ETSI TS 132 393 V9.0.0 (2010-02)

Technical Specification

Digital cellular telecommunications system (Phase 2+);
Universal Mobile Telecommunications System (UMTS);
LTE;
Telecommunication management;
Delta synchronization Integration Reference Point (IRP);
Common Object Request Broker Architecture (CORBA)
Solution Set (SS)
(3GPP TS 32.393 version 9.0.0 Release 9)

Reference
RTS/TSGS-0532393v900

Keywords
GSM, LTE, UMTS

ETSI

650 Route des Lucioles
F-06921 Sophia Antipolis Cedex - FRANCE

Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16

Siret N° 348 623 562 00017 - NAF 742 C
Association à but non lucratif enregistrée à la
Sous-Préfecture de Grasse (06) N° 7803/88

Intellectual Property Rights

IPRs essential or potentially essential to the present document may have been declared to ETSI. The information pertaining to these essential IPRs, if any, is publicly available for **ETSI members and non-members**, and can be found in ETSI SR 000 314: *"Intellectual Property Rights (IPRs); Essential, or potentially Essential, IPRs notified to ETSI in respect of ETSI standards"*, which is available from the ETSI Secretariat. Latest updates are available on the ETSI Web server (http://webapp.etsi.org/IPR/home.asp).

Pursuant to the ETSI IPR Policy, no investigation, including IPR searches, has been carried out by ETSI. No guarantee can be given as to the existence of other IPRs not referenced in ETSI SR 000 314 (or the updates on the ETSI Web server) which are, or may be, or may become, essential to the present document.

Foreword

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

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

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

Contents

Foreword

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

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

Version x.y.z

where:

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

Introduction

The present document is part of a TS-family covering the 3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; Telecommunication management; as identified below:

32.391: Delta Synchronization Integration Reference Point (IRP); Requirements.

32.392: Delta Synchronization Integration Reference Point (IRP); Information Service (IS).

32.393: Delta Synchronization Integration Reference Point (IRP); Common Object Request Broker Architecture (CORBA) Solution Set.

32.395: Delta Synchronization Integration Reference Point (IRP); eXtensible Markup Language (XML) file format definition.

The Itf-N interface is built up by a number of IRPs and a related Name Convention, which realise the functional capabilities over this interface. The basic structure of the IRPs is defined in 3GPP TS 32.101 [1] and 3GPP TS 32.102 [2].

IRPManagers (typically Network Management Systems) and IRPAgents (typically EMs or NEs) synchronize their data concerning alarms or configuration data. In certain scenarios this synchronization is lost or not done. This IRP provides functionality to significantly reduces the amount of data which needs to be transferred in order to re-establish synchronization.

1 Scope

The purpose of Delta Synchronization IRP is to define an interface through which an IRPManager can request only those data which changed (i.e. changed, were created or deleted) from a synchronization point onwards.

The present document is the "CORBA Solution Set" of Delta Synchronization IRP for the IRP whose semantics is specified in Delta Synchronization IRP: Information Service (3GPP TS 32.392 [5]).

2 References

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

- References are either specific (identified by date of publication, edition number, version number, etc.) or non-specific.
- For a specific reference, subsequent revisions do not apply.
- For a non-specific reference, the latest version applies. In the case of a reference to a 3GPP document (including a GSM document), a non-specific reference implicitly refers to the latest version of that document in the same Release as the present document.

[1] 3GPP TS 32.101: "Telecommunication management; Principles and high level requirements".

[2] 3GPP TS 32.102: "Telecommunication management; Architecture".

[3] 3GPP TS 32.302: "Telecommunication management; Configuration Management (CM); Notification Integration Reference Point (IRP): Information Service (IS)".

[4] 3GPP TS 32.391: "Configuration Management (CM); Delta Synchronization Integration Reference Point (IRP): Requirements".

[5] 3GPP TS 32.392: "Configuration Management (CM); Delta Synchronization Integration Reference Point (IRP): Information Service (IS)".

[6] 3GPP TS 32.622: "Telecommunication management; Configuration Management (CM); Generic network resources Integration Reference Point (IRP): Network Resource Model (NRM)".

[7] 3GPP TS 32.312: "Telecommunication management; Generic Integration Reference Point (IRP) management; Information Service (IS)".

[8] 3GPP TS 32.602: "Telecommunication management; Configuration Management (CM); Basic CM Integration Reference Point (IRP) management; Information Service (IS)".

[9] 3GPP TS 32.303: "Telecommunication management; Configuration Management (CM); Notification Integration Reference Point (IRP): Common Object Request Broker Architecture (CORBA) Solution Set (SS)".

[10] OMG TC Document telecom/98-11-01: "OMG Notification Service". http://www.omg.org/technology/documents/

3 Definitions and abbreviations

3.1 Definitions

For the purposes of the present document, the terms and definitions given in 3GPP TS 32.101 [1], 3GPP TS 32.102 [2] and 3GPP TS 32.391 [4] apply.

IRP: See 3GPP TS 32.101 [1].

IRPAgent: See 3GPP TS 32.102 [2].

IRPManager: See 3GPP TS 32.102 [2].

Changed: See 3GPP TS 32.391 [4].

Changed instance: See 3GPP TS 32.391 [4].

Delta Synchronisation: See 3GPP TS 32.391 [4].

Delta Synchronisation Point: See 3GPP TS 32.391 [4].

Full Synchronisation: See 3GPP TS 32.391 [4].

3.2 Abbreviations

For the purposes of the present document, the following abbreviations apply:

EM	Element Manager
IRP	Integration Reference Point
IS	Information Service (see 3GPP TS 32.101 [1])
Itf-N	Interface N
NE	Network Element
SS	Solution Set

4 Architectural features

The overall architectural feature of Delta Synchronization IRP is specified in 3GPP TS 32.382 [5].

5 Mapping

5.1 General mappings

Not applicable.

5.2 Operation and notification mapping

The Delta Synchronization IS defines semantics of operations visible across the Itf-N. Table 5.2-1 indicates mapping of these operations and notifications to their equivalents defined in this SS.

Table 5.2-1: Mapping from IS Operation to SS equivalents

IS Operation / Notification (3GPP TS 32.392)	SS Method	Qualifier
manageDeltaSynchronization	manageDeltaSynchronization	M
getAvailableDeltaSynchPoints	getAvailableDeltaSynchPoints	O
triggerDeltaSynchOfCMData	triggerDeltaSynchOfCMData	O
triggerDeltaSynchOfAlarms	triggerDeltaSynchOfAlarms	O
notifyStatusOfDeltaSynchronization	notifyStatusOfDeltaSynchronization	M
notifyNewDeltaSynchPoint	notifyNewDeltaSynchPoint	O

5.3 Operation parameter mapping

The Delta Synchronization IS defines semantics of parameters carried in operations across the Itf-N. The following tables indicate the mapping of these parameters, as per operation, to their equivalents defined in this SS.

Table 5.3-1: Mapping from IS `manageDeltaSynchronization` parameters to SS equivalents

IS Operation parameter	SS Method parameter	Qualifier
managerReference	DeltaSynchronizationConstDefs::ManagerReference	M
manageDeltaSynchForAlarmData	DeltaSynchronizationConstDefs::ManageDeltaSynchForXDataConditional	CM
manageDeltaSynchForCMData	DeltaSynchronizationConstDefs::ManageDeltaSynchForXDataConditional	CM
status	Exceptions: DeltaSynchronizationConstDefs::ManageDeltaSynchronization, GenericIRPManagementSystem::ParameterNotSupported, GenericIRPManagementSystem::InvalidParameter, GenericIRPManagementSystem::ValueNotSupported, GenericIRPManagementSystem::OperationNotSupported	M

Table 5.3-2: Mapping from IS `getAvailableDeltaSynchPoints` parameters to SS equivalents

IS Operation parameter	SS Method parameter	Qualifier
managerReference	DeltaSynchronizationConstDefs::ManagerReferenceOpt	O
synchPointsForCMDataRequested	DeltaSynchronizationConstDefs::SynchPointsRequestedConditional	CM
synchPointsForAlarmDataRequested	DeltaSynchronizationConstDefs::SynchPointsRequestedConditional	CM
synchPointListForAlarms	DeltaSynchronizationConstDefs::SynchPointListConditional	CM
synchPointListForCMData	DeltaSynchronizationConstDefs::SynchPointListConditional	CM
status	Exceptions: DeltaSynchronizationConstDefs::DeltaSynchNotSupportedForCMData, DeltaSynchronizationConstDefs::DeltaSynchNotSupportedForAlarmData, DeltaSynchronizationConstDefs::DeltaSynchNotActive, DeltaSynchronizationConstDefs::DeltaSynchForCMDataDeactivated, DeltaSynchronizationConstDefs::DeltaSynchForAlarmDataDeactivated, GenericIRPManagementSystem::ParameterNotSupported, GenericIRPManagementSystem::InvalidParameter, GenericIRPManagementSystem::ValueNotSupported, GenericIRPManagementSystem::OperationNotSupported	M

Table 5.3-3: Void

Table 5.3-4: Mapping from IS `triggerDeltaSynchOfCMData` parameters to SS equivalents

IS Operation parameter	SS Method parameter	Qualifier
managerReference	DeltaSynchronizationConstDefs::ManagerReferenceOpt	O
dataRequested	DeltaSynchronizationConstDefs::CMDataRequestedOpt	O
baseMOInstance	KernelCmConstDefs::DNOpt	O
scope	KernelCmConstDefs::ScopeTypeOpt	O
synchPoint	DeltaSynchronizationConstDefs::SynchPoint	M
deltaLists	DeltaSynchronizationConstDefs::DeltaListsConditional	CM
newSynchPoint	DeltaSynchronizationConstDefs::SynchPointConditional	CM
status	Exceptions: DeltaSynchronizationConstDefs::TriggerDeltaSynchOfCMData, DeltaSynchronizationConstDefs::SynchronizationPointTooLongAgo, DeltaSynchronizationConstDefs::TooManyChangesFullSynchronizationRecommended DeltaSynchronizationConstDefs::SynchPointUnknown, DeltaSynchronizationConstDefs::DeltaSynchNotSupportedForCMData, DeltaSynchronizationConstDefs::DeltaSynchForCMDataDeactivated, DeltaSynchronizationConstDefs::DeltaSynchNotActive, GenericIRPManagementSystem::ParameterNotSupported, GenericIRPManagementSystem::InvalidParameter, GenericIRPManagementSystem::ValueNotSupported, GenericIRPManagementSystem::OperationNotSupported	M

Table 5.3-5: Mapping from IS `triggerDeltaSynchOfAlarms` parameters to SS equivalents

IS Operation parameter	SS Method parameter	Qualifier
managerReference	DeltaSynchronizationConstDefs::ManagerReference	M
dataRequested	DeltaSynchronizationConstDefs::AlarmDataRequested	M
baseMOInstance	KernelCmConstDefs::DN	O
scope	KernelCmConstDefs::ScopeTypeOpt	O
synchPoint	DeltaSynchronizationConstDefs::SynchPoint	M
deltaLists	DeltaSynchronizationConstDefs::DeltaListsConditional	CM
newSynchPoint	DeltaSynchronizationConstDefs::SynchPointConditional	CM
status	Exceptions: DeltaSynchronizationConstDefs::TriggerDeltaSynchOfAlarms, DeltaSynchronizationConstDefs::SynchronizationPointTooLongAgo, DeltaSynchronizationConstDefs::TooManyChangesFullSynchronizationRecommended DeltaSynchronizationConstDefs::SynchPointUnknown, DeltaSynchronizationConstDefs::DeltaSynchNotSupportedForAlarms, DeltaSynchronizationConstDefs::DeltaSynchForAlarmsNotActive, DeltaSynchronizationConstDefs::DeltaSynchNotActive, GenericIRPManagementSystem::ParameterNotSupported, GenericIRPManagementSystem::InvalidParameter, GenericIRPManagementSystem::ValueNotSupported, GenericIRPManagementSystem::OperationNotSupported	M

5.4 Notification parameter mapping

The delta synchronization Information Service defines semantics of parameters carried in notifications. The following table indicates the mapping of these parameters to their OMG CORBA Structured Event (defined in OMG Notification Service [n2!!]) equivalents. The composition of OMG Structured Event, as defined in the OMG Notification Service [n2!!], is:

```
Header
      Fixed Header
           domain_name
           type_name
           event_name
      Variable Header
Body
      filterable_body_fields
      remaining_body
```

The following tables list all OMG Structured Event attributes in the second column. The first column identifies the delta synchronization Information Service defined notification parameters.

Table 5.4-1: Mapping for `notifyStatusOfDeltaSynchronization`

IS Parameters	OMG CORBA Structured Event attribute	Qualifier	Comment
There is no corresponding IS attribute.	domain_name	M	It carries the IRP document version number string. See clause 3.1. It indicates the syntax and semantics of the Structured Event as defined by the present document.
notificationType	type_name	M	This is the NotifyDeltaSynchOfInstancesDeactivated of module DeltaSynchronizationNotifications.
There is no corresponding IS attribute.	event_name	M	It carries no information.
There is no corresponding IS attribute.	Variable Header		
objectClass, objectInstance	One NV pair of filterable_body_fields	M	NV stands for name-value pair. Order arrangement of NV pairs is not significant. The name of NV-pair is always encoded in string. Name of this NV pair is the MANAGED_OBJECT_INSTANCE of interface AttributeNameValue of module NotificationIRPConstDefs. Value of NV pair is a string. See corresponding table in Notification IRP: CORBA SS (3GPP TS 32.303 [1]).
notificationId	One NV pair of remaining_body	M	Name of NV pair is the NOTIFICATION_ID of interface AttributeNameValue of module NotificationIRPConstDefs. Value of NV pair is a long. See corresponding table in Notification IRP: CORBA SS (3GPP TS 32.303 [1]).
eventTime	One NV pair of filterable_body_fields	M	Name of NV pair is the EVENT_TIME of interface AttributeNameValue of module NotificationIRPConstDefs. Value of NV pair is IRPTime. See corresponding table in Notification IRP: CORBA SS (3GPP TS 32.303 [1]).
systemDN	One NV pair of filterable_body_fields	M	Name of NV pair is the SYSTEM_DN of interface AttributeNameValue of module NotificationIRPConstDefs. Value of NV pair is a string. See corresponding table in Notification IRP: CORBA SS (3GPP TS 32.303 [1]).
managerReference	One NV pair of remaining_body	M	Name of NV pair is the MANAGER_REFERENCE of interface notifyDeltaSynchOfInstancesDeactivated of module DeltaSynchronizationNotifications. Value of NV pair is ManagerReference of module DeltaSynchronizationConstDefs.
deltaSynchStatusForCMData	One NV pair of remaining_body	M	Name of NV pair is the DELTA_SYNCH_STATUS_FOR_CMDATA of interface notifyDeltaSynchOfInstancesDeactivated of module DeltaSynchronizationNotifications. Value of NV pair is DeltaSynchStatus of module DeltaSynchronizationConstDefs.
deltaSynchStatusForAlarmData	One NV pair of remaining_body	M	Name of NV pair is the DELTA_SYNCH_STATUS_FOR_ALARM_DATA of interface notifyDeltaSynchOfInstancesDeactivated of module DeltaSynchronizationNotifications. Value of NV pair is DeltaSynchStatus of module DeltaSynchronizationConstDefs.

Table 5.4-2: Mapping for `notifyNewDeltaSynchPoint`

IS Parameters	OMG CORBA Structured Event attribute	Qualifier	Comment
There is no corresponding IS attribute.	domain_name	M	It carries the IRP document version number string. See subclause 3.1. It indicates the syntax and semantics of the Structured Event as defined by the present document.
notificationType	type_name	M	This is the NotifyDeltaSynchOfAlarmsDeactivated of module DeltaSynchronizationNotifications.
There is no corresponding IS attribute.	event_name	M	It carries no information.
There is no corresponding IS attribute.	Variable Header		
objectClass, objectInstance	One NV pair of filterable_body_fields	M	NV stands for name-value pair. Order arrangement of NV pairs is not significant. The name of NV-pair is always encoded in string. Name of this NV pair is the MANAGED_OBJECT_INSTANCE of interface AttributeNameValue of module NotificationIRPConstDefs. Value of NV pair is a string. See corresponding table in Notification IRP: CORBA SS (3GPP TS 32.303 [1]).
notificationId	One NV pair of remaining_body	M	Name of NV pair is the NOTIFICATION_ID of interface AttributeNameValue of module NotificationIRPConstDefs. Value of NV pair is a long. See corresponding table in Notification IRP: CORBA SS (3GPP TS 32.303 [1]).
eventTime	One NV pair of filterable_body_fields	M	Name of NV pair is the EVENT_TIME of interface AttributeNameValue of module NotificationIRPConstDefs. Value of NV pair is IRPTime. See corresponding table in Notification IRP: CORBA SS (3GPP TS 32.303 [1]).
systemDN	One NV pair of filterable_body_fields	M	Name of NV pair is the SYSTEM_DN of interface AttributeNameValue of module NotificationIRPConstDefs. Value of NV pair is a string. See corresponding table in Notification IRP: CORBA SS (3GPP TS 32.303 [1]).
newSynchPoint	One NV pair of remaining_body	M	Name of NV pair is the NEW_DELTA_SYNCH_POINT of interface notifyDeltaSynchOfAlarmsDeactivated of module DeltaSynchronizationNotifications. Value of NV pair is SynchPoint of module DeltaSynchronizationConstDefs.
requestedSynchPoint	One NV pair of remaining_body	M	Name of NV pair is the REQUESTED_SYNCH_POINT of interface notifyDeltaSynchOfAlarmsDeactivated of module DeltaSynchronizationNotifications. Value of NV pair is SynchPoint of module DeltaSynchronizationConstDefs.
deltaSynchPointType	One NV pair of remaining_body	M	Name of NV pair is the DELTA_SYNCH_POINT_TYPE of interface notifyDeltaSynchOfAlarmsDeactivated of module DeltaSynchronizationNotifications. Value of NV pair is DeltaSynchPointType of module DeltaSynchronizationConstDefs.

triggeredByAgentOrManager	One NV pair of remaining_body	M	Name of NV pair is the TRIGGERED_BY_AGENT_OR_MANAGER of interface notifyDeltaSynchOfAlarmsDeactivated of module DeltaSynchronizationNotifications. Value of NV pair is TriggeredByAgentOrManager of module DeltaSynchronizationConstDefs.
agentOrManagerReference	One NV pair of remaining_body	M	Name of NV pair is the AGENT_OR_MANAGER_REFERENCE of interface notifyDeltaSynchOfAlarmsDeactivated of module DeltaSynchronizationNotifications. Value of NV pair is AgentOrManagerReference of module DeltaSynchronizationConstDefs.

Annex A (normative): IDL specifications

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

```
// File: DeltaSynchronizationConstDefs.idl
#ifndef _DELTA_SYNCHRONIZATION_CONST_DEFS_IDL_
#define _DELTA_SYNCHRONIZATION_CONST_DEFS_IDL_

#include <TimeBase.idl>

#include <DeltaSynchronizationConstDefs.idl>
#include <GenericIRPManagementConstDefs.idl>
#include <KernelCmConstDefs.idl>
#include <FileTransferIRPConstDefs.idl>

//FileTransferIRPConstDefs::FileLocation value;


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


/* ## Module: DeltaSynchronizationConstDefs */

module DeltaSynchronizationConstDefs
{

/*****************************************************************************/
/* definition of types used in several operations for Delta Synchronization: */
/*****************************************************************************/

/* types used in several operations: */

typedef string ManagerReference;

typedef string AgentOrManagerReference;

/*
ManagerReferenceConditional is a type carrying an optional parameter.
The boolean shall be TRUE, if the condition described in TS 32.392 is fulfilled. In this case the
value is present. Otherwise the value is be absent.
*/
union ManagerReferenceConditional switch (boolean)
      {
      case TRUE: ManagerReference value;
      };

/*
ManagerReferenceOpt is a type carrying an optional parameter.
The boolean shall be TRUE, if the operation request uses this parameter. In this case the value is
present. Otherwise the value is absent.
*/
union ManagerReferenceOpt switch (boolean)
      {
      case TRUE: ManagerReference value;
      };
```

```
enum ManageDeltaSynchForXData {ACTIVATE, DEACTIVATE};

/*
ManageDeltaSynchForXDataConditional is a type carrying a conditional parameter.
The boolean shall be TRUE, if the condition described in TS 32.392 is fulfilled. In this case the
value is present. Otherwise the value is absent.
*/
union ManageDeltaSynchForXDataConditional switch (boolean)
      {
      case TRUE: ManageDeltaSynchForXData value;
      };


typedef TimeBase::UtcT SynchPoint;

/*
SynchPointConditional is a type carrying a conditional parameter.
The boolean shall be TRUE, if the condition described in TS 32.392 is fulfilled. In this case the
value is present. Otherwise the value is absent.
*/
union SynchPointConditional switch (boolean)
      {
      case TRUE: SynchPoint value;
      };

/*
SynchPointOpt is a type carrying an optional parameter.
The boolean shall be TRUE, if the operation request uses this parameter. In this case the value is
present. Otherwise the value is absent.
*/
union SynchPointOpt switch (boolean)
      {
      case TRUE: SynchPoint value;
      };

/*
ScopeTypeOpt is a type carrying an optional parameter.
The boolean shall be TRUE, if the operation request uses this parameter. In this case the value is
present. Otherwise the value is absent.
*/
union ScopeTypeOpt switch (boolean)
      {
      case TRUE: KernelCmConstDefs::ScopePara value;
      };

/*
BaseMOInstanceOpt is a type carrying an optional parameter.
The boolean shall be TRUE, if the operation request uses this parameter. In this case the value is
present. Otherwise the value is absent.
*/
union BaseMOInstanceOpt switch (boolean)
      {
      case TRUE: GenericIRPManagementConstDefs::DN value;
      };


enum Status {SUCCESS, FAILURE};


/*****************************************************************/
/* types used in operation manageDeltaSynchronization            */
/*****************************************************************/

enum ActivatedStatus {ACTIVATED, DEACTIVATED };
typedef ActivatedStatus ManageDeltaSynchMode;



/*****************************************************************/
/* types used in operation getAvailableDeltaSynchPoints          */
/*****************************************************************/

typedef boolean SynchPointsRequested;

/*
SynchPointsRequestedConditional is a type carrying a conditional parameter.
The boolean shall be TRUE, if the condition described in TS 32.392 is fulfilled. Otherwise the value
may be absent.
```

```
*/
union SynchPointsRequestedConditional switch (boolean)
      {
      case TRUE: SynchPointsRequested value;
      };


typedef sequence <SynchPoint> SynchPointList;

/*
SynchPointListConditional is a type carrying an optional parameter.
The boolean shall be TRUE, if the condition described in TS 32.392 is fulfilled. In this case the
value is present. Otherwise the value is be absent.
*/
union SynchPointListConditional switch (boolean)
      {
      case TRUE: SynchPointList value;
      };



/*********************************************************************/
/* types used in operation triggerDeltaSynchOfCMData and             */
/*            in operation triggerDeltaSynchOfAlarmData              */
/*********************************************************************/


/*
AttributeListConditional is a type carrying a conditional parameter.
The boolean shall be TRUE, if the operation's dnsOnly=FALSE. In this case the value is present.
Otherwise the value is absent.
*/
union AttributeListConditional switch (boolean)
   {
   case TRUE: GenericIRPManagementConstDefs::MOAttributeSet value;
   };


struct ListedInstance
   {
   GenericIRPManagementConstDefs::DN moInstance;   /* DN is a string; */
   AttributeListConditional attributeList;
   };

typedef sequence <ListedInstance> ListOfInstances;

struct DeltaListsWithRealLists
   {
   TimeBase::UtcT startTime;
   TimeBase::UtcT endTime;
   ListOfInstances listOfCreatedInstances;
   ListOfInstances listOfChangedInstances;
   ListOfInstances listOfDeletedInstances;
   };


struct AlarmDeltaListsWithRealLists
   {
   TimeBase::UtcT startTime;
   TimeBase::UtcT endTime;
   AlarmIRPConstDefs::AlarmInformationSeq listOfNewAlarms;
   AlarmIRPConstDefs::AlarmInformationSeq listOfChangedAlarms;
   AlarmIRPConstDefs::AlarmInformationIdSeq listOfDeletedAlarms;
   };


typedef sequence <FileTransferIRPConstDefs::FileLocation> FileLocationList;

struct DeltaListsWithFileReferences
   {
   TimeBase::UtcT startTime;
   TimeBase::UtcT endTime;
   FileTransferIRPConstDefs::FileLocationList fileList;
/* if several files are used, then they shall be processed by the IRPmanager in sequence, i.e. first
file first, second file as second, … */
   };
```

```
enum DeltaListContentChoice {REAL_LISTS, FILE_REFERENCES};

//The CmDeltaList may contain a list of ListOfInstances or a list of filenames
   union CmDeltaLists switch (DeltaListContentChoice)
   {
      case REAL_LISTS: DeltaListsWithRealLists deltaListRealLists;
      case FILE_REFERENCES: DeltaListsWithFileReferences deltaListFileReferences;
   };


//The AlarmDeltaLists may contain a list of ListOfInstances or a list of filenames
   union AlarmDeltaLists switch (DeltaListContentChoice)
   {
      case REAL_LISTS: AlarmDeltaListsWithRealLists deltaListRealLists;
      case FILE_REFERENCES: DeltaListsWithFileReferences deltaListFileReferences;
   };

/*
CmDeltaListsConditional is a type carrying a conditional parameter.
The boolean shall be TRUE, if the condition described in TS 32.392 is fulfilled. In this case
the value is present. Otherwise the value is be absent.
*/
union CmDeltaListsConditional switch (boolean)
   {
   case TRUE: CmDeltaLists value;
   };

/*
AlarmDeltaListsConditional is a type carrying a conditional parameter.
The boolean shall be TRUE, if the condition described in TS 32.392 is fulfilled. In this case
the value is present. Otherwise the value is absent.
*/
union AlarmDeltaListsConditional switch (boolean)
   {
   case TRUE: AlarmDeltaLists value;
   };


/*********************************************************************/
/* types used in operation triggerDeltaSynchOfAlarmData              */
/*********************************************************************/

enum AlarmDataRequested { ALARM_IDS_ONLY, COMPLETE_ALARM_INFORMATION };

/*
AlarmDataRequestedOpt is a type carrying an optional parameter.
The boolean shall be TRUE, if the operation request uses this parameter. In this case the value
 is present. Otherwise the value is absent.
*/
union AlarmDataRequestedOpt switch (boolean)
      {
      case TRUE: AlarmDataRequested value;
      };


/*********************************************************************/
/* types used in operation triggerDeltaSynchOfCMData                 */
/*********************************************************************/

enum CMDataRequested { DNS_ONLY, COMPLETE_DATA_SET };

/*
CMDataRequestedOpt is a type carrying an optional parameter.
The boolean shall be TRUE, if the operation request uses this parameter. In this case the value
 is present. Otherwise the value is absent.
*/
union CMDataRequestedOpt switch (boolean)
      {
      case TRUE: CMDataRequested value;
      };



/*********************************************************************************/
/* definition of types in notifications for Delta Synchronization                */
/*********************************************************************************/

enum DeltaSynchPointType { DELTA_SYNCH_POINT_FOR_ALARM, DELTA_SYNCH_POINT_FOR_CM_DATA };
```

```
typedef ActivatedStatus DeltaSynchStatus;

enum TriggeredBy { IRP_AGENT, IRP_MANAGER };


/*********************************************************************************/
/* Definition of parameters specified in notifications for Delta Synchronization */
/*********************************************************************************/

interface AttributeNameValue
   {
   const string MANAGER_REFERENCE = "MANAGER_REFERENCE";
   const string AGENT_OR_MANAGER_REFERENCE = "AGENT_OR_MANAGER_REFERENCE";
   const string DELTA_SYNCH_STATUS_FOR_CMDATA = "DELTA_SYNCH_STATUS_FOR_CMDATA";
   const string DELTA_SYNCH_STATUS_FOR_ALARM_DATA = "DELTA_SYNCH_STATUS_FOR_ALARM_DATA";
   const string NEW_DELTA_SYNCH_POINT = "NEW_DELTA_SYNCH_POINT";
   const string DELTA_SYNCH_POINT_TYPE = "DELTA_SYNCH_POINT_TYPE";
   const string REQUESTED_SYNCH_POINT = "REQUESTED_SYNCH_POINT";
   const string TRIGGERED_BY_AGENT_OR_MANAGER = "TRIGGERED_BY_AGENT_OR_MANAGER";
   };


};

#endif // _DELTA_SYNCHRONIZATION_CONST_DEFS_IDL_
```

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

```
//File: DeltaSynchronizationSystem.idl
#ifndef _DELTA_SYNCHRONIZATION_SYSTEM_IDL_
#define _DELTA_SYNCHRONIZATION_SYSTEM_IDL_

#include <KernelCmConstDefs.idl>
#include <DeltaSynchronizationConstDefs.idl>
#include <GenericIRPManagementSystem.idl>
#include <AlarmIRPConstDefs.idl>
#include <AlarmIRPSystem.idl>
#include <NotificationLogIRPSystem.idl>

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

/* ## Module: DeltaSynchronizationSystem */

module DeltaSynchronizationSystem
{

   /*
   If the system fails to complete an operation, then it can provide a reason
   to qualify the exception. The semantics carried in this reason are outside
   the scope of the present document.
   */

   exception ManageDeltaSynchronization { string reason; };
   exception GetAvailableDeltaSynchPoints { string reason; };
   exception TriggerDeltaSynchOfCMData { string reason; };
   exception TriggerDeltaSynchOfAlarms { string reason; };
   exception SynchronizationPointTooLongAgo { string reason; };
   exception TooManyChangesFullSynchronizationRecommended { string reason; };
   exception DeltaSynchNotSupportedForCMData { string reason; };
   exception DeltaSynchNotSupportedForAlarmData { string reason; };
   exception DeltaSynchNotActive { string reason; };
   exception DeltaSynchForCMDataDeactivated { string reason; };
   exception DeltaSynchForAlarmDataDeactivated { string reason; };
   exception SynchPointTooLongAgo{ string reason; };
   exception SynchPointUnknown { string reason; };
   exception DeltaSynchNotSupportedForAlarms { string reason; };
   exception DeltaSynchForAlarmsNotActive { string reason; };


   interface DeltaSynchGenericParts
   {

      DeltaSynchronizationConstDefs::Status manageDeltaSynchronization
      /* for the purpose of this operation see 3GPP TS 32.392 */
         (
         in DeltaSynchronizationConstDefs::ManagerReference managerReference,
         in DeltaSynchronizationConstDefs::ManageDeltaSynchForXDataConditional
             manageDeltaSynchForAlarmData,
         in DeltaSynchronizationConstDefs::ManageDeltaSynchForXDataConditional
             manageDeltaSynchForCMData
         )
      raises
         (
         ManageDeltaSynchronization,
         GenericIRPManagementSystem::ParameterNotSupported,
         GenericIRPManagementSystem::InvalidParameter,
         GenericIRPManagementSystem::ValueNotSupported,
         GenericIRPManagementSystem::OperationNotSupported
         );


      DeltaSynchronizationConstDefs::Status getAvailableDeltaSynchPoints
      /* for the purpose of this operation see 3GPP TS 32.392 */
         (
         in DeltaSynchronizationConstDefs::ManagerReferenceOpt managerReference,
         in DeltaSynchronizationConstDefs::SynchPointsRequestedConditional
            synchPointsForCMDataRequested,
         in DeltaSynchronizationConstDefs::SynchPointsRequestedConditional
```

```
         synchPointsForAlarmDataRequested,
      out DeltaSynchronizationConstDefs::SynchPointListConditional synchPointListForAlarms,
      out DeltaSynchronizationConstDefs::SynchPointListConditional synchPointListForCMData
      )
   raises
      (
      GetAvailableDeltaSynchPoints,
      DeltaSynchNotSupportedForCMData,
      DeltaSynchNotSupportedForAlarmData,
      DeltaSynchNotActive,
      DeltaSynchForCMDataDeactivated,
      DeltaSynchForAlarmDataDeactivated,
      GenericIRPManagementSystem::ParameterNotSupported,
      GenericIRPManagementSystem::InvalidParameter,
      GenericIRPManagementSystem::ValueNotSupported,
      GenericIRPManagementSystem::OperationNotSupported
      );


};



interface DeltaSynchOfCMData
{

   DeltaSynchronizationConstDefs::Status triggerDeltaSynchOfCMData
   /* for the purpose of this operation see 3GPP TS 32.392 */
      (
      in DeltaSynchronizationConstDefs::ManagerReferenceOpt managerReference,
      in DeltaSynchronizationConstDefs::CMDataRequestedOpt cmDataRequested,
      in DeltaSynchronizationConstDefs::BaseMOInstanceOpt baseMOInstance,
      in DeltaSynchronizationConstDefs::ScopeTypeOpt scope,
      in DeltaSynchronizationConstDefs::SynchPoint synchPoint,
      out DeltaSynchronizationConstDefs::CmDeltaListsConditional deltaLists,
      out DeltaSynchronizationConstDefs::SynchPointConditional newSynchPoint
      )
   raises
      (
      TriggerDeltaSynchOfCMData,
      SynchronizationPointTooLongAgo,
      TooManyChangesFullSynchronizationRecommended,
      SynchPointUnknown,
      DeltaSynchNotSupportedForCMData,
      DeltaSynchForCMDataDeactivated,
      DeltaSynchNotActive,
      GenericIRPManagementSystem::ParameterNotSupported,
      GenericIRPManagementSystem::InvalidParameter,
      GenericIRPManagementSystem::ValueNotSupported,
      GenericIRPManagementSystem::OperationNotSupported
      );

};



interface DeltaSynchOfAlarmData
{

   DeltaSynchronizationConstDefs::Status triggerDeltaSynchOfAlarms
   /* for the purpose of this operation see 3GPP TS 32.392 */
      (
      in DeltaSynchronizationConstDefs::ManagerReferenceOpt managerReference,
      in DeltaSynchronizationConstDefs::AlarmDataRequestedOpt alarmDataRequested,
      in DeltaSynchronizationConstDefs::BaseMOInstanceOpt baseMOInstance,
      in DeltaSynchronizationConstDefs::ScopeTypeOpt scope,
      in DeltaSynchronizationConstDefs::SynchPoint synchPoint,
      out DeltaSynchronizationConstDefs::AlarmDeltaListsConditional deltaLists,
      out DeltaSynchronizationConstDefs::SynchPointConditional newSynchPoint
      )
   raises
      (
      TriggerDeltaSynchOfAlarms,
      SynchronizationPointTooLongAgo,
      TooManyChangesFullSynchronizationRecommended,
      SynchPointUnknown,
      DeltaSynchNotSupportedForAlarms,
      DeltaSynchForAlarmsNotActive,
```

```
         DeltaSynchNotActive,
         GenericIRPManagementSystem::ParameterNotSupported,
         GenericIRPManagementSystem::InvalidParameter,
         GenericIRPManagementSystem::ValueNotSupported,
         GenericIRPManagementSystem::OperationNotSupported
         );

   };

   interface DeltaSynchIRPSystem : DeltaSynchGenericParts, DeltaSynchOfCMData,
      DeltaSynchOfAlarmData, GenericIRPManagementSystem::GenericIRPManagement{};

};
#endif // _DELTA_SYNCHRONIZATION_SYSTEM_IDL_
```

A.3 IDL specification (file name "DeltaSynchronizationNotifications.idl")

```
//File: DeltaSynchronizationNotifications.idl
#ifndef _DELTA_SYNCHRONIZATION_NOTIFICATIONS_IDL_
#define _DELTA_SYNCHRONIZATION_NOTIFICATIONS_IDL_

#include <DeltaSynchronizationConstDefs.idl>
#include <NotificationIRPNotifications.idl>

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

/* ## Module: DeltaSynchronizationNotifications
This contains the specification of notifications of Delta Synchronization.
=========================================================================
*/
module DeltaSynchronizationNotifications
{

   /* Constant definitions for the NotifyDeltaSynchOfInstancesDeactivated notification */

   interface notifyDeltaSynchOfInstancesDeactivated: NotificationIRPNotifications::Notify
   {
      const string EVENT_TYPE = "notifyStatusOfDeltaSynchronization";

      /**
      * This constant defines the name of the ManagerReference property,
      * which is transported in the filterable_body fields.
      * The data type for the value of this property is
      * DeltaSynchronizationConstDefs::ManagerReferenceConditional.
      */
      const string MANAGER_REFERENCE =
         DeltaSynchronizationConstDefs::AttributeNameValue::MANAGER_REFERENCE;

      /**
      * This constant defines the name of the DeltaSynchStatusForCMData property,
      * which is transported in the filterable_body fields.
      * The data type for the value of this property is
      * DeltaSynchronizationConstDefs::DeltaSynchStatus.
      */
      const string DELTA_SYNCH_STATUS_FOR_CMDATA =
         DeltaSynchronizationConstDefs::AttributeNameValue::DELTA_SYNCH_STATUS_FOR_CMDATA;

      /**
      * This constant defines the name of the DeltaSynchStatusForAlarmData property,
      * which is transported in the filterable_body fields.
      * The data type for the value of this property is
      * DeltaSynchronizationConstDefs::DeltaSynchStatus.
      */
      const string DELTA_SYNCH_STATUS_FOR_ALARM_CMDATA =
         DeltaSynchronizationConstDefs::AttributeNameValue::DELTA_SYNCH_STATUS_FOR_ALARM_DATA;

   };



   /* Constant definitions for the notifyNewDeltaSynchPoint notification */

   interface notifyNewDeltaSynchPoint: NotificationIRPNotifications::Notify
   {
      const string EVENT_TYPE = "notifyNewDeltaSynchPoint";

      /**
      * This constant defines the name of the AgentOrManagerReference property,
      * which is transported in the filterable_body fields.
      * The data type for the value of this property is
      * DeltaSynchronizationConstDefs::AgentOrManagerReference.
      */
      const string AGENT_OR_MANAGER_REFERENCE =
         DeltaSynchronizationConstDefs::AttributeNameValue::AGENT_OR_MANAGER_REFERENCE;

      /**
      * This constant defines the name of the NewDeltaSynchPoint property,
```

```
      * which is transported in the filterable_body fields.
      * The data type for the value of this property is
      * DeltaSynchronizationConstDefs::SynchPoint.
      */
      const string NEW_DELTA_SYNCH_POINT =
         DeltaSynchronizationConstDefs::AttributeNameValue::NEW_DELTA_SYNCH_POINT;

      /**
      * This constant defines the name of the RequestedSynchPoint property,
      * which is transported in the filterable_body fields.
      * The data type for the value of this property is
      * DeltaSynchronizationConstDefs::SynchPoint.
      */
      const string REQUESTED_SYNCH_POINT =
         DeltaSynchronizationConstDefs::AttributeNameValue::REQUESTED_SYNCH_POINT;

      /**
      * This constant defines the name of the DeltaSynchPointType property,
      * which is transported in the filterable_body fields.
      * The data type for the value of this property is
      * DeltaSynchronizationConstDefs:: DeltaSynchPointType.
      */
      const string DELTA_SYNCH_POINT_TYPE =
         DeltaSynchronizationConstDefs::AttributeNameValue::DELTA_SYNCH_POINT_TYPE;

      /**
      * This constant defines the name of the TriggeredByAgentOrManager property,
      * which is transported in the filterable_body fields.
      * The data type for the value of this property is
      * DeltaSynchronizationConstDefs::TriggeredBy.
      */
      const string TRIGGERED_BY_AGENT_OR_MANAGER =
         DeltaSynchronizationConstDefs::AttributeNameValue::TRIGGERED_BY_AGENT_OR_MANAGER;


   };


};

#endif // _DELTA_SYNCHRONIZATION_NOTIFICATIONS_IDL_
```

Annex B (informative): Change history

Change history								
Date	**TSG #**	**TSG Doc.**	**CR**	**R**	**Subject/Comment**	**Cat**	**Old**	**New**
Mar 2007	SA_35	SP-070054	--	--	Submitted to SA#35 for Information	--	1.0.0	
Jun 2007	SA_36	SP-070285	--	--	Submitted to SA#36 for Approval	--	2.0.0	7.0.0
Sep 2007	SA_37	SP-070612	0001	--	Remove operation setDeltaSynchPoint - Align with IS in 32.392	F	7.0.0	7.1.0
Sep 2007	SA_37	SP-070612	0002	--	Corrections to IDL Definitions	F	7.0.0	7.1.0
Dec 2008	SA_42	--	--	--	Upgrade to Release 8	--	7.1.0	8.0.0
Dec 2009	-	-	-	-	Upgrade to Release 9	--	8.0.0	9.0.0

History

Document history		
V9.0.0	February 2010	Publication