Final draft ETSI ES 201 915-7 V1.2.1 (2002-05)

ETSI Standard

Open Service Access (OSA);
Application Programming Interface (API);
Part 7: Terminal Capabilities SCF

Reference

RES/SPAN-120076-7

Keywords

API, OSA, IDL, UML

ETSI

650 Route des Lucioles
F-06921 Sophia Antipolis Cedex - FRANCE

Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16

Siret N° 348 623 562 00017 - NAF 742 C
Association à but non lucratif enregistrée à la
Sous-Préfecture de Grasse (06) N° 7803/88

Important notice

Individual copies of the present document can be downloaded from:
http://www.etsi.org

The present document may be made available in more than one electronic version or in print. In any case of existing or perceived difference in contents between such versions, the reference version is the Portable Document Format (PDF). In case of dispute, the reference shall be the printing on ETSI printers of the PDF version kept on a specific network drive within ETSI Secretariat.

Users of the present document should be aware that the document may be subject to revision or change of status. Information on the current status of this and other ETSI documents is available at
http://portal.etsi.org/tb/status/status.asp

If you find errors in the present document, send your comment to:
editor@etsi.fr

Copyright Notification

No part may be reproduced except as authorized by written permission.
The copyright and the foregoing restriction extend to reproduction in all media.

© European Telecommunications Standards Institute 2002.
© The Parlay Group 2002.
All rights reserved.

DECTTM, **PLUGTESTS**TM and **UMTS**TM are Trade Marks of ETSI registered for the benefit of its Members.
TIPHONTM and the **TIPHON logo** are Trade Marks currently being registered by ETSI for the benefit of its Members.
3GPPTM is a Trade Mark of ETSI registered for the benefit of its Members and of the 3GPP Organizational Partners.

Contents

Intellectual Property Rights

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

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

Foreword

This ETSI Standard (ES) has been produced by ETSI Technical Committee Services and Protocols for Advanced Networks (SPAN), and is now submitted for the ETSI standards Membership Approval Procedure.

The present document is part 7 of a multi-part deliverable covering Open Service Access (OSA); Application Programming Interface (API), as identified below. The API specification (ES 201 915) is structured in the following parts:

Part 1:	"Overview";
Part 2:	"Common Data Definitions";
Part 3:	"Framework";
Part 4:	"Call Control SCF";
Part 5:	"User Interaction SCF";
Part 6:	"Mobility SCF";
Part 7:	**"Terminal Capabilities SCF";**
Part 8:	"Data Session Control SCF";
Part 9:	"Generic Messaging SCF";
Part 10:	"Connectivity Manager SCF";
Part 11:	"Account Management SCF";
Part 12:	"Charging SCF".

The present document has been defined jointly between ETSI, The Parlay Group [24] of ES 201 915-1 and the 3GPP, in co-operation with a number of JAIN™ Community [25] of ES 201 915-1 member companies.

The present document forms part of the Parlay 3.1 set of specifications.

1 Scope

The present document is part 7 of the Stage 3 specification for an Application Programming Interface (API) for Open Service Access (OSA).

The OSA specifications define an architecture that enables application developers to make use of network functionality through an open standardised interface, i.e. the OSA APIs.

The present document specifies the Terminal Capabilities Service Capability Feature (SCF) aspects of the interface. All aspects of the Terminal Capabilities SCF are defined here, these being:

- Sequence Diagrams
- Class Diagrams
- Interface specification plus detailed method descriptions
- State Transition diagrams
- Data Definitions
- IDL Description of the interfaces

The process by which this task is accomplished is through the use of object modelling techniques described by the Unified Modelling Language (UML).

2 References

The references listed in clause 2 of ES 201 915-1 contain provisions which, through reference in this text, constitute provisions of the present document.

ETSI ES 201 915-1: "Open Service Access (OSA); Application Programming Interface (API); Part 1: Overview".

3 Definitions and abbreviations

3.1 Definitions

For the purposes of the present document, the terms and definitions given in ES 201 915-1 apply.

3.2 Abbreviations

For the purposes of the present document, the abbreviations defined in ES 201 915-1 apply.

4 Terminal Capabilities SCF

The following clauses describe each aspect of the Terminal Capabilities Capability Feature (SCF).

The order is as follows:

- The Sequence diagrams give the reader a practical idea of how each of the SCF is implemented.
- The Class relationships clause show how each of the interfaces applicable to the SCF, relate to one another.
- The Interface specification clause describes in detail each of the interfaces shown within the Class diagram part.

- The State Transition Diagrams (STD) show the transition between states in the SCF. The states and transitions are well-defined; either methods specified in the Interface specification or events occurring in the underlying networks cause state transitions.
- The Data Definitions clause show a detailed expansion of each of the data types associated with the methods within the classes. Note that some data types are used in other methods and classes and are therefore defined within the Common Data types part of this specification.

5 Sequence Diagrams

There are no Sequence Diagrams for the Terminal Capabilities SCF.

6 Class Diagrams

Terminal Capabilities Class Diagram:

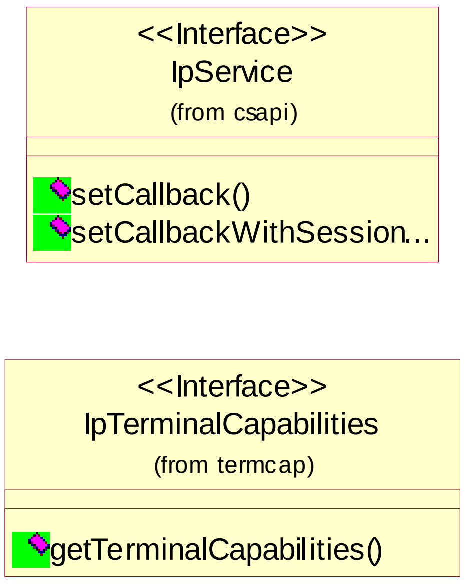


Figure 1: Package Overview

7 The Service Interface Specifications

7.1 Interface Specification Format

This clause defines the interfaces, methods and parameters that form a part of the API specification. The Unified Modelling Language (UML) is used to specify the interface classes. The general format of an interface specification is described below.

7.1.1 Interface Class

This shows a UML interface class description of the methods supported by that interface, and the relevant parameters and types. The Service and Framework interfaces for enterprise-based client applications are denoted by classes with name `Ip<name>`. The callback interfaces to the applications are denoted by classes with name `IpApp<name>`. For the interfaces between a Service and the Framework, the Service interfaces are typically denoted by classes with name IpSvc<name>, while the Framework interfaces are denoted by classes with name IpFw<name>

7.1.2 Method descriptions

Each method (API method "call") is described. Both synchronous and asynchronous methods are used in the API. Asynchronous methods are identified by a '`Req`' suffix for a method request, and, if applicable, are served by asynchronous methods identified by either a '`Res`' or '`Err`' suffix for method results and errors, respectively. To handle responses and reports, the application or service developer must implement the relevant `IpApp<name> or IpSvc<name>` interfaces to provide the callback mechanism.

7.1.3 Parameter descriptions

Each method parameter and its possible values are described. Parameters described as 'in' represent those that must have a value when the method is called. Those described as 'out' are those that contain the return result of the method when the method returns.

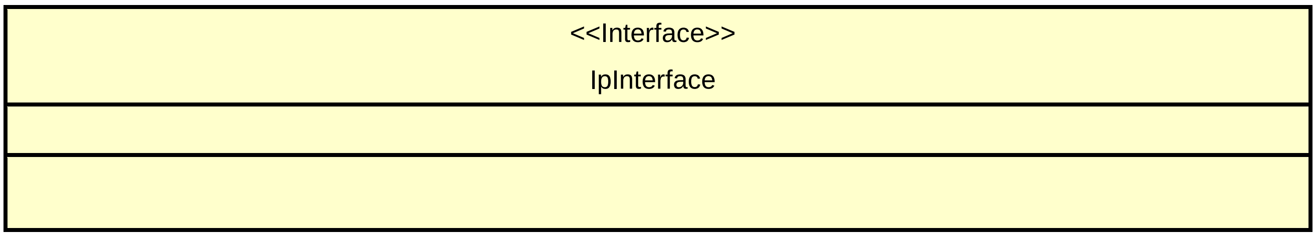
7.1.4 State Model

If relevant, a state model is shown to illustrate the states of the objects that implement the described interface.

7.2 Base Interface

7.2.1 Interface Class IpInterface

All application, framework and service interfaces inherit from the following interface. This API Base Interface does not provide any additional methods.



7.3 Service Interfaces

7.3.1 Overview

The Service Interfaces provide the interfaces into the capabilities of the underlying network - such as call control, user interaction, messaging, mobility and connectivity management.

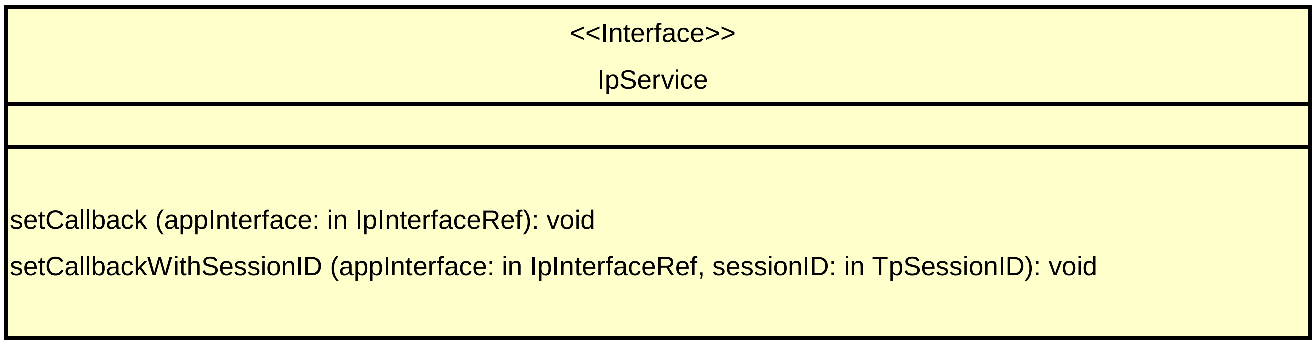
The interfaces that are implemented by the services are denoted as 'Service Interface'. The corresponding interfaces that must be implemented by the application (e.g. for API callbacks) are denoted as 'Application Interface'.

7.4 Generic Service Interface

7.4.1 Interface Class IpService

Inherits from: IpInterface

All service interfaces inherit from the following interface.



setCallback (appInterface: in IpInterfaceRef): void

setCallbackWithSessionID (appInterface: in IpInterfaceRef, sessionID: in TpSessionID): void

Method
`setCallback()`

This method specifies the reference address of the callback interface that a service uses to invoke methods on the application. It is not allowed to invoke this method on an interface that uses SessionIDs.

Parameters

`appInterface: in IpInterfaceRef`

Specifies a reference to the application interface, which is used for callbacks

Raises

`TpCommonExceptions, P_INVALID_INTERFACE_TYPE`

Method
`setCallbackWithSessionID()`

This method specifies the reference address of the application's callback interface that a service uses for interactions associated with a specific session ID: e.g. a specific call, or call leg. It is not allowed to invoke this method on an interface that does not use SessionIDs.

Parameters

`appInterface: in IpInterfaceRef`

Specifies a reference to the application interface, which is used for callbacks

`sessionID: in TpSessionID`

Specifies the session for which the service can invoke the application's callback interface.

Raises

`TpCommonExceptions, P_INVALID_SESSION_ID, P_INVALID_INTERFACE_TYPE`

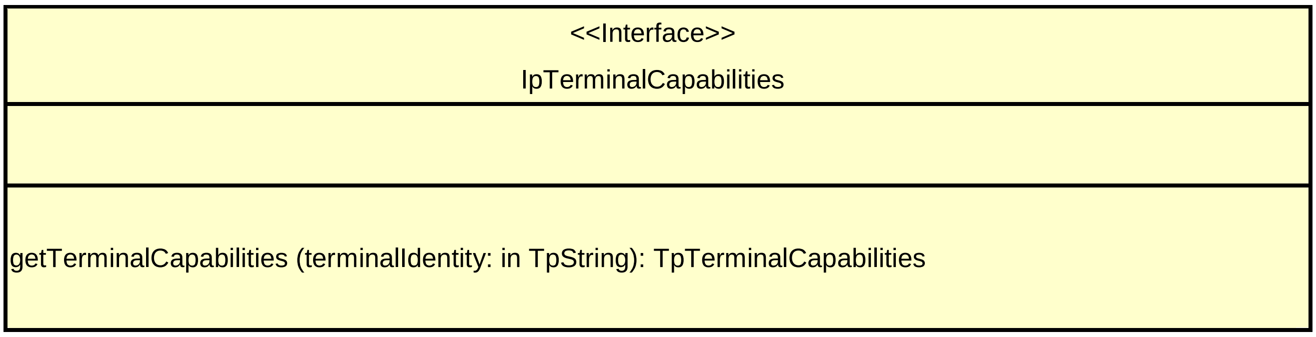
8 Terminal Capabilities Interface Classes

The Terminal Capabilities SCF enables the application to retrieve the terminal capabilities of the specified terminal. The Terminal Capabilities service provides a SCF interface that is called IpTerminalCapabilities. There is no need for an application interface, since IpTerminalCapabilities only contains the synchronous method getTerminalCapabilities.

8.1 Interface Class IpTerminalCapabilities

Inherits from: IpInterface.

The Terminal Capabilities SCF interface IpTerminalCapabilities contains the synchronous method getTerminalCapabilities. The application has to provide the terminaIdentity as input to this method. The result indicates whether or not the terminal capabilities are available in the network and, in case they are, it will return the terminal capabilities (see the data definition of TpTerminalCapabilities for more information).


<<Interface>>
IpTerminalCapabilities
getTerminalCapabilities (terminalIdentity: in TpString): TpTerminalCapabilities

Method

`getTerminalCapabilities()`

This method is used by an application to get the capabilities of a user's terminal. Direction: Application to Network.

Returns result: Specifies the latest available capabilities of the user's terminal.

This information, if available, is returned as CC/PP headers as specified in W3C and adopted in the WAP UAProf specifications (see references in Part 1 of this specification). It contains URLs; terminal attributes and values, in RDF format; or a combination of both.

Parameters

`terminalIdentity: in TpString`

Identifies the terminal. It may be a logical address known by the WAP Gateway/PushProxy.

Returns

`TpTerminalCapabilities`

Raises

`TpCommonExceptions, P_INVALID_TERMINAL_ID`

9 State Transition Diagrams

There are no State Transition Diagrams for the Terminal Capabilities SCF.

10 Terminal Capabilities Data Definitions

The constants and types defined in the following sections are defined in the *org.csapi.termcap* package.

All data types referenced but not defined in this clause are common data definitions which may be found in ES 201 915-2.

10.1 terminalIdentity

Identifies the terminal.

Name	Type	Documentation
`terminalIdentity`	`TpString`	Identifies the terminal. It may be a logical address known by the WAP Gateway/PushProxy.

10.2 TpTerminalCapabilities

This data type is a Sequence of Data Elements that describes the terminal capabilities. It is a structured type that consists of:

Sequence Element Name	Sequence Element Type	Documentation
`StatusCode`	`TpBoolean`	Indicates whether or not the TerminalCapabilities are available.
`TerminalCapabilities`	`TpString`	Specifies the latest available capabilities of the user's terminal. This information, if available, is returned as CC/PP headers as specified in W3C (see [6] in ES 201 915-1)and adopted in the WAP UAProf specification (see [9] in ES 201 915-1). It contains URLs; terminal attributes and values, in RDF format; or a combination of both.

10.3 TpTerminalCapabilitiesError

Defines an error that is reported by the Terminal Capabilities SCF.

Name	Value	Description
`P_TERMCAP_ERROR_UNDEFINED`	0	Undefined.
`P_TERMCAP_INVALID_TERMINALID`	1	The request cannot be handled because the terminal id specified is not valid.
`P_TERMCAP_SYSTEM_FAILURE`	2	System failure. The request cannot be handled because of a general problem in the terminal capabilities service or the underlying network.

11 Exception Classes

The following are the list of exception classes which are used in this interface of the API.

Name	Description
`P_INVALID_TERMINAL_ID`	The request cannot be handled because the terminal id specified is not valid.

Each exception class contains the following structure:

Structure Element Name	Structure Element Type	Structure Element Description
`ExtraInformation`	`TpString`	Carries extra information to help identify the source of the exception, e.g. a parameter name

Annex A (normative): OMG IDL Description of Terminal Capabilities SCF

The OMG IDL representation of this interface specification is contained in a text file (termcap.idl contained in archive es_20191507v010201m0.ZIP) which accompanies the present document.

Annex B (informative): Contents of 3GPP OSA R4 Terminal Capabilities

All of the present document is relevant for TS 129 198-7 V4 (Release 4).

Annex C (informative): Summary of differences between V1.1.1 (Parlay 3.0) and V1.2.1 (Parlay 3.1)

C.1 IpService

setCallback() and setCallbackWithSessionID() now both raise P_INVALID_INTERFACE_TYPE.

History

Document history			
V1.1.1	February 2002	Publication	
V1.2.1	May 2002	Membership Approval Procedure	MV 20020705: 2002-05-07 to 2002-07-05