基于XML实现NGOSS一致性测试资源管理

张 川[1] 王 柏[1] 艾 波[2]

(北京邮电大学通信软件工程中心 北京100876)[1] (中国联通信息系统部 北京100032)[2]

摘 要 介绍了NGOSS一致性测试资源的内容及分类,分析了对其进行管理的特点,提出了一个通用的NGOSS一致性测试资源管理模型,并详细研究了该模型的数据组织方式。最后,举例说明了如何利用XML技术实现该模型。

关键词 NGOSS一致性测试,资源管理,XML

The Implement of Test Resource Management in NGOSS Compliance Test with XML

ZHANG Chuan[1] WANG Bai[1] AI Bo[2]

(Telecommunication Software Engineering Research Center, Beijing University of Posts and Telecommunications, Beijing 100876)[1]

(Information System Dept., China Unicom, Beijing 100032)[2]

Abstract This paper introduces the meaning and classification of test resource in NGOSS compliance test. With elaborated analyses of the characteristics, a general test resource management model as well as its resource organization mechanism is proposed. Finally a case study is presented on how to implement the model with XML.

Keywords NGOSS compliance test, Test resource management, XML

1 引言

NGOSS(Next Generation Operation Support System,下一代运营支撑系统)是TMF(Telecommunication Management Forum,电信管理论坛)针对电信业务运营支撑系统的开发、实现和发布而制定的集成框架。目的是定义一系列标准的数据交换接口和业务集成点,使电信运营过程中的各个参与者的信息模型和商业流程能够互相参考甚至复用,从而实现电信运营过程的自动化。针对NGOSS需要进行多角度的测试,如功能测试、性能测试和一致性测试等,其中NGOSS一致性测试是指判断软件系统是否符合NGOSS原则的过程,这里的NGOSS原则包括针对各种的电信业务需求而制定出的各项业务标准及接口规范。

NGOSS一致性测试通过测试系统与被测系统服务接口之间业务消息的交互来判断被测系统的实现是否符合NGOSS原则。在NGOSS一致性测试中涉及到大量的如测试准备数据、测试流程脚本及测试结果数据等测试资源。本文的目的是研究在NGOSS一致性测试中如何科学有效地管理这些测试资源,使其更好地服务于NGOSS的自动化测试工作。

本文的研究是以北京邮电大学通信软件工程中心实验室研制的统一测试支撑系统(Universal Test Support System,简称Uni-TSS)为依托的。该系统是一个具有一定通用性的测试工具,能够支持一致性测试、功能测试及性能测试等,本文研究的测试资源管理器是其中的一个重要的组成部分。下面以NGOSS一致性测试中的资源管理为例说明该测试资源管理器的内容、参考模型及技术实现。

2 NGOSS一致性测试资源管理模型

首先对测试资源作一个定义:一般来说,测试资源是测试工作中涉及到的数据及相关配置文件和脚本文件的总称,包括测试流程信息和相关测试数据的内容及格式等。

2.1 测试资源的内容及分类

NGOSS一致性测试资源由以下几部分组成:

·动态资源:动态资源是测试流程中控制流的定义,描述被测电信业务测试消息交互的先后顺序。测试流程需要描述顺序、循环及条件转移等控制信息,在测试执行时,这些信息会根据测试系统与被测系统交互的消息内容确定下一步的测试动作,因此也被称为动态资源。动态资源只描述测试流程的控制流信息,不包括每一测试步骤中涉及到的测试数据

张 川 博士研究生,研究方向:通信软件。王 柏 博士,教授,博士生导师,研究方向:分布式计算。艾 波 博士,教授,博士生导师,研究方向:通信软件。

等具体信息。

·静态资源：是指在测试执行中结构化的、保持不变的测试资源，包括测试准备数据、配置信息和格式转换算法。

·测试准备数据指按照 NGOSS 原则和实际测试需求编写出的业务消息内容，包括向被测系统发送的测试消息及相应的预期返回消息。特别地，本文中的数据资源只包括数据内容本身，而与数据格式相关的信息在格式转换算法中说明。

·配置文件是指测试执行时的通信协议及目标主机位置等相关信息。由于被测系统采用的技术不同，测试系统需要采用不同的协议与被测系统交互，同时每次测试时被测主机 IP 地址和端口号也会发生变化，因此需要对这些信息进行专门的管理。

·格式转换算法是指由于测试系统需要与不同的应用系统交互消息，这些消息格式可能是异构的。因此，需要针对每一种数据定义出相应的格式转换算法，屏蔽这些数据格式的异构性，使得同一套测试数据适用于不同格式的数据。

·测试流程资源：指将动态资源与静态资源结合起来，生成可被测试系统解释并执行的脚本文件。它既能表达测试流程控制信息，也包括每一步骤的测试数据及相应格式转换算法等。

·测试结果：指在测试执行过程中，测试系统接收到被测系统返回的消息内容，与静态数据中的预期返回消息的格式相同，也包括消息包头和消息包体两部分内容。

·结果比对规则：指根据 NGOSS 原则，并结合测试需求编写出的一系列布尔表达式。

·一致性报告：测试系统将事先定义好的比对规则应用在测试结果上，得到的比对结果称为一致性事件，将一致性事件汇总并统计后，得到一致性报告。

综上，测试支撑系统的资源应该包括动态资源、静态资源、测试流程、测试结果、结果比对规则以及一致性报告五个部分。

2.2 测试资源管理的特点

NGOSS 一致性测试作为一致性测试的研究方向之一，其测试资源管理具有以下特点：

·数据格式复杂多样。NGOSS 一致性测试数据格式的多样性体现在以下两个方面。

·NGOSS 一致性测试需要针对公共通信平台、合约定义接口、服务注册与查找、外部工作流控制以及共享信息等五个角度获得数据进行逻辑完备性测试[4]，不同角度的数据项不尽相同。例如针对公共通信平台的测试，需要获得关于消息的发起端、接收端及消息发布方式等信息；而针对服务注册与查找则需要记录服务接口的名称、描述及输入输出参数等信息。

·NGOSS 一致性测试需要与各个子系统交互消息，它们可能采用不同类型的中间件技术，对应的数据存储格式也不相同，如 XML、FML 或 String 等。

·资源可复用性要求较高。由于 NGOSS 系统规模庞大，需要多个开发商同时参与建设，NGOSS 一致性测试中会测试多个开发商的同一类产品。因此，测试资源需要进行多次重复利用，这要求测试系统有效地组织测试资源，提高资源可复用性。

·测试结果文件数量庞大。NGOSS 一致性测试基本不涉及对用户界面的测试和评价，但需要针对服务接口上交互的消息格式和内容进行校验，由于电信业务中会产生大量的消息交互，因此需要对大量结果文件进行管理。

2.3 测试资源管理模型

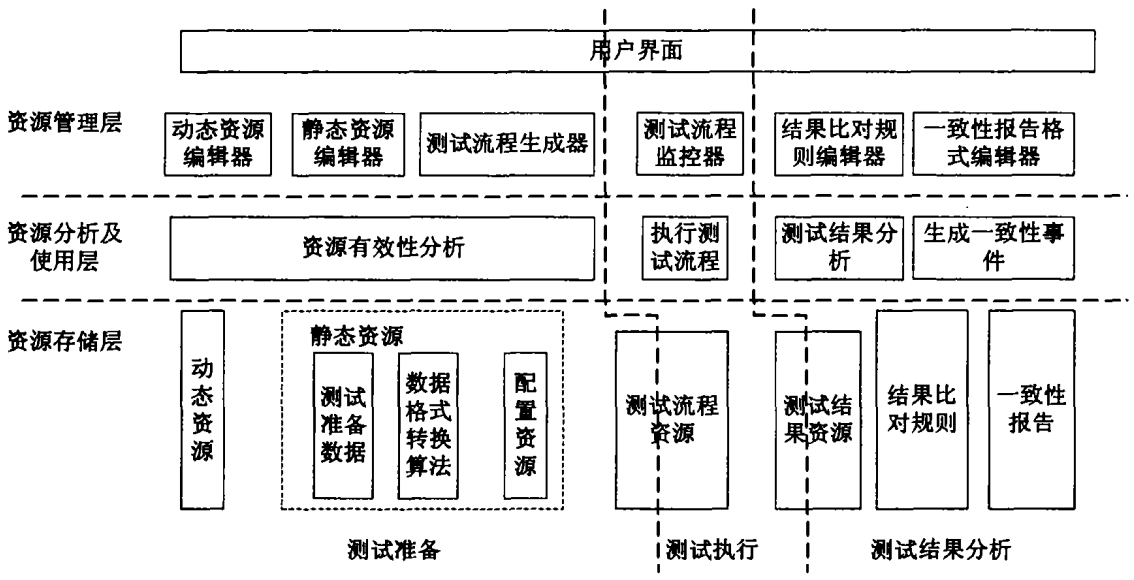

图1 NGOSS 一致性测试资源管理模型

根据以上特点，本文按照测试工作的三个阶段提出 NGOSS 一致性测试资源管理模型。

·测试准备阶段：生成动态及静态资源，并按照不同的测试需求将它们进行整合，生成测试流程。在资源管理层，用户首先利用动态资源编辑器和静态资源编辑器，按照被测业务的要求定制动态资源及测试准备数据，并根据实际情况编写配置文件和格式转换算法；然后在测试流程生成器中结合动态资源与静态资源，生成可以被测试系统解释执行的测试流程脚本；在资源分析及使用层，对生成的各种资源进行有效性分析，主要检查生成的资源是否真正符合测试系统的要求，以及资源之间是否存在重复或冲突的情况；在资源存储层，以文件形式保存这些生成的测试资源。

·测试执行阶段：在资源管理层通过测试流程监控器对测试流程的执行过程进行监控，在资源分析及使用层解释执行测试准备阶段生成的测试流程，与被测系统进行交互，同时将测试得到的结果文件作为测试结果资源存入资源存储层。

·测试结果分析阶段：在资源管理层，用户根据 NGOSS 一致性测试的相关要求，利用结果比对规则编辑器和一致性报告格式编辑器定制本次测试的结果比对规则和报告格式文件；在资源分析及使用层，将结果比对规则应用到测试执行后的测试结果资源上，得出比对结果，生成一致性事件，并按照一致性测试报告的格式要求给出一致性报告；结果比对规则和一致性报告都保存在资源存储层。

从以上分析可以看出，在这个测试资源管理模型中，测试资源管理工作被分为三个阶段，每一个阶段的工作被分在三个层次上实现，使得测试资源管理工作的界线非常清晰，便于测试资源管理系统的实现和进一步扩展。

3 NGOSS 一致性测试资源组织方式

根据 NGOSS 一致性测试资源管理模型，可以看出 NGOSS 一致性测试资源分为四个部分进行组织，包括：动态和静态资源及其实现、测试流程、测试结果资源及比对规则、一致性测试报告。其组织方式如图2所示。

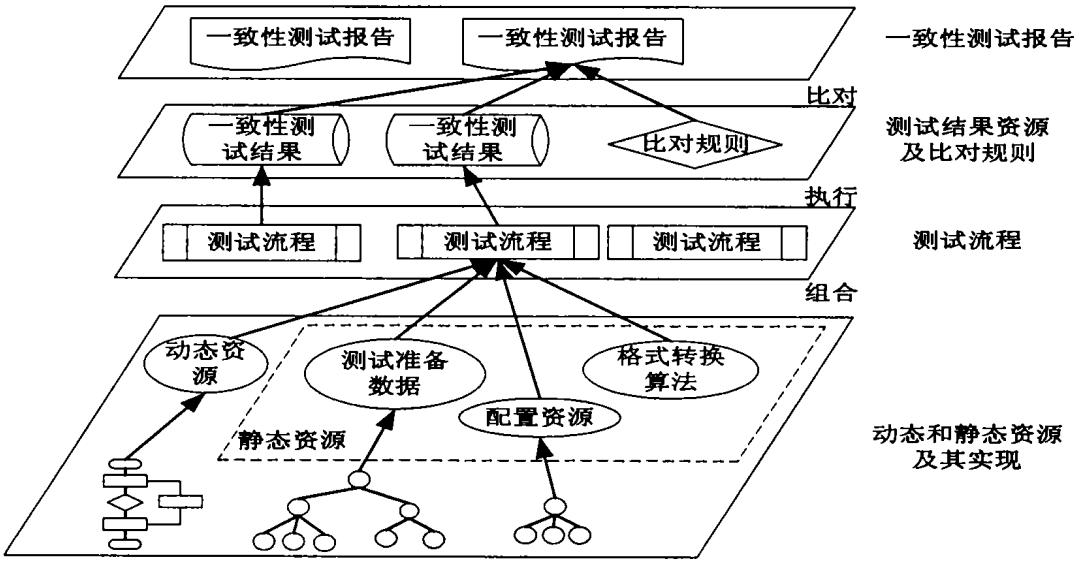

图2 NGOSS 一致性测试资源组织方式

3.1 动态和静态资源及其实现

在测试准备阶段对各种动态资源、静态资源及其具体实现进行了定义：

·动态资源包括若干个测试控制流脚本，每个测试控制流由若干测试步骤组成，描述了测试行为的控制方式，包括顺序、循环及条件转移等。

·静态资源包括测试准备数据、配置资源和格式转换算法等。

·测试准备数据由多个数据集组成，每个数据集包括发送数据文件和预期返回数据文件，它们都包括消息包头和包体两部分，各自又分别由多个字段组成，因此测试准备数据集是深度为3的多叉树。

·配置资源由多个对应特定测试环境的配置文件组成，包括测试执行时的通信协议及目标主机位置等配置信息，因此配置文件是深度为2的多叉树。

·格式转换算法指针对不同的数据格式，定义出相应的格式转换算法，该算法一般用参数化的程序脚本来实现。通过对格式转换算法的独立管理，可以在生成测试流程的时候经过简单的配置，利用已有的格式转换算法方便地支持新的测试需求。

3.2 测试流程资源

当用户通过界面将动态资源与静态资源结合起来后，就生成了可以被真正执行的测试流程资源。如图3所示，测试流程资源由多个测试流程组成，每个

测试流程不仅包括控制流向信息，还包括每一步骤涉及到的测试准备数据、配置文件及格式转换算法等。

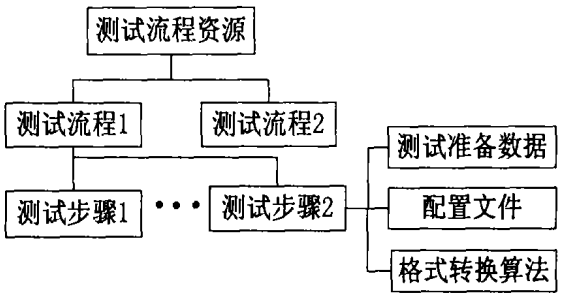

图3 测试流程资源组织方式

3.3 测试结果

测试流程执行后生成测试结果文件，该文件由多个结果集组成，每个结果集对应一次消息交互的返回内容，包括包头和包体信息，各自又分别由若干字段组成。测试结果文件信息与静态资源中测试准备数据集的格式一致，都是深度为3的多叉树，这样可以很方便地进行结果数据比对。

3.4 一致性测试报告

在对测试结果进行分析之后，生成一致性事件，对其统计后形成一致性报告，该报告分为通过测试的流程和未通过测试的流程两部分，其中针对每个未通过测试的流程需要记录其不一致现象描述和原因分析。最后，将一致性报告作为最终测试结论提交给用户。

4 用 XML 实现 NGOSS 一致性测试资源管理

在比较了多种技术之后，我们选定了 XML 语言作为测试资源管理模型的数据描述基础。XML 的特点是数据结构化、自我描述性和用户可自定义标记的灵活性，强调的是描述数据内容的组织存放结构。XML Schema 是一种描述信息结构的模型，它为 XML 文档建立了一个模式，可以描述各个字段的名称、类型和其他相关信息。与传统测试软件工具中使用关系型数据库存储资源信息相比，用户利用 XML 可以描述更多类型的数据，并且能够根据测试需求的变化灵活地调整文件格式。此外，在测试资源分析的过程中，需要对其进行格式检查和内容比对等文件操作，利用 XML Schema 可以很容易地实现这些工作。

4.1 静态资源的管理

利用 XML 可以很方便地实现对静态资源的管理：

·测试系统中的静态资源大都采用树形结构进行组织，与 XML 描述数据的方式一致。

·XML 实现了数据内容与数据格式的分离，具有机器可理解的强大的数据描述能力，可以十分方便地定义测试准备数据和编写相应的格式转换脚本。

·可通过 XML schema 方便地对 XML 的数据格式进行校验，实现测试结果的比对。

以静态资源中的测试准备数据为例加以说明：测试准备数据由多个数据集组成，每个数据集包括发送数据文件和预期返回数据文件，它们结构相同，都由包头信息和包体信息组成，包头信息包括描述信息如日期、序列号和包类型信息等，包体信息包括被测服务接口的输入及返回参数，例如姓名、年龄和性别等信息。使用 XML 表示一个发送数据文件结构如下所示：

```
〈request〉
    〈data_head〉…〈/data_head〉
    〈data_body〉…〈/data_body〉
〈/request〉
```

以 XML 表示的数据资源具有良好的可扩展性，当测试数据项随着需求发生变化时，可以通过增加或修改 XML 文件的相应标记实现对测试数据项的改动。

4.2 动态资源的管理

动态资源实质是对测试流程中控制流向的描述，因此可以采用任何一种工作流定义语言描述，同样地，XML 经过扩展也可以描述流程的运行：

·顺序结构：用〈sequence〉标识一个顺序结构。一个〈sequence〉中包含一个或多个按顺序执行的活动，这些活动执行的顺序就是在〈sequence〉中出现的顺序。如下所示：

```
〈sequence〉
    〈step name="step1"type="invocation"〉
      〈action〉InsertUserInfo〈action〉
    〈/step〉
    〈step name="step2"type="reponse"〉
      〈action〉QueryUserInfo〈action〉
    〈/step〉
〈/sequence〉
```

其中，〈sequence〉包括两个测试步骤 step1和 step2，它们按照顺序执行，用〈step〉标识，属性 type 表示该步骤是向 CDI 发送消息(invocation)还是接收 CDI 的返回消息(response)，这两个步骤对应的活动为 InsertUserInfo 和 QueryUserInfo，用〈action〉标识，它们是在测试流程中涉及到的 CDI 在 NGOSS 中注册的句柄信息。

·循环结构：用〈while〉来标识一个循环结构。如下所示：

```
〈while〉
    〈whilecondition name="condition1"〉
        age<30
    〈/whilecondition〉
    〈sequence〉 … 〈/sequence〉
  …
〈/while〉
```

其中，〈whilecondition〉标识循环条件，包括条件名 condition1和条件表达式“age<30”。当有多个

循环条件时，可定义多个〈whilecondition〉元素，并用关系运算符将其连接。接着是对循环中活动的描述，和顺序结构类似。

·条件转移结构：条件转移结构用〈switch〉标识。如下所示：

```
〈switch〉
    〈case condition="condition1"〉
        〈sequence〉 … 〈/sequence〉
    〈/case〉
    …
    〈fault〉 … 〈/fault〉
〈/switch〉
```

其中，〈switch〉中含一个或多个〈case〉和一个〈fault〉，〈case〉标识各个转移条件下(condition1)的活动，其 condition 属性标示转移条件，〈fault〉表示所有〈case〉的条件都不满足的情况下的默认处理。

4.3 测试流程

测试流程可以由测试系统解释执行脚本文件。该脚本不仅要描述流程的控制信息，还要在流程执行的过程中根据测试的需要随时与被测服务接口交互测试消息。下面以扩展的 XML 语言描述顺序测试流程为例进行说明：

```
〈sequence〉
   〈step name="step1"〉
       〈action〉InsertUserInfo〈action〉
       〈output schema="ResultOfInsert.xsd"〉ResultOfInsert.xml〈/output〉
       〈input schema="NewUserInfo.xsd"〉NewUserInfo.xml〈/input〉
       〈pack〉PackArithmetic1.dll〈/pack〉
       〈configure〉Configure1.xml〈/configure〉
   〈/step〉
〈/sequence〉
```

〈step〉中基本活动类型被扩展为测试步骤、调用外部服务、接收外部消息、格式转换程序及配置文件等五种，分别用〈action〉、〈output〉、〈input〉、〈pack〉和〈configure〉标识。

其中〈action〉与静态资源中的含义相同；〈output〉中包含该步骤对应的预期返回文件及相关信息(其 schema 属性标识对应的 schema 文件名)；同样地，〈input〉中包含该步骤中接收外部传进的消息文件名；〈pack〉中标明该步骤采用的格式转换程序，本例使用动态链接程序(PackArithmetic1.dll)完成该过程；〈configure〉中指示的 XML 文件(Configure1.xml)记录了该步骤执行时所需的配置信息，包括通信方式、主机 IP 及端口号等。

4.4 测试结果的描述和分析

测试执行完毕后，需要按照 NGOSS 一致性原则比对测试结果，并生成一致性测试报告。测试结果文件与预期结果文件格式基本相同，也是由包头信息和包体信息组成，包头信息包括描述信息，如日期、序列号和包类型等，包体信息包括被测服务接口的输入参数或返回参数，例如姓名、年龄和性别等。使用 XML 描述测试结果资源的结构如下所示：

```
〈ResulSet〉
    〈data_head〉…〈/data_head〉
    〈data_body〉…〈/data_body〉
〈/ResulSet〉
```

在得到测试结果并将其用 XML 描述后，就可以很方便地实现测试结果比对分析工作。

结论 目前，作者所在的北京邮电大学通信软件工程中心已按照 TMF 测试体系结构参考模型研制成功了一个 NGOSS 测试通用测试支撑系统(Uni-TSS)，在该系统中采用了本文研究的测试资源管理模型实现了对测试资源的有效管理。并且已经使用了 Uni-TSS 针对国内某电信运营商某市分公司的电信业务支撑系统进行了实际测试，在测试过程中，由于实现了测试资源的有效管理，使测试工作的效率和准确性大大提高，并且使测试资源得到了充分的复用。实践证明上述模型是一种行之有效的测试资源管理方法。

在本文提出的测试资源管理模型中，没有涉及对测试人员及整个测试工作实施策略的管理，下一步研究方向是将测试工作中其他相关因素都作为测试资源管理起来，使整个测试工作更加高效。

参 考 文 献

1 Khoumsi A. A temporal approach for testing distributed systems. Software Engineering. IEEE Transactions on, 2002, 28(11):1085～1103

2 GB914,System Integration Map

3 GB922,Shared Information/Data Model(SID)

4 TMF050～TMF053,NGOSS 系列规范

5 张川，王君珂，王柏. 基于 XML 开发接口测试平台. 见：第六届全国计算机应用联合学术会议论文集，2002.11

6 张川，王柏，艾波. 分布式系统中不确定性分析及其对 Uni－CRM 测试工作的影响. 北京邮电大学学报，2003，26：96～101

7 尹霞，吴建平. 基于测试序列动态选择技术的测试管理的设计与实现. 小型微型计算机系统，2000.1