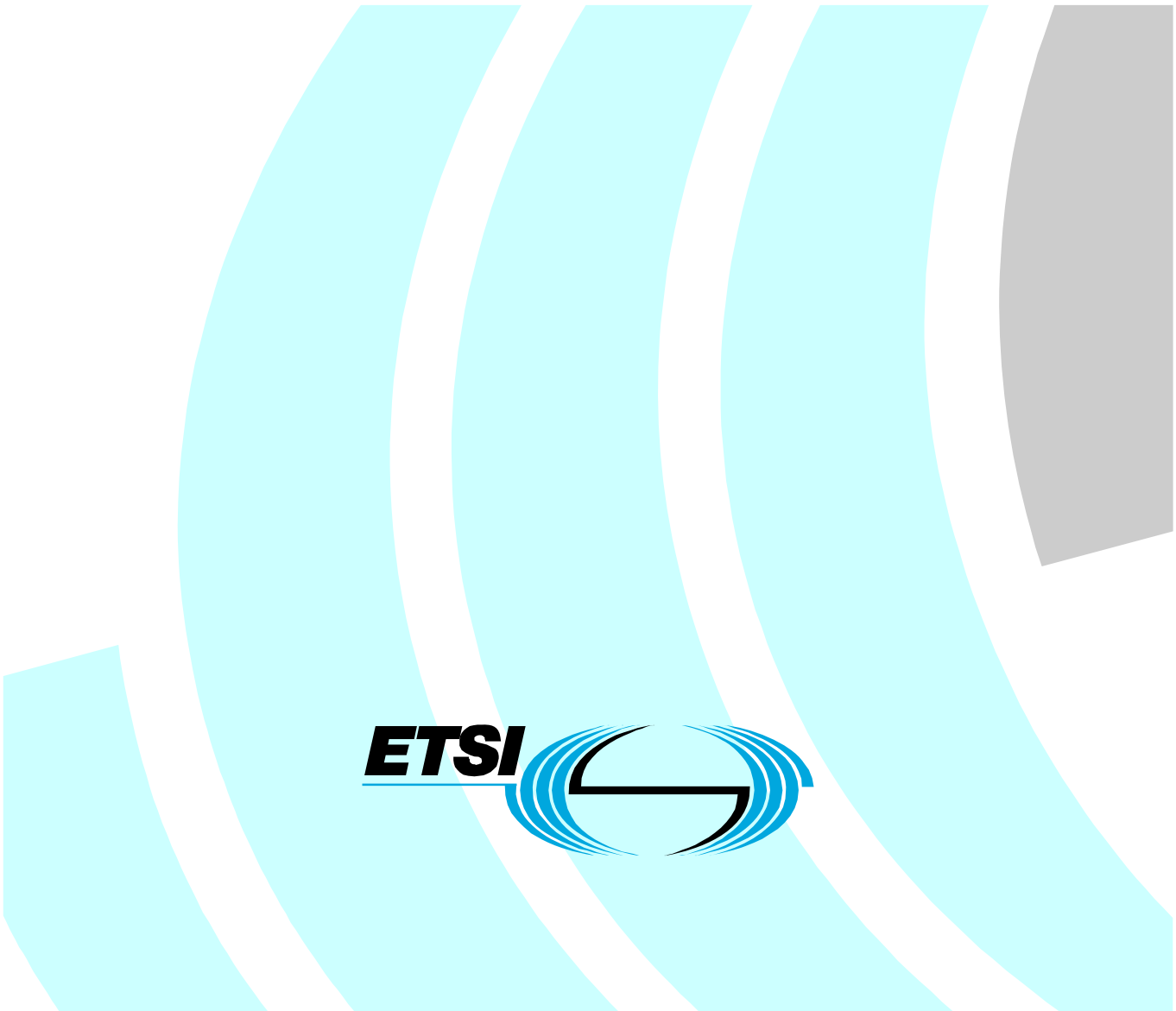
ETSI TS 102 027-3 V4.2.5 (2008-01)

Technical Specification

Methods for Testing and Specification (MTS); Conformance Test Specification for SIP (IETF RFC 3261); Part 3: Abstract Test Suite (ATS) and partial Protocol Implementation eXtra Information for Testing (PIXIT) proforma



Reference
RTS/MTS-0097-3[3]

Keywords
TTCN, ATS, IP, PIXIT, SIP, conformance, testing, VoIP

ETSI

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

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

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

Contents

Intellectual Property Rights

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

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

Foreword

This Technical Specification (TS) has been produced by ETSI Technical Committee Methods for Testing and Specification (MTS).

The present document is part 3 of a multi-part deliverable covering Conformance Test Specification for SIP (IETF RFC 3261 [1]), as identified below:

Part 1: "Protocol Implementation Conformance Statement (PICS) proforma";

Part 2: "Test Suite Structure and Test Purposes (TSS&TP)";

Part 3: "Abstract Test Suite (ATS) and partial Protocol Implementation eXtra Information for Testing (PIXIT) proforma".

1 Scope

The present document specifies the Abstract Test Suite (ATS) for the Session Initiation Protocol (SIP) as defined in RFC 3261 [1].

The objective of the present document is to provide a basis for conformance tests for SIP equipment giving a high probability of inter-operability between different manufacturer's SIP equipments.

Annex A provides the TTCN-3 part of the ATS.

Annex B provides the Partial Protocol Implementation Extra Information for Testing (PIXIT) Proforma of the ATS.

Annex C provides the Protocol Conformance Test Report (PCTR) Proforma of the ATS.

2 References

References are either specific (identified by date of publication and/or edition number or version number) or non-specific.

- For a specific reference, subsequent revisions do not apply.
- Non-specific reference may be made only to a complete document or a part thereof and only in the following cases:
  - if it is accepted that it will be possible to use all future changes of the referenced document for the purposes of the referring document;
  - for informative references.

Referenced documents which are not found to be publicly available in the expected location might be found at http://docbox.etsi.org/Reference.

For online referenced documents, information sufficient to identify and locate the source shall be provided. Preferably, the primary source of the referenced document should be cited, in order to ensure traceability. Furthermore, the reference should, as far as possible, remain valid for the expected life of the document. The reference shall include the method of access to the referenced document and the full network address, with the same punctuation and use of upper case and lower case letters.

NOTE: While any hyperlinks included in this clause were valid at the time of publication ETSI cannot guarantee their long term validity.

2.1 Normative references

The following referenced documents are indispensable for the application of the present document. For dated references, only the edition cited applies. For non-specific references, the latest edition of the referenced document (including any amendments) applies.

[1] IETF RFC 3261: "SIP: Session Initiation Protocol".

[2] ETSI ES 201 873-1: "Methods for Testing and Specification (MTS); The Testing and Test Control Notation version 3; Part 1: TTCN-3 Core Language".

[3] ETSI ES 201 873-5: "Methods for Testing and Specification (MTS); The Testing and Test Control Notation version 3; Part 5: TTCN-3 Runtime Interface (TRI)".

[4] ISO/IEC 9646-4: "Information technology - Open Systems Interconnection - Conformance testing methodology and framework - Part 4: Test realization".

[5] ISO/IEC 9646-5: "Information technology - Open Systems Interconnection - Conformance testing methodology and framework - Part 5: Requirements on test laboratories and clients for the conformance assessment process".

[6] ISO/IEC 9646-6: "Information technology - Open Systems Interconnection - Conformance testing methodology and framework - Part 6: Protocol profile test specification".

[7] IETF RFC 2224: "NFS URL Scheme".

[8] ETSI TS 102 027-2: "Methods for Testing and Specification (MTS); Conformance Test Specification for SIP (IETF RFC 3261); Part 2: Test Suite Structure and Test Purposes (TSS & TP)".

3 Definitions and abbreviations

3.1 Definitions

For the purposes of the present document, the terms and definitions given in RFC 3261 [1], ES 201 873-1 [2] and ES 201 873-5 [3] apply.

3.2 Abbreviations

For the purposes of the present document, the abbreviations given in ES 201 873-1 [2], ES 201 873-5 [3], RFC 3261 [1] and the following apply:

ABNF	Augmented BNF (for syntax specification - see RFC 2224 [7])
ATM	Abstract Test Method
ATS	Abstract Test Suite
BNF	Backus Naur Form
CRLF	Carriage Return Line Feed
EDS	Encoding/Decoding System
ETS	Executable Test Suite
IP	Internet Protocol
IUT	Implementation Under Test
MOT	Means Of Testing
MTC	Main Test Component
PA	Platform Adapter
PCTR	Protocol Conformance Test Report
PDU	Protocol Data Unit
PICS	Protocol Implementation Conformance Statement
PIXIT	Protocol Implementation eXtra Information for Testing
PTC	Parallel Test Component
SA	SUT Adapter
SIP	Session Initiation Protocol
SUT	System Under Test
TC	Test Cases
TCI	TTCN-3 Control Interface
TCP	Transmission Control Protocol
TE	TTCN-3 Executable
TM	Test Management
TP	Test Purpose
TRI	TTCN-3 Runtime Interface
TS	Test System
TSS	Test Suite Structure
TTCN-3	Testing and Test Control Notation (version 3)
UDP	User Datagram Protocol

4 Abstract Test Method (ATM)

This clause describes the ATM used to test Draft IETF SIP RFC as defined in RFC 3261 [1].

4.1 Network architecture

The basic SIP network architecture is defined in figure 1. The ATS defines test cases for the IUT being in the role of each displayed entity.

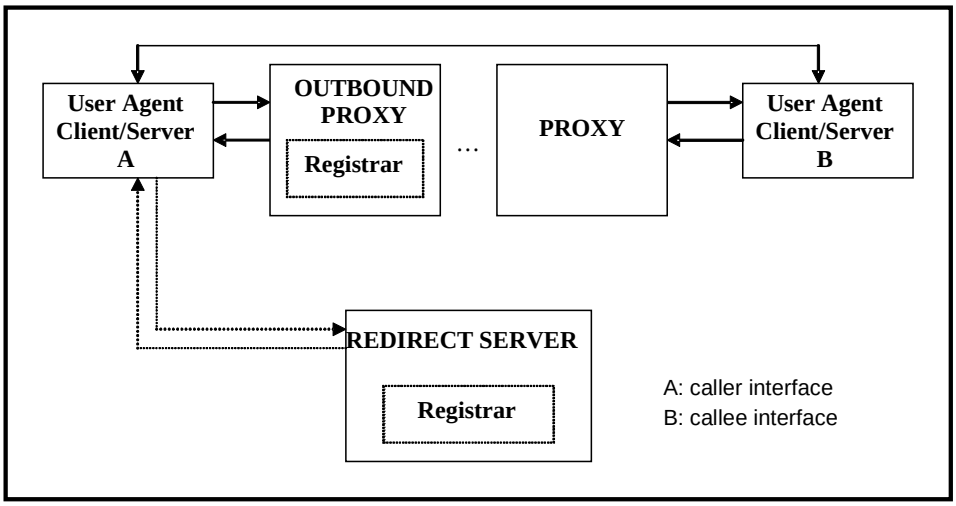


Figure 1: SIP network architecture

4.2 Protocol architecture

The Implementation Under Test (IUT) for which this test case specification applies consists of the SIP protocol (see figure 2).

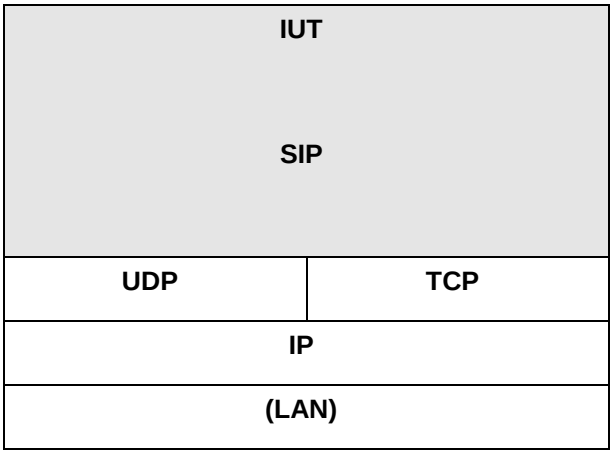


Figure 2: SIP protocol architecture

4.3 Test System architecture

Test systems that implement this ATS shall conform to the requirements as defined in this clause.

4.3.1 Structure

An abstract architecture for a Test System (TS) implementing a TTCN-3 ATS is displayed in figure 3 and also stated in TRI.

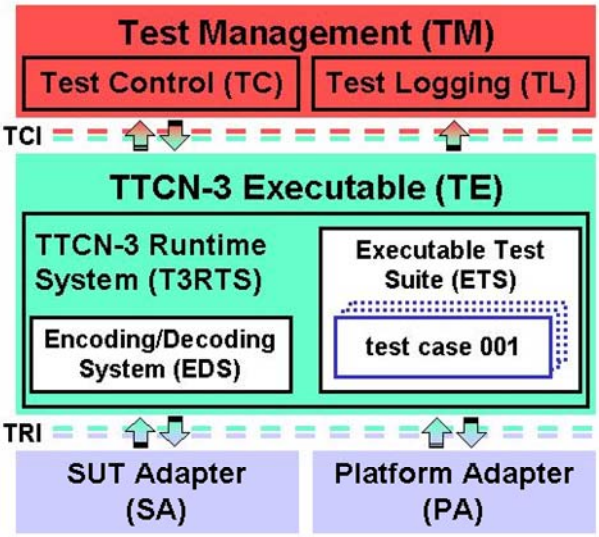


Figure 3: Abstract Test System architecture

A TS has two interfaces, the TTCN-3 Control Interface (TCI) and the TTCN-3 Runtime Interface (TRI), which specify the interface between Test Management (TM) and TTCN-3 Executable (TE) entities, and TE, SUT Adapter (SA) and Platform Adapter (PA) entities, respectively. Out of these two interfaces the TRI has been standardized in ES 201 873-5 [3], whereas the specification and implementation of the TCI is currently considered to be proprietary.

The part of TS that deals with interpretation and execution of TTCN-3 modules, i.e. the Executable Test Suite (ETS), is shown as part of the TTCN-3 Executable (TE). This ETS corresponds either to the executable code produced by a TTCN-3 compiler or a TTCN-3 interpreter from the TTCN-3 ATS in a TS implementation. The remaining part of the TS, which deals with any aspects that cannot be concluded from information being present in the TTCN-3 ATS alone, can be decomposed into Test Management (TM), SUT Adapter (SA), and Platform Adapter (PA) entities. In general, these entities cover a TS user interface, test execution control, test event logging, communication of test data with the SUT, and timer implementation.

The SA for a SIP TS shall implement the TRI adaptation as well as the SIP transport protocol architecture described in clause 4.2. The Encoding/Decoding System (EDS) entity with the TE and Test Logging (TL) entity within the TM shall comply with the conventions defined in following clauses.

4.3.2 Encoding/Decoding System requirements

SIP is a text-based protocol that allows different syntactical presentations of the same information. In general, an implementation of this ATS should use a EDS to parse received encoded messages into TTCN-3 type structures and values, and encode structured TTCN-3 type structures and values into encoded messages. This EDS is not part of the ATS. Still all encoded messages, i.e. the messages as they are transmitted by the SA to or received by the SA from the SUT, shall be logged.

The following terms shall be used for the conventions defined below:

Syntactic delimiter	syntactic delimiters are characters like "=" or ";" that are used to separate encoded values.
LWS	linear white spaces as defined in RFC 3261 [1].
Parameter name	name of header parameters as defined in RFC 3261 [1].

Parameter value	the value of a parameter as defined in RFC 3261 [1].
Undefined method	an undefined method is a method other than: "INVITE", "ACK", "OPTIONS", "BYE", "CANCEL" and "REGISTER".
Undefined header	an undefined header is a header other than general-header, entity-header, request-header and response header as defined in RFC 3261 [1].
Unexpected header	an unexpected header is a header, which shall not be present in a specific request message. This definition complies to the definition of NOT APPLICABLE in RFC 3261 [1], section 20 for request messages.

4.3.2.1 Decoding requirements

TTCN-3 fields should not contain syntactic delimiters like white space, semicolon, equal characters etc. in fully decoded fields. Instead the information provided by a parser shall be used to build the decoded message in TTCN-3. Decoded messages shall use the TTCN-3 enumeration types where ever appropriate, e.g. for the method and the header field name.

For **`charstring`** fields the following decoding rules shall be applied by the EDS:

1) Subsequent LWS shall compress to a single space character " ".

2) Decoded parameter names shall use only lower case letters.

3) Parameter values containing an integer value shall be decoded to a TTCN-3 integer value where a TTCN-3 **`integer`** type is used for a SIP parameter value.

The following decoding rules shall be applied by the EDS to each received message in the following order:

1) In case a request message indicating an undefined method is received by the test system, the message shall not be passed in the TE to the ETS. However the message is subject to logging as defined in clause 4.3.3 ("Logging conventions").

2) In case an undefined header has been received the header field shall be decoded as **`UndefinedHeader`** field.

RFC 3261 [1] allows for multiple header field values of the same kind to either arrive in one or multiple occurrences of the corresponding header field. The SIP ATS has been written assuming only the first format. Therefore, should the EDS receive multiple header fields of the same kind in a SIP message, e.g. of a Via header field, it shall convert them into the equivalent single header field with multiple values. This can be achieved by adding the value of, e.g. the second received Via header field as the last value to the value(s) of the first Via header field.

4.3.2.2 Encoding requirements

Encoders shall follow all encoding rules that are defined in RFC 3261 [1] when encoding structured values received from templates. This applies in particular to but it is not restricted to section 7.3.1 of RFC 3261 [1].

Values of type **`Raw`** shall be send to the SUT without any modification.

4.3.3 Logging conventions

As the ATS defines on an abstract level the message exchange between TS and SUT the messages encoded messages send and received shall be logged. The TM entity in the TS shall provide access to this log.

5 Untestable Test Purposes (TP)

This clause gives a list of TP, which are not implemented in the ATS due to the chosen ATM or other restrictions:

- None.

6 ATS conventions

The ATS conventions are intended to give a better understanding of the ATS, but they also describe the conventions made for the development of the ATS. These conventions shall be considered during any later maintenance or further development of the ATS.

The ATS conventions contain two clauses, the naming conventions and the implementation conventions. The naming conventions describe the structure of the naming of all ATS elements. The implementation conventions describe the functional structure of the ATS.

6.1 Naming conventions

6.1.1 Type definitions

This clause describes the naming conventions used for structured and unstructured types as well as for the field names of structured types.

6.1.1.1 General

Type identifiers use mixed cased with the first letter of each internal word capitalized.

EXAMPLE 1: `RequestLine`

Field identifiers use mixed cased with a lowercase first letter. Internal words start with a capital letter.

EXAMPLE 2: `requestLine`

Names of type as well as field identifiers attempt to follow the naming chosen of ABNF elements (if a counterpart exists) as closely as possible.

EXAMPLE 3:
```
NameAddr
hostName
```

In case type and identifier names should give a hint on their structure the term describing the structure should be separated with an underscore ("_") at the end of the name.

EXAMPLE 4:
```
CommaParam_List
ContactBody_Union
```

6.1.1.2 PDU Type Definition

Names of types used as PDUs follow the general conventions as defined in clause 6.1.1.1.

EXAMPLE: `Request`

6.1.2 Template definition

Template identifiers consist of the type name, an identifier denoting whether the template is for sending or receiving and a sequential number.

EXAMPLE: `Request_r_1` denotes a template from type `Request` that is intended for reception.
`CommaParam_List_s_25` denotes a template from type `CommaParam_List` that is intended for sending.

The sequential number is used only to distinguish between templates for the same type and direction and includes no other information.

6.1.3 Constant declarations

Identifiers for either internal or external constants, use only uppercase letters. Internal words are separated by an underscore ("_").

EXAMPLE: `SIP_VERSION`

6.1.4 Enumeration declarations

While identifiers for the enumeration type follow the conventions as defined in clause 6.1.1. "Type definitions", enumerations elements use only uppercase letters, which are suffixed by "_E" to distinguish them from constants. Internal words are separated by an underscore character ("_").

EXAMPLE: Enumeration type
`FieldName`

Enumeration value
`FROM_E`

6.1.5 Module parameter declarations

Identifiers for module parameters follow the general rules as defined in clause 6.1.3. Numbers are separated from words using the underscore character "_".

EXAMPLE: CAP_1

6.1.6 Variable declarations and formal parameters

Identifiers for variables follow the general rules for field names as defined in clause 6.1.1. "Type definitions" with the following addition: component variables are prefix with "v_", variables of local visibility (declared at the beginning of the control part, of a test case, a function, an altstep or of a statement block) are prefixed with "vl_" except single-letter variable names (like i, j).

Identifiers for formal parameters follow the general rules for field names as defined in clause 6.1.1. "Type definitions" with the addition that are prefixed with "loc_".

6.1.7 Function declarations

Identifiers for either internal or external functions use mixed case with a lowercase first letter. Internal words start with a capital letter, and "to" is abbreviated with "2".

EXAMPLE: `synchronisePtc2Check ()`

6.1.8 Test Case declarations

6.1.8.1 General

All test cases are listed in the order in which they appear in the Test Suite Structure (TSS) and TP document. Grouping is used to reflect the TSS.

6.1.8.2 Test Case (TC) identifier

The identifier of the test case is constructed in the same way as for the test purpose described in TS 102 027-2 [8]. The identifier of a TC is built according to table 1.

Table 1: TP identifier naming convention scheme

Identifier: **<protocol>_<main functionality>_<role>_<functionality>_<type>_<nn>**	
<protocol>	SIP
<main functionality>	Registration (RG), Session (SS), Message Parsing (MP)
<role> =	Registrant (RT), Registrar (RR) Originating Endpoint (OE), Terminating Endpoint (TE), Proxy (PR), Redirect (RD)
<functionality> (optional)	Call Establishment (CE), Call Release(CR)
<type>	Valid Behaviour (V), Invalid Behaviour (I), Inopportune Behaviour (O)
<nnn> =	sequential number (001 to 999)

If variants for one TP have been specified a lower case letter, starting with "a" has been appended to each of the variants.

EXAMPLE:
```
SIP_RG_RT_V_001
SIP_RG_RT_V_004a
SIP_RG_RT_V_004c
```

If more than 26 variants have been defined an additional lower case letter, starting with "a" will be appended.

6.1.9 Timer declarations

Two types of timers can be identified:

1) Standardized:

   - Those defined in RFC 3261 [1], e.g. `T1`. They use exactly the same name as in the standard.

   As there is a tolerance margin accepted for these timers, three values are needed:

   - the maximum value allowed, which will use the suffix `"_max`";
   - the minimum value allowed, which will use the suffix "`_min`";
   - the value actually implemented, with no suffix;

EXAMPLE 1: `T1 _max`, `T1_min`, and `T1`.

2) Not standardized:

   - Those not defined in the protocol standard, i.e. for execution use, e.g. a timer waiting for a response. These timers begin with the prefix "`T_`", followed by a string in lowercase letters.

EXAMPLE 2: `T_resp` represents a timer for controlling the response time of the IUT.

6.1.10 Group names

Group names follow the same general conventions as defined in clause 6.1.1.1.

EXAMPLE: SubtypesTemplateDeclarations

Where appropriate group names reflect the hierarchic group structure.

6.2 Implementation conventions

6.2.1 Type definitions

The following clause gives an overview on the mapping of SIP messages and structures as defined in RFC 3261 [1] and their corresponding TTCN-3 types.

6.2.1.1 Messages

Distinct types have been defined for SIP request and response messages, as they have a different internal structure. Messages are defined as a record structure containing three fields, a request/status line field, a header field and a message body field, and a payload field which contains the whole message as it has been received in its text format. For syntactic delimiters, like Carriage Return Line Feed (CRLF), colon ":", etc. no extra fields are defined as they are already removed by the EDS.

The definition of invalid messages for sending and receiving is discussed in clause 6.2.1.1.3.

6.2.1.1.1 Request messages

For all possible valid requests one generic type has been defined.

EXAMPLE: The generic request message

```
type record Request {
  RequestLine     requestLine,
  MessageHeader   msgHeader,
  charstring      messageBody optional,
  Payload         payload optional
}
```

6.2.1.1.2 Response messages

For all possible valid responses one generic type has been defined.

EXAMPLE: The generic response message

```
type record Response {
    StatusLine      statusLine,
    MessageHeader   msgHeader,
    charstring      messageBody optional,
    Payload         payload optional
}
```

6.2.1.1.3 Raw messages

For defining syntactic torture or syntactically invalid messages a distinct type **Raw** and its type aliases **Response_Raw**, **Raw_REGISTER_Request**, **Raw_BYE_Request**, **Raw_Unknown_Request** and **Raw_INVITE_Request** have been defined. The **Raw** type is defined as a **charstring**. Messages using these types define exactly how the message shall be transmitted, thus giving the possibility to define the message on a character level.

EXAMPLE: A torture message

```
type charstring Raw
type Raw Raw_INVITE_Request;
template Raw_INVITE_Request rawMessage_s_1 =
    "INVITE sip:joe@foo.com SIP/2.0" & CRLF &
    "TO     :   " & CRLF
    " sip:joe@foo.com   ;" & TAB & "    tag   = 1918181833n" & CRLF &
    "Via   : SIP   /   2.0" ;
```

The **Raw** type and its aliases have not been used to describe valid receiving templates. Therefore, valid receiving templates will always use structured messages.

6.2.1.2 Headers

The following clause defines the mapping of the header field section of a SIP message as defined in RFC 3261 [1].

For all message one message header type has been defined. These message header type include all possible header fields that are allowed to be present in the header section of SIP messages according to RFC 3261 [1]. As individual header fields may appear in any order in a SIP message these header types are defined as **set**s. Header fields which are optional in all SIP messages use the **optional** keyword.

EXAMPLE 1: A header structure for an SIP request.

```
type set MessageHeader {          Accept                accept optional,
  AcceptEncoding        acceptEncoding optional,
  AcceptLanguage        acceptLanguage optional,
  AlertInfo         alertInfo optional,
  Allow             allow optional,
  AuthenticationInfo    authenticationInfo optional,
  Authorization     authorization optional,
  CallId                callId optional,
  CallInfo          callInfo optional,
  Contact               contact optional,
  ContentDisposition    contentDisposition optional,
  ContentEncoding       contentEncoding optional,
  ContentLanguage       contentLanguage optional,
  ContentLength     contentLength optional,
  ContentType           contentType optional,
  CSeq              cSeq optional,
  Date              date optional,
  ErrorInfo             errorInfo optional,
  Expires           expires optional,
  From              fromField optional,
  InReplyTo             inReplyTo optional,
  MaxForwards           maxForwards optional,
  MimeVersion           mimeVersion optional,
  MinExpires            minExpires optional,
  Organization      organization optional,
  Priority          priority optional,
  ProxyAuthenticate proxyAuthenticate optional,
  ProxyAuthorization    proxyAuthorization optional,
  ProxyRequire      proxyRequire optional,
  RecordRoute           recordRoute optional,
  ReplyTo               replyTo optional,
  Require           require optional,
  RetryAfter            retryAfter optional,
  Route                 route optional,
  Server                server optional,
  Subject               subject optional,
  Supported             supported optional,
  Timestamp             timestamp optional,
```

```
    To                    toField optional,
    Unsupported           unsupported optional,
    UserAgent             userAgent optional,
    Via                   via optional,
    Warning               warning optional,
    WwwAuthenticate       wwwAuthenticate optional,
    UndefinedHeader_List  undefinedHeader_List optional
}
```

Each header field is substructured into record and consists of at least the field fieldName.

EXAMPLE 2: The Accept header field type:

```
type record Accept {
    FieldName   fieldName (ACCEPT_E),
    AcceptBody_List acceptArgs optional
}
```

The field name type is an enumeration type that contains an entry for each header field type. All enumerations are capitalized and suffixed with "_E" to distinguish enumeration values from constants.

EXAMPLE 3:

```
type enumerated FieldName {
   TO_E, FROM_E, …
}
```

Depending on the definition of every particular header field in the ABNF [1] and its relevance to the SIP TP, types for header field values have been defined and substructured differently. In general, every header field value is at least structured according to its header field definition in the SIP ABNF.

EXAMPLE 4: A header field type with a simple header field value type like **charstring:**

```
type record Priority {
    FieldName   fieldName(PRIORITY_E),
    charstring  priorityValue
}
```

EXAMPLE 5: A header field type with a header field value consisting of multiple fields:

```
type record MimeVersion {
    FieldName   fieldName (MIME_VERSION_E),
    integer     majorNumber,
    integer     minorNumber
}
```

Types for header field values may, however, be further sub structured if the SIP ABNF allows.

EXAMPLE 6: A header field type with a sub structured header field value (see the Accept header definition above):

```
type set of AcceptBody AcceptBody_List;
```

The level of substructure is again header field dependent. Should a header field value have a potential for further sub structuring (in the SIP ABNF) it is declared as a type equivalent to **charstring**. Note that values of the latter types still contain syntactic delimiters.

EXAMPLE 7: The value of the below type still contains the "@" delimiter:

```
type charstring CallidString:
```

In case a SIP header field might contain a choice of types of information **unions** have been used.

EXAMPLE 8: Sub structured union field value type:

```
type union ContactBody {
..charstring        wildcard,
..ContactAddress_List   contactAddresses
  }
```

The contact body can contain either a wildcard or a contact address list.

A list, i.e. the **set of** construct has been used for collecting information of the same type. In case the order is important the **record of** construct has been used. List types have been created for header fields that may have multiple occurrences of header field values as well as for parameters in header fields. To avoid multiple encodings for the same type, the naming of list types derived from generic parameters has been based on its encoding.

EXAMPLE 9: Collection of multiple header field values of the same type:

```
type record Via {
    FieldName       fieldName (VIA_E),
    ViaBody_List    viaBody
}

type record of ViaBody ViaBody_List;
```

`ViaBody_List` contains all values of the `Via` header field.

EXAMPLE 10: Collection of multiple generic parameters:

```
type set of GenericParam SemicolonParam_List;
```

6.2.2 Constant definitions

6.2.2.1 Constants

Syntactic delimiters for raw messages, e.g. "<", or "@" have been defined as constants.

EXAMPLE:

```
type charstring specialchar length (1);
const specialchar AT := "@";
```

6.2.2.2 External Constants

No external contant definition is used.

6.2.3 Module Parameters

PICS/PIXIT parameters are defined as TTCN-3 module parameters.

6.2.4 Template definitions

Modifications of templates have been used to allow focusing on the essential parts of a test message. However these mechanisms have to be checked very carefully as they can be counter intuitive.

6.2.5 Dynamic part

No conventions for the dynamic part have been defined yet.

7 PCTR conformance

A test laboratory, when requested by a client to produce a PCTR, is required, as specified in ISO/IEC 9646-5 [5], to produce a PCTR conformant with the PCTR template given in annex B of ISO/IEC 9646-5 [5].

Furthermore, a test laboratory, offering testing for the ATS specification contained in annex C, when requested by a client to produce a PCTR, is required to produce a PCTR conformant with the PCTR proforma contained in annex A.

A PCTR which conforms to this PCTR proforma specification shall preserve the content and ordering of the clauses contained in annex A. Clause A.6 of the PCTR may contain additional columns. If included, these shall be placed to the right of the existing columns. Text in italics may be retained by the test laboratory.

8 PIXIT conformance

A test realizer, producing an executable test suite for the Abstract Test Suite (ATS) specification contained in annex C, is required, as specified in ISO/IEC 9646-4 [4], to produce an augmented partial PIXIT proforma conformant with this partial PIXIT proforma specification.

An augmented partial PIXIT proforma which conforms to this partial PIXIT proforma specification shall, as a minimum, have contents which are technically equivalent to annex B. The augmented partial PIXIT proforma may contain additional questions that need to be answered in order to prepare the Means Of Testing (MOT) for a particular Implementation Under Test (IUT).

A test laboratory, offering testing for the ATS specification contained in annex C, is required, as specified in ISO/IEC 9646-5 [5], to further augment the augmented partial PIXIT proforma to produce a PIXIT proforma conformant with this partial PIXIT proforma specification.

A PIXIT proforma which conforms to this partial PIXIT proforma specification shall, as a minimum, have contents which are technically equivalent to annex B. The PIXIT proforma may contain additional questions that need to be answered in order to prepare the test laboratory for a particular IUT.

9 ATS Conformance

The test realizer, producing a Means Of Testing (MOT) and Executable Test Suite (ETS) for the present document, shall comply with the requirements of ISO/IEC 9646-4 [4]. In particular, these concern the realization of an Executable Test Suite (ETS) based on each ATS. The test realizer shall provide a statement of conformance of the MOT to the present document.

An ETS which conforms to the present document shall contain test groups and test cases which are technically equivalent to those contained in the ATS in annex C. All sequences of test events comprising an abstract test case shall be capable of being realized in the executable test case. Any further checking which the test system might be capable of performing is outside the scope of the present document and shall not contribute to the verdict assignment for each test case.

Test laboratories running conformance test services using this ATS shall comply with ISO/IEC 9646-5 [5].

A test laboratory which claims to conform to this ATS specification shall use an MOT which conforms to this ATS.

Annex A (normative): Abstract Test Suite (ATS)

A.1 The ATS in TTCN-3 core (text) format

This ATS has been produced using the Testing and Test Control Notation (TTCN) according to ES 201 873-1 [2].

The TTCN-3 core (text) representation corresponding to this ATS is contained in an ASCII files (SIP_Steps.ttcn, SIP_Messaging.ttcn, SIP_TypesAndConf.ttcn, SIP_Registration.ttcn, SIP_QueryingCapabilities.ttcn, SIP_Templates.ttcn, SIP_CallControl.ttcn and SIP_MainModule.ttcn contained in archive ts_10202703v040201p0.zip) which accompanies the present document.

Where an ETSI Abstract Test Suite (in TTCN-3) is published in both core and tabular format these two forms shall be considered equivalent. In the event that there appears to be syntactical or semantic differences between the two then the problem shall be resolved and the erroneous format (whichever it is) shall be corrected.

Annex B (normative): Partial PIXIT proforma

The PIXIT Proforma is based on ISO/IEC 9646-6 [6]. Any needed additional information can be found in there.

B.1 Identification summary

Table B.1

PIXIT Number:	
Test Laboratory Name:	
Date of Issue:	
Issued to:	

B.2 ATS summary

Table B.2

Protocol Specification:	
Protocol to be tested:	
ATS Specification:	
Abstract Test Method:	

B.3 Test laboratory

Table B.3

Test Laboratory Identification:	
Test Laboratory Manager:	
Means of Testing:	
SAP Address:	

B.4 Client identification

Table B.4

Client Identification:	
Client Test manager:	
Test Facilities required:	

B.5 SUT

Table B.5

Name:	
Version:	
SCS Number:	
Machine configuration:	
Operating System Identification:	
IUT Identification:	
PICS Reference for IUT:	
Limitations of the SUT:	
Environmental Conditions:	

B.6 Protocol layer information

B.6.1 Protocol identification

Table B.6

Name:	
Version:	
PICS References:	

B.6.2 IUT information

Table B.7: IP parameters

Name	Type	Comments	Value
PX_UDP	boolean	True if UDP transport is used by the IUT (NOTE: TCP is currently not supported in this SIP ATS, so that this PIXIT value shall always be Yes)	Yes
PX_TRANSPORT	charstring	Used transport in uppercase "UDP" / "TCP"	
PX_IUT_PORT	integer	Default port number used by the IUT to exchange SIP messages	
PX_IUT_IPADDR	charstring	Default IP address used by the ETS to exchange SIP messages	
PX_IUT_PORT2	integer	Default port number used by the IUT to exchange SIP messages on PTC side	
PX_IUT_IPADDR2	charstring	Default IP address used by the IUT to exchange SIP messages on PTC side	
PX_ETS_PORT	integer	Default port number used by the ETS to exchange SIP messages on MTC side	
PX_ETS_IPADDR	charstring	Default IP address used by the IUT to exchange SIP messages on MTC side	
PX_ETS_PORT2	integer	Default port number used by the ETS to exchange SIP messages on PTC1 side	
PX_ETS_IPADDR2	charstring	Default IP address used by the IUT to exchange SIP messages on PTC1 side	
PX_ETS_PORT3	integer	Default port number used by the ETS to exchange SIP messages on PTC2 side	
PX_ETS_IPADDR3	charstring	Default IP address used by the IUT to exchange SIP messages on PTC2 side	
PX_PROXY_PORT	integer	Default port number used in 305 "Use Proxy" message	
PX_PROXY_IPADDR	charstring	Default IP address used in 305 "Use Proxy" message	

Table B.8: Registration parameters

Name	Type	Comments	Value
PX_DELTA_REGISTRATION	charstring	delta-seconds used in expires header field in 200 OK message to answer REGISTRATION request	
PX_ETS_LOCAL_DOMAIN	charstring	identity of the tester local domain on MTC side	
PX_ETS_LOCAL_USER	charstring	identity of the tester local user on MTC side	
PX_ETS_LOCAL_DOMAIN2	charstring	identity of the tester local domain on MTC or PTC side	
PX_ETS_LOCAL_USER2	charstring	identity of the tester local user on PTC side	
PX_HOME_REGISTRATION	boolean	IUT needs to register itself to its home register first	
PX_PR_MTC_REGISTRATION	boolean	True if the ETS shall register itself before running proxy test cases on MTC side	
PX_PR_PTC_REGISTRATION	boolean	True if the ETS shall register itself before running proxy test cases on PTC side	

Table B.9: Roaming parameters

Name	Type	Comments	Value
PX_ETS_LOCAL_THIRD_USER	charstring	identity of another tester local user (third party tester)	

Table B.10: Session parameters

Name	Type	Comments	Value
PX_TE_CALLEE_DOMAIN	charstring	hostname of the callee when IUT is the callee	
PX_TE_CALLEE_USERINFO	charstring	userinfo of the callee when IUT is the callee	
PX_UNKNOWN_DOMAIN	charstring	unknown hostname when IUT is the callee	
PX_UNKNOWN_USERINFO	charstring	unknown userinfo when IUT is the callee	
PX_UNAUTHORIZED_USER	charstring	Identity of the tester local user on MTC side, which is not authorized to update registration	

Table B.11: Header parameters

Name	Type	Comments	Value
PX_CREDENTIALS	credentials	Value used in authorization header in REGISTER sent to the IUT	
PX_STR_CREDENTIALS	charstring	String Value used in authorization header in REGISTER sent to the IUT for MG group	
PX_REALM	charstring	realm value understood by the IUT	
PX_CONTENCOD_UNSUPPORTED	charstring	Content encoding mechanism that is not supported by the IUT	
PX_OPTION_UNSUPPORTED	charstring	Option set in Request header field that is not supported	

Table B.12: Body parameters

Name	Type	Comments	Value
PX_ SDPBODY	charstring	SDP parameter proposed by the ETS	
PX_SDPBODY2	charstring	additional SDP parameter proposed by the ETS	
PX_SDPBODY_UNSUPPORTED	charstring	SDP parameter proposed by the ETS that is not supported by the IUT	
PX_SDPBODY_65535	charstring	SDP parameter proposed by the ETS that makes the 200 OK or INVITE request of 63 535 bytes long	
PX_SDPBODY_TOO_LARGE	charstring	SDP parameter proposed by the that makes the 200 OK or INVITE request too large	

Table B.13: Timers

Name	Type	Comments	Value
PX_T1	float	T1 RTT estimate (500 ms)	
PX_T2	float	T2 maximum retransmit interval for non-invite requests and invite responses (4 000 ms)	
PX_T4	float	Maximum duration a message will remain in the network (5 000 ms)	
PX_TWAIT	float	TWait default value for waiting an operator action	
PX_TACK	float	default value for waiting an acknowledgement	
PX_TRESP	float	TResp default value for waiting for a response from the IUT	
PX_TNOACT	float	value for waiting no message from the IUT. Default value for waiting no message from the IUT. The given value should be less than value of SHORT_REGISTRATION constant (which is currently "3" (seconds))	
PX_TSYNC	float	default value to synchronize ptc	
PX_TGUARD	float	default value for an extra long timer to limit test execution	

Table B.14: Authentication parameters

Name	Type	Comments	Value
PX_REGISTRATION_AUTHENTICATION_ENABLED	boolean	True if authentication is enabled for registration messages	
PX_RFC2617_QOP	charstring	Quoted string of one or more tokens indicating the "quality of protection" values supported by the server. The value "auth" indicates authentication; the value "auth-int" indicates authentication with integrity protection. EXAMPLE: "auth, auth-int".	
PX_RFC2617_USERNAME	charstring	The name of user in the specified realm.	
PX_RFC2617_PASSWD	charstring	Password of USERNAME.	
PX_RFC2617_URI	charstring	URI	

Annex C (normative): PCTR proforma

The PCTR proforma is based on ISO/IEC 9646-6 [6]. Any needed additional information can be found in there.

C.1 Identification summary

C.1.1 Protocol conformance test report

Table C.1

PCTR Number:	
PCTR Date:	
Corresponding SCTR Number:	
Corresponding SCTR Date:	
Test Laboratory Identification:	
Test Laboratory Manager:	
Signature:	

C.1.2 IUT identification

Table C.2

Name:	
Version:	
Protocol specification:	
PICS:	
Previous PCTR if any:	

C.1.3 Testing environment

Table C.3

PIXIT Number:	
ATS Specification:	
Abstract Test Method:	
Means of Testing identification:	
Date of testing:	
Conformance Log reference(s):	
Retention Date for Log reference(s):	

C.1.4 Limits and reservation

Additional information relevant to the technical contents or further use of the test report, or the rights and obligations of the test laboratory and the client, may be given here. Such information may include restriction on the publication of the test report.

.....

.....

.....

.....

.....

.....

C.1.5 Comments

Additional comments may be given by either the client or the test laboratory on any of the contents of the PCTR, for example, to note disagreement between the two parties.

.....

.....

.....

.....

.....

.....

C.2 IUT Conformance status

This IUT has or has not been shown by conformance assessment to be non conforming to the specified protocol specification.

Strike the appropriate words in this sentence. If the PICS for this IUT is consistent with the static conformance requirements (as specified in clause C.3) and there are no "FAIL" verdicts to be recorded (in clause C.6) strike the words "has or", otherwise strike the words "or has not".

C.3 Static conformance summary

The PICS for this IUT is or is not consistent with the static conformance requirements in the specified protocol.

Strike the appropriate words in this sentence.

C.4 Dynamic conformance summary

The test campaign did or did not reveal errors in the IUT.

Strike the appropriate words in this sentence. If there are no "FAIL" verdicts to be recorded (in clause C.6t) strike the words "did or" otherwise strike the words "or did not".

Summary of the results of groups of test:

.....

.....

.....

.....

.....

.....

.....

.....

C.5 Static conformance review report

If clause C.3 indicates non-conformance, this clause itemizes the mismatches between the PICS and the static conformance requirements of the specified protocol specification.

.....

.....

.....

.....

.....

.....

.....

.....

C.6 Test campaign report

Table C.4

ATS Reference	Selected?	Run?	Verdict	Observations (Reference to any observations made in clause C.8)
SIP_RG_RT_V_001				
SIP_RG_RT_V_002				
SIP_RG_RT_V_003				
SIP_RG_RT_V_004				
SIP_RG_RT_V_005				
SIP_RG_RT_V_007				
SIP_RG_RT_V_008				
SIP_RG_RT_V_009				
SIP_RG_RT_V_011				
SIP_RG_RT_V_012				
SIP_RG_RT_V_013				
SIP_RG_RT_V_014				
SIP_RG_RT_V_015				
SIP_RG_RT_V_016				
SIP_RG_RT_V_017				
SIP_RG_RT_V_018				
SIP_RG_RT_V_019				
SIP_RG_RT_V_020				
SIP_RG_RT_TI_001				
SIP_RG_RT_TI_002				
SIP_RG_RT_TI_003				
SIP_RG_RT_TI_004				
SIP_RG_RT_TI_005				
SIP_RG_RT_TI_006				
SIP_RG_RT_TI_007				
SIP_RG_RR_V_001				
SIP_RG_RR_V_002				
SIP_RG_RR_V_003				
SIP_RG_RR_V_004				
SIP_RG_RR_V_005				
SIP_RG_RR_V_006				
SIP_RG_RR_V_007				
SIP_RG_RR_V_008				
SIP_RG_RR_V_009				
SIP_RG_RR_V_010				
SIP_RG_RR_V_011				
SIP_RG_RR_V_012				
SIP_RG_RR_V_013				
SIP_RG_RR_V_014				
SIP_RG_RR_V_015				
SIP_RG_RR_V_016				
SIP_RG_RR_V_017				
SIP_RG_RR_V_018				
SIP_RG_RR_V_019				
SIP_RG_RR_V_020				
SIP_RG_RR_V_021				
SIP_RG_RR_V_022				
SIP_RG_RR_I_001				
SIP_RG_RR_I_002				
SIP_RG_RR_I_003				
SIP_RG_RR_I_004				

ATS Reference	Selected?	Run?	Verdict	Observations (Reference to any observations made in clause C.8)
SIP_RG_RR_O_001				
SIP_RG_RR_O_002				
SIP_RG_RR_O_003				
SIP_CC_OE_CE_V_001				
SIP_CC_OE_CE_V_002				
SIP_CC_OE_CE_V_003				
SIP_CC_OE_CE_V_004				
SIP_CC_OE_CE_V_005				
SIP_CC_OE_CE_V_006				
SIP_CC_OE_CE_V_007				
SIP_CC_OE_CE_V_008				
SIP_CC_OE_CE_V_009				
SIP_CC_OE_CE_V_010				
SIP_CC_OE_CE_V_011				
SIP_CC_OE_CE_V_012				
SIP_CC_OE_CE_V_013				
SIP_CC_OE_CE_V_014				
SIP_CC_OE_CE_V_015				
SIP_CC_OE_CE_V_016				
SIP_CC_OE_CE_V_017				
SIP_CC_OE_CE_V_018				
SIP_CC_OE_CE_V_019				
SIP_CC_OE_CE_V_020				
SIP_CC_OE_CE_V_021				
SIP_CC_OE_CE_V_022				
SIP_CC_OE_CE_V_023				
SIP_CC_OE_CE_V_024				
SIP_CC_OE_CE_V_025				
SIP_CC_OE_CE_V_026				
SIP_CC_OE_CE_V_027				
SIP_CC_OE_CE_V_028				
SIP_CC_OE_CE_V_029				
SIP_CC_OE_CE_V_030				
SIP_CC_OE_CE_V_031				
SIP_CC_OE_CE_V_032				
SIP_CC_OE_CE_V_033				
SIP_CC_OE_CE_V_034				
SIP_CC_OE_CE_V_035				
SIP_CC_OE_CE_V_036				
SIP_CC_OE_CE_V_037				
SIP_CC_OE_CE_V_038				
SIP_CC_OE_CE_V_039				
SIP_CC_OE_CE_V_040				
SIP_CC_OE_CE_V_041				
SIP_CC_OE_CE_V_042				
SIP_CC_OE_CE_V_043				
SIP_CC_OE_CE_V_044				
SIP_CC_OE_CE_V_045				
SIP_CC_OE_CE_V_046				
SIP_CC_OE_CE_V_047				
SIP_CC_OE_CE_V_048				
SIP_CC_OE_CE_V_049				
SIP_CC_OE_CE_V_050				
SIP_CC_OE_CE_V_051				
SIP_CC_OE_CE_V_052				
SIP_CC_OE_CE_V_053				
SIP_CC_OE_CE_TI_001				
SIP_CC_OE_CE_TI_002				
SIP_CC_OE_CE_TI_003				
SIP_CC_OE_CE_TI_004				

ATS Reference	Selected?	Run?	Verdict	Observations (Reference to any observations made in clause C.8)
SIP_CC_OE_CE_TI_005				
SIP_CC_OE_CE_TI_006				
SIP_CC_OE_CE_TI_007				
SIP_CC_OE_CE_TI_008				
SIP_CC_OE_CE_TI_009				
SIP_CC_OE_CE_TI_010				
SIP_CC_OE_CE_TI_011				
SIP_CC_OE_CE_TI_012				
SIP_CC_OE_CR_V_001				
SIP_CC_OE_CR_V_002				
SIP_CC_OE_CR_V_003				
SIP_CC_OE_CR_V_004				
SIP_CC_OE_CR_V_005				
SIP_CC_OE_CR_V_006				
SIP_CC_OE_CR_V_007				
SIP_CC_OE_CR_V_008				
SIP_CC_OE_CR_V_009				
SIP_CC_OE_CR_V_010				
SIP_CC_OE_CR_V_011				
SIP_CC_OE_CR_V_012				
SIP_CC_OE_CR_V_013				
SIP_CC_OE_CR_V_014				
SIP_CC_OE_CR_V_015				
SIP_CC_OE_CR_I_001				
SIP_CC_OE_CR_TI_001				
SIP_CC_OE_CR_TI_002				
SIP_CC_OE_CR_TI_003				
SIP_CC_OE_CR_TI_004				
SIP_CC_OE_CR_TI_005				
SIP_CC_OE_CR_TI_006				
SIP_CC_OE_CR_TI_007				
SIP_CC_OE_CR_TI_008				
SIP_CC_OE_SM_V_001				
SIP_CC_OE_SM_V_002				
SIP_CC_TE_CE_V_001				
SIP_CC_TE_CE_V_002				
SIP_CC_TE_CE_V_003				
SIP_CC_TE_CE_V_004				
SIP_CC_TE_CE_V_005				
SIP_CC_TE_CE_V_006				
SIP_CC_TE_CE_V_007				
SIP_CC_TE_CE_V_008				
SIP_CC_TE_CE_V_009				
SIP_CC_TE_CE_V_010				
SIP_CC_TE_CE_V_011				
SIP_CC_TE_CE_V_012				
SIP_CC_TE_CE_V_013				
SIP_CC_TE_CE_V_014				
SIP_CC_TE_CE_V_015				
SIP_CC_TE_CE_V_016				
SIP_CC_TE_CE_V_017				
SIP_CC_TE_CE_V_018				
SIP_CC_TE_CE_V_019				
SIP_CC_TE_CE_V_020				
SIP_CC_TE_CE_V_021				
SIP_CC_TE_CE_V_022				
SIP_CC_TE_CE_V_023				

ATS Reference	Selected?	Run?	Verdict	Observations (Reference to any observations made in clause C.8)
SIP_CC_TE_CE_V_024				
SIP_CC_TE_CE_V_025				
SIP_CC_TE_CE_V_026				
SIP_CC_TE_CE_V_027				
SIP_CC_TE_CE_V_028				
SIP_CC_TE_CE_V_029				
SIP_CC_TE_CE_V_030				
SIP_CC_TE_CE_V_031				
SIP_CC_TE_CE_V_032				
SIP_CC_TE_CE_V_033				
SIP_CC_TE_CE_V_034				
SIP_CC_TE_CE_V_035				
SIP_CC_TE_CE_V_036				
SIP_CC_TE_CE_I_001				
SIP_CC_TE_CE_I_002				
SIP_CC_TE_CE_TI_001				
SIP_CC_TE_CE_TI_002				
SIP_CC_TE_CE_TI_003				
SIP_CC_TE_CE_TI_004				
SIP_CC_TE_CE_TI_005				
SIP_CC_TE_CE_TI_006				
SIP_CC_TE_CE_TI_007				
SIP_CC_TE_CE_TI_008				
SIP_CC_TE_CE_TI_009				
SIP_CC_TE_CE_TI_010				
SIP_CC_TE_CE_TI_011				
SIP_CC_TE_CE_TI_012				
SIP_CC_TE_CR_V_001				
SIP_CC_TE_CR_V_002				
SIP_CC_TE_CR_V_003				
SIP_CC_TE_CR_V_004				
SIP_CC_TE_CR_V_005				
SIP_CC_TE_CR_V_006				
SIP_CC_TE_CR_V_007				
SIP_CC_TE_CR_V_008				
SIP_CC_TE_CR_V_009				
SIP_CC_TE_CR_V_010				
SIP_CC_TE_CR_V_011				
SIP_CC_TE_CR_V_012				
SIP_CC_TE_CR_V_013				
SIP_CC_TE_CR_V_014				
SIP_CC_TE_CR_V_015				
SIP_CC_TE_CR_V_016				
SIP_CC_TE_CR_V_017				
SIP_CC_TE_CR_V_018				
SIP_CC_TE_CR_V_019				
SIP_CC_TE_CR_V_020				
SIP_CC_TE_CR_V_021				
SIP_CC_TE_CR_I_001				
SIP_CC_TE_CR_I_002				
SIP_CC_TE_CR_I_003				
SIP_CC_TE_CR_I_004				
SIP_CC_TE_CR_I_005				
SIP_CC_TE_CR_TI_001				
SIP_CC_TE_SM_V_001				

ATS Reference	Selected?	Run?	Verdict	Observations (Reference to any observations made in clause C.8)
SIP_CC_TE_SM_V_002				
SIP_CC_TE_SM_V_003				
SIP_CC_TE_SM_I_001				
SIP_CC_PR_MP_RQ_V_001				
SIP_CC_PR_MP_RQ_V_002				
SIP_CC_PR_MP_RQ_V_003				
SIP_CC_PR_MP_RQ_V_004				
SIP_CC_PR_MP_RQ_V_005				
SIP_CC_PR_MP_RQ_V_006				
SIP_CC_PR_MP_RQ_V_007				
SIP_CC_PR_MP_RQ_V_008				
SIP_CC_PR_MP_RQ_V_009				
SIP_CC_PR_MP_RQ_V_011				
SIP_CC_PR_MP_RQ_V_012				
SIP_CC_PR_MP_RQ_V_013				
SIP_CC_PR_MP_RQ_V_014				
SIP_CC_PR_MP_RQ_V_015				
SIP_CC_PR_MP_RQ_V_016				
SIP_CC_PR_MP_RQ_V_017				
SIP_CC_PR_MP_RQ_V_018				
SIP_CC_PR_MP_RQ_V_019				
SIP_CC_PR_MP_RQ_V_020				
SIP_CC_PR_MP_RQ_V_021				
SIP_CC_PR_MP_RQ_V_022				
SIP_CC_PR_MP_RQ_V_023				
SIP_CC_PR_MP_RQ_V_024				
SIP_CC_PR_MP_RQ_V_025				
SIP_CC_PR_MP_RQ_V_026				
SIP_CC_PR_MP_RQ_V_027				
SIP_CC_PR_MP_RQ_V_029				
SIP_CC_PR_MP_RQ_V_030				
SIP_CC_PR_MP_RQ_V_031				
SIP_CC_PR_MP_RQ_V_032				
SIP_CC_PR_MP_RQ_V_033				
SIP_CC_PR_MP_RQ_V_034				
SIP_CC_PR_MP_RQ_V_035				
SIP_CC_PR_MP_RQ_V_036				
SIP_CC_PR_MP_RQ_V_037				
SIP_CC_PR_MP_RQ_V_038				
SIP_CC_PR_MP_RQ_V_039				
SIP_CC_PR_MP_RQ_V_040				
SIP_CC_PR_MP_RQ_V_041				
SIP_CC_PR_MP_RQ_V_042				
SIP_CC_PR_MP_RQ_V_043				
SIP_CC_PR_MP_RQ_V_044				
SIP_CC_PR_MP_RQ_V_045				
SIP_CC_PR_MP_RQ_V_046				
SIP_CC_PR_MP_RQ_V_047				
SIP_CC_PR_MP_RQ_V_048				
SIP_CC_PR_MP_RQ_V_049				
SIP_CC_PR_MP_RQ_V_050				
SIP_CC_PR_MP_RQ_V_051				
SIP_CC_PR_MP_RQ_V_052				
SIP_CC_PR_MP_RQ_V_053				
SIP_CC_PR_MP_RQ_V_054				
SIP_CC_PR_MP_RQ_V_055				
SIP_CC_PR_MP_RQ_V_056				
SIP_CC_PR_MP_RQ_V_057				
SIP_CC_PR_MP_RQ_V_059				

ATS Reference	Selected?	Run?	Verdict	Observations (Reference to any observations made in clause C.8)
SIP_CC_PR_MP_RQ_V_060				
SIP_CC_PR_MP_RQ_V_061				
SIP_CC_PR_MP_RQ_V_062				
SIP_CC_PR_MP_RQ_V_063				
SIP_CC_PR_MP_RQ_V_064				
SIP_CC_PR_MP_RQ_V_065				
SIP_CC_PR_MP_RQ_V_066				
SIP_CC_PR_MP_RQ_V_067				
SIP_CC_PR_MP_RQ_I_001				
SIP_CC_PR_MP_RQ_I_002				
SIP_CC_PR_MP_RQ_I_003				
SIP_CC_PR_MP_RQ_I_004				
SIP_CC_PR_MP_RS_V_002				
SIP_CC_PR_MP_RS_V_003				
SIP_CC_PR_MP_RS_V_004				
SIP_CC_PR_MP_RS_V_005				
SIP_CC_PR_MP_RS_V_006				
SIP_CC_PR_MP_RS_V_007				
SIP_CC_PR_MP_RS_V_008				
SIP_CC_PR_MP_RS_V_009				
SIP_CC_PR_MP_RS_V_010				
SIP_CC_PR_MP_RS_V_011				
SIP_CC_PR_MP_RS_V_012				
SIP_CC_PR_MP_RS_V_013				
SIP_CC_PR_MP_RS_V_014				
SIP_CC_PR_MP_RS_V_015				
SIP_CC_PR_MP_RS_V_016				
SIP_CC_PR_MP_RS_V_017				
SIP_CC_PR_MP_RS_V_018				
SIP_CC_PR_MP_RS_V_019				
SIP_CC_PR_MP_RS_V_020				
SIP_CC_PR_MP_RS_V_021				
SIP_CC_PR_MP_RS_V_022				
SIP_CC_PR_MP_RS_V_023				
SIP_CC_PR_MP_RS_V_024				
SIP_CC_PR_MP_RS_V_025				
SIP_CC_PR_MP_RS_V_026				
SIP_CC_PR_MP_RS_V_027				
SIP_CC_PR_MP_RS_V_028				
SIP_CC_PR_MP_RS_V_029				
SIP_CC_PR_MP_RS_V_030				
SIP_CC_PR_MP_RS_V_031				
SIP_CC_PR_MP_RS_V_032				
SIP_CC_PR_MP_RS_V_033				
SIP_CC_PR_MP_RS_V_034				
SIP_CC_PR_MP_RS_V_035				
SIP_CC_PR_MP_RS_V_036				
SIP_CC_PR_MP_RS_V_037				
SIP_CC_PR_MP_RS_V_038				
SIP_CC_PR_MP_RS_V_039				
SIP_CC_PR_MP_RS_V_040				
SIP_CC_PR_MP_RS_V_041				
SIP_CC_PR_MP_RS_V_042				
SIP_CC_PR_MP_RS_V_043				
SIP_CC_PR_MP_RS_V_044				
SIP_CC_PR_MP_RS_V_045				
SIP_CC_PR_MP_RS_V_046				
SIP_CC_PR_MP_RS_V_047				
SIP_CC_PR_MP_RS_V_048				
SIP_CC_PR_MP_RS_V_049				

ATS Reference	Selected?	Run?	Verdict	Observations (Reference to any observations made in clause C.8)
SIP_CC_PR_MP_RS_V_050				
SIP_CC_PR_MP_RS_V_051				
SIP_CC_PR_MP_RS_V_052				
SIP_CC_PR_MP_RS_V_053				
SIP_CC_PR_TR_CL_V_001				
SIP_CC_PR_TR_CL_V_002				
SIP_CC_PR_TR_CL_V_003				
SIP_CC_PR_TR_CL_V_004				
SIP_CC_PR_TR_CL_V_005				
SIP_CC_PR_TR_CL_V_006				
SIP_CC_PR_TR_CL_V_007				
SIP_CC_PR_TR_CL_V_008				
SIP_CC_PR_TR_CL_V_009				
SIP_CC_PR_TR_CL_V_010				
SIP_CC_PR_TR_CL_V_011				
SIP_CC_PR_TR_CL_V_012				
SIP_CC_PR_TR_CL_V_013				
SIP_CC_PR_TR_CL_V_014				
SIP_CC_PR_TR_CL_TI_001				
SIP_CC_PR_TR_CL_TI_002				
SIP_CC_PR_TR_CL_TI_003				
SIP_CC_PR_TR_CL_TI_004				
SIP_CC_PR_TR_CL_TI_005				
SIP_CC_PR_TR_CL_TI_007				
SIP_CC_PR_TR_CL_TI_008				
SIP_CC_PR_TR_CL_TI_009				
SIP_CC_PR_TR_CL_TI_010				
SIP_CC_PR_TR_CL_TI_012				
SIP_CC_PR_TR_CL_TI_013				
SIP_CC_PR_TR_CL_TI_014				
SIP_CC_PR_TR_CL_TI_015				
SIP_CC_PR_TR_CL_TI_016				
SIP_CC_PR_TR_CL_TI_017				
SIP_CC_PR_TR_CL_TI_018				
SIP_CC_PR_TR_CL_TI_019				
SIP_CC_PR_TR_SE_V_001				
SIP_CC_PR_TR_SE_V_002				
SIP_CC_PR_TR_SE_V_003				
SIP_CC_PR_TR_SE_V_004				
SIP_CC_PR_TR_SE_V_005				
SIP_CC_PR_TR_SE_V_006				
SIP_CC_PR_TR_SE_V_007				
SIP_CC_PR_TR_SE_V_008				
SIP_CC_PR_TR_SE_V_009				
SIP_CC_PR_TR_SE_V_010				
SIP_CC_PR_TR_SE_V_011				
SIP_CC_PR_TR_SE_V_012				
SIP_CC_PR_TR_SE_V_013				
SIP_CC_PR_TR_SE_V_014				
SIP_CC_PR_TR_SE_V_015				
SIP_CC_PR_TR_SE_V_016				
SIP_CC_PR_TR_SE_V_017				
SIP_CC_PR_TR_SE_V_018				
SIP_CC_PR_TR_SE_V_019				
SIP_CC_PR_TR_SE_V_020				
SIP_CC_PR_TR_SE_V_021				
SIP_CC_PR_TR_SE_V_022				
SIP_CC_PR_TR_SE_V_023				
SIP_CC_PR_TR_SE_V_024				

ATS Reference	Selected?	Run?	Verdict	Observations (Reference to any observations made in clause C.8)
SIP_CC_PR_TR_SE_V_025				
SIP_CC_PR_TR_SE_V_026				
SIP_CC_PR_TR_SE_V_027				
SIP_CC_PR_TR_SE_V_028				
SIP_CC_PR_TR_SE_V_029				
SIP_CC_PR_TR_SE_TI_001				
SIP_CC_PR_TR_SE_TI_002				
SIP_CC_PR_TR_SE_TI_003				
SIP_CC_PR_TR_SE_TI_004				
SIP_CC_PR_TR_SE_TI_005				
SIP_CC_PR_TR_SE_TI_006				
SIP_CC_PR_TR_SE_TI_007				
SIP_CC_PR_TR_SE_TI_008				
SIP_CC_PR_TR_SE_TI_009				
SIP_CC_RD_CE_V_001				
SIP_CC_RD_CE_V_002				
SIP_CC_RD_CE_V_003				
SIP_CC_RD_CE_V_004				
SIP_CC_RD_CE_V_005				
SIP_CC_RD_CE_V_006				
SIP_CC_RD_CE_V_007				
SIP_CC_RD_CE_V_008				
SIP_CC_RD_CE_V_009				
SIP_CC_RD_CE_V_010				
SIP_CC_RD_CR_V_001				
SIP_CC_RD_CR_V_002				
SIP_CC_RD_CR_V_003				
SIP_QC_OE_V_001				
SIP_QC_OE_V_002				
SIP_QC_OE_V_003				
SIP_QC_OE_V_004				
SIP_QC_OE_V_005				
SIP_QC_OE_V_006				
SIP_QC_OE_V_007				
SIP_QC_OE_V_008				
SIP_QC_OE_V_009				
SIP_QC_OE_V_010				
SIP_QC_OE_V_011				
SIP_QC_OE_V_012				
SIP_QC_OE_V_013				
SIP_QC_OE_V_014				
SIP_QC_OE_V_015				
SIP_QC_OE_TI_001				
SIP_QC_OE_TI_002				
SIP_QC_OE_TI_003				
SIP_QC_OE_TI_004				
SIP_QC_OE_TI_005				
SIP_QC_OE_TI_006				
SIP_QC_OE_TI_007				
SIP_QC_TE_V_001				
SIP_QC_TE_V_002				
SIP_QC_TE_V_003				
SIP_QC_TE_V_004				
SIP_QC_TE_V_005				
SIP_QC_TE_V_006				
SIP_QC_TE_V_007				
SIP_QC_TE_V_008				
SIP_QC_TE_V_009				
SIP_QC_TE_V_010				
SIP_QC_PR_V_001				

ATS Reference	Selected?	Run?	Verdict	Observations (Reference to any observations made in clause C.8)
SIP_QC_PR_V_002				
SIP_QC_PR_V_003				
SIP_QC_PR_V_004				
SIP_QC_PR_V_005				
SIP_QC_PR_V_006				
SIP_QC_PR_V_007				
SIP_QC_PR_V_008				
SIP_QC_PR_V_009				
SIP_QC_PR_V_010				
SIP_QC_PR_V_011				
SIP_QC_PR_V_012				
SIP_QC_PR_V_013				
SIP_MG_RT_V_001				
SIP_MG_RT_V_002				
SIP_MG_RT_V_003				
SIP_MG_RT_V_004				
SIP_MG_RT_V_005				
SIP_MG_RT_V_006				
SIP_MG_RT_V_007				
SIP_MG_RT_V_008				
SIP_MG_RT_I_001				
SIP_MG_RT_I_002				
SIP_MG_RT_I_003				
SIP_MG_RT_I_004				
SIP_MG_RT_I_005				
SIP_MG_RR_V_001				
SIP_MG_RR_V_002				
SIP_MG_RR_V_003				
SIP_MG_RR_V_004				
SIP_MG_RR_V_005				
SIP_MG_RR_V_006				
SIP_MG_RR_V_007				
SIP_MG_RR_V_008				
SIP_MG_RR_V_009				
SIP_MG_RR_V_010				
SIP_MG_RR_V_011				
SIP_MG_RR_V_012				
SIP_MG_RR_V_013				
SIP_MG_RR_I_001				
SIP_MG_RR_I_002				
SIP_MG_RR_I_003				
SIP_MG_RR_I_004				
SIP_MG_RR_I_005				
SIP_MG_RR_I_006				
SIP_MG_OE_V_001				
SIP_MG_OE_V_002				
SIP_MG_OE_V_003				
SIP_MG_OE_V_004				
SIP_MG_OE_V_005				
SIP_MG_OE_V_006				
SIP_MG_OE_V_007				
SIP_MG_OE_V_008				
SIP_MG_OE_V_009				
SIP_MG_OE_V_010				
SIP_MG_OE_V_011				
SIP_MG_OE_V_012				
SIP_MG_OE_V_013				
SIP_MG_OE_V_014				
SIP_MG_OE_V_015				
SIP_MG_OE_I_001				

ATS Reference	Selected?	Run?	Verdict	Observations (Reference to any observations made in clause C.8)
SIP_MG_OE_I_002				
SIP_MG_OE_I_003				
SIP_MG_OE_I_004				
SIP_MG_OE_I_005				
SIP_MG_OE_I_006				
SIP_MG_OE_I_007				
SIP_MG_OE_I_008				
SIP_MG_OE_I_009				
SIP_MG_OE_I_010				
SIP_MG_TE_V_001				
SIP_MG_TE_V_002				
SIP_MG_TE_V_003				
SIP_MG_TE_V_004				
SIP_MG_TE_V_005				
SIP_MG_TE_V_006				
SIP_MG_TE_V_007				
SIP_MG_TE_V_008				
SIP_MG_TE_V_009				
SIP_MG_TE_V_010				
SIP_MG_TE_V_011				
SIP_MG_TE_V_012				
SIP_MG_TE_V_013				
SIP_MG_TE_V_014				
SIP_MG_TE_V_015				
SIP_MG_TE_I_001				
SIP_MG_TE_I_002				
SIP_MG_TE_I_003				
SIP_MG_TE_I_004				
SIP_MG_TE_I_005				
SIP_MG_TE_I_006				
SIP_MG_TE_I_007				
SIP_MG_TE_I_008				
SIP_MG_PR_V_001				
SIP_MG_PR_V_002				
SIP_MG_PR_V_003				
SIP_MG_PR_V_004				
SIP_MG_PR_V_005				
SIP_MG_PR_V_006				
SIP_MG_PR_V_007				
SIP_MG_PR_V_008				
SIP_MG_PR_V_009				
SIP_MG_PR_V_010				
SIP_MG_PR_V_011				
SIP_MG_PR_V_012				
SIP_MG_PR_V_013				
SIP_MG_PR_V_014				
SIP_MG_PR_V_015				
SIP_MG_PR_V_016				
SIP_MG_PR_I_001				
SIP_MG_PR_I_002				
SIP_MG_PR_I_003				
SIP_MG_PR_I_004				
SIP_MG_PR_I_005				
SIP_MG_PR_I_006				
SIP_MG_PR_I_007				
SIP_MG_PR_I_008				
SIP_MG_PR_I_009				
SIP_MG_PR_I_010				
SIP_MG_RD_V_001				

ATS Reference	Selected?	Run?	Verdict	Observations (Reference to any observations made in clause C.8)
SIP_MG_RD_V_002				
SIP_MG_RD_V_003				
SIP_MG_RD_V_004				
SIP_MG_RD_V_005				
SIP_MG_RD_V_006				
SIP_MG_RD_V_007				
SIP_MG_RD_V_008				
SIP_MG_RD_V_009				
SIP_MG_RD_V_010				
SIP_MG_RD_V_011				
SIP_MG_RD_V_012				
SIP_MG_RD_V_013				
SIP_MG_RD_V_014				
SIP_MG_RD_I_001				
SIP_MG_RD_I_002				
SIP_MG_RD_I_003				
SIP_MG_RD_I_004				
SIP_MG_RD_I_005				
SIP_MG_RD_I_006				
SIP_MG_RD_I_007				
SIP_MG_RD_I_008				

C.7 Void

C.8 Observations

Additional information relevant to the technical content of the PCTR is given here.

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

Annex D (informative): Bibliography

- ISO/IEC 9646-1: "Information technology - Open Systems Interconnection - Conformance testing methodology and framework - Part 1: General concepts".
- ISO/IEC 9646-2: "Information technology - Open Systems Interconnection - Conformance testing methodology and framework - Part 2: Abstract Test Suite specification".
- ISO/IEC 9646-3: "Information technology - Open Systems Interconnection - Conformance testing methodology and framework - Part 3: The Tree and Tabular Combined Notation (TTCN)".
- ISO/IEC 9646-7: "Information technology - Open Systems Interconnection - Conformance testing methodology and framework - Part 7: Implementation Conformance Statements".

History

Document history		
V3.1.1	November 2004	Publication
V3.2.1	July 2005	Publication
V4.1.1	July 2006	Publication
V4.2.5	January 2008	Publication