ETSI ES 201 873-5 V4.8.1 (2017-05)

Methods for Testing and Specification (MTS);

The Testing and Test Control Notation version 3;

Part 5: TTCN-3 Runtime Interface (TRI)

**ETSI Standard**

Reference

RES/MTS-201873-5 T3ed481TRI

Keywords

interface, methodology, runtime, testing, TRI, TTCN-3

***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

***Important notice***

The present document can be downloaded from:  
<http://www.etsi.org/standards-search>

The present document may be made available in electronic versions and/or in print. The content of any electronic and/or print versions of the present document shall not be modified without the prior written authorization of ETSI. In case of any existing or perceived difference in contents between such versions and/or in print, the only prevailing document is the print of the Portable Document Format (PDF) version kept on a specific network drive within ETSI Secretariat.

Users of the present document should be aware that the document may be subject to revision or change of status. Information on the current status of this and other ETSI documents is available at <https://portal.etsi.org/TB/ETSIDeliverableStatus.aspx>

If you find errors in the present document, please send your comment to one of the following services:  
<https://portal.etsi.org/People/CommiteeSupportStaff.aspx>

***Copyright Notification***

No part may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm except as authorized by written permission of ETSI.  
The content of the PDF version shall not be modified without the written authorization of ETSI.  
The copyright and the foregoing restriction extend to reproduction in all media.

© European Telecommunications Standards Institute 2017.

All rights reserved.

**DECT**TM, **PLUGTESTS**TM, **UMTS**TM and the ETSI logo are Trade Marks of ETSI registered for the benefit of its Members.  
**3GPP**TM and **LTE**™ are Trade Marks of ETSI registered for the benefit of its Members and  
of the 3GPP Organizational Partners.  
**oneM2M** logo is protected for the benefit of its Members  
**GSM**® and the GSM logo are Trade Marks registered and owned by the GSM Association.

#### 6.3.2.5 TriMessageType

**TriMessageType** is mapped to the following interface:

// TRI IDL TriMessageType

package org.etsi.ttcn.tri;

public interface TriMessage {

public byte[] getEncodedMessage();

public void setEncodedMessage(byte[] message);

public int getNumberOfBits();

public void setNumberOfBits(int amount);

public java.io.InputStream getInputStream();

public void setInputStream(java.io.InputStream stream, int amountOfBits);

public boolean equals(TriMessage message);

}

**Methods:**

* getEncodedMessage  
  Returns the message encoded according the coding rules defined in the TTCN‑3 specification. In some cases, the message data are available only in the form of a stream and cannot be converted into a continuous byte array (e.g. because of memory restrictions). In such cases, this method returns null and the getInputStream method shall be used for reading the data.
* setEncodedMessage  
  Sets the encoded message representation of this TriMessage to message.
* getNumberOfBits  
  Returns the amount of bits of the message.
* setNumberOfBits  
  Sets the amount of bits in the message.
* getInputStream  
  Returns the input stream that can be used for reading message content. The method is usually used in case of large messages when putting all data into a continuous byte array would be impractical.
* setInputStream  
  Sets the input stream that can be used for reading message content. The method is usually used in case of large messages when putting all data into a continuous byte array would be impractical.
* equals  
  Compares message with this TriMessage for equality. Returns true if and only if both messages have the same encoded representation, false otherwise.

#### 8.5.2.8 TriMessage

A value of type TriMessage is encoded test data that either is to be sent to the SUT or has been received from the SUT. It is mapped to following pure virtual class:

class TriMessage {

public:

virtual ~TriMessage ();

virtual const Tbyte \*getData()const=0;

virtual void setData(const Tbyte \*str, Tsize bitLen)=0;

virtual Tsize getBitsDataLen()const=0;

virtual std:istream \* getInputStream()=0;

virtual void setInputStream(std:istream \* stream, Tsize bitLen)=0;

virtual Tboolean operator== (const TriMessage &msg) const =0;

virtual TriMessage \* cloneMessage () const =0;

virtual Tboolean operator< (const TriMessage &msg) const =0;

}

**Methods:**

* ~TriMessage

Destructor.

* getData

Gets binary string data (array of characters). In some cases, the message data are available only in the form of a stream and cannot be converted into a continuous byte array (e.g. because of memory restrictions). In such cases, this method returns null and the getInputStream method shall be used for reading the data.

* setData

Set the binary string data (array of characters).

* getBitsDataLen

Gets data length.

* getInputStream

Returns the input stream that can be used for reading message content. The method is usually used in case of large messages when putting all data into a continuous byte array would be impractical.

* setInputStream

Sets the input stream that can be used for reading message content. The method is usually used in case of large messages when putting all data into a continuous byte array would be impractical.

* operator==

Returns true if both TriMessage objects are equal.

* cloneMessage

Returns a copy of the TriMessage.

* operator<

Operator < overload.

#### 9.4.2.6 TriMessageType

**TriMessageType** is mapped to the following interface:

public interface ITriMessage {  
 byte [] EncodedMessage { get; set; }  
 int NumberOfBits { get; }  
 void SetEncodedMessage(byte[] data, int numberOfBits);

System.IO.Stream InputStream { get; }

void SetInputStream(System.IO.Stream stream, int numberOfBits);  
 bool Equals(ITriMessage msg);  
}

**Members:**

* EncodedMessage  
  Gets or sets the message encoded according the coding rules defined in the TTCN-3 specification. In case the message is set, the property assignment call produces the same result as calling the SetEncodedMessage method with the second parameter equal to byte array length \* 8. In some cases, the message data are available only in the form of a stream and cannot be converted into a continuous byte array (e.g. because of memory restrictions). In such cases, this method returns null and the InputStream property shall be used for reading the data
* NumberOfBits  
  Returns the amount of bits of the message.
* SetEncodedMessage  
  Sets the encoded message representation of this TriMessage to message. The number of bits has to be less or equal to data.Length \* 8.
* InputStream  
  Returns the input stream that can be used for reading message content. The method is usually used in case of large messages when putting all data into a continuous byte array would be impractical.
* SetInputStream  
  Sets the input stream that can be used for reading message content. The method is usually used in case of large messages when putting all data into a continuous byte array would be impractical.
* Equals  
  Compares a message with this TriMessage for equality. Returns true if and only if both messages have  
  the same encoded representation, false otherwise.