3GPP TSG SA Plenary **Tdoc SP-000680**
Meeting #10, Bangkok, Thailand
11th –14th December 2000

Source: **TSG CN**

Title: **CRs to TS 23.127**

Agenda item: **8.1.1**

Document for: **APPROVAL**

Introduction:

This document contains 3 CRs on **R99** Work Item **"OSA"**, that have been **AGREED by TSG CN WG5, ENDORSED by TSG CN**, and are forwarded to TSG SA Plenary meeting #10 for **APPROVAL**.

Spec	CR	Rev	Doc-2nd-Level	Phase	Subject	Cat	Ver_C
23.127	040		N5-000291	R99	Removal of the parameter serviceProperties in the method selectService	F	3.1.0
23.127	041		N5-000266	R99	setCallBackWithSessionID is missing from the base service interface	F	3.1.0
23.127	042		N5-000265	R99	getCriteria has superfluous assignmentID parameter.	F	3.1.0

3GPP Meeting CN5 #7
Sophia Antipolis, November 7-8

Document N5-000265
e.g. for 3GPP use the format TP-99xxx
or for SMG, use the format P-99-xxx

CHANGE REQUEST

Please see embedded help file at the bottom of this page for instructions on how to fill in this form correctly.

23.127	CR	042	Current Version:	3.1.0

GSM (AA.BB) or 3G (AA.BBB) specification number ↑ *↑ CR number as allocated by MCC support team*

For submission to:	SA#10	for approval	X	strategic		(for SMG use only)
list expected approval meeting # here ↑		for information		non-strategic		

Form: CR cover sheet, version 2 for 3GPP and SMG *The latest version of this form is available from: ftp://ftp.3gpp.org/Information/CR-Form-v2.doc*

Proposed change affects: *(at least one should be marked with an X)* (U)SIM [ ] ME [ ] UTRAN / Radio [ ] Core Network [X]

Source: TSG_CN **Date:** 10th Dec 2000

Subject: getCriteria has superfluous assignmentID parameter.

Work item: OSA

Category: *(only one category shall be marked with an X)*

F	Correction	X	**Release:**	Phase 2	
A	Corresponds to a correction in an earlier release			Release 96	
B	Addition of feature			Release 97	
C	Functional modification of feature			Release 98	
D	Editorial modification			Release 99	X
				Release 00	

Reason for change: When the application requests the criteria it is interested in all the criteria that this application did enable. The result will be a set of criteria. Each criteria in the set will contain the corresponding assignmentID.

Clauses affected: 7.1.1

Other specs affected:

Other 3G core specifications	[ ]	→ List of CRs:
Other GSM core specifications	[ ]	→ List of CRs:
MS test specifications	[ ]	→ List of CRs:
BSS test specifications	[ ]	→ List of CRs:
O&M specifications	[ ]	→ List of CRs:

Other comments:

help.doc

<--------- double-click here for help and instructions on how to create a CR.

7.1.1 Call Manager

The generic call manager interface provides the management functions to the generic call Service Capability Features. The application programmer can use this interface to enable or disable call-related event notifications.

Method	**`enableCallNotification()`** This method is used to enable call notifications to be sent to the application.
Direction	Application to network
Parameters	appInterface 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 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 "incoming call attempt reported by network", "answer", "no answer", "busy".
Returns	assignmentID Specifies the ID assigned by the generic call control manager object for this newly-enabled event notification.
Errors	USER_NOT_SUBSCRIBED Returned if the end-user is not subscribed to the application APPLICATION_NOT_ACTIVATED Returned if the end-user has de-activated the application USER_PRIVACY_VIOLATION Returned if the requests violates the end-user's privacy setting

Method	**`changeCallNotification()`** This method is used to change the notification criteria initially set with enableCallNotification().
Direction	Application to network
Parameters	eventCriteria Overrides the set of event criteria initially defined with enableCallNotification(). assignmentID Specifies the ID returned with enableCallNotification().
Returns	
Errors	USER_NOT_SUBSCRIBED Returned if the end-user is not subscribed to the application APPLICATION_NOT_ACTIVATED Returned if the end-user has de-activated the application USER_PRIVACY_VIOLATION Returned if the requests violates the end-user's privacy setting

Method	**`disableCallNotification()`** This method is used by the application to disable call notifications.
Direction	Application to network
Parameters	assignmentID Specifies the assignment ID given by the generic call control manager object when the previous `enableNotification()` was called.
Returns	-
Errors	`INVALID_ASSIGNMENTID` Returned if the assignment ID does not correspond to one of the valid assignment IDs.

Method	**`getCriteria()`** This method is used to retrieve the call event notification criteria set with enableCallNotification() or changeCallNotification().
Direction	Application to network
~~**Parameters**~~	~~assignmentID~~ ~~Specifies the assignment ID given by the generic call control manager object when the previous `enableNotification()` was called.~~
Returns	eventCriteria Specifies the event specific criteria currently set.
~~**Errors**~~	~~`INVALID_ASSIGNMENTID`~~ ~~Returned if the assignment ID does not correspond to one of the valid assignment Ids.~~

Method	**`callEventNotify()`** This method notifies the application of the arrival of a call-related event.
Direction	Network to application
Parameters	callReference Specifies the call session ID and the reference to the call object to which the notification relates. eventInfo Specifies data associated with this event. These data include originating address, original destination address, redirecting address and application information, which consists of teleservice information, bearer service information, calling party's category, presentation address, additional calling party address, alerting mechanism, network access type, interworking indicators and generic info for operator specific information. assignmentID Specifies the assignment ID which was returned by the `enableNotification()` method. The application can use assignment ID to associate events with event-specific criteria and to act accordingly. appInterface Specifies a reference to the application object which implements the callback interface for the new call.
Returns	-
Errors	-

Method	**callAborted()**
	This method indicates to the application that the call object has aborted or terminated abnormally. No further communication will be possible between the call object and the application.
Direction	Network to application
Parameters	callReference Specifies the call object that has aborted or terminated abnormally.
Returns	-
Errors	-

Method	**callNotificationInterrupted()**
	This method indicates to the application that all event notifications have been temporary interrupted (for example, due to faults detected). Note that more permanent failures are reported via the Framework (integrity management).
Direction	Network to application
Parameters	-
Returns	-
Errors	-

Method	**callNotificationContinued()**
	This method indicates to the application that event notifications will again be possible.
Direction	Network to application
Parameters	-
Returns	-
Errors	-

3GPP Meeting CN5 #7
Sophia Antipolis, November 7-8

Document N5-000266
e.g. for 3GPP use the format TP-99xxx
or for SMG, use the format P-99-xxx

CHANGE REQUEST

Please see embedded help file at the bottom of this page for instructions on how to fill in this form correctly.

23.127	CR	041	Current Version:	3.1.0

GSM (AA.BB) or 3G (AA.BBB) specification number ↑ *↑ CR number as allocated by MCC support team*

For submission to:	SA#10	for approval	X	strategic		(for SMG use only)
list expected approval meeting # here ↑		for information		non-strategic		

Form: CR cover sheet, version 2 for 3GPP and SMG *The latest version of this form is available from: ftp://ftp.3gpp.org/Information/CR-Form-v2.doc*

Proposed change affects: *(at least one should be marked with an X)*

(U)SIM		ME		UTRAN / Radio		Core Network	X

Source: TSG_CN **Date:** 10th Dec 2000

Subject: The setCallBackWithSessionID is missing from the base service interface. This provides an extra parameter to specify for which session the callBack applies.

Work item: OSA

Category: *(only one category shall be marked with an X)*

F	Correction	X	**Release:**	Phase 2	
A	Corresponds to a correction in an earlier release			Release 96	
B	Addition of feature			Release 97	
C	Functional modification of feature			Release 98	
D	Editorial modification			Release 99	X
				Release 00	

Reason for change: Most of the SCFs use sessionIDs in the methods to allow addressing of a specifc session. This allows an implemenation to use a 'single object' to handle multiple sessions (e.g., calls). The callback objects are session related. Therefore, the application must be able to specify which session the setCallBack applies to.

The method is already present in the 29.198 v 3.1.0

Clauses affected: 5.4.2

Other specs affected:

Other 3G core specifications		→ List of CRs:	
Other GSM core specifications		→ List of CRs:	
MS test specifications		→ List of CRs:	
BSS test specifications		→ List of CRs:	
O&M specifications		→ List of CRs:	

Other comments:

help.doc

<--------- double-click here for help and instructions on how to create a CR.

5.4.2 Base Service Interface

This interface provides the base for all interfaces described in the following clauses. It allows an application to set an interface reference to be used by the OSA interfaces for requests and asynchronous responses to the application. For example, when an application wants to be notified upon the receipt of the "called party busy" event, the Service Capability Server must know where to send the notification. This reference can be provided by the application with the setCallBack method across the OSA API.

Name	Base_Service_Interface
Method	**`setCallback()`** This method specifies the reference address of the callback interface that an SCF uses to invoke methods on the application.
Direction	Application to Framework
Parameters	appInterface Specifies a reference to the application interface which is used for callbacks.
Returns	
Errors	

Name	Base_Service_Interface
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.
Direction	Application to Framework
Parameters	appInterface Specifies a reference to the application interface which is used for callbacks. sessionID Specifies the session for which the service can invoke the application's callback interface
Returns	
Errors	

3GPP TSG-CN WG5 Meeting #7
Sophia Antipolis, France, 7th – 8th November, 2000

***Document* N5-000291**
e.g. for 3GPP use the format TP-99xxx
or for SMG, use the format P-99-xxx

CHANGE REQUEST

Please see embedded help file at the bottom of this page for instructions on how to fill in this form correctly.

23.127	CR	040	Current Version:	3.1.0

GSM (AA.BB) or 3G (AA.BBB) specification number ↑ *↑ CR number as allocated by MCC support team*

For submission to:	TSG-SA#10	for approval	X	strategic		(for SMG use only)
list expected approval meeting # here ↑		for information		non-strategic		

Form: CR cover sheet, version 2 for 3GPP and SMG *The latest version of this form is available from: ftp://ftp.3gpp.org/Information/CR-Form-v2.doc*

Proposed change affects: *(at least one should be marked with an X)*	(U)SIM		ME		UTRAN / Radio		Core Network	X

Source: TSG_CN **Date:** 10th Dec 2000

Subject: Removal of the parameter serviceProperties in the method selectService

Work item: OSA

Category: *(only one category shall be marked with an X)*

F	Correction	X	**Release:**	Phase 2	
A	Corresponds to a correction in an earlier release			Release 96	
B	Addition of feature			Release 97	
C	Functional modification of feature			Release 98	
D	Editorial modification			Release 99	X
				Release 00	

Reason for change: The serviceProperties need not be supplied any more at this phase as the specific serviceID has been determined already.

Clauses affected: 6.1.3

Other specs affected:

Other 3G core specifications		→ List of CRs:
Other GSM core specifications		→ List of CRs:
MS test specifications		→ List of CRs:
BSS test specifications		→ List of CRs:
O&M specifications		→ List of CRs:

Other comments:


help.doc

<--------- double-click here for help and instructions on how to create a CR.

6.1.3 OSA Access

During an authenticated session accessing the Framework, the application will be able to select and access an instance of a framework or network service capability feature.

Access to framework SCFs is gained by invoking the obtainInterface, or obtainInterfaceWithCallback methods. The latter is used when a callback reference is supplied to the framework. For example, a network SCF discovery interface reference is returned when invoking obtainInterface with "discovery" as the SCF name.

In order to use network SCFs, the application must first be authorised to do so by establishing a service agreement with the Home Environment. The application uses the discovery SCF to retrieve the ID of the network SCF they wish to use.They may then use the accessCheck method to check that they are authorised to use the network SCF. The selectService method is used to tell the Home Environment that the application wishes to use the network SCF. The signServiceAgreement method is used to digitally sign the agreement, and provide non-repudiation for both parties in agreeing that the SCF would be available for use.

Establishing a service agreement is a business level transaction, which requires the HE-VASP that owns the application to agree terms for the use of an SCF with the Home Environment. Service agreements can be reached using either off-line or on-line mechanisms. Off-line agreements will be reached outside of the scope of OSA interactions, and so are not described here. However, applications can make use of service agreements that are made off-line. Some Home Environments may only offer off-line mechanisms to reach service agreements.

After a service agreement has been established between the application and the Home Environment domains, the application will be able to make use of this agreement to access the network SCF.

The accessCheck method allows the application to check whether it has permission to access (read, write, etc) to a specified SCF, and specific SCF features. The application defines the security domain and context of access to the SCF. The access control policy is based on a number of conditions, events and permissions that determine whether the application is authorised to access the SCF/feature.

The accessCheck method is optional, in that can be called by the application to check that it has permission to use specific SCF features, before starting an SCF instance. It is not compulsory for the application to make this check before selecting a network SCF and signing a service agreement to use an instance of the SCF. If the accessCheck method confirms that the application has permission to use a specific SCF feature, then this feature should be available to the application when using the SCF instance. The Home Environment may include the results of the accessCheck as part of the service agreement, that is signed before using an SCF instance, thereby assuring the application that the SCF features will be available.

The selectService method is used to identify the SCF that the application wishes to use. ~~A list of service properties initialises the SCF, and a~~An SCF token is returned. The application and Home Environment must sign a copy of the service agreement to confirm the use of the SCF. The framework invokes signServiceAgreement method on the applications's Access callback interface with the service agreement text to be signed. The application uses its digital signature key to sign the agreement text, and return the signed text to the framework. The application then calls the signServiceAgreement method on the OSA Access SCF. The framework signs the agreement text, retrieves a reference to a network manager interface for the selected SCF (using the getServiceManager method defined in clause 8), and returns this reference to the client application. In addition, the OSA Access interface may be invoked by SCSs in the context of SCF registration, see subclause 8.1.

The OSA Access framework SCF is defined by a single interface, which consists of the following methods.

Method	**`obtainInterface ()`** The application uses this method to obtain interface references to other framework SCFs (e.g. discovery, load manager). (The obtainInterfacesWithCallback method should be used if the application is required to supply a callback interface to the framework.)
Direction	Application to network
Parameters	**interfaceName** The name of the framework SCF to which a reference to the interface is requested.
Returns	**fwInterface**

	This is the reference to the SCF interface requested.
Errors	**INVALID_INTERFACE_NAME** Returned if the interfaceName is invalid.

Method	**`obtainInterfaceWithCallback ()`** The application uses this method to obtain interface references to other framework SCFs (e.g. discovery, load manager), when they are required to supply a callback interface to the framework. (The obtainInterface method should be used when no callback interface needs to be supplied.)
Direction	Application to network
Parameters	**interfaceName** The name of the framework SCF to which a reference to the interface is requested. **appInterface** This is the reference to the application interface which is used for callbacks. If an application interface is not needed, then this method should not be used. (The obtainInterface method should be used when no callback interface needs to be supplied.)
Returns	**fwInterface** This is the reference to the SCF requested.
Errors	**INVALID_INTERFACE_NAME** Returned if the interfaceName is invalid.

Method	**`accessCheck()`** This method may be used by the application to check whether it has been granted permission to access the specified SCF. The response is used to indicate whether the request for access has been granted or denied and if granted the level of trust that will be applied.
Direction	Application to network
Parameters	**securityContext** A context is a group of security relevant attributes that may have an influence on the result of the accessCheck request. **securityDomain** The security domain in which the application is operating may influence the access control decisions and the specific set of features that the requestor is entitled to use. **group** A group can be used to define the access rights associated with all applications that belong to that group. This simplifies the administration of access rights. **serviceAccessTypes** These are defined by the specific Security Model in use but are expected to include: Create, Read, Update, Delete as well as those specific to SCFs.
Returns	**serviceAccessControl** This is a structure containing:

	• policy: indicates whether access has been granted or denied. If granted then the parameter trustLevel must also have a value. • trustLevel: The trustLevel parameter indicates the trust level that the Home Environment has assigned to the application.
Errors	

Method	**`selectService ()`** This method is used by the application to identify the network SCF that the application wishes to use.
Direction	Application to network
Parameters	**serviceID** This identifies the SCF required. ~~**serviceProperties**~~ ~~This is a list of the properties that the SCF should support. These properties (names and values) are used to initialise the SCF instance for use by the application.~~
Returns	**serviceToken** This is a free format text token returned by the framework, which can be signed as part of a service agreement. This will contain operator specific information relating to the service level agreement. The serviceToken has a limited lifetime. If the lifetime of the serviceToken expires, a method accepting the serviceToken will return an error code (`INVALID_Service_TOKEN`). Service Tokens will automatically expire if the application or framework invokes the endAccess method on the other's corresponding access interface.
Errors	**`INVALID_SERVICE_ID`** Returned if the serviceID is not recognised by the framework **`INVALID_SERVICE_PROPERTY`** Returned if a property is not recognised by the framework