3GPP TSG CN Plenary Meeting #15 **NP-020113**
6 – 8 March 2002. Jeju, KOREA

Source:	**CN5 (OSA)**
Title:	**Rel-5 CRs 29.198-07 OSA API Part 7: Terminal capabilities**
Agenda item:	**9.4**
Document for:	**Decision**

Doc-1st-Level	Spec	CR	R	Phase	Subject	Cat	Ver-Curr	Ver-New	Doc-2nd-Level	Workitem
NP-020113	29.198-07	005		Rel-5	**Addition of terminal capability change notifications**	B	4.4.0	5.0.0	N5-020161	OSA1

3GPP TSG-CN5 Meeting #16
Hong Kong, CHINA, 4th – 8th February 2002

N5-020161

CR-Form-v6.1

CHANGE REQUEST

⌘	**29.198-07**	**CR**	**005**	⌘ **rev**	**-**	⌘	Current version:	**4.4.0** ⌘

*For **HELP** on using this form, see bottom of this page or look at the pop-up text over the ⌘ symbols.*

Proposed change affects: ⌘ (U)SIM ☐ ME/UE ☐ Radio Access Network ☐ Core Network **X**

Title:	⌘	Addition of terminal capability change notifications		
Source:	⌘	CN5		
Work item code:	⌘	OSA2	***Date:*** ⌘	08/02/2002
Category:	⌘	**B**	***Release:*** ⌘	REL-5
		Use one of the following categories: ***F*** *(correction)* ***A*** *(corresponds to a correction in an earlier release)* ***B*** *(addition of feature),* ***C*** *(functional modification of feature)* ***D*** *(editorial modification)* Detailed explanations of the above categories can be found in 3GPP TR 21.900.		*Use one of the following releases:* *2 (GSM Phase 2)* *R96 (Release 1996)* *R97 (Release 1997)* *R98 (Release 1998)* *R99 (Release 1999)* *REL-4 (Release 4)* *REL-5 (Release 5)*

Reason for change:	⌘	Requirements for terminal capability change notification exist in 22.127 v. 5.2.0 section 12.1. Where it is currently stated: " • The Terminal Capabilities are changed. when the capabilities of a terminal change (e.g. when a keyboard is attached) and this event is armed by an application, that application shall be notified. Note: The ability to support this function is dependent on the ability of a terminal (through e.g. MExE or WAP) to notify changes in its capabilities. Therefore this function will *not* be able to supply event notifications for terminals not supporting notification of their terminal capabilities."
Summary of change:	⌘	A new interface with four new methods is defined. An application starts monitoring the terminal capability changes by invoking a start method and stops it by a stop method call. The SCS reports the changes and errors by respective method invocations on the application side. The terminal can be addressed either by the subscriber or terminal identification. Also the criteria for changes and the scope of the interesting capabilities can be provided by the application. The version number of WAP has been removed from the references section, because limitation to version 1.2 is not appropriate. A sequence diagram has been added. Furthermore the inheritance of IpTerminalCapabilities is changed to IpService instead of IpInterface. Also a new error value has been specified for the case when the terminal capability information is not available.
Consequences if not approved:	⌘	Missing definitions.

Clauses affected:	⌘	5, 6, 8, 10

Other specs affected:	⌘	☐	Other core specifications Test specifications O&M Specifications	⌘
		☐		
		☐		
Other comments:	⌘	SA1 has indicated in the agreed CR (see the CN5 output Liaison statement in N5-011161 or S1-011111) that OSA does not require that all SCFs, to which OSA provides an API interface, need to be 3GPP standardised entities, nor that the existence of a standardised interface / protocol to communicate with that SCF is required. In this case the underlying mechanisms are not fully standardised within 3GPP.		

How to create CRs using this form:
Comprehensive information and tips about how to create CRs can be found at: http://www.3gpp.org/3G_Specs/CRs.htm. Below is a brief summary:

1) Fill out the above form. The symbols above marked ⌘ contain pop-up help information about the field that they are closest to.

2) Obtain the latest version for the release of the specification to which the change is proposed. Use the MS Word "revision marks" feature (also known as "track changes") when making the changes. All 3GPP specifications can be downloaded from the 3GPP server under ftp://ftp.3gpp.org/specs/ For the latest version, look for the directory name with the latest date e.g. 2001-03 contains the specifications resulting from the March 2001 TSG meetings.

3) With "track changes" disabled, paste the entire CR form (use CTRL-A to select it) into the specification just in front of the clause containing the first piece of changed text. Delete those parts of the specification which are not relevant to the change request.

2 References

The following documents contain provisions which, through reference in this text, constitute provisions of the present document.

- References are either specific (identified by date of publication, edition number, version number, etc.) or non-specific.
- For a specific reference, subsequent revisions do not apply.
- For a non-specific reference, the latest version applies. In the case of a reference to a 3GPP document (including a GSM document), a non-specific reference implicitly refers to the latest version of that document *in the same Release as the present document*.

[1] 3GPP TS 29.198-1 "Open Service Access; Application Programming Interface; Part 1: Overview".

[2] 3GPP TS 22.127: "Stage 1 Service Requirement for the Open Service Access (OSA) (Release 4)".

[3] 3GPP TS 23.127: "Virtual Home Environment (Release 4)".

[4] World Wide Web Consortium Composite Capability/Preference Profiles (CC/PP): A user side framework for content negotiation (www.w3.org)

[5] Wireless Application Protocol (WAP), ~~Version 1.2,~~ UAProf Specification (www.wapforum.org)

3 Definitions and abbreviations

3.1 Definitions

For the purposes of the present document, the terms and definitions given in TS 29.198-1 [1] apply.

3.2 Abbreviations

For the purposes of the present document, the abbreviations given in TS 29.198-1 [1] apply.

4 Terminal Capabilities SCF

The following clauses describe each aspect of the Terminal 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 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 section 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.~~ The following example sequence diagram illustrates how the terminal capabilities can be retrieved and their changes monitored.

1: The application retrieves the terminal capability of a terminal.

2: The application creates an object to implement IpAppExtendedTerminalCapabilities.

3: The terminal capabilities changes are started to be monitored.

4: The terminal capabilities have changed and they are reported as requested.

5: The report is forwarded internally to the application.

6: The terminal capabilities have changed and they are reported as requested.

7: The report is forwarded internally to the application.

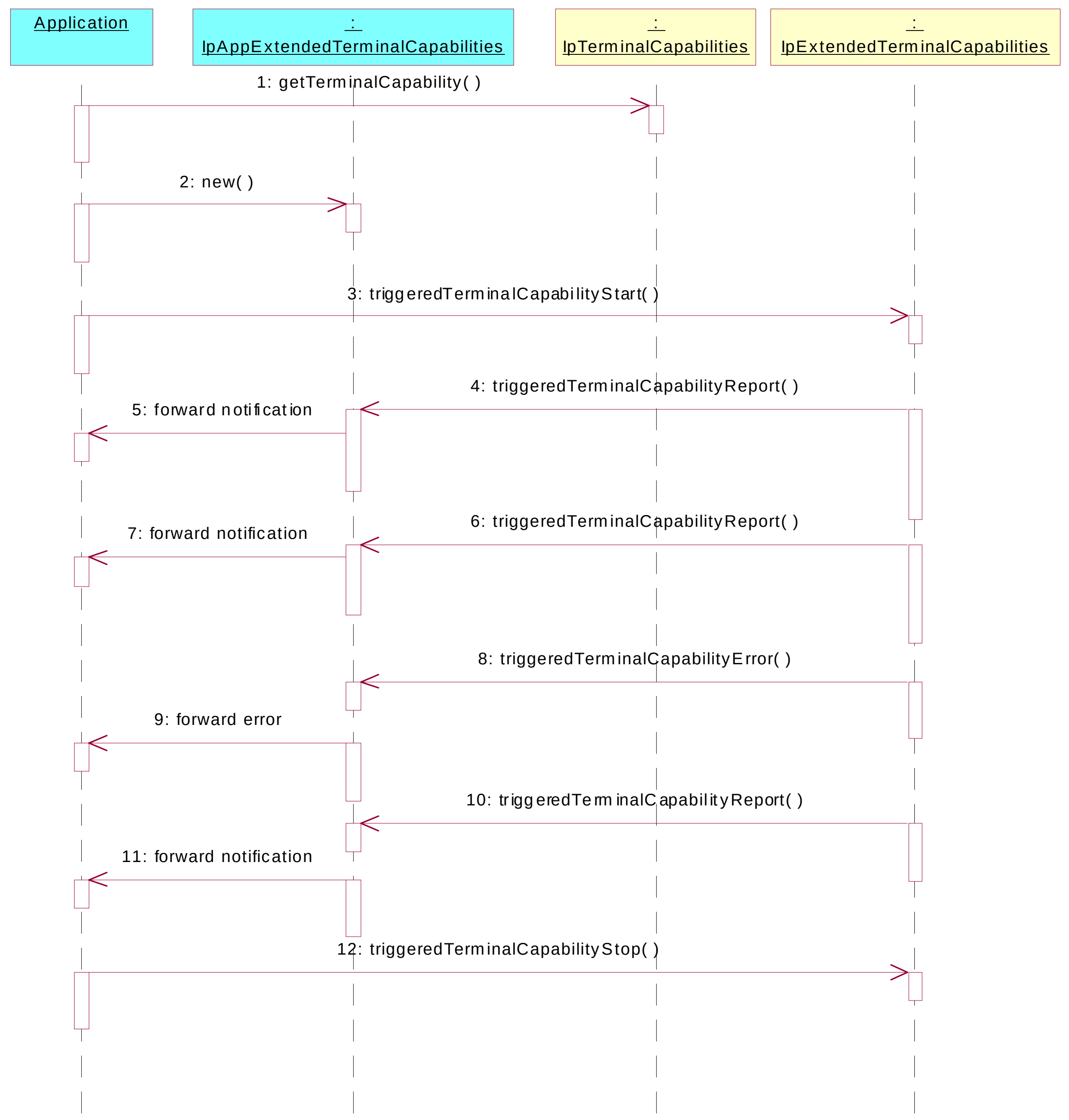
8: An error has happened in the monitoring and it is reported.

9: The error report is forwarded internally to the application.

10: The terminal capabilities have changed and they are reported as requested.

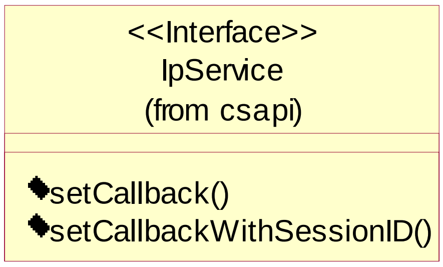
11: The report is forwarded internally to the application.

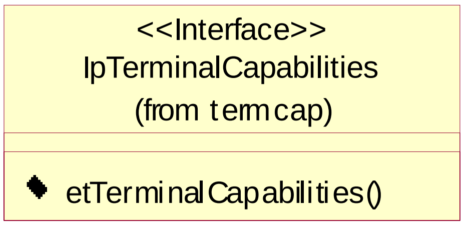
12: The terminal capability monitoring is stopped.


Application
: IpAppExtendedTerminalCapabilities
: IpTerminalCapabilities
: IpExtendedTerminalCapabilities
1: getTerminalCapability( )
2: new( )
3: triggeredTerminalCapabilityStart( )
4: triggeredTerminalCapabilityReport( )
5: forward notification
6: triggeredTerminalCapabilityReport( )
7: forward notification
8: triggeredTerminalCapabilityError( )
9: forward error
10: triggeredTerminalCapabilityReport( )
11: forward notification
12: triggeredTerminalCapabilityStop( )

6 Class Diagrams

Terminal Capabilities Class Diagram:





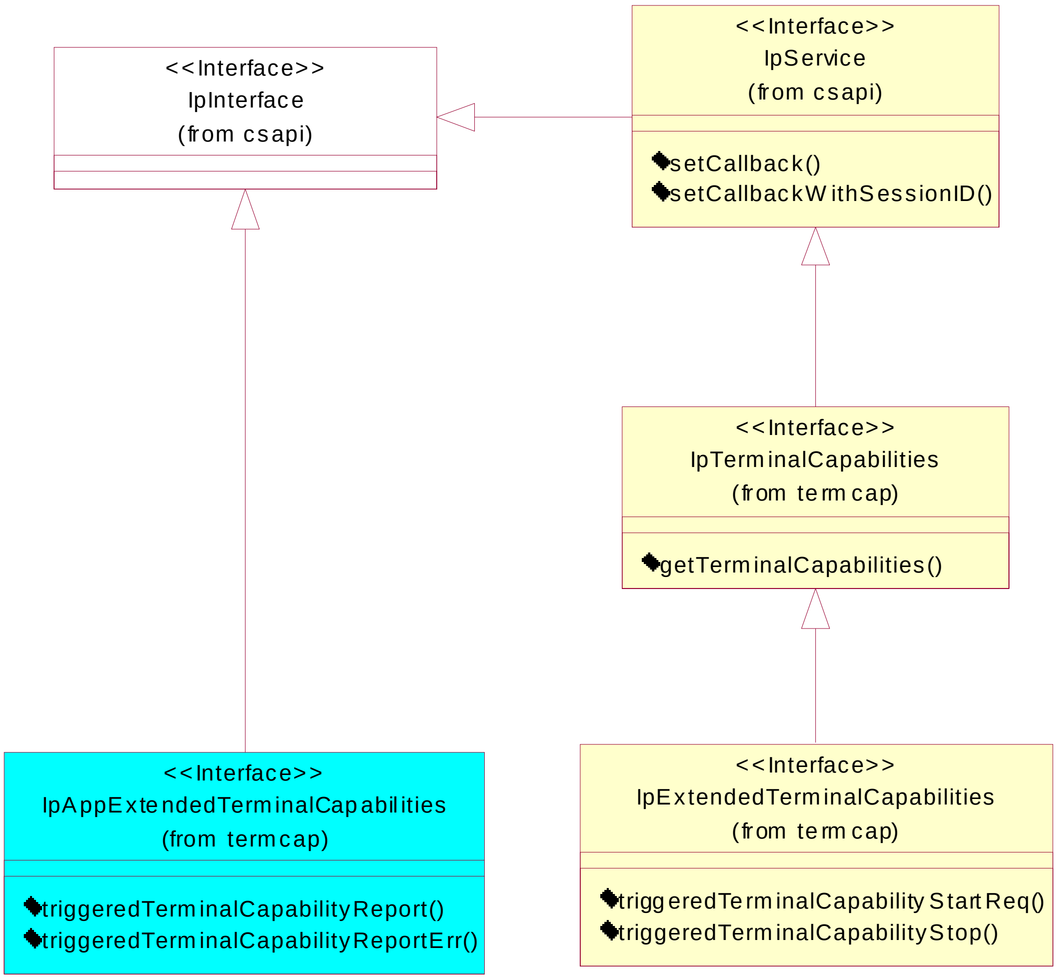

<<Interface>>
IpInterface
(from csapi)
<<Interface>>
IpService
(from csapi)
setCallback()
setCallbackWithSessionID()
<<Interface>>
IpTerminalCapabilities
(from termcap)
getTerminalCapabilities()
<<Interface>>
IpAppExtendedTerminalCapabilities
(from termcap)
triggeredTerminalCapabilityReport()
triggeredTerminalCapabilityReportErr()
<<Interface>>
IpExtendedTerminalCapabilities
(from termcap)
triggeredTerminalCapabilityStartReq()
triggeredTerminalCapabilityStop()

Figure: Terminal Capabilities Class Diagrams~~Package Overview~~

7 The Service Interface Specifications

7.1 Interface Specification Format

This section 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. All methods in the API return a value of type `TpResult`, indicating, amongst other things, if the method invocation was sucessfully executed or not.

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.

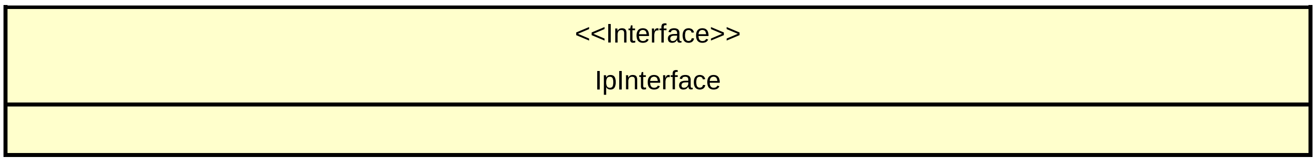
7.1.4 State Model

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.

<<Interface>>

IpService

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 SessionID's.

Parameters

`appInterface : in IpInterfaceRef`

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

Raises

`TpCommonExceptions`

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 uses SessionID's.

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`

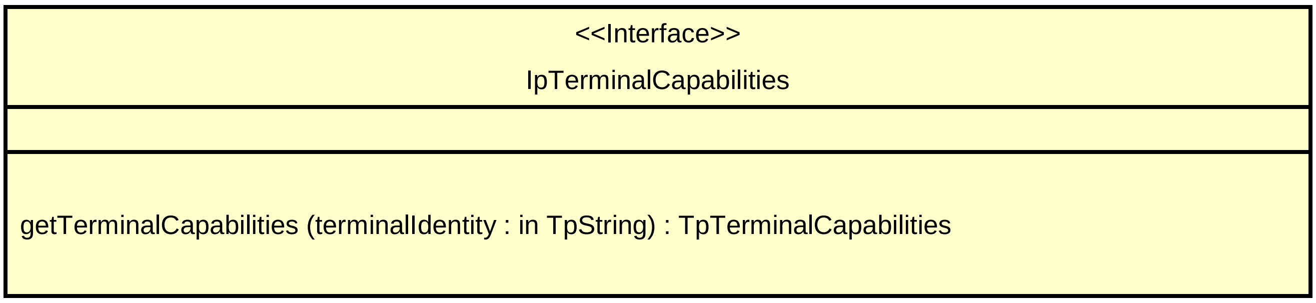
8 Terminal Capabilities Interface Classes

The Terminal Capabilities SCF enables the application to retrieve the terminal capabilities of the specified terminal. Additionally it is possible for the application to request notifications when the capabilities of the terminal change in some way. The Terminal Capabilities service provides ~~a~~ SCF interfaces ~~that is called~~ IpTerminalCapabilities and IpExtendedTerminalCapabilities. The application side interface for the reporting is called IpAppExtendedTerminalCapabilities. ~~There is no need for an application interface, since IpTerminalCapabilities only contains the synchronous method getTerminalCapabilities.~~

8.1 Interface Class IpTerminalCapabilities

Inherits from: ~~IpInterface~~IpService.

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). The network may override some capabilities that have been indicated by the terminal itself due to network policies or other restrictions or modifications in the supported capabilities.


<<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 [1] and adopted in the WAP UAProf specification [2]. 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`

8.2 Interface Class IpExtendedTerminalCapabilities

Inherits from: IpTerminalCapabilities.

This interface can be used as an extended version of terminal capability monitoring.

The application programmer can use this interface to request terminal capability reports that are triggered by their changes. Note that the underlying mechanisms for this network feature are currently not fully standardised.

<<Interface>> IpExtendedTerminalCapabilities
triggeredTerminalCapabilityStartReq (appTerminalCapabilities : in IpAppExtendedTerminalCapabilitiesRef, terminals : in TpAddressSet, capabilityScope : in TpString, criteria : in TpTerminalCapabilityChangeCriteria) : TpAssignmentID triggeredTerminalCapabilityStop (assignmentID : in TpAssignmentID) : void

Method
`triggeredTerminalCapabilityStartReq()`

Request for terminal capability reports when the capabilities change or when the application obviously does not have the current terminal capability information when this method is invoked.

Returns: assignmentID

Specifies the assignment ID of the triggered terminal capability reporting request.

Parameters

`appTerminalCapabilities : in IpAppExtendedTerminalCapabilitiesRef`

Specifies the application interface for callbacks.

`terminals : in TpAddressSet`

Specifies the terminal(s) for which the capabilities shall be reported. TpAddress fields have the following use:

- Plan: Used to indicate the numbering plan
- AddrString: Used to indicate the subscriber address
- Name: Used to indicate the terminal identity. May be applied also together with AddrString to indicate subscriber's particular terminal. The precise format is not defined.
- Presentation: No defined use
- Screening: No defined use
- SubAddressString: No defined use

Hence it is possible to indicate the subscriber and/or the terminal identification. This terminal addressing is implementation specific e.g. subscriber identification may not always be sufficient information to get the capabilities of the terminal.

`capabilityScope : in TpTerminalCapabilityScope`~~`String`~~

Specifies the scope of the capabilities that the application is interested in. The contents are implementation specific. One possibility is to use the CC/PP definitions as in TpTerminalCapabilities.

`criteria : in TpTerminalCapabilityChangeCriteria`

Specifies the trigger conditions for the reports e.g. software or hardware update.

Returns

`TpAssignmentID`

Raises

`TpCommonExceptions, P_INFORMATION_NOT_AVAILABLE, P_INVALID_INTERFACE_TYPE, P_INVALID_CRITERIA, P_INVALID_TERMINAL_ID`

Method
`triggeredTerminalCapabilityStop()`

Stop reporting for terminal capability changes that were started by triggeredTerminalCapabilityStartReq().

Parameters

`assignmentID : in TpAssignmentID`

Specifies the assignment ID for the task to be stopped.

Raises

`TpCommonExceptions, P_INVALID_ASSIGNMENT_ID`

8.3 Interface Class IpAppExtendedTerminalCapabilities

Inherits from: IpInterface.

IpAppExtendedTerminalCapabilities interface is used to send triggered terminal capability reports. It is implemented by the client application developer.

<<Interface>> IpAppExtendedTerminalCapabilities
triggeredTerminalCapabilityReport (assignmentID : in TpAssignmentID, terminals : in TpAddressSet, criteria : in TpTerminalCapabilityChangeCriteria, capabilities : in TpTerminalCapabilities) : void triggeredTerminalCapabilityReportErr~~or~~ (assignmentID : in TpAssignmentID, terminals : in TpAddressSet, cause : in TpTerminalCapabilitiesError) : void

Method
triggeredTerminalCapabilityReport()

This terminal capability report is issued when the capabilities of the terminal have changed in the way specified by the criteria parameter in the previously invoked triggeredTerminalCapabilityStartReq () method.

Parameters

assignmentID : in TpAssignmentID

Specifies the assignment ID of the report.

terminals : in TpAddressSet

Specifies the terminal(s) either by subscriber or terminal ID or both as described for the triggeredTerminalCapabilityStartReq () method.

criteria : in TpTerminalCapabilityChangeCriteria

Specifies the criteria that caused the report to be sent.

capabilities : in TpTerminalCapabilities

Specifies the capabilities of the terminal. The network may override some capabilities that have been indicated by the terminal itself due to network policies or other restrictions or modifications in the supported capabilities.

Method
triggeredTerminalCapabilityReportErr~~or~~()

This method indicates that the requested reporting has failed. Note that errors may concern the whole assignment or just some terminals. In the former case no terminals are specified.

Parameters

assignmentId : in TpAssignmentID

Specifies the assignment ID.

terminals : in TpAddressSet

Specifies the terminal(s) either by subscriber or terminal ID or both as described for the triggeredTerminalCapabilityStartReq () method.

cause : in TpTerminalCapabilitiesError

Specifies the error that led to the failure.

9 State Transition Diagrams

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

10 Service Properties

List of Service Properties

The following table lists properties relevant for this SCF.

Property	Type	Description
P_TRIGGERED_REPORTING_SUPPORTED	BOOLEAN_SET	Value = TRUE : The triggered reporting of terminal capabilities is supported by the SCF. Value = FALSE : The triggered reporting of terminal capabilities is not supported by the SCF.

11 Terminal Capabilities Data Definitions

The constants and types defined in the following clauses are defined in the *org.osa.termcap* package.

terminalIdentity

Identifies the terminal.

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

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 [4] and adopted in the WAP UAProf specification [5]. It contains URLs; terminal attributes and values, in RDF format; or a combination of both.

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 can not 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.
P_TERMCAP_INFO_UNAVAILABLE	3	The terminal capability information is not available.

TpTerminalCapabilityChangeCriteria

Defines the type of the terminal capability changes to be reported. The values may be combined by a logical 'OR' function.

Name	Value	Description
P_TERMINAL_CAPABILITY_CHANGE_CRITERIA_UNDEFINED	00h	Undefined
P_TERMINAL_CAPABILITY_CHANGE_CRITERIA_GENERAL	01h	Any change in the terminal capabilities.
P_TERMINAL_CAPABILITY_CHANGE_CRITERIA_HW_UPDATE	02h	The terminal device hardware has been modified or replaced completely.
P_TERMINAL_CAPABILITY_CHANGE_CRITERIA_SW_UPDATE	04h	The software of the terminal has been updated in any way. Also changes in configuration or preferences may be concerned.
P_TERMINAL_CAPABILITY_CHANGE_CRITERIA_INITIAL	08h	The initial device capabilities reported when monitoring has been started by an application.

TpTerminalCapabilityScopeType

Defines a specific type of the terminal capability scope definition.

Name	Value	Description
P_TERMINAL_CAPABILITY_SCOPE_TYPE_UNDEFINED	0	Undefined.
P_TERMINAL_CAPABILITY_SCOPE_TYPE_CCPP	1	Indicates that the terminal capability scope is expressed as CC/PP headers as specified in W3C [4] and adopted in the WAP UAProf specification [5]. It contains URLs; terminal attributes and values, in RDF format; or a combination of both.

TpTerminalCapabilityScope

Defines the `Sequence of Data Elements` that specify the scope of the terminal capabilities.

Sequence Element Name	Sequence Element Type
ScopeType	TpTerminalCapabilityScopeType
Scope	TpString

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 can not 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 2919807IDL.ZIP) which accompanies the present document.

Annex B (informative): Differences between this draft and 3GPP TS 29.198 R99

getTerminalCapabilities now throws TpCommonExceptions and individual, identified exceptions

All methods now return void or the former out parameter.

Annex C (informative): Change history

Change history							
Date	**TSG #**	**TSG Doc.**	**CR**	**Rev**	**Subject/Comment**	**Old**	**New**
Mar 2001	CN_11	NP-010134	047	--	CR 29.198: for moving TS 29.198 from R99 to Rel 4 (N5-010158)	3.2.0	4.0.0
Jun 2001	CN_12	NP-010330	001	--	Corrections to OSA API Rel4	4.0.0	4.1.0
Sep 2001	CN_13	NP-010470	002	--	Changing references to JAIN	4.1.0	4.2.0
Dec 2001	CN_14	NP-010600	003	--	Replace Out Parameters with Return Types	4.2.0	4.3.0