编者按　软件测试是提高软件质量与可靠性的重要一环，它和软件质量的概念是密不可分的。测试是手段，质量是目的，因此软件测试已逐渐成为现代软件工程学研究和应用的热点之一。前不久，“中国软件测评机构联盟”在京举行了联盟成立大会暨第一次会议，借此契机，本刊特约专家撰写“软件系统性能测试方法初探”，以便于性能测试工作者学习、参考。

软件系统性能测试方法初探

Study of Software System Performance Testing

上海市计算机软件评测重点实验室(上海计算机软件技术开发中心)　蔡立志　杨根兴

摘　要　介绍了基于网络的大型分布式应用系统性能测试的基本概念和一般过程，包括目的、时机、需求分析、策略设计等关键性内容，为性能测试工作者提供参考和借鉴作用。

关键词　软件系统　性能测试　压力测试　负荷测试

Abstract: *This paper introduced the basic concept and general process of software performance for the distributed software application system based on the network, including the object, opportunity, the requirement analysis and strategy designing of performance testing. It is used for reference for performance testing engineer.*

Keywords: *software system; performance testing; stress testing; load testing*

随着计算机技术和社会生产的不断发展，IT系统已经由早期的单机系统发展为基于网络的B/S模式或C/S模式的大型应用系统，其基本特征是服务器同时为成千上万个用户提供服务，网上银行、电子政务、网上电子商城等等都是比较典型的例子。应用业务量的不断增加意味着系统负载的加重，系统的性能是否能够满足不断增加的负载需求，直接影响到企业对外提供服务的质量。在保证业务逻辑功能正确性的基础上，软件性能作为软件质量的一个重要指标，开始受到更多的关注。

1 性能测试的目的

性能测试通过模拟大量用户操作，对服务器发出请求，增加服务器的负载，同时监控数据库服务器、应用服务器及网络资源的使用情况，考察系统业务的响应时间和这些资源之间的关系，旨在验证系统能力和找出系统瓶颈。性能测试包含以下几个目的：

● 评估系统的能力

系统性能测试是评估系统能力的一个重要手段，通过性能测试用户能够了解在不同的状态下系统业务的响应时间，和在用户可以接受的范围内系统能够处理的并发用户量。系统性能测试通过负载和响应时间这两个关键指标来评估系统的能力，并帮助用户做出决策。

● 识别系统中的瓶颈

基于网络的分布式大型应用系统涉及到客户端、网络、WEB服务器、应用服务器、数据库服务器等诸多方面，其

*本文得到上海市科技发展基金项目支持(项目编号:036505001)

中的每一个方面都有可能成为系统的瓶颈。在一台服务器中的内存、CPU、磁盘 IO 等不同部分对系统性能的影响也不尽相同。性能测试通过不断增加并发的虚拟用户数量，使得系统负载增加到一个极端的水平，从而发现应用系统的瓶颈所在：网络还是服务器，应用服务器还是数据库服务器，在同一个服务器中是内存、CPU 还是磁盘。性能测试为识别系统的瓶颈提供了一个有效的手段，同时也是系统硬件的升级的重要参考依据。

● 系统调优

通过重复不断地运行性能测试，并根据测试结果调整系统的代码和配置，从而达到改进性能的目的。这些调整包括硬件资源和软件配置的修改，内存的大小、应用服务器和数据库服务器的连接池设置、WEB 服务器和 Servlet 引擎之间的连接数、操作系统允许的共享内存配置、允许产生的线程数量等属于修改和调整的范围。

● 检测软件中的问题

尽管性能测试的目的在于测试系统的性能，但性能测试可以发现一些系统中的隐含的问题。由于性能测试的请求次数非常多，可以在较短的时间内发现应用程序存在的内存泄漏或者数据存储的分配和应用的规划不相适应，从而导致应用系统由于部分数据无法正确存储而失败。

● 验证稳定性(Resilience)和可靠性(Reliability)

在一个生产负载下执行测试一定的时间是评估系统稳定性和可靠性是否满足要求的唯一方法。

2 性能测试的时机

系统性能测试可能会出现在以下几个时期：系统试运行阶段、系统运行期间、系统升级。

在完成系统集成并通过功能测试后，对于基于网络的关键性应用系统，应当提供一个阶段的试运行，这是系统性能测试的最好时机。因为系统已经具备了生产所要求的一些必要数据，并且在试运行阶段的环境已经非常接近实际的生产环境，测试结果比一般测试环境具有更大的参考价值，且可避免测试产生的垃圾数据和测试对系统运行的影响。

在系统运行期间，如果用户希望评估系统的性能或者系统出现了性能问题时都会驱动性能测试的产生。后者目的是为了查找性能问题的原因，测试时应注意尽可能地采用运行出现问题的数据，有利于性能问题的重现，而不能

另外构造新的数据。考虑到系统是一个在线的生产系统，在测试前备份运行的数据以保护生产数据是一个必要的步骤，定制专门的策略来查找性能问题的原因是该性能测试的关键。

系统升级包括软件系统升级和硬件系统升级。查找旧系统性能上的瓶颈或者性能关键的业务应用，为升级提供参考依据，从而保护用户的投资效益，在系统的升级后也需性能测试，使用户掌握在系统升级以后系统性能的提升效果。

3 性能测试类型

根据性能内容和方法，性能测试包括以下类型：

● 负载测试：确定在各种工作负载下系统的性能，目标是测试当负载逐渐增加时，系统各项性能指标的变化情况；

● 强度测试：确定在系统资源特别低的条件下软件系统运行情况；

● 容量测试：在用户可接受的响应范围内，确定系统可处理同时在线的最大用户数；

● 压力测试：通过确定一个系统的瓶颈或者最大使用极限的测试；

● 疲劳强度测试：以系统稳定运行情况下能够支持的最大并发用户数或者日常运行用户数，持续执行一段时间业务，通过综合分析交易执行指标和资源监控指标来确定系统处理最大工作强度性能的过程；

● 大数据量测试：大数据量测试侧重点在于数据的量上，包括独立的数据量测试和综合数据量测试。独立的数据量测试针对某些系统存储、传输、统计、查询等业务进行大数据量测试，而综合数据量测试一般和压力性能测试、

负载性能测试、疲劳性能测试相结合。

4 性能测试的需求分析

性能测试的步骤包括：分析测试需求、制定测试策略、制定测试计划、设计测试用例、运行测试用例、分析测试结果。和功能测试一样，进行性能测试的前提必须分析用户的需求。性能测试的需求来源于三个方面：一是用户明示的测试需求，它确定用户对于性能的需求和期望，例如用户希望评估系统的性能状况还是找出系统的极限点或者系统的最大容量；二是应用系统的业务自身对系统提出许多特殊要求，例如实时的生产GIS系统、110报警系统等，业务的实际要求以间接的方式提出了系统性能的测试需求；三是应用所部署的底层支撑系统，显然不同的底层支撑系统对于测试的要求也不尽相同，这些不同点也将部分转化为性能测试不同的需求。根据用户的需求，可以制定不同情况下的测试目标：

- 了解系统状态：给出在正常情况下关键个业务的响应时间；
- 系统容量： 应当告诉用户，在可接受的范围内，系统最多可以承受多少用户同时在线；
- 系统调优：通过系统优化以后，业务的响应时间较调优前是否有比较大的提升；
- 系统升级：比较系统升级前后的系统性能变化，也是基准测试的一种方式；
- 疲劳测试：检查系统长时间运行状态和设计目标的符合性。

5 系统分析

5.1 架构分析

一个基于网络的大型应用系统，通常由客户端或者浏览器、WEB服务器、应用服务、数据库等构成。对于C/S架构的应用系统有：由客户端/服务器构成的两层架构，由客户端、应用服务器、数据库服务器构成的三层架构，以及由客户端、N个应用服务器、数据库服务器等构成的多层架构等。对于一个B/S系统也有类似的情况，但是一般情况下B/S系统至少有一个WEB浏览器和一个WEB服务器。显然不同层次结构和不同的系统类别对于性能测试有着不同的影响。系统的架构和类别是测试采用不同技术的依据，同时对测试工程师也提出了不同的测试要求，对于B/S结构的系统要求测试工程师必须理解HTTP协议、HTML语言、Cookie的处理等等；而C/S结构的系统要求测试工程师必须掌握TCP/IP、WINSOCK、CORBA、COM、TUXEDO等分布式系统中通用的通信方法和原理。

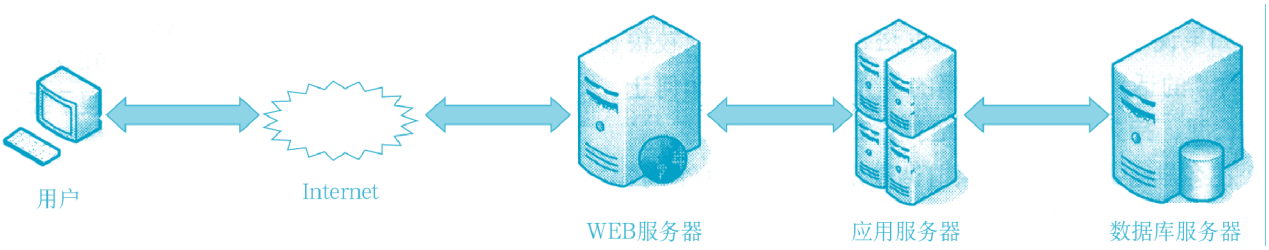

图1 一个典型的4层架构的企业应用系统

5.2 子系统性能需求分析

由于条件的限制，在进行性能测试时不可能对所有的功能点都进行性能测试。不同的功能点或者子系统对于性能的影响差别较大，例如在电力数配电生产管理系统中同时存在GIS子系统和FM子系统，GIS子系统含有反映数配电设备上实时传输过来的数据，对于性能有非常明确的要求，而FM是设备的台帐管理，要求数据的正确性，但对响应时间没有特别的要求。实时子系统和非实时子系统相结合系统，在测试前对子系统的性能需求进行分析，掌握系统不同的子系统对资源的不同需求，使得选择的测试功能比较接近系统的实际运行情况，又能体现性能测试的价值，反应出系统总体对于资源的需求。

6 测试策略的制定

在整个测试过程中，测试策略的制定直接关系到测试的成功与否。测试策略包括负荷策略、网络测试策略、业务策略、监控策略。在实际的应用系统的测试过程中，测试策略将是这些具体测试策略的不同组合。制定策略的基本依据是用户的需求和测试的目标。对希望评估系统状态的负载测试而言，其关键是估算测试强度；容量测试的关

键是确定用户可接受的性能点，即某一事务的响应时间的最大值，并且使系统达到该性能点；在压力测试中设定并发用户的初始值和并发虚拟用户数的增长规律是比较关键的。

6.1 负载策略

负载策略是指在测试时采用什么负载模式向系统增加压力。

固定负载在测试期间系统的负载基本保持不变，这种模式对于测试系统在特定的负载下的状态时非常有用，评估系统在正常用户状态下的系统响应情况就属于这种情况。固定负载模式下，在整个测试期间虚拟用户数量保持不变，每一个虚拟用户执行在测试脚本中定义的事务。当一个事务完成，虚拟用户马上重新执行该事务，在事务之间没有延时，在原指定的测试时间到达以后测试停止。为了提高测试的准确性，固定负载模式在测试开始时会预留一段时间作为系统预热阶段(WARMUP)。在WEB应用的测试中预热阶段特别重要，因为JSP和ASP.NET页面在初次请求时需要进行编译，第一次获得所请求页面的时间特别长，这段编译时间不能用来建立性能基准。预热期间系统还会产生各种缓存，应用程序的正常运行过程将使用这些缓存。WARMUP以确保在您开始测量之前每个页面均被请求了数次，以消除这些因素对测试带来的影响。同样的理由，在测试结束期间，通常需要安排一段较短的关闭时间(CLOSE DOWN)使得服务器有时间处理遗留的问题。整个固定负载测试可以分成三个阶段：预热阶段、计量阶段、关闭阶段。只有在计量阶段才将测试结果记录在结果文件中。图2给出了一种固定负载的模型。

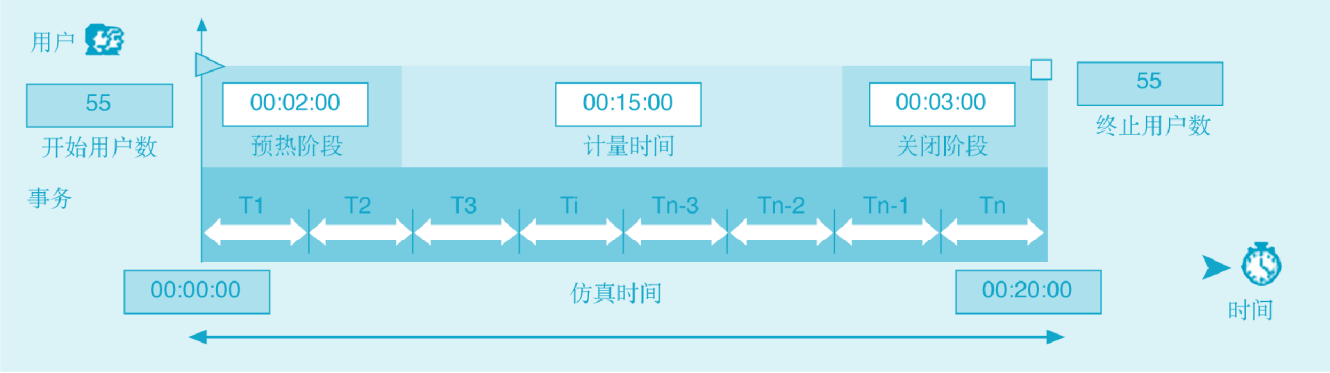

图2 固定负载测试示意图

增量式负载由两个阶段组成，第一个阶段为增长阶段，第二个阶段为固定负载阶段。增测试开始时测试工具仿真的不是用户指定的全部虚拟用户数，而仅仅是部分虚拟用户，然后按照用户指定的方式逐步递增虚拟用户数。增量式负载模式对于找到系统在什么样的负载情况下崩溃，或者查找在什么样的负载下系统不能在用户可接受的时间范围和错误范围内响应是十分有用的。增量式负载模式中，测试工程师需要指定开始虚拟用户数、最大虚用户数、虚拟用户数增长的时间间隔、每次增长的虚拟用户数等参数。增量式负载方式，测试的关键点在于何时能找到系统的极限点。图3为一个增量式负载的示意图。

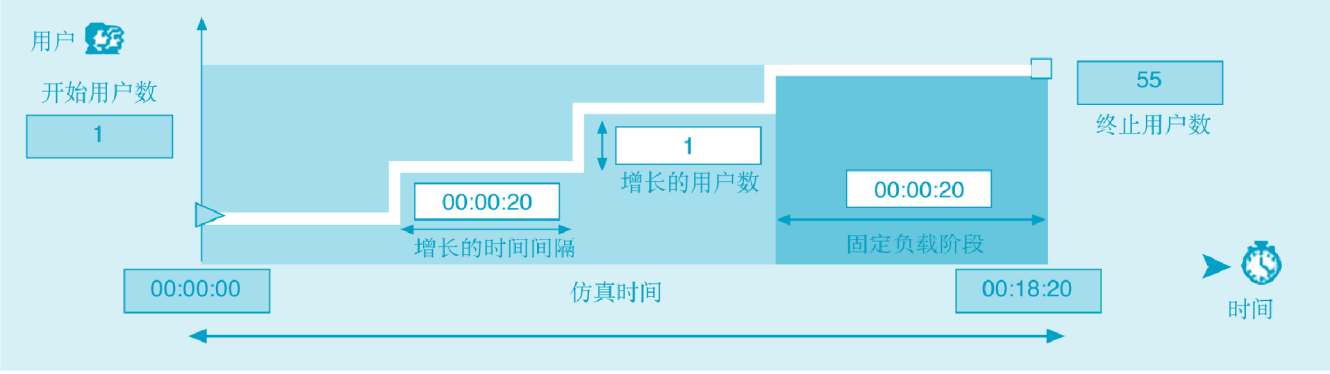

图3 增量式负载测试示意图

动态负载只要指定最大的虚拟用户数量，在测试期间可以增加或者减少系统的负载。如果希望在运行测试期间，能够经历不同的负载压力，并且能够对不同的负载水平加以控制，这种测试方式特别有用。

6.2 网络测试策略

网络作为应用系统运行的基础架构，和系统性能存在密切的关系。在测试应用系统性能的同时必须关注网络状态，测试网络是应用系统测试的一个重要辅助手段。例如网络带宽、延迟、负载和TCP端口的变化是如何影响的响应时间；利用网络应用性能分析工具，能够发现应用的瓶颈，知道应用在运行时的每个阶段网络的情况，在投产前预测应用的响应时间；利用测试工具或者网络仿真应用性能，根据最终用户在不同网络配置环境下的响应时间，用户可以根据自己的条件决定应用投产的网络环境。测试内容包括：运行应用时的监测网络、不同网络条件应用系统的状态、利用网络工具仿真系统应用。

网络监测可以帮助用户分析关键应用系统的性能，定位问题的根源是在客户端、服务器、应用程序还是网络。同时用户还可以借助网络监测自己较关心的网络问题：哪些应用程序占用大量带宽，哪些用户产生了最大的网络流量等等。

另外，可以采用网络测试工具产生网络背景流量，考察在不同网络条件下应用系统的运行状态。在恶劣的网络环境下，如果没有进行恰当的处理的话，可能系统会导致崩溃。在网络恢复以后应用程序是否能够自动恢复良好的状态，是考察系统健壮性的一个方面。

对于一些比较通用的应用，如HTTP、SMTP 、POP3、 FTP、流媒体等可以在应用部署前利用网络仿真工具如Chariot、Smartbits等进行仿真。

6.3 测试的监测策略

由于基于网络的大型应用系统本身结构的复杂性，为了达到有效的测试结果，在测试的同时必须对相关的资源加以监测。监测的对象包括客户端资源、WEB服务器资源、应用服务器资源、数据库服务器资源、网络、中间件状态等等。对于性能调优或者出现性能问题而进行的性能测试而言，测试的监测策略必须和测试的目的相结合，由于设计的需要监测的参数非常多，确定监测的重点变得非常重要，需要系统的开发人员或者维护人员的配合才能找到监测的重点。

6.4 业务策略

性能测试有两种不同业务功能测试方式：单一功能点测试和混合功能的性能测试。

在单一功能点的测试中，所有的虚拟用户都运行同样的事务脚本。其优点是能够精确定位不同的事务操作对于系统资源的需求和该事务的精确响应时间，但其缺点是和业务实际运行情况差距较大，因为不可能系统仅仅运行一个功能。单一功能点测试要求尽量选取几个最消耗系统资源的功能，并覆盖不同的交易形态，这样才有可能最大限度地检查出该部分的性能瓶颈。

混合功能测试在测试时按照系统实际运行时不同事务的比例进行多脚本混合测试。其优点是运行模式比较接近实际运行的情况，用户易于理解和接受；容易暴露服务器端应用处理的一些缺陷，如数据库死锁。其缺点是出现问题时较难定位，无法区分不同的功能点对系统资源的需求。由于混合功能压力测试需要完全模拟实际生产情况，所以业务的抽样选取相对比较复杂，通常需要进行当前业务量的收集和预测性能测试业务量，更重要的是确定业务的配比。系统运行的历史数据可作为确定业务分配的一个重要依据，一种简单的办法就是提取历史上交易量最大的10种业务作为代表，重新计算交易配比。应当指出，一些交易量很少但业务正常运行必不可少

的功能点必须加入到混合功能的性能测试中。

在测试过程中,应当根据用户的需求采用不同的方式,或者两种方式相结合。

7 性能强度目标的描述和估算

性能强度目标的描述必须是明确、无二意的、可度量的指标。例如系统能够支持200个用户的业务受理,业务受理的响应时间在5秒以内。有些业务自身的性能强度目标非常明确,例如119火警系统在最大负荷的情况下也必须保证系统在一个限定的时间内响应。有些业务没有直接给出性能强度目标,可以根据业务的历史数据进行估算。假设某一个业务系统,业务的性能强度可以按如下方式进行估算:

在估算时做如下假设(当然不同的业务可能会不太一样):

全年的业务量集中在8个月完成,每个月20个工作日,每个工作日8个小时;

● 采用80～20原理,每个工作日中80%的业务在20%的时间内完成,即每天80%的业务在1.6小时内完成;

● 依据历史数据,去年处理业务量为100万笔;

● 每年的业务增量为15%,考虑到业务发展的需要,测试需按现有业务量的2倍进行。

估算办法:

测试的全年业务总量:100 × 2=200(万笔)

平均每天的业务总量:200/(8 × 20)= 1.25(万笔)

平均每秒的业务总量:1.25 × 10000/(8 × 3600 × 80%)= 0.54(笔 / 秒)

8 性能测试的执行过程中的几个要点

8.1 系统配置的描述

基于分布式多层结构的应用系统,由于配置的多样性以及不同服务组件之间的关联性,性能问题很多是由于系统的配置不合理造成的,而不是编码引起的,在性能测试过程中对于系统配置的描述是非常重要的,在基于配置的调优测试更是如此。在描述各个组件的基础上必须描述他们之间的逻辑和拓扑关系。系统配置的描述包括以下几个方面:

● 连接到系统的用户数;

● 应用程序客户端计算机的配置情况(硬件、内存、操作系统、软件、开发工具等);

● 使用的数据库和Web 服务器的类型(硬件、数据库类型、操作系统、文件服务器等);

● 服务器与应用程序客户端之间的通信方式;

● 前端客户端与后端服务器之间的中间件配置和应用程序服务器;

● 可能影响响应时间的其他网络组件(调制解调器等)。

8.2 测试数据的准备

测试数据的准备包括两个方面:一个是存在应用中的数据,这些数据使得应用系统能够正常运行,并且比较接近在生产过程中的实际情况;另一方面测试过程中测试工具脚本需要大量的输入数据。

(1) 应用数据的准备

在测试环境中,系统的应用数据量越接近实际运行环境,测试效果越理想。如果应用程序访问数据库,则数据库应包含实际数目的记录,并且测试使用数据项应当随机但是有效。测试数据库太小,数据库服务器的缓存效果将产生不符合实际情况的测试结果。

应用程序除了要在数量上接近实际运行产生的数据外,数据的格式和长度、不同业务数据的逻辑关联性需要符合应用的要求。如果输入或访问数据的方式不符合实际情况,则测试结果也可能不符合实际情况。

对于一个已经有运行历史的系统而言,比较理想的是历史生产数据。由于数据在测试过程中可能被破坏,所以在应用数据的准备时要保证其足够且可重新生成。

(2) 测试参数的选用

测试参数主要包括以下的几种数据:

● 登录的系统账号和密码,对于一个账号只允许一次登录的系统必须为测试准备足够的账号和密码。账号的分布需要和系统运行的角色相匹配;

● 业务信息,指和业务相关的各种数据;

● 数据之间的约束关系,由于应用中各个数据之间存在一定的逻辑约束关系,要使应用系统能够正常运行,数据之间必须遵守这种约束关系。在应用系统自身的运行过程中,这种约束关系往往已经由客户端或者浏览器自动进行了校验和过滤,但在测试过程中可能会越过这些校验,为了保证程序的正常运行,在准备测试数据时必须由人工

保证这些数据的约束关系；

● 多IP地址的仿真问题，由于许多应用为提高系统的安全系数，禁止在同一个IP地址上有多个用户登录，测试工具往往使用一个测试代理机仿真多个用户登录系统。在这种情况下必须使用IP地址仿真技术，使得服务器认为不同用户来自不同的IP地址。

8.3 测试计划的制定和测试的执行

一份性能测试计划一般包括：测试目标、测试策略、测试环境与测试工具、测试资源配置、进度安排、测试终止准则等方面。测试终止准则是测试在什么时候终止的原则。测试终止准则一般要求描述精确无歧义。例如“所有待验证指标都达到”、“所有的测试用例都已经至少执行一次”。

在测试的执行过程中，测试的记录非常重要。记录的内容涵盖测试日志、测试环境、测试结果等。测试日志记录了测试过程中一些必要的信息，如测试中测试环境的变更、测试中遇到的技术难点以及解决办法等。在测试过程中，测试日志的记录是非常重要的，测试日志一方面作为测试过程中测试质量监控的一种有效手段，同时也是后期测试过程中知识积累的来源。测试环境的记录主要包括前面描述的系统配置。由于性能测试所涉及到的内容非常多，包括服务器监控文件、客户端监控文件、网络监控文件、测试驱动数据文件、测试工具产生的结果报告等等，涉及到的文件的种类多，格式也不相同。管理这些文件，使其达到版本的一致性是个非常重要的问题。在测试之前，应该做好规划和设计。一种有效的方法是借助于软件开发过程中的版本控制工具来实施测试数据和测试环境的管理。

测试结果分析，性能测试的结果一般包括响应时间、最大/最小并发数、失败的次数、正常连续运行的最长/最短时间、并发数与失败的关系等。包含预先定义的关键性能指标的数据、变化曲线的分析和结论。

9 测试时需要注意的几个问题

功能校验问题，性能测试工具偏向于多用户的并发操作，侧重于负载压力的产生和服务器的监控，忽略了负载压力情况下的功能不稳定问题。在负载压力测试过程中记录所有虚拟用户的操作及服务器的响应是当前负载压力测试技术发展的最大挑战，但测试过程中的附加记录会导致资源消耗、操作行为增加以及产生大量日志等问题，所以性能测试工具功能的正确性校验相对较弱，无法记录功能

性的错误。在测试过程中，可以同时使用功能测试工具作为补充进行系统正确校验，使得功能测试工具和性能测试工具能够发挥各自优点。

中间件的license、数据库的用户数有时会影响系统的性能。测试结果没有达到预期并不一定是被测对象的问题，可能是中间件的license不足或者是使用的数据库系统的并发用户数的限制所导致，在测试时需要综合分析。

参考文献

1 夏海涛. CS/CSS架构应用的软件性能测试模型分析. 测试员. 2004；2

2 Tony Northrup..NET Framework部署的性能调整. 微软技术网，http://www.microsoft.com/china/technet/itsolutions/net/Maintain/PTuneNET.asp

3 Capers Jones(美)，韩柯等译. 软件评估、基准测试与最佳实践. 机械工业出版社，2003

4 Brian Marick(美)，韩柯等译. 软件子系统测试. 机械工业出版社，2003

（收稿日期：2005-06-16）

作者简介：

蔡立志，硕士，高级工程师，专业从事基于网络的大型应用软件系统的性能测试和调优研究工作。近期，主持设计和完成了“上海市公务网”、“上海电力输配电系统”等系统的性能调优测试。

杨根兴，博士，研究员，国内软件质量与测试的知名专家，目前担任上海市软件质量管理联盟秘书长、上海市软件质量专业职业资格专家委员会副主任、中国软件测评机构联盟副理事长、中国软件测试外包促进会执行秘书长。近期，主持和完成了多项国家863和上海市科技发展基金项目。