一种适用于构件系统的软件抗衰技术框架*)

郑贤福　杨　群　许满武

(软件新技术国家重点实验室　南京大学计算机科学与技术系　南京 210093)

摘　要　近年来,软件抗衰技术已被证实是解决软件衰老问题的有效途径。本文针对构件系统特点,将 Micro-Reboot 思想引入到软件抗衰技术中,也即将单个构件作为抗衰技术中检测和措施的对象,一方面使得每个构件能够长时间保持在良好状态,从而提高整个体系的性能,另一方面引入 Request-Retry 机制,提高系统的可用性。本文基于 J2EE 构件模型开发出一种新的软件抗衰技术框架,相关理论已成果并在江苏省科技攻关项目"城域网海量视听信息实时点播系统"中应用。

关键词　软件抗衰,构件系统,抗衰粒度,可用性,性能优化

Research on the Component Rejuvenation in Component-Based System

ZHENG Xian-Fu　YANG Qun　XU Man-Wu

(State Key Lab for Novel Software Technology, Department of Computer Science and Technology, Nanjing University, Nanjing 210093)

Abstract　As a software application runs, it's performance and efficiency degrades over time due to factors such as memory fragmentation, counter inflation, and software errors. This is referred to as "Software Aging". "Software Rejuvenation" is a set of techniques to rectify this problem and return the software to its highest performance state. There are two methods to achieve this: 1) a complete restart of the affected software system 2) "Micro-reboot" which restarts only affected components of a software system. When using "Micro-Reboot", a component undergoing restart may not be temporary available during the restart process. Therefore a "Request-Retry" protocol is required, to insure the continued operation of the application during this process. This new software rejuvenation framework is based on the J2EE component model.

Keywords　Software rejuvenation, Component-based system, Rejuvenation granularity, Reliability, Performance tuning

1　引言

近年来,基于构件的软件开发方法已经被证明是一种有效的软件开发技术。但随着构件和构件系统自身复杂度的不断提高,"软件衰老"(Software Aging)成为影响这类系统可用性和运行性能的主要因素。软件衰老又称为"软件老化"(Outaging),是指长时间运行的软件系统随着运行时间的增长,运行性能逐渐下降并最终变为不可用的过程。但另一方面人们对软件系统的可用性、稳定性和运行性能也不断地提出越来越高的要求,软件抗衰技术作为提高可靠性和性能保持的有效方法,本文将结合构件系统的特点,提出一种软件抗衰技术框架。

软件抗衰(Software Rejuvenation)是基于前摄(Proactive)思想的重要的抗衰方法。它能够帮助用户在受到影响前预测、识别并修复故障,避免了系统突发故障及由此带来的严重后果,提高 MTTF(Mean Time To Fail)。Huang 等[1]分析了软件抗衰的起因、数学模型并进行对比分析。目前软件抗衰技术主要有基于时间度量和基于资源消耗度量两大类。文[2]详细介绍了软件抗衰技术的分类和各类方法的优缺点,文[8]介绍了一种适用于 Web 应用系统软件抗衰技术。

许多研究[10~12]表明重启(reboot)技术能够消除软件系统在运行过程中积累的一些错误状态,包括那些原因不明的错误,因此重启能有效恢复软件系统至初始良好状态。在传统软件抗衰技术中,往往对整个软件系统进行检测并在适当的时候对其进行重启从而达到解决"软件老化"的目标。

Recursive Restartability[8]研究的目的是缩短大型软件系统的恢复时间(MTTR:Mean Time To Recovery),针对大型软件系统的时间等资源消耗大而提出一种新的重启策略。它首先尝试重启某个构件集合(往往开始时为单个构件),如果已经解决问题则,到此为止。反之,则需要重启更大构件集合,再判断问题是否得到解决,直至问题解决或重启整个系统。

构件系统是由一个个相对独立、相互之间有严格边界的构件组成。由于提供或开发构件的组织和个人不一样,导致了构件之间在质量、效率上存在差异。因此,部分构件的软件老化会导致整个构件系统的性能下降,而在传统的软件抗衰研究框架下则将这部分构件的软件老化认为是整个构件系统的软件老化,对整个系统进行重启。事实上,在这种情况下,大部分构件还处于相对良好的状态,因此传统的研究角度没有考虑构件之间的差异,降低了软件抗衰技术的效果。文[13,14]研究了 IS(Internet Service)系统失败的原因,并提出通过构件的冗余技术来提高整体系统的可用性。

在构件系统中,构件之间有严格的边界并且存在差别,因此能够只关注存在衰老因素的那部分构件,将单个构件作为软件抗衰研究的对象。由此,本文提出一种适用于构件系统的软件抗衰技术,并证实了从系统层次细化单个构件层次能极大提高整个构件系统的可用性和性能,相关成果/理论在江苏省科技攻关项目——城域网海量视听信息实时点播系统中

*)本项目得到国家自然科学基金(60273055)、江苏省科技攻关项目(BE2003064)资助。**郑贤福**　硕士生,主要研究方向为软件方法学、软件抗衰、自主计算;**杨　群**　博士生,主要研究方向为软件方法学、自主计算、软件自愈;**许满武**　教授、博导,主要研究方向为软件方法学、新型程序设计。

得到应用。

2 构件和构件系统

2.1 状态分析

一般来说,构件和构件系统状态(如图1)有如下4种:初始健康状态(S_0)、衰老过程(S_p)、抗衰阶段(S_r)、不可用状态(S_f)。如果不存在衰老因素则保持在 S_0 否则进入到 S_p,当处于 S_p 状态,如果检测到该衰老并采取抗衰措施进入到抗衰阶段(S_r),否则进入不可用状态(S_f)。从不可用状态转化到健康状态只能通过其它外力干预,一般是人工干预。

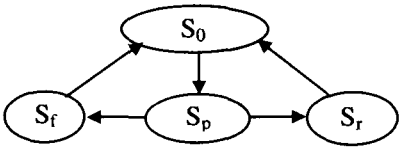

图1 构件和构件系统运行期状态转化

处于状态 S_p 时有两种可能,因此定义 $S_0 -> S_p -> S_r -> S_0$ 和 $S_0 -> S_p -> S_f -> S_0$ 都为一个转化周期。根据我们的研究以及文[4],这种周期性比较固定,并且其参数服从一定的概率分布。

2.2 J2EE构件系统框架

J2EE[15],CCM(Corba Component Model)[17]和.NET[16]是目前主流的构件中间件平台[18]。它们要求不同的构件模型,但都提供构件之间相互引用并提供性能监测、权限等基本服务。J2EE是三者之中唯一的一个开放的技术,并且有像JBoss[19]那样优秀的开源服务器,可以充分获取资料,也可以按照自己要求进行"定制"J2EE服务器。

J2EE服务器中包含有许多容器,构件都在容器中运行,通过名字从所运行的容器中获取所请求构件的引用。服务器和容器对构件有完全控制权,可以获取到构件的所有运行期的性能数据。在J2EE EJB3.0中,定义Interceptor[19]机制,允许对请求进行拦截和增加自定义处理措施,这样使可以在不改变构件和JBOSS程序的情况下获取构件的性能数据。

2.3 Request Retry机制

由于在我们的研究方法里允许对构件系统中单个构件进行软件抗衰,出现两种情况:当某个构件处于不可用时,准备请求该构件的服务的其他构件该如何处理;该构件如何处理正在处理的请求。在此,我们引入Request Retry机制,首先当服务请求构件获得服务处理构件处于不可用状态的信息后,需要每隔一定的时间进行尝试,直到成功接受请求或超过一定的次数。其次,如果发起请求构件在给定的时间内没有收到服务处理构件处理完毕的消息需要重新尝试。

引入Request Retry机制后,可以保证构件系统中任何一个单独的构件在任何时刻都可以采取软件抗衰措施,而不需要等待正在处理的请求处理完毕。

2.4 性能

可用性是描述软件系统,特别是对外提供服务的系统重要指标。计算公式A=MTTF/(MTTF+MTTR)已被广泛接受[6],由于构件和构件系统(用字母C表示)在上述4种状态(S_0,S_p,S_r,S_f)之间相互转化,由此对于某个构件或系统可用性可定义为:

$$\text{Availability}=\frac{\text{time}(S_0)+\text{time}(S_p)}{\text{time}(S_0)+\text{time}(S_p)+\text{time}(S_r)+\text{time}(S_f)}$$

其中time(x)定义为处于状态 x 的时长。

由于我们的研究对象是构件系统中单个构件,引入Request Retry机制后,构件系统的可用性定义为接受外界客户请求的构件的可用性。

文[5]指出响应时间适合作为在线服务类软件系统的性能评价标准。在此我们也选用响应时间作为判断标准。响应时间客观地反映了当前构件/系统的实际状态,是指服务请求构件或客户端发送请求到接收到处理结果的时间间隔。

在J2EE EJB3.0中提供了Interceptor机制能够拦截构件系统内部的请求。利用J2EE的这种机制,收集请求的响应时间。通过分析收集得到的性能数据判断构件所处的状态,然后根据构件的当前状态采取适当的软件抗衰措施或不采取任何措施,而软件抗衰措施则是通过J2EE容器来进行。

3 适用于构件系统的软件抗衰技术框架

3.1 总体框架

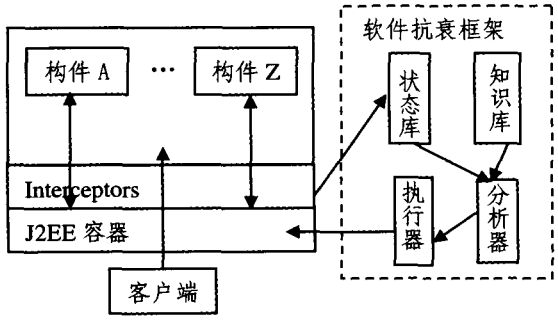

图2 适用于构件系统的软件抗衰技术框架

图2展示了适用于构件系统的软件抗衰技术框架示意图,整个构件系统运行在J2EE容器内。构件之间通过容器获取其他构件的对象引用,每次请求调用也都被Interceptor拦截。Interceptor将记录每次请求的响应时间到外部状态库。因此,状态库包含了所有构件的状态信息。

在容器外部,分析器和执行器是软件抗衰技术中主要组成部分。分析器是最重要的部件,分析状态库信息,并根据知识库决定是否对构件采取措施和采取何种措施;执行器则是最终执行抗衰措施。

3.2 构件状态的持久化

操作系统在进程切换的时候,需要对"进程的上下文"进行保存和恢复,同样当分析器决定要对某个构件抗衰措施(reboot),执行器负责调用J2EE容器服务,对构件进行重启。在这个过程中也需要对构件的状态进行保存和恢复。

这里的构件状态是那些与具体请求无关但又影响到请求的处理逻辑的构件参数的集合。在重启之前,将构件状态信息持久化到数据库或文件系统;重新启动后重新将状态信息初始化到构件。这样,可以保证构件在重启后与不重启的运行一致性。

3.3 软件抗衰的检测技术

随着软件抗衰技术研究的深入,提出了许多检测技术,主要有两大类:基于时间度量和基于资源消耗[2]。基于时间度量主要是分析软件系统运行规律,由分析器根据所获得的时间规律来决定采取何种措施,适用于那些有明显时间规律的软件系统;基于资源消耗则是根据对系统性能有重大影响的关键资源的消耗情况来判断软件的状态。

3.4 技术框架的特点和优势

软件抗衰技术作为一种相对成熟的技术,IBM已经广泛运用在Tivoli,DB2等大型商业软件系统[21],虽然具体实现不

一样，但技术框架基本上沿用分析器和执行器、状态库加知识库的模式。我们的技术框架可以认为是软件抗衰技术在构件系统的应用。

构件系统的构件之间，由于设计、开发等具体原因存在质量上的差异。设计良好、有良好的模型检查和完备的测试等要素的构件，能保持高可用性和高性能；反之，设计不精、没有完善测试的构件则无法保持高性能，长时间运行后会成为构件系统的性能瓶颈。

我们的技术框架的最大特点和优势有两点：

(1) 引入 Request Retry 机制后，构件系统的可用性定义为接受外界客户请求的构件的可用性。很显然从 A＝MTTF/(MTTF+MTTR)公式看，接受外部服务这单个构件的可用性必然大于整个系统的可用性。

(2)响应时间是系统性能的重要指标，文[5]指出在对于提供服务的系统，响应时间是最好的性能最好指标。而我们的技术框架从响应时间角度，可概括为：关注成为性能瓶颈的构件。根据著名的木桶理论，提高构件系统的性能，关键在于缩短瓶颈构件的响应时间。也就是说，将每个构件保持在最佳的状态，从而保证系统的响应时间最短。

J2EE[15]是一个被学术界和工业界广泛接受的构件模型，被广泛地采用[18]。采用该构件模型作为研究有两个方面优势：

(1)J2EE 的设计良好，易于将其他构件模型移植到 J2EE 体系。

(2)由于 J2EE 体系的开放性，市场上存在许多开源 J2EE 服务器，可以对其进行定制开发。

正是由于这两个原因，许多研究都采用 J2EE 作为研究框架。文[6,7,21]利用 J2EE 模型，定制开发 J2EE 服务器或容器以满足研究需求。

4 实验框架

随着互联网应用的发展，视频点播服务正在迅猛发展，它要求服务器能够长时间运行在高性能的状态，对软件系统的可用性和性能有极高要求。江苏省科技攻关项目“城域网海量视听信息实时点播系统”的目标是解决城域网环境下海量视听信息的检索、实时点播和服务器的性能保持。在实验系统中引入软件抗衰技术进行性能保持，软件抗衰框架使用本文所提及的技术框架，将实时点播系统软件系统中的构件作为抗衰的检测和采取措施的对象。

实时点播系统的登录操作是最终用户接触系统的第一步，该模块的可用性和性能直接关系到用户的使用感受和是否愿意继续使用实时点播系统。因此，我们将登录模块作为实验分析的对象，分别使用传统的技术框架和本文所提及的技术框架进行实现，并对实验结果进行分析。

软件抗衰是一种基于前摄(Proactive)的技术，分析器的设计直接决定抗衰的效果。在我们的对比实验中，软件抗衰框架中的分析器和知识库采用同样的设计。

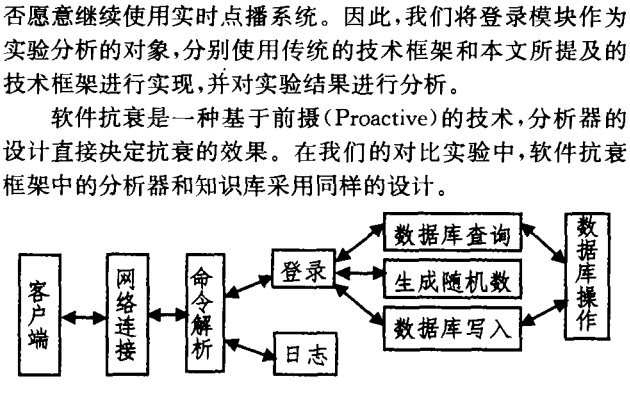

图 3 VOD 系统登录系统

图 3 是登录模块的示意结构图，在图中网络连接为入口构件。在 J2EE 构件模型下，构件之间的联系是通过容器进行，图中箭头起到示意作用。

为了对两种实现方案进行比较，我们开发了客户端模拟器。该模拟器的主要功能就是产生随机、大量的访问请求，并记录每次请求的系统可用性和响应时间。为了两种研究角度的对比分析，我们也记录采用传统研究框架的每次请求的响应时间。

5 实验数据分析

软件抗衰的主要目标是提高软件系统可用性和缩短系统响应时间两个目标。通过对实验运行数据的分析，我们的技术框架能够有效提高软件系统的性能。

5.1 系统可用性分析

引入 Request Retry 机制，极大地降低了系统不可用时长，从而在某种程度上提高了系统可用性。软件系统已经采用了抗衰策略，可用性处在一个很高的水平。

5.2 新框架较传统方案之性能数据

从性能对比图(图 4)中可以看出，采用传统的软件抗衰技术能够使软件系统在性能恶化的时候将其回复到初始干净状态，从而获得高性能。采用我们的技术框架后，由于抗衰的对象是软件系统中一个构件，因此使得性能波动更小，从而更进一步缩短响应时间，提高软件抗衰的效果。

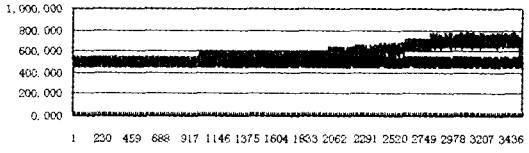

图 4 性能对比

结论和进一步工作 对于构件系统，我们利用构件之间的差异性和边界性，将单个构件作为软件抗衰检测和采取措施的对象，并由此提出适用于构件系统的软件抗衰技术框架。在传统的软件抗衰研究基础上，讨论了新技术框架的优点。在江苏省科技攻关项目“城域网海量视听信息实时点播系统”的应用表明，新技术框架的两点优点能提高构件系统的可用性和性能。

在我们的研究框架内，依赖于 J2EE 构件模型，响应时间的检测也依赖于 J2EE 的 Interceptor 机制。在我们下一步研究中，将构件研究框架推广到其它的构件模型。

利用 Interceptor 机制对每个请求访问进行记录响应时间，也就是说对每次请求都额外增加了操作步骤。减少请求拦截次数，也是我们下一步研究的重点。

参 考 文 献

1 Huang Y, Kintala C, Kolettis N, et al. Software Rejuvenation: Analysis, Modules and Application. In: Proc. of 25th Symposium on Fault Tolerant Computer Systems. Pasadena, California, June 1995

2 陈博. 软件恢复：从理论到实践. 南京大学：南京大学计算机系，2003

3 Wang Y M, Huang Y, Vo K P, et al. Checkpointing, and Its Application. In: Proc. of 25th Intl. Symposium on Fault-Tolerant Computing. Pasadena, California, June 1995

4 李正，万群丽，许满武. 软件恢复技术研究. 计算机科学，2003，30(8)：150～155

5 Broadwell P M. Response Time as a Performability Metric for Online Services. University of California at Berkeley ; Computer Science Division of University of California at Berkeley, 2004

6 Candea G, Cutler J, Fox A. Improving Availability with Recursive Micro-Reboots: A Soft-State System Case Study. Performance Evaluation Journal, Summer 2003

(下转第 289 页)

2)路由配置。用于控制流程的流向,不同的路由指向不同的操作节点,通过操作属性选择下一步的流向,同一节点可以单一操作或多个操作,操作属性通常包括操作名称、编号、源节点、目标节点、涉及范围对象、提醒方式(文件到来时的提醒方式,选项包括邮件、网络寻呼、短信息等)。

流程配置的步骤,首先定义点(流程节点),其次通过单向线(路由)将点对点进行连接,然后再定义分支与循环关系最终形成流程图。一般来说,流程配置比较复杂,容易出错,容易形成断点。可通过设计可视化流程配置工具(图 3 以发文流程图为例),直观反映流程图的整体情况,并与通过检查工具确保流程配置的正确性。

在公文管理中,除了复杂的业务逻辑之外,还必须考虑到用户的工作习惯及公文传输过程的留痕等问题,需要提供与 Word、Excel、WPS 等流行常用的 Office 软件无缝连接。在公文流转过程中,同一公文提交多人同时办理或依次办理时,能够自动地对每次操作进行控制,并将多人处理结果进行有机整合,通过留痕管理对文件的修改过程进行全方位跟踪,对修改人、修改时间作出明确的记录。

2.3 消息管理子系统

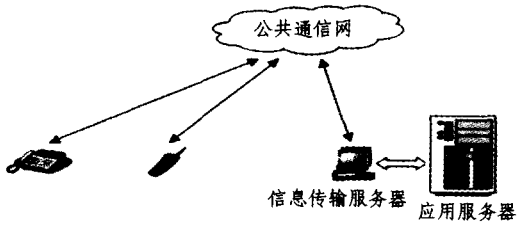

图 4 消息传输图

在传统办公自动化系统中,由于受通信条件的限制,用户的待办事宜唯有通过被动的查询方式获得业务的办理情况。消息管理子系统主要是利用当前能够提供公共通信服务,如:电信、移动等提供与手机、电话通信的功能,提供手机短信提示、电话提醒等功能。使待办公文能够主动及时地通知办理者,从而提高公文办理效率。消息管理子系统除了对登录用户实时提醒外,还能够通过手机、固定电话等通信工具通过公共通信网与本地的信息传输服务器、应用服务器系统连接,查询到相关信息的功能。如图 4 所示,信息传输服务器主要是按照传输协议处理公共通信网与内部应用系统之间的通信和数据转换。

2.4 模块管理子系统

模块管理子系统主要用于组织模块的层次结构。通过定义模块的分组、分层结构,使各模块按照配置好的层次加以展现。模块配置管理组件化可以增强系统的灵活性和可扩展性,可以快速将二次开发完成的模块或组件发布到系统中。模块管理主要包括如下配置项:模块所属目录、模块所在服务器、模块位置等。

2.5 安全管理子系统

安全管理子系统具备统一的身份认证、统一的门户管理、多重安全保护等功能。通过对数据链路层、网络层的信道加密和数字证书应用管理系统(或与第三方 PKI/CA 体系)进行身份认证、访问和系统的监控、日志分析,保证办公自动化应用系统能够方便地管理和访问不同的硬件平台、操作系统、网络协议和数据,使系统的安全风险降到最低限度,从而实现"单点登录,Web 应用漫游"。

结束语 上述办公自动化系统已在某地方高校开发成功,投入使用。该系统具有完备的接口和良好的耦合性,与按照教育部信息化建设标准建立的学校基础数据库连接,实现了与以前开发的应用系统有效整合。实际应用表明,应用 J2EE 多层次架构开发的办公自动化系统具有很好的开放性、实用性、安全性、可扩展性和可维护性,应用效果良好。

参 考 文 献

1 [美]Horstmann C S, Cornell C. Java2 核心技术(第 2 卷):高级性能[M]. 北京:机械工业出版社,2003
2 [美]Horstmann C S, Cornell C. Java2 核心技术(第 1 卷):基础知识[M]. 北京:机械工业出版社,2003
3 [美]KEOGH J. J2EE 参考大全 M]. 宁建平,等译. 北京:电子工业出版社,2003. 9~25
4 [美]Couch J. J2EE 宝典[M]. 马琳,等译. 北京:电子工业出版社

(上接第 277 页)

7 Candea G, Kawamoto S, Fujiki Y, et al. Microreboot: A Technique for Cheap Recovery. In: Proc. of the 6th Symposium on Operating Systems Design and Implementation(OSDI), Dec 2004
8 Candea G, Fox A. Recursive Restartability-Turning the Reboot Sledgehammer into a Scalpel. In: Proc. of the 8th Workshop on Hot Topics in Operating Systems(HotOS-VIII), May 2001
9 万群丽,杨群,李正,许满武. 一种基于 Agent 适用于 Web 应用的软件抗衰方法,计算机应用研究,2004,21(8):18~21,26
10 Sullivan M, Chillarege R. Software defects and their impact on system availability: a study of failures in operating systems. In: Proc. 21st Intl. Symposium on Fault-Tolerant Computing, Montr'eal, Canada, 1991
11 Gray J. Why do computers stop and what can be doneabout it?. In: Proc. 5th Symp. on Reliability in Distributed Software and Database Systems, Los Angeles, CA, 1986
12 Chou T C. Beyond fault tolerance. IEEE Computer, 1997, 30(4):31~36
13 Oppenheimer D, Ganapathi A, Patterson D A. Why do Internet services fail, and what can be done about it?. In:4th Usenix Symposium on Internet Technologies and Systems(USITS '03), 2003
14 Ganapathi A. Failure Analysis of Internet Services. Computer Science Division, University of California, Berkeley
15 Sun Microsystems. http://java.sun.com/j2ee/
16 Microsoft:http://www.microsoft.com/net/
17 Omg: http://www.omg.org/technology/documents/formal/components.htm
18 Szyperski C. Component Software: Beyond Object-Oriented Programming. Addison-Wesley, 2002
19 JBoss: www.jboss.org/
20 Diaconescu A, Mos A, Murphy J. Automatic Performance Management in Component Based Software Systems. Performance Engineering Laboratory, Dublin City University
21 Patterson D. Recovery Oriented Computing(ROC): Motivation, Definition, Techniques, and Case Studies. University of California at Berkeley