面向云计算的分布式应用自动部署框架

李 超[1],花 磊[2],宋云奎[2]
(1. 新华通讯社 中国经济信息社有限公司,北京 100083;
2. 中国科学院 软件研究所,北京 100190)

摘 要:云计算环境下的分布式应用规模巨大、交互复杂、层次众多,增加了应用部署的复杂度。同时,应用通常同时部署在多个云平台上,而异构云服务 API 缺少统一标准,导致了多云之间不可互操作。针对以上问题,面向云计算环境,提出一种细粒度弹性应用自动部署框架 CADep,以管理云计算平台内部服务与基于云平台的业务组件之间的交互关系,并连接各种部署进程和资源。该框架使用轻量级的微内核实现了应用部署的核心机制,并通过可定制组件扩展部署功能。同时,通过分层描述应用及其执行环境,实现了细粒度的资源管理。实验结果表明,与当前分布式应用部署框架相比,CADep 具有较短的部署时间,有效减轻了部署的工作量,降低了部署的难度。

关键词:分布式应用;应用部署;部署框架;云计算
中图分类号:TP39 **文献标识码**:A **文章编号**:1673-629X(2018)06-0012-05
doi:10.3969/j.issn.1673-629X.2018.06.003

Deployment Framework for Distributed Applications in Cloud Computing

LI Chao[1],HUA Lei[2],SONG Yun-kui[2]
(1. China Economic Information Service Co.,Ltd.,Xinhua News Agency,Beijing 100083,China;
2. Institute of Software,Chinese Academy of Sciences,Beijing 100190,China)

Abstract:Distributed applications in cloud computing have many components,complex interactions and various layers,so deploying these applications is difficult. Furthermore,multiple cloud computing platforms have different service APIs,so these cloud services cannot cooperate with each other. For this,we propose a fine-grained flexible deployment framework CADep for distributed applications in cloud computing to manage cloud services and interactions with application components,and then integrate various deployment processes and resources. CADep with a light-weight micro-kernel uses a component-based framework to improve the extensibility of application deployment. Furthermore,it uses the hierarchy description of applications and contexts to manage components in a fine granularity. Experiment shows that compared with typical deployment frameworks,CADep has short deployment time,reduces work amount and decreases operational difficulty.

Key words:distributed application;application deployment;deployment framework;cloud computing

0 引 言

在云计算环境下,分布式应用通常具有多个应用组件,同时部署在众多节点,因此云应用的配置管理面临着巨大挑战。首先,云计算引入了虚拟机(VM),增加了配置管理的复杂度;同时,应用通常需要同时部署在多个云上,而云计算平台的 API 缺少统一标准,导致了多云之间不可互操作。以云计算环境下典型的三层架构电子商务应用 Rubis 部署为例,分析分布式应用部署所面临的问题。Rubis 是基于 Servlet 的 JavaEE 应用,实现了类似于 eBay 的网上交易应用,提供了交互型事务处理功能,如注册新用户、浏览、购买商品、付款。管理员使用 Apache 作为 Web 服务器分发请求,Tomcat 作为应用服务器处理请求,MySQL 作为数据库服务器提供数据存取。客户请求处理分为与资金相关的关键型业务处理和与浏览相关的非关键业务处理。在实际应用中,通常将关键业务部署在公司的私有云计算平台,由本地主机集群组成;而非关键业务,部署在接近客户的公有云中托管。公司会根据云计算

收稿日期:2017-08-08 **修回日期**:2017-12-13 **网络出版时间**:2018-02-24
基金项目:国家自然科学基金;金融信息平台“新华 08”金融财经数据云服务平台科研专项(TC16037XX/01)(61602454)
作者简介:李 超(1984-),男,工程师,研究方向为系统架构设计、微服务与容器管理、DevOps 技术。

市场上的虚拟机租用价格,将应用运行在不同的公有云计算平台(如阿里云、百度云),当价格调整则将应用从一个云迁移到另一个云并进行自动化配置部署。同时,需要细粒度的应用管理,这样可以根据更新或迁移需要来决定操作 WAR 包还是其中的某个 Servlet。

针对该典型的应用部署实例,当前的分布式应用部署技术无法适应云计算环境,主要表现在以下几点:在云计算环境中,应用在 VM 中运行,而 VM 运行在物理机(PM)上,需要考虑多层部署问题;应用的运行环境是动态的,在整个生命周期内会在不同的云之间迁移,需要支持混合云部署;多个应用组件在不同云平台上部署,并且组件依赖关系存在着多样性,需要支持细粒度的应用部署管理。因此,文中提出了一个面向多云环境的可伸缩细粒度的应用自动部署框架 CADep,用来管理云平台内部与基于云的业务单元之间的交互,以连接各种部署进程和相关资源。CADep 主要设计思想如下:使用分层描述方法对应用和执行环境进行细粒度的描述;提供一组可重用组件,其中每个组件执行特定功能;实现了基本部署管理机制以感知环境,并根据环境与业务应用做出相应调整。

1　相关工作

当前的许多应用部署研究都是专门针对单个云平台或单一类型应用。文献[1]提出了基于 Xen 的云平台自动部署框架。文献[2-3]提出了 PM、VM 和应用等三个层次的云应用部署的描述语言。文献[4]提出了与云平台无关的可移植的应用部署的编排语言。文献[5]提出一种自动化部署与监控 Hadoop 平台的系统方案,自动化部署 Hadoop 平台,并监控集群及其中各个服务的状态。文献[6]提出一种结合 Docker 容器技术部署集群的解决方案,把 Ambari 及其运行环境和配置构建成 Docker 镜像,并把多节点容器的运行和 Hadoop 集群的部署过程写成 Shell 脚本,通过命令实现集群的自动化部署。文献[7]对 Apache ODE 结构以及部署、执行 BPEL 流程的原理进行分析,设计算法自动生成部署描述文件和服务的 WSDL 描述等,从而实现 Apache ODE 引擎环境下服务组合的自动化部署和执行。文献[8]基于用户审计与 SSH 单向信任安全机制,着力于规范化、流程化,设计自动化部署方案。文献[9]提出一种可运营的网络自动化部署架构,并对电信运营商开展多业务、多租户承载的云资源池网络自动化部署技术选型给出了建议。文献[10]利用可扩展的通用部署引擎来管理各种应用,从而减少生成定制引擎的步骤以减少构建开销。Bootware[11]创建部署引擎以部署应用实例,其使用通用部署引擎。文献[12]复用现有多样化开源系统,将不同类型的部署可执行程序转换为基于 TOSCA 标准的构件。云提供商提供了建模和编排工具[13],同时提供了 PaaS(Platform-as-a-Service)服务[14],调用该服务可以部署和管理应用实例,而不需要显式建模应用拓扑。文献[15]提出一种便于移动应用开发和部署的整合方案,从设备属性、用户偏好以及 QoS 需求等多重维度入手,采用体系结构驱动的方法对应用进行建模,并生成满足用户个性化需求的部署方案。以上方法难以实现多云混合部署场景,将应用或部分应用自动化迁移非常困难。同时,必须通过开发自定义的粘合代码来手动包装,因此只适用于特定部署场景,缺乏通用性。

2　分布式云应用自动部署框架 CADep

2.1　架构设计

CADep 设计目标是能够描述复杂分布式应用,并自动实现整个应用或各应用组件的灵活部署。该框架提供了微内核机制,通过动态插入新的功能组件来应对新的应用和部署环境,从而具有灵活性,以支持部署规模的伸缩以及应用的动态重配。图 1 描述了 CADep 架构,由以下几个模块构成:

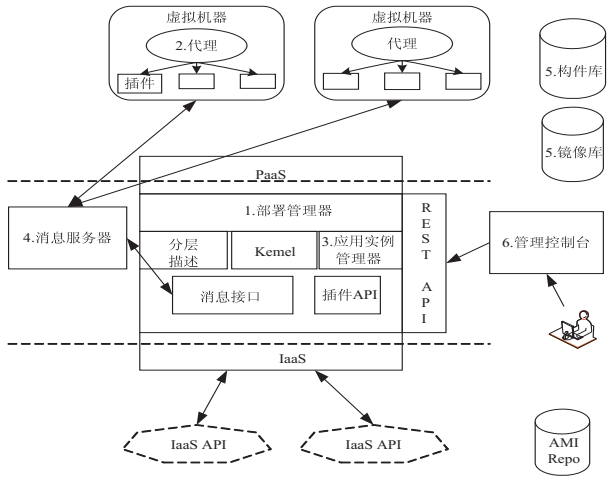

图 1　CADep 系统架构

(1)部署管理器(DM):提供 VM 接口以管理 VM 及其应用,在 PM 中实例化 VM 及其应用。

(2)代理:部署在 VM 的软件组件,使用插件对应用实例进行部署和管理,并支持不同部署技术(如 Ansible、Puppet)。CADep 使用轻量级微内核,插件可以灵活加入内核,通过异步消息实现服务器间通信。

(3)应用实例管理器:作为 CADep 的插件,在不同的软件平台上监控应用实例并管理其生命周期。

(4)消息服务器:负责在 DM 和代理之间通过消息服务器进行异步通信,包括消息定义、消息服务器与应用实例的交互接口。

(5)构件和镜像库:可以从本地或公共存储库中管理和检索软件包与 VM 镜像。镜像库将每个需要的

VM 镜像映射到相应的基础设施组件,所需的 VM 镜像可以是公共 VM 镜像库中的可用镜像或者是人工创建的镜像文件。

(6)管理控制台:基于 Shell 的控制台和基于 AngularJS 的 Web 应用,提供不同用户界面,将 Java Bean 转换为 JSON 并通过 REST 与部署管理器进行交互。

CADep 负责启动 VM 并部署软件,在软件组件之间动态解析依赖关系,更新组件配置,并启动所有组件。CADep 以非侵入方式管理应用生命周期,支持应用组件弹性伸缩的热重配,并且保持组件启动或停止时的一致性。应用组件为其他应用组件提供服务,并且依赖于其他应用组件所提供的服务。应用组件使用消息队列进行通信,并根据部署或启动的服务执行相应的操作,而这些操作由插件实现(如 Java 的 Servlet、OSGi 的 Bundle)。CADep 采用基于 RESTFul 的分布式通信技术,支持 IaaS 层或 PaaS 层的云应用部署,能够满足多云需求,如组件细粒度的分层描述,动态的依赖管理,并发的组件部署,多云的分布式部署,以及制定可配置的部署计划。

2.2 部署方法

2.2.1 系统配置

CADep 将分布式应用作为相互调用的组件集合,并且定义相关导入\导出的相关变量。典型的分布式架构应用(如 Rubis)通常包括表示层(如 Apache)、逻辑层(如 Tomcat)和数据层(如 MySQL)。其中,组件是指 Apache,而实例是指部署在特定主机的 Apache 实例,组件之间交换数据是字符串或结构化数据。分布式应用的组件需要导入\导出变量(如 IP 地址或端口号),需要来自其他应用的变量为“导入变量”,其他应用组件需要的变量为“导出变量”,例如,Apache 导入来自于 Tomcat 变量,即应用服务器的 IP 地址和端口号。此外,组件可以继承导入\导出变量的默认值,如 Tomcat 组件可以继承“应用服务器”组件的属性。文中定义了组件与实例的导入\导出变量,表 1 给出了 Apache 和 MySQL 的组件配置,其中,安装参数是强制的组件属性,指定了 CADep 的插件以管理部署组件实例。文中采用 Ansible 作为插件,导入变量列出了启动前需要被解析的变量。例如,Tomcat 导出 IP 变量,依赖组件将导入该变量;MySQL 导出了其他组件所需要的变量,而无需从其他组件导入变量。

表 1 组件配置

部署组件	安装参数	导入变量	导出变量
Apache	Ansible	Tomcat. portAJP, Tomcat. ip	X
MySQL	Bash	X	MySQL. ip MySQL. port

配置文件包括应用描述符与应用关系图,前者包含应用的元信息(名称、版本限定符),后者描述了软件组件(如虚拟机、云平台、应用包),并且定义了包含和运行时关系。包含关系描述了可以在一个组件上部署另一个组件,例如,在 VM 上部署 Tomcat 服务器,或者在 Tomcat 服务器上部署 Web 应用。运行时关系描述了共同工作的组件,例如,Web 应用需要数据库(包括数据库的 IP 地址和端口)。这些信息通常采用硬编码,而 CADep 在运行时解析,通过配置或管理 API 来更新组件(如 JMX、REST),可以并行部署 Apache、Tomcat 和 MySQL。Tomcat 部署时,在获得数据库位置之前,将无法启动。数据库部署并启动后,CADep 将更新 Tomcat 配置,以便获得 MySQL 位置,从而形成运行时间依赖关系。

CADep 列出了开发人员希望部署或可能部署的应用组件,包括部署的根节点(如 VM、设备、远程主机)、数据库、应用服务器、应用模块(如 WAR,ZIP)。将应用作为组件的层次结构,准确追踪系统中实现的实例,从而在动态部署中做出决策。分层模型解析包含关系,以及导入\导出变量模型的运行时关系,同时,使用图来决定哪些组件可以实例化以及如何部署。图中建模的内容是各种粒度的组件,将应用组件进行捆绑,例如将给定的 WAR 与应用服务器相关联。

CADep 为部署的应用提供三个文件。第一个文件包含 CADep 配置的应用描述文件,该文件描述了应用信息,如应用名、功能描述、模型文件的位置等;第二个文件描述了应用组件之间的关联图,其中,组件是安装的软件,也可以是部署的 VM 等;第三个文件提供了部署组件所需的资源(如脚本、软件包、配置文件)。在运行时,CADep 首先实例化组件,对实例进行分层结构描述,然后对其配置以准备部署。如果根组件带有子组件,那么必须在根实例中定义子实例,并且将实例的属性覆盖组件的属性,例如,如果 Tomcat 组件导出具有默认值 8080 的端口属性,则实例可能会以实际端口(如 8081)覆盖。

2.2.2 初始部署

文中使用三层架构的应用实例来描述 CADep 的工作方式。图 2 描述了组件之间的依赖关系,在云计算平台弹性伸缩的场景中,可以对多个 Tomcat 节点进行增\删,以适应负载的变化,但需要对 Apache 节点进行动态重新配置,使得请求转发模块可以感知所有可用的 Tomcat 节点。CADep 在 3 个独立的虚拟机中部署 Apache、MySQL 和 Tomcat,并在特定依赖关系下更新配置文件,例如,CADep 发现 MySQL 的 IP 地址和端口号时,发送到 Tomcat 节点,这样它就可以更新配置并启动。VM 支持 Apache、Tomcat 或 MySQL 组件

的部署,并且每个组件都以导入\导出形式进行描述。有了该描述信息,CADep 就能够感知何时可以启动部署组件,通过导入\导出配置信息来实现组件之间的数据交换,同时,实现应用组件的生命周期管理,例如,在导入配置信息时启动组件。

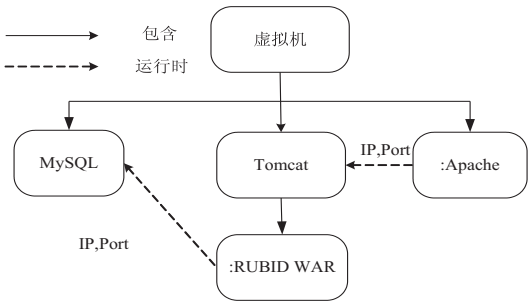

图 2　组件描述

2.2.3　动态重部署

当系统处于运行状态,创建、更改或删除实例时,需要指定要实例化或修改的组件,重命名组件实例并定义部署位置。例如,由于负载增加,需要自动添加新 Tomcat 实例的 VM,创建新的 VM 实例,代理会监测到运行环境的变化,并向 DM 发送通知。CADep 提供了规则来响应检测到的变化,DM 执行相应动作对环境变化做出响应。例如,CADep 从本地 Zabbix 的 Agent 中获取 CPU 等度量,如果度量超过阈值,则将运行状态发送给 DM。DM 调用请求处理程序来响应该消息,添加完整路径以及两个实例到应用模型中。在添加过程的开始阶段,这两个新程序都没有启动,后来由 DM 部署并启动它们。DM 首先创建 VM,向 VM 发送部署命令,并在上面部署新的 Tomcat 实例;然后,CADep 代理将该实例的相关信息进行发布,即新 Tomcat 实例的 IP 地址和端口号;Apache 负载均衡器会收到该信息,部署在该 VM 上的代理将调用一个 CADep 插件来更新 Apache 服务器的配置文件,这样负载均衡器就可以感知这两个 Tomcat 服务器。

3　实例研究

3.1　实验 1:部署时间比较

3.1.1　实验环境

CADep 解析组件的动态依赖关系,同时并发部署组件。CADep 与典型的部署框架 Cloudify、RightScale 和 Scalr 进行部署时间的对比,每个框架分别重复部署 10 次。文中选择阿里云作为部署应用的云计算平台,使用 Ansible 作为 CADep 的安装插件,部署 Rubis 应用。如果用户可以使用 Web 浏览器连接并登录 Administration 页面,并创建数据库,则表明部署成功。每个框架都有自己的生命周期管理,将所有这些状态分成启动和运行等两个阶段。

(1)启动阶段:客户端发送“部署”请求到 DM,DM 处理请求,并获得脚本和其他必要的文件,启动脚本(包括设置代理、向 DM 发送启动信息等)。具体步骤包括:资源供给、启动虚拟机、部署 Apache、MySQL、Administration。

(2)运行阶段:系统执行脚本,在服务器上运行服务或组件,包括预设状态(下载 tarball、传输脚本和必要的文件到 VM、准备运行时环境等),配置,安装(将资源和配置文件放置到正确位置,并设置权限等),预启动(建立软件组件的依赖关系),启动,生效(更新变量,配置监控)。具体步骤包括:启动 Apache、MySQL、Administration。

3.1.2　实验结果与分析

表 2 给出了应用部署时间的比较,部署时间为从启动到最后一个组件正确安装的时间。实验结果表明,CADep 的部署时间明显比其他框架都要低;Scalr 与 CADep 的部署阶段的时间最为接近;运行阶段的时间比 CADep 的少,这是由于其不解析服务间的动态依赖关系;在 Scalr 中,组件之间的依赖关系是通过 WebUI 中的配置来交换部署信息,虽然时间较短,但操作复杂,易于出错。

表 2　应用部署时间比较　　s

	CADep	Cloudify	RightScale	Scalr
启动阶段	164	227	178	182
运行阶段	209	384	286	197
总计	373	611	464	379

3.2　实验 2:部署工作量

3.2.1　实验环境

该实验基于 OSGi 的 SPECjms2007 基准测试应用,部署在阿里云的 VM 上。OSGi 应用通常由一个或多个 OSGi 组件构成,以提供应用的运行时环境并管理 OSGi 组件,如 Joram、JNDI。使用阿里云的两个 VM 实例,分别选择 Felix 或 Equinox 作为底层 OSGi 框架以管理 Joram 实例、OSGi JNDI 实例和 JMS OSGi 客户端,由“osgi-bundle”安装程序管理。由于 Cloudify 提供了脚本语言,可以用来表达分布式应用结构,故选择了 Cloudify 作为比较对象,但 CADep 以分层结构描述应用,而 Cloudify 则以扁平结构描述应用。

3.2.2　实验结果与分析

CADep 用户只需要为每个阿里云 VM 实例分别编写一个部署计划,而多个 OSGi 组件(包括 Joram、CADep、JNDI)可以重用相同的部署计划。然而,Cloudify 用户则需要为 VM 实例和每个组件分别编写一个对应的部署计划,总共需要编写六个部署计划,是 CADep 编写部署计划工作量的两倍。该实验结果表明,CADep 具有细粒度描述组件的能力,具有较好的

适用性,能够减少开发部署计划的工作量。

3.3 实验 3:部署复杂度

3.3.1 实验环境

在多云平台上部署流数据处理系统 Storm 集群以执行流数据处理,将 CADep 和根据说明书手动部署时间进行比较。Storm 由 Zookeeper 集群、Nimbus 服务器、Storm 管理器组成,需要安装 JZMQ、ZeroMQ 和 Python。该实验在多云环境中进行,使用两个公有云(阿里云和百度云)和一个私有云(Xen Serve)。CADep 为这些云提供了三个 IaaS 插件,以协调 IaaS 平台间的服务调用,其中每个插件都需要实现 API Java 接口。实现这些插件的代码行数分别为:阿里云 196 行代码,百度云 387 行代码,而 Xen Server 是 132 行代码。Zookeeper 安装在阿里云,Nimbus 安装在百度云,Storm 安装在 Xen Server 私有云平台。

3.3.2 实验结果与分析

Storm 的在线安装指南有 8 页长度的篇幅,同时也有许多依赖的外部文档链接。要求一名未曾安装过 Storm 的软件运维工程师按照安装指南手动对其进行配置,其第一次安装用了 7 小时 18 分钟,第二次安装用了 4 小时 12 分钟,而第三次安装则用了 1 小时 14 分钟。理解描述不够详细的安装说明、解决环境问题、寻找\下载所需的依赖包以及调试等操作消耗了其大量时间。同样的工作是由另一位从未配置安装过 Storm 的软件运维工程师使用 CADep 完成,时间主要用于编写 Zookeeper, Nimbus, Supervisors, JZMQ, ZeroMQ 和 Python 的部署计划。他编写 140 行左右的 Storm 组件的部署脚本,配置之后可以通过 CADep 对 Storm 进行管理(部署、启动、停止、反部署、更新),并自动连接到其他应用。部署脚本开发时间大约是 2 小时 35 分钟,其中,38 分钟用来设计组件,75 分钟用来编写脚本,42 分钟用于调试和测试。如果需要从 Internet 上下载所需的包,安装 Storm 需要 26 分钟;如果从本地存储库检索包,安装 Storm 大约 9 分钟。使用 CADep 自动化安装 Storm,可以使得开发者在多云上部署其应用,而无需了解实现的技术细节。同时,CADep 保证了过程的可重复,为大规模系统部署提供了便利。该实验结果表明,CADep 能够支持多云分布式环境的部署,同时降低了分布式应用部署的复杂性。

4 结束语

CADep 是细粒度可伸缩的多云部署框架,基于轻量级微内核实现了基本部署机制,使用动态可插拔组件实现了应用部署的改进和扩展。同时,框架多层次细粒度描述应用和执行环境,实现了组件(如 PM、VM、应用)的精细管理。实验结果表明,与典型的分布式应用部署框架相比,CADep 具有较短的部署时间,有效降低了部署的工作量,减轻了部署的难度,适用于多云环境下复杂分布式应用的高效部署。

参考文献:

[1] ZHANG Youhui, LI Yanhua, ZHENG Weimin. Automatic software deployment using user-level virtualization for cloud-computing [J]. Future Generation Computer Systems, 2013,29(1):323-329.

[2] DE ALFONSO C, CABALLER M, ALVARRUIZ F, et al. Infrastructure deployment over the cloud[C]//Third international conference on cloud computing technology and science. Athens, Greece: IEEE, 2011:517-521.

[3] SEINTURIER L, MERLE P, FOURNIER D, et al. A component-based middleware platform for reconfigurable service-oriented architectures[J]. Software Practice & Experience, 2012,42:559-583.

[4] BINZ T, BREITER G, LEYMANN F, et al. Portable cloud services using TOSCA[J]. IEEE Internet Computing, 2012, 16(3):80-85.

[5] 于金良,朱志祥,李聪颖. Hadoop 平台的自动化部署与监控研究[J]. 计算机与数字工程, 2016, 44(12): 2457-2461.

[6] 李 杰,刘广钟. Hadoop 分布式集群的自动化容器部署研究[J]. 计算机应用研究,2016,33(11):3404-3407.

[7] 黄 亮,姚放吾,金仙力. Apache ODE 环境下 Web 服务组合技术的研究[J]. 计算机技术与发展,2011,21(7):98-100.

[8] 王海侠,吴爱华,曾卫明. 基于 JBoss 和 Tomcat 的自动化部署研究[J]. 现代计算机,2015(13):38-42.

[9] 赖培源,马卫民,刘 艺,等. 云资源池网络自动化部署技术研究与实践[J]. 电信科学,2015,31(7):96-103.

[10] LU Hongbin, SHTERN B, SIMMONS M, et al. Pattern-based deployment service for next generation clouds[C]// Ninth world congress on services. Santa Clara, CA, USA: IEEE,2013:464-471.

[11] VUKOJEVIC-HAUPT K, KARASTOYANOVA D, LEYMANN F. On-demand provisioning of infrastructure, middleware and services for simulation workflows[C]//6th international conference on service-oriented computing & applications. Koloa, HI, USA: IEEE, 2013:91-98.

[12] ANDRIKOPOULOS V, REUTER A, GOMEZ-SAEZ S, et al. A GENTL approach for cloud application topologies [C]//Service-oriented and cloud computing. [s. l.]: [s. n.], 2014:148-159.

[13] ROSNER T. Learning AWS OpsWorks[M]. UK: Packt Publishing Ltd, 2013.

[14] COUTERMARSH M. Heroku cookbook[M]. UK: Packt Publishing Ltd, 2014.

[15] 张晓薇,曹东刚,陈向群,等. 一种网络化移动应用部署方案优化方法[J]. 软件学报,2011,22(12):2866-2878.