ETSI TS 129 198-4-1 V9.0.0 (2010-01)

Technical Specification

Digital cellular telecommunications system (Phase 2+);
Universal Mobile Telecommunications System (UMTS);
LTE;
Open Service Access (OSA)
Application Programming Interface (API);
Part 4: Call control;
Subpart 1: Call control common definitions
(3GPP TS 29.198-04-1 version 9.0.0 Release 9)

Reference
RTS/TSGC-0029198-04-1V900

Keywords
GSM, LTE, UMTS

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

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 Technical Specification (TS) has been produced by ETSI 3rd Generation Partnership Project (3GPP).

The present document may refer to technical specifications or reports using their 3GPP identities, UMTS identities or GSM identities. These should be interpreted as being references to the corresponding ETSI deliverables.

The cross reference between GSM, UMTS, 3GPP and ETSI identities can be found under http://webapp.etsi.org/key/queryform.asp.

Contents

Foreword

This Technical Specification has been produced by the 3rd Generation Partnership Project (3GPP).

The contents of the present document are subject to continuing work within the TSG and may change following formal TSG approval. Should the TSG modify the contents of the present document, it will be re-released by the TSG with an identifying change of release date and an increase in version number as follows:

Version x.y.z

where:

- x the first digit:
  - 1 presented to TSG for information;
  - 2 presented to TSG for approval;
  - 3 or greater indicates TSG approved document under change control.
- y the second digit is incremented for all changes of substance, i.e. technical enhancements, corrections, updates, etc.
- z the third digit is incremented when editorial only changes have been incorporated in the document.

Introduction

The present document is part 4, sub-part 1 of a multi-part TS covering the 3rd Generation Partnership Project: Technical Specification Group Core Network; Open Service Access (OSA); Application Programming Interface (API), as identified below. The **API specification** (3GPP TS 29.198) is structured in the following Parts:

Part 1:	"Overview";	
Part 2:	"Common Data Definitions";	
Part 3:	"Framework";	
Part 4:	**"Call Control";**	
	Sub-part 1: "Call Control Common Definitions";	
	Sub-part 2: "Generic Call Control SCF";	
	Sub-part 3: "Multi-Party Call Control SCF";	
	Sub-part 4: "Multi-Media Call Control SCF";	
	Sub-part 5: "Conference 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";	(not part of 3GPP Release 8)
Part 10:	"Connectivity Manager SCF";	(new in 3GPP Release 8)
Part 11:	"Account Management SCF";	
Part 12:	"Charging SCF".	
Part 13:	"Policy Management SCF";	
Part 14:	"Presence and Availability Management SCF";	
Part 15:	"Multi Media Messaging SCF";	
Part 16:	"Service Broker SCF".	

The **Mapping specification of the OSA APIs and network protocols** (3GPP TR 29.998) is also structured as above. A mapping to network protocols is however not applicable for all Parts, but the numbering of Parts is kept. Also in case a Part is not supported in a Release, the numbering of the parts is maintained.

Table: Overview of the OSA APIs & Protocol Mappings 29.198 & 29.998-family

OSA API specifications 29.198-family						OSA API Mapping - 29.998-family	
29.198-01	Overview					29.998-01	Overview
29.198-02	Common Data Definitions					*29.998-02*	*Not Applicable*
29.198-03	Framework					*29.998-03*	*Not Applicable*
Call Control (CC) SCF	**29.198-04-1 Common CC data definitions**	29.198-04-2 Generic CC SCF	29.198-04-3 Multi-Party CC SCF	29.198-04-4 Multi-media CC SCF	29.198-04-5 Conf CC SCF	29.998-04-1	Generic Call Control – CAP mapping
						29.998-04-2	*Generic Call Control – INAP mapping*
						29.998-04-3	*Generic Call Control – Megaco mapping*
						29.998-04-4	Multiparty Call Control – ISC mapping
29.198-05	User Interaction SCF					29.998-05-1	User Interaction – CAP mapping
						29.998-05-2	*User Interaction – INAP mapping*
						29.998-05-3	*User Interaction – Megaco mapping*
						29.998-05-4	User Interaction – SMS mapping
29.198-06	Mobility SCF					29.998-06-1	User Status and User Location – MAP mapping
						29.998-06-2	User Status and User Location – SIP mapping
29.198-07	Terminal Capabilities SCF					*29.998-07*	*Not Applicable*
29.198-08	Data Session Control SCF					29.998-08	Data Session Control – CAP mapping
29.198-09	*Generic Messaging SCF*					*29.998-09*	*Not Applicable*
29.198-10	Connectivity Manager SCF					*29.998-10*	*Not Applicable*
29.198-11	Account Management SCF					*29.998-11*	*Not Applicable*
29.198-12	Charging SCF					*29.998-12*	*Not Applicable*
29.198-13	Policy Management SCF					*29.998-13*	*Not Applicable*
29.198-14	Presence & Availability Management SCF					*29.998-14*	*Not Applicable*
29.198-15	Multi-media Messaging SCF					*29.998-15*	*Not Applicable*
29.198-16	Service Broker SCF					*29.998-16*	*Not Applicable*

1 Scope

The present document is Part 4, Sub-part 1 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 concepts and the functional architecture for the OSA are contained in 3GPP TS 23.198 [3]. The requirements for OSA are contained in 3GPP TS 22.127 [2].

The present document specifies the common definitions used by the Call Control Service Capability Features (SCF).

This specification has been defined jointly between 3GPP TSG CT WG5, ETSI TISPAN and the Parlay Group, in co-operation with a number of JAIN™ Community member companies.

Maintenance of up to 3GPP Rel-8 and new OSA Stage 1, 2 and 3 work beyond Rel-9 was moved to OMA in June 2008.

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 29.198-01: "Open Service Access (OSA) Application Programming Interface (API); Part 1: Overview".

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

[3] 3GPP TS 23.198: "Open Service Access (OSA); Stage 2".

[4] 3GPP TS 22.002: "Circuit Bearer Services (BS) supported by a Public Land Mobile Network (PLMN)".

[5] ISO 4217 (1995): "Codes for the representation of currencies and funds ".

[6] 3GPP TS 24.002: "GSM-UMTS Public Land Mobile Network (PLMN) Access Reference Configuration".

[7] 3GPP TS 22.003: "Circuit Teleservices supported by a Public Land Mobile Network (PLMN)".

[8] ITU-T Recommendation Q.1238-2 (2000): "Interface Recommendation for Intelligent Network Capability Set 3: SCF - SSF interface".

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 Call Control SCF

Four flavours of Call Control (CC) APIs have been included in 3GPP Release 7. These are Generic Call Control (GCC), Multi-Party Call Control (MPCC), Multi-Media Call Control (MMCC) and Conference Call Control (CCC). The GCC is the same API as was already present in the Release 99 specification (TS 29.198 v3.3.0). Multi-Party Call Contorl was introduced in the Release 4 specifications, and Multi-Media Call Control is introduced in Release 5. Conference Call Control was introduced in Release 7.

The joint work between 3GPP CT5, ETSI TISPAN and the Parlay CC Working group with collaboration from JAIN has been focussed on the MPCC and MMCC APIs. A number of improvements on CC functionality have been made and are reflected in these APIs. For this it was necessary to break the inheritance that previously existed between GCC and MPCC.

The joint CC group has furthermore decided that the MPCC is to be considered as the future base CC family and the technical work will not be continued on GCC. Errors or technical flaws will of course be corrected.

4.1 Call Model Description

The call model used for the Call Control SCFs has the following objects.

* a call object. A call is a relation between a number of parties. The call object relates to the entire call view from the application. E.g., the entire call will be released when a release is called on the call. Note that different applications can have different views on the same physical call, e.g., one application for the originating side and another application for the terminating side. The applications will not be aware of each other, all 'communication' between the applications will be by means of network signalling. The API currently does not specify any feature interaction mechanisms.

* a call leg object. The leg object represents a logical association between a call and an address. The relationship includes at least the signalling relation with the party. The relation with the address is only made when the leg is routed. Before that the leg object is IDLE and not yet associated with the address.

* an address. The address logically represents a party in the call.

* a terminal. A terminal is the end-point of the signalling and/or media for a party. This object type is currently not addressed.

The call object is used to establish a relation between a number of parties by creating a leg for each party within the call.

Associated with the signalling relationship represented by the call leg, there may also be a bearer connection (e.g., in the traditional voice only networks) or a number (zero or more) of media channels (in multi-media networks).

A leg can be attached to the call or detached from the call. When the leg is attached, this means that media or bearer channels related to the legs are connected to the media or bearer channels of the other legs that are attached to the same call. I.e., only legs that are attached can 'speak' to each other. A leg can have a number of states, depending on the signalling received from or sent to the party associated with the leg. Usually there is a limit to the number of legs that are in being routed (i.e., the connection is being established) or connected to the call (i.e., the connection is established). Also, there usually is a limit to the number of legs that can be simultaneously attached to the same call.

Some networks distinguish between controlling and passive legs. By definition the call will be released when the controlling leg is released. All other legs are called passive legs. There can be at most one controlling leg per call. However, there is currently no way the application can influence whether a Leg is controlling or not.

There are two ways for an application to get the control of a call. The application can request to be notified of calls that meet certain criteria. When a call occurs in the network that meets these criteria, the application is notified and can control the call. Some legs will already be associated with the call in this case. Another way is to create a new call from the application.

4.2 Structure of Call Control SCF Documentation

Each of the Call Control SCFs is specified under the following headings:

- The Sequence diagrams give the reader a practical idea of how each of the SCF is implemented.
- The Class relationships clause shows 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 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 shows 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 (TS 29.198-2).

4.3 General requirements on support of methods

An implementation of one of the call control APIs which supports or implements a method described in one of the sub-parts of TS 29.198-04, shall support or implement the functionality described for that method, for at least one valid set of values for the parameters of that method.

Where a method is not supported by an implementation of a Service interface, the exception P_METHOD_NOT_SUPPORTED shall be returned to any call of that method.

Where a method is not supported by an implementation of an Application interface, a call to that method shall be possible, and no exception shall be returned.

4.4 Application Control of a Call or Session

4.4.1 Introduction

Services should be provision-able by multiple independent parties and therefore multiple applications may apply control to the same instance of a call or a session. How this may be enabled is further described in the following.

However, first some reflections on what is meant with application control:

Single application control may be classified as to allow at the same point in time during call or session processing only one application to be capable to influence the call or session. This does not exclude more applications on the same call, but they cannot operate at the same time. This is referred to as ***"Single point of Control (SPC)"*** in IN terminology.

Multiple application control may be classified as to allow at the same point in time during call or session processing more than one application to be capable to influence the call or session. This is referred to as ***"Multiple Points of Control (MPC)"*** in IN terminology.
MPC will demand some rules for event handling among multiple applications on a call like the cascaded chain principle as applied in IN CS3 [8], where MPC has been introduced.

4.4.2 Concept of Multiple Points of Control

The term "multiple points of control" refers to the situation when multiple concurrently executing applications apply control to one and the same instance of a call or session.

General Objective:

"If there is more than one controlling application acting on the same call or session, then the event notification detection point processing requested by any of the involved applications shall be performed in the same way as if notification reporting had occurred in different call or session control instances, which are separated by a Network Node interface".

NOTE: The objective description above is taken from [8], but slightly generalized to become non-IN specific.

The MPC general objective signifies the cascaded chain principle, which is further explained below through an informative cascaded chain model.

4.4.2.1 Cascaded Chain Model

When services running in different nodes apply control to a call or session, even if they are unaware of each other, they provide a cascading of applications. They provide a natural ordering of applications. This ordering can be said to be upstream or downstream. A downstream ordering is the ordering of applications as they are invoked downstream in the network from the calling party (e.g. origin client, UAC) toward the called party (e.g. destination server, UAS). An upstream ordering is the ordering of applications as they are invoked upstream in the network from the called party toward the calling party.

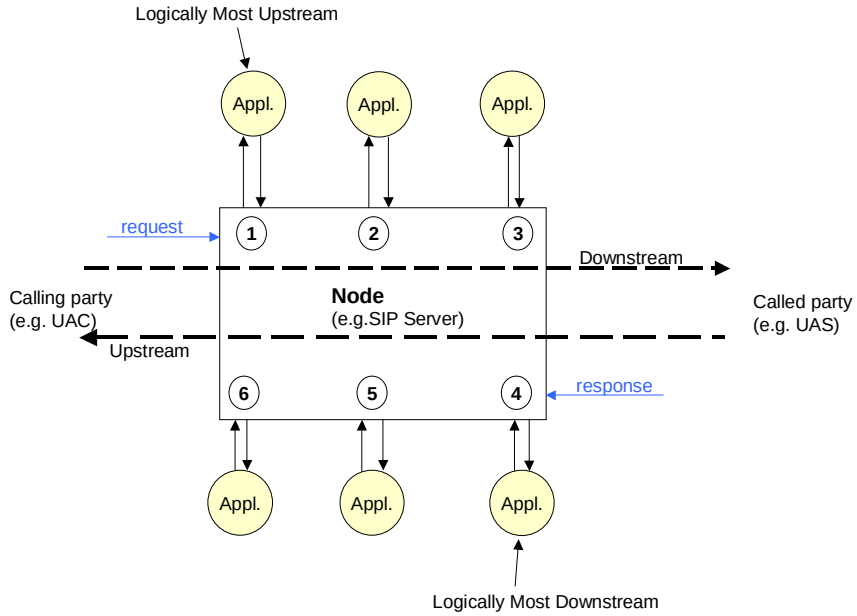


Figure 1 : Cascaded Chain Model

When applications that apply control to one instance of a call or session are invoked in some order, on the same node, the applications can be thought of as cascaded.

When a "request" is received then the earlier the application is invoked, based on this event, the more logically upstream it is considered to be in the chain of cascaded applications. When a response is received then the earlier the applications is invoked, based on this event, the more logically downstream it is considered to be in the chain of cascaded applications.

This means that the applications are treated as if they were triggered in different nodes (or hosts). This model is conceptually simple and may provide a natural algorithm for resolving conflicts between the instructions of multiple service applications at the same call or session event.

On reception of a call or session event in the network the actions are executed in order of priority in the following manner:

1) Control is passed to the first application.

2) Some response is received from the first application.

3) Control is passed to the second application.

4) Some response is received from the second application and so on.

In this way a decision about whether to invoke a subsequent application can depend on the output from the previous application.

If the first application terminates the request then the second application must not be invoked.

The most simple form of cascading is an order based in which the applications are triggered on a single event.

If for example three application "X1" "Y1" and "Z1" need to be triggered on event "e"; then specifying an order say X1-Y1-Z1 for event e means that first "X1" is contacted and its instructions taken, then the output of this is passed to "Y1" and finally the output of "Y1" is passed to "Z1". (The letter represents the application and the number an instance of the application running for a given user.)

How about future events that should invoke the same instances of "X" "Y" and "Z"? For these events there could be a totally different order, specified say YXZ. It would however be administratively very complex if one has to specify an order for all instances for all events. Thus a general basic principle may be useful like the cascaded chain for defining a default ordering in the cases where this is not possible. The actual order depends on whether the event is coming from the calling party (downstream) or the called party (upstream).

If however a new application A1 should be triggered on "er" then it would be necessary to place this somewhere in the order. However, a default general assumption can be that event reporting to already invoked applications should have precedence over the invocation of new applications [8].

Anyhow, as a network operator's option any MPC generalized rules specified may be replaced or enhanced by network operator specific rules.

4.4.3 Service Interactions

A variety of different services, service enablers, and capabilities are being standardized. There are potential feature interactions among these various services, service enablers, and capabilities.

Conflicts, incompatibilities, or modifications of one feature's characteristics due to interactions with another active feature need to be resolved in case of multiple services.

In case there are multiple initial notification requests (filter criteria) assigned for one subscriber, a priority describe the order in which the applications shall be contacted when the call or session encounters an event that matches the initial filter criteria. Handling of service/application interaction issues to prevent undesired interactions when multi services are to be supported is outside the scope of the present document. It is the basic assumption that the network operator is responsible for the provisioning of triggers in the network as in this domain full awareness exists of all other services and applications.

4.4.4 Multiple services - applicability of MPC for Parlay/OSA

As far as the OSA API is concerned a user can choose his clients as he wishes, i.e. the user may subscribe to more services over the network, each one being to a separate client application, not necessary placed in the same physical entity.

From an OSA gateway side it is a network implementation issue if multiple or single point of control mechanisms are supported. Multi service support extends the number of applications that can register for notifications. An example of multi service support network could be an OSA Gateway in a UMTS IMS network or IN network complying with the multiple points of control principle as defined for IN CS3.

Overlapping criteria have been defined for GCCS and MPCCS to prevent multiple points of control, leading to possible interaction problems. Where Multi service support is provided, the overlap criteria rules as defined to secure single point of control can be overruled.

5 The Service Interface Specifications

5.1 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.

5.1.1 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>.

5.1.2 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.

5.1.3 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.

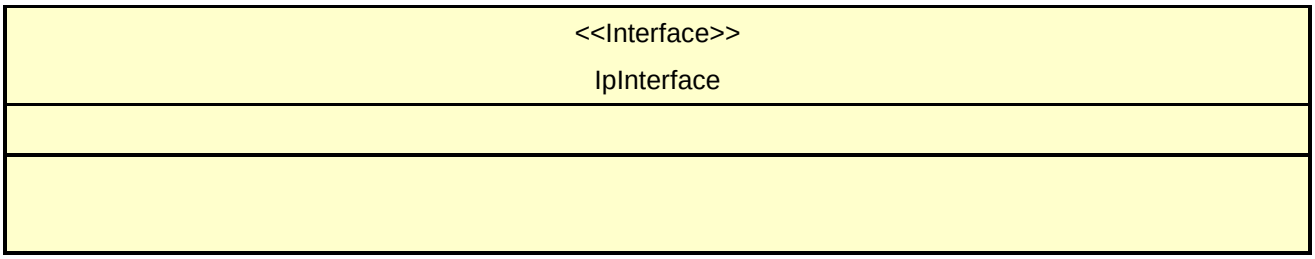
5.1.4 State Model

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

5.2 Base Interface

5.2.1 Interface Class IpInterface

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



5.3 Service Interfaces

5.3.1 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.

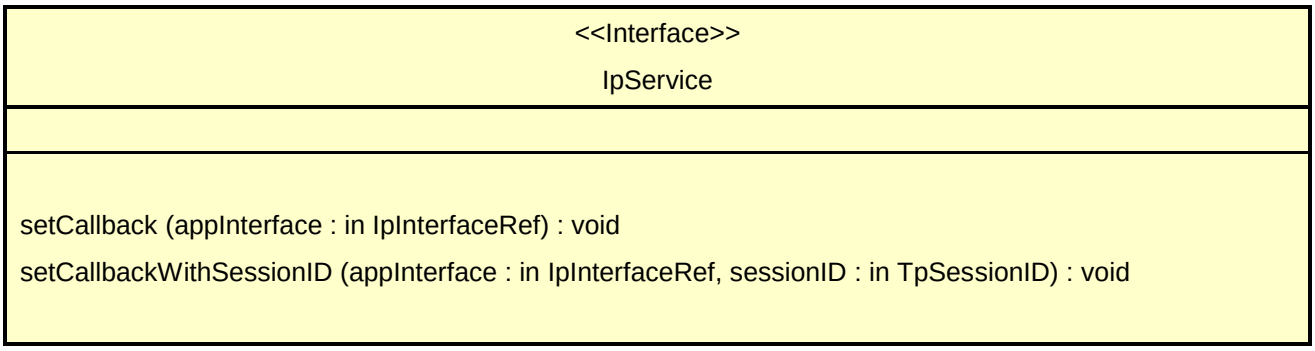
The interfaces that are implemented by the services are denoted as 'Service Interface'. The corresponding interfaces that must be implemented by the application (e.g. for API callbacks) are denoted as 'Application Interface'.

5.4 Generic Service Interface

5.4.1 Interface Class IpService

Inherits from: IpInterface

All service interfaces inherit from the following interface.



5.4.1.1 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. Multiple invocations of this method on an interface shall result in multiple callback references being specified. The SCS shall use the most recent callback interface provided by the application using this method. In the event that a callback reference fails or is no longer available, the next most recent callback reference available shall be used.

Parameters

`appInterface : in IpInterfaceRef`

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

Raises

`TpCommonExceptions, P_INVALID_INTERFACE_TYPE`

5.4.1.2 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. Multiple invocations of this method on an interface shall result in multiple callback references being specified. The SCS shall use the most recent callback interface provided by the application using this method. In the event that a callback reference fails or is no longer available, the next most recent callback reference available shall be used.

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

6 Common Call Control Data Types

The following data types referenced in this clause are defined in 3GPP TS 29.198-5:

TpUIInfo

All other data types referenced but not defined in this clause are common data definitions which may be found in 3GPP TS 29.198-2.

6.1 TpCallAlertingMechanism

This data type is identical to a TpInt32, and defines the mechanism that will be used to alert a call party. The values of this data type are operator specific.

6.2 TpCallBearerService

This data type defines the type of call application-related specific information (Q.931: Information Transfer Capability, and 3G TS 22.002).

Name	Value	Description
P_CALL_BEARER_SERVICE_UNKNOWN	0	Bearer capability information unknown at this time
P_CALL_BEARER_SERVICE_SPEECH	1	Speech
P_CALL_BEARER_SERVICE_DIGITALUNRESTRICTED	2	Unrestricted digital information
P_CALL_BEARER_SERVICE_DIGITALRESTRICTED	3	Restricted digital information
P_CALL_BEARER_SERVICE_AUDIO	4	3,1 kHz audio
P_CALL_BEARER_SERVICE_DIGITALUNRESTRICTED TONES	5	Unrestricted digital information with tones/announcements
P_CALL_BEARER_SERVICE_VIDEO	6	Video

6.3 TpCallChargePlan

Defines the `Sequence of Data Elements` that specify the charge plan for the call.

Sequence Element Name	Sequence Element Type	Description
ChargeOrderType	TpCallChargeOrderCategory	Charge order
TransparentCharge	TpOctetSet	Operator specific charge plan specification, e.g. charging table name / charging table entry. The associated charge plan data will be sent transparently to the charging records. Only applicable when transparent charging is selected.
ChargePlan	TpInt32	Pre-defined charge plan. Example of the charge plan set from which the application can choose could be : (0 = normal user, 1 = silver card user, 2 = gold card user). Only applicable when predefined charge plan is selected.
AdditionalInfo	TpOctetSet	Descriptive string which is sent to the billing system without prior evaluation. Could be included in the ticket.
PartyToCharge	TpCallPartyToChargeType	Identifies the entity or party to be charged for the call or call leg.
PartyToChargeAdditionalInfo	TpCallPartyToChargeAdditionalInfo	Contains additional information regarding the charged party.

6.4 TpCallPartyToChargeAdditionalInfo

Defines the Tagged Choice of Data Elements that identifies the entity or party to be charged.

	Tag Element Type		
	TpCallPartyToChargeType		

Tag Element Value	Choice Element Type	Choice Element Name
P_CALL_PARTY_ORIGINATING, ,	NULL	Undefined
P_CALL_PARTY_DESTINATION,	NULL	Undefined
P_CALL_PARTY_SPECIAL	TpAddress	CallPartySpecial

6.5 TpCallPartyToChargeType

Defines the type of call party to charge.

Name	Value	Description
P_CALL_PARTY_ORIGINATING	0	Calling party, i.e. party that initiated the call. For application initiated calls this indicates the first party of the call
P_CALL_PARTY_DESTINATION	1	Called party
P_CALL_PARTY_SPECIAL	2	An address identifying e.g. a third party, a service provider

6.6 TpCallChargeOrderCategory

Defines the type of charging to be applied.

Name	Value	Description
P_CALL_CHARGE_TRANSPARENT	0	Operator specific charge plan specification, e.g. charging table name / charging table entry. The associated charge plan data will be send transparently to the charging records
P_CALL_CHARGE_PREDEFINED_SET	1	Pre-defined charge plan. Example of the charge plan set from which the application can choose could be : (0 = normal user, 1 = silver card user, 2 = gold card user).

6.7 TpCallEndedReport

Defines the Sequence of Data Elements that specify the reason for the call ending.

Sequence Element Name	Sequence Element Type	Description
CallLegSessionID	TpSessionID	The leg that initiated the release of the call. If the call release was not initiated by the leg, then this value is set to –1.
Cause	TpReleaseCause	The cause of the call ending.

6.8 TpCallError

Defines the Sequence of Data Elements that specify the additional information relating to a call error.

Sequence Element Name	Sequence Element Type
ErrorTime	TpDateAndTime
ErrorType	TpCallErrorType
AdditionalErrorInfo	TpCallAdditionalErrorInfo

6.9 TpCallAdditionalErrorInfo

Defines the Tagged Choice of Data Elements that specify additional call error and call error specific information. This is also used to specify call leg errors and information errors.

	Tag Element Type	
	TpCallErrorType	

Tag Element Value	Choice Element Type	Choice Element Name
P_CALL_ERROR_UNDEFINED	NULL	Undefined
P_CALL_ERROR_INVALID_ADDRESS	TpAddressError	CallErrorInvalidAddress
P_CALL_ERROR_INVALID_STATE	NULL	Undefined
P_CALL_ERROR_RESOURCE_UNAVAILABLE	NULL	Undefined

6.10 TpCallErrorType

Defines a specific call error.

Name	Value	Description
P_CALL_ERROR_UNDEFINED	0	Undefined; the method failed or was refused, but no specific reason can be given.
P_CALL_ERROR_INVALID_ADDRESS	1	The operation failed because an invalid address was given
P_CALL_ERROR_INVALID_STATE	2	The call was not in a valid state for the requested operation
P_CALL_ERROR_RESOURCE_UNAVAILABLE	3	There are not enough resources to complete the request successfully

6.11 TpCallInfoReport

Defines the Sequence of Data Elements that specify the call information requested. Information that was not requested is invalid.

Sequence Element Name	Sequence Element Type	Description
CallInfoType	TpCallInfoType	The type of call report.
CallInitiationStartTime	TpDateAndTime	The time and date when the call, or follow-on call, was started.
CallConnectedToResourceTime	TpDateAndTime	The date and time when the call was connected to the resource. This data element is only valid when information on user interaction is reported.
CallConnectedToDestinationTime	TpDateAndTime	The date and time when the call was connected to the destination (i.e. when the destination answered the call). If the destination did not answer, the time is set to an empty string. This data element is invalid when information on user interaction is reported with an intermediate report.
CallEndTime	TpDateAndTime	The date and time when the call or follow-on call or user interaction was terminated.
Cause	TpReleaseCause	The cause of the termination.

A callInfoReport will be generated at the end of user interaction and at the end of the connection with the associated address. This means that either the destination related information is present or the resource related information, but not both.

6.12 TpCallInfoType

Defines the type of call information requested and reported. The values may be combined by a logical 'OR' function.

Name	Value	Description
P_CALL_INFO_UNDEFINED	00h	Undefined
P_CALL_INFO_TIMES	01h	Relevant call times
P_CALL_INFO_RELEASE_CAUSE	02h	Call release cause

6.13 TpCallLoadControlMechanism

Defines the Tagged Choice of Data Elements that specify the applied mechanism and associated parameters.

	Tag Element Type	
	TpCallLoadControlMechanismType	

Tag Element Value	Choice Element Type	Choice Element Name
P_CALL_LOAD_CONTROL_PER_INTERVAL	TpCallLoadControlIntervalRate	CallLoadControlPerInterval

6.14 TpCallLoadControlIntervalRate

Defines the call admission rate of the call load control mechanism used. This data type indicates the interval (in milliseconds) between calls that are admitted. This data type is identical to a TpInt32.

Name	Value	Description
P_CALL_LOAD_CONTROL_ADMIT_NO_CALLS	0	Infinite interval (do not admit any calls)
	1 - 60000	Duration in milliseconds

6.15 TpCallLoadControlMechanismType

Defines the type of call load control mechanism to use.

Name	Value	Description
P_CALL_LOAD_CONTROL_PER_INTERVAL	0	admit one call per interval

6.16 TpCallMonitorMode

Defines the mode that the call will monitor for events, or the mode that the call is in following a detected event.

Name	Value	Description
P_CALL_MONITOR_MODE_INTERRUPT	0	The call event is intercepted by the call control service and call processing is interrupted. The application is notified of the event and call processing resumes following an appropriate API call or network event (such as a call release)
P_CALL_MONITOR_MODE_NOTIFY	1	The call event is detected by the call control service but not intercepted. The application is notified of the event and call processing continues
P_CALL_MONITOR_MODE_DO_NOT_MONITOR	2	Do not monitor for the event

6.17 TpCallNetworkAccessType

This data defines the bearer capabilities associated with the call. (3G TS 24.002) This information is network operator specific and may not always be available because there is no standard protocol to retrieve the information.

Name	Value	Description
P_CALL_NETWORK_ACCESS_TYPE_UNKNOWN	0	Network type information unknown at this time
P_CALL_NETWORK_ACCESS_TYPE_POT	1	POTS
P_CALL_NETWORK_ACCESS_TYPE_ISDN	2	ISDN
P_CALL_NETWORK_ACCESS_TYPE_DIALUPINTERNET	3	Dial-up Internet
P_CALL_NETWORK_ACCESS_TYPE_XDSL	4	xDSL
P_CALL_NETWORK_ACCESS_TYPE_WIRELESS	5	Wireless

6.18 TpCallPartyCategory

This data type defines the category of a calling party. (Q.763: Calling Party Category / Called Party Category).

Name	Value	Description
P_CALL_PARTY_CATEGORY_UNKNOWN	0	calling party's category unknown at this time
P_CALL_PARTY_CATEGORY_OPERATOR_F	1	operator, language French
P_CALL_PARTY_CATEGORY_OPERATOR_E	2	operator, language English
P_CALL_PARTY_CATEGORY_OPERATOR_G	3	operator, language German
P_CALL_PARTY_CATEGORY_OPERATOR_R	4	operator, language Russian
P_CALL_PARTY_CATEGORY_OPERATOR_S	5	operator, language Spanish
P_CALL_PARTY_CATEGORY_ORDINARY_SUB	6	ordinary calling subscriber
P_CALL_PARTY_CATEGORY_PRIORITY_SUB	7	calling subscriber with priority
P_CALL_PARTY_CATEGORY_DATA_CALL	8	data call (voice band data)
P_CALL_PARTY_CATEGORY_TEST_CALL	9	test call
P_CALL_PARTY_CATEGORY_PAYPHONE	10	payphone

6.19 TpCallServiceCode

Defines the `Sequence of Data Elements` that specify the service code and type of service code received during a call. The service code type defines how the value string should be interpreted.

Sequence Element Name	Sequence Element Type
CallServiceCodeType	TpCallServiceCodeType
ServiceCodeValue	TpString

6.20 TpCallServiceCodeSet

Defines a Numbered Set of Data Elements of TpCallServiceCode.

6.21 TpCallServiceCodeType

Defines the different types of service codes that can be received during the call.

Name	Value	Description
P_CALL_SERVICE_CODE_UNDEFINED	0	The type of service code is unknown. The corresponding string is operator specific.
P_CALL_SERVICE_CODE_DIGITS	1	The user entered a digit sequence during the call. The corresponding string is an ASCII representation of the received digits.
P_CALL_SERVICE_CODE_FACILITY	2	A facility information element is received. The corresponding string contains the facility information element as defined in ITU Q.932
P_CALL_SERVICE_CODE_U2U	3	A user-to-user message was received. The associated string contains the content of the user-to-user information element.
P_CALL_SERVICE_CODE_HOOKFLASH	4	The user performed a hookflash, optionally followed by some digits. The corresponding string is an ASCII representation of the entered digits.
P_CALL_SERVICE_CODE_RECALL	5	The user pressed the register recall button, optionally followed by some digits. The corresponding string is an ASCII representation of the entered digits.

6.22 TpCallSuperviseReport

Defines the responses from the call control service for calls that are supervised. The values may be combined by a logical 'OR' function.

Name	Value	Description
P_CALL_SUPERVISE_TIMEOUT	01h	The call supervision timer has expired
P_CALL_SUPERVISE_CALL_ENDED	02h	The call has ended, either due to timer expiry or call party release. In case the called party disconnects but a follow-on call can still be made also this indication is used.
P_CALL_SUPERVISE_TONE_APPLIED	04h	A warning tone has been applied. This is only sent in combination with P_CALL_SUPERVISE_TIMEOUT
P CALL SUPERVISE UI FINISHED	08h	The user interaction has finished.
P CALL SUPERVISE QOS PARAM CHANGE	10h	The Quality of Service parameters were renegotiated during the active call.

6.23 TpCallSuperviseTreatment

Defines the treatment of the call by the call control service when the call supervision timer expires. The values may be combined by a logical 'OR' function.

Name	Value	Description
P_CALL_SUPERVISE_RELEASE	01h	Release the call when the call supervision timer expires
P_CALL_SUPERVISE_RESPOND	02h	Notify the application when the call supervision timer expires
P_CALL_SUPERVISE_APPLY_TONE	04h	Send a warning tone to the originating party when the call supervision timer expires. If call release is requested, then the call will be released following the tone after an administered time period

6.24 TpCallTeleService

This data type defines the tele-service associated with the call. (Q.763: User Teleservice Information, Q.931: High Layer Compatibility Information, and 3G TS 22.003).

Name	Value	Description
P_CALL_TELE_SERVICE_UNKNOWN	0	Teleservice information unknown at this time
P_CALL_TELE_SERVICE_TELEPHONY	1	Telephony
P_CALL_TELE_SERVICE_FAX_2_3	2	Facsimile Group 2/3
P_CALL_TELE_SERVICE_FAX_4_I	3	Facsimile Group 4, Class I
P_CALL_TELE_SERVICE_FAX_4_II_III	4	Facsimile Group 4, Classes II and III
P_CALL_TELE_SERVICE_VIDEOTEX_SYN	5	Syntax based Videotex
P_CALL_TELE_SERVICE_VIDEOTEX_INT	6	International Videotex interworking via gateways or interworking units
P_CALL_TELE_SERVICE_TELEX	7	Telex service
P_CALL_TELE_SERVICE_MHS	8	Message Handling Systems
P_CALL_TELE_SERVICE_OSI	9	OSI application
P_CALL_TELE_SERVICE_FTAM	10	FTAM application
P_CALL_TELE_SERVICE_VIDEO	11	Videotelephony
P_CALL_TELE_SERVICE_VIDEO_CONF	12	Videoconferencing
P_CALL_TELE_SERVICE_AUDIOGRAPH_CONF	13	Audiographic conferencing
P_CALL_TELE_SERVICE_MULTIMEDIA	14	Multimedia services
P_CALL_TELE_SERVICE_CS_INI_H221	15	Capability set of initial channel of H.221
P_CALL_TELE_SERVICE_CS_SUB_H221	16	Capability set of subsequent channel of H.221
P_CALL_TELE_SERVICE_CS_INI_CALL	17	Capability set of initial channel associated with an active 3,1 kHz audio or speech call.
P_CALL_TELE_SERVICE_DATATRAFFIC	18	Data traffic.
P_CALL_TELE_SERVICE_EMERGENCY_CALLS	19	Emergency Calls
P_CALL_TELE_SERVICE_SMS_MT_PP	20	Short message MT/PP
P_CALL_TELE_SERVICE_SMS_MO_PP	21	Short message MO/PP
P_CALL_TELE_SERVICE_CELL_BROADCAST	22	Cell Broadcast Service
P_CALL_TELE_SERVICE_ALT_SPEECH_FAX_3	23	Alternate speech and facsimile group 3
P_CALL_TELE_SERVICE_AUTOMATIC_FAX_3	24	Automatic Facsimile group 3
P_CALL_TELE_SERVICE_VOICE_GROUP_CALL	25	Voice Group Call Service
P_CALL_TELE_SERVICE_VOICE_BROADCAST	26	Voice Broadcast Service

6.25 TpCallTreatment

Defines the `Sequence of Data Elements` that specify the treatment for calls that will be handled only by the network (for example, call which are not admitted by the call load control mechanism).

Sequence Element Name	Sequence Element Type
CallTreatmentType	TpCallTreatmentType
ReleaseCause	TpReleaseCause
AdditionalTreatmentInfo	TpCallAdditionalTreatmentInfo

6.26 TpCallTreatmentType

Defines the treatment for calls that will be handled only by the network.

Name	Value	Description
P_CALL_TREATMENT_DEFAULT	0	Default treatment
P_CALL_TREATMENT_RELEASE	1	Release the call
P_CALL_TREATMENT_SIAR	2	Send information to the user, and release the call (Send Info & Release)

6.27 TpCallAdditionalTreatmentInfo

Defines the `Tagged Choice of Data Elements` that specify the information to be sent to a call party.

	Tag Element Type	
	TpCallTreatmentType	

Tag Element Value	Choice Element Type	Choice Element Name
P_CALL_TREATMENT_DEFAULT	NULL	Undefined
P_CALL_TREATMENT_RELEASE	NULL	Undefined
P_CALL_TREATMENT_SIAR	TpUIInfo	InformationToSend

6.28 TpMediaType

Defines the media type of a media stream. The values may be combined by a logical 'OR' function.

Name	Value	Description
P_AUDIO	1	Audio stream
P_VIDEO	2	Video stream
P_DATA	4	Data stream (e.g., T.120)

Annex A (normative): OMG IDL Description of Common Call Control Data Types

The OMG IDL representation of this interface specification is contained in the text file common_cc_data.idl (contained in archive 2919804-1V800IDL.ZIP) which accompanies the present document.

Annex B (informative):

W3C WSDL Description of Common Call Control Data Types

The W3C WSDL representation of this interface specification is contained in zip file 2919804-1V800WSDL.ZIP, which accompanies the present document.

Annex C (informative): Java API Description of the Call Control SCFs

The Java API realisation of this interface specification is produced in accordance with the Java Realisation rules defined in Part 1 of this specification. These rules aim to deliver for Java, a developer API, provided as a realisation, supporting a Java API that represents the UML specifications. The rules support the production of both J2SE and J2EE versions of the API from the common UML specifications.

The J2SE representation of this interface specification is provided as Java Code, contained in archive 2919804-1V800J2SE.ZIP that accompanies the present document.

The J2EE representation of this interface specification is provided as Java Code, contained in archive 2919804-1V800J2EE.ZIP that accompanies the present document.

Annex D (informative): Description of Call Control Sub-part 1: Call Control Common Definitions for 3GPP2 cdma2000 networks

This annex is intended to define the OSA API Stage 3 interface definitions and it provides the complete OSA specifications. It is an extension of OSA API specifications capabilities to enable operation in cdma2000 systems environment. They are in alignment with 3GPP2 Stage 1 requirements and Stage 2 architecture defined in:

[1] 3GPP2 P.S0001-B: "Wireless IP Network Standard", Version 1.0, September 2000.

[2] 3GPP2 S.R0037-0: "IP Network Architecture Model for cdma2000 Spread Spectrum Systems", Version 2.0, May 14, 2002.

[3] 3GPP2 X.S0013: "All-IP Core Network Multimedia Domain", December 2003.

These requirements are expressed as additions to and/or exclusions from the 3GPP specification.
The information given here is to be used by developers in 3GPP2 cdma2000 network architecture to interpret the 3GPP OSA specifications.

D.1 General Exceptions

The terms 3GPP and UMTS are not applicable for the cdma2000 family of standards. Nevertheless these terms are used (3GPP TR 21.905) mostly in the broader sense of "3G Wireless System". If not stated otherwise there are no additions or exclusions required.

CAMEL and CAP mappings are not applicable for cdma2000 systems.

D.2 Specific Exceptions

D.2.1 Clause 1: Scope

There are no additions or exclusions.

D.2.2 Clause 2: References

Normative references on 3GPP TS 23.078 and on 3GPP TS 29.078 are not applicable for cdma2000 systems.

D.2.3 Clause 3: Definitions and abbreviations

There are no additions or exclusions.

D.2.4 Clause 4: Call Control SCF

There are no additions or exclusions.

D.2.5 Clause 5: The Service Interface Specifications

There are no additions or exclusions.

D.2.6 Clause 6: Common Call Control Data Types

There are no additions or exclusions.

D.2.7 Annex A (normative): OMG IDL Description of Common Call Control Data Types

There are no additions or exclusions.

D.2.8 Annex B (informative): W3C WSDL Description of Common Call Control Data Types

There are no additions or exclusions.

D.2.9 Annex C (informative): Java™ API Description of the Call Control SCFs

There are no additions or exclusions.

Annex E (informative): Change history

Change history							
Date	**TSG #**	**TSG Doc.**	**CR**	**Rev**	**Subject/Comment**	**Old**	**New**
Mar 2007	CT_35	CP-070047	0018	--	Update document for conversion to Release 7	6.5.1	7.0.0
Dec 2008	CT_42				Upgraded unchanged from Rel-7	7.0.0	8.0.0
2009-12	-	-	-	-	Update to Rel-9 version (MCC)	8.0.0	9.0.0

History

Document history		
V9.0.0	January 2010	Publication
V9.0.0	January 2010	Publication