构建高性能J2EE应用的五种核心策略 PDF转换可能丢失图片或格式，建议阅读原文 https://www.100test.com/kao_ti2020/245/2021_2022__E6_9E_84_E5_BB_BA_E9_AB_98_E6_c104_245767.htm 对于J2EE，我们知道当开发应用时，在架构设计阶段的决定将对应用的性能和可扩展性产生深远的影响。现在当开发一个应用项目时，我们越来越多地注意到了性能和可扩展性的问题。应用性能的问题比应用功能的不丰富问题往往更为严重，前者会影响到所有用户，而后者只会影响到碰巧使用该功能的那些用户。 作为应用系统的负责人，一直被要求"要少花钱多办事"----用更少的硬件，更少的网络带宽，以及更短的时间完成更多的任务。J2EE通过提供组件方式和通用的中间件服务是目前首选的最优方式。而要能够构建一个具有高性能和可扩展性的J2EE应用，需要遵循一些基本的架构策略。 缓存(Caching) 简单地说，缓存中存放着频繁访问的数据，在应用的整个生命周期中，这些数据存放在持久性存储器或存放在内存中。在实际环境中，典型的现象是在分布式系统中每个JVM中有一个缓存的实例或者在多个JVM中有一个缓存的实例。 缓存数据是通过避免访问持久性存储器来提高性能的，否则会导致过多的磁盘访问和过于频繁网络数据传输。 复制 复制是通过在多台物理机器上创建指定应用服务的多个拷贝来获得整体更大吞吐效率。理论上看，如果一个服务被复制成两个服务，那么系统将可处理两倍的请求。复制是通过单一服务的多个实例的方式从而减少每个服务的负载来提高性能的。 并行处理 并行处理将一个任务分解为更为简单的子任务，并能够同时在不同的线程中执行。 并行处理是通过利用J2EE层执

行模式的多线程和多CPU特点来提高性能。与使用一个线程或CPU处理任务相比，以并行方式处理多个子任务可以使操作系统在多个线程或处理器中进行分配这些子任务。 异步处理 应用功能通常被设计为同步或串行方式。异步处理只处理那些非常重要的任务部分，然后将控制立即返回给调用者，其他任务部分将在稍后执行。 异步处理是通过缩短那些在将控制返回给用户之前必须处理的时间来提高性能的。虽然都做同样多的事情，但是用户不必等到整个过程完成就可以继续发出请求了。 资源池 资源池技术使用的是一套准备好的资源。与在请求和资源之间维持1：1的关系的不同，这些资源可被所有请求所共享。资源池的使用是有条件的，需要衡量下面两种方式的代价： A、维持一套可被所有请求共享资源的代价 B、为每个请求都重新创建一个资源的代价 当前者小于后者时，使用资源池才是有效率的。 100Test 下载频道开通，各类考试题目直接下载。详细请访问 www.100test.com