

IEEE Standard for Automatic Test Markup Language (ATML) Unit Under Test (UUT) Description

IEEE Standards Coordinating Committee 20

Sponsored by the
IEEE Standards Coordinating Committee 20 on
Test and Diagnosis for Electronic Systems

IEEE
3 Park Avenue
New York, NY 10016-5997
USA

IEEE Std 1671.3™-2017
(Revision of
IEEE Std 1671.3-2007)

IEEE Standard for Automatic Test Markup Language (ATML) Unit Under Test (UUT) Description

Sponsor

IEEE Standards Coordinating Committee 20 on Test and Diagnosis of Electronic Systems
of the
IEEE Standards Board

Approved 6 December 2017

IEEE-SA Standards Board

Abstract: An exchange format utilizing Extensible Markup Language (XML) for both the static description of unit under test (UUT) and the specific description of UUT instance information is defined in this standard.

Keywords: automatic test equipment (ATE), Automatic Test Markup Language (ATML), ATML Instance Document, automatic test system (ATS), IEEE 1671.3™, unit under test (UUT), XML schema

The Institute of Electrical and Electronics Engineers, Inc.
3 Park Avenue, New York, NY 10016-5997, USA

Copyright © 2018 by The Institute of Electrical and Electronics Engineers, Inc.
All rights reserved. Published 13 April 2018. Printed in the United States of America.

IEEE is a registered trademark in the U.S. Patent & Trademark Office, owned by The Institute of Electrical and Electronics Engineers, Incorporated.

PDF: ISBN 978-1-5044-4567-2 STD22925
Print: ISBN 978-1-5044-4568-9 STDPD22925

IEEE prohibits discrimination, harassment, and bullying.

For more information, visit <http://www.ieee.org/web/aboutus/whatis/policies/p9-26.html>.

No part of this publication may be reproduced in any form, in an electronic retrieval system or otherwise, without the prior written permission of the publisher.

Important Notices and Disclaimers Concerning IEEE Standards Documents

IEEE documents are made available for use subject to important notices and legal disclaimers. These notices and disclaimers, or a reference to this page, appear in all standards and may be found under the heading “Important Notices and Disclaimers Concerning IEEE Standards Documents.” They can also be obtained on request from IEEE or viewed at <http://standards.ieee.org/IPR/disclaimers.html>.

Notice and Disclaimer of Liability Concerning the Use of IEEE Standards Documents

IEEE Standards documents (standards, recommended practices, and guides), both full-use and trial-use, are developed within IEEE Societies and the Standards Coordinating Committees of the IEEE Standards Association (“IEEE-SA”) Standards Board. IEEE (“the Institute”) develops its standards through a consensus development process, approved by the American National Standards Institute (“ANSI”), which brings together volunteers representing varied viewpoints and interests to achieve the final product. IEEE Standards are documents developed through scientific, academic, and industry-based technical working groups. Volunteers in IEEE working groups are not necessarily members of the Institute and participate without compensation from IEEE. While IEEE administers the process and establishes rules to promote fairness in the consensus development process, IEEE does not independently evaluate, test, or verify the accuracy of any of the information or the soundness of any judgments contained in its standards.

IEEE Standards do not guarantee or ensure safety, security, health, or environmental protection, or ensure against interference with or from other devices or networks. Implementers and users of IEEE Standards documents are responsible for determining and complying with all appropriate safety, security, environmental, health, and interference protection practices and all applicable laws and regulations.

IEEE does not warrant or represent the accuracy or content of the material contained in its standards, and expressly disclaims all warranties (express, implied and statutory) not included in this or any other document relating to the standard, including, but not limited to, the warranties of: merchantability; fitness for a particular purpose; non-infringement; and quality, accuracy, effectiveness, currency, or completeness of material. In addition, IEEE disclaims any and all conditions relating to: results; and workmanlike effort. IEEE standards documents are supplied “AS IS” and “WITH ALL FAULTS.”

Use of an IEEE standard is wholly voluntary. The existence of an IEEE standard does not imply that there are no other ways to produce, test, measure, purchase, market, or provide other goods and services related to the scope of the IEEE standard. Furthermore, the viewpoint expressed at the time a standard is approved and issued is subject to change brought about through developments in the state of the art and comments received from users of the standard.

In publishing and making its standards available, IEEE is not suggesting or rendering professional or other services for, or on behalf of, any person or entity nor is IEEE undertaking to perform any duty owed by any other person or entity to another. Any person utilizing any IEEE Standards document, should rely upon his or her own independent judgment in the exercise of reasonable care in any given circumstances or, as appropriate, seek the advice of a competent professional in determining the appropriateness of a given IEEE standard.

IN NO EVENT SHALL IEEE BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO: PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE PUBLICATION, USE OF, OR RELIANCE UPON ANY STANDARD, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE AND REGARDLESS OF WHETHER SUCH DAMAGE WAS FORESEEABLE.

Translations

The IEEE consensus development process involves the review of documents in English only. In the event that an IEEE standard is translated, only the English version published by IEEE should be considered the approved IEEE standard.

Official statements

A statement, written or oral, that is not processed in accordance with the IEEE-SA Standards Board Operations Manual shall not be considered or inferred to be the official position of IEEE or any of its committees and shall not be considered to be, or be relied upon as, a formal position of IEEE. At lectures, symposia, seminars, or educational courses, an individual presenting information on IEEE standards shall make it clear that his or her views should be considered the personal views of that individual rather than the formal position of IEEE.

Comments on standards

Comments for revision of IEEE Standards documents are welcome from any interested party, regardless of membership affiliation with IEEE. However, IEEE does not provide consulting information or advice pertaining to IEEE Standards documents. Suggestions for changes in documents should be in the form of a proposed change of text, together with appropriate supporting comments. Since IEEE standards represent a consensus of concerned interests, it is important that any responses to comments and questions also receive the concurrence of a balance of interests. For this reason, IEEE and the members of its societies and Standards Coordinating Committees are not able to provide an instant response to comments or questions except in those cases where the matter has previously been addressed. For the same reason, IEEE does not respond to interpretation requests. Any person who would like to participate in revisions to an IEEE standard is welcome to join the relevant IEEE working group.

Comments on standards should be submitted to the following address:

Secretary, IEEE-SA Standards Board
445 Hoes Lane
Piscataway, NJ 08854 USA

Laws and regulations

Users of IEEE Standards documents should consult all applicable laws and regulations. Compliance with the provisions of any IEEE Standards document does not imply compliance to any applicable regulatory requirements. Implementers of the standard are responsible for observing or referring to the applicable regulatory requirements. IEEE does not, by the publication of its standards, intend to urge action that is not in compliance with applicable laws, and these documents may not be construed as doing so.

Copyrights

IEEE draft and approved standards are copyrighted by IEEE under U.S. and international copyright laws. They are made available by IEEE and are adopted for a wide variety of both public and private uses. These include both use, by reference, in laws and regulations, and use in private self-regulation, standardization, and the promotion of engineering practices and methods. By making these documents available for use and adoption by public authorities and private users, IEEE does not waive any rights in copyright to the documents.

Photocopies

Subject to payment of the appropriate fee, IEEE will grant users a limited, non-exclusive license to photocopy portions of any individual standard for company or organizational internal use or individual, non-commercial use only. To arrange for payment of licensing fees, please contact Copyright Clearance Center, Customer Service, 222 Rosewood Drive, Danvers, MA 01923 USA; +1 978 750 8400. Permission to photocopy portions of any individual standard for educational classroom use can also be obtained through the Copyright Clearance Center.

Updating of IEEE Standards documents

Users of IEEE Standards documents should be aware that these documents may be superseded at any time by the issuance of new editions or may be amended from time to time through the issuance of amendments, corrigenda, or errata. An official IEEE document at any point in time consists of the current edition of the document together with any amendments, corrigenda, or errata then in effect.

Every IEEE standard is subjected to review at least every ten years. When a document is more than ten years old and has not undergone a revision process, it is reasonable to conclude that its contents, although still of some value, do not wholly reflect the present state of the art. Users are cautioned to check to determine that they have the latest edition of any IEEE standard.

In order to determine whether a given document is the current edition and whether it has been amended through the issuance of amendments, corrigenda, or errata, visit the IEEE Xplore at <http://ieeexplore.ieee.org/> or contact IEEE at the address listed previously. For more information about the IEEE-SA or IEEE's standards development process, visit the IEEE-SA Website at <http://standards.ieee.org>.

Errata

Errata, if any, for all IEEE standards can be accessed on the IEEE-SA Website at the following URL: <http://standards.ieee.org/findstds/errata/index.html>. Users are encouraged to check this URL for errata periodically.

Patents

Attention is called to the possibility that implementation of this standard may require use of subject matter covered by patent rights. By publication of this standard, no position is taken by the IEEE with respect to the existence or validity of any patent rights in connection therewith. If a patent holder or patent applicant has filed a statement of assurance via an Accepted Letter of Assurance, then the statement is listed on the IEEE-SA Website at <http://standards.ieee.org/about/sasb/patcom/patents.html>. Letters of Assurance may indicate whether the Submitter is willing or unwilling to grant licenses under patent rights without compensation or under reasonable rates, with reasonable terms and conditions that are demonstrably free of any unfair discrimination to applicants desiring to obtain such licenses.

Essential Patent Claims may exist for which a Letter of Assurance has not been received. The IEEE is not responsible for identifying Essential Patent Claims for which a license may be required, for conducting inquiries into the legal validity or scope of Patents Claims, or determining whether any licensing terms or conditions provided in connection with submission of a Letter of Assurance, if any, or in any licensing agreements are reasonable or non-discriminatory. Users of this standard are expressly advised that determination of the validity of any patent rights, and the risk of infringement of such rights, is entirely their own responsibility. Further information may be obtained from the IEEE Standards Association.

Participants

At the time this IEEE standard was completed, the 1671.3 Working Group had the following membership:

Ion A. Neag, *Chair*

Malcom Brown
Chris Gorringer
Anand Jain

Teresa Lopes
Mukund Modi
Leslie Orlidge
Mike Seavey

John Stabler
Joseph Stanco
Ronald Taylor

The following members of the individual balloting committee voted on this standard. Balloters may have voted for approval, disapproval, or abstention.

W. Larry Adams Jr.
Malcom Brown
Juan Carreon
Chris Gorringer
Randall Groves
Werner Hoelzl
Bernard Homes
Noriyuki Ikeuchi

Anand Jain
Teresa Lopes
Mukund Modi
Ion A. Neag
Leslie Orlidge
Mike Seavey
Robert Spinner
Joseph Stanco

Michael Stora
Walter Struppler
Marcy Stutzman
Ronald Taylor
Louis Ungar
John Vergis
Oren Yuen
Daidi Zhong

When the IEEE-SA Standards Board approved this standard on 6 December 2017, it had the following membership:

Jean-Philippe Faure, *Chair*
Gary Hoffman, *Vice Chair*
John D. Kulick, *Past Chair*
Konstantinos Karachalios, *Secretary*

Chuck Adams
Masayuki Ariyoshi
Ted Burse
Stephen Dukes
Doug Edwards
J. Travis Griffith
Michael Janezic

Thomas Koshy
Joseph L. Koepfinger*
Kevin Lu
Daleep Mohla
Damir Novosel
Ronald C. Petersen
Annette D. Reilly

Robby Robson
Dorothy Stanley
Adrian Stephens
Mehmet Ulema
Phil Wennblom
Howard Wolfman
Yu Yuan

*Member Emeritus

Introduction

This introduction is not part of IEEE Std 1671.3-2017, IEEE Standard for Automatic Test Markup Language (ATML) Unit Under Test (UUT) Description.

This child, or “dot,” standard, also known as an ATML component standard, provides for the definition of the unit under test XML schemas, and contains references to XML instance document examples, both of which accompany this standard.

The XML schemas defined by this standard provide for the identification and definition of a unit under test (UUT).

Where appropriate, the XML schemas utilize and reference components of the ATML for Exchanging Automatic Test Equipment and Test Information via XML Standard (IEEE Std 1671) schema set. ATML’s XML schemas define the basic information required within any test application and provide a vehicle for formally defining the test environment by defining a class hierarchy corresponding to these basic information entities and provide several methods within each to enable basic operations to be performed on these entities. ATML component standards within the ATML framework define the particular requirements within the test environment.

Contents

1. Overview	9
1.1 Scope	9
1.2 Application	9
1.3 Conventions used within this document	10
2. Normative references.....	11
3. Definitions, acronyms, and abbreviations	12
3.1 Definitions	12
3.2 Acronyms and abbreviations	13
4. <i>UUTDescription</i> schema	14
4.1 Applicability	14
4.2 Describing UUT hierarchy	14
4.3 Using the <i>hc:HardwareItemDescription</i> type in the <i>UUTDescription</i> schema	14
4.4 Describing UUT digital serial buses.....	22
5. <i>UUTInstance</i> schema.....	22
5.1 Applicability	22
5.2 Describing UUT instance hierarchy	23
6. ATML UUT Description XML schema names and locations	23
7. ATML XML schema extensibility	24
8. Conformance	25
Annex A (normative) XML schemas	26
A.1 <i>UUTDescription</i> XML schema.....	26
A.2 <i>UUTInstance</i> XML schema	79
Annex B (informative) IEEE download website material associated with this document	88
Annex C (informative) Describing UUT serial digital buses	89
C.1 Describing serial buses.....	90
C.2 Describing serial bus nodes.....	92
C.3 Describing serial bus messages.....	94
Annex D (informative) User information and examples	99
D.1 Line-replaceable unit UUT	99
D.2 Circuit card assembly UUT	100
D.3 <i>UUTInstance</i>	101
D.4 Description of digital serial buses.....	101
Annex E (informative) Bibliography.....	102

IEEE Standard for Automatic Test Markup Language (ATML) Unit Under Test (UUT) Description

1. Overview

Automatic Test Markup Language (ATML) is a collection of IEEE standards and associated Extensible Markup Language (XML) schemas that allows automatic test system (ATS) and test information to be exchanged in a common format adhering to the XML specifications.

The ATML framework and the ATML family of standards have been developed and are maintained under the guidance of the IEEE Standards Coordinating Committee 20 (SCC20) to serve as a comprehensive environment for integrating design data, test strategies, test requirements, test procedures, test results management, and test system implementations, while allowing test program, test asset interoperability, and unit under test (UUT) data to be interchanged between heterogeneous systems.

This standard (as well as the XML schemas and XML instance document examples that accompany this standard) is intended to be used in identifying and documenting a UUT.

1.1 Scope

This standard defines an exchange format, utilizing Extensible Markup Language (XML), for both the static description of a UUT and the specific description of UUT instance information.

1.2 Application

1.2.1 Of this document

This standard provides for the identification of static characteristics of either a hardware or a software UUT. Characteristics are descriptive attributes relating to a UUT's form, fit, and function. This collection of characteristics defines either a class or type of UUT (as represented by the "UUTDescription.xsd" schema defined in Clause 4) or a specific UUT (as represented by the "UUTInstance.xsd" schema defined in Clause 5). Either collection of characteristics may be used for the purposes of developing test fixtures (holding and/or interface devices), interfacing (such as electrical cabling or optical interfaces), or defining a test configuration.

The information contained in XML documents conforming to this standard will be useful to:

- Test Program Set (TPS) developers
- TPS maintainers
- Automatic test equipment (ATE) system developers
- ATE system maintainers
- Instrument developers
- Developers of ATML-based tools and systems
- UUT developers and maintainers

1.2.2 Of the annexes

This document includes five annexes. Annex A specifies the XML Schemas defined in Annex B through Annex E are informative, and thus are provided strictly as information, for users, implementers, and maintainers of this document.

1.3 Conventions used within this document

1.3.1 General

In accordance with the *IEEE Standards Style Manual*, any schema examples will be shown in Courier font.¹ In cases where instance document examples are necessary to depict the use of a schema type or element, such examples will also be shown in Courier font. When the characters “...” appear in an example, it indicates that the example component is incomplete.

All simple types, complex types, attribute groups, and elements will be listed. Explanatory information will be provided, along with examples if additional clarification is needed. The explanatory information shall include information on the intended use of the elements and/or attributes where the name of the entity does not clearly indicate its intended use. For elements derived from another source type (e.g., an abstract type), only attributes that extend the source type shall be listed. Details regarding the base type shall be listed along with the base type.

In tables that describe XML elements, the column “Use” indicates the following occurrence constraints for each element:

- “required” indicates that the element shall appear exactly once
- “optional” indicates that the element may appear once or not at all
- “1..∞” indicates that the element shall appear at least once and may appear multiple times
- “0..∞” indicates that the element may appear multiple times, once, or not at all

The namespace prefixes “uut:” and “uuti:” identify that the type or attribute group is contained in an XML schema associated with this document. See Annex B for a description of where these XML schemas can be located.

¹ The numbers in brackets correspond to those of the bibliography in Annex E.

When referring to an attribute of an XML element, the convention of [element]@[attribute] shall be used. In cases where an attribute name is referred to with no associated element, the attribute name shall be enclosed in single quotes. Element and type names shall always be italicized when appearing in text.

This standard uses the vocabulary and definitions of relevant IEEE standards. In case of conflict of definitions, except for those portions quoted from standards, the following precedence shall be observed: 1) Clause 3 of this standard, and 2) The *IEEE Standards Dictionary Online*.²

1.3.2 Precedence

In the event of conflict between this document and a normatively referenced standard (see Clause 2), the normatively referenced standard, as it applies to the information being produced, shall take precedence.

The *UUTDescription* schema (“UUTDescription.xsd”) element, child element, and annotation information shall take precedence over the descriptive information contained in Clause 4.

The *UUTDescription* schema and the material contained in Clause 4 shall take precedence over the example information represented in Annex D.

The *UUTInstance* schema (“UUTInstance.xsd”) element, child element, and annotation information shall take precedence over the descriptive information contained in Clause 5.

The *UUTInstance* schema and the material contained in Clause 5 shall take precedence over the example information represented in Annex D.

1.3.3 Word usage

In accordance with the *IEEE Standards Style Manual*, the word “shall” is used to indicate mandatory requirements strictly to be followed in order to conform to the standard and from which no deviation is permitted (“shall” equals “is required to”). The use of the word “must” is used only to describe unavoidable situations. The use of the word “will” is only used in statements of fact.

The word “should” is used to indicate that among several possibilities one is recommended as particularly suitable, without mentioning or excluding others (“should” equals “is recommended that”).

The word “may” is used to indicate a course of action permissible within the limits of the standard (“may” equals “is permitted to”).

The word “can” is used for statements of possibility and capability (“can” equals “is able to”).

2. Normative references

The following referenced documents are indispensable for the application of this document (i.e., they must be understood and used, so each referenced document is cited in text and its relationship to this document is explained). For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments or corrigenda) applies.

IEEE Std 1671-2010™, IEEE Standard for Automatic Test Markup Language (ATML) for Exchanging Automatic Test Equipment and Test Information via XML.^{3, 4}

² *IEEE Standards Dictionary Online* is available at: <http://dictionary.ieee.org>.

3. Definitions, acronyms, and abbreviations

3.1 Definitions

For the purposes of this document, the following terms and definitions apply. The *IEEE Standards Dictionary Online* should be consulted for terms not defined in this clause. In the event a term is explicitly redefined, or defined in more detail in this standard, the definition included in this standard shall be normative.

abstract type: A declared type that can only be used by reference. Thus, in an instance document, any element of an abstract type must also be explicitly declared to be of an `xsi:[type]`.

attribute: A documenting characteristic of an entity.

automatic test equipment (ATE): A system providing a test capability for the automatic testing of one or more units under test (UUTs). The ATE system consists of a controller, test resource devices, and peripherals. The controller directs the testing process and interprets the results. The test resource devices provide stimuli, measurements, and physical interconnections. The peripherals, such as displays, keyboards, printers, mass storage, and so on, supply the necessary capability for information management.

Automatic Test Markup Language (ATML) instance document: An Extensible Markup Language (XML) instance document that conforms to an XML schema defined in an ATML component standard.

Automatic Test Markup Language (ATML) namespace: A collection of names; identified by a Uniform Resource Identifier (URI) reference.

automatic test system (ATS): Includes the automatic test equipment (ATE) as well as all support equipment, software, test program (TP), and adapters.

digital rights management: Access control technologies that may be utilized to impose limitations on the usage of digital content material that are not foreseen by the content provider. Within the context of Automatic Test Markup Language (ATML), the digital content materials are the associated XML schemas, and the content provider is the IEEE.

entity: Something that has a distinct separate instance.

error: In this standard, a status condition reported by the UUT indicating incorrect operation. An error can be the manifestation of a UUT failure, the result of an incorrect UUT input or control signal, etc. Errors can be reported through communication buses, visual or acoustical indicators, and so on.

Extensible Markup Language (XML) attribute: Name-value pair associated with an XML element.

Extensible Markup Language (XML) document: A data object that conforms to the XML requirements for being well formed. In addition, the data object is valid if it additionally conforms to semantic rules of the XML schema.

Extensible Markup Language (XML) element: A bounded component of the logical structure of an Extensible Markup Language (XML) document that has a type and that may have XML attributes and content.

³ The IEEE standards or products referred to in Clause 2 are trademarks owned by the Institute of Electrical and Electronics Engineers, Incorporated.

⁴ IEEE publications are available from the Institute of Electrical and Electronics Engineers (<http://standards.ieee.org/>).

Extensible Markup Language (XML) instance document: An Extensible Markup Language (XML) document that conforms to a particular XML schema.

Extensible Markup Language (XML) schema: The definition of a class of XML documents, which are typically expressed in terms of constraints on the structure and the content of documents of that class, above and beyond the basic syntax constraints imposed by XML itself.

failure: The loss of UUT's ability to perform a required function.

fault: A defect in a UUT component; for example, a short circuit, a broken wire, an out of tolerance characteristic, or an incorrect step, process, or data definition in the UUT's firmware.

fault isolation: The process of reducing the number of anomalies that constitute a diagnosis; identification of an anomaly or anomalies to a degree sufficient to undertake an appropriate corrective action.

framework: A collection of classes created specifically to serve the needs of an application area.

hardware unit under test (UUT): Physical equipment designed for the purpose of performing some function.

software unit under test (UUT): Software programs, procedures, data, and rules designed for the purpose of performing some function.

test: A set of stimuli, either applied or known, combined with a set of observed responses and criteria for comparing these responses to a known standard.

test program (TP): A program specifically intended for the testing of a unit under test (UUT).

Test Program Set (TPS): The complete set of hardware, software, and documentation needed to evaluate a unit under test (UUT) on a given test system.

unit under test (UUT): The entity to be tested. It may range from a simple component to a complete system.

well formed: Conforming to all of Extensible Markup Language's (XML's) syntax rules.

3.2 Acronyms and abbreviations

ATE	automatic test equipment
ATML	Automatic Test Markup Language
ATS	automatic test system
BCD	binary coded decimal
BIT	built-in test
ESD	electrostatic discharge
ISO	International Organization for Standardization
SCC20	Standards Coordinating Committee 20

TPS	Test Program Set
TSF	Test Signal Framework
UML	Unified Modeling Language
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
UTF-8	8-bit Unicode Transformation Format
UUT	unit under test
W3C	World Wide Web Consortium
XML	Extensible Markup Language

4. *UUTDescription* schema

4.1 Applicability

The *UUTDescription* schema, specified in A.1 of this document, provides a standard format for exchanging descriptive information for a model, class, or type of UUT. Information from an XML instance document that conforms to this schema shall apply to all UUTs with a specific model name or part number.

UUTDescription instance documents may be used for the purposes of developing test fixtures (holding and/or interface devices), interfacing (such as electrical cabling or optical interfaces), or defining a test configuration.

4.2 Describing UUT hierarchy

UUTDescription instance documents allow the specification of “child” components within a UUT through *uut:UUTDescription/hc:Components/hc:Component* elements. These elements describe other models, classes, or types of hardware or software items. If a “child” item is a UUT itself, then the *c:DescriptionDocumentReference* child element of *hc:Component* can be used to reference the corresponding *UUTDescription* instance document.

Similarly, *UUTDescription* instance documents allow the specification of a UUT’s “parent” components through *uut:UUTDescription/hc:ParentComponents/hc:Component* elements, which describe other models, classes, or types of hardware items. If a “parent” item is a UUT itself, then the *c:DescriptionDocumentReference* child element of *hc:Component* can be used to reference the corresponding *UUTDescription* instance document.

4.3 Using the *hc:HardwareItemDescription* type in the *UUTDescription* schema

The complex type *uut:UUTDescription* extends type *hc:HardwareItemDescription* defined in the XML schema *HardwareCommon* (file “HardwareCommon.xsd”). The type *hc:HardwareItemDescription*

supports the description of various hardware entities, including instruments, test stations, test adapters, and UUTs.

Because the *hc:HardwareItemDescription* type definition is generic, the use of elements and attributes inherited in from that type for the purpose of describing UUTs is subject to several additional requirements, described in the following subclauses.

4.3.1 Using existing *hc:HardwareItemDescription* elements and attributes

When UUT Components are described in the instance document, the attribute '*designator*' of their descendant *c:/Definition/c:/Identification* shall be used to specify the reference designator of the component (e.g., "R1"). The descendant *c:/Definition/c:/Identification/c:/ModelName* shall be used to specify the type of the component (e.g., "Resistor"). Refer to Annex D for examples of describing UUT components.

4.3.2 Assigning types defined in the *UUTDescription* schema to *hc:HardwareItemDescription* elements

As the complex type *uut:UUTDescription* extends type *hc:HardwareItemDescription*, some elements of the base type were found to be insufficient for describing UUTs. To overcome this limitation, the *UUTDescription* schema defines new complex types that can be assigned to elements within *hc:HardwareItemDescription*. These complex types extend types defined in the *Common* or *HardwareCommon* schemas.

The Common-derived types that can be assigned to elements within *hc:HardwareItemDescription* are shown in the Unified Modeling Language (UML) static structure diagram in Figure 1. Note that the classes contained in this diagram are not complete representations of the corresponding XML schema types; they include only the attributes necessary for describing relationships to other classes from the diagram.

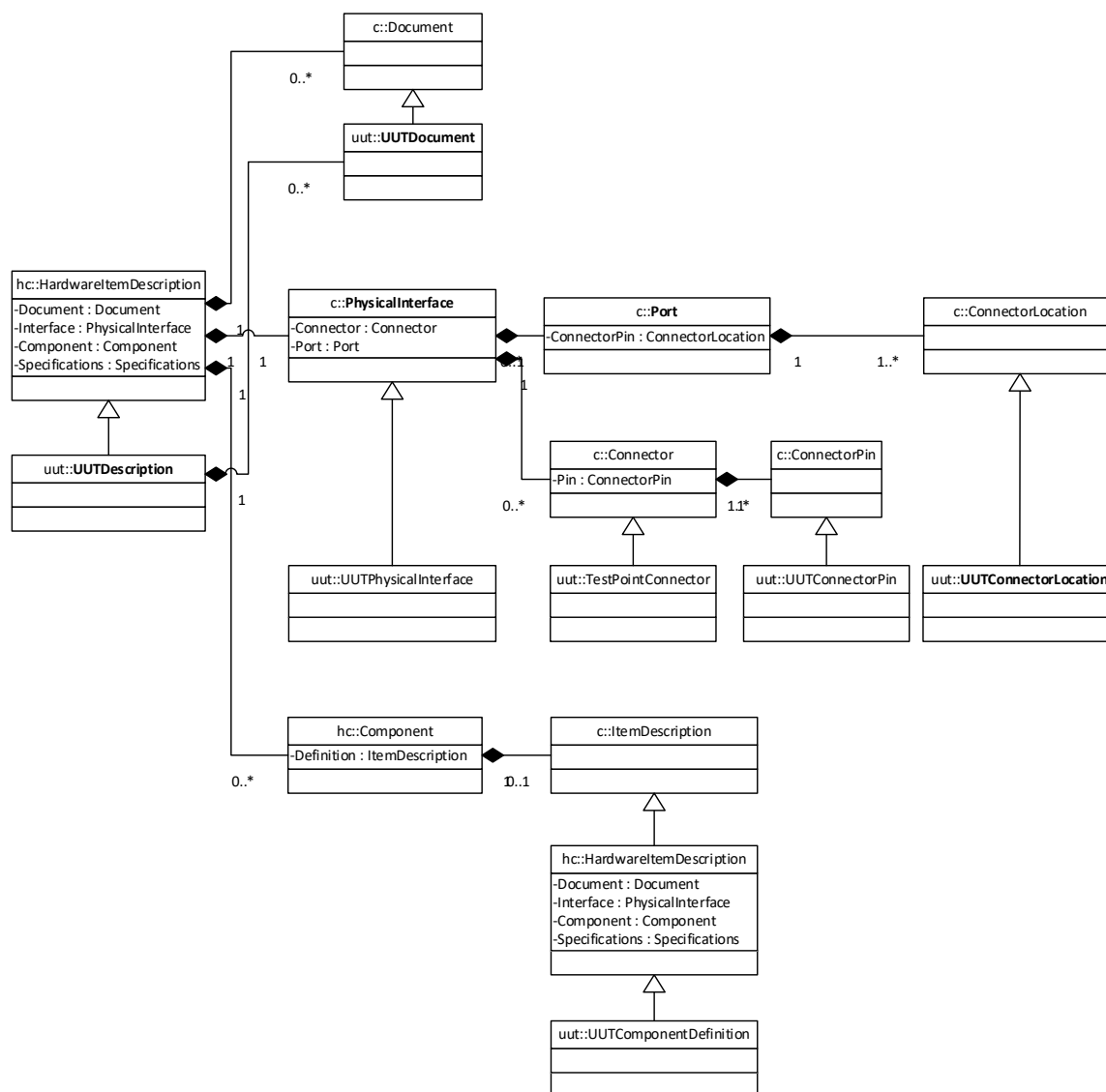


Figure 1—*UUTDescription* types that extend Common types

The following rules shall apply when assigning the Common-derived types to elements of the type *hc:HardwareItemDescription*:

- a) The type *uut:UUTDocument* (derived from *c:Document*) shall be assigned to all *hc:Documentation/hc:Document* descendants.
- b) The type *uut:UUTPhysicalInterface* (derived from *c:PhysicalInterface*) shall be assigned to the *hc:Interface* child element when it is necessary to describe the performance characteristics of the UUT.
- c) The type *uut:TestPointConnector* (derived from *c:Connector*) shall be assigned to *hc:Interface/c:Connectors/c:Connector* descendants when it is necessary to describe UUT test points.
- d) The type *uut:UUTConnectorPin* (derived from *c:ConnectorLocation*) shall be assigned to the *hc:Interface/c:Connectors/c:Connector/c:Pins/c:Pin* descendants when it is necessary to describe a cable recommended by the manufacturer for external connections to the UUT.

- e) The type *uut:UUTConnectorLocation* (derived from *c:ConnectorLocation*) shall be assigned to the *hc:Interface/c:Ports/c:Port/c:ConnectorPins/c:ConnectorPin* descendants when it is necessary to describe the role or function of the pin within a port. This type can be used, for example, to specify the functions of pins that belong to a TIA/EIA-232 serial port as RxD, TxD, CTS, RTS, etc.
- f) The type *uut:UUTComponentDefinition* (derived from *c:ItemDescription*) shall be assigned to the *hc:Components/hc:Component* descendants when it is necessary to describe the connectors or pins of a UUT component. See Annex D for examples of describing UUT components.

4.3.2.1 Describing UUT documentation

As indicated in 4.3.2, the type *uut:UUTDocument* (derived from *c:Document*) shall be assigned to all *hc:Document* child elements of *hc:Documentation*. The attribute ‘category’ of *uut:UUTDocument* shall be used to identify the category to which each document belongs.

The supported categories, described in simple type *UUTDocumentCategory*, are as follows:

- UUT Theory of Operation
- UUT Operating Instructions
- UUT Test Strategy Report
- UUT Failure Modes Effects Criticality Analysis (FMECA) Report
- UUT Configuration Information (outline drawings)
- UUT Installation Drawings (main drawings)
- UUT Assembly Drawings (detail and subassembly)
- UUT Module Sub-Assembly Schematics; referenced schematics shall show the circuitry contained within each module and subassembly, including the reference designators of each part and test point(s). Schematics can also include the amplitude and frequency of each input and output.
- UUT Specification Control Drawings (for nonstandard parts)
- UUT Wiring Diagrams; referenced diagrams shall show the actual physical wiring arrangement. Wire Run Lists or tables can supplement the diagrams.
- Detailed UUT Logic Diagrams; referenced diagrams shall conform to IEEE Std 91/91a™-1984 [B3].
- UUT Functional Block Diagrams; referenced diagrams shall identify parameters and associated signal tolerances at the external and internal interfaces of the UUT.
- UUT Parts Lists; referenced documents shall list the component part(s) of each assembly or subassembly. Components shall be listed by reference designator and manufacturer’s part number.
- Printed Circuit Board (PCB) Layouts. Referenced documents shall specify the location of components, junctions, terminals, etc.

- UUT Family Tree; referenced documents shall identify each functional module and subassembly. This information is utilized to enable proper part number and configuration accountability of all UUT subassemblies and modules requiring test requirement definition.
- UUT Test Requirements Document (TRD)
- Other document category (not included in the above list)

Note that the type of *hc:Document* elements, *c:Document*, has the required attribute 'uuid'. When referencing documents that are not ATML instance documents and not otherwise identified by a UUID, the value of this attribute shall be the null GUID, i.e., {00000000-0000-0000-0000-000000000000}.

4.3.2.2 Describing UUT test points

As indicated in 4.3.2, the type *uut:TestPointConnector* can be assigned to *uut:UUTDescription* descendants *hc:Interface/c:Connectors/c:Connector* to describe UUT test points.

For test points located on physical connectors, *c:Connector* elements are already in place to describe these connectors. Type *uut:TestPointConnector* shall be assigned to the elements. For each test point, one *c:Pins/c:Pin* descendant shall be created under the appropriate *c:Connector* element.

For test points that are not located on physical connectors, “virtual” connectors shall be used as containers, as follows:

- a) One or more *c:Connector* elements shall be created to represent the “virtual” connector, for example:
 - 1) A single *c:Connector* element could be used. In this case the value of the 'location' attribute (inherited from the base type and restricted to the enumerated values "Front" and "Back") is irrelevant. Any of these enumerated values can be assigned.
 - 2) Alternatively, one *c:Connector* element could hold the test points located on the front side of the printed circuit board and another *c:Connector* element could hold the test points located on the back side. For each element, the 'location' attribute will indicate the side.
- b) Because the “virtual” connector does not have a physical correspondent, some of the XML attributes and elements are not applicable. Because these attributes and elements are required in the *Common* schema, an arbitrary string (for example "NA") shall be assigned to them. This requirement applies to the following items:
 - 1) Attribute 'type' of element *c:Connector*
 - 2) Attribute 'designator' of child element *c:Identification*
 - 3) Descendant element *c:Identification/c:ModelName*
- c) For each test point, one *c:Pins/c:Pin* descendant shall be created under the appropriate *c:Connector* element.

Ports are generally required for referencing test points from other elements of the instance document. The *uut:UUTDescription/hc:Interface/c:Ports/c:Port* elements describing these ports shall reference the previously created Connector and Pin elements. The following guidelines are used:

- One *c:Port* can reference an individual test point. This approach can be used for test points accessed through manual probing.
- Alternatively, one *c:Port* can reference multiple test points. This approach is suitable for test points connected to an IEEE 1149.1 [B5] (JTAG) bus.

4.3.2.3 Describing UUT connector pins

While types *uut:UUTConnectorPin* and *uut:UUTConnectorLocation* both provide descriptive information for pins, they serve different purposes. Because the types are assigned to different elements within the *hc:HardwareItemDescription* type, instance documents can use either one of the types, or both. See Annex D for an example of describing UUT connectors and pins.

4.3.2.4 Describing UUT components

UUTs typically contain multiple components, which could be, for example, discrete electronic parts or sub-assemblies. The description of UUT components in *UUTDescription* instance documents is optional. A description is generally needed to support the following use cases:

- a) Use cases where components and ports are referenced from the current instance document or from external ATML instance documents:
 - 1) Referencing components when describing adjust operations. Adjust operations are described in *TestDescription* instance documents (which reference *UUTDescription* instance documents or contain embedded *UUTDescription* content). (See IEEE Std 1671.1TM, [B9].)
 - 2) Referencing component ports when describing faults. Faults can be described in *UUTDescription* instance documents, or in separate *TestDescription* instance documents.
 - 3) Referencing component ports when describing signal connections. Signal connections are described in *TestDescription* instance documents.
- b) Use cases where component and port information is required by software applications that consume the XML, for example:
 - 1) Applications that generate UUT parts lists, for example to support ordering of replacement components for repair. To support this use case, the component descriptions must contain part number and manufacturer information.
 - 2) Applications that perform diagnostic analysis (e.g., FMECA).
 - 3) Applications that display component failure information in a graphical format.

When there is a need to describe UUT components without pin information, the *uut:UUTDescription/hc:Components/hc:Component* elements shall have their original type, *c:ItemDescription*.

When there is a need to describe UUT components and their pins, the type *uut:UUTComponentDefinition* shall be assigned to the *uut:UUTDescription/hc:Components/hc:Component* elements.

NOTE—In most cases the pins of UUT components are referenced indirectly through ports.⁵

For additional information regarding the use of *c:Port*, *c:Connector*, and *c:Pin* elements refer to Annex E of IEEE Std 1671-2010, Pins, ports, connectors, and wire lists in ATML.⁶

4.3.3 Describing UUT firmware or UUT drivers

The *hc:Control/hc:Firmwares* element in type *hc:HardwareItemDescription* allows the specification of UUT firmware. The maximum or minimum version of each firmware component shall be specified using the ‘*version*’ and ‘*qualifier*’ attributes of the *hc:Firmware* descendant.

In some applications the firmware versions that can be loaded in the UUT are determined by the firmware versions loaded in the parent component. Describe these dependencies as follows:

- a) The type *hc:HardwareItemDescription* shall be assigned to element *uut:UUTDescription/hc:ParentComponents/c:Component/c:Definition*. The attributes of the *hc:Control/hc:Firmwares/hc:Firmware* element shall be used to specify the compatible versions of the parent component’s firmware.
- b) The ‘*name*’ attribute of the *hc:Control/hc:Firmwares/hc:Firmware* element shall be used to identify in free-form text the parent component firmware versions compatible with a particular UUT firmware version.

The *hc:Control/hc:Drivers* element in type *hc:HardwareItemDescription* allows the specification of UUT drivers. The maximum or minimum version of each driver component shall be specified using the ‘*version*’ and ‘*qualifier*’ attributes of the *hc:Driver* descendant.

4.3.4 Describing UUT errors, faults, and failures

The elements *uut:UUTDescription/uut:Faults/uut:Fault* and *uut:UUTDescription/uut:Failures/uut:Failure* describe conditions present in the UUT, which cause incorrect operation or performance degradation. Faults and failures are defined as follows:

- A fault is a condition that causes a component of the UUT to fail to perform in a required manner (e.g., broken wire, short circuit, value out of tolerance).
- A failure is a change in operating characteristics of the UUT resulting in degradation of its performance or the loss of its ability to perform a required function. A UUT failure is the manifestation of a fault in a UUT component.

The purpose of automatic testing is to detect and isolate faults and failures.

The *uut:UUTDescription/hc:Errors/hc>Error* elements describe errors and other status conditions reported by the UUT during normal operation or during testing. For example, errors could be numeric codes transmitted by the UUT, through a MIL-STD-1553B bus, to another unit located within the next-higher level of assembly. As another example, errors could be text messages transmitted by the UUT, through a TIA/EIA-422 bus, while the unit is being tested; these messages are received and interpreted by the ATE.

⁵ Notes in text, tables, and figures of a standard are given for information only and do not contain requirements needed to implement this standard.

⁶ Information on references can be found in Clause 2.

Note that errors may be indications of faults or failures within the UUT, or indications of problems external to the UUT (for example, “Ethernet bus not connected”). Furthermore, not all faults and failures result in errors being reported by the UUT.

The following rules shall apply when describing the errors, faults, failures, and their relationships:

- a) *hc:Error* elements (placed under the *uut:UUTDescription/hc:Errors* container) shall be used to describe errors and other status conditions reported by the UUT. A unique identifying string shall be assigned to attribute ‘*ID*’. Attribute ‘*type*’ shall specify the severity of the status condition, for example: *Warning*, *Error*, or *Fatal*.
- b) *uut:Fault* elements (placed under the *uut:UUTDescription/hc:Faults* container) shall be used to describe UUT faults.
- c) *uut:Failure* elements (placed under the *uut:UUTDescription/hc:Failure* container) shall be used to describe UUT failures.
- d) *uut:Fault/uut:Errors/uut:Error* elements shall be used to indicate the error conditions generated by the UUT when a fault occurs. The ‘*ID*’ of the referenced *hc:Error* element shall be assigned to attribute ‘*errorID*’ of *hc:Fault*.
- e) *uut:Failure/uut:Errors/uut:Error* elements shall be used to describe the error conditions generated by the UUT when a failure occurs. The ‘*ID*’ of the referenced *hc:Error* element shall be assigned to attribute ‘*errorID*’ of *hc:Failure*.

Refer to Annex D for examples of describing UUT errors and their relationship to faults and failures.

4.3.5 Describing software UUTs

The complex type *uut:UUTDescription* can be used to describe software UUTs such as instrument drivers or computer firmware:

- a) Required descendant *c:Identification/c:ModelName* shall contain the name of the software component.
- b) Optional descendants *c:Identification/c:IdentificationNumbers* can contain part numbers or model numbers for the software UUT.
- c) Required child element *c:Interface* is not relevant in the context of a software UUT. This element can contain placeholder data, for example a port named “N/A”.
- d) Where applicable, the child element *hc:Components* can contain or reference the description of “child” software components. For example, the child components of a software UUT representing a firmware program can be individual modules of that program.
- e) Where applicable, the child element *hc:ParentComponents* can contain or reference the description of “parent” hardware or software components. For example, the parent components of a software UUT representing a firmware program can be the hardware UUTs where the firmware runs.
- f) Where applicable, the child element *hc:SoftwareEntryPoints* can reference the callable entities of the software UUT, such as functions or class methods.
- g) The descendants *uut:UUTDescription/uut:Failures/uut:Failure* shall describe possible software defects.

- h) The child element *uut:UUTDescription/uut:Faults* shall be absent. Faults are not applicable to software UUTs.

4.3.6 Describing UUT warnings

In the context of the *UUTDescription* schema, elements *uut:UUTDescription/uut:Warnings/uut:Warning* shall be used to document hazard or warning information related to operation of the UUT, such as electrostatic discharge (ESD) warnings or high voltage warnings. This information can appear on warning labels attached to the physical UUT, in UUT documentation, etc.

Note that the warnings referenced above are distinct from the status conditions reported by the UUT, which could have a severity level of “Warning”. For describing status conditions reported by the UUT, use *hc:Error* elements as described in 4.3.4.

4.3.7 Describing UUT signal specifications

Elements of type *hc:Specifications* shall be used to describe UUT signal specifications, as follows:

- a) Elements *uut:UUTPhysicalInterface/uut:PerformanceCharacteristics/uut:Inputs/uut:Input/uut:Specifications* for UUT specifications applicable to input signals
- b) Elements *uut:UUTPhysicalInterface/uut:PerformanceCharacteristics/uut:Outputs/uut:Output/uut:Specifications* for UUT specifications applicable to output signals
- c) Elements *uut:UUTPhysicalInterface/uut:PerformanceCharacteristics/uut:Controls/uut:Control/uut:Specifications* for UUT specifications applicable to control signals
- d) Elements *uut:UUTConnectorPin/uut:Specifications* for UUT specifications applicable to signals at individual UUT connector pins

4.4 Describing UUT digital serial buses

When UUTs contain digital serial buses, element *uut:UUTDescription/uut:BusDescription* shall be used to describe the structure, connectivity, and operation of these buses. Refer to Annex C for detailed information on using the *UUTDescription* schema to describe UUT digital serial buses.

5. *UUTInstance* schema

5.1 Applicability

The *UUTInstance* schema, specified in A.2 of this document, provides a standard format for exchanging descriptive information for a specific UUT (i.e., a UUT “instance”). Information from an XML instance document that conforms to this schema shall apply to a single UUT, typically identified by a unique serial number.

In addition to recording the serial number, *UUTInstance* instance documents can be used to track time-dependent information applicable to the UUT instance, such as: manufacture date, current calibration state, total number of power-on cycles that the UUT has been submitted to over its lifetime, etc.

If a *UUTDescription* instance document exists for the UUT model, then the *uuti:UUTInstance/c:DescriptionDocumentReference* element can be used to reference that document.

Note that the distinction between a class of UUTs and a UUT instance is irrelevant for software UUTs, because in general all copies of a software program behave identically when tested. Consequently, the *UUTInstance* XML schema does not support the description of “software UUT instances”.

5.2 Describing UUT instance hierarchy

UUTInstance instance documents allow the identification of “child” components within a UUT instance through *uuti:UUTInstance/c:Components/c:Component* elements. These elements identify other item instances. If a “child” item is a UUT itself, then the *c:InstanceDocumentReference* child element of *c:Component* can be used to reference the corresponding *UUTInstance* instance document.

Similarly, *UUTInstance* instance documents allow the identification of a UUT instance’s “parent” component through the *uuti:UUTInstance/c:ParentComponent* element. This element identifies another item instance. If the “parent” instance is a UUT itself, then the *c:DescriptionDocumentReference* child elements of *c:ParentComponent* can be used to reference the corresponding *UUTInstance* instance document.

6. ATML UUT Description XML schema names and locations

The IEEE provides a download website for material published in association with published IEEE standards, presented in machine-friendly format. This material is digital rights management restricted use material. The ATML family of standards utilizes this download website to allow easy accessibility to all of the ATML family XML schemas (and in some cases, example XML instance documents). As depicted by Figure 2, the IEEE download website (<http://standards.ieee.org/downloads/>) contains several folders, each folder labeled by an associated IEEE standards number (e.g., IEEE 1671 series standards are in the 1671 folder). Each folder under this “base” IEEE standard number contains the material (XML schemas, etc.) for that ATML family component standard. ATML family component standards are identified by their IEEE 1671 series “dot” standard number and the year in which that standard was published by the IEEE.

NOTE 1—Standards that are revised will contain a folder for the year in which the standard is “reissued.” Both folders (for each year the standard was published) will be present on the IEEE download website.

NOTE 2—Folders for a particular standard are not available until the standard is published by the IEEE, and providing the standard has associated material that is to be made available via the download website.

Figure 2 depicts a portion of the entire IEEE download Web site, as it pertains to the UUT Description ATML family standard.

The IEEE SCC20 ATML website (<http://grouper.ieee.org/groups/scc20/tii>) provides access to material not yet published as an IEEE standard. This material is also digital rights management restricted use material. The ATML family of standards utilize this site to allow easy accessibility to any of the ATML family XML schemas (and in some cases, example XML instance documents) not yet approved by the IEEE Standards Board.

<http://standards.ieee.org/downloads/>

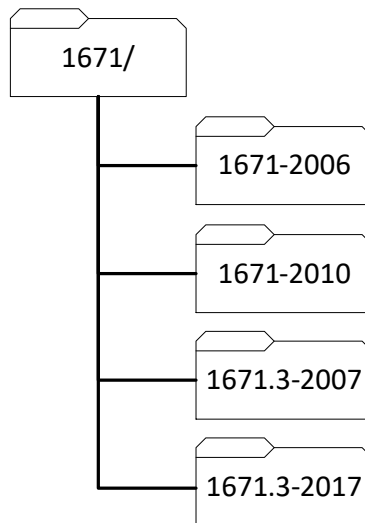


Figure 2—ATML UUT Description related IEEE download website structure

The UUT Description ATML family component standard, where the component is defined, their associated XML schemas names, and the IEEE download website folder name (where the XML schemas shall be located), is as defined in Table 1.

Table 1—ATML family XML schema name and folder location

Component	Defined in clause	XML schema name	IEEE download website folder (see Figure 2)
<i>UUTDescription</i>	4	UUTDescription.xsd	1671.3-2017
<i>UUTInstance</i>	5	UUTInstance.xsd	1671.3-2017

The XML schema identified in Table 2 includes ATML common elements. The ATML common element (e.g., component), where the component is defined, the associated XML schemas name, and the IEEE download website folder name (where the XML schema shall be located).

Table 2—ATML Common element XML schema name and location

Component	Location defined in IEEE Std 1671-2010	XML schema name	IEEE download website folder (see Figure 2)
<i>Common</i>	Annex B.1	Common.xsd	1671-2010
<i>HardwareCommon</i>	Annex B.2	HardwareCommon.xsd	1671-2010

7. ATML XML schema extensibility

The provision of an extension mechanism is necessary to ensure the viability of the specification and allow producers and consumers of ATML instance documents to interoperate in those cases where there is a requirement to exchange relevant data that are not included in the *UUTDescription* and *UUTInstance* XML schemas. The use of the extensions shall be done in a way that ensures that a conformant consumer can utilize the extended instance document without error, discard, or otherwise sidestep the extended data, and use the non-extended portions of the data as they are intended, without error or loss of functionality.

Extensions shall be additional information added to the content model of the element being extended.

Extensions shall not repackage existing information entities that are already supported by the *UUTDescription* and *UUTInstance* XML schemas.

An extended instance document shall be accompanied by the extension XML schema and documentation sufficient to explain the need for the extension as well as by the underlying semantics and relationship(s) to the base *UUTDescription* or *UUTInstance* schema.

The *UUTDescription* and *UUTInstance* XML schemas allow for two forms of extension, as follows:

- Wildcard-based extensions allow for the extension of the *UUTDescription* and *UUTInstance* XML schemas with additional elements.
- Type derivation allows for extending the set of data types by deriving a new type from an existing common element type.

8. Conformance

The minimal expectations for *UUTDescription* conformant XML instance documents shall be that a completely populated instance document is considered valid if said document complies with the following:

- The constraints expressed in the *UUTDescription* schema associated with this standard
- Any constraints imposed by inherited elements from Annex B of IEEE Std 1671-2010

The minimal expectations for *UUTInstance* conformant XML instance documents shall be that a completely populated instance document is considered valid if said document complies with the following:

- The constraints expressed in the *UUTInstance* schema associated with this standard
- Any constraints imposed by inherited elements from Annex B of IEEE Std 1671-2010

Extensions are permitted to both the *UUTDescription* and the *UUTInstance* schemas, but they shall only occur through the facility of the extensibility mechanism described in Clause 7. In short, extensions may only appear in the specific *<Extension>* tags provided in the associated XML schema. As defined in the W3C XML schema standard, an extended schema must conform to the W3C XML schema specification and shall not contain any entities defined in the base schema.

Annex A

(normative)

XML schemas

Should the reader not have a general understanding of XML schemas or XML terminology, there are several XML schema tutorials available for reference. The *XML Schema Part 0: Primer* [B13], the *XML Schema Tutorial* [B16], and the *XML Schema Tutorial, Part 1* [B17] are three available on the World Wide Web. These tutorials will help with the understanding of the contents of both this annex and the ATML *Common* schemas (Annex B of IEEE Std 1671-2010), of which this annex makes reference.

Prefixes utilized in this annex are as follows:

- The prefix “*uut:*” represents that the type or attribute group is defined in the “UUTDescription.xsd” XML schema associated with this standard.
- The prefix “*uuti:*” represents that the type or attribute group is defined in the “UUTInstance.xsd” XML schema associated with this standard.
- The prefix “*c:*” represents that the type or attribute group is defined in the “Common.xsd” XML schema associated with IEEE Std 1671-2010.
- The prefix “*hc:*” represents that the type or attribute group is defined in the “HardwareCommon.xsd” XML schema associated with IEEE Std 1671-2010.

A.1 UUTDescription XML schema

This subclause describes the *UUTDescription* XML schema, defined in file “UUTDescription.xsd”.

A.1.1 element UUTDescription

type	extension of <u>uut:UUTDescription</u>			
children	c:Description c:Identification c:Extension hc:Interface hc:Components hc:ParentComponents hc:Control hc:Documentation hc:ConfigurationOptions hc:FactoryDefaults hc:CalibrationRequirements hc:OperationalRequirements hc:EnvironmentalRequirements hc:PowerRequirements hc:PhysicalCharacteristics hc:Errors hc:NetworkList hc:LegalDocuments <u>uut:TsfLibraries uut:BusDescription uut:Specifications uut:Capabilities uut:SoftwareEntryPoints</u> <u>uut:Warnings uut:Failures uut:Faults</u>			
attributes	Name	Type	Use	Description
	version	c:NonBlankString	optional	A string designating the version of the described item.
	name	c:NonBlankString	optional	A descriptive or common name for the described item.
	uuid	c:Uuid	required	A universal unique identifier for the element containing this attribute.

	classified		xs:boolean	optional	An indication that the element is or is not classified.
	securityClassification		c:NonBlankString	optional	A use-case determined string declaring the security classification level of the element containing this attribute and the subordinate branch of the XML document.
identity constraints		Name	Refer	Selector	Field(s)
	unique	errorIdUnique		../hc:Errors	@ID
	unique	warningIdUnique		../uut:Warning	@ID
	key	tsfLibraryKey		../uut:TsfLibrary	@ID
	keyref	tsfLibraryKeyRef	<u>uut:tsfLibraryKey</u>	../uut:Protocol/uut:TsfClasses/uut:TsfClass	@tsfLibraryID
	key	componentKey		../hc:Component	@ID
	keyref	componentKeyRef	<u>uut:componentKey</u>	../uut:Node	@componentID
annotation	The UUTDescription XML schema defines the static information associated with a model, class, or type of UUT. Information from an XML instance document that conforms to this schema shall apply to all UUTs with a specific model name or part number.				

A.1.2 complexType Bus

type	extension of hc:Item			
children	hc:Description hc:Extension <u>uut:Protocols</u> <u>uut:Nodes</u> <u>uut:Ports</u> <u>uut:Attributes</u>			
used by	element <u>UUTDescription/BusDescription/Buses/Bus</u>			
attributes	Name	Type	Use	Description
	name	c:NonBlankString	required	A descriptive or common name for the hardware item.
	ID	c:NonBlankString	required	Unique identifier, enables referencing the bus. The value of this attribute shall be unique across all Bus elements defined in the instance document.
annotation	Shall describe one serial digital bus of the UUT.			

A.1.3 element Bus/Protocols

children	<u>uut:Protocol</u>
annotation	Shall reference the applicable specifications for all the protocols supported by the UUT on the current serial digital bus.

A.1.4 element Bus/Protocols/Protocol

attributes	Name	Type	Use	Description
	protocolDescriptionID	c:NonBlankString	required	Identifies the protocol. Shall contain the ID of a ProtocolDescription element.
annotation	Shall reference the specifications of one protocol used by the UUT.			

A.1.5 element Bus/Nodes

children	<u>uut:Node</u>
annotation	Shall contain the list of all UUT nodes of the digital bus.

A.1.6 element Bus/Nodes/Node

type	<u>uut:Node</u>			
children	hc:Description hc:Extension <u>uut:Attributes</u>			
attributes	Name	Type	Use	Description
	name	c:NonBlankString	required	A descriptive or common name for the hardware item.
	ID	c:NonBlankString	required	Unique identifier, enables referencing the node. The value of this attribute shall be unique across all Node elements defined under the ancestor Bus element.
	nodeDescriptionID	c:NonBlankString	required	Identifies the type of node. Shall contain the ID of a NodeDescription element.
	componentID	c:NonBlankString	optional	Shall identify the UUT component where the bus node resides. If present, this attribute shall contain the ID attribute of a Component defined for the UUT.
	role	<u>uut:NodeRole</u>	optional	Shall specify the node role, i.e., the direction of data transfers between the current node and other nodes located on the same bus.
	active	xs:boolean	optional	Shall indicate if the current node is active.
annotation	Shall describe one node of the digital bus.			

A.1.7 element Bus/Ports

children	uut:Port
annotation	Shall reference the ports where the bus is connected to the UUT interface.

A.1.8 element Bus/Ports/Port

type	hc:Network			
children	hc:Node hc:Description hc:Extension			
attributes	Name	Type	Use	Description
	baseIndex	xs:int	optional	Starting index for the items.
	count	xs:int	optional	Number fo identical items
	incrementBy	xs:int	optional	Specifies the value to increment by when calculating the value with which the replacement character is replaced. Allows for defining items where paired items have different characteristics. Defaults to 1.
	replacementCharacter	c:NonBlankString	optional	Character replaced with calculated index when repeated items are expanded.
annotation	Each /hc:Node/hc:Path descendant shall reference one port the where the bus is connected to the UUT interface.			

A.1.9 element Bus/Attributes

children	uut:Attribute				
identity constraints		Name	Refer	Selector	Field(s)
	unique	nameUnique2		uut:Attribute	@name
annotation	Shall contain the list of all bus attributes. The attributes included in this list can be specific to the bus protocol. For example, the bit rate of an ARINC429 bus can be specified through a bus attribute.				

A.1.10 element Bus/Attributes/Attribute

type	c:NamedValue			
children	c:Datum c:Collection c:IndexedArray			
attributes	Name	Type	Use	Description
	name	c:NonBlankString	required	A descriptive or common name for the subject data value.
annotation	Shall describe one bus attribute. The 'name' attribute shall uniquely identify each attribute within the parent collector element.			

A.1.11 complexType Failure

type	extension of uut:FaultFailureType			
children	<u>uut:Description</u> <u>uut:FailureRate</u> <u>uut:Location</u> <u>uut:Errors</u> <u>uut:Repair</u> <u>uut:DetectionIsolationComment</u> <u>uut:Extension</u>			
used by	element <u>UUTDescription/Failures/Failure</u>			
attributes	Name	Type	Use	Description
	insertable	xs:boolean	optional	Shall indicate if the fault or failure is insertable.
	severity	<u>uut:SeverityLevel</u>	optional	Shall describe the severity of the fault or failure. Shall be one of: Warning, Error, Fatal. Note that this attribute supports a fixed set of severity levels. The Extension child element shall be used to specify severity levels that do not belong to that set.
	source	c:NonBlankString	optional	Shall describe the source of the fault or failure through free-form text.
	name	c:NonBlankString	required	A descriptive or common name for the failure.
	type	<u>uut:FailureType</u>	required	Shall specify the type of the failure, as one of: Detectable, Non-Detectable, Non-Functional.
annotation	Shall describe a possible failure of the UUT. A failure is a change in operating characteristics of the UUT resulting in degradation of its performance. The Failures defined for the UUT typically correspond to the prime failure modes of all inputs and outputs to each next lower assembly of the UUT.			

A.1.12 complexType Fault

type	extension of uut:FaultFailureType			
children	<u>uut:Description</u> <u>uut:FailureRate</u> <u>uut:Location</u> <u>uut:Errors</u> <u>uut:Repair</u> <u>uut:DetectionIsolationComment</u> <u>uut:Extension</u> <u>uut:FailureMode</u>			
used by	element <u>UUTDescription/Faults/Fault</u>			
attributes	Name	Type	Use	Description
	insertable	xs:boolean	optional	Shall indicate if the fault or failure is insertable.
	severity	<u>uut:SeverityLevel</u>	optional	Shall describe the severity of the fault or failure. Shall be one of: Warning, Error, Fatal. Note that this attribute supports a fixed set of severity levels. The Extension child element shall be used to specify severity levels that do not belong to that set.
	source	c:NonBlankString	optional	Shall describe the source of the fault or failure through free-form text.
	componentID		required	ID of the Component for which the current Fault is defined.
	detectability	<u>uut:Detectability</u>	required	Detectability of the fault. Shall be one of: DetectableInsertable, DetectableDestructive, NonDetectable.
annotation	Shall describe a possible fault of a component within the UUT. A fault is a condition that causes the component to fail to perform in a required manner. (e.g., broken wire, short circuit, value out of tolerance, etc.). Note that Faults can be defined only for UUTs for which Components are also defined.			

A.1.13 element Fault/FailureMode

type	c:NonBlankString
annotation	Shall specify the failure mode, for example: Open, Short, Stuck at 0, Stuck at 1, Out of Tolerance.

A.1.14 complexType FaultFailureType

children	<u>uut:Description</u> <u>uut:FailureRate</u> <u>uut:Location</u> <u>uut:Errors</u> <u>uut:Repair</u> <u>uut:DetectionIsolationComment</u> <u>uut:Extension</u>			
used by	complexType <u>Failure Fault</u>			
attributes	Name	Type	Use	Description
	insertable	xs:boolean	optional	Shall indicate if the fault or failure is insertable.