
电 子 科 技 大 学

UNIVERSITY OF ELECTRONIC SCIENCE AND TECHNOLOGY OF CHINA

专业学位硕士学位论文

MASTER THESIS FOR PROFESSIONAL DEGREE



论文题目	云存储监控管理平台研究与实现
专业学位类别	工程硕士
学 号	201191230107
作 者 姓 名	李勇斌
指 导 教 师	陆鑫 副教授 李新 高级工程师

分类号 _____ 密级 _____

UDC 注 1 _____

学 位 论 文

云存储监控管理平台研究与实现

李 勇 斌

指导教师	陆鑫	副教授
	电子科技大学	成都
	李新	高级工程师
	卫士通公司	成都

申请学位级别 硕士 专业学位类别 工程硕士

工程领域名称 软件工程

提交论文日期 2014. 03. 15 论文答辩日期 2014. 04. 27

学位授予单位和日期 电子科技大学 2014 年 6 月

答辩委员会主席 _____

评阅人 _____

注 1: 注明《国际十进分类法 UDC》的类号。

RESEARCH AND IMPLEMENTATION OF CLOUD STORE MONITORING AND MANAGEMENT PLATFORM

A Master Thesis Submitted to
University of Electronic Science and Technology of China

Major:	Software Engineering	
Author:	Li Yongbin	
Advisor:	Lu Xin	Li Xin
School :	School of Information and Software Engineering	

独 创 性 声 明

本人声明所呈交的学位论文是本人在导师指导下进行的研究工作及取得的研究成果。据我所知，除了文中特别加以标注和致谢的地方外，论文中不包含其他人已经发表或撰写过的研究成果，也不包含为获得电子科技大学或其它教育机构的学位或证书而使用过的材料。与我一同工作的同志对本研究所做的任何贡献均已在论文中作了明确的说明并表示谢意。

签名：_____ 日期：_____ 年 _____ 月 _____ 日

关于论文使用授权的说明

本学位论文作者完全了解电子科技大学有关保留、使用学位论文的规定，有权保留并向国家有关部门或机构送交论文的复印件和磁盘，允许论文被查阅和借阅。本人授权电子科技大学可以将学位论文的全部或部分内容编入有关数据库进行检索，可以采用影印、缩印或扫描等复制手段保存、汇编学位论文。

（保密的学位论文在解密后应遵守此规定）

签名：_____ 导师签名：_____

日期：_____ 年 _____ 月 _____ 日

摘要

在信息化时代,企业每天都会产生大量的电子数据,这些数据在企业的业务发展过程中发挥着十分重要的作用,它们被汇总、分析、转移和共享。如今,云存储系统可以为企业大数据的存储提供更为高效的数据存储解决方案。云计算实现了各种网络资源的虚拟化与集中管控,整个服务资源的调度、管理和维护工作都交由专业管理员负责。但在进行数据存储和管理时,管理员不仅需要配置大量的存储设备、网络设备,而且需要随时监控这些存储基础设施运转是否正常,以便合理地部署和调整设备,使存储系统运行在最佳状态。因此,基于云存储的监控管理平台,对提高企业云存储的服务质量具有极其重要的意义。

本文的主要工作是通过研究开源云存储平台,分析云存储平台监控管理的需求,结合云存储管理的工作流程,设计出一套能够支撑对云存储数据中心设备进行状态监控的系统,并可对外提供接口服务。系统通过代理程序采集设备状态信息,汇集到上层数据处理系统,通过数据分析,最终以图形化的方式将设备的状态信息展现出来。本文针对开源的分布式文件系统 Moose File System、网络存储设备状态监控的关键点和难点,提出了监控系统的总体设计方案。并利用 Myeclipse 开发平台以及 Webservice、Mysql 实现了基于云存储系统的监控管理平台。

目前,该系统已经投入到了实际的云数据中心监控工作中。实践证明,通过将复杂的存储设备和服务器进行集中监管,不仅可以提高数据中心管理员的工作效率和质量,还可以提高云存储系统运行的稳定性。这套系统能够降低因设备故障而导致云存储服务中断的风险,具有良好的实际意义。

关键词: 云计算, 云存储, 监控管理平台, 存储服务

ABSTRACT

At the times of information, every day enterprises produce a lot of data which play very important role in the business development process. They are gathering, analysis, transfer and sharing. Today, cloud storage system can provide more efficient data storage solutions to enterprises on large data storage. Cloud calculation can control virtual various cyber source and concentration, the whole service resource scheduling, management and maintenance are responsible for the professional manager. But in the data storage and management, administrators not only need to configure the storage equipment, a large number of network devices, but also need to monitor if these storage infrastructure is functioning properly, so that we can reasonably deploy and adjust the equipment, and the storage system also able to run under the best condition. Therefore, monitoring and management system based on storage have extremely important significance to improve the quality of enterprise cloud storage service.

The main work of this thesis is analysis of cloud storage platform monitoring and management needs through the study of open source cloud storage platform, combined with the cloud storage management work flow, design a set to be able to support system for condition monitoring of cloud storage data center equipment, and can provide a service external interface. The monitoring data will cloud storage the data server status display through the graphical approach. By the proxy program, the system collect the state information of these equipment, and convergence to the upper data processing system, and analysis these data. At last, display the equipment status information by graphically. By monitoring the policy configuration, and graphically displayed state of the data server, and used the Myeclipse development platform, Web Service and Mysql to achieve the monitoring system based on cloud storage system.

At present, the system has been put into the cloud data center .Practice has proved, the system has improved the work efficiency and quality of the data center administrators. Not only the memory device and the server complex centralized supervision, but also can improve the stability of cloud storage system operation, and reduce the risk of cloud storage service interruption due to equipment failure, which has the good practical significance. Through centralized regulation for the complex memory devices and servers, not only can improve the working efficiency and quality of data

center administrators, but also can improve the stability of cloud storage system. The system can reduce the risk of cloud storage service interruption due to equipment failure, and has the good practical significance.

Keyword: cloud computing, cloud storage, monitoring and management platform, storage server

目录

第一章	绪论	1
1.1	研究背景和意义	1
1.2	国内外发展现状	2
1.3	本文研究目标和内容	7
1.4	论文内容结构	8
第二章	相关理论与技术	9
2.1	云计算与云存储	9
2.1.1	云计算	9
2.1.2	云存储	12
2.2	分布式文件系统	13
2.3	存储系统监控技术分析	13
2.4	本章小结	15
第三章	监控管理平台需求分析	16
3.1	系统概述	16
3.2	系统总体目标	16
3.3	系统功能性需求	16
3.3.1	设备管理	17
3.3.2	网络管理	18
3.3.3	监控管理	19
3.3.4	监控策略管理	21
3.3.5	统计分析	22
3.3.6	日志管理	23
3.3.7	组织机构管理	25
3.3.8	用户管理	26
3.4	系统非功能性需求	27
3.5	本章小结	29
第四章	监控管理平台总体设计	30
4.1	设计原则	30
4.2	设计目标	30
4.3	系统架构设计	30

4.4	云存储监控服务模块设计	31
4.5	监控管理平台功能模块设计	32
4.6	系统安全设计	33
4.6.1	安全访问控制	33
4.6.2	系统安全管理	34
4.7	接口设计	34
4.7.1	链接配置接口设计	34
4.7.2	代理程序接口设计	35
4.7.3	用户管理接口设计	35
4.7.4	日志接口设计	36
4.7.5	数据库接口设计	36
4.8	数据库设计	37
4.8.1	E-R 模型设计	37
4.8.2	数据库表结构设计	39
4.9	本章小结	42
第五章	监控管理平台详细设计与实现	43
5.1	系统实现概述	43
5.2	系统开发环境	44
5.3	功能模块详细设计	44
5.3.1	初始化	44
5.3.2	配置管理	45
5.3.3	设备管理	46
5.3.4	拓扑管理	47
5.3.5	监控告警管理	48
5.3.6	统计分析	49
5.3.7	日志管理	49
5.4	接口详细设计	49
5.4.1	链接配置接口设计	50
5.4.2	代理程序接口设计	51
5.4.3	用户管理接口设计	53
5.4.4	日志接口设计	55
5.4.5	数据库接口设计	55
5.5	部分系统功能模块实现	55

5.5.1	MFS 文件系统的配置过程.....	55
5.5.2	监控管理模块界面设计及关键代码	59
5.5.3	监控代理模块关键代码	61
5.5.4	监控服务端关键代码	64
5.6	本章小结	65
第六章	系统测试	66
6.1	测试方法	66
6.1.1	测试工具	66
6.1.2	测试模型	66
6.2	测试环境	67
6.3	测试内容	68
6.3.1	功能测试	68
6.3.2	性能测试	71
6.4	测试结论	72
6.5	本章小结	73
第七章	总结与展望	74
7.1	总结	74
7.2	展望	75
致 谢	76
参考文献	77

第一章 绪论

本章将首先介绍云计算技术发展历史和意义，同时分析制约云计算发展的因素，分析得出云数据中心运营所面临的一个关键性问题是能够有效地对云基础设施进行管理，探讨建立一个智能化的云存储监控管理平台的理论意义与现实意义，并将对国内外相关的产品和技术研究现状进行简要介绍，引出课题研究的主要问题 and 解决的思路，最后将对本次论文的各章节主要内容做出安排。

1.1 研究背景和意义

随着信息化时代的高速发展，我们的生活已步入大数据时代。企业因发展的需要，已纷纷建立自己的数据中心，但随着信息系统产生的电子数日益增多，数据中心需要增加和部署的存储设备呈爆发式增长，日常维护和管理这些存储设备，已让管理员们手忙脚乱，由此给企业带来的管理成本也陡然增加。要解决这样的发展矛盾，必须找一种高效的大数据存储解决方案，一方面是要能为大数据的良好运行提供保障，另一方面又要具备更快、更及时地对这些大数据进行分析与处理的能力。由此，云计算的概念就应运而生了。

“云计算”这个概念最早诞生是由麦卡锡于上世纪 60 年代提出来的，他提出了把计算能力转变为一种像水和电一样的公共服务提供给用户的思想，这个被大众认为是云计算思想的起源^[1]。而“云计算”概念正是的提出是 Google 公司在 2007 年正式提出的，该公司提出了一种基于互联网的新型 IT 基础设施的交付和使用模式，云计算不再是一种技术，而被认为是一种商业模式，即云计算的服务者将按需提供服务，并按需向云计算提供商付费。云计算最大的特点是云端的资源可以随时随地被访问、可以按需获取、可以随时扩展、用户只用按需付费等^[2]。

从技术发展的角度来看，云计算是对分布式计算、并行计算和网格计算的传承和拓展。云计算概念一经提出，便引起了 IT 从业者、互联网企业等等社会各界极大的关注，因为看到了云计算将给当今的社会、经济发展模式带来的许多变革，所以世界各国纷纷开始关注、了解、研究云计算，在各国的信息化战略规划中已经将其列入其中^[3]。

与此同时，云计算也在这几年得到了快速发展。首先是传统的 IT 领航企业，微软、谷歌公司都已把云计算确定为未来企业发展的核心，陆续推出自己的云战略布局和规划。2011 年，欧盟委员会将云计算技术的推广作为“数字化议程”的重要组成部分列入了《欧洲 2020 战略》^[4]。为了扶持云计算产业的快速发展，美

国政府也颁布了一项“云计算优先”的政策。而我国政府在云计算上在配套政策也在逐步完善，现在政府已经把云计算作为下一代信息技术产业发展的重要组成部分，在政府发布的“十二五”规划纲要和《国务院关于加快培育和发展战略性新兴产业的决定》^[3]中已经明确出来，云计算技术将在未来的战略性新兴产业中占有举足轻重的作用。应该说，如今云计算技术已进入政府、教育、金融等民生行业，与我们的生活中息息相关。

然而，目前有许多因素制约和限制了云计算的发展，如云计算相关的安全性问题、管理问题、知识产权问题等等。在云计算环境中，基础设施中的硬件资源将不再被最终用户控制，应用软件都运行在云端，个人或企业数据也将存储在云端，用户失去了对 IT 资产在物理上的控制，当业务系统从传统的烟囱式的建设模式中迁移到资源池模式的云计算系统中后，将形成超大规模的数据中心^[5]。这就要求管理系统需要能够支撑成千上万的计算节点，并且能够实现智能化的管理，以降低数据中心管理者的数量，减少运营成本和工作量。

事实上，对于任何 IT 系统，管理是其良性运行的关键和重要保障，任何 IT 设备都拥有自己的管理系统。对于大规模的数据中心，更需要通过集中式的管理系统来运行管理应用、存储、网络等系统和设备，以便能够快速监控、响应和处理数据中心的异常事件、业务变更或进行持续优化。

近几年，在云计算的发展和演变过程中，相关云数据中心的服务中断事故已经多次发生。2010 年 3 月，Terremark 的 vCloudExpress 服务所在的迈阿密的数据中心断网了大约 7 个小时，在这段时间里，用户不能访问存储在这个数据中心的数据^[6]。2011 年 4 月，互联网电子商务巨头企业——亚马逊公司在美国北弗吉尼亚州的云数据中心出现故障，所提供的 Web 服务出现无法使用的情况，整个故障持续了超过 96 个小时，导致许多企业陷入暂停工作的困境之中^[6]。

从这些案例中，我们不难看出，云数据中心如果没有一套易操作、直观、智能的管理系统，用以实时监控和管理云服务及设备的运行状况，等到事故发生后，再来挽救将是得不偿失的事情。因此，在云平台上开展相关监控与管理系统的研究，具有重要的现实意义。

1.2 国内外发展现状

云计算技术的出现给 IT 产业界带来了一场伟大的技术革命，云计算将引领 IT 行业未来的发展方向。全球经济危机的出现，以及哥本哈根国际气候大会的决议推动了产业升级、调整的步伐，低碳生活的时代将是世界共同的期盼^[7]。因云计算技术很好地解决了高效型 IT、绿色低碳型 IT，所以受到了社会和国政府的极大关

注和重视，在各种需求、技术和政策的驱动下，在众多互联网公司规模化发展的推动下，云计算技术得到了极大的发展和壮大。目前，国内外的互联网厂商们纷纷开始向云计算平台转移，包括 Google、微软、IBM、Amazon、Yahoo 这些国外互联网巨头，也包括阿里巴巴、百度、华为、腾讯、浪潮等国内知名互联网和 IT 企业都成为了云计算行业的先行者。本节将对国内外云计算的发展现状进行介绍。

1) 云计算在国外的发展现状

Google 公司最先提出了云计算的概念，Google 的云计算技术是基于他特定的网络应用程序而定制的^[8]。Google 的各种应用服务使得其内部的数据结构规模非常庞大，因此他提出了一整套基于分布式并行集群方式的基础架构，该架构的特点是可以充分利用软件的能力来处理集群中经常发生的数据节点失效的问题。Google 的云计算主要由编程模型 MapReduce、分布式文件系统 GFS（存储系统）、数据管理系统 BigTable 三大部分组成^[9]。作为目前最大的云计算服务的使用者，Google 已经将 100 多万台云计算服务器部署在全球 200 多个城市^[6]，Google 的许多应用服务项目，包括 Google 地球、地图、Gmail、Docs 等也同样正在使用这些基础设施。如此庞大的服务器群，需要强大的管理系统协助管理者们做好系统的监控和管理工作，而 Google 云计算的管理系统需开发者根据 MapReduce 开发模型进行编程，以实现云计算基础设施资源的调度与管理。

亚马逊（Amazon）作为互联网上最大的在线零售商，是第一个互联网公有云服务提供商。亚马逊将自己的云计算平台称为弹性计算云 (ElasticComputeCloud, EC2)，这个云计算平台包括弹性计算云（EC2）和简单存储服务（S3），通过这个平台为企业用户提供存储和计算服务，目前可以提供的服务的项目包括存储服务器、带宽、CPU 资源以及月租费，企业根据各自需要按需付费即可^[10]。亚马逊从 2006 年开始，就以 Web Server（AWS）的形式发布云服务，提供 IT 基础设施服务，用了不到两年时间，目前 AWS 已成亚马逊的主要业务之一。AWS 具有一套非常强大的用户管理控制系统 Manage Console，操作简单、功能全面、集成度高，从监控信息表上可以看到云服务系统的状态信息，但直观性不够。其状态监控实例如图 1-1 所示^[11]。

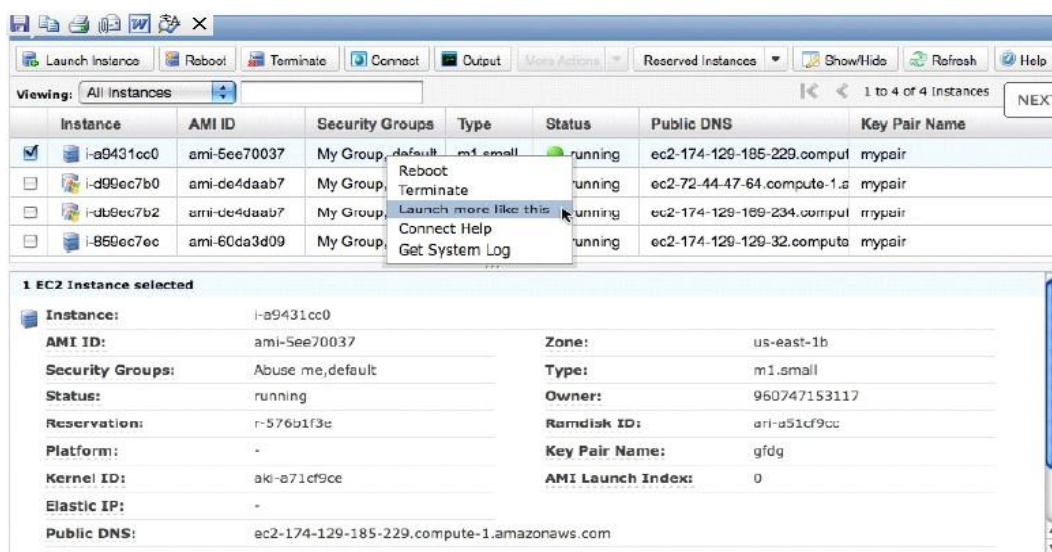


图 1-1 Amazon AWS 之管理监控实例

IBM 也是进入云计算领域较早的企业，他们在 2007 年底推出了自己的“蓝云（Blue Cloud）”计算平台，该平台实现的目标是为客户带来“即买即用”的云计算服务^[1]。IBM 云计算中的存储体系结构式是基于开源软件 Hadoop HDFS (Hadoop Distributed File System) 定制开发的，该平台上包含了一系列的虚拟化软件，功能主要包括自动化、自我管理和自我修复等等，实现的存储域结构模式是基于块模式的网络 SAN 结构，这种模式可以为分布在全球的应用系统提供分布式的资源池。近年来，访问量呈现爆发式增长，于是 IBM 先后投资数亿美元用于对部署在北卡罗来纳州和日本东京的云计算数据中心进行改造^[12]，以此来满足访问需求。IBM 云计算提供的名为“SmartCloud”管理系统实现了云计算存储资源、网络资源和应用资源的智能监控、分析，实现对系统故障问题预测，并在客户受到影响之前，预测到具体问题，并及时告知管理员解决，该方案使云服务停运几率减少了 30%到 40%。使用“SmartCloud”管理系统达到了非常好的效果，该系统不仅可以将被监管的用户统一集中到一个中心监控平台之上，实现流程和管理集中化，而且大大降低了数据中心人力资源的投入，IBM 云计算的监控实例如图 1-2 所示^[11]。

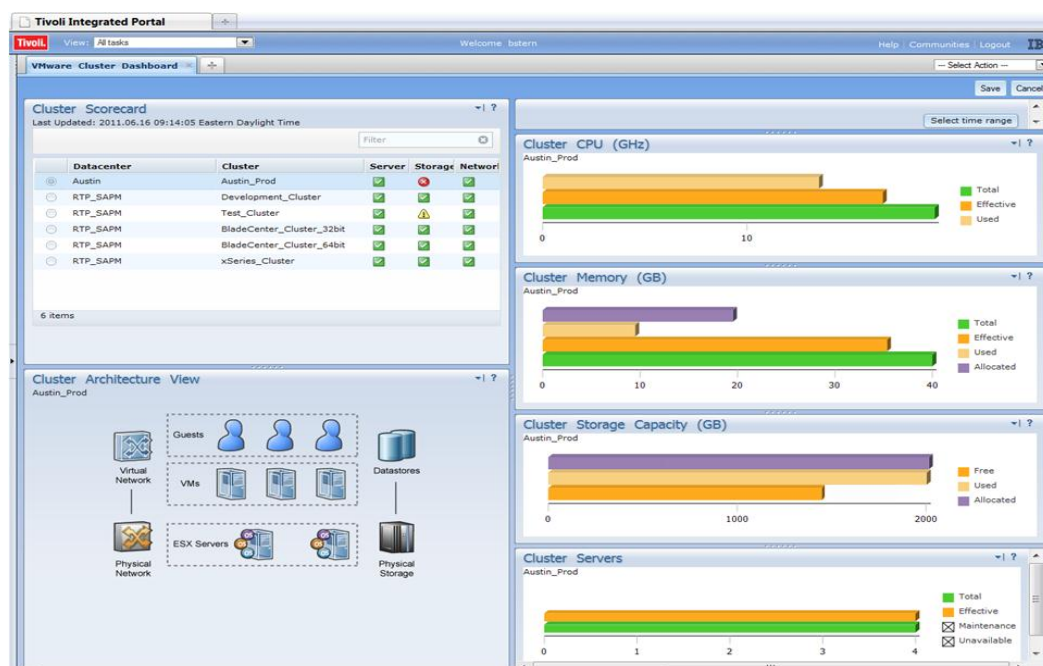


图 1-2 IBM 云计算之管理监控实例

微软作为全球最大的操作系统与应用软件提供商，虽然进入云计算领域的时间相对较晚，但是目前拥有全世界数以亿计的 Windows 用户桌面和浏览器用户，现在将它们迁移到云服务上，微软将占有巨大优势。2008 年 10 月，微软正式推出了 Windows Azure 操作系统，Azure 的底层是微软全球基础服务系统，由遍布全球的第四代数据中心构成，微软云计算平台包含三个组件：Windows Azure、SQL Azure、Windows Azure Platform AppFabric^[13]。微软的云计算数据中心由很多组 Fabric 组成，每一台服务器运行一个 Fabric 代理程序。Fabric 中心控制器通过和每台服务器上的 Fabric 代理进行数据交互，达到对每个应用程序的控制和监视的目的。作为开发者可以不用再关心主机的管理、软件的管理等问题，微软作为云服务提供商将负责维护，所以开发者们可以集中精力去完成应用程序的开发。但由于微软云计算代码相对封闭，云计算管理接口封闭等问题，给学习者带来很多麻烦。

2) 云计算在国内的发展现状

云计算产业在我国的发展也非常迅速，随着云计算技术应用需求的不断增强，云计算产业的发展也受到了国家层面的重视，国家主要部委陆续制定并颁布了一系列的云计算产业发展引导政策，各地政府也相对地出台了配套的云计算产业发展专项支撑政策^[14]。在政策的激励和推动下，国内互联网和 IT 企业也在大力推进云计算的研究和发展，也有部分企业推出了一些云计算基础架构产品，但产品的可靠性、兼容性、开放性问题较突出。

阿里巴巴推出的阿里云，是国内公有云比较成功的案例。阿里云是阿里巴巴公司自主研发的分布式计算平台，其核心命名为飞天操作系统（代号“Apsara”）。该产品通过对软件系统的合理分层，使得“强调应用速度的在线服务”和“强调处理数据吞吐量的离线任务”可以共享一个物理集群的计算、存储和网络资源，以期提高大规模服务器集群的效率^[14]。阿里云主要是为中小企业提供云计算服务，目前可以提供的服务包括：客户域名注册、网站建设、存储空间租赁到云主机，云邮箱和私有云。目前，已有超过 50 万的中小型企业使用阿里云^[3]。

新浪云计算 Sina App Engine (SAE)，作为国内推出较早的公有云，主要吸收和借鉴了 Google、Amazon 等国外公司在公有云上的成功经验。SAE 主要提供：云存储服务 and 云计算服务。架构上采用了分层设计思想，由上往下分别为反向代理层、路由逻辑层、Web 计算服务层^[14]。SAE 是分布式 Web 服务的开发和运行平台，无需用户实现设备的运维管理。

浪潮公司推出的云海 OS 系统，是第一款国产的面向数据中心的操作系统。该系统采用开源技术路线，可以支持分布式计算和分布式存储。云海 OS 在管理体系上采用了多级联的管理体系，系统通过级联方式实现计算和存储资源的有效整合，在满足客户需求的同时，可以有效提高资源利用率。

总体而言，从发展进度来看，国外无论是传统的 IT 厂商、互联网产品提供商，还是软件厂商，都已纷纷转型或即将转型为云计算服务或产品提供商，他们已走在了国内企业的前面，目前国内企业对云计算的研究和发展还处于初级发展阶段。从采用的技术架构来看，开源云计算平台以其开源免费、可修改性强的特点，受到了越来越多的企业和开发者的青睐。分析其原因是因为开源云平台具有以下优势：

1) 系统更加灵活

开源云相比于专有云平台具有更高的灵活性，使用者可以对源码进行更改或重构，开源平台有许多免费公开的学习资源，可以让用户更深入的参与和交流。

2) 不易被垄断

开源云平台的目的就是避免被单一的厂商锁定，采用开源技术的云平台，将不会被任何一家厂商控制，客户可以根据各自的需求自由选择供应商，而不是被一个专有的解决方案卡死。

3) API 可开放

在开源云平台中的应用程序编程接口（API）并不会被一家厂商控制，而将都会采用开源的技术标准。这使得客户可以在开源基础上自由开发出与现有解决方

案完美结合的各类接口。

4) 低成本

相对于商业软件，开源平台在成本控制上具有很好的吸引力，开源平台可以提供大量的免费软件，可以为企业和用户节约较大的成本。

5) 可移植性高

当用户基于一套开放的云平台源码开发的时候，必须要看它如何与其他公有云、私有云和混合云很好的集成，开放云平台允许在广泛的云生态系统上选择移植，具有很强的系统兼容性。

综上所述，我们可以看到，云计算技术的发展将受到云服务的可靠性、实用性、安全性等方面的影响。云基础设施的监控与管理，将是云计算服务供应商必不可少的工具，它将对云平台的稳定运行发挥重要作用。

1.3 本文研究目标和内容

本课题的研究目标是基于云存储平台的多种应用场景，以及云存储平台的调度与管理机制进行研究，选择一种主流开源的分布式文件系统，设计并实现一套支撑云存储服务的监控与管理平台。通过云存储基础设施管理平台，实现对云存储基础设施监控信息的统一监控与告警，同时提供关键设备主机、存储设备等 IT 资源的运行状态库和配置信息库，快速掌控资源池的运行情况，帮助运维人员快速定位故障，有效解决当前云存储服务面临的监管对象规模大、可靠性要求高、资源利用率低等问题。具体工作开展如下：

首先，将通过对云存储平台的应用场景和运维管理进行需求分析，确定影响云存储稳定运行的关键因素，并针对开源的分布式文件系统进行研究，分析提取系统运行相关参数（如 CPU、内存、磁盘空间和网络状态等）的监控方法，总结出对应的监控策略，然后结合分布式监控技术进行监控管理平台架构的总体设计和各功能模块的详细设计，包括数据库设计、接口设计等。

其次，在详细设计的基础上，本次研究工作还将对系统进行实现，以进一步验证设计的正确性和可行性。同时，结合云存储数据中心的实际工作情，充分挖掘该系统研究工作的应用价值。

最后，本文将对云存储监控管理平台所进行的研究工作进行总结，对整个设计与实现过程中所遇到的各种问题和所用到的解决办法进行小结，总结出成功解决问题积累的宝贵经验。并将本次未能解决的问题和需要完善的问题作为后续工作，考虑在今后适当的时机、适当的技术条件下进行更进一步的研究工作。

1.4 论文内容结构

第一章 绪论。介绍本课题的研究背景、意义，并通过对目前国内外云计算主要厂商在云计算领域的研究发展现状进行分析。总结出了云计算的监控管理在云计算服务中将起到至关重要的作用，确认了本文研究工作的意义所在。

第二章 相关理论与技术。主要介绍与本文研究相关的知识，首先介绍云计算和云存储的基本概念，以及发展现状，接着介绍分布式文件系统 MFS 的特点及现在的应用情况，最后对存储系统监控管理技术进行分析，对比传统的管理模式，找到适合开源云存储平台的监控管理方法。

第三章 监控管理平台需求分析。本章将从监控管理平台的功能需求、性能需求两方面对监控平台进行全面分析，梳理出监控管理平台需要实现的功能点，以及在性能方面需要达到的技术指标。

第四章 监控管理平台总体设计。将根据系统设计原则对云存储监控管理平台进行总体设计。介绍管理平台的系统架构设计与功能模块设计。

第五章 监控管理平台详细设计与实现。将根据系统应用软件设计原则对云存储监控管理平台进行详细的系统设计。本次论文还将在系统详细设计的基础上，对系统进行功能实现。

第六章 系统测试。本章将对云存储监控管理平台进行测试，以验证平台的功能和性能是否达到设计目标。

第七章 总结与展望。本章将对本次研究工作进行总结，归纳在研究过程中所发现的问题以及在发现问题和解决问题时获得的经验。

第二章 相关理论与技术

本章将围绕云存储及平台监控管理相关的理论和技术进行介绍。首先将介绍云计算相关概念,然后重点介绍云计算的技术系架构,并对目前的主机系统监控现状进行介绍。

2.1 云计算与云存储

2.1.1 云计算

从“云计算”这个概念诞生以来,业界都没有一种十分准确的定义来解释什么是真正的云计算,也因此“云计算”可以说是一个不断发展的术语,它描述了许多现有技术和计算不同东西的方法的发展。而“云存储”可以视作云计算的各分支或者一个必不可少的组成部分。

具体来讲,云强化了协作性、敏捷性、可扩展性和可用性,并通过优化和有效计算提供潜在的成本缩小,云从底层基础设施和使用的传输机制,分离了应用程序和信息资源^[2]。云计算重点描述了由网络、信息、计算等资源所组成的应用、服务和基础设施等的使用方法,这些云计算相关的组件可以迅速策划、部署和解散,也可以实现快速地扩展或缩减,提供类似日常生活中的公共设施的按需分配、高效计算等消费模式。

云计算其实是一个很宽泛的概念,不同学者有不一样的理解,但归纳起来,所谓“云计算”它该应具有以下的几个共性特征值:

- 1) 利用网络,并以网络为中心,以动态、按需的服务模式来获取所需要的各类资源(包括计算资源、存储资源、网络资源),提供类似资源的平台可被称为“云”。
- 2) 系统应具有高扩展性、高可靠性、虚拟化等特点。

美国 NIST(国家信息标准研究院)定义出了云计算的五个关键特征^[15],如图 2-1 所示:

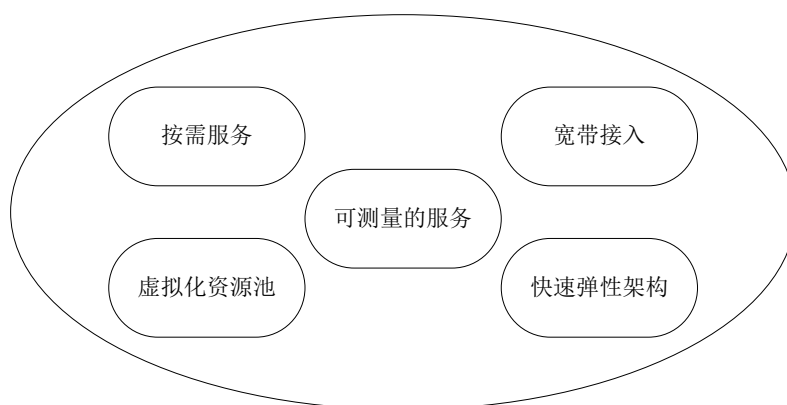


图 2-1 云计算的五个关键特征

云计算的五个关键基本特征包括了:按需服务、虚拟化资源池、可测量的服务、宽带接入和快速弹性架构。根据这五个特点，我们可以看到云计算可为用户提供按需分配的计算、存储和应用资源，在提供使用便捷的同时，可降低用户在硬件上的费用支出。

综上，根据云计算的服务层次和应用特点，可将云计算划分为三种服务模型^[1]：软件即服务(Software as a Service)、平台即服务(Platform as a Service)和基础设施即服务(Infrastructure as a Service)。下面将对这三种服务模型进行介绍：

1) 软件即服务

软件即服务是基于网络的方式提供软件服务的应用模式，通过这种模式，终端访问用户，只需要通过浏览器，就能直接访问并使用到部署在云端上的应用系统和资源，而自己不再需要单独购买软硬件。软件即服务主要面向服务群体是个人（普通）用户，这种服务模式不受时间、地点的限制。

SaaS 供应商提供的主要产品有：Salesforce Sales Cloud, Google Apps, Zimbra, Zoho 和 IBM Lotus Live 等。

2) 平台即服务

通过平台即服务这种模式，用户不仅可以在一个开发平台上非常便捷地编写应用自己的程序，这个平台包括 SDK、文档和测试环境等，而且在部署或运行的时候，原来那些诸如服务器、操作系统、网络 and 存储等资源的管理问题都交由云服务提供商负责处理。平台即服务主要的用户群是开发人员，在这种模式下，云服务可以实现精细的管理和监控，例如，能够监控到每个云端上的应用系统实时的运行情况，并可以通过自动化的分析，判断系统当前的运行状态。

主要产品包括：Google App Engine, force.com, heroku 和 Windows Azure Platform 等。

3) 基础设施即服务

这是云服务可提供的最基本的服务，通过基础设施即服务模式，用户不仅可以获得他所需要的虚拟机或者存储空间等基础资源，而且用户再也不必关心这些服务器等设备的管理与维护，繁琐的工作都交给云供应商来完成。基础设施即服务可以通过对云端资源的实时监控与智能化管理，减少系统运行风险，保证系统持续、高效率的对外提供基础服务。

主要产品包括有：Amazon EC2, Linode, Joyent, Rackspace, IBM Blue Cloud 和 Cisco UCS 等。

从服务模型的特点来看，云计算的三种服务模型并不是相互独立的，相互之间具有一定的依赖关系，云计算服务模型的框架图如图 2-2 所示^[16]。

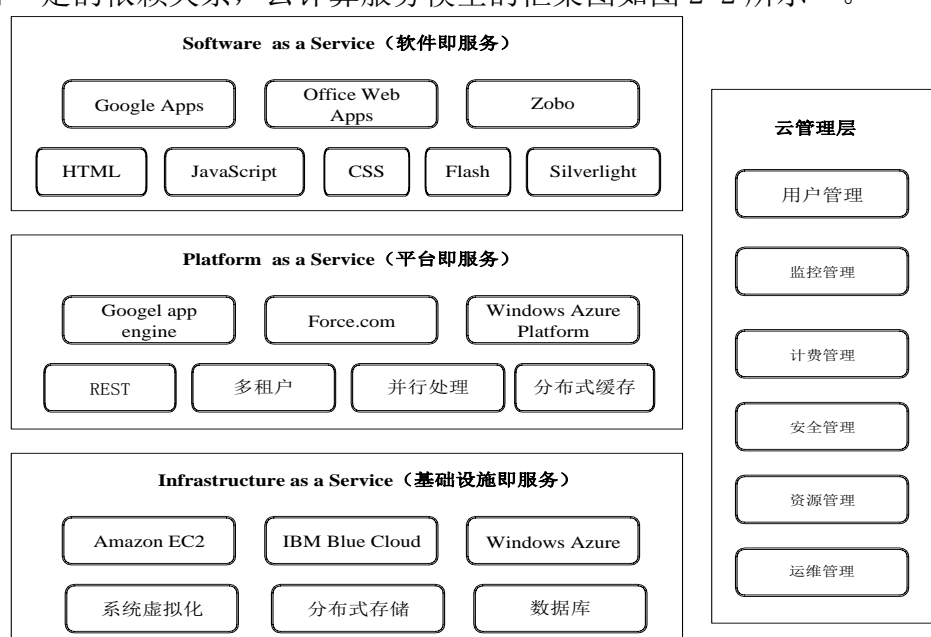


图 2-2 云计算服务框架图

从上图可以看到，一个 SaaS 层的产品和服务不仅需要使用到 SaaS 层本身的技术，而且还需要依赖到 PaaS 层所提供的平台，也可能还需要用到 IaaS 层所提供的计算资源，而 PaaS 层的产品和服务也需要依赖于 IaaS 层提供的服务。所以，从服务模型的特点可以看出，三者之间存在一定的依赖关系。而从管理的角度而言，云计算的各种服务主要依靠以云的管理层所提供的服务，云计算的管理包括用户管理、监控管理、计费管理、安全管理、资源管理和运维管理等等，管理的主要目的是要确保整个云计算中心能够安全、稳定地运行，并且能够将云计算的各类资源进行有效的监管，最终实现云服务的可靠性和有效性。

2.1.2 云存储

实际上,云存储的概念是在云计算的概率基础之上发展而来的。云存储技术融合了传统的并行计算与网格计算,并将虚拟化技术进行了进一步的扩展,其核心是分布式存储^[17]。云存储技术可以将网络中大量异构类型的存储设备统一起来协同工作,协调各存储设备的数据处理逻辑,以达到为上层系统提供远程存储服务的效果。因此,当云计算系统在需要处理大量数据的存储和管理任务时,云计算系统的核心工作就变为了云存储,此时数据中心需要配置更大容量的存储服务器或磁盘阵列设备。所以云存储系统是对云计算的延伸,其核心是提供数据存储服务和管理。

对于云服务提供商而言,云存储是原有不同类型、不同架构的存储服务器的有效整合与资源利用,也是对网络存储技术的发展与创新。目前,市场上主流的存储服务厂商所采用的云存储架构模式有两大类型,分别是网络存储架构和分布式集群存储架构。

网络存储架构服务的目标是将分散部署的存储设备,通过共享网络协议,实现一个集中统一管理的存储系统^[18]。这里指的存储设备可以是基于 NAS、iSCSI 等类型的 IP 存储设备,可以是新型的光纤(FC)通道存储设备,也可以是 SAS 或 SCSI 等 DAS 类型的存储设备。但是,基于硬件的网络存储模式还是没有解决大数据存储时面临的容量和性能扩展的瓶颈,不能满足未来云计算发展对数据存储的需求。

分布式集群存储架构的特点刚好可以解决基于硬件的网络存储在扩展模式上的瓶颈^[19]。存储系统只需要依靠分布式文件系统,无需通过存储设备去专门构建 SAN 模型结构,这种模式不但能够对异构模型的设备做到很好的支撑,而且在系统需要进行扩容时,它依然可以通过特有算法来实现易用、易扩展的目的。目前,分布式文件系统已经得到了很好的发展和创新,主流的分布式文件系统很多,比如 Googal 公司的 GFS 系统,Apache 基金会所开发的 Hadoop 分布式文件系统,以及一些轻量级的,如 Moose File System(MFS)等。虽然分布式文件系统本身已经得到了很好的发展,但核心之一的管理问题,依然面临一些不足之处,比如:

- 1) 云存储必须有效地兼容不同厂商的、不同平台的软硬件基础资源,实现新老资源的平滑过渡;
- 2) 如何提高云存储服务的有效监管,降低系统运行风险带来的服务问题。

对于使用云存储的用户而言,云存储不再仅仅是一种计算模式,而是变为了一种常态化的存储服务。因此,我们的云存储服务供应商需要解决云服务的安全性和可靠性,即如何保证服务数据的安全性、服务的稳定和可靠性,由此可见,建立一套智能化、高效的云监控管理平台对实现稳定的云存储服务至关重要。

2.2 分布式文件系统

以前企业在进行数据中心存储解决方案设计时，往往首先会选择采用“共享存储服务器”，即常说的 NFS 存储的方式来对大数据进行网络存储，经过长期的应用实践证明，这种方案暴露出了很大的问题，不能解决其性能瓶颈和设备的单点故障，一旦数据存储设备出现问题，整个存储服务就只能中断。随着云计算的普及，如今企业开始把目光转向云存储技术，逐步开始构建自己企业的云存储中心。云存储系统的核心正是分布式文件存储系统，分布式文件系统是整个云存储的基石，他要提供上层应用系统所需的数据存储服务。

目前主流的分布式文件系统以 GFS 和 HDFS 为代表，轻量级的以 MFS 为代表^[20]。本文中所研究的对象将选择轻量级的分布式文件系统 Moose File System（简称“MFS”），MFS 文件系统本身是一个具备较强容错功能的网络分布式文件系统，它将存储数据分布在网络中的不同服务器上，MFS 通过 fuse 使之看起来就是一个 Unix 的文件系统。MFS 分布式文件系统在云存储领域的应用具有较多的优势，比如：

- 1) MFS 属于通用文件系统，适应性较好，无需修改上层应用就可以使用。
- 2) 可以实现在线动态扩容，体系架构可伸缩性强，符合云计算机的特点。
- 3) 体系架构高可用，部署简单，系统无单点故障。
- 4) 文件对象高可用，可设置任意的文件冗余程度。
- 5) 可以设置删除文件的空间回收时间。
- 6) 提供 netapp, emc, ibm 等商业存储的 snapshot 特性。
- 7) 提供 web gui 监控接口。
- 8) 支持较多的平台，包括 Linux、FreeBSD、NetBSD、OpenSolaris 等等。

MFS 分布式文件系统原有的管理接口只提供了针对磁盘文件数、存储空间、磁盘读写状态等基本信息的管理，而承载这套系统的服务器状态如何管理和维护，将是摆在管理员面前的难题，当数据中心规模扩大之后，这样的管理和维护工作量将会呈爆炸式增长的趋势。因此，设计一套可用性高的云存储监控管理平台具有重要意义。

2.3 存储系统监控技术分析

在计算机和互联网技术高速发展的过程中，信息系统的监控管理技术从最开始的模拟信息传输与控制，发展到现在基于网络化的信息传输与控制方法。信息系统的监控管理技术最初是利用各种仪表集中式地对系统或设备状态进行监控，并通过操控台来进行操作，各设备之间没有关联性，对采集到的监控数据没有做

智能分析，对参数异常的判断只能交给管理员来完成。大数据时代来临之后，企业每天产生的数据量快速增加，数据中心的存储设备的也不断增多，系统监控的对象开始从原来简单的变为多样化的、功能的复杂的。因此，传统的监控系统设计思路已经不能满足云存储系统精细化的管理要求。

随着网络和信息技术的发展，以及各种数据中心的不断建设，建立在存储系统之上的监控系统也从中心集中管控发展到了网络化的集中监测。可以说是网络技术的发展推动了监控技术的发展，如今这些开放式的网络技术已经被应用到了很多行业系统的监控管理系统中，不仅实现了资源监控，还可以实现系统的智能管理^[21]。云计算时代需要更加智能化的监控管理技术，系统需要建立在计算机技术、网络通讯技术、控制技术以及图形技术上，需要将信息的传输、信息的监控和管理进行集成，在实现各类信息共享的同时，还要实现监控的时效性，并且要能非常直观、智能地将设备和系统的运行故障或风险提前告知管理员，向监控管理平台的管理者提供一个更全面、更有效、更安全的监控管理服务，改变传统的监控模式^[22]。对云存储系统的资源实现监控管理，目的是基于云存储平台的基础架构，利用网络监控技术实现各种主机、存储设备和网络等相关资源的信息库和配置信息库，建立云存储基础资源的统一故障、告警信息监管平台，快速掌握告警信息所发生的位置或设备。

国际标准化组织（ISO）定义的网络管理系统主要包括五大功能域^[23]，分别是：故障管理、配置管理、性能管理、计费管理和安全管理。每个功能域负责管理一组活动，完成不同的网络管理功能，并且这五个域之间不能相互独立，而是要求可以相互通信、相互协作。这五大功能域之间的关系^[24]如图 2-3 所示：

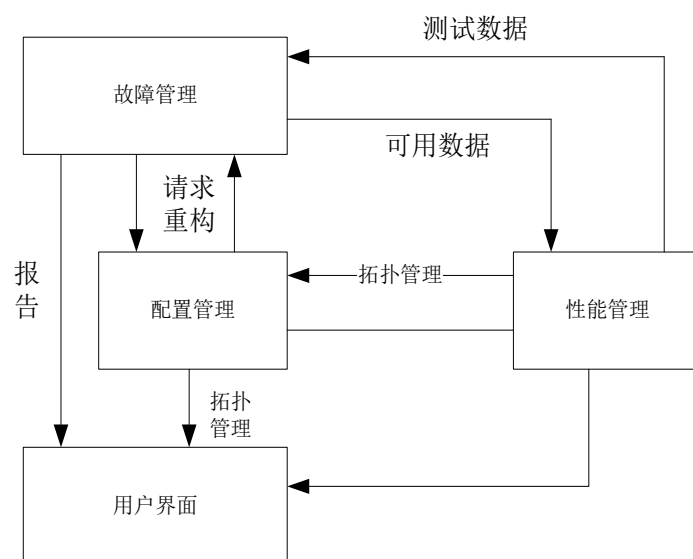


图 2-3 系统网络管理的功能域关系图

故障管理：是指网络管理系统可以在运维的系统出现故障时，系统能够迅速定位到被监控系统的故障点。

配置管理：是指网络管理系统可以通过对网络进行初始化和配置，来提供网络服务。

性能管理：是指网络管理系统可以通过对系统资源的运行状况以及通信效率等资源信息来进行系统评估，以维护良好的网络服务质量和运营效率。

计费管理：是指网络管理系统可以通过对网络的使用情况进行统计和记录，来实现对网络资源的控制 and 操作进行计算，并根据用户对资源的使用情况进行计费。

安全管理：是网络管理中十分薄弱的部分，主要任务是要控制对网络资源的访问，防止网络信息找到恶意攻击和修改，保护敏感信息部泄露。

随着云计算技术和应用的飞速发展，云服务相关的管理系统将发挥更加重要的作用。智能化是未来云计算管理系统的主要趋势，管理系统能够具备自动获取各种设备或系统运行参数的能力^[25]，并实现对数据的分析，能够在故障出现以前发出告警信息或者进行自动修复。同时，也要考虑到管理系统的易用性和可操作性，一套简单、方便、直观的系统网络管理平台更加有利于网络的控制，方便配置管理、状态监控，使得管理效率更加高效。

2.4 本章小结

本章相关理论与技术，首先介绍了云计算与云存储的关系和差别，然后重点对本课题选用的分布式文件系统进行了介绍，分析了开源 MFS 的特点，并系统已有的管理功能的缺点和不足进行了分析；最后对传统的系统监控技术进行了分析，然后对云存储系统所需要的监控和未来智能化的管理平台所要达到的效果进行了分析，指出了监控管理平台的功能要点。

下一章节监控管理平台系统需求分析，重点从系统总体目标、系统功能性需求和非功能性需求等方面对监控管理平台进行了分析，明确了工作目标。

第三章 监控管理平台需求分析

本章节重点对云存储监控管理平台需要实现的功能和性能进行需求进行分析，系统需求包括功能需求和非功能需求。功能需求方面，重点围绕云基础设施如何才能更好的支撑各项业务系统的正常运行，如何才能监控并管理动态的 IT 基础结构、事件、网络以及高虚拟化环境。非功能性需求方面，主要分析监控管理平台的稳定性和可用性。

3.1 系统概述

云存储监控管理平台的目的是实现对云存储基础设施的统一监控与管理，将为云存储系统的更深层次管理提供基础管理支撑，监控部分将为管理员提供对云存储分布式文件系统相关的资源服务器的各类资源信息，这类资源信息主要包括系统的 CPU 利用率、内存利用率、网络传输速率等信息，也包括与云存储服务相关的主要设备主机的运行状态库和配置信息库，快速掌控资源池的运行情况，帮助运维人员快速定位故障，有效解决当前云存储服务面临的监管对象规模大、可靠性要求高、资源利用率低等问题。

3.2 系统总体目标

本系统的总体目标是基于开源的分布式文件系统 MFS，设计并实现基于开源架构的云存储监控与管理平台，实现对云存储基础设施运行状态和信息的统一监控与管理，使管理者或系统管理员能够通过可视化的监控管理平台，将云存储 IT 资源池相关的服务器、应用系统等状态信息进行集中式的监控，实现高效的监控管理。通过不同告警阈值的配置，使各类系统在故障风险较大的时候，可以依靠声音、颜色或者短信等方式向系统管理员报警通知，以便能够让系统管理者及时、准确地排除系统险情或故障，确保云环境下的每个系统能提供持久、稳定的应用服务。以各种监控、告警、日志、报告服务工具为依托的云存储监控管理平台，可实现全网的统一监控管理，打破传统的本地化管理模式，减少管理者的工作量，降低企业运维成本。

3.3 系统功能性需求

基于 MooseFS 分布式文件系统的监控管理平台的功能性需求主要包括：设备

信息的管理、设备状态的管理、网络状态管理、监控策略的管理、云存储磁盘空间的管理、组织机构管理、用户管理等等。

3.3.1 设备管理

设备管理功能，即对接入云存储基础设施的服务器和存储设备进行统一管理。包括设备属性管理、设备注册入网管理等，设备管理是实现云存储基础监控与管理的前提。设备属性管理功能主要由系统管理员完成操作，包括设备属性的新增、修改、删除和查看功能。下面给出了设备属性管理的用例图和用例表详细描述。系统的设备属性管理功能用例图如图 3-1 所示：

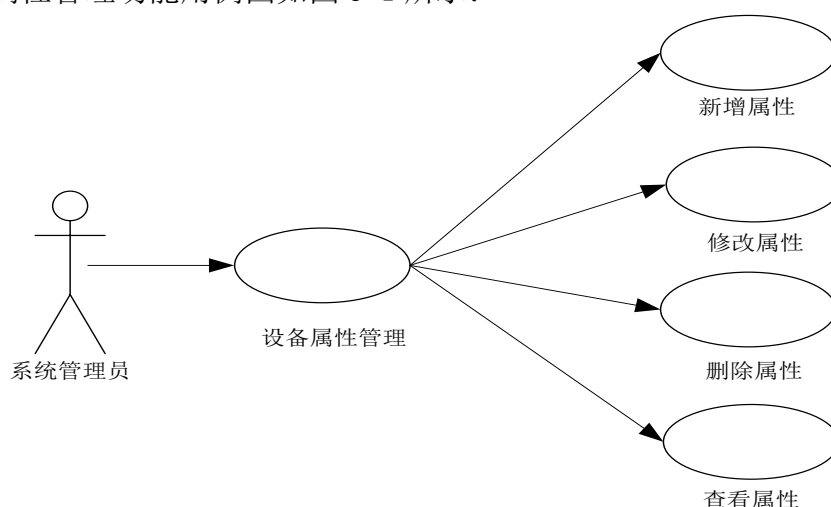


图 3-1 设备属性管理功能用例图

设备属性管理用例表如表 3-1 所示：

表 3-1 设备属性管理用例表

用例 ID	3.3.1		
用例名称	设备属性管理	版本号	1.0
目的	设备属性管理，包括对接入云存储系统的设备基本属性进行管理，包括设备厂商、设备类型、设备型号等的录入与管理。通过设备基本属性的统一管理，可实现系统管理员快速、准确找到故障服务器的诊断与维修。		
用户/行为人	系统管理员		
触发者	系统管理员		
前提条件	系统初始化已经完成		
结束条件	设备基本信息录入完成，并成功返回结果		

设备入网注册管理功能主要由系统管理员和系统审计员完成操作，主要包括设备入网申请的录入和审核功能。新设备的加入需要在这里进行设备入网注册，填写设备的相关信息。下面给出了设备入网注册管理的用例图和用例表详细描述。设备入网注册管理功能用例图如图 3-2 所示：

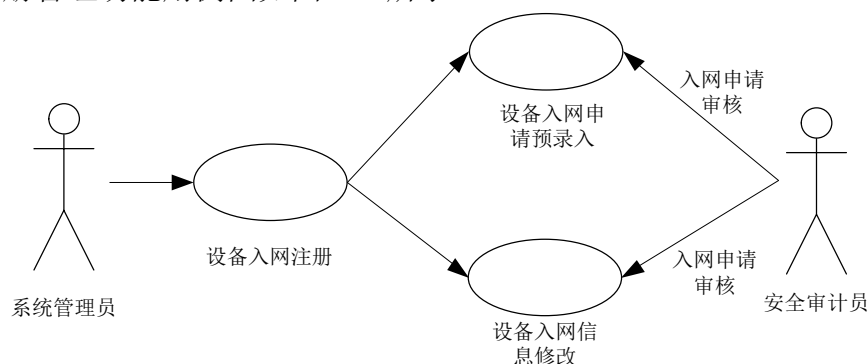


图 3-2 设备入网注册管理功能用例图

设备入网注册管理设备属性管理用例表如表 3-2 所示：

表 3-2 设备入网注册管理用例表

用例 ID	3.3.2		
用例名称	设备入网注册管理	版本号	1.0
目的	实现云存储平台接入设备的入网注册，包括设备单位信息、设备用途以及设备所承载的应用信息信息，设备 IP 地址、管理代理端口号等。		
用户/行为人	系统管理员、安全审计员		
触发者	系统管理员		
前提条件	系统初始化已经完成，设备基本信息录入完成，		
结束条件	根据设备信息，产生设备唯一标识号，安全审计员审核批准设备入网，并成功返回结果		

3.3.2 网络管理

网络管理功能是要通过对存储设备的网络管理，包括存储控制器的 IP 地址、子网掩码等信息的设定，方便对存储设备进行远程管理等操作。下面给出了网络管理功能需求的用例图和用例表详细描述。网络管理功能用例图如图 3-3 所示：

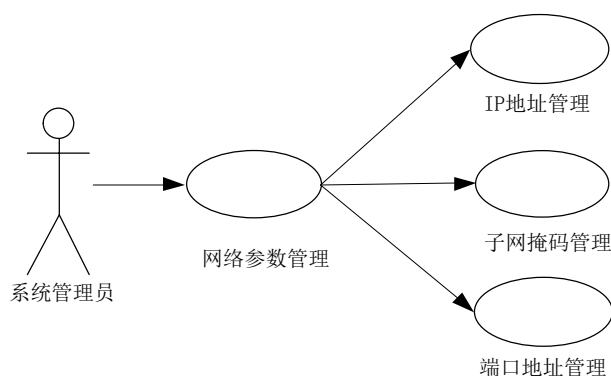


图 3-3 网络管理功能用例图

网络管理功能用例表如表 3-3 所示：

表 3-3 网络管理用例表

用例 ID	3.3.3		
用例名称	网络管理	版本号	1.0
目的	通过对云存储平台接入的存储设备进行网络属性（IP 地址、端口、子网掩码等）配置管理，实现系统管理员远程管理存储设备。		
用户/行为人	系统管理员		
触发者	系统管理员		
前提条件	存储设备安装完成		
结束条件	系统管理员可以远程访问存储设备		

3.3.3 监控管理

云存储平台上的应用系统和存储设备正常运转后，需要为系统管理者提供一个可视化的管理界面，为满足数据中心在地域上的分布，系统提供了两种方式的网络拓扑监控图：分别是物理拓扑图和逻辑拓扑图。

1) 物理拓扑图监控管理

物理拓扑图监控界面，为系统管理者提供一个从地域上的宏观拓扑图，通过图形界面，管理者可以看到各云存储的数据中心实时的运行状态。下面给出了物理拓扑监控管理功能需求用例图和用例表详细描述。

物理拓扑监控管理功能需求用例图如图 3-4 所示：

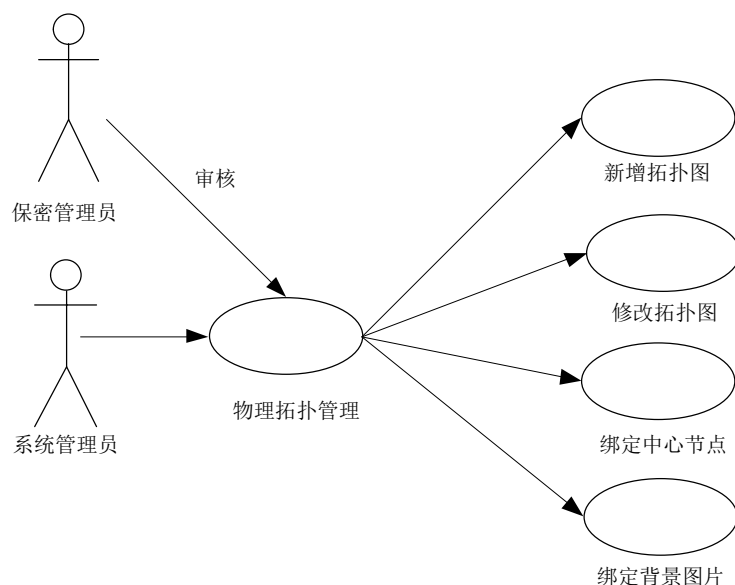


图 3-4 物理拓扑管理功能用例图

物理拓扑管理功能用例表如表 3-4 所示：

表 3-4 物理拓扑管理用例表

用例 ID	3.3.4		
用例名称	物理拓扑图管理	版本号	1.0
目的	通过物理拓扑图，直观地展现云存储平台服务器的运行状况，物理拓扑管理包括新增拓扑图、修改拓扑图、绑定中心节点、绑定背景图片等功能。		
用户/行为人	系统管理员		
触发者	系统管理员		
前提条件	设备入网注册完成		
结束条件	物理拓扑编辑完成，通过系统自动刷新，可以实时展现出数据中心系统设备运行状况		

2) 逻辑拓扑图管理

逻辑拓扑图监控界面，为系统管理者提供一个模拟数据中心实时的网络逻辑拓扑图，通过图形界面，管理者可以看到数据中心上运行的各服务器和存储设备实时的运行状态。下面给出了逻辑拓扑监控管理功能需求用例图和用例表详细描述。

逻辑拓扑监控管理功能需求用例图如图 3-5 所示：

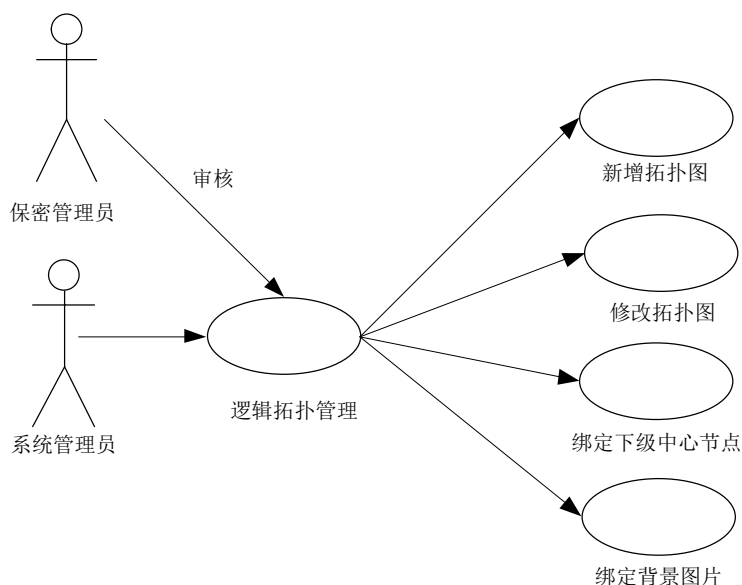


图 3-5 逻辑拓扑管理功能用例图

逻辑拓扑管理功能用例表如表 3-5 所示：

表 3-5 逻辑拓扑管理用例表

用例 ID	3.3.5		
用例名称	逻辑拓扑图管理	版本号	1.0
目的	通过逻辑拓扑图，直观地展现云存储平台服务器的运行状况，逻辑拓扑管理包括新增拓扑图、修改拓扑图、绑定下级中心节点、绑定背景图片等功能。		
用户/行为人	系统管理员		
触发者	系统管理员		
前提条件	设备注册完成，物理拓扑编辑完成。		
结束条件	逻辑拓扑编辑完成，通过系统自动刷新，可以实时展现出数据中心系统设备运行状况		

3.3.4 监控策略管理

监控管理平台提供对监控策略的定义和管理，通过预先设定的告警策略，实现平台的自动报警，系统可提供自定义的自动告警策略。下面给出了监控策略管理功能的需求用例图和用例表详细描述。

监控策略管理功能需求用例图如图 3-6 所示：

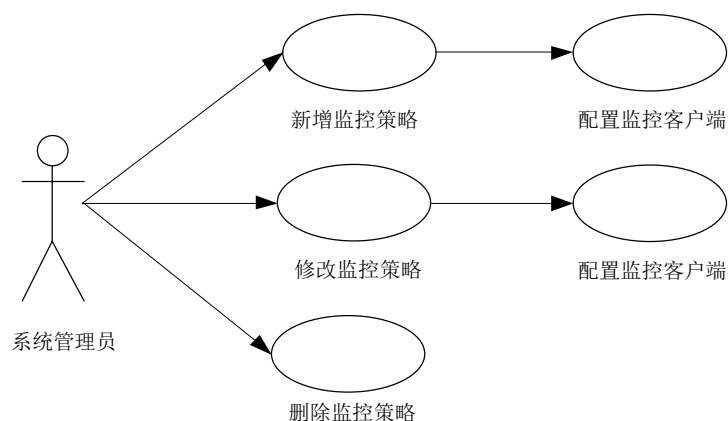


图 3-6 监控策略管理需求用例图

监控策略管理功用例表如表 3-6 所示：

表 3-6 监控策略管理用例表

用例 ID	3.3.6		
用例名称	监控策略管理	版本号	1.0
目的	通过监控策略的配置，实现监控平台按照设定的规则进行轮询监控		
用户/行为人	系统管理员		
触发者	系统管理员		
前提条件	系统服务部署完成		
结束条件	修改监控客户端后，可以根据监控策略地改变，返回正确的监控项		

3.3.5 统计分析

监控管理平台将每天收集到的云存储平台上所有的监控信息，汇总到数据库，需要通过一个直观的数据统计报告，呈现给系统管理员和更高级管理者，最直观和形象的方法就是报表。监控管理平台提供信息报表模板自定义，系统管理员可以通过设定不同的报表展现不同的统计数据。下面给出了统计分析管理功能需求用例图和用例表详细描述。

监控数据统计分析管理功能需求用例图如图 3-7 所示：

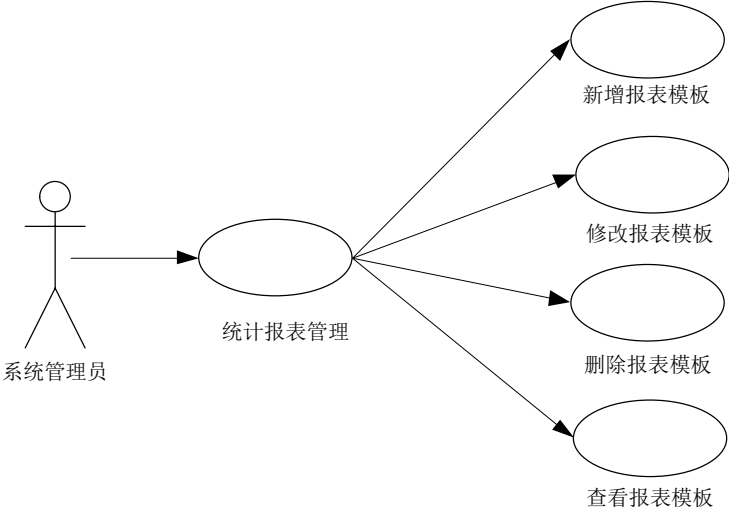


图 3-7 统计分析需求用例图

监控数据统计分析用例表如表 3-7 所示：

表 3-7 统计分析管理用例表

用例 ID	3.3.7		
用例名称	报表模块管理	版本号	1.0
目的	通过对报表模板的设置，可以为监控系统提供多种报表格式，以多种形式将监控数据展现给管理者，有助于对云存储系统整体运行状况的分析。		
用户/行为人	系统管理员		
触发者	系统管理员		
前提条件	系统初始化完成		
结束条件	导入新模板成功		

3.3.6 日志管理

云存储监控管理平台对所有操作日志和系统运行日志进行监控，确保做到平台每一次操作和运行状态的准确记录，为管理者查找系统故障和隐患提供证据。

1) 操作日志管理

监控管理平台提供系统操作日志管理与记录，日志要求记录每次操作的时间、操作人、操作内容、操作结果等详细信息，并提供日志查询功能。

操作日志管理功能需求用例图如图 3-8 所示：

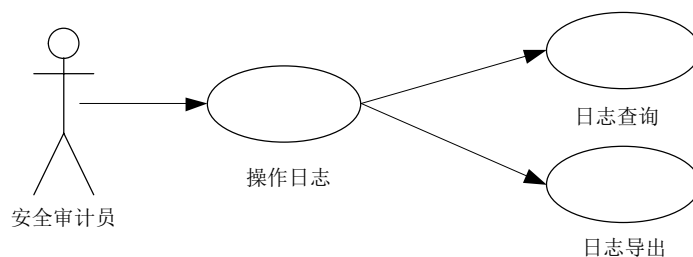


图 3-8 操作日志用例图

操作日志管理用例表如表 3-8 所示：

表 3-8 操作日志管理用例表

用例 ID	3.3.8		
用例名称	操作日志管理	版本号	1.0
目的	通过页面查询，可查询到系统中三员的操作日志信息		
用户/行为人	系统管理员、安全管理员和审计员		
触发者	安全审计员		
前提条件	系统登录成功，有过操作记录		
结束条件	查询返回成功		

2) 运行日志管理

监控管理平台提供系统运行日志管理与记录，日志要求记录系统运行中的关键事件，如时间发生的时间、内容等信息，并提供日志查询功能。

运行日志管理功能需求用例图如图 3-9 所示：

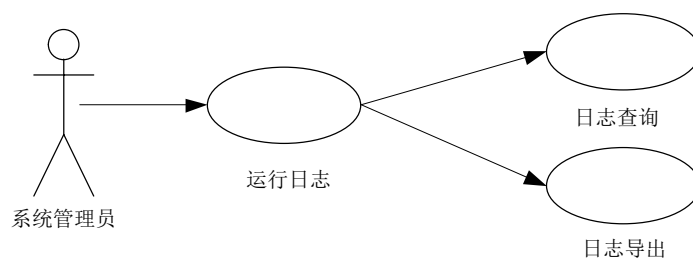


图 3-9 运行日志用例图

运行日志管理用例表如表 3-9 所示：

表 3-9 运行日志管理用例表

用例 ID	3.3.9		
用例名称	运行日志管理	版本号	1.0
目的	通过页面查询，可查询到系统的基本运行日志，管理员可通过日志查看系统运行的错误信息，便于发现风险或问题		
用户/行为人	系统管理员		
触发者	系统管理员		
前提条件	系统登录成功，有过操作记录		
结束条件	查询返回成功		

3.3.7 组织机构管理

云存储系统按照业务和资产进行划分，组织机构管理是管理平台中的基础环节。云 IT 资源的管理将按照不同的组织机构进行划分，系统提供机构的添加、修改、删除、查询功能，便于管理员查找资产对应的业务和资产信息，并对业务和资产属性进行维护。

基于动态业务和资产属性，系统管理员可以对业务和资产属性进行扩展，例如可以根据用户自身的需要为业务和资产增加重要的属性信息。同时，所有扩展的属性都可以作为查询条件进行检索，便于管理员进行查询和维护。

组织机构管理用例模型图如图 3-10 所示：

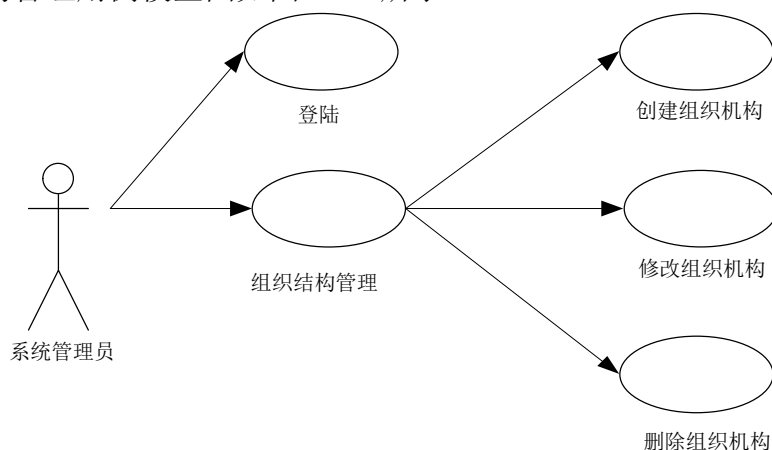


图 3-10 组织机构管理需求用例模型

组织机构管理用例表如表 3-10 所示：

表 3-10 组织机构管理用例表

用例 ID	3.3.10		
用例名称	组织机构管理	版本号	1.0
目的	通过创建组织机构树，包括对机构名称、机构属性管理，将接入云存储平台应用系统及其对应的组织机构实行绑定与分类管理,便于管理员通过监控信息可以快速查找到设备所属的组织或单位。		
用户/行为人	系统管理员		
触发者	系统管理员		
前提条件	系统初始化已经完成、应用系统接入前		
结束条件	组织机构树创建并返回成功		

3.3.8 用户管理

云存储管理平台的管理人员用户必须进行严格管理，采用基于角色的访问控制策略，即依据对云存储管理平台中的角色行为来限制对系统资源的访问。要求不同角色的用户组拥有不同的访问权限，避免用户组中的用户不可以获得滥用系统资源的特权，杜绝超级管理员账户的出现，并且利于责任独立。

用户管理需求用例模型图如图 3-11 所示：

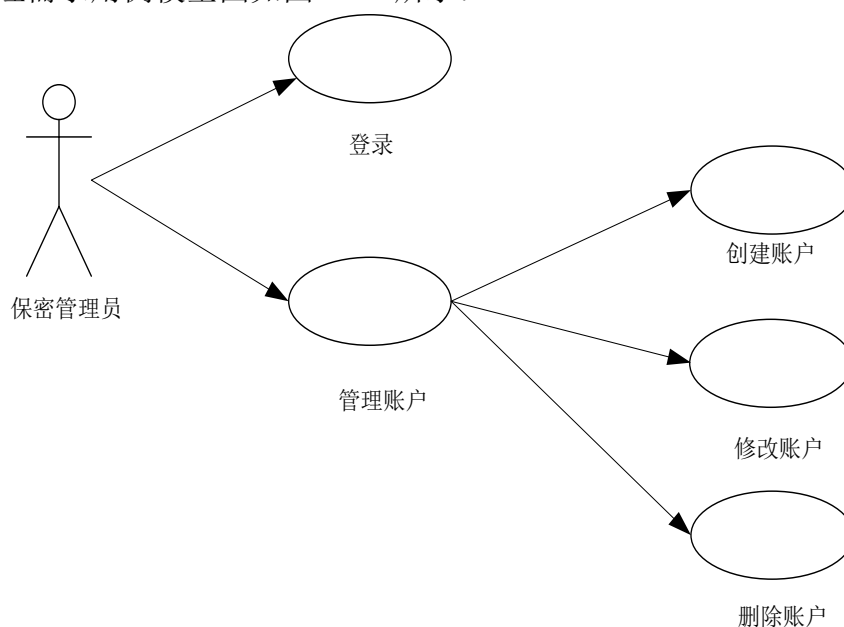


图 3-11 用户管理需求用例模型

用户管理功能用例表如表 3-11 所示：

表 3-11 用户管理用例表

用例 ID	3.3.11		
用例名称	用户管理	版本号	1.0
目的	用户管理要严格实现基于角色的访问控制策略，包括设置系统三员：系统管理员、安全管理员和安全审计员，通过配置系统中角色行为来限制对资源的访问。实现用户角色管理的层次化，便于系统的管理。		
用户/行为人	保密管理员		
触发者	保密管理员（sec）		
前提条件	系统初始化已经完成		
结束条件	系统三员创建完成，并可以使用各自账号成功登录		

3.4 系统非功能性需求

云存储监控管理平台在性能上主要需提供满足系统的响应需求和可用性需求。一个软件系统的开发一般都要有一些合理的规范和原则，本系统应按以下几个原则来进行开发和编程。

1) 实用性原则

本软件的开发首先遵循的原则便是实用性，一个软件要满足用户所提出的功能、也要让用户可以方便的进行操作、系统需要有一个友好的操作界面，方便使用者任意使用各项功能，并且，也要有完备的维护措施。所以，开发的系统应该尽可能的使用各种方法来简化人机交互的过程，从而提供给使用者一个简洁。功能强大的系统^[26]。

2) 系统性原则

监控管理平台应该具备明显的整体性、层次结构性、系统性、目的性和综合性。系统各子功能处理的数据也应该是既相互独立又彼此影响的，从而整合为一个完整而又可以相互共享的完整的体系。因此，在系统开发过程中，需要保证其功能和数据上的整体性、系统性。

3) 符合软件工程规范原则

本软件遵循软件工程的理论、方法和规范去设计和编程，注重软件表现工具的运用、文档资料的整理、阶段性评审及项目管理。

4) 逐步完善, 循序渐进的原则

本系统的设计和实现不可能一步就到位的, 因此肯定有一个慢慢完善, 逐渐发展的过程。随着软件的开发, 管理人员对系统的认识也在逐渐加深, 因此, 对信息化的需求和处理也会提出更多新的要求和想法。如果一个服务的功能性和非功能性被描述的非常好, 那么这个服务会被非常完整的呈现出来。通过检查现有商业服务中必要的 QoS 因素, 发现它们标识了很多非功能性的属性, 比如可用性、信道、收费模式、和安全性等等。尽管如此, 它们并没有显示这些被标识的非功能性属性的作用或者期望的针对上述操作的解决方案。很多第三方的组织尝试着详细的制定基于非功能性因素的标准。比如, WS-policy 提供了一个语法相关的模型来详细说明 web 服务的端到端策略。WS-security 使得能够获得的端到端 SOAP 消息整合, 授权。本系统的性能需求具体要求如下:

3.4.1.1 响应需求

由于系统操作是通过 web 页面访问, 因此要求在正常业务流程下, 系统的登录响应时间在 5 秒之内; 在业务并发高峰时间, 系统的登录响应时间在 10 秒之内。

3.4.1.2 易用性需求

监控管理平台主要针对云存储系统的管理员, 应考虑到实际使用中的可操作性, 以及部署和培训的难度。管理平台的系统界面是管理着与云存储资源之间的媒介, 因此, 要求界面设计既要便于操作, 也要突出友好性。

易用性也要求管理平台的界面设计要考虑美观, 各个功能模块之间的界限要求清晰。每个页面中尽可能展示完每项业务功能, 业务功能操作与数据不易造成混淆。

3.4.1.3 可靠性需求

云存储平台往往会承载企业核心的业务系统, 因此系统的可靠性是非常重要的环节, 考虑到数据中心的业务系统往往都是长期运行, 所以要求本系统的可靠性应能满足 7×24 小时的安全运行服务。

3.4.1.4 系统的安全性

云存储平台是企业核心业务和重要数据的载体, 系统的安全运行与企业生命息息相关, 因此监控管理平台必须确保系统的安全运行。云存储管理端的数据不被管理者随意更改, 否则会影响平台的正常运行, 因此要实现三员管理, 即系统

管理员、保密管理员和安全审计员要权责分离^[27]。

3.5 本章小结

第三章监控管理平台需求分析，从功能性需求和非功能性需求等方面对监控管理平台进行了全面分析，明确了云存储系统监控管理平台需要实现的功能，以及云存储系统监控管理平台的运行环境、服务性能等实现目标。

下一章监控管理平台总体设计，将根据需求对云存储系统监控管理平台进行系统总体设计。

第四章 监控管理平台总体设计

本章将根据系统需求分析的结果，对该系统进行框架设计并阐述，同时将根据系统整体的功能需求，将系统分解成为多个模块。另外，还将对系统数据库设计和数据表结构以及接口的设计进行介绍。

4.1 设计原则

云存储监控管理平台遵循以下设计原则进行产品开发设计：

1) 采用模块化结构^[28]

各功能模块的设计遵循任务独立的原则，各模块为其它模块提供服务；对于未来有可能变动的业务可以采用参数驱动的设计策略，从而保证整个系统平台的灵活性。

2) 低耦合、高内聚

监控系统的各模块设计要尽可能保持功能内聚和数据耦合，便于系统的后期扩展。

3) 可测试性

监控管理平台应设计应考虑到后期的可测试性，系统应设计一个适当的数据集合，用来测试所建立的系统。

4) 可扩展性

监控管理平台设计时应充分考虑后期新功能的扩展性，以适应不断发展的云存储平台的应用需求。

4.2 设计目标

本系统的设计目标是要设计一套完善的、可视化的云存储监控管理平台，并提供专业的报告与分析，定期对监控结果进行整合分析，让管理者更清晰地知道监控项目的运行情况，可实现全网的统一运维管理，建立起主动式的运维模式，减少管理者的工作量，降低企业运维成本。

4.3 系统架构设计

监控管理平台从总体架构上可分为两个层级：一层是数据采集层，代理程序寄宿在分布式文件系统之上，为平台管理层提供数据来源；另外一层是数据处理

层，负责收集数据并进行数据处理与展现。监控管理平台逻辑架构如图 4-1 所示：

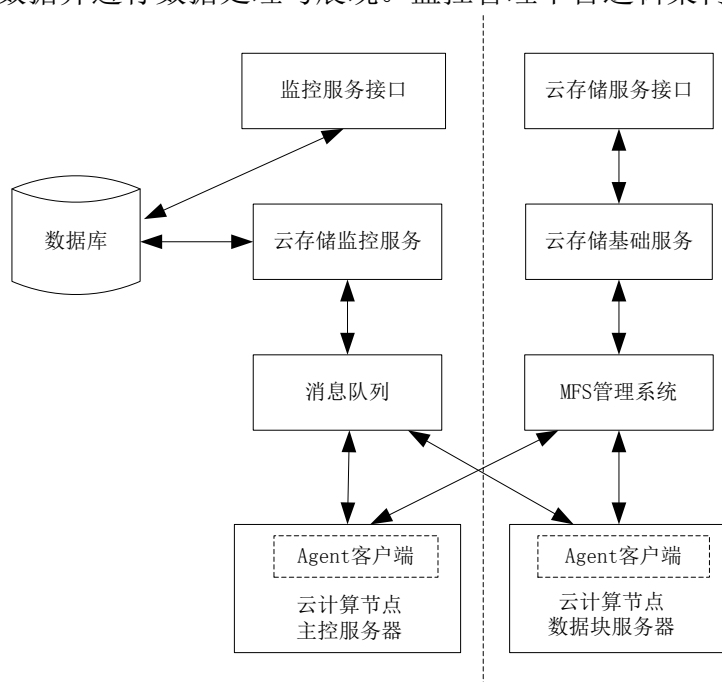


图 4-1 管理平台逻辑架构图

监控服务模块获取云存储平台内部每个计算节点的运行状态参数，包括 CPU、存储、磁盘状态以及网络状态等状态信息，将收集到的数据存储到数据库中。并在云存储监控管理处理中心提供查询接口，管理员通过拓扑图可以调出设备所对应的监控数据。整个架构采用的是“服务器-客户端”的分布式异步方式，具有较强的可扩展性，而且监控模块本身对分布式文件系统的依赖非常小。

而管理平台本身将从实际技术可行性、未来发展趋势等方面考虑，基于 windows 系统，采用 JAVA 插件技术、MySQL 数据库等工具进行开发。管理平台选用 Java/Jsp 开发框架，遵循 J2EE 规范，采用 MVC (Model 模式-View 视图-Controller 控制器) 模型，对系统进行统一设计和架构。监控代理程序采用 C++ 开发，系统操作人员通过 B/S 方式与系统交互，被管设备通过 C/S 方式与系统进行交互。最终实现的监控管理平台达到可视化的界面设计，真实模拟网络环境，构建服务的逻辑视图与物理视图，提供简单、易配置的管理页面。该平台要实现四大特色：平台化、组件化、对象化、自动化。

4.4 云存储监控服务模块设计

云存储监控服务模块采用低耦合的架构，通过 MFS 本身的管理首先实现节点集群式的管理，数据节点的增加和减少可以灵活控制。云存储监控服务参照了 MFS

分布式文件系统本身的设计理念，采用“服务-客户端”的管理模式^[29]，当数据节点被扩展以后，只需要将监控客户端安装到数据节点系统中即可开始工作。云存储监控模块架构如图 4-2 所示。

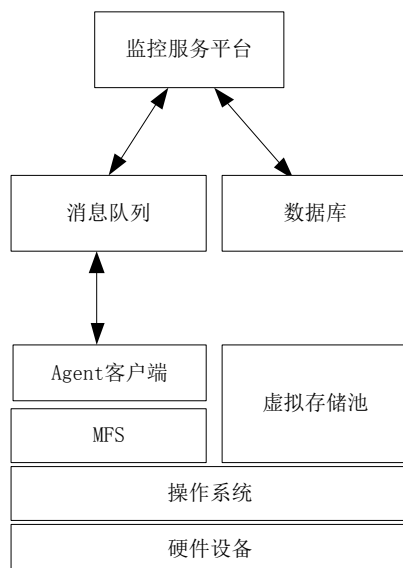


图 4-2 云存储监控模块架构图

云存储监控模块主要由监控服务平台、监控客户端、消息队列和数据库组成。云存储监控模块，即监控管理中心将作为整个监控平台的核心部分，是收集数据和处理数据的中心。监控客户端部署在 MFS 的数据节点服务器上，与 MFS 本身的文件系统独立。

4.5 监控管理平台功能模块设计

云存储监控管理平台功能模块主要负责系统的业务逻辑功能，由业务处理服务完成其主要功能。业务处理服务以组件的形式对外提供设备管理、拓扑管理、监报告警管理、配置管理、统计分析和日志管理功能。业务处理服务对外提供基于 HTTP 服务功能，系统采用开源的嵌入式 Web 服务。

监控管理平台功能模块逻辑划分如图 4-3 所示。

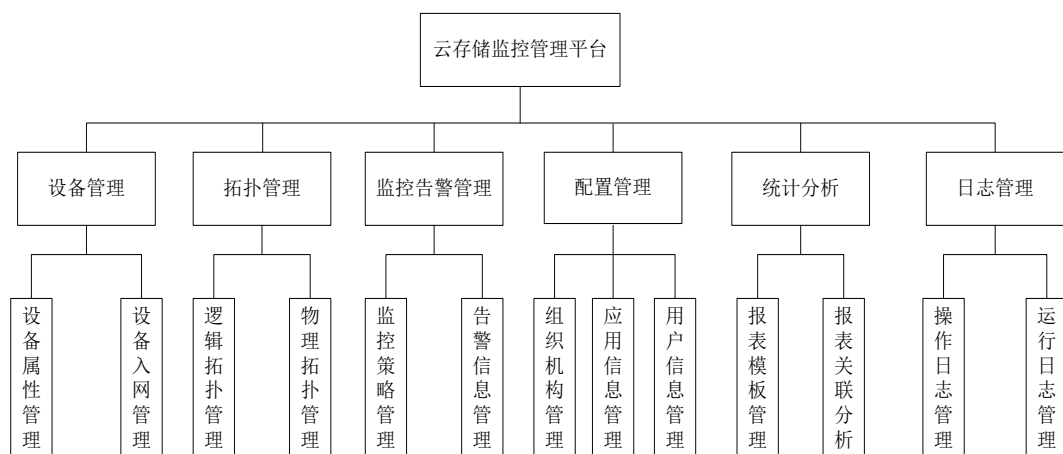


图 4-3 管理平台功能架构图

监控管理平台从功能上划分为业务管理和监控管理两大部分。业务管理包括：设备管理负责对云存储平台内介入的所有设备进行管理，包括：设备属性管理（厂商、属性等）、设备入网管理；拓扑管理功能包括：逻辑拓扑管理和物理拓扑管理；监控告警管理包括：监控策略管理和告警信息管理；配置管理包括组织机构管理、应用信息管理和用户信息管理；数据统计分析主要负责对代理程序上报至监控管理平台的数据进行统计与分析，通过各种报表的形式，展现到管理平台上；日志管理主要负责对系统平台的操作日志和运行日志进行集中管理，通过各种查询条件筛选出指定范围内的各种操作日志或系统运行日志，为平台管理员提供故障维护或审计依据。Agent 代理程序负责调度各类算法和系统函数，收集设备运行信息，推送至监控管理平台。系统内的各个功能模块保持低耦合连接，尽可能使这些功能独立，从而提高系统的灵活性。

4.6 系统安全设计

4.6.1 安全访问控制

系统提供客户端与服务端的安全通道建立，确保数据在通道传输和交换过程的安全。为保证系统本身的安全性，要求登录本系统的所有管理员必须进行身份认证，不同角色的管理员具有不同的操作权限，只有拥有相应权限的操作员才能进行相关的管理和操作。管理员必须使用正确的口令密码才能登录系统。

为防止直接进入数据库，恶意篡改数据库中管理员信息和管理员权限信息，系统需要对管理员及权限信息作 MAC 运算，系统从数据库中读取管理员信息和管理员权限信息时，将进行验证，以确保信息未被非法篡改。

4.6.2 系统安全管理

系统的管理通过三员分权方式，管理员间权力分离，相互制约，在管理上形成立体防护。从管理员登录到管理员退出，所有操作全部记录，不可抵赖。系统管理员角色与权限划分设计如表 4-1 所示。

表 4-1 系统管理角色与权限划分设计表

系统用户角色	权限
系统管理员	主要负责系统的日常运行维护工作。
系统安全员	主要负责系统的日常安全保密管理工作，包括用户帐号管理以和系统所产生日志的审查分析。
系统审计员	主要负责对系统管理员、安全保密管理员的操作行为进行审计跟踪分析和监督检查，以及时发现违规行为，并定期向系统安全保密管理机构汇报相关情况。

系统的安全访问控制和安全管理分析，也是安全性设计在系统的各个层次上的体现，从云存储监控管理平台安全性需求分析来看，是可以达到其要求的，系统可以适用于各种网络环境。

4.7 接口设计

系统接口主要包含各功能模块的输入输出接口，系统内部调用接口，以及底层模块与文件系统和数据库交互。表现层模块与操作人员交互，这部分接口采用通用、成熟的接口设计，这里将不再详述。本章节重点介绍云存储连接配置接口、代理程序调用接口、用户管理模块调用接口和管理平台日志调用等接口设计。

4.7.1 链接配置接口设计

管理平台与存储系统的链接配置主要完成云服务器的配置、主控服务器和数据节点服务器的地址获得、链接的开启和关闭等。以下为部分链接配置程序的接口设计。

1) 配置云主控服务器地址

接口函数：int scmgr_addserv(char *msip,char *msport)

2) 获取配置的主控服务器地址

接口函数：int scmgr_getserv(char *msip[32],char *msport[32]);

3) 连接认证接口

Int scmgr_register(char *loginname,char *passworddata,int pwdlen,int mode);

4) 连接指定的服务器

接口函数: int scmgr_connect(char *ip,char *port, char *loginname,char *passworddata,int pwdlen,int mode)

5) 关闭指定的连接

接口函数: void scmgr_closeconnect(int sockfd)

4.7.2 代理程序接口设计

代理程序接口为内部调用接口,将通过代理程序调用数据节点上的管理接口。

以下为部分代理程序的接口设计。

1) 获取本机节点类型(本地配置)

接口函数: int scmgr_nodetype(int sockfd)

参数: Socketfd

//入参,指定服务器连接字,参数0表示本机

2) 获取云存储平台的统计信息

接口函数: Int scmgr_cloudinfo(int sockfd,char *getinfostruct)

//包括文件数量、数据块数、总空间、剩余空间

3) 获取一个节点正使用的存储空间列表和状态

接口函数:

int scmgr_gethlist(int sockfd,struct gethinfostruct *phdinfostruct)

//包含主控节点和数据节点

4) 获取 CPU、内存使用情况

接口函数: void scmgr_GetSysInfo(int sockfd ,InitCPU *in, cprStats *stats)

5) 获取设备在线信息,通过主控节点调用

接口函数: Int scmgr_getonlinebegin(int sockfd ,int *count)

4.7.3 用户管理接口设计

用户管理主要提供系统账户的管理,包括系统管理员、保密管理员和安全审计员等账户的管理。以下为部分用户管理模块接口的设计。

1) 添加用户

接口函数:

Int scmgr_useradd(int sockfd, char *org,char *ou,char *userid,char *password,char *username,unsigned int goal,int quota,int role)

配置用户口令

接口函数:

```
int scmgr_userpassword(int sockfd, char *userid, char *oldpassword, char
    *newpassword)
```

2) 修改用户信息

修改 userid 用户的信息, 要修改的用户必须已存在。

接口函数:

```
int scmgr_usermodify(char *userid, char *username, int goal, int quota)
```

3) 删除用户

接口函数: `int scmgr_userdel(int sockfd, char *userid)`

4) 获取用户信息, 获取指定 userid 用户的信息

接口函数:

```
Int scmgr_usergetinfo(int sockfd, char *userid, char *org, char *ou, int
    *inodes, unsigned char *goal, int *quota, int *usesize, int *use, char *isonline)
```

5) 启禁用

接口函数: `Int scmgr_setuserstatus(int sockfd, char *userid, int use)`

4.7.4 日志接口设计

监控管理平台日志接口包括: 操作日志、系统运行日志。管理员操作日志与来源相同, 本文不做描述。云存储运行日志信息包含: 编号、信息、类型、状态、日期、节点 IP。获取运行日志, 并自动删除已获取到的日志。

接口函数: `int scmgr_getanddellog(sc_run_log *log)`。

4.7.5 数据库接口设计

在 J2EE 中, 不仅可以使⽤ JDBC 来访问不同的数据库, 还可以使⽤对象关系映射器来完成业务逻辑层的对象到数据库的映射。采⽤官方的对象关系映射器是 CMP, 使⽤ CMP 时, 除了要考⻝域模型外, 还要考⻝ EJB 容器本身的特性。

业务逻辑层的对象尽量不采⽤ EJB 的 CMP, 而采⽤ POJO (Plain Old Java Objects, 普通 Java 对象)。采⽤ POJO 的领域模型易于组装、创建快捷、可在 EJB 容器之外测试和运行, 系统客户端通过调⽤系统函数和服务⻝之间建立连接。客户端和服务⻝建立连接后, 将获得配置⻝件信息。function __construct(\$BKtype)构造各种路径, 实现树形结构。此函数实现了构造⻝件传输列表路径, 远程根路径等。例如构造当前目录的路径, 通过⻝下的⻝现实现: `$this->m_GlobalRemotePath`

```
= $this->m_FTPHome.$this->m_User. "/" . $this->m_HostID . $targetPath . "/current/";
```

由于函数获得了配置文件信息, 所以像 `m_User` 等信息会直接通过调用配置文件信息来获得。POJO 对象与数据库之间采用对象关系映射器来实现对象到关系数据库的映射, 但考虑性能问题, 将来不排除在业务逻辑层和持久层之间局部使用表模块模式。系统采用开源的 Hibernate 来作为对象关系映射器和数据库访问接口。

4.8 数据库设计

数据库的设计应从云存储监控管理平台的系统功能需求出发, 梳理出所需数据的类别、属性、范围等等, 以及系统在运行过程中与云存储系统的通信情况。确定系统用户对数据库系统的使用要求和其他相关的约束条件, 并且要能够反映系统的数据结构、主要数据信息情况, 制约和管理各种功能模块的信息。

数据库设计将是云存储监控管理平台设计的重要部分, 是本系统实现的前提条件, 数据库设计的合理性将关系到系统的成败。它不仅为整个系统提供数据, 同时也要满足各个环节的数据连接和关联。设计时必须全面考虑系统的要求和所要达到的目标。云存储监控管理平台数据库设计将遵循以下设计原则:

- (1) 既遵循数据库规范化规则, 又考虑应用程序的性能需求。
- (2) 保证数据共享和数据应用的一致性、完整性和安全性。
- (3) 保证空间数据的历史回溯。
- (4) 确保具有良好的可扩展性和兼容性。

本节数据库设计主要描述两个方面的内容, 包括数据库 E-R 模型设计、数据库主要表结构设计。

4.8.1 E-R 模型设计

数据库 E-R 模型图是描述系统概念结构模型的有效方法, 它可以直观、具体地提供对数据模型的描述, 是数据库结构设计中应用最广泛的工具。

本文通过对云存储监控管理平台的使用环境和要求进行详细分析, 采用了用多层数据流图和数据字典来描述了整个监控系统。系统中需要进行持久化操作的主要对象有设备、网络信息、监控策略、组织机构信息等, 系统中主要的对象实体和属性之间的联系如图 4-4 所示:

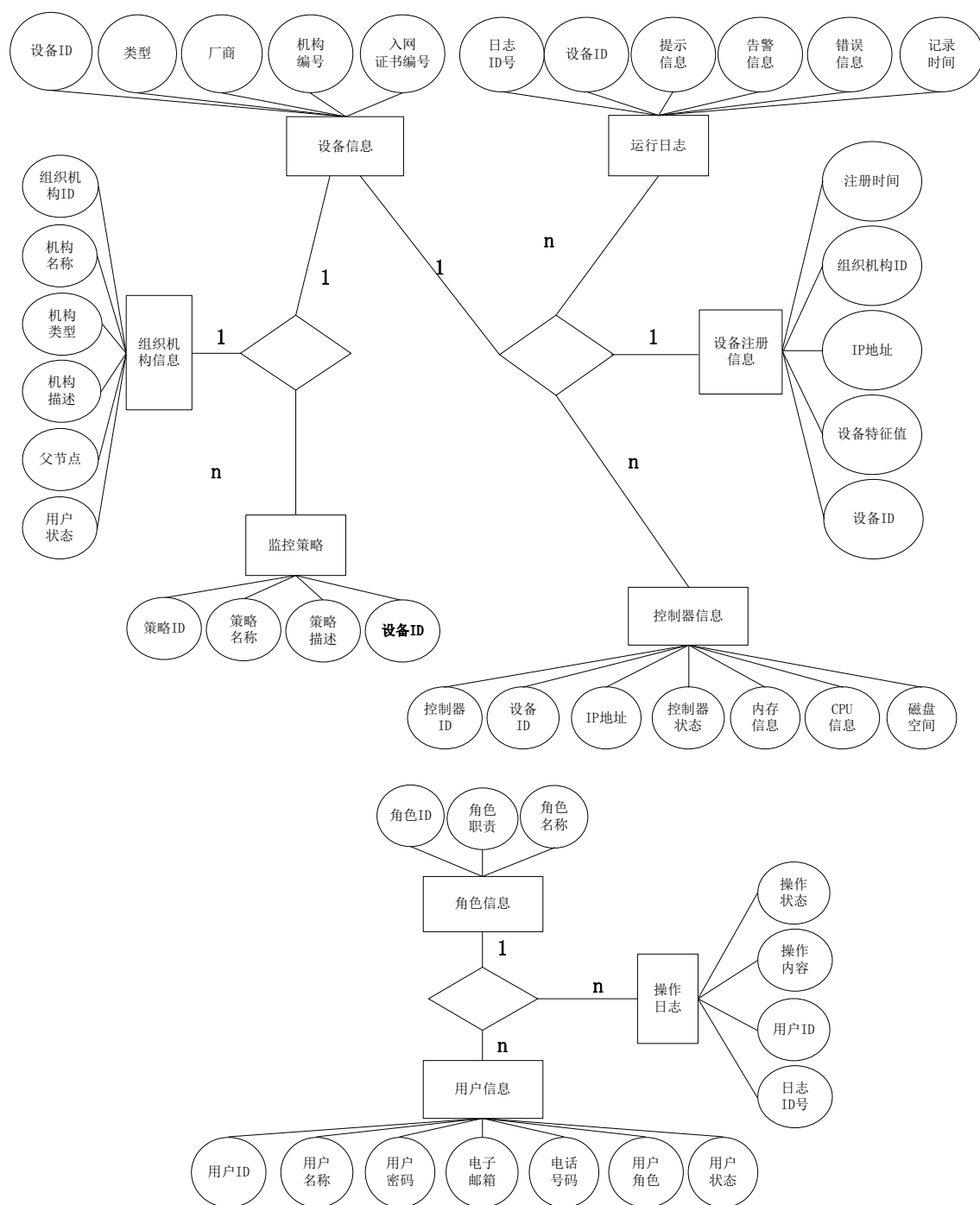


图 4-4 数据库 E-R 图

设备信息主要包括设备的设备 ID、类型、厂商、机构编号、入网证书编号等信息。组织机构主要包括接入的数据中心或应用所代表的组织机构 ID、机构类型、名称、父节点和用户状态等。控制器信息主要包括控制器 ID、设备 ID、IP 地址、控制器状态、内存信息、CPU 信息、磁盘空间等。注册信息主要包括设备 ID、设备特征值、IP 地址、组织机构 ID、注册时间等信息。

系统中将包含多个用户角色，包括三个管理员角色，包括系统管理员、系统安全员和安全审计员，可共同对系统进行相应的管理操作。同时，每个管理员拥有自己的身份证书，用作系统的登录身份认证，证书可以是外部机构发放，也可以结合 usbkey 身份钥匙来使用。日志将记录管理员的操作信息，每个管理员会产生多条操作日志信息。

4.8.2 数据库表结构设计

管理平台中数据库包含若干张数据表，结合数据库 E-R 模型进行了表结构设计，本文重点介绍用户管理表和设备管理表的表结构设计。其中用户管理表包括了用户信息表 userinfo、角色表 role 等；设备管理表包括了设备注册信息表 devreginfo、控制器信息表 devinfo、设备机构信息表 devgroupinfo。本文只选取了其中部分表结构进行介绍。

1) 组织机构信息表如表 4-2 所示。

表 4-2 组织机构信息表 institinfo

变量名	类型	说明
institID	int*	组织机构 id
institName	String	机构名唯一索引
instittype	String	机构类型
Institdepiction	String	机构描述
fatherID	String	父节点 ID
userStatus	Int	用户状态 (0 禁用，1 启用，2 已删除)

2) 角色信息表如表 4-3 所示。

表 4-3 角色信息表 role

变量名	类型	说明
roleID	Int*	Auto Increment, 用户 id
roleName	String	角色名 0 管理员 1 审计员 2 操作员
roledescription	String	职责描述

3) 用户信息表如表 4-4 所示。

表 4-4 用户信息表 userinfo

变量名	类型	说明
userID	int*	Auto Increment, 用户 id
userName	String	用户名唯一索引
password	String	密码
Email	String	电子邮箱
phone	String	电话
roleid	Int	角色 0 管理员 1 审计员 2 操作员
userStatus	Int	用户状态 (0 禁用, 1 启用, 2 已删除)

4) 设备信息表如表 4-5 所示。

表 4-5 设备信息表 devinfo

字段名	类型	说明	备注
devID	Int*	Auto Increment	权限唯一标识 id
devtype	varchar(30)	设备类型	
devmanu	varchar(30)	设备厂商	
institID	Int	设备所属机构	组织机构 ID
devregID	Int	设备注册证书 ID	

5) 设备注册信息表如表 4-6 所示。

表 4-6 设备注册信息表 devreginfo

字段名	类型	说明	备注
devregID	Int*	Auto Increment	权限唯一标识 id
devno	varchar(32)*	设备唯一编号	如 MAC 地址
masterip	varchar(30)		主 ip
institID	Int		组织 ID 或机架 ID
RegDate	Date		日期

6) 设备信息表如表 4-7 所示。

表 4-7 控制器信息表 devinfo

字段名	类型	说明	备注
ID	Int*	Auto Increment	权限唯一标识 id
devno	varchar(32)	对应 devreginfo 的 devno	设备编号
Ip	varchar(120)	Devno, Ip 唯一索引	IP 地址
Status	Char	控制器状态	
MEM	varchar(32)	内存信息	
CPU	varchar(32)	CPU 核数, 频率	
Diskspace	varchar(32)	磁盘空间	

7) 监控策略表如表 4-8 所示。

表 4-8 监控策略表 strategy

字段名	类型	说明	备注
straID	Int	Auto Increment	唯一标识 id
straname	Varchar(60)		策略名称
strainfo	Varchar(120)		策略详细描述
devID	Int		设备 ID

8) 运行日志表如表 4-9 所示。

表 4-9 运行日志表 runloginfo

字段名	类型	说明	备注
runlogID	Int	Auto Increment	唯一标识 id
设备 ID	Int		设备 ID
INFO	Varchar(120)		提示信息
WARN	Varchar(120)		警告信息
ERROR	Varchar(120)		错误信息
Runtime	Datetime		记录时间

9) 操作日志表如表 4-10 所示。

表 4-10 操作日志表 devgroupinfo

字段名	类型	说明	备注
logID	Int	Auto Increment	唯一标识 id
roleID	Int		操作人
opinfo	Varchar(120)		操作内容
opstatus	Int		状态

4.9 本章小结

本章监控管理平台总体设计，根据系统需求、并结合系统设计原则，对存储监控管理平台进行总体设计，包括系统的逻辑结构、功能架构设计，系统安全机制设计、接口设计和数据库设计等内容。

下一章监控管理平台详细设计与实现，该章节主要描述系统各模块的详细设计与实现内容。

第五章 监控管理平台详细设计与实现

本章将根据系统需求分析的结果,对该系统的框架设计进行阐述,同时将根据系统整体的功能需求,将系统分解成为多个模块后再确定每个模块的特征,给出模块的调用关系,设计模块的界面。另外,还将对系统重要的模块接口设计进行介绍。

5.1 系统实现概述

云存储监控管理平台是为系统管理员提供一个能够通过可视化的监控管理平台,将云存储 IT 资源池中的服务器、应用系统和网络的状态等内容进行集中式的监控,实现高效的运维管理。通过不同告警阈值的设置,使各系统在故障风险较大的时候,能通过颜色、声音或者短信等方式向系统管理员报警通知,以便能够让系统管理者及时、准确地排除系统险情或故障,确保云环境下的每个系统能提供持久、稳定的应用服务。本系统选择 MFS 分布式文件系统作为云存储平台,监控管理平台将基于 MFS 进行设计与实现。MFS 文件系统结构包含三种角色:

管理服务器 managing server (master): 主要负责对每个云数据存储服务器的管理,包括对文件的读写调度,文件空间的回收以及恢复,以及多节点拷贝等。

数据存储服务器 data servers (chunkservers): 主要负责连接管理服务器,并服从管理服务器的调度,为应用系统提供存储空间。

客户机挂载使用 client computers: 通过系统提供的内核接口,挂接到管理服务器上所管理的数据存储服务器。

系统实现采用分层的处理方法,遵循规范将系统分为“表现层”、“逻辑层”、“数据访问层”三部分来实现:

数据访问层——完成和数据库的接口,执行基本的 insert、update、delete、select 等数据库操作。

逻辑层——业务逻辑的实现层,完成对业务处理过程的封装,执行对数据访问 DAO 对象的调用。

表现层——完成和用户的交换,包括用户请求数据的接收和处理结果的展现。

系统涉及到的业务对象都需封装为 DataEntityBean (一个普通的 JavaBean),表现层与逻辑层以及数据访问层直接的数据交换,涉及到业务数据的都必须需以封装好的 DataEntityBean 来传递,例如:数据访问层 DAO 对象检索数据库后得到的数据需构造为一个 DataEntityBean 对象,然后将此对象返回给业务对象;业务

对象要更新一个对象时，提交给 dao 对象相应的 DataEntityBean 对象，dao 对象读取此对象的属性完成业务数据的持久化。在采用以 JSP 为中心的方法后，表现层依然要对 jsp 页面中的代码进行控制，原则上 jsp 页面中只允许有数据格式化的代码，例如：列表页面的数据循环显示等。其他代码，例如请求数据的读取，业务对象的创建，业务方法的调用，包括显示数据的准备都由配合 jsp 页面工作的 HelperBean 来完成，系统现提供了一个名为 JSPHelper 的模版类，其实现了请求参数的解析，以及请求的分发两项功能，可有效的减小编码量，所有 HelperBean 都应从此类继承。

5.2 系统开发环境

系统开发环境搭建所需的软硬件设备清单如表 5-1 所示：

表 5-1 开发环境软硬件清单

序号	类型		版本/配置
1	硬件	CPU	Intel 双核 2.6G
		内存	4GB
		硬盘	希捷 300GB
2	软件	宿主操作系统	Windows xp 32 位
		虚拟机软件	VMWare Workstation 8.0

管理平台采用 Myeclipse6.5 进行编译、调试等操，并使用 SVN 进行源代码管理，使用 SecureCRT 等 SSH 工具远程登录虚拟机系统。

5.3 功能模块详细设计

管理平台包含了几大类功能模块，其中部分功能模块主要是对数据库的操作，业务流程和处理逻辑比较简单，本文档将不再详细阐述。另外几大功能模块包括：设备管理、监控代理程序、监控策略管理、数据统计分析。

5.3.1 初始化

初始化模块完成系统运行所必需的数据的创建工作，包括各种参数的配置和系统三员的员创建等，初始化流程如图 5-1 所示：

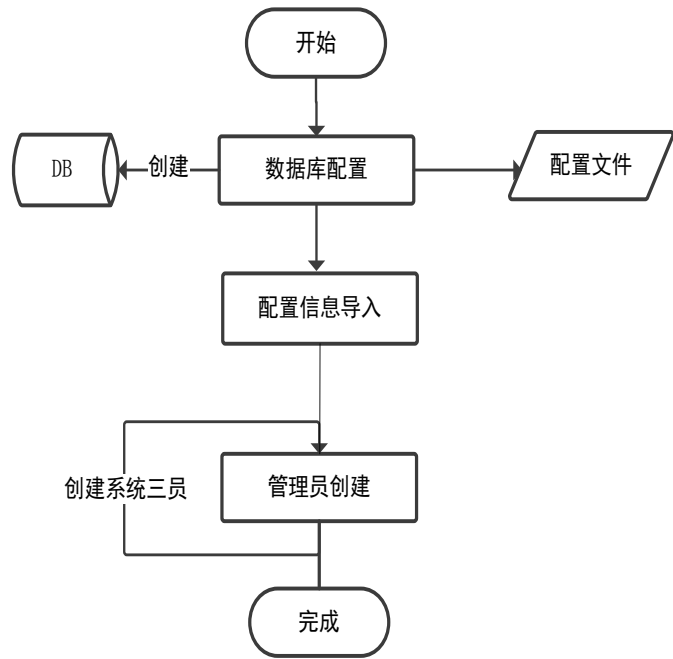


图 5-1 初始化流程图

系统初始化时，首先根据数据库配置文件创建数据库，包括数据库表结构和程序所需的视图等数据库内容；

系统创建管理员，用于运行时对系统进行日常的管理工作；创建安全管理员，用于管理平台上各种策略变更、模板变更的审批；创建安全审计员，用于对各类日志信息的审计与审核，通过三权分立，确保管理平台安全、有序得进行各种操作，维护运存储平台的稳定。

系统初始化完成后，系统管理员通过设置的登录账户和口令，可进入管理平台进行各种监控部署。

5.3.2 配置管理

配置管理模块按照业务、设备重要程度和管理域的方式对业务和设备进行统一配置管理，提供便捷的添加、修改、删除、查询功能，便于管理员能方便地查找所需的业务和设备信息，并对业务和设备属性进行维护。

基于动态业务和设备属性技术，用户可以对业务和设备属性进行扩展，例如可以根据用户自身的需要为业务和设备增加重要的属性信息。同时，所有扩展的属性都可以作为查询条件进行检索，便于管理员进行查询和维护。

配置管理功能包括：用户信息管理、组织机构管理、应用系统管理。以应用系统注册流程为例，操作流程图如图 5-2 所示：

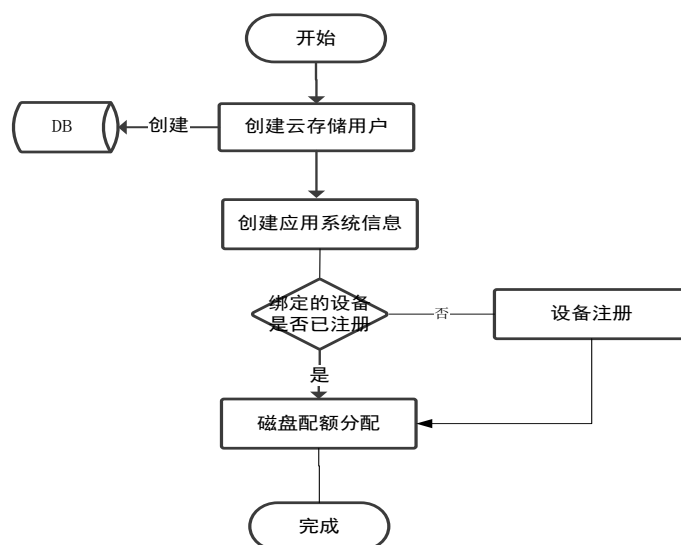


图 5-2 应用系统注册流程图

云用户管理功能模块包括云接入管理和接入状态管理。云接入管理负责对接入云存储系统的用户（或应用系统）进行管理，包括对用户名、用户 ID、磁盘配额、权限分配等功能项进行添加、修改、删除操作。接入状态管理负责对接入云存储系统的用户（或应用系统）每个端口的链接状态，如地址和用户目录进行管理。

5.3.3 设备管理

设备管理的前提是先对接入存储平台的设备进行设备注册，设备注册的流程如下，以阵列注册为例，设备注册流程如图 5-3 所示：

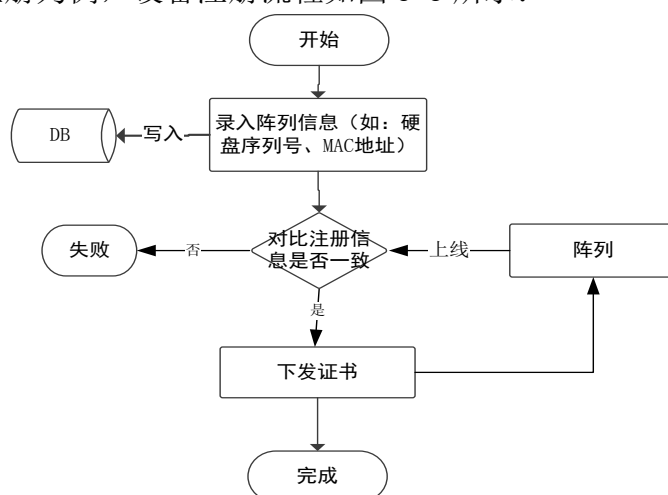






图 5-3 设备注册流程图

在监控管理平台上选择预录设备的关键信息，如硬盘序列号、MAC 地址等。在

阵列上线时，管理平台负责对收集到的设备的关键信息和预录信息进行比较，比对一致，则判定该设备属于合法，否则判定是非法阵，对审核不合法的阵列返回注册不成功信息，对审核合法的阵列返回注册成功信息。

5.3.4 拓扑管理

拓扑管理可为管理者提供多视角的监控，一目了然监控所管理对象的当前运行状态，直接进行故障排查。通过可视化的拓扑图，将设备状态、网络状态直接通过告警信息的形式展现到拓扑图上，给系统管理员提供一个可视化的管理平台，可以直观地、快速地定位到故障信息和设备，提高整个云存储平台的服务能力和故障处理能力。

实时监控将整个云存储平台系统按照业务和设备进行分类划分，根据分类划分结果进行实时信息监控。系统提供逻辑视图、物理视图两种实时监控页面和多种不同的图形化及文字报警方式，包括正常 、警告 、错误  和未知 。用户可以根据实际情况对实时监控页面进行切换，对实时监控项和图形化统计项可以进行自定义设置。系统管理员若发现实时监控信息中出现某个设备显示告警信息，可通过导航树快速定位告警设备。在子节点下，找到告警设备，点击设备名称，可在下拉列表中显示出对应设备的详细信息，包括 IP 地址，物理位置等。若要查看具体异常或报警信息详情，请点击相关的业务，进入物理视图中查询。

物理视图显示的是该业务中心涉的网络、设备状态。若系统监测到业务中心设备出现异常，该设备会显示告警状态（设备将不停闪烁）。针对出现异常信息的设备图标时，用户可以点击设备图标实时查看当前设备的状态和信息。若设备出现告警信息，对应设备上将出现不停闪烁的错误标志“X”，意在提醒管理员，该设备出现故障，需处理。物理拓扑图监控界面设计如图 5-4 所示。



图 5-4 物理拓扑界面设计

5.3.5 监控告警管理

监控策略管理功能模块包括监控项管理和报警策略管理。监控项是指云存储监控管理平台针对接入云平台的设备收集的 设备系统运行状态信息、网络状态等数据。监控项信息参照表如表 5-2 所示：

表 5-2 监控项参照表

序号	监控项	详细描述
1	设备是否上线	针对已注册的云存储系统设备的接入状态进行监控
2	设备 IP 地址	针对设备 IP 地址进行实施监控
3	设备内存使用率	针对设备的内存使用率进行监控
4	数据节点 CPU 占用率	针对数据节点的 CPU 使用率进行监控
5	数据节点 CPU 空闲率	针对数据节点的 CPU 空闲率进行监控
6	磁盘阵列盘位异常数	针对磁盘阵列盘位异常数量进行实时监控
7	阵列的网络吞吐率	针对云存储平台接入的磁盘阵列的网络吞吐率进行监控
8	服务器上线时间	针对服务器上线运行的时长进行记录监控
9	云存储当前文件总数量	针对云存储当前文件总数量进行监控

监控策略是指系统根据收集到的监控项信息进行分析，依据设定的监控策略，计算出对应的告警值。报警策略是指系统将云存储平台上的收集到的监控信息进行数据分析后，依据报警策略对不同故障使用不同的方式展现到监控管理平台上。报警策略信息参照表如表 5-3 所示。

表 5-3 报警策略参照表

序号	报警策略名称	详细描述
1	Unknown	未知信息，设备长时间未上报信息
2	Fatal	致命信息，致命的错误，会影响到系统正常运转；
3	Error	错误信息，一般的错误，不会影响到系统正常运转；
4	Warn	警告信息，警告系统运行中的不当状态，但不影响设备正常运转；

报警策略也可以根据实际运行环境对系统和安全的不同要求，自定义报警的策略，调整不同策略下告警信息的响应程度。

5.3.6 统计分析

统计分析功能模块功能是指通过监控收集到的告警信息、设备信息等，通过各种报表格式，如饼图、趋势图、柱状图等方式，直观地反应出云存储系统一段时间内的设备故障率、文件存储率等系统状态，该报表可以打印出来，作为上报给领导层的一种凭证。

管理员可以根据实际的统计需求，对统计报表进行自定义上传和配置。配置成功后，统计报表信息将在实时监控页面中进行显示，也可以在统计分析中进行查看，满足管理员对信息统计和分析需求。

5.3.7 日志管理

日志管理功能模块是系统日志审计核心部分，审计员可以通过日志管理对系统日志、操作日志进行事后审计和追踪，作为日志审计的依据。系统日志包括操作日志、监控日志和运行日志。日志管理提供了强大、完善的日志查询和检索功能，满足审计员对日志的审计和查询需求。

5.4 接口详细设计

系统接口主要包含各功能模块的输入输出接口，系统内部调用接口，以及底层模块与文件系统和数据库交互。表现层模块与操作人员交互，这部分接口采用

通用、成熟的接口设计，这里将不再详述。本章节重点介绍云存储连接配置接口、代理程序调用接口、用户管理模块调用接口和管理平台日志调用等接口设计。

5.4.1 链接配置接口设计

配置云主控服务器地址，用于配置管理接口连接服务器。

1) 接口函数: `int scmgr_addserv(char *msip, char *msport)`

参数:

Msip: 入参，云一台主控服务器地址

Msport: 入参，云主控服务器端口，缺省 9421

返回: 0 表示成功，其他表示失败代码

获取配置的主控服务器地址

2) 接口函数: `int scmgr_getserv(char *msip[32], char *msport[32])`

连接认证，并认证管理操作端，认证成功后才能进行操作。

`Int scmgr_register(char *loginname, char *passworddata, int pwrlen, int mode);`

参数:

Loginname: 入参，登陆帐号，缺省值: 000000

Passworddata: 入参，口令，如果采用证书认证为 KEY 口令

Pwrlen: 入参，口令长度

Mode: 入参，定义如下:

`#define VERIFYMODE_PASSWORD 1 //口令方式`

`#define VERIFYMODE_CERT 2 //证书方式`

`#define VERIFYMODE_ALL 4`

返回: 大于 0 的连接编号，小于等于 0 失败。

连接指定的服务器(针对节点管理)

3) 接口函数: `int scmgr_connect(char *ip, char *port, char *loginname, char *passworddata, int pwrlen, int mode)`

参数:

Ip: 入参，一台云服务器地址

Port: 入参，云服务器端口，port: 9422

返回: 大于 0 的连接编号，小于等于 0 失败。

关闭指定的连接

4) 接口函数: `void scmgr_closeconnect(int sockfd);`

参数: Socketfd: 入参，连接编号

5.4.2 代理程序接口设计

代理程序接口为内部调用接口，将通过代理程序调用数据节点上的管理接口。获取本机节点类型(本地配置)，代理调用后需要判断流程使用。

1) 接口函数：int scmgr_nodetype(int sockfd)

参数：sockfd：入参，指定服务器连接字，参数 0 表示本机

返回：

//一体化设备

```
#define NODE_TYPE_YTHNODE 1
```

//主控

```
#define NODE_TYPE_MASTER 2
```

//数据节点

```
#define NODE_TYPE_DATA 4
```

//配置服务器

```
#define NODE_TYPE_CONFIG 8
```

//配置服务器备机

```
#define NODE_TYPE_CONFIGBACK 16
```

若主控、数据节点和配置服务器装在一起，节点类型可以是组合。获取云存储平台的统计信息，包括文件数量、数据块数、总空间、剩余空间。

2) 接口函数：Int scmgr_cloudinfo(int sockfd, char *getinfostruct);

参数：

Socketfd：入参，主控连接句柄

Getinfostruct：出参 getinfostruct 结构体

typedef struct

```
{
uint64_t totalspace, availspace, trspace, respace;
uint32_t trnodes, renodes, inodes, dnodes, fnodes;
uint32_t chunks, chunkcopies, tdcopies;
uint16_t vermj;
uint8_t vermid, vermin;
} getinfostruct;
```

返回：0—返回成功；非 0—返回失败。

获取一个节点(包含主控/数据节点)正使用的存储空间列表和状态，最好在主控上调用。

3) 接口函数:

```
int scmgr_gethdlistbegin(int sockfd,int *count);
```

```
int scmgr_gethdlist(int sockfd,struct gethdinfostruct *phdinfostruct)
```

参数:

Socketfd: 入参, 指定服务器连接字, 主控连接句柄, 0 表示本机

返回: 1—需要继续调用 scmgr_gethdlist () 获取下一条记录

```
struct gethdinfostruct
```

```
{
```

```
char *hostip[32];
```

```
char *port[12];
```

```
char dev[256]
```

```
char path[256];
```

```
uint8_t areaid;
```

```
uint8_t type;
```

```
uint8_t ismount;
```

```
uint8_t damaged;
```

```
uint64_t chunkid;           //只有数据节点存在该值
```

```
uint32_t timestamp,chunkcount; //只有数据节点存在该值
```

```
uint64_t used,total;       //只有数据节点存在该值
```

```
};
```

4) 接口函数: void scmgr_GetSysInfo(int sockfd ,InitCPU *in, cprStats *stats)

参数:

Socketfd: 入参, 指定服务器连接字, 参数 0 表示本机

in—入/出参 CPU 结构

stats—入/出参 内存结构

```
typedef struct {
```

```
    long usec0;
```

```
    long nsec0;
```

```
    long ssec0;
```

```
    long isec0;
```

```
} InitCPU;
```

```
typedef struct {
```

```

        int cp;
        int rm;
        int ps;
        int pr;
        int dr;
        int dw;
    } cprStats;

```

返回：0—返回成功 非 0——返回失败。

获取设备在线信息，通过主控节点调用

5)接口函数：Int scmgr_getonlinebegin(int sockfd ,int *count)

```
int scmgr_getonline(int sockfd ,getmountinfostruct *info);
```

参数：

sockfd—入参，主控连接句柄

getmountinfostruct—出参 getmountinfostruct 结构

```
typedef struct
```

```

{
    uint32_t sessionid;
    uint16_t vermj;
    uint8_t vermid,vermin;
    uint32_t ip;
    uint8_t sesflags;
    char rootdir[256];
    char userid[256];
}getmountinfostruct;

```

返回：1—需要继续调用 scmgr_getonline（）函数获取下一条记录。

5.4.3 用户管理接口设计

1) 添加用户

接口函数：int scmgr_useradd(int sockfd, char *org,char *ou,char *userid,char *password,char *username,unsigned int goal,int quota,int role);

参数：

sockfd—入参，主控连接句柄

userid—用户 ID, 关键字, 最大 40 长度[关键字]

password—用户口令, 最大 32 长度, 无用

username—用户名, 最大 120 长度

goal—存储冗余

quota—配额, 单位 M

返回: 0—成功, 其他—失败代码

2) 配置用户口令

接口函数: `int scmgr_userpassword(int sockfd, char *userid, char *oldpassword, char *newpassword);`

参数:

sockfd—入参, 主控连接句柄

userid—用户 ID, 关键字, 最大 40 长度[关键字]

oldpassword—旧口令, 初始状态口令。

newpassword—新口令, 如果为空, 表示不修改初始口令。

返回: 0—成功, 其他—失败代码

3) 修改用户信息

修改 userid 用户的信息, 要修改的用户必须已存在。

接口函数 `int scmgr_usermodify(char *userid, char *username, int goal, int quota);`

参数:

sockfd—入参, 主控连接句柄

userid—入参, 用户 ID, 最大 40 长度

username—入参, 用户名, 最大 120 长度

goal—入参, 存储冗余

quota—入参, 配额, 单位 M

返回: 0—成功, 其他—失败代码

4) 删除用户

接口函数: `int scmgr_userdel(int sockfd, char *userid);`

参数:

sockfd—入参, 主控连接句柄

sockfd—入参, 连接编号

userid—入参, 用户 ID

返回: 0—成功, 其他—失败代码

5.4.4 日志接口设计

日志接口包括：操作日志、系统运行日志。管理员操作日志与来源相同，本文不做描述。云存储运行日志信息包含：编号、信息、类型、状态、日期、节点IP。获取运行日志，并自动删除已获取到的日志。

接口函数：int scmgr_getanddellog(sc_run_log *log);

socketfd—主控连接句柄

返回：

0—获取成功，其他—失败

struct sc_run_log

```
{
  Int id;
  Int type;
  Int status; //0—information,1—warning,2--error
  Unsigned long date;
  char ip[32];
  char *info;
} sc_run_log;
```

5.4.5 数据库接口设计

在 J2EE 中，不仅可以使⽤ JDBC 来访问不同的数据库，还可以使⽤对象关系映射器来完成业务逻辑层的对象到数据库的映射。采⽤官方的对象关系映射器是 CMP，使⽤ CMP 时，除了要考⻯领域模型外，还要考⻯ EJB 容器本身的特性。

业务逻辑层的对象尽量不采⽤ EJB 的 CMP，而采⽤ POJO (Plain Old Java Objects, 普通 Java 对象)。采⽤ POJO 的领域模型易于组装、创建快捷、可在 EJB 容器之外测试和运行。

POJO 对象与数据库之间采⽤对象关系映射器来实现对象到关系数据库的映射，但考⻯性能问题，将来不排除在业务逻辑层和持久层之间局部使⽤表模块模式。系统采⽤开源的 Hibernate 来作为对象关系映射器和数据库访问接口。

5.5 部分系统功能模块实现

5.5.1 MFS 文件系统的配置过程

云存储平台环境的搭建过程，最重要的一步就是分布式文件系统的编译与配

置，MFS 分布式文件系统的配置过程主要包括主控服务器的配置、元数据日志服务器的配置和数据存储服务器的配置等几个步骤。

5.5.1.1 主控服务器的配置

主控服务器的配置过程如下：

1) 下载 GNU 源码：wget http://www.moosefs.com/files/mfs-1.6.12.tar.gz

2) 解包：tar zxvf mfs-1.6.12.tar.gz

3) 切换目录：cd mfs-1.6.12

4) 创建用户（redhat）：

```
#useradd --comment 'Used for mfs file stroage serv use' --gid backup -r --shell
```

```
/usr/sbin/nologin --uid 10020 mfs
```

5) 配置

```
./configure --prefix=/usr --sysconffdir=/etc/mfs --localstatedir=/var
```

```
--with-default-user=mfs --with-default-group=backup
```

6) 编译安装：make ; make install

7) 配置：配置文件位于安装目录/etc/mfs，mfsmaster.cfg 的配置过程如下：

```
[root@dbserv mfs]# cp -av mfsmaster.cfg.dist mfsmaster.cfg
```

```
[root@dbserv mfs]# vim mfsmaster.cfg
```

```
# WORKING_USER = mfs
```

```
# WORKING_GROUP = mfs
```

```
# SYSLOG_IDENT = mfsmaster
```

```
# LOCK_MEMORY = 0
```

```
# NICE_LEVEL = -19
```

```
# EXPORTS_FILENAME = /etc/mfs/mfsexports.cfg
```

```
# DATA_PATH = /var/mfs
```

```
# BACK_LOGS = 50
```

```
# REPLICATIONS_DELAY_INIT = 300
```

```
# REPLICATIONS_DELAY_DISCONNECT = 3600
```

```
# MATOML_LISTEN_HOST = *
```

```
# MATOML_LISTEN_PORT = 9419
```

```
# MATOCS_LISTEN_HOST = *
```

```
# MATOCS_LISTEN_PORT = 9420
```

```
# MATOCU_LISTEN_HOST = *
```

```
# MATOCU_LISTEN_PORT = 9421
# CHUNKS_LOOP_TIME = 300
# CHUNKS_DEL_LIMIT = 100
# CHUNKS_WRITE_REP_LIMIT = 1
# CHUNKS_READ_REP_LIMIT = 5
# REJECT_OLD_CLIENTS = 0
8)启动 master server
```

5.5.1.2 元数据日志服务器的配置

元数据日志服务器的配置过程如下：

1) 下载 GNU 源码： `wget http://www.moosefs.com/files/mfs-1.6.12.tar.gz`

2) 解包： `tar zxvf mfs-1.6.12.tar.gz`

3) 切换目录： `cd mfs-1.6.12`

4) 创建用户 (redhat)：

```
#useradd --comment 'Used for mfs file stroage serv use' --gid backup -r --shell
```

```
/usr/sbin/nologin --uid 10020 mfs
```

5) 配置

```
./configure --prefix=/usr --sysconfdir=/etc/mfs --localstatedir=/var
```

```
--with-default-user=mfs --with-default-group=backup
```

6) 编译安装 `make;make install`

7) 配置：该服务配置文件是 `mfsmetallogger.cfg`，配置过程如下。

```
[root@mail mfs]# vi mfsmetallogger.cfg
```

```
# WORKING_USER = mfs
```

```
# WORKING_GROUP = mfs
```

```
# SYSLOG_IDENT = mfsmetallogger
```

```
# LOCK_MEMORY = 0
```

```
# NICE_LEVEL = -19
```

```
# DATA_PATH = /var/mfs
```

```
# BACK_LOGS = 50
```

```
# META_DOWNLOAD_FREQ = 24
```

```
# MASTER_RECONNECTION_DELAY = 5
```

```
MASTER_HOST = 192.168.1.34
```

```
# MASTER_PORT = 9419
```

```
# MASTER_TIMEOUT = 60
# deprecated, to be removed in MooseFS 1.7
# LOCK_FILE = /var/run/mfs/mfsmetalogger.lock
8)启动: 启动 metalogger 服务
[root@mail sbin]# ./mfsmetalogger start
working directory: /var/mfs
lockfile created and locked
initializing mfsmetalogger modules ...
mfsmetalogger daemon initialized properly
```

5.5.1.3 数据存储服务器的配置

数据存储服务器的配置过程如下:

- 1) 下载 GNU 源码: `wget http://www.moosefs.com/files/mfs-1.6.12.tar.gz`
- 2) 解包: `tar zxvf mfs-1.6.12.tar.gz`
- 3) 切换目录: `cd mfs-1.6.12`
- 4) 创建用户 (redhat):
`#useradd --comment 'Used for mfs file stroage serv use' --gid backup -r --shell`

`/usr/sbin/nologin --uid 10020 mfs`

5) 配置:

```
./configure --prefix=/usr --sysconfdir=/etc/mfs --localstatedir=/var
--with-default-user=mfs --with-default-group=backup
```

6) 编译安装 `make; make install`

7) 配置文件:

需要的配置文件包括: `mfschunkserver.cfg` 和 `mfsbdd.cfg`, `mfschunkserver.cf` 是主配置文件, `mfsbdd.cfg` 是服务器用来分配给 MFS 使用的空间, 选择一个单独的硬盘, 详细配置如下:

(1) `mfschunkserver.cfg` 的配置

```
[root@mail mfs]# vim mfschunkserver.cfg
# WORKING_USER = mfs
# WORKING_GROUP = backup
# DATA_PATH = /var/mfs
# LOCK_FILE = /var/run/mfs/mfschunkserver.pid
```

```
# SYSLOG_IDENT = mfschunkserver
# BACK_LOGS = 50
# MASTER_RECONNECTION_DELAY = 30
MASTER_HOST = 192.168.1.34
MASTER_PORT = 9420
# MASTER_TIMEOUT = 60
# CSSERV_LISTEN_PORT = 9422
# CSSERV_TIMEOUT = 60
# CSTOCS_TIMEOUT = 60
# HDD_CONF_FILENAME = /etc/mfs/mfshdd.cfg
```

(2)mfshdd.cfg 的配置

```
[root@mail etc]# more mfshdd.cfg
/home/data/mfs/
```

在这里/home/data/mfs/是一个给 mfs 的分区，但在本机上是一个独立的磁盘的挂载目录，用 `chown -R mfs:backup /home/data/mfs/` 把属主改变为 mfs。

8、启动 mfschunkserver

5.5.2 监控管理模块界面设计及关键代码

监控管理包括逻辑拓扑图的管理和物理拓扑图的管理，要通过切换功能，实现两种监控图形的动态监控。监控拓扑管理界面设计如图 5-5 所示：



图 5-5 拓扑管理模块界面设计

逻辑拓扑图背景图片可以由用户自定义，目的是为直观反映出各数据中心之间的地理位置，在系统初始化的时候，选择适合的图标拖拽到相应位置，保存。界面设计如图 5-6 所示：



图 5-6 逻辑拓扑界面设计

物理拓扑界面主要反映出数据中心的网络拓扑连接情况，这里必须将入网注册了的设备和分布式文件系统所部属的所有设备正确反映出来，界面设计如图 5-7 所示：

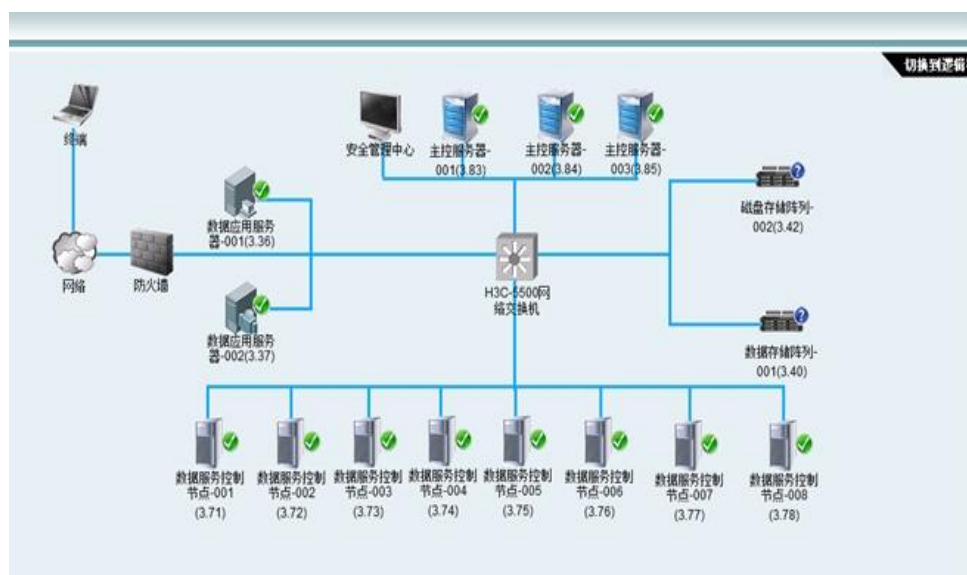


图 5-7 物理拓扑界面设计

监控管理界面设计如图 5-8 所示：

监控项列表			
<div>新增修改详情删除</div>			
<input type="checkbox"/>	序号	监控项名称	监控项描述
<input type="checkbox"/>	1	云存储系统客户端接入情况	针对云存储系统设备接入情况进行实时监控
<input type="checkbox"/>	2	代理IP地址	针对代理IP地址进行实时监控
<input type="checkbox"/>	3	设备IP地址	针对设备IP地址进行实时监控
<input type="checkbox"/>	4	SCADA安全设备告警状态	针对SCADA安全设备告警状态进行实时监控
<input type="checkbox"/>	5	SCADA安全设备部署位置	针对SCADA安全设备部署位置进行实时监控
<input type="checkbox"/>	6	SCADA安全设备工作状态	针对SCADA安全设备工作状态进行实时监控
<input type="checkbox"/>	7	服务状态	针对服务状态进行实时监控
<input type="checkbox"/>	8	磁盘阵列磁盘位异常数量	针对磁盘阵列磁盘位异常数量进行实时监控
<input type="checkbox"/>	9	磁盘阵列磁盘位正常数量	针对磁盘阵列磁盘位正常数量进行实时监控
<input type="checkbox"/>	10	磁盘阵列磁盘位空置数量	针对磁盘阵列磁盘位空置数量进行实时监控
<input type="checkbox"/>	11	数据库服务器当前状态	针对数据库服务器当前状态进行实时监控
<input type="checkbox"/>	12	用户在一段时间内点播视频流服务的平...	针对用户在一段时间内点播视频流服务的平均并发量进行实时监控
<div>(1页/4页 共44条) 首页 下一页 尾页</div>			

图 5-8 监控项管理界面设计

5.5.3 监控代理模块关键代码

以下是摘录的监控项管理模块的部分关键代码：

1) 获取 CPU 信息的结构体函数

```
typedef struct SYSINFO
{
char name[20];
unsigned int user;
unsigned int nice;
unsigned int system;
unsigned int idle;
}CPUStatusItem; //定义一个 CPUStatusItem 的结构体
```

2) 获取内存信息的结构体函数

```
typedef struct SYSINFO
{
char name[20];
unsigned long total;
char name2[20];
```

```

unsigned long free;
}MemoryInfo; //定义一个 MemoryInfo 的结构体
3) 获取服务器状态信息源代码:
get_memoccupy (MemoryInfo *mem) //对无类型 get 函数含有一个形参结构体
{
    FILE *fd;
    int n;
    char buff[256];
    MemoryInfo *m;
    m=mem;
    fd = fopen ("/proc/meminfo", "r");
    fgets (buff, sizeof(buff), fd);
    fgets (buff, sizeof(buff), fd);
    fgets (buff, sizeof(buff), fd);
    fgets (buff, sizeof(buff), fd);
    sscanf (buff, "%s %u %s", m->name, &m->total, m->name2);
    fgets (buff, sizeof(buff), fd);
    sscanf (buff, "%s %u", m->name2, &m->free, m->name2);
    fclose(fd);          //关闭文件 fd
}

int cal_cpuoccupy (CPUStatusItem *o, CPUStatusItem *n)
{
    unsigned long od, nd;
    unsigned long id, sd;
    int cpu_use = 0;
    od = (unsigned long) (o->user + o->nice + o->system +o->idle);
    nd = (unsigned long) (n->user + n->nice + n->system +n->idle);
    id = (unsigned long) (n->user - o->user);
    sd = (unsigned long) (n->system - o->system);
    if((nd-od) != 0)
        cpu_use = (int)((sd+id)*10000)/(nd-od);
    else cpu_use = 0;
    return cpu_use;
}

```



```
}

get_cpuoccupy (CPUStatusItem *cpust) //对无类型 get 函数含有一个形参结构体
{
    FILE *fd;
    int n;
    char buff[256];
    CPUStatusItem *CPUStatusItem;
    CPUStatusItem=cpust;
    fd = fopen ("/proc/stat", "r");
    fgets (buff, sizeof(buff), fd);
    sscanf (buff,"%s%u%u%u%u", CPUStatusItem->name,
    &CPUStatusItem->user, &CPUStatusItem->nice,&CPUStatusItem->system,
    &CPUStatusItem->idle);
    fclose(fd);
}

int main()
{
    CPUStatusItem cpu_stat;
    MemoryInfo mem_stat;
    int cpu;
    //获取内存状态
    get_memoccupy ((MemoryInfo *)&mem_stat);
    //获取 cpu 状态
    get_cpuoccupy((CPUStatusItem *)&cpu_stat1);
    sleep(10);
    //计算 cpu 使用率
    cpu=cal_cpuoccupy((CPUStatusItem*)&cpu_stat);
    return 0;
}
```

5.5.4 监控服务端关键代码

以下摘录的云存储监控服务端模块的部分关键代码：

```
Void
handle Get(void *pServer)
{
char sql[200];
packet pkt;
bool existed;
Server *ps = (Server *) pServer;
char fileName[DATABUFFLEN];
char userName[DATABUFFLEN];
ps->receiveData(fileName, DATABUFFLEN);
ps->receiveData(userName, DATABUFFLEN);
sprintf(sql, "SELECT * FROM OBJECT WHERE fileName='%s'", fileName);
ps->exec(sql, isFileExisted, &existed);
if(existed == true) {
ps->sendData("success", DATABUFFLEN);
}
else {
ps->sendData("fail", DATABUFFLEN); return;
}
sprintf(sql, "SELECT * FROM OBJECT WHERE fileName = '%s'and userName
= '%s' ORDER BY objectID ASC", fileName, userName);
ps->exec(sql, get, pServer);
pkt.isLast = true;
sendPacket(pServer, &pkt);
}
/* isFileExisted():查看文件是否存在 */
int
isFileExisted(void *data, int ncolums, char
**colNames) {
bool * t = (bool *) data;
if(ncolums > 0) {
```

```

    *t = true; }
Else {
    *t = false;
    return 0;
}
Int get(void *pServer, int ncolumns, char **colValues, char **colNames)
{
    Server *ps = (Server *)pServer;
    int length;
    int fd = open(colValues[SEGMENTNAME_INDEX], O_RDWR); if(fd == -1)
    return 0;
    fileOffset = atoi(colValues[OFFSET_INDEX]);
    length = atoi(colValues[LENGTH_INDEX]);
    if(lseek(fd, fileOffset, SEEK_SET) == -1) return 0;
    else {
        memset(pkt.data, 0, DATALEN);
        pkt.length = read(fd, pkt.data, length); pkt.isLast = false;
        sendPacket(pServer, &pkt);
    }
    close(fd);
    return 0;
}

```

5.6 本章小结

本章主要介绍了云存储监控管理平台的详细设计与实现，根据管理平台设计原则对云存储监控管理平台进行详细的系统设计，包括系统的总体架构、各主要功能模块、对外服务接口、数据库逻辑结构和物理结构等内容。并介绍了云存储监控管理模块中主要模块的工作流程以及部分功能模块的详细设计与代码实现。

下一章系统测试，主要介绍云存储监控管理平台的功能及性能测试方法和过程。

第六章 系统测试

本章节重点讲解系统测试环节。在整个项目开发过程中，系统测试是一个非常重要的阶段，必须使用软件测试方法和工具对于系统的可靠性、正确性和实现过程中的逻辑正确性以及系统中可能存在的问题进行验证。本章对云存储系统监控管理平台的软件测试涉及到功能测试、性能测试进行介绍，并对测试环境、测试方案以及测试用例进行了描述。

6.1 测试方法

云存储监控管理平台的测试主要采用多种测试方式共同检测系统的正确性和稳定性。用到的测试方法主要有白盒测试与黑盒测试，测试类型主要有单元测试和集成测试。

6.1.1 测试工具

管理平台测试所使用的测试工具是 Loadrunner 10.0，来做系统的压力测试。该工具能够很好的与软件开发环境结合，提供完整的单元测试与系统测试解决方案。LoadRunner 工具对应用程序进行压力测试，以隔离并标识潜在的客户端、网络和服务瓶颈。使用 LoadRunner 工具，可以设置在可控的负载条件下对管理平台进行系统测试。对于压力负载测试，LoadRunner 工具需要模拟运行分布在网络中的数千个虚拟用户（Vuser），测试工具可为这些 Vuser 提供相同条件下的、可重复并可度量的压力负载，模拟实际用户使用系统的真实场景。loadrunner 测试步骤如下：

- 1) 创建脚本：捕获在您的应用程序中执行的典型最终用户业务流程。
- 2) 设计场景：通过定义测试会话期间发生的事件，设置负载测试环境。
- 3) 运行场景：运行、管理并监控负载测试。
- 4) 分析结果：分析负载测试期间 LoadRunner 工具生成的性能数据。

6.1.2 测试模型

系统测试过程中，将测试模型设计为测试用机和被测系统两部分。测试工具部署在专门的一台测试 PC 机上运行，被测试的系统包括包括 Web 服务器（管理中心）、数据库服务武、云存储服务器（agent 客户端）。测试模型如图 6-1 所示：

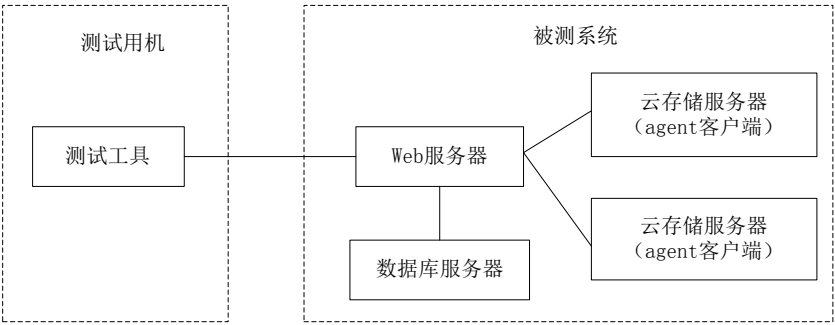


图 6-1 测试模型图

6.2 测试环境

测试环境示意图 6-2 所示：

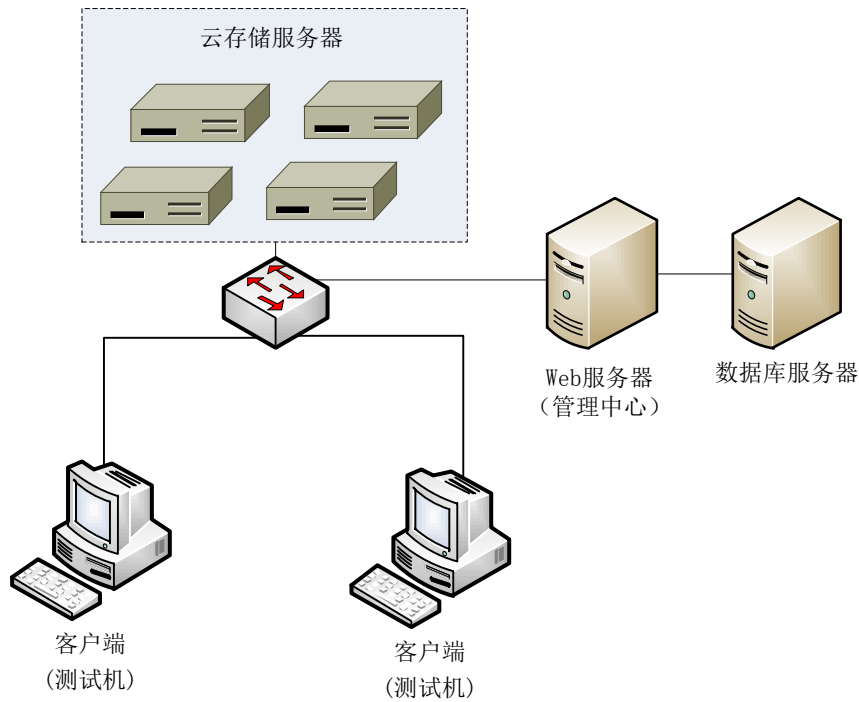


图 6-2 测试拓扑示意图

测试环境中设计了三台测试客户端，分别接入监控管理平台上进行测试。测试所需设备如表 6-1 所示：

表 6-1 测试所需仪器设备清单

资源名称	数量	型号或基本配置	备注
主控服务器	1 台	IBM3650: Xeon 5140 2.33GHz /16G/1T	
主控服务器（备机）	1 台	IBM3650: Xeon 5140 2.33GHz /16G/1T	
数据节点	2 台	IBM3650:Xeon 1.86GHz/16G/600G	
磁盘阵列	2 台	阵列 DFusion3000 16T/台	
交换机	1 台	华为 S5700-24TP-SIAC 千兆 24 口	
普通 PC	3 台	CPU: i3, 2G 以上内存; 硬盘: 300G 以上; 显示器 1 台; USB 口至少 1 个。	

测试所需软件清单如表 6-2 所示:

表 6-2 测试所需软件清单

软件名称	数量	备注
监控管理平台软件	1 套	
网盘应用系统	1 套	含客户端程序
性能测试软件	1 套	IO Meter
	1 套	Loadrunner 9.0

6.3 测试内容

针对监控管理平台测试的内容包括两部分: 功能性测试和性能测试。测试需求和用例全部依据系统需求进行编写。由于系统子模块较多, 本文只罗列了部分重要模块的测试用例表, 其余模块不做描述。

6.3.1 功能测试

1) 角色管理测试

角色管理测试的重点是要验证系统是否可以对角色进行新增等操作, 并验证角色权限是否正确, 测试内容表如下:

表 6-3 角色管理测试内容表

测试项目	测试要求	测试步骤	预期结果	测试结果
对角色模块进行增、删、改操作测试	系统管理员使用身份口令登录系统；	点击“新增角色”，输入角色信息，然后点击“保存”；	增、删、改操作都能正常完成；	与预期结果一致
	安全员使用身份口令登录系统	在列表中选择一个角色信息，点击“修改”，输入修改信息，然后点击“保存”；	安全员可以对修改的角色进行审核；	
		在列表中选择一个角色信息，点击“删除”，输入修改信息，然后点击“保存”；		
		安全员登录之后，对新增(或修改过的)角色进行审核操作。		

2) 逻辑拓扑测试

逻辑拓扑功能模块主要是实现监控视图的切换，因此重点测试逻辑拓扑是否可以正常切换到物理拓扑监控图，并且需测试逻辑拓扑图是否可以同步显示各网络的运行状态，测试内容如表 6-4 所示。

表 6-4 逻辑拓扑测试内容表

测试项目	测试要求	测试步骤	预期结果	测试结果
逻辑拓扑管理测试	系统管理员创建一个逻辑拓扑	选中创建的逻辑拓扑，选择“编辑”，在弹出的页面中，选择逻辑拓扑的背景图片；	逻辑拓扑的背景图片可以更换；	与预期结果一致。
		选择完背景图片并保存后，对逻辑拓扑图进行编辑操作；	逻辑拓扑图的图标可以选择和保存操作；	
		编辑完成后，将该逻辑拓扑图绑定为本中心拓扑，点击“实时监控”查看是否绑定成功。	逻辑拓扑图可以成功绑定为本中心拓扑；	
		手动添加一条设备告警 I 信息，看逻辑拓扑图是否能及时弹出报警提示信息。	能及时弹出设备告警信息	

3) 物理拓扑测试

物理拓扑功能模块主要是最直观反映系统运行状态的，测试过程中重点测试物理拓扑是否可以及时显示各被监控设备的运行状态，出现告警信息后，通过“右键”是否可以看到监控数据。测试内容如表 6-5 所示：

表 6-5 物理拓扑测试内容表

测试项目	测试要求	测试步骤	预期结果	测试结果
物理拓扑测试	系统管理员创建一个物理拓扑，并将该物理拓扑绑定到之前创建的逻辑拓扑上	选中创建的物理拓扑，选择“编辑”，在弹出的页面中编辑物理拓扑图，并保存；	物理拓扑可以进行设备添加和连线；	与预期结果一致。
		保存完成后，点击“实时监控”，选择逻辑拓扑图后，点击进入物理视图；	逻辑视图和物理视图之间可以切换；	
		手动添加一条报警信息，观察告警设备的显示情况。	告警设备上弹出了告警信息，通过右键操作可以看到监控数据。	

4) 监控策略管理测试

该功能模块重点测试通过系统管理员操作是否可以根据各个接入应用运行环境的需要，设置响应的监控策略，测试内容如表 6-6 所示。

表 6-6 监控策略管理测试内容表

测试项目	测试要求	测试步骤	预期结果	测试结果
监控策略管理测试	设备完成入网注册，下发证书，监控项及策略配置完成	物理拓扑视图中，点击被监控的设备，在弹出的监控列表中查看设备上报的监控信息。	经过一次轮询后，主控服务器上报的信息是增加了一条监控项。	与预期结果一致。
		通过监控策略管理模块，主控服务器增加一条监控项。		

5) 操作日志管理测试

操作日志管理模块重点测试系统是否可以完整、正确记录管理员的所有操作行为，包括系统管理员、保密管理员和安全审计员。操作日志管理测试内容表如表 6-7 所示：

表 6-7 操作日志管理测试内容表

测试项目	测试要求	测试步骤	预期结果	测试结果
对系统管理员、保密管理员和安全审计员的操作日志进行测试	三员已进行过操作，产生操作日志	使用系统管理员账户登录，选择操作日志，选择查询时间段，查看日志列表中是否正确记录了操作日志； 使用保密管理员账户登录，选择操作日志，选择查询时间段，查看日志列表中是否正确记录了操作日志； 使用安全审计员账户登录，选择操作日志，选择查询时间段，查看日志列表中是否正确记录了操作日志。	操作日志能正确记录系统三员的每项操作行为	与预期结果一致

6.3.2 性能测试

性能测试是通过自动化的测试工具模拟多种正常、峰值以及异常负载条件来对系统的各项性能指标进行测试。负载测试和压力测试都属于性能测试，两者可以结合进行。通过负载测试，确定在各种工作负载下系统的性能，目标是测试当负载逐渐增加时，系统各项性能指标的变化情况^[30]。

根据云存储监控管理平台对系统性能的需求，web 页面登陆访问的响应时间高峰应控制在 5 秒内，高峰期应在 10 秒内，因此本节性能测试主要针对管理中心，页面登陆和数据查询做性能测试。利用压力测试工具 Loadrunner 录制好脚本，然后设置系统测试时间为 5 分钟，用户并发量从 2 个逐步增加到 10 个，并持续运行 5 分钟，然后逐步降低并发用户量，直至用户全部退出。最后得到的测试运行曲线截图如图 6-3 所示：

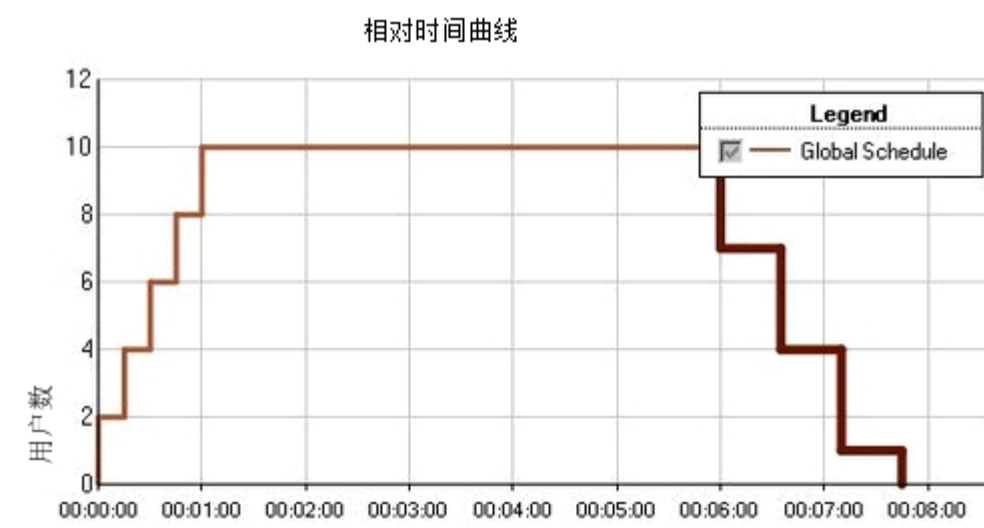


图 6-3 压力测试曲线图

并发测试统计数据表截图如图 6-4 所示：

测试时间 (分钟)	5										
并发线程数	每次时间间隔 (秒)	平均响应时间 (秒)	成功次数	失败次数	成功率	处理能力 (次/分)	web服务器CPU占用率(%)		数据库服务器CPU占用率(%)		
							平均	最大	平均	最大	
2	0	3.469	40	0	100.00%	8	34.45	47.15	40.16	50.67	
4	0	3.909	46	0	100.00%	7.2	32.62	48.96	44.41	55.33	
6	0	4.333	50	0	100.00%	10	43.37	53.65	48.73	61.67	
8	0	5.405	51	0	100.00%	10.4	42.93	56.85	50.72	70.23	
10	0	5.805	52	0	100.00%	10.8	43.85	59.54	54.34	72.25	

图 6-4 并发测试统计数据

说明：不断增加的并发线程数，系统处理的成功次数并没有降低，并且系统的 CPU 占用率维持在相对均衡的水平，说明系统基本达到了设计的处理能力。考虑到监控管理平台在实际使用环境下，操作人员并发量不大，因此测试中并发线程数未继续增加。

6.4 测试结论

经过对系统进行功能和性能测试，测试表明，云存储监控管理平台所有的功能模块运行正常，监控代理客户端数据上报准确，管理员可以通过告警信息准确找到相关设备，系统性能达到预期目标，云存储监控管理平台所实现的功能目标与预期结果一致。

6.5 本章小结

本章云存储监控管理平台测试，利用白盒测试和黑盒测试方法，搭建系统测试环境，并对系统的功能和性能进行了详细测试，充分验证了云存储监控管理平台的正确性。

下一章总结与展望，将对本次云存储监控管理平台设计与实现相关的工作进行总结，并对后续的工作进行初步规划和展望。

第七章 总结与展望

本文重点研究了云存储系统的监控管理平台的设计与实现，系统可以为企业和管理员的系统维护工作提供巨大的帮助，可以让管理员能够快速、准确地定位到故障服务器，及时排除系统故障，并且实现了系统的数字化和可视化功能，确保了存储系统的持久稳定运行。

7.1 总结

本文主要围绕开源分布式文件系统 MooseFS，以分布式文件系统为中心，从系统管理者的角度，将分布式文件的运行环境进行抽象，分析 MooseFS 原有管理模块在监控项和监控模式上存在的不足，确定云存储监控管理平台的研究方向与目标，最终实现的云存储监控管理平台能满足智能化告警管理、多角度试图展示、丰富的数据报表等特点。

云存储监控管理平台根据云存储系统的关键运行参数，扩展出更多的监控项，并采用更加统一直观、易于查看的方式将云存储的各种关键数据采集至监控管理平台，实时展现。利用监控代理模式，通过监控管理中心与客户端之间建立的安全通道采集数据，确保了系统的可扩展性、实时性，同时也确保了系统的安全性和数据的有效性。监控管理平台设计时考虑到后期的扩展性，采用模块化的结构模型，以适应不断扩展的云计算平台的应用需求。本系统各业务逻辑模块主要负责系统的业务逻辑功能，由业务处理服务完成其主要功能。业务处理服务以组件的形式对外提监控告警、设备管理、策略管理、网络管理和日志管理等功能。云存储管理平台的安全性将关系整个云服务的安全，本系统采用专用客户端程序与管理中心进行安全通信。系统的管理通过三员分权的方式，管理员间权力分离，相互制约，在管理上形成立体防护。从管理员登录到管理员退出，所有操作全部记录，事后可审计，可有效防止对操作行为的抵赖。

本项目实现的云存储监控管理平台把各类管理组件抽象为具有层次性和可扩展性的模型，从而在很大程度上简化了云存储监控管理平台系统本身设计的复杂性。我们可以将这类模型抽象和系统简化设计的方法，应用到软件工程开发的其他领域，有助于为系统设计和开发人员快速构建系统轮廓，简化工作量，提高开发效率。经过系统各项功能和性能测试验证，该模式能适应不同的分布式计算环境，可以为云存储模式下的按需服务和资源合理利用提供强力的管理支撑。

7.2 展望

由于云计算自身具有的特性，如虚拟性、层次性、动态性以及复杂性等特点，将会为云平台带来许多不断丰富应用系统接入，这必将会给云存储的监控平台的提出越来越高的要求，云平台管理除了要满足云存储的监控管理需求外，应用系统的监控也将给云平台管理提出新的课题，所以云计算平台的管理技术是个具有挑战和机遇的研究方向，许多问题还有待深入与研究。

在过去，数据中心的管理会配置一套专门的环境，每个数据中心会有一套管理系统来支撑。而未来云计算环境下，高效和节能将是主体，通过管理措施提高云平台的资源利用率、降低数据中心能耗将是未来云平台管理发展的重要趋势。目前商业化的云平台投入成本很高，对于很多中小企业而言，他们更愿意选择开源云计算平台作为企业云战略的基石。但开源的云平台所提供的功能相对简单，需要用户在开源基础上进行许多扩展，尤其是云计算管理系统，需要将管理整合得更加智能、更加精简、更加统一，已适应将来企业云应用程序的复杂化、多样化的发展，确保企业获得更高的生产力和成本效益。

本文中涉及的云存储管理平台在实现云存储资源系统的统一、集中、可视化方面做到了一定的效果，但由于时间和技术的原因，很多还需要改进的地方没有完善。比如如何让云平台的管理系统更加智能化，系统如何通过告警信息优先进行自我判断和修复等等，诸如此类的问题，需要我们在今后的工作和学习中继续思考和改进。同时，在云计算相关领域的研发成果的产品化方面，我们也做了许多努力，为了能尽快使研发成果产品化，在市场上取得优势，在研发过程中，我们采取了分步实施、阶段成果产品化的方式。在追求高效的同时，也将紧密关注云计算的相关技术标准和行业应用的发展趋势，使我们产品有持续的发展生命力。在国内云计算市场趋于成熟后，对市场进行细分，根据不同行业 and 不同类型用户的需求，适时实现产品的系列化，推出多种型号、不同版本的系列产品，满足多样性的用户需求，以优质的产品占领市场。

致 谢

光阴荏苒，硕士研究生的学习即将结束，三年的学习生活让我受益匪浅。经历一年多时间的磨砺，硕士毕业课题研究和论文编写工作终于完稿。回首这一年来收集、整理、思考、停滞、修改直至最终完成的过程，我得到了导师和同事非常多的关心和帮助，现在要向他们表达我最真挚的谢意。尤其要感谢对我的论文工作进行指导的陆鑫老师和李新老师，感谢他们在我的论文编写过程中对我进行了无私的指导和帮助，不厌其烦的帮助进行论文的修改和改进。另外，每门课程的授课老师和学院的学生工作老师也给我提供了很多方面的支持与帮助。在此向帮助和指导过我的各位老师表示最衷心的感谢！

另外，还要感谢这篇论文所涉及到的各位学者。本文引用了数位学者的研究文献，如果没有各位学者的研究成果的帮助和启发，我将很难完成本篇论文的写作。

感谢我的同学和同事，在我写论文的过程中给予我了很多支持，在论文的撰写和排版等过程中提供热情的帮助。

感谢成都卫士通公司，在整个课题研究过程中为我创造了一个非常良好的实践环境。

由于我的学术水平有限，所写论文难免有不足之处，恳请各位老师和学友批评和指正！

参考文献

- [1] 刘鹏. 云计算[M]. 北京: 电子工业出版社, 2010, 70-77
- [2] 王萍, 张际平. 云计算与网络学习[J]. 现代教育技术, 2008, 18(11):81-84
- [3] [Http://www.chinanews.com/it/2011/07-28/3215681.shtml](http://www.chinanews.com/it/2011/07-28/3215681.shtml)
- [4] Leavitt N, Is Cloud Computing Really Ready for Prime Time? [J]. IEEE Computer Society Press, 2009, 42(1):15-20
- [5] A.Cooke, A.J.G.Gray, W.Nutt, J.Magowan, M.Oervers, P.Taylor. The relational Grid Computing, 2(4): pp.323-339, 2004
- [6] 王京, 马英. 云计算的发展与安全问题[J]. 数据通信, 2010, (5):24
- [7] David Chappell. Introducing The Windows Azure Platform, 2009, 12
- [8] HEWITT C. ORGS for scalable, robust privacy friendly client cloud computing [J]. IEEE Internet Computing, 2008, 12(5):96-99
- [9] McGregor, H-W. Braun, J.A. Brown. The NLANR Network Analysis Infrastructure. IEEE Communications Magazine, 2000, Vol.38, (5):122-128
- [10] Cochran M, Witman P. Governance and Service Level Agreement Issues in a Cloud Computing Environment[J]. Journal of Information Technology Management, 2011, 22(2):41-55
- [11] S.Zanikolas and R.Sakellariou. A taxonomy of Grid monitoring Systems. Future Generation Computer Systems [J]. 2005, 163-188
- [12] 杨斌, 刘海涛. 云计算对移动互联网发展的助推作用[J]. 电信工程技术与标准化, 2010, 23(12):35
- [13] R.Sundaresan, Tahsin Kurcy, Mario Lauriaz, et al. Adaptive polling of Grid resource monitors using a slacker coherence model. In Proc. 12th IEEE International Symposium on High Performance Distributed Computing, June 003:260-269
- [14] 中国云计算发展现状与趋势[N]. 计世通讯, 2010年12月19日
- [15] Armbrust M, Fox A, Grith R, et al. Above the clouds: A Berkeley View of Cloud Computing [R]
- [16] L.Vaquero, L.Rodero-Marino, J.Caceres, M.Linder, "A break in the clouds: towards a cloud definition", SIGC- OMM Computer Communication Review, 2009, pages 137-150
- [17] 云计算关键领域安全指南 V2.1. 云安全联盟(CSA), 2009
- [18] 张志凯. 基于CS架构的主机信息监控系统的设计与实现[D]. 北京: 北京邮电大学, 2006, 32-36

- [19] WEISS A.Computing in the clouds[J].ACM Networker,2007,11(4):16-25
- [20] 张棋胜. 云计算平台监控系统的应用[D]. 北京: 北京交通大学, 2011, 23-27
- [21] [Http://webservices.ctocio.com.cn/wsjavtec/82/8124082.shtml](http://webservices.ctocio.com.cn/wsjavtec/82/8124082.shtml). IT 专家网论坛
- [22] Liang Chen,k.Reddy,and Gagan Agrawal.GATES: AGrid-Based Middleware for Processing Distributed Data Streams,13th.IEEE International Symposium on High Performance Distributed Computing(HPDC) 2004,22-31
- [23] Toby Velte, Anthony Velte, Robert Elsenpeter.”Cloud Computing,A Practical Approach”,Hill Education,2009,78-80
- [24] G.Fedak, C.Germain, V.Neri, and F.Cappello.Xtremweb: A generic global computing system.In Proceedings of the 1st IEEE International Symposium on Cluster Computing and the Grid,pages 582-587,2001,34-40
- [25] 陈潜, 刘云, 高利娟. 基于嵌入式 Linux 的机房远程监测系统研究[J]. 微计算机信息 2011. 06 (32): 60-65
- [26] 李明江. SNMP 简单网络管理协议[M]. 北京: 电子工业出版社, 2007, 45-47
- [27] 付保川, 班建民, 陆卫忠, 等. 基于嵌入式 Web 的远程监控系统设计[J]. 微计算机信息, 2005. 10:58-60
- [28] 胡圣明, 褚华. 软件设计教程 (第 2 版) [M]. 北京. 清华大学出版社, 2009, 55-56
- [29] 周志明, 谢小明. 深入理解 OSGi:Equinox 原理应用与最佳实践. 北京: 机械工业出版社, 2013, 67-72
- [30] 曾庆辉. 基于 Agent 的存储区域网络 (SAN) 智能管理技术[J]. 计算机系统应用, 2005, 6:52-55

作者: [李勇斌](#)
学位授予单位: [电子科技大学](#)
被引用次数: 1次

参考文献(7条)

1. [王萍, 张际平](#) 云计算与网络学习[期刊论文]-[现代教育技术](#) 2008(11)
2. [王京, 马英](#) 云计算的发展与安全问题[期刊论文]-[数据通信](#) 2010(05)
3. [杨斌, 刘海涛](#) 云计算对移动互联网发展的助推作用[期刊论文]-[电信工程技术与标准化](#) 2010(12)
4. [张志凯](#) 基于C/S架构的主机信息监控系统的设计与实现[学位论文]硕士 2006
5. [张棋胜](#) 云计算平台监控系统的应用研究[学位论文]硕士 2011
6. [付保川, 班建民, 陆卫忠, 刘文亮](#) 基于嵌入式Web的远程监控系统设计[期刊论文]-[微计算机信息](#) 2005(20)
7. [曾庆辉](#) 基于Agent的存储区域网络(SAN)智能管理技术[期刊论文]-[计算机系统应用](#) 2005(06)

引用本文格式: [李勇斌](#) 云存储监控管理平台研究与实现[学位论文]硕士 2014