文章编号:1671-4598(2019)02-0118-05 DOI:10.16526/j.cnki.11—4762/tp.2019.02.027 中图分类号:TP277 文献标识码:A

ATML 测试系统软件平台架构及测试结果建模研究

杨 起,王竹林,王 成
(陆军工程大学石家庄校区 导弹工程系,石家庄 050003)

摘要:ATML 是现今测试领域的国际通用标准;在深入学习了 ATML 标准的基础上,对 ATML 标准进行简要概述;设计了测试系统软件平台的基本架构,为测试系统的开发提供了依据;为了实现测试结果信息的共享性及交互性,对其中测试结果的描述方法进行了研究,并依据面向对象的思想,建立了测试结果的标准化 XML Schema;开发了相应的自动配置工具,该工具可实现测试结果模型自动配置,使得测试信息按统一的格式和规范的内容进行交互。

关键词:ATML;测试结果;XML Schema;配置工具

Research on ATML Test System Software Platform Architecture and Test Result Modeling

Yang Qi, Wang Zhulin, Wang Cheng
(Department of Missile Engineering, Army Engineering University, Shijiazhuang 050003, China)

Abstract: ATML is an international common standard in today's test field. Based on an in-depth study of the ATML standard, a brief overview of the ATML standard and a description of test results in the standard are conducted, the basic architecture of the test system software platform is designed to provide a basis for the development of the test system. in order to achieve the sharing and interactivity of the test result information, the description method of test results in the standard was studied, and the standardized XML Schema of the test results was established based on the object-oriented thinking; a corresponding automatic configuration tool was developed to automatically configure the standardized description of the test results, so that the test information was exchanged according to a uniform format and the content of the specification, It brings great convenience to the testing process and lays a good foundation for the sharing and reuse of test information.

Keywords: ATML; test result; XML Schema; configuration tool

0 引言

随着电子技术、计算机技术的不断进步发展,对 ATS(Automatic Test System,自动测试系统)越来越高。传统的 ATS 软件平台是以仪器为核心,存在很大的局限性,主要在于仪器的互换性差和 TPS(Test Program Sets,测试程序集)不可移植两个方面,极大阻碍了 ATS 的通用化发展。而面向信号的测试软件平台是当前 ATS 发展的主流,它以信号为核心,整个测试过程实质上就是信号的激励与响应过程,这种以信号来描述测试需求和仪器能力的思想为提高 TPS 可移植能力以及仪器互换能力提供了可能。构建通用的 ATS 软件平台,需要对多种格式测试诊断信息以及 ATS 中各种接口进行标准化、统一化,而 ATML(Automatic Test Markup Language,自动测试标记语言)的提出,为这一问题提供了方法。ATML 标准使用 XML(eXtention Marked Language,可扩展标记语言)来实现测试系统中测试信息的标准化描述,通过建立符合 ATML 标准的文档,可实现测试结果、测试仪器功能、测试策略需求、信号的特征信息、诊断信息以及被测对象的规格、需求等信息共享、互换,进而实现测试系统软件平台的开发[1-2]。

1 ATML 标准概述

ATML 标准作为自动测试系统中的标准交换媒介,提供了测试系统中信息交换的标准数据方式,实现了对所有测试信息的支持。ATML 标准体系中又划分为 9 个子标准,表 1 列出了标准集合以及 Schema 定义。

ATML 标准定义了构建一个 ATS 需要的大部分的要素,提供了测试信息交换和共享的框架结构及详细内涵,它定义的各个组件标准构成了一次测试执行的过程中需要共享和交换的测试信息的全集,提供了获得测试和测试程序的静态信息以及实时的测试信息的机制。因此,ATML 标准的应用将使 ATS 实现以下目标:

支持测试信息的流动;支持整个 TPS 生命周期;支持 TPS 的移植和复用;支持 ATML 的开发工具;支持动态测试序列以充分利用历史数据[3-4]。

收稿日期:2018-05-28; **修回日期**:2018-08-03。

作者简介:杨 起(1994-),男,黑龙江海林人,硕士研究生,主要从事武器系统自动测试与故障诊断研究。

2　基于 ATML 的测试系统软件平台架构

2.1　软件平台总体设计说明

测试系统软件主要包括计算机操作系统、测试程序开发和管理环境、测试程序运行环境，以及在测试程序开发环境中开发、在测试程序运行环境中执行的信号模型、系统模型、测试资源控制程序、自检校准测试程序等。测试系统软件支持 ATML 测试程序的开发和执行[5-7]。

表 1　窗口长度为 N 的不同窗函数性能对比

组件名称	XML Schema
公共部分	Common. xsd
测试描述	TestDescription. xsd
仪器描述	InstrumentDescription. xsd
被测对象	UUTDescription. xsd
测试适配器	TestAdaptor. xsd
测试平台	TestStation. xsd
测试配置	TestConfiguration. xsd
测试结果	TestResults. xsd
故障诊断	FaultTreeModel. xsd
	BayesNetworkModel. xsd
	D－MatrixInferenceModel. xsd
	DiagnosticLogicModel. xsd

测试系统软件结合系统硬件设计采用分布式结构，进行系统功能分解后实现管理、测试功能双分离，管理和测试功能之间通过组件接口进行数据交互，支持软件模块化、可重用。

开发/管理功能部分包括系统管理平台、TPS 开发平台和综合数据库，实现系统资源配置、操作权限管理、数据模型管理、仪器驱动管理、TPS 管理、TPS 开发、TPS 移植、TPS 导入/导出、测试数据管理、故障诊断、数据信息化应用、系统备份/恢复等功能，提供测试系统工作状态以及综合数据等信息。

测试运行功能部分主要为 TPS 运行平台，调度各类系统资源实现 TPS 的运行、模拟运行和调试等功能，满足测试对象的功能检查、性能测试、设备调教和历程记录等需求。

测试系统软件依据 ATML 标准建立软件架构、数据和信号模型，并将其纳入综合数据库进行管理及配置，通过系统管理平台、TPS 开发平台和 TPS 运行平台实现 TPS 开发和运行过程的规范化、通用化和标准化。

测试系统软件基于 IEEE－1641 标准定义信号，基于 IEEE－1671 和 IEEE－1636 等系列标准进行测试信息及结果的描述和应用，通过 TPS 开发平台匹配综合数据库内的数据模型形成被测对象 TPS，由系统管理平台统一管理和调度，TPS 运行平台具体负责 TPS 的运行支持和交互。

测试系统软件建立开放式、标准化的软件体系结构，其管理和测试功能各组成部分均为独立功能软件。

综合数据库为测试系统软件综合数据源。

系统管理平台是综合数据库的管理和应用前端，提供测试系统软件管理功能的人机接口，负责测试系统所有硬件、软件及数据的管理和配置功能。

TPS 开发平台挂接于系统管理平台，本地访问综合数据库，实现 TPS 的开发和移植。

TPS 运行平台远程访问综合数据库，提供测试系统软件测试功能的人机接口，负责被测对象 TPS 的测试运行和调试功能。

2.2　TPS 开发平台

TPS 开发平台基于 ATML 标准，构建面向信号的通用 TPS 开发平台，支持 TPS 移植。TPS 开发平台采用开放式结构，由一组用于 TPS 开发和辅助的功能软件集组成，同时形成 TPS 开发规范用于指导和约束 TPS 开发[8-10]。

TPS 开发平台分为建模层、开发层和应用层，如图 1 所示。

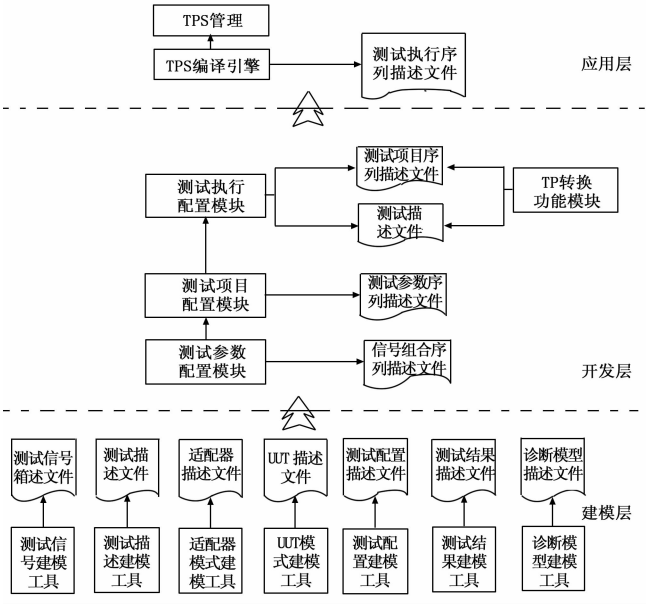

图 1　TPS 开发平台架构示意

建模层提供测试系统系统建模工具，根据测试系统系统数据模型生成符合 ATML 要求的被测对象描述文件集合。

开发层为 TPS 开发平台的功能模块组合，通过图形和表单方式完成测试信号、测试参数、测试项目的组合开发，也用于外部 TPS 的验证和移植，形成符合被测对象描述文件要求的测试项目序列。

应用层是 TPS 开发平台的执行引擎，将被测对象测试项目序列与测试系统系统数据模型进行匹配，形成可供 TPS 运行平台使用的测试执行序列。

开发人员按照测试系统软件开发规范，使用 TPS 开发平台可以形成各测试对象的 TPS，完成 TPS 开发和移植，主要功能包括：

UUT 建模：建立匹配测试系统数据模型和具体被测对

象的 UUT 描述实例，包括 UUT 信息的定义、UUT 插针定义、UUT 插针组、UUT 信号与 UUT 插针组的匹配定义、UUT 与接口适配器的连接定义等；

接口适配器建模：建立匹配测试系统数据模型和具体被测对象的接口适配器描述实例，包括接口适配器的基本信息、插针定义、以及与测试系统、UUT 之间的连接关系定义等；

测试信号建模：建立匹配测试系统数据模型和具体被测对象的信号描述实例，包括信号的名称、类型、方向等测试信号定义，以及参数、范围、分辨率、精确度等信号参数属性的描述信息；

测试配置建模：建立匹配测试系统数据模型和具体被测对象的测试配置描述实例，包括被测对象关联的硬件、软件和相关文档等配置信息；

测试描述建模：建立匹配测试系统数据模型和具体被测对象的测试描述实例，包括 UUT 执行测试时所需要的测试需求（激励信号、响应信号等）、测试诊断、接口需求以及测试的执行行为等物理特性及操作需求信息；

测试结果建模：建立匹配测试系统数据模型和具体被测对象的测试结果描述实例，包括测量值、测试状态、以及相关的操作员、测试站和环境等信息；

TPS 开发：采用表单和图形的方式描述测试流程、UUT、适配器等，通过面向信号的测试程序开发方式，形成 ATML 测试程序，满足 UUT 检测需求；

TPS 导入/导出：支持 UUT、适配器、测试配置、测试需求等描述信息文件在不同测试系统平台的导入、导出，并编译生成 TPS；

TPS 转换：对 TPS 进行移植检查和验证，将其转换为符合测试系统软件识别和运行的 TPS。

3 测试结果建模方法

测试结果信息的主要作用是向测试系统软件传递各个测试项目的结果判据信息，其标准化描述是实现不同平台之间测试结果信息交换的关键，并且标准化的测试结果信息对于实现 ATS 被测单元的通用性起着至关重要的作用。利用 ATML 标准中的 TestResults. xsd 可对测试结果进行标准化描述。

ATML 标准中的 TestResults. xsd 是用来指导用户对描述测试结果的架构文件。符合 TestResults. xsd 的测试结果文档必须以 TestResults 为根元素，其下面的子元素涵盖所有在测试中与测试结果有关的信息，包括 Personnel、PreTestRepairs、References、ResultSet、Site、TestDescription、TestProgram、TestStation、UUT、Workorder 以及 Extension 元素。由此可知 TestResults. xsd 的结构相对复杂，因为在标准制定时需要考虑其全面性和功能的可扩展性，但对于用户来说大多信息略显冗余，因此 Schema 文件给了用户极大的自由度去从这些子元素中筛选、删减，选择最需要的内容来进行描述，这也充分体现了在工程中对 ATML 的灵活运用。通过对 TestResults. xsd 文件的充分研究，简化了其复杂的树形结构，简化后的 TestResults 的 Schema 视图如图 1 所示。

从图中我们可以看出，该文件仍然以 TestResults 为根元素，其包含四个子元素，分别是 tr：Personnel（tr 为名称空间前缀，是一种用来解决名称冲突的机制）、tr：ResultSet、tr：TestDescrption 和 tr：UUT。其中 tr：Personnel 元素利用其子元素 tr：CustomerRepresentative 描述测试相关人员信息，tr：ResultSet 元素用来描述测试结果的集合，属于 tr：TestGroup 类型，它包含一个 tr：OutCome 元素和 tr：TestGroup 元素（该元素个数不受限制），tr：OutCome 元素用来描述整体测试结果；tr：TestDescrption 元素，属于在 Common. xsd 文件中定义的 c：ItemDescriptionReference 类型，用来标记测试描述的相关信息；tr：UUT 元素，属于 Common. xsd 文件中定义的 c：ItemInstance 类型，用来标记被测对象名称。

tr：TestGroup 为此文档中的核心，对其进行修改后，其逻辑视图如图 2 所示。该部分用于描述测试的结果数据，是整个架构中最重要的一部分，该部分的描述方法也最为关键。

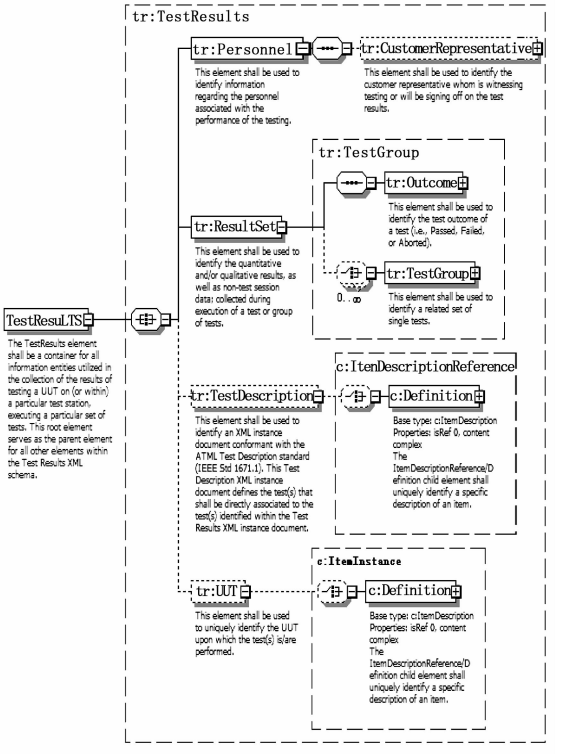

图 2 修改后的测试结果 Schema 文件逻辑视图

测试结果数据描述的部分由 tr：ResultSet 元素构成，它继承于 tr：TestGroup 类型。tr：TestGroup 类型逻辑类型如图 3～4 所示。

考虑到测试结果数据数目的不确定，ATML 用递归思

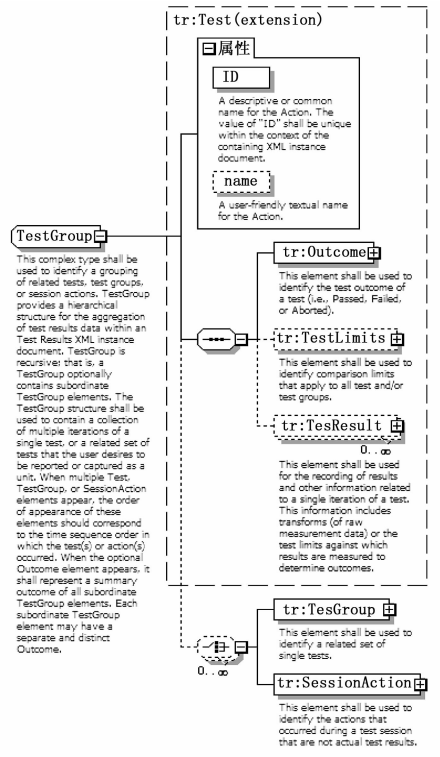

图 3　TestGroup 逻辑视图

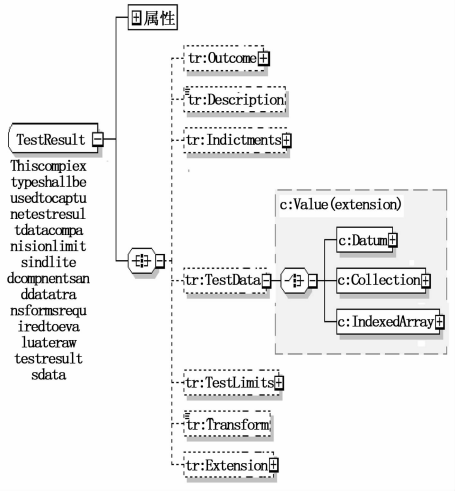

图 4　TestResult 逻辑视图

想来定义了 tr：TestGroup 类型。tr：TestGroup 不仅继承了 tr：Test 的全部元素和属性，同时还扩展了 tr：Test、tr：SessionAction 以及一个嵌套的 tr：TestGroup 元素。其中，tr：Outcome 元素用来标记测试结果是否正确；tr：TestLimits 用来标记标准值的范围；tr：TestResult 用来存储测试结果的数据；两个属性 ID 和 name 用来标识每组数据的 ID 号和具体信号名称。由于测试结果数据往往是多维数据，为了全面详细描述测试结果的所有参数信息，选择用 tr：TestGroup 循环嵌套的方式来存储所有结果数据；tr：SessionAction 元素用来标记测试结果描述结束的信息。其余冗余的元素我们仍然可以选择舍弃。

下面主要介绍 tr：TestResult，该类型逻辑视图如图 3 所示。

该元素中，最重要的子元素是 tr：TestData，用来存储具体数据。由图可知 tr：TestData 继承自 c：Value（前缀“c：”表示该元素在 Common. xsd 中定义），具体代码如下：

```
<xs:element name="TestData" minOccurs="0">
<xs:complexType>
  <xs:complexContent>
  <xs:extension base="c:Value"/>
  </xs:complexContent>
</xs:complexType>
</xs:element>
```

通过继承的方式，tr：TestData 就可以使用 c：Value 的子元素。而 Value 元素可以在 c：Datum、c：Collection 与 c：IndexedArray 三者中进行选择，这种定义方式允许测试数据可以有不同的形式，充分考虑了测试需求多样性的特点，体现了面向对象中的多态性思想。根据定义，我们可以利用 c：Datum 元素描述单个数据，c：Collection 用来描述数据集合，而 IndexedArray 元素用来描述波形数据类型的数值数组。c：Datum 元素为 c：DatumType 类型，该类型在 Schema 中被定义为：

```
<xs:complexType name="DatumType" abstract="true">
  <xs:group ref="c:DatumQuality"/>
  <xs:attributeGroup ref="c:UnitAttributes"/>
</xs:complexType>
```

在该段代码中，abstract=" true" 属性的定义表明该类型不可以被直接使用，而必须指明具体类型（其扩展类型），否则，使用无效。这种定义方式充分体现了面向对象中纯虚类思想。

4　测试结果模型配置工具开发

由于 XML 文件格式要求相对而言较严格，如果直接利用文本编辑工具手动输入则非常容易出错，而利用 XML Spy 类似工具填充 XML 元素与属性数据，则开发效率非常低。为此，我们开发了 ATML 标准的测试结果配置工具，以实现测试结果模型的自动配置，自动生成符合 ATML 标准的 XML 文件。

测试结果配置工具采用 Visual Studio 2013 中的 MFC 并结合了 COM 组件技术进行开发。在组件中，建立并对外开放 ITR 接口，以供客户端程序调用来实现 XML 文档的自动配置。在 ITR 接口中，设计的方法定义如下：

```
STDMETHOD_(HRESULT, put_UUT_Name)( BSTR newVal);
STDMETHOD_(HRESULT, Savefile)( BSTR path);
STDMETHOD_(HRESULT, put_Descripition)( BSTR newVal);
STDMETHOD_(HRESULT, put_OutcomeValue)( short ne-
```

```
wVal);
    STDMETHOD_(HRESULT, put_TestResultID)(short newVal);
    STDMETHOD_(HRESULT, AddData)(BSTR itemName, BSTR qualifier, short valueLength, double min, double max, double * value, short resultFlag);
```

可以看出,客户端可以调用接口中的多种方法,来实现对测试结果中的被测对象名称添加、相关测试信息的描述、测试结果 ID 号的添加、测试数据中数据值以及相关信息的添加、存储并生成相应的 XML 文件等功能。其中 AddData 方法至关重要,该方法中通过指针遍历存储数据的单链表结构以实现对多组数据的访问及添加,部分关键代码如下:

```
STDMETHODIMP_(HRESULT) TResult::XTR::AddData(BSTR itemName, BSTR qualifier, short valueLength, double min, double max, double * value, short resultFlag)
    {
    TestData * * td = &m_pTestGroup;
    char * itemNameString, * qualifierString;
    itemNameString = (char * )malloc(strlen((itemName)) + 1);
    strcpy(itemNameString, OLE2T(itemName));
    qualifierString = (char * )malloc(strlen((qualifier)) + 1);
    strcpy(qualifierString, OLE2T(qualifier));
    for (int i = 0; i<m_dataNumber; i++)
    {
    td = &((* td)->next);//指针指向下一组数据
    }
    (* td) = (TestData * )malloc(sizeof(TestData));;
    strcpy((char * )(* td)->testdataName, (char * )itemNameString); //存储 itemName 信息
    strcpy((char * )(* td)-> testqualifier, (char * )qualifierString); // 存储 qualifier 信息
    (* td)-> Length = valueLength; // 存储 valueLength 信息
    (* td)-> min_limit = (double * )malloc(sizeof(double));
    (* td)-> max_limit = (double * )malloc(sizeof(double));
    (* td)->data = (double * )malloc(sizeof(double) * valueLength);
    ……
    (* td)->next = NULL;
    }
```

对于客户端界面的设计,如图 5 所示。由图可见,我们可以在客户端中直接对 TestResults 的中的相应元素、属性进行配置以及添加测试数据及相关信息,其中,用户使用时间可有后台程序自动设定。在测试数据信息添加完成后,即可点击保存按钮,程序会自动生成符合标准的测试结果 XML 文件。

5 总结

本文通过对 ATML 标准深入研究后,提出了测试系统软件平台的基本架构;设计完成了符合 ATML 标准的测试结果 Schema,为测试结果的建模提供了依据;并在此基础上,设计并开发了相应的配置工具,实现了测试结果中信息的自动配置。在实际中的应用中,基于 ATML 的软件平台架构为测试系统的开发提供了依据,为未来自动测试系统发展奠定良好基础;且本文开发测试结果建模工具效果良好,可生成所需的测试结果 XML 文件为测试过程带来了极大便利,为测试信息的共享和重用奠定了良好的基础。

图 5 配置工具界面

参考文献:

[1] IEEE Standards Coordination Committee 20. IEEE Standard for Automatic Test Markup Language (ATML) for exchanging automatic test equipment and test information via XML [S]. IEEE Standards Coordination Committee, USA, 2010.

[2] IEEE Standards Coordination Committee 20. IEEE Trial Use Standard for Automatic Test Markup Language (ATML) for exchanging automatic test equipment and test information via XML: Exchanging Test Descriptions [S]. IEEE Standards Coordination Committee, USA, 2008.

[3] 严 乐, 司 斌, 张从霞, 等. 基于 ATML 标准的空空导弹 ATS 标准化描述 [J]. 测控技术, 2016, 35 (2): 152-156.

[4] 刘斌斌, 胡建旺. 基于 ATML 的 ATS 开关系统描述及工具开发 [J]. 计算机测量与控制, 2013, 21 (3): 811-812.

[5] 刘乃强. 通用测试系统软件架构及关键技术的设计与实现 [D]. 太原: 中北大学, 2016.

[6] 张桂英, 侯 倩, 范利花, 等. 基于 ATML 的可移植 TPS 开发技术研究 [J]. 测控技术, 2017, 36 (1): 112-115.

[7] 林志文, 贺 喆, 刘松风. 基于 ATML 的系统级 TPS 开发及综合诊断应用 [J]. 仪器仪表学报, 2010. 31 (5): 1010-1016.

[8] 钟建林, 何 友, 齐玉东. 基于 IEEE1641 标准的自动测试系统体系结构 [J]. 计算机测量与控制, 2009, 17 (5): 854-856.

[9] 张 文, 杨京礼. 一种面向信号的自动测试系统资源分配方法设计 [J]. 现代电子技术, 2013, 36 (19): 42-45.

[10] 赵鹏鹏, 崔少辉, 王诗源. ATML 中的能力描述方法研究 [J]. 信息技术, 2013 (1): 88-91.