

SIP PROXY

Test Specification Reference

Philipp Haupt
Matthias Hürlimann

December 14, 2006

Contents

1	Test Specification Reference	3
1.1	Document Information	4
1.1.1	History	4
1.2	Introduction	5
1.2.1	Purpose	5
1.2.2	Scope	5
1.3	Reference Overview	6
1.4	Elements	8
1.4.1	TestCase [ROOT]	8
1.4.2	Request	8
1.4.3	MessageModifiers	9
1.4.4	ContentLength	9
1.4.5	LinePermutator	10
1.4.6	Content	10
1.4.7	Timeout	10
1.4.8	Response	11
1.4.9	Regex	11
1.4.10	Variables	12
1.4.11	Var	12
1.4.12	ResponseVariables	13
1.4.13	ResponseVar	13
1.4.14	ResponseCatcher	13
1.4.15	NumberCounter	14
1.4.16	NumberRangeFuzzer	14
1.4.17	StringMutationFuzzer	14
1.4.18	StringExpansionFuzzer	15
1.4.19	ExpansionString	16
1.4.20	Character Set	16
1.4.21	ValueListFuzzer	17
1.4.22	HexValueListFuzzer	17
1.4.23	Digest	18
1.4.24	ConfigValue	19
1.4.25	ClearText	20
1.5	Using Vars	21
1.5.1	Example	21
1.6	XMLSchema for validation	22

List of Figures

1.1	Test Specification Reference	7
1.2	Proxy Mode paramNames	19
1.3	Test Case Mode paramNames	20

Chapter 1

Test Specification Reference

1.1 Document Information

1.1.1 History

Date	Version	Author	Description
02.11.2006	1.00	PH	Created document
03.11.2006	1.01	PH	Described Test Specification Reference
05.11.2006	1.02	MH	Supplemented Test Specification Languages, Reference
06.11.2006	1.03	MH	Examples for Test Specification Reference
07.11.2006	1.04	PH	Review of document and built a new structure of elements
09.11.2006	1.05	PH	Described the Modules and reviewed document
13.11.2006	1.06	MH	Adopted time attributes
20.11.2006	1.07	MH	Test case, cycles attribute is optional
20.11.2006	1.08	PH	Test case, added new attribute repeatList-Value for the ValueListFuzzer module
21.11.2006	1.09	PH	Added a new feature for the StringMutationFuzzer (own character set)
22.11.2006	1.10	MH	Added a new feature for the StringExpansionFuzzer (own character set)
23.11.2006	1.11	PH	Added description of the XMLSchema which will be used for validation
27.11.2006	1.12	PH	Added ContentLength module
27.11.2006	1.13	PH	Added configuration variables
29.11.2006	1.14	PH	Added NumberCounter module
29.11.2006	1.15	PH	Optimized document structure, added short description of each module
04.12.2006	1.16	PH	Added LinePermutator module
08.12.2006	1.17	PH	Review and correction
12.12.2006	1.18	MH	Proofreading

1.2 Introduction

1.2.1 Purpose

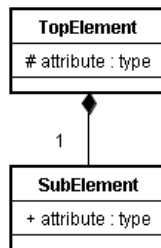
This document gives a quick overview about the all elements and attributes of a test case specification.

1.2.2 Scope

This document is valid over the whole project life time cycle.

1.3 Reference Overview

Figure 1.1 gives a quick overview about all elements and attributes that can be used within an XML test specification file. The following example will illustrate the general syntax usage:



```

<TopElement attribute="value">
  <SubElement attribute="value" />
</TopElement>
  
```

Attributes tagged with a '#' are mandatory.
Attributes tagged with a '+' are optional.

```

string  = "abc abc"
int     = Number between -2.147.483.648
              and +2.147.483.647
long    = Number between -9.223.372.036.854.775.808
              and +9.223.372.036.854.775.807
boolean = "true" or "false"
  
```

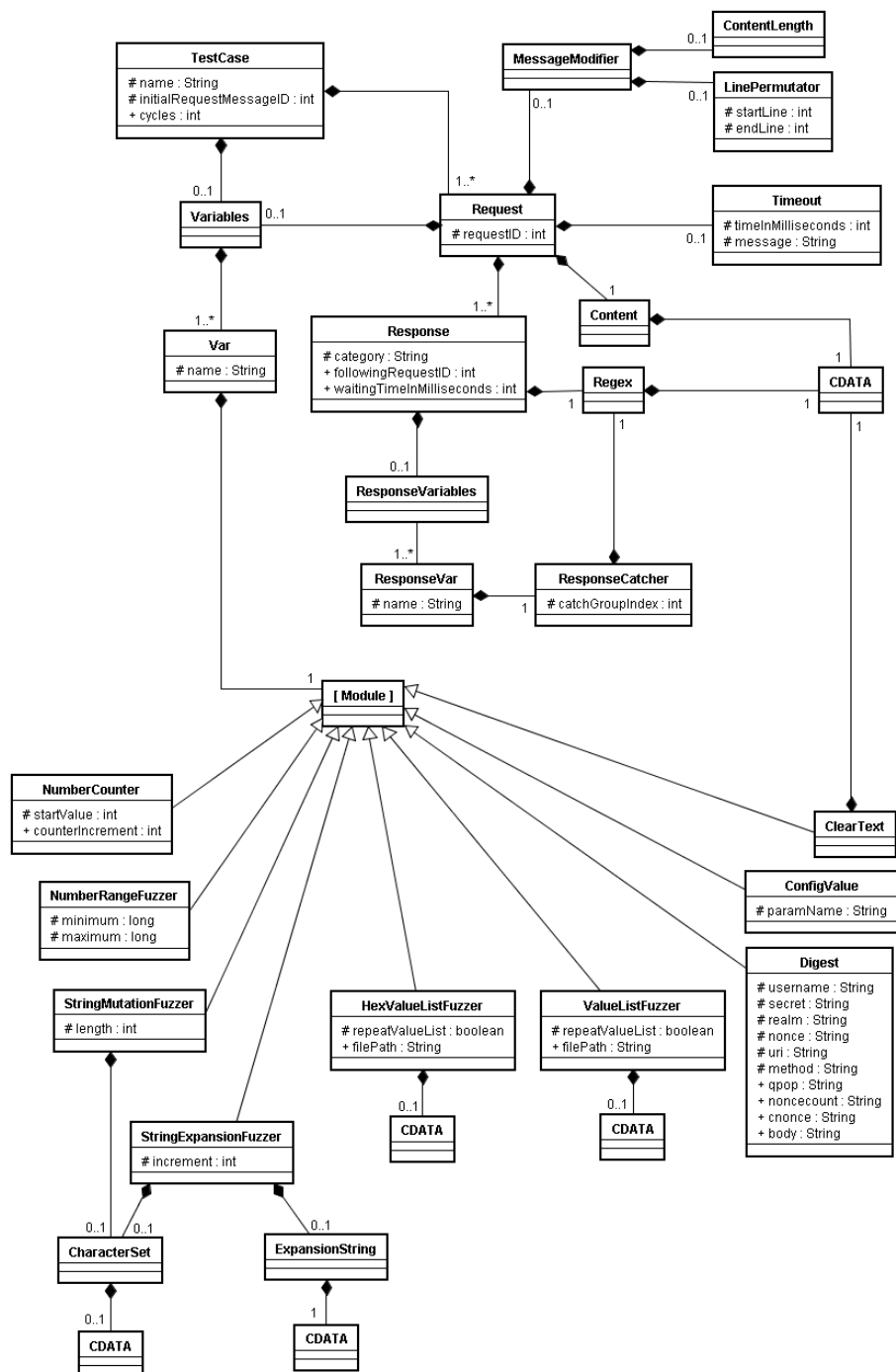


Figure 1.1: Test Specification Reference

1.4 Elements

1.4.1 TestCase [ROOT]

This element is the root element of every test case specification. Please make use of the XMLSchema, located in the /TestCase directory, to validate the file (refer to [1.6](#)).

Attributes

- name
Meaningful name of the test case.
- initialRequestMessageID
The ID of the initial request message to be sent
- cycles [optional]
Number of cycles to repeat the test case. Default value is infinite.

Subelements

- Request
- Variables

Example

```
<TestCase name="SampleTestCase" initialRequestMessageID="1" cycles="20"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="TestCaseSchema.xsd">
  ...
</TestCase>
```

1.4.2 Request

This will be a request message to be sent.

Attributes

- requestID
Unique ID of a request message (unique within test case / file)

Subelements

- Variables
- MessageModifier
- Content
- Timeout
- Response

Example

```
<Request requestID="1">
  <Variables>
    ...
  </Variables>
  <MessageModifier>
    ...
  </MessageModifier>
  <Content>
    ...
  </Content>
  <Timeout ... />
  <Response ...>
    ...
  </Response>
</Request>
```

1.4.3 MessageModifiers

Activates a specific module which will modify a request message before sending it to the target.

Subelements

- ContentLength
- LinePermutator

Example

```
<MessageModifier>
  ...
</MessageModifier>
```

1.4.4 ContentLength

Activates the ContentLength module which will provide a valid value for the Content-Length header

Example

```
<MessageModifier>
  <ContentLength/>
</MessageModifier>
```

1.4.5 LinePermutator

Activates the LinePermutator module which will shuffle the message lines within a certain line number range.

Attributes

- startLine
First line number which will be included in the shuffling list.
- endLine
Last line number which will be included in the shuffling list.

Example

```
<MessageModifier>  
  <LinePermutator startLine="2" endLine="5"/>  
</MessageModifier>
```

1.4.6 Content

Subelements

- CDATA (Text)
Content of the request message.

Example

```
<Content>  
  <![CDATA[INVITE sip:*98@192.168.0.12 SIP/2.0  
Via: SIP/2.0/UDP 192.168.0.12:5060  
...  
]]>  
</Content>
```

1.4.7 Timeout

Attributes

- timeInMilliseconds
Amount of time in millisecond within an expected response message has to arrive.
- message
Description of situation, if timeout occurs.

Example

```
<Timeout timeInMilliseconds="2000" message="PBX unavailable"/>
```

1.4.8 Response

Attributes

- **category**
Category of the response message which must be one of the following:
 - **okay**
No security flaw or bug has been found
 - **warning**
Security flaw or bug has been found
- **followingRequestID** [optional]
ID of request message which will be sent after this response message arrived.
- **waitingTimeInMilliseconds** [optional]
Time to wait in milliseconds until next Request message is being sent.

Subelements

- **Regex**
- **ResponseVariables**

Example

```
<Response category="okay" followingRequestID="3" waitingTimeInMilliseconds="500">  
  <Regex>  
    <![CDATA[SIP/2.0 (401|407)]]>  
  </Regex>  
</Response>
```

1.4.9 Regex

Regular expression.

Subelements

- **CDATA**
Regular expression

Example

```
<Regex>  
  <![CDATA[[(SIP/2.0 200)((.*) (\r\n))]]>  
</Regex>
```

1.4.10 Variables

Variables which can be addressed in a test case specification.

Subelements

- Var

Example

```
<Variables>
  <Var name="varName1">
    ...
  </Var>
  <Var name="varName2">
    ...
  </Var>
</Variables>
```

1.4.11 Var

Variable which can be addressed afterwards.

Attributes

- name
Name of the variable. This name can be used afterwards to address the variable

Subelements

- *Module*
A specific “module” which returns the value for the specified variable. The “module” itself is not an element, it represents an element of the type “module”.

Example

The “module” in this example is the ClearText-Module.

```
<Var name="varNameXY">
  <ClearText>
    <![CDATA[121212]]>
  </ClearText>
</Var>
```

1.4.12 ResponseVariables

Variables which can be addressed afterwards in subsequent elements.

Subelements

- ResponseVar

Example

```
<ResponseVariables>
  <ResponseVar name="...">
    ...
  </Var>
  <ResponseVar name="...">
    ...
  </Var>
</ResponseVariables>
```

1.4.13 ResponseVar

Variable which can be addressed afterwards.

Attributes

- name
Name of the response variable. This name can be used afterwards to address the variable

Subelements

- ResponseCatcher

Example

```
<ResponseVar name="nonceRegister">
  <ResponseCatcher catchGroupIndex="2">
    ...
  </ResponseCatcher>
</ResponseVar>
```

1.4.14 ResponseCatcher

A module which will catch specific information from a response message. The regular expression must be defined using capturing groups (please refer to the “**RegularExpression.pdf**” document).

Attributes

- catchGroupIndex
catchGroupIndex which provides the wanted value.

Subelements

- Regex

Example

```
<ResponseCatcher catchGroupIndex="2">
  <Regex>
    <![CDATA[(nonce=\\")(\\.*)\\")]]>
  </Regex>
</ResponseCatcher>
```

1.4.15 NumberCounter

Module which will provide a linear incrementing long value.

Attributes

- startValue
A start value which must be a long type.
- counterIncrement [optional]
Increment value which must be a long type. Default value is '1'.

Example

```
<NumberCounter startValue="1" counterIncrement="2"/>
```

1.4.16 NumberRangeFuzzer

Module which will provide a random long value within a specified number range.

Attributes

- minimum
Minimum value of a random long number.
- maximum
Maximum value of a random long number.

Example

```
<NumberRangeFuzzer minimum="0" maximum="1000"/>
```

1.4.17 StringMutationFuzzer

Module which will provide a random string of a specific length.

Attributes

- length
Length of the random string value.

Subelements

- `CharacterSet`
Individual character set to be used instead of default character set.

Example

```
<StringMutationFuzzer length="30"/>  
  
<StringMutationFuzzer length="30">  
  <CharacterSet>  
    ...  
  </CharacterSet>  
</StringMutationFuzzer>
```

1.4.18 StringExpansionFuzzer

Module which will provide an incremented string. If no expansion string is set, a random string will be computed.

Attributes

- `increment`
A string will be expanded by a multiple of the expansion string. If no expansion string is defined, a random string will be used which length is equal to the increment value.

Subelements

None or one of the following sub elements can be defined:

- `ExpansionString`
- `CharacterSet`
Only useful if no `ExpansionString` is defined

Examples

```
<StringExpansionFuzzer increment="5"/>  
  
<StringExpansionFuzzer increment="5">  
  <ExpansionString>  
    ...  
  </ExpansionString>  
</StringExpansionFuzzer>  
  
<StringExpansionFuzzer increment="5">  
  <CharacterSet>  
    ...  
  </CharacterSet>  
</StringExpansionFuzzer>
```


1.4.19 ExpansionString

Contains a CDATA element to define a specific expansion string.

Subelements

None or one of the following sub elements can be defined:

- CDATA
Specification of the string to be used for expansion.

Examples

```
<ExpansionString>
  <![CDATA[abc]]>
</ExpansionString>
```

1.4.20 Character Set

A individual character set can be defined.

Subelements

- CDATA
Specification of character set.

Character set specification

A character set can be defined with either hexadecimal or char values. Each value or value-range has to be separated by a comma. Refer to the following samples:

Escape Character = '\'

The characters '\'

For example: \\ and \,

DEFAULT Character Set: <![CDATA[a-z,A-Z,0-9]]>

Specification: <![CDATA[a-c,1-3,!,-,#,&]]>

Characters: 'a','b','c','1','2','3','!','-','#','&'

Specification: <![CDATA[X,! ,0,]]>

Characters: 'X','!','0',' ']

Specification: <![CDATA[\\,\,,-]]>

Characters: '\',' ','-'

Specification: <![\\,f-h]]>

Characters: '\','f','g','h'

Specification: <![0x41-0x44,0x23,0x26,0x45-0x47]]>

Characters: 'A','B','C','D','#','&','E','F','G'

Example

```
<CharacterSet>
  <![CDATA[a-c,1-3,!,#,&]]>
</CharacterSet>

<CharacterSet>
  <![0x41-0x44,0x23,0x26,0x45-0x47]]>
</CharacterSet>
```

1.4.21 ValueListFuzzer

Module which will provide a specific value from a list.

Attributes

- **filePath** [optional]
System Path to a external value list file (in case there is no CDATA value list defined).
- **repeatValueList**
If the value is “false” the module will return empty value after the last value has been used.
If the value is “true” the module will repeat the value list after the last value has been used.

Subelements

- **CDATA**
Value list (in case there is no filePath defined)

Example

```
<ValueListFuzzer filePath="TestCases/valuelist.txt" repeatValueList="true"/>

<ValueListFuzzer repeatValueList="true">
  <![CDATA[127.0.0.1
192.168.0.1
192.168.1.1
localhost.localdomain]]>
</ValueListFuzzer>
```

1.4.22 HexValueListFuzzer

Module which will provide a specific hexadecimal value from a list.

Attributes

- **filePath** [optional]
System Path to a external value list file (in case there is no CDATA value list defined). This file must contain any ASCII values in hex notation only.

- repeatValueList
If the value is “0” the module will return empty value after the last value has been used.
If the value is “1” the module will repeat the value list after the last value has been used.

Subelements

- CDATA
Value List (in case there is no filePath defined). This list must contain any ASCII values in hex notation only.

Example

```
<HexValueListFuzzer filePath="TestCases/valuelist.txt" repeatValueList="false"/>

<HexValueListFuzzer repeatValueList="true">
<![CDATA[0x5B
0x5C
0x5D
0x5E]]>
</HexValueListFuzzer>
```

1.4.23 Digest

Module which will compute a digest.

Attributes

- username
The username for a digest authentication.
- secret
The secret for a digest authentication.
- realm
The realm for a digest authentication.
- nonce
The nonce for a digest authentication.
- uri
The uri for a digest authentication.
- method
The method for a digest authentication.
- qop [optional]
The qop for a digest authentication.
- noncecount [optional]
The noncecount for a digest authentication.

- cnonce [optional]
The cnonce for a digest authentication.
- body [optional]
The body for a digest authentication.

Example

```
<Digest username="alice" secret="123456" realm="asterisk.biloxy.com"
  nonce="30419c6c" uri="sip:192.168.0.10" method="REGISTER"/>
```

1.4.24 ConfigValue

Module which will provide a specific configuration value which have been defined in the preferences window.

Attributes

- paramName
The paramName must be one of the following predefined names:

ProxySocketIP
ProxySocketPort
PbxIP
PbxPort
ClientIP
ClientPort
TestCaseSocketIP
TestCaseSocketPort
TargetIP
TargetPort

ParamName Overview

Figure 1.2 and 1.3 gives an overview for all possible configuration parameter names.

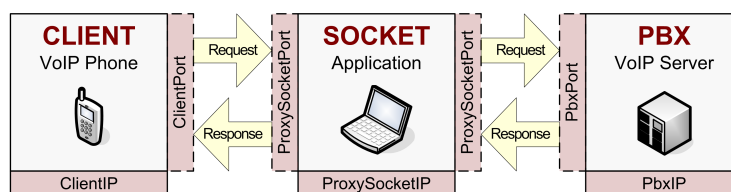
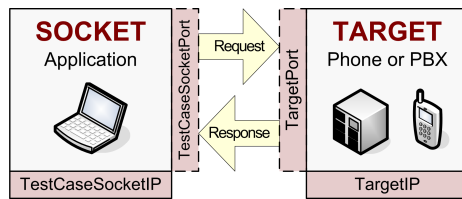


Figure 1.2: Proxy Mode paramNames

Example

```
<ConfigValue paramName="ProxySocketIP"/>
```

**Figure 1.3:** Test Case Mode paramNames

1.4.25 ClearText

Module which will provide a specific clear text value.

Subelements

- CDATA
Contains a value for any user defined variable

Example

```
<ClearText>  
  <![CDATA[abc]]>  
</ClearText>
```

1.5 Using Vars

Vars can be addressed with the following syntax:

```
/'varNameXY'/
```

1.5.1 Example

Call-ID: /'register_ID'/@biloxi.com

1.6 XMLSchema for validation

All Test Case Specification xml files must be well formed. Furthermore, they will be validated. The XMLSchema “**TestCaseSchema.xsd**” for the validation is located in the folder “**src/persistence**”. The XML Schema can be included in a test case specification file to validate the file with an external XML editor. To include the schema use the following attribute in the `<TestCase>`-element:

```
<TestCase initialRequestMessageID="1" name="Sample"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="..\src\persistence\TestCaseSchema.xsd">
```