

基于 DAnCE 的跨平台分布式组件部署研究

王 涵, 李 晶, 李石君

(武汉大学计算机学院, 湖北 武汉 430072)

摘 要: 分析了 DAnCE 的源码以及体系结构, 并根据系统实时监控及动态迁移的需要修改并扩展了其中的核心代码, 设计并实现了一种基于 DAnCE 的跨平台分布式组件部署的解决方案. 最后验证了该解决方案的可行性, 保证在跨平台分布式应用环境中组件间的通信和对组件的远程管理.

关键词: CCM 模型; DAnCE 框架; 分布式组件; 中间件; 部署与配置

中图分类号: TP311

文献标识码: A

文章编号: 1000-7180(2015)02-0105-05

Research on Cross-Platform and Distributed Components Deployment Based on DAnCE

WANG Han, LI Jing, LI Shi-jun

(School of Computer, Wuhan University, Wuhan 430072, China)

Abstract: This paper analyzed the source code and architecture of DAnCE, modified and extended the core code according to the actual need of real-time monitoring and dynamic migration, and proposed a solution of deploying and configuring components in cross-platform and distributed system based on DAnCE. Finally, the feasibility of the solution, which can ensure the communication between the components and remote management of the components.

Key words: Corba Component Model; Deployment and Configuration Engine framework; distributed component; middleware; deployment and configuration

1 引言

在高度异构、复杂多变的分布式应用系统环境中,如何在跨平台分布式应用环境下对组件进行部署和配置,是该领域面临的挑战之一. DAnCE (Deployment and Configuration Engine) 是基于 CORBA 实现的一种组件部署与配置框架,该框架能够解决分布式软件产品在异构跨平台的环境下对组件的部署和配置问题.

在基于组件的分布式应用系统中,软件的部署生命周期是至关重要的. 本文研究的是分布式应用系统中组件的部署和配置过程. 与部署生命周期相关的主要活动有组件的发布、组装、激活、去活、重配置、更新以及删除等^[1]. 本文通过对 DAnCE 框架进

行研究,并根据实际的跨平台分布式应用系统进行修改并完善,设计并实现了一种跨平台分布式组件部署与配置的解决方案. 该方案能够有效地支持计算环境的异构性,并通过部署计划建模来支持组件在分布式环境中的自动化部署与配置.

2 相关理论及研究

2.1 CCM (CORBA Component Model)

OMG 组织在 CORBA3.0 标准中新增了 CCM 规范^[2],即 CORBA 组件模型规范. CCM 的主要贡献是将组件中间件技术进入到 CORBA 里. 组件中间件技术通过更高层次的抽象定义,弥补了面向对象中间件技术的一些缺陷,定义了组件的虚拟边界、发布方法以及组件与其他组件和客户端进行交互的

收稿日期: 2014-04-25; 修回日期: 2014-06-17

基金项目: 国家自然科学基金项目 (61272109)

方式.同时,通过使用组件中间件技术,分布式应用系统可以使用一种“装配的”方式来构建,支持二进制级别的重用,而不像传统的工程方法那样只支持源码级别的重用. CCM 的体系结构主要包含抽象组件模型、组件实现框架、组件的打包组装及部署、组件容器框架等.

CCM 抽象组件模型提供了称为端口的多种外部接口,以便与客户、其他组件、CORBA 服务等进行交互. CCM 组件支持以下五种基本端口,如图 1 所示.

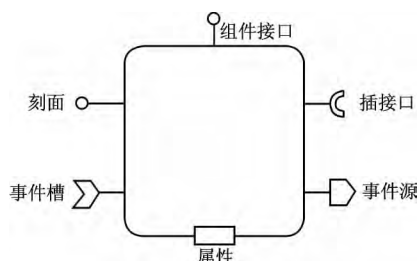


图 1 CCM 抽象组件模型

(1)剖面:组件实现且可供其他组件客户使用的接口;(2)插接口:组件使用外部对象或组件引用的连接点,一般用于组件之间的互连或组装;(3)事件源:负责事件的发布,用于向一个或多个关注某类特定事件的订阅者或事件通道发布该类型的事件.其中,事件源分为 Emitter 和 Publisher 两类. Emitter 规定同一时间只允许一个接收者与其连接,而 Publisher 则允许同一时间有多个接收者与其连接;(4)事件槽:负责事件的接收,用于接收外界向其推送的特定类型事件;(5)属性:主要用于组件的配置,定义了组件在运行时可配置的特性.另外,设计者还为每个组件定义 Home,一个 Home 管理某一类组件的所有实例. Home 要求所有实例都在一个地方生成,通过工厂设计模式实现对组件生命周期的管理.

部署在分布式应用系统的不同结点上的组件可以通过组件容器来与外部进行交互,而且通过 ORB 可以实现异构环境之间的交互. 组件容器的主要功能是为组件提供一个运行环境及上下文对象,除了在运行时负责组件的创建和管理、组件实例的生命周期管理、组件状态激活去活管理以及容错、负载平衡等 QOS 服务管理,组件容器还负责根据组件开发者的声明为组件提供其需要的各种标准服务. CCM 组件容器框架如图 2 所示.

和传统的软件开发方法相比, CORBA 组件模型(CORBA Component Model, CCM)支持异构环

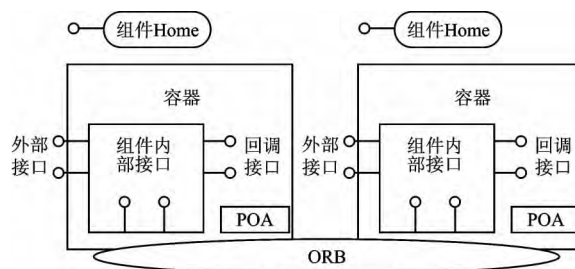


图 2 CCM 组件容器框架

境下分布式应用系统的开发和互操作,具有与底层硬件、操作系统、网络、通信协议以及编程语言无关的特点,所以被广泛地应用于大型的分布式应用系统中.

目前,基于 CCM 的研究和实现有很多,国内外典型的基于 CCM 组件技术的平台包括法国里尔大学开发的 OpenCCM 系统、美国华盛顿大学开发的 CIAO (Component-Integrated ACE ORB)、ICMG (Internet Component Management Group) 开发的 K2-CCM、MICO 项目组开发的 MicoCCM、CPI 开发的 EJCCM、D. O. G 开发的 JavaCCM 以及国防科学技术大学的 StarCCM 等等.

2.2 DAnCE (Deployment and Configuration Engine)

为了规范基于组件的软件开发过程,OMG 把基于组件的软件开发方法划分为组件开发、运行平台定制、组件应用部署三个阶段^[3]:在组件开发阶段,由组件开发人员开发组件,组件组装人员通过组装现有的组件形成复合组件,组件打包人员将多个同种功能组件的不同实现打包在组件包中;在运行平台定制阶段,域管理人员负责制定系统运行的目标平台及部署分配,如计算结点、网络资源和共享资源等方面;在组件应用部署阶段,组件部署与配置人员根据组件库中的组件包和运行平台及部署分配等相关信息制定出部署计划,把组件仓库中的组件组装后部署到运行平台上,保证分布式应用系统的正常运行.

当组件开发和运行平台定制两个阶段完成之后,就进入了组件的应用部署阶段. OMG 提出的 Deployment & Configuration (&C) 规范^[4]则提供了针对应用的组件部署和配置指南. D&C 规范基于以下目的而制定:(1)更好促进组件重用;(2)通过组装已经存在的组件来构造复杂的大规模及超大规模的应用系统;(3)通过抽象出应用程序在部署和配置过程中的所涉及公共元素,以便在组装与部署基于组件的应用时,能重用这些元素;(4)能动态地、自

动化地把组件部署到目标平台上。

DAnCE^[5,6]是在 CORBA、TAO(The ACE ORB)以及 CIAO 的基础上,基于 D&C 规范实现的部署与配置框架,负责处理面向组件的应用的部署与配置,可以兼容各种硬件和操作系统平台。DAnCE 框架的部署配置体系结构如图 3 所示。

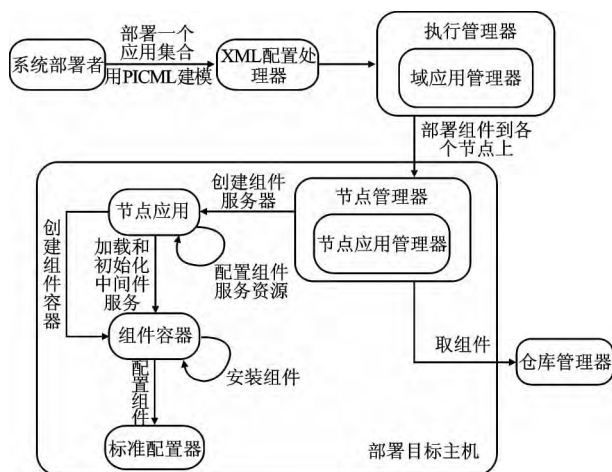


图 3 DAnCE 框架的体系结构

首先,应用框架必须由专用的部署描述文件(CDP 文件)进行定义,定义的内容包括组件实例和组件间连接等。这些文件将作为部署框架的输入,由部署框架负责组件的实例化和配置,同时也需要实现组件间的连接,最终形成一个可运行、可配置、可远程管理的分布式系统。

3 基于 DAnCE 的修改和扩展

本节主要对 DAnCE 框架核心代码进行研究,对 DAnCE 框架内的几种主要服务对象进行修改和扩展其功能,来实现一种跨平台分布式组件部署的解决方案。

DAnCE 框架主要包括以下几个服务对象:TargetManager(域管理服务)、NodeManager(节点管理服务)、ExecutionManager(执行管理服务)、RepositoryManager(仓库管理服务)等。下面就这几种服务对象进行研究并对其进行修改和扩展。

3.1 TargetManager

TargetManager 域管理服务用来管理一个域中的各种资源,域即组件所运行的目标环境。域管理服务是一个单例服务,即一个域中只含有一个 TargetManager。它负责分配和释放本域内所有资源,并且在系统运行当中能够及时回收所有可用的资源。本文对 TargetManager 进行修改和扩展,如图 4

所示。

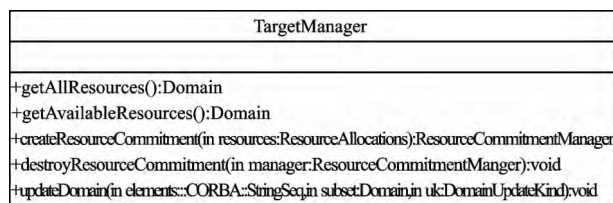


图 4 TargetManager 修改扩展类图

前四个函数专门负责获取本域内所有节点的资源信息,包括内存使用情况、进程数量及详细信息、IP 地址以及该节点上部署的组件的类型和数量,对于系统内节点的实时监控和管理起到至关重要的作用。第五个函数负责及时更新该域内的信息,以便该域内节点成员及时了解哪些节点加入或退出了该域。

3.2 NodeManager

NodeManager 节点管理服务运行在每个节点上的守护进程中,负责管理部署在该节点上的所有组件,不管这些组件是跟那些应用程序相关。组件是通过 NodeApplication 组件服务器进程这个组件容器创建出来。而 NodeManager 来创建并维护 NodeApplicationManager,来管理 NodeApplication 组件服务器进程,从而增强这个节点上共享于应用程序间的组件的重用。本文对 NodeManager 进行修改和扩展,如图 5 所示。

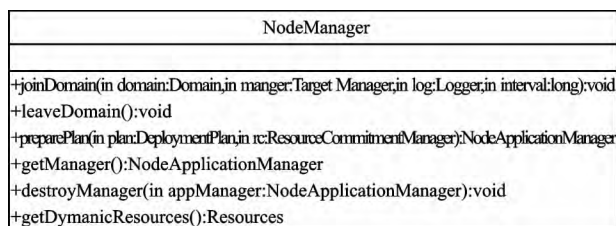


图 5 NodeManager 修改扩展类图

前两个函数用于节点向 TargetManager 域管理服务注册和注销,当注册到某个域中,就受到该域内的 TargetManager 管理,向其报告自己的资源信息及组件部署情况,以便域管理服务对其管理及实时监控。第三个函数是用于部署计划的实施,当部署计划发布后,ExecutionManager 执行管理服务将针对本节点的部署任务发送过来,本节点按照要求完成组件的部署,过程中需要向 RepositoryManager 来下载要部署的组件部件。第四、五个函数负责创建和注销 NodeApplicationManager。第六个函数负责收集本节点的资源信息。

3.3 ExecutionManager

ExecutionManager 执行管理服务作为一个守护进程运行,负责管理一个或者多个域的部署进程. ExecutionManager 通过使用工厂设计模式来产生一系列的 DomainApplicationManager,并协调多个 DomainApplicationManager,以此来管理分布式应用环境中的多个域. DomainApplicationManager 负责管理一个域中的所有组件的部署,创建和管理 DomainApplication,将一个完整的部署计划分成若干个子计划,然后分发到域中的各个节点上去. 本文对 ExecutionManager 进行修改和扩展,如图 6 所示.

ExecutionManager
<pre> +preparePlan(in plan:DeploymentPlan,in rc:ResourceCommitmentManager):DomainApplicationManager +order(in plan:DeploymentPlan,in myComponentID.Resources,in orderID:long):long +getManager():DomainApplicationManager +destroyManager(in manager:DomainApplicationManager):void +getDynamicResources():Resources </pre>

图 6 ExecutionManager 修改扩展类图

前两个函数负责将部署计划分发到各个节点上去,再由 NodeManager 完成本节点的部署任务. 第三、四个函数负责 DomainApplicationManager 的创建和注销. 第五个函数负责收集一个或多个域内的相关信息.

3.4 RepositoryManager

RepositoryManager 仓库管理服务主要管理组件包,并对其他的应用程序提供一系列接口,用户可以通过这些接口完成下载、安装、查询相关组件包的功能.

RepositoryManager 的作用主要体现在:

(1)当分布式节点特别多,且节点的运行平台环境各不相同,某些嵌入式设备没有磁盘这种外部存储系统,这时候这些设备要安装相关组件的话,就只能从 RepositoryManager 下载组件包,进行在线安装,然后完成其功能;(2)能更好地进行组件的版本升级,当有升级后的组件发布时,只需要通知各节点去 RepositoryManager 下载最新的组件包并安装而不需要到各个节点上分别更新组件包;(3)能更好地统一管理组件包;(4)RepositoryManager 在动态迁移中也发挥重要作用.

以上这些服务对象都被实现为 CORBA 对象,可以远程调用它们提供的函数,POA 在创建并激活它们的时候都会将它们注册到命名服务上去.

3.5 具体的开发与部署过程

• 开发过程

(1)编写 IDL 接口文件

将系统需要的功能模块及数据结构写入 IDL 接口文件中;

(2)生成 MPC 文件

%CIAO_ROOT%\bin\generate_component_mpc.pl -n InterfaceName 命令生成 IDL 接口文件相应的 MPC 文件;

%CIAO_ROOT%\bin\generate_component_mpc.pl -p InterfaceName ComponentName 命令生成 IDL 组件文件相应的 MPC 文件;

(3)生成 VS 项目

%ACE_ROOT%\bin\mwcc.pl - type vc9 生成 VS 项目文件;

(4)编译 VS 项目文件中的子项目

主要编译 IDL 接口文件和 IDL 组件文件,可以根据需要适当地调整相应参数;

(5)实现虚方法

IDL 接口文件和 IDL 组件文件编译完成后,根据系统的功能需要对其中的虚方法进行实现来完成组件的开发;

• 部署过程

(1)用 IDL 文件生成 GME 工程文件

用 COSMIC 的 idl_to_picml - x AppName - r 命令生成相应的 GME 工程文件. xme;

(2)部署操作

用通用建模环境 GME^[12] (General Modeling Environment, GME) 打开前面生成的. xme 文件,然后进行部署操作,最后生成 NodeMap. cdd 文件和 deploymentplan. cdp 文件

(3)启动系统平台服务

启动一系列的基础服务,在要部署的每台机器上启动 dance_node_manager,来作为组件服务器管理本机器上部署的所有组件;

(4)加载部署文件

在管理机上使用 dance_execution_manager 加载前面生成的 cdd 文件,然后运行 dance_plan_launcher 加载 cdp 文件完成部署工程;

综上所述,通过上述这些对于 DAnCE 部署框架的修改和扩展,结合分布式应用系统在跨平台的异构环境下运行的需要,以及系统对组件运行状态的实时监控和实现组件的动态迁移的要求,基本实现了基于 DAnCE 的跨平台分布式组件部署的解决

方案.

4 项目案例实现

本节通过一个具体的工程项目案例来验证本文提出并实现的跨平台分布式组件部署解决方案的可行性. 该项目是某电力系统平台的一个原型子系统. 项目运行的相关系统平台: Windows 操作系统、Linux 操作系统以及 Solaris 操作系统; 使用的相关软件: ACE-5.5+TAO-1.5+CIAO-0.5, xerces-c 3.1.1, zlib 1.2.7; 开发工具为 vs2005; 开发语言使用的是 C++; 打包部署环境: GME、UDM、COSMIC. 主要使用 CORBA 和 CCM 技术, 以组件形式提供分布式服务. 组件分别部署在异构分布式环境中, 接受 DDS(Data Delivery Service, OMG 颁布的分布式实时系统数据分发规范) 传送的信息并且进行信息处理, 最终将结果通过 DDS 传送到显示界面上.

4.1 系统的部署与配置

系统的部署配置体系结构及具体的组件部署流程如图 7 所示.

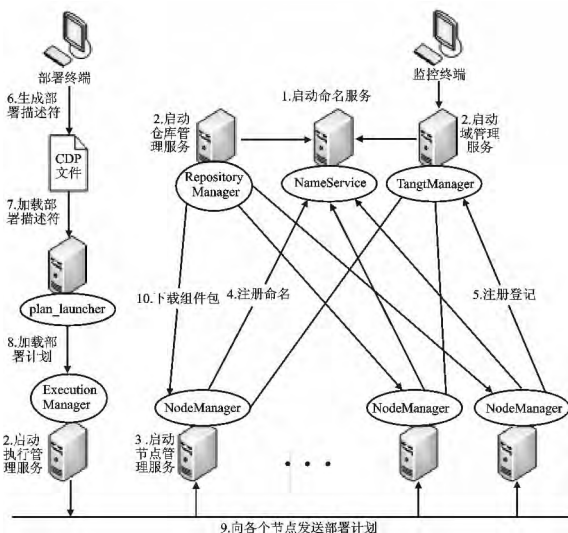


图 7 部署配置体系结构

4.2 节点的实时监控及动态迁移

系统在运行时面临这很多问题, 如组件失效、节点崩溃等. 为了可以及时发现并纠正这些错误, 系统的动态迁移及重部署就显得尤为重要.

在应用程序运行管理平台中, TargetManager 节点定时询问本域内所有 NodeManager 的实时状态并按一定的算法处理查询结果, 若询问过程失败即认为该节点失效. 扩展后的框架提供了运行时管理服务的接口, 基于这些服务接口, 可以实现系统的动态迁移及重部

署. 具体过程如图 8 所示.

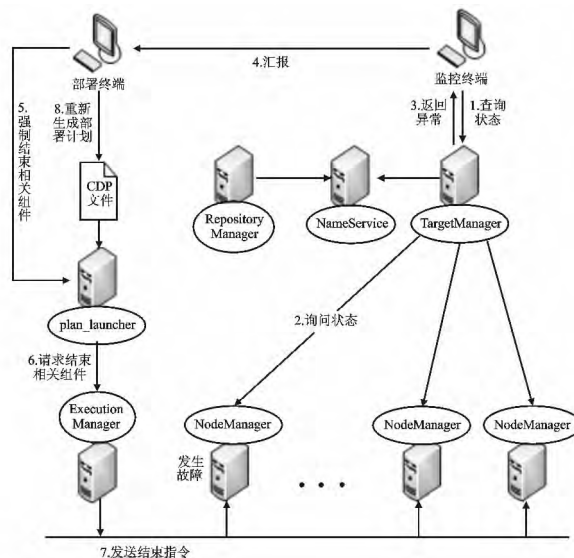


图 8 实时监控及动态迁移

4.3 实验结果

通过在由三台 Windows 主机、三台 Linux 主机以及两台 Solaris 主机组成的模拟分布式环境中分别部署扩展和修改后的 DAnCE 框架的相应基础管理服务以及不同功能的组件, 几台主机协同完成了系统所需要的业务功能, 不同主机上的组件互相之间正常地传递相关信息, 各个基础管理服务能正常的管理各个节点的实时状态信息都在监控终端上通过图形化的界面展现出来, 并模拟了其中一台主机出现故障后, 实现故障机上的组件动态地迁移到一台备用主机上面, 系统继续处于正常稳定的运行状态, 成功验证了本文所提出并实现的跨平台分布式组件部署解决方案的可行性, 进而成功运用于实际的某电力系统平台中.

5 结束语

本文设计并实现了一种基于 DAnCE 的跨平台分布式组件部署的解决方案, 并成功运用到实际的分布式应用系统中. 该解决方案不仅能够成功地实现在跨平台分布式应用中组件的部署与配置, 而且在出现故障时能实现动态迁移和组件的重部署, 具体非常良好的稳定性和容错性.

下一步的研究是将组件的部署配置及动态迁移与事务、安全、事件等方面的要素结合起来, 进一步增强分布式应用系统的稳定性和可靠性.

(下转第 114 页)

地实现曲线分割,分割效果更接近车辆的实际轮廓.在车辆距离过近和相互遮挡下,车牌对应凹点连接线产生缺帧.采用传统方法无法得到遮挡区域内各个车辆的准确轮廓,车牌重构效果不好,而采用本文算法,通过空间亮度均衡化平滑预处理,采用曲线分割方法准确分割遮挡车辆,提高了图像的三维重构性能.

为了定量分析本文算法的性能,采用本文算和传统算法,通过选取大量交通视频图像进行实验.实现对违章车辆的车牌检测,以查准率和查全率为指标,查准率反映的是虚检率,值越大虚检率越低;查全率反映的是漏检率,本文算法的 F 量测的均值在 0.9 左右,而文献[11]中的处理结果在 0.8 上下且很不稳定,采用本文算法,能有效提高对遮挡车辆的检测准确率,查全率和查准率都高于传统算法.

5 结束语

本文提出一种基于空间亮度均衡化平滑和曲线分割的缺帧环境下车牌图像的三维重建模型.设计车辆的区域检测算法,提取遮挡车辆分割线,进行空间亮度均衡化处理,对遮挡车牌进行曲线分割,以此为基础进行车牌的三维图像重建,提高对车牌的检测识别能力,仿真结果表明,本文算法能实现对缺帧环境下的车牌图像三维重构,效果较好,有效提高对遮挡车辆的检测准确率,查全率和查准率都高于传统算法,性能优越,展示了较大的实践价值.

参考文献:

- [1] 孟春芝,何凯,焦青兰.自适应样本块大小的图像修复方法[J].中国图象图形学报,2012,17(3):337-341.
- [2] 刘怀愚,李璟,洪留荣.利用局部特征的子空间车辆识

别算法[J].计算机工程与应用,2010,46(30):156-180.

- [3] 杨靖宇,张永生,邹晓亮,等.利用暗原色先验知识实现航空影像快速去雾[J].武汉大学学报:信息科学版,2010,35(11):1292-1295.
- [4] 雷亮,汪同庆,杨波.图像关联规则挖掘研究[J].计算机应用研究,2009(6):2374-2376.
- [5] 单家凌.基于无线网络车牌识别系统识别算法的研究[J].计算机测量与控制,2011,19(1):124-126.
- [6] FAN N. Object classification and occlusion handling using quadratic feature correlation model and neural networks[J]. International Journal of Pattern Recognition and Artificial Intelligence, 2011, 25(3):287-298.
- [7] 申晓磊,周武能.压缩感知在车牌识别系统中的应用研究[J].科学技术与工程,2012,12(19):4797-4803.
- [8] 郭荣艳,胡雪惠.BP神经网络在车牌字符识别中的应用研究[J].计算机仿真,2010,27(9):299-301.
- [9] 蔡勇智.一种改进的神经网络算法车牌识别算法研究[J].科技通报,2012,28(10):128-130.
- [10] 焦波,闫了了,李伟.实时部分遮挡车辆分割及阴影消除[J].小型微型计算机系统,2012,8(33):1825-1829.
- [11] 吴忻生,刘洋,戚其丰.基于凹性分析的粘连车辆分割[J].计算机应用研究,2012,29(1):344-347.
- [12] 魏玉强,王成儒.多车辆跟踪时分割粘连车辆的方法[J].电视技术,2009,33(11):107-109.

作者简介:

郭慧玲 女,(1979-),硕士研究生,讲师.研究方向为图像处理、数据挖掘等. E-mail:332897817@qq.com

乔蕊 女,(1983-),研究生,讲师.研究方向为计算机控制、传感器技术、图像处理.

(上接第 109 页)

参考文献:

- [1] 张涌,王渊峰,钱乐秋.一个集成式的软件构件描述框架[J].计算机学报,2002,25(5):502-507.
- [2] Object Management Group. CORBA Component Model Specification[S]. 2006.
- [3] 史殿习,丁博,崔巍,等.轻量级构件部署与配置工具的研究与实现[J].计算机科学,2011(06):165-168.
- [4] Object Management Group. V4.0 Deployment and Configuration of Component-based Distributed Applications Specification[S]. 2006.
- [5] G Deng, D C Schmidt, A Gokhale. CaDAnCE: a criticality-aware deployment and configuration engine[C]// Proceedings of the 11th IEEE International Symposium

on Object/Component/Service-oriented Real-time Distributed Computing. Orlando, Florida; IEEE, 2008.

- [6] The Deployment And Configuration Engine[EB/OL]. (2006-10-09)[2014-04-20]. http://www.dre.vanderbilt.edu/~schmidt/DOC_ROOT/CIAO/docs/index.html

作者简介:

王涵 男,(1990-),硕士研究生.研究方向为分布计算、计算机仿真. E-mail:wanghan_work@163.com

李晶 男,(1967-),博士,教授,博士生导师.研究方向为计算机仿真、多媒体技术等.

李石君 男,(1964-),博士,教授,博士生导师.研究方向为互联网搜索与挖掘、数据挖掘等.