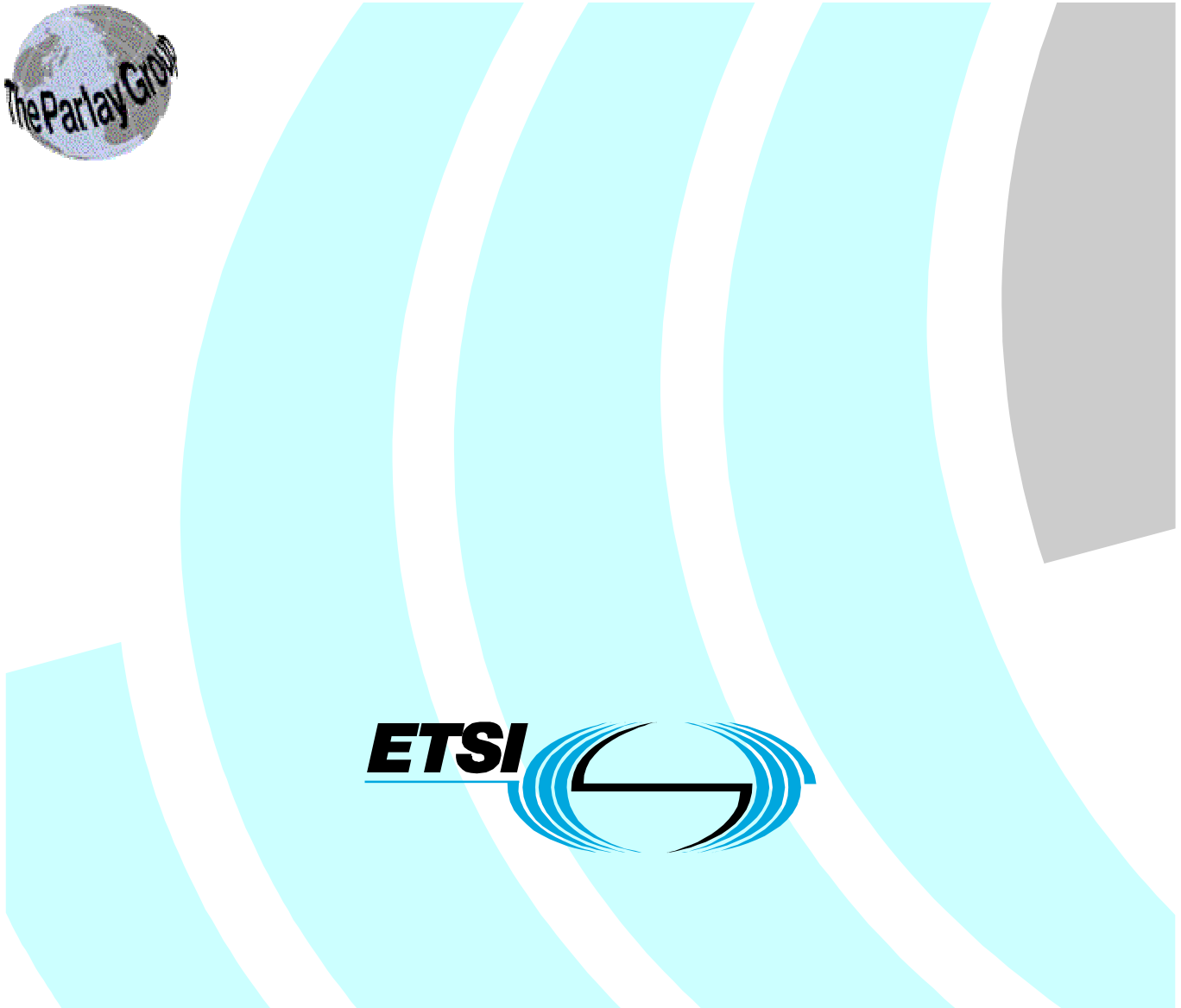
Final draft ETSI ES 201 915-8 V1.6.1 (2006-10)

ETSI Standard

Open Service Access (OSA); Application Programming Interface (API); Part 8: Data Session Control SCF (Parlay 3)



Reference
RES/TISPAN-01027-08-OSA

Keywords
API, IDL, OSA, 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

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 Telecommunications and Internet converged Services and Protocols for Advanced Networking (TISPAN), and is now submitted for the ETSI standards Membership Approval Procedure.

The present document is part 8 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 (http://www.parlay.org)and the 3GPP, in co-operation with a number of JAIN™ Community (http://www.java.sun.com/products/jain) member companies.

The present document forms part of the Parlay 3.5 set of specifications.

The present document is equivalent to 3GPP TS 29.198-8 V4.9.0 (Release 4).

1 Scope

The present document is part 8 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 standardized interface, i.e. the OSA APIs.

The present document specifies the Data Session Control Service Capability Feature (SCF) aspects of the interface. All aspects of the Data Session Control 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 (Parlay 3)".

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.

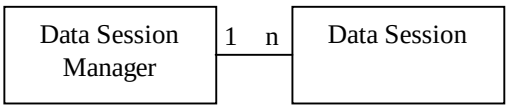
4 Data Session Control SCF

The Data Session control network service capability feature consists of two interfaces:

1) Data Session manager, containing management functions for data session related issues.

2) Data Session, containing methods to control a session.

A session can be controlled by one Data Session Manager only. Data Session Manager can control several sessions.



NOTE: The term "data session" is used in a broad sense to describe a data connection/session. For example, it comprises a PDP context in GPRS.

Figure 1: Data Session control interfaces usage relationship

The Data Session Control service capability features are described in terms of the methods in the Data Session Control interfaces. Table 1 gives an overview of the Data Session Control methods and to which interfaces these methods belong.

Table 1: Overview of Data Session Control interfaces and their methods

Data Session Manager	Data Session
createNotification	connectReq
destroyNotification	connectRes
dataSessionNotificationInterrupted	connectErr
dataSessionNotificationContinued	release
reportNotification	superviseDataSessionReq
dataSessionAborted	superviseDataSessionRes
getNotification	superviseDataSessionErr
changeNotification	dataSessionFaultDetected
	setAdviceofCharge
	setDataSessionChargePlan

The session manager interface provides the management functions to the data session service capability features. The application programmer can use this interface to enable or disable data session-related event notifications.

The following clauses describe each aspect of the Data Session Control Service 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 the present document.

4.1 General requirements on support of methods

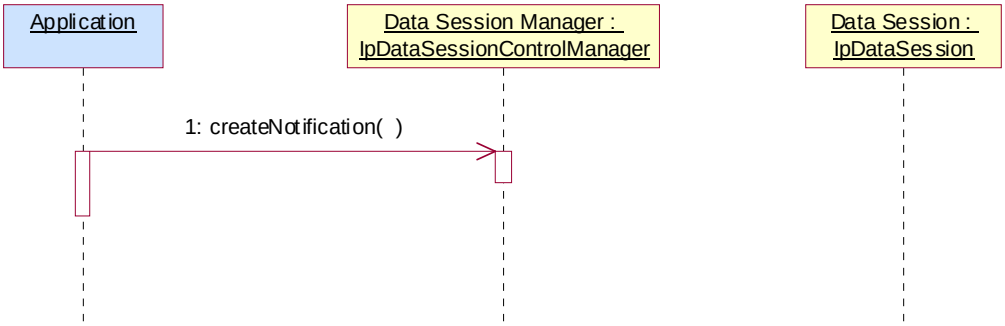
An implementation of this API which supports or implements a method described in the present document, shall support or implement the functionality described for that method, for at least one valid set of values for the parameters of that method.

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.

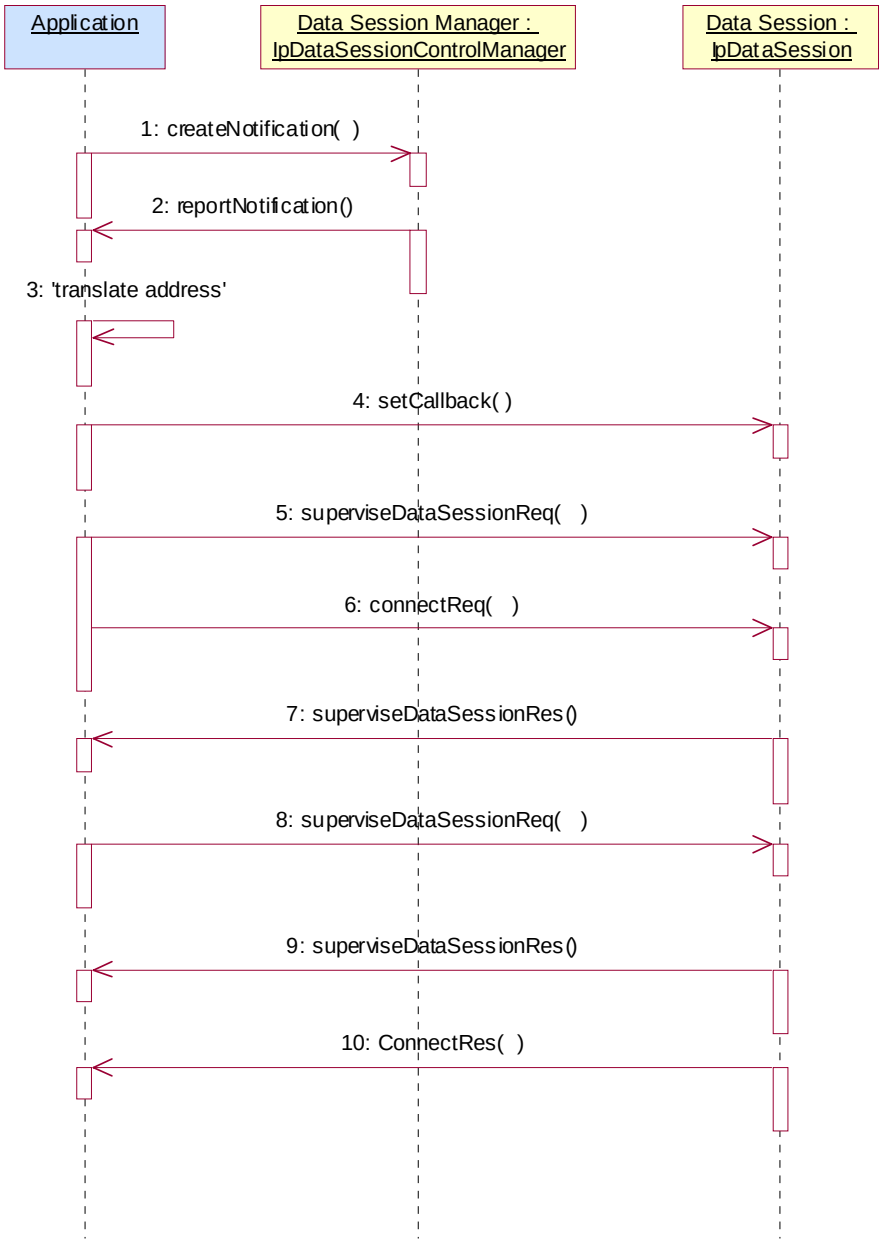
Where a method is not supported by an implementation of an Application interface, a call to that method shall be possible, and no exception shall be returned.

5 Sequence Diagrams

5.1 Enable Data Session Notification



5.2 Address Translation With Charging



6 Class Diagrams

Data Session Control Class Diagram:

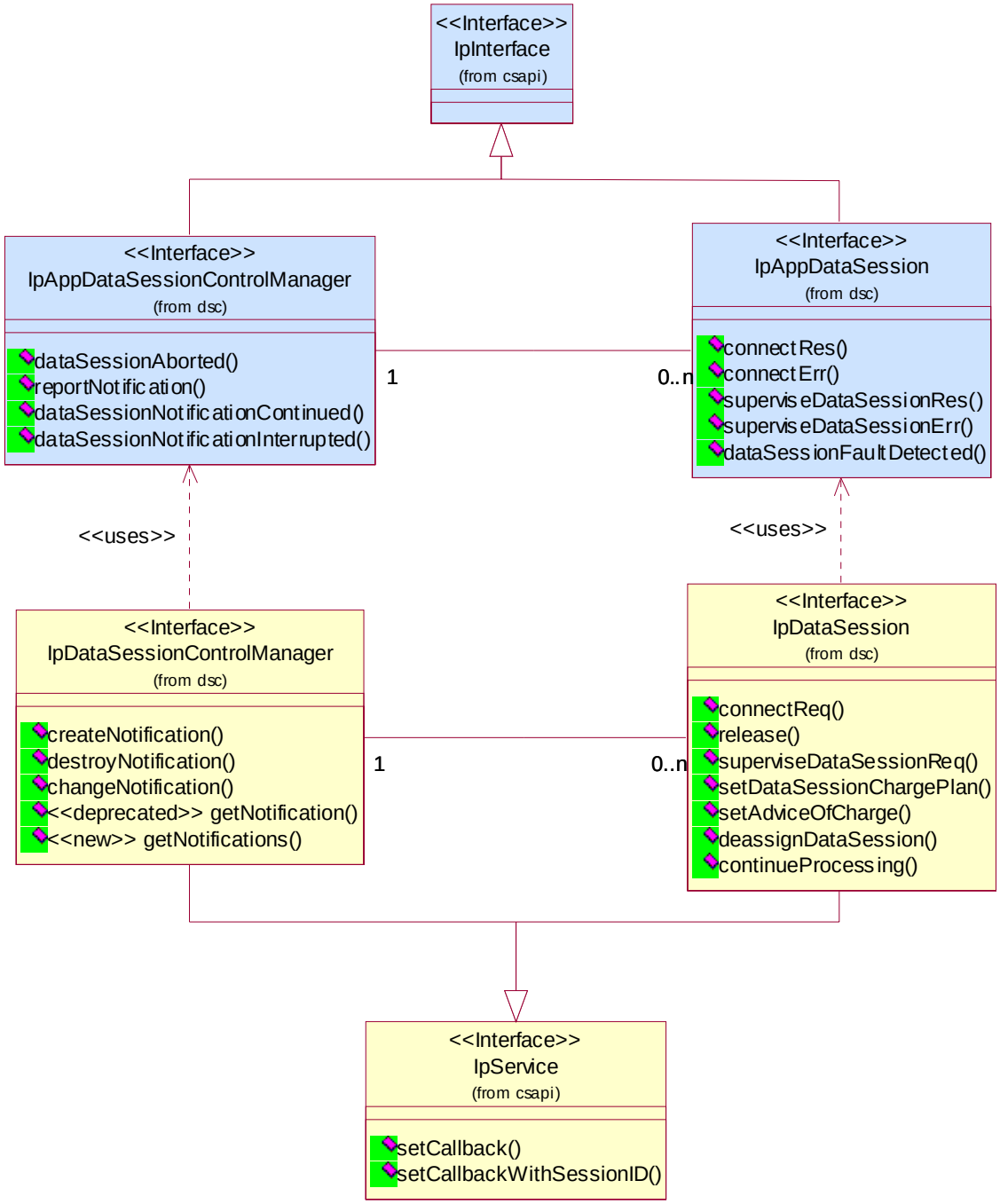


Figure 2: Package Overview

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.

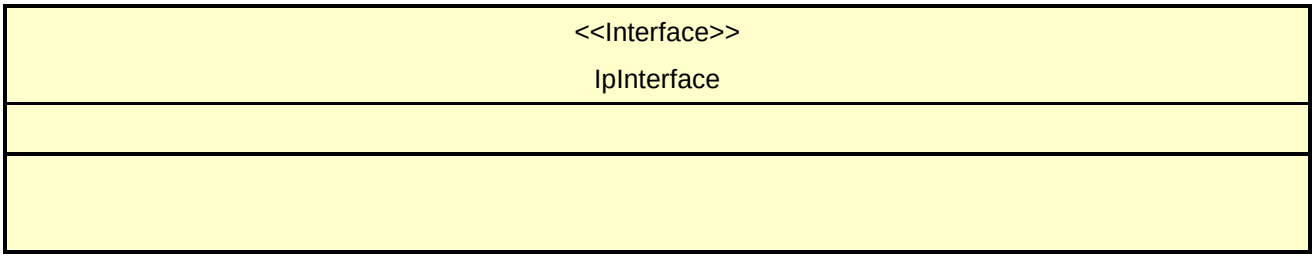
7.1.4 State Model

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.

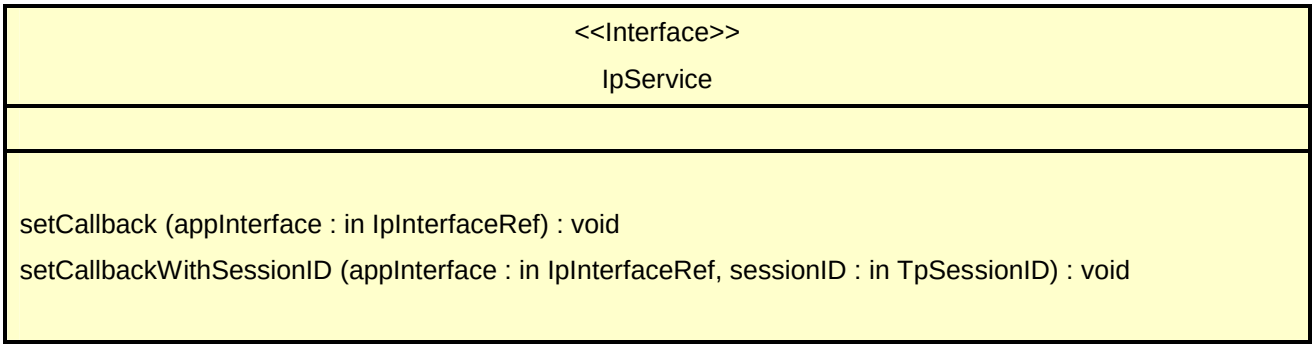
The interfaces that are implemented by the services are denoted as "Service Interface". The corresponding interfaces that must be implemented by the application (e.g. for API callbacks) are denoted as "Application Interface".

7.4 Generic Service Interface

7.4.1 Interface Class IpService

Inherits from: IpInterface.

All service interfaces inherit from the following interface.



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 Data Session Control Interface Classes

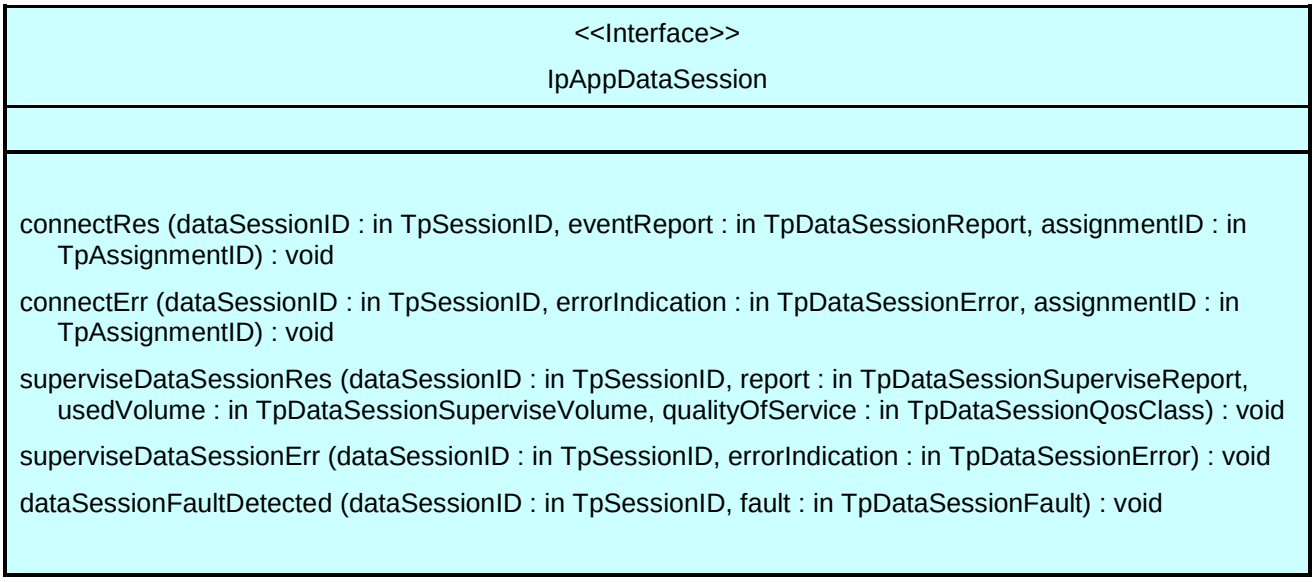
The Data Session Control provides a means to control per data session basis the establishment of a new data session. This means especially in the GPRS context that the establishment of a PDP session is modelled not the attach/detach mode. Change of terminal location is assumed to be managed by the underlying network and is therefore not part of the model. The underlying assumption is that a terminal initiates a data session and the application can reject the request for data session establishment, can continue the establishment or can continue and change the destination as requested by the terminal.

The modelling is similar to the Generic Call Control but assumes a simpler underlying state model. An IpDataSessionControlManager object and an IpDataSession object are the interfaces used by the application, whereas the IpAppDataSessionControlManager and the IpAppDataSession interfaces are implemented by the application.

8.1 Interface Class IpAppDataSession

Inherits from: IpInterface.

The application side of the data session interface is used to handle data session request responses and state reports.



Method

connectRes()

This asynchronous method indicates that the request to connect a data session with the destination party was successful, and indicates the response of the destination party (e.g. connected, disconnected).

Parameters

dataSessionID : in TpSessionID

Specifies the session ID of the data session.

eventReport : in TpDataSessionReport

Specifies the result of the request to connect the data session. It includes the network event, date and time, monitoring mode, negotiated quality of service and event specific information such as release cause.

assignmentID : in TpAssignmentID

Method

connectErr()

This asynchronous method indicates that the request to connect a data session with the destination party was unsuccessful, e.g. an error detected in the network or the data session was abandoned.

Parameters

dataSessionID : in TpSessionID

Specifies the session ID.

errorIndication : in TpDataSessionError

Specifies the error which led to the original request failing.

assignmentID : in TpAssignmentID

Method

superviseDataSessionRes()

This asynchronous method reports a data session supervision event to the application. In addition, it may also be used to notify the application of a newly negotiated set of Quality of Service parameters during the active life of the data session.

Parameters

dataSessionID : in TpSessionID

Specifies the data session.

report : in TpDataSessionSuperviseReport

Specifies the situation, which triggered the sending of the data session supervision response.

usedVolume : in TpDataSessionSuperviseVolume

Specifies the used volume for the data session supervision (in the same unit as specified in the request).

qualityOfService : in TpDataSessionQosClass

Specifies the newly negotiated Quality of Service parameters for the data session.

Method

superviseDataSessionErr()

This asynchronous method reports a data session supervision error to the application.

Parameters

dataSessionID : in TpSessionID

Specifies the data session ID.

errorIndication : in TpDataSessionError

Specifies the error which led to the original request failing.

Method

dataSessionFaultDetected()

This method indicates to the application that a fault in the network has been detected which cannot be communicated by a network event, e.g. when the user aborts before any establishment method is called by the application.

The system purges the Data Session object. Therefore, the application has no further control of data session processing. No report will be forwarded to the application.

Parameters

dataSessionID : in TpSessionID

Specifies the data session ID of the Data Session object in which the fault has been detected.

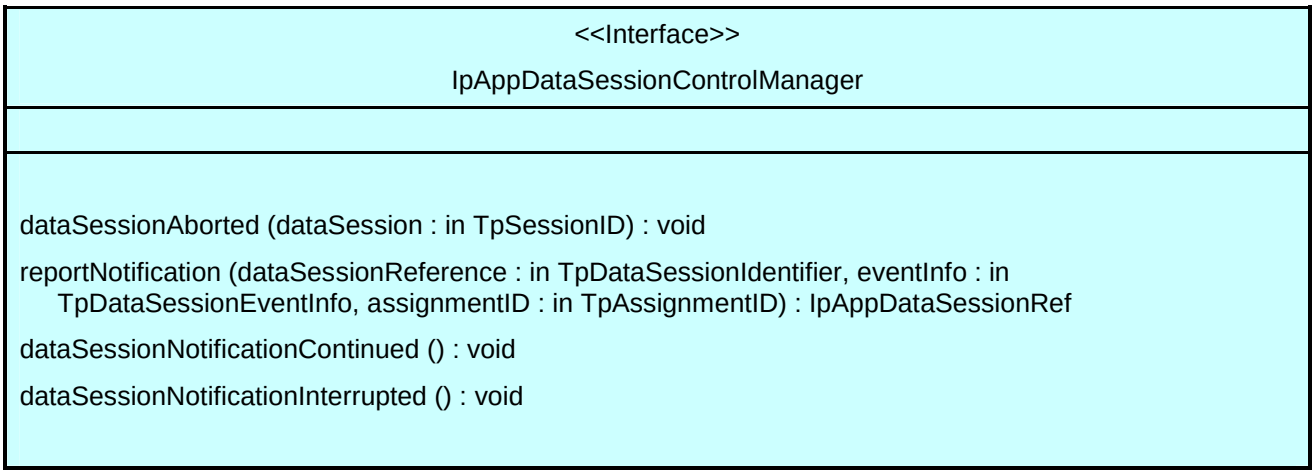
fault : in TpDataSessionFault

Specifies the fault that has been detected.

8.2 Interface Class IpAppDataSessionControlManager

Inherits from: IpInterface.

The data session control manager application interface provides the application data session control management functions to the data session control SCF.



Method

dataSessionAborted()

This method indicates to the application that the Data Session object has aborted or terminated abnormally. No further communication will be possible between the Data Session object and the application.

Parameters

dataSession : in TpSessionID

Specifies the session ID of the data session that has aborted or terminated abnormally.

Method
reportNotification()

This method notifies the application of the arrival of a data session-related event.

Returns appDataSession : Specifies a reference to the application object which implements the callback interface for the new data session. If the application has previously explicitly passed a reference to the IpAppDataSession interface using a setCallbackWithSessionID() invocation, this parameter may be null, or if supplied must be the same as that provided during the setCallbackWithSessionID().

This parameter will be null if the notification is in NOTIFY mode.

Parameters

dataSessionReference : in TpDataSessionIdentifier

Specifies the session ID and the reference to the Data Session object to which the notification relates. If the notification is being given in NOTIFY mode, this parameter shall be ignored by the application client implementation, and consequently the implementation of the SCS entity invoking reportNotification may populate this parameter as it chooses.

eventInfo : in TpDataSessionEventInfo

Specifies data associated with this event. This data includes the destination address provided by the end-user and the quality of service requested or negotiated for the data session.

assignmentID : in TpAssignmentID

Specifies the assignment id which was returned by the createNotification() method. The application can use assignment ID to associate events with event-specific criteria and to act accordingly.

Returns

IpAppDataSessionRef

Method
dataSessionNotificationContinued()

This method indicates to the application that all event notifications are resumed.

Parameters
No Parameters were identified for this method.

Method
dataSessionNotificationInterrupted()

This method indicates to the application that event notifications will no longer be sent (for example, due to faults detected).

Parameters
No Parameters were identified for this method.

8.3 Interface Class IpDataSession

Inherits from: IpService.

The Data Session interface provides basic methods for applications to control data sessions. This interface shall be implemented by a Data Session Control SCF. As a minimum requirement, the connectReq(), release(), deassignDataSession() and continueProcessing() methods shall be implemented.

<<Interface>> IpDataSession
connectReq (dataSessionID : in TpSessionID, responseRequested : in TpDataSessionReportRequestSet, targetAddress : in TpAddress) : TpAssignmentID release (dataSessionID : in TpSessionID, cause : in TpDataSessionReleaseCause) : void superviseDataSessionReq (dataSessionID : in TpSessionID, treatment : in TpDataSessionSuperviseTreatment, bytes : in TpDataSessionSuperviseVolume) : void setDataSessionChargePlan (dataSessionID : in TpSessionID, dataSessionChargePlan : in TpDataSessionChargePlan) : void setAdviceOfCharge (dataSessionID : in TpSessionID, aoCInfo : in TpAoCInfo, tariffSwitch : in TpDuration) : void deassignDataSession (dataSessionID : in TpSessionID) : void continueProcessing (dataSessionID : in TpSessionID) : void

Method

`connectReq()`

This asynchronous method requests the connection of a data session with the destination party (specified in the parameter TargetAddress). The Data Session object is not automatically deleted if the destination party disconnects from the data session.

Returns assignmentID : Specifies the ID assigned to the request. The same ID will be returned in the connectRes or Err. This allows the application to correlate the request and the result.

Parameters

`dataSessionID : in TpSessionID`

Specifies the session ID.

`responseRequested : in TpDataSessionReportRequestSet`

Specifies the set of observed data session events that will result in a connectRes() being generated.

`targetAddress : in TpAddress`

Specifies the address of destination party.

Returns

TpAssignmentID

Raises

TpCommonExceptions, P_SERVICE_INFORMATION_MISSING, P_SERVICE_FAULT_ENCOUNTERED, P_INVALID_NETWORK_STATE, P_INVALID_ADDRESS, P_INVALID_SESSION_ID

Method

release()

This method requests the release of the data session and associated objects.

Parameters

dataSessionID : in TpSessionID

Specifies the session.

cause : in TpDataSessionReleaseCause

Specifies the cause of the release.

Raises

TpCommonExceptions, P_SERVICE_INFORMATION_MISSING, P_SERVICE_FAULT_ENCOUNTERED, P_INVALID_NETWORK_STATE, P_INVALID_SESSION_ID

Method

superviseDataSessionReq()

The application calls this method to supervise a data session. The application can set a granted data volume for this data session. If an application calls this function before it calls a connectReq() or a user interaction function the time measurement will start as soon as the data session is connected. The Data Session object will exist after the data session has been terminated if information is required to be sent to the application at the end of the data session.

Parameters

dataSessionID : in TpSessionID

Specifies the data session.

treatment : in TpDataSessionSuperviseTreatment

Specifies how the network should react after the granted data volume has been sent.

bytes : in TpDataSessionSuperviseVolume

Specifies the granted number of bytes that can be transmitted for the data session.

Raises

TpCommonExceptions, P_SERVICE_INFORMATION_MISSING, P_SERVICE_FAULT_ENCOUNTERED, P_INVALID_NETWORK_STATE, P_INVALID_SESSION_ID

Method

setDataSessionChargePlan()

Allows an application to include charging information in network generated CDR.

Parameters

dataSessionID : in TpSessionID

Specifies the session ID of the data session.

dataSessionChargePlan : in TpDataSessionChargePlan

Specifies the charge plan used.

Raises

TpCommonExceptions, P_SERVICE_INFORMATION_MISSING, P_SERVICE_FAULT_ENCOUNTERED, P_INVALID_NETWORK_STATE, P_INVALID_SESSION_ID

Method

setAdviceOfCharge()

This method allows the application to determine the charging information that will be sent to the end-users terminal.

Parameters

dataSessionID : in TpSessionID

Specifies the session ID of the data session.

aoCInfo : in TpAoCInfo

Specifies two sets of Advice of Charge parameter according to GSM.

tariffSwitch : in TpDuration

Specifies the tariff switch that signifies when the second set of AoC parameters becomes valid.

Raises

TpCommonExceptions, P_SERVICE_INFORMATION_MISSING, P_SERVICE_FAULT_ENCOUNTERED, P_INVALID_NETWORK_STATE, P_INVALID_TIME_AND_DATE_FORMAT

Method

deassignDataSession()

This method requests that the relationship between the application and the data session and associated objects be de-assigned. It leaves the data session in progress, however, it purges the specified data session object so that the application has no further control of data session processing. If a data session is de-assigned that has event reports, data session information reports requested, then these reports will be disabled and any related information discarded.

The application should always either release or deassign the data session when it is finished with the data session, unless dataSessionFaultDetected is received by the application.

Parameters

dataSessionID : in TpSessionID

Specifies the session ID of the data session.

Raises

TpCommonExceptions, P_INVALID_SESSION_ID

Method

continueProcessing()

This operation continues processing of the data session. Applications can invoke this operation after session handling was interrupted due to detection of a notification or event the application subscribed its interest in.

Parameters

dataSessionID : in TpSessionID

Specifies the session ID of the data session.

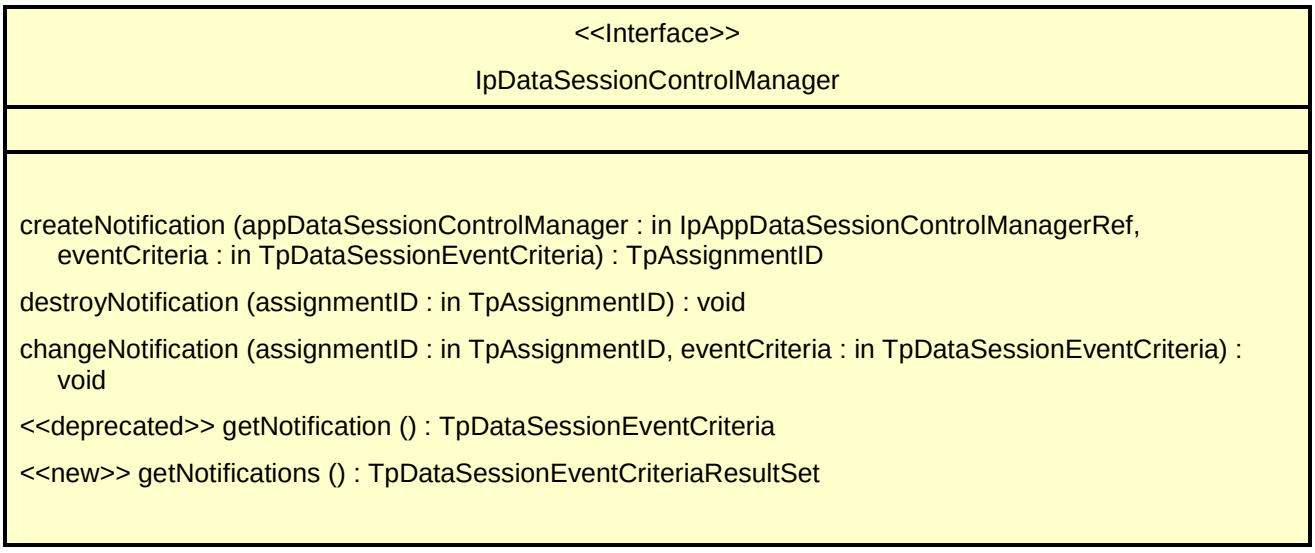
Raises

TpCommonExceptions, P_INVALID_SESSION_ID, P_INVALID_NETWORK_STATE

8.4 Interface Class IpDataSessionControlManager

Inherits from: IpService.

This interface is the "SCF manager" interface for Data Session Control. This interface shall be implemented by a Data Session Control SCF. As a minimum requirement, the createNotification() and destroyNotification() methods shall be implemented.



Method

createNotification()

This method is used to enable data session notifications so that events can be sent to the application. This is the first step an application has to do to get initial notifications of data session happening in the network. When such an event happens, the application will be informed by reportNotification(). In case the application is interested in other events during the context of a particular data session it has to use the connectReq() method on the data session object. The application will get access to the data session object when it receives the reportNotification().

The createNotification method is purely intended for applications to indicate their interest to be notified when certain data session events take place. It is possible to subscribe to a certain event for a whole range of addresses, e.g. the application can indicate it wishes to be informed when a data session is setup to any number starting with 800.

If some application already requested notifications with criteria that overlap the specified criteria, the request is refused with P_INVALID_CRITERIA. The criteria are said to overlap if both originating and terminating ranges overlap and the same number plan is used.

If the same application requests two notifications with exactly the same criteria but different callback references, the second callback will be treated as an additional callback. Both notifications will share the same assignmentID. The gateway will always use the most recent callback. In case this most recent callback fails the second most recent is used. In case the createNotification contains no callback, at the moment the application needs to be informed the gateway will use as callback the callback that has been registered by setCallback().

Returns assignmentID : Specifies the ID assigned by the Data Session Manager object for this newly-enabled event notification.

Parameters

appDataSessionControlManager : in IpAppDataSessionControlManagerRef

If this parameter is set (i.e. not NULL) it specifies a reference to the application interface which is used for callbacks. If set to NULL, the application interface defaults to the interface specified via the setCallback() method.

eventCriteria : in TpDataSessionEventCriteria

Specifies the event specific criteria used by the application to define the event required. Individual addresses or address ranges may be specified for destination and/or origination. Examples of events are "Data Session set up".

Returns

TpAssignmentID

Raises

TpCommonExceptions, P_SERVICE_INFORMATION_MISSING, P_SERVICE_FAULT_ENCOUNTERED, P_INVALID_NETWORK_STATE, P_INVALID_ADDRESS, P_INVALID_CRITERIA, P_INVALID_EVENT_TYPE

Method

destroyNotification()

This method is used by the application to disable data session notifications.

Parameters

assignmentID : in TpAssignmentID

Specifies the assignment ID given by the data session manager object when the previous createNotification() was done.

Raises

TpCommonExceptions, P_SERVICE_INFORMATION_MISSING, P_SERVICE_FAULT_ENCOUNTERED, P_INVALID_NETWORK_STATE, P_INVALID_ASSIGNMENT_ID

Method

changeNotification()

This method is used by the application to change the event criteria introduced with the createNotification method. Any stored notification request associated with the specified assignmentID will be replaced with the specified events requested.

Parameters

assignmentID : in TpAssignmentID

Specifies the ID assigned by the manager interface for the event notification.

eventCriteria : in TpDataSessionEventCriteria

Specifies the new set of event criteria used by the application to define the event required. Only events that meet these criteria are reported.

Raises

TpCommonExceptions, P_SERVICE_INFORMATION_MISSING, P_SERVICE_FAULT_ENCOUNTERED, P_INVALID_NETWORK_STATE, P_INVALID_ASSIGNMENT_ID, P_INVALID_CRITERIA, P_INVALID_EVENT_TYPE

Method

<<deprecated>> getNotification()

This method is deprecated and its use is discouraged. It will be removed in a later release. It is replaced with getNotifications.

This method is used by the application to query the event criteria set with createNotification or changeNotification.

Returns eventCriteria : Specifies the event criteria used by the application to define the event required. Only events that meet these requirements are reported.

Parameters

No Parameters were identified for this method.

Returns

TpDataSessionEventCriteria

Raises

TpCommonExceptions, P_SERVICE_INFORMATION_MISSING, P_SERVICE_FAULT_ENCOUNTERED, P_INVALID_NETWORK_STATE

Method

<<new>> getNotifications()

This method replaces getNotification().

This method is used by the application to query the event criteria set with createNotification or changeNotification.

Returns eventCriteria: the list of event criteria for the notifications requested by the application. If there is no information to return (e.g. no notifications requested by the application), an empty set (zero length) is returned.

Parameters

No Parameters were identified for this method.

Returns

TpDataSessionEventCriteriaResultSet

Raises

TpCommonExceptions, P_INVALID_NETWORK_STATE

9 State Transition Diagrams

9.1 State Transition Diagrams for IpDataSession

The state transition diagram shows the application view on the Data Session object.

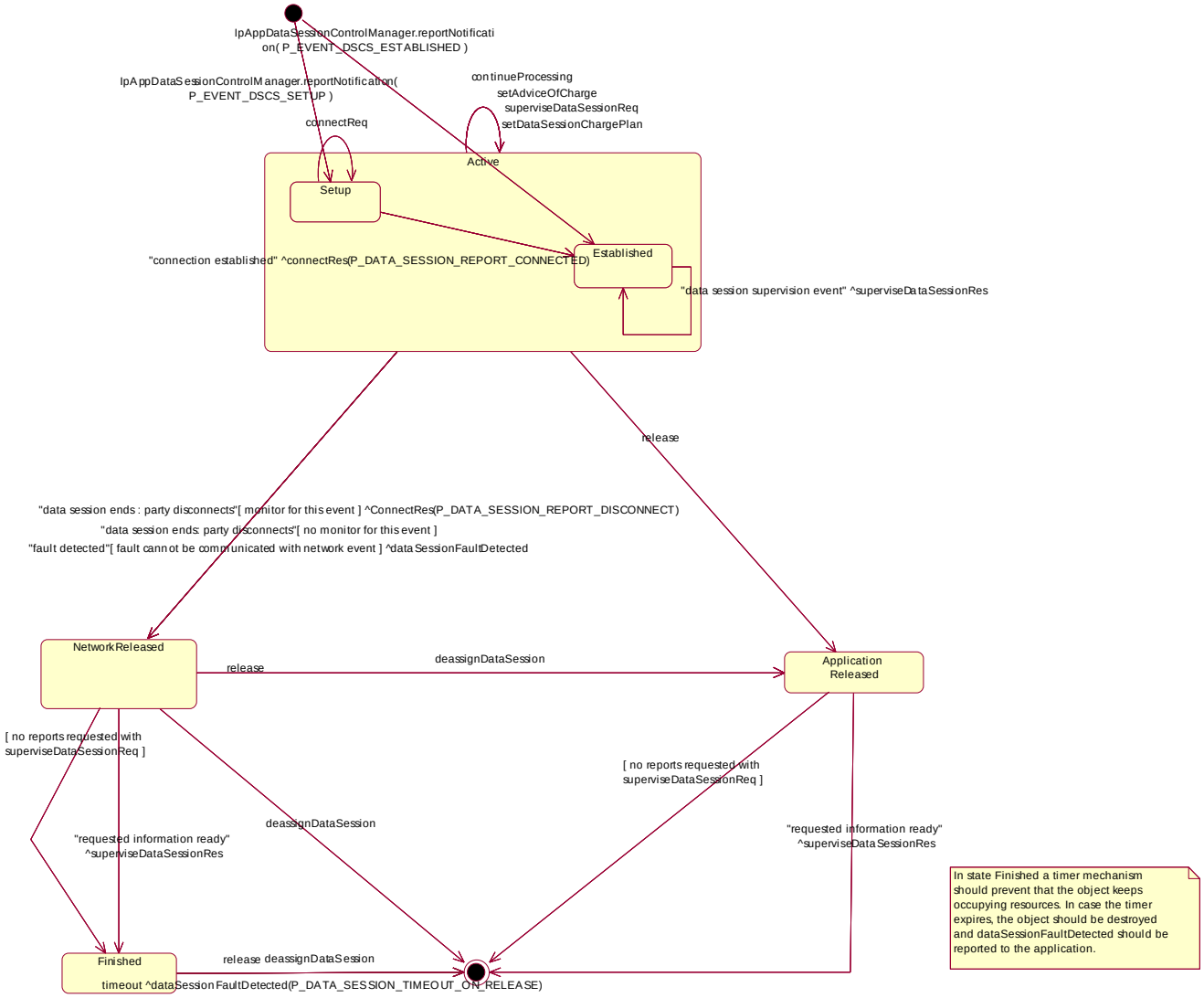


Figure 3: Application view on the Data Session object

9.1.1 Network Released State

In this state the data session has ended. In the case on a normal user disconnection the transition to this state is indicated to the application by the disconnect report of connectRes(). But this will only happen if the application requested monitoring of the disconnect event before. An abnormal disconnection is indicated by dataSessionFaultDetected(). The application may wait for outstanding superviseDataSessionRes().

9.1.2 Finished State

In this state the data session has ended and no further data session related information is to be send to the application. The application can only release the data session object. Calling the deassignDataSession() operation has the same effect. If the application fails to invoke release() within a certain period of time the gateway should automatically release the object and send a timeout indication to the application.

9.1.3 Application Released State

In this state the application has released the data session object. If supervision has been requested the gateway will collect the information and send superviseDataSessionRes() to the application.

9.1.4 Active State

In this state a data connection between two parties is being setup or established (refer to the substates for more details). The application can request the gateway for a certain type of charging by calling setDataSessionChargePlan(), send advice of charge information by calling setAdviceOfCharge(), and request supervision of the data session by calling superviseDataSessionReq().

9.1.5 Setup State

The Setup state is reached after a reportNotification() indicates to the application that a data session is interested in being connected. If the application is going to connect the two parties by invoking connectReq() it may call the charging or supervision methods before.

9.1.6 Established State

In this state the data connection is established. If supervision has been requested the application expects the corresponding superviseDataSessionRes().

10 Data Session Control Service Properties

The following table lists properties relevant for the Data Session Control API.

Property	Type	Description/Interpretation
P_TRIGGERING_EVENT_TYPES	INTEGER_SET	Indicates the static event types supported by the SCS. Static events are the events by which applications are initiated.
P_DYNAMIC_EVENT_TYPES	INTEGER_SET	Indicates the dynamic event types supported by the SCS. Dynamic events are the events the application can request for during the context of a call.
P_ADDRESSPLAN	INTEGER_SET	Indicates the supported address plans (defined in TpAddressPlan.) E.g. P_ADDRESS_PLAN_IP. Note that more than one address plan may be supported.

The previous table lists properties related to the capabilities of the SCS itself. The following table lists properties that are used in the context of the Service Level Agreement, e.g. to restrict the access of applications to the capabilities of the SCS.

Property	Type	Description/Interpretation
P_TRIGGERING_ADDRESSES (Deprecated)	ADDRESSRANGE_SET	Indicates for which numbers the notification may be set. For terminating notifications it applies to the terminating number, for originating notifications it applies only to the originating number.
P_NOTIFICATION_ADDRESS_RANGES	XML_ADDRESS_RANGE_SET	Indicates for which numbers notifications may be set. More than one range may be present. For terminating notifications they apply to the terminating number, for originating notifications they apply only to the originating number.
P_MONITOR_MODE	INTEGER_SET	Indicates whether the application is allowed to monitor in interrupt and/or notify mode. Set is: P_INTERRUPT P_NOTIFY
P_NUMBERS_TO_BE_CHANGED	INTEGER_SET	Indicates which numbers the application is allowed to change or fill for legs in an incoming call. Allowed value set: {P_TARGET_NUMBER}.
P_CHARGEPLAN_ALLOWED	INTEGER_SET	Indicates which charging is allowed in the setDataSessionChargePlan indicator. Allowed values: {P_CHARGE_PER_VOLUME, P_TRANSPARANT_CHARGING, P_CHARGE_PLAN}
P_CHARGEPLAN_MAPPING	INTEGER_INTEGER_MAP	Indicates the mapping of charge plans (we assume they can be indicated with integers) to a logical network charge plan indicator. When the P_CHARGEPLAN_ALLOWED property indicates P_CHARGE_PLAN, then only charge plans in this mapping are allowed.
P_CURRENCY_ALLOWED	STRING_SET	Indicates the currencies that are allowed to be set for the charge plan in the setDataSessionChargePlan. The valid values for the string set are according to ISO-4217. E.g. {"EUR", "NLG"}.

11 Data Definitions

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

11.1 Data Session Control Data Definitions

11.1.1 IpAppDataSession

Defines the address of an `IpAppDataSession` Interface.

11.1.2 IpAppDataSessionRef

Defines a Reference to type `IpAppDataSession`.

11.1.3 IpAppDataSessionControlManager

Defines the address of an `IpAppDataSessionControlManager` Interface.

11.1.4 IpAppDataSessionControlManagerRef

Defines a Reference to type `IpAppDataSessionControlManager`.

11.1.5 IpDataSession

Defines the address of an `IpDataSession` Interface.

11.1.6 IpDataSessionRef

Defines a Reference to type `IpDataSession`.

11.1.7 IpDataSessionControlManager

Defines the address of an `IpDataSessionControlManager` Interface.

11.1.8 IpDataSessionControlManagerRef

Defines a Reference to type `IpDataSessionControlManager`.

11.2 Event Notification data definitions

11.2.1 TpDataSessionEventName

Defines the names of events being notified with a new call request. The following events are supported. The values may be combined by a logical "OR" function when requesting the notifications. Additional events that can be requested/received during the call process are found in the TpDataSessionReportType data-type.

Name	Value	Description
P_EVENT_NAME_UNDEFINED	0	Undefined
P_EVENT_DSCS_SETUP	1	The data session is going to be setup.
P_EVENT_DSCS_ESTABLISHED	2	The data session is established by the network.
P_EVENT_DSCS_QOS_CHANGED	4	A change in QoS class has taken place during the life of the data session.

11.2.2 TpDataSessionMonitorMode

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_DATA_SESSION_MONITOR_MODE_INTERRUPT	0	The data session event is intercepted by the data session control service and data session establishment is interrupted. The application is notified of the event and data session establishment resumes following an appropriate API call or network event (such as a data session release).
P_DATA_SESSION_MONITOR_MODE_NOTIFY	1	The data session event is detected by the data session control service but not intercepted. The application is notified of the event and data session establishment continues.
P_DATA_SESSION_MONITOR_MODE_DO_NOT_MONITOR	2	Do not monitor for the event.

11.2.3 TpDataSessionEventCriteria

Defines the Sequence of Data Elements that specify the criteria for a event notification.

Of the addresses only the Plan and the AddrString are used for the purpose of matching the notifications against the criteria.

Sequence Element Name	Sequence Element Type	Description
DestinationAddress	TpAddressRange	Defines the destination address or address range for which the notification is requested.
OriginationAddress	TpAddressRange	Defines the origination address or a address range for which the notification is requested.
DataSessionEventName	TpDataSessionEventName	Name of the event(s).
MonitorMode	TpDataSessionMonitorMode	Defines the mode that the Data Session is in following the notification. Monitor mode P_DATA_SESSION_MONITOR_MODE_DO_NOT_MONITOR is not a legal value here.

11.2.4 TpDataSessionEventInfo

Defines the Sequence of Data Elements that specify the information returned to the application in a Data Session event notification.

Sequence Element Name	Sequence Element Type	Description
DestinationAddress	TpAddress	Defines the destination address for which the notification is reported.
OriginatingAddress	TpAddress	Defines the origination address for which the notification is reported.
DataSessionEventName	TpDataSessionEventName	Name of the event(s).
MonitorMode	TpDataSessionMonitorMode	Defines the mode in which the Data Session is reporting the notification. Monitor mode P_DATA_SESSION_MONITOR_MODE_DO_NOT_MONITOR is not a legal value here.
QoSClass	TpDataSessionQosClass	Defines the Quality of Service (QoS) class for the Data Session. QoSClass NULL is not a legal value when DataSessionEventName is set to P_EVENT_DSCS_QOS_CHANGED. For this particular event, the QoSClass defines the new QoS class effective after the change.

11.2.5 TpDataSessionQosClass

Defines the Quality of Service (QoS) classes for a data session.

Name	Value	Description
P_DATA_SESSION_QOS_CLASS_CONVERSATIONAL	0	Specifies the Conversational QoS class, as specified in TS 123 107.
P_DATA_SESSION_QOS_CLASS_STREAMING	1	Specifies the Streaming QoS class, as specified in TS 123 107.
P_DATA_SESSION_QOS_CLASS_INTERACTIVE	2	Specifies the Interactive QoS class, as specified in TS 123 107.
P_DATA_SESSION_QOS_CLASS_BACKGROUND	3	Specifies the Background QoS class, as specified in TS 123 107.

11.2.6 TpDataSessionChargePlan

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

Sequence Element Name	Sequence Element Type	Description
ChargeOrderType	TpDataSessionChargeOrder	Charge order.
Currency	TpString	Currency unit according to ISO-4217.
AdditionalInfo	TpString	Descriptive string which is sent to the billing system without prior evaluation. Could be included in the ticket.

Valid Currencies are:

ADP, AED, AFA, ALL, AMD, ANG, AON, AOR, ARS, ATS, AUD, AWG, AZM, BAM,

BBD, BDT, BEF, BGL, BGN, BHD, BIF, BMD, BND, BOB, BOV, BRL, BSD, BTN,

BWP, BYB, BZD, CAD, CDF, CHF, CLF, CLP, CNY, COP, CRC, CUP, CVE, CYP,

CZK, DEM, DJF, DKK, DOP, DZD, ECS, ECV, EEK, EGP, ERN, ESP, ETB, EUR,

FIM, FJD, FKP, FRF, GBP, GEL, GHC, GIP, GMD, GNF, GRD, GTQ, GWP, GYD,

HKD, HNL, HRK, HTG, HUF, IDR, IEP, ILS, INR, IQD, IRR, ISK, ITL, JMD,

JOD, JPY, KES, KGS, KHR, KMF, KPW, KRW, KWD, KYD, KZT, LAK, LBP, LKR,

LRD, LSL, LTL, LUF, LVL, LYD, MAD, MDL, MGF, MKD, MMK, MNT, MOP, MRO,

MTL, MUR, MVR, MWK, MXN, MXV, MYR, MZM, NAD, NGN, NIO, NLG, NOK, NPR,

NZD, OMR, PAB, PEN, PGK, PHP, PKR, PLN, PTE, PYG, QAR, ROL, RUB, RUR,

RWF, SAR, SBD, SCR, SDD, SEK, SGD, SHP, SIT, SKK, SLL, SOS, SRG, STD,

SVC, SYP, SZL, THB, TJR, TMM, TND, TOP, TPE, TRL, TTD, TWD, TZS, UAH,

UGX, USD, USN, USS, UYU, UZS, VEB, VND, VUV, WST, XAF, XAG, XAU, XBA,

XBB, XBC, XBD, XCD, XDR, XFO, XFU, XOF, XPD, XPF, XPT, XTS, XXX, YER,

YUM, ZAL, ZAR, ZMK, ZRN, ZWD.

XXX is used for transactions where no currency is involved.

11.2.7 TpDataSessionChargeOrder

Defines the Tagged Choice of Data Elements that specify the charge plan for the call.

	Tag Element Type	
	TpDataSessionChargeOrderCategory	

Tag Element Value	Choice Element Type	Choice Element Name
P_DATA_SESSION_CHARGE_PER_VOLUME	TpChargePerVolume	ChargePerVolume
P_DATA_SESSION_CHARGE_NETWORK	TpString	NetworkCharge

11.2.8 TpDataSessionChargeOrderCategory

Name	Value	Description
P_DATA_SESSION_CHARGE_PER_VOLUME	0	Charge per volume.
P_DATA_SESSION_CHARGE_NETWORK	1	Operator specific charge plan specification, e.g. charging table name / charging table entry.

11.2.9 TpChargePerVolume

Defines the Sequence of Data Elements that specify the time based charging information. The volume is the sum of uplink and downlink transfer data volumes.

Sequence Element Name	Sequence Element Type	Description
InitialCharge	TpInt32	Initial charge amount (in currency units * 0,0001).
CurrentChargePerKilobyte	TpInt32	Current tariff (in currency units * 0,0001).
NextChargePerKilobyte	TpInt32	Next tariff (in currency units * 0,0001) after tariff switch. Only used in setAdviceOfCharge().

11.2.10 TpDataSessionIdentifier

Defines the Sequence of Data Elements that unambiguously specify the Data Session object

Sequence Element Name	Sequence Element Type	Sequence Element Description
DataSessionReference	IpDataSessionRef	This element specifies the interface reference for the Data Session object.
DataSessionID	TpSessionID	This element specifies the data session ID of the Data Session.

11.2.11 TpDataSessionError

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	TpDataSessionErrorType
AdditionalErrorInfo	TpDataSessionAdditionalErrorInfo

11.2.12 TpDataSessionAdditionalErrorInfo

Defines the Tagged Choice of Data Elements that specify additional Data Session error and Data Session error specific information.

	Tag Element Type	
	TpDataSessionErrorType	

Tag Element Value	Choice Element Type	Choice Element Name
P_DATA_SESSION_ERROR_UNDEFINED	NULL	Undefined
P_DATA_SESSION_ERROR_INVALID_ADDRESS	TpAddressError	DataSessionErrorInvalidAddress
P_DATA_SESSION_ERROR_INVALID_STATE	NULL	Undefined

11.2.13 TpDataSessionErrorType

Defines a specific Data Session error.

Name	Value	Description
P_DATA_SESSION_ERROR_UNDEFINED	0	Undefined; the method failed or was refused, but no specific reason can be given.
P_DATA_SESSION_ERROR_INVALID_ADDRESS	1	The operation failed because an invalid address was given.
P_DATA_SESSION_ERROR_INVALID_STATE	2	The data session was not in a valid state for the requested operation.

11.2.14 TpDataSessionFault

Defines the cause of the data session fault detected.

Name	Value	Description
P_DATA_SESSION_FAULT_UNDEFINED	0	Undefined
P_DATA_SESSION_FAULT_USER_ABORTED	1	User has finalized the data session before any message could be sent by the application.
P_DATA_SESSION_TIMEOUT_ON_RELEASE	2	This fault occurs when the final report has been sent to the application, but the application did not explicitly release data session object, within a specified time. The timer value is operator specific.
P_DATA_SESSION_TIMEOUT_ON_INTERRUPT	3	This fault occurs when the application did not instruct the gateway how to handle the call within a specified time, after the gateway reported an event that was requested by the application in interrupt mode. The timer value is operator specific.

11.2.15 TpDataSessionReleaseCause

Defines the Sequence of Data Elements that specify the cause of the release of a data session.

Sequence Element Name	Sequence Element Type
Value	TpInt32
Location	TpInt32
NOTE: The Value and Location are specified as in ITU-T Recommendation Q.850.	

11.2.16 TpDataSessionSuperviseVolume

Defines the Sequence of Data Elements that specify the amount of volume that is allowed to be transmitted for the specific connection.

Sequence Element Name	Sequence Element Type	Sequence Element Description
VolumeQuantity	TpInt32	This data type is identical to a TpInt32, and defines the quantity of the granted volume that can be transmitted for the specific connection. The volume specifies the sum of uplink and downlink transfer data volumes.
VolumeUnit	TpInt32	In Order to enlarge the range of the volume quantity value the exponent of a scaling factor (10^VolumeUnit) is provided. When the unit is for example in kilobytes, VolumeUnit shall be set to 3.

11.2.17 TpDataSessionSuperviseReport

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

Name	Value	Description
P_DATA_SESSION_SUPERVISE_VOLUME_REACHED	01h	The maximum volume has been reached.
P_DATA_SESSION_SUPERVISE_DATA_SESSION_ENDED	02h	The data session has ended, either due to data session party to reach of maximum volume or calling or called release.
P_DATA_SESSION_SUPERVISE_MESSAGE_SENT	04h	A warning message has been sent.

11.2.18 TpDataSessionSuperviseTreatment

Defines the treatment of the call by the data session control service when the supervised volume is reached. The values may be combined by a logical "OR" function.

Name	Value	Description
P_DATA_SESSION_SUPERVISE_RELEASE	01h	Release the data session when the data session supervision volume is reached.
P_DATA_SESSION_SUPERVISE_RESPOND	02h	Notify the application when the call supervision volume is reached.
P_DATA_SESSION_SUPERVISE_INFORM	04h	Send a warning message to the originating party when the maximum volume is reached. If data session release is requested, then the data session will be released following the message after an administered time period.

11.2.19 TpDataSessionReport

Defines the Sequence of Data Elements that specify the data session report specific information.

Sequence Element Name	Sequence Element Type
MonitorMode	TpDataSessionMonitorMode
DataSessionEventTime	TpDateAndTime
DataSessionReportType	TpDataSessionReportType
AdditionalReportInfo	TpDataSessionAdditionalReportInfo

11.2.20 TpDataSessionAdditionalReportInfo

Defines the Tagged Choice of Data Elements that specify additional data session report information for certain types of reports.

	Tag Element Type	
	TpDataSessionReportType	

Tag Element Value	Choice Element Type	Choice Element Name
P_DATA_SESSION_REPORT_UNDEFINED	NULL	Undefined
P_DATA_SESSION_REPORT_CONNECTED	NULL	Undefined
P_DATA_SESSION_REPORT_DISCONNECT	TpDataSessionReleaseCause	DataSessionDisconnect

11.2.21 TpDataSessionReportRequest

Defines the Sequence of Data Elements that specify the criteria relating to data session report requests.

Sequence Element Name	Sequence Element Type
MonitorMode	TpDataSessionMonitorMode
DataSessionReportType	TpDataSessionReportType

11.2.22 TpDataSessionReportRequestSet

Defines a Numbered Set of Data Elements of TpDataSessionReportRequest.

11.2.23 TpDataSessionReportType

Defines a specific data session event report type.

Name	Value	Description
P_DATA_SESSION_REPORT_UNDEFINED	0	Undefined
P_DATA_SESSION_REPORT_CONNECTED	1	Data session established.
P_DATA_SESSION_REPORT_DISCONNECT	2	Data session disconnect requested by data session party

11.2.24 TpDataSessionEventCriteriaResult

Defines a sequence of data elements that specify a requested data session event notification criteria with the associated assignmentID.

Sequence Element Name	Sequence Element Type	Sequence Element Description
EventCriteria	TpDataSessionEventCriteria	The event criteria that were specified by the application.
AssignmentID	TpAssignmentID	The associated assignmentID. This can be used to disable the notification.

11.2.25 TpDataSessionEventCriteriaResultSet

Defines a set of TpDataSessionEventCriteriaResult.

12 Exception Classes

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

Name	Description
P_SERVICE_INFORMATION_MISSING	Information relating to the Data Session Control SCF could not be found
P_SERVICE_FAULT_ENCOUNTERED	Fault detected in the Data Session Control SCF

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 Data Session Control SCF

The OMG IDL representation of this interface specification is contained in a text file (dsc.idl contained in archive es_20191508v010601m0.zip) which accompanies the present document.

Annex B (informative): Contents of 3GPP OSA R4 Data Session Control

All of the present document is relevant for TS 129 198-8 V4 (Release 4).

Annex C (informative): Record of changes

The following is a list of the changes made to the present document for each release. The list contains the names of all changed, deprecated, added or removed items in the specifications and not the actual changes. Any type of change information that is important to the reader is put in the *Others* part of this annex.

Changes are specified as changes to the prior major release, but every minor release will have its own part of the table allowing the reader to know when the actual change was made.

C.1 Interfaces

C.1.1 New

Identifier	Comments
Interfaces added in ES 201 915-8 version 1.4.1 (Parlay 3.3)	
Interfaces added in ES 201 915-8 version 1.5.1 (Parlay 3.4)	
Interfaces added in ES 201 915-8 version 1.6.1 (Parlay 3.5)	

C.1.2 Deprecated

Identifier	Comments
Interfaces deprecated in ES 201 915-8 version 1.4.1 (Parlay 3.3)	
Interfaces deprecated in ES 201 915-8 version 1.5.1 (Parlay 3.4)	
Interfaces deprecated in ES 201 915-8 version 1.6.1 (Parlay 3.5)	

C.1.3 Removed

Identifier	Comments
Interfaces removed in ES 201 915-8 version 1.4.1 (Parlay 3.3)	
Interfaces removed in ES 201 915-8 version 1.5.1 (Parlay 3.4)	
Interfaces removed in ES 201 915-8 version 1.6.1 (Parlay 3.5)	

C.2 Methods

C.2.1 New

Identifier	Comments
Methods added in ES 201 915-8 version 1.4.1 (Parlay 3.3)	
IpDataSessionControlManager.getNotifications	replaces getNotification
Methods added in ES 201 915-8 version 1.5.1 (Parlay 3.4)	
Methods added in ES 201 915-8 version 1.6.1 (Parlay 3.5)	

C.2.2 Deprecated

Identifier	Comments
Methods deprecated in ES 201 915-8 version 1.4.1 (Parlay 3.3)	
IpDataSessionControlManager.getNotification	replaced by getNotifications
Methods deprecated in ES 201 915-8 version 1.5.1 (Parlay 3.4)	
Methods deprecated in ES 201 915-8 version 1.6.1 (Parlay 3.5)	

C.2.3 Modified

Identifier	Comments
Methods modified in ES 201 915-8 version 1.4.1 (Parlay 3.3)	
Methods modified in ES 201 915-8 version 1.5.1 (Parlay 3.4)	
Methods modified in ES 201 915-8 version 1.6.1 (Parlay 3.5)	

C.2.4 Removed

Identifier	Comments
Methods removed in ES 201 915-8 version 1.4.1 (Parlay 3.3)	
Methods removed in ES 201 915-8 version 1.5.1 (Parlay 3.4)	
Methods removed in ES 201 915-8 version 1.6.1 (Parlay 3.5)	

C.3 Data Definitions

C.3.1 New

Identifier	Comments
Data Definitions added in ES 201 915-8 version 1.4.1 (Parlay 3.3)	
TpDataSessionEventCriteriaResult	
TpDataSessionEventCriteriaResultSet	
P_EVENT_DSCS_QOS_CHANGED	missing from IDL, present in Document.
Data Definitions added in ES 201 915-8 version 1.5.1 (Parlay 3.4)	
Data Definitions added in ES 201 915-8 version 1.6.1 (Parlay 3.5)	

C.3.2 Modified

Identifier	Comments
Data Definitions modified in ES 201 915-8 version 1.4.1 (Parlay 3.3)	
TpDataSessionFault	P_DATA_SESSION_USER_ABORTED corrected to P_DATA_SESSION_FAULT_USER_ABORTED (document only)
TpDataSessionEventCriteria	OriginatingAddress changed to OriginationAddress to match IDL
Data Definitions modified in ES 201 915-8 version 1.5.1 (Parlay 3.4)	
Data Definitions modified in ES 201 915-8 version 1.6.1 (Parlay 3.5)	

C.3.3 Removed

Identifier	Comments
Data Definitions removed in ES 201 915-8 version 1.4.1 (Parlay 3.3)	
Data Definitions removed in ES 201 915-8 version 1.5.1 (Parlay 3.4)	
Data Definitions removed in ES 201 915-8 version 1.6.1 (Parlay 3.5)	

C.4 Service Properties

C.4.1 New

Identifier	Comments
Service Properties added in ES 201 915-8 version 1.4.1 (Parlay 3.3)	
Service Properties added in ES 201 915-8 version 1.5.1 (Parlay 3.4)	
P_NOTIFICATION_ADDRESS_RANGES	Replaces P_TRIGGERING_ADDRESSES
Service Properties added in ES 201 915-8 version 1.6.1 (Parlay 3.5)	

C.4.2 Deprecated

Identifier	Comments
Service Properties deprecated in ES 201 915-8 version 1.4.1 (Parlay 3.3)	
Service Properties deprecated in ES 201 915-8 version 1.5.1 (Parlay 3.4)	
P_TRIGGERING_ADDRESSES	Replaced by P_NOTIFICATION_ADDRESS_RANGES
Service Properties deprecated in ES 201 915-8 version 1.6.1 (Parlay 3.5)	

C.4.3 Modified

Identifier	Comments
Service Properties modified in ES 201 915-8 version 1.4.1 (Parlay 3.3)	
Service Properties modified in ES 201 915-8 version 1.5.1 (Parlay 3.4)	
Service Properties modified in ES 201 915-8 version 1.6.1 (Parlay 3.5)	

C.4.4 Removed

Identifier	Comments
Service Properties removed in ES 201 915-8 version 1.4.1 (Parlay 3.3)	
Service Properties removed in ES 201 915-8 version 1.5.1 (Parlay 3.4)	
Service Properties removed in ES 201 915-8 version 1.6.1 (Parlay 3.5)	

C.5 Exceptions

C.5.1 New

Identifier	Comments
Exceptions added in ES 201 915-8 version 1.4.1 (Parlay 3.3)	
Exceptions added in ES 201 915-8 version 1.5.1 (Parlay 3.4)	
Exceptions added in ES 201 915-8 version 1.6.1 (Parlay 3.5)	

C.5.2 Modified

Identifier	Comments
Exceptions modified in ES 201 915-8 version 1.4.1 (Parlay 3.3)	
Exceptions modified in ES 201 915-8 version 1.5.1 (Parlay 3.4)	
Exceptions modified in ES 201 915-8 version 1.6.1 (Parlay 3.5)	

C.5.3 Removed

Identifier	Comments
Exceptions removed in ES 201 915-8 version 1.4.1 (Parlay 3.3)	
Exceptions removed in ES 201 915-8 version 1.5.1 (Parlay 3.4)	
Exceptions removed in ES 201 915-8 version 1.6.1 (Parlay 3.5)	

C.6 Others

None.

History

Document history		
V1.1.1	February 2002	Publication
V1.2.1	July 2002	Publication
V1.3.1	October 2002	Publication
V1.4.1	July 2003	Publication
V1.5.1	February 2005	Publication
V1.6.1	October 2006	Membership Approval Procedure MV 20061208: 2006-10-10 to 2006-12-08