3GPP TSG CN Plenary Meeting #19 **NP-030034**
12- 14 March 2003, Birmingham, UK

Source: **CN5 (OSA)**

Title: **Rel-5 CRs 29.198-08 OSA API Part 8: Data session control**

Agenda item: **8.2**

Document for: **APPROVAL**

Doc-1st-Level	Spec	CR	Rev	Phase	Subject	Cat	Version-Current	Doc-2nd-Level	Workitem
NP-030034	29.198-08	022	-	Rel-5	Inconsistent description of use of secondary callback	F	5.1.0	N5-021039	OSA2
NP-030034	29.198-08	023	-	Rel-5	Promotion of TpDataSessionQosClass data type definition to the Common Data Types	F	5.1.0	N5-021116	OSA2

joint API group (Parlay, ETSI Project OSA, 3GPP TSG_CN WG5) **N5-021039**
Meeting #21, Dublin, IRELAND, 28 – 31 October 2002

CR-Form-v5

CHANGE REQUEST

⌘ **29.198-08** **CR** **022** ⌘ **rev** **-** ⌘ Current version: **5.1.0** ⌘

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

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

Title:	⌘	Inconsistent description of use of secondary callback
Source:	⌘	N5
Work item code:	⌘	OSA2
Date:	⌘	10/10/2002
Category:	⌘	**F**
Release:	⌘	*REL-5*

Use one of the following categories:
F *(correction)*
A *(corresponds to a correction in an earlier release)*
B *(addition of feature),*
C *(functional modification of feature)*
D *(editorial modification)*
Detailed explanations of the above categories can be found in 3GPP TR 21.900.

Use one of the following releases:

2	*(GSM Phase 2)*
R96	*(Release 1996)*
R97	*(Release 1997)*
R98	*(Release 1998)*
R99	*(Release 1999)*
REL-4	*(Release 4)*
REL-5	*(Release 5)*

Reason for change:	⌘	OSA Specification describes use of secondary callback interface inconsistently between the different parts which confuses application developers.
Summary of change:	⌘	Describe that most recent call back will be used as the callback interface. Only if this one does not work, the initially provided callback interface is used.
Consequences if not approved:	⌘	Interoperability problems.

Clauses affected:	⌘			
Other specs affected:	⌘	☐ Other core specifications ☐ Test specifications ☐ O&M Specifications	⌘	
Other comments:	⌘			

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

1) Fill out the above form. The symbols above marked ⌘ contain pop-up help information about the field that they are closest to.
2) Obtain the latest version for the release of the specification to which the change is proposed. Use the MS Word "revision marks" feature (also known as "track changes") when making the changes. All 3GPP specifications can be downloaded from the 3GPP server under ftp://ftp.3gpp.org/specs/ For the latest version, look for the directory name with the latest date e.g. 2001-03 contains the specifications resulting from the March 2001 TSG meetings.
3) With "track changes" disabled, paste the entire CR form (use CTRL-A to select it) into the specification just in front of the clause containing the first piece of changed text. Delete those parts of the specification which are not relevant to the change request.

Introduction

The OSA Specifications contains the following descriptions about the use of a secondary callback interface:

Part 1:

7.12 Notification Handling

It is possible to recreate a (set of) notification(s) or re-register for notifications. This is only useful when providing a different callback interface reference. In this case, the last provided interface is used for reporting notifications. ***The earlier provided callback interface is used as "backup" interface (this can be the one provided with setCallback() or setCallbackWithSessionID() if NULL was provided initially). Notifications are reported on this interface when calls to the most recent provided callback interface fail (object providing the interface is crashed or overloaded).*** When re-creating or re-registering, the same assignment ID is returned.

Part 4-2:

6.1.2 Method enableCallNotification()

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 enableCallNotification 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().

Part 4-3:

6.1.2 Method createNotification()

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().

6.1.7 Method <<new>> enableNotifications()

If the same application requests to enable notifications for a second time with a different IpAppMultiPartyCallControlManager reference (i.e. without first disabling them***), the second callback will be treated as an additional callback. This means that the callback will only be used in cases when the first callback specified by the application is unable to handle the callEventNotify (e.g. due to overload or failure).***

Part 4-4:

6.1.1 Method createMediaNotification()

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 createMediaNotification contains no callback, at the moment the application needs to be informed the gateway will use as callback the one that has been registered by setCallback().

Part 5:

8.1.3 Method createNotification()

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. This means that the callback will only be used in case when the first callback specified by the application is unable to handle the reportNotification (e.g., due to overload or failure).***

8.1.7 Method <<new>> enableNotifications()

If the same application requests to enable notifications for a second time with a different IpAppUIManager reference (i.e. without first disabling them***), the second callback will be treated as an additional callback. This means that the callback wil only be used in cases when the first callback specified by the application is unable to handle the callEventNotify (e.g. due to overload or failure).***

Part 8:

8.4.1 Method <<deprecated>> createNotification()

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().

8.4.5 Method <<new>> enableNotifications()

If the same application requests to enable notifications for a second time with a different IpAppDataSessionControlManager reference (i.e. without first disabling them), ***the second callback will be treated as an additional callback. This means that the callback wil only be used in cases when the first callback specified by the application is unable to handle the callEventNotify (e.g. due to overload or failure).***

8.4.8 Method <<new>> createNotifications()

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().

Part 11:

8.1.1 Method createNotification()

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 enableCallNotification 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().

8.1.7 Method <<new>> enableNotifications()

If the same application requests to enable notifications for a second time with a different IpAppAccountManager reference (i.e. without first disabling them), ***the second callback will be treated as an additional callback. This means that the callback will only be used in cases when the first callback specified by the application is unable to handle the reportNotification (e.g. due to overload or failure).***

Solution

The intended use of the 2nd callback interface is as described in part 1, therfore the changes to the following method descriptions are proposed:

- Part 4-3, method enableNotifications()
- Part 5, method createNotification()
- Part 5, method enableNotifications()
- Part 8, method enableNotifications()
- Part 11, method enableNotifications()

This contribution proposes the changes for Part 8.

Proposed Changes

8.4.5 Method <<new>> enableNotifications()

This method is used to indicate that the application is able to receive which are provisioned from within the network (i.e. these notifications are NOT set using createNotification() but via, for instance, a network management system). If notifications provisioned for this application are created or changed, the application is unaware of this until the notification is reported.

If the same application requests to enable notifications for a second time with a different IpAppDataSessionControlManager reference (i.e. without first disabling them), the second callback will be treated as an additional callback. The gateway will always use the most recent callback. In case this most recent callback fails the second most recent is used.~~This means that the callback wil only be used in cases when the first callback specified by the application is unable to handle the callEventNotify (e.g. due to overload or failure).~~

When this method is used, it is still possible to use createNotification() for service provider provisioned notifications on the same interface as long as the criteria in the network and provided by createNotification() do not overlap. However, it is NOT recommended to use both mechanisms on the same service manager.

The methods changeNotification(), getNotification(), and destroyNotification() do not apply to notifications provisoned in the network and enabled using enableNotifications(). These only apply to notifications created using createNotification().

Returns assignmentID: Specifies the ID assigned by the manager interface for this operation. This ID is contained in any reportNotification() that relates to notifications provisioned from within the network. Repeated calls to enableNotifications() return the same assignment ID.

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.

Returns

TpAssignmentID

Raises

TpCommonExceptions

CR-Form-v7

CHANGE REQUEST

⌘ **29.198-08** **CR** **023** ⌘ **rev** **-** ⌘ Current version: **5.1.0** ⌘

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

Proposed change affects: UICC apps⌘ ☐ ME ☐ Radio Access Network ☐ Core Network **X**

Title:	⌘	Promotion of TpDataSessionQosClass data type definition to the Common Data Types
Source:	⌘	N5
Work item code:	⌘	OSA2
Date:	⌘	31/10/2002
Category:	⌘	**F**
Release:	⌘	*REL-5*

Use one of the following categories:
F *(correction)*
A *(corresponds to a correction in an earlier release)*
B *(addition of feature),*
C *(functional modification of feature)*
D *(editorial modification)*
Detailed explanations of the above categories can be found in 3GPP TR 21.900.

Use one of the following releases:

2	*(GSM Phase 2)*
R96	*(Release 1996)*
R97	*(Release 1997)*
R98	*(Release 1998)*
R99	*(Release 1999)*
Rel-4	*(Release 4)*
Rel-5	*(Release 5)*
Rel-6	*(Release 6)*

Reason for change:	⌘	QoS class reporting functionality has been included in Multi Media Call Control, reusing a data type from Data Session Control. This has now become a common data type, and therefore needs to be removed from the Data Session Control specification.
Summary of change:	⌘	Propote data type definition of TpDataSessionQosClass to the Common Data Types
Consequences if not approved:	⌘	Duplication of data types, giving rise to maintenance problems.

Clauses affected:	⌘	11.2.5

Other specs affected: ⌘	Y	N		⌘
	☐	☐	Other core specifications	
	☐	☐	Test specifications	
	☐	☐	O&M Specifications	

Other comments: ⌘

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

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

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

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

11 Data Definitions

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

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.
OriginatingAddress	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 3G TS 23.107.~~
~~P_DATA_SESSION_QOS_CLASS_STREAMING~~	~~1~~	~~Specifies the Streaming QoS class, as specified in 3G TS 23.107.~~
~~P_DATA_SESSION_QOS_CLASS_INTERACTIVE~~	~~2~~	~~Specifies the Interactive QoS class, as specified in 3G TS 23.107.~~
~~P_DATA_SESSION_QOS_CLASS_BACKGROUND~~	~~3~~	~~Specifies the Background QoS class, as specified in 3G TS 23.107.~~

11.2.5~~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-4217:1995 [4]
AdditionalInfo	TpString	Descriptive string which is sent to the billing system without prior evaluation. Could be included in the ticket.