Technical Specification Group Services and System Aspects
Meeting #16, Marco Island, Florida, 10-13 June 2002

**TSGS#16(02)0328**

**Source:** **SA5 (Telecom Management)**

**Title:** **4 new Rel-5 draft v100 TSs 32.321/2/3/4 (Test management IRP; Requirements/ IS/ CORBA SS/ CMIP SS) - for Information**

**Document for:** **Information**

**Agenda Item:** **7.5.3**

| | | | | | |
|---|---|---|---|---|---|
| TS | 32.321 | Telecommunication management; Test management IRP; Requirements | Rel-5 | 1.0.0 | POLLAKOWSKI, Olaf |
| TS | 32.322 | Telecommunication management; Test management IRP; Information service | Rel-5 | 1.0.0 | POLLAKOWSKI, Olaf |
| TS | 32.323 | Telecommunication management; Test management IRP; CORBA solution set | Rel-5 | 1.0.0 | POLLAKOWSKI, Olaf |
| TS | 32.324 | Telecommunication management; Test management IRP; CMIP solution set | Rel-5 | 1.0.0 | POLLAKOWSKI, Olaf |

# 3GPP TS 32.321 V1.0.0 (2002-05)

*Technical Specification*

## 3rd Generation Partnership Project;
## Technical Specification Group Services and System Aspects;
## Telecommunication Management; Test Management;
## Test Management IRP: Requirements;
## (Release 5)

The present document has been developed within the 3rd Generation Partnership Project (3GPP $^{TM}$) 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 $^{TM}$ system should be obtained via the 3GPP Organizational Partners' Publications Offices.

Keywords

Configuration management

***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

# Contents

# Foreword

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

The present document is part the 32.300-series covering the 3rd Generation Partnership Project: Technical Specification Group Services and System Aspects; Telecommunication Management; Notification Management, as identified below:

**32.321: "Test Management Integration Reference Point: Requirements";**

32.322: "Test Management Integration Reference Point: Information Service Version 1";

32.323: "Test Management Integration Reference Point: CORBA Solution Set Version 1:1";

32.324: "Test Management Integration Reference Point: CMIP Solution Set Version 1:1";

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

A 3G telecommunication network is composed of a multitude of different network elements (NE). For a successful operation of the network the operator must be provided with mechanisms allowing him to manage the network. These management activities can be grouped into several areas: configuration management, fault management, performance management, accounting management and security management.

A management function assisting in different high level management areas such as fault management and performance management is test management. The purpose of testing is to get information about the functionality and performance of the 3G managed network subject to the test.

The present document is part of a set of technical specifications defining the telecommunication management (TM) of 3G systems. The TM principles are described in 3GPP TS 32.101 [5]. The TM architecture is described in 3GPP TS 32.102 [6]. The other specifications define the interface (Itf-N) between the managing system (manager), which is in general the network manager (NM) and the managed system (agent), which is either an element manager (EM) or the managed NE itself. The Itf-N is composed of a number of integration reference points (IRPs) defining the information in the agent that is visible for the manager, the operations that the manager may perform on this information and the notifications that are sent from the agent to the manager. One of these IRPs is the Test Management IRP.

Each IRP is specified by four TS, the requirements part, the information service (IS) part, the CORBA solution set (SS) and the CMIP solution set.

# 1 Scope

The present document describes, in addition to the requirements defined in [1] and [2], the requirements for the Test Management IRP.

# 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 TS 32.101: "3G Telecom Management principles and high level requirements".

[2] 3GPP TS 32.102: "3G Telecom Management architecture".

[3] ITU-T Recommendation X.745:"Information Technology-Open Systems Interconnection-Systems Management: Test Management".

[4] 3GPP TS 32.301: "Notification IRP: Requirements".

[5] 3GPP TS 32.304: "Notification IRP: CMIP Solution Set".

[6] 3GPP TS 32.xxx: "Log Management IRP: Requirements".

[7] 3GPP TS 32.xxx: "Log Management IRP: CMIP Solution Set".

[8] 3GPP TS 32.671: "State Management IRP: Requirements".

[9] 3GPP TS 32.672: "State Management IRP: CMIP Solution Set".

[10] ITU-T Recommendation X.737: "Information Technology; Open Systems Interconnection; Systems Management: Confidence and Diagnostic Test Categories"

# 3 Definitions and Abbreviations

## 3.1 Definitions

**Test Action Request Receiver (TARR)**: An instance of a managed object class with attributes, operations and notifications allowing to receive test requests and to create and delete TOs. The managed object may also have attributes, notifications and operations pertaining to other capabilities than the TARR functionality.

**Test Category**: One or more tests sharing a common purpose and similar characteristics.

**Test Object (TO)**: A managed object that is instantiated for the purpose of monitoring and controlling a test invocation. Each test invocation has one associated TO. TOs are created and deleted by managed objects with TARR functionality.

## 3.2 Abbreviations

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

| | |
|---|---|
| NE | Network Element |
| NM | Network Management Center |
| TARR | Test Action Request Receiver |
| TO | Test Object |

# 4 Purpose

The purpose of testing is to get information about the functionality and performance of the 3G managed network subject to the test. This information can be used in other network management areas, for instance in fault management.

Testing is an activity that involves the operator, the managing system (the OS) and the managed system (the NE). Generally the operator requests the execution of tests from the OS and the managed NE autonomously executes the tests without any further support from the operator.

In case of intrusive tests it is required that the network resources to be tested are locked prior to test execution. During the test execution the telecommunication service provided by the network resources is interrupted. After completion of the test, depending on  the test result, the network resources shall be set to the most appropriate state.

Test management capabilities should be provided over the Itf-N in order to allow the NM operator to perform tests on network resources. This capability is especially important at times when only the NM but not the OMC is attended, which might be the case at night or during weekends.

In the context of fault management the NM operator may use the testing capabilities over the Itf-N for numerous purposes

- When a fault has been detected and if the information provided in the alarm report is not sufficient to localise the faulty resource, tests can be executed in order to localise the fault.
- When a fault has been detected and if the information provided in the alarm report specifies the faulty resource, tests can be executed on that resource in order to determine the required repair action.
- During normal operation of the NE, tests can be executed for the purpose of discovering undetected faults.
- After a faulty resource has been repaired or replaced and before it is restored to service, tests can be executed on that resource in order to make sure that it is fault free.

However, regardless of the context where the testing is used, its target is always the same: verify if a system's physical or functional resource performs properly and, in case it happens to be faulty, provide all the information to help the operator to localise and correct the fault.

The requirements for the test management service shall be based on ITU-T Recommendation X.745 [3].

# 5 Requirements

## 5.1 Overview

The IRPManager (test conductor) must be able to initiate and terminate tests. For this purpose special test initiation and termination requests may be used. The IRPAgent (test performer) must be able to receive and react upon these requests. Special objects may be created for the purpose of monitoring and control of the test execution. During test execution the IRPManager shall be able to monitor the test.

Test results shall be made available to the IRPManager by test result reports (notifications). It shall be possible to log these test result reports.

This gives rise to the following requirements for the test management function to be satisfied

- the ability for the manager to initiate tests
- the ability for the manager to terminate tests

- the ability for the manager to monitor the test execution
- the specification of a mechanism to report the test results to the manager
- the specification of a mechanism to log test results in the agent

These capabilities are outlined in the following sections in more detail.

## 5.2.1 Test Initiation

The IRPManager shall be able to initiate a test in the IRPAgent by sending a test request to the IRPAgent. The IRPAgent must have at least one object instance capable of receiving these test requests. This functionality is called Test Action Request Receiver (TARR) functionality. In response to a test request one or more tests shall be initiated. Each test shall be controlled and monitored individually by special objects instantiated exclusively for this purpose. These objects are called Test Objects (TOs). Each of the tests initiated in response to a test request shall have a single associated TO.

The test request shall include the following information

- The network resources to be tested
- Information about the managed objects (e. g. TOs) assisting in the test execution
- Any other information useful for the test execution.

In response to the test request the IRPAgent shall send a test request response to the IRPManager. For a successful test request this response shall contain the following information

- A unique identifier for each test initiated by the test request
- Information about the managed objects assisting to execute the test

For a failed test request the response shall contain the following information

- Information about the reason for the failure

## 5.2.2 Test Termination

A test may terminate in two different ways, either by request (explicit test termination) or spontaneously (implicit test termination).

### 5.2.2.1 Explicit Test Termination

During the lifetime of a test explicit test termination may be requested by the IRPManager in two different ways

- Emission of a test termination request
- Deletion of the TO related to the test

The test termination request must be directed to the object in the IRPAgent which received the test request. The test termination request shall provide the following information

- The identifiers of the tests to be terminated

All TO(s) related to the tests to be terminated shall be deleted.

After reception of a test termination request a test termination response shall be generated by the IRPAgent and forwarded to the IRPManager. In case the test termination request is successful the test termination response shall contain the following information

- The identifiers of all tests that have been successfully terminated.

If one or more of the tests specified in the test termination request cannot be terminated a test termination request failure response shall be generated. This response shall contain the following information

- The identifiers of the tests that could not be deleted

- Information about the reason for the failure

#### 5.2.2.2 Implicit Test Termination

Implicit test termination may be triggered by three events

- Fulfilment of conditions for a successful completion of the test
- Fulfilment of conditions for a premature termination of the test, e. g. expiry of a time-out period
- Occurrence of abnormal conditions, e. g. fault situations

### 5.2.3 Test Monitoring

The IRPManager shall be able to monitor the tests, i. e. get information about the tests while they are still executing. For this purpose the IRPManger shall be able to inquire the values of attributes containing specific information about the test execution.

### 5.2.4 Test Result Reporting

Test results shall be made available to the IRPManager by one or more notifications emitted by the TO that is related to the test. These notifications shall be generated in an automatic manner upon occurrence of predefined triggering events without that any action of the IRPManager is required (unsolicited reporting). The triggering events depend on the test category and shall be defined by the TO class behaviour.

Test result notifications may be emitted during the test execution (intermediate test result reporting) and at the end of the test execution (final test result reporting).

The event triggering the emission of final test result reports is the termination of the test execution, irrespective of if the test terminates by request or spontaneously except for the case where the test is terminated by deleting its related TO. This applies to all test categories.

The events triggering intermediate test result reporting depend on the test category. They may include

- Arrival of the test execution at specified points, for example at the end of test phases
- Expiry of specified reporting time intervals
- Fulfilment of certain predefined criteria

A test result notification shall include the following information

- The identifier of the test for which the results are reported
- Information about the outcome of the test, e. g. if the test terminated successfully upon test completion or prematurely
- Information about the test and the test execution, e. g. tested network resources, proposed repair actions in case of fault detection
- Any other useful information pertaining to the test

For test result reporting the Notification IRP as defined in 3GPP TS 32.301 [4] to 3GPP TS 32.304 [5] shall be used. According to the operations provided by the Notification IRP the IRPManager is able to specify filter conditions selecting the notifications that are forwarded over the Itf-N to the IRPManager. Notifications not satisfying the filter conditions are discarded.

## 5.3 Test Result Logging

It shall be possible to store the test results in the IRPAgent (test performer) for later retrieval by the IRPManager.

This log functionality shall be realised using the LogIRP defined in 3GPP TS 32.xxx [6] to 3GPP TS 32.xxx [7]. According to the mechanism defined by the Log IRP an appropriate log has to be instantiated in the IRPAgent. The test result notifications to be stored are selected by a filter mechanism. The filter conditions are specified by the IRPManager.

## 5.4 Test States

For the purpose of monitoring tests TOs shall provide information about the current state of a test. Therefore each TO shall support the operational state attribute and the procedural status attribute. Both attributes are defined in 3GPP TS 32.672 [9]. The operational state may have one of the following values: enabled, disabled. The procedural status may assume one or more of the following values: not initialised, initialising, reporting, terminating

In addition to this, a special test state attribute shall be supported. The test state may assume one of the following values: not initialised, idle, initialising, testing, terminating, suspended, disabled. The actual test state value shall be derived from the actual operational state and procedural status according to the mapping table specified in ITU-T X.745 [3].

Any state change shall be reported to the NM using the Notification IRP.

## 5.5 Test Categories

Tests can be classified into test categories. In ITU-T X.737 [10] eight different test categories are defined. This specification stipulates support for the resource self test. Other test categories may be added in the future.

### 5.5.1 Resource Self Test

Resource self tests are used to investigate the ability of a simple resource in the managed network (e. g. a hardware board) to perform its allotted function. For the specification of the resource to be tested a single MORT is required. No associated objects are necessary.

The resource self test may be intrusive or non-intrusive. In case of intrusive tests the MORT has to be placed in the appropriate state before the test may start. If this is not possible the test request is rejected.

# Annex A (informative): Change history

| Change history | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Date** | **TSG #** | **TSG Doc.** | **CR** | **Rev** | **Subject/Comment** | **Old** | **New** |
| Jun 2002 | S_16 | SP-020328 | -- | -- | Submitted to TSG SA #16 for Information | 1.0.0 | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |

# 3GPP TS 32.324 V1.0.0 (2002-05)

*Technical Specification*

## 3rd Generation Partnership Project;
## Technical Specification Group Services and System Aspects;
## Telecommunication Management; Test Management;
## Test Management IRP: CMIP SS;
## (Release 5)

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

Configuration management

***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

# Contents

# Foreword

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

The present document is part the 32.300-series covering the 3rd Generation Partnership Project: Technical Specification Group Services and System Aspects; Telecommunication Management; Notification Management, as identified below:

32.321: “Test Management Integration Reference Point: Requirements”;

32.322: “Test Management Integration Reference Point: Information Service Version 1”;

32.323: “Test Management Integration Reference Point: CORBA Solution Set Version 1:1”;

**32.324: “Test Management Integration Reference Point: CMIP Solution Set Version 1:1”;**

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

A 3G telecommunication network is composed of a multitude of different network elements (NE). For a successful operation of the network the operator must be provided with mechanisms allowing him to manage the network. These management activities can be grouped into several areas: configuration management, fault management, performance management, accounting management and security management.

A management function assisting in different high level management areas such as fault management and performance management is test management. The purpose of testing is to get information about the functionality and performance of the 3G managed network subject to the test.

The present document is part of a set of technical specifications defining the telecommunication management (TM) of 3G systems. The TM principles are described in 3GPP TS 32.101 [5]. The TM architecture is described in 3GPP TS 32.102 [6]. The other specifications define the interface (ITf-N) between the managing system (manager), which is in general the network manager (NM) and the managed system (agent), which is either an element manager (EM) or the managed NE itself. The Itf-N is composed of a number of integration reference points (IRPs) defining the information in the agent that is visible for the manager, the operations that the manager may perform on this information and the notifications that are sent from the agent to the manager. One of these IRPs is the Test IRP.

Each IRP is specified by four TS, the requirements part, the information service (IS) part, the CORBA solution set (SS) and the CMIP solution set.

# 1 Scope

The present document specifies the Common Management Information Protocol (CMIP) Solution Set (SS) for the Test Management IRP: Information Service defined in 3GPP TS 32.322 [8]. In detail:

- Clause 4 contains an introduction to some concepts that are the base for some specific aspects of the CMIP interfaces.
- Clause 5 contains the GDMO definitions for the Test Management IRP over the CMIP interfaces
- Clause 6 contains the ASN.1 definitions supporting the GDMO definitions provided in clause 5

This Solution Set specification is related to 3GPP 32.322 V5.0.0.

# 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 TS 32.101: "3G Telecom Management principles and high level requirements".

[2] 3GPP TS 32.102: "3G Telecom Management architecture".

[4] 3GPP TS 32.301: " Telecommunication management; Configuration Management; Notification IRP: Requirements".

[5] 3GPP TS 32.304: " Telecommunication management; Configuration Management; Notification Integration Reference Point: CMIP Solution Set".

[6] 3GPP TS 32.312: " Telecommunication management; Generic IRP management; Information service".

[7] 3GPP TS 32.321: " Telecommunication management; Test management IRP; Requirements".

[8] 3GPP TS 32.322: " Telecommunication management; Test management IRP; Information service".

[9] 3GPP TS 32.xxx: "Log Management IRP: Requirements".

[10] 3GPP TS 32.xxx: "Log Management IRP: CMIP Solution Set".

[11] 3GPP TS 32.671: " Telecommunication management; 3G Configuration Management; State Management IRP: Requirements".

[12] 3GPP TS 32.672: " Telecommunication management; 3G Configuration Management; State Management IRP: Information service".

[13] ITU-T Recommendation X.710: “Information Technology - Open Systems Interconnection - Common Management Information Service”

[14] ITU-T Recommendation X.745:”Information Technology - Open Systems Interconnection - Systems Management: Test Management”.

# 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 32.321 [7] apply.

## 3.2 Abbreviations

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

| | |
|---|---|
| IOC | Information Object Class |
| MOC | Managed Object Class |
| NE | Network Element |
| NMC | Network Management Center |
| TARR | Test Action Request Receiver |
| TO | Test Object |

# 4 Basic Aspects

## 4.1 Mapping

The semantics of the Test Management IRP are defined in 3GPP TS 32.322 [8]. The definitions of the management information defined there are independent of any implementation technology and protocol. This clause maps these protocol independent definitions onto their equivalents of the CMIP Solution Set of the Test Management IRP.

### 4.1.1 Mapping of Information Object Classes

Table 1 maps the IOCs defined in 3GPP TS 32.322 Test Management IRP: Information Service [8] onto the corresponding managed object classes (MOCs) defined in this CMIP Solution Set. The MOCs are qualified either as mandatory (M) or optional (O).

**Table 1: Mapping of IOC**

| IS IOC Name | MOC or Attributes of the CMIP solution set | Qualifier |
|---|---|---|
| TestManagementIRP | testManagentIRP | M |
| TestActionPerformer | testActionPerformer (ITU-T X.745 [14]) | M |
| TesterObject | testObject (ITU-T X.745 [14]) | M |
| TestInvocation | testObject (ITU-T X.745 [14]) | M |
| ResourceSelfTestObject | resourceSelfTestObject (ITU-T X.745 [14]) | M |

### Mapping of Information Object Class Attributes

### 4.1.2 Mapping of Operations

Table 2 and Table 3 map the operations defined in 3GPP TS 32.322 Test Management IRP: Information Service [8] and 3GPP TS 32.312 Generic IRP Management: Information Service [6] onto corresponding GDMO actions and CMISE services. The operations are qualified either as mandatory (M) or optional (O).

The CMISE services are defined in ITU-T REc. X.710 [13].

**Table 2: Mapping of operations of the Test Management IRP: IS**

| Interface | Operation | GDMO Action or CMISE of CMIP SS | Qualifier |
|---|---|---|---|
| TestManagementIRPControlOperations | initiateTest | testRequestControlledAction<br>(ITU-T X.745 [14]) | M |
| | terminateTest | testTerminateAction<br>(ITU-T X.745 [14]) | M |
| TestIManagementRPMonitorOperations | monitorTest | M-GET (CMISE)<br>Retrieval of the test object attribute values for the operational state, the procedural state, the test state and the test outcome | M |

**Table 3: Mapping of operations inherited from the Generic IRP Management: IS**

| Interface | Operation | GDMO Action or CMISE of CMIP SS | Qualifier |
|---|---|---|---|
| GenericIRPVersionsOperations | getIRPVersion | getTestManagementIRPVersion | M |
| GenericIRPProfileOperations | getOperationProfile | getOperationProfile | O |
| | getNotificationProfile | getNotificationProfile | O |

## 4.1.3 Mapping of Operation Parameters

The tables in the following subclauses show the parameters of each operation defined in the Test Management IRP: Information Service [8] and their equivalents in the CMIP Solution Set.

### 4.1.3.1 Parameter Mapping of the Operation *initiateTest*

The operation *initiateTest* is mapped to the GDMO action *testRequestControlledAction* defined in ITU-T X.745 [14]. This action shall be implemented using the CMISE M-ACTION service.

All input parameters are mapped to the M-ACTION request parameter ‘Action information’. The syntax and semantics of this parameter is specified in ITU-T X.745 by the ASN.1 definition *TestRequestControlledInfo*.

The output parameter *successResponse* is mapped to the M-ACTION response parameter ‘Action reply’, which is specified in ITU-T X.745 by the the ASN.1 definition *TestRequestControlledResponse*.

**Table 4: Parameter mapping of the operation *initiateTest***

| IS Parameter Name | IN/OUT | CMIP SS Equivalent | Qualifier |
|---|---|---|---|
| testInvocationInitiator | IN | This parameter is conditional and not used in the CMIP SS. | -- |
| maxTestingStateDuration | | TestRequestControlledInfo: timeoutPeriod | |
| toBeInitiatedTests: toBeTestedMORT | IN | M-ACTION parameter ‘Action information’ (TestRequestControlledInfo): toBeTestedMORTs | M |
| toBeInitiatedTests: testerObjectClass | IN | TestRequestControlledInfo: testObjectList: tOClass | M |
| toBeInitiatedTests: testerObjectInitialAttributeList | | TestRequestControlledInfo: testObjectList: initialAttributeList | |
| response | OUT | All tests were successfully initiated:<br>TestRequestControlledResponse: CHOICE independentTestResponseList: testInvocationId<br><br>At least one test failed to be initiated:<br>M-ACTION response parameter ‘Errors’ | M |

### 4.1.3.1 Parameter Mapping of the Operation *terminateTest*

The operation *initiateTest* is mapped to the GDMO action *testTerminateAction* defined in ITU-T X.745 [14]. This action shall be implemented using the CMISE M-ACTION service.

The M-ACTION request parameter ‘Action information’ is specified in ITU-T X.745 by the ASN.1 definition *TestTerminateInfo*, which is the CMIP SS equivalent of the IS parameter *indicatedTest*.

The parameter *successresponse* is mapped to the M-ACTION response parameter ‘Action reply’, which is specified in ITU-T X.745 by the the ASN.1 definition *TestTerminateResult*.

**Table 5: Parameter mapping of the operation *terminateTest***

| IS Parameter Name | IN/OUT | CMIP SS Equivalent | Qualifier |
|---|---|---|---|
| tARRObjectInstance | IN | M-ACTION request parameter ‘Base object instance’ | M |
| indicatedTests | IN | TestTerminateInfo: indicatedTests with CHOICE SET OF testInvocationId | M |
| successResponse | OUT | TestTerminateResult | M |
| failureResponse | OUT | M-ACTION response parameter ‘Errors’ | M |

### 4.1.3.1 Parameter Mapping of the Operation *monitorTest*

The TO attributes reflecting the status of the test can be retrieved by the manager using the CMISE M-GET service. The TOs to be monitored may be selected by a scoping and filtering mechanism.

The attribute values are returned in the M-GET response parameter ‘Attribute list’.

**Table 6: Parameter mapping of the operation *monitorTest***

| IS Parameter Name | IN/OUT | CMIP SS Equivalent | Qualifier |
|---|---|---|---|
| baseObjectInstance | IN | M-GET request parameter ‘Base object instance’ | M |
| scope | IN | M-GET request parameter ‘Scope’ | O |
| filter | IN | M-GET request parameter ‘Filter’ | O |
| --- | IN | M-GET request parameter ‘Attribute identifier list’: attribute identifier of the TO attributes *operationalState*, *proceduralStatus*, *testState* and *testOutcome* | O |
| monitoredAttributes | OUT | M-GET response parameter ‘Attribute list’: attribute identifier and value of the TO attributes *operationalState*, *proceduralStatus*, *testState* and *testOutcome* | M |
| error | OUT | M-GET response parameter ‘Errors’ | M |

## 4.1.4 Mapping of Notifications

The notification *notifyTestResults* is mapped to the GDMO notification *testResultNotification* defined in ITU-T X.745 [14]. This notification shall be implemented using the CMISE M-EVENT-REPORT service.

**Table 7: Mapping of notifications of the Test Management IRP IS**

| Interface | Operation | GDMO Notification or CMISE of CMIP SS | Qualifier |
|---|---|---|---|
| TestManagementIRPNotifications | notifyTestResults | testResultNotification | M |

## 4.1.5 Mapping of Notification Parameters

The tables in the following subclauses show the parameters of each notification defined in the 3GPP TS 32.322 Test Management IRP: Information Service [8] and their equivalents in the CMIP Solution Set.

### 4.1.5.1 Parameter Mapping of the Notification *notifyTestResults*

All parameters defined in the IS are mapped to the M-EVENT-REPORT parameter ‘Event information’. The syntax and semantics of this structured parameter is defined in ITU-T X.745 by the ASN.1 definition *TestResultInfo*.

**Table 8: Parameter mapping of the notification *notifyTestResults***

| IS Parameter Name | CMIP SS Equivalent | Qualifier |
|---|---|---|
| objectClass | M-EVENT REPORT parameter ‘Managed object class’ | M |
| objectInstance | M-EVENT REPORT parameter ‘Managed object instance’ | M |
| notificationId | M-EVENT REPORT parameter ‘Event information’ (TestResultInfo): notificationIdentifier | O |
| eventTime | M-EVENT REPORT parameter ‘Event time’ | M |
| systemDN | This IS parameter is conditional and not used in the CMIP SS. | -- |
| notificationType | M-EVENT REPORT parameter ‘Event type’ | M |
| testInvocationId | M-EVENT REPORT parameter ‘Event information’ (TestResultInfo): testInvocationId | O |
| testInvocationInitiator | This IS parameter is conditional and not used in the CMIP SS. | -- |
| testOutcome | M-EVENT REPORT parameter ‘Event information’ (TestResultInfo): testOutcome | O |
| mORT | M-EVENT REPORT parameter ‘Event information’ (TestResultInfo): mORTs | O |
| proposedRepairActions | M-EVENT REPORT parameter ‘Event information’ (TestResultInfo): proposedRepairActions | O |
| additionalInformation | M-EVENT REPORT parameter ‘Event information’ (TestResultInfo): additionalInformation | O |
| fileReference | M-EVENT REPORT parameter ‘Event information’ (TestResultInfo): additionalInformation Define this parameter as GraphicString | M, see Note 1 |
| fileExpiryDate | M-EVENT REPORT parameter ‘Event information’ (TestResultInfo): additionalInformation Define this parameter as GeneralizedTime | M, see Note 1 |

Note 1: This parameter contains only information, if the test result data are captured in a file. Otherwise it shall contain no information or be absent.

# 5 GDMO Definitions

## 5.1 Managed Object Classes

### 5.1.1 testManagementIRP

```
testManagementIRP MANAGED OBJECT CLASS
  DERIVED FROM
    "Rec. X.721 | ISO/IEC 10165-2 : 1992":top;
  CHARACTERIZED BY
    testMangementIRPIdPackage,
    testMangementIRPVersionPackage;
  CONDITIONAL PACKAGES
    testManagementIRPProfilePackage PRESENT IF “an instance supports it”;
REGISTERED AS {ts32-324ObjectClass 1};
```

## 5.2 Packages

### 5.2.1 testManagementIRPIdPackage

```
testManagementIRPIdPackage PACKAGE
  BEHAVIOUR
    testManagementIRPIdPackageBehaviour;
  ATTRIBUTES
    testManagementIRPId;
  REGISTERED AS {ts32-324Package 1};

testManagementIRPIdPackageBehaviour BEHAVIOUR
DEFINED AS
```

"An instance of the MOC *testManagementIRP* is identified by the value of the attribute *testManagementIRPId*.";

### 5.2.2 testManagementIRPVersionPackage

```
testManagementIRPVersionPackage PACKAGE
  BEHAVIOUR
    testManagementIRPVersionPackageBehaviour;
  ATTRIBUTES
    supportedTestManagementIRPVersions    GET;
  ACTIONS
    getTestManagementIRPVersion;
REGISTERED AS {ts32-324Package 2};
```

```
notificationIRPVersionPackageBehaviour BEHAVIOUR
DEFINED AS
```

“This package has been defined to allow the IRPManager to get information about the Test Management IRP versions supported by the IRPAgent.

The attribute *supportedTestManagementIRPVersions* indicates all versions of the Test Management IRP currently supported by the IRPAgent.

The action *getTestManagementIRPVersion* is invoked by the IRPManager to get information about the Test Management IRP versions supported by the IRPAgent.”;

### 5.2.3 testManagementIRPProfilePackage

```
testManagementIRPProfilePackage PACKAGE
  BEHAVIOUR
    testManagementIRPProfilePackageBehaviour;
  ACTIONS
    getOperationProfile,
    getNotificationProfile;
REGISTERED AS {ts32-324Package 3};
```

**DEFINED AS**

"This package has been defined to allow the Manager to get detailed information about the profile of the Test Management IRP. The action *getOperationProfile* is invoked by the Manager to get detailed information about the operations supported by the Test Management IRP. The action *getNotificationProfile* is invoked by the Manager to get detailed information about the notifications supported by the Test Management IRP.";

## 5.3 Actions

### 5.3.1 getTestManagementIRPVersion (M)

```
getTestManagementIRPVersion ACTION
  BEHAVIOUR
    getTestManagementIRPVersionBehaviour;
  MODE
    CONFIRMED;
  WITH REPLY SYNTAX
    TS32-324TypeModule.GetTestManagementIRPVersionReply;
REGISTERED AS {ts32-324Action 1};
```

```
getTestManagementIRPVersionBehaviour BEHAVIOUR
DEFINED AS
```

"The Manager invokes this action to get information about the Test Management IRP versions supported by the Agent. The 'Action information' field contains no data. The 'Action response' is composed of the following data:

- *versionNumbersList*
- *status*

The parameter *versionNumbersList* defines a list of Test Management IRP versions supported by the Agent. A list containing no element, i.e. a NULL list, means that the concerned Agent doesn't support any version of the Test

Management IRP. The parameter *status* contains the results of the Manager action. Possible values: noError (0), error (the value indicates the reason of the error).";

## 5.3.2 getNotificationProfile (O)

```
getNotificationProfile ACTION
  BEHAVIOUR
    getNotificationProfileBehaviour;
  MODE
    CONFIRMED;
  WITH INFORMATION SYNTAX
    TS32-324TypeModule.IRPVersionNumber;
  WITH REPLY SYNTAX
    TS32-324TypeModule.GetNotificationProfileReply;
REGISTERED AS {ts32-324Action 2};
```

```
getNotificationProfileBehaviour BEHAVIOUR
DEFINED AS
```

"A Manager invokes this action to enquiry about the notification profile (supported notifications and supported parameters) for this specific Test Management IRP version.

The 'Action information' contains the following data:

- *irpVersionNumber*

This mandatory parameter identifies the Test Managemnt IRP version.

The 'Action response' is composed of the following data:

- *notificationNameProfile*
- *notificationParameterProfile*
- *status*

The parameter *notificationNameProfile* contains a list of notification names, i.e. a NULL list means that the Test Management IRP doesn't support any notification. The parameter *notificationParameterProfile* contains a set of elements, each element corresponds to a notification name and is composed by a set of parameter names. The parameter *status* contains the results of this action. Possible values: noError (0), error (the value indicates the reason of the error).";

## 5.3.3 getOperationProfile (O)

```
getOperationProfile ACTION
  BEHAVIOUR
    getOperationProfileBehaviour;
  MODE
    CONFIRMED;
  WITH INFORMATION SYNTAX
    TS32-324TypeModule.IRPVersionNumber;
  WITH REPLY SYNTAX
    TS32-324TypeModule.GetOperationProfileReply;
REGISTERED AS {ts32-324Action 3};
```

```
getOperationProfileBehaviour BEHAVIOUR
DEFINED AS
```

"A Manager invokes this action to enquiry about the operation profile (supported operations and supported parameters) for this specific Test Management IRP version.

The 'Action information' contains the following data:

- *irpVersionNumber*

This mandatory parameter identifies the Test Management IRP version.

The 'Action response' is composed of the following data:

- *operationNameProfile*
- *operationParameterProfile*
- *status*

The parameter *operationNameProfile* contains a list of operation names. The parameter *operationParameterProfile* contains a set of elements, each element corresponds to an operation name and is composed by a set of parameter names. The parameter *status* contains the results of this action. Possible values: noError (0), error (the value indicates the reason of the error).";

## 5.4 Attributes

### 5.4.1 testManagementControlId

```
testManagementControlId ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    TS32-324TypeModule.GeneralObjectId;
  MATCHES FOR
    EQUALITY;
  BEHAVIOUR
    testManagementControlIdBehaviour;
REGISTERED AS {ts32-324Attribute 1};

testManagementControlIdBehaviour BEHAVIOUR
DEFINED AS
```

"This attribute names an instance of the MOC *testManagementControl*.";

### 5.4.2 supportedTestManagementIRPVersion

```
supportedTestManagementIRPVersions ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    TS32-324TypeModule.SupportedTestManagementIRPVersions;
  MATCHES FOR
    EQUALITY;
  BEHAVIOUR
    supportedTestManagementIRPVersionsBehaviour;
REGISTERED AS {ts32-324Attribute 2};

supportedTestManagementIRPVersionsBehaviour BEHAVIOUR
DEFINED AS
```

"This attribute provides the information concerning the Test Management IRP versions currently supported by the Agent.";

# 6 ASN.1 Definitions

```
TS32-324TypeModule {itu-t(0) identified-organization(4) etsi(0) mobileDomain(0) umts-Operation-
Maintenance(3) ts32-324(324) informationModel(0) asn1Module(2) version1(1)}

DEFINITIONS IMPLICIT TAGS ::=
BEGIN
--EXPORTS everything
--IMPORTS nothing

baseNodeUMTS           OBJECT IDENTIFIER ::= { itu-t (0) identified-organization (4) etsi (0)
                                               mobileDomain (0) umts-Operation-Maintenance (3)}

ts32-324               OBJECT IDENTIFIER ::= {baseNodeUMTS ts32-324                (324)}
ts32-324InfoModel      OBJECT IDENTIFIER ::= {ts32-324 informationModel            (  0)}
ts32-324ObjectClass    OBJECT IDENTIFIER ::= {ts32-324InfoModel managedObjectClass (  3)}
ts32-324Package        OBJECT IDENTIFIER ::= {ts32-324InfoModel package            (  4)}
ts32-324Parameter      OBJECT IDENTIFIER ::= {ts32-324InfoModel parameter          (  5)}
ts32-324Attribute      OBJECT IDENTIFIER ::= {ts32-324InfoModel attribute          (  7)}
ts32-324Action         OBJECT IDENTIFIER ::= {ts32-324InfoModel action             (  9)}
ts32-324Notification   OBJECT IDENTIFIER ::= {ts32-324InfoModel notification       ( 10)}

ErrorCauses ::= ENUMERATED
{
```

```
noError (0),                  -- operation / notification successfully performed
unspecifiedErrorReason (255)  -- operation failed, specific error unknown
}

GetNotificationProfileReply ::= SEQUENCE
{
notificationNameProfile       NotificationList,
notificationParameterProfile  ParameterListOfList,
status                        ErrorCauses
}

GetOperationProfileReply ::= SEQUENCE
{
operationNameProfile          OperationList,
operationParameterProfile     ParameterListOfList,
status                        ErrorCauses
}

GetTestManagementIRPVersionReply ::= SEQUENCE
{
versionNumberList             SupportedTestManagementIRPVersions,
status                        ErrorCauses
}

GeneralObjectId ::= INTEGER

IRPVersionNumber ::= GraphicString

NotificationList ::= SET OF NotificationName

NotificationName ::= GraphicString

OperationList ::= SET OF OperationName

OperationName ::= GraphicString

ParameterList ::= SET OF ParameterName

ParameterListOfList ::= SET OF ParameterList

ParameterName ::= GraphicString

SupportedTestManagemntIRPVersions ::= SET OF IRPVersionNumber
```

END – of module TS32-324TypeModule

# Annex A (informative): Change history

| Change history | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Date** | **TSG #** | **TSG Doc.** | **CR** | **Rev** | **Subject/Comment** | **Old** | **New** |
| Jun 2002 | S_16 | SP-020328 | -- | -- | Submitted to TSG SA #16 for Information | 1.0.0 | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |

# 3GPP TS 32.323 V1.0.0 (2002-05)

*Technical Specification*

## 3rd Generation Partnership Project;
## Technical Specification Group Services and System Aspects;
## Telecommunication Management;
## Test Management;
## Part 3: Test Management IRP: CORBA Solution Set
## (Release 5)

The present document has been developed within the 3rd Generation Partnership Project (3GPP TM) 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 TM system should be obtained via the 3GPP Organizational Partners' Publications Offices.

Keywords

Fault Management, Alarms

***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

# Contents

# Foreword

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

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

Part 1: “3G Fault Management Requirements”;

Part 2: “Test Management Integration Reference Point: Information Service”;

**Part 3: “Test Management Integration Reference Point: CORBA Solution Set”;**

Part 4: “Test Management Integration Reference Point: CMIP Solution Set”.

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.

# 1 Scope

The present document specifies the CORBA Solution Set (SS) for the IRP whose semantics is specified in Test Management IRP: Information Service (IS) (3G TS 32.322 [6]).

Clause 1 to 3 provides background information. Clause 4 provides key architectural features supporting the SS. Clause 5 defines the mapping of operations, notification, parameters and attributes defined in IS to their SS equivalents. Clause 6 describes the notification interface containing the push method. Annex A contains the IDL specification.

This Solution Set specification is related to 3G TS 32.323 V5.0.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.

[1] OMG TC Document telecom/98-11-01: “OMG Notification Service”.

[2] OMG CORBA Services: “Common Object Services Specification, Update: November 22, 1996” (Clause 4 contains the Event Service specification).

[3] 3GPP TS 32.300: " Telecommunication management; 3G configuration management; Name convention for Managed Objects". [not used in the document]

[4] 3GPP TS 32.302: "Telecommunication management; Configuration Management; Notification Integration Reference Point; Information Service". [not used in the document]

[5] 3GPP TS 32.303: " Telecommunication management; Configuration Management; Notification Integration Reference Point; CORBA solution set".

[6] 3GPP TS 32.322: " Telecommunication management; Test management IRP; Information service".

[7] 3GPP TS 32.312: " Telecommunication management; Generic IRP management; Information service".

# 3 Definitions and abbreviations

## 3.1 Definitions

In addition to the terms and definitions defined in TS 32.322 [6], there are no additional definitions applicable to this document.

## 3.2 Abbreviations

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

| | |
|---|---|
| CORBA | Common Object Request Broker Architecture |
| IDL | Interface Definition Language |
| IRP | Integration Reference Point |

| | |
|---|---|
| IOC | Information Object Class |
| IS | Information Service |
| NE | Network Element |
| OMG | Object Management Group |
| SS | Solution Set |

## 3.3 IRP document version number string

The IRP document version number (sometimes called “IRP version” or “version number”) string is used to identify this specification. The string is derived using the following rule.

Take the 3GPP document number on the front page of this specification, such as “3GPP TS 32.ABC-D VE.F.G (2000-12)”. Discard the leading “3GPP TS ”. Discard all characters after and including the last period. Eliminate leading and trailing spaces. Reduce multiple consecutive spaces with one space. Express the resultant in a string. Capitalised the string. For example, if the 3GPP document version number is “3GPP TS 32.106-3 V3.2.0 (2000-12)”, then the IRP document version number shall be “32.106 V3.2”.

This string is returned in `get_test_management_IRP_version` method and is carried in the first field of the notification header of all notifications related to Test Management IRP. This string is also returned in `get_notification_categories` method of the Notification IRPAgent, in case that IRPAgent is responsible for emitting notifications related to Test Management IRP.

# 4 Architectural Features

The overall architectural feature of Test Management IRP is specified in 3G TS 32.322 [6]. This clause specifies features that are specific to the CORBA SS.

## 4.1 Notification Services

In implementations of CORBA SS, IRPAgent conveys Test Management Information to IRPManager via OMG Notification Service (OMG Notification Service [1]).

OMG Event Service [2] provides event routing and distribution capabilities. OMG Notification Service provides, in addition to Event Service, event filtering and Quality Of Service (QOS) as well.

A necessary and sufficient sub set of OMG Notification Services shall be used to support `Test ManagementIRPNotifications` notifications as specified in 3G TS 32.322 [6].

## 4.2 Push and Pull Style

OMG Notification Service defines two styles of interaction. One is called push style. In this style, IRPAgent pushes notifications to IRPManager as soon as they are available. The other is called pull style. In this style, IRPAgent keeps the notifications till IRPManager requests for them.

This Notification CORBA SS [5] specifies that support of Push style is Mandatory (M) and that support of Pull style is Optional (O).

## 4.3 Support multiple notifications in one `push` operation

For efficiency reasons, IRPAgent may send multiple notifications using one single `push` operation. To pack multiple notifications into one `push` operation, IRPAgent may wait and not invoke the `push` operation as soon as notifications are available. To avoid IRPAgent to wait for an extended period of time that is objectionable to IRPManager, IRPAgent shall implement an IRPAgent wide timer configurable by administrator. On expiration of this timer, IRPAgent shall invoke `push` if there is at least one notification to be conveyed to IRPManager. This timer is re-started after each `push` invocation.

# 5 Mapping

## 5.1 Operation and Notification mapping

Test Management IRP: IS 3G TS 32.322 [6] defines semantics of operation and notification visible across the Test Management IRP. Table 1 indicates mapping of these operations and notifications to their equivalents defined in this SS.

**Table 1: Mapping from IS Operations and Notification to SS equivalents**

| IS Operations/ notification 3G TS 32.322 [6] | SS Method | Qualifier |
|---|---|---|
| initiateTests | initiate_tests | M |
| terminateTests | terminate_tests | M |
| monitorTest | monitor_test | M |
| getIRPVersion | get_test_management_IRP_version | M |
| getOperationProfile (see note) | get_test_management_IRP_operation_profile | O |
| getNotificationProfile (see note) | get_test_management_IRP_notification_profile | O |
| notifyTestResult | push_structured_event (See clause 6.1) | M |
| NOTE: This operation is of ManagedGenericIRP IOC specified in [7]. The TestManagementIRP IOC of [6] inherits from it. | | |

## 5.2 Operation parameter mapping

The Test Management IRP: IS [6] defines semantics of parameters carried in operations across the Test Management IRP. The following tables indicate the mapping of these parameters, as per operation, to their equivalents defined in this SS.

**Table 2: Mapping from IS `initiateTests` parameters to SS equivalents**

| IS Operation parameter | SS Method parameter | Qualifier |
|---|---|---|
| testInvocationInitiator | TestManagementIRPConstDefs::TestInvocationInitiator test_invocation_initiator | M |
| maxTestingStateDuration | long maxtestingStateDuration | M |
| toBeInitiatedTests | TestManagementIRPConstDefs::ToBeInitiatedTestSeq to_be_initiated_test_seq | M |
| response | TestManagementIRPConstDefs::InitiateTestsResponse<br>Exceptions:<br>InitiateTests, ManagedGenericIRPSystem::ParameterNotSupported, ManagedGenericIRPSystem::InvalidParameter | M |

**Table 3: Mapping from IS `terminateTests` parameters to SS equivalents**

| IS Operation parameter | SS Method parameter | Qualifier |
|---|---|---|
| toBeTerminatedTests | TestManagementIRPConstDefs::ToBeTerminatedTestSeq to_be_terminated_test_seq | M |
| response | TestManagementIRPConstDefs::TerminateTestsResponse<br>Exceptions:<br>TerminateTests,<br>ManagedGenericIRPSystem::OperationNotSupported,<br>ManagedGenericIRPSystem::ParameterNotSupported,<br>ManagedGenericIRPSystem::InvalidParameter | M |

**Table 4: Mapping from IS `monitorTest` parameters to SS equivalents**

| IS Operation parameter | SS Method parameter | Qualifier |
|---|---|---|
| toBeMonitoredTO | TestManagementIRPConstDefs::ToBeMonitoredTO<br>to_be_monitored_TO | M |
| attributeList | TestManagementIRPConstDefs::TOAttributeList tO_attribute_list | M |
| error | ManagedGenericIRPConstDefs::Signal<br>Exceptions:<br>MonitorTest, ManagedGenericIRPSystem::OperationNotSupported, ManagedGenericIRPSystem::ParameterNotSupported, ManagedGenericIRPSystem::InvalidParameter | M |

## 5.3 Notification parameter mapping

The Test Management IRP: IS [6] 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 [1]) equivalents. The composition of OMG Structured Event, as defined in the OMG Notification Service [1], is:

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

The following table lists all OMG Structured Event attributes in the second column. The first column identifies the Test Management: IS [6] defined notification parameters.

### Table 8: Mapping for `notifyTestResult`

| 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 sub-clause 3.3.<br>It indicates the syntax and semantics of the Structured Event as defined by this specification. |
| notificationType | type_name | M | This is the NOTIFY_TM_TEST_RESULT of interface NotificationType of module TestManagementIRPConstDefs. |
| 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.<br>Name of this NV pair is the MANAGED_OBJECT_INSTANCE of interface AttributeNameValue of module NotificationIRPConstDefs.<br>Value of NV pair is a string. See corresponding table in Notification IRP: CORBA SS [5]. |
| notification Id | One NV pair of filterable_ body_fields | M | Name of NV pair is the NOTIFICATION_ID of interface AttributeNameValue of module NotificationIRPConstDefs.<br>Value of NV pair is a long. See corresponding table in Notification IRP: CORBA SS [5]. |
| eventTime | One NV pair of filterable_ body_fields | M | Name of NV pair is the EVENT_TIME of interface AttributeNameValue of module NotificationIRPConstDefs.<br>Value of NV pair is IRPTime. See corresponding table in Notification IRP: CORBA SS [5]. |
| systemDN | One NV pair of filterable_ body_fields | M | Name of NV pair is the SYSTEM_DN of interface AttributeNameValue of module NotificationIRPConstDefs.<br>Value of NV pair is a string. See corresponding table in Notification IRP: CORBA SS [5]. |
| testInvocationInitiator | One NV pair of filterable_ body_fields | | Name of NV pair is the TEST_INVOCATION_INITIATOR of interface AttributeNameValue of module TestManagementIRPConstDefs.<br>Value of NV pair is a string. |
| testInvocationId | One NV pair of filterable_ body_fields | | Name of NV pair is the TEST_INVOCATION_ID of interface AttributeNameValue of module TestManagementIRPConstDefs.<br>Value of NV pair is a string. |
| testActualStartTime | One NV pair of filterable_ body_fields | | Name of NV pair is the TEST_ACTUAL_START_TIME of interface AttributeNameValue of module TestManagementIRPConstDefs.<br>Value of NV pair is IRPTime. See corresponding table in Notification IRP: CORBA SS [5]. |
| testActualStopTime | One NV pair of filterable_ body_fields | | Name of NV pair is the TEST_ACTUAL_STOP_TIME interface AttributeNameValue of module TestManagementIRPConstDefs.<br>Value of NV pair is IRPTime. See corresponding table in Notification IRP: CORBA SS [5]. |
| testOutcome | One NV pair of filterable_ body_fields | | Name of NV pair is the TEST_OUTCOME of interface AttributeNameValue of module TestManagementIRPConstDefs.<br>Value of NV pair is a string. |
| mORT | One NV pair of filterable_ body_fields | | Name of NV pair is the MORT of interface AttributeNameValue of module TestManagementIRPConstDefs.<br>Value of NV pair is a string. |
| proposedRepairActions | One NV pair of | | Name of NV pair is the PROPOSED_REPAIR_ACTIONS of |

| IS Parameters | OMG CORBA Structured Event attribute | Qualifier | Comment |
|---|---|---|---|
| | filterable_body_fields | | interface AttributeNameValue of module TestManagementIRPConstDefs.<br>Value of NV pair is a string. |
| additionalInformation | Two NV pair of filterable_ body_fields | O | Name of one NV pair is the TEST_MANAGEMENT_VS_DATA of interface AttributeNameValue of module TestManagementIRPConstDefs.<br>Value of NV pair is a string. |
| fileReference | One NV pair of filterable_ body_fields | O | Name of NV pair is FILE_REFERENCE of interface AttributeNameValue of module TestManagementIRPConstDefs.<br>Value of NV pair is a string. |
| fileExpiryDate | One NV pair of filterable_ body_fields | O | Name of NV pair is the FILE_EXPIRY_DATE of interface AttributeNameValue of module TestManagementIRPConstDefs.<br>Value of NV pair is IRPTime. See corresponding table in Notification IRP: CORBA SS [5]. |
| There is no corresponding IS attribute. | remaining_ body | | |

# 6 TestManagementIRPNotification Interface

OMG CORBA Notification push operation is used to realise the notification of AlarmIRPNotifications. All the notifications in this interface are implemented using this push_structured_event method.

## 6.1 Method push (M)

```
module CosNotifyComm {

      …

      Interface SequencePushConsumer : NotifyPublish {
            void push_structured_events(

                  in CosNotification::EventBatch notifications)

         raises( CosEventComm::Disconnected);

            …

      }; // SequencePushConsumer

      …

}; // CosNotifyComm
```

NOTE 1: The push_structured_events method takes an input parameter of type EventBatch as defined in the OMG CosNotification module (OMG Notification Service [1]). This data type is the same as a sequence of Structured Events. Upon invocation, this parameter will contain a sequence of Structured Events being delivered to IRPManager by IRPAgent to which it is connected.

NOTE 2: The maximum number of events that will be transmitted within a single invocation of this operation is controlled by IRPAgent wide configuration parameter.

NOTE 3: The amount of time the supplier (IRPAgent) of a sequence of Structured Events will accumulate individual events into the sequence before invoking this operation is controlled by IRPAgent wide configuration parameter as well.

NOTE 4: IRPAgent may push `EventBatch` with only one Structured Event.

# Annex A (normative): IDL specification

## IDL specification (file name "TestManagementIRPConstDefs.idl")

```
#ifndef TestManagementIRPConstDefs_idl
#define TestManagementIRPConstDefs_idl

#include "CosNotification.idl"
#include "ManagedGenericIRPConstDefs.idl"

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

/* ## Module: TestManagementIRPConstDefs
This module contains commonly used definitions for Alarm IRP
================================================================
*/
module TestManagementIRPConstDefs
{
   /*
   This block defines the notification types of this Test Management
      IRP version.
   */
   interface NotificationType
   {
      const string NOTIFY_TM_TEST_RESULT = "x1";
   };


   /*
   This block defines the test state
   */
   interface TestState
   {
      const short NOT_INITIALIZED = 1;
      const short IDLE = 2;
      const short INITIALIZING = 3;
      const short TESTING = 4;
      const short TERMINATING = 5;
      const short DISABLED = 6;
   };



   /*
   This block defines the test outcome
   */
   interface TestOutcome
   {
      const short INCONCLUSIVE = 1;
      const short PASS = 2;
      const short FAIL = 3;
      const short TIME_OUT = 4;
      const short PREMATURE_TERMINATION = 5;
   };
```

```
/*
This block defines notification attributes of this IRP.
These attribute values should not clash with those used
    in Notification header (see IDL of Notification IRP).
*/
interface AttributeNameValue
{
   const string TEST_INVOCATION_INITIATOR = "f";
   const string TEST_INVOCATION_ID = "g";
   const string TEST_ACTUAL_START_TIME = "h";
   const string TEST_ACTUAL_STOP_TIME = "i";
   const string TEST_OUTCOME = "j";
   const string MORT = "k";
   const string PROPOSED_REPAIR_ACTIONS = "l";
   const string ADDITIONAL_INFORMATION = "m";
   const string FILE_REFERENCE = "n";
   const string FILE_EXPIRY_DATE = "o";
};


typedef string TestInvocationInitiator;
typedef string ToBeMonitoredTO;


/*
Define a seq of to-be-initiated-test
*/
struct ToBeInitiatedTest
{
    string toBeTestedMORT; //DN of the to be tested MORT
    string vSTOC;          //class of the to be tested MORT
    string vSTONVPair;     //seq of name value pairs
};
typedef sequence <ToBeInitiatedTest> ToBeInitiatedTestSeq;

/*
Define the structure returned by initiate_tests
*/
struct InitiateTestsResponseElement
{
   // If failureReason is NULL, the test is initiated successfully and
   //     testInvocationId contains the invocation id.
   // Else, the test initiation fails and failureReason contains
   //     the failure reason and testInvocationId contains garbage.
   string failureReason;
   string testInvocationId;
};
typedef sequence <InitiateTestsResponseElement> InitiateTestsResponse;


/*
Define a seq of to-be-terminated-test
*/
typedef string TestInvocationId;
typedef sequence <TestInvocationId> ToBeTerminatedTestSeq;

/*
Define the structure returned by terminate_tests
*/
struct TerminateTestsResponseElement
{
   // If failureReason is NULL, the test has terminated successfully and
   //     testInvocationId identifies the terminated invocation.
```

```
      // Else, the test termination fails and failureReason contains
      //     the failure reason and testInvocationId contains garbage.
      string failureReason;
      string testInvocationId;
   };
   typedef sequence <TerminateTestsResponseElement> TerminateTestsResponse;

   /*
   Define the structure of a TOAttributes of 3 elements.
   One element is encoded by a NV pair.  It is a list of NV pairs.
   The name and value of the NV pair is separated by a ’=’.
   NV pairs are separated by a ’,’.
   The type of all name and value of NV pairs is string.
   The name of NV pair carries the attribute name, case insensitive.
   The value of NV pair is attribute value.  If the attribute value
     is of type string, value of NV pair is an exact copy.  If
     the attribute value is of type short/long, the value of NV pair
     is the stringified digit of the short/long.
   Note that these attributes support only simple types such as
     string and short/long.
   */
   struct TOAttributes
   {
       short testState;  //use defn in interface TestState
       short testOutcome; //use defn in interface TestOutcome
       string attributesInNVPairs;
   };

   /*
   Define a list of TOAttributes
   */
   typedef sequence <TOAttributes> TOAttributeList;

};
#endif
```

# IDL specification (file name "TestManagementIRPSystem.idl")

```
#ifndef TestManagementIRPSystem_idl
#define TestManagementIRPSystem_idl

#include "TestManagementIRPConstDefs.idl"
#include "ManagedGenericIRPSystem.idl"

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

/* ## Module: TestManagementIRPSystem
This module contains the specification of all operations of TestManagement IRP
Agent.
================================================================
*/
module TestManagementIRPSystem
{
   /*
   System fails to complete the operation.  System can provide reason
   to qualify the exception.  The semantics carried in reason
   is outside the scope of this IRP.
   */
   exception GetTestManagementIRPVersions { string reason; };
   exception GetTestManagementIRPOperationsProfile { string reason; };
   exception GetTestManagementIRPNotificationProfile { string reason; };
   exception InitiateTests { string reason; };
   exception TerminateTests { string reason; };
   exception MonitorTest { string reason; };


   interface TestManagementIRP
   {
      /*
      Return the list of all supported TestManagement IRP versions.
      */
      ManagedGenericIRPConstDefs::VersionNumberSet
      get_Test_Management_IRP_versions (
      )
      raises (GetTestManagementIRPVersions);


      /*
      Return the list of all supported operations and their supported
      parameters for a specific TestManagement IRP version.
      */
      ManagedGenericIRPConstDefs::MethodList
      get_Test_Management_IRP_operations_profile (
         in ManagedGenericIRPConstDefs::VersionNumber
             test_management_irp_version
      )
      raises (GetTestManagementIRPOperationsProfile,
              ManagedGenericIRPSystem::OperationNotSupported,
              ManagedGenericIRPSystem::InvalidParameter);

      /*
      Return the list of all supported notifications and their supported
      parameters for a specific TestManagement IRP version.
      */
      ManagedGenericIRPConstDefs::MethodList
      get_Test_Management_IRP_notification_profile
      (
         in ManagedGenericIRPConstDefs::VersionNumber
test_management_irp_version
      )
      raises (GetTestManagementIRPNotificationProfile,
              ManagedGenericIRPSystem::OperationNotSupported,
              ManagedGenericIRPSystem::InvalidParameter);


      /*
      Request to initiate tests.
      */
      TestManagementIRPConstDefs::InitiateTestsResponse
      initiate_tests (
         in TestManagementIRPConstDefs::TestInvocationInitiator
            test_invocation_initiator,
         in unsigned long max_testing_state_duration,
         //in seconds;0 -> no limit
         in TestManagementIRPConstDefs::ToBeInitiatedTestSeq
            to_be_initiated_test_seq
      )
      raises (InitiateTests, ManagedGenericIRPSystem::ParameterNotSupported,
              ManagedGenericIRPSystem::InvalidParameter);


      /*
      Request to terminate tests.
      */
      TestManagementIRPConstDefs::TerminateTestsResponse
      terminate_tests (
         in TestManagementIRPConstDefs::ToBeTerminatedTestSeq
             to_be_terminated_test_seq
      )
      raises (TerminateTests,
              ManagedGenericIRPSystem::OperationNotSupported,
              ManagedGenericIRPSystem::ParameterNotSupported,
              ManagedGenericIRPSystem::InvalidParameter);


      /*
      Request test info.
      */
      ManagedGenericIRPConstDefs::Signal monitor_test (
         in TestManagementIRPConstDefs::ToBeMonitoredTO
             to_be_monitored_TO,
         out TestManagementIRPConstDefs::TOAttributeList tO_attribute_list
      )
      raises (MonitorTest, ManagedGenericIRPSystem::OperationNotSupported,
              ManagedGenericIRPSystem::ParameterNotSupported,
              ManagedGenericIRPSystem::InvalidParameter);




   };
};

#endif
```

# Annex B (informative): Change history

| Change history | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Date** | **TSG #** | **TSG Doc.** | **CR** | **Rev** | **Subject/Comment** | **Old** | **New** |
| Jun 2002 | S_16 | SP-020328 | -- | -- | Submitted to TSG SA #16 for Information | 1.0.0 | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |

# 3GPP TS 32.322 V1.0.0 (2002-05)

*Technical Specification*

## 3rd Generation Partnership Project;
## Technical Specification Group Services and System Aspects;
## Telecommunication Management; Test Management;
## Test Management IRP: Information Service;
## (Release 5)

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

Configuration management

***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

# Contents

# Foreword

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

The present document is part the 32.xxx-series covering the 3rd Generation Partnership Project: Technical Specification Group Services and System Aspects; Telecommunication Management; Test Management, as identified below:

32.321: "Test Management Integration Reference Point: Requirements";

**32.322: "Test Management Integration Reference Point: Information Service";**

32.323: "Test Management Integration Reference Point: CORBA Solution Set";

32.324: "Test Management Integration Reference Point: CMIP Solution Set";

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

A 3G telecommunication network is composed of a multitude of different network elements (NE). For a successful operation of the network the operator must be provided with mechanisms allowing him to manage the network. These management activities can be grouped into several areas: configuration management, fault management, performance management, accounting management and security management.

A management function assisting in different high level management areas such as fault management and performance management is test management. The purpose of testing is to get information about the functionality and performance of the 3G managed network subject to the test.

The present document is part of a set of technical specifications defining the telecommunication management (TM) of 3G systems. The TM principles are described in 3GPP TS 32.101 [5]. The TM architecture is described in 3GPP TS 32.102 [6]. The other specifications define the interface (ITf-N) between the managing system (manager), which is in general the network manager (NM) and the managed system (agent), which is either an element manager (EM) or the managed NE itself. The Itf-N is composed of a number of integration reference points (IRPs) defining the information in the agent that is visible for the manager, the operations that the manager may perform on this information and the notifications that are sent from the agent to the manager. One of these IRPs is the Test Management IRP.

Each IRP is specified by four TS, the requirements part, the information service (IS) part, the CORBA solution set (SS) and the CMIP solution set.

# 1 Scope

The present document defines the IS part of the Test Management IRP, which describes the semantics of the information and the interactions visible across Itf-N in a protocol independent way. The information is specified by means of information object classes and the interactions by means of operations and notifications. This document does not specify the syntax (encoding) of the information.

# 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 TS 32.302: “Notification IRP: Information Service”

[2] 3GPP TS 32.312: “Generic IRP Management: Information Service”

[3] 3GPP TS 32.622: “Generic Network Resources IRP: Network Resource Model”

[4] ITU-T Rec. X.733: “Information Technology – Open Systems Interconnection – Systems Management: Alarm Reporting Function”

[5] ITU-T Rec. X.745: “Information Technology – Open Systems Interconnection – Systems Management: Test Management Function”

[6] 3GPP TS 32.101: “3G Telecom Management: Principles and high-level Requirements”

[7] 3GPP TS 32.102: “3G Telecom Management Architecture”

[8] 3GPP TS 32.321: “Test Management IRP: Requirements”

[9] 3GPP TS.32.672: “State Management IRP: Information Service”

[10] ITU-T Rec. X.737: “Information Technology – Open Systems Interconnection – Systems Management: Confidence and Diagnostic Test Categories

# 3 Definitions and Abbreviations

## 3.1 Definitions

For the purposes of the present document, the terms and definitions given in 3GPP TS 32.101 [6], 3GPP TS 32.102 [7] and 3GPP TS 32.321 [8] apply.

## 3.2 Abbreviations

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

| | |
|---|---|
| IOC | Information Object Class |
| IRP | Integration Reference Point |

| | |
|---|---|
| IS | Information Service |
| ME | Element Manager |
| MORT | Managed Object Referring to Test |
| NE | Network Element |
| TM | Telecommunication Management |

# 4 System Overview

Figures 1 and 2 show the system context of this document in terms of implementations called IRPAgent and IRPManager.

The term IRPManager refers to a process that interacts with IRPAgent for the purpose of test management via this IRP. An example of an IRPManager can be a Network Management System. IRPAgent implements and supports the Test Management IRP.

IRPAgent can be one Network Element (NE) (Fig. 2) or it can be one Element Manager (EM) with one or more NEs (Fig. 1). In the latter case, the interfaces (represented by a thick dotted line) between the EM and the NEs are not subject of this IRP. Whether EM and NE share the same hardware system is not relevant to this document either. By observing the interaction across the Test Management IRP, one cannot deduce if EM and NE are integrated in a single system or if they run in separate systems.

As indicated in Figure 1 and 2, the subject document need to be complemented with the Notification IRP [1] (to allow IRPManager to subscribe to notifications issued by IRPAgent and (optionally) product-specific resource models describing the MOs maintained by the IRPAgent).

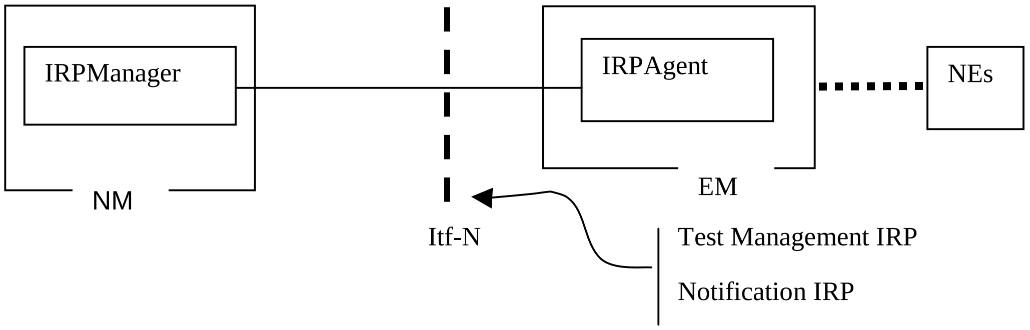


Figure 1: System Context A

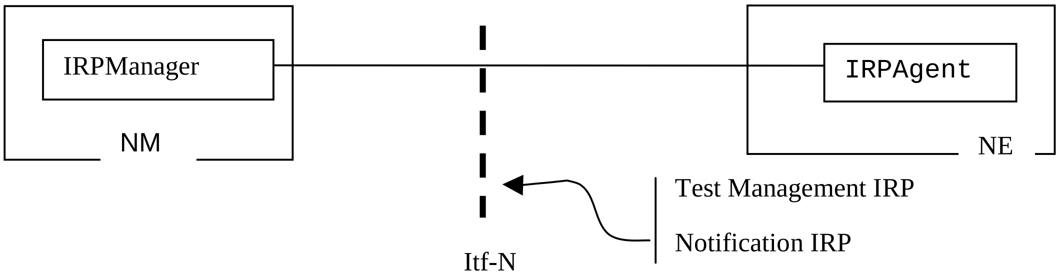


Figure 2: System Context B

# 5 Information Object Classes

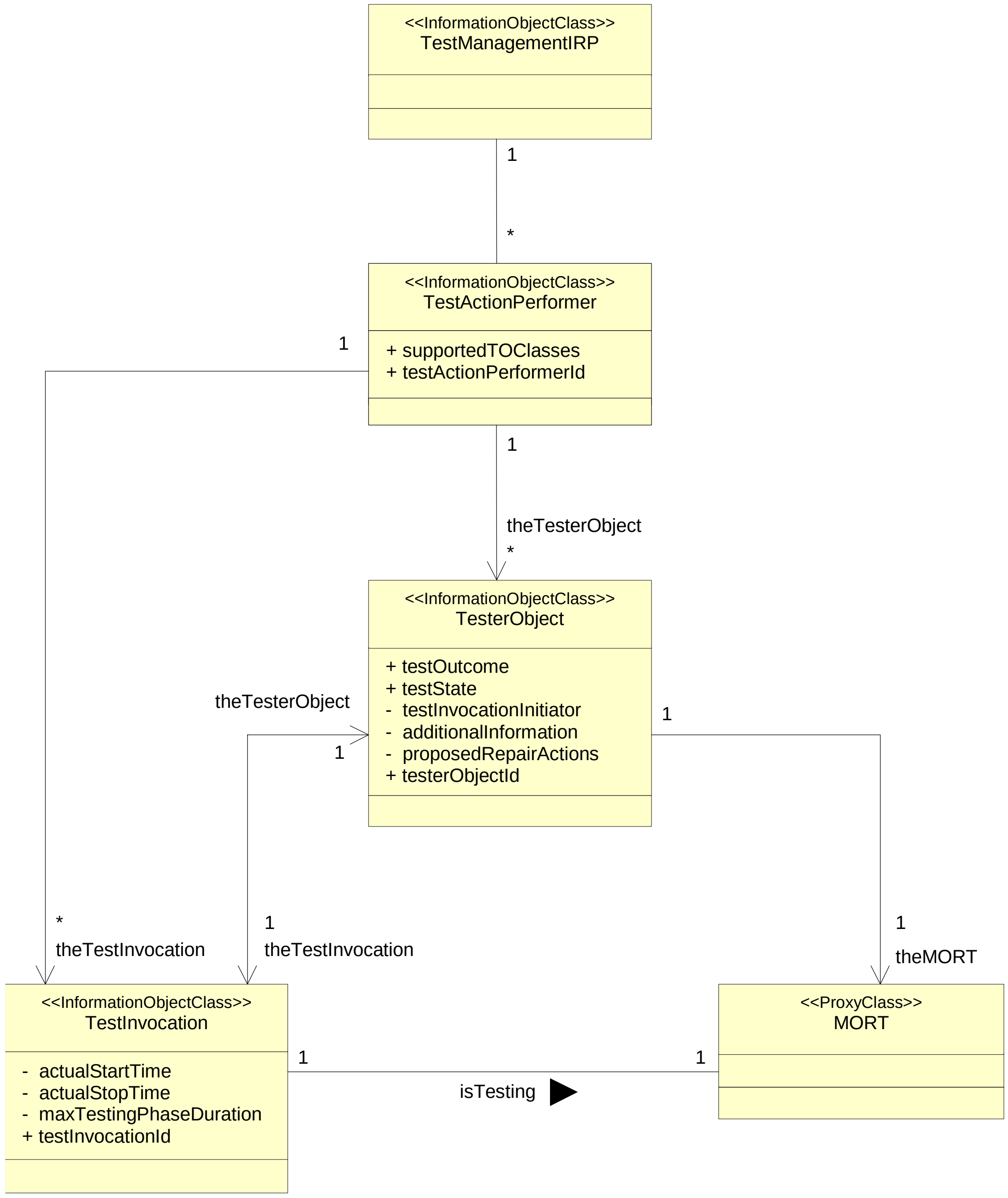
## 5.1 Information Entities imported and local Labels

| **Label reference** | **Local label** |
|---|---|
| 32.622 [3], information object class, Top | Top |
| 32.622 [3], information object class, IRPAgent | IRPAgent |
| 32.312 [2], information object class, managedGenericIRP | managedGenericIRP |

## 5.2 Class Diagram

### 5.2.1 Attributes and Relationships

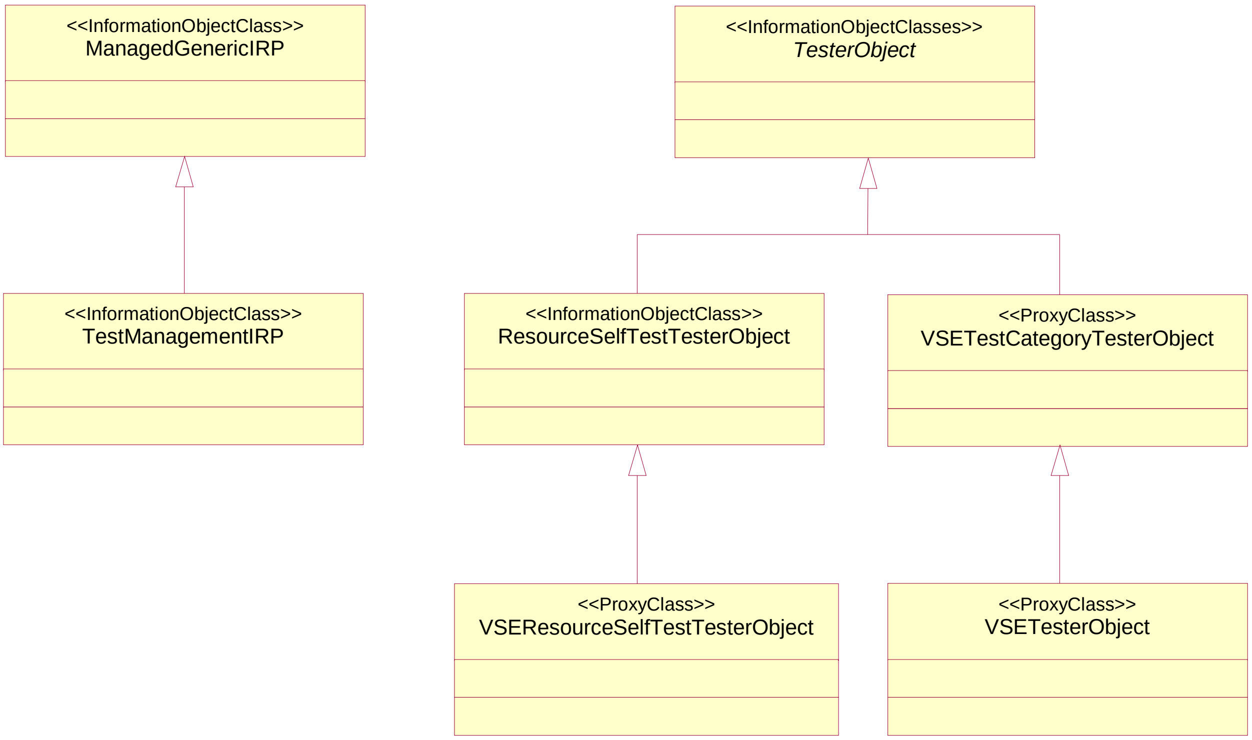
The following figure shows, for the Test Management IRP, the class definitions and the relationships between the classes.



From the cardinalities can be seen that each instance of *TestManagementIRP* may have several instances of *TestActionPerformer*. Each instance of *TestActionPerformer* can have multiple instances of associated *TesterObjects*. Each instance of *TesterObject* in turn has one instance of *TestInvocation* and one instance of *MORT*.

### 5.2.2 Inheritance

The following figure depicts the inheritance relationships between the information object classes. As can be seen the IOC *TestManagementIRP* inherits from *ManagedGenericIRP*, the Proxy Class *VSEResourceSelfTestTesterObject* inherits from the IOC *ResourceSelfTestTesterObject* which in turn inherits from *TesterObject*. The Proxy Class *VSETesterObject* inherits from the Proxy Class *VSETestCategoryTesterObject* which inherits from the IOC TesterObject. By default IOCs inherit from the IOC *Top*.



<<InformationObjectClass>>
ManagedGenericIRP
<<InformationObjectClasses>>
TesterObject
<<InformationObjectClass>>
TestManagementIRP
<<InformationObjectClass>>
ResourceSelfTestTesterObject
<<ProxyClass>>
VSETestCategoryTesterObject
<<ProxyClass>>
VSEResourceSelfTestTesterObject
<<ProxyClass>>
VSETesterObject

# 5.3 Information Object Classes Definition

## 5.3.1 Information Object Class *TestManagementIRP*

### 5.3.1.1 Definition

The IOC *TestManagementIRP* together with the IOC *TestActionPerformer* represent the test management capabilities defined by this specification. To conduct a test of network resources, this object may require capabilities of other objects such as *TesterObject*. The IOC *TestManagementIRP* inherits from the IOC *ManagedGenericIRP* specified in 3GPP TS 32.312 [2].

### 5.3.1.2 Attributes

The IOC *TestManagementIRP* has no own attributes, only those inherited from the IOC *ManagedGenericIRP*.

## 5.3.2 Information Object Class *TestActionPerformer*

### 5.3.2.1 Definition

The IOC *TestActionPerformer* provides the ability to receive and react upon test requests. This class must also be able to instantiate and delete tester objects or, in case the tester objects are permanently instantiated, to allocate and reserve them for their usage. This specification does not require this IOC to be instantiated. It may be abstract and used for inheritance purposes only. In this way the ability to receive and react upon test requests may be included in any other IOC.

### 5.3.2.2 Attributes

| Attribute name | Support Qualifier |
|---|---|
| `supportedTOClasses` | M |
| `testActionPerformerId` | M (NOTE 1) |

NOTE 1: This attribute is only mandatory in case the IOC *TestActionPerformer* is instantiated. In case this IOC is an abstract class and used for inheritance purposes only the attribute shall be omitted.

### 5.3.3 Information Object Class *TesterObject*

#### 5.3.3.1 Definition

The IOC *TesterObject* monitors and controls the testing of a *MORT* instance and reports the outcome of the test execution. Tester Objects (TOs) are instantiated by the IOC *TestActionPerformer* in response to a valid test initiation request (*initiateTests*). They are deleted after termination of the test. It is also possible that TOs are permanently instantiated. In this case they are allocated to a certain *TestActionPerformer* during the test execution. After termination of the test they are released.

The IOC *TesterObject* defines a generic TO. It shall be used as an abstract class from which more specific tester objects shall be derived by specialisation for each test category. Test categories and the associated test category specific TOs are defined in ITU-T X.737 [10]. These test category specific TOs can be specialised further by defining vendor-specific-extended (VSE) TOs. The generic TO defines attributes pertaining to a test and required for all test categories.

Each test invocation shall have only one associated TO.

Only test category specific TOs or VSE TOs shall be instantiated.

For simplicity this specification will often use only the term TO. In this case either the test category specific TO or the VSE TO is referred to depending on which is actually instantiated.

#### 5.3.3.2 Attributes

| Attribute name | Support Qualifier |
|---|---|
| `testInvocationInitiator` | C |
| `testOutcome` | M |
| `additionalInformation` | O |
| `proposedRepairActions` | O |
| `testerObjectId` | M |

### 5.3.4 Information Object Class *ResourceSelfTestTesterObject*

#### 5.3.4.1 Definition

The IOC *ResourceSelfTesterObject* is a specialised TO for the resource self test. It inherits from the IOC *TesterObject* and the IOC *TestInvocation.* It specifies the triggering events for the emission of the test result notifications.

#### 5.3.4.2 Attributes

This IOC has no own attributes, only those inherited from the generic IOC *TesterObject*.

### 5.3.5 Proxy Class *VSETestCategoryTesterObject*

#### 5.3.5.1 Definition

Certain tests may not fit in any of the test categories defined in X.737 [10]. In this case vendors may define new (VSE) test categories and the associated test category specific TOs. The Proxy Class *VSETestCategoryTesterObject* represents the set of these VSE test category tester objects

The IOCs represented by the Proxy Class *VSETestCategoryTesterObject* shall inherit from the IOC *TesterObject*.

NOTE:

A vendor may also claim 3GPP compliance to a certain release in case that a specific test fits into one of the ITU-T test categories without that the corresponding ITU-T test category specific TO is supported in this release supposed that this test category specific TO will be added in a later release than the current one. The vendor shall update this specification in due time.

#### 5.3.5.2 Attributes

The attributes of this IOC are vendor specific.

### 5.3.6 Proxy Class *VSEResourceSelfTestTesterObject*

#### 5.3.6.1 Definition

In case the IOC *ResourceSelfTestTesterObject* does not fulfil the specific requirements of a certain resource self test, vendors may define proprietary IOCs by further specialisation. The Proxy Class *VSEResourceSelfTestTesterObject* represents these IOCs.

The IOCs represented by the Proxy Class *VSEResourceSelfTestTesterObject* shall inherit from the IOC *ResourceSelfTestTesterObject*.

#### 5.3.6.2 Attributes

Apart from the attributes inherited the attributes of the IOCs represented by this Proxy Class are vendor specific.

### 5.3.7 Proxy Class *VSETesterObject*

#### 5.3.7.1 Definition

In case an IOC represented by the Proxy Class *VSETestCategoryTesterObject* does not fulfil the specific requirements of a certain test, vendors may define proprietary IOCs by further specialisation. The Proxy Class *VSETesterObject* represents these IOCs.

The IOCs represented by the Proxy Class *VSETesterObject* shall inherit from the associated IOC represented by the Proxy Class *VSETestCategoryTesterObject*.

#### 5.3.7.2 Attributes

Apart from the attributes inherited the attributes of the IOCs represented by this Proxy Class are vendor specific.

### 5.3.8 Proxy Class *MORT*

#### 5.3.8.1 Definition

The ProxyClass *MORT* represents a network resource that is under test. Its class definition shall be one defined in the various 3GPP Network Resource Model specifications or defined by a VSE network resource class.

#### 5.3.8.2 Attributes

This IOC has no attributes.

### 5.3.9 Information Object Class *TestInvocation*

#### 5.3.9.1 Definition

The IOC *TestInvocation* is the abstract representation of a test invocation. A test invocation shall aim to test one or more capabilities of a MORT. The IRPManager can request for the establishment of a test invocation using the operation *initiateTests*.

A MORT can be complex in that there are multiple capabilities that can be subject to test. Therefore, it is possible to have multiple test activities active, all aimed at the same MORT but on its different capabilities. Whether multiple test activities can be testing the same MORT capabilities at the same time is an implementation decision, probably based on the nature and behaviour of the TO, and therefore, is outside the scope of this specification.

#### 5.3.9.2 Attributes

| Attribute name | Support Qualifier |
|---|---|
| actualStartTime | O |
| actualStopTime | O |
| maxTestingPhaseDuration | O |
| testInvocationId | M |

## 5.4 Information Relationships Definition

### 5.4.1 Relationship between *TestManagementIRP* and *TestActionPerformer*

#### 5.4.1.1 Definition

This relationship defines a binary association between the IOC *TestManagementIRP* and the IOC *TestActionPerformer*.

#### 5.4.1.2 Roles

This relationship has no roles.

### 5.4.2 Relationship between *TestActionPerformer* and *TesterObject*

#### 5.4.2.1 Definition

This relationship defines a binary association between the IOC *TestActionPerformer* and the IOC *TesterObject*. The association is navigable from the *TestActionPerformer* to the *TesterObject*.

#### 5.4.2.2 Role

| Name | Definition |
|---|---|
| theTesterObject | This rolename provides a name allowing to navigate from an instance of *TestActionPerformer* to the associated instances of *TesterObject*. If *tap* is an instance of *TestActionPerformert*, the expression *tap.theTesterObject* yields the set of object instances of *TesterObject*. |

### 5.4.3 Relationship between *TestActionPerformer* and *TestInvocation*

#### 5.4.3.1 Definition

This relationship defines a binary association between the IOC *TestActionPerformer* and the IOC *TestInvocation*. The association is navigable from the *TesterObject* to the *TestInvocation*.

#### 5.4.3.2 Role

| Name | Definition |
|---|---|
| `theTestInvocation` | This rolename provides a name allowing to navigate from an instance of *TestActionPerformer* to the associated instances of *TestInvocation*. If *tap* is an instance of *TestActionPerformert*, the expression *tap.theTestInvocation* yields the set of object instances of *TestInvocationt*. |

### 5.4.4 Relationship between *TesterObject* and *TestInvocation*

#### 5.4.4.1 Definition

This relationship defines a binary association between the IOC *TesterObject* and the IOC *TestInvocation*. The association is navigable in both directions..

#### 5.4.4.2 Role

| Name | Definition |
|---|---|
| `theTesterObject` | This rolename provides a name allowing to navigate from an instance of *TestInvocation* to the associated instance of *TesterObject*. If *ti* is an instance of *TestInvocation*, the expression *ti.theTesterObject* yields an object instance of *TesterObject*. |
| `theTestInvocation` | This rolename provides a name allowing to navigate from an instance of *TesterObject* to the associated instance of *TestInvocation*. If *to* is an instance of *TesterObject*, the expression *to.theTestInvocation* yields an object instance of *TestInvocation*. |

### 5.4.5 Relationship between *TesterObject* and *MORT*

#### 5.4.5.1 Definition

This relationship defines a binary association between the IOC *TesterObject* and the Proxy Class *MORT*. The association is navigable from the *TesterObject* to the *MORT*.

#### 5.4.5.2 Role

| Name | Definition |
|---|---|
| `theMORT` | This rolename provides a name allowing to navigate from an instance of *TesterObject* to the associated instance of *MORT*. If *to* is an instance of *TesterObject*, the expression *to.theMORT* yields an object instance of *MORT*. |

### 5.4.6 Relationship between *TestInvocation* and *MORT*

#### 5.4.6.1 Definition

This relationship defines an association between the IOC *TestInvocation* and the IOC *MORT*. This association specifies that the latter is testing the former.

#### 5.4.6.2 Role

Instead of roles this relationship has a role name.

# 5.5 Information Attributes Definition

## 5.5.1 Definition and legal Values

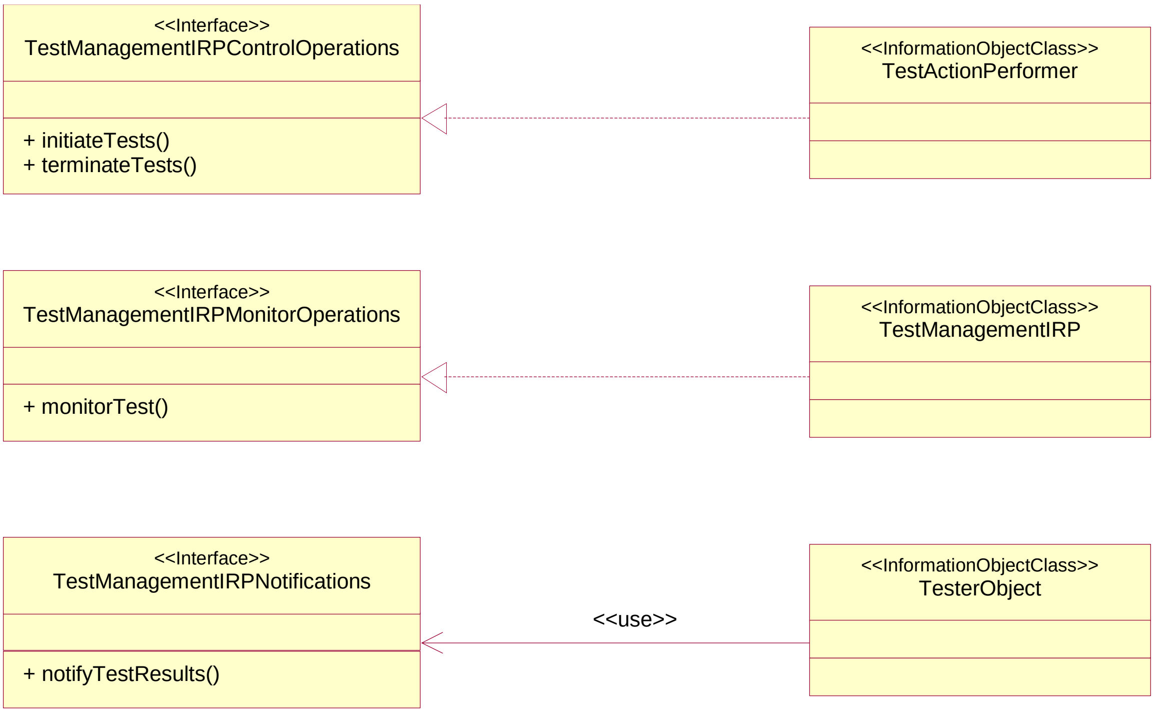
| Attribute Name | Definition | Legal Values |
|---|---|---|
| testInvocationId | his attribute uniquely identifies an instance of a *estInvocation* within the TestManagementIRP. he test invocation identifier is assigned by the estActionPerformer. When a testInvocationId an be reused is outside the scope of this pecification. | |
| testState | This attribute reflects the actual test state (ITU-T X.745 [5]). | ENUM {notInitialized, idle, initializing, testing, terminating, disabled} |
| testOutcome | This attribute provides information about the test result, as perceived by the associated TO, in a standardised manner.<br>The information in this parameter is only valid after termination of the test activity.<br>This information shall be present in the last test result notification emitted by a TO prior to its deletion. | ENUM {pass, fail, inconclusive, timed-out, premature-termination}<br>- Pass indicates that the test exercise of the test invocation has executed correctly and has found no problem.<br>- Fail indicates that the test exercise of the test invocation has executed correctly and has found one or more problems.<br>- Inconclusive indicates that the TO has not determined if the execution is Pass or Fail.<br>- Timed-out indicates that the TO has terminated its execution because of the expiry of the timer (i.e., the current time – TestSession.sessionStartTime >= TesterObject.timeOut).<br>- Premature termination indicates that the TO has (a) never started execution or (b) terminated its execution prematurely, either by *TestManagementIRP* and its associated objects internal problems or in response to a *terminateTests* operation. |
| supportedTOClasses | This attribute identifies the TO classes that are supported by a certain managed object instance whose class has inherited from *TestActionPerformer* or whose class is the *TestActionPerformer*. | SET OF TO class name |
| testActionPerformerId | This attribute unambiguously identifies an instance of a *TestActionPerformer*. | |
| testerObjectId | This attribute unambiguously identifies an instance of a *TesterObject*. | |
| testInvocationInitiator | It identifies the IRPManager. | How multiple IRPManagers choose their identifier so that they are distinguishable is outside the scope of this specification. |
| additionalInformation | This attribute holds a set of additional information pertaining to the test. | The semantics of this parameter are outside the scope of this specification |

| Attribute Name | Definition | Legal Values |
|---|---|---|
| proposedRepairActions | This attribute suggests one or more repair actions if the reason for a failure is known. | The semantics of this parameter are outside the scope of this specification |
| actualStartTime | This attribute specifies the time at which the TO will enter or has entered the test state *testing*. Before the TO enters the testing state this is an estimated time. After entering the testing state this is the actual time. Note that this is not the time of the invocation of the operation *initiateTests*. | All values indicating a valid time. |
| actualStopTime | This attribute specifies the time at which the TO will leave or has left the test state *testing*. Before the TO leaves the testing state this is an estimated time. After leaving the testing state this is the actual time. Note that this is not the time of the invocation of the operation *terminateTests*. | All values indicating a valid time later than the *actualStartTime*. |
| maxTestingPhaseDuration | This attribute specifies the maximum amount of time that a TO may spend in the testing state. | All values indicating a valid amount of time. |

# 6 Interface Definition

## 6.1 Class diagram representing interfaces

The following diagram depicts the interfaces of the Test Management IRP with their corresponding operations and notifications.


<<Interface>>
TestManagementIRPControlOperations
+ initiateTests()
+ terminateTests()
<<InformationObjectClass>>
TestActionPerformer
<<Interface>>
TestManagementIRPMonitorOperations
+ monitorTest()
<<InformationObjectClass>>
TestManagementIRP
<<Interface>>
TestManagementIRPNotifications
+ notifyTestResults()
<<use>>
<<InformationObjectClass>>
TesterObject

## 6.2 Generic rules

Rule 1: each operation with at least one input parameter supports a pre-condition valid_input_parameter which indicates that all input parameters shall be valid with regards to their information type. Additionally, each such operation supports an exception operation_failed_invalid_input_parameter which is raised when pre-condition valid_input_parameter is false. The exception has the same entry and exit state.

Rule 2: Each operation with at least one optional input parameter supports a set of pre-conditions supported_optional_input_parameter_xxx where "xxx" is the name of the optional input parameter and the pre-condition indicates that the operation supports the named optional input parameter. Additionally, each such operation supports an exception operation_failed_unsupported_optional_input_parameter_xxx which is raised when (a) the pre-condition supported_optional_input_parameter_xxx is false and (b) the named optional input parameter is carrying information. The exception has the same entry and exit state.

Rule 3: each operation shall support a generic exception operation_failed_internal_problem that is raised when an internal problem occurs and that the operation cannot be completed. The exception has the same entry and exit state.

## 6.3 Interface *testManagementIRPControlOperations*

The interface *TestManagementIRPControlOperations* contains the operations *initiateTests* and *terminateTests*. It must be implemented by every object with the ability to receive and react upon test requests, for example by every instance of *TestActionPerformer*.

### 6.3.1 Operation *initiateTests* (M)

#### 6.3.1.1 Definition

The IRPManager uses this operation to request the IRPAgent to initiate controlled tests. A single test request may initiate multiple (one or more) tests.

For each test to be initiated the managed object representing the network resource to be tested and the tester object class must be specified.

The initiated tests are independent and not related to each other. This implies that independent test result notifications are sent for each of the tests initiated by s single *initiateTests* operation.

#### 6.3.1.2 Input parameters

| Parameter Name | Qualifier | Information Type | Comment |
|---|---|---|---|
| testInvocationInitiator | C | TesterObject. testInvocationInitiator | This parameter identifies the IRPManager.. |
| maxTestingStateDuration | O | TestInvocation.maxTestingStateDuration | This parameter specifies the timeout period of the tests to be initiated. A certain value shall indicate forever. |
| toBeInitiatedTests | M | SET OF SET {<br>toBeTestedMORT<br>testerObjectClass<br>testerObjectInitialAttributeList<br>} | This sequence specifies the tests to be initiated.<br>For each test the parameter *toBeTestedMORT* specifies the instance of the *MORT* to be tested. The parameter *testerObjectClass* specifies the class of the associated tester object. The parameter testerObjectInitialAttributeList carries some or all the values of the attributes of the TO instance responsible for the test. The syntax and semantics of this attribute value is dependent on the specific TO class |

| Parameter Name | Qualifier | Information Type | Comment |
|---|---|---|---|
| | | | definition and is outside the scope of 3GPP. |

### 6.3.1.3 Output parameters

| Parameter Name | Qualifier | Matching Information | Comment |
|---|---|---|---|
| response | M | Resource self test:<br>SEQUENCE OF CHOICE {<br>testInitiated<br>testNotInitiated<br>}<br>testInitiated = TestInvocation.testInvocationId<br>testNotInitiated = failureReason | The number and the order, related to the tests to be initiated, of elements in this sequence and in the set of the input parameter *toBeInitiatedTests* shall be identical.<br>For a successfully instantiated test the parameter *testInitiated* returns the test invocation identifier of the test. For a failed test instantiation the parameter *testNotInvoked* returns the reason why the instantiation of the test failed.<br>Failure reasons are<br>• TO class is not existing<br>• MORT is not existing<br>• MORT is not available<br>• others |

### 6.3.1.4 Pre-condition

The precondition must hold true before the operation is invoked. The pre-condition depends on the test category.

**Resource Self Test:**

For at least one of the specified tests to be instantiated the following must hold true

theIndicatedMORTIsExisting **AND** theIndicatedMORTIsAvailable **AND** theIndicatedTOClassIsExisting

| Assertion Name | Definition |
|---|---|
| theIndicatedMORTIsExisting | The MORT indicated by the subject operation for this test is existing |
| theIndicatedMORTIsAvailable | The MORT indicated by the subject operation for this test is available. |
| theIndicatedTOClassIsExisting | The TO class indicated by the subject operation for this test is existing. |

### 6.3.1.5 Post-condition

The post-condition must hold true after the completion of the operation.

allIndicatedTOsInstantiated **OR** notAllTestsInitiated **OR** noTestInitiated

| Assertion Name | Definition |
|---|---|
| allTestsInitiated | All tests indicated by the subject operation were initiated successfully. |

| | |
|---|---|
| notAllTestsInitiated | Not all but at least one test indicated by the subject operation was initiated successfully. |
| noTestInitiated | No test indicated by the subject operation was initiated successfully. |

### 6.3.1.6 Exceptions

| Exception Name | Definition |
|---|---|
| operationFailedEntirely | **Condition**: noTestInitiated = TRUE<br>**Returned information**: The response parameter is returned.<br>**Exit state**: Entry state |
| opeartionFailedPartly | **Condition**: notAllTestsInitiated = TRUE<br>**Returned information**: The response parameter is returned.<br>**Exit state**: Entry state |

## 6.3.2 Operation *terminateTests* (M)

### 6.3.2.1 Definition

The IRPManager uses this operation to request the IRPAgent to terminate tests during their life time. A single *terminateTests* operation may terminate multiple (one or more) tests.

The tests to be terminated are identified by their test invocation identifiers. The IRPManager terminating a test may be different from the IRPManager that initiated the test. The *terminateTests* operation must be invoked on the object which received the corresponding *initiateTests* operation.

### 6.3.2.2 Input parameters

| Parameter Name | Qualifier | Information Type | Comment |
|---|---|---|---|
| toBeTerminatedTests | M | SET OF TestInvocation.testInvocationId | This parameter specifies the tests that shall be terminated. |

### 6.3.2.3 Output parameters

| Parameter Name | Qualifier | Matching Information | Comment |
|---|---|---|---|
| response | M | SEQUENCE OF CHOICE {<br>testTerminated<br>testNotTerminated<br>}<br>testTerminated = TestInvocation.testInvocationId<br>testNotTerminated = SEQUENCE {<br>TestInvocation.testInvocationId, | The number and the order, related to the test invocation identifier, of elements in this sequence and in the set of the input parameter *toBeTerminatedTests* shall be identical.<br>It specifies the test invocation ids of the tests, that were successfully terminated, and the ids of the tests, that failed to be terminated successfully together with the reason for the |

| Parameter Name | Qualifier | Matching Information | Comment |
|---|---|---|---|
| | | failureReason<br>} | failure.<br>Failure reasons are<br>• test invocation id is not existing<br>• others |

### 6.3.2.4 Pre-condition

The precondition must hold true before the operation is invoked.

allIndicatedTestInvocationIdsAreExisting **OR** notAllIndicatedTestInvocationIdsAreExisting

| Assertion Name | Definition |
|---|---|
| allIndicatedTestInvocationIdsAreExisting | All test invocation identifiers specified by the subject operation are existing. |
| notAllIndicatedTestInvocationIdsAreExisting | Not all but at least one test invocation identifier specified by the subject operation is existing. |

### 6.3.2.5 Post-condition

The post-condition must hold true after the completion of the operation.

allIndicatedTestsTerminated **OR** notAllIndicatedTestsTerminated **OR** noIndicatedTestTerminated

| Assertion Name | Definition |
|---|---|
| allIndicatedTestsTerminated | All tests indicated in the subject operation were terminated successfully. |
| notAllIndicatedTestsTerminated | Not all but at least one test indicated in the subject operation aaws terminated successfully |
| noIndicatedTestTerminated | No test indicated in the subject operation aaws terminated successfully |

### 6.3.2.6 Exceptions

| Exception Name | Definition |
|---|---|
| operationFailedEntirely | **Condition**: noIndicatedTestTerminated = TRUE<br>**Returned information**: The response parameter is returned.<br>**Exit state**: Entry state |
| operationFailedPartly | **Condition**: notAllIndicatedTestInvocationIdsAreExisting = TRUE **OR** notAllIndicatedTestsTerminated = TRUE<br>**Returned information**: The response parameter is returned.<br>**Exit state**: Entry state |

## 6.4 Interface *TestManagementIRPMonitorOperations*

The interface *TestManagementIRPMonitorOperations* contains the operation *monitorTest*. It has a realisation relationship with the IOC *TestManagementIRP*.

### 6.4.1 Operation *monitorTest* (M)

#### 6.4.1.1 Definition

IRPManager shall be able to retrieve information about the test as observed by the TO during the test execution by reading the relevant attributes of the TO associated to the test. Also after the test execution the manager shall be able to read these attributes as long as the TO exists. Attributes conveying information about the test execution are *testState* and *testOutcome*. Depending on the specific test category specific TO or the VSE TO other attributes may also contain information about the test execution. In this case the subject operation may also allow to read the values of these attributes.

#### 6.4.1.2 Input parameters

| Parameter Name | Qualifier | Information Type | Comment |
|---|---|---|---|
| toBeMonitoredTO | M | tOInstance | This parameter specifies the instance of the tester object, whose attribute values of *testState*, *testOutcome* and other attributes shall be retrieved. |

#### 6.4.1.3 Output parameters

| Parameter Name | Qualifier | Matching Information | Comment |
|---|---|---|---|
| attributeList | M | SET {<br>TesterObject.testState (M)<br>TesterObject.testOutcome (M)<br>other attributes (O)<br>} | This parameter shall be returned if all attributes were read successfully and may be returned, if at least one attribute was read successfully.<br>The values to be returned are those prevalent at the time of the reception of the subject operation. |
| error | M | failureReason | This parameter shall be returned if the specified tester object instance is not existing or, in case the tester object instance is existing, at least one attribute could not be read, i. e. if<br>operationFailedEntirely = TRUE<br>OR<br>operationFailedPartly = TRUE<br>The parameter returns the failure reason. |

#### 6.4.1.4 Pre-condition

The precondition must hold true before the operation is invoked.

indicatedTOInstanceIsExisting

| Assertion Name | Definition |
|---|---|
| toBeMonitoredTOIsExisting | The TO instance indicated by the subject operation is existing. |

### 6.4.1.5 Post-condition

The post-condition must hold true after the completion of the operation.

allAttributeValuesRead **OR** notAllAttributeValuesRead **OR** noAttributeValueRead

| Assertion Name | Definition |
|---|---|
| allAttributeValuesRead | All attributes of the TO indicated by the subject operation were read successfully. |
| notAllAttributeValuesRead | Not all but at least one attribute of the TO indicated by the subject operation were read successfully. |
| noAttributeValueRead | No attribute of the TO indicated by the subject operation was read successfully. |

### 6.4.1.5 Exception

| Exception Name | Definition |
|---|---|
| operationFailedEntirely | **Condition**:<br>toBeMonitoredTOIsExisting = FALSE **OR** noAttributeValueRead = TRUE<br>**Returned information**: The error parameter returns the object identifier of the TO that does not exist or the reasons, why the attributes could not be read.<br>**Exit state**: Entry state |
| operationFailedPartly | **Condition**:<br>toBeMonitoredTOIsExisting = TRUE **AND** notAllAttributeValuesRead = TRUE<br>**Returned information**: The error parameter returns the reason why an attribute could not be read. The attribute that could be read my be returned in the parameter *error* or the parameter *attributeList*.<br>**Exit state**: Entry state |

# 6.4 Interface *TestManagementIRPNotifications*

## 6.4.1 Notification *notifyTestResults* (M)

### 6.4.1.1 Definition

Test results are made available to the IRPManager by one or more notifications *notifyTestResults* emitted by the TO that is related to the test invocation.

Depending on the nature of the test and the specification of the TO behaviour, the TO may need to convey to the IRPManager a test result data set. There are two ways to convey this kind of information. One way is to use the parameter additionalInformation of the notification. In this case, the fileReference and fileExpiryDate shall contain no information or be absent. The other way is to use a file to capture the test result data set. In this case, the additionalInformation parameter may contain no information or be absent and the fileReference and fileExpiryDate shall be present. The file that captures the test result data set shall contain VSE attributes and other 3GPP attributes such as testerObjectClass, testOutcome, etc.

The use of the additionalInformation parameter or a file to capture the test result data set is specified by the class specification of the VSE TO.

In case the TO uses additionalInformation (and not a file) to capture the test result data set, that TO may emit this notification to transfer intermediate (non-final) test results. In this kind of notifications, the *testOutcome* parameter shall be absent. The TO should emit at least one more notification regarding the subject test invocation in the future. The last notification pertaining to a particular test invocation shall be indicated by including the *testOutcome* parameter in the notification.

In the case the TO uses a file to capture the test result data set, that TO shall not issue any notifications to transfer intermediate test results. The TO may capture the non-final test results in the file used to capture the final test result data set.

The events triggering the emission of test result notifications depend on the specific test. They shall be specified by the TO that is actually instantiated, i. e. either by the test category specific TO or the VSE TO. Some generic triggering events are included in this specification. It is expected that vendors specify more triggering events.

| Parameter Name | Qualifier | Matching Information | Comment |
|---|---|---|---|
| objectClass | M, F | TesterObject.testerObjectClass | This parameter is specified by NotificationIRPNotification defined in 3GPP TS 32.302 [1]. It specifies the class of the TO emitting the subject notification. |
| objectInstance | M, F | TesterObject.testerObjectId | This parameter is specified by NotificationIRPNotification defined in 3GPP TS 32.302 [1]. It specifies the instance of the TO emitting the subject notification. |
| notificationId | O | | This parameter is specified by NotificationIRPNotification defined in 3GPP TS 32.302 [1]. It carries the semantics of the notification identifier. |
| eventTime | M, F | | This parameter is specified by NotificationIRPNotification defined in 3GPP TS 32.302 [1]. It carries the time when the subject notification is emitted. |
| systemDN | C, F | IRPAgent.systemDN | This parameter is specified by NotificationIRPNotification defined in 3GPP TS 32.302 [1]. It carries the *systemDN* of the *IRPAgent* related to the *TestManagementIRP*. |
| notificationType | M, F | "notifyTestResults" | This parameter is specified by NotificationIRPNotification defined in 3GPP TS 32.302 [1] |
| testInvocationId | O | TestInvocation.testInvocationId | |
| testInvocationInitiator | C, F | TesterObject.testInvocationInitiator | |
| testOutcome | O, see Note 1 | TesterObject.testOutcome | This parameter shall be included only in the last notification emitted by a TO. In this way the TO indicates that it is sending no more notifications. |
| mORT | O | TesterObject.theMORT | This parameter identifies the object instance of the MORT that was subject to the test. |
| proposedRepairActions | O | TesterObject.proposedRepairActions | |
| additionalInformation | O | TesterObject.additionalInformation | This parameter allows the inclusion of any |

| Parameter Name | Qualifier | Matching Information | Comment |
|---|---|---|---|
| | | | additional information in the notification. As such, it may carry a test result data set.<br>The exact semantics of this parameter is outside the scope of this specification.<br>This parameter may contain no information or be absent, if the test results are captured in a file. It may be present if the test results are not captured in a file. |
| fileReference | M, see Note 1 | It carries the reference to a file that contains the test result data set. | This parameter shall contain no information or be absent if there is no test result captured in a file. It shall contain information if the test results are captured in a file. |
| fileExpiryDate | M, see Note 1 | It carries the date and time after which the file, whose reference is carried by fileReference of this notification, may be removed. | This parameter shall contain no information or be absent if fileReference carries no information or absent. Otherwise, it shall contain a valid future date and time. |

Note 1: As for the correct interpretation of this qualifier refer to the comment column.

### 6.4.1.3 Triggering Events for the Resource Self Test

For the resource self test the events triggering the emisson of test result notifications are:

- Termination of the test execution

The resource self test may be terminated explicitly by a test termination request. The events triggering an implicit termination are

- Fulfillment of the conditions for a successful termination of the test
- Fulfillment of the conditions for a premature termination of the test
- Occurance of an error situation

#### 6.4.1.3.1 From-State

testTerminateRequestReceived OR testCompleted OR prematureTermination OR testTimedOut OR errorSituationOccured

| Assertion Name | Definition |
|---|---|
| testTerminateRequest Received | The object with the ability to receive and react upon test requests has received a test termination request (see Note 1). |
| testCompleted | The predefined conditions for a successful completion of the test are fulfilled (see Note 1). |
| prematureTermination | The predefined conditions for a premature termination of the test are fulfilled (see Note 1). |
| errorSituationOccured | An error situation has occurred during the test execution and the tester object has aborted the test invocation (see Note 1). |

Note 1: The conditions to satisfy this trigger are related to the VSE TO definition and therefore, their specifications are outside the scope of 3GPP.

#### 6.4.1.3.2 To-State

testTerminated

| Assertion Name | Definition |
|---|---|
| testTerminated | The test has been terminated successfully. |

# Annex A (informative): Change history

| Change history | | | | | | | |
|---|---|---|---|---|---|---|---|
| Date | TSG # | TSG Doc. | CR | Rev | Subject/Comment | Old | New |
| Jun 2002 | S_16 | SP-020328 | -- | -- | Submitted to TSG SA #16 for Information | 1.0.0 | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |