Atitit 代码之美 ati文摘

目录

[1.1. 4.5 可靠性 1](#_Toc30736)

[1.2. 监测 1](#_Toc2434)

[1.3. 4.6.1 动态重配置 2](#_Toc28647)

[1.4. 4.6.2 热交换 2](#_Toc19642)

From smp ver

## **4.5 可靠性**

可靠的代码能够持续良好地执行而不会遇到问题。程序很少崩溃。你可以想像到，在火

星漫步者上的代码必须是极其可靠的，因为要进行现场的维护将是非常困难的事情。不过，

MER 任务也希望任务控制中心所使用的基地软件同样是可靠的。一旦任务开始进行，没有

人希望因软件问题而中断整个操作。

正如前面所提到的，CIP 项目采用了几种方法来确保系统内在的可靠性：

●

遵循工业标准和最佳实践，包括 J2EE。

●

在任何可能的地方都使用那些已被验证过的 COTS 软件，包括来自于权威中间件

制造商的商业应用程序服务器。

●

使用带有模块化服务的面向服务架构

●

实现简单、直接的中间件服务

我们通过额外的技术来进一步提高可靠性：服务日志和监测。这些功能不但在调试小型程序

时是很有用的，而且在跟踪大型应用程序的运行行为上也能够起到重要的作用。

## 监测

日志使我们能够通过研究服务在一段时期内的操作来分析服务的性能。与日志方法不同

的是，运行时监测在定位问题及其成因中是最有帮助的，它能有助于我们看到服务当前的运

行情况。它使我们能够进行动态调节以改善性能或者防止任何潜在的问题。正如在前面提到

的，对于任何大型应用程序来说，监测运行时的行为对于应用程序的成功来说是非常重要的。

在前面给出的代码中包含了对在全局静态对象中保存的性能数据进行更新的语句，这些

全局对象是通过 cacheStats 和 fileStats 这两个域来引用的。中间件监测服务将根据请求来查

询这个性能数据。虽然并没有给出这些域所引用的全局对象，但是你应该能够想像出它们所

包含的内容。关键是收集有用的运行时性能数据并不是件复杂的任务。

我们把 CIP 中

## 4.6.1 **动态重配置**

大多数中间件服务都有关键的运行参数。例如，正如我们在前面所看到的，流服务按照

块的方式下载文件，因此它就有一个有关块大小的参数。我们不是把块大小硬编码到程序中，

而是把这个值放在一个参数文件中，服务在每次启动的时候都会去读取这个参数文件。这个

操作发生在流服务提供者 bean 被加载的时候（“日志”一节代码的第 3-5 行）。

## 4.6.2 **热交换**

热交换（Hot Swapping）是我们在 CIP 中间件中采用的商业应用服务器的重要功能之一。

这使得我们可以用一个中间件服务来替换另一个正在运行的中间件而无需先停止它（以及

CIP）。

当我们需要在修改参数值之后强制服务重新进行加载时，可以使用热交换，我们可以通

过重新加载一次服务来实现。当然，像流服务这样使用有状态会话 bean（文件读取器和文

件写入器 bean）的服务将会丢失所有的状态信息。因此，只有当服务处于“安静”时期，

我们才可以对服务进行热交换，因为此时我们知道服务目前并没有被使用。对于流服务来说，

我们可以通过中间件监测工具的 Files 标签页（参见图 4-8）来知道何时处于这种情况。

热交换在大型企业应用程序的运行环境中是最有意义的，在这些环境中，在替换部分程

序的同时保持其他部分仍然运行是一项很重要的需求。而对于小型程序来说，最好的方式就

是把程序送回去修改