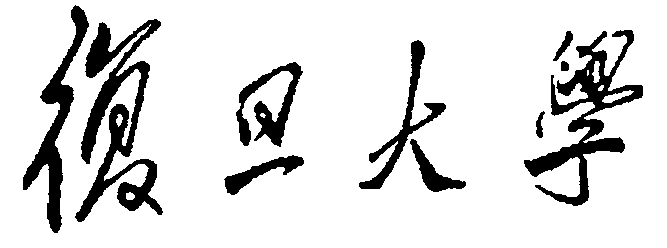
|  |  |
| --- | --- |
|  | 学校代码： 10246 |
|  | 学 号： |



|  |
| --- |
| 硕 士 学 位 论 文 |

（专 业 学 位）

|  |
| --- |
| **基于kvm虚拟化技术的**  **运维自动化平台**  **的设计与实现** |

|  |  |
| --- | --- |
| 院 系： | 软件学院 |
| 专 业： | 软件工程 |
| 姓 名： |  |
| 指 导 教 师： |  |
| 完 成 日 期： |  |

目录

[摘 要 1](#_Toc431938335)

[Abstract 2](#_Toc431938336)

[第一章 绪论 4](#_Toc431938337)

[1.1 课题研究背景 4](#_Toc431938338)

[1.2 课题研究的意义 4](#_Toc431938339)

[1.3 国内外研究现状 5](#_Toc431938340)

[1.4 本文主要工作 6](#_Toc431938341)

[1.5 论文的结构安排 7](#_Toc431938342)

[第二章 相关技术介绍 8](#_Toc431938343)

[2.1 KVM虚拟化技术 8](#_Toc431938344)

[2.1.1 虚拟化技术 8](#_Toc431938345)

[2.1.2 KVM架构 8](#_Toc431938346)

[2.1.3 KVM与Xen的比较 9](#_Toc431938347)

[2.1.4 KVM安装和使用 10](#_Toc431938348)

[2.2 Open vSwitch虚拟交换技术 12](#_Toc431938349)

[2.3 操作系统的自动化网络安装技术 13](#_Toc431938350)

[2.4 基于基线的应用自动化部署技术 15](#_Toc431938351)

[第三章 需求分析与设计 17](#_Toc431938352)

[3.1 项目背景 17](#_Toc431938353)

[3.2 系统详细需求 17](#_Toc431938354)

[3.2.1 硬件资源管理模块的需求 17](#_Toc431938355)

[3.2.2 虚拟资源管理模块的需求 18](#_Toc431938356)

[3.2.3 资源分配管理模块的需求 18](#_Toc431938357)

[3.2.4 环境部署模块的需求 19](#_Toc431938358)

[3.3 用户分类 20](#_Toc431938359)

[3.4 系统用例设计 20](#_Toc431938360)

[3.4.1 角色识别 20](#_Toc431938361)

[3.4.2 用例识别 21](#_Toc431938362)

[3.4.3 用例描述 21](#_Toc431938363)

[第四章 运维自动化平台的设计和实现 26](#_Toc431938364)

[4.1 系统结构总体设计 26](#_Toc431938365)

[4.2系统各层实现技术方案设计 28](#_Toc431938366)

[4.2.1 前端运维管理平台WEB系统 28](#_Toc431938367)

[4.2.2 后端控制端程序 29](#_Toc431938368)

[4.2.3 前后端通讯协议 30](#_Toc431938369)

[4.2.4 公用资源包管理 30](#_Toc431938370)

[4.3 硬件资源管理模块的设计和实现 30](#_Toc431938371)

[4.3.1 网络安装及主机探测的流程设计 31](#_Toc431938372)

[4.3.2 硬件资源监控及报警流程设计 32](#_Toc431938373)

[4.3.3 硬件资源管理类设计和实现 34](#_Toc431938374)

[4.4 虚拟资源管理模块的设计和实现 35](#_Toc431938375)

[4.4.1 kvm虚拟机的定制及创建功能的设计 35](#_Toc431938376)

[4.4.2 kvm虚拟机的管理的功能设计和实现 36](#_Toc431938377)

[4.4.3 kvm虚拟化的应用实现 37](#_Toc431938378)

[4.4.4 虚拟资源管理对象交互接口的设计 38](#_Toc431938379)

[4.5 资源分配管理模块的设计和实现 39](#_Toc431938380)

[4.5.1 运维环境申请流程的设计 39](#_Toc431938381)

[4.5.2 运维环境回收流程的设计 40](#_Toc431938382)

[4.5.3 运维资源管理流程类设计和实现 41](#_Toc431938383)

[4.6 环境自动化部署模块的设计和实现 42](#_Toc431938384)

[4.6.1 环境部署流程设计 43](#_Toc431938385)

[4.6.2 主机定制问题的描述与分析 44](#_Toc431938386)

[4.6.3 主机定制问题的解决方案 44](#_Toc431938387)

[4.6.4 应用自动化部署问题的解决方案 47](#_Toc431938388)

[4.6.5 应用自动化部署流程的设计 48](#_Toc431938389)

[4.7 本章小结 49](#_Toc431938390)

[第五章 系统运行分析 51](#_Toc431938391)

[5.1 系统物理部署 51](#_Toc431938392)

[5.2 系统运行平台 52](#_Toc431938393)

[5.3 系统用户体验效果分析 53](#_Toc431938394)

[5.4 系统使用前后效果分析 56](#_Toc431938395)

[5.5 与类似的方案比较 58](#_Toc431938396)

[第六章 总结与展望 59](#_Toc431938397)

[6.1 本文工作总结 59](#_Toc431938398)

[6.2 展望 60](#_Toc431938399)

[参考文献 61](#_Toc431938400)

[致 谢 63](#_Toc431938401)

# 摘 要

随着云计算时代的到来，虚拟化技术在运维工作中变得不可缺少。虚拟化技术的发展，带来了虚拟服务器、虚拟网络等资源越来越多，越来越难以管理的问题。特别是对于拥有很多自有服务器的公司来说，如何规范管理越来越多的虚拟资源，成为一个亟待解决的问题。虚拟资源如何高效的集成安装使用人员需要的应用，分配给使用人员，并且在合适的时间回收资源，也是需要运维自动化平台解决的问题。对于很多公司来说，稳定的基于虚拟化技术的管理运维平台，能使开发测试流程变得更高效，能使服务器发挥更优的性能，能最大程度的降低运维成本。

针对这些问题，本文提出了基于KVM虚拟机技术的运维自动化平台解决方案。本文提出的解决方案能帮助系统管理员管理和统计现有硬件资源，并通过KVM虚拟化技术在现有的硬件资源基础上创建虚拟资源，并快速自动安装部署指定的服务，分配给最终用户。

本文论述的运维自动化平台系统是按照分层思想来逐层设计和开发的。本文对于整个系统的逐层技术选择进行了分析，表示层用B/S模式，服务层用python脚本控制KVM虚拟机的安装和管理，中间层XMLRPC通讯， 并且给出了选用各项技术的原因和优劣分析。本文主要对硬件资源管理，虚拟机管理，资源分配管理，环境自动部署这几个模块进行了详细设计和分析，对设计过程中需要解决的问题，实现逻辑，实现技术等进行详细阐述，为其他开发者解决类似问题提供了很好的借鉴作用。

本文论述了运维自动化平台系统的设计和实现过程。本系统投入使用以来，为系统管理员管理硬件和虚拟资源提供了很大帮助，大大缩短了测试环境的搭建时间，并为虚拟资源的使用者简化了申请流程，提高了工作效率。

关键词 KVM虚拟化, 自动化运维, 自动部署, 服务器管理

# Abstract

With the advent of the era of cloud computing, virtualization technology has become indispensable in the operation and maintenance work. The development of virtualization technology has brought about the more and more virtual server, virtual network and other resources, and on the other hand it was more and more difficult to manage the resource. Especially for companies with a lot of their own servers, how to standardize the management of more and more virtual resources, become an urgent problem to be solved. How to efficiently integrate the virtual resources in the application, the use of the personnel, and the use of the appropriate time to recycle resources, but also need to solve the problem of operation and maintenance automation platform. For a lot of companies, the stability of the management and operation platform based on virtualization technology, can make the development of testing process more efficient, can make the server to play a more excellent performance, can the greatest degree of reducing operation and maintenance costs.

To solve these problems, this paper presents a solution for the operation and maintenance automation platform based on KVM virtual machine technology. The program group system administrators to manage the existing hardware resources, and through the KVM virtualization technology to create a virtual resource based on the existing hardware resources, and quickly automatically install the deployment of the specified service, assigned to the end user.

The automatic operation and maintenance platform is designed and developed by the methods of multilayer. In this research, we choose the technology of the whole system, which is based on B/S mode, and the service layer uses Python script to control the installation and management of KVM virtual machine, and the intermediate layer XMLRPC communication. This research mainly focuses on the management with hardware resource and virtual machine, resource allocation and management, and the environment automatic deployment of these modules are detailed design and analysis of the design process to solve the problem, to achieve logic, technology, etc., to provide a good reference for other developers to solve similar problems.

This main research discusses the design and implementation of operation and maintenance platform. The operation and maintenance automation platform has been put into use, it has provided a great help to the system administrators to manage hardware and virtual resources, greatly reducing the time of testing environment, and simplifying the application process and improving the working efficiency.

Keywords KVM virtualization, automatic maintenance, automatic deployment, server management

# 绪论

## 1.1 课题研究背景

软件所运行的环境往往是很复杂的，比如开发软件的公司都会搭建开发环境，测试环境，仿真环境等一系列环境来模拟真实环境，测试他们开发的软件。而这些环境的搭建经常会耗费比较大的时间和精力，要考虑的因素很多，比如服务器硬件环境是否和真实环境一致，网络是否能仿真等问题。 另一方面，现在的软件，无论其规模的大小，部署起来步骤常常是比较繁琐，有很多参数需要设置，如果是分布式集群形式，还要考虑各个集群的配置的差异问题。在部署过程中，通常会遇到一些不可预料的问题，需要回退到上一个版本。

为了解决这些问题，自动化的运维管理平台应运而生。运维管理平台需要统筹管理公司的服务器等硬件资源，对硬件资源中机房架构，网络分布有良好的规划，对服务器的当前使用情况有详尽的统计，并及时反馈给系统管理员。运维管理系统需要结合持续集成的观点，提供自动化部署开发环境，测试环境和仿真环境的功能，开发环境和测试环境可以对真实环境做适当简化，仿真环境要最大程度的模拟真实环境。

系统管理员的工作往往是比较繁琐的，他们需要维护整个服务器环境正常运行的任务，不断的处理各种突发事件，以及服务器扩容，网络改造，各种应用服务宕机等问题。如果有自动化程度比较高的运维管理系统，就能大大提升系统管理员的工作效率，减少环境问题对其他部门的影响，从而提升公司的竞争力。

## 1.2 课题研究的意义

稳定可靠的运维管理平台是软件企业管理资源，提高生产效率的关键。目前在企业中运维工作普遍存在很多问题，例如自动化程度不够，集成运维的规划不够等。企业的IT运维的自动化做得不够好，不仅影响效率，还会使企业的运维人力成本高得惊人。据调查，现在很多企业仍然处于被动低效率的“救火式”的IT运维方式[8]。运维人员终日忙碌，只是处理一些技术含量不高的重复性运维工作，例如人工巡检服务器等，而运维系统的故障预警机制不够完善，出了问题后才临时处理[9]。有的企业做了运维自动化的尝试，但简单的自动化带来了更多的问题，系统管理员每日处理很多系统警告，其中很多是误报。运维自动化的另一个问题是对应用支持的不够，很多企业的上线系统的维护补丁是运维人员手工执行的，这种做法不仅导致很大的出错概率，而且效率非常低下。如果软件运行环境比较复杂，运维人员新搭建一套仿真测试系统也需要花费非常多的精力。

## 1.3 国内外研究现状

近年来运维平台出现了一些新技术，例如虚拟化管理平台的代表OpenStack，CloudStack等。OpenStack是一个由NASA（[美国国家航空航天局](http://baike.baidu.com/view/85211.htm)）和Rackspace[合作研发](http://baike.baidu.com/view/1506447.htm)并发起的，Apache许可证授权的云计算管理平台项目[4]。Openstack的目标是利用虚拟化技术，提供基础化设施的创建和维护，为企业内部提供共享资源。Openstack能解决包括对象存储，镜像服务，资源计算等问题，但是它的缺点是对实施人员的要求非常高，架构过于庞大，需要有专业人员的指导才能实际运用。另外，openstack对于应用部署的支持不够，不足以支持企业应用中复杂的业务场景。

CloudStack是一个开源的具有高可用性及扩展性的云计算平台。支持管理大部分主流的hypervisors，如KVM，XenServer，VMware，Oracle VM，Xen等[5]。CloudStack 是一个开源的云操作系统，它可以帮助用户利用自己的硬件提供类似于Amazon EC2那样的公共云服务。CloudStack可以通过组织和协调用户的虚拟化资源，构建一个和谐的环境。CloudStack具有许多强大的功能，可以让用户构建一个安全的多租户云计算环境。CloudStack 兼容Amazon API 接口。

国内对于基于虚拟化的运维管理平台的研究也在迅速发展。近年来也出现了云服务平台，为企业或个人提供私有云服务基础设施，代表性的有青云QingCloud公司。青云主要提供分布式存储、一体化网络等基础设施服务，也提供一些数据库，缓存，消息队列等Paas层的服务，按用户使用时间收费。 同类型的还有阿里云，提供弹性计算，数据存储于CDN等服务。

关于自动化运维方面还有fit2cloud运维平台，是由杭州飞致云信息科技有限公司成立于2014年，致力于为云基础设施和应用提供配置、 监控、自动化和管理服务，让企业能够轻松部署和管理跨公有云、私有云的重要商业应用。但是应用该平台，用户仍然需要编写自己的部署脚本，需要定制的任务很多。

这些云平台技术对于企业自动化运维平台是一个很好的参考，缺点是稳定性和安全性不能很好的保证，也不能解决运维中应有自动化部署等问题，不适合大型企业使用。

本文将探讨基于KVM的虚拟化技术的运维管理平台，有管理实体机服务器，申请分配虚拟资源，自动部署应用等功能。

本文所设计实现的运维管理平台基于KVM虚拟机技术，加入云计算思想，在以下几点上有创新：

1. 具备弹性和可扩展性。新加入的实体机服务器安装操作系统后，可以由本运维平台探测得到服务器信息，避免手动录入，提高效率，方便扩展。
2. 按需使用，定时回收。虚拟资源的使用人员申请资源后，本运维平台将按其使用的服务器的情况，到期时间确定是否收回。合理回收并注销应用环境，合理利用运维资源。
3. 自动分配虚拟资源，自动安装应用。虚拟资源使用人员申请后，本运维平台评估用户需求，按需提供必要虚拟资源，并根据应用版本等信息找到应用的基线自动部署并冒烟测试，提供给用户准备完善的开发测试环境。

## 1.4 本文主要工作

本论文的主要研究工作和成果如下：

1. 设计了基于kvm虚拟化技术的运维自动化平台的整体架构，将其划分为硬件资源管理、虚拟资源管理、资源分配流程、环境自动化部署这四个主要功能模块。并分别对每个功能模块进行了详细的需求分析、设计和实现；
2. 通过采用实现一套操作系统的自动化网络安装技术，并设计了服务器的主机探测流程，使新增服务器纳入本运维管理平台的过程完全自动化，简化了以前新增服务器需要系统管理员手工录入信息等繁琐流程；
3. 设计了运维资源的监控流程，实现了服务器出现故障的及时告警，以及对于达到监控指标阈值的预警；
4. 设计了基于kvm的虚拟机的管理流程，实现了虚拟资源的自动化定制创建、扩展等管理工作；
5. 通过设计一套完整的针对运维资源的申请、分配和回收的流程，使运维资源的管理非常高效，解决了运维资源空置、性能达不到使用需求，以及资源的使用者和管理者沟通不畅等问题；
6. 通过设计了应用环境自动化部署的流程，解决了不同应用程序在部署中需要对环境不同定制化的问题。只要在本系统中定义了模板，就可以自动化部署用户需要的不同应用。

本文最后对本运维平台的实际运行状况进行了详细分析。通过本运维平台的使用，可以实现运维的全自动化，使运维资源得到高效利用。同时，本文的系统设计和实现经验将为其它需要解决此类运维自动化的问题的项目起到一个很好的借鉴作用。

## 1.5 论文的结构安排

本文共分六个章节。

第一章为绪论，阐述了本论文课题的研究背景，以及研究的意义和国内外研究的现状，并且对本论文的整体结构有个总体的结构描述。

第二章为相关技术综述，首先介绍了本论文所使用到的一些与虚拟化相关技术，然后对本文研究中使用的自动化安装技术进行描述，以明确说明使用这些技术的意义和价值。

第三章主要描述的是本运维平台系统的需求分析。介绍了本系统的开发背景，以及需求要点，并对这些需求要点进行深入分析。

第四章是阐述了本系统的设计和实现。本章为本文研究重点，设计并实现了自动化运维平台。本章中首先设计了整体架构方案，然后对本系统的资源管理及监控方案、资源的申请分配流程，以及自动化部署等模块的设计开发原理及实现方案进行详细的阐述。

第五章为系统运行分析。首先阐述了系统的运行平台和部署方案，然后对本系统的运行效果进行了分析。

第六章全文总结。总结全文工作，分析研究工作不足之处，并对未来工作做出展望。

# 相关技术介绍

在本运维平台系统的设计和实现过程中，虚拟化技术是本系统得以实现的基础，自动化部署技术是本系统实现自动化的保证。在本章中，前两节将介绍kvm虚拟化，和虚拟网络的技术实现；后两节介绍在实现自动化部署中应用的相关技术。

## 2.1 KVM虚拟化技术

### 2.1.1 虚拟化技术

随着云计算技术的发展，虚拟化技术也越来越成熟了。虚拟化是一种对计算机资源的抽象，虚拟化技术，即在物理计算机资源（计算机硬件，网络，存储等）的基础上，创建出逻辑意义上的多个操作系统。

虚拟机化技术的出现，解决了运维管理中原有的一台服务器主机上只能创建单一操作系统的问题。在虚拟化技术发展之前，系统管理员往往需要在一台服务器上建多个用户，将用户分配给不同的使用者，来达到一机多用的目的，但是这种办法效果并不好，往往会出现端口冲突，出现问题无法定位负责人等种种问题。虚拟化技术的应用，可以让系统管理员在分配资源时将同一台服务器上的不同用户完全隔离开，从而避免了交叉影响的问题。

虚拟化技术包括准虚拟化和全虚拟化两种。准虚拟化在本质上是对虚拟机特殊指令的被动截获，将其转化为虚拟机所在主机操作系统能识别的指令。准虚拟化需要运行在经过修改的特定的操作系统下。准虚拟化的例子有 Xen 和 User-Mode-Linux (UML)。全虚拟化则可以理解为不需要对虚拟机所在的实体机做任何修改即可基于X86平台的操作系统和软件。全虚拟机化为虚拟机提供了完整的虚拟平台，包括处理器，内存和外设。这是全虚拟化对准虚拟化的的巨大优势。

### 2.1.2 KVM架构

KVM是一种全虚拟化技术。和Xen相比，KVM还是相对较新的虚拟化技术，但是KVM目前正被大量使用。KVM的优势在于，它刚诞生不久就被Linux社区接纳，成为随Linux内核发布的轻量型模块。

KVM的主要功能是初始化CPU等硬件，打开虚拟化模式，为运行在实体机上的虚拟机提供支持。首先，KVM会加载实体机服务器的Linux内核模块，初始化内部的数据模块，然后执行虚拟化模式，最后，KVM模块创建特殊的设备文件/dev/kvm，来和用户空间交互。初始化完成后，接下来虚拟机的创建将是用户空间的应用程序（QEMU）和KVM模块的交互配合的过程。KVM虚拟化组件的结构图如下图：

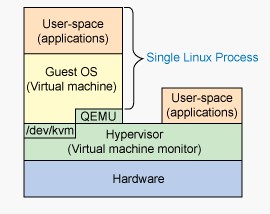


图2.1 KVM虚拟化组件

对上图中QEMU介绍，QEMU本身是一个著名的开源虚拟化软件，但是它不能实现硬件的虚拟化。KVM上文提到加载了KVM模块实现初始化后，还需要有一个用户空间的工具给用户来操作，因为用户无法直接通过内核模块来操作。这个用户空间工具，就是开源虚拟化软件QEMU。KVM使用了QEMU的一部分，并在此基础上做了修改，成为可控制KVM的用户空间工具。

### 2.1.3 KVM与Xen的比较

除KVM虚拟化技术以外，还有很多虚拟化技术，例如Xen等。KVM和这些虚拟化技术相比有哪些优点和缺点呢。

Xen也是Linux下的一个虚拟化解决方案。Xen的实现方法是运行实现Xen功能的kernel。在Linux系统中使用这个kernel，并启动服务器后，就可以在该服务器上运行qemu软件，从而虚拟出多个虚拟系统。Xen的缺点在于，每次需要更新Xen的版本时，都需要重新编译整个内核，这样容易产生很多问题，导致系统无法启动[6]。而KVM更加简化，它更新时不需要重新编译内核。并且它代码量更小，结构更精简。并且在某些方面，性能比Xen更胜一筹。

### 2.1.4 KVM安装和使用

安装KVM首先需要一个安装了Linux操作系统的宿主机。这一步在2.2 小节会详细讲解自动化网络安装的方法。

安装KVM的步骤如下：   
1、下载解压并配置kvm软件包：

下载KVM包后，打开终端并输入下列命令：

|  |
| --- |
| tar xzf kvm-release.tar.gz  cd kvm-release  ./configure --prefix=/usr/local/kvm  make  sudo make install  sudo /sbin/modprobe kvm-intel |

2、检查系统中是否包含下面四个虚拟机软件包，如没有需用yum方式安装：

1. kmod-kvm
2. kvm
3. kvm-qemu-img
4. bridge-utils

安装完成后须重启，否则报如下错：

open /dev/kvm: No such file or directory

Could not initialize KVM, will disable KVM support

3、具体安装步骤如下：

1. 创建网桥
2. 创建虚拟机硬盘文件

#qemu-img create /images/kvm01.CentOS54.img 30G

文件大小根据应用需要，示例中为30G，文件名（kvm01.CentOS54.img）、文件大小（30G）、安装位置(/images/)，实际情况根据需要自定。

1. 制作软盘镜像文件

软盘镜像文件(a.flp)中包含安装CentOS所需的kick-start文件。

1. 准备CentOS安装光盘镜像文件

例如采用CentOS-5.4-x86\_64-bin-DVD.iso

1. (可选)将输出重定向到监控机

#export DISPLAY=192.168.106.158:0.0

监控机需预先安装XWindows服务器程序.

1. 虚拟机MAC地址设置

网卡MAC地址前面三个字代表生产商。本系统中KVM虚拟机的这三位缺省设置为52:54:00，

虚拟机MAC地址的后续两位拟设置为16进制表示的IP网址后两位。虚拟机MAC地址的最后一位拟从01开始，依次分配给该虚拟机内的各虚拟网卡。

例如：虚拟机192.168.86.39两个虚拟网卡的MAC地址分别设为：

|  |
| --- |
| 52:54:00:56:27:01  52:54:00:56:27:02 |

为简化配置，接入存储私网的网卡MAC不再根据存储私网IP计算得到，而直接从上述MAC地址简单末位递增得到。

1. 安装虚拟机操作系统

安装Linux

1. 将软盘镜像文件(a.flp),CentOS光盘镜像文件(CentOS-5.4-x86\_64-bin-DVD.iso),引导程序initrd.img,内核文件vmlinuz都放到当前目录,执行安装命令启动安装过程:
2. 执行命令

|  |
| --- |
| #qemu-kvm –m 2048 –hda /images/kvm01.CentOS54.img –cdrom ~/ CentOS-5.4-x86\_64-bin-DVD.iso –net nic,vlan=0,model=e1000, macaddr = 52:54:00:56:27:01 -net tap, vlan=0, ifname=tap00, script=/etc/qemu-ifup –net nic, vlan=1, model=e1000, macaddr = 52:54:00:56:27:02 -net tap, vlan=1, ifname=tap10, script=/etc/qemu-ifup -kernel vmlinuz –drive file=a.flp,index=0,if=floppy –append ks=floppy:/ks-dell-trade.cfg –initrd initrd.img –daemonize |

安装Windows

可采用交互式安装，如执行下面命令安装Windows2003

|  |
| --- |
| #/usr/libexec/qemu-kvm -m 2048 -localtime -hda /images/kvm86195.win2003std.img -cdrom ./Win2003Std.iso -net nic,vlan=0,model=e1000,macaddr=52:54:00:56:C3:01 -net tap,vlan=0,ifname=tap05,script=/etc/qemu-ifup -net nic,vlan=1,model=e1000,macaddr=52:54:00:56:C3:02 -net tap,vlan=1,ifname=tap15,script=/etc/qemu-ifup --daemonize --boot d： |

## 2.2 Open vSwitch虚拟交换技术

Open vSwitch是一个虚拟交换软件，主要用于虚拟机环境，作为一个虚拟交换机来使用。它支持Xen、 KVM、 VirtualBox多种虚拟化技术。

一台机器的虚拟化的环境中，一个虚拟交换机主要有两个作用：传递虚拟机VM之间的流量，以及实现VM和外界网络的通信。

Open vSwitch主要是通过网桥实现的，网桥也叫桥接器，是连接两个局域网的一种存储/转发设备，它能将一个大的LAN分割为多个网段，或将两个以上的LAN互联为一个逻辑LAN，使LAN上的所有用户都可访问服务器。

当我们创建了一个网桥，会产生一个虚拟网卡，名字就是br name，然后，我还需要为这个交换机增加端口(port)，类似于物理网卡，当网卡加入到这个交换机之后，其工作方式就和普通交换机的一个端口的工作方式类似了。

下面介绍Open vSwitch的主要逐渐及其功能：

* ovs-vswitchd：是open vswitch的最重要的组件，它通过netlink协议直接和OVS的内核模块进行通信。ovs-vswitchd是守护程序，实现交换功能，和Linux内核兼容模块一起，实现基于流的交换flow-based switching，并且将交换机配置等信息实时保存起来，使得即使系统被重启后，交换机的相关配置仍然能存在[12]。
* ovsdb-server：是OVS中轻量级的数据库服务，主要保存了整个OVS的配置信息，包括接口数据信息，交换内容，VLAN信息等等。ovs-vswitchd会根据数据库中的配置信息工作。Ovsdb-server需要保持和ovs-vswitchd服务的联系，以便配置信息能及时的保存起来。
* ovs-dpctl：是OVS中的一个工具，用来配置交换机内核模块，可以控制转发规则，使得ovs-vswitchd服务不依赖数据库的情况下去管理内核空间的数据通路[12]。
* ovs-vsctl：是OVS用于交换机管理的基本工具。它的主要功能是是获取或者更改ovs-vswitchd的配置信息，用户可以登录到交换机部署的服务器上通过ovs-vsctl管理OVS交换机。
* ovs-appctl：是向OVS守护进程发送命令的工具，通过发送一些内部命令给ovs-vswitchd组件以改变其配置。
* ovsdbmonitor：GUI工具来显示ovsdb-server中数据信息。
* ovs-controller：一个简单的OpenFlow控制器。
* ovs-ofctl：用来控制OVS作为OpenFlow交换机工作时候的流表内容。

Open vswitch的使用方式示例如下：

1. 创建ovsdb数据库，具体指令如下：

ovsdb-tool create /usr/local/etc/openvswitch/conf.db vswitchd/vswitch.ovsschema

1. 启动ovsdb-server，具体指令如下：

ovsdb-server /usr/local/etc/openvswitch/conf.db --remote=punix:/usr/local/var/run/openvswitch/db.sock --remote=db:Open\_vSwitch,manager\_options --pidfile --detach --log-file

1. 启动vswitchd daemon，具体指令如下：

需要指定和ovsdb-server通信的机制，默认是unix:/usr/local/var/run/openswitch/db.sock

ovs-vsctl --no-wait init

ovs-vswitchd --pidfile --detach unix:/usr/local/var/run/openvswitch/db.sock --log-file –verbose

## 2.3 操作系统的自动化网络安装技术

在本运维自动化平台的环境自动部署过程中，最开始的步骤为安装操作系统。在测试管理系统中，主要通过自动化的网络安装方式来执行这一步骤。

操作系统的自动化网络安装需要结合以下几种工具：

PXE：Pre-boot Execute Environment(预启动执行环境)，是intel公司设计的一组硬件规范，其目的是实现网卡引导的系统启动。PXE客户机首先通过DHCP服务获取网卡配置和pxelinux启动程序位置(在tftp服务器上)，然后从tftp服务器获取pxelinux启动程序(类似grub的启动器)并加载运行，最后加载用户选择的linux内核镜像并引导系统启动。

DHCP：Dynamic Host Configuration Protocol(动态主机设置协议)，用于获取主机的网络配置。

TFTP：Trivial File Transfer Protocol(简单文件传输协议)，类似FTP的用于文件传输的网络协议。

Anaconda：Redhat和Fedora Linux系统的安装管理程序。

Kickstart：Anaconda使用的自动化安装脚本，它能帮助完成Redhat系统的无人值守安装。

一台服务器的自动化系统安装的完整流程图如下：

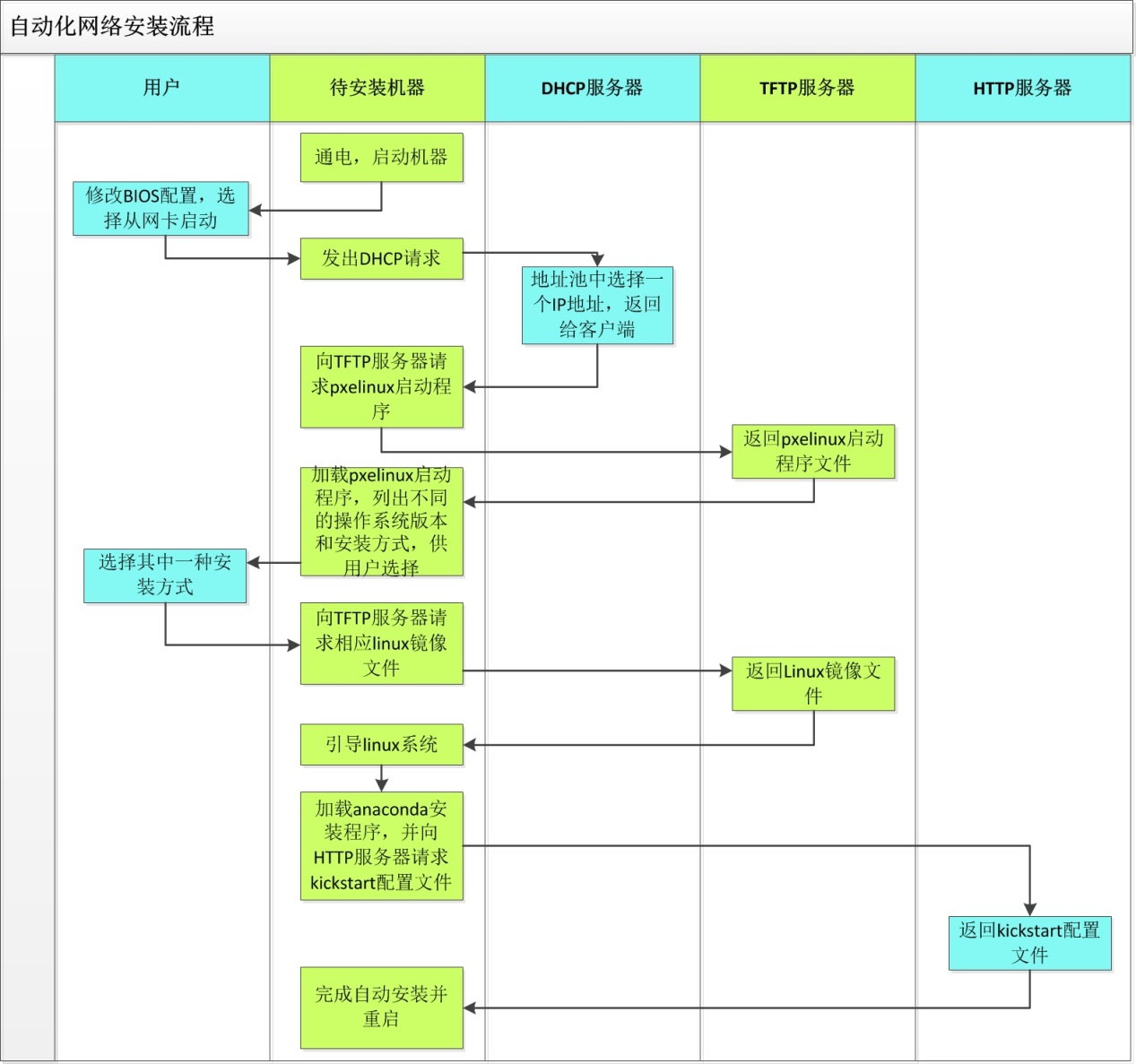


图2.2 自动化系统安装流程

Cobbler是一套用于快速搭建网络安装环境的软件，它可以帮助你配置DHCP服务、TFTP服务和YUM源，并且可以灵活定义不同的系统安装方式。该软件使用python开发，小巧轻便，同时支持命令行和web方式进行管理，并提供API接口，可以方便地进行二次开发。

Cobbler定义了三层概念，用于定义个性化的系统安装方式：

Distro：定义了安装源和linux镜像文件，通常用来确定某个特定的操作系统版本

Profile：继承自某个Distro，在此基础上又定义了kickstart配置文件

System：继承自某个Profile，在此基础上再细化了系统信息，如hostname、ip等

Cobbler软件包还提供一个重装工具koan，使用它可以很方便地对系统进行远程重装。koan会在重启机器前，修改系统grub配置，使其在下次重新引导系统时，采用类似网络安装的方式启动系统，从而完成系统的重装。

## 2.4 基于基线的应用自动化部署技术

在搭建应用测试环境时，我们希望最后发布的结果能够尽可能地和生产环境保持一致，这样在测试环境下得到的测试结果才更有意义。但是现实中我们无法通过简单地通过拷贝粘贴文件来复制一个和生产环境一致的环境，而完全依靠手动的部署一方面耗费时间，另一方面其结果又有较大的不确定性，且无法精确还原历史版本，故在本系统中采用基于基线的环境发布技术来解决上述问题。

基线是对环境发布过程中具体操作过程的总结和抽象，它根据某一版本的生产环境实际情况制定，是测试环境发布的基准。基线由配置工具、配置模板和版本信息三部分组成。

（1） 配置工具主要用于在发布过程中结合配置模板和用户提供的配置参数修改配置项。每个配置项都有其对应的配置工具。配置工具的形式可以是脚本或者是可执行程序。

（2） 配置模板是为规范配置项的配置而提供的一组文件，可以简化用户在环境安装部署中的配置过程，并尽量减少在这一过程中发生的错误。某些配置模板直接使用生产环境的配置项收集结果（比如系统参数配置/etc/sysctl.conf），某些配置模板需要在生产环境收集结果的基础上进行加工处理（比如网卡配置/etc/sysconfig/network-scripts/ifcfg-eth0，由于其中配置了IP地址，所以无法直接使用生产环境的配置）。

（3）版本信息包括操作系统的版本号，数据库版本号，应用系统版本号以及应用系统版本与操作系统版本的关系。

由于基线反映了某一版本生产环境的配置情况，故当生产环境发生变更时，也必须建立新的基线来适应这些变更。不过，只要这些变更只涉及到与环境无关的配置项时（即这些配置项在所有环境都是相同的，比如/etc/sysconfig/iptables），可以使用生产环境的收集结果来自动地建立新的基线。总体来看，需要手动方式新建基线的情况还是比较少的。

基于基线的测试环境发布主要具有如下几个特点：

（1）确定性，即在基线相同且用户配置相同的情况下进行多次发布，发布的结果应当都是相同的。

（2）可预见性，即当用户选定了基线，并且提供了相应的配置参数后，发布结束后各配置项的配置情况应当是符合预期的；

（3）全面性，即在发布过程中，不需要对基线规定范围之外的配置项进行配置。

# 需求分析与设计

## 3.1 项目背景

软件公司每年批次采购服务器，管理自有机房中的服务器的成本越来越高；另一方面，随着近年来上线的业务系统的不断增多，新版本的上线的频率逐渐增加，对运维资源的需求量也在越来越大，所要求的支持效率也越来越高。

本课题来源于上海期货信息技术公司为管理自有运维资源，实现DevOps，提升运维效率而设计开发的运维管理平台。公司的运维面临三大挑战，首先，公司软件所在的生产环境网络复杂，准备一套仿真环境用来测试及其复杂；其次，维护补丁版本的迭代时间压缩的越来越少，对于开发测试运维的各环节分配的时间也越来越少；最后，确保其中一个系统的修改，是否会对整个系统的流程产生影响，需要有一套快速评估的系统。

对于以上提到的问题，首先运维管理平台需要基于虚拟化技术，因为期货交易系统需要五个不同的网段，而传统方法无法解决网段的划分，尤其是多套测试系统同时工作的时候，更需要部署在不同的网段中，对交换机的数量需求太大，所以，用虚拟机配合虚拟网络，才能解决网络的问题。其次，更新版本的频繁，需要运维管理平台具备快速自动部署系统，自动冒烟测试的功能，最大化的降低集成部署在整个开发测试环境中所消耗的时间。最后，需要管理运维平台具有可选的部署系统的功能，能根据测试系统的类型灵活的选择不同相关的上下游流程系统来进行部署。

本运维平台的建设目标是利用KVM等云计算技术，搭建出一套能快速满足用户对于测试需求，性能稳定的运维管理平台。它能对现有的服务器资源，网络资源的梳理，以便让系统管理员清晰的了解运维资源的使用状况，能方便的扩展和替换运维资源。另外它也具备快速整合运维资源，自动搭建出符合开发人员和测试人员需要的集成测试环境的功能。

## 3.2 系统详细需求

### 3.2.1 硬件资源管理模块的需求

本模块首先需要对机房中的服务器的主机信息进行详细的统计。服务器主机资源信息包括以下内容：

* 主机名称
* 主机硬件信息：包括主机序列号、主机型号、CPU个数、内存大小
* 主机的管理网段IP地址
* 主机的机架位置信息：包括机房名称、机架号和机器位置

初始数据构建完成后，对于新增的服务器主机资源，在本系统中可以采用如下步骤来让新上架的服务器纳入本运维管理平台的管理范围：

1. 导入新增的主机序列号列表；
2. 针对列表中的主机，进行操作系统的网络安装；
3. 主机探测，在本系统中更新新增主机信息。

### 3.2.2 虚拟资源管理模块的需求

运维资源的管理和监控。当有虚拟机的创建需求时，需要运维平台快速判断出当前的硬件资源的负载程度是否能满足需求，如果不能满足需求要考虑扩展资源。这就需要对硬件资源严密监控，实时了解当前的硬件资源使用状况，以及对使用年限比较长的服务器提前更换的预警。

本系统中对于运维资源的分配流程如下所示：

1. 用户填写对于运维资源的需求，提交管理员审核；
2. 管理员审核并分配环境；
3. 环境的维护，包括虚拟网络的修改、环境使用人员的权限分配等；

### 3.2.3 资源分配管理模块的需求

本运维平台系统的用户有系统管理员，以及需要使用系统资源的开发人员、测试人员、运维人员等使用者。

为了满足资源使用者的需求，同时将虚拟机、网络等资源得到有效利用。本系统设计流程为使用者用户提出申请，明确需要何种类型的虚拟机和网络，以及部署何种应用程序，同时确定使用期限，到期归还。系统管理员接到用户的申请，选择确认或驳回。如管理员确认满足该用户需求，就为该用户建立一套用户单元环境，用运维管理系统自动创建虚拟机，并部署应用程序。

具体操作流程如下：

1. 测试人员在本系统中提出环境申请，明确需要的应用类型和版本；
2. 系统管理员收到申请，决定是否批准申请；
3. 如果批准申请，转入环境搭建流程；
4. 系统计算出该申请所需要的虚拟机和网络资源，自动创建虚拟机并配置网络；
5. 系统按需求部署用户申请的应用版本；
6. 系统通知测试人员环境已准备好可以使用。

### 3.2.4 环境部署模块的需求

快速部署应用程序是指本运维管理平台基于部署内容，自动完成测试环境中各主机的操作系统安装和配置，以及各应用系统的发布部署的过程。

按需求分析，环境部署模块需要实现的主要功能如下：

1. 待部署应用的主机需要完成准备；
2. 主机准备完成后，系统根据环境模板选择确定需要部署的应用；
3. 系统从资源库中获取指定基线的应用；
4. 系统执行部署脚本，完成部署；
5. 运行冒烟检查脚本，检查部署的应用是否可用。

其中主机准备环节的流程图如下：



图 3.1 主机准备流程

主机定制配置需要考虑如下因素：

1. BIOS配置
2. 主机名称
3. 网卡IP配置
4. 路由配置
5. DNS配置
6. SYSCTL系统参数配置
7. NTP配置
8. IPTABLES配置
9. 用户和组
10. SSH信任域配置
11. YUM源配置
12. RPM包
13. 开机自启动服务配置
14. USB KEY配置
15. 存储设备配置

## 3.3 用户分类

本系统分为系统管理员、审批管理员、环境申请人员两类用户。每种用户在系统中拥有不同的操作权限，从而起到了权限控制作用。

1．系统管理员：系统管理员是本系统主要服务的用户，拥有创建硬件资源，创建KVM虚拟机，管理KVM虚拟机，创建测试环境，部署应用等主要功能；

2．审批管理员：该类用户的主要功能为审批搭建环境的需求，判断环境申请人员的申请要求是否合理。此角色一般由运维部领导担任，也可以由系统管理员兼任；

3．环境申请人员：该类用户主要为公司的开发人员、测试人员、集成人员。当他们需要使用服务器资源时，在本系统提出申请建立一个单元环境，由审批管理员和系统管理员审核后搭建环境，部署指定的应用程序。

## 3.4 系统用例设计

### 3.4.1 角色识别

经过分析可以得到，本系统对应的主要角色有：

1. 系统管理员
2. 审批管理员
3. 环境申请人员

### 3.4.2 用例识别

在角色识别的基础上，经过分析可以得到本运维管理平台涉及到的主要用例如下图所示：

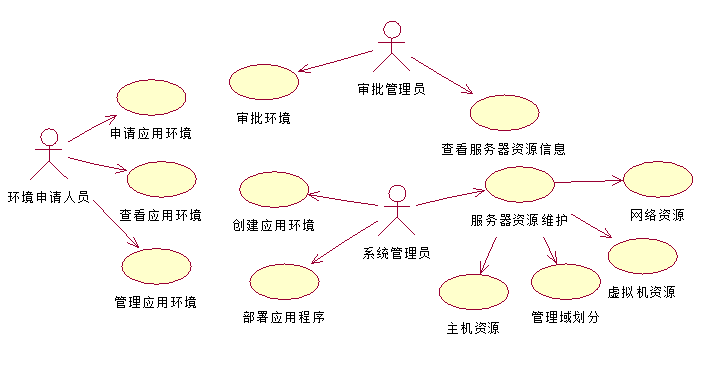


图3.2 系统用例图

### 3.4.3 用例描述

根据上述分析所得到的基本模型，我们对其中的主要用用例进行具体描述。

* 用例 1：服务器资源维护

1. 服务器资源维护共分为主机资源管理、网络资源管理、虚拟机资源管理这四个方面；
2. 主机资源管理为维护服务器主机的信息。服务器主机需要管理运维平台自动扫描得出，不需要管理员手动录入；
3. 系统管理员登陆系统后，进入服务器硬件资源管理模块，点击“扫描硬件资源”，开始对公司的内网网络内的IP地址逐一扫描，扫描完成后列出所有的服务器主机，包括主机CPU类型和核数、内存信息、硬盘信息、主机名、IP地址、网卡信息、hba卡信息等；
4. 列出主机信息后，管理员需要针对服务器主机录入其在机房机柜上位置，系统管理员进入硬件资源管理模块，点击“主机上架”，录入服务器主机所在机柜号，服务器高度，服务器所在机柜的层数；
5. 主机查询，系统管理员在所有主机的列表页面上，可以通过输入IP地址来查询出指定的服务器主机。查询IP需要联想功能，根据用户输入主动补全，列出IP地址供用户选择；
6. 网络资源管理为管理运维环境中服务器实体机的网卡，虚拟机的虚拟网卡。

系统管理员登陆系统后，进入“资源管理”->“网络资源管理”菜单，显示现有的网络资源，有“网络地址”、“掩码方式”、“网络类型”等属性。

此时用户可操作的功能包括：

1. 点击“新增”按钮，录入新增网络的基本信息，包括网络名称、网络的用途等信息，点击“确定”按钮即可生效。
2. 点击“修改”按钮，显示已有网络信息，修改完成后，点击“确定”按钮即可保存。
3. 虚拟机资源管理

因为虚拟机的管理离不开其所在的实体机服务器。所以对虚拟机的管理需要首先确认它在哪台实体机上。

系统管理员登陆系统后，进入“资源管理”->“主机资源管理”菜单，显示所有的实体机服务器，在此界面上展示实体机基本信息及在该实体机上部署的虚拟机。用户可以按虚拟机的IP地址搜索，搜索结果将展示该虚拟机及其所在的实体机。

此时用户可操作的功能包括：

1. 点击“搜索”按钮，录入虚拟机IP，得到虚拟机和实体机的信息，包括虚拟机的硬件基本信息，以及该虚拟机做何种用途，即在该虚拟机上部署了哪些应用程序；
2. 点击“新增”按钮，录入新增KVM虚拟机的基本信息，包括该虚拟机的IP、主机名、CPU核数、硬盘信息、内存信息等，点击“确定”按钮即可生效。
3. 点击“修改”按钮，显示该KVM虚拟机的配置，包括修改CPU核数、硬盘信息、内存信息等，其中硬盘大小只能在原来基础上增加不能减少（即硬盘扩容），修改完成后，点击“确定”按钮即可保存。
4. 点击“重启”按钮，将该虚拟机重启；
5. 点击“监控”按钮，列出该虚拟机近期的CPU、内存、硬盘使用率情况的波动曲线图，方便管理员针对了解该虚拟机上应用程序对资源的消耗情况；

* 用例 2：创建应用环境

系统管理员登陆系统后，进入“环境管理”->“新建应用环境”菜单，可以直接穿件一个新的应用环境。新建应用环境的属性有“环境名称”、“环境负责人”、“环境类型”、“到期时间”、“主管理网段”、“环境描述”等属性。

此时用户可操作的功能包括：

1. 在新增应用环境的界面上，录入该应用环境的基本信息，选择环境负责人、到期时间等，点击“确定”按钮即可生效；
2. 环境创建完成后可以点击“修改”按钮，显示已创建的环境的基本信息，修改完成后，点击“确定”按钮即可保存；
3. 环境的主机管理。环境创建完成后点击“主机管理”按钮，选择为该环境添加已有的主机（包括实体机服务器或者虚拟机），还是直接新建虚拟机。如选择添加已有的主机，则可以查看该环境的所有空闲的主机情况（即没有被其它环境使用的实体机和虚拟机），选择哪台实体机或者该实体机上的虚拟机添加进来，也可以直接搜索实体机或虚拟机的IP地址添加；如选择为该环境新建虚拟机，可以让本运维平台自动选择实体机服务器安装KVM虚拟机，或者系统管理员手动指定服务器安装。添加完成后，点击“确定”按钮即可保存；
4. 环境的网络管理。环境创建完成后点击“网络管理”按钮，选择网段或者虚拟网段添加倒本环境中。网络管理界面列出“网络编号”、“网络地址”、“掩码位数”、“网络类型”（实际网段还是虚拟网段）等信息，系统管理员选定后，点击“确定”按钮即可保存。如果开发或测试环境对网络没有要求，即只需要单一网段即可的情况下，本步骤可以跳过；
5. 环境的用户管理。环境创建完成后点击“用户管理”按钮，可以将本系统中的用户添加到本环境中来，使其拥有查看和管理该环境的权限。该操作可以由系统管理员或者系统管理员指定的本应用环境的管理者来执行，添加或删除用户完成后，点击“确定”按钮即可保存；

* 用例 3：部署应用程序

系统管理员登陆系统后，进入“环境管理”->“环境列表”菜单，选择应用环境后，就可以点击“部署应用程序”菜单，进入部署界面。

此时用户可操作的功能包括：

1. 为该应用环境选择部署何种应用程序，本运维管理平台将公司的应用程序归类，用户可以选择需要的应用程序;
2. 选择应用程序版本，本运维管理平台将列出应用程序的历史版本，用户可以根据需要选择部署最新的或者版本库中的某一版本；
3. 用户选择完成以上两项后，点击“确定”按钮即可生效，本系统将开始自动将用户选中的应用程序版本部署到该应用环境，以进度条显示部署进度；

* 用例 4：申请应用环境

环境申请人员（即开发人员、测试人员、集成人员）登陆系统后，进入“环境申请”->“我的申请”菜单，系统将列出该用户所有的申请记录。

此时用户可操作的功能包括：

1. 点击“申请新环境”按钮，进入应用环境申请页面，录入环境用途、预计归还时间、预计的主机类型、性能和数量，点击“确认”即可提交申请；
2. 点击“续借环境”按钮，选择为该用户使用的到期环境的环境名称，录入“预计归还时间”、“续借原因”，提交审批管理员审核；

* 用例 5：查看应用环境

在上一用例中环境申请人员申请如审批通过后，即可查看应用环境。

环境申请人员登陆系统后，进入“环境列表”->“我的环境”菜单，系统将列出该用户拥有使用权限的所有环境。

此时用户可以查看的内容包括：

1. 应用环境的基本信息，包括“环境名称”、“环境负责人”、“环境类型”、“到期时间”、“主管理网段”、“环境描述”等属性；
2. 应用环境的上部署的应用程序，包括应用程序“名称”、“类型”、“版本号”、应用程序部署所在的主机信息等；

* 用例 6：管理应用环境

环境申请人员登陆系统后，进入“环境列表”->“我的环境”菜单，选择其中一个应用环境，进入管理界面。

此时用户可操作的功能包括：

1. 管理该应用环境的使用用户。如果是该应用环境的负责人（即由本人申请的环境），可以为该应用环境增加或删除使用人员。增加的使用人员为本运维平台的使用者，输入用户名时系统会自动识别补全，选择需要的用户，点击“确认”按钮确认添加；
2. 重新部署应用程序；应用环境的负责人可以重新部署应用程序的版本，但不能更改应用程序的类型，因为创建应用环境时用途已经确定。环境负责人选择应用程序版本后，点击“重新部署”，即可自动重新按指定版本部署，以进度条显示部署进度；
3. 管理本应用环境的虚拟机。虚拟机如果发生内存泄露等问题，会发生无法连接的问题，点击“重启”按钮即可重启虚拟机；另外如果虚拟机的性能无法满足部署的应用程序的要求，就会频繁出问题，这时候最好的办法是向系统管理员申请扩展虚拟机的性能，点击“虚拟机扩容”，输入扩展的要求，即可向系统管理员发送请求。系统管理员确认需求后，即可完成虚拟机扩展性能的操作；

* 用例 7：审批应用环境

审批管理员登陆系统后，进入“申请审批”->“我的任务”菜单，系统列出所有待审批的应用环境使用需求，包括入环境用途、预计归还时间、预计的主机类型、性能和数量等信息。

此时用户可操作的功能包括：

1. 批准环境申请人员的申请需求。点击“确认”按钮确认审批通过，该环境需求将被由系统管理员执行创建任务；
2. 驳回环境申请人员的申请需求。录入驳回理由，点击“驳回”按钮驳回申请，该环境申请将驳回给申请人员，由申请人员确认修改申请需求后重新申请，还是取消申请；

* 用例 8：查看服务器资源信息

审批管理员要确认是否通过应用环境申请，需要对运维平台的硬件资源有个总体的了解。

审批管理员登陆系统后，进入“资源管理”->“资源总览”菜单，系统将以报表形式列出资源总体信息，具体有如下几张报表：

1. 服务器机型分类表。列出服务器实体机的机器类型分类，服务器机型如DELL-R710、DELL-R720、DELL-R805、HP-360G5，统计每种机型的服务器数量及性能情况；
2. 服务器负载情况表。列出各服务器的负载及服务器的总负载比例。

# 运维自动化平台的设计和实现

根据前述章节介绍的虚拟化技术及自动化部署技术，并在以上需求分析的基础上，本章节将对本运维平台系统进行详细设计和实现，设计了系统架构的分层结构及并分析了各层的技术实现方案。

按功能划分本系统为四个主要模块，分别为硬件资源管理模块、虚拟资源管理模块、资源分配管理模块和环境自动化部署模块，对各主要功能模块需要解决的主要问题进行详细的描述和分析，并设计了解决方案。

## 4.1 系统结构总体设计

根据前述章节的需求分析，本运维管理平台系统需要在各个服务器上实现虚拟化并监控服务器运行状态，所以需要在各个服务器上部署客户端，同时需要服务端来接收监控信息，并发出指令；另一方面，本系统的使用者需要有良好用户体验的用户界面来操作，并与控制端实现接口交互。

本系统的系统架构，设计为三部分。第一部分是管理平台，即管理平台web应用，主要功能为系统管理员提供运维资源的管理操作功能，以及为用户申请资源提供平台；第二部分为控制器，通过RPC接口接收前台发出的管理指令，控制整个运维资源；第三部分为受控机群上配置控制程序，提供监控信息和执行控制器发出的指令。

本运维管理平台系统的总体架构设计如下图所示：

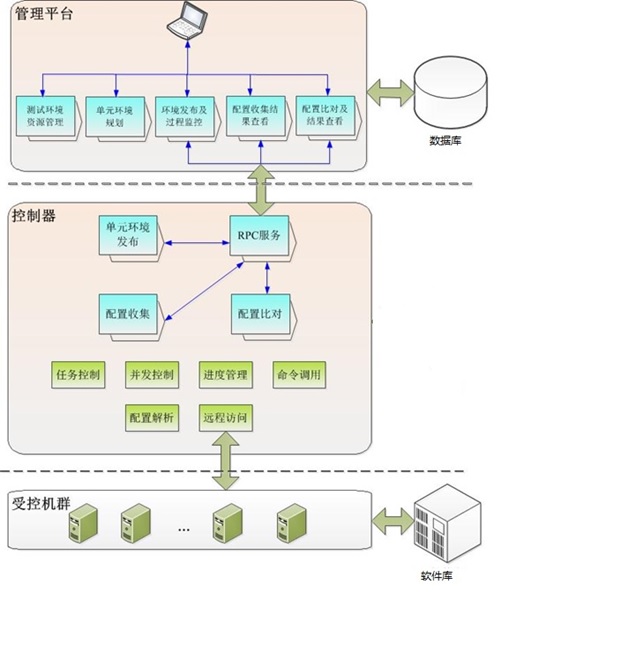


图4.1 系统体系结构

从图中可以看出，本系统的结构设计为由管理平台、远程受控端程序、数据库、软件库等四个组件组成。

数据库中主要存储环境信息、环境发布情况以及一些基础数据。数据库仅与管理平台发生交互。

软件库主要用于存放操作系统镜像、数据库软件、应用程序等，并同时承担网络操作系统安装服务器的角色。在测试环境发布时，软件库将通过SFTP方式向各受控机提供操作系统安装和相关软件下载的服务。

## 4.2系统各层实现技术方案设计

下面阐述本运维平台系统对各层的技术方案的设计。

### 4.2.1 前端运维管理平台WEB系统

运维管理平台向用户提供了操作界面，需要提供测试环境资源管理、配置收集结果查看、比对结果查看和单元环境发布及过程监控等功能的操作入口。本系统设计为使用基于[Spring MVC](http://docs.spring.io/spring/docs/3.0.x/reference/validation.html#format-configuring-FormatterRegistry)框架实现的B/S架构。

本系统按设计分为表示层、业务逻辑层、持久化层和数据库层。各层的说明如下所示：

1. 表现层

表现层是向最直接用户展示的窗口，为了提供良好的用户体验，提升开发效率，设计采用HTML4.0 + bootstrap + CSS + JQUERY架构技术实现。Bootstrap来自Twitter，是目前比较受欢迎的前端框架之一。Bootstrap使用方便灵活，使前端开发方便快捷，并提供丰富的简洁风格的WEB组件，是界面摆脱传统的html组件形式，提升了美观度。

在本系统中，为了直观的显示机房中服务器在机架上的位置，我们也自定义了一些jquery插件。

1. 业务逻辑层

业务逻辑层设计采用 Spring MVC 技术实现。spring mvc是spring中的一个模块。spring对于spring mvc的控制器管理更加简单方便，而且从Spring 3.0开始提供了全注解方式进行管理，各种功能的注解都比较全面，使用简单。所以Spring MVC在应用中受欢迎程度已经大大超过同类型的框架，例如Struts2，struts2需要采用XML很多的配置参数来管理。

1. 持久化层

持久化层设计采用 Hibernate技术实现。Hibernate对数据库结构提供了较为完整的封装，Hibernate的O/R Mapping实现了POJO和数据库表之间的映射，以及SQL的自动生成和执行。相比同类型的iBATIS，Hibernate提供的hql，使用更方便，数据库无关性好。

1. 数据库层

数据库层使用 MYSQL 数据库作为数据库选型。本系统对于数据库并没有严格的实时性等要求，所以使用相对比较轻量级的Mysql数据库。

在开发本系统的过程中，注重了项目管理和单元测试，以保证项目的质量。具体说明如下：

1. 项目管理

采用Maven + Svn进行项目管理实现，使用Jenkins实现持续集成。在敏捷开发中不可避免的要用到持续集成。持续集成的最大好处在于降低风险，简单地来说就是尽早暴露问题，能让开发人员及早发现并修复，从而降低修复成本。

开发本系统过程中，我们采用了Jenkins + Maven +Svn 的持续集成方案。每天开发人员提交在本地测试过的源码，jenkins自动检测到源码更新，

1. 单元测试

采用JUNIT4进行单元测试。在敏捷开发中，单元测试同样是非常重要的一个环节。JUNIT4是一个回归测试框架，它提供单元测试（Unit Test）的支持框架，用来编写和执行重覆性的测试。它能方便的组织测试组，进行系统主流程的快速测试，适合在每次迭代提交代码时调用。

### 4.2.2 后端控制端程序

控制端程序分为控制器和远程受控端程序，其中远程受控端程序需要部署在运维管理平台管辖的每一台实体机服务器上。这部分程序设计用Python语言进行实现。以RPC服务的方式对外提供API，供运维管理平台调用。

控制端程序以服务的方式运行。它的主要业务逻辑是处理虚拟机的相关任务。例如创建和删除虚拟机，修改虚拟机网络配置，虚拟机扩容，虚拟机启停等。

### 4.2.3 前后端通讯协议

因为本系统前后端分别用java和Python语言实现，所以在考虑前后台通讯技术方案的时候，我选择了RPC[远程过程调用](http://baike.baidu.com/view/431455.htm)协议的方案。相比与Rest, 本系统中更多的是WEB端直接发布指令到后端执行，所以RPC的方式更加符合需求。RPC通讯方式中具体采用XML-RPC协议，XML-RPC将RPC信息封包编码为XML，然后通过 HTTP传输封包。

在后端服务器实体机上需要运行监听服务，前台WEB端调用该服务发布指令。后端python程序引用SimpleXMLRPCServer库建立RPCServer，前台引用Apache XML-RPC访问对应的RPC URL。

### 4.2.4 公用资源包管理

在创建虚拟机和部署应用程序的时候，需要用到一些公用资源，这些公用资源包括各个操作系统的标准虚拟机镜像、需要部署的应用程序源文件、数据库初始数据等等。如果采用从控制端推送到远程服务器的方式，在远程服务器数量比较大，并且同步部署的情况下，系统性能将受到很大影响。

本系统在设计中将这些资源保存在公用资源库，有数台服务器组成，以SFTP的形式对外提供下载服务。这样设计的好处是在执行部署应用程序等场景，就可以直接下载需要的资源包到虚拟机上，避免了性能问题。

## 4.3 硬件资源管理模块的设计和实现

由需求分析可知，硬件资源管理模块需要实现的功能为管理并监控服务器、交换机等资源状态，使系统管理员能够全面的、动态的掌握硬件资源的情况。

在以往的运维管理平台中，需要使用人员录入或者运维人员在配置文件中配置大量的服务器信息，从而使管理平台能够根据这些信息管理硬件资源。而本系统的特色是设计并实现了网络安装操作系统以及主机探测的流程，规避了这些录入的步骤，简化了流程。实现了硬件资源管理的全自动化。

### 4.3.1 网络安装及主机探测的流程设计

本流程的需要实现的功能为服务器远程网络安装指定操作系统，并将服务器纳入本系统管理。主流程设计为主动探测机房中的服务器，并与系统中已记录的服务器信息比对；比对结果中发现有服务器与本系统中主机信息不符，邮件发送系统管理员确认修正；对于新上机架的服务器裸机，网络远程安装系统管理员指定的操作系统。

其具体流程设计如下图所示：

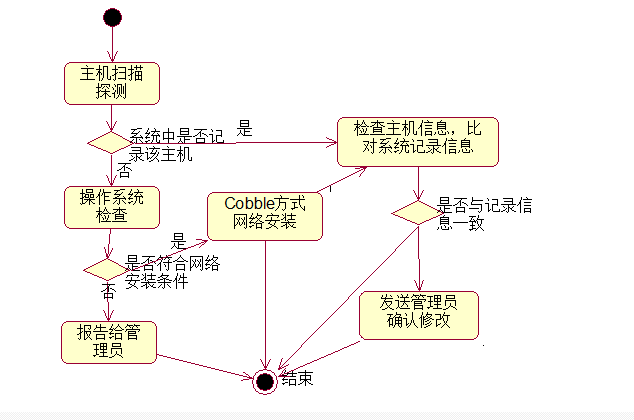


图4.2网络安装及主机探测流程图

流程具体设计如下：

1. 用户录入机房名称和机架信息，机架信息包括所属机房、机架号、机架高度
2. 对于已经安装好操作系统的主机，本系统可以使用探测的方式，自动发现这些主机，并挖掘出它的所有信息(除机架位置信息外)。自动探测的前提是，用户配置好管理网段IP地址，并提供对应于控制器的SSH信任关系。
3. 对于未安装操作系统的主机，需要用户提供这些主机的序列号列表，列表中的信息包括机器序列号、机架位置信息和IP地址。
4. 针对上一步骤的主机，使用网络安装方式，安装系统管理员指定的操作系统。操作系统安装好之后，我们将自动检测出该主机的其他基本信息。操作系统的自动化网络安装技术具体请见第二章相关技术介绍。
5. 至此，系统已经掌握了除机架位置信息外的所有其他主机信息。对于机架位置信息缺失的主机，需要用户录入其机架位置信息。

### 4.3.2 硬件资源监控及报警流程设计

硬件资源监控及报警流程需要实现的主要功能为监控服务器的状态是否正常，当遇到硬件故障等问题或达到设置的预警阈值时及时报警给系统管理员。

硬件资源监控的实现方案设计为定时扫描探测机房内的所有服务器的状态，并及时预警故障。在扫描探测过程中是以ping的方式分网段探测的，发现响应的服务器后与本系统信息进行核对，如果发现无法连接服务器，超时后报警。确认可以连接服务器后，连接服务器检查监控指标，如果达到或超过监控阈值报警。

本系统指定监控服务器的指标示例如下：

1. 主机是否可以连通；
2. CPU的使用率。如果该值持续超过90%，表明瓶颈是CPU。可以考虑增加一个处理器或换一个更快的处理器。如果服务器专用于数据库,可适当放宽预警值，合理使用的范围在30%至70%。
3. 磁盘的使用率。防止硬盘过满；
4. 磁盘I/O。如果该参数值一直很高，表明I/O有问题。可考虑更换更快的硬盘系统;
5. 内存的使用率；监控自由内存和交换内存的大小；
6. 日志缓冲区。如果日志缓冲区申请的值较大，则应加大LOG\_BUFFER参数的值。；

硬件资源监控的流程的设计如下：

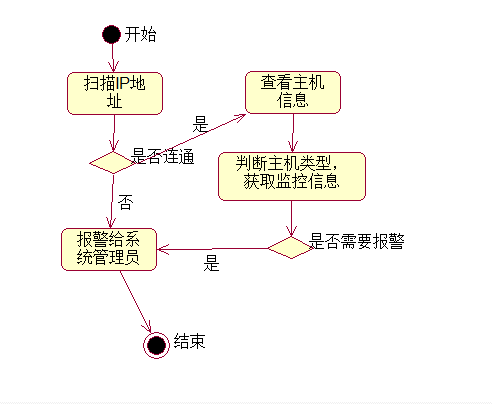


图4.3监控及报警子模块流程图

接下来对流程做详细阐述如下：

1. 扫描系统信息中IP地址；
2. 发现服务器响应后，检查是实体机服务器还是虚拟机，并在本系统中查找是否有该服务器信息；
3. 如果是虚拟机，检查操作系统、CPU、内存、运行状态是否正常，如达到预警阈值立即报警给系统管理员；
4. 如果是实体机，除了上一步检查的内容外，还需要磁盘挂载等信息，并对达到预警阈值立即报警给系统管理员；
5. 在本系统中记录报警信息日志，如系统管理员处理后，可以讲报警信息重置状态；
6. 如果系统中记录的服务器没有探测成功，作为服务器异常报警给系统管理员。

系统监控模块将服务器探测功能作为定时任务，配置在系统中。本系统采用spring框架自带的定时任务工具spring-task，在配置文件中配置本功能需要的接口，频率默认为1分钟/次。系统中关于采样频率的配置如下所示：

|  |
| --- |
| <task:scheduled-tasks scheduler=*"springScheduler"*>  <!-- timer(60 second)-->  <task:scheduled ref="monitorService" method="checkServerReachable" fixed-rate="60000"/>  </task:scheduled-tasks> |

### 4.3.3 硬件资源管理类设计和实现

本系统中对JAVA类的设计遵守开放封闭原则等JAVA类设计原则。例如对服务器信息设计为Server类，实体机和kvm虚拟机分别设计为其子类PCServer类和KvmServer类。这样设计的好处是当虚拟机或者实体机需要修改成员变量时，父类可以保持不变，只要在子类中修改。主要JAVA类图如图4.4所示：

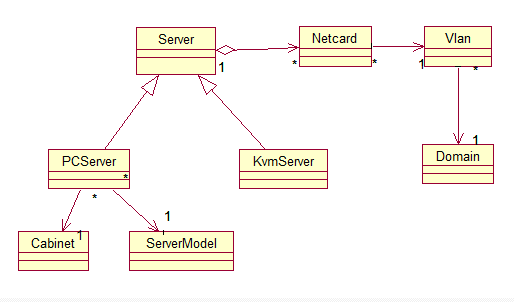


图4.4硬件资源管理类图

其中，Server类是服务器信息是硬件资源管理模块的核心，Server类定义了服务器的硬件属性等基本信息。主要成员变量设计有serverType，作用为定义主机属性，是实体机服务器或是kvm虚拟机；netcard，作用为定义服务器的主管理网段的网卡编号，与NetCard类关联；os，作用为定义服务器上安装的操作系统类型；state，作用为定义服务器的状态，是开启、关闭或异常状态。

PCServer类，该类是Server的子类，定义了实体机服务器需要的相关信息。其主要成员变量设计有Indate，作用为定义服务器开始使用时间；model，作用为定义服务器模式，与ServerModel类相关联；cabinet用来描述该服务器在机柜的位置；kvmNum用来定义该服务器上创建的虚拟机数量。

KvmServer类也是Server的子类，设计为描述kvm虚拟机需要的信息，主要成员变量有host，用来描述所属实体机信息；deployState用来记录应用在虚拟机的部署状态。

## 4.4 虚拟资源管理模块的设计和实现

根据前述章节的虚拟资源管理模块的需求分析可知，该模块需要实现kvm虚拟机的定制与创建、kvm虚拟机的管理功能这两部分功能，以下详细阐述这两个功能的设计与实现过程。

### 4.4.1 kvm虚拟机的定制及创建功能的设计

根据需求分析可知，kvm虚拟机的定制和创建功能需要实现的主要功能如下：

1、虚拟机的创建和管理。因为需要安装多个版本的多种操作系统，所以需要对各个版本操作系统的镜像进行维护，在用户选择操作系统和版本后进行相应的安装。安装完成的虚拟机需要备份，在虚拟机奔溃时恢复；

2、权限控制和安全性。运维系统搭建完成环境后，要给指定的集成人员和测试人员分配访问权限，对其他人员禁止访问权限；

3、出现错误的应急处理措施。在实体机或虚拟机出现故障，要采取应急措施快速恢复环境。

安装虚拟机的流程设计如下图所示：

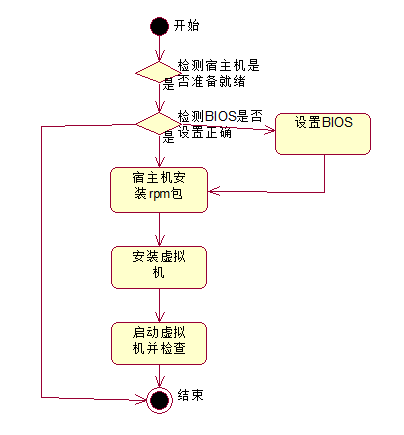


图 4.5 虚拟机的创建流程

其中在检测步骤中主要的检查项包括：

1. 是否能够连通各个实体机的管理网段IP；
2. 是否能够使用ssh登录到实体机的root用户；
3. Linux虚拟机yum源是否配置正确；

另外，系统还需要检查各宿主机的BIOS设置中的“Virtualization Technology”选项是否开启，如果没有开启，需要对其BIOS进行设置使其满足安装虚拟机的需要。

该步骤完成的判定条件为：

1. 各虚拟机正常安装并正常运行；
2. 能通过管理网段的IP连接到虚拟机；
3. Linux虚拟机配置了正确的yum源；

### 4.4.2 kvm虚拟机的管理的功能设计和实现

根据前述章节的需求分析可知，kvm虚拟机管理模块需要实现的主要功能如下：

1. kvm虚拟机的定制和创建；
2. kvm启停、扩展和回收功能；
3. 镜像的管理；

kvm虚拟机的管理子模块由两部分组成，一部分作为远程受控端提供RPC接口，部署在各实体机服务器上，另一部分Web服务端，展示给用户操作，并按用户操作指令调用远程受控端。

Web服务端直观的展示虚拟机的属性，并将其与所在的实体机服务器结合起来。用户通过实体机的机型，所属组的分类，将列出所有的实体机列表，同时展示该实体机上的虚拟机的情况。虚拟机创建以后，管理员可以很方便的搜索出需要的虚拟机，执行启动，停止等操作，并可以扩展磁盘容量、内存指标，CPU核数等信息，本系统调用服务器上的远程受控端程序实现。

对于虚拟机镜像的管理，本平台采取定时克隆备份和手动执行备份相结合的操作，将使用的虚拟机镜像备份处理。在虚拟机的使用过程中，使用人员也可以根据需要，手动执行保存当前的虚拟机状态的按钮，将当前虚拟机的镜像保存下来，以备以后还原到当前状态。

### 4.4.3 kvm虚拟化的应用实现

在本系统设计中，选用了相对轻量级的KVM虚拟化技术，并用自动化安装脚本实现kvm虚拟机的安装步骤。

KVM虚拟机自动化安装采用脚本实现，该脚本位于远程受控服务器上，是受控程序的一部分。它的输入参数为用户指定的IP地址，内存和cpu核数，根据这些设置信息创建kvm虚拟机。

该脚本具体实现的步骤如下：

1. 创建镜像

前提是实体服务器上已安装QEMU，并准备了各操作系统的标准镜像。以安装Redhat Linux5.5操作系统为例：

|  |
| --- |
| #1. create kvm.img  qemu-img create kvm${NETHOST}.RHEL55.img 50G |

1. 配置网络

以标准镜像创建的kvm虚拟机需要修改网络设置。以Redhat Linux为例，需要修改eth0网卡和bridge网桥的配置。

1. 安装虚拟机

确认了镜像和操作系统，就可以安装虚拟机的操作系统了。以下部分代码展示了安装虚拟机的安装设置，其中“-m 8196 -smp 2”表示内存为8G，cpu核数为2。

|  |
| --- |
| /usr/libexec/qemu-kvm -m 8196 -smp 2 -localtime -name kvm${NETHOST},process=kvm${NETHOST} -pidfile file${SNUM} -no-reboot -hda /images/kvm${NETHOST}.RHEL55.img -cdrom ./rhel-server-5.5-x86\_64-dvd.iso -net  … -initrd initrd.img --daemonize |

1. 启动虚拟机

启动虚拟机的命令也需要设置IP地址和macaddr地址等信息，需要网络配置步骤相对应，否则虚拟机将无法访问。

通过以上步骤，kvm虚拟机就可以快速创建并使用了。实际测试，一台kvm虚拟机的创建并安装操作系统的时间平均为3分钟。

### 4.4.4 虚拟资源管理对象交互接口的设计

由于本系统在设计中采用前后端分离的架构，即后端控制器和前端管理平台分离，采用RPC接口的方式通讯。前后端之间的交互方式由后端控制器提供调用的RPC接口，前端管理平台根据需要调用。

控制端的接口设计为三部分，第一部分接口作用为网络配置的接口，如vport.create，作用是基于OpenvSwitch配置网卡，配置指定虚拟VLAN；vport.delete，作用为删除指定的网卡信息；

第二部分接口是为虚拟机的管理配置设计的接口。如vm.boot，作用为启动kvm虚拟机；vm.terminate，作用为停止虚拟机进程；

第三部分接口是为镜像管理设计的接口。如image.list，作用为列出虚拟机镜像文件列表；image.copy，作用为复制指定的虚拟机镜像；image.backup，作用为备份虚拟机镜像；image.extend，作用为将虚拟机硬盘扩大到指定额度；

## 4.5 资源分配管理模块的设计和实现

根据前述章节的资源分配管理模块的需求分析可知，该模块需要实现运维资源申请、审批、分配和回收的流程，以下就该流程的具体设计与实现进行详细分析。

### 4.5.1 运维环境申请流程的设计

本系统在设计中对资源的分配是按用途以环境为单元进行划分的。当本系统的用户（包括开发人员、测试人员、集成人员）需要本系统提供工作环境时，可以在本系统中申请，由管理员审核通过后，本系统根据用户需要自动搭建并部署环境。

运维环境申请的具体流程设计如下图所示：

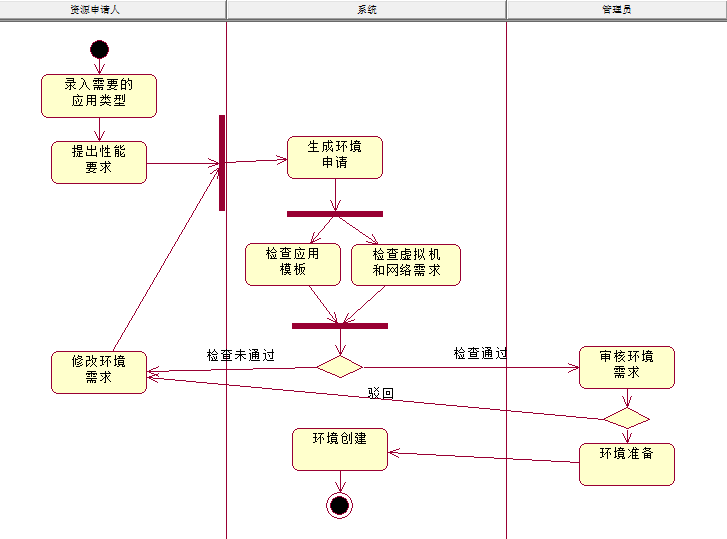


图 4.6 运维资源申请流程

该流程的具体设计描述如下：

1. 用户提出对于运维资源的用途，例如需要部署某种类型的应用；
2. 用户提出自己对于主机性能的需求，如果对性能没有特殊要求可以使用默认模板；
3. 系统检查用户需求的应用程序模板、虚拟机和网络的需求，如果需求不合理，例如提出超过系统接受的需求，则驳回用户重新填写；
4. 管理员的审核用户需求；
5. 审核通过后，系统分配对应的运维资源，创建单元环境。

### 运维环境回收流程的设计

当运维资源按环境分配后，在使用完毕后需要回收注销环境，以释放运维资源。回收有使用人员主动归还和系统强制回收两种形式。回收流程的设计如下：

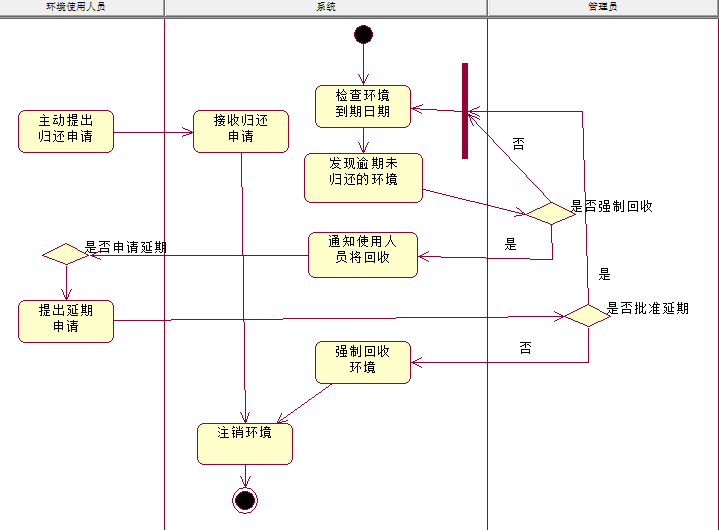


图 4.7 运维资源回收流程

对该回收流程的具体设计如下：

1. 系统根据应用环境的到期日期检查当前是否有环境到期。环境的到期日期在创建时指定；
2. 如有环境即将到期，本系统提交系统管理员是否需要强制回收；
3. 如果系统管理员确认强制回收，本系统将通知环境使用人员即将回收环境；
4. 如果环境使用人员提出延期要求，需要系统管理员确认；
5. 如果系统管理员批准延期要求，系统将到期日调整为下一个应用周期的到期日期；
6. 如果系统管理员不批准延期请求，系统将强制回收；
7. 系统注销环境，释放运维资源。

### 4.5.3 运维资源管理流程类设计和实现

根据以上的流程设计，设计了本模块中定义的主要JAVA类，其类图如下所示：

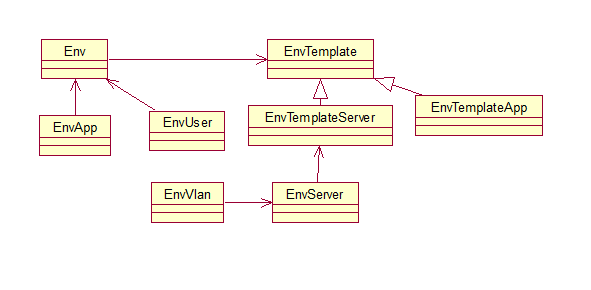


图4.8资源分配模块管理类图

其中，Env类是核心类，作用是描述环境的信息，其主要成员变量有deployState，作用是描述环境的部署状态，expireDat作用是记录环境到期时期，用户申请环境时需要设定期限，到期需要将环境归还。如超期未重新续借，管理员可以将环境强制回收；EnvApp类的作用是描述环境中需要部署的应用，依照EnvTemplateApp的设计生成；EnvUser类的作用是定义对该环境有使用权限的用户；EnvVlan类作用是描述环境中应用的网络资源；EnvServer类描述了环境的部署所在的服务器；

EnvTemplate类是模板的具体说明类，定义环境的基线模板；EnvTemplateApp描述环境应用模板中有哪些应用；EnvTemplateServer的作用是定义环境模板中的主机的具体信息，例如其对cpu，内存和硬盘的需求；

根据流程设计，资源分配管理流程时序图设计如下：

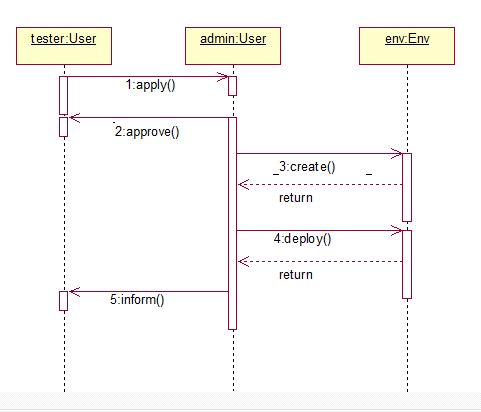


图4.9资源分配管理分时图

## 4.6 环境自动化部署模块的设计和实现

根据前述章节的环境自动化部署模块的需求分析可知，该模块分为环境准备、应用部署这两个子模块组成，以下就这两个子模块的具体设计与实现进行详细分析。

### 4.6.1 环境部署流程设计

环境部署的流程在设计上，按功能划分为两个阶段，第一个阶段是主机准备，第二个阶段是部署应用系统。主机准备阶段分为寻找适合的服务器安装虚拟机、虚拟机的环境定制两个步骤；部署应用系统阶段分为发布应用系统和冒烟测试等两个步骤。在主机准备阶段，由于部署过程对于各个虚拟机来说是独立的，所以以主机为单位并行执行。在应用系统部署阶段，各个部署步骤一般以应用组为单位并行执行。

设计了环境自动化部署的整体流程如下图：

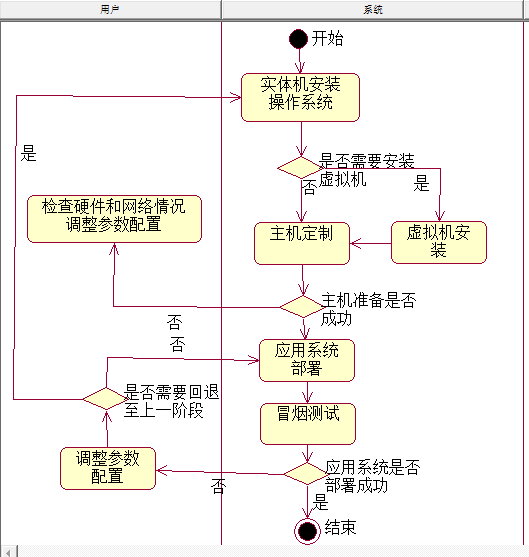


图4.10环境部署流程图

系统会在每一个步骤开始前对上一个步骤是否执行成功加以检查。各个部署步骤都是可以重复执行的，如果在部署过程中发现错误，用户可以通过调整配置并重复执行某些部署步骤来进行修正。

### 4.6.2 主机定制问题的描述与分析

主机定制是根据各主机的最终用途，对操作系统进行一些必要的配置。在应用部署时，对主机的需求是不一致的，例如Linux系统下数据库的安装，需要对操作系统参数进行配置，安装依赖的包等；此外，不同主机在定制相同的配置项时，其具体的配置内容也是不同的。所以主机定制是应用自动化部署前的重要准备工作。

因为应用程序的对环境的需求差别，主机定制在不同主机上的操作也不一致。主机定制会遇到以下问题：

1. 如何确认当前主机环境是否符合要求；
2. 如何远程安装应用程序所依赖的包；
3. 如何配置主机网络；

### 4.6.3 主机定制问题的解决方案

针对上一小节提出的问题，设计的解决方案如下：

* 1. 在主机定制步骤的开始，首先检查当前主机的配置信息，与基线中应用程序的具体要求比较，得出当前主机配置是否符合标准的结论。判定标准将在下文详细论述；
  2. 远程复制需要的脚本和软件到目标宿主机，执行自动安装脚本。自动安装脚本的实现将在下文详细论述；
  3. 通过网络配置脚本修改网络设置 /etc/sysconfig/network-scripts下的配置。网络配置脚本的实现将在下文详细论述；

在主机定制过程中需要检查的主要配置内容主要有以下方面：

1. BIOS设置；检查bios.conf文件，该文件不是OS标准配置文件，而是一个类似ini格式的配置文件，包括以下几项具体bios配置：
   * + - C1E：enable/disable
       - CStates：enable/disable
       - TurboMode：enable/disable
       - PowerManagement：max
       - VT：enable/disable
2. 网络路由配置。需要检查以下相关配置：
   * + - default-gateway：默认网关地址
       - eth-list：eth设备列表
       - ethX-static-routes：ethX设备上的静态路由配置，比如：

|  |
| --- |
| 172.16.60.0/24:172.16.80.100 172.16.70.0/24:172.16.80.100 |

表示两条静态路由

* + - * bond-list：bond设备列表
      * bondX-static-routes：bondX设备上的静态路由配置

1. 用户管理。检查组/用户新建，删除，密码管理，权限管理，用户Key等；
2. 系统参数。检查/etc/sysctl.conf中的配置；
3. ntp配置。检查ntp-server是否为ntp服务器对应的主机名称；
4. iptables防火墙。检查/etc/sysconfig/iptables中是否符合了基线中要求；
5. dns配置。检查dns server是否配置；
6. 检查需要安装的rpm包；
7. 检查需要启动的服务；

以上检查项都通过时，就可以确认主机配置检查完成。

远程自动安装脚本安装需要的包，使用yum安装，结合expect实现了自动安装，以安装kvm-tools为例示例如下：

|  |
| --- |
| #!/usr/bin/expect  set timeout 3  spawn /usr/bin/yum install kvm  expect "N]:"  send "y\r"  interact |

对于其他应用包，同理也按expect的方式实现自动安装。

网络配置脚本需要修改网络设置 /etc/sysconfig/network-scripts下的配置，ifcfg-eth0，ifcfg-br0等配置文件，修改完成后需要重启网络使其生效。

网络配置脚本的具体实现如下所示：

|  |
| --- |
| #!/bin/bash  cd /etc/sysconfig/network-scripts/  mkdir bak  cat ifcfg-eth0 |grep IP >> net.temp  cat ifcfg-eth0 |grep NET >> net.temp  mv ifcfg-eth0 ./bak  # edit for eth0  echo "DEVICE=eth0" >> ifcfg-eth0  echo "BOOTPROTO=static" >> ifcfg-eth0  echo "ONBOOT=yes" >> ifcfg-eth0  echo "BRIDGE=br0" >> ifcfg-eth0  echo "USERCTL=no" >> ifcfg-eth0  # edit for br0  echo "DEVICE=br0" >> ifcfg-br0  echo "TYPE=Bridge" >> ifcfg-br0  echo "PROTO=static" >> ifcfg-br0  echo "ONBOOT=yes" >> ifcfg-br0  echo "USERCTL=no" >> ifcfg-br0  cat net.temp >> ifcfg-br0  # reboot network  service network restart |

### 4.6.4 应用自动化部署问题的解决方案

应用部署是发布过程中最关键的一步。应用系统部署过程中，会遇到如下一些问题：

1. 如何确定应用组件的发布顺序，哪些组件需要优先部署；
2. 应用发布完成后，如何检查应用环境是否能正常使用；

针对这些问题，在设计中经过试验，提出了以下解决方案：

1. 应用组件在模板中设置优先级；
2. 应用发布完成后，启动冒烟测试验证；

以下对本系统设计的解决方案作具体阐述：

在应用组件中，有些组件是属于公共组件，所以必须优先进行部署。在公用组件部署结束后，剩余的部署工作一般都可以以应用组为单位并行进行。

在实际情况中，大多数应用系统都有相应的控制台负责系统发布和启停等管理工作，对于这一类应用系统的部署，本系统的做法是首先搭建应用自己的控制台，剩余的工作都通过调用控制台的接口来完成。对于没有控制台的系统（比如数据库），整个部署工作都由测试管理系统完成。

组件部署完成的判定条件如下，必须在这些判定条件满足以后，才能认为该应用组件部署完成，进入下一优先级的应用部署：

1. 软件已正确安装（包括数据库等）；
2. 应用系统已正确发布；
3. 各应用系统的参数配置正确；

在用户配置过程中，用户需要依据基线中的模板提供控制台的配置参数（仅针对由控制台管理的系统）以及应用系统的配置参数。对于某些有关联的参数，系统会加以校验，保证用户的配置是正确的。

冒烟测试是整个部署过程的最后一步。冒烟测试是为了验证应用环境部署的正确性，如果冒烟测试通过，则表示应用系统都已正常安装，且均处于可用状态。冒烟测试的方法根据系统的不同而不同，简述如下：

1. 业务系统
   * 查看业务服务进程是否启动；
   * 执行业务系统的冒烟测试功能进行一些基本检查。
2. 数据库系统
   * 检查基本的数据库实例进程是否已经启动；
   * 测试数据库连接，检查数据库用户是否创建；

### 4.6.5 应用自动化部署流程的设计

为实现应用的自动化部署，设计了基于基线的应用自动化部署流程。本流程的具体方案设计为：

1. 确定发布的内容。首先用户要先选择需要部署的应用。本运维管理平台根据对于自动化部署应用，能实现的范围为本系统能识别的 应用模板，首先确定应用部署的范围，需要的虚拟机数量，虚拟网络类型，然后创建虚拟机。
2. 虚拟资源准备完成后，进行主机定制，配置出应用程序所需要的环境变量等，提示用户选择版本。
3. 根据用户指定的版本，启动自动化部署步骤，分组的部署指定的应用，并在用户界面上以进度条的方式直观的展示部署程序的进展。
4. 配置比对。配置比对主要用于在环境发布完成后，比较应用环境的发布结果和基线是否一致。由于测试环境的规模可能比生产环境要小，故在比对时应当仅比对两者中共有的主机和应用配置。在配置比对过程中，有很多配置文件（比如主机的网卡配置文件，路由配置文件，/etc/hosts文件等）由于体现了IP信息，所以比对的结果一般情况下必定是不一致的（除非测试单元环境中的IP地址和基线环境完全一致），这样可能会对总体的比对结果形成干扰，不利于用户判定发布结果是否实质上一致。可以采取两类方法部分地解决这个问题：第一种方式是比对时过滤某些配置文件不进行比较；第二种方式是在比对之前将这些配置文件中的IP全部自动替换为相同的字符串，然后在替换的结果基础上进行比对。
5. 基线的修订。基线是测试环境发布的基准，它规定了在安装部署过程中需要修改的配置项范围，各个配置项的修改方法、修改顺序以及用户必须提供的配置参数。基线中还决定了将要发布的各个系统的版本，用户可以通过选择基线来决定是发布一个生产环境的最新版本或某一历史版本。基线将保证不同系统之间版本的兼容性。初始的基线在系统上线时由人工构造的方式产生。其中配置模板和版本号都应当与该应用的生产系统的实际情况保持一致。在初始的基线建立之后，为了适应生产环境对于操作系统和应用系统方面的变更，必须不断地建立新的基线。除了少数变更不需要修改基线外（比如系统用户的例行密码修改），大部分变更都会导致新基线的建立。如果变更仅修改了环境无关的配置项（如iptables新增端口），则新的基线可以使用生产环境的收集结果自动建立；如果变更修改了环境相关的配置项（如应用系统配置文件修改，增加了某个和环境有关的参数），则新的基线一般无法自动生成，应通过补丁的形式手工修改。

## 4.7 本章小结

基于前述章节描述的需求和用例，本章描述了本运维管理系统的架构设计，以及主要模块的设计和实现过程，并对其主要功能模块进行了详细描述，包括实现逻辑，效率优化方案，实现技术等。

本章首先设计了本系统的总体架构设计。基于虚拟化操作、监控功能设计的考虑，本系统架构划分了管理平台、控制器和远程受控端程序三个部分，并对各部分的架构做了详细描述；

本章接着对按功能模块划分的各功能模块分别进行了详细设计和实现。对硬件资源管理模块中自动化的问题，设计了网络安装和主机探测的流程，避免了硬件资源管理的录入流程；对虚拟资源管理模块，设计了kvm虚拟的定制创建和管理命令的流程和实现方法；对于资源分配管理模块，设计了资源申请、审核、分配、回收一整套流程，解决了空闲资源再利用等问题；对于应用环境的自动化部署模块，设计了环境部署的流程，针对主机定制问题，分析了主机定制的远程安装等问题，接着重点分析了应用自动化部署中的问题并给出了解决方案。

# 系统运行分析

本文设计与实现的运维自动化平台已在公司实际环境中部署，统一管理公司的运维资源。本章节将从运维自动化平台的物理部署、运行平台和实际运行中的情况等几个角度进行系统运行分析。

## 5.1 系统物理部署

由于运维自动化平台需要管理包括实体机服务器和虚拟机在内的所有服务器，所以本系统所部署的服务器需要全面访问各网络的权限。在实际使用环境中，其具体无力部署如图5.1所示：

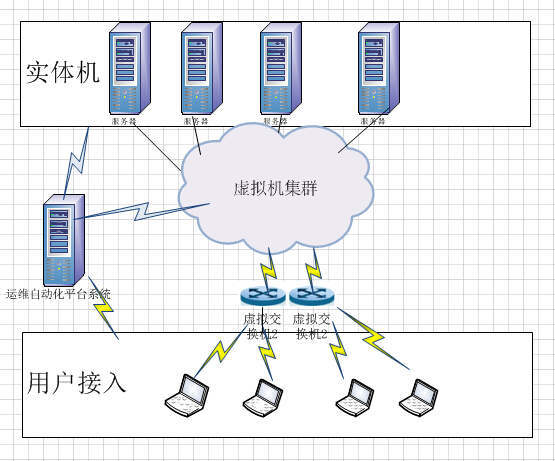


图5.1 物理部署图

从图5-1所知，虚拟机用户访问虚拟机集群是通过虚拟交换机介入的。在虚拟交换机服务器上部署了Open vSwitch虚拟交互软件来创建虚拟网络，从而让服务器上的虚拟机可以以不同的虚拟网段来分离开来，各自之间不能连通，保证了独立性。

从用户访问的安全性和独立性来考虑，为了保证用户接入虚拟网络后，只能访问系统指定的虚拟机和虚拟网络，我们在虚拟交换机上安装了 openvpn server认证服务，用户在自己的笔记本上需要安装openvpn client，通过其以windows域账号登录到openvpn server。Openvpn server需要维护用户域账号于访问权限的关系，从而确定用户可以访问虚拟资源。

运维自动化平台所部署的服务器需要有访问实体资源的权限和虚拟化资源，所以该服务器上需要配置多块网卡，分别连接到实体资源和虚拟化资源。

## 5.2 系统运行平台

系统运行平台是运维自动化平台能够稳定运行的保证。平台选型是一项专业性强，技术难度高的工作，既要保证系统的正常运行，又要考虑扩展性，可移植性等问题。本次通过遵循开放性、实用性和可扩展性等选型原则，并结合系统实际使用情况进行平台的选型。以下将从软件平台和硬件平台两个方面进行阐述。

1. 软件平台

软件平台主要是开发系统所用到的开发、建模、管理工具，系统所用的数据库，以及系统所部署的应用服务器等。本系统所使用的软件平台如表5.1所示：

表5.1软件平台配置表

|  |  |  |  |
| --- | --- | --- | --- |
| 软件类别 | 软件名称 | 软件类别 | 软件名称 |
| 操作系统 | Redhat Linux6.6 | 开发语言 | Java SE1.6, python |
| 开发工具 | Eclipse3.4.1 | 数据库系统 | Mysql5.5 |
| 项目管理工具 | Microsoft Project 2007 | 测试管理工具 | Junit4 |
| Web服务器 | Tomcat6.0.30 | 配置管理工具 | subversion |
| 建模工具 | Rational Rose 2003 | 质量管理工具 | BugFree |

1. 硬件平台

硬件平台主要指服务所部署的服务器，按照用途来分，可以分为Web服务器、数据库服务器以及应用服务器（后端执行指令的程序）等3个部分。

根据系统物理部署设计，硬件平台的具体配置如表5.2所示：

表5.2硬件平台配置表

|  |  |  |
| --- | --- | --- |
| 服务器种类 | 硬件需求描述 | 数量 |
| 数据服务器 | 两路4核CPU，内存64G，1T本地硬盘，4个网络接口 | 2 |
| Web服务器 | 两路4核CPU，内存64G，500G本地硬盘，4个网络接口 | 2 |
| 应用服务器 | 部署在需要管理的实体机服务器上 | 视具体情况部署 |

## 5.3 系统用户体验效果分析

系统的运行效果是系统是否受欢迎的重要因素，在越来越重视用户体验的情况下，系统运行的界面效果是我们必须关注的方面。下面我们来看一看本文提到的一些模块的运行效果示例。

1. 硬件资源管理

为了让系统管理员更清晰的了解出现问题的服务器的具体位置，我们通过在服务器拓扑图中标识出出错的服务器，从而可以让运维人员方便的到机房中找到服务器，排除故障。服务器拓扑图示例如下，其中红色表示服务器出故障，鼠标放上服务器会显示具体IP地址：



图5.2硬件资源管理页面示例图

1. 环境自动化部署

环境自动化部署中必然会有一些等待的时间，如何能让用户在等待的时候确切的知道系统正在做什么事情，从而不必为无聊的等待失去耐心呢？而同时，在系统出错的时候，又要能准备的反映给用户，让用户得知在什么地方出错，如何处理。

以下为系统自动部署某应用的界面示例：



图5.3 应用发布界面

如上图，执行成功界面显示绿色，而一旦出错，会显示醒目的红色。同时，每一步执行的事物也很清晰的显示在页面上，一旦中途出错，系统就会用红色标识出出错的步骤，机器出错的环境套数和名称，这样同时发布多套环境的时候，相互之间不会干扰，出错的环境也能很明显的显示具体的出错阶段，非常方便用户了解环境部署的进展情况。

1. 系统参数设置

系统中不可避免有需要参数设置的地方，本系统应用bootstrap前台框架，在参数设置中考虑了人性化的设计。例如true和false的设置可以用开关按钮实现等，具体如下图所示：



图5.4参数设置界面

## 5.4 系统使用前后效果分析

在本系统投入使用以前，运维资源的管理面临很多问题，例如应用环境搭建进展缓慢，使得软件开发周期延长等问题。而在本运维自动化平台系统投入使用后，情况得到很大的改观。以下从几个方面对比系统使用前后的改善情况：

1. 运维资源管理方面的比较。

在本系统使用前，系统管理员是使用Microsoft Excel记录服务器信息的。但是随着服务器的不断增多，单纯用Excel记录服务器信息的方式使得系统管理员对运维资源的管理越来越力不从心。而且随着虚拟机的创建与发展，单纯记录的方式变得不可能了。

本运维平台系统投入使用后，服务器的信息都记录在本系统内，并且可以按需要导出服务器信息报表。目前本系统管理300多台实体机服务器，以及服务器上创建的虚拟机服务器约500台。本系统按服务器的机型等多种条件对实体机服务器进行分类处理，使系统管理员能够按多维查询条件迅速找到需要的服务器。另外，本系统提供了按IP地址查询的功能，能迅速按IP地址找到所查询的服务器。经实际测试，用户查询所消耗的时间在100ms以下。以下是本系统主机查询的多维查询条件查询界面效果：



图5.5查询条件界面

1. 应用环境部署的效率方面的比较。

在本系统使用前，运维人员是以手动的方式部署应用环境。由于应用环境通常对网络需求比较复杂，所以运维人员部署一套应用环境平均需要一至两天时间，这不仅使得运维人员的工作非常繁重，而且极大的影响了项目开发的进展时间。

本系统使用后，由于对应用环境使用自动化部署，使应用环境的准备时间大大缩短，应用环境从搭建到部署完成的平均时间为3小时。应用环境部署的自动化，使运维人员从不断的准备环境的繁重工作中解放出来，缩短了项目开发流程中的环境准备时间。

1. 环境申请流程方面的比较。

本系统使用前，应用环境的需求人员需要向系统管理员邮件申请，系统管理员也以邮件回应。有时需求人员没有将需求描述清楚，需要多次邮件沟通，极大的影响工作效率。另外，由于应用环境没有到期回收的机制，导致运维资源分配出去后，不能及时的回收利用，对于运维资源来说是一个很大的浪费，而且使新的应用环境没有足够的资源可以利用。

本系统使用后，应用环境的需求人员，可以在本系统填写应用环境申请。由于有统一的应用环境需求模板，所以可以明确的将需求描述清楚。应用环境申请填写完成后，本系统会提醒系统管理员审批。如果系统管理员驳回申请，会立刻通知申请人员。如果系统管理员批准申请，在通知申请人员的同时，系统会计算运维资源，准备应用环境，从而极大的节省应用环境的申请和准备环节所耗费的时间。

## 5.5 与类似的方案比较

本运维自动化平台系统与类似的平台系统比较如下：

* 1. 与Openstack比较

Openstack平台在管理虚拟化资源方面比较有优势，它能集成kvm虚拟化技术或其它虚拟化技术，在虚拟机的管理方面提供接口，使得虚拟机的创建，注销、迁移等操作通过接口可以实现。但是它的弱点是在自动化部署方面的功能没有本运维自动化平台全面。在部署复杂的应用程序的时候，本运维平台的基于基线和模板的自动化部署解决方案将更加实用。

* 1. 与chef比较

Chef是一个自动化部署的工具。通过Chef提供编程接口，可以定制应用程序的自动化发布。它与本运维平台系统比较的缺点是，应用程序的部署没有和环境申请等流程结合起来，所以在自动化发布的效率上不如本系统。此外，应用程序的自动化部署程序需要通过Ruby编程语言编写，对于不熟悉Ruby语言的运维人员来说，又增添了学习新语言的负担。

# 总结与展望

本章以下部分将对上述几个章节所阐述的内容做一个总结，并在此基础上对下一步的研究工作进行展望。

## 6.1 本文工作总结

本文阐述了基于KVM虚拟化技术的管理运维平台系统的设计和实现过程。本课题的研究目的是为了解决虚拟化资源的创建、管理、分配和自动化部署的问题，即如何构建一个管理平台，将运维资源合理的规划，利用虚拟化技术切分为具体的应用环境，并按用户需求自动化部署。

本文通过设计与实现一个基于kvm虚拟化技术的运维自动化平台系统来解决以上提出的问题，本系统能帮助系统管理员管理硬件资源，创建并维护虚拟资源并自动部署应用，也能将虚拟资源合理的分配到硬件上，分析用户的需求，规划合理的虚拟资源来创建应用环境。

本文研究的工作内容主要有以下几个方面：

1. 对开发实现本运维平台系统所用到的相关技术，以及需要解决的关键性问题进行了详细探讨和分析；
2. 对本运维平台系统按功能模块进行了划分。根据运维自动化平台的需求，将功能划分为硬件资源的管理，虚拟资源的管理，资源申请分配管理，应用环境自动化部署这四个主要的方面，并对应用环境自动化部署这一模块的实现逻辑以及实现方法进行了重点描述。
3. 对本运维平台系统进行了系统架构设计。本系统采取前后端分开，用RPC异步调用的架构设计。这种实现方式能使前后端解耦合性更好，前后端分别采用java和python语言实现，更大程度的发挥不同技术语言的优势。可扩展性也是另一个重要的优点。本系统可以自动检查新加入的服务器，采集硬件信息保存到本系统中，使服务器在机房的上架下架操作非常方便。
4. 本文的特色在于将KVM虚拟化技术、自动部署应用的技术应用在自动化运维系统中，实现了KVM虚拟机在受控主机上的安装、启动、克隆和迁移等操作，并自动部署标准化的应用程序。另外，本系统在分配资源环境，对硬件和虚拟资源的可视化管理，也探索出一套成熟的解决方案。自投入实际使用以来，规范了公司运维资源的管理，极大的提升了效率，简化了系统管理员的工作。

## 6.2 展望

本文介绍的基于KVM虚拟化技术的运维自动化平台，完成预订的任务，投入使用以来状态稳定，但是还是有许多地方值得进一步研究和改善。主要包含以下几个方面：

(1) 持续交付与本系统的结合。本系统自动部署部分的主要功能部署已上线的维护项目，对开发项目的灵活性支持不够，下一步需要和持续交付系统结合，构建从设计、开发到上线的一站式功能，更好的提升效率。

(2)监控与分析。本系统目前已实现服务器级别的监控，在硬件服务器和虚拟主机发生故障时能及时发出预警，但应用程序级别的监控还没有实现。

（3）自动计算回收功能。本系统分配环境资源到使用人员后，如果不自动回收，会造成资源的浪费。运维自动化平台可以提供一个更高效的回收机制，利用回收算法回收不需要在使用的环境资源，优化虚拟资源的使用率。

# 参考文献

[1]岳斌. 基于TAM的IT系统能力分析[J]. 邮电设计技术,2014,11:18-23.

[2]王锐. 谈企业电信业务运行支撑平台[J]. 统计与管理,2014,12:132-133.

[3]赵凯,杨照宇. 基于Vmware vSphere虚拟化的技术实践[J]. 山东通信技术,2014,03:34-37.

[4]王朋涛. [大规模OpenStack集群自动化部署与系统管理研究](http://www.cnki.net/kcms/detail/detail.aspx?filename=1014227801.nh&dbcode=CMFD&dbname=CMFD2014&v=)[D]. 杭州电子科技大学 2014

[5]林凯. [基于CloudStack的云计算资源监控系统的研究与实现](http://www.cnki.net/kcms/detail/detail.aspx?filename=1014078805.nh&dbcode=CMFD&dbname=CMFD2015&v=)[D]. 上海大学 2014

[6]刘国乐,何建波,李瑜. Xen与KVM虚拟化技术原理及安全风险[J]. 保密科学技术,2015,04:24-30.

[7]赵峰,石佳磊,李祉岐,孙磊. 基于OpenStack的电力信息网运维平台研究[J]. 制造业自动化,2015,13:148-152.

[8]庄天天. 安全运维平台关键技术的研究与实现[D].北京邮电大学,2013.

[9]肖增. IT运行管理系统的研究与实现[D].华东师范大学,2009.

[10]彭琳,蒲立新,曲建明,高忠军. 基于医院信息化系统的IT智能运维平台的设计和实现[J]. 中国数字医学,2014,04:58-61.

[11]张毅. 基于OpenStack的虚拟桌面云系统服务端设计与实现[D].华南理工大学,2013.

[12] 雷葆华. 网络管理[D]. 电子工业出版社,2013.

[13]秦学东. 开源虚拟化——KVM的构建[J]. 现代图书情报技术,2011,11:89-92.

[14]车翔. QEMU-KVM设备虚拟化研究与改进[D].成都理工大学,2012.

[15]徐燕雯. 基于KVM的桌面虚拟化架构设计与实现[D].上海交通大学,2012.

[16]黄煜,罗省贤. KVM虚拟化技术中处理器隔离的实现[J]. 计算机系统应用,2012,01:179-182.

[17]余平. QEMU-KVM实现虚拟机动态负载均衡[J]. 电子商务,2012,08:59-62.

[18]李斌,胡屹峰. 基于KVM的服务器虚拟化环境实现[J]. 洛阳理工学院学报(自然科学版),2013,02:66-69+76.

[19]赵天麒. 云计算环境下WEB应用的自动部署与资源分配[D].哈尔滨工业大学,2014.

[20]刘广宇. 客户端程序自动部署系统的设计与实现[D].东北大学,2011.

[21]Brodkin, Jon. Linux KVM virtualization gains steam in cloud computing market[J]. Network World (Online),2010,:.

[22]Anonymous. Data Security; Belkin Announces New Advanced Secure KVM Switch Lineup[J]. Technology &amp; Business Journal,2011,:.

[23]Haebel, Annette. KVM technology[J]. Broadcast Engineering,2011,536:.

[24]Niels Veerman. Automated mass maintenance of a software portfolio[J]. Science of Computer Programming,2006,623:.

[25]Youhui Zhang,Yanhua Li,Weimin Zheng. Automatic software deployment using user-level virtualization for cloud-computing[J]. Future Generation Computer Systems,2013,291:.

[26]Feng Wang,Xiaoyang Sun,Shi Li,Yong Wang,Bingjia Xiao,Sidi Chang. The implementation of virtualization technology in EAST data system[J]. Fusion Engineering and Design,2014,:.

[27]Ning Liu,Xiaoping Li,Weiming Shen. Multi-granularity resource virtualization and sharing strategies in cloud manufacturing[J]. Journal of Network and Computer Applications,2014,46:.

[28]Adjidé V,Fournier P,Vassault A. The maintenance of automatic analysers and associated documentation.[J]. Annales de Biologie Clinique,2011,68HS1:.

# 致 谢

本次论文写作从立题、资料收集、起草到最后成文得到了许多老师、同学和亲友的帮助。在此谨向各位老师、同学和亲友致以真诚的谢意！

首先要感谢我的导师，老师在论文写作方面提供很多宝贵的材料。在论文的选题、资料收集、撰写和最后的成文过程中，也得到了导师悉心的指导。在论文开始写作的时候，对论文的写作没有计划，他在繁忙的教学任务中多次给我提出了许多宝贵的意见和建议，这些无疑将对我未来的工作产生很大的影响。在此谨向恩师表示衷心的感谢！

其次还有要感谢我的同学，在我平时的学习中给予了很多帮助。

最后需要感谢我的家人，尤其是我的妻子对我的支持，使我得以安心地进行论文的写作。

**论文独创性声明**

本论文是我个人在导师指导下进行的研究工作及取得的研究成果。论文中除了特别加以标注和致谢的地方外，不包含其他人或其它机构已经发表或撰写过的研究成果。其他同志对本研究的启发和所做的贡献均已在论文中作了明确的声明并表示了谢意。

作者签名： 日期：

**论文使用授权声明**

本人完全了解复旦大学有关保留、使用学位论文的规定，即：学校有权保留送交论文的复印件，允许论文被查阅和借阅；学校可以公布论文的全部或部分内容，可以采用影印、缩印或其它复制手段保存论文。保密的论文在解密后遵守此规定。

作者签名： 导师签名： 日期：