测试管理工具能力评估框架研究*)

白晓颖[1] 赵冲冲[1] 徐 睿[2]

(清华大学计算机科学与技术系 北京 100084)[1] (北京奥运会组织委员会 北京 100007)[2]

摘 要 管理工具有助于有效利用测试资源、合理安排测试流程、系统组织测试活动。本文提出了一个评估框架,从流程管理、资产管理和实施管理三个方面评估测试管理工具的能力,每个方面都由一组特性刻画,并对目前市场上主流的工具进行了定性的分析和比较。

关键词 测试管理,工具,评估

Research on Evaluation of Test Management Tools

BAI Xiao-Ying[1] ZHAO Chong-Chong[1] XU Rui[2]

(Department of Computer Science and Technology, Tsinghua University, Beijing 100084)[1]

(Beijing Organizing Committee for the Games of the XXIX Olympiad, Beijing 100007)[2]

Abstract Test management tools are essential to facilitate effective resource usage, feasible process schedule and systematical activity organization. However, it is a challenge to evaluate and select a proper tool with acceptable maturity. To counter the challenge, the paper proposes a framework for evaluating the capabilities of test management tools from three aspects: process management, project management, and usability. Each evaluation aspect is further supported by a set of desired features that a test management tools should possess. The framework is illustrated by a case study on three popular marketing products, which are analyzed and compared based on the evaluation criteria.

Keywords Test management, Tools, Evaluation

1 引言

随着技术的发展,软件系统的规模急剧增大,采用国际协作的模式,由位于世界上不同国家不同城市的多个团队联合开发软件系统已经成为目前软件开发的主要趋势[3]。与之相适应,测试也需要物理上分布的多个团队共同参与,各个团队承担不同的任务,并有着不同的项目管理模式。为保证整个系统的一致的、有效的质量控制,测试管理至关重要[10,11]。测试管理有助于系统地、规范地管理各种测试资源和测试活动,以提高测试的效率和质量。

对于大型系统测试,测试管理工具可以帮助组织测试资产、监督项目状态、集成自动化测试工具以及度量测试效果[3,4],能够为所有这些参与者提供一个交流和协作的平台,是项目管理中必不可少的。

近年来,测试工具的应用越来越普遍,很多工具都能提供一定程度的测试管理功能。但是,要选择并应用合适的工具需要认真细致地分析工具的能力和成熟度[10,11]。迄今为止,不论是定量还是定性分析,都仅有少量的研究工作尝试研究测试工具的评估[5,7]。现有的评估方法往往针对各种类型的测试工具。本文针对测试管理工具提出了一个评估框架,从三个方面评估测试管理工具的能力:测试过程管理、资产管理和测试实施管理。每个评估方面都由一系列测试管理所特有的特性组成。

本文第2节对与工具评估相关的研究工作进行了总结;第3节介绍本文提出的评估框架和评估方法;第4节选择三个主流的测试管理工具产品(IBM Rational公司的 TestManager、Mercury Interactive 公司的 TestDirector 以及 Compuware 公司的 QACenter),应用该框架进行评估;最后对全文进行总结并指出未来的研究方向。

2 相关工作

1991年发布了 IEEE 标准 1175[5,7],定义了计算系统-工具交互的参考模型,并发布了一个工具评估系统。该系统支持多种因素的评估,包括工具依赖的因素如性能和可靠性、环境依赖的因素如费用和工具-机构、工具-平台以及工具之间的相互影响等。该系统使用数据表格收集数据并分析信息,可对各种标准进行加权、分级和总结。1992年,R. M. Poston 和 M. P. Sexton 进一步改善了该系统和评估标准[7]。

受美国国防部资助,IDA 主持了一个关于工具分析和评估的研究,并于1992年和1993年发表了两篇报告[10,11]。报告详细说明了每个投资购买的工具的分析方法和结果,包括工具在分析类别、环境需求、工具交互特点等方面的能力,以及一些通用的工具信息如价格、图形化支持和用户数量等。

10年后,J. B. Michael 和他的同事重新考虑该问题并为软件测试工具的评估定义了度量[5]。他们根据被测系统,将测试工具及其评估标准分为两类:过程软件测试工具和面向对象软件测试工具,并定义了一系列通用的度量来评测工具的功能性和非功能性特点,包括用户界面设计、成熟度、工具管理、易用性等。

目前,测试管理已经不再局限于测试文档化。测试管理

*)本文受到以下项目资助:国家科技攻关项目,编号:2002BA904B13;国家自然科学基金项目,编号:60403022。白晓颖 博士,讲师,研究方向为软件工程、软件测试。

工具通常都具有更复杂的功能来实现有效的管理。因此，数据表格和提问形式的通用工具使用评估方法已经不能满足需要。本文建立了一个多维度的定性分析框架，用以评估测试管理工具的能力。

3 评估框架

如图 1 所示，该框架从三个方面评估测试管理工具的能力：流程管理、资产管理和实施管理。图 2 显示了每个层次的特性定义。

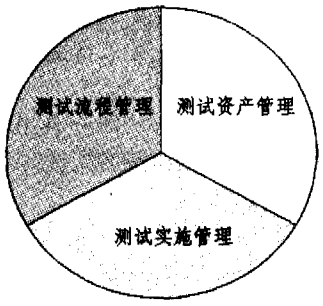

图 1 测试管理工具评估的三层结构

• 测试流程管理的目的是评估工具对贯穿整个生命周期的关键测试活动（如设计测试、执行测试、收集结果、分析结果）以及测试工作流（如缺陷跟踪）的支持能力。

• 测试中通常都要涉及一定的人力、设备和其他资源。测试资产管理评估的目的是评定工具对测试组织、资源分配和调度、规划安排的支持能力。

• 测试实施管理从测试开展的角度对测试管理工具进行评估。测试实施管理评估的目的是评定工具对开展测试和实施测试的支持能力。

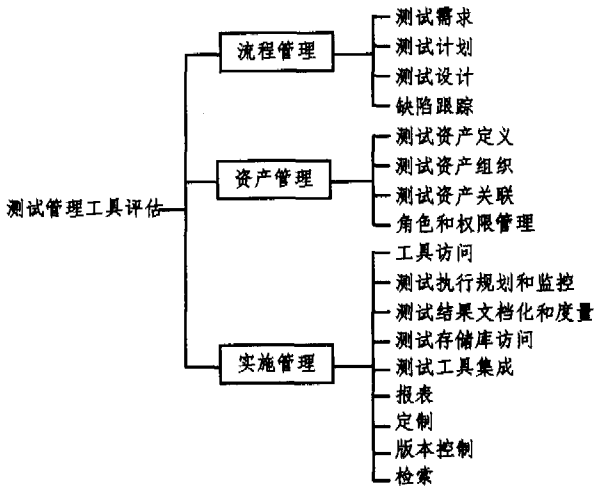

图 2 测试管理工具评估特性

3.1 测试流程管理

如同 Baetjer 所言[1]，构建软件是一个反复迭代的社会学习过程，在这个过程中收集信息、积累知识并生成解决方案。测试是一个贯穿生命周期的流程，如著名的 V 模型所示[8]。Graham Davis 介绍了一种通用的四个阶段测试流程[2]：测试分析从源文件中获取测试需求，即确定“测试什么”；测试设计把测试规格说明和测试脚本记录成文档，用于实现在分析阶段确定的需求，即定义“如何去测试”；测试规划，定义测试集合和测试脚本的运行顺序，即规划“何时测试”；测试执行，评估收集并分析执行结果，即说明“发生什么”。

测试流程管理是测试管理中的重要工作，也是测试管理工具必须具备的能力。本文确认了一组测试管理工具应拥有的关键特征。

3.1.1 *测试需求* 需求定义了测试的目标，测试应根据需求来验证和确认系统。测试需求来自于系统需求，为定义测试范围、确定测试执行和缺陷的优先权、分配测试资源、规划测试任务以及分析测试覆盖提供良好的基础。

理想情况下，测试管理工具应该提供对测试需求定义和组织的支持，并将需求集成到测试生命周期过程中的其他测试活动中，如设计、执行和发现缺陷。

3.1.2 *测试计划* 测试设计建立测试项目、定义范围、分解任务、确认里程碑以及每个工作的起始和结束时间。为了支持测试设计，一个测试管理工具必须具备如下能力：

• 测试任务定义和分解；

• 为每个工作进行测试规划；定义里程碑；

• 为每个工作分配测试资源。

3.1.3 *测试设计* 可以从两个方面定义测试：测试用例和测试场景。测试用例是测试的规格说明，包括测试的前置条件、后置条件、目标需求、输入、执行步骤和期望输出等。测试场景是测试用例执行的环境说明，包括运行策略、压力设计等。测试用例和场景规格说明来自于系统需求（黑盒测试）或系统实现（白盒测试）。

通常，测试管理工具需要在两个层面上支持测试定义。高一层的是通用属性的描述信息，如 ID、名称、作者等等，这些一般都使用由自然语言描述的、基于模板的规格说明。低一层的是使用计算机可读的脚本，脚本能够被计算机编译或解释从而完成自动化执行。

测试定义是测试设计中的一个关键活动。设计可能是测试流程中最费时的阶段，因此，工具能够为测试定义提供有效的支持对及时有效地实施测试至关重要。为了评估测试定义的能力，可以考虑如下属性：

• 可理解性：测试设计人员应该能够很容易地使用定义说明测试，该定义也应该很容易地被随后的测试执行人员理解。

• 可扩展性：为了满足各种不同的使用环境，测试定义应能够随着用户的属性以及应用所特有的目标和行为而扩展。

• 兼容性：测试脚本语言应该独立于技术平台、设计方法和编程语言。

• 复用性：测试定义可以在不同的粒度上复用，如测试数据复用和测试用例之上的测试步骤序列的复用。

3.1.4 *缺陷跟踪* 一旦某次测试运行后没有得到预期的结果，就可能发现了一个缺陷。针对缺陷的操作通常如下：识别、报告、评审、确认、制订优先级、修复、重新提交进行回归测试。测试的缺陷处理过程中涉及了不同的角色，而且不同的角色之间需要相互协作。测试管理工具必须跟踪这样的协作过程以及缺陷状态的转换。特别是当测试内容更新或发生变更时，工具应该能够自动地通知相关人员。例如，向开发人员通报缺陷报告，向测试人员通报缺陷修正信息等。电子邮件是及时通知方式中的一种，很多工具都支持电子邮件并能够自动触发电子邮件系统和自动发送信息。

3.2 测试资产管理

测试中涉及了大量的资产。除了传统的用于实施测试的计算机和网络设备外，现在的测试管理还需要考虑测试中其

他的“资源”，如测试脚本、测试人员等。将测试中的“硬件”(资产)和“软件”(流程)分开考虑，有利于更深入、更准确地评估测试管理工具的能力。

3.2.1 *测试资产定义* 在最初的 Ad hoc 测试中，完成测试所需的硬件设备(计算机、网络设备和终端设备等)是主要的测试资产。随着测试逐步地系统化和工程化，需要管理越来越多的对象，例如可以将测试设计中的很多对象(脚本、用例和场景等)当作资产的一部分系统加以管理。

基于对测试的不同认识，不同的测试管理工具对测试资产有不同的定义。通常来说，测试资产划分的粒度越小、范围越广，测试管理工具对测试资产的管理能力越好。

3.2.2 *测试资产组织* 测试人员对测试的认识总是需要一个过程，随着测试过程的深入，测试将会越来越具体越来越细化。因此，将测试分类并分层组织是必然也是必要的。一个清晰的多层分组机制有助于分析测试的临界状态和分析测试资源，还可以帮助分析测试的完备性和一致性(C&C)，避免出现相互矛盾的测试和重复的测试。

测试可以根据不同的策略分组。例如可以以需求分解为一种策略，每个需求都和一组验证测试相关联，包括正常输入、不正常输入、正常路径和异常处理。另一种策略可以是系统分解，每个模块都和一组执行其各种功能和接口的测试相关。

测试管理工具需要在多个层次上和不同粒度上支持使用各种策略分解测试，还需要支持在组内或在多个组之间对测试进行 C&C 检查。

3.2.3 *测试资产关联* 测试资产——诸如测试用例、测试场景、测试运行和缺陷——之间相互联系，而且也和其他软件元素相关联，如需求、软件组件等。保持这些连接信息被记录是很重要的，因为一个部分的变更可能会扩散到其他很多部分，这被称作涟漪效应[9]。一旦发生某个变更，必须定位所有受影响的部分。例如，当一个功能变更请求被接受，所有相关软件模块都应被修改并确认、所有相关的测试用例都应被选出并重新确认。只有识别出所有这些依赖并且使用文档记录下这些引用，这个过程才能在受控的条件下进行。

3.2.4 *角色和权限管理* 人员是非常特殊的一种测试资源。测试流程中通常要涉及很多部门的人员，包括设计人员、开发人员、测试人员、质量控制和项目管理人员。为了实现不同的用户可以访问不同的数据并执行不同的操作，测试管理工具必须具备角色和权限管理。基于角色的权限控制是测试管理工具中广泛使用的一种策略，设计良好的角色和权限管理机制可以为项目的安全性提供有力的保障。工具应具有的用户管理能力如下：

— 为每个角色维护一个权限说明，定义该角色在测试流程中在每个数据上的允许的操作；

— 为每个用户维护一个角色说明；

— 支持定制角色权限定义，理想情况下工具应该能够在多个粒度上定制；

— 支持用户-角色定义，以便于用户在必要时能够被赋予不同的角色。

3.3 测试实施管理

除了流程管理和资产管理，测试管理工具还需要对测试的实施提供支持。如果能够对测试实施提供有效地管理，可以显著地提高工具的易用性。

3.3.1 *工具访问* 目前，软件项目趋向于更大、更复杂以及更分散[3]。测试团队通常是松散组织的，并且分布于世界各地，相互之间距离遥远。很多情况下一个团队不可能完成全部的工作，而是由多个团队协同完成。每个团队都执行子合同或者面对外包的软件模块。然而不管怎样，为了确保产品集成后的整体质量，需要所有的团队在一个统一的管理框架内工作。为了促进分布环境内的协作和合作，测试管理工具必不可少的一个功能就是支持任何人从任何地方容易快捷地访问工具。

3.3.2 *测试执行规划和监控* 可以采用两种方式执行测试：手工和自动。对于手工测试，可以使用测试管理工具记录测试执行的结果；对于自动化测试，测试管理工具需要提供更全面的支持，体现在如下方面：

— 分配计算资源：为了模拟实际的使用环境，测试中通常需要涉及到多台计算机。测试管理工具应该能够在分布式环境下向每台计算机分配任务。

— 规划测试运行：测试可以在不同的粒度上执行，如测试脚本、单个的测试用例/场景、测试用例组、测试场景组等等。规划的目的是定义测试执行的序列，还可能需要定义在测试用例之间切换的条件，如时间限制、重复执行的频率、前置条件和后置条件等。

— 监控执行状态：执行状态可以从两方面来说明。一方面是测试用例的状态，如激活/结束/阻塞状态下的测试用例百分比。另一方面是被测系统的状态，测试管理工具应该能够表现系统状态的各种视图，如内存利用率、CPU 使用率、响应时间、每秒钟页面访问次数和网络阻塞等等。

— 远程控制：在分布式系统测试中，远程测试越来越普遍。通常，测试代理器和测试控制器构成测试环境。测试代理器生成测试负载并在被测系统上运行测试；测试控制器和测试代理器之间允许交互，调度测试运行、监测状态并综合测试结果。因此，测试管理工具应该具有支持远程控制计算代理器的能力，如监控代理器的有效性和调用处理过程等。

3.3.3 *测试结果文档化和度量* 为确认已经满足退出测试的条件，测试结果应被保存成文档。测试过程中记录的信息包括：

- 日期、时间、脚本名以及测试运行的执行人。
- 测试运行的成功/失败结果，以及与结果相关的各种信息。
- 任何缺陷报告的参考。

分析结果数据可以识别缺陷、修正缺陷、评估测试的覆盖效果、评估测试的有效性和生产率以及评估软件的可靠性。为了测量结果，需要使用各种度量。例如，可以使用测试覆盖度量测试的质量，常见的如路径覆盖、数据流覆盖、白盒测试的决策覆盖、需求覆盖、子系统覆盖以及黑盒测试的接口覆盖。

测试管理工具还应该支持测试执行日志和统计报告的维护，便于测试人员在需要时能够快速地找到与特定测试相关的测试日志和测试结果。

3.3.4 *测试存储库访问* 测试存储库保存了所有必需的测试资产。可以从两个方面评估测试存储库的访问。首先是工具所能支持的存储类型的数量，例如文本文件、XML 文件和关系数据库。存储库支持的类型越标准、越通用、越灵活，工具的开放性就越高。其次是访问存储库的方法，集中式的访问控制能够增强安全性而直接访问可以提供更快的响应时间。

3.3.5 测试工具集成 和其他自动化测试工具集成的能力也是非常重要的。测试管理工具在很高的层次上提供了项目管理和控制，而其他工具可以提供具体的、特定的自动化支持，如仿真测试。和其他测试工具集成的能力表现了测试管理工具的开放性和扩展性。

3.3.6 报表 报表是数据处理和分析的一种有效手段，系统必须为不同用户提供不同的视图，用及时、灵活的方式提供所需信息。因此，在整个测试流程(包括测试规划、测试设计、测试执行、测试结果分析和回归测试)中，工具的报表能力是成功管理测试的一个关键因素。

3.3.7 定制 不同的测试项目在组织结构、资产定义和流程方面有所差异。定制功能对于管理工具来说非常重要，可以帮助工具适应不同的环境。在测试管理工具中，常见的定制功能包括：

- 测试定义模板，支持用户自定义的属性；
- 工具的角色权限定义，用户权限定义；
- 流程定义，在测试活动之上定义工作流。

3.3.8 版本控制 考虑到缺陷修改、需求变更等原因，测试过程中需要不断地检查测试的更新情况。保持测试资产和被测试系统之间版本的一致性是非常必要的，版本控制有助于跟踪测试流程的当前状态并为正确的软件版本维护正确的测试数据。

3.3.9 检索 在测试流程中积累了数量庞大的数据，如前所述，这些数据被指定、引用和版本管理。测试管理工具应支持有效的检索机制以便于用户能够便捷高效地查找所需信息。

4 实例分析

已有很多的测试工具对各个测试阶段中的测试活动提供支持，目前的趋势是将各个工具组合到一起以提供更高层次的测试支持。在本文的实例分析中，我们从目前先进的测试工具集中选择了三个有代表性的工具：Compuware 公司的 QACenter v4.65[12]、Mercury Interactive 公司的 TestDirector v7.6[13]和 IBM Rational 公司的 TestManager v2002[14]。

4.1 测试流程管理能力比较

TestDirector 为测试提供强有力的、集成的测试过程支持。它将测试流程划分为四个阶段：需求、测试设计、测试实验室和缺陷。对于每个发现的缺陷，TestDirector 将记录其详细的说明，并将该缺陷与测试集合、测试脚本相关联，跟踪其状态转移并在日志中记录下所有的修改信息。

TestManager 支持核心的测试活动，包括计划、执行、结果和分析。它可以和 RequisitePro 工具结合使用以支持需求管理，和 ClearQuest 结合使用以实现缺陷跟踪，和 ClearCase 结合使用以实现版本控制。

QACenter 没有明确清晰地区分不同阶段的测试，而是在各个阶段使用一个树形结构组织管理所有的测试设计和测试执行。测试可以在三个层次上进行设计：1)测试脚本说明了测试最基本的操作；2)流程是测试脚本的一个有序序列；3)测试类是测试流程的集合，测试类可以递归分解为子类。QACenter 支持在每个层次的测试设计之上执行测试，即测试可以在三种粒度上运行。此外，QACenter 还可以将测试执行与其他的测试工具和覆盖分析工具结合使用，增强某个阶段测试流程的效果，如 BoundsChecker、、Fault Detection、TrueCoverage 和 TrueTime。

总之，TestDirector 的集成度比 TestManager 和 QACenter 更高，它提供了清楚的、结构化的生命周期过程定义和支持。然而，变更控制和回归测试是这些工具都有的弱点。

4.2 测试资产管理能力比较

TestDirector 可以管理如下测试资产：用户、用户组、测试需求、测试设计、测试用例、测试脚本、测试场景、测试结果、缺陷、分析报表、测试计算机和网络配置。TestDirector 强调需求驱动的过程，需求按树形结构组织，测试资产中的对象之间相互连接，这些都为需求覆盖分析奠定了基础。TestDirector 在三个层次上为测试设计提供支持：测试场景、测试脚本和测试步骤。这三个层次是三种测试组织的形式也提供了在不同粒度上的复用。测试执行按测试集合组织管理，测试集合是若干测试脚本的有序序列。TestDirector 提供了两种方法定义这个序列，一种是被称作执行网格的基于表格的定义，另一种是被称作执行流的有向图定义。测试结果按测试集合中的每个测试脚本记录。

TestManager 可以管理如下资产：用户、用户组、测试设计、测试用例、测试脚本、测试场景、测试结果、分析报表、测试计算机和网络配置(需求和缺陷由其他工具辅助完成)。TestManager 的特点之一是强调软件项目的本质——迭代，而这也符合 Rational Unified Process 模型。因此，它为每个测试计划定义迭代，每个迭代都具有明确的目标、里程碑和质量标准，质量标准可以通过测试覆盖度量。TestManager 使用集合 suite 来组织管理测试执行。suite 可以嵌套定义，也就是说一个 suite 可以包含一组测试用例和另一个(或多个)已定义的 suite。在 suite 中，TestManager 还支持一些在测试用例之上的运行控制参数，如 synchronization、delay、transactor、Selector 和 Event。

因为 QACenter 没有明确区分的测试阶段，所以也没有明确的测试资产管理。所有可以管理的测试资产都包含在用于管理测试设计和执行的树型结构中，包括测试设计、测试脚本、测试流程、测试类、测试结果和测试计算机。不同资产之间的关联通过树型结构的父子、兄弟关系体现。

这三个工具在测试中都有项目的定义。在 TestDirector 中，一个项目就是所有的测试资产和活动的容器，也就是说必须先创建项目然后才能向其中添加其他的测试活动。而在 TestManager 和 QACenter 中，项目和其他的测试活动分别维护，它们之间松散耦合并且可以互相关联。

对于用户管理，TestDirector 支持基于角色的权限控制。结合工具 Administrator，TestManager 支持角色定义和权限定义。QACenter 不支持角色定义而是为每个用户都要单独地定义权限。此外，TestDirector 提供了集中式的用户管理即所有项目的用户集中在一起管理，而在 TestManager 和 QACenter 中采用了分散管理即每个项目各自维护项目相关的人员。

从比较中可以看出，TestDirector 使用更灵活更便利，其测试资产划分更细致全面，支持在更精细的粒度上进行角色和权限定义(TestManager 需要借助其他工具)。此外，虽然都记录了交叉引用，但是仍需要更智能的方法来实现 C&C 检查和变更影响分析。

4.3 测试实施管理比较

QACenter 和 TestDirector 都是独立工具，即由一个工具为测试管理提供全面的支持。TestManager 仅支持核心的测试活动，而且必须和其他工具结合使用完成全面的测试活动，

如 Administrator 工具完成项目管理、RequisitePro 完成需求管理、ClearQuest 完成缺陷管理、SoDA 完成报表生成和管理。

TestDirector 采用 B/S 架构，而 TestManager 和 QACenter 采用 C/S 架构。这三个工具都可以使用多种第三方数据库系统用于测试资产存储，如 MS Access、MS SQL 和 Oracle。除了测试管理工具，MI、Rational 和 Compuware 公司还提供很多端对端测试解决方案和工具，这三个工具都可以和测试工具集中的其他工具合作。

TestDirector 和 QACenter 重在管理测试实施的流程，而自动化测试的具体运行则由其他专业的工具完成，如 WinRunner 和 LoadRunner 辅助 TestDirector 完成功能测试和性能测试，QARun 和 QALoad 辅助 QACenter 完成功能测试和性能测试。TestManager 除了完成测试流程的管理，还完成具体的性能的测试执行(Robot 能够独立完成功能测试，但是只能完成性能测试的脚本编写)，即性能测试的监控也由 TestManager 完成。

TestDirector 的分析模块提供了各种各样的分析报表，支持定制，并能够将报表内容生成多种文档。TestDirector 没有提供专业的版本控制，而是通过目录管理和关联机制实现一定程度的版本控制。支持普通的基于文本的查询功能。

TestManager 支持多种分析报表，可以结合 SoDA 工具实现文档化。还可以结合 ClearCase 实现专业的版本控制，结合 ClearQuest 实现变更管理和检索功能。

定制方面，这三个工具都支持缺陷跟踪的过程定制。相比之下，TestDirector 的角色定制和权限定制比 TestManager 和 QACenter 更详细、粒度更小。此外，TestDirector 还支持用户定义项目属性以及电子邮件通知的触发规则。

结论和未来研究工作 测试管理工具对于大型测试至关重要。本文分析了一个成功的测试管理工具所应具有的核心特性并建立了一个评估框架。通过对目前市场上主流产品的实例分析，显示该框架能够全面地展示工具的能力。这样的研究对于选择测试管理工具非常有益，同时也有益于测试管理工具的开发。

我们未来的研究工作包括：1)进一步使用可度量属性强化评估特性；2)开发定量评估的度量单位；3)对更多的实例进行调研，根据实验结果进一步完善本文提出的框架。

参 考 文 献

1 Baetjer H Jr. Software As Capital: And Economic Perspective on Software Engineering. IEEE Computer Society Press, 1997
2 Davis G. Managing the Test Process. In: Proc. of Intl. Conf. on Software Methods and Tools, 2000. 119～126
3 Hefner R H. Collaborative Test Management. In: Proc. of the 26th Annual NASA Goddard Software Engineering Workshop, 2001
4 Liu L, Robson D J. SEMST-A Support Environment for the Management of Software Testing. In: Proc. of the Second Symposium on Assessment of Quality Software Development Tools, 1992. 11～20
5 Michael J B, Bossuyt B J, Snyder B B. Metrics for Measuring the Effectiveness of Software-Testing Tools. In: Proc. of ISSRE 2002, 2002
6 Ohtera H, Yamada S. Optimal Allocation and Control Problems for Software-Testing Resources. IEEE Transactions on Reliability, 1990, 39(2)
7 Poston R M, Sexton M P. Evaluating and Selecting Testing Tools. IEEE Software, 1992, 9(3): 33～42
8 Pfleeger S L. Software Engineering Theory and Practice. Prentice Hall, 1998
9 Yau S S, Kishimoto Z. A Method for Revalidation Modified Programs in the Maintenance Phase. In: Proc. of IEEE COMPSAC, 1987. 272～277
10 Youngblut C, Brykczynski B. An Examination of Selected Software Testing Tools: 1992: [IDA Paper P-2769]. Inst. For Defense Analysiss, Alexandria, Va., Dec. 1992
11 Youngblut C, Brykczynski B. An Examination of Selected Software Testing Tools: 1993: [IDA Paper P-2925]. Inst. For Defense Analysiss, Alexandria, Va., Oct. 1993
12 Compuware Corporation. Available at: http://www.compuware.com
13 Mercury Interactive. Available at: http://www-heva.mercuryinteractive.com
14 Rational Software Corporation. Available at: http://www.rational.com

(上接第 3 页)

7 Johnson D B, Maltz D A. Dynamic source routing in ad hoc wireless networks. In: Mobile Computing, Kluwer Academic Publishers, 1996, 353
8 Perevalov E, Blum R. Delay limited capacity of ad hoc networks: asymptotically optimal transmission and relaying strategy. In: IEEE INFOCOM'03, 2003, 2: 1575～1582
9 Gamal A E, Mammen J, Prabhakar B, et al. Throughput-delay trade-off in wireless networks. IEEE INFOCOM'04, Hong Kong, China, 2004
10 Dousse P T O, Hasler M. Connectivity in ad-hoc and hybrid networks. IEEE INFOCOM'02, 2002
11 Liu B, Liu Z, Towsley D. On the capacity of hybrid wireless networks. In: IEEE INFOCOM'03, 2003, 2: 1543～1552
12 Kozat U C, Tassiulas L. Throughput capacity of random ad hoc networks with infrastructure support. In: ACM MobiCom'03, San Diego, CA, 2003. 55～65
13 Yi S, Pei Y, Kalyanaraman S. On the capacity improvement of ad hoc wireless networks using directional antennas. In: ACM MobiHoc'03, 2003. 108～116
14 Peraki C, Servetto S D. On the maximum stable throughput problem in random networks with directional antennas. ACM MobiCom'03, 2003
15 Negi R, Rajeswaran A. Capacity of power constrained ad-hoc network. IEEE INFOCOM'04, Hong Kong, China, 2004
16 Zhang H, Hou J. Capacity of wireless ad hoc networks under ultra wide band with power constraint. IEEE INFOCOM'05, 2005
17 Dana A, Hassibi B. Power bandwidth trade-off in sensory and ad-hoc wireless networks. In: International Symposium on Information Theory (ISIT'04), Chicago, IL, USA, 2004
18 Dana A, Hassibi B. Power bandwidth trade-off in sensory and ad-hoc wireless networks. Fourth Annual Workshop on Advanced Networking, Pasadena, CA, USA, 2004
19 Dousse O, Thiran P. Connectivity vs capacity in dense ad hoc networks. IEEE INFOCOM'04, Hong Kong, China, 2004
20 Toumpis S, Goldsmith A J. Large wireless networks under fading, mobility, and delay constraints. IEEE INFOCOM'04, Hong Kong, China, 2004
21 Xie L, Kumar P R. A network information theory for wireless communication: Scaling laws and optimal operation. IEEE Transactions on Information Theory, 2004, 50(5): 748～767
22 Gastpar M, Vetterli M. On the capacity of wireless networks: the relay case. IEEE INFOCOM'02, 2002, 3: 1577～1586
23 Toumpis S. Capacity bounds for three classes of wireless networks: Asymmetric, cluster and hybrid. In: ACM MobiHoc'04, 2004. 133～144
24 Li J Y, Blake C, De Couto D S J, et al. Capacity of ad hoc wireless networks. MobiCom'01, Rome, Italy, Jul. 2001. 61～69