.511, and the TCAP services that are used for reporting operating errors are described in clause 12.

9.6 IN-Search procedure

9.6.1 General description

The IN-Search procedure uses the X.511 Search operation.

This is used to search a portion of the SDF resident DIT for entries of interest and to return selected information from those entries. For a full description of the Search operation, refer to ITU-T X.511.

9.6.2 Parameters

9.6.2.1 Argument parameters

The operation argument consists of the following parameters. These parameters are defined in clause 10.

– baseObject;

– subset;

– filter:

  The filter parameter is used to eliminate entries from the search space. Only the filter parameter should be sent because the extendedFilter parameter is not used in this IN capability set.

– searchAliases:

  The searchAliases parameter indicates whether the aliases encountered in the search space (except the base object) should be considered. Since in IN aliases are always dereferenced when searching, this parameter should be used only with its default value.

– selection (in-EntryInformationSelection):

  The selection parameter indicates what information from the entries, e.g. types and values, is requested.

(– *pagedResults*)

  The pagedResults parameter is used to request a page by page result. The pagedResults parameter is used to present the results of a search operation in a page format. This type of information is not needed in the IN capability set since the SCF treats the results.

– matchedValuesOnly

(– *extendedFilter*):

  The extendedFilter parameter was added in the 1997 version of the Directory for compatibility reasons and should therefore not be sent.

– checkOverspecified:

  This parameter requests to return an overspecFilter item in partialOutcomeQualifier if the result of the IN-Search operation is empty and the Directory is able to determine that this is due to the filter being overspecified.

– in-CommonArguments.

9.6.2.2 Result parameters

The operation result consists of the following parameters. These parameters are defined in clause 10.

– searchInfo:
  - name;
  - entries:

    The entries parameter contains the entries that satisfy the filter.

    – in-EntryInformation;
  - partialOutcomeQualifier:
    – limitProblem;
    – unexplored;
    – unknownErrors;
    (– *queryReference*);
    – overspecFilter:

      This parameter is set only when requested by the "checkOverspecified" parameter set to TRUE, which is a default value, in the IN-Search operation argument.
  - commonResults; or

– uncorrelatedSearchInfo.

9.6.3 Invoking entity (SCF)

9.6.3.1 Normal procedure

Refer to the Normal procedure of Invoking entity (SCF) in IN-Execute operation procedure.

9.6.3.2 Error handling

Generic error handling for the operation-related errors is described in clause 11 and in ITU-T X.511, and the TCAP services that are used for reporting operating errors are described in clause 12.

9.6.4 Responding entity (SDF)

9.6.4.1 Normal procedure

SDF Precondition:

1) SDSM: "SCF Bound" or "Bind Pending".

SDF Postcondition:

1) SDSM: "SCF Bound".

When this operation is received in the state "Bind Pending", the SDF does not proceed to the operation until an IN-DirectoryBind operation has been successfully executed. It remains in the same state.

When this operation is received in the state "SCF Bound", the SDF waits for sending back the response of the operation.

Before retrieving the data as specified in the operation parameters, the SDF takes the following actions:

– verify that the objects accessed by the request exist;
– verify that the user on behalf of whom the request is performed has sufficient access rights to access the objects and attributes reached during the execution of the operation;
– verify that attributes on which an operation should be performed exist in the object.

After the specified actions indicated above are successfully executed, the SDF returns all the possible attributes that satisfy the retrieval criteria to the SCF.

9.6.4.2 Error handling

Generic error handling for the operation-related errors is described in clause 11 and in ITU-T X.511, and the TCAP services that are used for reporting operating errors are described in clause 12.

9.7 IN-Unbind procedure

9.7.1 General description

This operation is used by the SCF at the end of a period of accessing the Directory.

This operation is also used by the SDF to end a TFC association between an SDF and an SCF.

9.7.1.1 Parameters

None.

9.7.2 Invoking entity (SCF in Directory Access)

9.7.2.1 Normal procedure

SCF Precondition:

1) SCSM: "SDF Bound".

SCF Postcondition:

1) SCSM: "Idle".

The SCSM has previously initiated a successful IN-DirectoryBind operation to the SDF directory. It is in state "SDF Bound". The service logic determines that the access to the SDF is to be terminated. It issues an IN-Unbind operation that causes the SCSM to transit back to the state "Idle".

9.7.2.2 Error handling

This operation does not have operation-related errors.

9.7.3 Responding entity (SDF in Directory Access)

9.7.3.1 Normal procedure

SDF Precondition:

1) SDSM: "SCF Bound".

SDF Postcondition:

1) SDSM: "Idle".

An IN-DirectoryBind operation was previously issued and the SDSM is in the state "SCF Bound" waiting for a request from the SCF and/or performing an operation. The reception of the IN-Unbind operation causes a transition to the state "Idle".

9.7.3.2 Error handling

This operation does not have operation-related errors.

9.7.4 Invoking entity (SDF in traffic flow control)

9.7.4.1 Normal procedure

SDF Preconditions:

a) SDME-FSM: "Wait for TFC";

b) SDME-FSM: "Wait for BindResult" or SDME-FSM: "Active"

SDF Postconditions:

The SDF determines that the TFC association with the SCF is to be terminated and takes the following action:

a) If the SDME-FSM is in state "Wait for TFC", the SDME-FSM transits back to the state "Idle".

b) If the SDME-FSM is in state "Wait for BindResult" or state "Active", the SDME-FSM constructs and sends an IN-Unbind operation to the SCF and transits back to the state "Idle".

9.7.4.2 Error handling

This operation does not have operation-related errors.

9.7.5 Responding Entity (SCF in traffic flow control)

9.7.5.1 Normal procedure

SCF Preconditions:

– SCME-FSM: "Bind Pending"; or

– SCME-FSM: "Active".

SCF Postcondition:

– SCME-FSM: "Idle".

9.7.5.2 Error handling

This operation does not have operation-related errors.

9.8 TFCBind procedure

9.8.1 General description

The TFCBind procedure uses the X.500 DirectoryBind operation.

This is used by the TFC initiator FE (SDF, SCF) to create an association between itself and a TFC Respondor FE. It carries the authentication information of the initiator FE if any. For a full description of the DirectoryBind operation, refer to ITU-T X.511.

9.8.1.1 Parameters

Refer to the IN-DirectoryBind procedure because all the parameters of this operation are equivalent to those of the IN-DirectoryBind operation.

9.8.2 Invoking entity (SDF)

9.8.2.1 Normal procedure

SDF Precondition:

1) SDME-FSM: "Idle".

SDF Postconditions:

1) SDME-FSM: "Active" in case of success.

2) SDME-FSM: "Idle" in case of failure.

The SDME-FSM is created when the SDF detects the need to apply traffic flow control (TFC) to a SCF and there is no existing SDME-FSM associated with the SCF.

Once created, and the SDME-FSM is in the state "Idle", a TFCbind operation can be sent. This causes a transition to the state "Wait for TFC", and the other operations are awaited. The SDME-FSM remains in the state "Wait for TFC" until either a sending of the first TFC operation is requested or a delimiter is indicated. Either event causes a transition to the state "Wait for BindResult". The TFCBind operation (and first TFC operation if present) is sent to the SCF. The SDME-FSM waits for the response from the SCF.

The reception of the result to the TFCBind operation causes a transition of the SDME-FSM to the state "Active". Otherwise the reception of an error moves the SDME-FSM back to the state "Idle".

9.8.2.2 Error handling

Generic error handling for the operation-related errors is described in clause 11, and the TCAP services that are used for reporting operating errors are described in clause 12.

9.8.3 Responding entity (SCF)

9.8.3.1 Normal procedure

SCF Precondition:

1) SCME-FSM: "Idle".

SCF Postconditions:

1) SCME-FSM: "Active" (success).

2) SCME-FSM: "Idle" (failure).

The reception of a "TFCBind" operation from an SDF causes the creation of an SCME-FSM within the SCF. Once the SCME-FSM is in the initial state "Idle", it receives the operation and transits to the state " Bind Pending". The SCF performs the TFCBind operation according to the contents of the argument. Once the SCF has completed the operation, the result or error indication is returned to the

SDF. The SCF returns to the state "Idle" if the TFCBind fails or to the state "Active" if the TFCBind is successful.

9.8.3.2 Error handling

Generic error handling for the operation-related errors is described in clause 11, and the TCAP services that are used for reporting operating errors are described in clause 12.

9.9 TrafficFlowControl procedure

9.9.1 General description

This operation is used to request the SCF to reduce the rate at which specific data base requests are sent to the SDF.

9.9.2 Parameters

9.9.2.1 Argument parameters

The operation argument consists of the following parameters. These parameters are defined in clause 10.

– tfcCriteria:

  The "sdf" is the only alternative used in this operation.

  - sdf:
    - object;

– duration:

  The value means as follows:

  0: gapping is to be removed;

  –1: an infinite duration;

  –2: a network specific duration; and

  other values: duration in seconds.

– controlType:

  The value means as follows:

  - "sCPOverloaded": when the SDF in the SCP is accessed, an automatic congestion detection and control mechanism has detected a congestion situation.
  - "manuallyInitiated": the service and or network/service management centre has detected congestion situation, or any other situation that requires manually initiated controls. The "manuallyInitiated" will have priority over "sCDOverloaded" traffic flow control.
  - "destinationOverload": an automatic congestion detection and control mechanism in the SDP, which is a destination physical entity, has detected a congestion situation.

– security;

– wasChained:

  The value is set to "TRUE" if the traffic source was relayed through an intermediate point.

9.9.3 Invoking entity (SDF)

9.9.3.1 Normal procedure

SDF Preconditions:

1) a) SDME-FSM: "Wait for TFC";
   b) SDME-FSM: "Wait for BindResult"; or SDME-FSM: "Active".
2) The SDME-FSM detects an overload condition persists and traffic flow control has to be initiated at the SCF; or the SDF receives a manually initiated traffic flow control request.

SDF Postcondition:

1) a) SDME-FSM: "Wait for BindResult";
   b) SDME-FSM: remains in the same state.

A congestion detection and control algorithm monitors the load of SDF resources. After detection of a congestion situation, the parameters for the "TrafficFlowControl" operation are provided.

If the congestion level changes, new "TrafficFlowControl" operations may be sent for active traffic flow controls but with new control values. If no congestion is detected, traffic flow control may be removed.

A manually initiated traffic flow control will prevail over an automatically initiated traffic flow control.

9.9.3.2 Error handling

Operation-related error handling is not applicable, due to class 4 operation.

9.9.4 Responding entity (SCF)

9.9.4.1 Normal procedure

SCF Preconditions:

1) SCME-FSM: "Bind Pending"; or SCME-FSM: "Active".

SCF Postconditions:

1) SCME-FSM: "Active".
2) Traffic flow control for tfcCriteria is activated, or Traffic flow control for tfcCriteria is renewed, or Traffic flow control for tfcCriteria is removed.

If the SCME-FSM is in state "Bind Pending", then the incoming "TrafficFlowControl" operation is stored until the bind process is resolved. If the "TFCBind" operation succeeds, then the "TrafficFlowControl" operation is processed, otherwise, it is discarded.

If the SCME-FSM is in state "Active", then the incoming "TrafficFlowControl" operation is processed.

In general, the manuallyInitiated traffic flow control will prevail over automatically initiated.

The traffic flow control process is stopped if the indicated duration equals ZERO.

9.9.4.2 Error handling

Operation-related error handling is not applicable, due to class 4 operation.

9.10 Operation timer values

Table 1 lists operation timer value range. The definitive timer value may be network-specific and has to be defined by the network operator.

Table 1/Q.1248.4 – Operation timer values

Operation	Timer value
IN-Search	Medium
IN-ModifyEntry	Medium
IN-AddEntry	Medium
IN-RemoveEntry	Medium
IN-Execute	Medium
TrafficFlowControl	Medium
Medium: 1 to 60 seconds	

10 Parameters

This clause defines the parameters used in the operations procedures as specified in clause 9.

10.1 BaseObject

This parameter indicates the object entry (or possibly the root) relative to which a search is to take place. For a detailed definition of this parameter, refer to ITU-T X.511.

10.2 Changes

This parameter indicates a sequence of modifications that are applied in the order specified. For a detailed definition of this parameter, refer to ITU-T X.511.

10.3 CheckOverspecified

This parameter specifies to request the Directory to return the result with an overspecFilter item in partialOutcomeQualifier parameter if the result of the IN-Search operation is empty and the Directory is able to determine that this is due to the filter being overspecified.

10.4 CommonResults

This parameter specifies the common information conveyed by Return Result to qualify the result of each retrieval operation that the Directory can perform. It comprises the following subparameters. For a definition of this parameter and the subparameters, refer to ITU-T X.511.

– securityParameters;

– performer;

– aliasDereferenced.

10.5 ControlType

This parameter indicates the reason for activating database request gapping. For a definition of this parameter, refer to ITU-T Q.1248.2.

10.6 Credentials

This parameter allows the Directory to establish the identity of the user or allows the user to establish the identity of the Directory. It comprises the following alternative subparameters. For a definition of this parameter and the subparameters, refer to ITU-T X.511.

– simple; or

– strong; or

– externalProcedure; or

– spkm.

10.7 Duration

This parameter specifies the total time interval during which database request gapping for the specified gap criteria will be active, that the gapping should be finished, that the interval is infinite, or that the interval is a network specific value.

10.8 Entry

This parameter contains the attribute information which constitutes the entry to be created. It comprises a list of (1 to n) attribute. For a definition of this parameter, refer to ITU-T X.511.

10.9 Filter

This parameter is used to eliminate entries from a search space which are not of interest. For a definition of this parameter, refer to ITU-T X.511.

10.10 IN-CommonArguments

This parameter specifies the common information conveyed for qualifying the invocation of an operation that the Directory can perform. This parameter is based on X.511 "CommonArguments" with some constraints. For the definition of the "CommonArguments", refer to ITU-T X.511.

This parameter comprises the following subparameters. The italic subparameters with parentheses are defined in ITU-T X.511, but not used in this IN capability set.

– serviceControls:

  This subparameter is based on X.511 "serviceControls" with some constraints. For the definition of the "serviceControls", refer to ITU-T X.511. It comprises the following subparameters:

  - options;
  - priority;
  - (• *timeLimit*):

    NOTE 1 – The timeLimit subparameter indicates the maximum elapsed time to fulfil a request. It is redundant with the operation timers of TCAP and therefore is not needed.

  - (• *sizeLimit*):

    NOTE 2 – The sizeLimit subparameter sets some size limits on the results in terms of objects. This is useful when requests are expected to be general (the requestor does not know the structure of the DSA), but in the case of IN, this type of limitation does not seem applicable.

  - (• *scopeOfReferral*);
  - (• *attributeSizeLimit*):

    NOTE 3 – The attributeSizeLimit subparameter sets some size limits on the results in terms of attributes. This is not needed for the same reason of the above sizeLimit.

  - manageDSAITPlaneRef;

– securityParameters;

– requestor;

– operationProgress;

(– *aliasedRDNs*):

NOTE 4 – The aliasedRDNs component is present in the third edition only for compatibility reasons. It should always be omitted in the third edition implementations of the Directory (assumption 1 in clause 4).

– criticalExtensions;

– referenceType;

– entryOnly;

– exclusions;

– nameResolveOnMaster;

– operationContexts.

10.11 Information

This parameter indicates the requested information. It comprises the following subparameters:

– entry:

This parameter indicates the requested information from the specified entry.

This parameter is based on X.511 "EntryInformation" with some constraints. For the definition of the "EntryInformation", refer to ITU-T X.511.

This parameter comprises the following subparameters:

- name;
- fromEntry:

  This parameter indicates if a copy or the entry itself is returned. Since this IN capability set does not use copy mechanisms (assumption 3 in clause 4), only the value "TRUE" should be used. This subparameter should be absent because the value is specified as the default.
- information:

  This parameter contains the relevant information which is returned. It comprises a list of (1 to n) the following parameter. Note that the choice of "attributeType" is not supported in this IN capability set.

  (– *attributeType or*);

  – attribute;
- incompleteEntry;
- partialName;

– commonResults:

For a definition of this parameter, refer to 10.4.

10.12 Input-assertions

This parameter provides a set of attribute values which are used as an input to the method execution. It comprises a list of the following subparameters' sequence.

– type;

– values;

– valuesWithContext:

This subparameter comprises a list of the following subparameters' sequence.

- value;

- contextList:

  This subparameter comprises a list of (1 to n) context. For a definition of the context, refer to ITU-T X.501 "Context".

10.13 MatchedValuesOnly

This parameter indicates that certain attribute values are to be omitted from the returned entry information. For a detailed definition of this parameter, refer to ITU-T X.511.

10.14 Method-id

This parameter indicates the method which is to be executed on the entry. The value of this parameter is identified from the ASN.1 description of the method defined in the data schema. It is unique for the associated object.

10.15 Null

This parameter indicates that no information is to be retrieved by an operation.

10.16 Object

The parameter identifies the entry of the SDF in the Directory Information Tree (DIT) to be added, deleted, modified, or executed. The Object may be an alternative name and may include context information. For a definition of this parameter, refer to ITU-T X.501.

10.17 Output-assertions

This parameter contains attribute values returned as a result of the method execution. It comprises a list of the following subparameters' sequence.

– type;

– values;

– valuesWithContext:

  This subparameter comprises a list of the following sub-parameters' sequence.

  - value;
  - contextList:

    This subparameter comprises a list of (1 to n) contexts. For a definition of the context, refer to ITU-T X.501 "Context".

10.18 SearchAliases

This parameter is used to decide that aliases shall be dereferenced or not.

For a detailed definition of this parameter, refer to ITU-T X.511.

10.19 SearchInfo

This parameter specifies the search result. This parameter is based on X.511 "SearchInfo" with some constraints. For the definition of the "SearchInfo", refer to ITU-T X.511.

This parameter comprises the following subparameters. The italic subparameters with parentheses are defined in ITU-T X.511, but not used in this IN capability set.

– name:

The **name** is the distinguished name of the entry or an alias name of the entry. It is present only if an alias has been dereferenced, RDNs have been resolved to primary RDNs, or context selection has been applied and the name to be returned differs from the **baseObject** name supplied in the operation argument.

– entries:

The **entries** parameter conveys the requested information from each entry (zero or more) which satisfied the filter. The names supplied as part of **entries** may be affected by contexts as described for **Name**. This parameter specifies a list of (1 to n) entry. For a definition of the entry, refer to the subparameter definition in 10.11 "Information".

– partialOutcomeQualifier:

This parameter shall be present whenever the result is incomplete because of a time limit, size limit, or administrative limit problem, because regions of the DIT were not explored, because some critical extensions were unavailable, because an unknown error was received, because paged results are being returned, or an overspecified filter is to be indicated. It contains information on the reasons why the search operation was not finished and on where the operation was stopped. This parameter is based on X.511 "partialOutcomeQualifier" with some constraints. For the definition of the "partialOutcomeQualifier", refer to ITU-T X.511.

This parameter comprises the following subparameters:

- limitProblem;
- unexplored;
- unknownErrors;
- (• *queryReference*):

  NOTE – The queryReference subparameter is used when paged results were requested and therefore is not needed.

- overspecFilter;

– commonResults:

For a definition of this parameter, refer to 10.4.

10.20 Security

This parameter indicates a security-related information.

10.21 Selection

This parameter indicates what information from a set of entries is requested. This parameter is based on X.511 "EntryInformationSelection" with a constraint. For the definition of the "EntryInformationSelection", refer to ITU-T X.511. This parameter comprises the following subparameters:

– attributes:

This parameter comprises the following alternative subparameters:

- allUserAttributes; or
- select;

The attributes subparameter specifies the attributes that should be returned in a retrieval service. The allUserAttributes option is kept even though it is advised to service specifiers to avoid its use which generates more traffic than needed. Instead the select option which precisely names the requested attributes should be used.

– infoTypes:

  The infoTypes subparameter specifies whether the attribute types and values should be returned or only the types. IN services are mainly interested in the attribute values that are relevant to the processing of the service. Therefore, the "attributeTypesAndValues" shall be always used. This subparameter should be absent because the value is specified as the default.

– extraAttributes;

– contextSelection;

– returnContexts.

10.22 Specific-input

This parameter indicates the additional information which is required on the entry in order to perform the method. The type of this parameter is identified by the ASN.1 description of the method defined in the data schema.

10.23 Specific-output

This parameter contains information returned as a result of the operation executed when the method on the entry is invoked. The type of this parameter is identified by the ASN.1 description of the method defined in the data schema.

10.24 Subset

This parameter indicates whether a search is to be applied to:

a) the **baseObject** only;

b) the immediate subordinates of the base object only (**oneLevel**);

c) the base object and all its subordinates (**wholeSubtree**).

For a definition of this parameter, refer to ITU-T X.511.

10.25 TargetSystem

This parameter indicates the SDF to hold the new entry. If this parameter is absent, it shall be taken to mean the same SDF as holds the superior of the new object. If this parameter is present, it shall be the SDF with the specified **AccessPoint**. The parameter shall be absent when subentries are to be added.

For a detailed definition of this parameter, refer to ITU-T X.511.

10.26 TfcCriteria

This parameter identifies the criteria for a service to be subject to traffic flow control. It comprises the following subparameter. Note that the choice of "scf" is not used in SDF related interface.

– sdf:

  This subparameter indicates that the request for traffic flow control is originating from an SDF. It comprises the following subparameter:

  - object:

    This subparameter indicates that the traffic control will be applied when the "object" of an SDF bind request attempt matches those specified in "tfcCriteria".

10.27 UncorrelatedSearchInfo

When the SCF has requested a protection request of signed, this parameter may comprise a list of result parameters originating from and signed by different components of the Directory. If no SDF in the chain can correlate all the results, the SCF must assemble the actual result from the various pieces.

For a definition of this parameter, refer to ITU-T X.511.

10.28 Versions

This parameter indicates the versions of the service which the SCF is prepared to participate in.

For a definition of this parameter, refer to ITU-T X.511.

10.29 WasChained

This is a boolean parameter to indicate if the traffic source node was a relay point used by distributed operations processing.

11 Errors

This clause defines the error procedures for the SCF-SDF interface. Error descriptions are provided in ITU-T Q.1248.1 and the following subclauses provide operation-related error procedures and when relevant, error procedures related to error conditions which are not directly related to the failure of an operation.

The error precedence rule defined in ITU-T X.511 should be applied.

The abandoned and abandonFailed errors are not considered because not supported by this IN capability set (assumption 1 and assumption 3 in clause 4).

11.1 Operation-related error procedures

The following subclauses define the generic error handling for the operation-related error procedures on the SCF-SDF interface. The errors are defined as operation errors in the ASN.1 operations-related description. The TCAP services which are used for reporting operations errors are described in ITU-T Q.1248.1.

Table 2 provides the list of operations which may return each of the errors used on the SCF-SDF interface.

Table 2/Q.1248.4 – Operation-related errors

Errors	AdE	Ex	ME	RE	Se	DB	TB
AttributeError	X	X	X		X		
ExecutionError		X					
IN-DirectoryBindError						X	
IN-ServiceError	X	X	X	X	X		
NameError	X	X	X	X	X		
Referral	X	X	X	X	X		
SecurityError	X	X	X	X	X		
TFCBindError							X
UpdateError	X	X	X	X			

AdE IN-AddEntry
Ex IN-Execute
ME IN-ModifyEntry
RE IN-RemoveEntry
Se IN-Search
DB IN-DirectoryBind
TB TFCBind

11.1.1 AttributeError

The "AttributeError" error procedure uses the AttributeError error defined in ITU-T Q.1248.1.

11.1.1.1 Operations SCF to SDF

This clause describes the procedure when the error for an operation invoked from the SCF occurs in the SDF. The relevant operations are described in Table 1.

11.1.1.1.1 Procedures at invoking entity (SCF)

a) *Sending Operation*

Precondition: refer to the relevant "Operation procedures SCF – precondition" clause.

Postcondition: refer to the relevant "Operation procedures SCF – postcondition" clause.

b) *Receiving Error*

Precondition: refer to the relevant "Operation procedures SCF – postcondition" clause.

Postcondition: remain in the same state.

Error Procedure is dependent on the Service Logic. If the SCF is able to change the request, it can do another SDF query, otherwise the service processing should be terminated.

11.1.1.1.2 Procedures at responding entity (SDF)

Precondition: refer to the relevant "Operation procedures SDF – precondition" clause.

Postcondition: remain in the same state.

The SDF could not perform the operation due to an attribute problem and therefore sends an Attribute error to the SCF. After returning the error, no further error treatment is performed.

11.1.2 ExecutionError

The "ExecutionError" error uses the ExecutionError error defined in ITU-T Q.1248.1.

11.1.2.1 Operations SCF to SDF

This clause describes the procedure when the error for an operation invoked from the SCF occurs in the SDF. The relevant operations are described in Table 1.

11.1.2.1.1 Procedures at invoking entity (SCF)

a) *Sending Operation*

Precondition: refer to the relevant "Operation procedures SCF – precondition" clause.

Postcondition: refer to the relevant "Operation procedures SCF – postcondition" clause.

b) *Receiving Error*

Precondition: refer to the relevant "Operation procedures SCF – postcondition" clause.

Postcondition: remain in the same state.

Error Procedure is independent of the Service Logic. The service processing should be terminated.

11.1.2.1.2 Procedures at responding entity (SDF)

Precondition: refer to the relevant "Operation procedures SDF – precondition" clause.

Postcondition: remain in the same state.

In the case that the SDF has not received the correct input value, the SDF sends an ExecutionError to the SCF with problem set to missingInputValues. In all other cases the SDF attempts to execute the entry method but fails due to the failure of an internal operation or data access logic. The SDF therefore sends an ExecutionError to the SCF with problem set to executionFailure. After returning the error, the SDF is returned to the state it was prior to the operation, that is the IN-Execute operation is considered to be an atomic operation.

11.1.3 IN-DirectoryBindError

The "IN-DirectoryBindError" error procedure is described as the DirectoryBindError in ITU-T Q.1248.1. For a definition of the error, refer to clause 12.

11.1.3.1 Operations SCF to SDF

This clause describes the procedure when the error for an operation invoked from the SCF occurs in the SDF. The relevant operations are described in Table 1.

11.1.3.1.1 Procedures at invoking entity (SCF)

a) *Sending Operation*

Precondition: refer to the relevant "Operation procedures SCF – precondition" clause.

Postcondition: refer to the relevant "Operation procedures SCF – postcondition" clause.

b) *Receiving Error*

Precondition: refer to the relevant "Operation procedures SCF – postcondition" clause.

Postcondition: SCSM-SDF state 1: Idle.

Error Procedure is independent of the Service Logic. The service processing should be terminated.

11.1.3.1.2 Procedures at responding entity (SDF)

Precondition: refer to the relevant "Operation procedures SDF – precondition" clause.

Postcondition: SDSM-SCF state 1: Idle.

The SDF could not perform the operation and therefore sends an IN-DirectoryBindError to the SCF. After returning the error, no further error treatment is performed.

11.1.4 IN-ServiceError

The "IN-ServiceError" error procedure is described as the ServiceError in ITU-T Q.1248.1. For a definition of the error, refer to clause 12.

11.1.4.1 Operations SCF to SDF

This clause describes the procedure when the error for an operation invoked from the SCF occurs in the SDF. The relevant operations are described in Table 1.

11.1.4.1.1 Procedures at invoking entity (SCF)

a) *Sending Operation*

Precondition: refer to the relevant "Operation procedures SCF – precondition" clause.

Postcondition: refer to the relevant "Operation procedures SCF – postcondition" clause.

b) *Receiving Error*

Precondition: refer to the relevant "Operation procedures SCF – postcondition" clause.

Postcondition: remain in the same state.

Error Procedure is dependent on the Service Logic. If there is an alternative SDF it may be possible to do another query, otherwise the service processing should be terminated.

11.1.4.1.2 Procedures at responding entity (SDF)

Precondition: refer to the relevant "Operation procedures SDF – precondition" clause.

Postcondition: remain in the same state.

The SDF could not perform the operation due to a service related problem and sends an IN-Service Error to the SCF. After returning the error, no further error treatment is performed.

11.1.5 NameError

The "NameError" error procedure uses the NameError error defined in ITU-T Q.1248.1.

11.1.5.1 Operations SCF to SDF

This clause describes the procedure when the error for an operation invoked from the SCF occurs in the SDF. The relevant operations are described in Table 1.

11.1.5.1.1 Procedures at invoking entity (SCF)

a) *Sending Operation*

Precondition: refer to the relevant "Operation procedures SCF – precondition" clause.

Postcondition: refer to the relevant "Operation procedures SCF – postcondition" clause.

b) *Receiving Error*

Precondition: refer to the relevant "Operation procedures SCF – postcondition" clause.

Postcondition: remain in the same state.

Error Procedure is dependent on the Service Logic. If the SCF is able to change the request, it can do another SDF query, otherwise the service processing should be terminated.

11.1.5.1.2 Procedures at responding entity (SDF)

Precondition: refer to the relevant "Operation procedures SDF – precondition" clause.

Postcondition: remain in the same state.

The SDF could not perform the operation due to a name problem and therefore sends a Name error to the SCF. After returning the error, no further error treatment is performed.

11.1.6 Referral

The "Referral" error procedure uses the Referral error defined in ITU-T Q.1248.1.

11.1.6.1 Operations SCF to SDF

This clause describes the procedure when the error for an operation invoked from the SCF occurs in the SDF. The relevant operations are described in Table 1.

11.1.6.1.1 Procedures at invoking entity (SCF)

a) *Sending Operation*

Precondition: refer to the relevant "Operation procedures SCF – precondition" clause.

Postcondition: refer to the relevant "Operation procedures SCF – postcondition" clause.

b) *Receiving Error*

Precondition: refer to the relevant "Operation procedures SCF – postcondition" clause.

Postcondition: remain in the same state.

After receiving a referral error, the SCF can continue to execute the service logic by accessing another SDF which is indicated by this error.

11.1.6.1.2 Procedures at responding entity (SDF)

Precondition: refer to the relevant "Operation procedures SDF – precondition" clause.

Postcondition: remain in the same state.

The SDF could not perform the operation due to a data location and therefore sends a Referral error to the SCF. After returning the error, no further error treatment is performed.

11.1.7 SecurityError

The "SecurityError" error procedure uses the SecurityError error defined in ITU-T Q.1248.1.

11.1.7.1 Operations SCF to SDF

This clause describes the procedure when the error for an operation invoked from the SCF occurs in the SDF. The relevant operations are described in Table 1.

11.1.7.1.1 Procedures at invoking entity (SCF)

a) *Sending Operation*

Precondition: refer to the relevant "Operation procedures SCF – precondition" clause.

Postcondition: refer to the relevant "Operation procedures SCF – postcondition" clause.

b) *Receiving Error*

Precondition: refer to the relevant "Operation procedures SCF – postcondition" clause.

Postcondition: remain in the same state.

Error Procedure is independent of the Service Logic. The service processing should be terminated.

11.1.7.1.2 Procedures at responding entity (SDF)

Precondition: refer to the relevant "Operation procedures SDF – precondition" clause.

Postcondition: remain in the same state.

The SDF could not perform the operation for security reasons and therefore sends a Security error to the SCF. After returning the error, no further error treatment is performed.

11.1.8 TFCBindError

The "TFCBindError" error procedure uses the TFCBindError error defined in ITU-T Q.1248.1.

11.1.8.1 Operations SDF to SCF

This clause describes the procedure when the error for an operation invoked from the SDF occurs in the SCF. The relevant operations are described in Table 1.

11.1.8.1.1 Procedures at invoking entity (SDF)

a) *Sending Operation*

Precondition: refer to the relevant "Operation procedures SDF – precondition" clause.

Postcondition: refer to the relevant "Operation procedures SDF – postcondition" clause.

b) *Receiving Error*

Precondition: refer to the relevant "Operation procedures SDF – postcondition" clause.

Postcondition: SDME FSM state 1: Idle.

The traffic flow control association should be terminated.

11.1.8.1.2 Procedures at responding entity (SCF)

Precondition: refer to the relevant "Operation procedures SCF – precondition" clause.

Postcondition: SCME FSM state 1: Idle.

The SCF could not perform the operation and therefore sends a TFCBindError to the SDF. After returning the error, no further error treatment is performed.

11.1.9 UpdateError

The "UpdateError" error procedure uses the UpdateError error defined in ITU-T Q.1248.1.

11.1.9.1 Operations SCF to SDF

This clause describes the procedure when the error for an operation invoked from the SCF occurs in the SDF. The relevant operations are described in Table 1.

11.1.9.1.1 Procedures at invoking entity (SCF)

a) *Sending Operation*

Precondition: refer to the relevant "Operation procedures SCF – precondition" clause.

Postcondition: refer to the relevant "Operation procedures SCF – postcondition" clause.

b) *Receiving Error*

Precondition: refer to the relevant "Operation procedures SCF – postcondition" clause.

Postcondition: remain in the same state.

Error Procedure is dependent on the Service Logic.

11.1.9.1.2 Procedures at responding entity (SDF)

Precondition: refer to the relevant "Operation procedures SDF – precondition" clause.

Postcondition: remain in the same state.

The SDF detects a problem related to the addition, deletion or modification of information and sends an UpdateError error to the SCF. After returning the error, no further error treatment is performed.

12 Service assumed from TCAP

All the services provided by the Directory ASEs are contained in a single AE. The Component Handler (CHA) of the Transaction Capabilities (TC) supports the request/reply paradigm of the operation. The Directory ASEs provide the mapping function of the abstract-syntax notation of the directory operation packages onto the services provided by TC. The Dialogue handler (DHA) of the TC supports the establishment and release of an application-association called "dialogue" between a pair of AEs. Dialogues between an SCF and an SDF are initiated only by the SCF.

All the services provided by the Traffic Flow Control (TFC) ASEs are contained in a single AE. The Component Handler (CHA) of the Transaction Capabilities (TC) supports the request/reply paradigm of the operation. The TFC ASEs provide the mapping function of the abstract-syntax notation of the TFC related operation packages onto the services provided by TC. The Dialogue handler (DHA) of the TC supports the establishment and release of an application-association called "dialogue" between a pair of AEs. Dialogues between an SDF and an SCF are initiated only by the SDF.

12.1 Normal procedures

12.1.1 SCSM/SDSM-related messages

Dialogue handling services are used to support the IN-DirectoryBind and IN-Unbind operations and to trigger the sending of the APDUs associated with the operations involved in the Directory packages.

Component grouping is performed under the control of the application-process through an appropriate usage of the TC-BEGIN and TC-CONTINUE service.

The TC-END service is solely used to support the unbind procedure (i.e. it is never used to trigger the sending of components).

On receipt of an empty TC-CONTINUE.req primitive, the SDF should ignore the primitive.

On receipt of a TC-END.req with a database request, the SDF should not perform the database request and consider the requested TC-END service as an unbind procedure. The dialogue is then terminated.

12.1.2 SCME/SDME-related messages

Dialogue handling services are used to support the TFCBind and IN-Unbind operations and to trigger the sending of the APDUs associated with the operations involved in the TFC package.

Component grouping is performed under the control of the application-process through an appropriate usage of the TC-BEGIN and TC-CONTINUE service.

The TC-END service is solely used to support the unbind procedure (i.e. it is never used to trigger the sending of components).

On receipt of an empty TC-CONTINUE.req primitive, the SDF should ignore the primitive.

On receipt of a TC-END.req with a TFC request, the SCF should perform the request and consider the requested TC-END service as an unbind procedure. The dialogue is then terminated.

12.2 Abnormal procedures

On receipt of a TC-U-REJECT.ind in the SDF, this primitive should be ignored.

On receipt of a TC-R-REJECT.ind in the SDF, the dialogue should be released with a TC-U-ABORT.req.

If reject situations are detected in the SDF, a TC-U-REJECT.req should be sent followed by a TC-CONTINUE.req.

The pre-arranged termination procedure is never used.

The application-process is the sole user of the TC-P-ABORT service and TC-NOTICE service.

The receipt of a TC-U-ABORT.ind or TC-P-ABORT.ind on a dialogue terminates all request processing. It is the application-process responsibility to confirm if requested modifications occurred.

It is an application-process responsibility to provide in the TC-BEGIN.req primitive a destination address which can be used by the underlying SCCP to route the message to the proper SDF.

12.3 Dialogue handling

Common procedures are described in ITU-T Q.1248.1.

12.3.1 Dialogue establishment

Common procedures are described in ITU-T Q.1248.1.

12.3.2 Dialogue continuation

Common procedures are described in ITU-T Q.1248.1.

12.3.3 Dialogue termination

Common procedures are described in ITU-T Q.1248.1.

12.3.4 User abort

Common procedures are described in ITU-T Q.1248.1.

12.3.5 Provider abort

Common procedures are described in ITU-T Q.1284.1.

12.3.6 Mapping to TC dialogue primitives

12.3.6.1 SCSM/SDSM-related messages

This clause defines the mapping of the IN-DirectoryBind and IN-Unbind services onto the services of the TC dialogue handling services defined in ITU-T Q.771.

Bind

The IN-DirectoryBind service is mapped onto TC-services as follows:

a) The TC-BEGIN service is used to invoke the **IN-DirectoryBind** operation and SETransfer operations.

b) The TC-CONTINUE service is used to report the success of the **IN-DirectoryBind** operation. If the SESE is included in the application context, the TC-CONTINUE service is used for the second and third exchanges.

c) The TC-U-ABORT service is used to report the failure of the **IN-DirectoryBind** operation and SETransfer operations.

The use of the parameter of these services is qualified in ITU-T Q.1248.1.

The use of parameters of the TC-BEGIN service is defined in ITU-T Q.1248.1 with the following qualifications:

– The Application Context Name parameter of the TC-BEGIN service shall take the value of the application-context-name field of the **iNdirectoryAccessAC, iNdirectoryAccessWith3seAC, inExtendedDirectoryAccessAC,** or **inExtendedDirectoryAccessWith3seAC** object.

– The Dialogue ID parameter shall be used as specified in ITU-T Q.1248.1.

– If the **DirectoryBindArgument** of the **in-DirectoryBind** operation is present, the User Information parameter of the TC-BEGIN service shall contain an EXTERNAL ASN.1 Type with the direct-reference field set to **id-as-indirectoryBindingAS** and the value to be encoded shall be of type **DAPBinding-PDUs.bind.bind-invoke** (**DirectoryBindArgument**).

  If the SESE is included in the application context, the User Information parameter of the TC-BEGIN service shall contain a value of type **seItem**.

– If the **DirectoryBindArgument** of the **in-DirectoryBind** operation is omitted, the User Information parameter of the TC-BEGIN service shall not contain an EXTERNAL ASN.1 Type with the direct-reference field set to **id-as-indirectoryBindingAS**.

  If the SESE is included in the application context, the User Information parameter of the TC-BEGIN service shall contain a value of type **seItem**.

The use of parameters of the TC-CONTINUE service is defined in ITU-T Q.1248.1 with the following qualifications:

– The Dialogue ID parameter shall be used as specified in ITU-T Q.1248.1.

– If the **DirectoryBindResult** of the **in-DirectoryBind** operation is present, the User Information parameter of the TC-CONTINUE service shall contain an EXTERNAL ASN.1 Type with the direct-reference field set to **id-as-indirectoryBindingAS** and the value to be encoded shall be of type **DAPBinding-PDUs.bind.bind-result (DirectoryBindResult)**.

  If the SESE is included in the application context, the User Information parameter of the TC-CONTINUE service shall contain a value of type **seItem**.

– If the **DirectoryBindResult** of the **in-DirectoryBind** operation is omitted, the User Information parameter of the TC-CONTINUE service shall not contain an EXTERNAL ASN.1 Type with the direct-reference field set to **id-as-indirectoryBindingAS**.

  If the SESE is included in the application context, the User Information parameter of the TC-CONTINUE service shall contain a value of type **seItem**.

The use of parameters of the TC-U-ABORT service is defined in ITU-T Q.1248.1 with the following qualifications:

– The Dialogue ID parameter shall be used as specified in ITU-T Q.1248.1.

– The Application Context Name parameter of the TC-U-ABORT service shall be set as defined in ITU-T Q.1248.1 unless the TC-U-ABORT is reporting the failure of the **IN-DirectoryBind** operation. In this case it may be set to the value of the application-context-name field of the **iNdirectoryAccessAC, iNdirectoryAccessWith3seAC, inExtendedDirectoryAccessAC,** or **inExtendedDirectoryAccessWith3seAC** object.

– The User Information parameter of the TC-U-ABORT service shall be used as specified in ITU-T Q.1248.1, unless the TC-U-ABORT is reporting the failure of the **IN-DirectoryBind** operation. In this case this parameter shall contain an EXTERNAL ASN.1 Type with the direct-reference field, if present, set to **id-as-indirectoryBindingAS** and the value to be encoded shall be of type **DAPBinding-PDUs.bind.bind-error** (**IN-DirectoryBindError**).

Unbind

The **IN-Unbind** service is mapped onto the TC-END service.

The use of the parameters of the TC-END service is defined in ITU-T Q.1248.1 with the following qualifications:

– The Dialogue ID parameter shall be used as specified in ITU-T Q.1248.1.

12.3.6.2 SCME/SDME-related messages

This clause defines the mapping of the TFCBind and IN-Unbind services onto the services of the TC dialogue handling services defined in ITU-T Q.771.

Bind

The TFCBind service is mapped onto TC-services as follows:

a) The TC-BEGIN service is used to invoke the **TFCBind** operation.

b) The TC-CONTINUE service is used to report the success of the **TFCBind** operation.

c) The TC-U-ABORT service is used to report the failure of the **TFCBind** operation.

The use of the parameter of these services is qualified in ITU-T Q.1248.1.

The use of parameters of the TC-BEGIN service is defined in ITU-T Q.1248.1 with the following qualifications:

– The Application Context Name parameter of the TC-BEGIN service shall take the value of the application-context-name field of the **trafficFlowControlAC** object.

– The Dialogue ID parameter shall be used as specified in ITU-T Q.1248.1.

– If the **TFCBindArgument** of the **tfcBind** operation is present, the User Information parameter of the TC-BEGIN service shall contain an EXTERNAL ASN.1 Type with the direct-reference field set to **id-as-tfcBindingAS** and the value to be encoded shall be of type **TFCBinding-PDUs.bind.bind-invoke** (**TFCBindArgument**).

– If the **TFCBindArgument** of the **tfcBind** operation is omitted, the User Information parameter of the TC-BEGIN service shall not contain an EXTERNAL ASN.1 Type with the direct-reference field set to **id-as-tfcBindingAS**.

The use of parameters of the TC-CONTINUE service is defined in ITU-T Q.1248.1 with the following qualifications:

– The Dialogue ID parameter shall be used as specified in ITU-T Q.1248.1.

– If the **TFCBindResult** of the **tfcBind** operation is present, the User Information parameter of the TC-CONTINUE service shall contain an EXTERNAL ASN.1 Type with the direct-reference field set to **id-as-tfcBindingAS** and the value to be encoded shall be of type **TFCBinding-PDUs.bind.bind-result (TFCBindResult)**.

– If the **TFCBindResult** of the **tfcBind** operation is omitted, the User Information parameter of the TC-CONTINUE service shall not contain an EXTERNAL ASN.1 Type with the direct-reference field set to **id-as-tfcBindingAS**.

The use of parameters of the TC-U-ABORT service is defined in ITU-T Q.1248.1 with the following qualifications:

– The Dialogue ID parameter shall be used as specified in ITU-T Q.1248.1.

– The Application Context Name parameter of the TC-U-ABORT service shall be set as defined in ITU-T Q.1248.1 unless the TC-U-ABORT is reporting the failure of the TFCBind operation.

– The User Information parameter of the TC-U-ABORT service shall be used as specified in ITU-T Q.1248.1, unless the TC-U-ABORT is reporting the failure of the TFCBind operation. In this case this parameter shall contain an EXTERNAL ASN.1 Type with the direct-reference field, if present, set to **id-as-tfcBindingAS** and the value to be encoded shall be of type **TFCBinding-PDUs.bind.bind-error** (**TFCBindError**).

Unbind

The **IN-Unbind** service is mapped onto the TC-END service.

The use of the parameters of the TC-END service is defined in ITU-T Q.1248.1 with the following qualifications:

– The Dialogue ID parameter shall be used as specified in ITU-T Q.1248.1.

12.4 Component handling

12.4.1 Procedures for INAP operations

Common procedures are described in ITU-T Q.1248.1.

12.4.2 Mapping to TC component parameters

The mapping of TC component parameters are described in ITU-T Q.1248.1.

ANNEX A

ASN.1 definitions

Annex A is available as an electronic attachment.

SERIES OF ITU-T RECOMMENDATIONS

Series A	Organization of the work of ITU-T
Series B	Means of expression: definitions, symbols, classification
Series C	General telecommunication statistics
Series D	General tariff principles
Series E	Overall network operation, telephone service, service operation and human factors
Series F	Non-telephone telecommunication services
Series G	Transmission systems and media, digital systems and networks
Series H	Audiovisual and multimedia systems
Series I	Integrated services digital network
Series J	Cable networks and transmission of television, sound programme and other multimedia signals
Series K	Protection against interference
Series L	Construction, installation and protection of cables and other elements of outside plant
Series M	TMN and network maintenance: international transmission systems, telephone circuits, telegraphy, facsimile and leased circuits
Series N	Maintenance: international sound programme and television transmission circuits
Series O	Specifications of measuring equipment
Series P	Telephone transmission quality, telephone installations, local line networks
Series Q	**Switching and signalling**
Series R	Telegraph transmission
Series S	Telegraph services terminal equipment
Series T	Terminals for telematic services
Series U	Telegraph switching
Series V	Data communication over the telephone network
Series X	Data networks and open system communications
Series Y	Global information infrastructure and Internet protocol aspects
Series Z	Languages and general software aspects for telecommunication systems

Printed in Switzerland
Geneva, 2002