Technical Specification Group Services and System Aspects
Meeting #27, Tokyo, JAPAN, 14 - 17 March 2005

TSGS#27(05)0044

Source: **SA5 (Telecom Management)**

Title: **CR 32602-3-4 Basic Configuration Management (CM) IRP**

Document for: **Approval**

Agenda Item: **7.5.3**

Doc-1st-Level	Spec	CR	R	Phase	Subject	Ca	VerCr	Doc-2nd-Level	Workitem
SP-050044	32.602	008	--	Rel-6	Apply Generic System Context	F	6.0.0	S5-056032	OAM-NIM
SP-050044	32.603	017	--	Rel-6	Generic System Context, update of reference to IS specification	F	6.2.0	S5-056075	OAM-NIM
SP-050044	32.604	005	--	Rel-6	Generic System Context, update of reference to IS specification	F	6.0.0	S5-056076	OAM-NIM
SP-050044	32.603	016	--	Rel-6	IDL incompliant to the style guide	F	6.2.0	S5-056070	OAM-NIM

3GPP TSG-SA5 (Telecom Management) **S5-056032**
Meeting #41, Lisbon, PORTUGAL, 24-28 January 2005

CR-Form-v7

CHANGE REQUEST

⌘ **32.602 CR 008** ⌘**rev** **-** ⌘ Current version: **6.0.0** ⌘

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 ☒

Title: ⌘	Apply Generic System Context		
Source: ⌘	**SA5** (Ericsson, thomas.tovinger@ericsson.com)		
Work item code: ⌘	OAM-NIM	***Date:*** ⌘	28/1/2005
Category: ⌘	**F**	***Release:*** ⌘	Rel-6

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: ⌘
1. Today we have redundant, time-consuming and error prone duplication of the same text for the System Context in all Interface IRPs.
2. Obsolete reference to earlier releases in subclause 4.2, and in clause 5 obsolete/redundant introduction to the IRP modelling concept which is now placed in 32.150 (aligning with the other IRPs).

Summary of change: ⌘
1. Align the title of subclause 4.1 with other Interface IRPs and modify the text of 4.1 with a generic text, referring to the new common definition in 32.150 for the System Context for all Interface IRPs, but keep the diagrams for readability.
2. Remove obsolete reference to earlier releases in subclause 4.2 and replace the obsolete/redundant introduction to the IRP modelling concept in clause 5 with a reference to 32.150.

Consequences if not approved: ⌘ Redundant, time-consuming and error prone duplication of the same text for many IRP TSs. Wrong references.

Clauses affected: ⌘ 2, 4.1, 4.2, 5.

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

Other comments: ⌘

Change in Clause 2

2 References

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

- References are either specific (identified by date of publication, edition number, version number, etc.) or non-specific.
- For a specific reference, subsequent revisions do not apply.
- For a non-specific reference, the latest version applies.

…

[13] 3GPP TS 32.622: "Telecommunication management; Configuration Management (CM); Generic network resources Integration Reference Point (IRP): Network Resource Model (NRM)".

[14] 3GPP TS 32.150: "Telecommunication management; Integration Reference Point (IRP) Concept and definitions".

End of Change in Clause 2

Change in Clause 4

4 System overview

4.1 System C~~c~~ontext

The general definition of the System Context for the present IRP is found in 3GPP TS 32.150 [14] subclause 4.7.

In addition, the set of related IRP(s) relevant to the present IRP is shown in the two diagrams below.~~Figure 4.1 and 4.2 identify system contexts of the IRP defined by the present specification in terms of its implementation called IRPAgent and the user of the IRPAgent, called IRPManager. For a definition of IRPManager and IRPAgent, see 3GPP TS 32.102 [2].~~

~~The IRPAgent implements and supports this IRP. The IRPAgent can reside in an Element Manager (EM) or a Network Element (NE) (see also [2] clause 8). In the former case, the interfaces (represented by a thick dotted line) between the EM and the NEs are not the subject of this IRP.~~

~~An NE can be managed via System Context A or B. The criterion for choosing System Context A or B to manage a particular NE is implementation dependent. An IRPAgent shall support one of the two System Contexts. By observing the interaction across the Itf-N, an IRPManager cannot deduce if the EM and NE are integrated in a single system or if they run in separate systems.~~

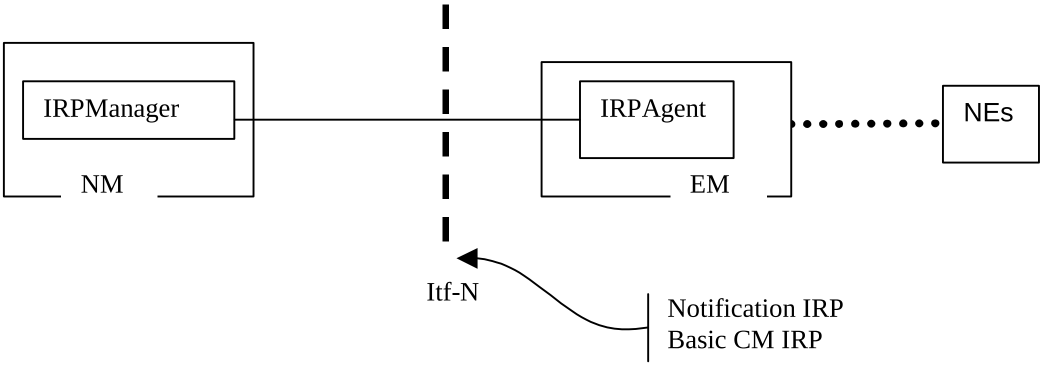

Figure 4.1: System Context A

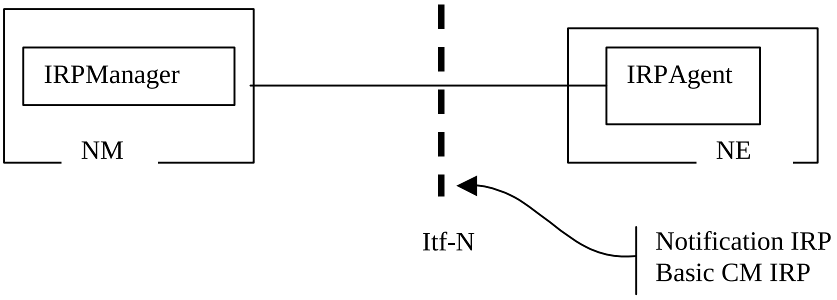

Figure 4.2: System Context B

4.2 Compliance rules

For general definitions of compliance rules related to qualifiers (Mandatory/Optional/Conditional) for *operations, notifications and parameters* (of operations and notifications) please refer to 3GPP TS 32.150~~02~~ [14~~2~~].

An IRPAgent that incorporates vendor-specific extensions shall support normal communication with a 3GPP SA5-compliant IRPManager with respect to all Mandatory and Optional managed object classes, attributes, associations, operations, parameters and notifications without requiring the IRPManager to have any knowledge of the extensions.

Given that

- rules for vendor-specific extensions remain to be fully specified, and
- many scenarios under which IRPManager and IRPAgent interwork may exist,

it is recognised that ~~in Release 4/5~~ the IRPManager, even though it is not required to have knowledge of vendor-specific extensions, may be required to be implemented with an awareness that extensions can exist and behave accordingly.

End of Change in Clause 4

Change in Clause 5

5 Modelling approach

See 3GPP TS 32.150 [14].~~This clause identifies the modelling approach adopted and used in this IRP.~~

~~As described in 3GPP TS 32.101 [1], an IRP comprises the following components:~~

~~(1)an IRP Information Model that specifies the interface in a protocol neutral manner, defined as an Information Service and/or one or more Network Resource Models,~~

~~(2)a number of IRP Solution Sets that provide the actual realization of the operations and notifications defined in the IRP Information Model for each protocol environment.~~

~~The present document defines one such Information Service – the Basic CM IRP: IS.~~

~~The IRP Information Service is a specification of the *operations* and *notifications* that are visible over the IRP. These operations/notifications are generic in the sense that they do not specify the Managed Objects that are retrieved/manipulated/informed about over the interface, and thus this IS is independent of the NRM being managed.~~

~~5.1 IRP Information Service modelling approach~~

~~The IRP Information Service of the subject IRP specifies a number of protocol-independent operations and notifications that are needed by an IRPManager to retrieve CM information from an IRPAgent.~~

~~The operations and notifications of the IRP Information Service are mainly based on the principles of the Common Management Information Service (CMIS) defined in ITU-T Recommendation X.710 [7] and ITU-T Recommendation X.721 [8] (M-GET etc.). Note however, that the Information Service of the subject IRP is focused on the operations and notifications needed for basic CM purposes and thus only covers a subset of the operations/notifications defined in ITU-T Recommendation X.710 [7]/ITU-T Recommendation X.721 [8].~~

~~It is expected that most Solution Sets will implement the operations and notifications by mapping them to standard operations (and possibly standard notifications) that are applicable in the corresponding protocol environment. A CMIP Solution Set should for instance map the operations to the more generic operations defined in CMIS, an SNMP Solution Set should map the operations to applicable SNMP operations, and a CORBA Solution Set should map the operations to applicable OMG/CORBA services.~~

End of Change in Clause 5
End of document

Annex A (informative): Change history

Change history							
Date	**TSG #**	**TSG Doc.**	**CR**	**Rev**	**Subject/Comment**	**Old**	**New**
Dec 2003	S_22	SP-030630	005	--	Correction of System Context	5.1.0	5.2.0
Mar 2004	S_23	SP-040119	007	--	Correction of System Context	5.2.0	5.3.0
Mar 2004	S_23	SP-040105	--	--	Automatic upgrade to Rel-6 (no CR)	5.3.0	6.0.0

3GPP TSG-SA5 (Telecom Management) *Tdoc ⌘S5-056070*
Meeting #41, Lisbon, PORTUGAL, 24 - 28 January 2005

CR-Form-v7.1

CHANGE REQUEST

⌘	**32.603**	**CR**	**016**	⌘ **rev**	**-**	⌘	Current version:	**6.2.0** ⌘

*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 ☒

Title:	⌘	IDL incompliant to the style guide		
Source:	⌘	**SA5** (huangsq@zte.com.cn)		
Work item code:	⌘	OAM-NIM	***Date:*** ⌘	28/01/2005
Category:	⌘	**F**	***Release:*** ⌘	Rel-6

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)*
Rel-7	*(Release 7)*

Reason for change:	⌘	The IDL does not reflect the format recommended by the style guide (TS 32.150).
Summary of change:	⌘	Add double slash between “ #endif “ and the macro.
Consequences if not approved:	⌘	The IDL won’t conform to the styleguide, and will compile erros when using java compilers (e.g., idlj.exe).

Clauses affected:	⌘	Annex A				
		Y	**N**			
Other specs affected:	⌘		**X**	Other core specifications	⌘	
			X	Test specifications		
			X	O&M Specifications		
Other comments:	⌘					

Change in Annex A

Annex A (normative): CORBA IDL, Access Protocol

A.1 IDL specification (file name "BasicCMIRPConstDefs.idl")

```
// File: BasicCMIRPConstDefs.idl

#ifndef _BASICCMIRPCONSTDEFS_IDL_
#define _BASICCMIRPCONSTDEFS_IDL_

// This statement must appear after all include statements
#pragma prefix "3gppsa5.org"

/* ## Module: BasicCMIRPConstDefs
This module contains commonly used definitions for BasicCMIRP.
================================================================
*/
module BasicCMIRPConstDefs
{

   /**
    * Defines the name of a Managed Object Class
    */
   typedef string MOClass;

   /**
    * The format of Distinguished Name (DN) is specified in 3GPP TS 32.300
    * "Name Conventions for Managed Objects".
    */
   typedef string DN;


   /**
    * Defines the name of an attribute of a Managed Object
    */
   typedef string MOAttributeName;

   /**
    * Defines the value of an attribute of a Managed Object in form of a CORBA
    * Any. Apart from basic datatypes already defined in CORBA, the allowed
    * attribute value types are defined in the AttributeTypes module.
    */
   typedef any MOAttributeValue;

   /**
    *
    * In this version the only allowed filter value is "TRUE" i.e. a filter that
    * matches everything.
    */
   typedef string FilterType;

    /**
     * ResultContents is used to tell how much information to get back
     * from the find_managed_objects operation.
     *
     * NAMES: Used to get only Distinguished Name
     *        for MOs.
```

```
  *        The name contains both the MO class
  *        and the names of all superior objects in the naming
  *        tree.
  *
  * NAMES_AND_ATTRIBUTES: Used to get both NAMES plus
  *    MO attributes (all or selected).
  */
 enum ResultContents
 {
    NAMES,
    NAMES_AND_ATTRIBUTES
 };

 /**
  * ScopeType defines the kind of scope to use in a search
  * together with SearchControl.level, in a SearchControl value.
  *
  * SearchControl.level is always >= 0. If a level is bigger than the
  * depth of the tree there will be no exceptions thrown.
  * BASE_ONLY: level ignored, just return the base object.
  * BASE_NTH_LEVEL: return all subordinate objects that are on "level"
  * distance from the base object, where 0 is the base object.
  * BASE_SUBTREE: return the base object and all of its subordinates
  * down to and including the nth level.
  * BASE_ALL: level ignored, return the base object and all of it's
  * subordinates.
  */
 enum ScopeType
 {
    BASE_ONLY,
    BASE_NTH_LEVEL,
    BASE_SUBTREE,
    BASE_ALL
 };

 /**
  * SearchControl controls the find_managed_object search,
  * and contains:
  * the type of scope ("type" field),
  * the level of scope ("level" field), level 0 means the "baseObject",
  *    level 1 means baseobject including its sub-ordinates etc..
  * the filter ("filter" field),
  * the result type ("contents" field).
  * The type, level and contents fields are all mandatory.
  * The filter field contains the filter expression.
  *  The string "TRUE" indicates "no filter",
  * i.e. a filter that matches everything.
  */
 struct SearchControl
 {
    ScopeType type;
    unsigned long level;
    FilterType filter;
    ResultContents contents;
 };

 /**
  * Represents an attribute: "name" is the attribute name
  * and "value" is the attribute value.
  */
struct MOAttribute
{
   MOAttributeName name;
   MOAttributeValue value;
};

 typedef sequence <MOAttribute> MOAttributeSet;


 struct Result
 {
    DN mo;
    MOAttributeSet attributes;
 };

 typedef sequence <Result> ResultSet;

 /**
  * AttributeErrorCategory defines the categories of errors, related to
  * attributes, that can occur during creation or modification of MOs.
  *
  * NO_SUCH_ATTRIBUTE: The specified attribute does not exist.
  * INVALID_ATTRIBUTE_VALUE: The specified attribute value is not valid.
  * MISSING_ATTRIBUTE_VALUE: An attribute value is required but none was
  *   provided and no default value is defined for the attribute.
  * INVALID_MODIFY_OPERATOR: The specified modify operator is not valid
  *   (e.g. operator ADD_VALUES applied to a non multi-valued attribute
  *   or operator SET_TO_DEFAULT applied where no default value is defined).
  * MODIFY_NOT_ALLOWED: The modification of the attribute is not allowed.
  * MODIFY_FAILED: The modification failed because of an unspecified reason.
  */
 enum AttributeErrorCategory
 {
    NO_SUCH_ATTRIBUTE,
    INVALID_ATTRIBUTE_VALUE,
    MISSING_ATTRIBUTE_VALUE,
    INVALID_MODIFY_OPERATOR,
    MODIFY_NOT_ALLOWED,
    MODIFY_FAILED
 };

 /**
  * DeleteErrorCategory defines the categories of errors that can occur
  * during deletion of MOs.
  *
  * SUBORDINATE_OBJECT: The MO cannot be deleted due to subordinate MOs.
  * DELETE_NOT_ALLOWED: The deletion of the MO is not allowed.
  * DELETE_FAILED: The deletion failed because of an unspecified reason.
  */
 enum DeleteErrorCategory
 {
    SUBORDINATE_OBJECT,
    DELETE_NOT_ALLOWED,
    DELETE_FAILED
 };

 /**
  * AttributeError represents an error, related to an attribute, that occured
  * during creation or modification of MOs.
  * It contains:
  * - the name of the indicted attribute ("name" field),
  * - the category of the error ("error" field),
  * - optionally, the indicted attribute value ("value" field),
  * - optionally, additional details on the error ("reason" field).
  */
 struct AttributeError
```

```
{
   MOAttributeName name;
   AttributeErrorCategory error;
   MOAttributeValue value;
   string reason;
};

typedef sequence <AttributeError> AttributeErrorSeq;

/**
 * DeleteError represents an error that occured during deletion of MOs.
 * It contains:
 * - the distinguished name of the indicted MO ("objectName" field),
 * - the category of the error ("error" field),
 * - optionally, additional details on the error ("reason" field).
 */
struct DeleteError
{
   DN objectName;
   DeleteErrorCategory error;
   string reason;
};

typedef sequence <DeleteError> DeleteErrorSeq;

/**
 * ModifyAttributeErrors represents errors that occured during
 * modification of attributes of a MO.
 * It contains:
 * - the distinguished name of the indicted MO ("objectName" field),
 * - a sequence containing the attribute errors ("errors" field).
 */
struct ModifyAttributeErrors
{
   DN objectName;
   AttributeErrorSeq errors;
};

typedef sequence <ModifyAttributeErrors> ModifyAttributeErrorsSeq;

typedef sequence <MOAttributeName> AttributeNameSet;

/**
 * ModifyOperator defines the way in which an attribute value is to be
 * applied to an attribute in a modification of MO attributes.
 *
 * REPLACE: replace the current value with the provided value
 * ADD_VALUES: for a multi-valued attribute, add the provided values to the
 *   current list of values
 * REMOVE_VALUES: for a multi-valued attribute, remove the provided values
 *   from the current list of values
 * SET_TO_DEFAULT: set the attribute to its default value
 */
enum ModifyOperator
{
   REPLACE,
   ADD_VALUES,
   REMOVE_VALUES,
   SET_TO_DEFAULT
};

/**
 * AttributeModification defines an attribute value and the way it is to
```

```
      * be applied to an attribute in a modification of MO attributes.
      * It contains:
      * - the name of the attribute to modify ("name" field),
      * - the value to apply to this attribute ("value" field),
      * - the way the attribute value is to be applied to the attribute
      *   ("operator" field).
      */
     struct AttributeModification
     {
        MOAttributeName name;
        MOAttributeValue value;
        ModifyOperator operator;
     };

     typedef sequence <AttributeModification> AttributeModificationSet;

};
#endif // _BASICCMIRPCONSTDEFS_IDL_
```

A.2 IDL specification (file name "BasicCMIRPSystem.idl")

```
// File: BasicCMIRPSystem.idl

#ifndef _BASICCMIRPSYSTEM_IDL_
#define _BASICCMIRPSYSTEM_IDL_

#include "ManagedGenericIRPConstDefs.idl"
#include "ManagedGenericIRPSystem.idl"
#include "BasicCMIRPConstDefs.idl"

// This statement must appear after all include statements
#pragma prefix "3gppsa5.org"

module BasicCmIRPSystem
{

    exception IllegalFilterFormatException {
       string reason;
    };
    exception IllegalDNFormatException {
       string reason;
    };
    exception IllegalScopeTypeException {
       string reason;
    };
    exception IllegalScopeLevelException {
       string reason;
    };
    exception UndefinedMOException {
       string reason;
    };

   exception UndefinedScopeException {
       string reason;
   };

   exception FilterComplexityLimit {
      string reason;
   };
```

```
exception DuplicateMO {};

exception CreateNotAllowed {};

exception ObjectClassMismatch {};

exception NoSuchObjectClass {
   BasicCMIRPConstDefs::MOClass objectClass;
};

 exception ParentObjectDoesNotExist {};

/**
 * System otherwise fails to complete the operation.  System can provide
 * reason to qualify the exception. The semantics carried in reason
 * is outside the scope of this IRP.
 */
exception NextBasicCmInformations { string reason; };
exception NextDeleteErrors { string reason; };
exception NextModifyErrors { string reason; };
exception DestroyException { string reason; };
exception GetBasicCmIRPVersion { string reason; };
exception GetBasicCmIRPOperationProfile { string reason; };
exception GetBasicCmIRPNotificationProfile { string reason; };
exception FindManagedObjects { string reason; };
exception CreateManagedObject { string reason; };
exception DeleteManagedObjects { string reason; };
exception ModifyManagedObjects { string reason; };

 /**
  The BasicCmInformationIterator is used to iterate through a snapshot of
   Managed Object Information when IRPManager invokes find_managed_objects.
   IRPManager uses it to pace the return of Managed Object Information.

  IRPAgent controls the life-cycle of the iterator. However, a destroy
   operation is provided to handle the case where IRPManager wants to stop
   the iteration procedure before reaching the last iteration.
  */
 interface BasicCmInformationIterator
 {

    /**
     This method returns between 1 and "how_many" Managed Object information.
      The IRPAgent may return less than "how_many" items even if there are
      more items to return. "how_many" must be non-zero. Return TRUE if there
      may be more Managed Object information to return. Return FALSE if there
      are no more Managed Object information to be returned.

      If FALSE is returned, the IRPAgent will automatically destroy the
       iterator.

      @parm how_many how many elements to return in the "fetchedElements" out
       parameter.
      @parm fetchedElements the elements.
      @returns A boolean indicating if any elements are returned.
       "fetchedElements" is empty when the BasicCmInformationIterator is
       empty.
     */

     boolean next_basicCmInformations (
         in unsigned short how_many,
         out BasicCMIRPConstDefs::ResultSet fetchedElements
     )
    raises (
        NextBasicCmInformations,
        ManagedGenericIRPSystem::InvalidParameter,
         ManagedGenericIRPSystem::OperationNotSupported);

    /**
    This method destroys the iterator.
   */

    void destroy ()
    raises (
         DestroyException,
         ManagedGenericIRPSystem::OperationNotSupported);

}; // end of BasicCmInformationIterator


/**
 The DeleteResultIterator is used to iterate through the list of deleted MOs
  when IRPManager invokes method "delete_managed_objects".
  IRPManager uses it to pace the return of Managed Object Information.

 IRPAgent controls the life-cycle of the iterator. However, a destroy
  operation is provided to handle the case where IRPManager wants to stop
  the iteration procedure before reaching the last iteration.
 */
interface DeleteResultIterator : BasicCmInformationIterator
{

   /**
    Inherited method "next_basicCmInformations" has the same behaviour as
     for interface BasicCmInformationIterator, except that:
     - The Managed Object information returned in parameter
       "fetchedElements" contains only the DNs of the deleted MOs
       (no attributes are returned).
     - If FALSE is returned, the IRPAgent will not automatically destroy the
       iterator.
    */

   /**
    This method returns between 0 and "how_many" deletion errors. The
     IRPAgent may return less than "how_many" items even if there are more
     items to return. "how_many" must be non-zero. Return TRUE if there are
     more deletion errors to return. Return FALSE if there are no more
     deletion errors to be returned.

     If FALSE is returned and last call to inherited method
     "next_basicCmInformations" also returned FALSE (i.e. no more Managed
     Object information to be returned), the IRPAgent will automatically
     destroy the iterator.

     @parm how_many: how many deletion errors to return in the
      "fetchedDeleteErrors" out parameter.
     @parm fetchedDeleteErrors: the deletion errors.
     @returns: a boolean indicating if any deletion errors are returned.
    */

    boolean next_deleteErrors (
        in unsigned short how_many,
        out BasicCMIRPConstDefs::DeleteErrorSeq fetchedDeleteErrors
    )
    raises (
        NextDeleteErrors,
        ManagedGenericIRPSystem::InvalidParameter);

};  // end of DeleteResultIterator


/**
 The ModifyResultIterator is used to iterate through the list of modified
  MOs when IRPManager invokes method "modify_managed_objects".
  IRPManager uses it to pace the return of Managed Object Information.

 IRPAgent controls the life-cycle of the iterator. However, a destroy
  operation is provided to handle the case where IRPManager wants to stop
  the iteration procedure before reaching the last iteration.
 */
interface ModifyResultIterator : BasicCmInformationIterator
{

   /**
    Inherited method "next_basicCmInformations" has the same behaviour as
     for interface BasicCmInformationIterator, except that:
     - The Managed Object information returned in parameter
       "fetchedElements" contains DNs and attributes of the modified MOs.
     - If FALSE is returned, the IRPAgent will not automatically destroy the
       iterator.
    */

   /**
    This method returns between 0 and "how_many" modification errors. The
     IRPAgent may return less than "how_many" items even if there are more
     items to return. "how_many" must be non-zero. Return TRUE if there are
     more modification errors to return. Return FALSE if there are no more
     modification errors to be returned.

     If FALSE is returned and last call to inherited method
     "next_basicCmInformations" also returned FALSE (i.e. no more Managed
     Object information to be returned), the IRPAgent will automatically
     destroy the iterator.

     @parm how_many: how many modification errors to return in the
      "fetchedModifyErrors" out parameter.
     @parm fetchedModifyErrors: the modification errors.
     @returns: a boolean indicating if any modification errors are returned.
    */

    boolean next_modificationErrors (
        in unsigned short how_many,
        out BasicCMIRPConstDefs::ModifyAttributeErrorsSeq
            fetchedModifyErrors
    )
    raises (
        NextModifyErrors,
        ManagedGenericIRPSystem::InvalidParameter);

};  // end of ModifyResultIterator

/**
  * The BasicCmIrpOperations interface.
  * Supports a number of Resource Model versions.
  */
 interface BasicCmIrpOperations
 {

   /**
    * Get the version(s) of the interface
    *
    * @raises GetBasicCmIRPVersion when the system for some reason
    *   can not return the supported versions.
    * @returns all supported versions.
    */
   ManagedGenericIRPConstDefs::VersionNumberSet get_basicCm_IRP_version()
      raises (GetBasicCmIRPVersion);

   /**
    * Return the operation profile for a specific Basic CM IRP version.
    *
    * @raises GetBasicCmIRPOperationProfile when the system for some reason
    *   cannot return the supported operations and parameters.
    * @returns the list of all supported operations and their supported
    *   parameters for the specified version.
    */
   ManagedGenericIRPConstDefs::MethodList get_basicCm_IRP_operation_profile
   (
       in ManagedGenericIRPConstDefs::VersionNumber basicCm_IRP_version
   )
   raises (
       GetBasicCmIRPOperationProfile,
       ManagedGenericIRPSystem::OperationNotSupported,
       ManagedGenericIRPSystem::InvalidParameter);

   /**
    * Return the notification profile for a specific Basic CM IRP version.
    *
    * @raises GetBasicCmIRPNotificationProfile when the system for some
    *   reason cannot return the supported notifications and parameters.
    * @returns the list of all supported notifications and their supported
    *   parameters for the specified version.
    */
   ManagedGenericIRPConstDefs::MethodList
       get_basicCm_IRP_notification_profile (
           in ManagedGenericIRPConstDefs::VersionNumber basicCm_IRP_version
   )
   raises (
       GetBasicCmIRPNotificationProfile,
       ManagedGenericIRPSystem::OperationNotSupported,
       ManagedGenericIRPSystem::InvalidParameter);

   /**
    * Performs a containment search, using a SearchControl to
    * control the search and the returned results.
    *
    * All MOs in the scope constitute a set that the filter works on.
    * The result BasicCmInformationIterator contains all matched MOs,
    * with the amount of detail specified in the SearchControl.
    * For the special case when no managed objects are matched in
    * find_managed_objects, the BasicCmInformationIterator will be returned.
    * Executing the next_basicCmInformations in the
    * BasicCmInformationIterator will return FALSE for
    * completion.
    *
    * @parm baseObject The start MO in the containment tree.
    * @parm searchControl the SearchControl to use.
    * @parm requestedAttributes defines which attributes to get.
    *   If this parameter is empty (""),  all attributes shall
    *   be returned. In this version this is the only supported semantics.
    *   Note that this argument is only
    *   relevant if ResultContents in the search control is
```

```
     *   specifed to NAMES_AND_ATTRIBUTES.
     *
     *
     * @raises ManagedGenericIRPSystem::ValueNotSupported if a valid but
     * unsupported parameter value is passed. E.g. the contents
     * field in the searchcontrol parameter contains the value NAMES and
     * the optional getContainment IS operation is not supported.
     * @raises UndefinedMOException The MO does not exist.
     * @raises IllegalDNFormatException The dn syntax string is
     * malformed.
     * @raises IllegalScopeTypeException The ScopeType in scope contains
     * an illegal value.
     * @raises IllegalScopeLevelException The scope level is negative
     * (<0).
     * @raises IllegalFilterFormatException The filter string is
     * malformed.
     * @raises FilterComplexityLimit if the filter syntax is correct,
     *   but the filter is too complex to be processed by the IRP agent.
     * @see SearchControl
     * @see BasicCmInformationIterator
     */
    BasicCmInformationIterator find_managed_objects(
        in BasicCMIRPConstDefs::DN baseObject,
        in BasicCMIRPConstDefs::SearchControl searchControl,
        in BasicCMIRPConstDefs::AttributeNameSet requestedAttributes)
    raises (
        FindManagedObjects,
        ManagedGenericIRPSystem::ParameterNotSupported,
        ManagedGenericIRPSystem::InvalidParameter,
        ManagedGenericIRPSystem::ValueNotSupported,
        ManagedGenericIRPSystem::OperationNotSupported,
        UndefinedMOException,
        IllegalDNFormatException,
        UndefinedScopeException,
        IllegalScopeTypeException,
        IllegalScopeLevelException,
        IllegalFilterFormatException,
        FilterComplexityLimit);

    /**
     * Performs the creation of a MO instance in the MIB maintained
     * by the IRPAgent.
     *
     * @parm objectName: the distinguished name of the MO to create.
     * @parm referenceObject: the distinguished name of a reference MO.
     * @parm attributes: in input, initial attribute values for the MO to
     *   create; in output, actual attribute values of the created MO.
     * @parm attributeErrors: errors, related to attributes, that caused the
     *   creation of the MO to fail.
     *
     * @raises ManagedGenericIRPSystem::OperationNotSupported: The operation
     *   is not supported.
     * @raises ManagedGenericIRPSystem::ParameterNotSupported: An optional
     *   parameter is not supported.
     * @raises ManagedGenericIRPSystem::InvalidParameter: An invalid
     *   parameter value has been provided.
     * @raises UndefinedMOException: The MO does not exist.
     * @raises IllegalDNFormatException: The DN syntax string is malformed.
     * @raises DuplicateMO: A MO already exist with the same DN as the one
     *   to create.
     * @raises CreateNotAllowed: The creation of the MO is not allowed.
     * @raises ObjectClassMismatch: The object class of the MO to create does
     *   not match with the object class of the provided reference MO.
```

```
 * @raises NoSuchObjectClass: The class of the object to create is not
 *   recognized.
 * @raises ParentObjectDoesNotExist: The parent MO instance of the
 *  ManagedEntity specified to be created does not exist.
 */
void create_managed_object (
    in BasicCMIRPConstDefs::DN objectName,
    in BasicCMIRPConstDefs::DN referenceObject,
    inout BasicCMIRPConstDefs::MOAttributeSet attributes,
    out BasicCMIRPConstDefs::AttributeErrorSeq attributeErrors
)
raises (
    CreateManagedObject,
    ManagedGenericIRPSystem::OperationNotSupported,
    ManagedGenericIRPSystem::ParameterNotSupported,
    ManagedGenericIRPSystem::InvalidParameter,
    UndefinedMOException,
    IllegalDNFormatException,
    DuplicateMO,
    CreateNotAllowed,
    ObjectClassMismatch,
    NoSuchObjectClass,
    ParentObjectDoesNotExist);

/**
 * Performs the deletion of one or more MO instances from the MIB
 * maintained by the IRPAgent, using a SearchControl to control the
 * instances to be deleted.
 *
 * All MOs in the scope constitute a set that the filter works on.
 * All matched MOs will be deleted by this operation.
 * The returned DeleteResultIterator is used to retrieve the DNs of the
 * MOs deleted and the errors that may have occurred preventing deletion
 * of some MOs.
 * For the special case when no managed objects are matched in
 * delete_managed_objects, the DeleteResultIterator will be returned.
 * Executing the next_basicCmInformations in the DeleteResultIterator
 * will return FALSE for completion.
 *
 * @parm baseObject: the start MO in the containment tree.
 * @parm searchControl: the SearchControl to use; field "contents" has no
 *   meaning here and shall be ignored.
  @returns: a DeleteResultIterator (see above).
 *
 * @raises ManagedGenericIRPSystem::OperationNotSupported: The operation
 *   is not supported.
 * @raises ManagedGenericIRPSystem::InvalidParameter: An invalid
 *   parameter value has been provided.
 * @raises UndefinedMOException: The MO does not exist.
 * @raises IllegalDNFormatException: The DN syntax string is malformed.
 * @raises IllegalScopeTypeException: The ScopeType in scope contains
 *   an illegal value.
 * @raises IllegalScopeLevelException: The scope level is negative (<0).
 * @raises IllegalFilterFormatException: The filter string is malformed.
 * @raises FilterComplexityLimit: The filter syntax is correct,
 *   but the filter is too complex to be processed by the IRPAgent.
 */
DeleteResultIterator delete_managed_objects (
    in BasicCMIRPConstDefs::DN baseObject,
    in BasicCMIRPConstDefs::SearchControl searchControl
)
raises (
    DeleteManagedObjects,
            ManagedGenericIRPSystem::OperationNotSupported,
            ManagedGenericIRPSystem::InvalidParameter,
            UndefinedMOException,
            IllegalDNFormatException,
            UndefinedScopeException,
            IllegalScopeTypeException,
            IllegalScopeLevelException,
            IllegalFilterFormatException,
            FilterComplexityLimit);

        /**
         * Performs the modification of MO attributes. One or more MOs attributes
         * may be modified according to a SearchControl.
         *
         * All MOs in the scope constitute a set that the filter works on.
         * All matched MOs will have their attributes modified by this operation.
         * The returned ModifyResultIterator is used to retrieve the DNs of the
         * modified MOs together with the values of the modified attributes, and
         * the errors that may have occurred preventing modification of some
         * attributes.
         * For the special case when no managed objects are matched in
         * modify_managed_objects, the ModifyResultIterator will be returned.
         * Executing the next_basicCmInformations in the ModifyResultIterator
         * will return FALSE for completion.
         *
         * @parm baseObject: the start MO in the containment tree.
         * @parm searchControl: the SearchControl to use; field "contents" has no
             meaning here and shall be ignored.
         * @parm modifications: the values for the attributes to modify and
             the way those values are to be applied to the attributes.
          @returns: a ModifyResultIterator (see above).
         *
         * @raises ManagedGenericIRPSystem::OperationNotSupported: The operation
         *   is not supported
         * @raises ManagedGenericIRPSystem::InvalidParameter: An invalid
         *   parameter value has been provided
         * @raises UndefinedMOException: The MO does not exist.
         * @raises IllegalDNFormatException: The DN syntax string is malformed.
         * @raises IllegalScopeTypeException: The ScopeType in scope contains
         *   an illegal value.
         * @raises IllegalScopeLevelException: The scope level is negative (<0).
         * @raises IllegalFilterFormatException: The filter string is malformed.
         * @raises FilterComplexityLimit: The filter syntax is correct,
         *   but the filter is too complex to be processed by the IRPAgent.
         */
        ModifyResultIterator modify_managed_objects (
            in BasicCMIRPConstDefs::DN baseObject,
            in BasicCMIRPConstDefs::SearchControl searchControl,
            in BasicCMIRPConstDefs::AttributeModificationSet modifications
        )
        raises (
            ModifyManagedObjects,
            ManagedGenericIRPSystem::OperationNotSupported,
            ManagedGenericIRPSystem::InvalidParameter,
            UndefinedMOException,
            IllegalDNFormatException,
            UndefinedScopeException,
            IllegalScopeTypeException,
            IllegalScopeLevelException,
            IllegalFilterFormatException,
            FilterComplexityLimit);

    };
```

```
};
#endif // _BASICCMIRPSYSTEM_IDL_
```

End of change in Annex A

Annex B (informative): Change history

Change history							
Date	**TSG #**	**TSG Doc.**	**CR**	**Rev**	**Subject/Comment**	**Old**	**New**
Jun 2001	S_12	SP-010283	--	--	Approved at TSG SA #12 and placed under Change Control	2.0.0	4.0.0
Sep 2001	S_13	SP-010476	001	--	Correction of invokeIdentifier usage	4.0.0	4.1.0
Mar 2002	S_15	SP-020019	002	--	Correction of erroneous CORBA module names and mapping tables	4.1.0	4.2.0
Mar 2002	S_15	SP-020019	003	--	Corrections to Basic CM IRP CORBA Solution Set IDLs	4.1.0	4.2.0
Mar 2002	S_15	SP-020038	004	--	Addition of missing CORBA exception "ManagedGenericIRPSystem::ValueNotSupported" onto CORBA method "find_managed_objects"	4.1.0	4.2.0
Jun 2002	S_16	SP-020294	005	--	Correcting IDL definitions of notification structured event Name Value pair names	4.2.0	4.3.0
Jul 2002	--	--	--	--	Updated the Version number (420->431) and the Date on the cover page	4.3.0	4.3.1
Sep 2002	S_17	SP-020483	006	--	Add Active Basic CM feature - CORBA Solution Set	4.3.1	5.0.0
Mar 2003	S_19	SP-030139	007	--	Add CORBA equivalents to IS operations "get{Operation Notification}Profile" - alignment with 32.602 & 32.312	5.0.0	5.1.0
Mar 2003	S_19	SP-030139	008	--	Correction of IDL errors	5.0.0	5.1.0
Mar 2003	S_19	SP-030144	009	--	Add description for notifications of each activeCM operation and one exception for createMO - alignment with 32.602, Information Service	5.0.0	5.1.0
Jun 2003	S_20	SP-030279	010	--	Alignment with Basic CM IRP information service (32.602) - add one exception for the operation createMO	5.1.0	5.2.0
Mar 2004	S_23	SP-040105	--	--	Automatic upgrade to Rel-6 (no CR)	5.2.0	6.0.0
Sep 2004	S_25	SP-040567	012	--	Removal of Rules for NRM extensions - Align with 32.622 (Generic NRM IS)	6.0.0	6.1.0
Sep 2004	S_25	SP-040566	014	--	Removal of unused/duplicate definition of types MOReference and MOReferenceSet	6.0.0	6.1.0
Dec 2004	S_26	SP-040806	015	--	Align the IDL style in the CORBA SS with the IDL Style Guide in 32.150	6.1.0	6.2.0

3GPP TSG-SA5 (Telecom Management) **S5-056075**
Meeting #41, Lisbon, PORTUGAL, 24-28 January 2005

CR-Form-v7

CHANGE REQUEST

⌘ **32.603 CR 017** ⌘ **rev** **-** ⌘ Current version: **6.2.0** ⌘

*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 ☒

Title:	⌘	Generic System Context, update of reference to IS specification
Source:	⌘	**SA5** (Ericsson, thomas.tovinger@ericsson.com)
Work item code:	⌘	OAM-NIM
Date:	⌘	28/1/2005
Category:	⌘	**F**
Release:	⌘	Rel-6

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:	⌘	The Information Service (IS) for this IRP is being updated due to an approved CR (to introduce the Generic System Context).
Summary of change:	⌘	Update the reference in Scope to the new latest IS version.
Consequences if not approved:	⌘	Wrong reference in Scope to the IS version.

Clauses affected:	⌘	Scope			
		Y	**N**		
Other specs affected:	⌘		**X**	Other core specifications	⌘
			X	Test specifications	
			X	O&M Specifications	
Other comments:	⌘	This CR should only be approved if the corresponding CR on the IS to introduce the Generic System Context is approved (see the related CR collection document for an overview of all involved CR Tdoc numbers).			

Change in Clause Scope

1 Scope

The purpose of this *Basic Configuration Management (CM) IRP: CORBA Solution Set* is to define the mapping of the Basic CM IRP: IS (see 3GPP TS 32.602 [4]) to the protocol specific details necessary for implementation of this IRP in a CORBA/IDL environment.

This document defines NRM independent data types and methods.

This Solution Set specification is related to 3G TS 32.602 V6.1~~0~~.X.

End of Change in Clause Scope

Annex B (informative): Change history

Change history							
Date	**TSG #**	**TSG Doc.**	**CR**	**Rev**	**Subject/Comment**	**Old**	**New**
Jun 2001	S_12	SP-010283	--	--	Approved at TSG SA #12 and placed under Change Control	2.0.0	4.0.0
Sep 2001	S_13	SP-010476	001	--	Correction of invokeIdentifier usage	4.0.0	4.1.0
Mar 2002	S_15	SP-020019	002	--	Correction of erroneous CORBA module names and mapping tables	4.1.0	4.2.0
Mar 2002	S_15	SP-020019	003	--	Corrections to Basic CM IRP CORBA Solution Set IDLs	4.1.0	4.2.0
Mar 2002	S_15	SP-020038	004	--	Addition of missing CORBA exception "ManagedGenericIRPSystem::ValueNotSupported" onto CORBA method "find_managed_objects"	4.1.0	4.2.0
Jun 2002	S_16	SP-020294	005	--	Correcting IDL definitions of notification structured event Name Value pair names	4.2.0	4.3.0
Jul 2002	--	--	--	--	Updated the Version number (420->431) and the Date on the cover page	4.3.0	4.3.1
Sep 2002	S_17	SP-020483	006	--	Add Active Basic CM feature - CORBA Solution Set	4.3.1	5.0.0
Mar 2003	S_19	SP-030139	007	--	Add CORBA equivalents to IS operations "get{Operation Notification}Profile" - alignment with 32.602 & 32.312	5.0.0	5.1.0
Mar 2003	S_19	SP-030139	008	--	Correction of IDL errors	5.0.0	5.1.0
Mar 2003	S_19	SP-030144	009	--	Add description for notifications of each activeCM operation and one exception for createMO - alignment with 32.602, Information Service	5.0.0	5.1.0
Jun 2003	S_20	SP-030279	010	--	Alignment with Basic CM IRP information service (32.602) - add one exception for the operation createMO	5.1.0	5.2.0
Mar 2004	S_23	SP-040105	--	--	Automatic upgrade to Rel-6 (no CR)	5.2.0	6.0.0
Sep 2004	S_25	SP-040567	012	--	Removal of Rules for NRM extensions - Align with 32.622 (Generic NRM IS)	6.0.0	6.1.0
Sep 2004	S_25	SP-040566	014	--	Removal of unused/duplicate definition of types MOReference and MOReferenceSet	6.0.0	6.1.0
Dec 2004	S_26	SP-040806	015	--	Align the IDL style in the CORBA SS with the IDL Style Guide in 32.150	6.1.0	6.2.0

3GPP TSG-SA5 (Telecom Management) **S5-056076**
Meeting #41, Lisbon, PORTUGAL, 24-28 January 2005

CR-Form-v7

CHANGE REQUEST

⌘	**32.604**	**CR**	**005**	⌘ **rev**	**-**	⌘	Current version:	**6.0.0**	⌘

*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	☒

Title:	⌘	Generic System Context, update of reference to IS specification		
Source:	⌘	**SA5** (Ericsson, thomas.tovinger@ericsson.com)		
Work item code:	⌘	OAM-NIM	***Date:*** ⌘	28/1/2005
Category:	⌘	**F**	***Release:*** ⌘	Rel-6
		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:	⌘	The Information Service (IS) for this IRP is being updated due to an approved CR (to introduce the Generic System Context).
Summary of change:	⌘	Update the reference in Scope to the new latest IS version.
Consequences if not approved:	⌘	Wrong reference in Scope to the IS version.

Clauses affected:	⌘	Scope				
		Y	**N**			
Other specs affected:	⌘		**X**	Other core specifications	⌘	
			X	Test specifications		
			X	O&M Specifications		
Other comments:	⌘	This CR should only be approved if the corresponding CR on the IS to introduce the Generic System Context is approved (see the related CR collection document for an overview of all involved CR Tdoc numbers).				

Change in Clause Scope

1 Scope

The present document specifies the Common Management Information Protocol (CMIP) Solution Set (SS) for the Basic CM Integration Reference Point (IRP): Information Service defined in TS 32.602 [6]. In detail:

- Clause 4 provides the basic concept of the CMIP SS and the mapping between the IOCs, operations and notifications defined in TS 32.602 (Basic Configuration Management IRP: Information Service) [6] to the corresponding CMIP SS equivalents.
- Clause 5 contains the GDMO definitions for the Basic Configuration Management IRP over the CMIP interfaces,
- Clause 6 contains the ASN.1 definitions supporting the GDMO definitions provided in clause 5.

This Solution Set specification is related to 3GPP TS 32.602 V6.1~~0~~.X.

End of Change in Clause Scope

Annex A (informative): Change history

Change history							
Date	**TSG #**	**TSG Doc.**	**CR**	**Rev**	**Subject/Comment**	**Old**	**New**
Jun 2001	S_12	SP-010283	--	--	Approved at TSG SA #12 and placed under Change Control	2.0.0	4.0.0
Sep 2001	S_13	SP-010478	001	--	Correction due to TS renumbering	4.0.0	4.1.0
Sep 2001	S_13	SP-010476	002	--	Correction of invokeIdentifier usage	4.0.0	4.1.0
Dec 2001	S_14	SP-010643	003	--	Alignment with ITU-T Rec. X.710 (CMISE) 1997	4.1.0	4.2.0
Dec 2002	S_18	SP-020749	004	--	Alignment of the CMIP SS with the Rel-5 version of the IS in 32.602	4.2.0	5.0.0
Mar 2004	S_23	SP-040105	--	--	Automatic upgrade to Rel-6 (no CR)	5.0.0	6.0.0