# 摘 要

目前，全球移动通信市场已进入3G时代，各种基于3G网络的新业务层出不穷，而以手机电视、移动视频电话、视频短消息等为代表的移动视频业务无疑是其中最具发展前景的业务之一。移动视频业务是指通过移动网络和移动终端为移动用户提供视频内容的一种新型通信服务，它的主要特征在于传送的内容是比文本、话音更加高级的视频图像，并可伴有音频信息。3G网络技术和移动终端的不断发展，为移动视频业务的出现和推广提供了必要的条件，市场需求为移动视频业务的发展提供了广阔空间。但是，目前提供的仅仅是3G的一个承载网络，在3G网络之上的增值业务创建、部署和管理是一个存在许多未知点的技术领域。本文从3G业务最为核心的视频业务入手，尝试设计和实现一个视频业务创建和部署的平台，解决3G业务中视频业务的创建和部署中的一些关键技术。

本文分析了IP多媒体子系统IMS对视频业务的支持，并分析了IMS的三种业务提供方式，经过分析可以看出，Parlay X开放业务接口技术是未来第三方增值业务接口的核心技术。本文重点从视频业务实现的三个方面，分析和研究了Parlay X开放业务接口对视频业务实现的支持与不足。针对其不足，借鉴Parlay X开放业务接口思想，参考IMS中MRF（媒体资源功能）的设计，并参考JSR 309（Java媒体服务器控制API）规范，设计了移动网络视频业务的媒体处理方案，通过通用媒体处理服务开放接口来提供移动视频业务所需的媒体处理服务。在此基础上，进一步设计了移动视频业务创建和部署平台的整体框架，通过Parlay X接口来提供视频业务媒体信道的建立、维护和拆除，以及媒体流的传送和控制，通过通用媒体处理服务开放接口来提供通用媒体处理服务。通用媒体处理服务还提供了视频业务定制过程中的关键服务，例如视频显示、视频录制要求等。

结合移动视频业务创建和部署平台的的设计方案，本文重点实现了视频业务涉及的通用媒体处理服务的提取和封装，并提供了对通用视频服务的透明调用方案。

最后通过在视频业务测试平台上开发具体的视频业务案例，并对业务进行部署和测试，验证了论文所设计的移动视频业务创建和部署平台对视频业务开发的通用支持。

**关键词： 移动视频业务，IP多媒体子系统，Parlay X，JSR 309，万维网服务**

# Abstract

At present, the global mobile communications market has entered the 3G era. A variety of new services based on 3G networks emerge in endlessly. Represented by mobile TV, mobile video telephony, video message, etc., mobile video services is one of the most promising services. Mobile video services are referring to the new type of communications services which supplying mobile users with video content through mobile networks and mobile terminals .The main characteristic of mobile video services is that the delivered contents are not text or voice but video image that accompanied by audio information. The development of 3G network technology and mobile terminals provide Mobile video service's appearance and promotion with necessary conditions, the market demand also provide a broad development space. However, currently only offering a bearing network of 3G, but development, deployment and management of value-added services on 3G is still a developing technical field. This thesis tries to design and implement a video services development and deployment platform to resolve the technical issues on video services's development and deployment.

This thesis analyzes the support with video services from IP Multimedia Subsystem, and the three services provided modes of IMS. Being analyzed, it is showed that Parlay X technology is the key of the future third party value-added services interface. Focused on three aspects of the video services implementation, this thesis analyzes and researches the support to the implementation of video operation and shortage of Parlay X. In allusion of the shortage, this article designs the media services scheme of the mobile video services. Making reference of the ideology of Parlay X, the design of the MRF in IMS, and the criterion of JSR 309, we supply the media services needed by mobile video services from CMPS API. On this basis, this thesis takes a deep research of the whole framework of the mobile video services platform which can supplies the establishment, maintenance and backout of the video services media stream channel, the transmission and controlment of the medium stream by Parlay X interface, supplies the universal media manage services by common media services platform. The common media services platform also supplies the key services in the customizing of video services, like the layout of video frame, the demand of video REC.

By the design of the platform, this thesis implements the extraction and encapsulation of the common media services, and supplies the transparent invoke of the common media services.

At last, this thesis validates the design of the platform and the support of the platform to mobile video services by testing a specific example on the platform.

**Keywords: Mobile Video Services, IMS, Parlay X, JSR 309, Web Services**

# 南京邮电大学学位论文原创性声明

本人声明所呈交的学位论文是我个人在导师指导下进行的研究工作及取得的研究成果。尽我所知，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，也不包含为获得南京邮电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了谢意。

研究生签名：陈小敏 日期：2009.4.12

# 南京邮电大学学位论文使用授权声明

南京邮电大学、中国科学技术信息研究所、国家图书馆有权保留本人所送交学位论文的复印件和电子文档，可以采用影印、缩印或其它复制手段保存论文。本文电子文档的内容和纸质论文的内容相一致。除在保密期内的保密论文外，允许论文被查阅和借阅，可以公布（包括刊登）论文的全部或部分内容。论文的公布（包括刊登）授权南京邮电大学研究生部办理。

研究生签名：陈小敏 导师签名：边培莘 日期：2009.4.12

# 第一章 绪论

## 1.1 研究背景及意义

目前，全球移动通信市场已进入3G[1]（Third-Generation，第三代移动通信系统）时代，各种基于3G网络的新业务层出不穷，而以手机电视、移动视频电话、视频短消息等为代表的移动视频业务无疑是其中最具发展前景的业务之一[2]。

移动视频业务是指通过移动网络和移动终端为移动用户提供视频内容的一种新型通信服务，它的主要特征在于传送的内容是比文本、话音更加高级的视频图像，并可伴有音频信息。3G网络技术和移动终端的不断发展，为移动视频业务的出现和推广提供了必要的条件，市场需求为移动视频业务的发展提供了广阔空间。同时，移动视频业务也是3G市场发展的主要推动力量，是未来移动通信新的市场增长点，真正使移动用户享受无线视频掌上未来。但是，目前提供的仅仅是3G的一个承载网络，在3G网络之上的增值业务创建、部署和管理是一个存在许多未知点的技术领域。因此，移动网络中的视频业务创建和部署技术的研究具有理论意义和应用价值。

## 1.2 当前国内外的研究现状

NGN[3]（Next Generation Network，下一代网络）已经成为语音、视频通信的主流技术之一，3G标准的R5阶段也将是以NGN为核心网的业务平台，当前以视频业务为代表的下一代多媒体业务已经被公认为是电信业增值业务的发展方向。基于开放API是NGN中多媒体业务开发的趋势。开放API技术给NGN多媒体业务平台带来了强大的优势。开放API有两种不同的方案：（1）对网络能力进行的封装（如SIP协议接口[4]、Parlay接口[5]）；（2）对业务能力进行的封装（Parlay X接口[6]）。文献[7][8][9]研究了基于SIP和Parlay接口来生成、执行和管理多媒体业务的方案。文献[10][11]分析了Parlay X接口对媒体业务的支持，从分析可以看出，现有的开放接口标准对多媒体处理的支持有限，Parlay X2.1仅支持基于IVR或是基于会议的简单多媒体业务，Parlay3.0虽然在一定程度上扩展了2.1版本，提供了基于IVR和基于会议组合的业务，但仍不能提供NGN中多种多样的多媒体业务尤其是视频业务所需要的媒体处理服务。针对现有开放接口对媒体业务支持的缺陷，文献[10]进一步提出了一系列通用媒体处理能力组件，并通过不同组件的结合来开发多媒体业务，但是

其对通用媒体处理能力组件的定义及具体实现并没有深入探讨，只是做了理论性的研究。

媒体处理是指音频混音、视频合成、多媒体录放和在音/视频域与用户的交互等等，在具体实现时，媒体处理既可以由终端自己来实现，也可由媒体服务器来集中提供。媒体服务器是提供集约式标准媒体处理服务的网元，与终端自己执行媒体处理相比，媒体服务器提供的媒体处理能力更加稳定和高效，并且能够根据业务需求快速模块化地提供新的媒体处理能力[10]。在某些特定的多媒体业务中，如视频会议业务，则必须有专门的媒体服务器来提供视频会议所需的音频混音、视频合成等媒体处理能力。现有多个厂家致力于媒体服务器的提供，其中最具影响力的有 Convedia 媒体服务器[12]、HP OpenCall 媒体平台[13]、Cantata SnowShore [14]媒体服务器平台等等。因为媒体服务器本身并不具有业务逻辑，所以必须提供标准化的媒体控制接口，与软交换或者应用服务器等网元进行信令交互，完成业务逻辑所需的媒体处理功能。目前，媒体服务器的主要控制协议包括 MGCP[15]， H.248[16] 和 SIP[17]。其中 SIP 协议自身在控制媒体处理的时候是不足的，必须要经过适当的扩展，才可以满足实现增值业务所需要的控制消息和通知消息。目前主流的扩展是原 SnowShore 公司提出的NETANN[18] / MSCML[19] 以及原 Convedia 公司(现 Radisys 公司媒体服务器事业部)提出的 MSML[20] / MOML[21]。

由于媒体服务器的提供厂家繁多，媒体服务器控制协议也不尽相同，业务开发人员在开发涉及到媒体处理的增值业务时，必然涉及到多个媒体服务器控制协议的掌握，这使得IT应用开发人员在开发多媒体增值业务时必须投入大量的时间和精力到新知识的学习。针对此，BEA、惠普、富士通、爱立信、Oracle等组织在07年联合向JCP(Java Community Process)提交了JSR 309[22]（JMSC，Java Media Server Control API）规范请求，JMSC致力于为为多媒体业务开发者提供一个通用的媒体服务器能力抽象接口。BEA以及Oracle公司分别在07、08年推出了对此规范的参考实现。

## 1.3 论文的主要工作

本文的研究目标是移动网络中的增殖业务尤其是视频业务的开发问题。在对现有媒体业务开发技术研究和分析的基础上，针对移动网络的特点，着眼于未来业务开放、融合的趋势，提出一种新型的视频业务创建和部署平台。

本文的主要工作有：

- 研究移动网络视频业务开发的关键技术，分析现有机制对媒体业务开发的支持，重点分析了 IMS 网络架构中的业务提供方式，在此基础上研究了基于 Parlay X 的

多媒体业务开发。

- 媒体处理是视频业务开发的重点和难点，深入分析了现有媒体处理的相关标准，针对移动网络的特点，设计移动网络环境下的媒体处理实现方案。
- 借鉴开放业务接口的思想，设计移动网络环境下的视频业务创建和部署平台。深入研究 Parlay X，JSR 309，Web Services[23]（万维网服务）技术，通过 Parlay X 接口来提供视频业务需要的电信网络能力，参考 JSR 309 设计通用媒体处理服务，采用 Web Services 分布式计算环境，实现通用媒体处理服务。通用媒体处理服务提供了视频业务定制过程中的关键服务，例如视频显示、视频录制要求等。
- 通过业务实例的创建、部署，验证了本文提出的视频业务创建和部署平台对视频业务的通用支持。

## 1.4 论文组织结构

第二章，首先简单概述了移动视频业务及其涉及的关键技术标准。其次介绍了 IMS 体系架构，分析了 IMS 对移动视频业务的支持。进一步分析了 IMS 的业务提供方式，比较了三种业务提供方式的优缺点。

第三章，分析移动视频业务的具体实现，结合前面的介绍，分析 Parlay X 开放业务接口对视频业务的支持与不足，并根据不足设计解决方案，提出移动视频业务创建和部署平台的整体框架设计方案。

第四章，结合第三章的设计方案，重点实现视频业务涉及的通用媒体处理服务，并提供了对通用视频服务的透明调用方案，最后分析了基于本文设计的视频业务创建和部署平台的业务创建和部署方式，并分析了基于视频业务创建和部署平台业务创建和部署的特征。

第五章，通过在视频业务创建和部署平台上开发具体的视频业务案例，并对业务进行部署和测试，验证视频业务创建和部署平台对视频业务开发的通用支持。

第六章，总结全文所做工作，指出所存在的问题并提出下一步的主要工作。

# 第二章 移动视频业务相关分析和研究

## 2.1 移动视频业务概述

### 2.1.1 移动视频业务分类

移动视频业务是指通过移动网络和移动终端为移动用户提供视频内容的一种新型通信服务，它的主要特征在于传送的内容是比文本、话音更加高级的视频图像，并可伴有音频信息。移动视频业务种类繁多，对它们可采用多种方式划分子类，例如按照通信实体划分，按照网络承载方式划分和按照业务内容划分等等。尽管视频业务的应用种类繁多，但从视频通信业务本身所具有的多媒体信息(特别是视频信息)的实时性、交互性的特点归结起来典型的视频业务主要有两大类：流媒体业务和视频会议业务[2]。

**移动流媒体业务**

移动流媒体业务就是流媒体技术在移动网络和终端上的应用。所谓流媒体技术就是把连续的影像和声音信息经过压缩处理后放到网络服务器上，让移动终端用户能够一边下载一边观看、收听，而不需要等到整个多媒体文件下载完成就可以即时观看的技术。流媒体是从互联网上发展起来的一种多媒体技术，流媒体技术有三大特点：1、能够实时播放音视频和多媒体内容，即“边下载，边播放”；2、播放的流媒体文件不需要在客户端保存，降低对客户端存储空间的要求；3、不在客户端保存，简化了媒体文件的版权保护。

移动流媒体业务根据内容的播放方式可以分为：

(1)在线播放

终端播放器实时从流媒体服务器上获取流媒体数据，边下载边播放，流媒体内容不需存储在用户的终端设备。如果用户需要多次播放同一内容，每一次都需要从流媒体服务器上重新下载数据。对于在线播放的内容，由于是边下载边播放，播放效果很大程度上依赖于网络带宽，所以需要根据运营商网络的实际带宽状况，选择合适的压缩参数制作内容。

(2)下载播放

用户将流媒体内容下载并存储到本地终端中，然后可以选择在任意时间进行播放。对于下载播放，主要的限制指标是终端的处理能力和终端的存储能力，内容提供商可以制作出较高质量的视频内容(高带宽，高帧速率)，但需要考虑内容的下载时间及终端的存储空

间。

根据内容的来源划分移动流媒体业务则可以分为：

(1)流媒体点播(VOD)

内容提供商将预先录制好的多媒体内容编码压缩成相应格式，存放在内容服务器上并把内容的描述信息以及链接放置在流媒体的门户上。最终用户就可以通过访问门户，发现感兴趣的内容，有选择的进行播放。

(2)流媒体直播

流媒体编码服务器将实时信号编码压缩成相应的格并经由流媒体服务器分发到用户的终端播放器。根据实时内容信号源的不同，又可以分为电视直播、远程监控等。目前我国移动运营商主要是通过2.5G或2.75G网络传输技术来播放“手机电视”节目的。也就是说移动通过GPRS，联通通过CDMA 1X。

由于流媒体技术的三大特点，决定了移动流媒体业务的广阔应用前景。首先，流媒体技术有效降低对传输带宽和抖动的要求，使得在无线传输环境实现实时媒体播放业务成为可能。其次，移动终端体积小、低能耗的要求决定了其有限的存储空间，而媒体文件不需要在终端中保存，避免了对存储空间的较高要求。此外，有效的版权保护，能够确保移动流媒体应用的商用模式。

总之，把视频下载和转送类、移动视频类、移动音乐类等业务都包括在移动流媒体业务之中。其中移动视频下载和转送类主要传送实时性要求不高或容量较小的流媒体内容，如影视视频短篇、音乐短篇、体育视频片断、热点新闻等等，可以通过移动手机邮件附件和彩信等进行传输。移动视频类主要为时传时播、容量较大、无法或不允许被存储在手机上的节目内容，如手机电视、影视VOD、手机动画、移动可视电话、移动视频会议、移动音乐、实时的交通路况等随着第三代移动通信技术的逐步成熟，将移动流媒体技术引入移动数据增值业务，已经成为目前全球范围内移动业务研究的热点之一。目前3GPP、3GPP2、OMA等标准化组织都已经开展了移动流媒体业务研究工作，并已经制定了相应的标准。

**移动视频会议业务**

网络从诞生到今天经历了三个发展阶段:第一阶段称之为“线性网络”，以网络设备为核心，人们考虑的是让设备更快地传递数据；第二阶段是“平面网络”，即以网络控制技术为中心，目的是基于一个可控的网络，最终能够保障网络为不同要求的业务，尤其是多媒体业务，提供端到端有质量的服务；第三阶段被称之为“立体网络”，真正做到“融合通信”的平台。这是一种单数据网向多媒体网络的进步形式，是一个从量变到质变的飞跃。立体网络是一个以IP技术为核心的分组多媒体网络，是一个融合了语音、数据和视频的网络，这

种融合，必将推动视频会议系统的发展。

移动视频会议业务是指多点环境下的通信，可以在无线网络上提供实时视频、音频或数据等媒体格式的任意组合，利用无线网络在移动设备上实现互通，从而让移动用户之间能够随时随地进行实时音、视频等的交互。移动视频会议业务首先需要实现的是移动终端之间的互通，其次要实现移动终端与PSTN、ISDN以及IP网等各种网络终端设备之间的互通。

在未来可预见的一段时间内，视频会议系统的技术发展趋势主要呈现以下三大特点：

一、视频、语音网络将与数据完全融合，推动产品发展。这是一种单数据网向多媒体网络的进步形式，是一个从量变到质变的飞跃。立体网络是一个以IP技术为核心的分组多媒体网络，是一个融合了语音、数据和视频的网络，这种融合，必将推动视频会议系统的发展。

二、视频会议系统将与协同办公融为一体。随着视频通信应用的不断深入，不同行业的用户越来越感觉到功能单一的“会议电视”不能满足他们的需求。例如，政府应用在大力推广电子政务的前提下，越来越强调远程办公、远程查看、共享或修改办公文档和数据报表；中小型企业用户不希望视频通信只用于显示上层经理的形象，他们更希望能用于产品设计讨论、产品发布、员工培训……这些需求显然是以往“会议电视”所不能满足的。在不久的将来，包括视音频信息编码技术、多媒体技术、计算机技术、网络技术等在内的超媒体技术的发展，有望将这些需求变为现实。

三、视频会议业务向移动通信网络的延伸，移动视频会议业务将逐渐成熟。移动视频电话/会议是比目前语音、数据通信更高级的一种通信形式，实现了点对多点的通信业务，能极大地满足了人们个人通信和商务交流的需求，为商务用户带来更大的便利。目前，基于2.5G以及CDMA的移动通信应用已经启动，但受带宽的影响，应用还不是很充分。但随着3G即将到来，各国的3G移动运营商都在将移动视频电话业务作为3G业务拓展的先锋。

### 2.1.2 移动视频的关键技术及其标准

移动视频业务的涉及的关键技术有：

（1）视频编码标准。数字视频技术广泛应用于通信、计算机、广播电视等领域，带来了会议电视、可视电话及数字电视、媒体存储等一系列应用，促使了许多视频编码标准的产生。ITU-T 与 ISO/IEC 是制定视频编码标准的两大组织，ITU-T 的标准包括 H.261、H.263、H.264，主要应用于实时视频通信领域，如会议电视；MPEG 系列标准是由 ISO/IEC 制定的，主要应用于视频存储(DVD)、广播电视、因特网或无线网上的流媒体等。AVS 是

由我国自主制定的音/视频编码技术标准，主要面向高清晰度电视、高密度光存储媒体等应用。AVS 标准以当前国际上最先进的 MPEG-4 AVC/H.264 框架为基础，强调自主知识产权，同时充分考虑了实现的复杂度。

（2）音频编码标准。语音编码既可用软件也可用硬件的方法实现。软件实现就是将压缩算法用软件方法实现，这样做的好处是成本低、修改方便灵活，但处理速度较慢，不易保证处理的实时性；采用硬件实现就是将语音压缩算法固化到专用 DSP 芯片中，这样处理速度快，便于实时处理。国际上，对语音信号压缩编码的审议在 CCITT 下设的第十五研究组进行，相应的建议为 G 系列，多由 ITU 发表。到目前为止已发布的相关系列的标准有 G.721、G.722、G.723、G.728、G.729。另外在目前应用的标准中还包括 MPEG 系列(MPEG I，MPEG II，MPEG IV，MPEG VII)中的音频编码和 DOLBY 实验室推出的 AC 系列(AC-1，AC-2，AC-3)。

（3）流媒体网络传输技术。数字视频和声音传输所涉及到的一个重要概念就是“流媒体”概念。流媒体是指视频、声音和数据从源端同时向目的地传输，它可以作为连续实时流在目的地被接收。流数据从服务器端应用传输后可由客户端应用接收并显示或回放。流媒体的一个重要特征是对时间的敏感性，其实现主要取决于网络带宽和压缩算法的提高。流媒体的传输技术主要有三种：单播(unicast)、多播(Multicast)和广播(Broadcast)，多播又称为组播。点对点的特点是流媒体的源和目的地是一一对应的，即流媒体从一个源（服务器端的应用）发送出去后只能到达一个目的地（客户端应用）。组播是一种基于“组”的广播，其源和目的地是一对多的关系，但这种一对多的关系只能在同一个组内建立，也就是说，流媒体从一个源（服务器端的应用）发送出去后，任何一个已经加入了与源同一个组号的目的地（客户端应用）均可以接收到，但该组以外的其他目的地（客户端应用）均接收不到。广播的源和目的地也是一对多的关系，但这种一对多的关系并不局限于组，也就是说，流媒体从一个源（服务器端的应用）发送出去后，同一网段上的所有目的地（客户端应用）均可以接收到，广播可以看作组播的一个特例。

## 2.2 IMS 为移动视频业务提供技术支持

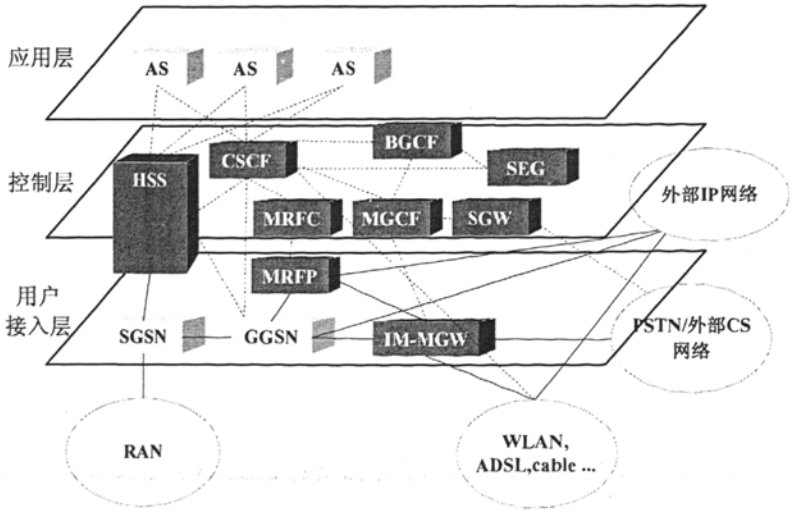
视频业务作为多媒体综合业务的典型，能为用户提供更直接、更全面的沟通交流，IMS[24][25][41]的出现为视频业务的发展和完善带来了新的推动力[2]。通过移动互联网接受的移动视频，需要运营商网络的支持，这就需要移动通信网络和有线网络的密切配合和交互，而有线网络和无线网络都在进行着IP化进程，同时用户需要的视频业务是个性化和多样化

的，这样就需要一个强大的多媒体体系，3GPP R5的分组域增加的IMS(IP Multimedia Subsystem, IP多媒体子系统)系统支持有线和无线网络的融合，支持语音、数据和多媒体业务，为最终用户提供多媒体化、个性化和多样化服务，能对视频流传输进行有效管理，所以IMS在移动视频业务中占有很重要的位置。

### 2.2.1 IMS 系统概述

IMS(IP Multimedia Subsystem, IP多媒体子系统)[24][25]是3G核心网络的重要组成部分，也是NGN核心网的发展方向之一。以IMS为核心的下一代网络已被认为是适应未来固定网与移动网融合(FMC，Fixed-Mobile Convergence)的网络架构。IMS是3GPP(3rd Generation Partnership Project，第三代合作伙伴计划)在版本5中提出的支持IP多媒体业务的子系统，IMS 定义了一个完整的体系结构和框架，允许在基于 IP 的基础设施上对声音、视频、数据和移动网络技术进行聚合。

IMS网络采取了分层架构进行体系设计，如图2-1所示，其网络分成三个主要层面：用户接入层面(User plane)、IMS核心控制层面 (Control plane)、应用层面(Application plane)。IMS分层的架构设计体现了“业务与控制分离”和“控制与接入或承载分离”的思想，使得不同的用户终端能够通过不同的无线或者有线接入技术接入IMS网络，享受统一的呼叫控制服务及其相应的增值业务。这种“层次化”的网络架构设计为不同网络的互联互通和业务的融合奠定了基础。




**CSCF：呼叫会话控制功能；S-CSCF：服务CSCF；I-CSCF：问询CSCF；P-CSCF：代理CSCF；HSS：归属用户服务器；MRFC：媒体资源功能控制；MRFP：媒体资源功能处理；BGCF：出口网关控制功能；MGCF：媒体网关控制功能；SGW：信令网关；IM-MGW：IP多媒体媒体网关控制器；**

图2-1 IMS网络架构

### 2.2.2 IMS 中的 MRF（媒体资源功能）

在 3GPP 的 IMS 架构里，媒体服务器被定义为 MRF[26]（Media Resource Function，媒体资源功能），并进一步细分为 MRFC（媒体资源控制器）和 MRFP（媒体资源处理器）。媒体资源控制器负责与 S-CSCF（呼叫会话控制服务器）交互，接收来自应用层业务对媒体处理的控制需求，他们之间的接口定义为 Mr。媒体资源控制器把收到的媒体处理控制需求进行处理，通过 Mp 接口调用媒体资源处理器进行媒体流的处理。Mp 现在使 H.248/MEGACO 协议。媒体资源处理器对外的 Mb 接口就是媒体流（RTP 流）。简单网络结构如下图所示：

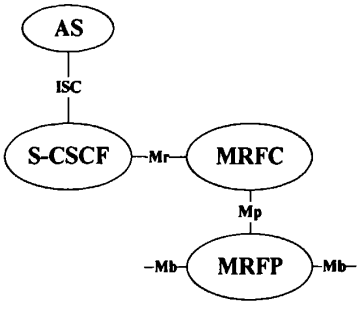



图 2-2 MRF 网络结构

### 2.2.3 IMS 业务提供方式

IMS 提供的连在 CSCF 上的三种业务平台，分别针对不同种类的业务提出的。

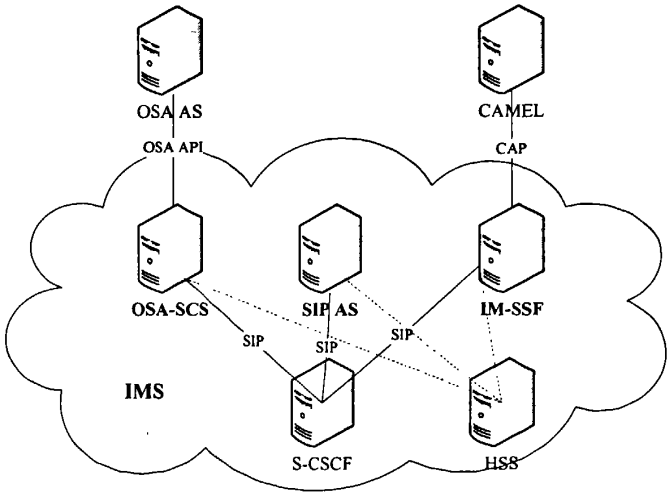



图 2-3 IMS 服务提供模型

1、基于 SIP 的业务应用

SIP 是由 IETF 提出的 IP 电话信令协议。主要目的是为了解决 IP 网中的信令控制，以

及与软交换节点的通信，从而构成下一代的增值业务平台，对电信、银行、金融等行业提供更好的增值业务。在 IMS 业务体系中，S-CSCF 相当于是软交换的呼叫控制平台，IMS 基于 SIP 的业务平台是通过专用的 SIP AS 实现的，可以实现 IM(即时通信)、点击呼叫、Web800、用户在线状态业务、会议电话、数字传真、多媒体消息、VOD、彩铃、彩名、可视电话等丰富的实时和非实时的多媒体业务。

2、基于 OSA（Open Service Architecture，开放业务系统）的业务应用

IMS 支持 PARLAY/OSA 标准业务接口，不仅可以完美地实现传统的基本电信业务，如：呼叫前转、呼叫等待、多方通话、主叫显示、主叫显示限制、呼叫限制等，而且可以为第三方业务供应商提供标准接口，提高运营商提供新业务的能力。

3、基于 CAMEL 的业务应用

CAMEL（Customised Applications for Mobile Network Enhanced Logic，移动网络增强逻辑的客户化应用）[27][28]业务是一种网络特性而不是补充业务，它采用智能网的原理，通过增加智能网的功能模块，即使当用户漫游出归属PLMN（Public Land Mobile Network，公众陆地移动电话网），网络运营商也可以为用户提供运营商特定的业务。为保护运营商已有的投资，IMS提供了对传统的智能网络的接口，可以通过SIP/INAP/CAMEL/WIN接口接入SCP（Service Control Point，业务控制点业务控制点），支持传统的智能业务，如预付费、虚拟专网、卡类业务等，目前移动网络中应用的预付费业务、VPN等业务都是采用这种方式实现的，从而实现了对已有智能业务的完美继承。

三种业务平台有利于快速推出各种层次的融合业务，最大程度地灵活支撑运营商未来网络的综合运营和业务推广，但三种业务平台在对业务的提供上各有特点[29]：(1) CAMEL 业务环境为使 IMS 用户仍继续享用传统智能网业务带来的便利，IMS 通过 CAP/ INAP 协议接入现有的 CAMEL/IN 业务,但无法为 IMS 用户提供更多的增值业务，且不支持对第三方业务开发；(2) SIP 会话控制机制为 SIP AS 和 CSCF 提供了对等的多次交互机制，具有可灵活触发、灵活嵌套的融合业务提供能力。但 SIP 存在着业务平台不够开放的问题，难以快速引入第三方 SP（Service Provider，服务提供商）或 CP（Content Provider 内容提供商）开发的业务；(3) Parlay/ OSA 是一种通用的技术标准，通过对底层网络的抽象，定义出一系列的 API，屏蔽了底层网络细节，方便了第三方业务的快速引入和部署。同时，其体系结构通用性较好，还可引入 Parlay X 标准来扩展 Parlay/ OSA 标准，提供一个更高抽象层次的基于 Web 服务器的技术，快速适应底层网络的变化，满足灵活多变的业务提供需求。

### 2.2.4 IMS 对视频业务的支持

IMS 网络的 QoS 为视频会话和媒体传输提供了质量保证；用户认证机制对进入 IMS 网络的用户进行身份验证，保证了视频业务的安全性；IMS 的业务触发和收费机制，减轻了业务开发负担；IMS 统一的业务平台丰富了视频业务的终端类型，兼容 SIP 协议的移动电话和计算机均可以享受基于 IMS 的视频业务；IMS 网络的媒体资源功能（MRF）为视频业务提供了集中高效的媒体处理能力。总之，IMS 为移动视频业务及其他业务应用提供了完善的网络平台，目前可以展望到的业务包括电话业务、多媒体电话会议、多媒体视频通信、统一消息、白板及应用共享业务等。很多标准化组织也参与 IMS 的完善发展。IETF[30]（Internet Engineering Task Force，互联网工程工作小组）为 IMS 提供了 SIP、SDP[31]等协议，并启动了 IMS 相关的工作。ITU（International Telecommunication Union，国际电信联盟）为 IMS 提供了 H.248、Q.1912、SIP 等协议。OMA(Open Mobile Alliance，开放移动联盟)定义了 IMS 架构下的业务，例如及时消息、移动对讲服务(PPT)等。ETSI（European Telecommunications Sdandards Institute，欧洲电信标准协会）同意在其下一代网络计划中采用 IMS。ATIS(Alliance for Telecommunication Industry Solutions，电信行业解决方案联盟)着重研究有线环境和无线环境中端到端的解决方案。

## 2.3 Parlay X 开放业务接口

### 2.3.1 Parlay/OSA 的分析

Parlay/OSA[5]是由Parlay、ETSI和3GPP组织制定、扩展、完善和标准化的，它致力于研究一种API标准使得企业可以访问使用电信网络的核心能力，并且该API标准应该独立于任何网络结构和网络技术，使得新应用的开发工作不再依赖于错综复杂的网络结构和网络技术。Parlay最本质的特征就是将网络能力抽象为接口的形式，这就允许业务开发商可以基于这些接口编写访问各种网络能力的业务。具体说来，这些接口分两类：框架接口(Framework Interfaces)和业务接口(Service Interfaces)。业务接口提供抽象的电信网业务能力，框架接口提供对业务能力接口的访问控制和计费管理。由于Parlay规范过于庞大和复杂，比较难以掌握， Parlay组织又推出了Parlay X[6]规范，基于Web Services技术对Parlay/OSA API进行了组合和封装。

### 2.3.2 Parlay X研究范围

如上所述，Parlay/OSA API用于向应用开发者开放底层电信网络的能力，它虽然对底层的网络细节进行了屏蔽，但是仍要求应用开发者具备电信背景知识并熟悉电信网络应用开发流程。这就限制了Parlay/OSA API的应用推广。为此，Parlay组织推出了Parlay X规范，其目的是为了促进不具备电信专业知识的IT开发人员开发下一代网络应用。

Parlay X是一组功能强大但简单、高度抽象的电信网络能力标准构件。无论开发人员是否具备电信专业知识都能够快速理解Parlay X，而且利用它开发出各具特色的应用。但是Parlay X不提供AAA(Authentication、Authorization、Accounting，认证、授权、记帐)、SLA(Service Level Agreement，服务等级协议)和其他与环境相关的功能。这些功能应当由Web Services框架来提供。

### 2.3.3 Parlay X提供的业务能力

Parlay X是Parlay API的更高层次的抽象并对其做了简化，2007年6月推出了最新版本3.0，抽象提供了20组网络业务能力[6]，分别是公共基础（Common）、第三方呼叫（Third Party Call）、呼叫通知（Call Notification）、短消息（Short Messaging）、多媒体消息（Multimedia Messaging）、付费（Payment）、账户管理（Account Management）、终端状态（Terminal Status）、终端位置（Terminal Location）、呼叫处理（Call Handling）、语音呼叫（Audio Call）、多媒体会议（Multimedia Conference）、地址列表管理（Address List Management）、在线状态（Presence）、消息广播（Message Broadcast）、地理编码和映射（Geocoding）、应用驱动QoS（Application-Driven Quality of Service）、设备能力和配置（Device Capabilities and Configuration）、多媒体流控制（Multimedia Streaming Control）、多媒体多播会话控制（Multimedia Multicast Session Management）。通过对上述网络业务能力的动态组合可以提供丰富的电信增值业务。

### 2.3.4 Parlay X应用体系架构

Parlay X的应用体系架构同Parlay API的应用架构类同，第三方应用服务器是Parlay X的客户端，利用Parlay X服务器也叫Parlay X网关（Parlay X Gateway）提供的业务能力向最终用户提供增值服务。Parlay X网关屏蔽下层电信网络、向上提供电信业务能力。Parlay X网关同各底层网络资源的接口目前还由网络运营商自己设定内部的通信协议提供，如采

用 JAIN、INAP、SIP 将 API 映射到低层网络，如图 2-4 所示:

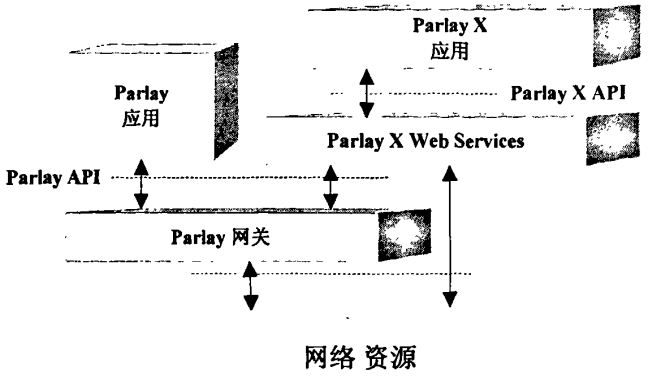



图 2-4 Parlay X 的应用体系架构

Parlay X 接口的优点主要有以下三点：

首先，在使用基于业务能力进行封装的 Parlay X 接口的时候，开发人员不需要关心网络的呼叫过程与资源管理过程，而只需要将关注点集中在具体的业务逻辑实现，因此 Parlay X 接口能够有效降低业务开发人员开发电信业务的技术门槛。

其次，承载 ParlayX 接口的 Web Service 技术是 Internet 行业被广泛应用的技术，出发点是实现 Internet 上分布的异构应用的共享，其客户和服务器之间是简单的“请求－响应”模式的面向服务交互过程，控制粒度在“服务”级别上，适合于普通的 IT 应用开发者。

最后，由于 ParlayX 是针对业务能力的封装，业务开发人员并不能直接访问到网络控制能力，因此具有很好的网络安全性，使运营商能够将这一接口开放给位于不可信任域的第三方增值业务开发与运营商使用。

由此可见，Parlay X 技术将是未来第三方增值业务接口的核心技术。

## 2.4 Web Services 技术介绍

Web Services[23]（万维网服务）技术是目前应用最为广泛的分布式组件技术，对企业业务集成和互联网上各种新业务的开发都有起到了巨大影响和推动，这种影响力也渗透到了移动电信领域，Parlay X API 就是下一代网络开放业务 API 的 Web Services 版本。Web Service 技术已经为业务融合奠定了良好的基础。未来移动业务的开放性、松散耦合性、可编程性的特点和万维网服务的特征非常相符。

### 2.4.1 Web Services 体系结构

Web Service 采用了面向服务( Service Oriented Architecture，SOA) 的体系结构，通过服务提供者、请求者和注册中心等实体之间的交互实现服务调用。交互具体涉及发布、发现和绑定操作[32]。这些角色和操作一起作用于万维网服务构件：万维网服务软件模块及其描述。在典型情况下，服务提供者提供可通过网络访问的软件模块（万维网服务的一个实现）。服务请求者使用服务的发现操作从服务注册中心查找服务描述，然后通过服务描述与服务提供者进行绑定，并调用相应的万维网服务实现，同它交互。图2-5展示了这些操作、角色以及它们之间的交互。

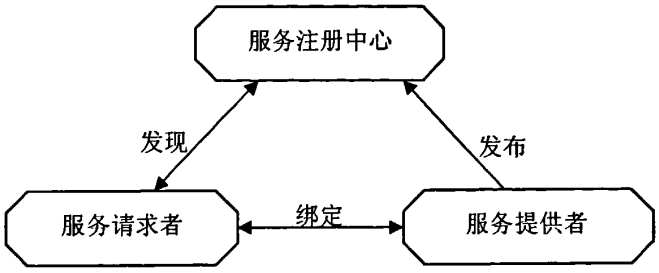



图 2-5　万维网服务体系架构模型

万维网服务体系结构中的角色包括：

1、服务提供者：从企业的角度看，这是服务的所有者。从体系结构的角度看，这是托管被访问服务的平台。

2、服务请求者：从企业的角度看，这是要求满足特定功能的企业。从体系结构的角度看，这是寻找并调用服务，或启动与服务的交互的应用程序。服务请求者角色可以由浏览器来担当，由人或无用户界面的程序（例如另外一个Web服务)来控制它。

3、服务注册中心：这是可搜索的服务描述注册中心，服务提供者在此发布他们的服务描述。在静态绑定开发或动态绑定执行期间，服务请求者查找服务并获得服务的绑定信息(在服务描述中)。对于静态绑定的服务请求者，服务注册中心是体系结构中的可选角色，因为服务提供者可以把描述直接发送给服务请求者。同样，服务请求者可以从服务注册中心以外的其它来源得到服务描述，例如本地文件、FTP站点、Web站点。

万维网服务体系结构中的具体操作包括：

1、发布：为了使服务可访问，需要发布服务描述以使服务请求者可以查找它。发布服务描述的位置可以根据应用程序的要求而变化。

2、发现：在发现操作中，服务请求者直接检索服务描述或在服务注册中心中查询所要求的服务类型。对于服务请求者，可能会在两个不同的生命周期阶段中牵涉到查找操作：在设计时，为了程序开发而检索服务的接口描述；而在运行时，为了调用而检索服务的绑定和位置描述。

3、绑定：最后需要调用服务。在绑定操作中，服务请求者使用服务描述中的绑定细节来定位、联系和调用服务，从而在运行时调用或启动与服务的交互。

### 2.4.2 Web Services 关键技术

Web Services涉及的最基本的技术规范包括XML，WSDL，SOAP和UDDI。WSDL是程序员描述Web Services的编程接口。Web Services可以通过UDDI来注册自己的特性，其他应用程序可以通过UDDI找到需要的Web服务。SOAP则提供了应用程序和Web 服务之间的通信手段。而WSDL，SOAP和UDDI 都建立在XML 基础之上[33]。

#### 2.4.2.1 XML

可扩展标记语言(eXtensible Markup Language，XML) 使用标记来界定内容，允许用户定义任意复杂度的结构，具有良好的扩展性；它具有自描述性，适合数据交换和共享；XML另一个优点是无关性，独立于具体的平台和厂商，确保了结构化数据的统一。目前XML已成为开放环境下描述数据信息的标准技术，也是Web Services中信息描述和交换的标准手段。

XML使用XML Schema 作为建模语言。它具有丰富的数据类型，支持类型继承，能对XML文件进行严格的合法性检查；使用与XML完全一致的文法，统一了分析和处理方式；引入了命名空间的概念，解决了可能的名称重复问题。XML Schema是Web Services中协议制订的标准语言，它和XML共同构成Web Services的基石。

#### 2.4.2.2 SOAP

XML使数据易于理解和共享，但应用实体之间要发送和接收XML文档，还需对网络协议、访问点等细节达成共识。Web Services实体间交互使用的是简单对象访问协议(Simple Object Access Protocol，SOAP) 。它提供了Internet 中交换结构化信息的轻量级机制，实现异构应用之间的互操作性。SOAP包括三个部分：封装结构、编码规则和RPC机制。封装结构定义了一个整体框架，描述消息内容、内容的属性以及谁负责处理。编码规则定义了交换应用程序数据的一系列机制，它支持XML Schema中全部的简单数据类型、以及结构和数组；RPC机制定义了远程过程调用和应答的协定，通过SOAP绑定，可以将SOAP信封在HTTP、SMTP等协议上进行传送。

#### 2.4.2.3 WSDL

在应用程序调用一个Web Service之前，必须知道其调用接口。Web Services具有松散耦合和自动集成的特点，要求接口描述方式能够被机器自动识别。Web Services采用WSDL (Web Service Description Language，万维网服务描述语言 )来描述其服务接口。

WSDL采用XML Schema定义，能够对各种语言实现的服务接口进行描述，具有语言无关性。WSDL将Web Service定义为网络端点的集合，使用类型、消息、端口等元素来描述服务接口。请求者据此可以知道服务要求的数据类型、消息结构、传输协议等，从而实现对Web Service的调用。

#### 2.4.2.4 UDDI

为了使服务申请者能够查找需要的服务，业界制订了注册和查找Web服务的UDDI技术规范。UDDI注册中心是对所有提供公共UDDI注册服务站点的统称，凡是实现UDDI规范的站点都可被称为UDDI 操作入口站点，站点之间通过复制机制保持彼此间的内容同步。服务提供者可以在服务注册中心发布自己提供的服务,服务请求者则在注册中心查找期望的服务。

## 2.5 本章小结

本章简单概述了移动视频业务及其涉及的关键技术标准。当前电信领域业务提供存在多种方案，业务开发技术也存在多种选择。重点介绍了 IMS 体系架构，分析了 IMS 对移动视频业务的支持。进一步分析了 IMS 的业务提供方式，比较了三种业务提供方式的优缺点。

# 第三章 移动视频业务创建和部署平台的设计

结合上一章的介绍，本章重点阐述移动视频业务创建和部署平台的详细设计。首先分析了 Parlay X 开放业务接口对视频业务的支持与不足，根据不足设计解决方案，并提出整个移动视频业务平台的框架设计。

## 3.1 设计需求分析

目前，全球移动通信市场已进入 3G 时代，各种基于 3G 网络的新业务层出不穷，而以手机电视、移动视频电话、视频短消息等为代表的移动视频业务无疑是其中最具发展前景的业务之一。3G 网络技术和移动终端的不断发展，为移动视频业务的出现和推广提供了必要的条件，市场需求为移动视频业务的发展提供了广阔空间。但是，目前提供的仅仅是 3G 的一个承载网络，在 3G 网络之上的增值业务创建、部署和管理是一个存在许多未知点的技术领域。

IMS(IP Multimedia Subsystem, IP 多媒体子系统)[24][25]是 3G 核心网络的重要组成部分，也是 NGN 核心网的发展方向之一。以 IMS 为核心的下一代网络已被认为是适应未来固定网与移动网融合(FMC，Fixed-Mobile Convergence)的网络架构。IMS 提供了连在 CSCF 上的三种业务平台，经过前两章的分析可以看出，Parlay X 开放业务接口技术是未来第三方增值业务接口的核心技术。本文基于 Parlay X 开放接口的设计思想，针对 Parlay X 对移动视频业务支持的不足，设计移动网络下的视频业务平台。

## 3.2 移动视频业务平台的设计思想

### 3.2.1 业务平台定义

对于业务平台的理解，并没有一个准确的定义，不过可以从它产生的目的开始探讨。平台的出现，就是要提高开发效率，减少人力成本，使得开发一个新的业务、产品，可以很容易。平台的诞生，是要解决两个问题。1、二进制级别的复用，在无需重新编译的情况下，重复利用已有的成果；2、业务模块级的重用，将能够解决用户核心业务问题的模块，封装成可以直接复用的平台基石，并可以在这些基石上重新构建完全客户化的新产品。

基于以上描述，本文提出的移动视频业务平台，旨在封装视频业务相关的通用服务，基于这些通用服务，开发 3G 相关的视频业务。

### 3.2.2 业务平台设计思想

移动视频业务平台的设计以 NGN[3]为理论背景。NGN 是一个多网融合的网络，为了在下一代多网融合的环境下研究开放的、分布的网络业务体系结构，如何跨越异构网络、方便快捷的生成业务是关键问题，也是难点问题。基于开放业务是 NGN 中业务生成的趋势[42][43][45]。基于开放业务的业务生成可以从三个层面来考虑：网络能力平面、业务开放平面、业务集成平面[44]，本文中移动视频业务平台的设计也从这三个层面来考虑。其中，网络能力平面属于异构网络这一层次，能够提供最基本的网络能力实体，不同的网络提供的能力不一样，这些能力通过其所在的网络协议来实现。网络能力平面不在本文的研究范围之内，下面分别从业务开放平面、业务集成平面来描述移动视频业务平台的设计思想。

**业务开放平面**

业务开放平面提供的是通过统一的开放标准开放出来的网络能力实体，开放后的网络能力实体是物理网络能力实体按照开放标准形式的抽象。此外，这一平面的实体还包括大量的业务提供商直接基于开放标准开发的业务。开放业务有两种不同的方案：（1）对网络能力进行的封装（如 SIP 协议接口[4]、Parlay 接口[5]）；（2）对业务能力进行的封装（如 Parlay X 接口[6]）。基于业务能力的封装有两个优点，首先，在使用基于业务能力封装的开放业务接口的时候，开发人员不需要关心网络的呼叫过程与资源管理过程，只需要将关注点集中在具体的业务逻辑实现，能够有效地降低业务开发人员开发电信业务的技术门槛；其次，基于业务能力的封装，使得业务开发人员不能直接访问到网络控制能力，因此具有很好的网络安全性，使运营商能够将这一接口开放给位于不可信任域的第三方增值业务开发商使用。本文在设计移动视频业务平台时，基于业务能力封装的思想，将底层的媒体服务器能力开放出来，对上层业务开发人员提供了通用的媒体处理能力。

**业务集成平面**

网络能力通过业务开放平面开放后，需要再转化成能够被集成的业务组件；通过集成业务开放平面的业务能力，能够生成更多更丰富的业务，集成使得各种业务组件能够最大程度被复用，也使得业务生成更方便快捷。在分布式环境下的业务集成，需要通过分布计算的相关技术来解决。较为代表性的分布计算技术有 RMI，CORBA，DCOM。然而对于跨网络的多业务承载平台上的业务集成问题，这些技术都难以满足分布式环境下的业务集成的通信异步性、松耦合性[44]。Web Services 是目前应用最为广泛的分布式计算技术，它

与一般的分布式计算技术的区别在于它更强调基于单个 Internet 标准(XML)来解决异构的分布式计算问题。Web Services 技术能支持面向服务的，松耦合的业务集成，并且通过统一的认证、动态工作流控制和交易控制的机制解决了 Internet 上各个业务平台的业务能力同步问题。本文在设计移动视频业务平台时，拟采用 Web Services 技术，对通用的媒体处理能力进行封装。

## 3.3 Parlay X 接口对移动视频业务的支持

### 3.3.1 多媒体业务实现

多媒体业务实现分为三方面：（1）媒体通道的建立、维护和拆除，在视音频通话中，就是呼叫的接续和中止；（2）媒体的传送，用户终端产生和获取媒体流，在交互式业务中媒体流是双向，在流媒体中是单向；（3）媒体的处理，会议中需要混音和分频，在某些场合需要进行媒体格式的转换，实现媒体流的存贮、转发，这些需要专门的设备进行处理，在视频会议系统中由MCU（Micro Controller Unit，多点控制器）实现，在软交换系统中可由MS（Media Server，媒体服务器）和MCU来实现，在流媒体业务中由流媒体服务器来实现。

### 3.3.2 基于 Parlay X 的多媒体业务开发

Parlay/OSA API 主要包括 2 类接口：框架接口和业务接口。Parlay X 属于业务接口，是对 Parlay/OSA API 的组合和封装，它高度抽象了 Parlay/OSA API 所提供的业务能力，并以简单的操作参数提供给业务开发者。Parlay X 最新版本 3.0 抽象提供了 20 组网络业务能力：

| 业务接口名称 | 接口功能 |
| --- | --- |
| 公共基础（Common） | 所有其他服务使用的公共基础设施和 XML 定义。 |
| 第三方呼叫（Third Party Call） | 由应用侧发起和控制的呼叫，业务开发者可以方便的通过此接口对呼叫双方发起呼叫控制。 |
| 呼叫通知（Call Notification） | 应用侧可以从网络侧获得呼叫过程中的呼叫状态信息。 |

| | |
|---|---|
| 短消息（Short Messaging） | 应用可以调用此接口通过网络发送短消息。 |
| 多媒体消息（Multimedia Messaging） | 应用可以调用此接口通过网络发送多媒体消息。 |
| 付费（Payment） | 提供预付费、后付费等付费应用接口。 |
| 账户管理（Account Management） | 支持账户查询，直接充值等用户账户管理功能。 |
| 终端状态（Terminal Status） | 终端状态信息，当终端状态发生变化时，可以通知应用终端状态的变化情况。 |
| 终端位置（Terminal Location） | 终端位置信息，当终端位置发生变化时，可以通知应用终端位置信息。 |
| 呼叫处理（Call Handling） | 预先设定呼叫处理规则，呼叫会按照规则进行处理。 |
| 语音呼叫（Audio Call） | 发起语音呼叫，可以播放的语音信息有文本、语音文件、VXML 文件。 |
| 多媒体会议（Multimedia Conference） | 对多媒体会议进行创建，并对相关媒体和会议成员进行管理。 |
| 地址列表管理（Address List Management） | 管理用户、用户组地址信息。 |
| 在线状态（Presence） | 提供用户的在线状态等信息。 |
| 消息广播（Message Broadcast） | 允许应用程序将消息向指定的用户终端广播。 |
| 地理编码和映射（Geocoding） | 将用户终端的坐标转换为地理位置，如可读的地址。 |
| 应用驱动 QoS（Application-Driven Quality of Service） | 受应用程序控制的服务质量。 |
| 设备能力和配置（Device Capabilities and Configuration） | 允许应用获得用户终端设备能力，以及配置终端设备能力。 |
| 多媒体流控制（Multimedia Streaming Control） | 支持多媒体流的请求、开始、重定向、暂定、终止、修改等功能，并对多媒体流的改变、终端状态的改变提供通知功能，允许终端用户在不同终端之间切换多媒体流。 |

| | |
|---|---|
| 多媒体多播会话控制（Multimedia Multicast Session Management） | 允许应用程序创建多媒体会议、动态管理参与者和管理所使用的媒体（audio、video 等），并提供了对会议参与者在线状态的获取和通知能力。 |

表格 3-1 Parlay X3.0 业务能力

根据业务属性，这些接口可以分为三大类，分别是：

（1）呼叫类业务接口：第三方呼叫、呼叫通知、呼叫处理、语音呼叫、多媒体会议、设备能力和配置、多媒体多播会话控制。

（2）数据业务类接口：短消息、多媒体消息、终端状态、终端位置、在线状态、消息广播、应用驱动 QoS、多媒体流控制。

（3）用户管理类接口：付费、账户管理、地址列表管理、地理编码和映射。

另外还有公共基础（Common）接口提供所有其他服务使用的公共基础设施和 XML 定义。

基于 Parlay X 的呼叫类业务接口，可以实现现有的电信呼叫业务，如电话业务、多媒体会议业务等；基于 Parlay X 的数据类业务接口，可以实现各类数据业务，如短消息、彩信、位置服务、天气预报、VOD（Video On Demand，视频点播业务）等；基于 Parlay X 的用户管理类接口，可以提供用户对个人帐户的管理功能，也可以与呼叫类或是数据类接口组合，完成业务中的计费等辅助功能。

### 3.3.3 Parlay X 接口对视频业务的支持与不足

多媒体业务的实现分为三方面，视频业务作为多媒体业务的典型，其实现也可从这三方面来考虑：

（1）媒体通道的建立、维护和拆除：这方面可由Parlay X的呼叫类业务接口来实现，呼叫类业务接口实际映射到底层的SIP协议，来实现第一方面的功能。如可采用多媒体会议（MC）和多媒体多播会话控制（MMSM）业务接口来实现视频会议业务，MC接口允许应用程序创建多媒体会议、动态管理参与者和管理所使用的媒体（audio、video等）。MMSM是3.0版本增加的业务接口，与MC类似，在其基础上增加了对会议参与者在线状态的获取和通知能力。另外，应用驱动QoS（ADQ）业务接口支持用户对媒体通道QoS的控制，如用户可以设置他每次连接到网络的默认QoS。

（2）媒体的传送：媒体传送中经常遇到两种不同类型的媒体，一种可以允许一定程度的丢包，而另一种则不允许，第一种类型的媒体有音频和视频信号，第二种类型的媒体有

网页和即时消息。在 IMS 架构中，通常是使用基于 UDP 的 RTP 完成媒体信息的非可靠传送，使用 TCP 和 SCTP（Stream Control Transmission Protocol，流控制传输协议）来完成媒体信息的可靠传送。Parlay2.1 中增加的应用驱动 QoS（ADQ）业务接口支持用户对媒体流传输质量的控制，可以根据用户业务的需要，修改媒体流传输的上下行带宽及其他 QoS 参数；Parlay3.0 版本中添加了多媒体流控制（MSC）业务接口，使得用户可以实现对媒体流的控制。MSC 接口支持媒体流的请求、开始、重定向、暂定、终止、修改等功能，并对媒体流的改变、终端状态的改变提供通知功能，允许终端用户在不同终端之间切换媒体流，如当用户在家中电视观看球赛时，如果此时有事需要出去，他可以将此时观看的媒体流迁移到移动设备上（与计费处理相关，此时用户已为观看的这段球赛付费），并继续观看。

（3）媒体的处理：媒体的处理，如视频会议中需要混音和分频，在某些场合需要进行媒体格式的转换，实现媒体流的存贮、转发，这些需要专门的媒体服务器进行处理。Parlay X 标准现有接口不支持媒体的处理功能。

基于以上的分析可知，Parlay X 接口为视频业务的开发提供了一定的支持，但是其没有提供媒体处理相关的业务接口，开发人员在开发视频业务时，需要自己来实现媒体的处理功能。

## 3.4 移动视频业务中媒体处理的研究与设计

本文所研究的移动网络视频业务创建和部署平台应该符合 IMS 的设计和标准，才有应用前景和实际意义。在研究移动视频业务中媒体处理的设计和实现之前，我们先来讨论下 IMS 网络架构中的 MRF（媒体资源功能）。

### 3.4.1 IMS 架构中的媒体处理

IMS 网络架构中媒体处理对应的模块为 MRF，MRF 代表了集约式提供媒体服务能力的网元，并进一步细分为媒体资源控制器（MRFC）和媒体资源处理器（MRFP）。媒体资源控制器负责与呼叫会话控制服务器（S-CSCF）交互，接收来自应用层业务对媒体处理的控制需求。媒体资源控制器把收到的媒体处理控制需求进行处理，通过调用媒体资源处理器进行媒体流的处理。可以总结出，IMS 网络架构定义中的媒体处理分为两个层面：媒体资源控制层和媒体资源处理层。

在具体实现时，媒体资源控制层（MRFC）可由现有的媒体服务器控制协议来提供。根据 IETF 下的 MediaCtrl（Media Server Control Working Group，媒体服务器控制工作组）

工作组制定的技术需求，媒体服务器控制协议需要实现的目标包括：

（1）媒体控制要求：

1、 协议应该支持一个或者多个应用服务器同时控制一个或者多个媒体服务器；

2、 协议应该使用可靠的底层传输协议；

3、 协议应该支持包括语音、音频信号、文本和视频在内的媒体类型；

4、 协议应该使用 SIP/SDP 来建立到媒体服务器的连接；

5、 应该支持跨媒体服务器的会议；

6、必须支持一个或者一组参与者从会议分离出来，而不需要跟媒体服务器建立新的连接，例如在会议中建立子会议；

7、 协议应该支持应用服务器查询媒体服务器的会话状态信息，信息报告应包括资源使用状况和网络质量等；

8、 媒体服务器应该支持上报应用服务器订阅的事件信息，如 STUN 请求，流量控制等；

（2）媒体混合/合成要求：

9、 应用服务器应该支持会议混音；

10、应用服务器应该支持用户自定义的视频输出窗口；

11、应用服务器应该支持视频流到指定输出窗口的映射或者告知媒体服务器视频流和输出窗口的映射关系

12、媒体服务器应该能把当前会议中的讲话者和视频源上报应用服务器，讲话者与视频源可能是不同的，如介绍一个播放的视频资料时；

13、媒体服务器应该支持告知应用服务器其所能支持的视频合成格式；

14、协议应该支持应用服务器控制媒体服务器对会议进行录音/像；

（3）IVR 要求：

15、应用服务器可以通过协议控制媒体服务器执行一个或者多个 IVR 脚本，并得到运行结果；脚本可以在服务器上也可以承载在控制信令上；

16、 应用服务器可以通过发送放音请求和接收应答（如 DTMF）来控制 IVR 会话过程；应用服务器根据收到的应答来决定 IVR 的流程；

17、应用服务器可以控制媒体服务器录制一小段媒体流然后回放；这不同于录音/像需求；

（4）操控要求：

18、应用服务器可以通过协议在媒体会话过程中改变媒体服务器的状态；

19、媒体服务器可以在媒体会话过程中上报自己的状态。

目前，媒体服务器的主要控制协议包括 MGCP，H.248 和 SIP。其中但是 SIP 协议自身在控制媒体处理的时候是不足的，必须要经过适当的扩展，才可以满足实现增值业务所需要的控制消息和通知消息。目前主流的扩展是原 SnowShore 公司提出的 NETANN/MSCML 以及原 Convedia 公司(现 Radisys 公司媒体服务器事业部)提出的 MSML/MOML。

而媒体资源处理层（MRFP）则由媒体服务器来提供。媒体服务器是指在 IP 网络上，可为软交换或者应用服务器等提供统一、集中媒体处理能力的服务器。与媒体网关不同，媒体服务器并不是接入层的设备，它属于核心网网元。软交换或者应用服务器执行业务逻辑，需要媒体处理能力的时候，通过 SIP 控制接口与媒体服务器进行交互，实现媒体服务器与媒体网关或者 VoIP 终端的媒体流连接（RTP 流）。媒体服务器根据业务控制的需求处理媒体流，并通过 SIP 控制接口返回必要信息。媒体服务器的核心是媒体处理，稳定高效的媒体处理能力是媒体服务器的基础，并能根据业务需求快速模块化提供新的媒体处理能力。因为媒体服务器本身并不具有业务逻辑，所以必须提供标准化的媒体控制接口，与软交换或者应用服务器等网元进行信令交互，完成业务逻辑所需的媒体处理功能。媒体处理是指音频混音、视频合成、多媒体录放和在音/视频域与用户的交互等。媒体服务器包含的媒体处理能力是通过操作控制 RTP 流来实现的。典型的操作包括多个 RTP 流的混合合成、一路 RTP 流的编解码转换、跨协议存储或者获取 RTP 流（如从 RTP 转换到 HTTP）、检测语音信号（如 DTMF）、文本转语音和自动语音识别等。当然媒体处理的范畴在不断的扩大，而且并没有一个确切的定义范围或者标准化定义，需要更深入的研究。现有多个厂家致力于媒体服务器的提供，其中最有影响力的有 Convedia 媒体服务器、HP OpenCall 媒体平台、Cantata SnowShore 媒体服务器平台等等。

可以得出，基于 IMS 架构中 MRF 定义的媒体处理实现如图 3-2 所示：

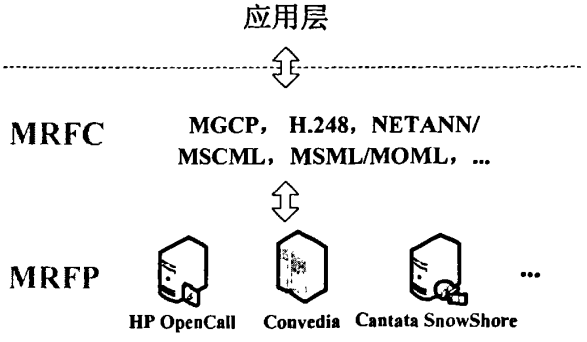



图 3-2 基于 MRF 的媒体处理实现

然而由于媒体服务器的提供厂家繁多，媒体服务器控制协议也不尽相同，业务开发人员在开发涉及到媒体处理的增值业务时，必然涉及到多个媒体服务器控制协议的掌握，这使得业务开发人员在开发多媒体增值业务时必须投入大量的时间和精力到新知识的学习。如何为业务开发提供一个统一的媒体服务器控制接口，使得业务开发人员在业务开发过程中不再涉及到多个不同的媒体服务器控制协议，而只需要调用通用的媒体服务器控制接口成了一个亟需解决的问题。JSR 309[22]规范正是基于此而提出。

### 3.4.2 JSR 309—Java 媒体服务器控制接口

JSR 309[22]（Java Media Server Control API，Java媒体服务器控制接口）是由多个组织（BEA、惠普、富士通、爱立信、Oracle等）联合向JCP(Java Community Process，Java 社区进程)提交的标准化技术规范请求，致力于为基于媒体服务器的电信业务提供媒体能力。JSR 309的主要目标是为多媒体业务开发者提供一个通用的媒体服务器能力抽象接口。在JSR 309的编程和对象模型中，媒体服务器控制独立于媒体服务器控制协议。它提供了多种媒体服务器能力，并提供了通用的应用函数集，如多方会议、多媒体混合、交互式会话等。JSR 309应用范围很广，它可以应用于一个简单的回铃音业务，也可以应用于一个复杂的会议业务。

JSR 309的目标是允许应用使用或组合典型的多媒体能力，它定义了三个主要的函数功能集MediaGroup、MediaMixer、NetworkConnection，如下图所示：

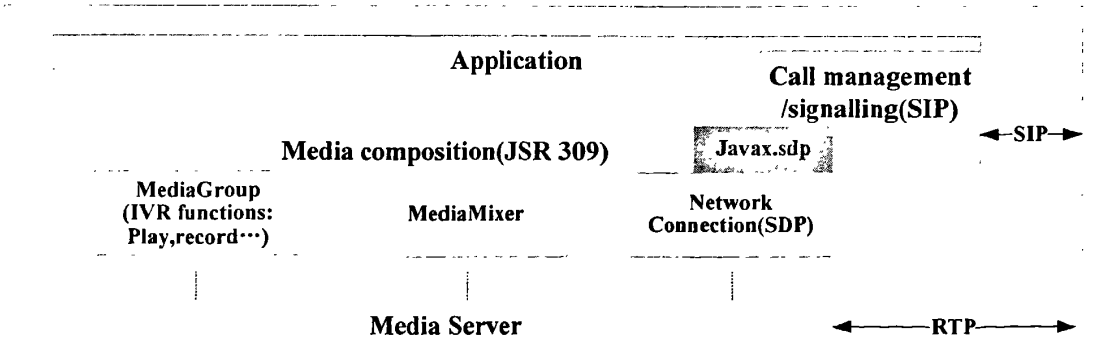



图3-3 JSR 309功能定义

并且定义了三种类型的媒体处理相关的功能对象：

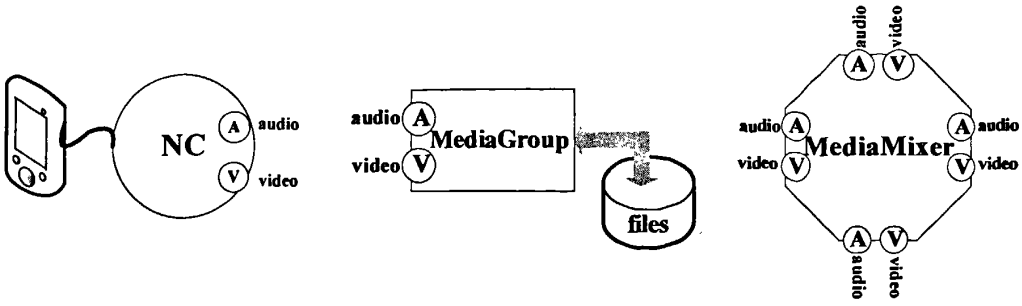



图3-4 JSR 309媒体处理功能对象

1、NetworkConnection（NC）对象，对应于一个终端用户，连接到电信媒体网络；

myMediaSession.createNetworkConnection(...);

2、MediaGroup对象，提供IVR相关功能，如播放提示，录制消息，监测DTMF信号等；

myMediaSession.createMediaGroup(...);

3、MediaMixer对象，有多个输入和输出，可以将多个媒体流混合成一个媒体流；

myMediaSession.createMediaMixer(...);

这些功能对象的不同组合可以提供多种不同的多媒体业务，如多媒体会议业务：

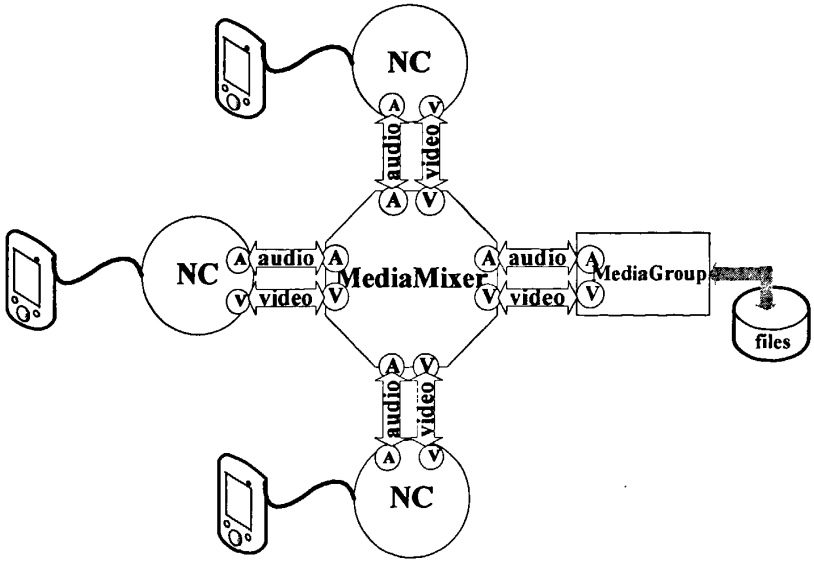



图3-5 基于JSR 309对象模型的多媒体会议业务

BEA、Oracle公司分别在07年、08年推出了对该规范的参考实现aspJMSC.jar（07）与mscontrol.jar（08）[34]。

### 3.4.3 移动视频业务的媒体处理方案设计

本文的研究目标是解决移动网络中的视频业务开发问题。移动视频业务通过移动网络和移动终端为移动用户提供视频内容。移动终端的不断发展，为移动视频业务的出现和推广提供了必要的条件。在设计视频业务创建和部署平台中的媒体处理方案时，基于IMS网络架构中MRF的设计思想，媒体处理也包含媒体资源控制和媒体资源处理两个层面，这使得本文设计的视频业务创建和部署平台可以无缝地应用到IMS网络架构中。并且进一步参照JSR 309规范、Parlay X技术，在媒体资源控制层面之上添加通用媒体服务器控制接口层面，通用媒体处理Web服务层面，通过Web Services技术对外开放通用媒体处理服务。媒体处理设计方案如图3-6所示：

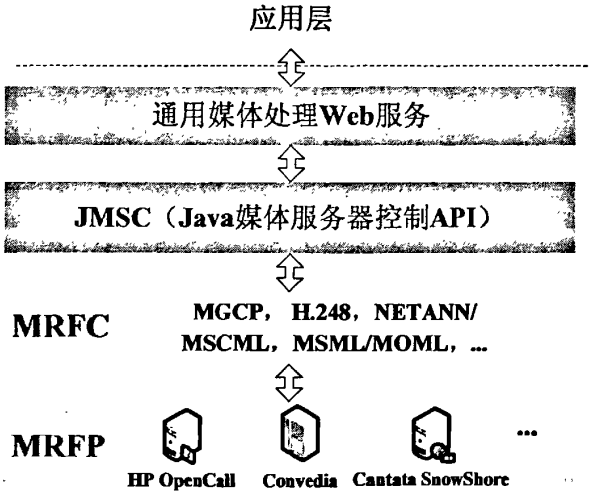



图3-6 移动视频业务的媒体处理方案

### 3.4.4 通用媒体处理能力的提取和封装

在上节设计的媒体处理方案中，在MRF设计之上添加了JMSC层、通用媒体处理Web服务层。通用媒体处理Web服务层，为上层应用提供了通用的媒体处理服务，从前面关于媒体服务器的描述中得知，典型的媒体操作包括多个RTP流的混合合成、一路RTP流的编解码转换、跨协议存储或者获取RTP流（如从RTP转换到HTTP）、检测语音信号（如DTMF），文本转语音和自动语音识别等等，但是媒体处理的范畴在不断的扩大，并没有一个确切的定义范围或者标准化定义，需要更多的跟进研究。所以，通用媒体处理服务并没有一个确切的定义参考。本文在设计通用媒体处理Web服务层时，在现有研究的基础上，参考JSR 309

规范以及NTT[1]网络通信系统实验室08年在IEICE[2]提出的MHE（Media Handling Enabler，媒体处理能力）通用平台，将媒体处理提取为五种通用能力。这些通用服务，构成了本文媒体处理方案中的通用媒体处理Web服务层，如下图：

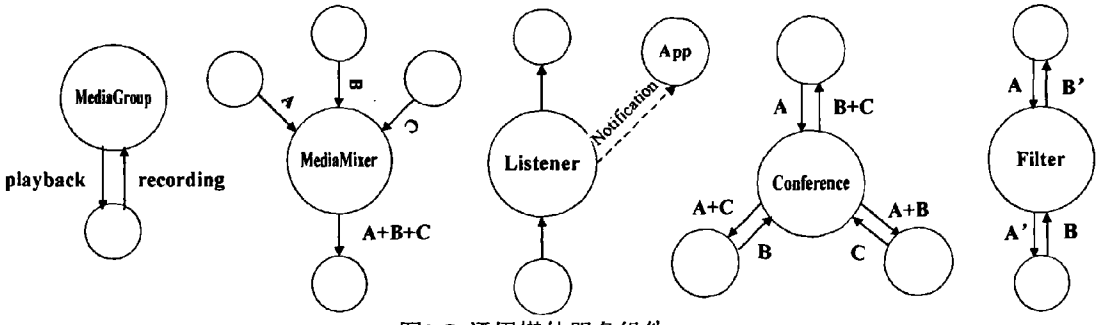



图3-7 通用媒体服务组件

1. MediaGroup 组件：提供媒体（音频，视频）文件的播放以及采集、存储功能；根据媒体文件 URL 播放媒体文件，并可以将接受的媒体流存储到媒体文件；
2. MediaMixer 组件：将接受的多个媒体流根据某种方案混合成一个媒体流；
3. Listener 组件：监听媒体流，并向上层应用通告接受到媒体流；
4. Conference 组件：提供多媒体会议所需的功能支持，混合多方媒体流（除了参会者本身）并发送给参会者，混合过程与 Mixer 组件类似；
5. Filter 组件：根据给定条件对媒体流提供过滤功能；

这些通用服务是对媒体服务器能力的高层抽象，对应用开发者屏蔽了底层的媒体服务器控制协议，使得业务开发者可以通过开放 API 来获取媒体服务器提供的媒体能力。不同的通用媒体服务能力可以由同一台媒体服务器提供，也可由不同的媒体服务器提供，通用服务处理服务开放接口根据用户的服务请求，映射到相应的服务器。

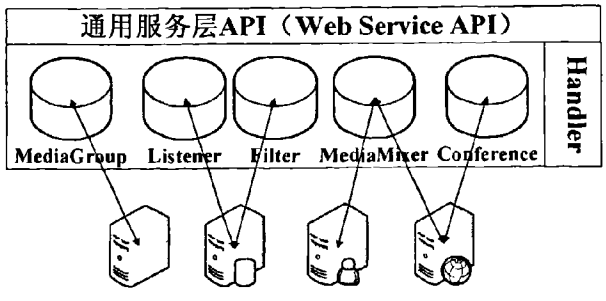



图 3-8 通用媒体处理服务体系架构

[1] NTT: 日本最大的移动通信运营商
[2] IEICE: The Institute of Electronics, Information and Communication Engineers，日本电子情报通信学会

## 3.5 移动视频业务平台整体框架设计

### 3.5.1 平台框架设计

前面，本文详细设计了移动视频业务中的媒体处理方案。在视频业务创建和部署平台的整体框架设计中，将这些视频相关的媒体处理通用能力定义为通用媒体处理服务开放接口（CMPS, Common Media Process Services）。通用媒体处理服务包括了媒体处理方案中的通用媒体处理Web服务层、通用媒体服务器控制接口层。其中，通用媒体处理Web服务层，对上层应用提供了基于Web Services技术封装通用媒体处理服务，通用媒体服务器控制接口层通过媒体服务器控制协议与底层的媒体服务器交互。并且，在通用媒体处理服务开放接口中提供了视频特有业务的定制功能，例如多方视频交互业务、视频快照、视频显示窗口等。

通用媒体处理服务提供了通用的媒体处理能力，而现实中的媒体服务器往往只负责某一具体的媒体服务能力，如视频服务器服务提供媒体文件资源，接受用户的点播请求，会议资源服务器负责视频会议涉及的相关处理，所以在我们的视频业务开发框架中引入了CMPS网关，网关提供了通用媒体处理能力的开放业务接口，应用通过这些接口可以访问到相应的媒体服务器，获得相应的媒体服务能力。在这种模式下，当有新的媒体服务器加入时，可以方便的在CMPS网关中发布自己的服务能力，供业务开发者使用。

构筑NGN多媒体业务平台需要应用服务器、Parlay网关、媒体服务器等关键设备。本文抽象了通用媒体服务能力，在媒体服务器之上添加了CMPS网关，参照NGN多媒体业务平台，本文提出的移动视频业务创建和部署平台的具体框架如下图：

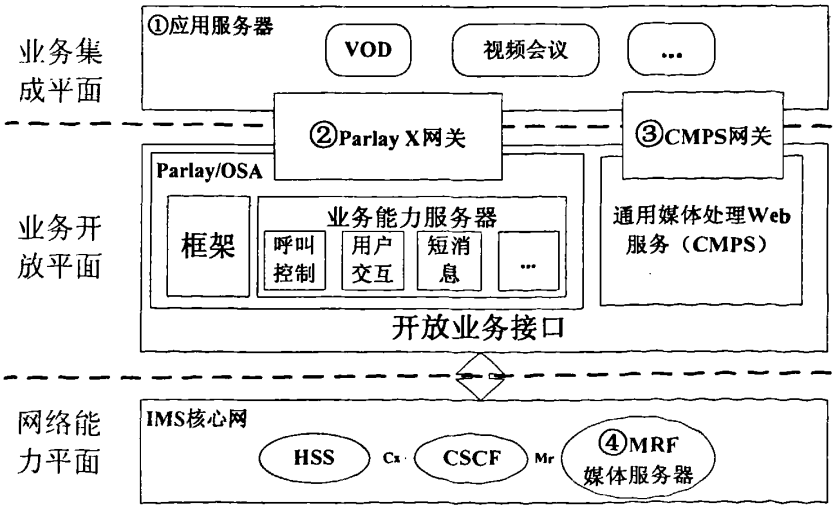



图 3-9 基于通用媒体服务的视频业务创建和部署平台

### 3.5.2 平台框架组成

移动视频业务创建和部署平台主要由四部分组成：

①应用服务器：即业务开发商开发和部署移动视频业务的平台，属于应用层，对应移动视频业务的应用逻辑。

②Parlay X网关：向应用服务器提供下层网络能力，业务开发者根据调用相关接口函数，获得底层网络能力，如呼叫控制，用户交互等。

③CMPS（通用媒体处理服务）网关：架构的核心模块，在通用媒体处理服务开放接口中，封装了视频相关的通用媒体处理服务。

④媒体服务器：如视频服务器，会议资源服务器等，提供相关的媒体服务能力的具体实现，它属于核心网网元。

## 3.6 平台设计方案的分析

### 3.6.1 业务平台分析

根据以上的设计分析可知，移动视频业务平台基于现有的 Parlay X 开放业务接口，并扩展了 Parlay X 接口，提出了通用媒体处理服务开放接口。通用媒体处理服务开放接口基于媒体服务器和现有的媒体服务器协议。

业务平台为业务开发人员提供了开发视频业务所需的开放业务接口，通过 Parlay X 接口来提供视频业务媒体信道的建立、维护和拆除，以及媒体流的传送和控制，通过通用媒体处理服务开放接口来提供通用媒体处理服务。

### 3.6.2 业务平台特点

移动视频业务创建和部署平台的设计基于开放业务的思想，遵循下一代网络的发展趋势，提出了基于通用媒体处理的移动视频业务开发模式，该业务平台具有以下特点：

（1）通过本文设计的视频业务平台开发移动视频业务的时候，开发人员不需要关心网络的呼叫过程与资源管理过程，只需要将关注点集中在具体的业务逻辑实现，通过调用 Parlay X 接口来获得网络呼叫和控制能力，通过调用通用媒体处理 Web 服务来获取媒体服务器提供的媒体处理能力。论文提出的新型的视频业务开发模式，能够有效地降低业务开发人员开发移动视频业务的技术门槛。

（2）视频业务平台采用的业务集成技术 Web Services 是 Internet 行业广泛应用的技术，基于 Web Services 技术，能够有效地实现异构网络的业务集成，实现 Internet 上分布的异构应用的共享。其客户和服务器之间是简单的“请求-响应”模式的面向服务交互过程，控制粒度在“服务”级别上，适合于普通的 IT 应用开发者。

（3）视频业务平台引入的 Parlay X 接口，设计的通用媒体处理服务接口，都是针对业务能力的封装，业务开发人员并不能直接访问到网络控制能力，以及底层的媒体服务器，因此具有很好的网络安全性，使运营商能够将视频业务平台提供的开放接口开放给位于不可信任域的第三方增值业务开发商使用。

## 3.7 本章小结

视频业务的实现可从三方面来考虑，媒体通道的建立、媒体的传送以及媒体的处理。Parlay X 接口提供了对前两方面的支持，但是对媒体处理没有提供相应的接口。本章分析了现有媒体处理相关研究，设计了移动视频业务开发中的媒体处理解决方案，并且参照 NGN 中的媒体业务平台，设计了移动视频业务创建和部署平台的整体框架。

下一章将在本章的设计基础上具体实现。

# 第四章 移动视频业务创建和部署平台的实现

上一章详细设计了移动视频业务中的媒体处理设计方案，并且设计了移动视频业务创建和部署平台的整体框架。本章在前面的设计基础上进行实现，重点实现视频业务创建和部署平台中的通用媒体处理服务开放接口。

## 4.1 通用媒体处理服务开放接口的技术实现架构

通用媒体处理服务开放接口在技术实现上分为三个层次，如图 4-1 所示。

媒体资源功能（MRF）层：IMS 架构里的媒体服务器，核心是媒体处理，提供稳定高效的媒体处理能力，并能根据业务需求快速模块化提供新的媒体处理能力。媒体处理是指音频混音、视频合成、多媒体录放和在音/视频域与用户的交互等。媒体服务器包含的媒体处理能力是通过操作控制 RTP 流来实现的。

媒体处理功能层：基于 JMSC 的 Java 接口进行万维网服务的开发，对上层提供了通用媒体处理能力 Web 服务，并对底层不同的媒体服务器提供了特定的驱动实现。

用户应用层：应用层使用万维网服务框架中提供的通用媒体处理服务，开发特定的多媒体视频应用。

通用媒体处理服务开放接口实现的技术架构逐层封装，屏蔽了底层媒体服务器平台的细节，在通用性和可扩展性方面有所突破。

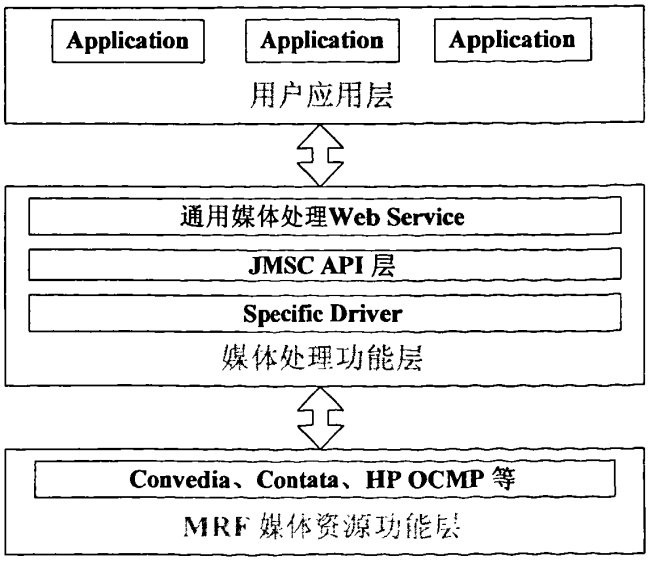



图 4-1 移动视频业务创建和部署框架的实现技术架构

## 4.2 通用媒体处理服务提取与封装基础

### 4.2.1 基于 JSR 309 开发通用媒体处理服务

关于JSR 309，在第二章已有详细介绍。JSR 309 实现了与媒体服务器的交互，对上层提供了通用媒体处理能力，本课题采用 Oracle 公司 2008 年 8 月发布的 JSR 309 参考实现 mscontrol.jar 作为基础，在其提供的 API 之上来开发通用媒体处理服务。Oracle 公司开发的 JMSC(mscontrol.jar)实现的媒体能力描述如下：

- 应用通过 MediaGroup 对象从媒体服务器获得一组媒体资源。MediaGroup 包含的操作接口有：record()，play(URI[])，receiveSignals()。
- 一个 NetworkConnection 对象代表了一个到终端的 RTP 连接。NetworkConnection 对象能够处理信令请求的媒体操作。
- 一个 MediaGroup 对象可以关联到一个 NetworkConnection 对象，这种情况下 MediaGroup 对象播放的媒体流可以被 NetworkConnection 对象连接到的远程终端收看，同时远程终端传过来的媒体流也可以被 MediaGroup 录制下来。
- 可以通过关联终端的 NetworkConnection 连接，把两个远程终端关联到一起。
- 通过 MediaMixer 对象，可以构建会议，或是其他多方应用。一个 MediaMixer 对象能够混合多路媒体流。

详情可以参考其主类的UML图：

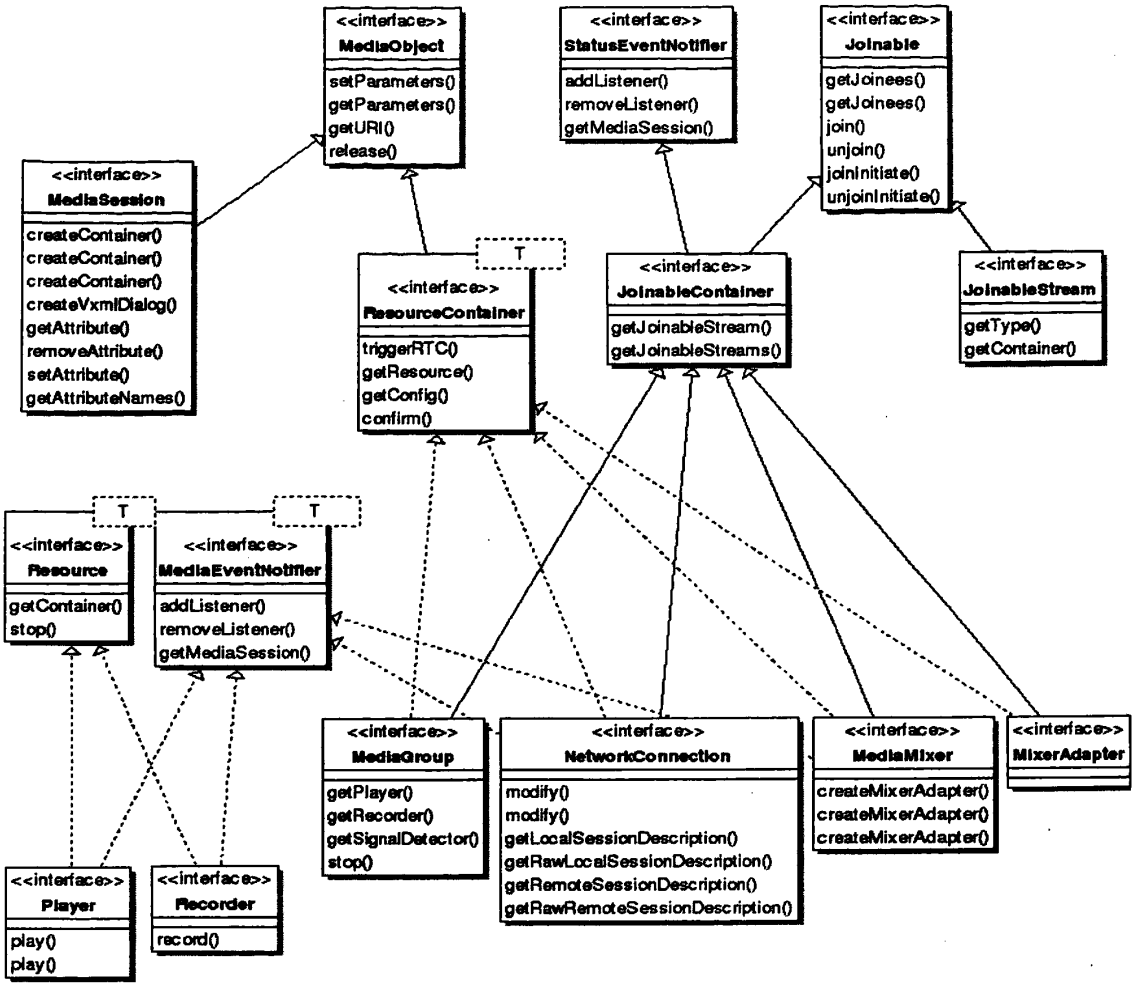



图 4-2 mscontrol（Media Server Controler API）主类 UML 图

JMSC 参考实现提供了与媒体服务器交互的接口，可以通过加载不同的驱动来连接不同的媒体器，代码如下：

```
// 根据配置文件，加载正确的JMSC媒体驱动
try {
    mediaResource = MediaResourceFactory.createMediaResource(
    config.driverType, // 媒体驱动类型，现有实现提供的驱动可为cantata/convedia/ocmp
    config.mediaServerAddress,
    Integer.parseInt(config.mediaServerPort),
    // 驱动HP OCMP媒体服务器时，需要添加 HTTP URL
    "http://" + config.myAddress + ":" + config.myPortHTTP + "/JMSCSampleCode");
} catch (JMSCException e)
{
    e.printStackTrace();
}
```

清单 4-3 加载媒体服务器驱动

### 4.2.2 用 WSDL 实现通用服务接口定义

关于通用服务接口采用何种方法定义，只有两个要求，没有特殊的限制。

1. 将所有的可供调用的服务都封装成统一的通用服务接口。
2. 通用服务接口需要能够描述服务的功能信息和服务的封装信息。

从另一个角度看，假设把这些底层服务都看作是万维网服务，即要求它们都额外地用 WSDL 接口描述，就达到了第一个要求。

WSDL 是以 XML 为基础的语言，用来定义和描述万维网服务以及如何访问万维网服务。由图 4-4 中 WSDL 的结构分析，可以看出整个 WSDL 文件由三个部分组成，分别表达了服务的“what”、“how”、“where”三个方面：

- 服务内容：包括接口名称(portType)，接口操作(operation)，输入和输出消息(input message、out put message)，输入和输出变量(types)。
- 绑定类型、传输协议：包括绑定(wsdl:binding)指向响应的接口名称、绑定方式(soap: binding，通过 Transport 定义传输协议、通过 Style 定义绑定类型)、每个接口操作的输入和输出消息的绑定类型(input message、output message)。一个 binding 定义了如何在一个抽象 PortType 与一个真实的服务格式和协议之间建立映射关系。例如，SOAP 绑定定义了编码风格、SOAPAction 头和 body（targetURI）的名称空间等等。
- 服务地址：包括 wsdl: port 指向绑定，address 指向位置服务。在现在的 WSDL 中，每一个 Port 有且只有一个 binding，而每一个 binding 有一个独一无二的 PortType。反过来（也是更重要的），每一个 Service（即 PortType）可能有多个 Port，而每一个 Port 则代表了一个作为替换的访问该服务的位置和绑定。

WSDL 的设计人员在一开始就考虑到了服务接口与实现的分离，WSDL 的服务内容部分是服务的抽象描述，仅描述了服务的操作、输入输出和类型。但仅有抽象部分是无法消费 Web Service 的，因此还需要将服务和具体的传输协议和访问位置关联，也就是后面两个部分。

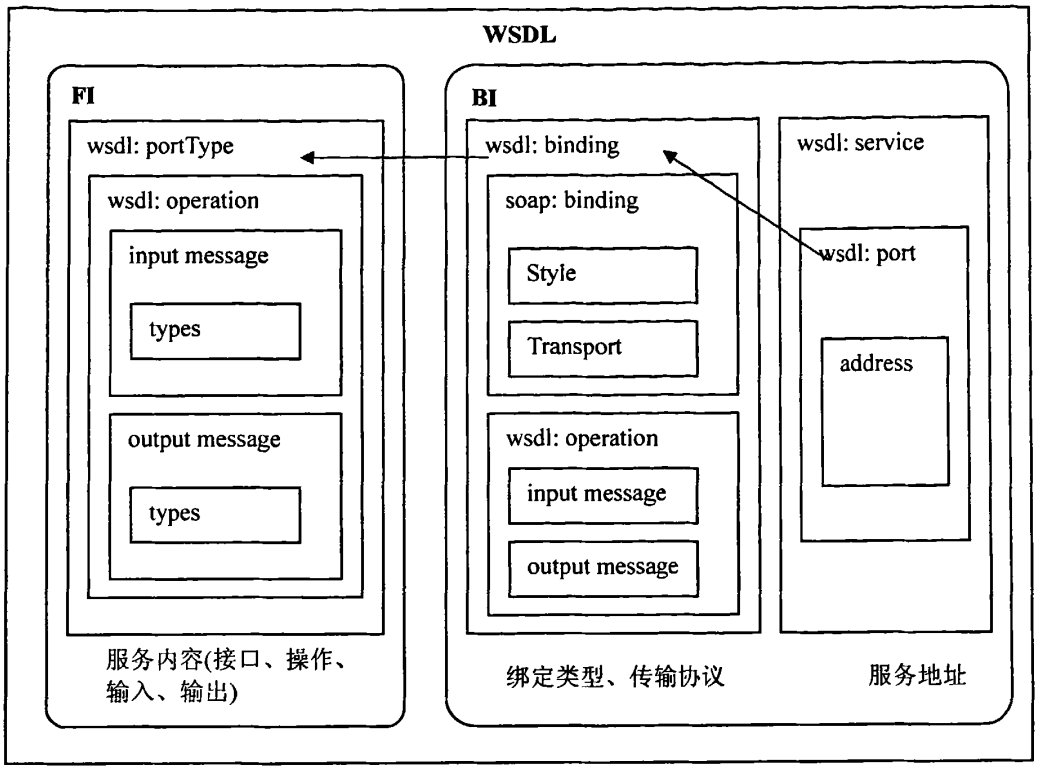



图 4-4 WSDL 结构分析

从图中可以看到 WSDL 文档的 portType 部分包含了通用服务接口所需要的 FI（Function Information）信息，而 WSDL 文档的 wsdl: binding 和 wsdl:service 部分则包含了通用服务接口的 BI（Binding Information）部分信息。这样 WSDL 也满足了实现服务通用接口的第二个要求。使用 WSDL 定义通用服务接口是可行的。

## 4.3 通用媒体处理服务提取与封装实现

在上一节的基础上，本节实现通用媒体服务的提取，并介绍如何把开发的抽象接口封装成 Web Service。媒体处理的具体实现由 JMSC API 提供，媒体处理通用接口的封装基于 WSDL 来实现。在这里仅介绍通用媒体处理能力的两个核心媒体处理能力的封装和实现，关于 Listener、Conference、Filter 的具体实现不再赘述。

### 4.3.1 MediaGroup

MediaGroup代表了一组简单的媒体处理资源，如播放器（Player）、录音（像）机（Recorder）、事件（信号）监测器等。实现时，MediaGroup作为一个web service，包括四个基本操作createMediaGroup、inviteNcToJoin、inviteMMToJoin、controlMediaGroup。应

用可以通过MediaGroup服务接口来获得相应的媒体处理能力：播放、录制、监测信号等。

#### 4.3.1.1 Operation：createMediaGroup

createMediaGroup操作根据媒体会话创建一个MediaGroup对象，createMediaGroup操作的具体实现是基于JMSC API的调用，具体实现的核心代码如下：

核心代码：

```
//根据 type 类型，创建相应类型的 MediaGroup 对象
if (type.equals("c_Player"))
    mediaGroup = mediaSession.createContainer(MediaGroupConfig.c_Player);
else if (type.equals("c_PlayerRecorderSignalDetector"))
    mediaGroup = mediaSession
    .createContainer(MediaGroupConfig. c_PlayerRecorderSignalDetector);
else if (type.equals("c_ PlayerSignalDetector"))
    mediaGroup = mediaSession
    .createContainer(MediaGroupConfig. c_ PlayerSignalDetector);
else if (type.equals("c_ SignalDetector"))
    mediaGroup = mediaSession
    .createContainer(MediaGroupConfig. c_ SignalDetector);
else return "false";
```

清单4-5 MediaGroup服务createMediaGroup操作接口内部实现核心代码

createMediaGroup操作有两个传入参数，一个是MediaSession参数，还有一个是类型参数。

输入消息: **createMediaGroupRequest**

| | 类型 | 选项 | 描述 |
|---|---|---|---|
| mediaSession | xsd: string | 必填 | 多媒体会话参数 |
| mediaGroupType | xsd: string | 必填 | 创建的MediaGroup类型，可选值为：c_Player，c_PlayerRecorderSignalDetector，c_PlayerSignalDetector，c_SignalDetector |

输出消息: **createMediaGroupResponse**

| | 类型 | 选项 | Description |
|---|---|---|---|
| mediaGroupIndentifier | xsd:string | 不可选 | MediaGroup描述符 |

#### 4.3.1.2 操作：inviteNcToJoin

应用调用操作inviteToJoin将NetworkConnection对象关联到创建的MediaGroup，一个NetworkConnection对象代表了一个到终端的会话连接，通过关联，使得终端拥有了MediaGroup对应的媒体能力。操作的具体实现是基于JMSC API的调用，具体实现的核心代

码如下：

核心代码：

```
//根据 streamType 参数，设置 join 操作的媒体流参数，如 audio、video 等
if (streamType.equals("double")) {
//根据 type 参数，设置 join 操作的属性参数，如 DUPLEX、RECV、SEND，对应媒体信道的单/双工属性
    if (type.equals("DUPLEX")) {
        mediaGroup.join(Joinable.Direction.DUPLEX, networkConnection);
    } else if (type.equals("RECV")) {
        mediaGroup.join(Joinable.Direction.RECV, networkConnection);
    } else if (type.equals("SEND")) {
        mediaGroup.join(Joinable.Direction.SEND, networkConnection);
    } else return "false";
} else if (streamType.equals("audio")) {
            if (type.equals("DUPLEX")) {
                mediaGroup.getJoinableStream(StreamType.audio).join(
                        Joinable.Direction.DUPLEX, networkConnection);
            } else if (type.equals("RECV")) {
                mediaGroup.getJoinableStream(StreamType.audio).join(
                        Joinable.Direction.RECV, networkConnection);
            } else if (type.equals("SEND")) {
                mediaGroup.getJoinableStream(StreamType.audio).join(
                        Joinable.Direction.SEND, networkConnection);
            } else
                return "false";
        } else if (streamType.equals("video")) {
            if (type.equals("DUPLEX")) {
                mediaGroup.getJoinableStream(StreamType.video).join(
                        Joinable.Direction.DUPLEX, networkConnection);
            } else if (type.equals("RECV")) {
                mediaGroup.getJoinableStream(StreamType.video).join(
                        Joinable.Direction.RECV, networkConnection);
            } else if (type.equals("SEND")) {
                mediaGroup.getJoinableStream(StreamType.video).join(
                        Joinable.Direction.SEND, networkConnection);
            } else
                return "false";
        } else if (streamType.equals("message")) {
            if (type.equals("DUPLEX")) {
                mediaGroup.getJoinableStream(StreamType.message).join(
                        Joinable.Direction.DUPLEX, networkConnection);
            } else if (type.equals("RECV")) {
```

```
                    mediaGroup.getJoinableStream(StreamType.message).join(
                            Joinable.Direction.RECV, networkConnection);
                } else if (type.equals("SEND")) {
                    mediaGroup.getJoinableStream(StreamType.message).join(
                            Joinable.Direction.SEND, networkConnection);
                } else
                    return "false";
            } else
return "false";
```

清单4-6 MediaGroup服务inviteNcToJoin操作接口内部实现核心代码

inviteNcToJoin操作有四个传入参数，MediaGroup参数，NetworkConnection参数，Steam参数，以及Type参数。

输入消息: **inviteNcToJoinRequest**

| | 类型 | 选项 | 描述 |
|---|---|---|---|
| mediaGroupIdentifier | xsd: string | 必填 | MediaGroup参数 |
| networkConnectionIdentifier | xsd: string | 必填 | NetworkConnection参数 |
| streamType | xsd: string | 必填 | 媒体流类型参数，可以为video（视频流），audio（音频流），message（消息），或是double（视音频流） |
| type | xsd: string | 必填 | 媒体信道参数，可以为RECV（单工接受），SEND（单工发送），DUPLEX（双工） |

输出消息: **inviteNcToJoinResponse**

| | 类型 | 选项 | 描述 |
|---|---|---|---|
| status | xsd:string | No | MediaGroup状态符 |

相应的，也可将 MediaMixer 对象关联到创建的 MediaGroup，对应的操作为 inviteMMToJoin，inviteMMToJoin 操作与 inviteNcToJoin 类似，这里不再赘述。

#### 4.3.1.3 Operation：controlMediaGroup

应用调用操作controlMediaGroup去控制MediaGroup对应的媒体能力，操作具体实现的核心代码如下：

核心代码：

```
//根据 action 参数执行相应的媒体操作，mediaStreamURI 是代处理的媒体资源地址
if (action.equals("play")) {
    mediaGroup.getPlayer().play(mediaStreamURI, null,
            Parameters.NO_PARAMETER);
```

```
    return "true";
} else if (action.equals("record")) {
    URI uri = mediaStreamURI[0];
    mediaGroup.getRecorder().record(uri, null, Parameters.NO_PARAMETER);
    return "true";
} else return "false";
```

清单4-7 MediaGroup服务controlMediaGroup操作接口内部实现核心代码

它有三个传入参数， MediaGroup参数，action参数（对应相应的媒体能力），还有一个是类型参数。

输入消息: **controlMediaGroupRequest**

| | 类型 | 选项 | 描述 |
|---|---|---|---|
| mediaGroupIdentifier | xsd: string | 必填 | MediaGroup参数 |
| action | xsd: string | 必填 | 媒体处理参数，可为play、record、receiveSignals，分别对应不同的媒体处理 |
| mediaStreamURI | xsd:anyURI[0..unbounded] | 选填 | 对应数据URI |

输出消息: **controlMediaGroupResponse**

| | 类型 | 选项 | 描述 |
|---|---|---|---|
| status | xsd:string | No | MediaGroup状态符 |

#### 4.3.1.4 MediaGroup 服务对应的 WSDL 服务端口

```
<wsdl:service name="MediaGroupService">
    <wsdl:port binding="impl:MediaGroupSoapBinding" name="MediaGroup">
        <wsdlsoap:address location="http://localhost:8080/jsr-ws/services/MediaGroup"
    </wsdl:port>
</wsdl:service>
```

清单 4-8 MediaGroup 服务 WSDL 文件中的服务端口

### 4.3.2 MediaMixer

MediaMixer能够将多路媒体流混合成一个媒体流，是实现多媒体会议或是其他多方应用的基础。实现时，Mixer作为一个web service，包括三个基本操作createMediaMixer、inviteNcToJoin、inviteMGToJoin，以及setVideoLayout。应用可以通过MediaMixer服务接口来实现多媒体会议或其他多方应用中的媒体流混合。

#### 4.3.2.1 Operation：createMediaMixer

应用调用操作createMediaMixer创建一个MediaMixer对象，操作的具体实现同样是基于JMSC API的调用，具体实现的核心代码如下：

核心代码：

```
//创建 MediaMixer 对象
mediaMixer = mediaSession.createContainer(MediaMixer.class, Parameters.NO_PARAMETER);
```

清单4-9 MediaMixer服务createMediaMixer操作接口内部实现核心代码

操作createMediaMixer具有一个传入参数， MediaSession参数。

输入消息: **createMediaGroupRequest**

| | 类型 | 选项 | 描述 |
|---|---|---|---|
| mediaSession | xsd: string | 必填 | 多媒体会话参数 |

输出消息: **createMediaGroupResponse**

| | 类型 | 选项 | Description |
|---|---|---|---|
| mediaMixerIndentifier | xsd:string | 不可选 | MediaGroup描述符 |

#### 4.3.2.2 Operation：inviteNcToJoin

应用调用操作inviteToJoin将NetworkConnection对象关联到创建的MediaMixer对象，一个NetworkConnection对象代表了一个到终端的会话连接，通过关联，将终端信号作为MediaMixer的输入媒体流，MediaMixer将多个输入媒体流混合成一个媒体流。MediaMixer服务inviteNcToJoin操作的实现代码与MediaGroup服务的同名操作inviteNcToJoin的实现代码是类似的：

核心代码：

```
//根据 streamType 参数，设置 join 操作的媒体流参数，如 audio、video 等
if (streamType.equals("double")) {
//根据 type 参数，设置 join 操作的属性参数，如 DUPLEX、RECV、SEND，对应媒体信道的单/双工属性
    if (type.equals("DUPLEX")) {
        mediaMixer.join(Joinable.Direction.DUPLEX, networkConnection);
    } else if (type.equals("RECV")) {
        mediaMixer.join(Joinable.Direction.RECV, networkConnection);
    } else if (type.equals("SEND")) {
        mediaMixer.join(Joinable.Direction.SEND, networkConnection);
    } else return "false";
} else if (streamType.equals("audio")) {
```

```
            if (type.equals("DUPLEX")) {
                mediaMixer.getJoinableStream(StreamType.audio).join(
                        Joinable.Direction.DUPLEX, networkConnection);
            } else if (type.equals("RECV")) {
                mediaMixer.getJoinableStream(StreamType.audio).join(
                        Joinable.Direction.RECV, networkConnection);
            } else if (type.equals("SEND")) {
                mediaMixer.getJoinableStream(StreamType.audio).join(
                        Joinable.Direction.SEND, networkConnection);
            } else
                return "false";
        } else if (streamType.equals("video")) {
            if (type.equals("DUPLEX")) {
                mediaMixer.getJoinableStream(StreamType.video).join(
                        Joinable.Direction.DUPLEX, networkConnection);
            } else if (type.equals("RECV")) {
                mediaMixer.getJoinableStream(StreamType.video).join(
                        Joinable.Direction.RECV, networkConnection);
            } else if (type.equals("SEND")) {
                mediaMixer.getJoinableStream(StreamType.video).join(
                        Joinable.Direction.SEND, networkConnection);
            } else
                return "false";
        } else if (streamType.equals("message")) {
            if (type.equals("DUPLEX")) {
                mediaMixer.getJoinableStream(StreamType.message).join(
                        Joinable.Direction.DUPLEX, networkConnection);
            } else if (type.equals("RECV")) {
                mediaMixer.getJoinableStream(StreamType.message).join(
                        Joinable.Direction.RECV, networkConnection);
            } else if (type.equals("SEND")) {
                mediaMixer.getJoinableStream(StreamType.message).join(
                        Joinable.Direction.SEND, networkConnection);
            } else
                return "false";
        } else
            return "false";
```

清单4-10 MediaMixer服务inviteNCToJoin操作接口内部实现核心代码

它有四个传入参数，MediaMixer参数，NetworkConnection参数，Steam参数，以及Type参数。

输入消息: **inviteNcToJoinRequest**

| | 类型 | 选项 | 描述 |
|---|---|---|---|
| mediaMixerIdentifier | xsd: string | 必填 | MediaGroup参数 |
| networkConnectionIdentifier | xsd: string | 必填 | NetworkConnection参数 |
| streamType | xsd: string | 必填 | 媒体流类型参数，可以为video（视频流），audio（音频流），message（消息），或是double（视音频流） |
| type | xsd: string | 必填 | 媒体信道参数，可以为RECV（单工接受），SEND（单工发送），DUPLEX（双工） |

输出消息: **inviteNcToJoinResponse**

| | 类型 | 选项 | 描述 |
|---|---|---|---|
| status | xsd:string | No | MediaGroup状态符 |

相应的，也可将 MediaGroup 对象关联到创建的 MediaMixer，对应的操作为 inviteMGToJoin，inviteMGToJoin 操作与 inviteNcToJoin 类似，这里不再赘述。

#### 4.3.2.3 Operation：setVideoLayout

应用调用操作setVideoLayout设置连接到MediaMixer的多路媒体流中视频流的版面布局，在JMSC标准实现中，video layout的设置是基于SMIL（Synchronized Multimedia Integration Language，同步多媒体集成语言）的，所以有必要先介绍一下SMIL标记语言。

**SMIL**

SMIL 是同步多媒体集成语言（Synchronized Multimedia Integration Language）的缩写，念做 smile。它是由 3W(World Wide Web Consortium)组织规定的，它已开始成为将多媒体集成到 Web 内容的重要新方法。SMIL 提供基于 XML 的方法来控制多媒体。SMIL 文件包含了描述多媒体程序所需的所有信息。存储 SMIL 文件的扩展名是 *.smil。SMIL 文件包含：

- 呈现的布局 (the layout of the presentation )
- 呈现的时间线 (The timeline of the presentation)
- 多媒体元素的源 (The source of the multimedia elements )

由于 SMIL 基于 XML，因此其标签对大小写敏感。所有的 SMIL 标记都必须是小写字母。SMIL 文档必须以 <smil> 标签开始，并以 </smil> 标签结束。它可包含一个 <head> 元素，且必须包含一个 <body> 元素。<head> 元素用于存储有关呈现布局的信息，以及其他的元信息。<body> 包含媒介元素。下面是一个基于 SMIL 来设置视频版面布局例子：

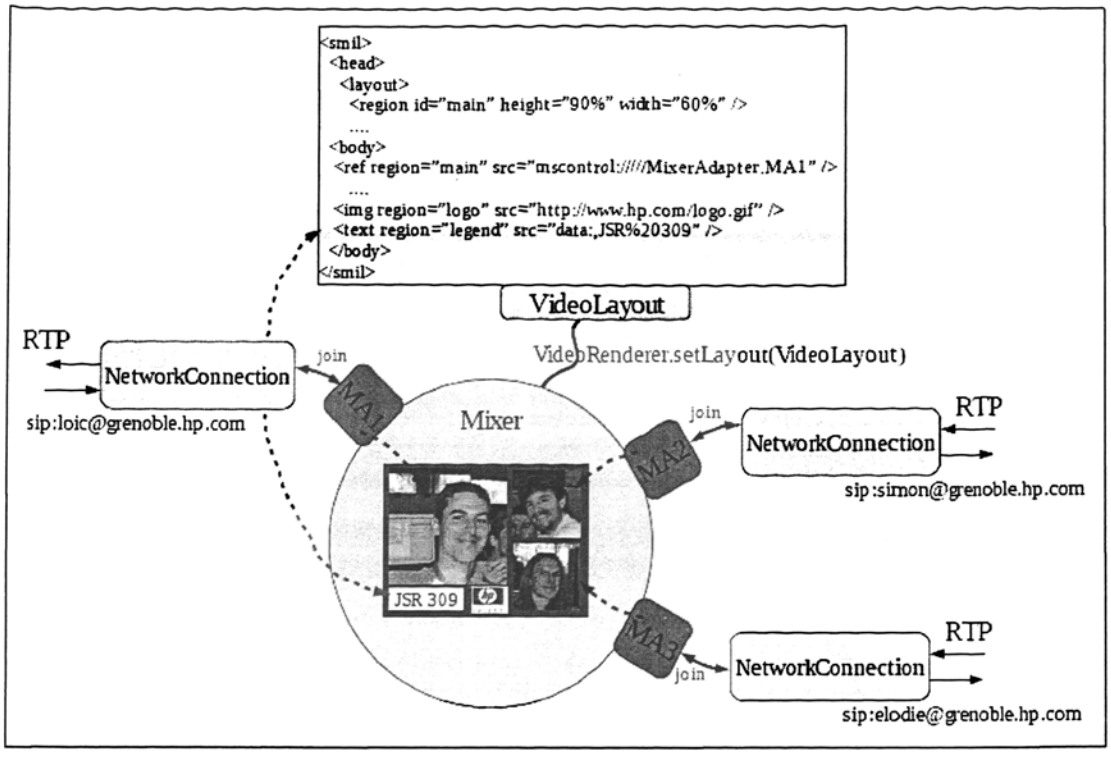



图 4-11 基于 SMIL 设置视频版面布局示例

从上面的例子，可以看到<head>子元素<region>标签定义了“main”的呈现布局信息，<body>子元素<ref>标签的属性 src 定义了区域“main”的媒介来源。在对 SMIL 有所了解后，下面接着介绍操作 setVideoLayout 的具体实现。

setVideoLayout 操作同样是基于对 JMSC 接口的调用，其实现核心代码如下：

核心代码：

```
//获得 VideoRenderer 资源对象
VideoRenderer myRenderer = mediaMixer.getResource(VideoRenderer.class);
//根据 smil xml 文件创建 VideoLayout 对象
VideoLayout myLayout = myMSFactory
        .createVideoLayout("application/smil+xml",smil);
//将 VideoLayout 对象附加到 myRenderer
myRenderer.setLayout(myLayout);
```

清单 4-12 MediaMixer 服务 setVideoLayout 操作接口内部实现核心代码

setVideoLayout 操作具有两个输入参数，smilURI 是已编辑好的 SMIL XML 文件的 URI 地址。

输入消息: **setVideoLayoutRequest**

| | 类型 | 选项 | 描述 |
|---|---|---|---|
| mediaMixerIdentifier | xsd: string | 必填 | MediaMixer参数 |
| smilURI | xsd: anyURI | 必填 | 编辑好的smil xml文件URI，包含了相应的版面布局信息 |

输出消息: **setVideoLayoutResponse**

| | 类型 | 选项 | 描述 |
|---|---|---|---|
| status | xsd:string | No | MediaGroup状态符 |

一个 VideoLayout 对象是一个只读对象，当我们需要更改视频布局时，需要重新调用操作 setVideoLayout，传入新的 SMIL XML 文件即可。

#### 4.3.2.4 MediaMixer 服务对应的 WSDL 服务端口

```
<wsdl:service name="MediaMixerService">
   <wsdl:port binding="impl:MediaMixerSoapBinding" name="MediaMixer">
      <wsdlsoap:address location="http://localhost:8080/jsr-ws/services/MediaMixer"
   </wsdl:port>
</wsdl:service>
```

清单 4-13 MediaMixer 服务 WSDL 文件中的服务端口

## 4.4 通用媒体服务的调用方式

### 4.4.1 用 WSIF 实现服务的透明调用

万维网服务调用框架(Web Service Invocation Framework, WSIF)[37]提供了一组基于 WSDL 文档驱动的简单 API 来调用 Web Service。只要服务采用 WSDL 文档描述，则不管服务是怎样提供或由哪里提供，均可以统一的模式来调用。过去仅局限于 SOAP 进行服务调用的模式转变为基于 WSDL 文档的通用服务调用模式。

万维网服务调用框架将 Web Service 调用的全部问题从以绑定为中心的观点转移到更抽象的级别上，与传统调用模式相比，WSIF 框架具有以下一些特点[37]：

1. 提供了一组对任何服务都独立与绑定访问的 API；
2. 内置了端口类型编译器产生存根，允许使用抽象服务接口调用；
3. 允许无存跟的服务调用；

4. 可以在运行时更新绑定；
5. 可以在运行时插入新的绑定；
6. 允许将绑定选择延后到运行时。

### 4.4.2 通用媒体服务调用测试

对一个特定的 Web Service（如 MediaGroup Web Service），只需要调用 WSIF API，由 WSIF 框架来解析 WSDL 文档，完成相应的特定调用。代码如下清单 4-14 所示：

```
//根据WSDL URI读取WSDL文件
Definition def = WSIFUtils.readWSDL(null, wsdlLocation);
WSIFDynamicPortFactory portFactory = new WSIFDynamicPortFactory(def, null, null);
//获取默认端口
WSIFPort port = portFactory.getPort();
//用户也可以明确的选择端口类型
//WSIFPort port = portFactory.getPort("SOAPPort");
//预备input message
WSIFMessage input = port.createInputMessage();
input.setPart("mediaSession", new WSIFJavaPart(String.class, symbol));
//预备output值的占位符
WSIFMessage output = port.createOutputMessage();
//执行WSDL operation
port.executeRequestResponseOperation("createMediaGroup", input, output, null);
//检查响应
WSIFPart part = output.getPart("mediaGroupID");
return ((Integer)part.getJavaValue()).intValue();
```

清单 4-14 基于 WSIF 的服务调用源代码

WSIFPort 是 WSIF 中的关键抽象，它代表了运行时的一个 WSDL Port，可看作一个服务端点。一旦 WSIFPort 接口被实现了，就能执行 WSDL 中的 operation。

## 4.5 移动视频业务创建和部署方式

### 4.5.1 业务创建和部署方式

基于通用媒体处理服务开放接口的移动视频业务创建和部署方式如图 4-15 所示：

1、根据业务需求设计业务逻辑；

2、在集成开发平台（如 eclipse）根据需求创建相应类型项目；

3、基于 Parlay X 以及通用媒体处理 Web 服务开发业务逻辑；

4、将开发好的项目部署到应用服务器（如 WebLogic Sip Server）；

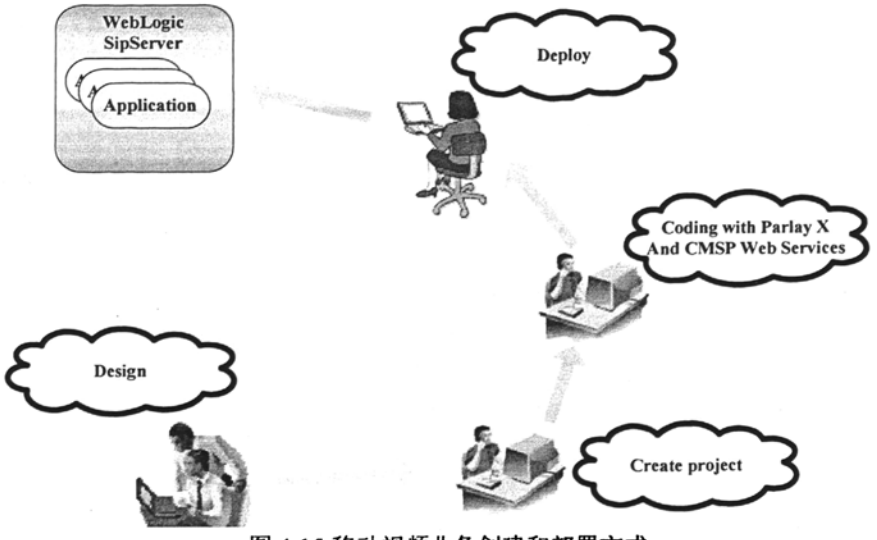



图 4-15 移动视频业务创建和部署方式

### 4.5.2 业务创建和部署特征

基于通用媒体处理服务开放接口的新型视频业务开发模式具有以下特征：

（1）业务开发呈“水平型”结构：业务与底层网络松耦合，业务提供商和网络运营商分离，创建业务层面的良好竞争环境，促进差异化的增值业务的快速提供；

（2）基于开放接口：业务基于开放标准接口开发，可发布于任何网络，具有快速、易用、灵活、动态的业务开发环境；支持第三方业务快速的生成、部署和更新；

（3）开发难度小：第三方开发者不用深入了解电信网络知识，媒体处理和媒体服务器控制知识，开发难度减小、开发周期缩短、投资收益高；

（4）支持业务融合：业务间具有良好的集成关系，业务提供商可以提供方便实用的融合业务，体现软交换网络的技术潜力和市场价值。

## 4.6 本章小结

本章结合第三章的设计方案，重点实现视频业务涉及的通用媒体处理服务，采用 Web Services 技术来封装通用媒体处理服务，并提供了对通用视频服务的透明调用方案。在本章最后，分析了基于设计的视频业务创建和部署平台的业务创建和部署方式，并分析了基于视频业务创建和部署平台的业务创建和部署的特征。视频业务创建和部署平台便利了视频业务的创建和部署，为 3G 时代的视频业务开发提供了一种新的、可参考的开发模式。

# 第五章 移动视频业务实例测试

本章将根据前面的设计与实现，搭建移动视频业务创建和部署的测试平台，进行业务实例的创建和测试。

## 5.1 测试平台

移动视频业务测试平台的搭建从四个方面来考虑：

一、移动视频业务的支撑环境：在本文中，移动视频业务的开发主要基于两类开放业务接口：Parlay X 以及 CMPS 开放接口。Parlay X 开放业务接口提供了电信网络的业务能力，在这里采用爱立信（Ericsson）公司 08 年发布的 tc_ws_sdk（Parlay X 网关仿真程序）开发包来提供，由 Sun 公司的 SJSAS（Sun Java System Application Server）应用服务器提供 Parlay X 网关仿真程序的运行环境。CMPS 开放接口是本文重点实现的部分，第四章已经详细描述了其实现，开发好的通用媒体处理 Web 服务将会部署到 Tomcat 服务器上，媒体服务的具体实现由惠普公司的 HP OpenCall 媒体平台提供。

二、移动视频业务的生成环境（SCE）：移动视频业务的开发主要基于 Eclipse 集成开发环境。

三、移动视频业务的执行环境（SLEE）：开发好的移动视频业务将会部署到BEA公司的WSS（WebLogic SIP Server）服务器上。WSS服务器可用来对使用 SIP 的应用程序进行呼叫控制和会话管理。

四、移动视频业务的客户端：本文采用CounterPath公司提供的X-Lite SIP软电话作为视频业务的测试终端。

### 5.1.1 测试平台网络架构

移动视频业务测试平台的网络架构如图 5-1 所示：

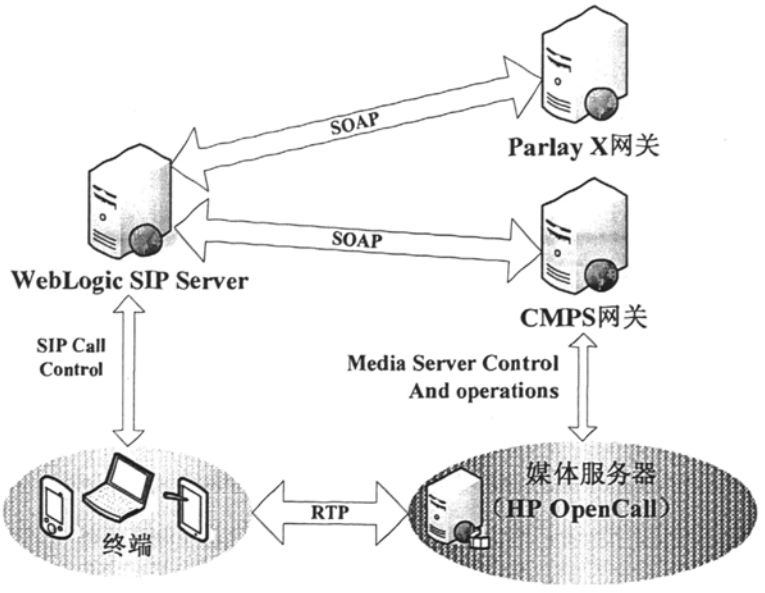



图 5-1 移动视频业务测试平台网络架构

移动视频业务测试平台系统需要至少一台媒体服务器、一台运行 Weblogic SIP Server 的应用服务器、Parlay X 网关以及通用媒体服务网关，两台或以上的运行 SIP Soft phone 的个人 PC。业务测试平台对硬件条件的要求以能运行相应的软件为准，没有特别的限制。

### 5.1.2 业务编辑环境主要部件

➢ Eclipse[35]集成开发环境：Eclipse 是由开放社区提供的开放工具集成平台，具有友好的用户图形界面，它的特点是允许第三方在其上开发各种 Eclipse 插件，功能非常强大。Lomboz 是开发一个 Eclipse 插件，可以辅助生成 Web Service。

### 5.1.3 业务执行环境主要部件

➢ WSIF[37]：Web Service 调用框架，提供了一组基于 WSDL 文档驱动的简单 API 来调用 Web Service。。

➢ WebLogic SIP Server：Web 应用服务器，可用来对使用 SIP 的应用程序进行呼叫控制和会话管理，在这里用来部署开发好的视频业务。

### 5.1.4 外部环境

#### 5.1.4.1 服务器端

➢ HP_OpenCall_Media_Platform 3.1[13]：基于 SIP 控制的媒体处理平台，拥有先进的语音和多媒体处理性能，可支持新型融合 IMS 语音和视频应用。在这里用来提供通用媒体处理服务的具体实现。

➢ CMPS（通用媒体处理服务开放接口）：本文重点实现的部分，根据 JSR 309 规范开发，提供了通用的媒体处理 web 服务，通过 WSDL 定义开放接口，应用开发者可以基于 WSIF 调用 CMPS 提供的媒体处理服务，通用媒体处理服务部署在 CMPS 网关上。

➢ Tomcat[36]服务器，是一种 Servlet/JSP 容器，也叫 Web 应用服务器。它为 CMPS（通用媒体处理服务开放接口）提供运行环境。

➢ Telecom Web Services Network Emulator[38]：一个 Parlay X 网关仿真程序，Parlay 网关可以把移动运营商网络中的业务能力，如 SMS、MMS、终端状态、WAP-Push 等，展现给第三方，是对移动网络环境的仿真。

➢ Sun Java System Application Server[39]：和 Tomcat 一样是 Web 应用服务器，在这里主要为 Parlay X 网关仿真器提供运行环境。

#### 5.1.4.2 客户端

➢ X-lite SIP Soft phone[40]：客户端采用 CounterPath 厂家提供的 SIP Phone 的软件实现。

## 5.2 业务实例测试

### 5.2.1 业务生成

为了测试本课题提出并实现的业务创建和部署平台对移动视频业务的支持，本章在搭建的视频业务开发平台上实现了典型的移动流媒体业务：移动视频点播业务。下面具体介绍该业务的开发流程。

1. 服务器端的业务开发

在这里不开发视频点播业务涉及的 web 页面，在配置文件里设置要播放的视频文件，对视频点播业务进行简单演示。视频点播业务的业务逻辑图如下：

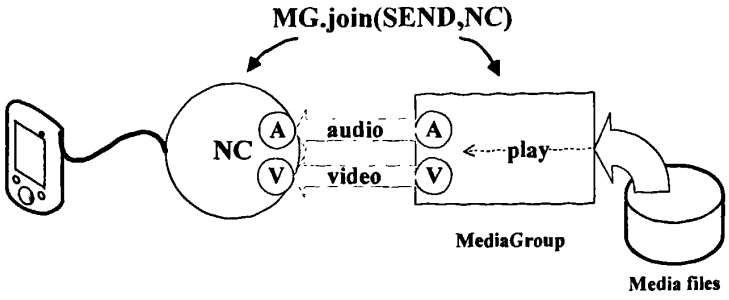



图 5-2 视频点播业务逻辑图

如上图所示，NC 代表了到移动终端的 RTP 连接，将 NC 关联到 MediaGroup 对象时设置的参数是“SEND”，代表视频流是从媒体服务器到移动终端的单向媒体流。视频业务的开发需要调用通用媒体处理服务 MediaGroup 服务的相关操作来实现。实现时，在 SIPServlet 中组合 MediaGroup 服务的几个操作，实现视频点播业务的业务逻辑。媒体文件的播放是由 MediaGroup 服务的 controlMediaGroup 操作完成的，controlMediaGroup 操作调用 player 对象来为移动终端播放指定的视频。视频业务的调用流程图如下图所示：

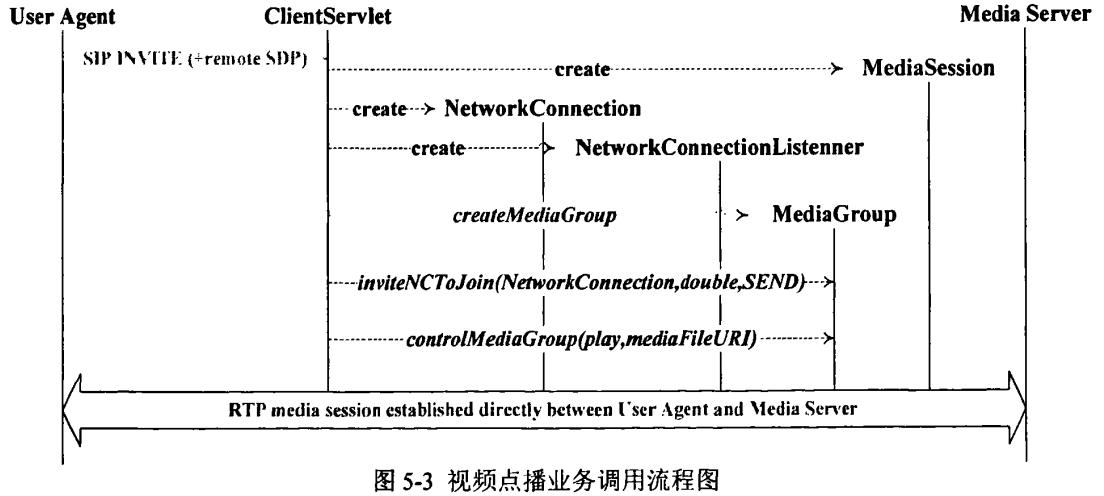



图 5-3 视频点播业务调用流程图

具体代码的开发在 Eclipse 集成开发环境中完成。通用媒体处理 Web 服务的调用参考第四章中基于 WSIL 的透明调用方式。

业务开发完成后，在业务部署到应用服务器之前，还需要设置配置文件 sip.xml，该文件设置了应用服务器端的监听程序 ClientServlet，并设置了 ClientServlet 的相关参数。其中 MS_ADDR 为安装媒体服务器的机器 IP 地址，MS_PORT 为媒体服务器的监听端口，DRIVER_TYPE 为媒体服务器对应的驱动类型，MY_ADDR 为安装 WebLogic Sip Server 的机器 IP 地址，MY_PORT_SIP 为 WebLogic Sip Server 的 SIP 监听端口，MY_PORT_HTTP 是 WebLogic Sip Server 的 HTTP 监听端口，MEDIA_FILE 为媒体文件的地址。sip.xml 文件中的<servlet-mapping>标签定义了 ClientServlet 的响应类型，本案例中，服务器端的

ClientServlet 监听 SIP 端口，当其收到终端发过来的 INVITE 请求时，触发视频点播业务。sip.xml 文件相关内容如下：

```
<servlet>
  <servlet-name>OCMPServlet</servlet-name>
  <servlet-class>com.bea.asf.jmsc.driver.ocmp.OCMPServlet</servlet-class>
  <init-param>
    <param-name>supportSessionTimer</param-name>
    <param-value>1</param-value>
    <description>
      Optional. Determine if Session Timer Support should be turned on.
    </description>
  </init-param>
  <load-on-startup>1</load-on-startup>
</servlet>

<servlet>
  <servlet-name>ClientServlet</servlet-name>
  <servlet-class>njupt.vod.ClientServlet</servlet-class>
  <load-on-startup>1</load-on-startup>
  <init-param>
    <param-name>MS_ADDR</param-name>
    <param-value>127.0.0.1</param-value>
  </init-param>
  <init-param>
    <param-name>MS_PORT</param-name>
    <param-value>5054</param-value>
  </init-param>
  <init-param>
    <param-name>DRIVER_TYPE</param-name>
    <param-value>OCMP</param-value>
  </init-param>
  <init-param>
    <param-name>MY_ADDR</param-name>
    <param-value>127.0.0.1</param-value>
  </init-param>
  <init-param>
    <param-name>MY_PORT_SIP</param-name>
    <param-value>5060</param-value>
  </init-param>
  <init-param>
    <param-name>MY_PORT_HTTP</param-name>
    <param-value>8001</param-value>
  </init-param>
  <init-param>
    <param-name>MEDIA_FILE</param-name>
    <param-value>file://D:/web/bluescreen2.mov</param-value>
  </init-param>
  </servlet>

<servlet-mapping>
  <servlet-name>ClientServlet</servlet-name>
  <pattern>
    <equal>
  <var>request.method</var>
  <value>INVITE</value>
    </equal>
  </pattern>
</servlet-mapping>
```

清单5-4 视频点播业务部署文件sip.xml

2. 业务部署

业务开发好之后，将业务应用部署到 Weblogic SIP Server 服务器中，步骤如下：

a. 进入域目录，运行服务器 startWebLogic.cmd，并登陆服务器控制台页面；

b. 通过控制台可视化页面部署应用；

c. 开启应用，并保证应用处于“Active”状态；

此时服务器已将所部署应用加载进内存，等待业务触发。

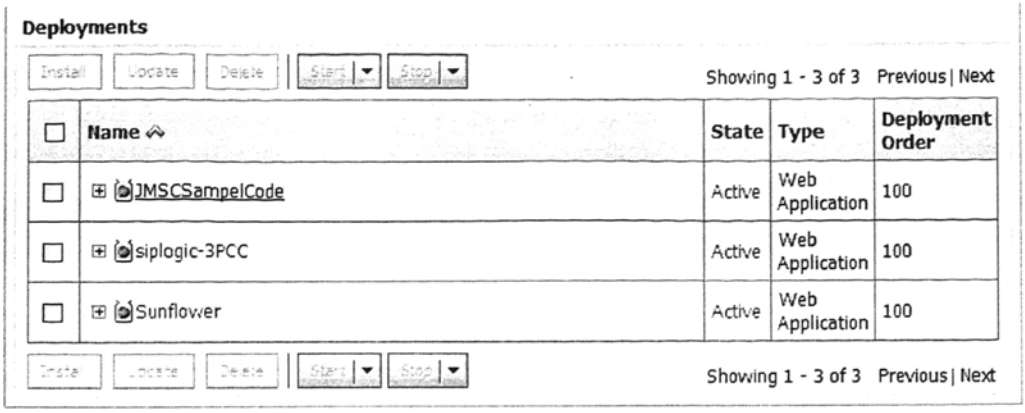



图 5-5 成功部署到 WebLogic SIP Server 的 JMSCSampleCode 案例

## 5.2.2 业务测试

测试该业务时，首先打开 X-Lite 终端，并对 X-Lite 软电话进行相关设置，如图：

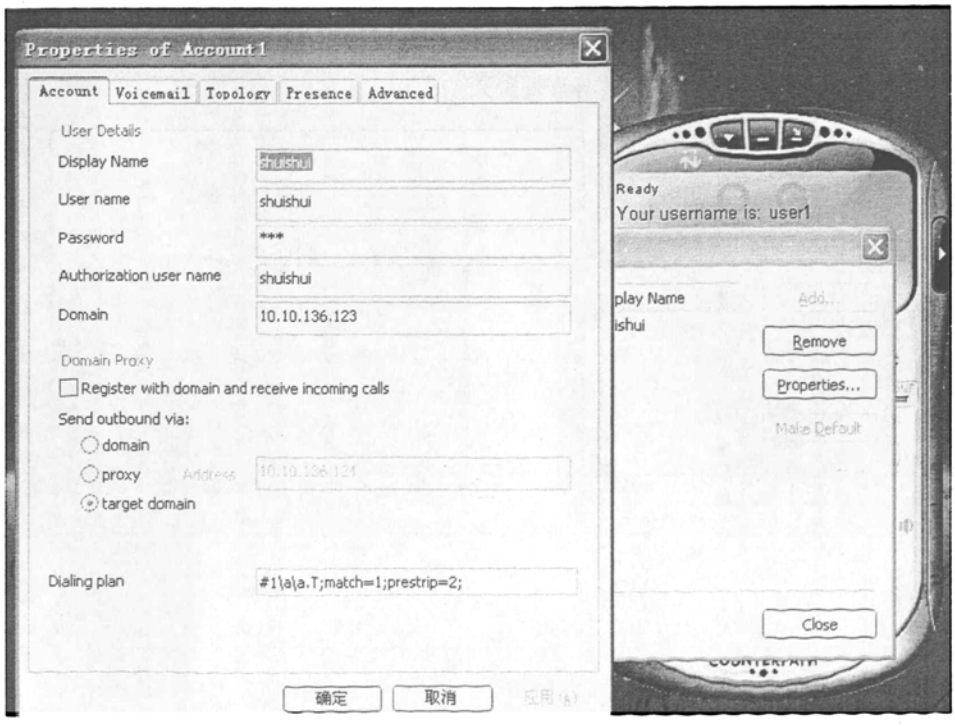


图 5-6 X-Lite 终端配置

此时用户对应的 sip 地址为：sip:shuishui@10.10.136.123，通过终端 INVITE SIP Server，用户名任意，部署在 WebLogic SIP 服务器端的 ClientServlet 匹配到 INVITE 类型的 SIP 消息后，会触发移动视频点播业务流程，通过调用开发好的通用媒体处理 Web 服务控制媒体服务器，建立媒体服务器到终端的 RTP 连接，此时媒体服务器将会为 SIP 终端播放配置文件中指定的视频文件，如图 5-7 所示，案例测试成功。

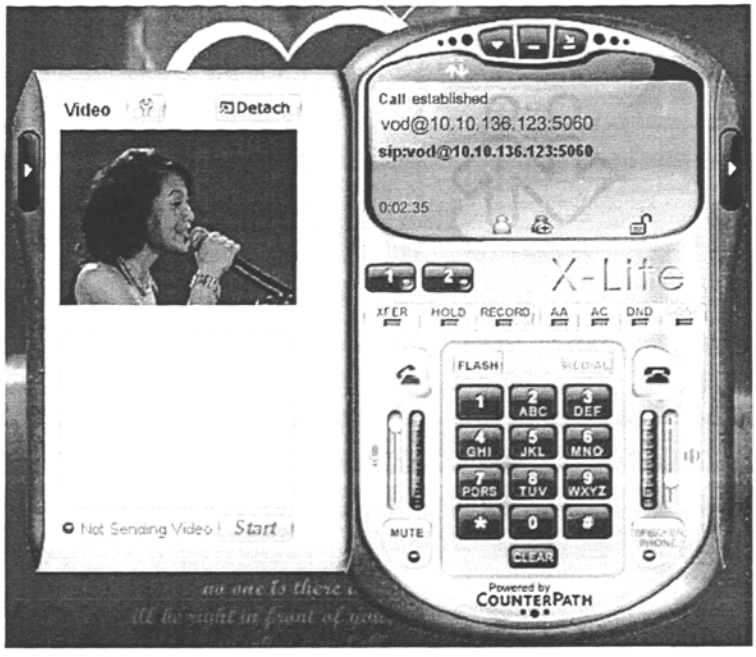



图 5-7 移动视频点播业务客户端画面

## 5.2.3 测试结果分析

基于测试结果，可以得出移动视频业务平台具有以下特点：

（1）易于扩展：移动视频业务平台的设计基于分层的设计思想，视频业务所需的媒体处理服务与底层的媒体服务器相分离，在这种模式下，当有新的媒体服务器加入时，可以方便的在 CMPS 网关中发布自己的服务能力，供业务开发者使用。异构的媒体服务器通过 JMSC 层的媒体驱动汇聚到通用媒体处理 Web 服务层。

（2）易于开发：业务开发基于统一接口，通过本文设计的视频业务平台开发移动视频业务的时候，只需要将关注点集中在具体的业务逻辑实现。在设计测试案例—移动视频点播业务的时候，开发人员只需调用 MediaGroup 服务的操作接口，不需要关心网络的呼叫过程与资源管理过程。论文提出的新型的视频业务开发模式，能够有效地降低业务开发人员开发移动视频业务的技术门槛。

（3）易于集成：通用媒体处理服务基于 Web Services 技术封装，Web Services 技术能够支持面向服务的，松耦合的业务集成，并且通过统一的认证、动态工作流控制和交易控制机制解决了 Internet 上各个业务平台的业务能力同步问题。如基于 MediaGroup 服务可以开发移动视频点播业务，基于 MediaMixer 服务可以开发移动视频会议业务，而将 MediaGroup 服务与 MediaMixer 服务组合使用时，可以方便地开发移动 K 歌业务。

（4）安全性：移动视频业务平台基于的开放接口，都是针对业务能力的封装，业务开发人员并不能直接访问到网络控制能力，以及底层的媒体服务器，因此具有很好的网络安全性。同时业务能力基于 Web Services 封装，Web Services 环境下可能的服务质量保障技术、访问控制技术、安全技术，如 WS-Security，都可以引入到移动视频业务平台中来。

## 5.3 本章小结

本章主要在第四章实现技术分析的基础上，搭建了移动网络环境下的视频业务创建和部署平台，通过在视频业务创建和部署平台上开发具体的视频业务案例，并对业务进行部署和测试，验证视频业务创建和部署平台对视频业务开发的通用支持。

# 第六章 总结与展望

## 6.1 全文总结

本文的研究目标是解决移动网络中的视频业务开发问题。移动视频业务通过移动网络和移动终端为移动用户提供视频内容。本文分析了 IP 多媒体子系统 IMS 对视频业务的支持，基于 IMS 网络架构中 MRF 的设计思想，并且进一步参照 JSR 309 规范、Parlay X 技术，设计了移动网络视频业务的媒体处理方案。在此基础上，设计了移动视频业务创建和部署平台的整体框架。最后，通过在移动视频业务测试平台上开发具体的视频业务案例，验证了论文所设计的移动视频业务创建和部署平台对视频业务开发的通用支持。

基于通用媒体处理的移动视频业务开发模式，遵循了下一代网络的发展趋势，具有以下特点：

（1）业务开发呈"水平型"结构：业务与底层网络松耦合，业务提供商和网络运营商分离，创建业务层面的良好竞争环境，促进差异化的增值业务的快速提供。

（2）基于开放接口：业务基于开放标准接口开发，可发布于任何网络，具有快速、易用、灵活、动态的业务开发环境；支持第三方业务快速的生成、部署和更新。

（3）调用方式和接口描述相对独立：如果服务能力的调用形式改变，只需更改服务绑定部分的描述，更新服务注册库信息，不会对业务逻辑造成影响，提高调用效率和业务灵活性。

（4）开发难度小：第三方开发者不用深入了解电信网络知识，媒体处理和媒体服务器控制知识，开发难度减小、开发周期缩短、投资收益高。

（5）支持业务融合：业务间具有良好的集成关系，支持分布式环境下的业务集成，业务提供商可以提供方便实用的融合业务；

## 6.2 进一步工作与展望

虽然通用媒体处理 Web 服务能够满足业务开发人员对媒体处理服务的需要，但是基于 Web Services 技术开发的通用媒体处理服务仍然有其不足：

首先，Web Services 技术通常用于完成持久性无状态的服务，持久性和无状态意味着 Web Services 服务并不会区分客户，不会记录客户的一系列相关操作。其认为每一个客户

的每一次请求都是相对独立的一次服务过程。因此需要通过应用扩展来支持服务的状态。

其次，通过 Web Services 技术提取和封装了 JMSC 提供的通用媒体服务器控制接口，虽然能够简化媒体处理服务的调用，但是本文实现的媒体处理业务能力还过于简单，更高一层的封装削弱了对媒体服务器的控制能力，需要进一步的深入研究，在本文实现的通用媒体服务能力上进行扩展。

最后，基于 Web Services 的无状态特性，现有呼叫控制类 API 没有对呼叫状态的维持，因此无法对呼叫的进展保持监控，也不能根据呼叫的具体进展情况做进一步的处理。

这一系列问题仍然需要进一步的研究，是本文下一步的研究方向。

# 致 谢

在硕士研究生生活即将结束，毕业论文即将完成之际，我非常感谢在这些年中所有帮助过我的老师和同学们。

首先，我要感谢我的导师沈苏彬老师。从最初的选题、课题方案的确定、到最终论文的完成，每一步都离不开沈老师的悉心指导和关怀。沈老师以其渊博的学识，开阔的科学视野、活跃的思维方式给予我全方位的指导和启发；沈老师一丝不苟的治学态度，兢兢业业，永不疲倦的工作精神教会了我做学问首先要端正态度、勤奋踏实，尽职尽责，做一个真正意义上的研究生，使我除了就科研本身还学到了许多做人的道理，许多待人处事的方法和克服困难的方法。这些道理和方法必将成为我未来在社会立足之本，职业发展之道，是伴随我一生的宝贵财富。

还要感谢实验室的其他老师和所有同窗好友，能够和大家一起度过人生中最宝贵的三年青春岁月必将是我最美好的回忆，感谢大家在一起学习交流和参与科研工作时对我的指点和建议，在我遇到困难时候给予的鼓励和帮助，也非常感谢大家对我生活上的关心和照顾。特别要感谢钱海忠师兄和欧阳志友老师在工作和论文中对我的指点和帮助。

最后，我深深感谢我的父母，感谢他们从物质上和精神上对我学业的大力支持，对我生活上无微不至的关怀，使我可以安心学习没有后顾之忧，他们的鼓励和支持让我有了在困难中前行的勇气，他们一直以来对我的信任让我充满了对未来和人生的信心。

我再次衷心感谢教导和帮助过我的各位师长、同学和朋友！感谢审查论文的各位专家、评委！谢谢！

本论文研究工作来源于国家863项目，项目编号为2006AA01Z208，在此一并表示感谢。

# 参考文献

[1] Jing Zhang, Xiong-jian Liang. 3G in China: Environment and Prospect. PICMET, Aug. 2007 Page(s):2988 – 2992.

[2] 梅玉平，杨维忠，张琳峰等.《3G 与固定视频业务的融合》.人民邮电出版社, 2005.

[3] Cochennec, J.-Y.Activities on next-generation networks under Global Information Infrastructure in ITU-T. Communications Magazine, IEEE, Volume 40, Issue 7, July 2002 Page(s):98 - 101

[4] Niladri Sekhar Nath. SIP Developer Suite 5.0.
http://www.radvision.com/Products/Developer/Protocol/SIPDeveloper/

[5] Moerdijk, A.-J. Klostermann, L. Opening the networks with Parlay/OSA: standards and aspects behind the APIs. Network, IEEE, Volume: 17, Issue: 3, May-June 2003, On page(s): 58- 64.

[6] Parlay X Web Services, Version 3.0. http://www.parlay.org/en/specifications/pxws.asp, UPDATED JUNE 2007.

[7] Pailer, R., Stadler, J., and Miladinovic I. Using PARLAY APIs Over a SIP System in a Distributed Service Platform for Carrier Grade Multimedia Services. Wireless Networks, Jul. 2003:353-363.

[8] Magedanz, T., Witaszek, D., Knuettel, K. The IMSPlayground @ Fokus - An Open Testbed for Next Generation Network Multimedia Services. In Proceedings of First International Conference on Testbeds and Research Infrastructures for the Development of Networks and Communities (TRIDENTCOM'05), IEEE Computer Society, Feb. 2005:2-11.

[9] Magedanz, T., Witaszek, D., Knuttel, K. Service Delivery Platform Options for Next Generation Networks within the national German 3G Beyond Testbed. SATNAC04, Stellenbosch, South Africa, Sep.2004.

[10] Michio Irie, Masashi Kaneko, Moo Hirano, etc. Highly Flexible Media Handling Enabler for NGN Service Delivery Platform. IEICE, April 2008 Page(s):126 – 131.

[11] Michio Irie, Masashi Kaneko, Moo Hirano, etc. Novel SIP Session Handling in a Service Delivery Platform for Telecom-Web Integration. IEICE, April 2008 Page(s):92 – 97.

[12] Charles Wong. 全业务运营下的 IP 媒体服务器. http://www.microvoip.com/broadcasting/

[13] HP OpenCall - Media Platform. http://h20208.www2.hp.com/opencall/products/media/ocmp/index.jsp

[14] SnowShore IP Media Server. http://www.cantata.com/promotions/mediaserver_download/index.cfm

[15] F. Andreasen,B. Foster. Media Gateway Control Protocol. IETF RFC 3435,January 2003.

[16] H.248/Megaco Information Site. http://www.packetizer.com/ipmc/h248/

[17] J. Rosenberg,H. Schulzrinne,G. Camarillo,etc. SIP: Session Initiation Protocol. IETF RFC 3261, June 2002.

[18] E. Burger, J. Van Dyke,A. Spitzer. Basic Network Media Services with SIP. IETF RFC 4240, December 2005.

[19] J. Van Dyke, E. Burger, A. Spitzer. Media Server Control Markup Language (MSCML) and Protocol. IETF RFC 5022, September 2007.

[20] A. Saleem, Y. Xin,Radisys, G. Sharratt. Media Server Markup Language (MSML), draft-saleem-msml-06. Internet Draft, February 11, 2008.

[21] A. Saleem, G. Sharratt. Media Objects Markup Language (MOML), draft-melanchuk-sipping-moml-06.

Internet Draft, October 21, 2005.

[22] Marc Brandt, Hewlett-Packard. JSR 309: Media Server Control API. http://jcp.org/en/jsr/detail?id=309,

[23] Web Services Activity. http://www.w3.org/2002/ws/

[24] 3GPP, 3GPP T5 23.228 IP Multimedia Subsystem(IMS), Stage2(R6). http://www.3gpp.org

[25] 3GPP, 3GPP TS 23.002 Network architecture (R6). http://www.3gpp.org

[26] A. Al-Hezmi, F. Carvalho de Gouveia, M. Sher, etc. Provisioning IMS-based Seamless Triple Play Services over Different Access Networks. IEEE, April 2008 Page(s):927 – 930.

[27] Lopez Lavado, Miguel Barba Marti, Antoni . Evolution of CAMEL Service Platform beyond 3G networks. Latin America Transactions, IEEE (Revista IEEE America Latina), Volume 4, Issue 5, Sept. 2006 Page(s):315 - 325

[28] Rutz, R. Richert, J. CAMeL: an open CACSD environment. Control Systems Magazine, IEEE , Volume 15, Issue 2, April 1995 Page(s):26 - 33.

[29] 孔 松, 张力军. 基于 IMS 的下一代网络融合架构研究[J ]. 邮电设计技术,2006 (2) :27231.

[30] IETF Home Page. http://www.ietf.org/

[31] M. Handley, V. Jacobson. SDP: Session Description Protocol. IETF RFC 2327, April 1998.

[32] 杨 涛,刘锦德，Web Services 技术综述——一种面向服务的分布式计算模式，计算机应用，2004，Vol. 24, No. 8.

[33] 岳 昆+，王晓玲，周傲英，Web 服务核心支撑技术研究综述，软件学报，2004，Vol.15, No.3.

[34] JSR-000309 Media Server Control API, http://jcp.org/aboutJava/communityprocess/pr/jsr309/index.html，2008.09.26

[35] Eclipse, http://www.eclipse.org

[36] Apache Tomcat, http://tomcat.apache.org/

[37] Apache WSIF Project, http://ws.apache.org/wsif/

[38] Telecom web services SDK,
http://www.ericsson.com/developer/sub/open/technologies/parlayx/tools/telecom_web_services

[39] Sun Java System Application Server, http://java.sun.com/javaee/downloads/index.jsp

[40] X-Lite, http://www.counterpath.com/x-lite.html&active=4

[41] Gonzalo Camarillo, Miguel A.Garcia-Martin. The 3G IP Multimedia Subsystem: Merging the Internet and the cellular worlds. Wiley. 2004.

[42] 沈苏彬. 基于开放业务的网络融合. 中兴通讯技术.2004,10(2):13~16.

[43] A-J Moerdijk, L.Klostermann. Opening the networks with Parlay/OSA: standards and aspects behind the APIs. IEEE Network,2003,17(3):58-64.

[44] 刘真，杨景. 基于业务级互联的下一代网络业务生成研究. 中国科学院，博士学位论文, 2006.

[45] A.R.Modarressi, S.Mohan. Control and management in next-generation networks: challenges and opportunities. IEEE Communications Magazine, 2000, 11838(10):94-102.