本科毕业设计（论文）

自动化服务部署平台设计与实现

Design and Implementation of Automatic Service Deployment Platform

学　　院：软件学院

专　　业：软件工程

学生姓名：XXX

学　　号：XXXXXXXX

指导教师：XXX

北京交通大学

2016 年 5 月

学士论文版权使用授权书

本学士论文作者完全了解北京交通大学有关保留、使用学士论文的规定。特授权北京交通大学可以将学士论文的全部或部分内容编入有关数据库进行检索，提供阅览服务，并采用影印、缩印或扫描等复制手段保存、汇编以供查阅和借阅。

（保密的学位论文在解密后适用本授权说明）

学位论文作者签名：　　指导教师签名：

签字日期：　　年　　月　　日　　签字日期：　　年　　月　　日

中文摘要

摘要：随着互联网的发展，越来越多的公司开始采用虚拟化服务器对外提供服务，为了实现负载均衡、动态迁移、故障自动隔离等特性，通常将一个服务分为多个子服务，再将这些子服务分别部署于不同的虚拟化服务器之上，为了保证服务的可用性，通常一个子服务也会部署在多台不同的虚拟化服务器上，然后有这些虚拟机协同向外提供服务。

为了提供一个服务通常可能需要几十台虚拟化服务器组成一个环境，这也就造成了服务器的监控和管理的麻烦，同时一套环境可能需要部署很多个分别用于研发和测试等不同的目的，但是由于一套环境中虚拟服务器众多，重新部署一套环境需要耗费一个人几天的时间。公司也就急需一套自动化服务部署平台。

自动化服务部署平台可以实现：服务快速部署，虚拟服务器监控与管理，代码批量和自动升级，虚拟服务器服务快速配置等功能。该平台开发过程中本人主要负责以下五个模块，平台首页：可以直观的监控平台所管理的虚拟机；虚拟机管理：提供了虚拟机快速创建，删除，重启和虚拟机服务代码升级功能；虚拟机配置：可以快速对虚拟机中的服务进行配置；日志管理：可以查看、备份和删除用户在平台中所进行的操作日志；用户管理功能：主要负责平台的用户添加和用户权限管理。

目前该平台已经开发完成并在部门内部投入使用，该平台上线以后为部门内部大量的虚拟服务器的管理提供极大了帮助，该平台提供了服务部署和配置功能极大了缩短了服务部署和配置的时间，代码批量升级功能也极大了缩短了代码升级时间，并且平台提供了虚拟机监控功能也能及时发现虚拟机中的服务异常。

该平台基于 Django 实现，支持 VMware vSphere 虚拟服务器，提供 web 图形用户界面以便于用户进行虚拟服务器管理以及服务部署等操作。

关键词：服务器虚拟化；Django；VMware vSphere；服务监控；服务管理；

ABSTRACT

ABSTRACT: With the development of the Internet, more and more company choose virtual machine over physical machine to provide services for customer. In order to achieve load balancing, failover and soon, service provider usually divide an service to several parts and deploy these parts in different machines, each part also will be deployed in several machine so that the service system will still run normally if an machine goes down, then all these machine provide an service to users together.

There are usually dozens even hundreds if virtual machines in order to combine an environment that can provide a well-functioning service. However, there are also some problem, it is a lot of work to manage these virtual machine, and we sometimes also need several same environments for the convenience of testing and development, and due to the massive amount of these virtual machine, it will be such a pain to deploy and configure these machine one by one. So we need a platform that can deploy and manage these virtual machines.

With Automatic Service Deployment Platform, we can deploy a service rapidly, monitor and manage our virtual machine, configure our virtual machine conveniently. And I am responsible for these 5 modules: the platform front page where we can monitor all virtual machine intuitively; virtual machine management where we can create, delete, reboot and upgrade virtual machines; virtual machine configuration where we can configure our service in these virtual machines; log management where we can look up, back up and delete operation logs; user management where we can add users and manage user's permissions.

Now the development is complete and Automatic Service Deployment Platform has been put to use in my department, and the platform help a lot in our virtual machine management work, the platform greatly reduce the time of deploying and configuring service, with virtual machine upgrade function we spend less time in service code upgrade, and virtual machine monitor function help us find service failure in time.

The platform is developed base on Django, provide a web-based interface for user to manage, deploy services and so on.

KEYWORDS： service virtualization; Django; VMware vSphere; service monitor; service management;

目　　录

1 引言

1.1 课题研究背景

本项目来源于在北京神州绿盟信息安全科技股份有限公司的实习项目，本人所在的云平台开发部门的主要任务是利用爬虫技术统计网上一些网站的暗链，漏斗等情况并产生将统计结果呈现给用户，部门的研发和测试环境主要由架设在 VMware vSphere 上的虚拟机构成，然后由一组虚拟机组成一个整体的环境对外提供服务。

一套环境中的虚拟机也扮演者不同的角色，比如有 web 类型的虚拟机，这类虚拟机主要向外提供 web 界面以便于客户的访问，有 spider 类型的虚拟机，这类虚拟机主要从监控的网站上爬取各类信息，有 MySQL 以及 Mongo 类型的虚拟机，这类虚拟机主要的的任务是存贮信息，以及 reportservice 类型的虚拟机，这类虚拟机主要将从网上抓取到的信息统计分析并且生成各种报表以供展示。

在一个环境中一个角色的虚拟机通常也有很多个，以此来保证服务的速度以及可用性，但是这样就造成了一个问题，一个环境中的虚拟机往往由十几个甚至是几十个构成，同时为了研发和测试的方便，一个环境往往同时需要很多个，这样一来同时管理的虚拟机数量就接近百个，而通过 VMware vSphere 的客户端也只能一台一台进行操作，这样如果需要进行服务配置或者代码升级就会及其麻烦，同时 VMware vSphere 的客户端也不能一次进行一个环境的部署，还是需要一台一台的创建然后进入虚拟机后台进行配置，如果研发或者测试人员急需一套环境的话可能就要耗费一个人一两天才能将一个环境整体搭建完成，极大的降低了开发和测试的效率。这样就急需一个平台对这些虚拟机进行管理。

1.2 课题研究意义

这个平台是一个自动化服务自动部署和管理的平台，通过这个平台可以实现:

- 快速进行服务部署的能力：由于服务的复杂性，一个服务通常由几十个虚拟化服务器构成，这些虚拟机又有很多不同的角色，从零开始构建配置一个虚拟机通常需要服务器创建，系统安装，软件安装，以及服务安装与配置等过程，为了将一个环境中的虚拟机全部配置完成这个工作通常需要重复几十遍，但是由于各个虚拟机的角色不同每个过程又有些许不同，这就造成了一个环境的搭建要耗费一个人几天的时间，但是利用该平台可以将服务部署几小时内完成，而

且用户要做的也只是几下点击。

- 虚拟服务器可用性实时监控的能力：为了保证服务的可用性，该平台提供了两套监控方法，一是可以通过平台的 web 界面直观的看到这台机器的网络可用性，当这台机器的网络出现问题时在平台首页可以立即观察到，平台首页的刷新效率已经达到秒级别，一台虚拟机网络出现问题后几秒钟内便可以知道，但是一个服务器的的网络联通也不能保证服务正常，然后就有了第二套监控方法，每一个从该平台部署的虚拟机上都会有一个服务监控进程，这个进程将在每秒中查看服务的状态，如果服务中断就会试图重新启动服务，并且同时相应的负责人。
- 虚拟服务器快速管理的能力：利用该平台可以快速实现整体环境的开机，关机以及重启，单台或者批量虚拟服务器的开机，关机或者重启，服务的停止，开始和重启，以及修改单台服务器配置等。
- 自动或者手动进行服务代码升级的能力：在这个平台之前如果需要对环境中的某些主机进行代码升级需要一台台进入虚拟机的后台执行相关的命令，利用这个平台可以实现代码自动升级，以及根据需要手动进行批量代码升级的能力，将原来几个小时的工作量压缩到几分钟。

1.3 个人主要工作

本人在整个平台开发中，本人主要负责以下几个方面：平台首页、服务器管理、服务器配置、日志系统、以及用户管理等模块，本人主要负责这几个模块的前台开发，后台实现，对这几个模块的简述如下：

- 平台首页：包括虚拟机网络状态监控，虚拟机资源状态监控，以及虚拟机信息简介。
- 服务器管理：包括环境或者虚拟机的快速构建，服务器的配置修改，以及服务代码的自动以及手动升级。
- 服务器配置：主要包括与公司业务相关的服务的配置。
- 日志系统：操作日志的查询以及备份和删除功能。
- 用户管理：包括平台用户的创建和删除，以及用户权限管理。

1.4 论文组织结构

本论文来源于在公司实习中的项目，基本依据软件开发的规范流程及需求分析、系统架构设计、模块详细设计和实现的顺序来进行论述的。本论文将主要从以下几个方面

对该自动化服务部署平台进行研究：

论文第 1 章：引言，从本论文课题的来源、背景、研究的内容及意义以及个人工作角度对本论文进行了简述，表明了本课题研究的必要性和研究价值。

论文第 2 章：对该平台的需求分析，首先是介绍了该项目简介和的模块划分，以及对项目中各个模块的具体功能的描述；然后分别对各个模块进行具体的需求分析。

论文第 3 章：介绍该平台的系统架构，首先是介绍了系统的总体架构和设计策略，包括系统的基本设计思想、技术支持和建模工具；然后是系统的整体数据库设计；最后介绍了项目具体的开发环境，包括软件环境和硬件环境。

论文第 4 章：介绍该平台的模块设计与实现，首先，在本章开始简单的介绍了系统的整体设计要求，然后分模块详细介绍了项目设计和实现的过程。

论文第 5 章：总结，首先对整个论文的撰写进行自我总结，然后是对于给予本人支持和帮助的指导老师、同学、学校表示感谢，最后列举了参考文献和附录。

2 自动化服务部署平台需求分析

2.1 平台简介

自动化服务部署平台是一个可以提供虚拟机服务部署， 配置以及管理的平台，用户可以借助该平台对生产和测试中的众多虚拟机以及建立在其上的服务进行快速管理，配置和监控。

借助该平台，可以极大的简化研发和测试过程中的环境部署和服务配置的问题，平台可是服务快速部署，服务快速配置，以及服务环境中的虚拟机状态监控以及虚拟机的配置管理，以及日志管和用户管理的功能；

2.2 平台模块划分

该平台可以大致分为以下几个模块：平台首页、虚拟机管理、虚拟机配置、以及日志和用户管理模块。

平台首页主要包括虚拟机网络状态监控，在平台主页可以直观的看到平台所管理的虚拟机的情况，包括虚拟机的总数，种类，以及每种虚拟机的数量，还可以看到每种类别的虚拟机中有哪些是网络正常，哪些网络出现异常，同时下面的列表还允许用户查看每一个虚拟机的名称以及配置等信息，同时查看详情功能还能为用户提供一个虚拟机的CPU 和内存情况利用折线图以便于用户直观的了解虚拟机的资源使用情况以便于及时调整；

虚拟机管理模块具有以下功能：以虚拟机为单位的虚拟机的增删改查，以及虚拟机的开关机和重启操作，以环境（虚拟机组）为单位的的增删改查，以及开关机和重启操作，同时也可以修改虚拟机的网络配置等信息；另外还有虚拟机服务代码的批量升级功能，只需要填入代码的版本号就可以批量升级服务代码，极大的简化了升级流程；

虚拟机配置：主要分为六个服务的配置，DNS、平稳度、网页篡改、爬虫、网页测速、三方通告（这六个服务均为实习所在部门的业务，该平台的配置功能只需要对这六个服务的配置文件继续进行修改，对这六个服务的具体功能不在赘述）；

日志管理：包括日志的增加、删除、查找以及备份功能，因为日志众多，查找方法将提供多种；包括

用户管理：包括用户的增加、删除、查找和修改功能，以及用户权限的管理；平台用户分为系统管理员和普通管理员。

该平台的整体模块划分如下图 2-1 所示：

图 2-1 系统功能模块划分图

2.3 平台首页

2.3.1 功能描述

该模块为平台的首页，为用户展示了该平台所管理的所有虚拟机的信息，首先为用户展示的是该平台所有虚拟机的网络连通性的情况，用户可以根据该部分直观的了解到该平台所管理的虚拟机的总数，以及每种类型的虚拟机的数量，所有网络正常和异常的虚拟机的数量，以及各个类型的虚拟机网络真差和和异常的数量，以便于即时确定网络

异常的虚拟机。

然后还有用表格方式列出平台所有虚拟机的配置情况，以及虚拟机的描述，指定分页操作，同时用户也可以指定每页显示的虚拟机的数量以便于查看，同时可以用查看详情功能查看虚拟机的资源利用图表，图表使用折线图，将默认显示 2 小时内该虚拟机的 CPU 和内存利用情况折线图，以便于用户即时发现某台虚拟机出现资源不足的情况并进行调整。

2.3.2 用例图

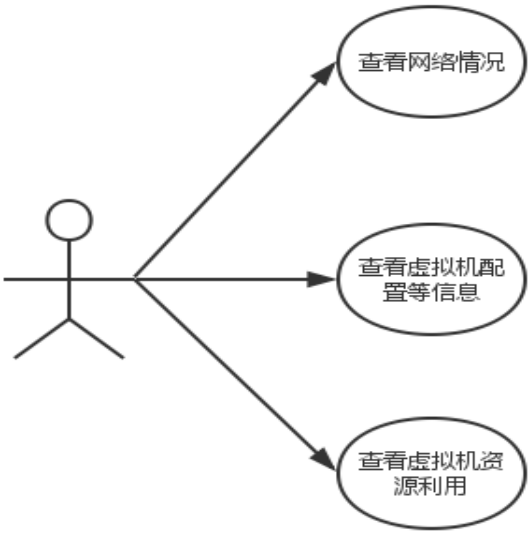

图 2-2 平台首页模块用例图

2.3.3 用例描述

1) 用户进入平台首页以后可以看到所管理所有虚拟机的网络连通性总概况，虚拟机将按照在环境中的角色分组，系统首页将会展示这个角色中的虚拟机一共有几个，有几个网络正常，有几个网络已经中断，正常以绿色表示，中断以红色表示。
2) 在网路连通性概况下面有现在所有虚拟机的列表，分页数量默认 10 个，用户也可以自定义每页虚拟机的个数，在每个虚拟机的后面会有这个虚拟机的配置信息，以及一些用户的备注信息。
3) 在每一个虚拟机的后面有一个查看详细信息的按钮，在点击以后用户可以看到

这个虚拟机的详细信息，包括虚拟机所属的环境，虚拟机配置信息，以及虚拟机过去一个小时以内的 CPU 和内存使用情况折线图。

2.4 虚拟机管理

2.4.1 功能概述

该模块主要是虚拟机的管理相关功能，将以环境为单位将虚拟机分组，可以以环境为单位进行操作，同时也可以细分为以虚拟机为单位的操作。

该模块将以表格方式显示该平台的所有环境列表，首先该模块以环境为单位对虚拟机进行分类，用户可以直观的看到该平台现在管理环境的数量以及每个环境的描述信息，然后用户也可以点击响应的环境查看具体某一个环境内部所包含的虚拟机的信息，同时用户也可以选择增加一个环境或是删除一个环境，并可以批量对所选择的环境同时进行环境的打开，关闭和重启操作，打开环境将打开该环境中所有的虚拟机，关机环境将关闭环境内所有的虚拟机，重启环境将对环境中所有的开机虚拟机进行重启，关机虚拟机打开。

用户可以查看某一个环境下所有虚拟机的详细信息，包括虚拟机的名称，IP，类型，以及现在在该虚拟机上运行的服务的信息。同时可以在每个虚拟机的后面对该虚拟机进行开关机，重启和删除等操作。同时也可以指定一个虚拟机完成虚拟机的网络配置修改和密码修改等操作。

在该模块用户也可以增加一个指定类型的虚拟机，用户可以在增加虚拟机时选择要增加的虚拟机的类型和所属的环境，以及可以指定要增加的虚拟机的网络配置以及密码等信息。之后该平台会完成该虚拟机的创建和初始化等操作，并在虚拟机正式开机完成之后向用户返回创建完成信息并且记录操作到平台日志。如果创建过程中出现问题，比如服务器资源不足或者不能连接服务器导致创建虚拟机失败也会给出提醒并将信息记录到系统的异常日志中。

另外该模块还有对虚拟机进行代码升级的功能，在升级子模块中会显示出该平台的所有虚拟机的名称，IP 和类型等信息，以及该虚拟机中服务的代码的 svn 版本号，用户可以在界面上输入要升级到的 svn 版本号，然后可以对某一台虚拟机进行代码升级或者批量对虚拟机进行代码升级，同时也支持服务代码版本号回退的操作。

2.4.2 用例图

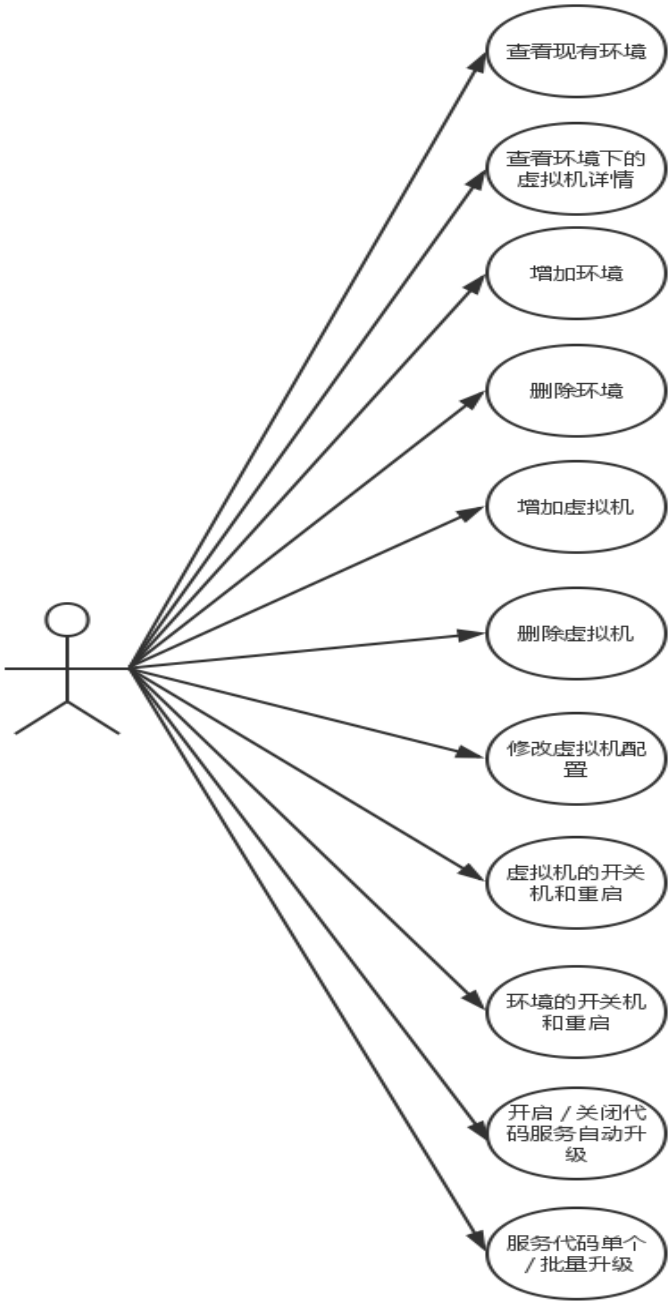

图 2-3 虚拟机管理模块用例图

2.4.3 用例描述

1) 用户进入该模块后会展示系统中现有的环境以及环境的备注信息；

2) 在点开每一个环境以后可以查看该环境下的虚拟机的详情，包括虚拟机的 IP，该虚拟机在环境中的类型，以及现在在该虚拟机上的服务情况；
3) 在页面上有增加环境按钮，用户可以选择增加一个环境；
4) 用户也可以选中环境并将现有的环境删除，环境中的虚拟机也随着一起删除；
5) 用户也可以选择新建一个虚拟机，选中新建虚拟机以后会让用户选择这个将放入哪一个环境，然后输入虚拟机的 IP 等信息便可以新增一个虚拟机；
6) 用户可以用每一个虚拟机后面的删除按钮删除一个虚拟机；
7) 用户可以选中虚拟机以后点击修改配置功能便可以修改虚拟机的网络配置或者密码等信息；
8) 用户可以选中环境以后进行一个或者多个环境的开机，关机和重启操作；
9) 用户可以使用每一个虚拟机后面的选项进行某一个虚拟机的开机，关机和重启操作。
10) 用户可以进入虚拟机升级的子栏目开启或者关闭自动升级功能；
11) 用户可以进入虚拟机升级的子栏目输入版本号以后选择单个或者批量升级虚拟机服务代码。

2.5 虚拟机配置

2.5.1 功能描述

该模块主要是部门内部所运行的各项服务的配置，该平台所支持的服务主要包括以下 DNS、平稳度、网页篡改、爬虫、网页测速、三方通告。

每一个模块的页面上的配置流程大致相同，都为先选择需要配置的环境，然后选择主机对环境中的主机的服务进行检测，最后选择需要配置的主机进行配置。主要不同在于用户选择配置以后后台所修改的配置文件不同。

用户可以在该模块中选择需要进行配置的服务，然后平台会显示现有的环境列表，用户需要对需要配置的环境进行选择，然后可以对该环境进行服务配置。

首先用户需要选择一些类型的虚拟机进行服务检测，Mongo、Mysql、NFS、RabbitMQ 和 Redis，这些选择的虚拟机的信息将会传入下一个流程用于服务配置，之后平台会对选中的虚拟机上的服务进行检测，当检测到这些虚拟机上的服务正常以后用户可以进入下一个流程，服务配置。

在服务配置的部分用户也需要选择需要进行配置的虚拟机，然后平台会利用前一页选择的服务器的 IP 信息对该页选中的服务器进行服务配置，主要是修改指定的配置文件中对应的 IP 地址，然后重启虚拟机上对应的服务便可以完成配置修改工作。

2.5.2 用例图

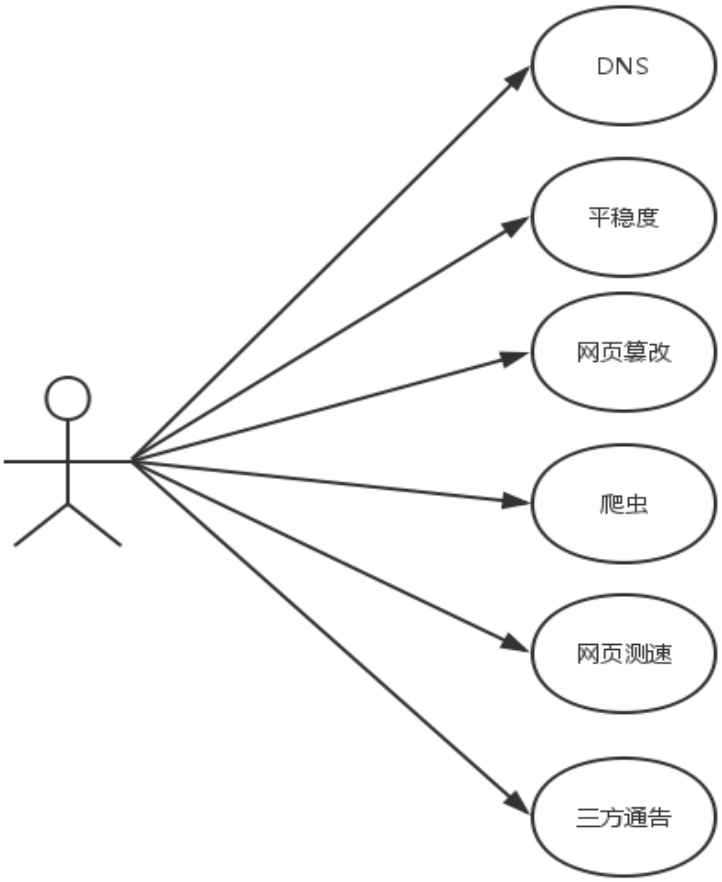

图 2-4 虚拟机配置模块用例图

2.5.3 用例描述

进入虚拟机配置栏目以后会有这六个子模块，然后用户选择其中的一个模块进行服务配置，选择需要配置的服务以后会展示现在平台上的环境列表，用户选择需要配置的环境，之后用户会进入服务检测的页面，在该页面用户需要从环境中选择五个类型的主机进行服务检测，分别为 Mongo、Mysql、NFS、RabbitMQ 和 Redis，在一个环境中可能存在多种一个类型的主机，用户只能选择其中一个，然后这写主机的 IP 等信息将会传入下一个页面用户具体配置，选择完主机以后用户需要点击服务检测，当这些主机上的服务检测结果正常以后才可以进入下一步进行服务配置，否则需要排查这些主机上服务的问题，比如进行重启服务等操作。

当所有主机的服务检测正常以后，用户可以点击下一步进行服务配置，在这一步中

用户需要选择环境中 3 个类型的主机进行配置，分别为 web，daemon 和 Celery，当选择以后用户可以点击配置按钮，界面将返回配置成功或者失败。

2.6 虚拟机监控

2.6.1 功能描述

该模块并不是由平台用户主动触发的操作，而是由该模块提供的监控机制将监控信息提供给该平台相应的用户。

该模块主要用于监控虚拟机中服务的运行状态，从该平台创建的虚拟机在创建之后都会传入服务监控脚本，该脚本启动以后将在每一秒钟对该主机上的所需监控的服务进程进行监控。

当脚本发现该服务器中的服务进程不存在时会对该服务进行指定次数的尝试重启，并将重启的结果以邮件的方式发送给提前指定好的邮箱地址。

不同的虚拟机需要监控的服务不同，该平台会对每一个针对每一个虚拟机的类型提供不同的监控配置文件模版，监控脚本将从这些监控配置文件中读取需要监控的配置信息，并在秒级别的时间内对主机上的进程进行监控，并且即时将结果通知给指定人员。

2.6.2 用例图

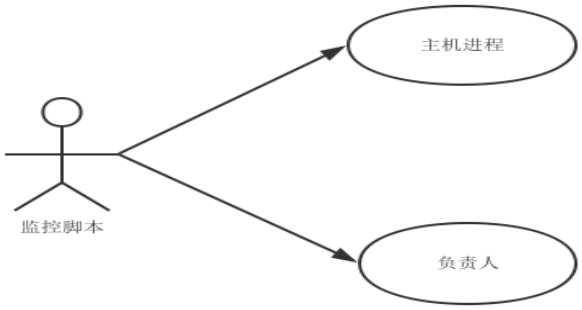

图 2-5 虚拟机监控模块用例图

2.6.3 用例描述

该平台推向各个主机的监控脚本对各台平台所管理的主机进行服务监控，当检测到服务中断以后将会自动重启，并通过邮件方式发送信息给对应负责人即部分平台用户。

2.7 日志系统

2.7.1 功能描述

日志系统主要用于查询在该平台的操作日志，在该平台的操作都会记录到数据库中，用户可以在日志管理模块进行日志的查询，删除和备份功能。

用户可以在该模块查看该平台所有记录的日志，包括该平台的操作信息和操作人的IP 地址以及操作的结果等信息。

因为日志数量较大，日志的查询支持多种方法，查询某个时间段的日志，或者是查询某一个或者是某几个类型的 虚拟机的操作日志，或者查询某一类日志，日志分为以下四种类型操作日志，错误日志，登陆日志和升级日志。

2.7.2 用例图

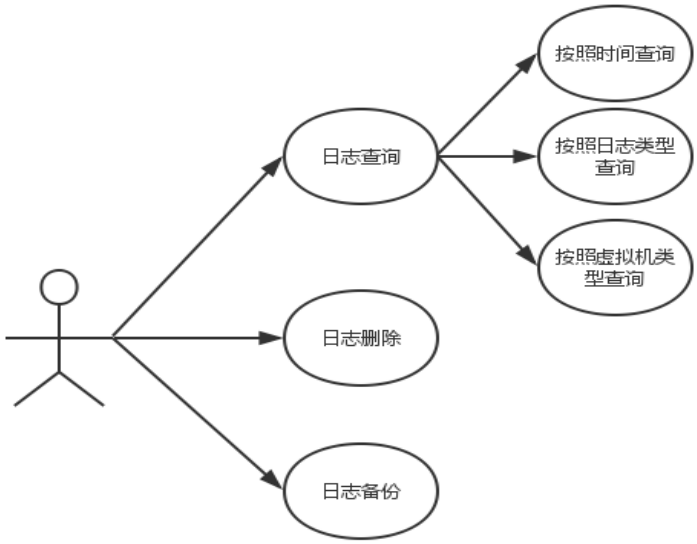

图 2-6 日志管理模块用例图

2.7.3 用例描述

1) 用户可以按照时间段对日志进行查询；
2) 用户可以查询某一个和几个类型的日志；
3) 用户可以查询某一类或者几类的虚拟机类型的操作日志；

4) 用户可以按照上面三个的并集进行查询；
5) 用户可以删除现有日志（只有系统管理员有权限）；
6) 用户可以备份现有日志。

2.8 用户管理

该模块主要是对平台的用户进行管理，由于该平台不对外开放，所以不提供注册功能，新的用户只有通过改模块的用户添加功能实现。

2.8.1 功能描述

该模块主要是平台中的用户管理，用户可以新建用户，删除用户，对用户的信息进行修改，以及管理用户的权限。按照权限可以分为两种用户，系统管理员和普通管理员。

系统管理员具有最高权限，可以进行以下操作：

- 环境操作：可以查看平台现有的环境以及环境的详细信息，增加一个新的环境，删除一个现有的环境，修改环境的备注信息，同时也可以对一个环境进行开机、关机和重启操作；
- 虚拟机操作：查看平台所管理的虚拟机的信息以及虚拟机的配置和资源使用情况，为某一个环境增加虚拟机或是删除虚拟机，修改虚拟机的密码和网络配置，对某台虚拟机进行开机、关机和重启操作；
- 日志操作：查看平台现有的日志，对日志进行查找，删除平台的日志或是对平台的日志进行备份；
- 用户操作：查看平台现有的用户信息，为平台增加用户或是删除一个先存用户，对用户的密码等信息进行修改，修改一个用户的权限，即是系统管理员还是普通管理员。

普通管理员可以进行以下操作。

- 环境操作：可以查看平台现有的环境以及环境的详细信息，同时也可以对一个环境进行开机、关机和重启操作；
- 虚拟机操作：查看平台所管理的虚拟机的信息以及虚拟机的配置和资源使用情况，修改虚拟机的网络配置或是对某台虚拟机进行开机、关机和重启操作；
- 日志操作：查看平台现有的日志，对日志进行查找或是对平台的日志进行备份。
- 用户操作：查看平台现有的用户信息，可以修改自己账户的密码。

2.8.2 用例图

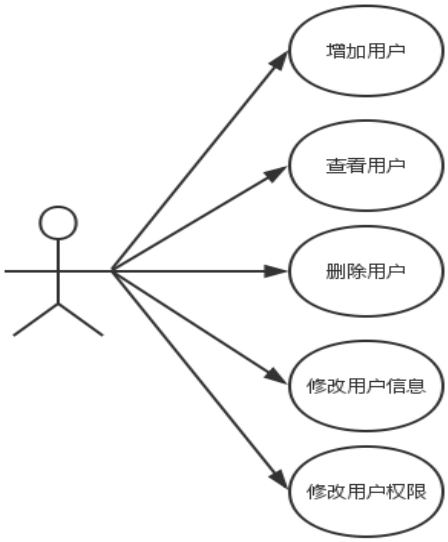

图 2-7 用户管理模块用例图

2.8.3 用例描述

1) 增加用户：系统管理员可以为改平台增加一个用户，点击增加用户以后输入该用户的用户名和密码，并选择改用户的角色点击添加可以增加用户；
2) 删除用户：系统管理员可以删除一个现有的用户；
3) 修改用户信息：系统管理员可以修改用户的用户名和密码等信息；
4) 修改用户权限：系统管理员可以修改任意用户为普通管理员或者系统管理员。

3 自动化服务部署平台总体架构

3.1 整体架构

系统将利用 Django Web framework 进行开发，然后利用 nginx 和 uWSGI 进行部署。

首先浏览器发起一个请求，经由 DNS 解析以后请求会达到我们的服务器，达到指定端口，Request Middlewares 中间件接受到 request 以后对 request 做一些预处理或者直接传递 response 请求。然后 URLConf 通过 url.py 的文件和请求的 URL 可以找到指定的 View。

View Middlewares 会被访问，它可以对 request 做一些处理或者直接返回一个 response 给客户端。

然后 View 中的响应函数会执行，根据传入的请求 View 函数会访问数据库进行数据存取，或者访问 VMware vSphere 执行相应的行为，并将结果返回给客户端。

settings 里设置了 WSGI application，也就是 WSGIHandler 类。

启动后，会生成一个 WSGIHandler 实例，以后每次请求响应都用这个实例。因此，只有第一次请求时会调用 load_middleware，以后不再调用了，按着 settings 配置顺序加载 Middleware 类。其中，request_middleware、view_middleware 是顺序，template_response_middleware、response_middleware、exception_middleware 是逆序。

然后，WSGIHandler 会根据入参 environ 构造 WSGIRequest 对象，更有用的是它的父类 HttpRequest。如果是 POST 请求，且 CONTENT_TYPE 含‘multipart/form-data’，还会加载 settings 配置的 uploadhandler，并解析。

在一些请求的情况下，Views 可以使用一个特殊的 Context，Context 被传给 Template 可以生成指定的页面，展示一些指定的信息，然后 Template 使用 Filters 和 Tags 去渲染输出，这些输出会被传回到 View 函数，然后 View 函数会通过 HTTPResponse 将这个 Response 返回到 Response Middlewares 进而返回到客户端。

系统前端将使用 html 和 CSS 提供一个友好的 web 界面，也将采用 JavaScript 和 ajax 技术以实现平台中部分信息的异步加载功能。

系统后端将采用 python 进行开发，一个请求经过 Django web 框架的处理之后会将请求传入响应的 python 文件中指定的函数，当函数接受到前端请求后会提取出请求中的信息并对请求进行处理，提供后台主要和三个对象进行交互，一个是 MySQL 数据库，存储系统的环境和虚拟机的信息，以及平台的日志和用户；一个是 vsphere 服务器，主要是和虚拟机管理相关操作；另一个是具体的虚拟服务器，要是是完成虚拟机配置的相关操作。

当前端发送的请求是数据查询类时，比如查看现在平台的环境列表和虚拟机列表，

为了保证效率后台会直接从数据库中取得这些信息进行展示。

当前端发送的是增加、删除或者是重启虚拟机相关的虚拟机管理操作时，后台会同时和 vsphere 服务器和 MySQL 数据库进行交互完成虚拟机信息的查找和虚拟机的增加、删除或是重启等相关的操作，最后完成数据库数据更新。

当前端发送的是对虚拟机网络或是服务进行配置的操作时，后台会与 MySQL 数据库和虚拟服务器继续交互，首先取得虚拟机的信息然后进入虚拟机后端对虚拟机配置进行修改，最后会更新数据库中相关信息。

整个过程可以用以下的流程图来表示：

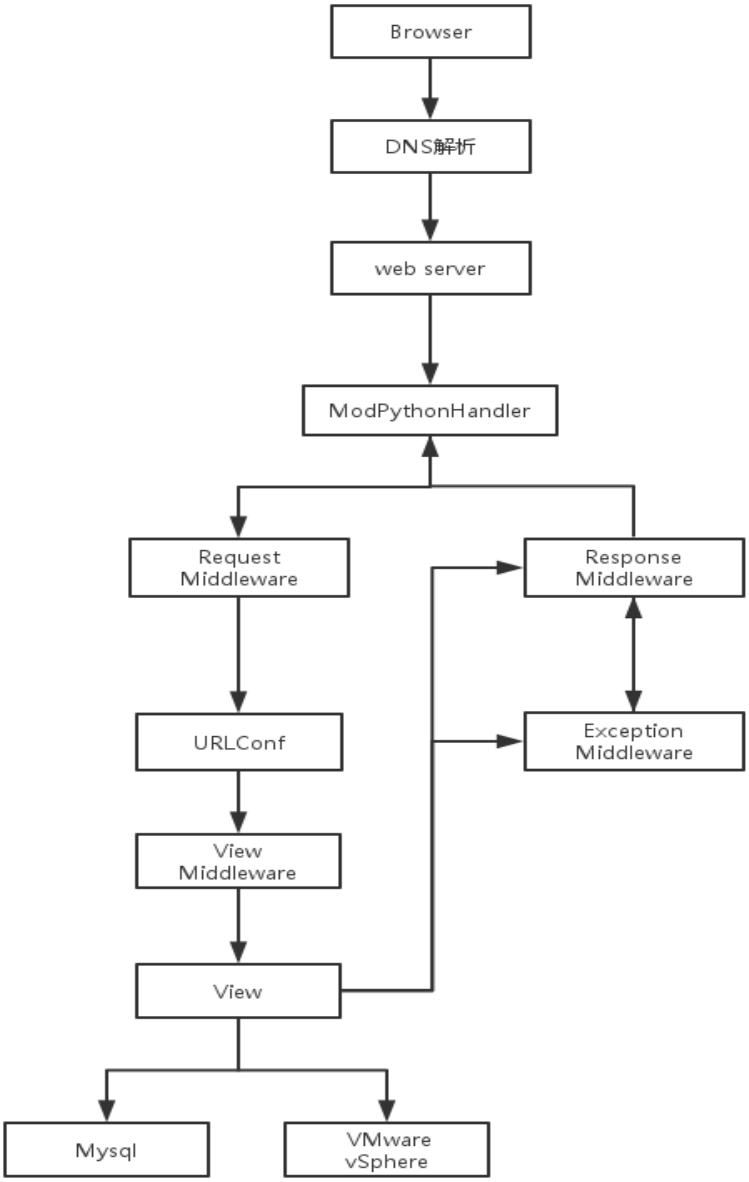

图 3-1 系统架构图

3.2 开发平台

操作系统：ubuntu
Python 版本：2.7.10
Web 框架：Django
数据库系统：MySQL
版本控制工具：svn
编译平台：ubuntu
服务器：Nginx

3.3 应用技术

改自动化服务部署平台将使用 Django web 框架进行开发，前台将使用 HTML 和 CSS 展示，同时结合 JavaScript 和 Ajax 实现异步加载等功能，在后台将使用 MySQL 作为存储数据库，同时后台根据需要也会与 VMware vSphere 服务器进行交互来操作平台之上的虚拟化服务器。改平台将搭建在 ubuntu 系统之上，同时用 Nginx 和 uWSGI 进行服务器的搭建。以下为关键技术介绍：

3.3.1 服务器虚拟化

为了使物理服务器达到更高的可用性、可维护性及可扩展性，金融科技部门运用虚拟化软件将服务器物理资源抽象成逻辑资源，变物理服务器为几台甚至几十台相互隔离的虚拟服务器[1]。

服务器虚拟化不仅向网络管理员提出了整合和等待时间等挑战．它还使进入每一个物理服务器的网络基础结构肩负了更沉重的负担。在一家公司减少物理服务器数量的时候．服务器人员却成比例地增加[2]。

服务器的虚拟化开辟了一系列的自主数据中心管理的新的可能性，通过可以被利用，以控制和监控虚拟机内运行的任务的新的自动化机制的可用性。这不仅提供新的和更灵活的控制使用的管理控制台操作，也更加强大和灵活自主控制，通过维持该系统所需的状态改变工作量和需求面对管理软件[3]。

3.3.2 Django

Django 是 Python 的一个优秀的开源 Web 开发框架，在 Web 开发方面具有简洁、清

晰、高效、安全的优点，特别适合快速构建各类 Web 应用。通过一个留言板的开发实例，论述了基于 Django 快速开发 Web 应用的基本原理和过程[4]。

随着 Web 技术的发展和软件工程的日益成熟,人们逐渐提出敏捷开发的要求。但是传统的 Web 框架像 Struts,Spring,.NET 等由于其复杂性而很难满足人们的要求,于是逐渐出现一些敏捷性 Web 框架。Django 是使用 Python 开发的优秀 Web 框架,作为一款开源的敏捷开发框架,受到越来越多的人的关注[5]。 它采用了 MVC 的软件设计模式，即模型 Model，视图 View 和控制器 Controller。整个系统可以分为三个层次：

1)最上层，是直接面向最终用户的"视图层"（View）。它是提供给用户的操作界面，是用户和整个系统交互的入口和系统为用户展现结果的空间。

2)最底层，是核心的"数据层"（Model），也就是程序需要操作的数据或信息，该层可以直接与数据库交互进行数据的存取。

4) 中间层，就是"控制层"（Controller），它可以接受用户从"视图层"输入的请求，然后得到"数据层"中的数据，然后对这些数据进行处理，然后再将结果返回 View 层以供用户查看。

需要注意的是，Django 将 MVC 中的 View 视图层进一步分解为 Django 视图 和 Django 模板两个部分，分别决定 “展现哪些数据” 和 “怎么展现这些数据”，使得 Django 的模板可以根据需要随时替换，而不仅仅限制于内置的模板。

而 MVC 控制器部分，由 Django 框架的 URLconf 来实现。URLconf 机制是使用正则表达式匹配 URL，然后调用后台中合适的消息处理函数。

3.3.3 Fabric

Fabric 是一个 Python 库，只要目标机器支持 ssh 访问，就可以借助 fabric 来进行远程操作（如在 host1 上对 host2 远程运行 shell 命令），显然，由于 fabric 是个 Python package，故其它 Python package 都可以被 import 到 fabric 特有的 fabfile.py 脚本中，这使得 fabric 如虎添翼，在功能的丰富程度和运维脚本的可维护性上，远远超过用 shell 实现的自动化部署脚本，更不要说与纯手工敲命令的上线方式相比所体现出的巨大优势了。

3.3.4 vSphere Management SDK+ PySphere

vSphere 是 VMware 公司推出一套服务器虚拟化解决方案，虚拟化打破了物理硬件与操作系统及在其上运行的应用程序之间的硬性连接。操作系统和应用程序在虚拟机中实现虚拟化之后，便不再因位于单台物理计算机中而受到种种束缚。物理元素（如交换机和存储器）的虚拟等效于在可跨越整个企业的虚拟基础架构内运行。vSphere5 中的核

心组件为 VMware ESXi 5.0.0（取代原 ESX）, ESXi 是一款可以独立安装和运行在裸机上的系统，因此与他我们以往见过的 VMware Workstation 软件不同的是它不再依存于宿主操作系统之上。在 ESXi 安装好以后，我们可以通过 vSphere Client 远程连接控制，在 ESXi 服务器上创建多个 VM（虚拟机），在为这些虚拟机安装好 Linux /Windows Server 系统使之成为能提供各种网络应用服务的虚拟服务器，ESXi 也是从内核级支持硬件虚拟化，运行于其中的虚拟服务器在性能与稳定性上不亚于普通的硬件服务器，而且更易于管理维护。

VMware 的 ESXi 的是开始使用虚拟化的最简单的方法 - 它是免费的。它使管理员能够整合他们的应用程序到更少的服务器，并开始通过减少硬件，电力，冷却存钱，管理成本[6]。

大数据处理时代的来临，使得虚拟服务器集群在为企业在电力和能源成本上带来巨大的经济与社会效益的同时，也在更好地利用现有资源、提供更多可管理的应用，改进数据中心的灾难恢复方式和提高稳定性的前提下降低维护的复杂性等方面给予了巨大的提升空间．而且，寻求经济、高效的 NoSQL 存储和分布式存储用以支持虚拟服务器集群的海量数据存储方案也渐渐成为应对爆炸式数据增长的研究方向与应用主流．VMware vSphere 环境，为企业级、高可用的虚拟服务器集群的构建提供了一种较为经济可行的解决方案[7].

vSphere Management SDK 是 VMWare 为 vSphere 提供的一套公共 API 接口，利用这套 API 可以完成基础的虚拟化服务器的管理，通常需要通过验证连接到 vSphere 虚拟化服务器，然后通过一些命令可以完成虚拟机相关操作。

需要注意的是在该平台中的虚拟机新建部分，由于我们需要新建出来的虚拟机可以立即向外提供服务，如果是利用 vSphere Management SDK 中的新建接口新建出来的是一个虚拟机裸机，不能向外提供任何服务，所以该平台设计中将转用虚拟机克隆的方法来完成虚拟机的新建，利用该平台新建指定的类型的虚拟机，该平台将会从指定的该类型的虚拟机模版中克隆出来该类型的虚拟机，这样克隆出来的虚拟机只需要进行相应的简单配置即可对外提供服务。

vSphere Management SDK 为我们提供了虚拟机克隆的接口，但是该接口只支持付费的 VMware vCenter，并不支持 vSphere，所以需要另寻他法完成克隆虚拟机的功能。

现在该平台先利用 vSphere Management SDK 提供的接口先新建一个和被克隆达到虚拟机配置相同的空白虚拟机，然后通过 ssh 服务进入 vSphere 服务器后台删除掉指定的该新建虚拟机的磁盘文件，之后可以利用 vmkfstools 克隆指定的虚拟机的磁盘文件，之后。

vmkfstools 在 VMware vSphere(ESXi) Server 中是一个文件系统管理工具。可用来创建虚拟硬盘，转换虚拟硬盘格式。他支持创建在一磁盘分区上创建 VMFS，和管理保

存在 VMFS 上的文件(如虚拟磁盘)。

克隆之后根据需要可以对指向虚拟磁盘文件的配置文件进行修改，一般如果新建的虚拟磁盘文件和之前的相同则不需要继续修改，然后便可以对虚拟机进行正常开机。但是这时的虚拟机是从之前的模版文件中克隆出来的，网络信息还未进行任何配置，一般无法联网的，也就无法对外提供服务或者是通过 ssh 服务登陆到该虚拟机。这是我们需要 PySphere 和 VMware Tools。

PySphere 是一种 python 与 vSphere Web SDK 接口的 API 函数库，对虚拟机的操作主要以脚本或者编程方式实现，方便程序员编写程序自动控制虚拟机。本文利用 pysphere 实现虚拟机电源管理、镜像恢复、传送文件、启动程序等操作[8]。

VMware Tools 是 VMware Workstation 虚拟机软件的增强工具包，是 VMware 提供的增强虚拟显卡和硬盘性能、以及同步虚拟机与主机文件的驱动程序。

将 PySphere 和 VMware Tools 结合，我们已经事先将所有被克隆的虚拟机模板都内置了 VMware Tools，我们就可以完成与新建出来的虚拟机进行通讯，在每个虚拟机克隆完成并且开始以后，平台都会利用 PySphere 向虚拟机发送指定的命令完成虚拟机的密码配置和网络配置等操作，并会向虚拟机传入服务监控脚本，然后对虚拟机进行重启，并启动服务监控脚本进程，至此一个虚拟机正式新建完成 。

3.3.5 Nginx+uWSGI

Nginx 是一个高性能的 HTTP 和 反向代理 服务器，也是一个 IMAP/POP3/SMTP 代理服务器 。 Nginx 是由 Igor Sysoev 为俄罗斯访问量第二的 Rambler.ru 站点开发的，它已经在该站点运行超过四年多了。Igor 将源代码以类 BSD 许可证的形式发布。自 Nginx 发布四年来，Nginx 已经因为它的稳定性、丰富的功能集、 示例配置文件和低系统资源的消耗而闻名了。目前国内各大门户网站已经部署了 Nginx，如新浪、网易、腾讯等；国内几个重要的视频分享网站也部署了 Nginx，如六房间、酷 6 等。 新近发现 Nginx 技术在国内日趋火热，越来越多的网站开始部署 Nginx。

uWSGI 是一个快速、自我修复、开发人员/系统管理员友好的容器服务器，代码完全用 C 编写。uWSGI 以客户端-服务器模型运作。你的 Web 服务器（例如 nginx、Apache）与一个 django-uwsgi 进程交互来服务动态的内容。它实现了 WSGI 协议、uwsgi、http 等协议。Nginx 中 HttpUwsgiModule 的作用是与 uWSGI 服务器进行交换。

3.3.6 Linux 进程间通讯

linux 下的进程通信手段基本上是从 Unix 平台上的进程通信手段继承而来的。而对

Unix 发展做出重大贡献的两大主力 AT&T 的贝尔实验室及 BSD（加州大学伯克利分校的伯克利软件发布中心）在进程间通信方面的侧重点有所不同。前者对 Unix 早期的进程间通信手段进行了系统的改进和扩充，形成了“system V IPC”，通信进程局限在单个计算机内；后者则跳过了该限制，形成了基于套接口（socket）的进程间通信机制。

其中，最初 Unix IPC 包括：管道、FIFO、信号；时于多用户、多任务的操作系统，进程间的通信(Inter-Process Communication，IPC)是非常重要的，它是使整个系统得以有条不紊工作的基础。Linux 操作系统提供了多种 IPC 机制，如信号、管道、信号量、消息队列、共享内存和套接字等。其中信号是系统必备的一种 IPC 机制，是内核不可分割的一部分，而其它的几种机制则是可选的[9]。

4 自动化服务部署平台功能模块设计与实现

本人主要负责设计和实现自动化服务部署平台的首页模块，虚拟机管理模块，虚拟机配置模块，日志模块以及用户管理模块，下面将逐个介绍各个模块的设计和实现的思路。

4.1 平台首页模块

4.1.1 设计描述

在该模块中，平台用户点击首页栏目，系统将会查询并展示当前平台所管理的所有虚拟机的网络连通情况，并显示出网络连接概况。同时也会将虚拟机进行分类，展示出每一类中虚拟机的总数和网络出现问题的虚拟机的数量。

同时在网络连接概况下方将会将平台所有管理的每个虚拟机进行信息分页展示，默认每一页将有是个虚拟机，用户可以用展示栏目上方或者下方的按钮到上一页或者下一页，同时也可以到达指定的某一页查看虚拟机的情况。

在虚拟机信息展示栏目中，将会展示虚拟机的网络状态，如果网络正常将会用绿色按钮表示，如果网络出现异常将会用红色按钮表示，然后会展示虚拟机的名称、该虚拟机所对应的 IP、该虚拟机的类型、虚拟机系统的版本以及改虚拟机的配置信息。

在每一个虚拟机的后面都会有一个查看详细信息的按钮，点击进去以后会显示该虚拟机的详细信息，包括虚拟机名称，IP，配置，类型，以及该虚拟机在过去 2 个小时内（时间段用户可选）CPU，内存的使用情况折线图。同时用户也可以修改折线图展示的时间段，让用户看到更长时间内虚拟机的资源利用情况。

同时在虚拟机展示栏目的上方有搜索按钮，用户可以对虚拟机的名称进行模糊搜索，查看指定的虚拟机的信息。之后下半部分虚拟机信息展示栏中的虚拟机信息会进行更新，只展示服务用户搜索的关键字的虚拟机，并展示出搜索出的的虚拟机的网络连通情况，虚拟机的名称、该虚拟机所对应的 IP、该虚拟机的类型、虚拟机系统的版本以及改虚拟机的配置信息。

我们也可以通过点击某一个类别的虚拟机进而查看某一个类别的虚拟机的详细信息，点击虚拟机网络情况总概部分的每一个类型的虚拟机都会引起下面虚拟机信息展示栏目的更新，用户可以通过这个方法查看某一个类别的虚拟机的详细信息，包括这个类别的虚拟机的网络情况，还有虚拟机的名称、该虚拟机所对应的 IP、该虚拟机的类型、虚拟机系统的版本以及改虚拟机的配置信息。

4.1.2 流程图

平台首页流程图如下所示：

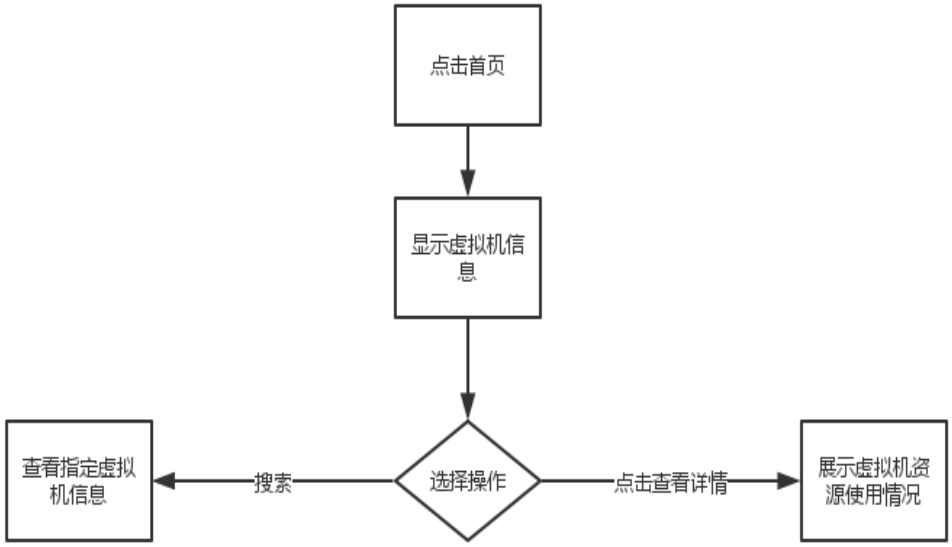

图 4-1 首页模块流程图

流程图说明：首先用户登陆以后点击平台首页栏目会进入平台首页，平台首页会首先展示当前平台所管理的虚拟机的总括信息，这些信息包括虚拟机的总数和每种类型的虚拟机总数以及每种类型中虚拟机的网络正常的个数和网络异常和个数。

之后会有每个虚拟机的详细信息，这些信息包括该虚拟机的网络连通情况，该虚拟机的类型、虚拟机名称以及虚拟机 IP 和虚拟机配置等信息，之后用户可以进行两种操作。

用户可以用搜索框搜索虚拟机的名称以找到某台指定的虚拟机，平台会展示该台虚拟机的信息，包括该虚拟机的网络情况、该虚拟机的类型、该虚拟机的 IP 和虚拟机配置情况（CPU，内存，磁盘等）。

用户可以点击每一台虚拟机信息后面的查看详情按钮查看某一台虚拟机的详细信息，包括该虚拟机的名称和 IP，以及改虚拟机的硬件配置信息，之后的页面除了会列出虚拟机的基本信息之外还会默认列出该虚拟机 2 小时以内该虚拟机的 CPU 和内存利用情况折线图。当用户的鼠标到达折线图中指定的点以后还会有虚拟机在该点 CPU 和内存使用情况的具体数值信息。

4.1.3 时序图

平台首页的时序图如下所示：

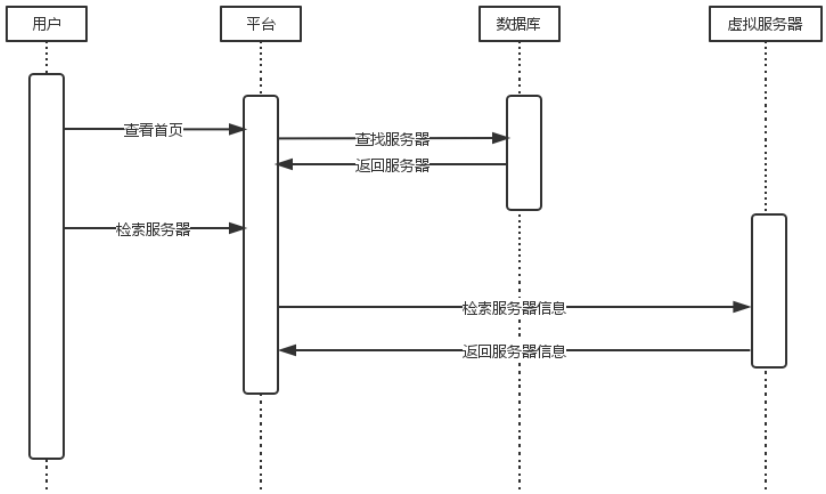

图 4-2 平台首页时序图

时序图说明：用户在进入首页之后平台后台会收到查看首页的请求，之后平台后台会查询 MySQL 数据库，找出平台中现在管理的所有虚拟机的信息，之后从这些信息中取出虚拟机的 IP 然后用 fping 命令指定一次 ping 操作（fping 可以一次批量对大量 IP 执行 ping 操作，比直接逐个对虚拟机进行网络测试节省了大量的时间），获得 fping 结果之后对结果进行解析并统计其中哪些虚拟机的网络无法联通，哪些网络正常，之后对这些信息进行包装并填入页面，返回给浏览器进行展示。由于平台首页每 3 秒进行一次刷新，所以用户处在平台首页时会一直向平台服务器发送该请求。除了第一次请求会返回整个页面，后来每次刷新请求会用 ajax 完成，每次只需要返回虚拟机的统计结果信息即可。

用户在 web 界面可以进行搜索操作；有两种搜索方式，一种时直接点击某个类型的虚拟机的总概信息框，之后会发送到平台服务器一个查看指定类型的虚拟机信息的请求，平台服务器收到这个请求之后会找出请求中的虚拟机类型，然后从 MySQL 数据库中查询这种类型的所有虚拟机的信息，然后再用 fping 找出这些虚拟机的网络连通情况，然后对返回的结果进行处理之后再包装返回到界面中。另一种搜索方式时直接用虚拟机信息展示部分上方的搜索框进行搜索，用户可以出入虚拟机的名字之后点击搜索，之后平台服务器会收到这个搜索请求，再从这个请求中找出要搜索的虚拟机的名称，之后再从数据库中找到虚拟机名称符合这个名称的虚拟机，然后用 fping 再去检查这些虚拟机的网络连通情况，之后分析结果包装并返回到页面中进行展示。

4.1.4 系统实现图

下图为平台首页部分实现效果图：

图 4-3 平台首页总概情况实现效果图

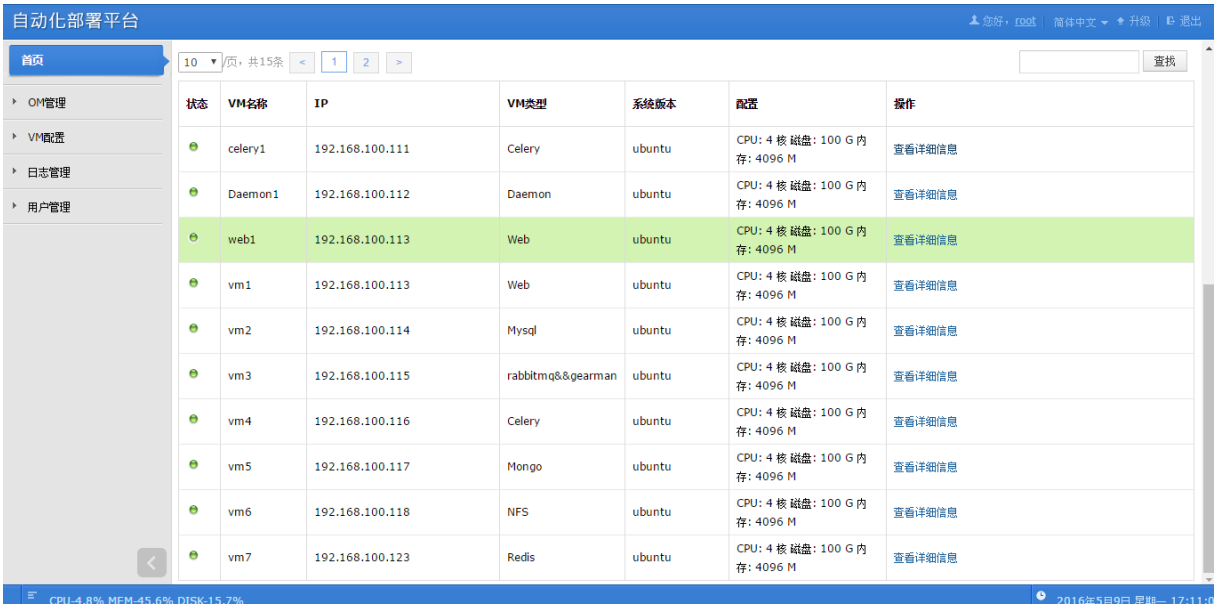

图 4-4 平台首页虚拟机信息实现效果图

4.2 虚拟机管理模块

4.2.1 设计描述

该模块有两方面内容：1)VM 管理 2)VM 升级

用户点击左边的ＯＭ管理之后会出现两个子栏目，ＶＭ管理和ＶＭ升级，用户将默认在ＶＭ管理子栏目，该栏目将显示出平台现有的环境列表，在每一个环境列表前都会有选择框，用户可以选择多个环境进行环境的打开环境，关闭环境，重启环境，以及删除环境的操作。

在环境列表的每一行，将显示当前环境的名称，环境中拥有的虚拟机的数量，以及对当前该环境的方法描述，同时在环境名称前有一个按钮可以用户查看该环境中虚拟机的详细信息。

环境列表默认每一页１０个环境，用户可用上方或者下方的按钮进入上一页或者下一个，或是指定的某一页。

用户可以点击页面的添加环境按钮，点击之后会出现添加环境的信息输入框，包括环境的名称以及对于该环境的描述，用户点击添加以后系统将会添加该环境并将该环境显示与环境列表当中，并将该信息记录到日志中。

用户可以点击页面的添加ＶＭ按钮添加一个虚拟机，点击之后会出现添加ＶＭ的信息输入框，要输入的信息包括，要添加的虚拟机的主机名，虚拟机的ＩＰ地址，虚拟机的子网掩码，默认网关，首先和备用ＤＮＳ服务器地址，已知虚拟机的类型和虚拟机所属的环境等信息。点击添加后平台将会为指定环境添加指定类型的虚拟机，添加成功以后界面将返回添加成功的提示，失败以后也将返回响应的提示，并将该记录计入平台日志当中，虚拟机添加成功以后该虚拟机将出现在对应的环境之中。

用户可以点击某一个环境前的展开按钮，点击之后将会以表格方式显示该环境之中的所有虚拟机列表，显示的信息包括虚拟机的名称，ＩＰ，虚拟机的类型，以及正在该虚拟机中运行的服务，同时在每一个虚拟机的最后一栏是可以对该虚拟机进行的操作，包括重启该虚拟机，打开或者关闭虚拟机，以及删除该虚拟机。平台会将相应操作的信息记录到平台日志当中。

用户也可以对某一个虚拟机进行修改密码，用户点击操作中的修改密码按钮，平台将会出现一个悬浮框让用户出入要修改的虚拟机密码的名称，以及新的密码和密码确认，当密码和密码确认以后平台将会为改虚拟机修改密码，并将修改的结果在界面中进行展示。同样将操作记录到日志当中。

用户也可以对某一个虚拟机进行修改网络配置的功能，用户点击操作中的修改网络配置按钮以后，将会出现修改网络配置的悬浮框，用户需要输入要修改配置的虚拟机的名称，以及新的ＩＰ地址，子网掩码，默认网关，首先和备用ＤＮＳ服务器地址的信息，然后点击修改按钮平台将对指定的虚拟机的网络配置进行修改，并将修改的结果返回到界面之上，并将该操作记录到日志当中。

用户可以点击ＶＭ升级子栏目进入ＶＭ的升级部分，该部分将默认用表格方式显示

平台中的所有虚拟机以及其中的代码版本信息，每页１０个虚拟机的信息，用户可以修改每页显示的虚拟机的个数，显示的信息包括，虚拟机的名称，虚拟机的 IP，虚拟机的类型，以及虚拟机的当前 svn 版本号，之后会有输入框用户可以输入要更新的 svn 版本号信息，之后会有更新和回退的按钮，用户可以点击相应的按钮完成对某一个虚拟机的版本更新和回退操作。

在每一个虚拟机的前面，会有一个复选框，用户可以同时选择多个虚拟机，然后点击界面上方的批量升级按钮，同时对选中的多个虚拟机进行批量的代码升级操作，平台也会记录相应的日志信息。

用户也可以在该ＶＭ升级栏目中打开代码自动升级，打开代码自动升级后每当 svn 服务器中的代码更新以后会向该平台发送请求并传入响应的升级信息，之后平台将会对指定的虚拟机进行代码升级操作。

4.2.2 流程图

该模块的流程图如下所示：

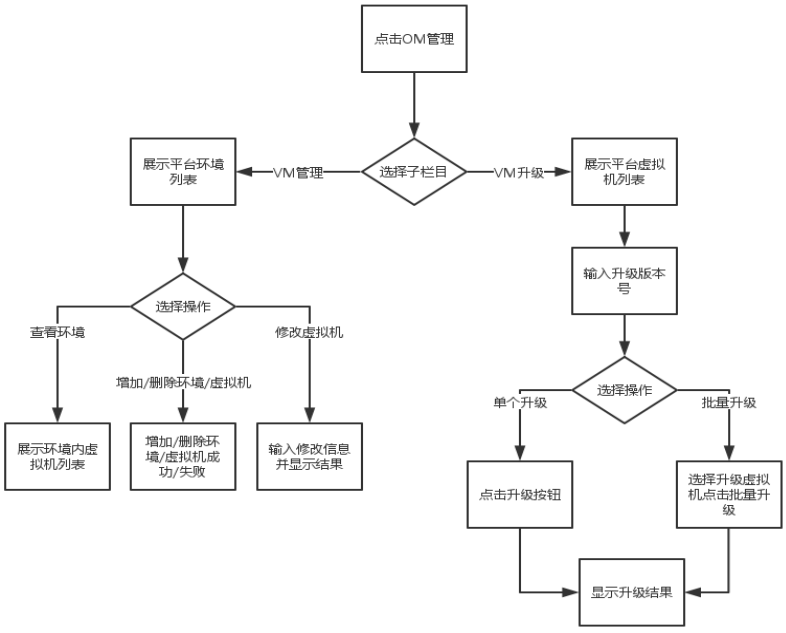

图 4-5 虚拟机管理模块流程图

流程图说明：用户点击左边导航栏的 OM 管理之后会进入虚拟机的管理模块，该模块有两个子栏目，一个是 VM 管理，一个是 VM 升级。

用户可以点击 VM 管理子栏目进入 VM 管理子模块，之后平台会展示该平台所管理的环境列表的信息，包括该环境的名称，该环境中拥有的虚拟机的个数，以及该环境的

描述信息，之后用户可以选择以下操作：显示某个环境中虚拟机的信息，增加删除环境，增加删除或者修改虚拟机配置。

用户可以点击每个环境前的展开按钮查看这个环境下的虚拟机的详细信息，点击展开按钮之后平台会为用户展示该环境下所有虚拟机的名称，IP，以及虚拟机的类型和正在该虚拟机上运行的服务，之后用户还可以用每个虚拟机后面的按钮对该虚拟机进行重启，开关机或者是删除的操作。

用户可以点击信息展示栏上方的增加环境按钮为平台添加一个环境，需要输入该环境的名称以及描述信息，点击添加即可添加完成，用户也可以点击上方的添加虚拟机按钮添加一个虚拟机，之后需要输入添加到的环境，虚拟机的类型以及虚拟机的网络配置等信息，点击添加就可添加虚拟机到指定的环境，或者是批量选择环境然后利用操作栏下拉中的删除按钮批量删除环境。

用户也可以选择操作中的修改密码和修改网络配置的功能，点击修改密码之后需要输入要修改的虚拟机的密码的名称，以及虚拟机的新密码和确认密码，之后平台检查以后可以对虚拟机的密码进行修改，点击修改网络配置之后需要输入要修改的虚拟机的名称，以及新的网络配置信息，包括虚拟机的 IP 地址，子网掩码，网关以及 DNS 地址等信息，然后平台会对该虚拟机的网络配置进行修改。

4.2.3 时序图

虚拟机管理模块的增加/删除环境的时序图如下所示：

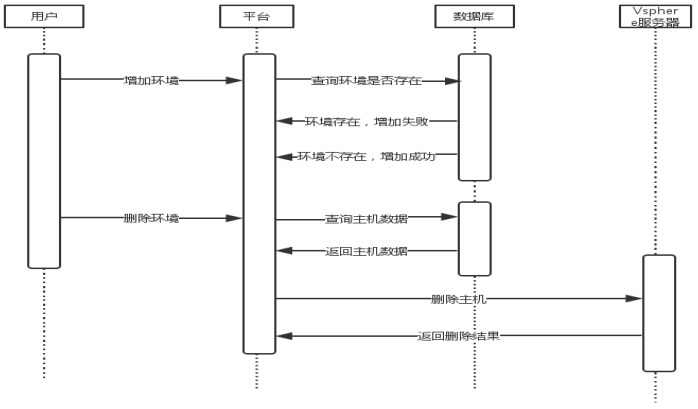

图 4-6 增加和删除环境时序图

增加/删除环境部分时序图说明：用户点击增加环境并添加指定的信息之后点击添加会触发一个增加环境的请求，之后平台收到该请求之后解析出该环境的名称和描述信息，之后将这些信息存储到 MySQL 的环境表之中，完成环境的添加，然后向界面返回添加的结果。

用户选择环境之后点击操作中的删除按钮会触发删除环境的请求，平台服务器收到请求之后解析出请求中的要删除的环境的名称，有时候要删除的环境有多个，然后平台会循环从 MySQL 数据库中找出每个环境中的虚拟机的信息，然后连接到 Vsphere 服务器，删除掉这些虚拟机并获得删除的结果，待所有虚拟机删除完成之后再删除掉数据库中的这些虚拟机的信息和环境的信息，再向前端返回删除的结果。

以下为增加/删除虚拟机的时序图：

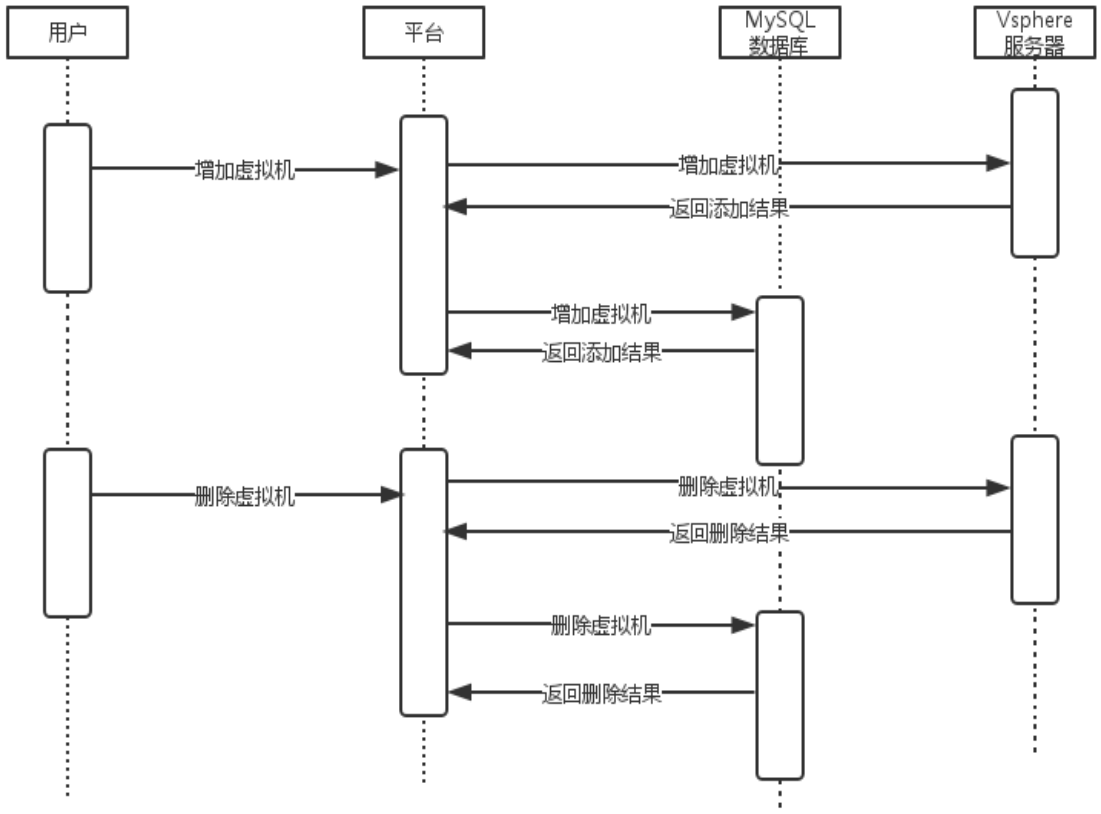

图 4-7 增加/删除虚拟机时序图

增加/删除虚拟机时序图说明：用户点击增加虚拟机并填入相关的信息之后会产生一个添加虚拟机的请求，平台的服务器接受到请求之后解析请求中的参数，找出该虚拟机的名称，类型以及网络配置等信息，之后连接 Vsphere 服务器进行虚拟机的添加，添加完以后开机修改虚拟机的网络配置等信息之后进行重启，然后返回操作的结果，如果操作成功然后再将虚拟机的信息添加到 MySQL 数据库中，然后平台显示虚拟机添加完成。

用户点击虚拟机删除按钮之后会触发一个删除虚拟机的请求，平台的服务器接受到请求以后解析出要删除的虚拟机的名称，然后连接 Vsphere 服务器删除掉这个虚拟机并得到删除的结果，如果删除成功以后再进入 MySQL 数据库中删除掉这个虚拟机的记录，然后平台显示删除虚拟机完成。

以下为虚拟机管理模块信息展示部分时序图：

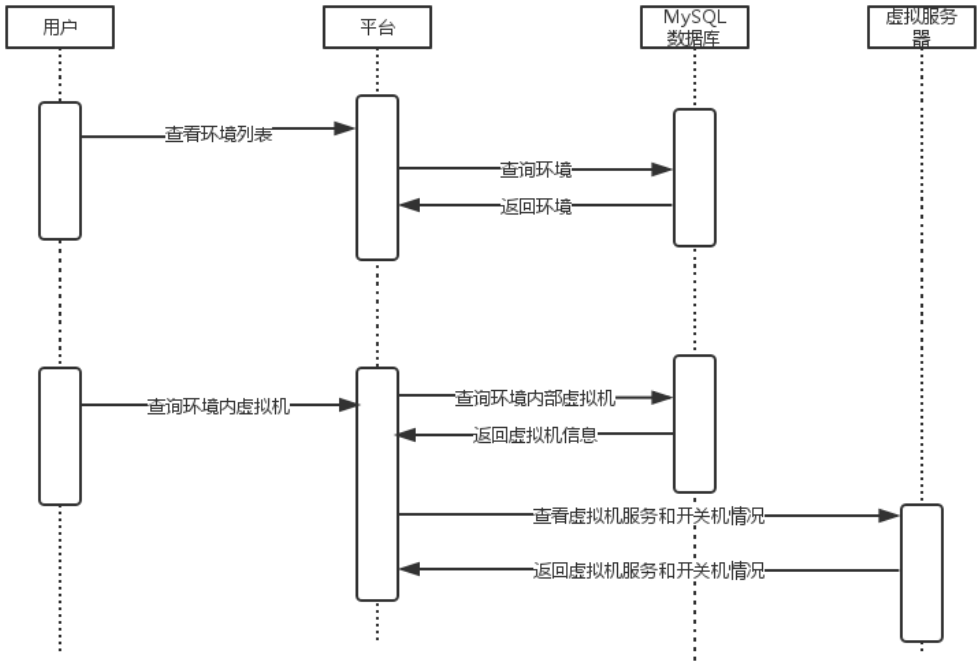

图 4-8 虚拟机管理信息展示时序图

虚拟机管理信息展示时序图说明：用户进入平台的 VM 管理子栏目之后会触发一个查看平台内环境列表的请求，平台服务器接到请求以后会查询 MySQL 数据库中该平台上所有的环境，然后将环境信息返回到平台的前端进行展示。

用户点击每一个环境前的展开按钮以后会触发一个查询环境内虚拟机的请求，平台服务器接受到请求以后解析出请求中环境的名称，然后从 MySQL 数据库中查出该环境中所有的虚拟机的信息，之后连接到 VSphere 再查看这些虚拟机的开关机情况，然后返回这些信息到平台进行展示。

如果在查看虚拟机的情况的时候就查看每一个主机上面运行的服务会造成请求处理的时间过长，造成界面会已知等待这服务器返回请求从未无法进行信息展示，所以运行服务显示的问题用以下方法解决。

在信息返回之后界面会用 ajax 技术发送请求查看虚拟机之上的服务情况，平台接受到请求之后会解析这个请求，之后平台还会从这些信息中得到虚拟服务器的 IP 然后进入虚拟服务器后台查看在该虚拟机之上运行的服务情况，最后将这些服务情况进行包装，然后再将这些信息返回到平台中进行展示，再在每一个虚拟机的运行服务情况那列展示服务查询的结果。

以下为环境的开关机和重启的时序图：

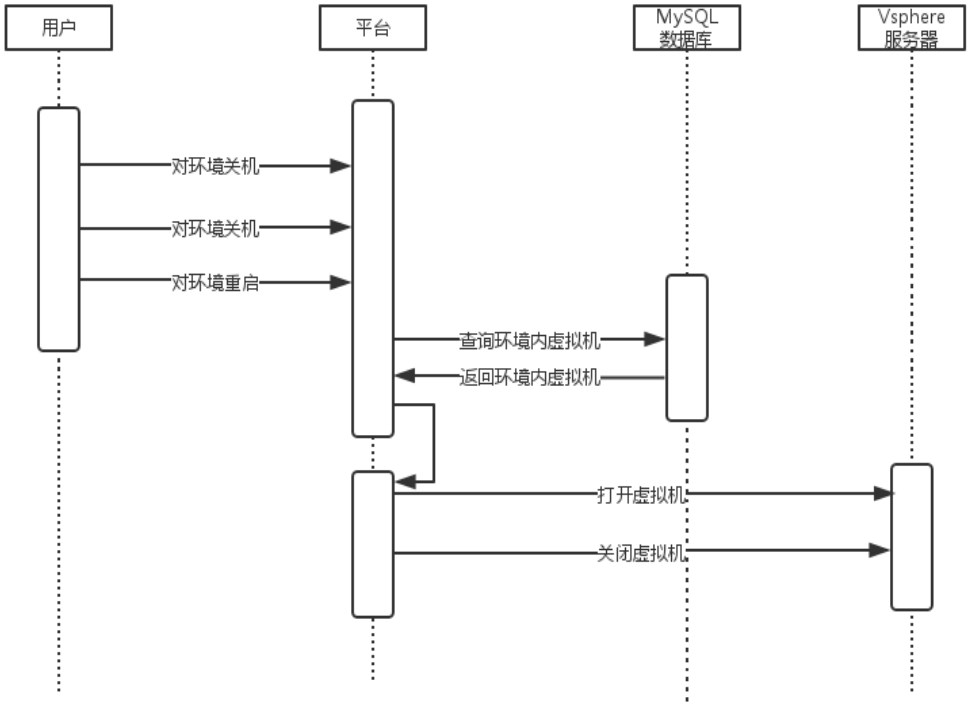

图 4-9 环境开关机和重启时序图

环境开关机和重启时序图说明：用户点击界面上对环境开机按钮以后会触发一个请求，平台后台收到这个请求以后解析出需要操作的环境的名称，然后再从 MySQL 数据库中找出这个环境中的虚拟机的信息，然后连接到 VSphere 服务器循环对这些虚拟机进行开机操作。

用户点击界面上对环境关机按钮以后会触发一个关机的请求，平台后台收到这个请求以后解析出需要操作的环境的名称以及操作的类型，然后再从 MySQL 数据库中找出这些环境中的虚拟机的信息，然后连接到 VSphere 服务器循环对这些虚拟机进行关机操作。

用户点击界面上对环境开机按钮以后会触发一个重启的请求，平台后台收到这个请求以后解析出需要操作的环境的名称以及操作的类型，然后再从 MySQL 数据库中找出这个环境中的虚拟机的信息，然后连接到 VSphere 服务器循环对这些虚拟机进行重启操作。

以下为对虚拟机进行开机，关机，重启或者修改配置的时序图：

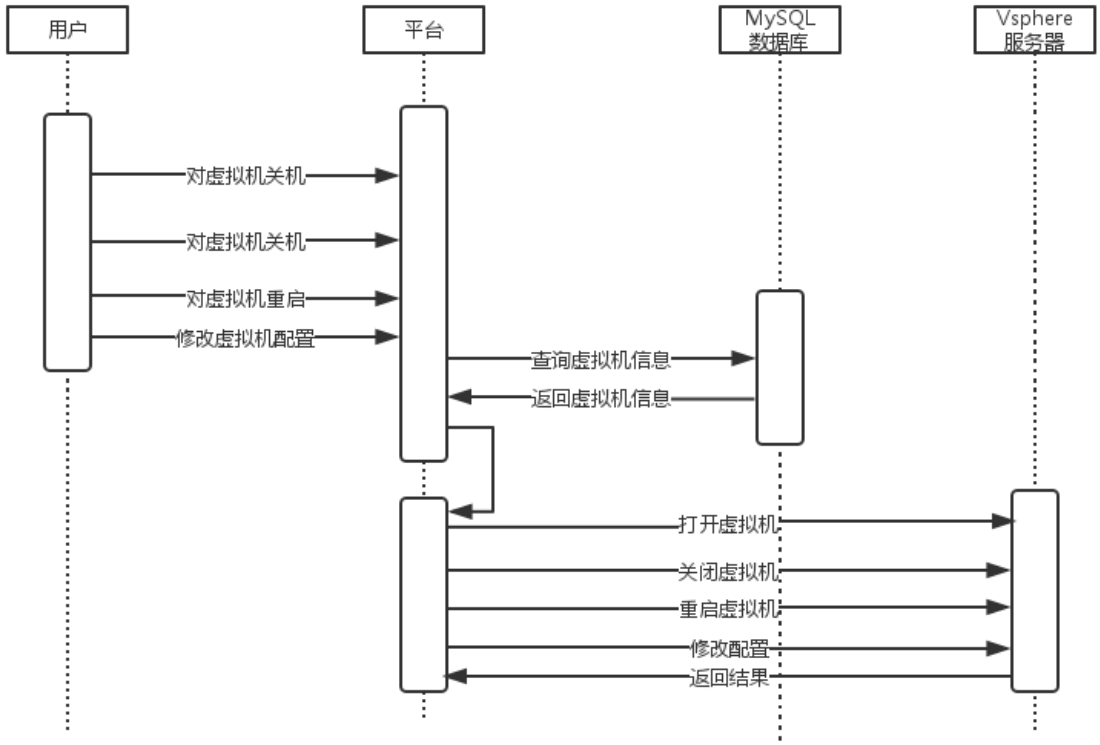

图 4-10 虚拟机开机关机重启和修改配置时序图

虚拟机开机关机重启和修改配置时序图说明：

用户可以利用界面上每一个虚拟机后面的开机、关机或者重启按钮触发一个开机、关机或者重启的请求，平台的服务器接受到请求以后会从请求中解析出要操作的主机的名称，然后利用该名称从 MySQL 数据中查出该虚拟机的信息，然后连接到 Vsphere 服务器利用 Pysphere 对虚拟服务器进行开机、关机或者重启等操作。

用户也可以点击界面上的修改密码功能进行虚拟机修改，再填入指定的信息以后会触发一个修改密码的请求，平台服务器接受到请求以后会从 MySQL 数据库中查到该虚拟机的密码等信息，然后连接到 Vsphere 服务器结合 Pysphere 和 vmware tools 对虚拟服务器的密码进行修改。然后将修改结果返回到平台，平台将修改的结果进行展示，最后还要对 MySQL 中的相关虚拟机记录进行更新。

用户也可以点击界面上的修改虚拟机网络配置功能进行虚拟机网络配置修改，再填入指定的信息以后会触发一个修改网络配置的请求，平台服务器接受到请求以后会从 MySQL 数据库中查到该虚拟机的密码等信息，然后连接到 Vsphere 服务器结合 Pysphere 和 vmware tools 对虚拟服务器的相关的配置文件进行修改。然后重启相关的网络服务，最后将修改结果返回到平台，平台将修改的结果进行展示，最后还要对 MySQL 中的相

关虚拟机记录进行更新。

以下为服务器代码升级时序图：

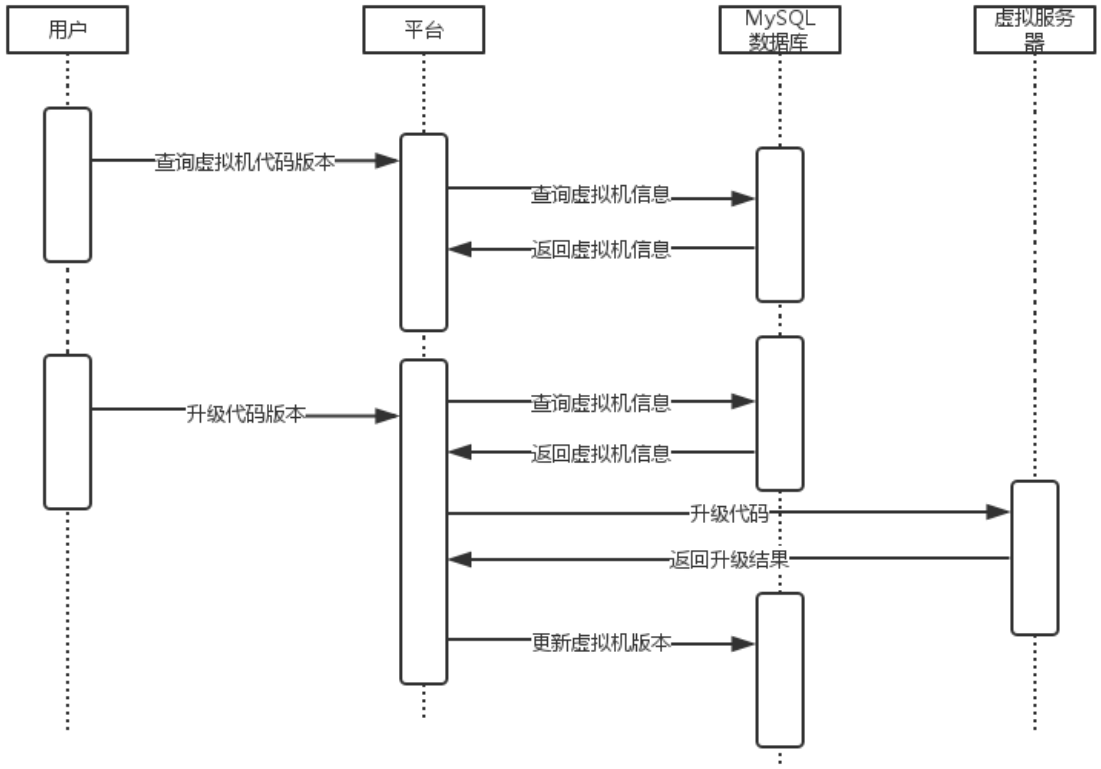

图 4-11 服务器代码升级时序图

服务器代码升级时序图说明：

首先用户进入 VM 升级子栏目以后会触发一个查看当前平台中所有虚拟机的代码版本的请求，平台服务器后端会接到这个请求，然后从数据库中查询平台中所有的虚拟机的信息然后返回平台，再由平台返回到前端进行展示，展示的信息包括虚拟机的名称，虚拟机的 IP，虚拟机的类型以及虚拟机中当前的代码 svn 版本号。

用户可以使用两种方法触发升级虚拟机代码版本的请求，一种是直接输入 svn 代码的版本号然后点击更新按钮，另一种是批量选择虚拟机并输入 svn 代码版本号然后点击界面上的批量升级按钮，平台服务器接受到请求以后会解析请求中的服务器信息和 svn 代码的版本信息，然后查询 MySQL 数据库中这些虚拟机的信息，接着利用数据库中虚拟机的 IP 和密码等信息登陆到这些虚拟机循环执行代码升级的命令，直到所有的虚拟机全部完成代码升级，最后再对 MySQL 中更新的虚拟机的代码版本进行更新，修改到最近的版本号。

4.2.4 表结构设计

1. 环境信息表 www_envir

表 4-1 环境信息表 www_envir

字段名	类型	允许空	描述	备注
id	int(11)	N	自增主键	主键
envir_name	varchar(30)	N	环境名称	
envir_desc	varchar(50)	N	环境描述	

2. 虚拟机类型表 www_vm_type

表 4-2 虚拟机类型表 www_vm_type

字段名	类型	允许空	描述	备注
id	int(11)	N	自增主键	主键
vtype_name	varchar(20)	N	类型名称	

3. 虚拟机信息表 www_vm

表 4-3 虚拟机信息表 www_vm

字段名	类型	允许空	描述	备注
id	int(11)	N	自增主键	主键
vhostname	varchar(50)	N	主机名	
vip	varchar(20)	N	主机 IP	
vtype_num_id	int(11)	N	主机类型	
vpwd	varchar(20)	N	密码	
vold_svn	varchar(10)	N	上一个 svn 版本	
vnew_svn	varchar(10)	N	现在 svn 版本	
vos	varchar(20)	N	主机系统	
vcpu	varchar(10)	N	主机 cpu 核数	
vmem	varchar(10)	N	主机内存大小	
vdisk	varchar(10)	N	主机硬盘大小	
vis_live	tinyint(1)	N	是否正在使用	
vgroup_id	int(11)	N	属于环境	

4.2.5 系统实现图

以下为虚拟机管理部分实现效果图

图 4-12 VM 管理实现图

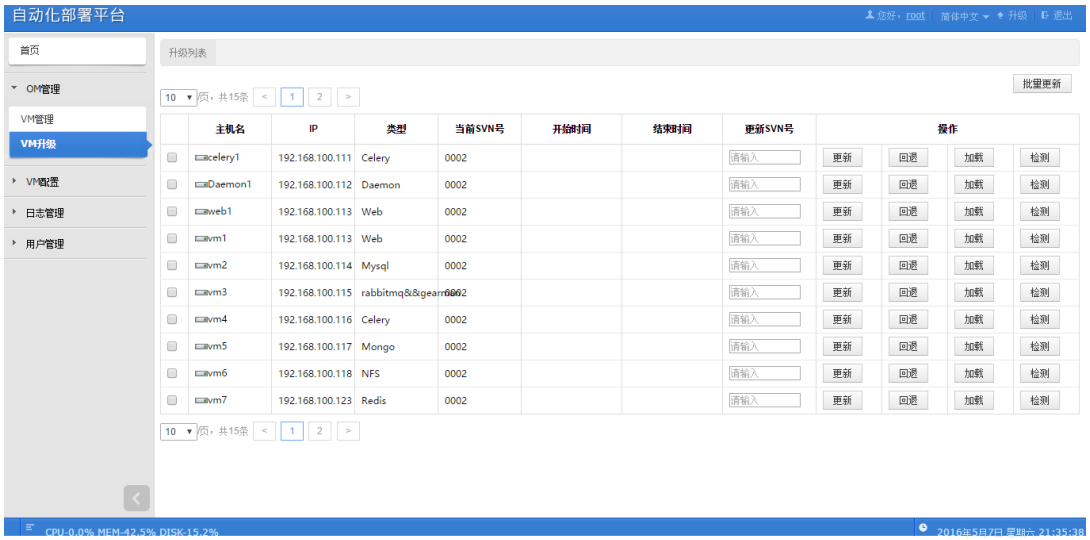

图 4-13 VM 升级实现图

4.3 虚拟机配置模块

4.3.1 设计描述

该模块主要是对某个环境中的某个服务进行配置。

该模块现在支持六个服务的配置，分别为 DNS、平温度、网页篡改、爬虫、网页测速、和三方通告，这六个服务均为本人所在的实习部门所开发的服务。

首先点击ＶＭ配置以后会出现这六个子栏目，用户可以点击其中一个进入该服务的配置，点击之后会出现平台中所有的环境列表，以表格方式展示，展示的信息有环境的名称，环境的描述，在每一个环境之后会有一个配置的按钮，在用户点击以后便可以进入该环境中该服务的配置流程。

点击配置以后用户会首先进入服务检测的页面，该页面用户需要选择 5 种类型的虚拟机，分别为 Mongo、Mysql、NFS、RabbitMQ 和 Redis，在该页面表格的 第一列将显示这五种虚拟机类型，然后显示针对每一种虚拟机所需要检测的服务，最后一列是下拉框，用户可以在下拉框中选择改环境中该类型的某一个虚拟机，当所有类型的虚拟机选择完毕以后，用户可以点击下方的开始检测按钮进行服务检测，平台会将检测结果返回到界面之上，当其中一台的服务检测出现问题时会在该虚拟机的后面显示出现的问题，可能出现的问题有无法连接该主机或者是服务不存在，当所有选择的虚拟机服务检测通过以后界面会显示服务检测正常，然后用户才可以点击进入配置按钮进入服务配置页面。

点击进入配置之后会进入服务配置界面，该界面以表格方式展示信息，用户在该页面需要选择三种类型的虚拟机进行服务配置，分别为 pamc_web、pamc_daemon、celery，在表格的第一列为这三种虚拟机的类型，第二列为这三种类型的虚拟机需要配置的内容，即配置过程中需要修改的配置文件，分别为 settings.py 和 conf.py、settings.py 和 conf.py、conf.py 和 supervisord.conf。最后一栏是下拉框，用户可以在该下拉框中选择该环境中该类型的虚拟机进行配置，当用户在下拉框中选择虚拟机完成以后点击开始配置按钮，然后平台会对这些虚拟机进行服务配置，并将配置的结果显示在界面之上，同样上述所有操作将被记录在平台日志当中。

这六个模块操作时在前台的流程一样，都要先选择要进行服务配置的环境，然后进入服务检测页面，需要选择进行服务检测的虚拟机，之后平台后台对根据虚拟机类型的不同确定该虚拟机上需要检测的服务，当所有选择的虚拟机服务检测正常以后才能进入配置的界面，如果某个虚拟机的服务出现异常则要再异常处理之后重新进行检测，检测通过以后才可能进行下一步的服务配置。

再下一步的服务配置之中用户也需要选择不同类型的虚拟机进行服务配置，平台后台会根据不同类型的虚拟机修改不同服务配置文件中的参数，这些参数会从前一页的服务检测页面中选择的虚拟机信息得到，然后服务配置完成后会给出服务配置的结果是成功还是某台虚拟机的服务配置失败。

如果配置失败平台应该显示失败的原因，比如无法正常连接到该服务器等原因，等问题解决以后，用户可以对这些虚拟机的服务重新进行服务配置。以上操作均会被记录到系统日志当中。

4.3.2 流程图

以下为虚拟机配置部分流程图

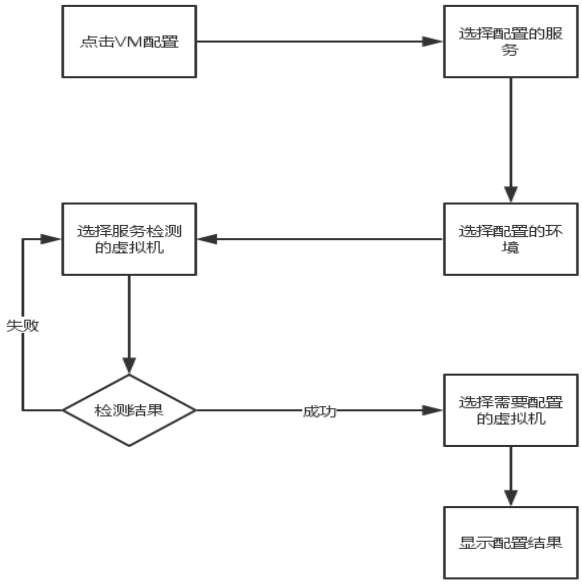

图 4-14 虚拟机配置模块流程图

虚拟机配置模块流程图说明：首先用户点击左边导航栏的 VM 配置以后会进入虚拟机配置部分，然后用户需要根据需要选择要进行配置的服务，现在该平台支持六种服务的配置，用户点击要配置的服务子栏目之后，平台会显示现在在该平台之上管理的环境的列表，显示的信息有环境的名称和环境的描述信息，然后用户可以选择一个环境点击环境最后的操作图表进行服务配置。

用户点击以后会进入服务检测页面，平台会列出再检测阶段需要选择的虚拟机的类型，用户需要为每一个类型选择一个虚拟机，这些虚拟机都属于之前选择的环境，用户可以选择的虚拟机平台为用下拉框的方式分类进行显示方便用户的选择，当用户选择完成之后点击服务检测，服务检测失败以后需要重新进行检测，检测成功以后才可以进入服务配置页面进行服务配置。

服务检测完成以后可以点击页面上的进入配置按钮进入服务配置的页面，平台依然会列出在服务配置界面需要选择的虚拟机的类型以及该类型的虚拟机对应的要修改的配置文件，然后用户从每一个类型的下拉框中选择出一个该环境中对应类型的虚拟机，然后点击开始配置按钮便可以对选择的虚拟机进行服务配置，平台再配置完之后会返回配置的结果是成功还是失败，如果配置失败平台会显示失败的原因，再解决问题以后用户可以进行重新配置

4.3.3 时序图

以下为虚拟机配置部分时序图：

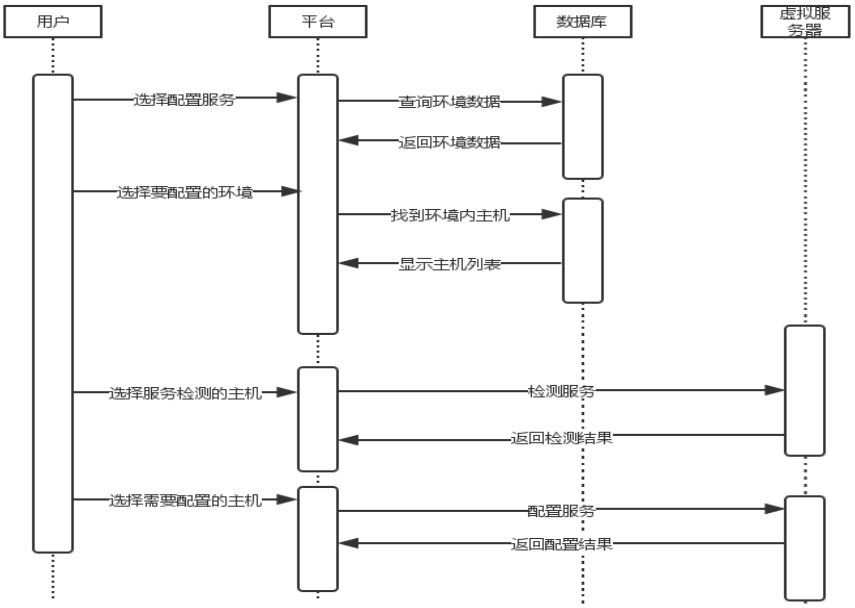

图 4-15 虚拟机配置时序图

虚拟机配置时序图说明：首先用户点击 VM 配置栏目以后选择需要配置的服务，界面会向平台服务器发送一个请求，服务器接到这个请求以后会从 MySQL 服务器中查询该平台中所有的环境以及环境的信息，然后将查询的结果返回到前端界面。

然后用户点击选择的环境进行服务配置，这时界面会向服务器发送一个请求，平台服务器接受到该请求以后会查询 MySQL 数据库中指定的类型的虚拟机并将这些虚拟机按照类型分组发送到前端，然后用户可以从这些虚拟机中选择要进行服务检测的虚拟机。

用户选择完需要服务检测的虚拟机之后点击服务检测按钮，界面会向平台服务器发送服务检测的请求，平台服务器会从这些请求中解析出需要进行服务检测的虚拟机的名称，然后从 MySQL 数据库中查询到这些服务器的 IP 以及密码等信息，然后用 fabric 批量进入这些虚拟机的后端指定指定的命令，然后从这些命令的返回信息可以获得该虚拟机上的服务运行情况，最后根据这些返回信息确定服务是否正常并将结果返回到界面。

如果所有的服务检测正常以后进入配置按钮可用，然后用户点击服务配置按钮以后会进入服务配置页面，界面会想平台服务器发送请求，之后平台服务器接受到请求以后会查询该环境中的制定类型的虚拟机信息并将这些信息打包分组发送到界面前端，最后前端对这些类型的虚拟机分组进行显示。

之后用户可以从这些分组的虚拟机中选择需要进行需要进行服务配置的虚拟机，然

后点击服务配置按钮，界面会将用户选择的虚拟机信息放在请求之中传到平台服务器，平台的服务器接受到请求之后解析出其中的虚拟机的信息，然后后从 MySQL 数据库中查询到这些服务器的 IP 以及密码等信息，然后用 fabric 批量进入这些虚拟机的后端指定指定的脚本或者命令，对虚拟机中指定的配置文件进行修改，最后对服务进行重启并返回执行结果，最后将这些结果进行分析并返回前端进行显示。

4.3.4 系统实现图

以下为虚拟机配置部分实现图：

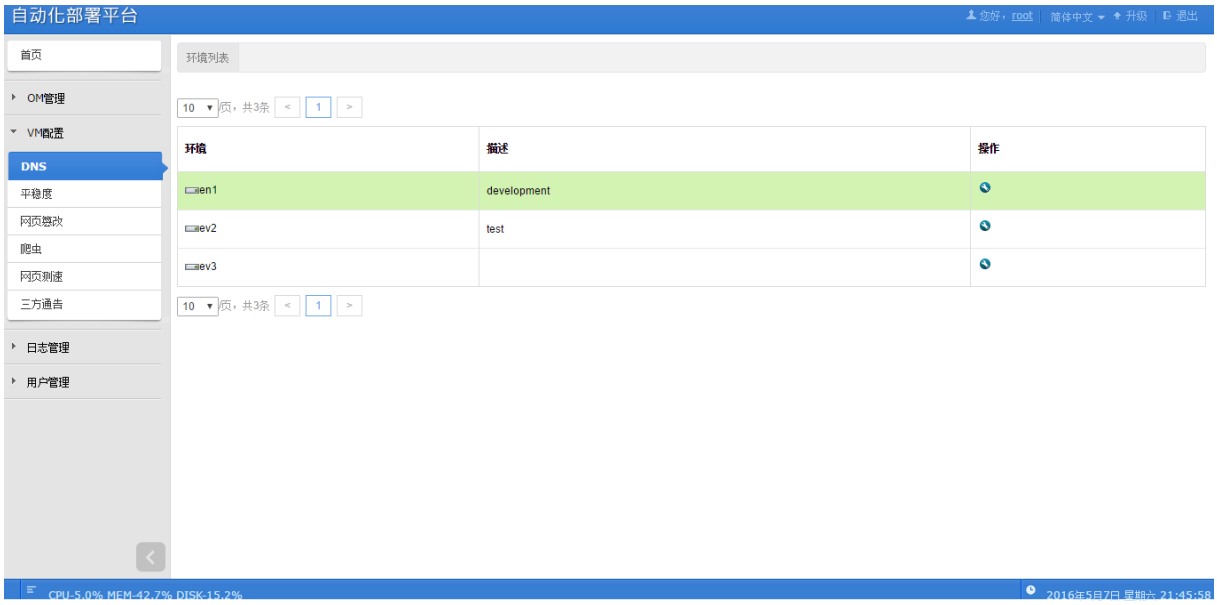

图 4-16 虚拟机配置环境选择实现图

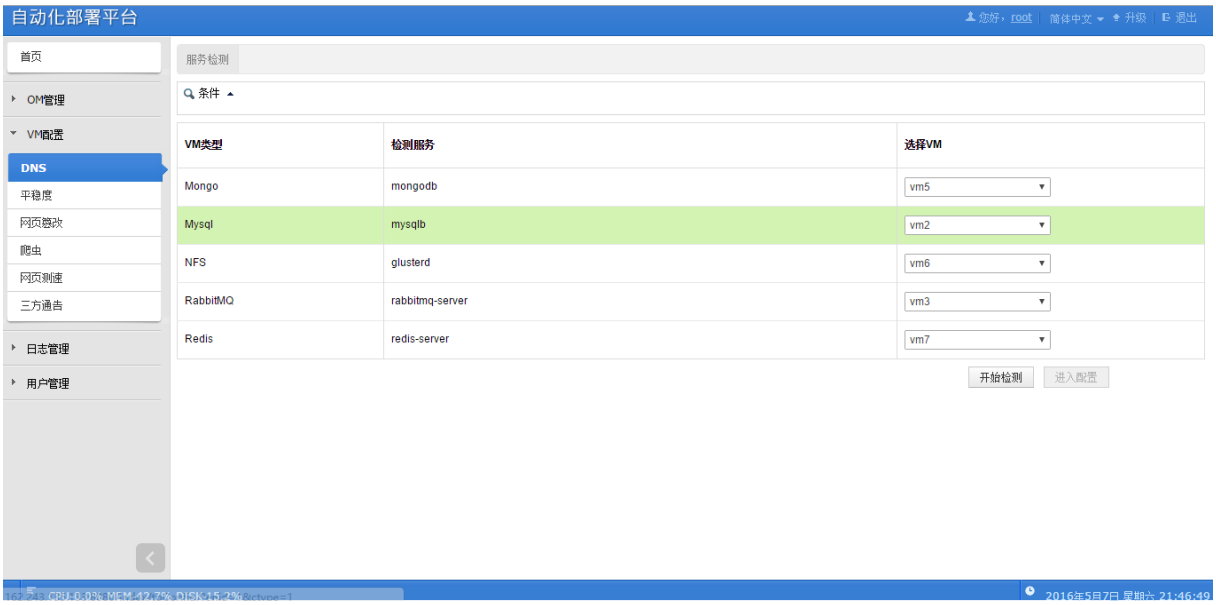

图 4-17 虚拟机配置服务检测实现图

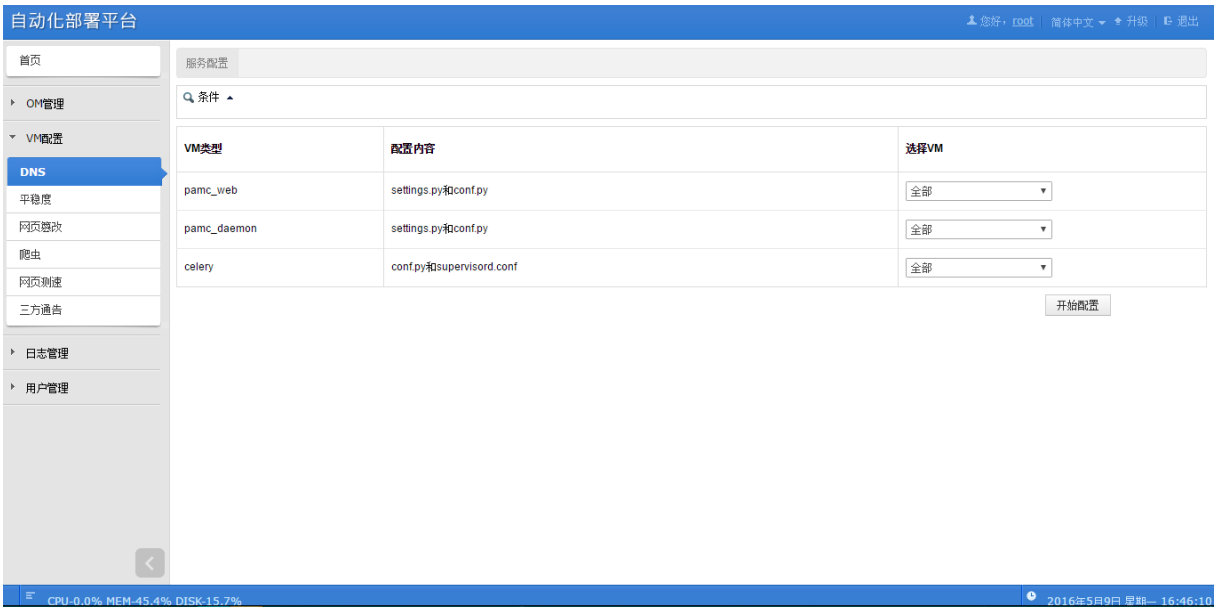

图 4-18 虚拟机配置服务配置实现图

4.4 虚拟机监控模块

4.4.1 设计描述

该模块主要是对平台所管理的虚拟机中的服务进行监控。

所有由该平台所部署的虚拟机在虚拟机初始化完成之后都会被传入一个服务监控脚本并运行该脚本，该脚本将从特定的目录读取需要监控的服务的信息，包括需要监控的服务的名称，开启服务的可执行文件所在的路径，以及开启服务的命令，尝试启动服务的次数和该主机的负责人邮箱等信息，然后该脚本会根据这些信息对服务进行监控。

该脚本将每秒钟查看在主机之上运行的服务进程，然后如果检测到该主机上的该服务已经停止运行，会尝试之前读取的信息中的开启服务的方法尝试启动该服务，并发邮件给该主机的负责人，如果超过指定的尝试启动次数之后，服务依然无法正常运行，然后会发给指定的负责人指定的邮件内容，包括服务宕机的主机 IP，尝试重启的次数，以及宕机的服务信息，这些信息也会录入平台的日志信息当中。

4.4.2 流程图

以下为虚拟机监控模块的流程图：

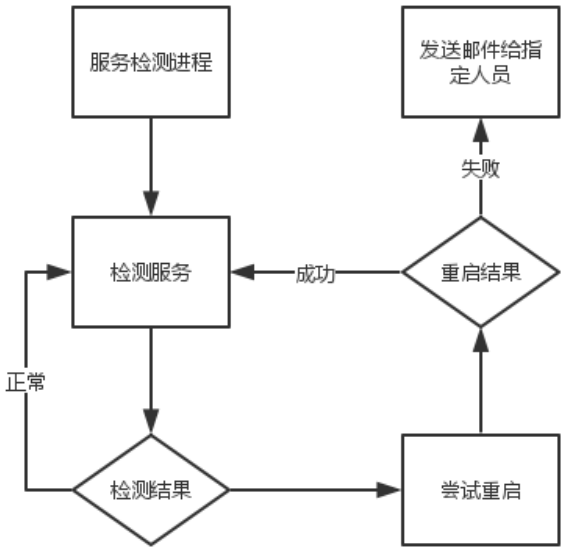

图 4-19 虚拟机监控流程图

虚拟机监控流程图说明：首先每个虚拟机中的服务检测进程会每秒中对虚拟机中的服务进行检测，当发现服务器中的服务进程宕机时会立即执行服务的启动命令或者是脚本，当启动尝试超过预设次数依然没有正常启动后会发邮件通知给预设的负责人服务出现宕机且无法正常启动，如果服务已经启动会发邮件给负责人说明服务宕机并且已经正常启动，并附上尝试启动的次数。

4.4.3 时序图

以下为虚拟机监控模块的时序图：

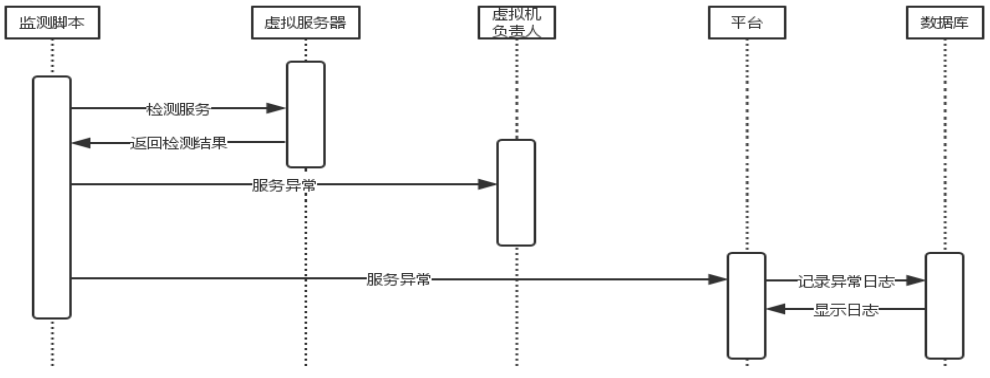

图 4-20 虚拟机监控时序图

虚拟机监控时序图说明：服务监测脚本进程会已知对虚拟服务器上的服务进行检测，如果出现服务异常之后会尝试服务启动，并且异常的信息发送到虚拟机的相关负责人，并且会向平台发送一个请求，请求中包含服务宕机的信息，包括虚拟机的 IP 和宕机时间等信息，然后平台会将这些信息计入平台日志当中。

4.4.4 系统实现图

由于该模块的特殊性，该模块的虚拟机监控脚本在虚拟机新建之初便存在与虚拟机之中并对服务进行监控，当服务出现问题时会用邮件进行通知，所以该模块在平台之上并没有对应的界面实现效果图。

4.5 日志系统模块

4.5.1 设计描述

该模块包括显示和管理平台日志内容。

平台的日志分为四种类型：登陆日志，操作日志，异常日志和升级日志。

登陆日志：主要包括用户登陆的信息，包括登陆人的登陆时间和登陆 IP 等信息。

操作日志：主要包括环境的增加和删除，环境的开机、关机和重启操作，记录的内容包括操作人的 IP 和操作时间以及登陆结果；虚拟机的增加和删除以及配置修改，虚拟机的开机关机和重启操作，记录的内容包括操作人的 IP、操作时间和操作的虚拟机的名称和类型以及操作的结果；虚拟机配置部分的服务检测和服务配置的信息，记录的内容包括操作人的 IP、操作时间和操作的虚拟机的名称和类型以及操作的结果。

用户点击日志管理栏目以后会进入日志管理模块，之后会以表格方式显示该平台中的所有日志信息，默认每页显示 10 条日志，用户可以通过上方或者下方的下拉框修改每页显示的日志条数。日志的内容包括该日志的序号，以及日志的类型，该日志的发生时间，操作的虚拟机名称，操作的虚拟机的类型，以及操作人的登陆的 IP 和日志详细信息（主要为操作的内容和结果）。用户可以用上方或者下方的按钮进行换页或者到达指定的某一页。

用户可以点开上方的查询按钮，会展开一个搜索栏，第一行为要查询的日志的时间段，用户可以用弹出来的日历选择时间，第二行为要搜索的日志的类型，第三行为要搜索的虚拟机的类型，第四行为要搜索的操作人的 IP，用户可以选择填入其中的某几项，之后平台会给出这几项的交集搜索结果。

用户也可以点击日志列表上方的清空按钮删除现在平台中的所有日志，或者点击备份按钮，平台将把现在平台中的所有日志写入一个文本文档放在服务器的指定目录当中。

4.5.2 流程图

以下为日志模块的流程图：

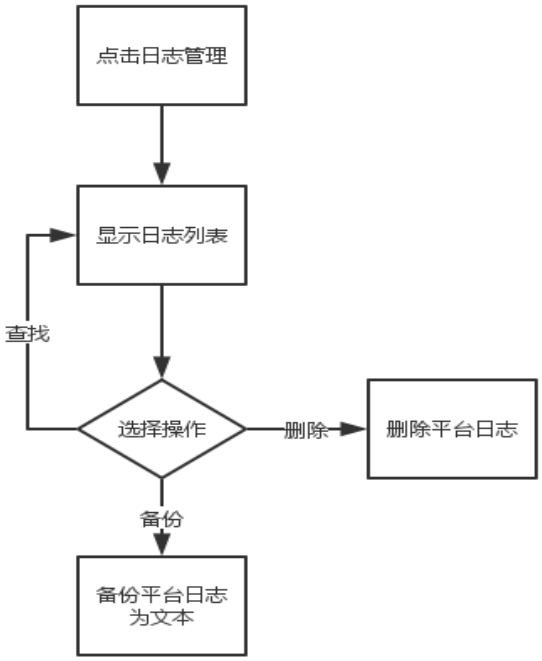

图 4-21 日志模块流程图

日志模块流程图说明：用户点击左边导航栏中的日志管理栏会进入日志管理模块，进入以后平台会为用户分页展示平台的日志信息，默认每页展示 10 条日志记录，用户也可以根据自己的需要对每页显示的日志数量进行调整，用户可以进行三种操作，根据某些信息搜索日志，删除日志和备份日志。

用户可以点击日志展示栏上面的查询按钮打开查询框，一共可以利用四个条件对日志进行筛选，可以指定日志的起始时间和终止时间，指定要查询日志的类型，是登陆日志，操作日志，升级日志还是异常日志，或是查询几种类型的虚拟机的操作，再 VM 类型一栏中可以利用下拉框选择多了虚拟机的类型进行查询，或是根据 IP 进行查询。

用户也可以点击删除按钮，之后平台会删除平台中记录的所有日志。

用户可以点击备份按钮对日志中的所有日志进行备份，之后平台会在服务器中产生一个文本文档，里面内容是平台产生的所有日志。

4.5.3 时序图

以下为日志模块的时序图：

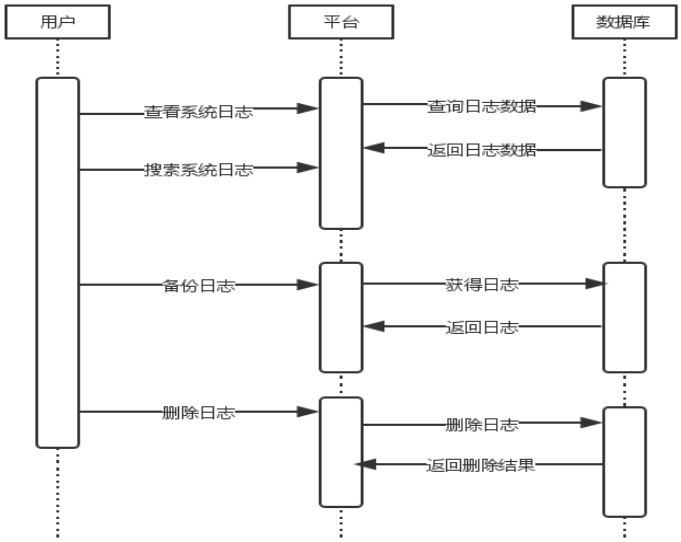

图 4-22 日志模块时序图

日志模块时序说明：用户进入日志模块以后会向平台服务器发送一个查看系统日志的请求，平台服务器接受到请求之后会向 MySQL 数据库中查询平台中的日志并把查询到的信息返回到平台前端。

用户输出搜索的关键字以后点击搜索按钮会触发一个搜索系统日志的请求，服务器接受到请求以后会解析请求中搜索的关键字，然后连接 MySQL 数据库并用这些关键字去过滤日志，然后将过滤后的日志返回到界面。

点击删除按钮之后平台会发送删除日志的请求，平台服务器接受到请求以后会删除 MySQL 数据库中的所有日志信息。

点击备份按钮以后平台会发送备份日志的请求，平台服务器接受到请求以后会查询 MySQL 数据库中的所有日志信息并将结果写入文件中。

4.5.4 表结构设计

表 4-4 日志信息表 logmanage_logmanage

字段名	类型	允许空	描述	备注
id	int(11)	N	自增主键	主键
vname	varchar(50)	Y	操作主机名称	
vip	varchar(20)	N	操作人 ip	
vtype	varchar(20)	N	操作主机类型	

dtime	datetime	N	操作时间	
description	varchar(500)	Y	操作描述	
type	int(11)	N	操作日志类型	

4.5.5 系统实现图

以下为日志模块系统实现图：

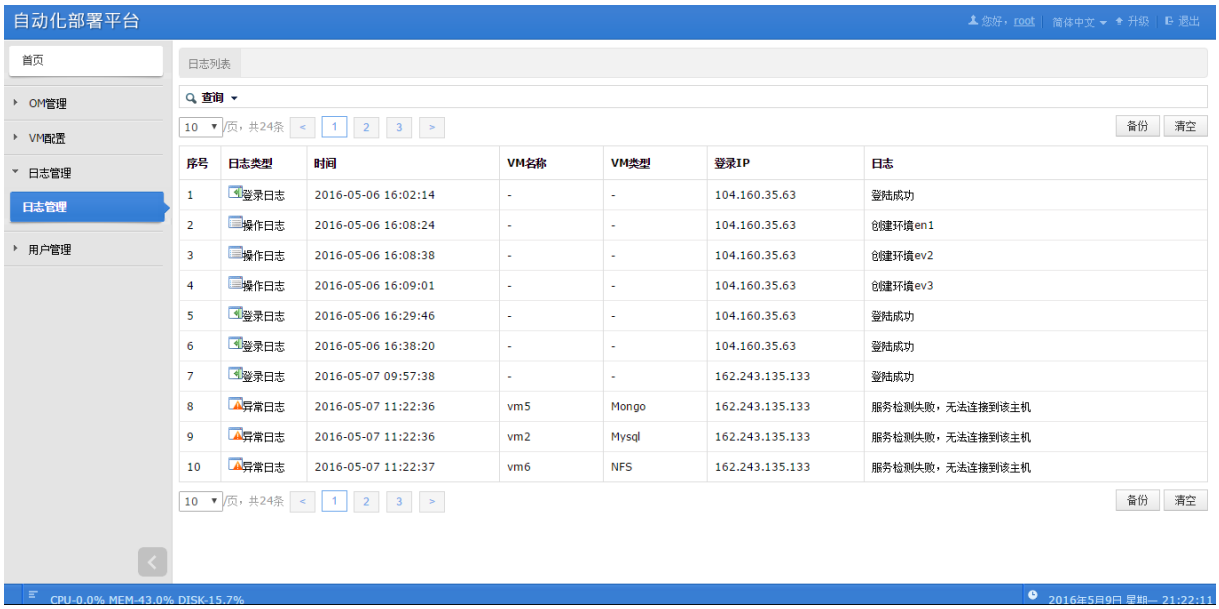

图 4-23 日志模块系统实现图

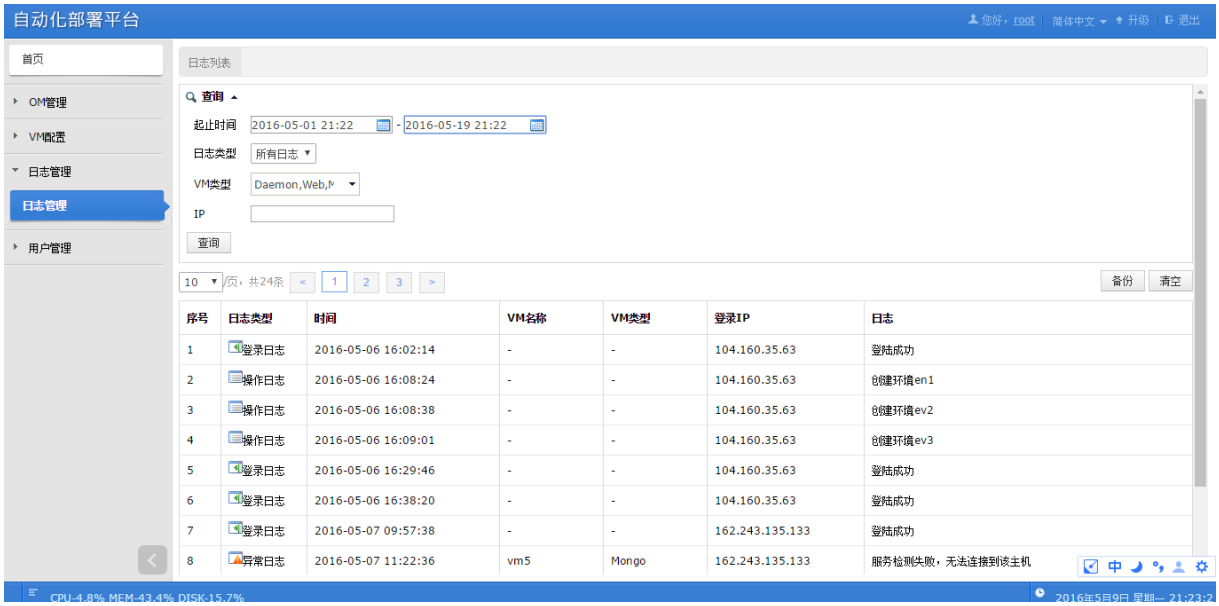

图 4-24 日志模块系统搜索实现图

4.6 用户管理模块

4.6.1 设计描述

该模块主要是对平台中的用户进行管理。

由于该平台的特殊性，只对公司中指定的人员开放，所以该平台没有注册功能，如果需要一个新用户只能向平台的系统管理员申请账户。

用户点击用户管理栏目后将进入用户管理模块，该模块会展示该平台中现在的所有用户，展示的信息包括该用户的姓名，用户的角色是系统管理员还是普通管理员，然后是对该用户可以进行的操作，包括对用户信息进行编辑和删除。

用户可以点击每一个用户最后的编辑按钮对该用户进行编辑，之后平台会弹出一个编辑悬浮框让用户输入指定的信息。

编译用户悬浮框首先会显示该用户的姓名（不可编辑），如果要修改密码的话用户需要在用户姓名下方的密码和确认密码框输入新密码，密码和确认密码匹配以后将对该用户进行密码修改，如果不需要修改密码则对该项留空，之后用户也可以用下方的下拉框修改该用户的角色，用户需要在下拉框中选择该用户是系统管理员还是普通管理员，然后点击确定按钮会对该用户完成信息修改。

用户本人和系统管理员可以修改用户密码，只有系统管理员才可以修改用户的角色即权限。

用户也可以点击添加按钮添加一个用户，点击用户信息列表上方的添加按钮会出现添加用户的悬浮框，用户需要输入需要添加用户的姓名，密码和确认密码，并需要从下拉框中选择该用户的角色，是普通管理员还是系统管理员，之后点击确认按钮完成用户添加。

只有系统管理员才有权限添加一个用户。如果用户角色为普通管理员无法进行此操作。

用户也可以点击每一个用户后的删除按钮，之后平台对让用户对操作进行确认，确认完成之后平台会删除该用户。

只有系统管理员才有权限删除一个用户。如果用户角色为普通管理员无法进行此操作。

4.6.2 流程图

以下为用户管理模块流程图：

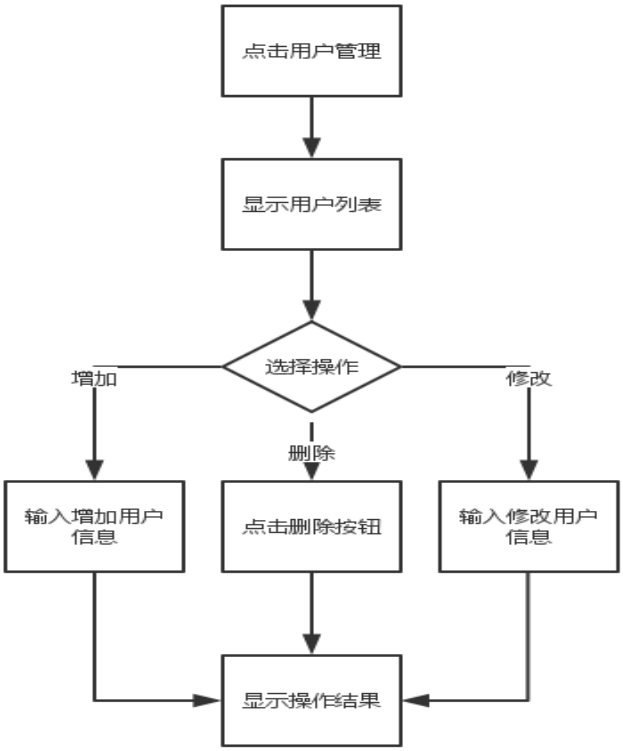

图 4-25 用户管理模块流程图

用户管理模块流程图说明：用户点击导航栏中的用户管理可以进入凭条的用户管理模块，平台会显示该平台现有的用户的信息，包括用户的用户名，用户的角色（系统管理员或者普通管理员），最后还有可以对该用户的操作，包括编辑和删除。

用户可以点击每一个用户后面的编辑图表对一个用户进行编辑，在弹出的编辑框中可以对用户的密码和用户的角色进行修改，修改之后点击确定即可修改完成。

用户可以点击每一个用户后面的删除图表对一个用户进行删除，在弹出的确认框中进行确认即可删除完成。

用户也可以对该平台进行用户添加，点击添加按钮以后会弹出用户添加框，需要输入用户的姓名，密码，确认密码，并从下拉框中为添加的用户选择角色，是系统管理员还是普通管理员，然后点击确定当确认密码和密码相同之后便可以完成用户的添加操作。

4.6.3 时序图

以下为用户管理模块时序图：

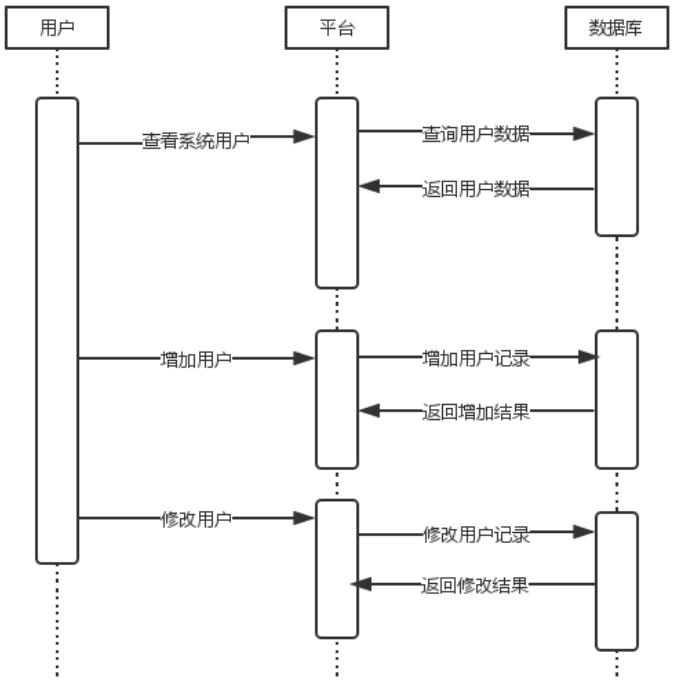

图 4-26 用户管理模块时序图

用户管理模块时序图说明：用户点击左边导航栏中的用户管理子栏目会进入用户管理模块，之后前端会向平台的服务器后端发送一个查看当前平台中用户的请求，平台服务器接受到请求之后会从 MySQL 数据库中查询当前平台的所有的用户，然后将这些用户信息返回到前端进行显示。

用户可以点击平台的添加用户按钮，在弹出添加用户框中添加完信息点击添加，之后平台前端会利用 JavaScript 去验证用户输入的密码和确认密码是否一致，一致后会将信息包装到一个 POST 请求发送到服务器，平台的服务器后台便可以收到添加用户的请求，之后平台服务器后台解析这个请求中的用户信息，再次验证密码和确认密码，然后还会从 MySQL 数据库中找到操作人的信息并对操作人的权限进行验证，如果通过便会将该用户加入到 MySQL 数据库中的用户表当中，如果权限不符合将返回结果表示权限不足，并将添加结果返回到界面进行显示。

用户可以点击平台的删除用户按钮，确认之后会将信息包装到一个 POST 请求发送到服务器，平台的服务器后台便可以收到添加用户的请求，之后平台服务器后台解析这个请求中的用户信息，然后还会从 MySQL 数据库中找到操作人的信息并对操作人的权限进行验证，如果通过便会将该用户的记录从 MySQL 数据库中删除，如果权限不符合将返回结果表示权限不足，并将添加结果返回到界面进行显示。

用户可以点击平台编辑用户按钮，在弹出编辑用户框中添加完信息点击确定，之后

平台前端会利用 JavaScript 去验证用户输入的密码和确认密码是否一致（如果为空则表示不对密码进行修改），一致后会将信息包装到一个 POST 请求发送到服务器，平台的服务器后台便可以收到添加用户的请求，之后平台服务器后台解析这个请求中的用户信息，再次验证密码和确认密码，然后还会从 MySQL 数据库中找到操作人的信息并对操作人的权限进行验证，如果通过便会将该用户的 MySQL 数据库中的信息进行更新，如果权限不符合将返回结果表示权限不足，并将添加结果返回到界面进行显示。

4.6.4 表结构设计

表 4-5 用户信息表 auth_user

字段名	类型	允许空	描述	备注
id	int(11)	N	自增主键	
password	varchar(128)	N	密码	
last_login	datetime	Y	上次登陆时间	
is_superuser	tinyint(1)	N	用户角色	
username	varchar(30)	N	用户名	
first_name	varchar(30)	Y	用户 first name	
last_name	varchar(30)	Y	用户 last name	
email	varchar(75)	Y	用户邮箱	
date_joined	datetime	N	用户加入时间	

4.6.5 系统实现图

以下为用户管理模块系统实现图：

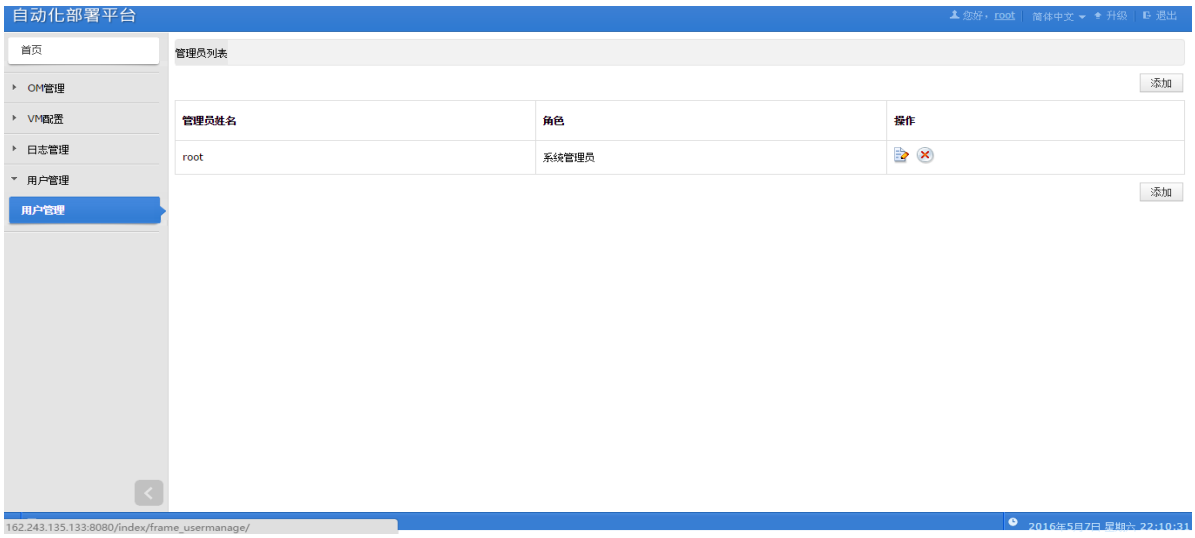

图 4-27 用户管理模块系统实现图

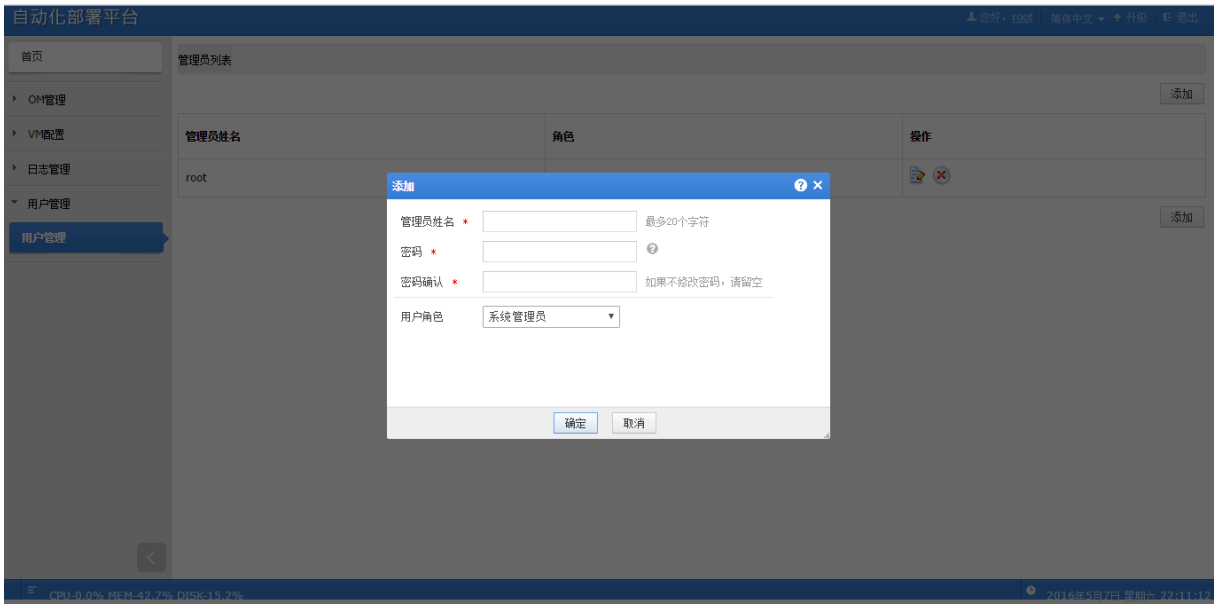

图 4-28 用户添加系统实现图

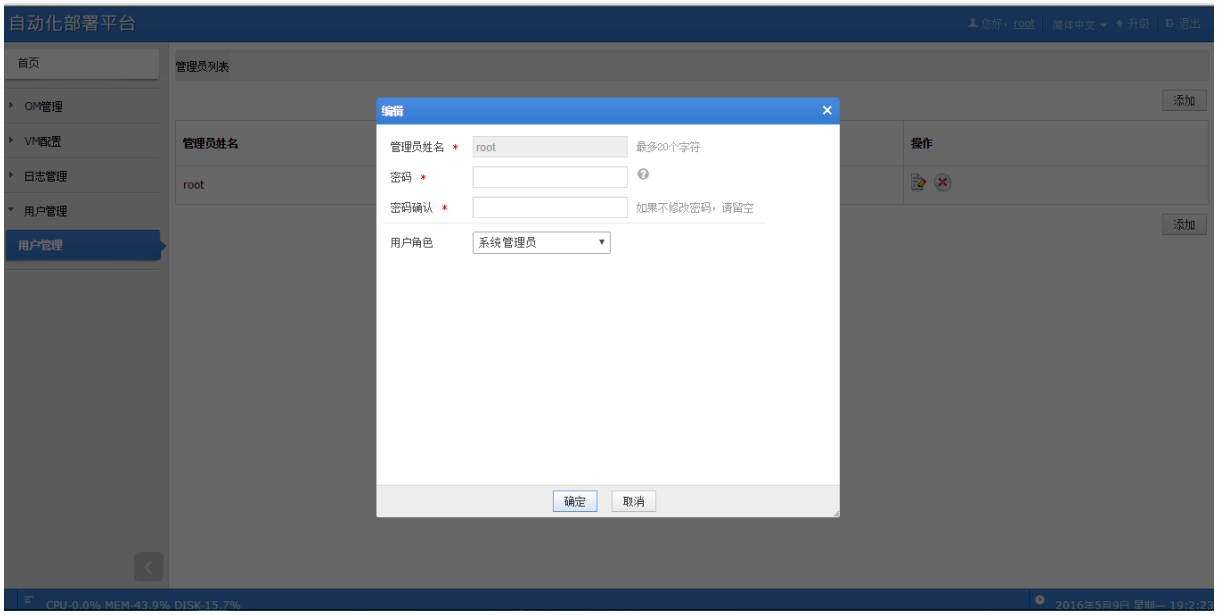

图 4-29 用户编辑系统实现图

5 结论

这个论文来源于本人所在实习公司的实际项目－－自动化服务部署平台，该平台主要致力于使公司的服务部署和虚拟机管理流程更加方便简介，由于平台的特殊性，该平台面向的用户主要是公司内部的部分人员，利用此平台可以实现服务的快速部署和配置，对研发和生产环境中的众多虚拟机进行统一的管理，并可以及时的监控服务的运行状态和方便的升级虚拟服务器中的服务代码，大大减少了服务配置和虚拟机管理的工作量，并可以对服务进行监控保证了服务的正常运行。

本平台采用 Django Web 框架进行开发，并在开发过程中将整个平台分为六个不同的模块，提供了统一的 web 图形界面方便用户进行管理操作，平台从 web 界面获得用户的操作，然后平台会和后台的 MySQL 中进行数据的存取和查询操作，并根据需要和 VMWare vSphere 进行交互以完成虚拟机的新建或者开关机等操作。系统开发完成之后使用 Nginx 和 uWSGI 进行部署到服务器。

该系统目前已在本人所实习的部门内部投入使用，虽然现在平台依然还会存在一些问题，后续会对该平台进行持续的完善，并可能加入更多的功能以方便部门里业务的开展和部署，通过这次开发本人也学到很多，通过这次论文更是对这次经历的一次总结。希望自己以后会获得更好的成长。

参考文献

[1] 王伟纲. 服务器虚拟化[J]. 金融科技时代, 2011, 19(5): 18-18.

[2] 闫红梅. 服务器虚拟化[J]. 网管员世界, 2008 (16): 27-27.

[3] Steinder M, Whalley I, Carrera D, et al. Server virtualization in autonomic management of heterogeneous workloads[C]//Integrated Network Management, 2007. IM'07. 10th IFIP/IEEE International Symposium on. IEEE, 2007: 139-148.

[4] 刘班. 基于 Django 快速开发 Web 应用[J]. 电脑知识与技术: 学术交流, 2009, 5(3): 1616-1618.

[5] 王冉阳. 基于 Django 和 Python 的 Web 开发[J]. 电脑编程技巧与维护, 2009 (2): 56-58.

[6] Mishchenko D. VMware ESXi: Planning, Implementation, and Security[M]. Cengage Learning, 2010.

[7] 郑小长, 闫格. 基于 VMware vSphere 环境下虚拟服务器集群的构建[J]. 漳州师范学院学报: 自然科学版, 2013 (1): 44-46.

[8] 韩奕, 姜建国, 仇新梁, 等. 基于云计算的恶意程序检测平台设计与实现 恶意程序检测平台设计与实现[J]. Computer Engineering, 2014, 40(4).

[9] 王文义, 武华北. Linux 中进程间信号通信机制的分析及其应用[J]. 计算机工程与应用, 2005, 41(3): 108-110.

[10] 王迎庆. 证券公司证券营业部网络系统的虚拟化设计技术[J]. 信息系统工程, 2014 (8): 52-53.

[11] 邓仲举. 高可靠性集群部署的设计与实现[D]. 华中科技大学, 2012.

[12] 段春乐. 虚拟化技术在图书馆服务器整合中的应用——以西安科技大学图书馆为例[J]. 科技情报开发与经济, 2014 (7): 95-96.

[13] 方斌. 基于虚拟化的金融实验教学平台搭建的选择标准[J]. 电子制作, 2014, 24: 085.

致　　谢

在论文的编写和平台开发的过程中，遇到了很多的困难，也遇到了很多对我有帮助的人，这些人让我更好的成长，让我这四年学到和很多知识和道理。

首先，感谢的我的导师，老师在论文的撰写过程中给了我很多具有建设性的意见，辛苦的对我进行指导，让我少走了很多弯路，在我迷惑时向我指明方向，不仅在论文编写的过程中提供了很大的帮助，也对我的生活和以后的工作提供了很多宝贵的建议。

接下来要感谢的是我的老师、同学以及朋友们，你们让我大学四年成长了很多，没有你们的教导与帮助我是无法完成这个论文的，因为你们使我变成了现在的自己，因为你们我变得更加优秀，因为你们我对生活理解的更多。

最后，感谢学校对我们四年的教导，这四年北京交通大学让我成长了很多，让我开阔了眼界，让我达到了以前从没想过的高度，更重要的，大学为我展示了我以后的方向，也许以后还会迷茫，但是我不会后退。

附　　录

附录 A VMware vSphere 4 的服务器虚拟化——外文原文

Server Virtualization with VMware vSphere 4

The server virtualization market is expanding rapidly as customers seek savings in both space and energy through efficient use of hardware resources, an increase in business agility through prompt addition and removal of servers, and a reduction in the total cost of ownership (TCO) by separating hardware and business-system life cycles. The virtualization solution offered by Fujitsu and used by a large number of its customers in Japan combines Fujitsu's high-reliability servers and middleware products, which optimize data center operations, with vSphere 4, VMware's virtual infrastructure. This paper describes the transition of VMware's server virtualization products over the years, the functions and features of the vSphere 4 virtual infrastructure, its relation to cloud environments, and examples of using Fujitsu middleware with vSphere 4.

1. Introduction

The server virtualization market is expanding rapidly as customers seek space and energy saving through more efficient usage of hardware resources, greater business agility through prompt addition and removal of servers, and lower total cost of ownership (TCO) through separation of hardware and business-system life cycles. VMware, Inc.,1) headquartered in Palo Alto, California, is a virtualization software vendor with approximately 190 000 customers worldwide and about 6000 customers in Japan (as of October 2010). Its most recent product offerings for building and operating server virtualization environments are VMware vSphere 4 and VMware vCenter Server (respectively referred to as "vSphere 4" and "vCenter" hereafter. Fujitsu started providing VMware products in 2006 and has since been providing its customers with virtualization solutions by combining these products with its own high-reliability servers as well as its middleware products for optimization of data center operations. These solutions are deployed by more than 1200 customers (as of November 2010). Fujitsu has abundant customer experience with these virtualization solutions and has released three case studies regarding cloud systems, which are currently attracting considerable attention.• In-house Case Study 1: "Deployment of VMware vSphere 4 and Enhancement of On-Demand Virtual Environment Hosting for SaaS Providers"2)• In-house Case

Study 2: “Putting the Cloud Into Practice: Numazu Software Development Cloud Center”3)• Oita Prefecture Case Study: “Virtualization of Mission-critical Systems by PRIMEQUEST, PRIMERGY, and VMware vSphere 4 toward Creation of a Public Sector Cloud”4)Fujitsu’s server virtualization solutions are based on a number of server virtualization products (such as VMware, Hyper-V, RHEL-Xen, RHEL-KVM, and XenServer). From among these, this paper describes the transition of VMware server virtualization products over the years and introduces the functions of VMware’s latest product, vSphere 4.

2.Transition of VMware server virtualization

This section introduces how VMware’s server virtualization has changed, focusing on two aspects—the “hypervisor” and the “virtualization product.” The hypervisor, occasionally referred to as “VM monitor,” is a control program designed to enable the creation and control of virtual machines (VMs). As shown in Figure 1, VMware’s hypervisor product (shaded boxes in the figure) has evolved through three main phases.1) VMware GSX (GSX)The GSX hypervisor runs as an application on a general purpose OS (the “host OS”) on hardware.

1) In this setup, the host OS is in charge of I/O processing, which enables the hypervisor itself to have a compact configuration. However, this type of hypervisor suffers from overhead and extensibility problems. Like VMware Workstation, it is optimal for PC virtualization.

2) VMware ESX (ESX)The ESX hypervisor runs directly on “bare metal,” i.e., hardware—which is currently the most popular form [2) in Figure 1]. It features higher performance than GSX and has a console OS (COS) for hypervisor management. Client management is done by vSphere Client, and adding vCenter simplifies management of multiple ESX hypervisors. Since COS is Linux-based, it must be frequently patched, thus security is one challenge that it presents.

3) VMware ESXi (ESXi)ESXi, the latest hypervisor of VMware, does not have a COS [3) in Figure 1]. As with ESX, vSphere Client is used to perform management tasks. While maintaining the same capabilities as ESX, ESXi is more compact since it has no COS, is easy to install and configure, and does not require COS patches (which reduces security risks).

As summarized above, VMware’s hypervisor has evolved into a compact and high-performance product with easy-to-install features.

1) VMware ESX Server 2. this product enables resources on an ESX-installed server (ESX server) to be shared among VMs. Centralized management of multiple ESX servers is

provided by vCenter.

2) VMware Infrastructure 3.x (VI3) Adding the concept of resource pooling to VMware ESX Server 2.x, VI3 has enhanced overall capabilities. The CPU and memory resources of multiple ESX servers can be managed as a resource pool, enabling them to be handled as a single resource pool.

3) VMware vSphere 4.xThis product has significantly enhanced functionalities compared to those of VI3. It broadens the management scope to include network resources and shared storage and enables management of server, network, and storage resources as a single virtual infrastructure. It is better able to deal with dynamic hardware changes and enables agile modification of the ESX/VM configuration in accordance with resource requirement changes. As described above, vSphere 4 represents the evolution of VMware's server virtualization products in two major streams: a) more compact, high-performing, and easy-to-install hypervisor and b) strengthened adaptability to system changes. In other words, vSphere 4 is suitable for environments such as the cloud, where agile creation and operation of systems is required.

3.Functions and features of vSphere 4 virtual infrastructure

In this chapter, we will dive into the functionalities and features of vSphere 4.vSphere 4 enables users to chose either ESX or ESXi as the hypervisor and to create a virtual infrastructure having the following features.

1) Wide management scope In general, this kind of infrastructure virtualizes and pools together physical resources and provides VMs with virtualized CPUs, virtualized memory, virtualized networks, and virtualized disks. On top of that, vSphere 4 enables dynamic and detailed management, including resource allocation, configuration change, priority control, and access control.

2) Flexibility vSphere 4 provides a variety of functionalities such as dynamic addition of virtualized CPUs and virtualized memory (Hot Add), dynamic reconfiguration of virtualized disks and networks (Hot Plug), dynamic extension of virtualized disks (Hot Extend of Virtual Disks), and dynamic load balancing of network I/O. These functions enable elastic business operations—responding flexibly to service requirements and hardware changes without disrupting services provided by VMs.

3) Benefits for applications vSphere 4 also provides advanced functions such as ones for ensuring business continuity (vMotion and Storage vMotion enable the

moving of resources point-to-point [P2P] without disrupting the operation of VMs), automatic load balancing (DRS: Distributed Resource Scheduler), and automatic VM reboot upon server failure (HA: VMware High Availability). Since these functions are part of the virtual infrastructure, there is no need to reconfigure the OS or applications running on the VMs—they can be used as is.

4) Integrated management vSphere 4 provides useful features and extensive function groups for creating and operating virtual infrastructures. With these functionalities, vSphere 4 has always been a step ahead of other hypervisors.

The virtual infrastructure is integrally managed by vCenter. This enables system managers to manage almost all management tasks through the vSphere Client graphical user interface (GUI), which displays each management function in icon form for easy operation, enabling the entire virtual infrastructure to be managed as a single environment, just like Windows control panels.

4. vSphere 4 and cloud environments

Taking infrastructure as a service (IaaS) as an example (typically lending and aggregation of VMs), the role and responsibility of vSphere 4 in cloud environments is set forth below. To acquire physical machine resources, users generally go through a process of budget, ring (the process of obtaining management approval for a plan by circulating a draft prepared by the plan originator), and purchase, so it may be several months before the equipment is actually deployed. To acquire VM resources, users usually go through a simple process of application, content review, resource adjustment, VM creation/configuration, and provisioning, as long as the amount of resources requested does not exceed the amount available from the computing pool. If it does exceed the amount available, the procedure for acquiring physical machine resources is followed. Though it depends on resource availability and the skill of the IT manager, a new VM can usually be created within several days. With vSphere 4's management uniformity and configuration change flexibility, it is relatively easy to acquire VM resources. IaaS virtually automates VM creation and service provisioning. However, providing virtual infrastructure as a service and automating virtual infrastructures are not the whole story—there are several difficulties, as described below. The HA function can be used in cloud environments such as IaaS to improve the level of service by minimizing service downtime upon server shutdown. An overview of HA is shown in Figure 4. The HA function detects a failure in an ESX server through mutual monitoring between ESX servers, which exchange "heartbeats"

via the network. When a failure is detected [1) in Figure 4], it automatically reboots the VM that had been running on the failed ESX server on another ESX server [2) in Figure 4]. An acceptable level of availability can therefore be obtained without having to reconfigure the virtual OS or applications running on the VM. However, as the system is based on “mutual monitoring,” a procedure is needed to ensure normal and safe shutdown of all ESX servers. Such a procedure consists of VM Figure 3 Screenshot of vSphere Client shutdown (shutdown of OS and applications on each VM), switching of ESX servers to maintenance mode, and shutdown of each ESX server. In other words, this procedure affects business task procedures and requires the use of a mutual monitoring mechanism. Failure to follow this procedure results in problems such as the need to reconfigure the HA function after rebooting. Moreover, system problems result if the shutdown procedure is not concluded in time after the onset of a power outage due to a large number of ESX servers being deployed or a lack of uninterruptible power supply (UPS) capacity.

Although the above may seem a worst case scenario, it illustrates how system operations easily performed manually in a conventional virtualization environment using only vSphere 4 may be difficult in a virtualization environment that assumes automation, such as the cloud.2) VM restart on another ESX server ESX server Mutual monitoring Mutual monitoring Mutual monitoringXX1) System failure detected through mutual monitoring between ESX servers Figure 4Overview of VMware’s High Availability function.

Given its extensive experience and considerable expertise, Fujitsu is fully equipped and capable of addressing and providing a variety of solutions (from the creation of cloud environments to the solving of operation problems like the ones described above) by optimally combining vSphere 4 with Fujitsu middleware.

5. Conclusion

The vSphere 4 product is certainly well suited for the emerging cloud era, in which flexibility and agility are the keys to success. Fujitsu will continue to provide and support VMware products in combination with Fujitsu’s highly reliable servers and middleware, which are essential to data center optimization, and, as a result, contribute to the virtualization market and most importantly to our customers’ business expansion.

附录 B VMware vSphere 4 的服务器虚拟化——翻译

VMware vSphere 4 的服务器虚拟化

服务器虚拟市场正在迅速扩大作为客户通过分离寻求在空间和能量通过有效使用硬件资源，通过提示除了增加业务灵活性和除去服务器，和所有权（TCO）的总成本的降低的积蓄硬件和商业系统的生命周期。富士通还提供有大量的客户在日本使用的虚拟化解决方案结合了富士通的高可靠性服务器和中间件产品，优化数据中心运营，与 vSphere 4 是 VMware 的虚拟基础架构。本文介绍了 VMware 的服务器虚拟化产品多年来的转变，职能和 vSphere 4 虚拟基础架构，它关系到云环境的特点，并采用富士通中间件与 vSphere 4 的例子。

1.介绍

服务器虚拟化市场正在迅速扩大的客户希望通过硬件和业务系统的分离空间和能量通过硬件资源的更有效使用节能，通过及时添加和删除服务器更高的业务灵活性和所有权（TCO）的总成本更低生活 cycles.VMware 公司，1）总部设在加利福尼亚州帕洛阿尔托，是在全球拥有约 19 万客户和 6000 的日本客户（虚拟化软件供应商截至 2010 年 10 月）。其最近的产品线建设和运营的服务器虚拟化环境是 VMware 的 vSphere 4 和 VMware vCenter 服务器（分别称为“vSphere 4 的”和“的 vCenter”下同）.Fujitsu 开始提供 VMware 产品于 2006 年，至今已提供其客户通过这些产品与自己的高可靠性服务器以及其中间件产品为数据中心操作的优化结合虚拟化解决方案。这些解决方案由 1200 多家客户部署（如 2010 年 11 月）。富士通与这些虚拟化解决方案丰富的客户经验，并已发布了关于云系统三个案例研究，这是目前吸引了相当多的关注•内部案例 1：“VMware 的 vSphere 4 和增强的按需虚拟环境中部署虚拟主机的 SaaS 提供商“2）•内部案例 2：”把云 - 实践沼津软件开发云中心“3）•大分县案例研究：”通过 PRIMEQUEST，PRIMERGY 关键任务系统的虚拟化和 VMware vSphere 4 向公共部门云计算创造了“4）富士通的服务器虚拟化解决方案是基于一些服务器虚拟化产品（如 VMware，Hyper-V 的，RHEL-的 Xen，RHEL-KVM，和 XenServer）。从这些当中，本文介绍了 VMware 服务器虚拟化产品多年来的过渡，并介绍了 VMware 的最新产品，vSphere 4 的功能。

2.VMware 服务器虚拟化的转变

本节介绍了 VMware 的服务器虚拟化如何改变，侧重于两个方面，对“管理程序”和“虚拟化产品。”该虚拟机管理程序，偶尔被称为“虚拟机监视器”，是旨在使创建和控制的控制程序虚拟机（VM）。如图 1 所示，VMware 的虚拟机管理程序产品（图中阴影部分）经历了三个主要 phases.1 ）的 VMware GSX （ GSX ）的 GSX 管理程序运行作为一个通用操作系统的应用程序（ “主机 OS” ）演变而来硬件。

1）在此设置中，主机 OS 负责 I / O 处理，使系统管理程序本身具有紧凑构造。然而，这种类型的管理程序的从塔顶和扩展问题的困扰。像 VMware 工作站，它是最佳的 PC 虚拟化。

2）的 VMware ESX（ESX）ESX 管理程序直接运行在“裸机”，即硬件，这是目前[图 1 2）]的最流行的形式。它的功能比 GSX 更高的性能和对虚拟机管理程序的管理控制台操作系统（COS）。客

户管理是通过 vSphere Client 中完成，并添加的 vCenter 简化了多台 ESX 虚拟机管理程序的管理。因为 COS 的基于 Linux 的，它必须被频繁地修补，从而安全是因为它提出的一个挑战。

3）的 VMware ESXi 的（ESXi）之间的 ESXi，VMware 公司最新的虚拟机管理程序，有一个 COS。与 ESX，vSphere Client 将用于执行管理任务。在保持相同的功能，ESX，ESXi 的更紧凑，因为它没有 COS，易于安装和配置，并且不需要 COS 补丁（减少安全隐患）。

如上概述，VMware 的管理程序已经演变成一个紧凑和高性能的产品，易于安装的特点。

1）VMware ESX Server 的 2.xThis 产品使一个 ESX 安装的服务器（ESX 服务器）的虚拟机之间共享的资源。的多个 ESX 服务器的集中管理由 vCenter 提供。

2）的 VMware Infrastructure 3.x 的（VI3）中添加资源池的概念，VMware ESX Server 的 2.X，VI3 增强了整体功能。多个 ESX 服务器的 CPU 和内存资源，可以作为管理的资源池，使他们能够作为单一资源池进行处理。

3）的 VMware vSphere 4.xThis 产品已显著增强功能相比，这些 VI3 的。它拓宽了管理范围，包括网络资源和共享存储并支持服务器，网络和存储资源管理作为一个单一的虚拟基础架构。它能够更好地处理动态硬件改动并允许根据资源需求的变化 ESX / VM 配置的灵活修改。如上所述，vSphere 4 的代表 VMware 的服务器虚拟化产品两大流的演变：一）更加紧凑，高性能和易于安装的虚拟机管理程序和 b）加强适应性系统的变化。换句话说，vSphere 4 的适合的环境中，如在云，其中需要敏捷的创建和系统操作。

3.功能和 vSphere 4 虚拟基础架构的特点

在本章中，我们将深入的 vSphere 的功能和特性 4.vSphere 4 允许用户选择或者 ESX 或 ESXi 作为虚拟机管理程序，并创建具有以下特征的虚拟基础架构。

1）宽的管理 scopeIn 一般情况下，这种基础设施的虚拟化和池物理资源在一起，并提供虚拟机与虚拟化的 CPU，内存虚拟化，虚拟化的网络和虚拟磁盘。最重要的是，vSphere 4 的实现了动态和细致的管理，包括资源分配，配置改变，优先级控制和访问控制。

2）FlexibilityvSphere 4 提供了多种功能，如动态增加虚拟化的 CPU 和虚拟内存（热添加），虚拟磁盘和网络（热插拔），虚拟磁盘的动态扩展（热扩展虚拟磁盘）的动态重新配置的，并网络 I / O 动态负载平衡。这些功能使弹性的业务运营，灵活应对业务需求和硬件的变化，而不会中断虚拟机所提供的服务。

3）应用程序的 vSphere 4 还提供了先进的功能优势，如那些用于确保业务连续性（vMotion 和 Storage vMotion 的使资源的移动点至点[P2P]不中断虚拟机的操作），自动负载平衡（DRS：分布式资源调度程序），并在服务器发生故障（HA 自动重启虚拟机：VMware 高可用性）。由于这些功能是虚拟基础设施的一部分，也没有必要重新配置操作系统或上运行的应用程序的虚拟机，它们可被用作是。

4）综合 managementvSphere 4 提供了有用的功能，以及用于创建和运行虚拟基础广泛的功能组。有了这些功能，vSphere 4 的一直领先其他 hypervisor 的步骤。

虚拟基础架构整体由 vCenter 管理。这使得系统管理员通过 vSphere Client 图形用户界面（GUI），它显示在图标的形式，便于操作的每个管理功能来管理几乎所有的管理任务，使整个虚拟基础架构作为一个单一的环境中进行管理，就像 Windows 的控制面板。

4.vSphere 4 和云环境

以基础设施即服务（IaaS）为例（通常为贷款和虚拟机的聚集），vSphere 4 的云环境中的作用和责任提出 below.To 获得物理机的资源，用户一般通过预算的过程中，林吉和购买，因此前该设备实际部署也可能是数月（获得由循环按计划鼻祖编写的草案计划管理审批程序）。获得 VM 资源，用户通常要经过应用，内容的综述，资源调整，虚拟机创建/配置和配置的一个简单的过程，只要请求不超过可从计算池中的量的资源量。如果它不超过可用的量，用于获取物理机器资源的程序被执行。虽然这取决于资源的可用性和 IT 经理的技能，新的 VM 通常可以在几个 days.With vSphere 4 的管理层均匀性和配置更改的灵活性创建的，它是比较容易获得 VM 资源。 IaaS 的几乎自动化虚拟机创建和服务提供。然而，提供虚拟基础架构即服务和自动化虚拟基础架构并不是故事的全部，有几个困难，因为描述 below.The HA 功能，可以在云环境中，如 IaaS 的使用由在减少服务时间，提高服务水平服务器关闭。 HA 概述如图 4 所示。HA 功能通过 ESX 服务器，它通过网络交换“心跳”之间的相互监督检测的 ESX 服务器失败。当[图 4 中 1）检测到故障时，它会自动重新启动已[2）图 4]失败的 ESX 服务器上运行的另一个 ESX 服务器上的虚拟机。的可用性可接受的水平可以在系统是基于因此可以无需重新配置的虚拟操作系统上或 VM.However 运行的应用程序得到的，“相互监视，”需要一个过程，以确保所有 ESX 服务器的正常和安全停机。这样的过程包括 vSphere Client 中关机 VM 图 3 截图（OS 和每个虚拟机应用的关断），以维护模式，每个 ESX 服务器的停机 ESX 服务器的切换。换言之，此过程会影响商业任务的程序和需要使用一个相互监视机制。不遵循问题此过程的结果，如需要重启后重新配置 HA 功能。此外，系统问题导致如果关机过程未在停电发生后的时间得出结论，由于大量的部署 ESX 服务器或缺乏的不间断电源（UPS）的能力。

虽然上面看起来可能最坏的情况下,它说明了如何系统操作容易地在另一个 ESX 仅使用 vSphere 4 的可能只有假定自动化，如 cloud.2 虚拟化环境困难）虚拟机重新启动手动进行在以往的虚拟化环境通过 VMware 的高可用性功能的 ESX serversXFigure4Overview 之间相互监督检测系统故障。

鉴于其丰富的经验和相当的专业知识，富士通是配备齐全，能够处理和优化的 vSphere4 与富士通中间件相结合，提供了多种解决方案（从创建云环境来像上面描述的操作问题的解决）的。

5.结论

在 vSphere 4 的产品肯定是非常适合新兴的云时代，灵活性和敏捷性是成功的关键。Fujitsu 将继续提供支持 VMware 产品与富士通的高可靠性服务器和中间件，这对于数据中心的重要结合优化，并作为一个结果，有助于虚拟化市场，最重要的是我们的客户的业务拓展。