**密级： 保密期限：**



**硕士学位论文**



**题目：一种适用于广电网的云应用服务管理方案的研究与实现**

**学 号： 2012111499**

**姓 名： 赵晓森**

**专 业： 计算机科学与技术**

**导 师： 王志谦**

**学 院： 网络技术研究院**

**2014年12月日**

**独创性（或创新性）声明**

本人声明所呈交的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。尽我所知，除了文中特别加以标注和致谢中所罗列的内容以外，论文中不包含其他人已经发表或撰写过的研究成果，也不包含为获得北京邮电大学或其他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了谢意。

申请学位论文与资料若有不实之处，本人承担一切相关责任。

本人签名： 日期：

**关于论文使用授权的说明**

学位论文作者完全了解北京邮电大学有关保留和使用学位论文的规定，即：研究生在校攻读学位期间论文工作的知识产权单位属北京邮电大学。学校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，允许学位论文被查阅和借阅；学校可以公布学位论文的全部或部分内容，可以允许采用影印、缩印或其它复制手段保存、汇编学位论文。（保密的学位论文在解密后遵守此规定）

保密论文注释：本学位论文属于保密在 年解密后适用本授权书。非保密论文注释：本学位论文不属于保密范围，适用本授权书。

本人签名： 日期：

导师签名： 日期：

一种适用于广电网的云应用服务管理方案的研究与实现

摘 要

在互联网发展越来越迅速、接触面越来广的今天，互联网为用户提供了更多的应用和内容。越来越多的广电网用户转向了互联网。经过近几年三网融合的发展，越来越多的家庭拥有具有双向交互能力的机顶盒。如何利用现有的平台开展新业务，为用户提供更多的应用选择和更丰富的内容，重新吸引用户的注意力，成为当前广电网运营商亟待解决的问题。

本课题提出了一种新的解决方案，即利用云的概念，将用户终端的处理能力与图像渲染能力剥离到云端，终端只需要具备基本的网络交互能力和音视频解码能力，由云端为用户完成应用程序的计算、处理以及图像的渲染，编码成音视频流，并将该音视频流经广电网络传送至终端，由终端为用户呈现。目前大多数家庭用机顶盒都具备该方案所需的硬件能力，广电网运营商不需要为用户进行大规模的终端升级即可提供更多的应用和更丰富的内容，并且完全可以实现广电网应用内容的可观可控以及应用版权的保护。

NGOD架构是应用于广电网的一种主要支持视频点播服务的架构。该架构合理细致地划分了各个模块的功能，详细规定了各个接口的规范，具有很强的可扩展性，能支持多种业务的扩展。为了在不影响现有业务的前提下部署云应用服务，本课题依照NGOD架构的规范，对其进行了扩展，增加了应用服务器、应用服务器管理器两个模块以及相应所需要的接口。

应用服务器为云应用提供运行平台，在其上实现了应用虚拟化技术，以便于应用的快速部署，并减少服务器端的消耗。应用服务器管理器负责管理分配应用服务器资源，本课题实现了一种资源分配策略，尽量充分利用应用服务器的资源，维持服务器间的负载均衡。

关键词：云应用服务 NOGD架构 负载均衡 广电网 应用虚拟化

RESEARCH AND IMPLEMENT OF PROGRAM OF CLOUD APPLICATION SERVICE MANAGEMENT APPLIED TO THE BROADCOST NETWORK

ABSTRACT

With more and more quick development, and more and more wide contact surface of the Internet, it provides much more application and content for users. More and more users of the broadcast network turn their attention to the Internet. After the development in recent years, more and more families have the set-top box with two-way interactive ability. How to use the existing platform to develop new business, to provide more choice of the applications and more rich content for the users, to attract the users’ attention back to the broadcast network, is the urgent problem to be solved for the broadcast network operators.

This paper proposes a new solution, that using the cloud concept, the terminal’s processing ability and image rendering capabilities are stripped to the cloud. The terminal only need to have the basic network interactive ability and the ability of decoding audio and video. The cloud complete the application’s calculation, processing and image rendering, then code the result into video stream. After that, the video stream is transferred to the terminal through the HFC network. The terminal will decode the stream and present it to the user. Most of the current domestic set-top boxes have the hardware capability required by the program. So the broadcast network operators don’t need to carry out large scale terminal upgrades for the users to provide more applications and more rich content. And this program can fully realize the broadcasting network application content considerable control and protect the copyright of the application.

The NGOD architecture is one that supports video on demand service used on broadcast network. This architecture divides the function into some modules carefully and reasonably. And it provides the specification of various interfaces detailedly. This architecture has strong scalability. So it can support a variety of business expansion. In order to protect the old services from affecting by the new business, this paper has carried on the expansion according to the NGOD architecture specification. We increase the application server module, the application server manager module and the interfaces needed by the new modules.

Application servers provide a platform for cloud applications running on. We implements application virtualization on the servers to deploy application rapidly and reduce the consumption. The application server manager is responsible for managing and allocating the resources of the application servers. This paper implements a resource allocation strategy to make full use of the resources of the application servers and maintain load balancing between servers.

KEY WORDS： cloud application service NGOD architecture load balancing broadcast network application virtualization

# 目录

[一种适用于广电网的云应用服务管理方案的研究与实现 I](#_Toc407114374)

[摘 要 I](#_Toc407114375)

[ABSTRACT II](#_Toc407114376)

[目录 IV](#_Toc407114377)

[第一章 绪论 1](#_Toc407114378)

[1.1 课题背景和意义 1](#_Toc407114379)

[1.2 课题研究可行性分析 1](#_Toc407114380)

[1.3论文研究目标及创新点 2](#_Toc407114381)

[1.4论文组织和结构 3](#_Toc407114382)

[第二章 对NGOD架构进行扩展 4](#_Toc407114383)

[2.1 NGOD架构 4](#_Toc407114384)

[2.2 VOD系统 6](#_Toc407114385)

[2.3 扩展NGOD架构 8](#_Toc407114386)

[2.3.1 云应用服务业务 8](#_Toc407114387)

[2.3.2 扩展NGOD架构 9](#_Toc407114388)

[2.3.3 RTSP协议 10](#_Toc407114389)

[2.3.4 VREP协议 11](#_Toc407114390)

[2.4 本章小结 12](#_Toc407114391)

[第三章 NGOD架构新增接口的定义 13](#_Toc407114392)

[3.1 定义S7接口 13](#_Toc407114393)

[3.1.1 S7 SETUP Request消息 14](#_Toc407114394)

[3.1.2 S7 SETUP Response消息 15](#_Toc407114395)

[3.1.3 S7 TEARDOWN消息 18](#_Toc407114396)

[3.1.4 S7 TEARDOWN Response消息 19](#_Toc407114397)

[3.1.5 S7 ANNOUNCE消息 19](#_Toc407114398)

[3.1.6 S7 ANNOUNCE Response消息 20](#_Toc407114399)

[3.2 定义R8接口 21](#_Toc407114400)

[3.2.1 R8 SETUP Request消息 22](#_Toc407114401)

[3.2.2 R8 SETUP Response消息 22](#_Toc407114402)

[3.2.3 R8 TEARDOWN Request消息 22](#_Toc407114403)

[3.2.4 R8 TEARDOWN Response消息 23](#_Toc407114404)

[3.2.4 R8 ANNOUNCE Request消息 23](#_Toc407114405)

[3.2.5 R8 ANNOUNCE Response消息 23](#_Toc407114406)

[3.3 定义D8接口 24](#_Toc407114407)

[3.3.1 OPEN消息类型 26](#_Toc407114408)

[3.3.2 UPDATE消息类型 28](#_Toc407114409)

[3.3.3 KEEPALIVE消息类型 31](#_Toc407114410)

[3.3.4 NOTIFICATION消息类型 31](#_Toc407114411)

[3.4 本章小结 33](#_Toc407114412)

[第四章 相关模块的设计与实现 34](#_Toc407114413)

[4.1 改进SM模块 34](#_Toc407114414)

[4.1.1 SM模块的结构设计 34](#_Toc407114415)

[4.1.2 SM模块的关键数据结构 37](#_Toc407114416)

[4.2 ASM模块的设计与实现 41](#_Toc407114417)

[4.2.1 ASM架构的研究与选择 41](#_Toc407114418)

[4.2.2 注册管理功能的实现 44](#_Toc407114419)

[4.2.3 资源分配功能的实现 46](#_Toc407114420)

[4.2.4 资源分配策略 49](#_Toc407114421)

[4.3 应用服务器组件的设计与实现 53](#_Toc407114422)

[4.3.1应用服务器相关接口 53](#_Toc407114423)

[4.3.2应用服务器端的虚拟化 54](#_Toc407114424)

[4.3.3 部署应用 54](#_Toc407114425)

[4.4本章小结 55](#_Toc407114426)

[第五章 系统测试 56](#_Toc407114427)

[5.1测试环境 56](#_Toc407114428)

[5.2测试用例和结果 56](#_Toc407114429)

[5.3本章小结 60](#_Toc407114430)

[第六章 总结和展望 61](#_Toc407114431)

[6.1全文总结 61](#_Toc407114432)

[6.2全文展望 61](#_Toc407114433)

[参考文献 62](#_Toc407114434)

[致谢 64](#_Toc407114435)

[硕士期间发表论文 65](#_Toc407114436)

# 

# 第一章 绪论

## 1.1 课题背景和意义

伴随着国内三网融合的不断推进，越来越多的用户家庭拥有具有双向交互能力的数字电视。同时由于电脑和手机、平板电脑等智能终端的迅猛发展，人们可以接触到的信息和应用服务越来越丰富，用户也越来越多的向互联网方向靠拢。国家新闻出版广电总局发展研究中心最新发布的《中国视听新媒体发展报告（2013）》显示受个人电脑、平板电脑、智能手机的冲击，北京地区电视机开机率从三年前的70%下降至30%，同时电视观看人群的年龄结构呈现“老龄化”趋势。在这样一个内容与体验为王的时代，如何利用现有平台基础，通过技术的改进创新，为用户提供更多的应用、内容，重新获得用户的关注，成为广电运营商亟待解决的问题。

国际上对此的解决方案主要有两种：一种是欧洲主流的HBBTV（Hybrid Broadcast Broadband TV）标准。该标准将广电网与互联网标准混合在一起，规定终端提供基于网络下载和执行应用程序的能力，对终端要求较高。另一种是美国主导的全部IP化的方案。

我国国内广电运营现状比较复杂，各地区之间差距较大。国内用户终端机顶盒大部分由运营商投放，多种多样，并且普遍配置低、运算能力差。如采用HBBTV方案，需要各运营商对用户终端进行统一升级，这对运营商来说耗时太长、投入太大。如采用全IP化方案，就要丢掉现有的广电网络平台，这对广电网运营商来说也是不可取的。

目前一些公司提出了一种新的解决方案，即采用云技术，将终端的运算能力剥离到云端，终端只需要具备基本的网络接入、视频解码和交互处理能力。把应用部署到云端，应用程序的处理、显示图像的渲染等在云端完成，运行结果以音视频流的方式通过网络传送到终端，终端完成音视频的解码并将结果呈现。用户的操作指令通过网络传送到云端运行的应用，应用处理后将响应结果再次以音视频的形式下发。该解决方案对终端能力要求低，现存的多数终端设备都能满足要求，避免了对终端的升级维护。同时在云端部署应用，可以实现应用的快速部署、内容的可管可控、对应用软件的版权保护，以及实现对资源统一有效利用。

## 1.2 课题研究可行性分析

基于云平台为广电网用户提供服务，目前还没有一套成熟的完整的方案。当前国内外只有几家公司提供类似的服务，提供了一些案例参考，但由于商业原因，这些公司都没有公开技术细节。

目前越来越多的广电网运营商选择NGOD架构来提供视频点播服务。NGOD是一种开放的视频点播架构，对各个功能组件进行了合理细致的划分，全面详细的规定了各组件间的协议交互和通信接口，非常有利于进行未来的业务扩展。在NGOD架构上进行扩展，既可以充分利用现有的功能组件、视频传输通道，又可以兼容现有的广电网的业务。

本课题着重研究基于NGOD架构，通过运用虚拟化技术，合理充分利用服务器资源，将云应用服务在广电网上部署。

## 1.3论文研究目标及创新点

1. 研究目标

(1)对广电网络现有的NGOD架构进行扩展，从而可以在广电网上部署云应用。

(2)研究一种云端应用服务器群的调度管理方法，能够针对用户不同的请求，选择合适的应用服务器部署应用，并确保应用服务器间负载均衡。

(3)研究一种适用于广电网的云应用服务管理方案，使得PC、智能终端上运行的单机应用、互联网上的应用可以移植到广电网上进行部署，为用户提供服务。

2. 创新点：

NGOD架构主要是针对视频点播服务的，并没有考虑在广电网上部署云应用。广电网上云应用服务与视频点播服务的区别在于，云应用服务对时延要求更高、云应用服务器管理更复杂，会话的上下行通路以及客户端操作命令的传输解析都与视频点播服务不同。课题需要在不影响原架构功能的基础上，对架构进行扩展。

对服务器资源的调度管理需要综合考虑用户选择应用程序的类型、用户的位置、服务器的负载情况等因素。目前，有人已实现NGOD架构中ERM、ODRM模块，在选择推流服务器和边缘资源管理器时采用靠近用户的原则选择设备。但是，广电网中的云应用对传播时延要求很高，在选择应用服务器的位置时需要综合考虑上行和下行通路的时延。不同的应用类型对服务器资源的需求不同，有些应用可能对CPU要求高，对内存、显卡等要求低，有些应用可能对显卡要求较高，例如3D游戏，显卡必须具备较高的性能才能对游戏画面完成渲染。服务器间负载平衡，有利于保证服务质量。假如部分服务器负载远高于其它服务器，如果这部分服务器宕机，会造成大部分会话丢失，降低了系统的QoS。

本课题综合考虑了上述因素，选择了一种分配策略，可以较合理的选择应用服务器，分配服务器资源、提高服务质量。

## 1.4论文组织和结构

本文共分七章，第一章介绍了课题的背景和意义，课题可行性、研究目标和创新点以及论文组织结构；第二章介绍了NGOD架构，分析了VOD系统，通过比较现有的功能架构与广电网上部署云应用的业务需求，对NGOD架构进行扩展，并介绍了NGOD架构应用到的协议；第三章介绍了对扩展NGOD架构新增加的接口的定义，以及对接口通信信息的详细规定和对相应协议的扩展；第四章详细介绍了对NGOD架构扩展的涉及模块的设计与实现，包括对会话管理模块的扩展、对应用服务器管理器的设计实现、对应用服务器的设计实现，重点介绍了应用服务器管理器的总体结构和采用的负载均衡策略、应用服务器的应用虚拟化技术；第六章介绍了测试环境，给出了测试拓扑设计和系统测试流程和结果分析；第七章为全文总结和展望。

# 第二章 对NGOD架构进行扩展

## 2.1 NGOD架构

NGOD（Next Generation On Demand，即下一代交互电视架构），是由美国Comcast公司提出的一种新的交互式的开放的服务网络框架结构[1]。该架构合理细致的划分出了各个逻辑功能组件，全面详尽地规定各逻辑组件间的通信接口和交互协议，支持不同厂商之间产品的交互，有很强的可扩展性，能支持未来多种业务。目前，国内广电网运营商开始逐步地转向基于NGOD架构开发和部署广电网内视频点播业务。

NGOD架构如图2-1所示：

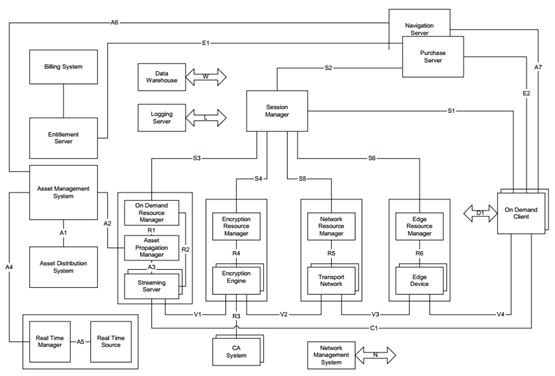


图2-1 NGOD架构

这里主要介绍以下与本课题直接相关的几个组件及接口：

On Demand Client：点播客户端，被定义为数字机顶盒或任何可以提供代理服务来同NGOD进行交互的网关服务器上的一个模块收集器。ODC与其它组件的关键信息和协议包括资产信息（接口A7）、授权信息（接口E2）、会话信号协议（接口S1）、推流控制协议（接口C1）、视频流（接口V4）、客户端自动检测接口（接口D1）。

Edge Device：边缘设备，主要功能是接收从IP传输网络通过UDP/IP传送过来的多MPEG SPTS，转成MPEG MPTS，并且产生QAM调制信号。

Edge Resource Manager：边缘资源管理器，负责分配和管理边缘设备上所需的资源，如QAM带宽（通过接口R6）。在从会话管理器（Session Manager）特定会话传来的资源请求上，边缘资源管理器需要确定边缘设备的使用，输入UDP端口和IP地址，输出频率和MPEG节目参数。边缘资源管理器的其它功能也包括带宽管理和服务质量。

Streaming Server：推流服务器，负责通过传输网络和边缘设备，将数字视频流向数字机顶盒。推流服务器也处理VCR类似的流控制，如暂停、快进、快退等。

On Demand Manager：点播资源管理器，负责分配和管理来自推流服务器的流资源。在会话建立来自客户端的请求之前，会话管理器从点播资源管理器请求资源（通过接口S3）。由点播资源管理器分配的资源包括选择的推流服务器（通过接口R1）和分配的流资源（通过接口R2）。

Session Manager：会话管理器（SM），负责为点播服务管理会话生命周期。会话管理器的一项重要功能是通过同适当的服务器和网络组建的资源管理的协商下，为会话获取必需的资源。

Asset Management System：资产管理系统（AMS），通过资产发布接口（A1），接收来自资产发布系统的资产数据，包括资产元数据和内容文件。

Asset Propagation Manager：媒资传播管理器（APM），通过接口A3，负责将来自各种内容源的资产传送到合适的推流服务器，这项重要的功能被叫作“传播服务”。

Data Warehouse：数据仓库，对来自架构中各种组件，包括会话管理器和资源管理器的点播统计数据进行收集、分析和记录。

Logging Server：日志服务器（LS），负责从架构中各种组件上收集事件日志数据。当出现故障需要进行诊断，或者某些特殊需要，要对事件进行跟踪时，使用LS来提供详细的点播会话和资源信息的日志。

NGOD架构中的接口都被定义成开放的、共有的，以此来促进将点播服务应用在多商家的环境中。NGOD架构对管理平面的接口分为以下几类：

资产管理接口：A1-A7，其主要负责管理或操纵资产元数据和内容。另外，这些接口也负责监控资产的状态。

会话管理接口：S1-S6，主要负责会话的建立和销毁，如同其它会话管理功能。这些接口要求非常实时，因此，诸如响应时间和吞吐量这样的性能问题将被放到接口设计中来考虑。其中，在SM和ODRM之间定义的会话和点播资源接口S3负责在SS上对请求会话的所需资源进行商议。在SM和ODRM之间定义的会话和边缘资源接口S6负责在边缘设置上对请求会话所需的资源进行商议。

资源管理接口：R1-R7。各种资源管理器使用这些接口来管理相应组件的资源。它们允许资源管理器获得相应组件的配置、拓扑、状态和可用资源。特别是，资源接口相互之间运行在同一层面，并且不与会话接口同步。其中，ODRM和APM之间的资产定位接口R1负责分配SS资源来对请求内容推流。APM维持一张内容资源表及其在SS上的位置，并将SS的位置返回给ODRM。ODRM和SS之间的推流服务器资源接口负责管理SS的资源。通过这个接口，ODRM将监控多个SS的配置、状态和可用资源，使用这些信息可以确定通过R1接口选定的SS来推流是有效的，并且有足够的带宽容量来对资产推流。

授权管理接口：E1-E2，负责执行授权验证、购买授权和通过ES（Entitlement Server，授权服务器）进行事务传递。

推流控制接口：C1，支持VCR像播放、暂停、快进、倒退这样的“欺骗模式”。如同会话管理，接口可以采用DSM-CC或者RTSP协议。特别的是，流控制信息由SS直接处理来确保较低的反应时间。

客户端自动监测接口：D1，负责允许ODC自动寻找自己在传输分布网络（HFC）中的位置并周期性地回报给数据转发器组件。对此，有很多种解决方案，例如，客户端可以自动寻找使用一套由机顶盒可见的边缘QAM指定的独特的MPEG传输流ID。

服务发现注册接口：D2-D7，数据平面的设备（例如边缘设备）利用这个接口向管理它们的模块（例如ERM）注册和更新自身的参数信息。资源管理模块利用这个接口获得资源的详细目录，同时也可以利用这个接口向SM注册和更新它们管理的资源的参数。

视频传输接口：V1-V4，推流服务器到边缘设备的接口，负责点播内容的传输。

网络管理接口：N，负责总体的网络管理功能，一般使用SNMP协议。

数据仓库汇报接口：W，负责对公共定义格式的数据的收集，这个接口可以用于各种NGOD组件，如SM、ERM和ODRM。

日志服务器的事件日志接口：L，负责公共定义格式的日志事件的收集，可以用于来自SM和各种资源管理器的会话日志和资源信号事件[2]。

## 2.2 VOD系统

VOD系统（全称Video on Demand，即视频点播系统），是一种交互式的视频播放系统，用户可以按照自身需要点播喜欢的节目[3]。该系统将多媒体数据和音视频素材存放在视频服务器中，然后通过家庭用的机顶盒将内容列表呈现给用户，用户可以随时随地通过接入网络进行多媒体内容的交互式查询，挑选自己喜爱的节目，对其进行点播。当电视用户向视频点播系统提出点播请求，如果经系统鉴权，该多媒体资源可以播放，系统就会将节目视频从视频服务器以传输流的形式通过网络传送到连接用户的边缘设备上，经过边缘设备的软硬件解压，视频资源会被推送到用户的家庭机顶盒上，实现视频的观看。同时，电视用户也可以利用与视频服务器的控制接口实现视频的播放控制功能[4]。

NGOD架构的核心是视频点播服务，不过可以将架构进行扩展来支持其它的点播服务，如交换广播视频或网络PVR。如下图所示为基于NGOD架构的VOD系统的简单结构图：

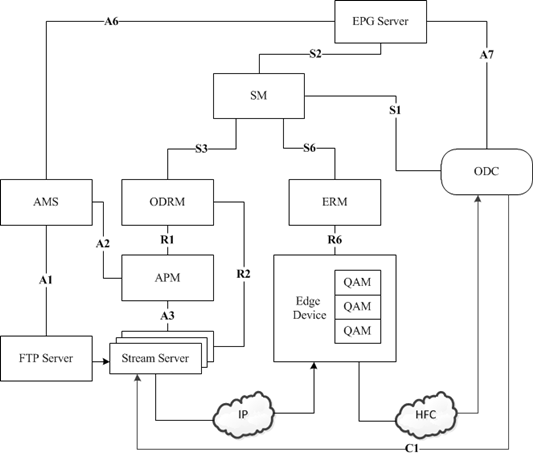


图2-2 基于NGOD架构的VOD系统

整个视频点播的过程是：点播客户端（一般指用户机顶盒），通过A7接口接收到VOD系统提供的内容资源列表，用户从中选择自己喜欢的内容，点播客户端通过S1接口发出点播请求。SM收到客户端的点播请求后会为客户端建立一个会话。SM通过S6接口向ERM边缘资源管理器请求QAM资源，通过S3接口向ODRM点播资源管理器请求推流服务器流资源。ERM通过解析SM请求中的QAM信息和带宽请求，根据自身数据库中的边缘QAM信息，依据一定的分配策略来分配QAM资源，并通过R6接口配置资源。ODRM在收到SM的请求后，通过R1接口向APM请求推流服务器流资源，APM会向ODRM上报自己所拥有的服务器资源信息，ODRM管理推流服务器信息并根据一定的负载均衡算法找到合适的推流服务器资源，通过R2接口配置推流服务器，推流服务器从缓存服务器获取用户请求的内容并进行推流[5]。APM与推流服务器之间定义了A3接口，目的是不同厂家的推流服务器可以通过相同的传播服务框架被引入到工作中。这个接口隐藏了推流服务器存储系统的内在实现。FTP Server代表视频服务器，通过各种单点/多点传送协议，将用户请求的内容直接传送到推流服务器上，推流服务器对其进行缓存。SM结合收到的资源，使得推流服务器和边缘设备有效配合完成点播。用户可以在播放过程中，通过C1接口向推流服务器发送指令，进行暂停、快进、快退等操作。

用户结束视频点播、点播客户端发出会话撤销命令时，SM通过和ERM和ODRM通信来撤销会话并回收资源。

## 2.3 扩展NGOD架构

### 2.3.1 云应用服务业务

在广电网上部署云应用服务，最好的选择是基于广电网现有的平台和传输网络，在不影响现有功能的基础上进行。

本课题提出的云应用服务的解决方案是：将终端的运算能力剥离到云端，终端只需要具备基本的网络接入、视频解码和交互处理能力，现在一般的家庭用机顶盒都可以满足这个要求。用户点播的应用在服务器端运行，应用的处理、显示图像的渲染等都在服务器上完成，运行结果以音视频流的方式通过网络传送到终端，终端完成音视频的解码并将结果呈现。用户的操作指令通过网络传送到服务器上运行的应用，应用处理后再次将响应结果以音视频的形式传输到终端。

云应用服务与VOD系统的区别：

1. VOD系统提供给用户的内容资源已经存储在视频服务器上，而云应用服务通过广电网络传输给客户端的是应用程序在服务器端运行、实时产生的音视频。
2. 云应用服务业务中应用程序需要运行在服务器上，而现有NGOD架构中并没有这样的服务器模块。
3. 在VOD系统中，推流服务器会对内容资源进行缓存，以方便用户对视频的操作；而云应用服务要求用户能接收到实时的运行结果，故云应用服务中推流服务器无需缓存。
4. 在VOD系统中，用户暂停、快进、快退等操作命令是通过点播客户端和推流服务器之间定义的C1接口来传输的，操作是在推流服务器上完成的。而云应用服务业务中，用户的操作命令必须传输到应用服务器，由应用服务器上正在运行的应用来完成操作。
5. 在VOD系统中，客户端与服务器端交互的下行通路是视频服务器→推流服务器→边缘设备→点播客户端，而云应用服务业务中，下行通路是应用服务器→推流服务器→边缘设备→点播客户端。

### 2.3.2 扩展NGOD架构

现有的NGOD架构并不支持在服务器端部署云应用服务，需要对其进行扩展。NGOD架构对各功能组件进行了细致的划分、对各接口进行了详细全面的规定，架构本身就是模块化的、可灵活扩展的。因此，依据NGOD架构现在的标准、协议，可以较为方便地对其进行扩展。如图2-3是对NGOD架构进行扩展后的部分结构图。

本次课题为NGOD架构新增加应用服务器管理器（ASM，Application Server Manager）和应用服务器（AS，Application Server）两个模块。ASM的功能包括资源管理功能和设备发现功能。AS模块代表应用服务器群，应用程序在这些服务器上运行。原有的接口定义无法满足新增的两个模块的功能需求，为此，在现有的接口的基础上，新增了三个接口。现有NGOD架构标准定义了S1-S6会话管理接口，在此基础上在SM和ASM之间定义了S7接口，负责在应用服务器上对会话请求所需的服务器资源进行商议。现有标准定义了R1-R7资源管理接口，在此基础上在ASM与AS之间定义了R8接口，ASM通过使用该接口，依据选定的负载均衡策略来管理应用服务器的资源。现有标准定义了D1-D7发现接口，在此基础上在ASM与AS之间新增了D8接口，AS通过该接口向ASM注册并提供可用资源的详细目录。

现有NGOD架构在点播客户端与推流服务器之间定义了流控制接口C1，用来向推流服务器传输用户的操作命令。本课题在点播客户端与应用服务器之间新定义了一个命令传输接口C2，用来向应用服务器上运行的应用程序传输操作命令。当用户请求的是云应用服务时，点播客户端不再与推流服务器建立连接，而是与服务器端为用户分配的应用服务器建立连接。

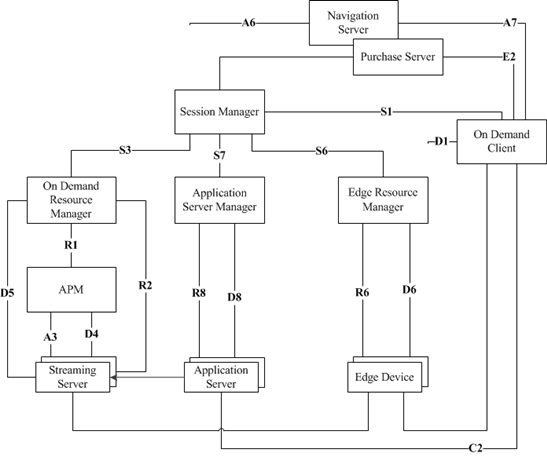


图2-3 扩展NGOD架构

### 2.3.3 RTSP协议

RTSP，Real-Time Streaming Protocol，实时流传输协议，是由RealNetworks和网景公司共同提出的如何有效地在IP网络上传输流媒体数据的应用层协议。RTSP协议建立并控制一个或者多个时间同步的连续媒体流，例如音频和视频。该协议自身并不会发送连续的媒体流，尽管连续的媒体流和控制流存在着交叉的可能。换句话说，RTSP协议扮演着多媒体服务器“网络远程控制”的角色[6]。

现在还没有RTSP连接的概念，一个服务器维护着由一个标识符标识的会话。一个RTSP会话并不会绑定到一个传输层连接，例如TCP连接。在一个RTSP会话期间，RTSP客户端可以打开和关闭多个对服务器的可靠的传输层连接来发出RTSP请求。该协议也可以选择无连接的传输层协议，例如UDP协议。

RTSP协议基于典型的客户端/服务器模型，与HTTP协议类似，它们都是基于文本的协议。二者不同之处在于RTSP协议引入了许多新的方法并且有一个不同的标识符；RTSP协议能够维护两个端点之间的连接状态，而HTTP协议则无法完成；同时RTSP协议是一种对称的协议，客户端既可以发送数据也可以接收数据，服务器端既可以接收数据也可以发送数据，即模糊了客户端与服务器之间的差别。而HTTP协议中，客户端只可以发送请求，服务器端只可响应请求数据[6]；RTSP使用ISO 10646（UTF-8）而非ISO 8859-1；RTSP中的请求URI总是包含完整的URI，但HTTP协议为了与过去的错误兼容，只在请求过程中传送绝对路径并将主机名置于另外的头字段。

RTSP协议的请求消息格式如下：

*请求行CRLF*

*消息头CRLF*

*CRLF*

*消息体*

响应消息的格式如下：

*状态行CRLF*

*消息头CRLF*

*CRLF*

*消息体*

请求和响应消息都通过一个CRLF来区分消息体与消息头。

基于文本的协议都有很好的扩展性，RTSP协议可以做出三种扩展：以新参数扩展、加入新方法、定义新版本协议。NGOD架构中的会话接口与资源接口都是基于RTSP协议的，NGOD对RTSP协议做出了扩展，增加了一些方法和头部来实现自己的功能。ERM模块、ODRM模块和SM模块都实现了这些扩展，本课题为了在NGOD架构上添加新的业务，也对RTSP协议做出了一些扩展。

### 2.3.4 VREP协议

VREP协议（Video Registration Protocol，即视频注册协议），是美国Comcast公司提出的一种资源注册管理协议。该协议是以RFC 3219（TRIP协议）为基础，扩展得来的[7]。TRIP协议（Telephony Routing Over IP，即IP上的电话路由选择协议）主要用于通告定位服务器间电话的可达性和通告到达目的地的路径属性[8]。TRIP协议独立于任何信令协议，因此TRIP协议可以为任何信令协议提供电话路由选择协议服务。VREP协议采用了TRIP协议的部分内容，同时继承了TRIP协议属性集的可扩展性，方便了NGOD架构增加新的业务功能。VREP协议将数据层设备与资源管理器都定义为VREP的发布者或者收听者，二者都可以初始建立一个VREP会话，一旦会话建立之后，它们之间就可以互相交换信息以确认连接的配置参数，同时互相协商通过这条连接要传递的信息的类型。但是在NGOD的第一版本中，只有VREP的发布者能与VREP的收听者建立一个VREP会话，因为VREP的收听者不能事先知道VREP发布者的信息。在2.0版本中VREP发布者会广播自己的信息，这样VREP收听者就能够获取VREP发布者的信息，因此它也可以主动建立一个会话。

NGOD架构中的服务发现注册接口D2-D7是基于VREP协议的。本课题为NGOD架构添加新业务时，为了与现有架构“无缝”兼容，新增加的D8接口实现的是VREP协议。

## 2.4 本章小结

本章对广电网上现有的NGOD架构进行了较为详细的介绍，介绍了VOD系统、云应用服务业务，并对这两者做了比较，在此基础上提出了对NGOD架构进行扩展，为后文的研究打下了基础。

# 第三章 NGOD架构新增接口的定义

上文提到对NGOD架构进行扩展，使之支持在广电网上部署云应用服务。为架构新增加了S7、R8、D8三个接口，为了不影响现有的功能，需要在现有标准的基础上定义实现这三个接口。

## 3.1 定义S7接口

S7接口位于ASM与SM之间，负责在应用服务器上对会话请求所需的服务器资源进行商议。S7接口属于会话管理接口，基于RTSP协议实现的，术语、消息语法、方法定义和状态码依据于RFC 2326和NGOD官方对RTSP协议做的扩展。消息类型包括SETUP、TEARDOWN和ANNOUNCE等。

下图是S7接口的消息交互流程：

|  |
| --- |
|  |
| C:\Users\109421\AppData\Local\Temp\ksohtml\wps1A03.tmp.png |  |

图3-1 S7接口消息交互流程

ODC（On Demand Client，点播客户端）通过S1接口向SM发出应用服务请求后，SM首先通过S6接口向ERM（Edge Resource Manager，边缘资源管理器）发送SETUP会话建立请求，选择ED（Edge Device，边缘设备）。ERM向SM返回SETUP RESPONSE消息，包括ED的IP、端口等信息。然后SM通过S3接口向ODRM（On Demand Resource Manager，点播资源管理器）发送包括ED的IP、端口等信息的SETUP会话建立请求，选择SS（Streaming Server，推流服务器）[9]。之后，SM作为RTSP客户端通过S7接口向ASM发出服务器资源请求SETUP。在SETUP报文中，除了点播客户端的信息、用户请求的应用信息外，还包括ODRM返回的SS的信息，以便于建立会话的下行通路。ASM在选定AS（Application Server，应用服务器）部署应用之后，返回SETUP RESPONSE消息，包括选定的AS的IP和部署的应用的进程号，SM将其转发给点播客户端，用来建立点播客户端到AS的上行通路。

用户结束使用云应用服务后，点播客户端向SM发送撤销会话的请求，或者SM与点播客户端之间的心跳检测超时的情况下，SM向ASM发送TEARDOWN消息，请求释放该会话占用的应用服务器资源。ASM在释放服务器资源后，向SM发送TEARDOWN RESPONSE消息。

在一些异常情况下，例如ASM检测到AS的异常，ASM会主动向已经建立会话的SM发送一些非请求消息（消息中包含发生的事件以及发生的时间等信息），这些消息一般通过RTSP ANNOUNCE Request和Response消息来实现。

RTSP消息包括URL和Header两个部分：

URL的语法是rtsp://<path>:<port>，其中rtsp表明传输层要建立TCP连接（如果是rtspu，则表明传输层是UDP协议），path代表应用服务器管理模块的IP地址或者域名，port代表应用服务器管理模块的RTSP会话建立端口，缺省是554[10]。

Header根据消息类型的不同而有所变化。

### 3.1.1 S7 SETUP Request消息

消息定义格式如下：

SETUP rtsp://asm\_ip:asm\_port RTSP/1.0

CSeq:<cseq>

Require:<require>

OnDemandSessionId:<on demand session id>

SessionGroup:<session group>

StreamControlProto:<stream control protocol>

ApplicationId:<application id>

ApplicationType:<application type>

Transport:

MP2T/DVBC/UDP;unicast;client=<client>;destination=<destination>;client\_port=<client\_port>;bandwidth=<bandwidth>

其中，CSeq字段指的是RTSP请求-响应的序列号。此字段必须存在于所有的请求和响应。对于每个包含给定序列号的RTSP请求，将有一个有相同序列号的响应。任何重传的请求必须包含与原始请求相同的序列号（即序列号不会因为重传递增）。

Require字段是NGOD架构为RTSP协议扩展的字段，它定义了一个机制，客户端在一个请求中标明特定的扩展必须满足该请求。特定的NGOD接口通过使用这个字段来定义严密的用法。例如：Requie:com.comcast.ngod.s1。

OnDemandSessionId字段是NGOD架构为RTSP协议扩展的一个字段，它标明了SM为一个会话指定的唯一标识。字段值是一个128位的IETF定义的UUID字符串。

SessionGroup是NGOD架构为RTSP协议扩展的一个字段，它定义了一个令牌，用来标识一组会话。SessionGroup头字段用于两个NGOD组件重新同步会话状态的过程。RTSP客户端生成一个SessionGroup令牌将其建立的会话组合在一起，并通过SETUP请求消息发往RTSP服务器，RTSP服务器为每一个会话记录该令牌。

StreamControlProto是NGOD架构为RTSP协议扩展的一个字段，它指明了接口C1和C2实现的协议，取值有rtsp（基于TCP的RTSP）、lscp（基于TCP的LSCP协议）、lscpu（基于UPD的LSCP协议）。

ApplicationId是本课题扩展的一个字段，它的值是用户请求应用的标识。

ApplicationType是本课题扩展的一个字段，它的值是用户请求应用的类型。

Transport头字段指明了传输媒体流的传输层协议，以及配置参数，例如destination目的地址、compression压缩格式等。client指明客户端的唯一标识；destination指明媒体流发送的目的IP，在S7接口中就是推流服务器的IP；client\_port指明媒体流发送的目的端口，在S7接口中就是推流服务器的输入端口；bandwidth指明推流服务器分配的带宽。

### 3.1.2 S7 SETUP Response消息

该消息是对S7 SETUP消息的相应。消息定义格式如下：

RTSP-version Status-Code Reason-Phrase

CSeq:<cseq>

Session:<session>

StreamControlProto:<protocol>

OnDemandSessionId:<on demand session id>

Path:<path>

Transport:

MP2T/DVBC/UDP;unicast;client=<client>;destination=<destination>;client\_port=<client port>;source=<source>;server\_port= <server port>; bandwidth=<bandwidth>

第一行是状态信息，包括RTSP版本号、状态码（Status Code）、状态码对应的文本信息（Reason Phrase）。

状态码是一个三位的整数，最高位代表了响应的状态：

1xx：临时响应，请求已被收到，继续处理；

2xx：请求已被成功接收、理解并接受；

3xx：重定向，发出请求的客户端需要进一步的操作；

4xx：客户端出错，请求消息可能包括错误的语义或者不满足协议要求；

5xx：服务器端出错，服务器在处理请求的过程中出现错误或者发生异常[10]。

6xx，7xx：NGOD架构对RTSP协议的扩展

具体的Status-Code和Reason-Phrase的定义如表3-1所示：

|  |  |
| --- | --- |
| Status Code | Reason Phrase |
| 100 | Continue |
| 200 | OK |
| 201 | Created |
| 250 | Low on Storage Space |
| 300 | Multiple Choices |
| 301 | Moved Permanently |
| 302 | Moved Temporarily |
| 303 | See Other |
| 305 | Use Proxy |
| 400 | Bad Request |
| 401 | Unauthorized |
| 402 | Payment Required |
| 403 | Forbidden |
| 404 | Not Found |
| 405 | Method Not Allowed |
| 406 | Not Acceptable |
| 407 | Proxy Authentication Required |

续上表

|  |  |
| --- | --- |
| Status Code | Reason Phrase |
| 408 | Request Time-out |
| 410 | Gone |
| 411 | Length Required |
| 412 | Precondition Failed |
| 413 | Request Entity Too Large |
| 451 | Parameter Not Understood |
| 452 | Conference Not Found |
| 453 | Not Enough Bandwidth |
| 454 | Session Not Found |
| 455 | Method Not Valid in This State |
| 456 | Header Field Not Valid for Resource |
| 457 | Invalid Range |
| 458 | Parameter Is Read-Only |
| 459 | Aggregate operation not allowed |
| 460 | Only aggregate operation allowed |
| 461 | Unsupported transport |
| 462 | Destination unreachable |
| 500 | Internal Server Error |
| 501 | Not Implemented |
| 502 | Bad Gateway |
| 503 | Service Unavailable |
| 504 | Gateway Time-out |
| 505 | RTSP Version not supported |
| 551 | Option not supported |
| 650 | SM Setup Failed - No Response |
| 651 | SM Setup Failed - Unknown QAM Group |
| 652 | SM Setup Failed - Invalid Request |
| 653 | SM Setup Failed - Internal Error |
| 770 | Server Setup Failed - No Response |
| 777 | Server Setup Failed - Invalid Request |
| 778 | Server Setup Failed - Internal Error |

表 3-1 status code和reason

Session字段标识该会话，有RTSP服务器端生成。

Path字段指用户点播的应用在应用服务器上部署的路径，该字段最终会通过S1接口传送到点播客户端，点播客户端与应用服务器建立会话上行通路后，通过该路径与点播的应用交互。

Transport字段中的source参数指流媒体的来源，在S7接口中指的是AS的IP。Server\_port指流媒体来源端口，在S7接口中指AS的输出端口。

### 3.1.3 S7 TEARDOWN消息

TEARDOWN请求会终止该会话并释放与该会话相关联的资源。S7 TEARDOWN请求消息格式如下：

TEARDOWN rtsp://asm\_ip:asm\_port RTSP/1.0

CSeq:<cseq>

Require:<require>

Reason:<reason-code> <reason>

Session:<session>

OnDemandSessionId:<on demand session id>

Reason字段是NGOD架构为RTSP协议扩展的字段[10]，它的定义如表3-2所示：

|  |  |
| --- | --- |
| Reason Code | Reason Description |
| 200 | User stop |
| 201 | End of stream |
| 202 | Beginning of stream |
| 203 | Pause timeout |
| 400 | Fail to tune |
| 401 | Loss of tune |
| 403 | RTSP failure |
| 404 | Channel failure |
| 405 | No RTSP server |
| 406 | Trick-play failed |
| 407 | Internal ODA issue |
| 408 | Unknown |
| 409 | Network Resource Failure |
| 420 | Settop Heartbeat Timeout |

续上表

|  |  |
| --- | --- |
| Reason Code | Reason Description |
| 421 | Settop Inactivity Timeout |
| 422 | Content Unavailable |
| 423 | Streaming Failure |
| 424 | QAM Failure |
| 425 | Volume Failure |
| 426 | Stream Control Error |
| 427 | Stream Control Timeout |
| 428 | Session List Mismatch |
| 550 | Session Timeout |

表3-2 TEARDOWN Reason Code

### 3.1.4 S7 TEARDOWN Response消息

消息格式如下：

RTSP-version Status-Code Reason-Phrase

CSeq:<cseq>

Session:<session>

OnDemandSessionId:<on demand session id>

### 3.1.5 S7 ANNOUNCE消息

在某些情况下，ASM会主动发送一些信息给SM，这些信息通过RTSP ANNOUNCE消息来发送。ANNOUCE Request消息的格式如下：

ANNOUNCE rtsp://<sm-path>:<sm-port> RTSP/1.0

CSeq:<cseq>

Require:<Require>

Session:<session>

Notice:<notice-code> “<text description>” event\_date=<datetime> npt=<npt-value>

OnDemandSessionId:<on demand session id>

sm-path是SM的IP或域名，sm-port是SM的端口号。

Notice字段是NGOD架构对RTSP协议的扩展字段。其中npt（normal play time）参数是用来记录最后的正常播放时间，这样下一次可以在同一个地方开始播放，如果无法获得这个参数值，可以设置为空[10]。

表3-3是notice-code和相应描述信息的详细列表：

|  |  |
| --- | --- |
| Code | Message |
| 2101 | End-of-Stream Reached, sent when the end-of-stream has been reached |
| 2104 | Start-of-Stream Reached, sent when the start-of-stream has been reached |
| 4400 | Error Reading Content Data, sent when there is an issue reading the data to be streamed |
| 5200 | Server Resource Unavailable, sent when there is an issue with availability of server resource |
| 5401 | Downstream Failure, sent when there is a downstream failure, e.g., QAM failure |
| 5402 | Client Session Terminated, sent as a notification that for some reason the RTSP client session was terminated by the server |
| 5502 | Internal Server Error, sent as a notification that there was an issue with the server |
| 5601 | In Band Stream Marker Mismatch |
| 5602 | Bandwidth Exceeded Limit |
| 5700 | Session In Progress, sent by RTSP server when a session timer expires |
| 6000 | Encryption Engine Failure/Fault |
| 6001 | Stream Bandwidth Exceeds that Available |
| 6004 | Downstream Destination Unreachable |
| 6005 | Unable to encrypt one or more components |
| 6006 | ECMG session failure |

表3-3 S7 ANNOUNCE消息notice code

### 3.1.6 S7 ANNOUNCE Response消息

消息格式如下：

RTSP-version Status-Code Reason-Phrase

CSeq:<cseq>

Session:<session>

OnDemandSessionId:<on demand session id>

## 3.2 定义R8接口

ASM与AS之间定义了R8接口，ASM通过使用该接口，依据选定的负载均衡策略来管理应用服务器的资源。R8接口也是基于RTSP协议的，包括SETUP、TEARDOWN、ANNOUNCE、GET\_PARAMETER、PING等信令。在ASM通过R8接口向AS建立会话连接之前，ASM必须已通过D8接口获得AS的信息。

|  |
| --- |
|  |
|  | C:\Users\109421\AppData\Local\Temp\ksohtml\wps1A52.tmp.png |

图3-2 资源分配流程

点播客户端向SM发送SETUP会话建立请求，其中包括点播应用的ID和点播应用的类型。SM向ERM、ODRM发出SETUP请求并受到回复后，会向ASM发出SETUP会话建立请求，ASM结合点播客户端的应用请求与各应用服务器的状态，依据负载均衡策略，选定应用服务器来部署应用，并通过R8接口向选定的AS发送SETUP报文，包括客户端请求的应用的信息。AS加载部署应用后，向ASM发送SETUP RESPONSE报文，包括IP、所部署应用的路径。

ASM收到SM发送的TEARDOWN报文后，进行一些内部处理，例如更新AS信息表，然后向AS发送TEARDOWN报文告知结束会话。AS结束会话后返回TEARDOWN RESPONSE报文。

在会话过程中发生异常时，AS可以通过发送ANNOUNCE报文向ASM反映异常。ASM通过周期性地向AS发送PING报文来维护会话的生命周期。在NGOD RTSP Usage Specification中，默认的会话超时时间被设为3个小时。

### 3.2.1 R8 SETUP Request消息

消息格式如下：

SETUP rtsp://<as-ip>:<as-port> RTSP/1.0

CSeq:<cseq>

Require:<require>

OnDemandSessionId:<on demand session id>

SessionGroup:<session group>

StreamControlProto:<stream control protocol>

ApplicationId:<application id>

ApplicationType:<application type>

Transport:

MP2T/DVBC/UDP;unicast;client=<client>;destination=<destination>;client\_port=<client\_port>;bandwidth=<bandwidth>

### 3.2.2 R8 SETUP Response消息

消息格式如下：

RTSP-version Status-Code Reason-Phrase

CSeq:<cseq>

Session:<session>

StreamControlProto:<protocol>

OnDemandSessionId:<on demand session id>

Path:<path>

Transport:

MP2T/DVBC/UDP;unicast;client=<client>;destination=<destination>;client\_port=<client port>;source=<source>;server\_port=<server port>;bandwidth=<bandwidth>

Status-Code的取值详见表3-1。

### 3.2.3 R8 TEARDOWN Request消息

ASM在收到SM发出的结束会话TEARDOWN请求消息后，会向AS发送TEARDOWN请求消息，在收到AS的应答后，再向SM发送TEARDOWN Response。

R8接口的TEARDOWN Request消息格式如下：

TEARDOWN rtsp://<as-ip>:<as-port> RTSP/1.0

CSeq:<cseq>

Require:<require>

Reason:<reason>

Session:<session>

OnDemandSessionId:<on demand session id>

Reason字段的取值详见表3-2。

### 3.2.4 R8 TEARDOWN Response消息

应用服务器AS在收到ASM的TEARDOWN请求消息后，会进行一些释放资源的处理，比如终止用户请求的应用、释放内存、关闭上行通路和下行通路的端口等，然后向ASM返回TEARDOWN Response消息。R8接口的TEARDOWN Response消息格式如下：

RTSP-version Status-Code Reason-Phrase

CSeq:<cseq>

Session:<session>

OnDemandSessionId:<on demand session id>

### 3.2.4 R8 ANNOUNCE Request消息

AS通过R8接口向ASM发送ANNOUNCE Request消息报告错误或异常事件，消息格式如下：

ANNOUNCE rtsp://<sm-path>:<sm-port> RTSP/1.0

CSeq:<cseq>

Require:<Require>

Session:<session>

Notice:<notice-code> “<text description>” event\_date=<datetime> npt=<npt-value>

OnDemandSessionId:<on demand session id>

notice-code和test description的取值详见表3-3。

### 3.2.5 R8 ANNOUNCE Response消息

ASM收到AS发送的ANNOUNCE Request消息后，发出应答消息，消息格式如下：

RTSP-version Status-Code Reason-Phrase

CSeq:<cseq>

Session:<session>

OnDemandSessionId:<on demand session id>

## 3.3 定义D8接口

数据平面设备通过服务发现注册接口向资源管理模块注册，NGOD架构中的资源管理模块负责管理一组数据平面设备，例如ERM负责分配边缘设备资源，ODRM负责分配推流服务器资源，本课题提出的ASM负责分配应用服务器资源。一个资源管理模块为了管理数据平面组件的资源，必须获得该数据平面组件的资源列表，并且维护一个资源数据库[11]。

服务发现注册接口还有一个重要需求就是支持一个节点及时检测相邻节点的故障。例如，一个ERM通过服务注册发现接口及时检测到一个边缘设备的故障，这个ERM就可以利用该信息将新的会话请求重定向到一个没有发生故障的边缘设备。

ASM和AS之间定义了服务发现与注册接口D8。AS通过该接口向ASM注册并提供可用资源的详细目录。D2-D6接口实现了VREP协议，为了便于与原系统的兼容，D8接口采用VREP协议。

VREP协议包括四种消息类型：OPEN、UPDATE、NOTIFICATION、KEEPALIVE。OPEN消息主要用于VREP传输会话的建立；UPDATE消息主要用于资源设备向其资源管理器上报自身配置参数；NOTIFICATION消息主要用于错误的报告；KEEPALIVE消息用来恢复和维持心跳[12]。

图3-3所示为D8接口消息交互流程：

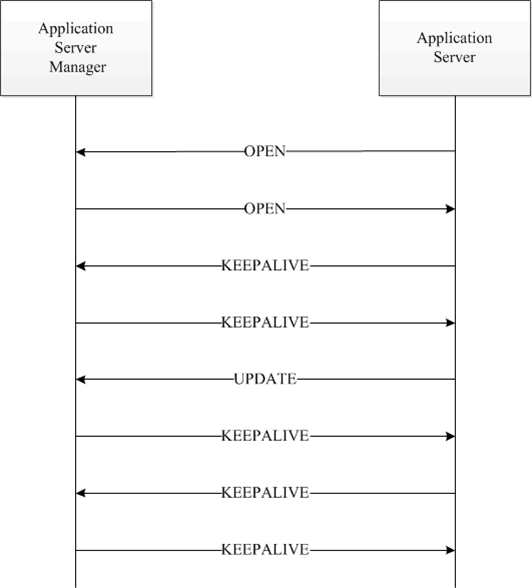


图3-3 D8接口VREP协议消息交互

AS接入系统后主动向相应的ASM发起连接请求，传输层连接建立后，两者首先通过OPEN消息协商参数信息，包括VREP协议版本、超时时间、VREP会话标识、支持的路由类型以及组件的收发能力等信息。OPEN消息被接受之后，会向对方发送KEEPALIVE消息确认。

在通过OPEN消息对组件的收发能力协商后，AS只具备发送UPDATE消息的能力，ASM的D8接口只具备接受UPDATE消息的能力。AS通过向ASM发送UPDATE消息报告自己的服务器资源详细信息，包括CPU信息和内存信息。ASM发送KEEPALIVE消息对它进行确认。

当AS和ASM在收发消息的过程中发现消息有误时，任何一方都会利用NOTIFACATION消息报告错误并断开连接。

每一个VREP消息都有一个固定大小的报文头部，根据消息类型的不同，后面可以附加消息内容或者为空，每个消息的最小长度是三个字节。VREP消息报文头部的格式如图3-4：

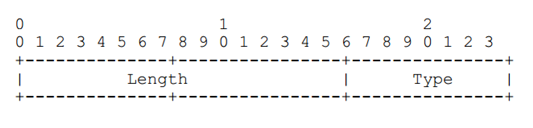


图 3-4 VREP Header

Length字段，占两个字节，指定消息总共占多少字节，包括头部，取值最小是3。

Type字段，占一个字节，指定消息类型，支持的取值包括：1、OPEN；2、UPDATE；3、NOTIFICATION；4、KEEPALIVE。

### 3.3.1 OPEN消息类型

在建立传输层连接后，VREP会话双方发出的第一个消息都是OPEN消息，如果接受了对方的OPEN消息，就会发回一个KEEPALIVE消息来确认。一旦OPEN消息被确认了，VREP会话双方就可以发送UPDATE、KEEPALIVE和NOTIFICATION消息了。

OPEN消息的最小长度是17字节。除了固定长度的VREP头部外，OPEN消息包含的字段如图3-5所示。

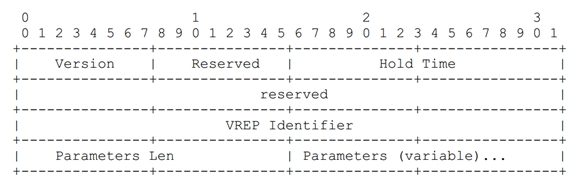


图3-5 VREP OPEN Header

Version字段占一个字节，指定协议版本号，当前VREP协议版本是1。

Hold Time字段占2个字节，其值为无符号整数。VREP会话双方会比较自身配置的Hold Time值和收到的OPEN消息的Hold Time值，选择较小的一个作为计时器值。该值指定了连续的KEEPALIVE和（或者）UPDATE消息之间间隔的最大秒数。Hold Time字段必须设置为0，或者最小值为3。

Reserved字段必须设置为0。

VREP Identifier字段，占4个字节，其值为无符号整数，指定了VREP发送方的标识。VREP Identifier在发送者或接收者所在的区域必须是唯一的，例如IP地址。

Parameters Len字段，值为2个字节的无符号整数，指定了Parameters字段的总长度，单位是字节。VREP OPEN消息有两个强制性的参数，一个是component name，组件名称，另一个是streaming zone，所在流区，其余参数都是可选的。

Parameters字段包含一个参数列表，每一个参数的形式如图3-6所示。

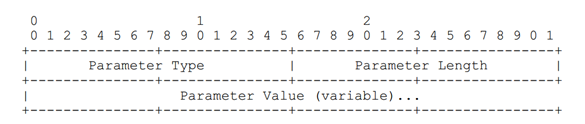


图3-6 VREP OPEN消息参数格式

Parameter Type字段占2个字节，取值包括：1、Capability Information；2、Streaming Zone；3、Component Name；4、Vendor Specific String。

Parameter Length字段占2个字节，指定Parameter Value字段的长度，单位是字节。

Parameter Value字段长度不固定，根据参数类型的不同而不同。

Capability Information，能力信息，VREP发送方通过该参数向对方表示自身的能力范围。该参数类型的格式如下：

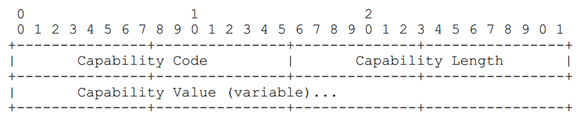


图3-7 Capability Information参数格式

Capability Length占2个字节，指定Capability Value的长度，单位是字节。Capability Code的定义如下：

|  |  |
| --- | --- |
| Code | Capability |
| 1 | Route Types Supported |
| 2 | Send Receive Capability |

表3-4 Capability Code

本课题没有用到Route Types Supported类型，在这里就不再描述。

类型2 Send Receive Capability取值有三种：1、Send Receive mode，结点既可以发送，也可以接受VREP消息；2、Send Only mode，一个处于该模式下的VREP结点发送UPDATE消息给对端，但对端不能向该结点发送UPDATE消息，如果该结点收到了UPDATE消息，必须丢掉；3、Receive Only mode，处于该模式下的VREP结点作为接收方，接收并处理对端发来的UPDATE消息，但不能向对端发送UPDATE消息。如果两个结点都处于Send Only mode或者Receive Only mode，这两个结点间不能建立连接。如果已经建立了连接的两个结点发现对端和自己都是处于Send Only mode或者Receive Only mode，这时必须发送NOTIFICATION消息结束会话，错误码是“Capability Mismatch”。

Streaming Zone，VREP结点所在的区域，是一个强制性的参数，其值的格式是<region>.<local name>。

Component Name，组件名称，也是一个强制性参数，其值的格式是<region>.<local name>。

Vendor Specific String，供应商标识，这是一个可选参数。

### 3.3.2 UPDATE消息类型

UPDATE消息主要用于资源设备向其资源管理器上报自身配置参数。根据各个AS发送的UPDATE消息，ASM维护一个应用服务器负载情况表，根据该表选择应用服务器部署用户请求的应用。

VREP UPDATE消息包括VREP头部和一系列属性信息，UPDATE消息的最小长度是3字节，即不包括任何属性信息。图3-8是不包括头部的UPDATE消息的格式。

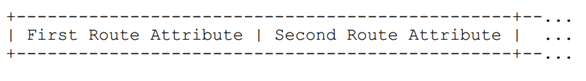


图3-8 VREP UPDATE消息格式

VREP UPDATE消息的属性的格式如图3-9所示。

|  |
| --- |
|  |
|  | C:\Users\109421\AppData\Local\Temp\ksohtml\wps1AB7.tmp.png |

图3-9 VREP UPDATE消息属性格式

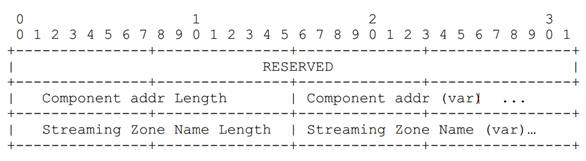
Attribute Flags字段用于指明一个参数是熟知参数还是未知参数。很明显，随着业务的增加，VREP协议肯定会相应的扩展，这时候提供一种机制来支持未知的参数类型就是必须的了。该字段就起这样的作用。该字段占一个字节，最高位（即第0位）为0，表明该参数是熟知的，最高位为1，表明该参数是未知的。该字段其它位没有用处，应该设置为0。

Attribute Type Code字段定义了该参数的类型，占一个字节，出现在UPDATE报文中的参数必须按照参数类型码的大小顺序排列，在同一个UPDATE报文中，一个参数不允许出现超过一次。

Attribute Length字段占两个字节，指明Attribute Value参数值的长度，单位是字节。

UPDATE报文中的熟知参数有很多，但没有一个是强制性的。这里只描述本课题用到的参数以及本课题定义的未知参数。

1、NextHopServer，下一跳服务器地址。Attribute Type Code是3。参数格式如下：

图3-10 NextHopServer格式

|  |
| --- |
|  |
|  |  |

Reserved字段设置为0。

Component Address Length，占两个字节，指定Component Address字段的长度，单位是字节。

Component Address，组件地址，长度可变，指定下一跳服务器的IP地址或者主机名。该字段的取值是一个ASCII码字符串，定义如下：

Server = host[“:”port]

host =<IP地址或主机域名>

port = 端口号，如果为空，则为默认端口

Streaming Zone Name Length，占两个字节，指定Streaming Zone Name字段的长度，单位是字节。

Streaming Zone Name，组件所在区域的域名。

Service Status，服务状态，Attribute Type Code是241。该参数指明了数据平面设备的使用状态，资源管理组件使用该参数来了解管理资源设备的状态。参数格式如图3-11所示。

图3-11 Service Status格式

该参数值是一个二进制编码的值，指明了一个数据平面设备的状态。参数取值如下：

（1）Operational可用的：设备是可用的，工作正常，资源管理组件可以向该设备分配任务。

（2）Shutting Down关闭：设备被正常关闭。资源管理组件不应该分配该设备的资源。

（3）Standby备用的：多余的设备，当一个设备出现故障时，可以用备用设备取代。

1. MemoryLoad，应用服务器内存占用百分比。该参数是本课题提出的，属于未知参数，Attribute Flags最高位设置为1。Attribute Type Code是247。参数格式如图3-12所示。该参数值占四个字节，以二进制形式存在的浮点数。应用服务器管理器通过该参数获得各个应用服务器的内存使用情况。

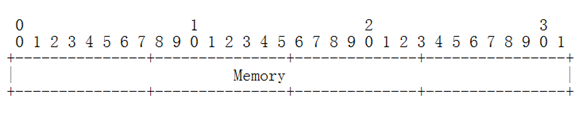


图3-12 Memory参数格式

1. CPULoad，应用服务器CPU负载情况。参数值以二进制的形式表示，占四个字节。应用服务器管理模块根据该参数获得应用服务器的负载情况。该参数是本课题提出的，属于未知参数，Attribute Flags最高位设置为1，Attribute Type Code是247。参数格式与MemoryLoad相同。应用服务器管理器通过该参数获得各应用服务器的CPU使用情况。

### 3.3.3 KEEPALIVE消息类型

VREP协议并不使用任何基于传输的心跳机制来检测对端的可达性。VREP会话双方通过足够频繁的交换KEEPALIVE消息来保证Hold Time不超时。KEEPALIVE消息之间最大的合适的时间间隔是Hold Time的三分之一，同时KEEPALIVE消息绝不能每隔3秒发送一次。如果经过UPDATE消息协商后的Hold Time值是0，那么会话双方不能发送KEEPALIVE消息。KEEPALIVE消息不包括任何消息内容，只有3个字节的VREP消息报文头部。

### 3.3.4 NOTIFICATION消息类型

当VREP会话的一方检测到错误或者异常发生时，会向对方发送NOTIFICATION消息，并且立刻关闭VREP传输连接。

除了固定大小的VREP报文头部，NOTIFICATION消息还包括如图3-13所示的一些字段。

|  |
| --- |
|  |
|  | C:\Users\109421\AppData\Local\Temp\ksohtml\wps1ADA.tmp.png |

图3-13 NOTIFICATION消息格式

Error Code（错误码）字段，占一个字节，无符号整数，指定NOTIFICATION消息的类型。Error Code的定义如表3-5所示。

|  |  |
| --- | --- |
| **Error Code** | **错误类型** |
| 1 | 消息头部错误 |
| 2 | OPEN消息错误 |
| 3 | UPDATE消息错误 |
| 4 | Hold Time超时 |
| 5 | 有限状态机错误 |
| 6 | 停止 |

表3-5 Error Code定义

Error Subcode（子错误码）字段对NOTIFICATION消息报告的错误提供了更详细的说明。每一个错误码都有一个或几个字错误码与之相关联。如果没有合适的子错误码，就将该字段设置为0。表3-6是Error Code 1 消息头部错误对应的子错误码的定义，表3-7是Error Code 2 OPEN消息错误对应的子错误码的定义，表3-8是Error Code 3 UPDATE消息错误对应的子错误的定义。

|  |  |
| --- | --- |
| **消息头部错误子错误码** | **错误类型** |
| 1 | 消息长度错误 |
| 2 | 消息类型错误 |

表3-6 消息头部错误子错误码定义

|  |  |
| --- | --- |
| **OPEN消息错误子错误码** | **错误类型** |
| 1 | 不支持的版本号 |
| 2 | 对端地址或域名错误 |
| 3 | VREP标识错误 |
| 4 | 不支持的可选参数 |
| 5 | 错误的Hold Time值 |
| 6 | 不支持的Capability参数 |
| 7 | Capability不能匹配 |

表3-7 OPEN消息错误子错误码定义

|  |  |
| --- | --- |
| **UPDATE消息错误子错误码** | **错误类型** |
| 1 | 参数列表顺序错误 |
| 2 | 未被识别的熟知参数 |
| 3 | 丢失熟知强制性参数 |
| 4 | Attribute Flags错误 |
| 5 | Attribute Length错误 |
| 6 | 无效的参数 |

表3-8 UPDATE消息错误子错误码

Data字段长度可变，该字段用来判断NOTIFICATION报告的错误的原因。Data字段的值根据错误码和错误子码而不同。可以通过下面这个公式计算出Data字段的长度：

Data长度 = 消息长度 - 5

可以看出NOTIFICATION消息的最小长度是5。

## 3.4 本章小结

本章详细介绍了对NGOD架构扩展新增的S7、R8、D8接口，包括其消息交互、消息的格式以及对其实现协议的扩展，为后文对NGOD架构扩展模块的设计与实现打下了基础。

# 第四章 相关模块的设计与实现

本课题对NGOD架构的扩展，主要是通过增加ASM和AS模块以及相应的接口来实现的。同时，有与增加了新的业务，以及新的业务上下行通路与原有的视频点播服务不同，所以需要对SM模块和点播客户端进行改进。改进点播客户端不属于本论文的内容范围。接下来的内容主要是介绍改进SM模块，设计实现ASM和AS模块。

## 4.1 改进SM模块

### 4.1.1 SM模块的结构设计

SM（Session Manager，会话管理模块）负责点播服务会话生命周期的管理。每一次客户端请求会话的建立，SM都需要负责对来自客户端请求数据的处理，并根据请求消息做出相应的响应，例如通过模块的内部接口完成和其它功能组件（例如ERM、ODRM、ASM等）的通信过程以获得相应的服务资源和网络资源。

SM具体应该实现的功能如下：  
 1、与点播客户端交互，完成会话建立、会话状态查询和会话释放。

2、与相应购买服务器通信，认证用户请求的会话。

3、资源分配能力，SM通过和各类资源管理器组件通信，完成对各类资源的申请和使用。

4、动态添加、删除或者修改与会话相关的资源，以支持多种点播服务集成。

5、保证整个系统会话通信质量。

6、管理整个系统中所有会话的生命周期。

基于以上的功能需求，课题设计的SM的架构如图4-1所示。SM的具体实现主要分为以下几个处理模块的实现：

1、会话管理模块

具体负责会话的建立、会话状态管理、会话释放。每一次会话的发起都是由客户端开始，根据客户端的MAC地址和其对应调谐器的ID可以生成一个对应的会话ID，这就是OnDemandSessionId，SM模块每次都需要记录下个会话ID，同时记录此次会话过程中出现的各类事件以及各个接口交互的成功与否，针对这些情况，SM模块会采取相应的处理措施（比如重新建立会话、终止会话、释放会话、发送ANNOUNCE广播等），这样可以保证每一次会话成功结束，或者是在会话为成功结束的情况下做出相应的操作。

2、资源管理模块

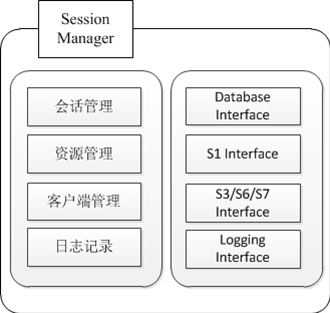
SM的主要功能是对各类资源的管理，包括推流服务器资源、边缘设备资源、应用服务器资源等[8]。该模块实现了S3、S6、S7接口的通信过程，通过S3接口与ODRM的交互过程请求推流服务器资源，通过S6接口同ERM的交互过程请求边缘设备的QAM信道资源。SM接收到客户端的应用服务请求后，首先和ERM通信，请求会话所需的边缘设备QAM通道资源，得到应答后和ODRM通信，请求会话所需的推流服务器资源，之后再与ASM通信，请求应用服务器资源、部署应用服务。最后将应用服务器的信息传回给客户端，由客户端建立与应用服务器的上行通路，传输操作命令。

图4-1 SM的架构图

3、客户端管理模块

整个云应用服务业务都是由点播客户端首先发起的，最后也是由点播客户端结束。SM模块需要知道点播客户端的实时状态，这通过SM和客户端的心跳测试和实时状态查询来实现的。

4、日志记录模块

整个NGOD架构的运行操作都要被日志文件记录下来。SM每收到一条消息、每一次具体的操作过程，开始和结束都要调用日志服务器的相应接口，记录消息内容和操作的运行状态。

5、SM模块的相关接口

SM模块设计到的接口可以分为三类：

1. 同其他功能组件通信的会话管理接口，S1、S3、S6、S7接口，其中S1、S3、S6接口是基于NGOD架构定义实现的，S7接口是本课题对NGOD架构扩展新增加的，这几个接口都是实现的RTSP协议。
2. 数据存储接口，云应用服务系统的各类数据都要保存到数据库中，本课题使用的是NGOD架构定义的数据存储接口。
3. 日志记录接口，云应用服务系统的所有运行状态都要记录到日志文件中去，用来跟踪系统的工作状态，本课题使用的是NGOD架构定义的日志接口。

本课题根据上述SM模块的功能需求和NGOD现有业务与新增业务的相同与不同，改进扩展了NGOD架构原有的SM模块，SM针对点播客户端的请求建立会话的过程如图4-2所示。

|  |
| --- |
|  |
|  | C:\Users\109421\AppData\Local\Temp\ksohtml\wps1B0B.tmp.png |

图4-2 SM内部模块调用关系

1. SM模块在S1接口收到一个会话请求时，内部的会话管理模块会创建一个新的会话，并处理请求中包含的参数信息。这个过程的具体实现是首先调用lockfile()函数获得对共享内存的写权限，同时将工作进程和线程的信息记录入日志，然后调用process\_manager()函数创建一个新的会话，具体任务的处理通过调用pthread\_SM()函数完成。
2. 在phread\_SM()函数的执行过程中，通过调用客户端管理模块来检查客户端当前状态和客户端应用服务请求的RTSP消息的参数是否合法。
3. 调用资源管理模块中的alloc\_stratage\_choose()函数选择合适的分配策略，继而通过S6接口向ERM请求边缘设备QAM通道资源，通过S3接口向ODRM请求推流服务器资源，通过S7接口向ASM请求应用服务器资源并部署云应用服务。
4. 获得资源分配的参数，返回会话管理模块，向客户端发送应答消息。
5. 在整个会话处理过程中，各个功能模块通过调用日志接口记录会话的实时状态。

### 4.1.2 SM模块的关键数据结构

这一部分主要介绍SM模块实现过程中涉及到的主要的数据结构。

1、typedef struct {

int flag;/\*标识该段内存数据是否为空\*/

char \*mmap\_queue;/\*sizeof(int)+sizeof(SOCKET\_DATA)\*/

}Queue\_info;

该数据结构定义了共享内存区域的结点指针信息，主进程通过传递该数据结构告知子进程获取共享内存区域的数据，在每一次写共享内存操作时要记录日志。

2、typedef struct{

int pid;/\*进程pid\*/

char \*pid\_queue\_addr;/\*共享内存节点指针\*/

}process\_info;

该数据结构定义了每个由父进程派生的子进程的相关信息，在每一次派生过程中记录日志。

3、typedef struct{

int data\_id;/\*共享内存区的数据编号\*/

int data\_flag;

int pthread\_flag;

int sd;

int data\_len;

char data[1024];

}pthread\_queue;

该数据结构用来存储每一个线程处理的具体数据内容，由子进程定义，传递给线程处理，在每一次子进程接受任务时生成。

4、typedef struct

{

int data\_flag;/\*有无数据\*/

int process\_flag;/\*进程标志\*/

int data\_len;

char data[1024];

}SOCKET\_DATA;

该数据结构是共享内存区域的数据结构，父进程在收到来自各个接口的消息后，生成该数据结构并存入共享内存。

5、typedef struct \_QAM\_INFO

{

INT64 bandwidth;

char qam\_name[128];

char client[128];

char modulation[128];

}QAM\_INFO;

该数据结构记录了S3、S6、S1消息中传递的QAM资源信息，每一次S3、S6、S1接口会话内容中包含该数据结构。

6、typedef struct \_S6\_SETUP\_MSG

{

char ip[40];

int cseq;

char require[256];

char session\_group[256];

char encryption\_type[128];

char cas\_id[128];

char encrypt\_control[256];

char ondemandsessionid[256];

char policy[128];

char inband\_marker[256];

QAM\_INFO qam\_info[QAM\_NUM\_MAX];

int qam\_num;

}S6\_SETUP\_MSG;

该数据结构是接口S6的SETUP消息的数据结构，SM通过该消息向ERM发出会话建立请求，请求边缘设备资源。

7、typedef struct \_S1\_SETUP\_MSG

{

char sm\_ip[40];

int sm\_port;

char rtsp\_version[10];

char require[40];

char purchase\_token[37];

char server\_id[40];

int cseq;

char client\_session\_id[50];

QAM qam[QAM\_NUM\_MAX];

int qam\_num;

char app\_id[256];

int app\_type;

}S1\_SETUP\_MSG;

该数据结构是接口S1的SETUP消息的数据结构，点播客户端向SM发送SETUP消息请求应用服务。

8、typedef struct \_S3\_SETUP\_MSG

{

char odrm\_ip[40];

int odrm\_port;

int cseq;

char require[40];

char ondemand\_session\_id[256];

char sop\_group[5][20];

S3\_QAM qam[QAM\_NUM\_MAX];

int qam\_num;

char session\_group[20];

int start\_point\_slot;

char start\_point\_npt[50];

char policy[128];

char inband\_marker[256];

char content\_type[30];

//sdp消息参数

int sdp\_version; //NGOD中默认为"0"

char email\_add[5]; //NGOD中默认为"-"

char ntp[50]; //the time that the session setup message was created

char add\_type[10]; //"IN"

char ip\_version[10]; //"IP4"

char sm\_ip[40]; //创建会话的服务器地址，SM或ODRM的IP，此处应为SM的IP

char s[2]; //NGOD中默认为" "

int time[2]; //Describes the validity start/end times of the session. 0 indicates media is always available

//a=X-playlist-item

char provider\_id[128]; //the CableLabs provider\_id for the content asset

char asset\_id[40]; //the CableLabs asset\_id for the content asset

int start\_npt; // an 8-digit hex ASCII value, no “0x” prefix and no leading zero

int stop\_npt; // an 8-digit hex ASCII value, no “0x” prefix and no leading zero

char c[15]; //NGOD中默认为"IN IP4 0.0.0.0 "

char m[17]; //NGOD中默认为"video 0 udp MP2T"

}S3\_SETUP\_MSG;

该数据结构是接口S3的SETUP消息的数据结构，SM向ODRM发送SETUP消息来请求推流服务器资源。

9、typedef struct \_S7\_SETUP\_MSG

{

char asm\_ip[40];

int asm\_port;

int cseq;

char require[256];

char session\_group[256];

char ondemandsessionid[128];

char policy[128];

char app\_id[256];

int app\_type;

S7\_SS ss;//推流服务器信息

}S7\_SETUP\_MSG;

该数据结构是S7接口的SETUP消息的数据结构，SM通过向ASM发送SETUP消息来请求应用服务器资源，并部署应用。

## 4.2 ASM模块的设计与实现

ASM模块负责分配和管理应用服务器的资源，选择应用服务器部署应用。ASM一收到SM发送的SETUP消息后，ASM就需要按照选定的资源分配策略选择应用服务器，然后向该应用服务器通过R8接口发送SETUP消息，通知该应用服务器部署应用。ASM还实现了D8接口，应用服务器通过该接口向ASM注册自身、上报自身的参数，ASM维护着一个自己管理的应用服务器的负载情况的信息表，应用服务器每20秒回通过D8接口的UPDATE消息更新自己的负载情况，ASM选择应用服务器时就是根据该信息表来选择。

### 4.2.1 ASM架构的研究与选择

服务器端的架构有很多种选择，每种都有着自己的优缺点，需要根据实际的需要情况来选择使用哪种架构。总体来看，服务器端的架构有三种选择：

1. 为每个业务请求派生一个进程进行处理并相应。
2. 预先派生子进程处理用户请求。
3. 预先派线程处理用户请求。

第一种，为每个业务请求派生一个进程处理并相应，这种实现方式比较简单，实现起来也很容易，大致的代码实现如下：

listenfd = socket();//创建监听套接字

bind(listenfd, ...);//为监听套接字绑定端口和IP地址

listen(listenfd, ...);//监听

while(true)

{

connfd = accept(listenfd, ...);//接受客户端的连接请求

if((pid = fork() == 0)//子进程

{

close(listenfd);

handle(connfd);//接收并处理消息

close(connfd);

}

close(connfd);

}

通过这种方式，一台服务器可以同时为多个客户端服务。这种方式的瓶颈就在于一个进程中能同时派生出多少个子进程。对于网络请求不繁忙的服务器来说，这种实现方式是可以满足需求的。但是当服务器的并发数量大幅增长后，这个模型肯定不能满足服务器的需要。它简单在于为每个用户派生一个进程即可，而最大的弊端也在与此，因为系统派生一个进程的开销太大了。当用户对服务器的请求大多是短链接的情况下，它的弊端暴露的尤其明显。用户请求完毕就关闭连接，而服务器却为短暂的连接消耗了大量的资源，服务器的大部分时间被浪费在了分配子进程资源上，而不是业务处理上。因此这种方式对于繁忙的边缘资源管理器来说是不可取的[7]。

第二种，预派生子进程的方式，这种方式是对第一种方式的改进。它并不等到用户的请求到达后再派生子进程，而是提前派生一定数量的子进程，每个子进程都在等待用户请求的到来。这样，当用户请求到达时，就可以立刻为用户服务了。

这种方式避免了用户请求到达时派生子进程的时间消耗，提高了响应的速度，而且更能适应短链接的情况。但是提前派生多少子进程，这没有统一的规定，要根据具体情况来分析。如果分配多了，势必会造成资源浪费。如果分配少了，那么当用户请求比较多时，又要派生新的子进程，又要产生派生子进程的时间消耗。另外，可能存在这样一种情况，有多个子进程同时都阻塞在accept调用上，当一个用户请求到达时，这些子进程都会被唤醒，但是却只有一个子进程会接收消息并处理，这种现象被称为“惊群”。尽管所有的子进程被唤醒，但是并不会影响程序的运行，那些没有得到连接的进程会继续进入阻塞状态，这种现象虽然不影响程序的逻辑，但是这种情况毕竟会造成子进程无谓的唤醒阻塞，浪费系统的资源[7]。有两种方法可以避免“惊群”现象的出现。一种是在在accept调用之前增加一个锁，让所有子进程都阻塞在这个锁上，只有一个子进程会获得锁并且阻塞在accept调用上，这样当一个用户请求到达时，就只会有一个子进程被激活并且接收处理用户的请求。获得请求之后，该子进程就会释放锁，由其他子进程继续获得锁并阻塞在accept调用上。在Linux操作系统环境下，可以选择文件锁或者线程锁两种方式。另外一种解决方案就是只有主进程阻塞在accept上，当主进程接受一个用户连接请求后，将该连接的套接字描述符传送给子进程，主进程继续阻塞在accept调用上，而由子进程通过这个套接字描述符接收处理用户的请求。这么做就避免了对所有的accept调用加锁解锁的过程，在性能上会有所提高，但同时也产生了向子进程传递套接字描述符以及管理子进程的运行状态的需要，势必会增加编写代码的复杂性。处理进程消耗系统资源过多的影响，综合其他因素考虑，这是一个不错的方案。

第三种，预先派生线程处理用户请求。这种方式与第二种类似，只不过将派生出来的进程改为了更轻量级的线程，对进程的处理方式也需要改为对线程的处理方式。由于操作系统在创建或者释放一个线程的时间开销要小于创建或释放一个进程的开销，因此，预先派生线程处理用户请求的方式较之预先派生进程处理用户请求的方式，性能要有所提高。

本课题需要设计实现的ASM模块，处理的并发量比较大，同时由于要管理分配应用服务器的资源，ASM还要请求建立与选定的应用服务器的连接，之后才能向SM发送SETUP Response消息，所以ASM与SM之间的连接时间会比较长，而ASM与应用服务器之间的RTSP会话时间要短一些。另外，ASM与应用服务器之间还要建立VREP会话连接，用于管理更新应用服务器的负载情况，这是一个长连接。在长连接和短链接都存在的情况下，考虑到进程派生的开销较大以及并发度较高，因此采用预先派生线程的方式。在Linux操作系统中，每一个进程可用的内存空间是受限的，通常情况是2G，而每个线程又要为自己保留大约1M的内存空间，所以理论上一个进程最多可以派生2048个线程。而在实际中，又要有其它的内存消耗，所以一个进程可以派生的进程数一定会小于2048。由此可以得出结论，只采用单个进程派生多线程的方式肯定不能满足本课题高并发的需求，需要采取多进程的方式。综上，本课题的ASM模块的设计实现采用多进程与线程池结合使用的方式。具体设计如图4-3所示。

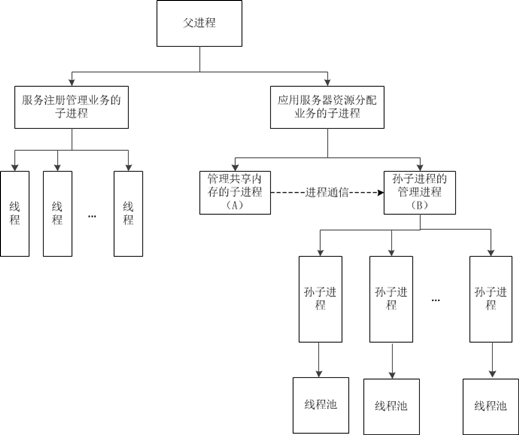


图4-3 ASM详细设计图

系统在启动时，ASM要同时负责应用服务器的服务注册与负载更新，以及应用服务器资源的管理与应用部署，因此主进程创建了两个子进程负责这两种业务。其中，服务注册管理业务的子进程负责接收处理D8接口的消息，并更新应用服务器负载信息表。负责应用服务器负载分配业务的子进程同时接收来自SM的RTSP消息和来自应用服务器的RTSP消息，根据当前系统的运行状态，选择是创建新的子进程还是使用现有子进程的线程池中的线程来完成业务逻辑。在这之后，依据选定的负载均衡策略，选择合适的应用服务器来部署应用。

### 4.2.2 注册管理功能的实现

ASM模块的注册管理功能主要是通过D8接口通信来实现的。注册管理功能主要针对的是应用服务器资源，在系统部署初期大致可以确定应用服务器的数量，在之后会有所增加，但不会太多。因此使用单个子进程线程池的方案足够。即主进程监听端口，当有应用服务器的VREP连接请求到达时，创建线程，并由该线程管理该连接套接字。在收到应用服务器发送的OPEN消息后，向应用服务器负载信息表中添加该应用服务器的信息，之后该线程等待应用服务器发送的UPDATE负载更新消息，收到后更新应用服务器负载信息表。

关键的数据结构如下：

1、

typedef struct \_OPEN //OPEN消息

{

short Len; //消息长度(包括自己)

byte type; //消息类别

short version; //协议版本

short holdtime; //ED申请的HoldTime值

byte ID[4]; //VREP标识(如果N接口没有配则设为设备的IPv4地址)

short parametersLen; //消息所带参数的总长度

Parameters parameters[256]; //参数队列

int ParameterNum; //参数个数

Capability capability[256]; //功能(参数的一种)队列

}OPEN;

该数据结构是D8接口的VREP OPEN消息的数据结构。ASM和AS通过发送OPEN消息建立VREP连接。

2、

typedef struct \_UPDATE //UPDATE消息

{

short Len; //消息长度

byte type; //消息类型

Attributes attributes[256]; //参数指针(指向所有参数的字符串，无实际意义)

int AttributeNum; //参数个数

int ServiceStatus; //服务状态

double cpu; //CPU最近1分中的负载

int memory; //应用服务器的可用内存

}UPDATE;

该数据结构是D8接口的VREP UPDATE消息的数据结构。ASM在通过OPEN消息协商建立VREP 连接后，只具有接收UPDATE消息的能力。AS在经过协商建立连接后，只具有发送UPDATE消息的能力。AS每隔20秒通过UPDATE消息向管理自己的ASM上报自身的负载情况，包括CPU负载和可用内存。ASM接收到OPEN消息后，通过解析获得消息内的参数，然后获得应用服务器负载信息表的锁，修改该信息表。

3、

typedef struct ASNode {

double load; //应用服务器CPU负载

double memory; //应用服务器已被占用内存的百分比

double weight; //应用服务器负载权值

char ip[40]; //应用服务器IP地址

int port; //应用服务器VREP的端口

}ASNode;

该数据结构是应用服务器负载信息表的结点的数据结构。

4、

typedef struct List {

ASNode\* nodes;

int length; //哈希表长

int number; //表中应用服务器的结点个数

int min; //表中负载最小的应用服务器结点的索引

}List;

该数据结构就是应用服务器负载情况信息表的数据结构。该信息表采用哈希表的实现形式，这样做的好处是查找定位简单。根据应用服务器的IP地址计算出哈希值，该值就是该应用服务器的信息在哈希表中的位置。对哈希表中出现的冲突，采用开放定址线性探测再散列的方法来处理。当表中结点的个数除以表的长度的值大于0.75时，需要对表进行扩展。同时，信息表维护着一个min值，该值指向了哈希表中负载权值最小的应用服务器的结点。如果新插入或者更新后的结点的权值小于min指向的结点的权值，就要更新min值。

### 4.2.3 资源分配功能的实现

资源分配功能的流图如图4-4所示。

|  |
| --- |
|  |
| C:\Users\109421\AppData\Local\Temp\ksohtml\wps1B4B.tmp.png |  |

图4-4 ASM模块资源分配模块的流图

为了应对并发量上升的情况，资源分配功能进程模型以及消息的接收处理比较复杂，下面就此进行详细的说明。正如前面所述，资源分配由一个单独的进程进行管理，它是整个资源分配过程的基础。多进程与线程池结合的方式可以应对高并发，但是不可以应对无上限的并发。原因就在于这个进程，一个系统可以生成的进程数是要受限的，一个进程可以派生的线程数也是受到限制的，当然线程池中的线程数量也受制于进程空间的限制[7]。

假如说我们有n1个进程，每个进程可以派生出n2个线程，但服务器端的并发能力不能简单算作n1 \* n2。因为在通常情况下，如果一个端口无法立即处理达到的请求，操作系统会将这个请求放入一个缓冲区中，这个缓冲区是用来保存那些已经完成三次握手但是还没有获得处理的连接。可以在调用listen函数时设置该缓冲区。这些未处理的连接不是一直保存的，超过一定时间后，这些连接就要从缓冲区中被删除。如果业务流程处理的比较快的话，所用时间小于一个连接在缓冲区中保留的时间，那么服务器端的并发能力就会大于n1 \* n2。这只是理论上的一些分析，还有一些其它因素未考虑，实际会更复杂一些。但是从上面可以得出一个结论，提高业务处理的速度，对提高服务器端的并发能力有很大帮助。

进程的处理流程如下：

1. ASM应用服务器资源分配的子进程负责接收来自所有接口的请求或响应，这部分是通过epoll方式来实现的。主进程通过多次调用mmap函数把多个文件映射为一个由多个共享内存区域组成的队列，作为派生的子进程以及线程池中的线程之间数据传送的中介。mmap函数能把一个文件或者一个Posix共享内存区对象映射到调用进程的地址空间。选定一个内存映射文件，在open该文件后，调用mmap把它映射到调用进程的地址空间，这样做有一个好处是，所有的I/O都在内核的掩盖下完成，程序员不需要调用read和write来执行I/O，只需要编写存取内存映射区中各个值的代码[13]。每一个孙子进程使用共享内存队列中的一个结点，然后再划分该结点的共享内存区域，为该子进程派生出的线程池中的每一个线程数分配一块内存来完成业务处理。在定位内存块的时候，通过使用队首指针与在此队中的索引即可。之所以使用共享内存是因为相对信号量，管道，消息队列等它是最高效的进程通信机制，因为它不需要在进程之间复制数据[7]。

其中，共享内存队列的结点的数据结构是：

typedef struct {

int flag;/\*标识该段内存数据是否为空\*/

char \*mmap\_queue;/\*sizeof(int)+sizeof(SOCKET\_DATA)\*/

}mmap\_queue\_info;

其中mmap\_queue即指向mmap映射的共享内存区域。

1. 之后ASM应用服务器资源分配进程会通过调用pipe()创建一个管道与调用socketpair()创建一个套接字，然后派生两个进程（图4-3A，B），其中A进程在监听套接字上阻塞，当有消息到来时将消息放入共享内存，并将共享内存的首地址与接受连接的描述符传递给B进程。共享内存是由A进程维护的，共享内存空间不够用时，该进程会需找释放一块共享内存空间。共享内存在队列里是线性排列的，即从前往后，如果最后找不到合适的空间就从前寻找未被占用的空间。

|  |
| --- |
|  |
|  |  |

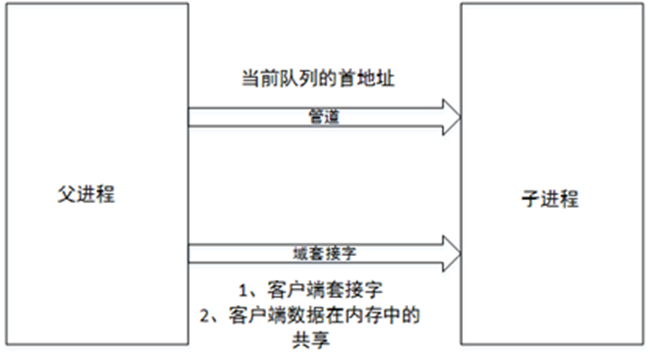


图4-5 父子进程通信示意图

1. 进程B通过调用pipe()创建一个管道，然后派生一个孙子进程，孙子进程在创建之后就会初始化一个线程池。然后根据A进程传送给自己的共享内存首地址和文件描述符，从线程池中取出一个空闲线程，把这两个参数传递给该线程。
2. 共享内存不为空时，线程会从共享内存中读取消息，解析该消息，然后处理消息中的参数。在解析消息的过程中，如果有参数不合法或者消息格式不正确，就会发送错误报告。针对不同的消息类型，会调用相应的函数来处理该消息。在处理过程中，同时完成数据库的存储和日志的记录。进程B会利用之前创建的管道等待接收孙子进程的共享内存用完的消息，并等待其派生的所有线程处理完后回收该共享内存[8]。

这样的实现方式的好处是主进程只负责接收消息和管理共享内存队列，当有消息到达后，主进程只需要把消息放入共享内存，具体的消息处理以及发送应答都由线程来完成。主进程不直接处理消息，这样就保证了大并发环境下主进程主进程不会因为任务过多而丢弃消息。同时，线程是由子进程创建的，这也减轻了主进程的负担。

### 4.2.4 资源分配策略

对于本课题提出的广电网上部署云应用服务的管理方案，应用服务器是最终的应用运行平台，如何管理应用服务器的资源、最大化的利用应用服务器的资源，同时保证各服务器之间的负载均衡，增强系统的可靠性，是该方案的重中之重。

对于应用服务器来说，影响其负载情况变化的有很多因素，包括服务器自身的硬件条件、操作系统、用户点播的应用的类型、资源消耗等。应用服务器上各应用运行时占用的资源是各不相同的，有的应用会非常占用服务器资源，有的应用占用的就少，有的应用占用CPU较多，有的应用比较耗内存。负载变化也是一个随机的过程，有很强的波动性，可能一个应用在运行中占用的CPU或内存会急剧增加。另外，负载变化与时间有很大的关联，过去的负载对现在的负载情况有很大影响，当然，也存在负载值突变的情况[14]。这些情况下，如何管理服务器的资源并维持负载均衡，是一个难点。

|  |
| --- |
|  |
|  | C:\Users\109421\AppData\Local\Temp\ksohtml\wps1B5D.tmp.png |

图4-6 基于客户代理的客户端负载均衡机制

图4-6所示为基于客户端代理的客户端负载均衡机制。目前，很多成熟的集群技术都是采用的客户代理的客户端负载均衡的模式作为其负载均衡机制的基础[15]。客户代理维护着对多个服务器的引用。每一次有客户请求到达时，客户代理都会根据当前各服务器的信息以及选定的负载均衡策略来决定采用哪一个服务器为客户服务，然后向该服务器发出请求。这种负载均衡机制的优点是：第一，这种机制具有很好的可伸缩性，增加或者减少服务器都很方便；第二，维护负载均衡的计算和开销都由客户代理来承担，服务器不需要担心因为并发的增多而引起的维护负载均衡的开销的增加；第三，这种机制使用了分布式的负载均衡格局，避免了集中式负载均衡方法有可能产生的负载瓶颈和单点失效问题。

本课题负载均衡的实现就采用了这种基于客户端代理的客户端负载均衡机制，其中，ASM充当客户端代理的角色。

我们常用的负载均衡算法主要分为两种：静态负载均衡和动态负载均衡[16]。静态负载均衡就是提前设定好固定的分配方案，不需要考虑服务器的实际负载情况。优点是开销小、实现简单，但是因为用户点播的应用占用资源情况不同、不固定，当服务器集群的状态发生变化时（尤其是本课题的云应用服务），很难通过修改参数或者调整策略来维持负载均衡，无法适应新的情况，因此静态负载均衡的实际效果不能令人满意。动态负载均衡考虑了各服务器的实时负载情况，选定一台服务器来分配任务，以此来提高整个系统的吞吐率和降低响应时间，实际运行情况要好于静态负载均衡。动态负载均衡的缺点是需要实时收集各服务器的负载情况，这会造成额外的开销，但由于具体实现时只能间歇地收集信息，有可能因为负载信息的不及时而造成误导。同时动态负载均衡的实现要比静态负载均衡复杂。综合以上的内容，本课题采用动态负载均衡算法。

反映服务器负载情况的因子很多，本课题只考虑CPU负载和内存占用。表4-1所示为一些根据CPU和内存的使用情况决定是否可以向一台应用服务器分配任务的示例。

|  |  |  |
| --- | --- | --- |
| CPU | Memory | 是否可以继续分配任务 |
| 55% | 91% | 否 |
| 55% | 61% | 是 |
| 64% | 52% | 是 |
| 65% | 49% | 否 |
| 83% | 57% | 否 |
| 58% | 66% | 是 |
| 51% | 77% | 否 |
| 74% | 78% | 否 |
| 58% | 89% | 否 |
| 46% | 43% | 是 |
| 82% | 67% | 否 |
| 90% | 87% | 否 |

表4-1 负载情况决策表示例

对此，本科题对系统负载划分了范围[16]。

|  |
| --- |
|  |
|  | C:\Users\109421\AppData\Local\Temp\ksohtml\wps1B6E.tmp.png |
|  | |
| C:\Users\109421\AppData\Local\Temp\ksohtml\wps1B7E.tmp.png | |  |

对应之前的示例可得：

|  |  |  |
| --- | --- | --- |
| CPU | Memory | 是否可以继续分配任务 |
| 1 | 3 | 否 |
| 1 | 2 | 是 |
| 2 | 1 | 是 |
| 2 | 1 | 否 |
| 3 | 1 | 否 |
| 2 | 2 | 是 |
| 1 | 2 | 否 |
| 2 | 2 | 否 |
| 2 | 3 | 否 |
| 1 | 1 | 是 |
| 3 | 2 | 否 |
| 3 | 3 | 否 |

表4-2 对照负载情况决策表示例

ASM会维护一个服务器负载情况信息表，并根据CPU使用率和内存占用百分比来计算负载值。可以引入一组动态系数wi来反映这两个因子的重要程度，其中 ，wi越大说明该因子的重要性越高。负载值计算公式是：

每一个应用服务器上都运行着一个后台进程，每个20秒，该进程会读取/proc路径下的文件，获得CPU使用情况和内存消耗情况，通过D8接口通知ASM负载变化情况，ASM会更新自身维护的相应应用服务器的负载信息。如果在一个更新周期内，没有收到某一台应用服务器的VREP UPDATE消息，则认为该应用服务器不可用，ASM将应用服务器负载信息表中该应用服务器对应的负载值设为无穷大。

本课题选择实现的动态分配算法是最小负载分配。原理是对于到达的每一个用户请求，ASM都选则负载信息表中负载值较小的应用服务器分配任务。具体流程如下：

1. 初始化负载信息表。
2. 如果负载信息表中各应用服务器的负载值均为空，那么一次将任务分配到每一台注册的应用服务器上。
3. 收到用户的应用服务请求后，从负载信息表中选择负载值最小的结点，向该结点对应的应用服务器发送SETUP消息，请求部署应用。
4. 收到应用服务器发送的VREP UPDATE消息，更新负载信息表，重新计算权值，并更新负载信息表中负载值最小结点的索引。
5. 若一个更新周期内没有收到某一台应用服务器的UPDATE消息，将负载信息表中该应用服务器对应的结点的负载值设为无穷大。
6. 如果有一台应用服务器新加进来，刚刚注册，则选择该应用服务器作为下一次任务分配的目标。

## 4.3 应用服务器组件的设计与实现

### 4.3.1应用服务器相关接口

应用服务器为用户请求的应用服务提供运行环境，并接受用户客户端的操作指令，与正在运行的应用结合，其结果经过应用服务器的显卡渲染后，被编码为音视频流传输至客户端。

应用服务器上实现的接口包括V5视频传输接口，C2用户操作指令传输接口，R8资源管理接口和D8服务注册发现接口。

用户在客户端请求应用服务，经过SM和ASM对请求消息的处理后，应用部署在了一个选定的应用服务器上。之后客户端会直接与该应用服务器建立会话。会话的上行通路客户端通过C2接口向应用服务器传输操作指令，该接口实现了RTSP协议。会话的下行通路是应用服务器通过V5接口向推流服务器传送经过编码后的视频流，推流服务器再将视频流不存储直接推至边缘设备，在经过HFC网络传输至客户端。V5接口实现的是RTP（Real-time Transport Protocol）协议。V5接口和C2接口并不在本论文的讨论范围内，所以不再详述。

R8接口定义于ASM和应用服务器之间，ASM通过该接口管理应用服务器资源。ASM收到SM的SETUP请求后，会选择一台应用服务器为用户服务，之后向该应用服务器发送SETUP请求。应用服务器收到该消息后，会将应用部署，然后通过R8接口向ASM发送应答消息。如果应用服务器自身出现错误或者异常，会通过R8接口向ASM发送ANNOUNCE消息报告事件。ASM通过R8接口向应用服务器发送TEARDOWN消息结束会话，应用服务器会立即终止正在运行的应用，关闭V5接口和C2接口，然后向ASM发送应答。

D8接口定义于ASM和应用服务器之间，应用服务器通过该接口向ASM注册资源并更新自身的负载情况。应用服务器和ASM通过OPEN消息建立VREP会话后，应用服务器就只具备发送UPDATE消息的能力，如果收到UPDATE消息，直接丢掉。ASM只具备接受UPDATE消息的能力。应用服务器每个20秒，就会获取一次自身的负载信息，包括CPU的负载和空闲内存数，然后通过UPDATE消息向ASM报告。

### 4.3.2应用服务器端的虚拟化

对云端资源实现虚拟化，主要有三种技术方式：服务器虚拟化、桌面虚拟化和应用虚拟化[17]。

服务器虚拟化是将服务器物理资源抽象成逻辑资源，将CPU、内存、磁盘、I/O等硬件变成可以动态管理的“资源池”，一台服务器变成几台甚至上千台服务器。将服务器硬件资源按需分配给每台虚拟机，虚拟机上再安装操作系统和用户所需的应用程序。采用服务器虚拟化方式，可以为每个用户分配一台虚拟机，这样便于资源管理和用户数据的管理，并且用户与用户之间有很好的隔离性。但是因为其实现方式是在系统里安装虚拟化软件，再在虚拟机上安装操作系统，所以就会产生原生系统和虚拟化软件两层消耗。即使以Hyper-V为代表的虚拟化技术实现了跨过原生系统，直接在硬件上安装虚拟化软件，但虚拟机上运行操作系统本身就是很大的消耗。随着用户的增多，需要配置越来越多的虚拟机，这会带来一个无法忽略的成本。另外，由于需要对每台虚拟机进行维护、安全检测，对系统管理员来说，工作量太大，并且可能因为无法及时对虚拟机进行维护检测带来很大的风险[18]。

桌面虚拟化指将计算机的桌面进行虚拟化，以达到桌面使用的安全性和灵活性。桌面虚拟化依赖于服务器虚拟化，与服务器虚拟化有着类似的优缺点。

应用虚拟化指将应用程序与操作系统解耦合，为应用程序提供一个虚拟的运行环境。在这个环境中，不仅包括应用程序的可执行文件，还包括它所需要的运行时环境[19]。从本质上说，应用虚拟化是把应用对底层的系统和硬件的依赖抽象出来。每个通过应用虚拟化技术封装的应用程序，在相互独立的虚拟化环境中运行，这样可以减少不同应用程序之间出现的冲突，提高应用程序的兼容性，同时可以实现在同一台计算机上运行同一种应用程序的不同版本，具有很大的灵活性。应用虚拟化简化了应用安装过程，提高了应用部署的速度。采用应用虚拟化技术，系统管理员可以直接在服务器上对系统进行维护检测，减少了工作量，降低了风险。

综上考虑，本课题采用应用虚拟化技术在应用服务器上部署应用。

### 4.3.3 部署应用

如果将所有的应用部署在每台应用服务器上，这是不现实的。应用服务器主要是提供应用运行的环境，虚拟化后的应用存储在专门的组件上，本科题添加了这样一个组件——存储服务器。

TFTP（Trivial File Transfer Protocol，简单文件传输协议）是TCP/IP协议族中的一个用来在客户机与服务器之间进行简单文件传输的协议，提供不复杂、开销不大的文件传输服务。端口号为69[20]。TFTP可用于UDP环境，代码所占的内存较小，这对于应用服务器来说极为重要。TFTP协议不需要验证客户端的权限，一般应用于局域网。而应用服务器与存储服务器位于同一个局域网，相互之间文件传输也是很安全的，并不需要验证客户端[21]。综上所述，本课题采用TFTP作为应用服务器与存储服务器之间的文件传输协议。

ASM在收到SM发送的SETUP消息后，根据当前各应用服务器的负载情况，依据上文给出的负载均衡策略，选定应用服务器为用户提供服务，然后通过R8接口向该应用服务器发送SETUP消息。应用服务器接收到该SETUP消息后，向存储服务器请求传输应用虚拟化后的应用软件，并将该应用部署到应用服务器上，最后通过R8接口发送RTSP SETUP Response消息，报告应用部署的路径。

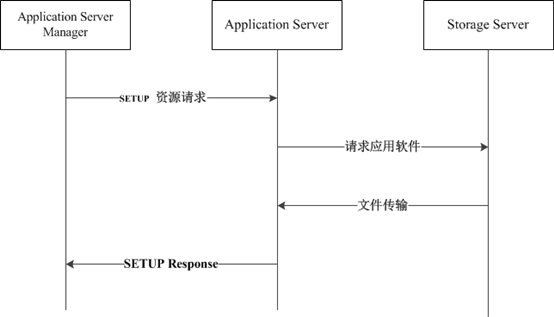


图4-7 应用部署时序

## 4.4本章小结

本章详细介绍了对NGOD架构的扩展实现，包括对SM模块功能的扩展，对ASM模块的设计实现，对应用服务器的设计实现。同时，还介绍了本科题实现的应用服务器负载均衡的策略。

# 第五章 系统测试

## 5.1测试环境

硬件环境：

1. PC电脑：Pentium(R) Dual-core cpu T4200 @2.0GHz + 2GB
2. 服务器：Intel(R) Xeon(R) cpu E5640 @2.67GHz + 3GB

软件环境：

1. 操作系统：Cent OS6.5
2. 软件服务器：ERM Server、ODRM Server、SM Server、ASM Server，它们的IP和端口配置如表5-1所示。

|  |  |  |
| --- | --- | --- |
| 组件 | IP | 端口 |
| SM | 192.168.10.11 | RTSP：6606 |
| ERM | 192.168.10.11 | RTSP：6608 |
| ODRM | 192.168.10.11 | RTSP：6610 |
| ASM | 192.168.10.11 | RTSP：6614 VREP：6616 |
| 应用服务器 | 192.168.10.212 | RTSP：6614 VREP：6616 |
| 应用服务器 | 192.168.10.213 | RTSP：6614 VREP：6616 |
| 应用服务器 | 192.168.10.214 | RTSP：6614 VREP：6616 |

表5-1 测试环境各服务器IP和端口配置

应用软件：ORACLE数据库

在PC上运行着SM、ERM、ODRM、ASM服务器程序。应用服务器有三台，等待应用部署。同时192.168.10.212兼做存储服务器。

## 5.2测试用例和结果

本次测试包括：

SM和ASM会话建立请求的交互

SM和ASM通过接口S7建立会话请求，通过SM向ASM发送RTSP SETUP Request消息，测试ASM的应答消息。

RTSP SETUP Request消息：

SETUP rtsp://192.168.10.11:6614 RTSP/1.0

CSeq:313

Require:com.comcast.ngod.S7

OnDemandSessionId:be074250-cc5a-11d9-8cd5-0800200c9a66

SessionGroup:sm1

StreamControlProto:RTSP

ApplicationId:001

ApplicationType:1

Transport:

MP2T/DVBC/UDP;unicast;client=00AF123456DE;destination=192.26.14.1;client\_port=4000;bandwidth=2920263

ASM应答消息如图5-1所示。

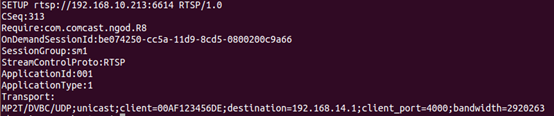
|  |
| --- |
|  |
|  | C:\Users\109421\AppData\Local\Temp\ksohtml\wps1BB1.tmp.png |

图5-1 ASM SETUP Response

该应答消息符合预期的结果，说明SM和ASM之间的交互没有问题，并且ASM能够正确处理SM发送的会话建立请求。

ASM和应用服务器会话建立请求的交互

ASM接收到SM的会话建立请求后，会根据当前应用服务器的负载情况选择一台应用服务器，通过R8接口向该应用服务器发送RTSP SETUP Request消息。消息如图5-2所示。

图5-2 ASM SETUP Request

|  |
| --- |
|  |
|  |  |

应用服务器的应答消息如图5-3所示，符合预期，说明ASM与应用服务器间的交互没有问题，应用服务器能正确处理ASM的SETUP请求，并部署应用。

|  |
| --- |
|  |
|  | C:\Users\109421\AppData\Local\Temp\ksohtml\wps1BD2.tmp.png |

图5-3 应用服务器SETUP Response

SM和ASM会话释放的交互

在收到客户端的会话释放请求后，SM会通过接口S7向ASM发送RTSP TEARDOWN消息，测试ASM的应答。消息内容是：

TEARDOWN rtsp://192.168.10.11:6614 RTSP/1.0

CSeq:314

Require:com.comcast.ngod.S7

Reason:200 User stop

Session:47112344

OnDemandSessionId:be074250-cc5a-11d9-8cd5-0800200c9a66

ASM的应答消息如图5-4所示。

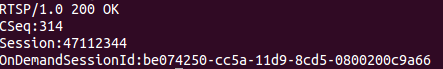


图5-4 ASM TEARDOWN Response

该应答消息符合预期的结果，说明SM和ASM之间的交互没有问题，并且ASM能够正确处理SM发送的会话释放请求。

ASM和应用服务器会话释放的交互

ASM在收到SM发送的TEARDOWN消息后，首先解析消息，然后从数据库中读出Session参数对应的会话的详细信息，向应用服务器发送TEARDOWN消息。消息如图5-5所示。

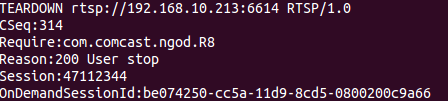


图5-5 ASM TEARDOWN Request

ASM收到的应答消息如图5-6所示。

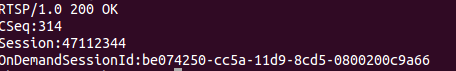


图5-6 应用服务器TEARDOWN Response

该应答消息符合预期。同时查看应用服务器可以看到，之前部署的应用的进程已被杀死。说明ASM和应用服务器之间通过R8接口的会话释放交互没有问题，应用服务器在收到ASM的会话释放请求后，能准确的关闭用户点播的应用。

ASM和应用服务器D8接口的交互。

D8接口实现的是VREP协议，该协议是二进制形式的。测试过程是：首先启动ASM，等待应用服务器发送资源注册信息，之后接收应用服务器发送的负载更新的信息，这两次消息都是VREP UPDATE消息。

ASM接收到的UPDATE消息如下所示：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 字段名 | 字段长度 | 字段值 | 注释 |
| 头部 | Length | 2 | 134 | 消息总长度 |
| Type | 1 | 2 | 更新消息 |
| 更新消息 | Attr Flags | 1 | 0x00 | 熟知属性 |
| Attr Type Code | 1 | 3 | 下一接收信息地址 |
| Attr Length | 2 | 24 | 属性长度 |
| Component addr Length | 2 | 13 | 地址长度 |
| Component addr | 13 | 192.168.10.11 | IPv4地址 |
| Streaming Zone Name Length | 2 | 3 | 区域域名长度 |
| Streaming Zone Name | 3 | SM1 | 区域域名 |
| Attr Flags | 1 | 0x00 | 熟知参数 |
| Attr Type Code | 1 | 241 | 服务状态 |
| Attr Length | 2 | 4 | 属性长度 |
| Service Status | 4 | 1 | 可用 |
| Attr Flags | 1 | 0x80 | 未知参数 |
| Attr Type Code | 1 | 247 | MemoryLoad |
| Attr Length | 2 | 4 | 属性长度 |
| Memory load | 4 | 0.54 | 内存占用百分比 |
| Attr Flags | 1 | 0x80 | 未知参数 |
| Attr Type Code | 1 | 248 | CPULoad |

续上表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 更新消息 | 字段名 | 字段长度 | 字段值 | 注释 |
| Attr Length | 2 | 4 | 属性长度 |
| CPU Load | 4 | 0.49 | CPU状态 |

表5-2 UPDATE消息

## 5.3本章小结

通过对以上测试结果的分析，本课题为NGOD架构扩展的SM、ASM和应用服务器模块及相应接口都能正常工作。ASM能够完成对应用服务器资源的管理和分配，应用服务器能够实现对应用的部署。所有的设计实现均达到预期的效果。

# 第六章 总结和展望

## 6.1全文总结

随着技术与社会的进步，人们对于文化生活水平的要求会越来越高，广电网络的发展也应该跟上这个步伐。能否为用户提供更多的应用服务和内容，成为决定广电网用户量的关键因素。由于中国国内广电网络的特殊，用户端机顶盒更新缓慢、硬件配置低、功能受限，瘦客户端方案仍是受广电运营商关注的一种应用部署解决方案。为此，本课题提出了一种借助于趋于成熟的虚拟化技术，在NGOD架构的基础上，扩展架构搭建可融合于现有广电网络的云应用服务平台的方案。

本文作者的主要工作是：

一，分析当前广电网的运营情况，选择了在广电网络上搭建云应用服务平台的方案为广电网新增业务。

二，分析了广电网现有的NGOD架构方案，并比较了云应用服务业务与现有的视频点播业务的不同，提出了基于现有平台，对NGOD架构进行扩展以兼容现有业务功能的服务端管理方案。

三，设计实现了对NGOD架构的扩展，新增加了应用服务器和应用服务器管理模块，扩展了会话管理模块，，并设计实现了相应的接口，扩展了RTSP协议和VREP协议。

## 6.2全文展望

本课题提出的服务端管理方案充分利用了现有平台的硬件软件资源，实现了新增业务与已有业务的兼容并存，但是，由于时间有限，本文设计实现的方案还有很多需要改进完善的地方。

应用服务器管理模块实现了对应用服务的负载均衡的管理。但是，本课题实现的最小负载分配策略仍偏简单，在实际运行中，有可能出现服务器负载过重的情况。同时，对应用服务器有效性监督是由一个20秒的时间间隔，如果在这20秒内服务器宕机，服务器管理模块不能立刻发现。如果单纯的缩短时间间隔，有会带来更多的开销。这需要一个新的服务器监测机制。

# 参考文献

[1] 胡朋 基于NGOD架构的VOD系统资源管理和服务发现研究[学位论文], 北京邮电大学, 2012

[2] Comcast NGOD:Overall Architecture(Version 2.0) January 2006

[3] 刘莹宇 适合远程教育的分布式VOD系统的研究[学位论文], 重庆大学, 2006

[4] 周宁 VOD视频服务器上关键技术的研究与实现[学位论文], 西安电子科技大学, 2003

[5] 索妮尔 基于NGOD架构的VOD系统内容管理与传播服务的研究与实现[学位论文], 北京邮电大学, 2013

[6] H. Schulzrinn，A. Rao，R. Lanphier RFC-2326 Real Time Streaming Protocol(RTSP), April 1998

[7] 宫伟俊 双向广电网络中统一边缘资源管理器的研究与实现[学位论文], 北京邮电大学, 2013

[8] J. Rosenberg，H. Salama，M. Squire. RFC-3219 Telephony Routing over IP(TRIP), January 2002

[9] 周帅 基于NGOD的SDV系统设计与开发[学位论文], 北京邮电大学, 2014

[10] Comcast. Next Generation On Demand (NGOD 2.0) RTSP Usage Specification. January 2006

[11] Comcast. Next Generation On Demand (NGOD 2.0) Service Discovery & registration Interface. January 2006

[12] 胡朋 基于NGOD架构的VOD系统资源管理和服务发现研究与实现[学位论文], 北京邮电大学, 2012

[13] W. Richard Stevens UNIX网络编程第2卷: 进程间通信 第三版 人民邮电出版社 2010 261-278

[14] 杨伟, 朱巧明, 李培峰 基于时间序列的服务器负载预测 计算机工程 32(19) 2006 143-145

[15] 王梓又, 周明辉, 梅宏 一种动态的客户端负载均衡机制 中国科学 43(1) 2013 60-72

[16] 陈亮, 王加阳 基于粗糙集的负载均衡算法研究 计算机工程与科学 31(1) 2010 101-104

[17] 付平武 应用虚拟化技术应用与研究 电脑知识与技术 9(20) 2013 4732-4733

[18] 周波 服务器虚拟化的优缺点分析及趋势 微型电脑应用 28(9) 2012 56-58

[19] 杨娴, 陈麟 云计算环境下的应用虚拟化的研究 软件 33(4) 2012 74-77

[20] 张西红, 周其刚, 张树京 在国际互联网上实现FTP与TFTP两种通信协议的比较 电信快报 (6) 1997 20-22

[21] 郭秋丽 基于TFTP协议的简单文件传输系统的研究 电子技术 (7) 2013 1-3

# 致谢

时光荏苒，岁月如梭。转眼间，我的硕士研究生阶段就要过去了。回望这两年多的时间，太多的事、太多的人需要我们去牢记。在此，对所有帮助过我的人，表示诚挚的感谢。

首先，我要感谢我的导师王志谦老师。我在各个方面都得到了王老师的帮助和指导。您渊博的知识、严谨的治学态度、丰富的经验值得所有人钦佩。每每在项目遇到瓶颈困难之际，您总是能够给予我启发和鼓励，教会我如何在碰到难题时勇往直前，如何在遇到瓶颈时扩展思路。在您的细心指导下，我学到了一系列解决问题的方法和宝贵的经验，这些经验将使我终身受用。可以说能在研究生期间遇到王老师这样的良师益友是我人生中最幸运的事。

感谢实验室的马严老师，黄小红老师，柯宏力老师和徐明昆老师，四位老师同样拥有广博的学识和丰富的经验，在我论文的撰写过程中给与我很大的帮助，提出了很多指导性的修改意见，鞭策我用更高的标准来要求自己。

感谢在工作和生活中给予我无私帮助的各位师兄师姐，周帅、陈建红、张丽勤和温平华。感谢你们在平日生活中解答我各样的问题，是你们耐心的帮助使得我顺利完成研究生的学业。

感谢和我一起度过两年多研究生时光的徐扬、高宇翔同学，还有信网中心的其他同学，我们一起欢笑、一同成长。也许毕业后我们要奔走四方寻找自己的前程，但是，同学们，不管身在哪里，我永远不会忘记你们。

然后，我要感谢我的父母，感谢他们为我的成长所付出的心血和汗水，感谢他们一直在我的身边鼓励我、支持我，他们是我前进的不竭动力。

最后，向每一位百忙中抽出时间审阅本论文的老师学者表示最诚挚的感谢。

赵晓森

2014年12月于北京

# 硕士期间发表论文

[1] 赵晓森，徐扬，王志谦。 一种基于NGOD架构的云应用服务管理方案[J]. 电视技术，2014,38(17):85-88.