

3GPP TS 32.533 V9.5.0 (2010-12)

Technical Specification

**3rd Generation Partnership Project;
Technical Specification Group Services and System Aspects;
Telecommunication management;
Software management (SwM);
Integration Reference Point (IRP);
Common Object Request Broker Architecture (CORBA)
Solution Set (SS)
(Release 9)**



The present document has been developed within the 3rd Generation Partnership Project (3GPP™) and may be further elaborated for the purposes of 3GPP.

The present document has not been subject to any approval process by the 3GPP Organizational Partners and shall not be implemented. This Specification is provided for future development work within 3GPP only. The Organizational Partners accept no liability for any use of this Specification. Specifications and reports for implementation of the 3GPP™ system should be obtained via the 3GPP Organizational Partners' Publications Offices.

Keywords

Management, IRP

3GPP

Postal address

3GPP support office address

650 Route des Lucioles - Sophia Antipolis
Valbonne - FRANCE
Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16

Internet

<http://www.3gpp.org>

Copyright Notification

No part may be reproduced except as authorized by written permission.
The copyright and the foregoing restriction extend to reproduction in all media.

© 2010, 3GPP Organizational Partners (ARIB, ATIS, CCSA, ETSI, TTA, TTC).
All rights reserved.

UMTS™ is a Trade Mark of ETSI registered for the benefit of its members
3GPP™ is a Trade Mark of ETSI registered for the benefit of its Members and of the 3GPP Organizational Partners
LTE™ is a Trade Mark of ETSI currently being registered for the benefit of its Members and of the 3GPP Organizational Partners
GSM® and the GSM logo are registered and owned by the GSM Association

Contents

Foreword	4
Introduction	4
1 Scope	5
2 References.....	5
3 Definitions and abbreviations	5
3.1 Definitions	5
3.2 Abbreviations.....	5
4 Architectural Features	6
5 Mapping.....	6
5.1 Operation and Notification mapping	6
5.2 Operation parameter mapping	7
5.3 Notification parameter mapping	10
Annex A (normative): IDL specifications	12
A.1 IDL specification (file name "SwMIRPConstDefs.idl")	12
A.2 IDL specification (file name "SwMIRPSystem.idl")	17
A.3 IDL specification (file name "SwMIRPNotifications.idl").....	21
Annex B (informative): Change history.....	27

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:

- 32.531: Telecommunication management; Software management; Concepts and Integration Reference Point (IRP) Requirements
- 32.532: Telecommunication management; Software management Integration Reference Point (IRP); Information Service (IS)
- 32.533: Telecommunication management; Software management Integration Reference Point (IRP); Common Object Request Broker Architecture (CORBA) Solution Set (SS)**
- 32.535: Telecommunication management; Software management Integration Reference Point (IRP); eXtensible Markup Language (XML) definitions

1 Scope

The present document is the "CORBA Solution Set" of Software Management IRP for the IRP whose semantics is specified in Software Management IRP Information Service (3GPP TS 32.532 [5]).

This Solution Set specification is related to 3GPP TS 32.532 [5] V9.3.X.

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. 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 TR 21.905: "Vocabulary for 3GPP Specifications".
- [2] 3GPP TS 32.101: "Telecommunication management; Principles and high level requirements".
- [3] 3GPP TS 32.102: "Telecommunication management; Architecture".
- [4] Void.
- [5] Void.
- [6] 3GPP TS 32.531: "Telecommunication management; Software management; Concepts and Integration Reference Point (IRP) Requirements".
- [7] 3GPP TS 32.532: "Telecommunication management; Software management Integration Reference Point (IRP); Information Service (IS)".
- [8] 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 TS 32.101 [2], TS 32.102 [3] and TR 21.905 [1] and the following apply. A term defined in the present document takes precedence over the definition of the same term, if any, in TS 32.532 [7], TS 32.531 [6], TS 32.101 [2], TS 32.102 [3] and TS 21.905 [1], in that order.

3.2 Abbreviations

For the purposes of the present document, the abbreviations given in TR 21.905 [1], TS 32.531 [6] and the following apply. An abbreviation defined in the present document takes precedence over the definition of the same abbreviation, if any, in TS 32.532 [7], TS 32.531 [6], TS 32.101 [2], TS 32.102 [3] and TS 21.905 [1], in that order.

4 Architectural Features

The overall architectural feature of Software Management IRP is specified in 3GPP TS 32.532 [7].

5 Mapping

5.1 Operation and Notification mapping

Software Management IRP: IS 3GPP TS (see 3GPP TS 32.532 [7]) defines semantics of operations and notifications visible across the Itf-N. Table 5.1.1 indicates mapping of these operations and notifications to their equivalents defined in this SS.

Table 5.1.1: Mapping from IS Notification/Operation to SS equivalents

IS Operation/ notification Software Management IRP: IS 3GPP TS 32.532[7]	SS Method	Qualifier
listSwMCapabilities	listSwMCapabilities	M
listSwMProfiles	listSwMProfiles	M
createSwMProfile	createSwMProfile	M
deleteSwMProfile	deleteSwMProfile	M
listSwMProcesses	listSwMProcesses	M
resumeSwMProcess	resumeSwMProcess	M
swFallback	swFallback	M
terminateSwMProcess	terminateSwMProcess	M
changeSwMProfile	changeSwMProfile	O
downloadNESw	downloadNESw	M
installNESw	installNESw	O
activateNESw	activateNESw	M
notifySwMProfileCreation	notifySwMProfileCreation	M
notifySwMProfileDeletion	notifySwMProfileDeletion	M
notifySwMProcessCreation	notifySwMProcessCreation	M
notifySwMProcessStage	notifySwMProcessStage	M
notifySwMProcessDeletion	notifySwMProcessDeletion	M
notifyNewSwAvailability	notifyNewSwAvailability	M
notifySwMProfileChange	notifySwMProfileChange	O
notifyDownloadNESwStatusChanged	notifyDownloadNESwStatusChanged	M
notifyInstallNESwStatusChanged	notifyInstallNESwStatusChanged	O
notifyActivateNESwStatusChanged	notifyActivateNESwStatusChanged	M

5.2 Operation parameter mapping

Reference 3GPP TS 32.532 [7] defines semantics of parameters carried in operations across the Itf-N. The following set of tables indicate the mapping of these parameters, as per operation, to their equivalents defined in this SS.

Table 5.2-1: Mapping from IS listSwMCapabilities parameters to SS equivalents

IS Operation parameter	SS Method parameter	Qualifier
nEInformation	SwMIRPCConstDefs::NEInformation	M
capabilitiesList	SwMIRPCConstDefs::SwMCapabilitiesList	M
result	Exceptions: SwMIRPCConstDefs::ListSwMCapabilities, GenericIRPManagementSystem::ParameterNotSupported, GenericIRPManagementSystem::InvalidParameter, GenericIRPManagementSystem::ValueNotSupported, GenericIRPManagementSystem::OperationNotSupported	M

Table 5.2-2: Mapping from IS listSwMProfiles parameters to SS equivalents

IS Operation parameter	SS Method parameter	Qualifier
nEInformation	SwMIRPCConstDefs::NEInformation	M
profileList	SwMIRPCConstDefs::SwMProfileList	M
result	Exceptions: SwMIRPCConstDefs::ListSwMProfile, GenericIRPManagementSystem::ParameterNotSupported, GenericIRPManagementSystem::InvalidParameter, GenericIRPManagementSystem::ValueNotSupported, GenericIRPManagementSystem::OperationNotSupported	M

Table 5.2-3: Mapping from IS createSwMProfile parameters to SS equivalents

IS Operation parameter	SS Method parameter	Qualifier
id	SwMIRPCConstDefs::IdOpt	O
nEInformation	SwMIRPCConstDefs::NEInformation	M
swVersionToBeInstalled	SwMIRPCConstDefs::SwVersionToBeInstalledConditional	CM
stepsAndSelectedStopPointList	SwMIRPCConstDefs::StepsAndSelectedStopPointList	M
selectedFinalAdministrativeState	SwMIRPCConstDefs::SelectedFinalAdministrativeState	M
result	Exceptions: SwMIRPCConstDefs::CreateSwMProfile, GenericIRPManagementSystem::ParameterNotSupported, GenericIRPManagementSystem::InvalidParameter, GenericIRPManagementSystem::ValueNotSupported, GenericIRPManagementSystem::OperationNotSupported	M

Table 5.2-4: Mapping from IS deleteSwMProfile parameters to SS equivalents

IS Operation parameter	SS Method parameter	Qualifier
id	SwMIRPCConstDefs::Id	M
result	Exceptions: SwMIRPCConstDefs::DeleteSwMProfile, GenericIRPManagementSystem::ParameterNotSupported, GenericIRPManagementSystem::InvalidParameter, GenericIRPManagementSystem::ValueNotSupported, GenericIRPManagementSystem::OperationNotSupported	M

Table 5.2-5: Mapping from IS listSwMProcesses parameters to SS equivalents

IS Operation parameter	SS Method parameter	Qualifier
nEIdentification	SwMIRPCConstDefs::NEIdentificationOpt	O
processList	SwMIRPCConstDefs::ProcessList	M
result	Exceptions: SwMIRPCConstDefs::ListSwMProcesses, GenericIRPManagementSystem::ParameterNotSupported, GenericIRPManagementSystem::InvalidParameter, GenericIRPManagementSystem::ValueNotSupported, GenericIRPManagementSystem::OperationNotSupported	M

Table 5.2-6: Mapping from IS resumeSwMProcess parameters to SS equivalents

IS Operation parameter	SS Method parameter	Qualifier
id	SwMIRPCConstDefs::Id	M
startStepName	SwMIRPCConstDefs::NameOfStep	M
result	Exceptions: SwMIRPCConstDefs::ResumeSwMProcess, GenericIRPManagementSystem::ParameterNotSupported, GenericIRPManagementSystem::InvalidParameter, GenericIRPManagementSystem::ValueNotSupported, GenericIRPManagementSystem::OperationNotSupported	M

Table 5.2-7: Mapping from IS swFallback parameters to SS equivalents

IS Operation parameter	SS Method parameter	Qualifier
filter	SwMIRPCConstDefs::Filter	M
nEList	SwMIRPCConstDefs::NEList	M
result	Exceptions: SwMIRPCConstDefs::SwFallback, GenericIRPManagementSystem::ParameterNotSupported, GenericIRPManagementSystem::InvalidParameter, GenericIRPManagementSystem::ValueNotSupported, GenericIRPManagementSystem::OperationNotSupported	M

Table 5.2-8: Mapping from IS terminateSwMProcess parameters to SS equivalents

IS Operation parameter	SS Method parameter	Qualifier
id	SwMIRPCConstDefs::Id	M
result	Exceptions: SwMIRPCConstDefs::TerminateSwMProcess, GenericIRPManagementSystem::ParameterNotSupported, GenericIRPManagementSystem::InvalidParameter, GenericIRPManagementSystem::ValueNotSupported, GenericIRPManagementSystem::OperationNotSupported	M

Table 5.2-9: Mapping from IS changeSwMProfile parameters to SS equivalents

IS Operation parameter	SS Method parameter	Qualifier
id	SwMIRPCConstDefs::Id	M
neInformation	SwMIRPCConstDefs::NEInformation	M
swVersionToBeInstalled	SwMIRPCConstDefs::SwVersionToBeInstalledConditional	CM
stepsAndSelectedStopPointList	SwMIRPCConstDefs::StepsAndSelectedStopPointList	M
selectedFinalAdministrativeState	SwMIRPCConstDefs::SelectedFinalAdministrativeState	M
versionNumber	SwMIRPCConstDefs::VersionNumber	M
conflictingProfileId	SwMIRPCConstDefs::ConflictingProfileIdConditional	C
result	Exceptions: SwMIRPCConstDefs::CreateSwMProfile, GenericIRPManagementSystem::ParameterNotSupported, GenericIRPManagementSystem::InvalidParameter, GenericIRPManagementSystem::ValueNotSupported, GenericIRPManagementSystem::OperationNotSupported	M

Table 5.2-10: Mapping from IS downloadNESw parameters to SS equivalents

IS Operation parameter	SS Method parameter	Qualifier
swToBeDownloaded	SwMIRPCConstDefs::SWToBeDownloaded	M
neIdentifier	SwMIRPCConstDefs::NEIdentifier	M
downloadProcessId	SwMIRPCConstDefs::RequestID	M
Reason	SwMIRPCConstDefs::Reason	O
result	Exceptions: SwMIRPCConstDefs::OperationFailed, SwMIRPCConstDefs::ResourceLimitation	M

Table 5.2-11: Mapping from IS installNESw parameters to SS equivalents

IS Operation parameter	SS Method parameter	Qualifier
swToBeInstalled	SwMIRPCConstDefs::SWToBeInstalled	M
neIdentifier	SwMIRPCConstDefs::NEIdentifier	M
installProcessId	SwMIRPCConstDefs::RequestID	M
reason	SwMIRPCConstDefs::Reason	O
result	Exceptions: SwMIRPCConstDefs::OperationFailed, SwMIRPCConstDefs::ResourceLimitation SwMIRPCConstDefs::SWNotAvailable	M

Table 5.2-12: Mapping from IS activateNESw parameters to SS equivalents

IS Operation parameter	SS Method parameter	Qualifier
swVersionToBeActivated	SwMIRPCConstDefs::SWVersion	M
neIdentifier	SwMIRPCConstDefs::NEIdentifier	M
activateProcessId	SwMIRPCConstDefs::RequestID	M
reason	SwMIRPCConstDefs::Reason	O
result	Exceptions: SwMIRPCConstDefs::OperationFailed, SwMIRPCConstDefs::ResourceLimitation	M

5.3 Notification parameter mapping

Reference 3GPP TS 32.532 [7] defines semantics of parameters carried in notifications. The following tables indicate the mapping of these parameters to their SS equivalents."

The following tables indicate the mapping of these parameters to their OMG CORBA Structured Event (defined in OMG Notification Service [8]) equivalents. The composition of OMG Structured Event, as defined in the OMG Notification Service [8], 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 Software Management IRP: IS [7] defined notification parameters.

Table 5.3.1: Mapping for notifySwMProfileCreation

IS Parameters	<SS> Parameters	Qualifier	Comment
id	SwMIRPCConstDefs::Id	M	
versionNumber	SwMIRPCConstDefs::VersionNumber	M	
nEInformation	SwMIRPCConstDefs::NEInformation	M	
swVersionToBeInstalled	SwMIRPCConstDefs::SwVersionToBeInstalledConditional	CM	
stepsAndSelectedStopPointList	SwMIRPCConstDefs::StepsAndSelectedStopPointList	M	
selectedFinalAdministrativeState	SwMIRPCConstDefs::SelectedFinalAdministrativeState	M	

Table 5.3.2: Mapping for notifySwMProfileDeletion

IS Parameters	<SS> Parameters	Qualifier	Comment
id	SwMIRPCConstDefs::Id	M	

Table 5.3.3: Mapping for notifySwMProcessCreation

IS Parameters	<SS> Parameters	Qualifier	Comment
id	SwMIRPCConstDefs::Id	M	
nEIdentification	SwMIRPCConstDefs::NEIdentification	M	
profileId	SwMIRPCConstDefs::ProfileId	M	
matchingNEInformation	SwMIRPCConstDefs::MatchingNEInformation	M	
stepInfoList	SwMIRPCConstDefs::StepInfoList	M	

Table 5.3.4: Mapping for notifySwMProcessStage

IS Parameters	<SS> Parameters	Qualifier	Comment
id	SwMIRPCConstDefs::Id	M	
stepInfoList	SwMIRPCConstDefs::StepInfoList	M	

Table 5.3.5: Mapping for notifySwMProcessDeletion

IS Parameters	<SS> Parameters	Qualifier	Comment
id	SwMIRPCConstDefs::Id	M	
triggerForDeletion	SwMIRPCConstDefs::TriggerForDeletion	M	
additionalInformation	SwMIRPCConstDefs::AdditionalInformationOptional	O	

Table 5.3.6: Mapping for notifyNewSwAvailability

IS Parameters	<SS> Parameters	Qualifier	Comment
nEandSWversion	SwMIRPCConstDefs::NEandSWversion	M	

Table 5.3.7: Mapping for notifySwMProfileChange

IS Parameters	<SS> Parameters	Qualifier	Comment
id	SwMIRPCConstDefs::Id	M	
versionNumber	SwMIRPCConstDefs::VersionNumber	M	
nEInformation	SwMIRPCConstDefs::NEInformation	M	
swVersionToBeInstalled	SwMIRPCConstDefs::SwVersionToBeInstalledConditional	CM	
stepsAndSelectedStopPointList	SwMIRPCConstDefs::StepsAndSelectedStopPointList	M	
selectedFinalAdministrativeState	SwMIRPCConstDefs::SelectedFinalAdministrativeState	M	

Table 5.3.8: Mapping for notifyDownloadNESwStatusChanged

IS Parameters	<SS> Parameters	Qualifier	Comment
downloadProcessId	SwMIRPCConstDefs::DownloadProcessId;	M	
downloadOperationStatus	SwMIRPCConstDefs::DownloadNESwOperationStatus	M	
downloadedNESwInfo	SwMIRPCConstDefs::DownloadedNESwInfo	O	
failedSwInfo	SwMIRPCConstDefs::FailedSwInfo	O	

Table 5.3.9: Mapping for notifyInstallNESwStatusChanged

IS Parameters	<SS> Parameters	Qualifier	Comment
installProcessId	SwMIRPCConstDefs::RequestID	M	
installOperationStatus	SwMIRPCConstDefs::InstallOperationStatus	M	
installNESwInfo	SwMIRPCConstDefs::InstalledNESwInfo	O	
failedSwInfo	SwMIRPCConstDefs::FailedSwInfo	O	

Table 5.3.10: Mapping for notifyActivateNESwStatusChanged

IS Parameters	<SS> Parameters	Qualifier	Comment
activateProcessId	SwMIRPCConstDefs::RequestID	M	
activateOperationStatus	SwMIRPCConstDefs::ActivateOperationStatus	M	
swVersionActivated	SwMIRPCConstDefs::SWVersion	CM	
failureReason	SwMIRPCConstDefs::FailureReason	CM	

Annex A (normative): IDL specifications

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

```
// File: SwMIRPCConstDefs.idl
#ifndef _SWM_IRP_CONST_DEFS_IDL_
#define _SWM_IRP_CONST_DEFS_IDL_

#include <NotificationIRPCConstDefs.idl>
#include <GenericIRPManagementConstDefs.idl>

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

/* ## Module: SwMIRPCConstDefs */

module SwMIRPCConstDefs
{

/*****
/* definition of types used in operations for Software Management : */
*****/

/* types used in several operations: */

    enum Result { SUCCESS, PARTLY_SUCCESSFUL, FAILURE, NE_INFORMATION_INTERSECTION,
STEPNAME_DOES_NOT_MATCH };

    enum ResultChangeOrCreateProfileOperation { CHANGEORCREATEPROFILE_SUCCESS,
CHANGEORCREATEPROFILE_FAILURE, CHANGEORCREATEPROFILE_NE_INFORMATION_INTERSECTION,
NOT_ALLOWED_BECAUSE_OF_ONGOING_ACTIVITY };

    enum ResultNaSwmNotification { REQUEST_ACCEPTED, REQUEST_FAILED,
NOT_ALLOWED_BECAUSE_OF_ONGOING_SWM_ACTIVITY };

    typedef GenericIRPManagementConstDefs::DN Id;

    /*
    IdOpt is a type carrying an optional parameter.
    If the boolean is TRUE, then the value is present.
    Otherwise the value is absent.
    */
    union IdOpt switch (boolean)
    {
        case TRUE: GenericIRPManagementConstDefs::DN value;
    };

    /*
    ConflictingProfileIdConditional is a type carrying a conditional parameter.
    The boolean shall be TRUE, if the condition described in TS 32.532 or 32.502 is fulfilled.
    In this case the value is present. Otherwise the value is absent.
    */
    union ConflictingProfileIdConditional switch (boolean)
    {
        case TRUE: Id value;
    };

    typedef string NEInformation;

    typedef NEInformation MatchingNEInformation;

    typedef string AdditionalInformation;

    /*
    AdditionalInformationOpt is a type carrying an optional parameter.
    The boolean shall be TRUE, if the condition described in TS 32.532 is fulfilled.

```

```

In this case the value is present. Otherwise the value is absent.
*/
union AdditionalInformationOpt switch (boolean)
{
    case TRUE: AdditionalInformation value;
};

typedef GenericIRPManagementConstDefs::DN NEIdentification;

/*
NEIdentificationOpt is a type carrying an optional parameter.
If the boolean is TRUE, then the value is present.
Otherwise the value is absent.
*/
union NEIdentificationOpt switch (boolean)
{
    case TRUE: NEIdentification value;
};

enum SwFallbackStatus { FALLBACK_SUCCESSFUL, FALLBACK_UNSUCCESSFUL };

struct NeListEntry
{
    SwMIRPConstDefs::NEIdentification neIdentification;
    SwMIRPConstDefs::SwFallbackStatus swFallbackStatus;
};

typedef sequence<NeListEntry> NeList;

typedef string SwVersionToBeInstalled;

/*
SwVersionToBeInstalledConditional is a type carrying a conditional parameter.
The boolean shall be TRUE, if the condition described in TS 32.532 is fulfilled.
In this case the value is present. Otherwise the value may be absent.
*/
union SwVersionToBeInstalledConditional switch (boolean)
{
    case TRUE: SwVersionToBeInstalled value;
};

enum NameOfStep
{
    SW_DOWNLOAD,
    SW_INSTALLATION,
    SW_ACTIVATION,
    PREPARE_BASIC_CONFIGURATION_AND_OAMLINK,
    RETRIEVE_CONFIGURATION_DATA,
    SETUP_PRECONFIGURED_SIGNALLING_LINKS,
    SET_FINAL_STATE_OF_NE
};
/*
The following values are not used in SWM IRP, but only in inheriting Self-Conf IRP TS 32.502:
PREPARE_BASIC_CONFIGURATION_AND_OAMLINK,
RETRIEVE_CONFIGURATION_DATA,
SETUP_PRECONFIGURED_SIGNALLING_LINKS,
SET_FINAL_STATE_OF_NE
*/

typedef unsigned short SequenceNumberInProgress;

enum StopPointCanBeSetBeforeThisStep { YES, NO };

struct StepsAndOfferedStopPointListEntry
{
    SwMIRPConstDefs::NameOfStep nameOfStep;
    SwMIRPConstDefs::SequenceNumberInProgress sequenceNumberInProgress;
    SwMIRPConstDefs::StopPointCanBeSetBeforeThisStep stopPointCanBeSetBeforeThisStep
};
;
};

```

```

typedef sequence<StepsAndOfferedStopPointListEntry> StepsAndOfferedStopPointList;

enum StopPointSetIndication { STOP_POINT_IS_SET_BEFORE_THIS_STEP, STOP_POINT_IS_NOT_SET };

struct StepAndSelectedStopPointListEntry
{
    SwMIRPCConstDefs::NameOfStep nameOfStep;
    SwMIRPCConstDefs::SequenceNumberInProcess sequenceNumberInProcess;
    SwMIRPCConstDefs::StopPointSetIndication stopPointSetIndication
;
};

typedef sequence<StepAndSelectedStopPointListEntry> StepsAndSelectedStopPointList;

enum StepProgress { NOT_YET_STARTED, RUNNING, COMPLETED, AWAITING_RESUME, STEPPROGRESS_FAILURE,
TERMINATED };

struct StepInfoListEntry
{
    SwMIRPCConstDefs::NameOfStep nameOfStep;
    SwMIRPCConstDefs::SequenceNumberInProcess sequenceNumberInProcess;
    SwMIRPCConstDefs::StopPointSetIndication stopPointSetIndication;
    SwMIRPCConstDefs::StepProgress stepProgress;
};

typedef sequence<StepInfoListEntry> StepInfoList;

struct SwMProcessListEntry
{
    SwMIRPCConstDefs::Id id;
    SwMIRPCConstDefs::NEIdentification nEIdentification;
    SwMIRPCConstDefs::StepInfoList stepInfoList;
};

typedef sequence<SwMProcessListEntry> SwMProcessList;

enum FinalAdministrativeStateValue { LOCKED, UNLOCKED, DETERMINED_BY_CONFIGURATION_DATA };

typedef FinalAdministrativeStateValue OfferedFinalAdministrativeStateValue;

typedef sequence<OfferedFinalAdministrativeStateValue>
OfferedFinalAdministrativeStateInformation;

typedef FinalAdministrativeStateValue SelectedFinalAdministrativeStateValue;

typedef sequence<SwVersionToBeInstalled> SwVersionToBeInstalledOfferList;

/*
SwVersionToBeInstalledOfferListConditional is a type carrying a conditional parameter.
The boolean shall be TRUE, if the condition described in TS 32.532 is fulfilled.
In this case the value is present. Otherwise the value may be absent.
*/
union SwVersionToBeInstalledOfferListConditional switch (boolean)
{
    case TRUE: SwVersionToBeInstalledOfferList value;
};

typedef unsigned short VersionNumber;

struct ProfileId
{
    SwMIRPCConstDefs::Id id;
    SwMIRPCConstDefs::VersionNumber versionNumber;
};

struct SwMCapability
{
    SwMIRPCConstDefs::Id id;

```

```

SwMIRPConstDefs::NEInformation nEInformation;
SwMIRPConstDefs::StepsAndOfferedStopPointList stepsAndOfferedStopPointList;
SwMIRPConstDefs::OfferedFinalAdministrativeStateInformation
offeredFinalAdministrativeStateInformation;
SwMIRPConstDefs::SwVersionToBeInstalledOfferListConditional swVersionToBeInstalledOfferList;
};

typedef sequence<SwMCapability> SwMCapabilitiesList;

struct SwMProfile
{
SwMIRPConstDefs::Id id;
SwMIRPConstDefs::VersionNumber versionNumber;
SwMIRPConstDefs::NEInformation nEInformation;
SwMIRPConstDefs::StepsAndSelectedStopPointList stepsAndSelectedStopPointList;
SwMIRPConstDefs::SelectedFinalAdministrativeStateValue selectedFinalAdministrativeState;
SwMIRPConstDefs::SwVersionToBeInstalledConditional swVersionToBeInstalled;
};

typedef sequence<SwMProfile> SwMProfilesList;

/*****
/* definition of types in notifications for software management : */
*****/

typedef string Filter;

typedef string NeAndSWVersion;

enum TriggerForDeletion { IRP_AGENT_TERMINATION, IRP_MANAGER_TERMINATION,
AUTOMATED_SWM_SUCCESFULLY_CONCLUDED, SELF_CONFIGURATION_SUCCESFULLY_CONCLUDED };
/*
The following values are not used in SWM IRP, but only in inheriting Self-Conf IRP TS 32.502:
SELF_CONFIGURATION_SUCCESFULLY_CONCLUDED
*/

/*****
* Definitions for Non-Automated Software Management
*****/

typedef GenericIRPManagementConstDefs::DN NEIdentifier;
typedef unsigned long RequestID;
typedef string Reason;

typedef string FileLocation; //The FileLocation may be a directory path or a URL
typedef unsigned long FileSize; //the unit is byte
typedef string FileCompression;
typedef string FileFormat;

struct SWInfo
{
SwMIRPConstDefs::FileLocation swLocation;
SwMIRPConstDefs::FileSize swFileSize;
SwMIRPConstDefs::FileCompression swFileCompression;
SwMIRPConstDefs::FileFormat swFileFormat;
};

typedef SwMIRPConstDefs::FileLocation SWToBeInstalled;

typedef sequence<SWInfo> SWToBeDownloaded;

typedef string SWVersion;

/*****
* Definitions for Non-Automatic Software Management Notifications
*****/
typedef RequestID DownloadProcessId;
typedef RequestID InstallProcessId;
typedef RequestID ActivateProcessId;

typedef string FailureReason;

typedef sequence<FileLocation> DownloadedNESwInfo;
typedef sequence<FileLocation> InstalledNESwInfo;

```



```

struct FailedSwEntry
{
    SwMIRPCConstDefs::FileLocation failedSw;
    SwMIRPCConstDefs::FailureReason failureReason;
};

typedef sequence<FailedSwEntry> FailedSwInfo;
typedef SWVersion swVersionActivated;

enum DownloadNESwOperationStatus {NE_SW_DOWNLOAD_SUCCESSFUL, NE_SW_DOWNLOAD_FAILED,
NE_SW_DOWNLOAD_PARTIALLY_SUCCESSFUL};
enum InstallOperationStatus {NE_SW_INSTALLATION_SUCCESSFUL, NE_SW_INSTALLATION_FAILED,
NE_SW_INSTALLATION_PARTIALLY_SUCCESSFUL};
enum ActivateOperationStatus {NE_SW_ACTIVATION_SUCCESSFUL, NE_SW_ACTIVATION_FAILED,
NE_SWACTIVATION_PARTIALLY_SUCCESSFUL};

interface AttributeNameValue
{
    const string ID = "ID";
    const string VERSION_NUMBER = "VERSION NUMBER";
    const string NE_INFORMATION = "NE INFORMATION";
    const string SW_VERSION_TO_BE_INSTALLED = "SW VERSION TO BE INSTALLED";
    const string STEPS_AND_SELECTED_STOP_POINT_LIST = "STEPS AND SELECTED STOP POINT LIST";
    const string SELECTED_FINAL_ADMINISTRATIVE_STATE = "SELECTED_FINAL_ADMINISTRATIVE_STATE";
    const string NE_IDENTIFICATION = "NE_IDENTIFICATION";
    const string PROFILE_ID = "PROFILE_ID";
    const string MATCHING_NE_INFORMATION = "MATCHING_NE_INFORMATION";
    const string STEP_INFO_LIST = "STEP_INFO_LIST";
    const string NE_AND_SW_VERSION = "NE AND SW VERSION";
    const string TRIGGER_FOR_DELETION = "TRIGGER FOR DELETION";
    const string ADDITIONAL_INFORMATION = "ADDITIONAL_INFORMATION";
    const string DOWNLOAD_PROCESS_ID = "DOWNLOAD_PROCESS_ID";
    const string DOWNLOAD_OPERATION_STATUS = "DOWNLOAD_OPERATION_STATUS";
    const string DOWNLOADED_NESW_INFO = "DOWNLOADED_NESW_INFO";
    const string INSTALL_PROCESS_ID = "INSTALL_PROCESS_ID";
    const string INSTALL_OPERATION_STATUS = "INSTALL_OPERATION_STATUS";
    const string INSTALLED_NESW_INFO = "INSTALLED_NESW_INFO";
    const string ACTIVATE_PROCESS_ID = "ACTIVATE_PROCESS_ID";
    const string ACTIVATE_OPERATION_STATUS = "ACTIVATE_OPERATION_STATUS";
    const string FAILURE_REASON = "FAILURE_REASON";
    const string SW_VERSION_ACTIVATED = "SW_VERSION_ACTIVATED";
    const string FAILED_SW_INFO = "FAILED_SW_INFO";    };
};

#endif // _SWM_IRP_CONST_DEFS_IDL_

```

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

```

//File: SwMIRPSystem.idl
#ifndef _SWM_IRP_SYSTEM_IDL_
#define _SWM_IRP_SYSTEM_IDL_

#include <SwMIRPConstDefs.idl>
#include <GenericIRPManagementSystem.idl>

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

/* ## Module: SwMIRPSystem */

module SwMIRPSystem
{
    /*
    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 ListSwMCapabilities { string reason; };
    exception ListSwMProfiles { string reason; };
    exception CreateSwMProfile { string reason; };
    exception DeleteSwMProfile { string reason; };
    exception ListSwMProcesses { string reason; };
    exception ResumeSwMProcess { string reason; };
    exception SwFallback { string reason; };
    exception TerminateSwMProcess { string reason; };
    exception ChangeSwMProfile { string reason; };
    exception OperationFailed { string reason; };
    exception ResourceLimitation { string reason; };
    exception SWNotAvailable { string reason; };

    interface SwMIRPOperations_1
    {
        /* for the purpose of this operation see 3GPP TS 32.532 */
        SwMIRPConstDefs::Result listSwMCapabilities
        (
            in SwMIRPConstDefs::NEInformation nEInformation,
            out SwMIRPConstDefs::SwMCapabilitiesList capabilitiesList
        )
        raises
        (
            ListSwMCapabilities,
            GenericIRPManagementSystem::ParameterNotSupported,
            GenericIRPManagementSystem::InvalidParameter,
            GenericIRPManagementSystem::ValueNotSupported,
            GenericIRPManagementSystem::OperationNotSupported
        );

        /* for the purpose of this operation see 3GPP TS 32.532 */
        SwMIRPConstDefs::Result listSwMProfiles
        (
            in SwMIRPConstDefs::NEInformation nEInformation,
            out SwMIRPConstDefs::SwMProfilesList profileList
        )
        raises
        (
            ListSwMProfiles,
            GenericIRPManagementSystem::ParameterNotSupported,
            GenericIRPManagementSystem::InvalidParameter,
            GenericIRPManagementSystem::ValueNotSupported,
            GenericIRPManagementSystem::OperationNotSupported
        );

        /* for the purpose of this operation see 3GPP TS 32.532 */
        SwMIRPConstDefs::ResultChangeOrCreateProfileOperation createSWMProfile
        (
            in SwMIRPConstDefs::IdOpt id,
            in SwMIRPConstDefs::NEInformation nEInformation,

```

```

    in SwMIRPConstDefs::SwVersionToBeInstalledConditional swVersionToBeInstalled,
    in SwMIRPConstDefs::StepsAndSelectedStopPointList stepsAndSelectedStopPointList,
    in SwMIRPConstDefs::SelectedFinalAdministrativeStateValue selectedFinalAdministrativeState
  )
  raises
  (
    CreateSwMProfile,
    GenericIRPManagementSystem::ParameterNotSupported,
    GenericIRPManagementSystem::InvalidParameter,
    GenericIRPManagementSystem::ValueNotSupported,
    GenericIRPManagementSystem::OperationNotSupported
  );

/* for the purpose of this operation see 3GPP TS 32.532 */
SwMIRPConstDefs::Result deleteSwMProfile
(
  in SwMIRPConstDefs::Id id
)
  raises
  (
    DeleteSwMProfile,
    GenericIRPManagementSystem::ParameterNotSupported,
    GenericIRPManagementSystem::InvalidParameter,
    GenericIRPManagementSystem::ValueNotSupported,
    GenericIRPManagementSystem::OperationNotSupported
  );

/* for the purpose of this operation see 3GPP TS 32.532 */
SwMIRPConstDefs::Result listSwMProcesses
(
  in SwMIRPConstDefs::NEIdentificationOpt nEIdentification,
  out SwMIRPConstDefs:: SwMProcessList processList
)
  raises
  (
    ListSwMProcesses,
    GenericIRPManagementSystem::ParameterNotSupported,
    GenericIRPManagementSystem::InvalidParameter,
    GenericIRPManagementSystem::ValueNotSupported,
    GenericIRPManagementSystem::OperationNotSupported
  );

/* for the purpose of this operation see 3GPP TS 32.532 */
SwMIRPConstDefs::Result resumeSwMProcess
(
  in SwMIRPConstDefs::Id id,
  in SwMIRPConstDefs::NameOfStep startStepName
)
  raises
  (
    ResumeSwMProcess,
    GenericIRPManagementSystem::ParameterNotSupported,
    GenericIRPManagementSystem::InvalidParameter,
    GenericIRPManagementSystem::ValueNotSupported,
    GenericIRPManagementSystem::OperationNotSupported
  );

/* for the purpose of this operation see 3GPP TS 32.532 */
SwMIRPConstDefs::Result swFallback
(
  in SwMIRPConstDefs::Filter filter,
  out SwMIRPConstDefs:: NeList nEList
)
  raises
  (
    SwFallback,
    GenericIRPManagementSystem::ParameterNotSupported,
    GenericIRPManagementSystem::InvalidParameter,
    GenericIRPManagementSystem::ValueNotSupported,
    GenericIRPManagementSystem::OperationNotSupported
  );

/* for the purpose of this operation see 3GPP TS 32.532 */

```

```

SwMIRPConstDefs::Result terminateSwMProcess
(
    in SwMIRPConstDefs::Id id
)
raises
(
    TerminateSwMProcess,
    GenericIRPManagementSystem::ParameterNotSupported,
    GenericIRPManagementSystem::InvalidParameter,
    GenericIRPManagementSystem::ValueNotSupported,
    GenericIRPManagementSystem::OperationNotSupported
);

};

interface SwMIRPOperations_2
{
    /* for the purpose of this operation see 3GPP TS 32.532 */
    SwMIRPConstDefs::ResultChangeOrCreateProfileOperation changeSWMPProfile
    (
        in SwMIRPConstDefs::Id id,
        in SwMIRPConstDefs::NEInformation neInformation,
        in SwMIRPConstDefs::SwVersionToBeInstalledConditional swVersionToBeInstalled,
        in SwMIRPConstDefs::StepsAndSelectedStopPointList stepsAndSelectedStopPointList,
        in SwMIRPConstDefs::SelectedFinalAdministrativeStateValue selectedFinalAdministrativeState
        out SwMIRPConstDefs::VersionNumber versionNumber
        out SwMIRPConstDefs::ConflictingProfileIdConditional conflictingProfileId
    )
    raises
    (
        ChangeSwMProfile,
        GenericIRPManagementSystem::ParameterNotSupported,
        GenericIRPManagementSystem::InvalidParameter,
        GenericIRPManagementSystem::ValueNotSupported,
        GenericIRPManagementSystem::OperationNotSupported
    );

};

interface SwMIRPOperations_3
{
    /* for the purpose of this operation see 3GPP TS 32.532 */
    SwMIRPConstDefs::ResultNaSwmNotification downloadNESw
    (
        in SwMIRPConstDefs::SWToBeDownloaded swToBeDownloaded,
        in SwMIRPConstDefs::NEIdentifier neIdentifier,
        out SwMIRPConstDefs::RequestID downloadProcessId,
        out SwMIRPConstDefs::Reason reason
    )
    raises
    (
        OperationFailed,
        ResourceLimitation
    );
    /* for the purpose of this operation see 3GPP TS 32.532 */
    SwMIRPConstDefs::ResultNaSwmNotification activateNESw
    (
        in SwMIRPConstDefs::SWVersion swVersionToBeActivated,
        in SwMIRPConstDefs::NEIdentifier neIdentifier,
        out SwMIRPConstDefs::RequestID requestID,
        out SwMIRPConstDefs::Reason reason
    )
    raises
    (
        OperationFailed,
        ResourceLimitation
    );
};

interface SwMIRPOperations_4
{
    /* for the purpose of this operation see 3GPP TS 32.532 */
    SwMIRPConstDefs::ResultNaSwmNotification installNESw
    (
        in SwMIRPConstDefs::SWToBeInstalled swToBeInstalled,
        in SwMIRPConstDefs::NEIdentifier neIdentifier,

```

```
        out SwMIRPConstDefs::RequestID installProcessId,
        out SwMIRPConstDefs::Reason reason
    )
    raises
    (
        OperationFailed,
        ResourceLimitation,
        SWNotAvailable
    );
};

};

#endif // _SWM_IRP_SYSTEM_IDL_
```

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

```
//File: SwMIRPNotifications.idl
#ifndef _SWM_IRP_NOTIFICATIONS_IDL_
#define _SWM_IRP_NOTIFICATIONS_IDL_

#include <SwMIRPConstDefs.idl>
#include <NotificationIRPNotifications.idl>

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

/* ## Module: SwMIRPNotifications
This contains the specification of notifications of Software Management.
=====
*/
module SwMIRPNotifications
{

    /* Constant definitions for the notifySwMProfileCreation notification */

    interface NotifySwMProfileCreation: NotificationIRPNotifications::Notify
    {
        const string EVENT_TYPE = "notifySwMProfileCreation";

        /**
         * This constant defines the name of the Id property,
         * which is transported in the filterable_body_fields.
         * The data type for the value of this property is
         * SwMIRPConstDefs::Id.
         */
        const string ID =
            SwMIRPConstDefs::AttributeNameValue::ID;

        /**
         * This constant defines the name of the VersionNumber property,
         * which is transported in the filterable_body_fields.
         * The data type for the value of this property is
         * SwMIRPConstDefs::VersionNumber.
         */
        const string VERSION_NUMBER =
            SwMIRPConstDefs::AttributeNameValue::VERSION_NUMBER;

        /**
         * This constant defines the name of the NEInformation property,
         * which is transported in the filterable_body_fields.
         * The data type for the value of this property is
         * SwMIRPConstDefs::NEInformation.
         */
        const string NE_INFORMATION =
            SwMIRPConstDefs::AttributeNameValue::NE_INFORMATION;

        /**
         * This constant defines the name of the SwVersionToBeInstalled property,
         * which is transported in the filterable_body_fields.
         * The data type for the value of this property is
         * SwMIRPConstDefs::SwVersionToBeInstalledConditional.
         */
        const string SW_VERSION_TO_BE_INSTALLED =
            SwMIRPConstDefs::AttributeNameValue::SW_VERSION_TO_BE_INSTALLED;

        /**
         * This constant defines the name of the StepsAndSelectedStopPointList property,
         * which is transported in the remaining_body.
         * The data type for the value of this property is
         * SwMIRPConstDefs::StepsAndSelectedStopPointList.
         */
        const string STEPS_AND_SELECTED_STOP_POINT_LIST =
            SwMIRPConstDefs::AttributeNameValue::STEPS_AND_SELECTED_STOP_POINT_LIST;

        /**
         * This constant defines the name of the SelectedFinalAdministrativeState property,

```

```

* which is transported in the remaining_body.
* The data type for the value of this property is
* SwMIRPCConstDefs::SelectedFinalAdministrativeState.
*/
const string SELECTED_FINAL_ADMINISTRATIVE_STATE =
    SwMIRPCConstDefs::AttributeNameValue::SELECTED_FINAL_ADMINISTRATIVE_STATE;
};

/* Constant definitions for the notifySwMProfileDeletion notification */
interface NotifySwMProfileDeletion: NotificationIRPNotifications::Notify
{
    const string EVENT_TYPE = "notifySwMProfileDeletion";

    /**
    * This constant defines the name of the Id property,
    * which is transported in the filterable_body_fields.
    * The data type for the value of this property is
    * SwMIRPCConstDefs::Id.
    */
    const string ID =
        SwMIRPCConstDefs::AttributeNameValue::ID;
};

/* Constant definitions for the notifySwMProcessCreation notification */
interface NotifySwMProcessCreation: NotificationIRPNotifications::Notify
{
    const string EVENT_TYPE = "notifySwMProcessCreation";

    /**
    * This constant defines the name of the Id property,
    * which is transported in the filterable_body_fields.
    * The data type for the value of this property is
    * SwMIRPCConstDefs::Id.
    */
    const string ID =
        SwMIRPCConstDefs::AttributeNameValue::ID;

    /**
    * This constant defines the name of the NEIdentification property,
    * which is transported in the filterable_body_fields.
    * The data type for the value of this property is
    * SwMIRPCConstDefs::NEIdentification.
    */
    const string NE_IDENTIFICATION =
        SwMIRPCConstDefs::AttributeNameValue::NE_IDENTIFICATION;

    /**
    * This constant defines the name of the ProfileId property,
    * which is transported in the remaining_body.
    * The data type for the value of this property is
    * SwMIRPCConstDefs::ProfileId.
    */
    const string PROFILE_ID =
        SwMIRPCConstDefs::AttributeNameValue::PROFILE_ID;

    /**
    * This constant defines the name of the MatchingNEInformation property,
    * which is transported in the remaining_body.
    * The data type for the value of this property is
    * SwMIRPCConstDefs::MatchingNEInformation.
    */
    const string MATCHING_NE_INFORMATION =
        SwMIRPCConstDefs::AttributeNameValue::MATCHING_NE_INFORMATION;

    /**
    * This constant defines the name of the StepInfoList property,
    * which is transported in the remaining_body.
    * The data type for the value of this property is
    * SwMIRPCConstDefs::StepInfoList.
    */
    const string STEP_INFO_LIST =
        SwMIRPCConstDefs::AttributeNameValue::STEP_INFO_LIST;
};

```

```
};

/* Constant definitions for the notifySwMProcessStage notification */
interface NotifySwMProcessStage: NotificationIRPNotifications::Notify
{
    const string EVENT_TYPE = "notifySwMProcessStage";

    /**
     * This constant defines the name of the Id property,
     * which is transported in the filterable_body_fields.
     * The data type for the value of this property is
     * SwMIRPCConstDefs::Id.
     */
    const string ID =
        SwMIRPCConstDefs::AttributeNameValue::ID;

    /**
     * This constant defines the name of the StepInfoList property,
     * which is transported in the remaining_body.
     * The data type for the value of this property is
     * SwMIRPCConstDefs::StepInfoList.
     */
    const string STEP_INFO_LIST =
        SwMIRPCConstDefs::AttributeNameValue::STEP_INFO_LIST;
};

/* Constant definitions for the notifySwMProcessDeletion notification */
interface NotifySwMProcessDeletion: NotificationIRPNotifications::Notify
{
    const string EVENT_TYPE = "notifySwMProcessDeletion";

    /**
     * This constant defines the name of the Id property,
     * which is transported in the filterable_body_fields.
     * The data type for the value of this property is
     * SwMIRPCConstDefs::Id.
     */
    const string ID =
        SwMIRPCConstDefs::AttributeNameValue::ID;

    /**
     * This constant defines the name of the TriggerForDeletion property,
     * which is transported in the remaining_body.
     * The data type for the value of this property is
     * SwMIRPCConstDefs::TriggerForDeletion.
     */
    const string TRIGGER_FOR_DELETION =
        SwMIRPCConstDefs::AttributeNameValue::TRIGGER_FOR_DELETION;

    /**
     * This constant defines the name of the AdditionalInformation property,
     * which is transported in the remaining_body.
     * The data type for the value of this property is
     * SwMIRPCConstDefs::AdditionalInformationOptional.
     */
    const string ID =
        SwMIRPCConstDefs::AttributeNameValue::ADDITIONAL_INFORMATION;
};

/* Constant definitions for the notifyNewSwAvailability notification */
interface NotifyNewSwAvailability: NotificationIRPNotifications::Notify
{
    const string EVENT_TYPE = "notifyNewSwAvailability";

    /**
     * This constant defines the name of the NEandSWversion property,
     * which is transported in the filterable_body_fields.
     * The data type for the value of this property is
     * SwMIRPCConstDefs::NEandSWversion.
     */
}
```



```
*/
const string NE_AND_SW_VERSION =
    SwMIRPCConstDefs::AttributeNameValue::NE_AND_SW_VERSION;
};

/* Constant definitions for the notifySwMProfileChange notification */
interface NotifySwMProfileChange: NotificationIRPNotifications::Notify
{
    const string EVENT_TYPE = "notifySwMProfileChange";

    /**
     * This constant defines the name of the Id property,
     * which is transported in the filterable_body_fields.
     * The data type for the value of this property is
     * SwMIRPCConstDefs::Id.
     */
    const string ID =
        SwMIRPCConstDefs::AttributeNameValue::ID;

    /**
     * This constant defines the name of the VersionNumber property,
     * which is transported in the filterable_body_fields.
     * The data type for the value of this property is
     * SwMIRPCConstDefs::VersionNumber.
     */
    const string VERSION_NUMBER =
        SwMIRPCConstDefs::AttributeNameValue::VERSION_NUMBER;

    /**
     * This constant defines the name of the NEInformation property,
     * which is transported in the filterable_body_fields.
     * The data type for the value of this property is
     * SwMIRPCConstDefs::NEInformation.
     */
    const string NE_INFORMATION =
        SwMIRPCConstDefs::AttributeNameValue::NE_INFORMATION;

    /**
     * This constant defines the name of the SwVersionToBeInstalled property,
     * which is transported in the filterable_body_fields.
     * The data type for the value of this property is
     * SwMIRPCConstDefs::SwVersionToBeInstalledConditional.
     */
    const string SW_VERSION_TO_BE_INSTALLED =
        SwMIRPCConstDefs::AttributeNameValue::SW_VERSION_TO_BE_INSTALLED;

    /**
     * This constant defines the name of the StepsAndSelectedStopPointList property,
     * which is transported in the filterable_body_fields.
     * The data type for the value of this property is
     * SwMIRPCConstDefs::StepsAndSelectedStopPointList.
     */
    const string STEPS_AND_SELECTED_STOP_POINT_LIST =
        SwMIRPCConstDefs::AttributeNameValue::STEPS_AND_SELECTED_STOP_POINT_LIST;

    /**
     * This constant defines the name of the SelectedFinalAdministrativeState property,
     * which is transported in the filterable_body_fields.
     * The data type for the value of this property is
     * SwMIRPCConstDefs::SelectedFinalAdministrativeState.
     */
    const string SELECTED_FINAL_ADMINISTRATIVE_STATE =
        SwMIRPCConstDefs::AttributeNameValue::SELECTED_FINAL_ADMINISTRATIVE_STATE;
};

/* Constant definitions for the notifyDownloadNESwStatusChanged notification */
interface notifyDownloadNESwStatusChanged: NotificationIRPNotifications::Notify
{
    const string EVENT_TYPE = "notifyDownloadNESwStatusChanged";

    /**
     * This constant defines the name of the downloadProcessId property used
     * during downloadNESw operation and transported in the

```

```

* filterable_body_fields. The data type for the value of this property
* is SwMIRPCConstDefs::RequestID.
*/
const string DOWNLOAD_PROCESS_ID =
    SwMIRPCConstDefs::AttributeNameValue::DOWNLOAD_PROCESS_ID;

/**
* This constant defines the name of the downloadOperationStatus
* property which is transported in the filterable_body_fields.
* The data type for the value of this property is
* SwMIRPCConstDefs::DownloadNESwOperationStatus.
*/
const string DOWNLOAD_OPERATION_STATUS =
    SwMIRPCConstDefs::AttributeNameValue::DOWNLOAD_OPERATION_STATUS;

/**
* This constant defines the name of the downloadedNESwInfo
* property, which is transported in the remaining_body.
* The data type for the value of this property is
* SwMIRPCConstDefs::DownloadedNESwInfo.
*/
const string DOWNLOADED_NESW_INFO =
    SwMIRPCConstDefs::AttributeNameValue::DOWNLOADED_NESW_INFO;

/**
* This constant defines the name of the FailedSwInfo property,
* which is transported in the remaining_body.
* The data type for the value of this property is
* SwMIRPCConstDefs::FailedSwInfo.
*/
const string FAILED_SW_INFO =
    SwMIRPCConstDefs::AttributeNameValue::FAILED_SW_INFO;
};

/* Constant definitions for the notifyInstallNESwStatusChanged notification */
interface notifyInstallNESwStatusChanged: NotificationIRPNotifications::Notify
{
    const string EVENT_TYPE = "notifyInstallNESwStatusChanged";

    /**
    * This constant defines the name of the InstallProcessId property used
    * during installNESw operation and transported in the
    * filterable_body_fields. The data type for the value of this property
    * is SwMIRPCConstDefs::RequestID.
    */
    const string INSTALL_PROCESS_ID =
        SwMIRPCConstDefs::AttributeNameValue::INSTALL_PROCESS_ID;

    /**
    * This constant defines the name of the InstallOperationStatus
    * property which is transported in the filterable_body_fields.
    * The data type for the value of this property is
    * SwMIRPCConstDefs::InstallOperationStatus.
    */
    const string INSTALL_OPERATION_STATUS =
        SwMIRPCConstDefs::AttributeNameValue::INSTALL_OPERATION_STATUS;

    /**
    * This constant defines the name of the InstalledNESwInfo property,
    * which is transported in the remaining_body.
    * The data type for the value of this property is
    * SwMIRPCConstDefs::InstalledNESwInfo.
    */
    const string INSTALLED_NESW_INFO =
        SwMIRPCConstDefs::AttributeNameValue::INSTALLED_NESW_INFO;

    /**
    * This constant defines the name of the FailedSwInfo property,
    * which is transported in the remaining_body.
    * The data type for the value of this property is
    * SwMIRPCConstDefs::FailedSwInfo.
    */
    const string FAILED_SW_INFO =
        SwMIRPCConstDefs::AttributeNameValue::FAILED_SW_INFO;
};

```

```
/* Constant definitions for the notifyActivateNESwStatusChanged notification */
interface notifyActivateNESwStatusChanged: NotificationIRPNotifications::Notify
{
    const string EVENT_TYPE = "notifyActivateNESwStatusChanged";

    /**
     * This constant defines the name of the activateProcessId property used
     * during activateNESw operation and transported in the
     * filterable_body_fields. The data type for the value of this property
     * is SwMIRPCConstDefs::RequestID.
     */
    const string ACTIVATE_PROCESS_ID =
        SwMIRPCConstDefs::AttributeNameValue::ACTIVATE_PROCESS_ID;

    /**
     * This constant defines the name of the ActivateOperationStatus
     * property which is transported in the filterable_body_fields.
     * The data type for the value of this property is
     * SwMIRPCConstDefs::ActivateOperationStatus.
     */
    const string ACTIVATE_OPERATION_STATUS =
        SwMIRPCConstDefs::AttributeNameValue::ACTIVATE_OPERATION_STATUS;

    /**
     * This constant defines the name of the software version property
     * activated, which is transported in the filterable_body_fields.
     * The data type for the value of this property is
     * SwMIRPCConstDefs::SWVersion.
     */
    const string SW_VERSION_ACTIVATED =
        SwMIRPCConstDefs::AttributeNameValue::SW_VERSION_ACTIVATED;

    /**
     * This constant defines the name of the failureReason property,
     * which is transported in the remaining_body.
     * The data type for the value of this property is
     * SwMIRPCConstDefs::FailureReason.
     */
    const string FAILURE_REASON =
        SwMIRPCConstDefs::AttributeNameValue::FAILURE_REASON;
};

};

#endif // _SWM_IRP_NOTIFICATIONS_IDL_
```

Annex B (informative): Change history

Change history							
Date	TSG #	TSG Doc.	CR	Rev	Subject/Comment	Old	New
2008-12	SP-42	SP-080718	--	--	Submitted to SA#42 for information and approval	1.0.0	8.0.0
2009-06	SP-44	SP-090408	001	--	Add missing start step parameter for resume operation	8.0.0	8.1.0
2009-06	SP-44	SP-090408	002	--	To add downloadNESw functionality to the Software Management CORBA Solution Set	8.1.0	9.0.0
2009-06	SP-44	SP-090408	003	--	To add installNESw functionality to the Software Management CORBA Solution Set	8.1.0	9.0.0
2009-06	SP-44	SP-090408	004	--	To add activateNESw functionality to the Software Management CORBA Solution Set	8.1.0	9.0.0
2009-09	SP-45	SP-090627	005	--	Addition of notifyDownloadNESw StatusChanged functionality to the Software Management CORBA Solution Set	9.0.0	9.1.0
2009-09	SP-45	SP-090627	006	--	Addition of notifyInstallNESw StatusChanged functionality to the Software Management CORBA Solution Set	9.0.0	9.1.0
2009-09	SP-45	SP-090627	007	--	Addition of notifyActivateNESw StatusChanged functionality to the Software Management CORBA Solution Set	9.0.0	9.1.0
2009-09	SP-45	SP-090627	008	--	Remove duplication of SWM functionalities	9.0.0	9.1.0
2009-09	SP-45	SP-090627	009	--	Adding missing error reasons	9.0.0	9.1.0
2009-12	SP-46	SP-090719	010	--	To correct IDL definitions and remove reference to Kernel CM for CORBA SS	9.1.0	9.2.0
2009-12	SP-46	SP-090719	012	--	Remove inconsistent notification parameter	9.1.0	9.2.0
2010-01	--	--	--	--	Removal of track changes	9.2.0	9.2.1
2010-03	SP-47	SP-100035	012	--	Correct the Input parameters of notifyActivateNESw StatusChanged	9.2.1	9.3.0
2010-03	SP-47	SP-100035	013	--	Rapporteur's cleanup	9.2.1	9.3.0
2010-09	SP-49	SP-100488	014	--	Correct IDL compile error	9.3.0	9.4.0
2010-12	SP-50	SP-090831	017	--	Align support qualifier of notifyNewSwAvailability with 32.532	9.4.0	9.5.0