

# 软件测试教程 性能测试概述

本节的主要目的是带大家了解什么是性能测试，性能测试的关键指标及其含义，性能测试的类型，以及性能测试工程师进行性能测试的流程以及关键点是啥。

常见的性能问题

为什么要进行性能测试

如何确定系统的性能需求

常见的性能指标

哪些人员关注性能

性能测试的关注领域

何时进行性能测试

## 常见的性能问题



系统内部以及软件的代码实现

- 1, 资源泄漏，包括内存泄漏。
- 2, CPU使用率达到100%，系统被锁定等。
- 3, 线程死锁，阻塞等造成系统越来越慢。
- 4, 查询速度慢，或者列表的效率低。
- 5, 受外部系统影响越来越大

## 为什么要进行性能测试

- 获取系统性能的指标，作为性能指标的基准
- 验证系统的性能指标是否达到要求（性能需求）

- 应用程序是否能够满足系统要求的各中性能指标
- 应用程序是否能处理预期的用户负载并有盈余能力
- 应用程序是否能处理业务所需要的事务数量
- 在预期和非预期的用户负载下，应用程序是否稳定
- 是否能确保用户在真正使用软件时获得舒服的体验
- 发现系统的性能瓶颈，内存泄漏等问题。
- 系统正常工作情况下的最大容量。
- 帮助系统运维部门能更好的规划硬件配置

## 性能测试实施的流程

- 分析性能测试需求
- 根据性能测试的目标，设计性能测试的场景
- 开发性能测试场景和性能测试脚本
- 分析性能测试报告
- 根据性能测试报告排查和定能系统的性能瓶颈

### 如何确定性能测试的需求

从以下两个方面去确定系统性能测试的需求。

#### 1. 关键性能指标分析

在进行性能测试需求之前，一定要清楚的知道系统的性能需求，不能过于简单或者过于模糊的描述性能需求，系统性能的需求必须通过具体数据进行量化，也就是人们常说的性能指标。性能指标一旦量化，就可以度量，才具备可测试性（可验证性），即能确认系统的性能指标是否符合设计的要求，是否符合客户的需求。

所以一般的性能测试，需要明确而量化的性能指标。请看下面的例子，

【例1】某电商交易系统，业务要求支持2亿用户（同一时刻1%的用户在线），每天支持2000万次交易量（交易的时间集中在早上8点到凌晨2点），交易响应时间要求在1s中之内。

此外，高峰期系统的处理能力要求是平均值的3倍；

本地交易响应时间正常低于1s，在高峰时可以高于1s，但是要在3s内。

正常及交易成功率100%，在高峰期不能低于99.9%

这样的业务要求，作为测试人员，如何转化为可以性能测试可以验证的指标呢？

（1）业务支持2亿用户

数据库要支持这个量级的数据存储。同时按1%的用户在线，那么意味着同时有200万用户在线，那系统就要支持这个数量级用户的各种增删改查操作。

（2）每天支持2000万次交易

根据题意，每天交易的时间在18小时，折算成每秒需要完成多少次交易，即 $20000000 / (18 * 60 * 60) = 309$ 次/s

.....

一组清晰定义的预期性能指标值，是性能测试的基本要素。

如果是一个全新的应用系统，无法确定具体的性能指标怎么办？

（1）可以通过“基准测试”，获取性能指标数据 （2）从业务，用户体验，竞品的性能指标信息来定义性能指标的数据

最终用户的体验：如2-5-10原则。

商业需求：一个基本的商业需求是软件产品的性能比“竞争对的产品性能要好，至少不比它差”。

技术需求：从技术角度看系统的性能，例如CPU的使用率不能超过70%等；

标准要求：行业标准定义了某些软件的性能指标，相关的软件必须遵守这些。

## 2. 关键业务分析

系统如果出现问题，看似整个系统无法正常运行，实际往往是因为系统运行时某一个环节出了问题，系统的性能问题也是一样，如果在某一些业务功能上不出现性能问题，那么系统就不会出现性能问题，而这些业务功能就是系统性能测试的关键业务所在。

根据帕雷托法则 (pareto Principle)，系统中各个功能的使用频率是不相同的，有20%的功能是常用的功能，用户80%以上的时间都在使用这些功能，这些功能就是性能测试当中我们测试人员需要关注的。

比如淘宝这样的购物平台，首页肯定是用户访问量最多的页面，其次，用户进来之后要搜索自己喜欢的商品，输入关键字之后，相关的产品就会根据人气，价格等排序并且展示出来。

所以，对于淘宝这样的购物平台，“打开首页”，“搜索”功能等操作就会被选择为性能测试的关键业务功能。

总而言之，在性能测试中，我们测试人员首先要了解哪些业务功能为用户最常使用的，以此来确定性能测试的关键业务功能。

那么仅仅依靠功能的使用频率来确定性能测试的关键业务吗？答案是否定的。我们还要看这个业务功能后面的计算量，它是否耗时，对系统资源的消耗。这里同学们可以想想为什么？

例如，淘宝购物当中95%的用户搜索商品，5%的用户才会提交订单，完成支付。

提交订单和支付这两个功能计算更耗时，计算量更大些，它关联着较多相关数据的读写，支付会涉及到外部接口（支付宝），所以这两个功能就是我们性能测试的关键业务。

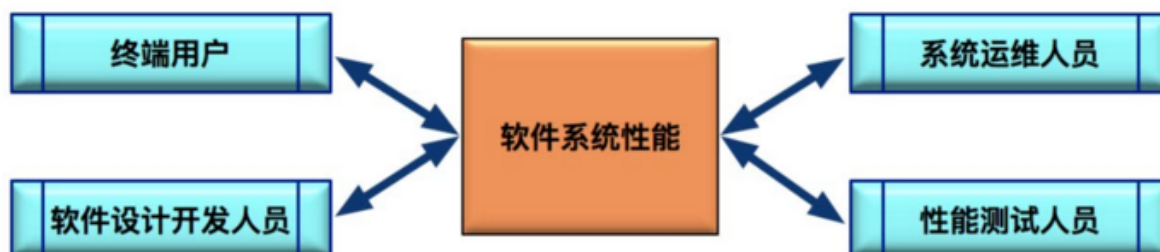
综上，要确定性能测试的关键业务要从业务功能的使用频率和功能的计算量，资源的消耗程度来决定，确定好关键业务之后，我们在进行性能测试的时候只要对关键的业务进行测试用例的设计，系统的性能测试脚本就会基于这些关键的业务进行开发。

## 常见的性能指标

确定了系统的关键业务之后，我们进行性能测试要关注这些业务功能的哪一些指标呢？

- 系统/事务的平均响应时间
- 事务处理效率TPS (Transaction Per Second)
- 吞吐率
- 每秒点击次数 (Hits Per Second)
- 服务器资源占用情况，内存和CPU使用率
- 软、硬件配置是否合适 (容量规划/硬件选型)

衡量软件性能的四个维度：



- 软件开发人员

系统架构：架构设计是否合理？

数据库设计：数据库设计是否存在问题？

算法：核心算法是否高效

设计和代码：系统中是否存在不合理的线程同步方式和不合理的资源竞争？

- 系统运维人员（操作系统、网络、服务器等等）

资源利用率：应用服务器和数据库资源使用状况合理吗？

系统容量：系统最多能支持多少用户的访问？系统最大的业务处理量是多少？

系统稳定性：系统是否能支持7X24小时的业务访问？

系统可扩展性：系统是否能够实现扩展？系统性能可能的瓶颈在哪里？更换哪些设备能够提高系统性能？

- 用户

从终端用户的维度来讲，软件性能表现为用户进行业务操作时的主观响应时间，具体来讲就是从用户在界面上完成一个操作开始，到系统把本次操作的结果以用户能察觉的方式全部展示出来的全部时间。

对终端用户来说，这个时间越短越好。

系统稳定性也是用户关注的重点

- 性能测试人员

以上所有层面都需要关注

是否能够发现系统中存在的瓶颈？

是否真实有效的评估系统性能能力？

#### WHERE:(关注领域)

- 能力验证

性能测试中最简单也是最常用的一个应用领域，主要关心的是“在给定的条件下，系统是否具有预期的表现”。

- 规划能力

规划能力领域与能力验证领域有些不同，其主要关心的是“应该如何才能使系统具有我们要求的性能能力”或是“在某种可能发生的条件下，系统具有如何的性能能力”

- 性能调优

主要应用于对系统性能进行调优。一般来说，性能调优活动会和其他领域的活动交杂在一起。是一种在开发阶段和测试阶段都可能会涉及到的性能测试应用领域。

- 发现缺陷

主要在该应用领域的主要目的是通过性能测试的手段来发现系统中存在的缺陷

误区：

- 1、性能测试独立于功能测试
- 2、性能测试就是并发用户测试
- 3、在开发环境测试一下性能就可以了

#### WHEN:（何时测试）

- 系统的功能稳定之后，即功能测试中后期

误区：

1、性能测试在其他测试完成后，测试一下就可以了

## 概念和术语介绍

性能测试是一项综合性的工作，致力于暴露性能问题，评估系统性能趋势。性能测试工作实质上是利用工具去模拟大量用户操作来验证系统能够承受的负载情况，找出潜在的性能问题分析并解决；找出系统性能变化趋势，为后续的扩展做准备。

一般地，它主要是针对系统的性能指标制定性能测试方案，执行测试用例，得出测试结果来验证系统的性能指标是否满足既定值。性能指标里包括系统各个方面的能力，如系统并发处理能力，系统响应时间，批量业务处理能力等等。

### 并发用户数

并发用户会对系统造成压力，首先对系统用户数，在线用户数，并发用户数做一个区分。

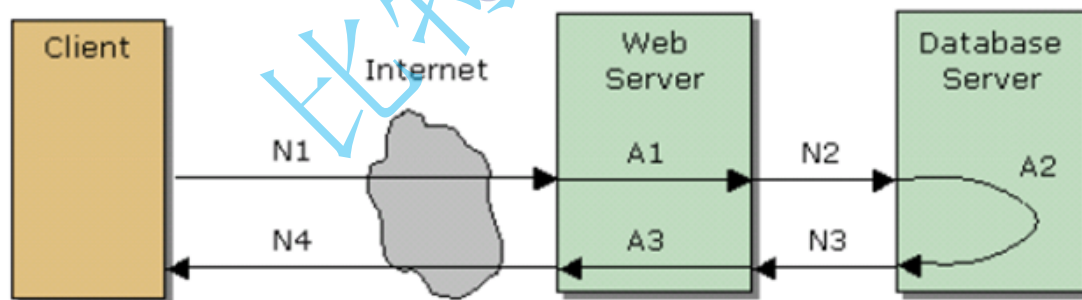
系统用户数：简单地说就是该系统的注册用户数。例如，BestTest论坛里存在6666个注册用户，他们可以是活跃的，也可以是僵尸的。

业务层面的并发用户数：指的是同时向服务器发送请求的用户数量。

后端服务器层面的并发用户数：指的是同时向服务器发送请求的请求数量。

### 响应时间/平均响应时间 (RT/ART)

从用户视角来考虑，响应时间反映了完成某个操作所需要的时间，标准定义是，应用系统从发出请求开始，到客户端接收完所有的字节数据所消耗的时间。



所以，响应时间分为前端展示时间和系统响应时间两部分。

前端展示时间指的是客户端收到服务器返回的数据后渲染前端页面，所耗费的时间。

系统的响应时间，分为web服务器，应用服务器，数据库服务器，等各种服务器之间通信和处理请求的时间。

所以严格的说，响应时间应该包含两层含义：用户主观感受时的时间定义，技术层面的标准定义。

对于软件服务器端的性能测试肯定要采用标准定义；

对于前端性能评估，则应该采用用户主观感受的时间定义；

### 事务响应时间 (Transaction Reponse Time)

每秒完成的事务数，通常指每秒成功的事务数，性能测试中重要的综合性性能指标。

这里的一个事务是一个业务度量单位，是指一组密切相关的子操作的组合。比如，一笔电子支付操作，后台处理的时候可能需要经过会员系统，账务系统，支付系统，银行系统等，这就是一个关于支付事务里面包含的操作。而对于用户，往往也只关注整个支付花费了多长时间。

### 每秒事务通过数 (Transaction Per Second)

TPS 是指每秒系统能够处理的事务数。它是衡量系统处理能力的重要指标。

当压力加大时，TPS曲线如果变化缓慢或者有平坦的趋势，很有可能是服务器开始出现瓶颈了。如果环境没有发生大的变化，对于同一系统会存在一个最大处理事务能力，它并不随着并发用户的增减而改变。

YB地铁检票机：

只有10台进站检票的机器，1台机器1秒能进1个人

并发用户数为5，则TPS为5

并发用户数为10，则TPS为10

并发用户数为100，则TPS仍为10

### 点击率 (Hit Per Second)

每秒点击数代表用户每秒向Web 服务器提交的HTTP请求数。点击率越大，服务器压力越大。

这里的点击并不是鼠标的一次点击，一次点击可能有多次HTTP请求。

### 吞吐量 (Throughput)

这里的吞吐量以单位时间为度量衡量；

单位时间内系统处理的客户请求的数量，直接体现软件系统的性能承载能力，一般来说用 Requests/second, Pages/Second, Bytes/Second, 从业务的角度，也可以用访问人数/天或是处理的业务数/小时来衡量，从网络设置的的角度来说，也可以用字节数/天来衡量。

### 思考时间 (Think Time)

指模拟正式用户在实际操作时的停顿间隔时间，从业务的角度来讲，思考时间指的是用户在进行操作时，每个请求之间的间隔时间。

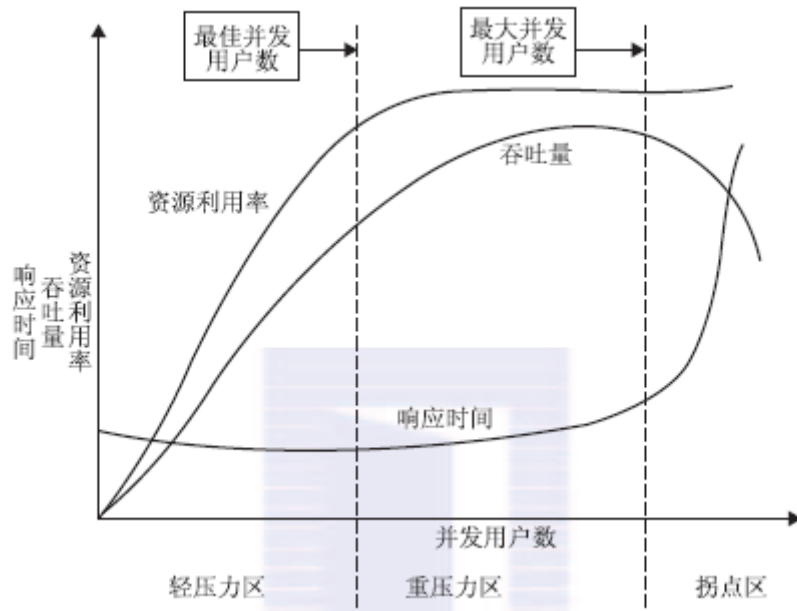
### 资源利用率

不同系统资源的使用情况。包含CPU，内存，硬盘，网络等。

## 性能测试模型

前面讲解了一些性能测试的概念和术语，现在来深入的理解一下。

### 曲线拐点模型



1) X轴代表并发用户数，Y轴代表资源利用率、吞吐量、响应时间。X轴与Y轴区域从左往右分别是轻压力区、重压力区、拐点区。

2) 随着并发用户数的增加，在轻压力区的响应时间变化不大，比较平缓，进入重压力区后呈现增长的趋势，最后进入拐点区后倾斜率增大，响应时间急剧增加。

3) 接着看吞吐量，随着并发用户数的增加，吞吐量增加，进入重压力区后逐步平稳，到达拐点区后急剧下降，说明系统已经达到了处理极限，有点要扛不住的感觉。

4) 同理，随着并发用户数的增加，资源利用率逐步上升，最后达到饱和状态。

5) 最后，把所有指标融合到一起来分析，随着并发用户数的增加，吞吐量与资源利用率增加，说明系统在积极处理，所以响应时间增加得并不明显，处于比较好的状态。但随着并发用户数的持续增加，压力也在持续加大，吞吐量与资源利用率都达到了饱和，随后吞吐量急剧下降，造成响应时间急剧增长。轻压力区与重压力区的交界点是系统的最佳并发用户数，因为各种资源都利用充分，响应也很快；而重压力区与拐点区的交界点就是系统的最大并发用户数，因为超过这个点，系统性能将会急剧下降甚至崩溃。

## 地铁模型

假设：

# 某地铁站进站只有3个刷卡机。

# 人少的情况下，每位乘客很快就可以刷卡进站，假设进站需要1s。

# 乘客耐心有限，如果等待超过30min，就暴躁、唠叨，甚至放弃。

场景一：只有1名乘客进站时，这名乘客可以在1s的时间内完成进站，且只利用了一台刷卡机，剩余2台等待着。

场景二：只有2名乘客进站时，2名乘客仍都可以在1s的时间内完成进站，且利用了2台刷卡机，剩余1台等待着。

场景三：只有3名乘客进站时，3名乘客还能在1s的时间内完成进站，且利用了3台刷卡机，资源得到充分利用。

场景四：A、B、C三名乘客进站，同时D、E、F乘客也要进站，因为A、B、C先到，所以D、E、F乘客需要排队。那么，A、B、C乘客进站时间为1s，而D、E、F乘客必须等待1s，所以他们3位在进站的时间是2s。

场景五：假设这次进站一次来了9名乘客，有3名的“响应时间”为1s，有3名的“响应时间”为2s（等待1s+进站1s），还有3名的“响应时间”为3s（等待2s+进站1s）。

场景六：如果地铁正好在火车站，每名乘客都拿着大小不同的包，包太大导致卡在刷卡机堵塞，每名乘客的进站时间就会又不一样。刷卡机有加宽的和正常宽度的两种类型，那么拿大包的乘客可以通过加宽的刷卡机快速进站（增加带宽）。

场景七：进站的乘客越来越多，3台刷卡机已经无法满足需求，为了减少人流的积压，需要再多开几个刷卡机，增加进站的人流与速度（提升TPS、增大连接数）。

场景八：终于到了上班高峰时间了，乘客数量上升太快，现有的进站措施已经无法满足，越来越多的人开始抱怨、拥挤，情况越来越糟。单单增加刷卡机已经不行了，此时的乘客就相当于“请求”，乘客不是在地铁进站排队，就是在站台排队等车，已经造成严重的“堵塞”，那么增加发车频率（加快应用服务器、数据库的处理速度）、增加车厢数量（增加内存、增大吞吐量）、增加线路（增加服务的线程）、限流、分流等多种措施便应需而生。

## 性能测试方法介绍

- 代码级别的性能测试
- 基准性能测试
- 并发测试
- 压力测试
- 配置测试
- 可靠性测试

### 代码级别的性能测试

代码级别的性能测试是指在单元测试阶段就对代码的时间性能和空间性能进行必要的测试和评估，以防止底层代码的效率问题在项目后期才发现的尴尬。

### 基准性能测试

系统的第一个版本，研发团队团队也不清楚系统的性能能达到怎样的水平，这时进行的性能测试，其目标是获得系统标准配置下，有关性能指标数据，作为将来性能改善的基准，这种测试称之为“性能基准测试”。

性能基准测试是通过性能测试获取系统的性能指标，建立一个性能基准，作为以后性能测试的参考。

系统进行性能基准测试可以在系统开发的较早的阶段发现性能问题。

### 并发测试 (Load Testing)

并发测试，指的是在同一时间内，同时调用后端服务，期间观察被调用服务在并发情况下的行为表现，目的是为了发现如资源竞争，死锁等问题。

这里的并发测试指的是严格的并发测试，也就是所有用户在同一时刻向后端服务器发送请求。

### 压力测试 (Stress Testing)

压力测试，通常指的是后端压力测试，一般采用后端性能测试的方法，不断对系统施加压力，验证系统长期处于临界饱和阶段的稳定性以及性能指标的变化。试图找到系统处于临界状态时的主要瓶颈点。

要更快的找出系统的性能瓶颈，一般会加大负载，甚至将负载一直增加上去，达到极限负载。这时候CPU、内存等处于极限状态，测试系统在极限状态下长时间运行是否稳定。确定系统是否稳定的指标包括TPS，RT，CPU Using，Mem Using等。

在不断地加压过程中，一旦系统出现拐点（从量变到质变），系统可能就会出现崩溃，揭露高负载下系统的问题，例如资源竞争、同步问题、内存泄漏等；然后逐渐较少压力，观察瘫痪的系统是否可以自愈。

### 配置测试 (Configuration Testing)

系统性能的好坏，不仅仅取决于软件自身的设计和实现，也取决于软件运行所依赖的硬件，网络环境。为了达到系统性能指标要求，就需要调整系统的硬件配置，如增加服务器或者服务器集群来达到更高的性能。这时，会在不同配置的情况下，来测定系统的性能指标，从而决定在系统部署时采用什么样的软，硬件配置。

- 1，通过基准测试，建立性能基线；
- 2，在此基础上，调整配置；
- 3，基于同样的性能基准测试，观察不同配置下系统性能的差异，目的是找出特定压力模式下的最佳配置；

这里注意一下，配置是一个广义的配置的概念，包含了以下多个层面的配置：

操作系统的配置

应用服务器的配置

数据库配置

JVM配置

网络环境的配置

.....

### 可靠性测试 (Reliability Testing)

验证系统在常规负载模式下长期运行的稳定性。

在一定的软硬件环境下，长时间运行一定的负载，确定系统在满足性能指标的前提下是否运行稳定。与压力测试不同的是系统的负载并不是处于极限的状态下。重点是满足性能要求的情况下，系统的稳定性，比如响应时间，TPS是否稳定。

## 性能测试实施与管理

性能测试前期准备

测试工具引入

测试方案

测试设计与开发

测试执行与管理

测试分析与调优

## 性能测试前期准备

- 系统基础功能验证

确保当前需要进行测试的应用系统具备了进行性能测试的条件

确保当前进行性能测试的应用系统版本已经稳定；

必须保证性能测试之前至少进行了一轮的系统功能覆盖测试，且性能测试选取的业务功能正常。

- 组建测试团队

确定团队内角色的构成，以及确定人员的技能

角色	职责	技能
测试经理	1、和用户等项目干系人交互，确保测试的外部环境 2、制定测试计划 3、监控测试进度 4、发现和处理测试中的风险	1、计划执行和监控能力 2、风险意识和能力 3、外交能力和灵活变通的能力

支持角色（系统）	系统支持，协助解决测试工程师无法解决的系统问题	专职的系统管理员
支持角色（网络）	网络方面的支持，协助测试工程师解决网络方面的问题，在必要时为测试分析角色提供网络方面的分析支持	专职的网络管理员
支持角色（数据库）	数据库方面的支持，在必要时为测试分析角色提供数据库方面的支持	专职的DBA
测试工程师	测试数据的准备	性能测试工程师

- 测试工具需求确认

根据被测系统的了解，评估性能测试工具所应具备的功能

被测系统环境	测试工具功能需求建议
操作系统环境	测试工具是否能运行在本操作系统上
测试工具是否能支持对本操作系统的监控	
应用服务器环境	测试工具能否支持对本应用服务器的监控
数据库环境	测试工具能否支持本数据库的监控
应用使用的协议	测试工具能否支持需要进行录制和产生负载的协议
网络环境	是否需要测试工具支持防火墙
是否需要测试工具支持负载均衡	
测试管理支持	测试工具是否能够提供方便的测试结果分析和管理

# 测试工具引入

- 工具选择

选择适合项目的性能测试工具，商业工具或者是自行开发的工具，Loadrunner, jmeter

- 工具应用技能培训

为项目组的相关参与者进行测试工具的应用技能培训，以使参与者能够具备测试需要的技能

- 确定工具应用过程

确定测试工具在测试中的具体应用范围，并不是“工具无所不能”，哪些工作使用工具完成，哪些无法使用工具完成

## 性能测试方案

### 1、调研测试需求

- 测试业务范围

关键的、常用的、压力较大的、有代表性的、不宜过多

- 测试环境

硬件环境：主机型号、配置...

软件环境：操作系统、数据库...

网络环境：带宽？交换机？防火墙？

- 测试目的：

上线前性能测试？对比性能测试？调优？查找缺陷？

- 性能指标

业务性能指标：

一般步骤是首先从需求和设计中分析出性能测试需求，性能测试需求的来源是多方面的，需求文档（非功能需求的描述）、设计文档、客户备忘录、历史经验的积累等等

系统性能指标：

cpu使用率、内存使用率

注：实际测试时，需要监控许多其他的性能指标：数据库、服务器系统、网络，用于定位问题

### 2、测试策略和测试资源需求

测试工具、测试方式、测试执行

人力资源：明确所需的人员类型（性能测试负责人、性能测试工程师、应用工程师、系统工程师、数据库工程师、网络工程师）、由何方提供、明确职责分工

硬件资源：明确测试时所需的硬件资源(测试工具要求机器的内存，磁盘空间)

### 3、性能测试计划

任务	起止时间	责任人	任务说明
1性能测试方案			
2测试环境准备	预计2天		
2.1应用软件部署、检查			
2.2数据库基础数据导入			
3测试脚本、测试数据	预计1天		
3.1脚本录制			
3.2脚本参数化			
3.3脚本调试			
4测试执行	预计5天		
4.1压力测试、系统调优			
4.2 负载测试			
5编写性能测试报告	预计1天		

## 性能测试设计与开发

### 1.测试环境设计

性能测试的结果与测试环境之间的关联性非常大，无论那种性能测试，都必须首先确定测试的环境，包括系统的软/硬件环境、数据库环境等等（50万条数据和空数据库执行操作的时间显然是不同的）

### 2.测试场景设计（测试用例）

测试场景模拟的一般是实际业务进行的一个剖面，其包括业务、业务比例、测试指标的目标、测试过程中需要监控的性能计数器

场景名称	混合交易性能表现测试			
目的	通过多个交易的混合场景运行，获得各个交易在混合情况下的平均响应时间，配合开发人员调优，同时监控服务器和数据库资源。在一定时间内经过调优确定参数后，获得性能表现。			
结果	获得各个交易的平均响应时间和监控数据			
配置信息	并发用户数分别为100、200、300 运行30分钟，用户一起增加，无思考时间，无迭代等待时间			
交易名称	百分比	100	200	300
Tran1	30	30	60	90
Tran2	30	30	60	90
Tran3	40	40	80	120

### 3.脚本开发

对测试场景进一步细化，一般包括测试类型、测试内容描述、前置条件、业务操作序列、参数化需求、验证点等

### 4.脚本和辅助工具开发

测试脚本是对业务操作的体现，一个脚本一般就是一个业务过程的描述，脚本的开发通常都基于“录制”，然后对脚本进行完善，以满足在性能测试中顺利使用。

辅助工具开发一般基于性能工具无法满足，或者是获取特定资源需要使用。

## 性能测试执行与管理

### 1.建立测试环境

搭建需要的测试环境，需要多个团队角色的参与，包括硬件、软件系统环境的搭建、数据库环境建立、应用系统的部署、系统设置参数的调整、以及数据库环境准备

### 2.部署测试脚本和测试场景

脚本和测试场景的部署最终需要保证场景与设计的一致性，保证需要监控的计数器都已经部署好了相应的监控手段。

### 3.执行测试和记录结果

可以依靠工具完成，对于工具不支持的，可以采用系统自带工具或自行开发工具解决。测试结果是最后分析的基础。

测试交易	Tran1	
测试时间	测试类型	场景说明
2006-05-06 14: 10—14: 20	单交易多用户测试	100用户 执行20 分钟; 模拟系 统延时 1s 无思 考时 间, 无 间隔时 间
测试结果描述及存在瓶颈		
测试结果: 每秒钟交易数7.19、事物平均响应时间5.56、交易成功率90% 存在的问题 虚拟用户渐进增加。虚拟用户少于40个时, 系统性能良好, 随着虚拟用户增加, 响应时间基本不变, TPS增加; 超过40个用户时, 响应时间变慢, TPS基本不变。部分事物因为超时失败。		

测试交易	Tran1	
测试环境配置及性能调优点		
应用系统服务器		
系统配置参数	日志	其他
Socket 连接池连接数=100	级别调试	
数据库服务器		
Oracle参数	索引	表分析
Pga_aggregate_target=524288000 Processes=1000、DB_CACHE_SIZE_2046m Sga_max_size=4094m		
调优情况及效果	改进状态跟踪和负责人	
资源监控明细		
应用服务器	DB服务器	模拟系统服务器
CPU: 20.3%;	CPU: 31.5%;	CPU: 20.3%;
测试场景及测试脚本	测试结果存放目录	
Fund_0701_Scensrio_base	Result\res_0701_vu100_15m	
备注:		

## 性能测试分析与调优

测试结果分析是最难的部分。是一个灵活的过程，每次性能测试结果的分析都需要测试分析人员具有相当程度的对软件性能、软件架构和各种性能测试指标的了解，性能测试分析需要借助各种图表。

通用方法之一就是“拐点分析的”方法。关注性能表现上的“拐点”，获得“拐点”附近使用情况，定位处系统的性能瓶颈。

## 性能测试报告

### 测试目标

指标要求:本次测试预期达到的性能要求。(TPS,ART, 交易成功率, 并发数等)

### 测试概要描述

系统结构

测试时间

测试地点和测试人员

工具和环境



测试过程简介

**测试结果和数据**

**测试结论**

测试遗留问题

建议

测试结论限制

比邻作业课  
万兴PDF