基于构件的软件性能模型及评估技术研究

冯　杰　赵文耘　杨明华
（复旦大学软件工程实验室，上海 200433）
E-mail：john_feng@etang.com

摘　要　该文讲述了在基于构件的软件开发中，建立应用系统的性能模型，以及对系统的性能进行预测和评估的方法，并介绍了构建一个满足预期性能目标的软件系统的步骤。

关键词　性能管理　性能预测　性能评估　性能模型

文章编号 1002-8331-(2004)14-0099-03　**文献标识码** A　**中图分类号** TP311

Component-Based Software Performance Model and Performance Evaluation

Feng Jie　Zhao Wenyun　Yang Minghua
(Laboratory of Software Engineering, Fudan University, Shanghai 200433)

Abstract: This paper discusses how to build the performance model for the Component-Based software and proviedes a method for performance evaluation based on performance model.The paper also provides a performance managing process for Component-Based development.

Keywords: performance management, performance evaluation, performance model, performance process

1　前言

软件性能是每个软件系统的重要质量指标，许多软件系统，因为性能问题而无法使用。所以，对软件性能的管理得到越来越多的重视，但一直以来，对软件性能改进的研究总是陷在队列理论、马尔科夫分析、深奥的进度算法的各种细节之中。软件性能主要指一个软件系统在时间上满足需求的程度，还包含了诸如内存使用等其它一些特征。

基于构件的软件开发(CBSD)能够在较短的时间开发出高质量的软件产品，CBSD 也给软件的性能管理带来了新的方法，使软件性能的评估变得容易。可以将软件性能工程(SPE)的方法应用到基于构件的软件开发之中。基于构件的软件开发经常着眼于系统的功能特性，对于其它特性比如性能的研究较少。如果能够为各个构件建立相应的性能模型，然后根据系统实际运行环境为系统的基本设施或设备建立性能模型，再根据构件和设备的性能模型以及系统的构架和设计来建立整个系统的性能模型，就能在软件开发的各个阶段对系统的性能进行评估，以确保最终的产品满足预期的性能要求。如图 1 所示。

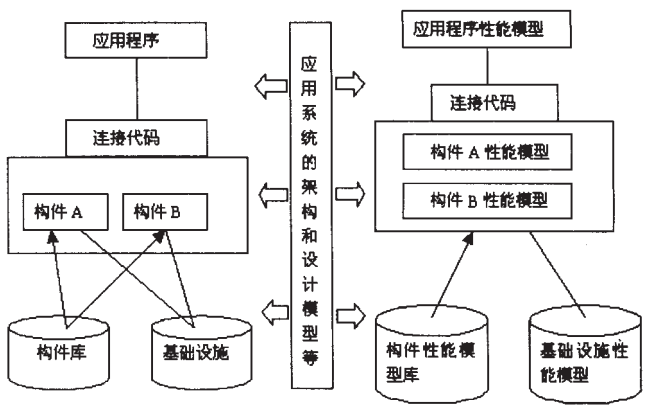

图 1　基于构件的软件开发的性能管理

2　基于构件的性能模型

通过为软件系统建立性能模型，来对系统的性能进行预测和评估。基于构件的软件性能模型的建立需要完成以下三个方面的工作：

(1)构件性能模型的描述以及实例化。

(2)如何建立各个设备的性能模型。

(3)根据构件和设备的性能模型以及整个系统的架构和设计来建立系统的性能模型。

2.1　构件性能模型的描述

为了进行性能管理，构件开发者需要将构件的性能属性加入到构件的描述中。构件的性能极大地依赖于它的运行环境，但这里需要得到与平台无关的性能模型。作者通过包含抽象和量化的性能环境参数来描述构件的性能模型。这样，得到的是特定环境下的构件性能模型，而不是特定平台下的构件性能模

基金项目：国家 863 高技术研究发展计划：基于 Internet，以构件库为核心的软件开发平台(编号：2001AA11307)资助
作者简介：冯杰(1978-)，男，复旦大学计算机与信息技术系硕士研究生，研究方向：软件工程。赵文耘(1964-)，男，复旦大学计算机与信息技术系教授，博士生导师，研究方向：软件工程。杨明华(1979-)，男，复旦大学计算机与信息技术系硕士研究生，研究方向：软件工程。

型。一旦建立性能目标,只需要根据具体的平台环境实例化参数,就可以验证给定的环境是否能够支持构件的性能模型。所以构件提供的性能模型必须考虑到各种兼容的环境。

假设对于一个给定的构件 C_i,它提供 $n(n>=1)$种服务 $S_j(j=1,2\cdots\cdots n)$,对它的性能进行描述就是 $Perf_{ci}(S_j(env-par))$,其中 *Perf* 表示作者考虑的某种性能属性(比如响应时间,通信延迟等),*env-par* 表示一个环境参数。

图 2 表示在构件的接口中加入对于构件性能属性的描述。

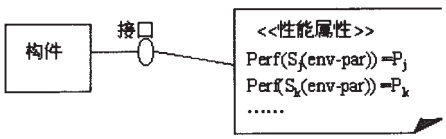

图 2 在构件接口描述中加入性能说明

构件的性能属性由构件开发者提供,可以通过统计等各种方法得到,对于如何得到构件的性能属性,这里不做介绍。

2.2 设备性能模型的建立

在确定系统的性能目标以后,再建立设备的性能模型,所以在建立性能模型以前,软件的运行环境已经被确定,只要确定各个设备在该特定环境下的性能即可,这可以根据实际设备的一些说明参数和当前环境来确定,比如网络的延迟,一次数据库操作所需时间等,对于一个提供 $m(m>=1)$种服务 $S_j(j=1,2\cdots\cdots m)$的设备,可用和构件的性能描述相似的方法来描述该设备的性能。如图 3 所示。

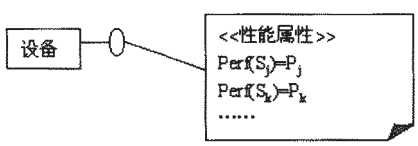

图 3 设备的性能描述

与构件的性能描述不同的是,这里描述某设备在特定环境下的性能,所以不需要一个环境参数。

2.3 构建系统的性能模型

整个系统的性能模型由各个构件和设备的性能模型以及整个系统的构架和设计得到,它的基本思想是把软件执行模型(EM)从运行环境和单元模型(UM)中分离出来。软件执行模型描述了软件系统的基本行为,可以用序列图来表示。序列图描述了系统完成某次行为所需的构件以及构件之间的交互,它可以从系统设计的用例图和时序图中得到。单元模型描述了各个构件和设备的性能模型以及相关的参数。通过软件执行模型、运行环境、单元模型和系统的构架和设计就能建立整个系统的性能模型。一旦建立了性能目标,就可以建立相应的执行模型和单元模型,整个系统的性能模型也能被构造出来,据此就可以对软件的性能进行评估。把执行模型、运行环境和单元模型三者分离后,同一个软件系统可以在不同的环境中进行评估。

软件的执行模型可以从系统的构架和设计模型中得到,系统开发者使用一系列的时序图、活动图等来描述软件系统的行为。作者用序列图来描述系统的执行模型。它提供了一种对于软件执行步骤视觉上的表示,用来解析软件的执行。序列图为性能评估提供了一种便利的表示形式。当然也可以直接对时序图或活动图加以分析,但各种附加信息、性能属性等,应被包含在相应的图形中。使用序列图有助于对分析步骤和模型结果的解析。而且,现在也没有什么工具可以用来对时序图或活动图的性能进行直接的解析。图 4 是一个系统序列图的具体例子。

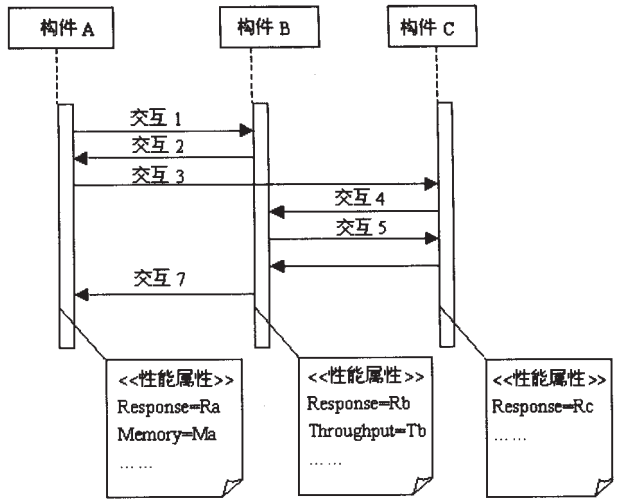

图 4 一个序列图的例子

从图中可以看到,每个构件的性能属性被包含在图中。有了软件的执行模型,为了描述系统的性能模型,还需要建立一个模型来描述执行模型中各个设备的使用情况,用一个配置图来表示这个模型。如图 5。

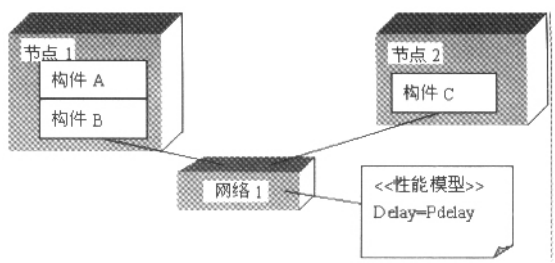

图 5 配置图例子

图 5 表示构件 A、构件 B 和构件 C 之间的交互要通过网络来完成。配置图中还可以包括其它的一些设备,如终端显示器、数据库等。图中应该表示出每个设备的性能属性。可以为每个执行模型单独建立它的配置图,也可以从总的配置图得到各个执行模型的配置图。

根据软件执行模型和配置图就可以得到系统的性能模型。性能模型描述了完成各个行为所需的构件以及设备,并且根据它们的执行模型和配置图给它们标上权重,在上面的例子中,这次交互共使用到 3 个构件和一个设备:构件 A、构件 B、构件 C 和网络 1。它们各自的性能属性被标志出来,根据序列图和配置图就可以给它们标上权重,构件 A,构件 B,构件 C,网络 1 的权重分别为:$P(C_A)=2/11$,$P(C_B)=3/11$,$P(C_c)=2/11$,$P(L_1)=4/11$。知道了构件和设备的权重后,还需要为每个构件和设备内部的服务计算标上权重,比如构件 A 提供 $h(h>1)$种服务,在本次操作中每种服务分别被调用 $N_1,N_2,\cdots N_h$ 次(N_k 可能为 0),共被调用 $N=N_1+N_2+\cdots\cdots N_h$ 次,那么第 i 种服务的权重 Ps_i 就是 N_i/N。对于设备提供的各种服务的权重计算方法也和构件一样。这些也可以从序列图和配置图中直接得到。最后得到本次操作的性能模型,如图 6。

据此,就可以为性能目标中的所有人感兴趣的行为建立它

们的性能模型。完成了性能模型创建以后,就可以进行性能的评估了。

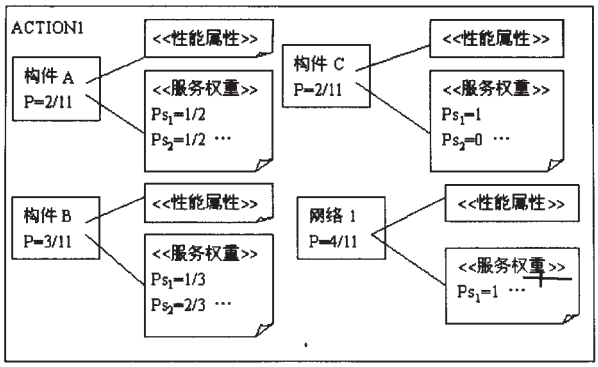

图 6　某次交互的性能模型

3　软件性能评估

建立了软件的性能模型后，就能对软件的性能进行评估，这里对最优情况下的性能进行评估,最优情况是指整个系统单独运行当前行为的性能,所以没有资源冲突的情况。

假设某次行为共用到 n 个构件和 k 个设备,而且在每个时刻系统只对某个构件或设备进行操作,那么整个操作的性能就用如下的公式来计算：

$$Perf(Application)=\sum P(C_i)*Perf_{Ci}(S(env))+\sum P(M_i)*Perf_{Mi}(S)$$

其中 $Perf(Application)$ 表示整个应用程序的性能,$P(C_i)$ 为构件 C_i 的权重,$Perf_{Ci}(S(env))$ 表示该构件的性能,$P(M_i)$ 表示设备 M_i 的权重,$Perf_{Mi}$ 表示设备 M_i 的性能。

对于每个构件 C_i,它的性能用如下的公式来计算：

$$Perf_{Ci}(S(env))=\sum P_{Si}*Perf_{Ci}(S_i(env))$$

对于每个设备 M_i,它的性能计算方法和构件类似：

$$Perf_{Mi}(S)=\sum P_{Si}*Perf_{Mi}(S_i)$$

这些公式中各个构件和设备的权重以及各构件和设备内部服务的权重和性能可以从性能模型中直接得到。用以上的公式计算得到的是某次行为中各个构件和设备性能的平均值,评估时可以根据实际情况进行转换,比如用以上公式得到各个构件和设备的平均响应时间,乘上交互次数就是完成本次行为所需要的总的响应时间。

据此，就能对性能目标中的各个行为的性能进行评估,实际的评估时可能考虑到通信延迟、资源竞争等情况。

4　基于构件的软件开发的性能管理步骤

对于软件性能的合理的管理方法是在软件开发的整个过程中系统地对正在开发的软件的性能进行规划和预测。有了系统的性能模型和性能评估的方法,就能在软件开发的各个阶段对软件的性能进行预测和评估,从而构建满足预期性能目标的软件系统。图 7 讲述了基于构件的软件性能管理的步骤。

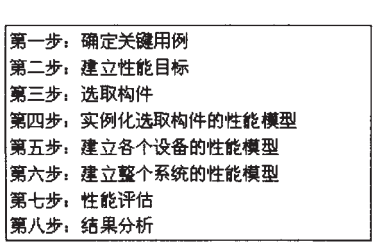

图 7　基于构件的性能管理步骤

第一步 确定关键用例

在这一步,系统开发者必须从系统的各个使用者出发来建立系统用例,如果一个系统有多个使用者和多个用例,那么就需要从中找出对整个系统性能影响较大的一些关键用例。关键用例的选择是风险驱动的。应该到那些如果性能问题得不到解决,整个系统就会失败或失效的地方去寻找关键用例。关键用例是所有用例的一个子集。在 UML 中是通过用例图来表示用例的。

第二步 建立性能目标

为在第一步中找出的关键用例定义出性能目标以及负荷强度。性能目标能够提供一个定量的标准,以便对开发中的系统进行定量的分析。这些目标可以通过多个途径加以描述 响应时间、吞吐量、或是对资源使用的限制等。

第三步 选取构件

这一步是从构件库中选取能够完成系统功能且具有较好性能的构件。

第四步 实例化选取构件的性能模型

由于性能目标已经建立,所以系统的运行环境就被确定下来，据此就可以对构件的性能模型中的环境参数进行实例化。一个构件 C 提供的每个服务 S_j 的性能用 $Perf_C(S_j(env-par))$ 来描述,其中的 $env-par$ 是描述该构件的运行环境的参数,在此,就可以根据当前的具体环境对它进行实例化,以验证给定的环境是否能够支持构件的性能模型。

第五步 建立各个设备的性能模型

和第四步一样,一旦系统的性能目标建立后,根据整个系统的架构和设计模型,系统运行所需的设备(如数据库、网络等)也被确定下来,然后根据实际的运行环境就可以为它们建立相应的性能模型。

第六步 建立系统的性能模型

整个系统的性能模型从构件和设备的性能模型以及系统的构架和设计模型得到。文中第三部分讲述了建立一个系统的性能模型的方法。

第七步 性能评估

完成了第六步性能模型创建以后,就可以在软件开发的各个阶段对性能目标中的各个行为的性能进行评估,实际评估时可能考虑到通信延迟、资源竞争等情况。

第八步 结果分析

在第七步的性能评估中，得到系统一系列行为的性能,比如系统完成某次行为所需的时间。另外还有资源的使用率等。把这些评估值和目标值进行比较,如果能够满足目标就可以进行系统的开发,否则回到第三步重复以上的步骤直到满足性能目标,如果都得不到满足,那么就需要改进设备等,或者改变性能目标,作者建立的性能模型也为性能的改进提供了方向。

5　结论和将来的工作

该文讲述了在基于构件的软件开发中,如何为各个构件和设备建立性能模型,并根据构件和设备的性能模型和系统的架构和设计模型来建立整个系统的性能模型和基于构件的软件开发的性能管理步骤。

将来的工作包括 :完善性能模型,使性能预测和评估更加精确。把软件性能管理集成到软件开发工具中,自动地完成一

(下转 122 页)

LZW、RunLength、Flate、CCITTFax、JBig2、DCT 等，必须实现对这些过滤器的解码算法才能从内容流中提取出正确的信息。

(2)文本属性的提取

文本属性包括文本内容，字体，颜色等等。PDF 支持 Type1，TrueType，Type3 等字体。要得到正确的文本内容，需要对字体的编码信息进行还原。除了 Type3 字体外，所有 PDF 字体都有一个内建的编码表。可以通过对编码表的映射还原出 PDF 所显示的文本内容。而 Type3 字体则是一种自定义的字体。在 Type3 字体中，字形的显示是由 PDF 图形操作符流所定义的。这些流和字符名联合起来，一个单独的编码表把字符码映射到显示相对应的字符名。因此必须通过 OCR 的方法才能提取出 Type3 字体的文本内容。

PDF 采用了 CIDFont 来表示以多字节为编码单位的字符集的编码，以显示亚洲地区的字体，如中文、日文、韩文等。这三种语言的字符集统称为 CJK(Chinese，Japanesea and Korean)。需要用一个叫作 CMap(Character Map)的映射文件来确定字符的编码所对应的 CID 编码。CIDFont 和 CMap 的信息可以在 CIDSystemInfo 对象中得到。

PDF 中对于色彩的定义可分为设备相关的色彩空间(Device-Dependent Color Spaces)，设备无关的色彩空间(Device-Independent Color Spaces)以及特殊色彩空间三种。设备相关的色彩空间包括 DeviceGray、DeviceRGB 和 DeviceCMYK 三种，设备无关的色彩空间包括 CalGray、CalRGB 和 Lab 三种，特殊色彩空间包括 Sepration (专色)、Indexed (颜色索引表)和 Pattern(底纹)三种。页面上的所有着色操作均与色彩空间相关，任何一种颜色都是由某一色彩空间及相应空间下的颜色值共同来表示的。各个色彩空间下的颜色值可以相互转换。

(3)篇章结构的实现

PDF 文档并没有提供文档的篇章结构信息，但是为了分析文档的结构，可以利用 PDF 的 Bookmark 来判断一个文档大致的篇章结构。PDF 文档的 Outline Tree 是一个树型层次结构，其中的每一个节点都是一个 Bookmark，每个 Bookmark 指向文档中的一个位置。可以通过 Bookmark 指向文档中的具体位置来给文档划分篇章结构。

具体的方法是将 Outline Tree 的结构作为整个文档的结构。Bookmark 节点在 Outline Tree 中的深度可以对应转换成的 XML 文档中章节的层次结构，而 Bookmark 的文本内容则可以作为所指向章节的标题。章节的划分以 Bookmark 指向文档中的具体位置为依据。

对于段落的判断可以通过一些规则来实现。PDF 文档并没有提供段落的信息，但是可以从文本显示的角度来大致地划分段落。PDF 中给出了每个所显示的字符在页面视图中的坐标，因此可以通过文本的坐标来判断段落的位置。一般说来在一个 PDF 文档中显示段落有两种方式，一种是通过增大段间距，一种是首行缩进。可以由此制定规则来划分文档的段落。如果发现两行文本之间的间距大于平均的行距，则认为这两行文本分属两个段落。同样如果行首文本的坐标大于前一行文本的行首坐标，则认为这一行文本是新段落的开始。

(4)链接的处理

PDF 中的链接类似于 HTML 中的链接，包含了链接的文本内容和指向的目的地址。因此可以很方便地将 PDF 中的链接转换为前面给出的 DTD 所定义的链接格式。

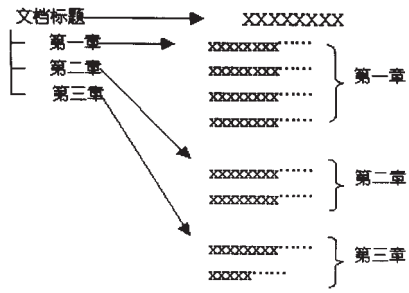

图 2　用 BookMark 来划分文档的篇章结构

5　结束语

在对网络文档进行信息检索、过滤、提取等处理操作中，文档格式转换是分析文档内容和结构的先决条件。该文就基于文档的文本内容和结构方面的 PDF 文档向 XML 文档的转换技术进行了讨论，对转换的相关技术进行了一些阐述。用户可以对转换得到的 XML 文档进行下一步的操作，从而实现对文档的内容提取、信息检索和过滤等功能。

(收稿日期：2003 年 7 月)

参考文献

1.Adobe Systems Incorporated.PDF Reference third edition，Adobe Portable Document Format Version 1.4

2.Extensible Markup Language(XML)1.0.Second Edition，http://www.w3.org/TR/REC-xml，2000-10

3.Norbert Fuhr.XML Information Retrieval and Information Extraction.http://ls6-www.informatik.uni-dortmund.de/bib/fulltext/ir/Fuhr:02a.pdf，2002

4.Danny Sullivan et al.Fifth Annual Search Engine Meeting Report[R].Boston，MA，http://websearch.about.com/internet/websearch/library/blsem.htm，1999-04

5.Elliotte Rusty Harold 著.杜大鹏，李善茂，傅烨等译.XML 实用大全[M].中国水利水电出版社，2000

(上接 101 页)

些功能，以便能够消除重复性的劳动，能够更为快速、更为容易地对各种可供选择的设计方案的性能进行评估。软件性能管理过程应该能够与软件开发过程紧密结合，而且能够比较容易地使用一个集成了软件性能管理的软件开发过程，以便将软件性能工程广泛地应用于实际软件系统的开发中。

(收稿日期：2004 年 1 月)

参考文献

1.Connie U Smith，Lloyd G Williams.Performance Solutions A Practical Guide To Creating Responsive.Scalable Software

2.Bachman F et al.Technical Concepts of Component-Based Software Engineering[R].T R U/SEI-2000-TR-008

3.Bertolino A，Mirandola R，Modeling and analysis of nonfunctional properties in component-based systems.to appear Tacos 2003

4.Antonia Bertolino Istituto di Scienza e Tecnologia，Raffaela Mirandola Dip.Informatica，Sistemi e Produzione.Towards Component-Based Software Performance Engineering

5.Xiuping Wu，David McMullan，Murray Woodside.Component Based Performance Prediction

6.http://www.perfeng.com