DHU

東華大學

# 硕士学位论文

学术学位

# DONGHUA UNIVERSITY MASTER DISSERTATION

论文题目：FFMPEG 转码技术在 HTML5 视频系统中的研究与应用

学科专业：计算机软件与理论

研究方向：

作者姓名：赵淑漫

学　　号：2111458

指导老师：乐嘉锦

完成日期：2013 年 12 月

学校代码:10255

学　号:2111458

# FFMPEG 转码技术在 HTML5 视频系统中的研究与应用

# Research and Application of FFMPEG Transcoding Technology Based on HTML5 Video System

学科专业：计算机软件与理论

作　　者：赵淑漫

指导教师：乐嘉锦

答辩日期：2014 年 1 月

2013 年 12 月

# 东华大学学位论文原创性声明

本人郑重声明：我恪守学术道德，崇尚严谨学风。所呈交的学位论文，是本人在导师的指导下，独立进行研究工作所取得的成果。除文中已明确注明和引用的内容外，本论文不包含任何其他个人或集体已经发表或撰写过的作品及成果的内容。论文为本人亲自撰写，我对所写的内容负责，并完全意识到本声明的法律结果由本人承担。

**学位论文作者签名：** 赵淑漫

**日期：** 2014 年 1 月 6 日

# 东华大学学位论文版权使用授权书

学位论文作者完全了解学校有关保留、使用学位论文的规定，同意学校保留并向国家有关部门或机构送交论文的复印件和电子版，允许论文被查阅或借阅。本人授权东华大学可以将本学位论文的全部或部分内容编入有关数据库进行检索，可以采用影印、缩印或扫描等复制手段保存和汇编本学位论文。

保密 □，在 ____ 年解密后适用本版权书。

本学位论文属于

不保密 ☑

学位论文作者签名：赵淑漫　　指导教师签名：

日期：2014 年 1 月 6 日　　日期：2014 年 1 月 6 日

# FFMPEG转码技术在HTML5视频系统中的研究与应用

## 摘 要

近年来，随着互联网技术的发展，单纯承载文字和图片的静态网页已经远远不能满足现在人们的需求，而使用 HTML 不能在浏览器上实现显示动画视频，音频，绘图等的动态效果，因此，当今唯一可实现在网页中嵌入视频，同时可使用户忽略浏览器和操作系统的方法是使用 Adobe Flash 插件结合<object>和<embed>标签，然而在安装 Flash 插件时，总是伴随着各类安全隐患问题。同时，Flash 插件占用资源大，非系统原生，与系统和浏览器的结合不够紧密，因此导致浏览器假死的故障时有发生。其次，随着网络的普及和 Web2.0 技术的发展，网络视频行业不断兴起，越来越多的用户选择通过网络视频平台来观看各种视频节目，因此，各种视频播放系统对人们的生活影响越来越大。在这种情况下，作为下一代 Web 规范之一的 HTML5 的多媒体优势凸显出来，HTML5 为浏览器提供了音视频，动画，绘图，表单等类型内容的新的嵌入标准，来完成音视频流媒体服务，可以为用户提供流畅，清晰的播放体验。

为使用户能够享受更流畅和更清晰的播放体验，同时减少网络带宽的负载，文章提出使用 HTML5 构建视频播放系统，用户可以上传自己的视频，与其他用户共享视频内容，同时可以收看互联网视频，

但是由于HTML5主要依靠其最新的Video标签和Audio标签来实现音视频播放，Video 标签仅支持三种特殊的视频格式：带有 Theora 视频编码和Vorbis音频编码的Ogg文件、带有H.264视频编码和AAC音频编码的 MPEG4 文件和带有 VP8 视频编码和 Vorbis 音频编码的 WebM文件，同时用户上传视频的格式是多种多样的，为了充分利用HTML5的多媒体标签，使用户能够享受更流畅和更清晰的播放体验，需要在基于HTML5的视频系统的中使用FFMPEG对视频进行转码，以供HTML5的Video标签播放。

基于以上问题，文章先介绍如何构建HTML5构建视频播放系统，然后对多媒体视频处理工具 FFMPEG 的架构进行深入研究分析，在此基础上，对 FFMPEG 进行相应的二次开发，实现将用户上传的各种格式的视频转成HTML5 Video标签可以播放的三种视频格式的转码功能，并将此转码功能应用到基于HTML5的视频系统中，通过实验结果显示，采用 HTML5 的多媒体标签结合 FFMPEG 转码技术构建的视频系统可以为用户提供更加流畅、清晰、优化的视频播放体验，同时使网络带宽负载更小，浏览器的性能消耗更低，同时具有一定的创新性和实用性。

**关键字**：HTML5;CSS3;FFMPEG;视频播放;视频转码

# RESEARCH AND APPLICATION OF FFMPEG TRANSCODING TECHNOLOGY BASED ON HTML5 VIDEO SYSTEM

## ABSTRACT

In recent years , with the development of Internet technology, simple static pages with text and pictures have been far from meeting the needs of people, and animated video, audio , graphics and other dynamic effects can not be achieved in the browser using HTML, so the only method to show videos in a web page , while allowing the user to override the browser and the operating system is the use of Adobe Flash plugin <object> and <embed> label , however, when installing the Flash plug-in, it is always accompanied by various safety problems . Meanwhile , Flash plug-occupied large resources , non- native systems , and systems and browsers is not combined tight enough , thus resulting in suspended animation problems when a fault has occurred, Secondly, with the popularization and development of Web2.0 technology networks , online video industry continues to rise , more and more users watch a variety of videos via online video platform , a variety of video playback system has increasing influence on people's lives. In this case , as the next generation of HTML , HTML5 provides a new standard about embedded audio and video , animation , graphics , forms, and other types of content for the browser to complete audio and video streaming services, and it can provide users with smooth , clear playback experience.

To provide smoother and clearer playback experience for users , while reducing the load on the network bandwidth, the article proposes to build the video playback system based on HTML5 , the user can upload his own videos and share video content with other users, However, due to mainly rely on its latest video and audio tags, HTML5 can support playing videos in the browser, Video tag supports only three special video formats: Ogg files with Theora video and Vorbis audio, MPEG4 files with H.264 video and AAC audio, WebM files with VP8 video and Vorbis audio, while videos uploadeb by users have different formats, in order to take full advantage of HTML5's

multimedia tags, allowing users to enjoy a smoother and clearer playback experience, it need to use FFMPEG transcoding the video to play for the HTML5 video tag in HTML5-based video systems.

Based on the above issues, the article first describes how to build the video playback systems based on HTML5, then takes deeply research and analysis for the video and multimedia processing tools FFMPEG, on this basis, makes secondary development to achieve transcoding function, and takes full use the function into the video system based on HTML5, experimental results show that using a combination of HTML5 multimedia label and FFMPEG transcoding technology to build video systems can provide a smoother, clearer, optimized video playback experience for the user, while the network bandwidth load is smaller, consumption of browser performance is lower , and it is showed innovative and practical.

Shuman Zhao (Computer Software and Theory)

Supervised by Jiajin Le

KEY WORDS : html5, css3, ffmpeg, video system, video transcoding

# 目录

# 第一章 绪论

随着网络技术的发展，为实现在网页中插入视频，同时不借助第三方的视频插件，提出使用 HTML5 的音视频元素 video 标签，为用户提供流畅、清晰的视频播放体验，为完善视频转码功能，提供更加全面的视频管理，采用 FFMPEG 技术实现视频编解码，本章节对课题的研究背景，研究现状，研究内容，研究目标等方面进行概述，分析课题的可行性。

## 1.1 课题研究背景

近几年，互联网技术的发展十分迅猛，已经在各行各业得到了广泛的应用，单纯承载的文字和图片的网页已经远远不能满足现在互联网用户对音频，视频的需求，而现行的 HTML 标准又不能单从浏览器上来实现各种各样的动画视频，音频，绘图等的效果。因此各种插件产生了，它在极大程度上满足了人们的需求，然而对于不用的操作平台和不同的浏览器对于插件的选择都是多种，并且他们之间是不能互相兼容的，而且我们在安装各种插件的同时，伴随而来的是计算机或者各种移动设备卡机，漏洞太多容易中毒等问题，插件数量越多，造成影响的可能性也就更大。因此，一些主要的浏览器厂商成立了 Web Hypertext Application Technology Working Group(What WG)来开发传统 HTML 的新版本[1]。W3C 也启动了自己的新一代 HTML 项目，双方的成员很多是相同的。这两个项目最终很可能合并，虽然很多细节还在争论中，但新一代版本 HTML（HTML5）的大体轮廓已经清楚了。

随着 HTML5 的发展，HTML5 中各种新特性引起互联网技术开发人员的广泛兴趣，其中 HTML5 中 video 标签的出现为互联网视频系统带了希望，传统互联网视频系统大多通过视频播发插件即 Flash 插件来进行播放，但这种方法有无法解决的问题，即插件安装的不安全性，以及在各种播放平台的兼容性，而 HTML5 的 video 标签相对 Flash 插件有着无可比拟的优势。

## 1.2 课题研究现状

随着网络的发展，视频播放系统蓬勃发展，Flash 插件与 HTML5 的 video 标签使用具有不可避免的缺陷，所以基于 HTML5 的视频系统正在不断的突现和广

泛应用，但是 video 标签的使用也有其局限性，即视频格式的限制，HTML5 的 video 标签仅仅支持以下三种格式视频，Ogg（带有 Theora 视频编码和 Vorbis 音频编码的 Ogg 文件）；MPEG4（带有 H.264 视频编码和 AAC 音频编码的 MPEG4 文件）；WebM（带有 VP8 视频编码和 Vorbis 音频编码的 WebM 文件），而且各个浏览器支持的视频格式各不相同，现在如果要在页面中使用 video 标签，则需要考虑三种情况，支持 Ogg Theora 或者 VP8 的（Opera、Mozilla、Chrome），支持 H.264 的（Safari、IE 9、Chrome），都不支持的（IE6、7、8）[2]。

面对各类浏览器之间的视频格式之争，为了跨越视频格式局限，需要对上传的视频进行转码，使用户不需要通过第三方插件，仅通过 HTML5video 标签的使用即可播放各类视频，得到更好的用户体验，因此提出使用 FFMPEG 对视频进行预转码，再作为输入供 HTML5 视频播出。

## 1.3 课题研究内容

本课题旨在通过使用 HTML5 新标签和 CSS3 新特性构建一套完整的多媒体视频播放平台，并同时将基于 FFMPEG 的视频转码模块应用其中，来更加完善基于 HTML5 的视频播放系统的功能，并给用户带来全新的视频播放体验和更好的跨平台，可扩展需求。

本课题中首先将研究 HTML5 新特性在各种平台的适用性，然后结合 CSS3 的新特性完成基于 HTML5 的视频播放系统，同时对 FFMPEG 这一开源可跨平台的音视频软件进行研究，并将其进行二次开发和编译后应用到 HTML5 视频系统中。具体的工作内容可以分为以下几部分：

（1）研究 HTML5 的多媒体新特性，分析其在各种平台和浏览器中的支持度和可行性，通过分析和比较得出构建基于 HTML5 的视频系统的最优方案；

（2）研究 FFMPEG 在音视频转码方面的基本流程和功能，搭建编译环境，对 FFMPEG 进行二次开发，提出满足需求的功能改写方案，并构建满足 HTML5 视频播放需求的音视频转码模块；

（3）基于 HTML5 的视频系统的播放平台开发。使用 HTML5 和 CSS3 以及 javascript 构建网页版基于 HTML5 的视频播放系统，在系统中，视频播放采用 HTML5 中的 video 标签来提供；

（4）基于 HTML5 的视频系统的管理平台的开发。管理平台主要实现用户管理，视频上传，基于 FFMPEG 的视频转码，视频信息管理以及视频数据库添加、删除、修改视频数据，并将数据记录生成 XML/JSON 等功能。同时添加一些辅助功能如视频缩略图的生成，视频可用性的测试等，来提供更好，更全面的视频系统管理功能；

（5）FFMPEG 转码技术在系统中的应用。搭建 FFMPEG 在 Windows 环境下的编译平台，在对 FFMPEG 源代码和架构深入研究分析的基础上，使用 FFMPEG 实现对用户上传视频的编解码，使其可以使用 HTML5 video 标签使用为用户提供清晰流畅的播放体验。

## 1.4 课题研究目标

本课题提出使用 HTML5 中的新特性 video 构建满足用户体验及具有更满足跨平台，可移植需求的视频播放系统，鉴于 video 标签对视频格式的特殊需求，使用 FFmpeg 这一开源免费跨平台的视频音频流方案来对视频进行重新转码，来满足 HTML5 视频播放系统的需求，以给用户带来全新的视频播放体验。

## 1.5 本章小结

本章对整个课题内容进行了简单阐述，分别从课题的研究背景、课题的研究现状、课题的研究内容、课题的研究目的等多个方面概述了课题的研究情况，为下述章节进行了背景的介绍

# 第二章 相关技术和知识介绍

本章节对 HTML5 视频系统和 FFMPEG 转码中所采用的相关技术和知识进行概述，分析相关技术和知识的特点和优势，从理论知识角度论述课题的可行性。

## 2.1 HTML5 技术

作为下一代 HTML 标准，HTML5 的提出为 web 应用的开发提供了各种新特性，强化了 Web 网页的表现性能，同时追加了本地数据库等 Web 应用的功能，本节对 HTML5 从发展历史到技术特性进行研究，分析 HTML5 技术的优势，为构建 HTML 视频系统提供理论和技术基础。

### 2.1.1 HTML5 技术概述

在互联网发展的初期，互联网由于没有一种网页技术呈现的标准，所以多家软件公司就合力打造了 HTML 标准，HTML 标准规定网页如何处理文字，如何安排图画等等，其中最著名的就是 HTML4，这是一个具有跨时代意义的标准，在 HTML4 标准提出之前，互联网上的标准非常混乱，当时的微软、网景等公司都提出了需要制定新的标准来规范互联网，所以 W3C 组织就于 1997 年提出了 HTML4 标准[2]。

由于 HTML4 提出时，互联网环境较差，网络带宽不足，网页的呈现形式也非常有限，在早期的网页上，主要的内容还仅仅是文字，但随着网络带宽的不断提高，人们对于互联网的要求也在不断提高，各个浏览器在发展过程中也在不断的支持各种标准，这使得 HTML4 过于混乱，同时 HTML4 所提供的样式和标记混淆，在 2004 年 W3C 组织提出了 XHTML 标准。

XHTML 只是 HTML 的扩展，对于数据类型要求更为严格，让 HTML 标准变得统一。不过 XHTML 并没有成功，大多数的浏览器厂商认为 XHTML 作为一个过渡化的标准并没有太大必要，所以 XHTML 并没有成为主流，而 HTML5 便因此孕育而生。

如今 IE、Firefox、Safari、Chorme、Opera 等多家浏览器共存竞争的市场格局推动了 Web 技术的进步，同时也引发了浏览器的兼容问题，且愈演愈烈，浏览器兼容问题成为 W3C 组织迫切需要解决的问题之一。HTML5 提供了全新的

功能使浏览器功能更强大，使 web 应用开发更加便捷[3]。

(1)全新的、更合理的标签(Tag)。多媒体对象将不再全部绑定在 object 或 embed 标签中，而是视频有视频的标签、音频有音频的标签；

(2)本地数据库。这个功能将在用户端建立一个内嵌的本地 SQL 数据库，以加速交互式搜索、缓存以及索引功能。同时，那些离线 Web 程序也将因此获益匪浅；

(3)视频不再需要插件。Canvas 对象将让浏览器能直接绘制矢量图，这意味着我们可以脱离 Flash 和 Silverlight，直接在浏览器中显示图形或动画。一些最新的浏览器，除了 IE ，已经开始支持 Canvas；

(4)提供类似桌面应用程序的功能。将提供 API 实现浏览器内的编辑、拖放以及各种图形用户界面的能力；

(5)内容修饰标签将被删除，而取而代之的是使用 CSS。

### 2.1.2 HTML5 技术的特点和优势

HTML5 是近十年来 Web 开发标准巨大的飞跃。和以前的版本不同，HTML5 并非仅仅用来表示 Web 内容，它的新使命是将 Web 带入一个成熟的应用平台，在 HTML5 平台上，视频、音频、图象、动画，以及同电脑的交互等操作都将被标准化。HTML5 最突出的一个特点即 video 标签的提出，Video 标签含有 src、poster、preload、autoplay、loop、controls、width、height 等几个属性，以及一 个内部使用的标签<source>。Video 标签内除了可以包含<source>标签外，还可以包含当指定的视频都不能播放时，返回的内容[2]。

（1）src 属性和 poster 属性

这个属性用于指定视频的地址。而 poster 属性用于指定一张图片，在当前视频数据无效时显示（预览图）；

（2）preload 属性

这此属性用于定义视频是否预加载。属性有三个可选择的值：none（不进行预加载）、metadata（部分预加载）、auto（全部预加载）。如果不使用此属性，默认为 auto；

（3）autoplay 属性

autoplay 属性用于设置视频是否自动播放，是一个布尔属性。当出现时，表示自动播放，去掉是表示不自动播放；

（4）loop 属性

loop 属性用于指定视频是否循环播放，同样是一个布尔属性；

（5）controls 属性

controls 属性用于向浏览器指明页面制作者没有使用脚本生成播放控制

器，需要浏览器启用本身的播放控制栏。控制栏须包括播放暂停控制，播放进度控制，音量控制等等；

（6）source 标签

source 标签用于给媒体指定多个可选择的（浏览器最终只能选一个）文件地址，且只能在媒体标签没有使用 src 属性时使用。浏览器按 source 标签的顺序检测标签指定的视频是否能够播放（可能是视频格式不支持，视频不存在等等），如果不能播放，换下一个。此方法多用于兼容不同的浏览器。

video 标签的使用也有其局限性，即视频格式的限制，HTML5 的 video 标签仅仅支持以下三种格式视频，Ogg(带有 Theora 视频编码和 Vorbis 音频编码的 Ogg 文件)；MPEG4(带有 H.264 视频编码和 AAC 音频编码的 MPEG4 文件)；WebM(带有 VP8 视频编码和 Vorbis 音频编码的 WebM 文件)，而且各个浏览器支持的视频格式各不相同，现在如果要在页面中使用 video 标签，则需要考虑三种情况，支持 Ogg Theora 或者 VP8 的（Opera、Mozilla、Chrome），支持 H.264 的（Safari、IE 9、Chrome），都不支持的（IE6、7、8）。

## 2.2 CSS3 技术

CSS3 可以有效地对页面的布局、字体、颜色、背景和其它效果实现更加精确的控制，在 HTML5 系统中，采用 CSS3 对系统页面进行布局和设置，可提供良好的用户体验，本节中通过 CSS3 的发展历史，技术特性和优势分析 CSS3 页面布局的优势。

### 2.2.1 CSS3 技术概述

CSS 即层叠样式表（Cascading Stylesheet）。 在网页制作时采用 CSS 技术，可以有效地对页面的布局、字体、颜色、背景和其它效果实现更加精确的控制。只要对相应的代码做一些简单的修改，就可以改变同一页面的不同部分，或者页数不同的网页的外观和格式[4]。随着 HTML 的成长，为了满足设计师的要求，HTML 获得了很多显示功能。随着这些功能的增加外来定义样式的语言越来越没有意义了。1994 年哈坤·利提出了 CSS 的最初建议。在 CSS 中，一个文件的样式可以从其他的样式表中继承下来。用户在有些地方可以使用他自己更喜欢的样式，在其他地方则继承，或"层叠"作者的样式。这种层叠的方式使作者和读者都可以灵活地加入自己的设计，混合各人的爱好。

### 2.2.2 CSS3 技术的特点

CSS3 是 CSS 技术的升级版本，CSS3 语言开发是朝着模块化发展的。以前的规范作为一个模块实在是太庞大而且比较复杂，所以，把它分解为一些小的模块，更多新的模块也被加入进来。这些模块包括：盒子模型、列表模块、超链接方式、语言模块、背景和边框、文字特效、多栏布局等。[4]

CSS3 提供的主要新特性：

（1）原生圆角表格；

（2）以往对网页上的文字加特效只能用 filter 这个属性，CSS3 则中专门定制了一个加文字特效的属性，而且可以添加阴影效果等原先实现起来复杂的特效；

（3）丰富了对链接下划线的样式，以往的下划线都是直线，在 CSS3 中有波浪线、点线、虚线等等，更可对下划线的颜色和位置进行任意改变；

（4）在文字下点几个点或打个圈以示重点。

## 2.3 Web Service 技术

### 2.3.1 Web Service 技术概述

Web Service 技术，能使得运行在不同机器上的不同应用无须借助附加的、专门的第三方软件或硬件，就可相互交换数据或集成。依据 Web Service 规范实施的应用之间，无论它们所使用的语言、平台或内部协议是什么，都可以相互交换数据。Web Service 是自描述、自包含的可用网络模块，可以执行具体的业务功能。Web Service 也很容易部署，因为它们基于一些常规的产业标准以及已有的一些技术，诸如 XML 和 HTTP。Web Service 减少了应用接口的花费。Web Service 为整个企业甚至多个组织之间的业务流程的集成提供了一个通用机制[6]。

### 2.3.2 Web Service 技术的组成

Web Service 也叫 XML Web Service Web Service 是一种可以接收从 Internet 或者 Intranet 上的其它系统中传递过来的请求，轻量级的独立的通讯技术。是：通过 SOAP 在 Web 上提供的软件服务，使用 WSDL 文件进行说明，并通过 UDDI 进行注册。

（1）XML：(Extensible Markup Language)扩展型可标记语言。面向短期的临时数据处理、面向万维网络，是 Soap 的基础；

（2）Soap：(Simple Object Access Protocol)简单对象存取协议。是 XML Web Service 的通信协议。当用户通过 UDDI 找到你的 WSDL 描述文档后，他通过可

以 SOAP 调用你建立的 Web 服务中的一个或多个操作。SOAP 是 XML 文档形式的调用方法的规范，它可以支持不同的底层接口，像 HTTP(S)或者 SMTP；

（3）WSDL：(Web Services Description Language) WSDL 文件是一个 XML 文档，用于说明一组 SOAP 消息以及如何交换这些消息。大多数情况下由软件自动生成和使用；

（4）UDDI (Universal Description, Discovery, and Integration) 是一个主要针对 Web 服务供应商和使用者的新项目，利用 SOAP 消息机制（标准的 XML/HTTP）来发布，编辑，浏览以及查找注册信息。它采用 XML 格式来封装各种不同类型的数据，并且发送到注册中心或者由注册中心来返回需要的数据。

### 2.3.3 Web Service 技术的优势

Web Service 的主要目标是跨平台的可互操作性。为了实现这一目标，Web Service 完全基于 XML（可扩展标记语言）、XSD（XML Schema）等独立于平台、独立于软件供应商的标准，是创建可互操作的、分布式应用程序的新平台。因此使用 Web Service 有许多优点[6]:

（1）跨防火墙的通信

调用 Web Service，可以直接使用 Microsoft SOAP Toolkit 或.net 这样的 SOAP 客户端，也可以使用自己开发的 SOAP 客户端，然后把它和应用程序连接起来。不仅缩短了开发周期，还减少了代码复杂度，并能够增强应用程序的可维护性；

（2）应用程序集成

通过 Web Service，应用程序可以用标准的方法把功能和数据“暴露”出来，供其它应用程序使用，完善应用程序的集成；

（3）B2B 的集成

Web Service 是 B2B 集成成功的关键。通过 Web Service，可以只需把关键的商务应用“暴露”给指定的供应商和客户就可以了，用 Web Service 来实现 B2B 集成可以轻易实现互操作性，同时减少花在 B2B 集成上的时间和成本；

(4)软件和数据重用

Web Service 在允许重用代码的同时，可以重用代码背后的数据。

## 2.4 视频编解码技术

本节中主要视频编解码技术进行概述，同时对视频编解码的现状进行分析，为采用 FFMPEG 技术构建视频编解码功能提供背景基础。

### 2.4.1 视频编解码技术概述

视频转码（Video Transcoding）是指将已经压缩编码的视频码流转换成另一个视频码流，以适应不同的网络带宽、不同的终端处理能力和不同的用户需求。转码本质上是一个先解码，再编码的过程，因此转换前后的码流可能遵循相同的视频编码标准，也可能不遵循相同的视频编码标准。

视频转码技术使用的目的不同，其实现的手段也各不相同。大致上可以分为两类[7]：

（1）不同编码格式之间的视频数据转码

此方法指通过转码方法改变视频数据的编码格式，这种数据转码会改变视频数据的现有码流和分辨率，如将基于 MPEG-2 格式的视频数据转换为 DV 、MPEG-4 或其它编码格式，同时根据其转码目的，指定转码产生视频数据的码流和分辨率。这种转码方式设计的算法较为复杂，涉及的算法复杂度和系统开销，是由转码所需图像质量要求及转码前后两种编码方式的相关度所决定的；

（2）相同编码格式之间的视频数据转码

此方法指不改变压缩格式，只通过转码手段改变其码流或头文件信息。根据其使用目的，可分为改变码流和不改变码流两种，如将 MPEG-2 全 I 帧 50Mbps 码流的视频数据转码为 MPEG-2 IBBP 帧 8Mbps 码流的视频数据，直接用于播出服务器用于播出。或者将基于 SONY 视频服务器头文件封装的 MPEG-2 全 I 帧 50Mbps 码流的视频文件，改变其头文件和封装形式，使之可以在给予 MATROX 板卡的编辑系统上直接编辑使用。这种转码方式的复杂度要小于不同编码格式转码的复杂度，而且对视频工程上而言，更加具有可操作性。

当前，市场上存在多种视频编码方式，如 MPEG-2，MPEG-4，H.263，H.264，等，如何改变同一格式视频内容的分辨率和码率以及实时转换不同格式的视频内容是向视频播放设备提供数字内容的关键技术，是目前一个研究热点。

## 2.5 FFMPEG 技术

本节中对 FFMPEG 技术从发展历史和技术核心内容进行深入研究，分析使用 FFMPEG 技术进行视频编解码的可行性以及使用性。

### 2.5.1 FFMPEG 技术概述

FFMPEG 是一套可以用来记录、转换数字音频、视频，并能将其转化为流的开源计算机程序。它包括了目前领先的音/视频编码库 libavcodec。 FFmpeg 是在 Linux 下开发出来的，但它可以在包括 Windows 在内的大多数操作系统中编译。可以轻易地实现多种视频格式之间的相互转换，FFMPEG 支持 MPEG, DivX, MPEG4, AC3, DV, FLV 等 40 多种编码， AVI, MPEG, OGG, Matroska, ASF 等 90

多种解码。TCPMP, VLC, Mplayer 等开源播放器都用到了 FFMPEG。FFMPEG 具有强大的音视频转码功能，而且是开源的免费软件，因此，可以在 FFMPEG 基础上进行相应的二次开发，将多种视频格式转码成 HTML5 video 标签可以播放的视频格式[8]。

### 2.5.2 FFMPEG 技术的特点及优势

多媒体视频处理工具 FFmpeg 有非常强大的功能，包括视频采集功能、视频格式转换、视频抓图、给视频加水印等[7]。

（1）视频采集功能

ffmpeg 视频采集功能非常强大，不仅可以采集视频采集卡或 USB 摄像头的图像，还可以进行屏幕录制，同时还支持以 RTP 方式将视频流传送给支持 RTSP 的流媒体服务器，支持直播应用；

（2）视频格式转换功能

ffmpeg 视频转换功能。视频格式转换，比如可以将多种视频格式转换为 flv 格式，可不是视频信号转换；

ffmpeg 可以轻易地实现多种视频格式之间的相互转换(wma,rm,avi,mod 等)，例如可以将摄录下的视频 avi 等转成现在视频网站所采用的 flv 格式；

（3）视频截图功能

对于选定的视频，截取指定时间的缩略图。视频抓图，获取静态图和动态图，不提倡抓 gif 文件;因为抓出的 gif 文件大而播放不流畅；

（4）给视频加水印功能

使用 ffmpeg 视频添加水印(logo)，提供版权保护功能。

## 2.6 本章小结

在本章节中对本次课题所要研究的基于 HTML5 视频系统和 FFMPEG 转码所要设计的相关知识和技术做一些简单介绍。

首先对本系统中所要用到的主要技术在基本概念和特点优势上做了简单说明，通过了解 HTML5 和 CSS3 技术的产生背景和发展历程的背景，并与最新的 WebService 技术相结合，通过研究技术的工作原理和核心内容来验证搭建基于 HTML5 视频系统的可能性和优势，然后对当前的视频转码技术现状进行研究，在此基础上提出 FFMPEG 转码技术的特点优势和可行性，通过以上相关技术知识的了解对接下来要进行的系统的分析设计和实现有重要作用。

# 第三章 HTML5 视频系统的系统分析

本章节对本课题要设计实现的基于 HTML5 的视频系统进行可行性分析和需求分析，通过可行性分析了解到开发本系统所使用的 HTML5 技术的可行性，其次通过需求分析明确本次课题要设计实现的系统的需求，为 HTML5 视频系统的开发奠定理论基础。

## 3.1 可行性分析

HTML5 视频系统是为实现视频资源的广泛传播和共享而建立，提供从视频上传到视频播放的一站式服务。系统的整体架构充分考虑 HTML5 的性能，并结合先进的网络传输及内容分发技术，为网络视频的发展提供了良好的空间，此视频系统主要分为两部分：

（1）视频播放平台：HTML5 与 Web Service 相结合实现视频的播放；

（2）视频管理平台：从多码率实施转码服务子系统、自适应多协议流媒体子系统、流媒体管理子系统、视频播放管理子系统四个主要子系统完成对视频从上传到播放的一站式管理。

在本章节中，就以上两个主要部分对基于 HTML5 视频系统的各个部分进行分析研究，在此基础上完成系统的具体实现。

## 3.2 功能需求分析

通过可行性分析，可知使用 HTML5 构建视频系统具有一定的技术可行性，在此基础上，提出 HTML5 视频系统的功能需求，分析系统为用户所能提供的各项功能，为实现 HTML5 视频系统提供理论依据。

### 3.2.1 系统管理平台需求分析

管理平台作为基于 HTML5 的视频系统的核心部分，实现对用户的视频进行视频上传、视频转码、视频发布、视频信息编辑等操作，根据具体情况，可支持视频批量上传，视频上传和转码进度实时监控，支持视频转码多码率和视频播放设备多选择，使经过一系列管理操作后的视频可以支持 PC、iPad、iPhone 三种设备的多码率视频播放，此管理平台采用模块化设计方法，为系统的可扩展性、

可维护性提供良好的支持。对管理平台的各个功能模块进行如下详细描述：

1）登录验证功能

（1）实现用户的登录和验证；

（2）验证成功后的用户可进入系统。

2）视频上传功能

（1）实现用户将本地视频上传到转码服务系统；

（2）对正在上传的视频实时显示上传进度，根据实时网速显示上传速度；

（3）对于待上传的视频进行排队等待，用户可通过拖拉动作对排队顺序进行调整。

3）视频转码功能

（1）对上传完毕的视频调用转码服务进行转码；

（2）对待转码的视频通过拖拉动作改变排队顺序；

（3）转码中及待转码视频可进行删除、暂停等操作。

4）视频发布功能

将转码好的视频进行发布，这样可以在视频播放平台用户就可以对此视频进行播放和共享。

5）视频列表功能

（1）将系统中的所有视频通过分页列表形式显示出来，包括视频的状态（正在转码、未转码、待发布、已发布）、视频标题、视频信息、视频缩略图等；

（2）通过可选操作对视频进行删除、编辑和快速发布等操作；

（3）在编辑页面中，可编辑视频的基本信息，包括：视频标题，视频内容，视频标签，视频播放平台等；

（4）在快速发布页面中，可预览视频的基本信息，确认无误后可将视频快速发布到播放页面，供用户观看和共享；

（5）提供多条件查询功能，管理员可通过关键字快速查询所需视频信息。

6）栏目管理功能

（1）对视频进行分类，通过树形结构显示视频栏目的层次结构；

（2）可对栏目进行查找、插入、删除、修改等基本操作，来实现对栏目层次结构的管理；

（3）可对特定视频的栏目进行修改，便于视频的属性管理。

7）皮肤设置功能

（1）支持更改视频门户的风格样式，提供两套皮肤供切换；

（2）支持更换视频门户的 logo 图片。

8）广告设置功能

（1）支持更换视频门户页面底部的三个宣传广告位的图片；

（2）支持设置广告超链接。

9）系统设置功能

（1）支持更改视频服务器的外网 IP 地址或者域名；

（2）支持修改管理员的用户名和密码；

（3）支持显示服务器运行时间，并提供重启、关闭服务器的功能；

（4）支持显示系统的日志。

### 3.2.2 视频播放平台需求分析

在基于HTML5的视频系统中，视频播放平台主要采用HTML5技术结合WebService来是实现将管理平台中用户上传的视频经过发布后直接播放，为用户提供跨平台、轻负载的高清视频播放体验，同时，还应完善用户的其他各种附加需求，包括栏目导航、最新视频排行、视频搜索、视频共享、广告位宣传等各方面需求，主要功能需求如下所述：

（1）栏目导航：显示在视频管理系统中创建的视频栏目，并作为网页的一级导航，点击栏目可在视频列表中显示该栏目下的所有视频，除“首页”栏目之外的其他栏目需要经过登录认证才可以观看；

（2）视频排行：用户可按最新发布和最热门两种方式列出前100个视频；

（3）视频搜索：可根据关键字搜索视频，搜索结果显示在视频列表中；

（4）视频列表：已分页列表的方式显示视频，支持按点击量、按发布时间、按标题正序或者倒序排列，同时记录和统计该视频的点击量；

（6）视频分享：点击分享按钮，可生成视频分享HTML代码，用于将其嵌入其他网页，同时可生成视频URL（Deep Link）并自动复制到剪贴板，供用户将该URL通过各种渠道分享给自己的好友；

（7）宣传广告位：显示视频管理系统中设置的三个宣传图片，并可点击图片链接到指定的网址。

## 3.3 性能需求分析

本课题为得到更好的用户体验，在满足用户基本功能需求的基础上，还需对系统的性能需求进行全面研究，主要包括以下方面

（1）准确性：确保视频信息准确无误；对视频内容管理正确；对视频点击排行的准确计算；对视频上传转码发布一系列操作的快速精准；对视频的搜索灵敏反应；对栏目导航的准确管理；

（2）实时性：在视频门户模块可以对视频内容实时播放；对视频排行实时刷

新；在视频管理模块可以对视频上传、转码、发布状态实时监控并更新；栏目添加删除时对视频的实时反应；

（3）健壮性：由于该视频系统支持大量视频、文字、图片等媒体元素，系统要时刻保持健壮性，对有大量视频上传转码操作时系统后台工作有序进行，保证前两者的准确性和实时性，并保证程序不会崩溃和占用大量的资源；

（4）易用性：整个系统使用方便，操作简单，以Windows风格的图形界面与用户交互；

（5）易维护性：系统的各个模块功能明确、结构清晰、接口简单、方便错误检测及性能维护；

（6）易扩展性：模块的独立性强，可以方便地对系统功能进行改动及扩展，以适应新的需求；

（7）易移植性：整个系统有很强的平台无关性，容易移植到各种不同的Windows平台中去。

## 3.4 本章小结

本章节对本课题要设计实现的基于HTML5的视频系统进行了可行性分析和需求分析，这对系统的的设计和实现是很必要的，首先，通过可行性分析了解到开发本系统所使用的HTML5技术是可行的，其次通过需求分析明确了本次课题要设计实现的系统的需求，并由此对系统的开发进度进行合理的规划和有效的调整，有利于保证系统的开发进度和遇到的问题可以及时解决。

# 第四章 HTML5 视频系统的设计

本章节中在可行性分析和需求分析基础上，提出 HTML5 视频系统的概要设计和详细设计，同时从系统的管理平台和播放平台两部分来分别介绍 HTML5 视频系统的构建，从系统的功能结构和系统的流程图角度来介绍系统所要实现的功能的大致设计思路。

## 4.1 系统管理平台的概要设计

本节中在需求分析的基础上，介绍 HTML5 视频系统管理平台的概要设计，从管理平台的总体设计，流程图和功能结构来概述管理平台的设计，为系统的详细设计奠定基础。

### 4.1.1 系统管理平台的总体设计

基于 HTML5 的视频系统的管理平台主要采用 JAVA、JSP、Struts、Spring、Hibernate、MySql 来开发实现用户视频的上传，转码，编辑，发布等一系列操作，在视频发布到播放平台供用户观看共享前，对视频的集中管理。总体结构如图所示：

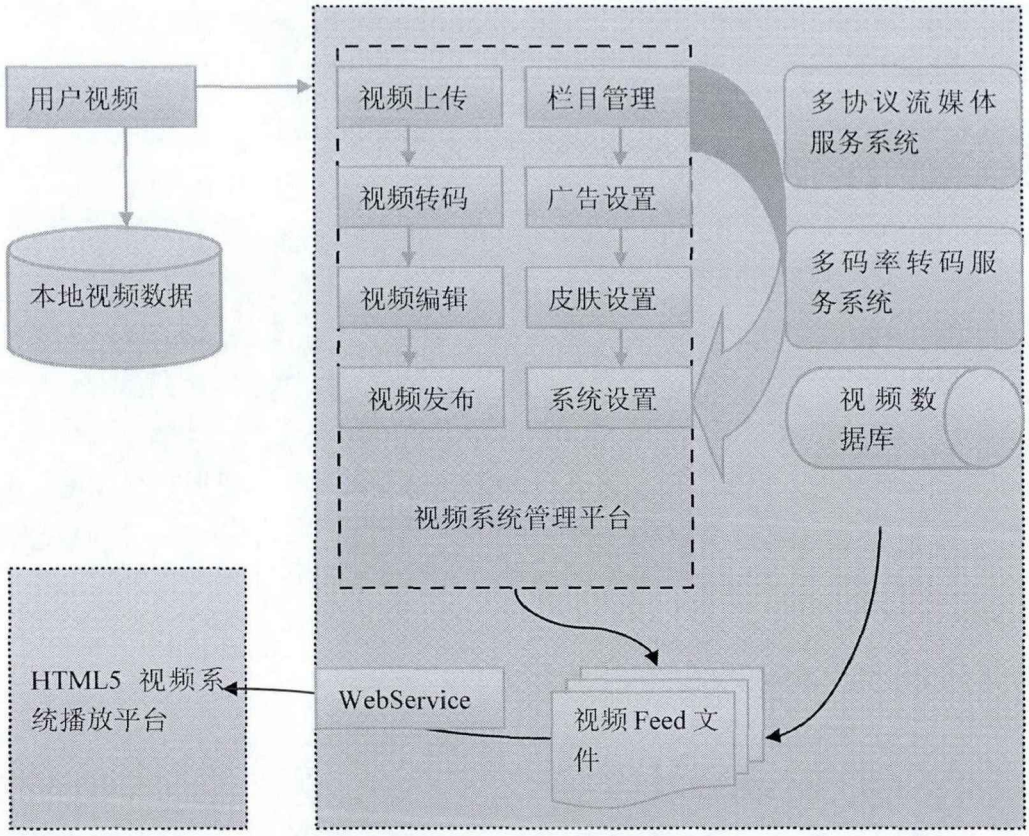


**图 4-1 管理平台总体结构图**

在基于 HTML5 的视频系统中，管理平台是其最核心的部分，由上述总体结构图可知，管理平台可实现视频系统的最主要功能，并结合最新的多协议流媒体服务系统和多码率转码服务系统，通过 Web Service 技术奖视频信息转成 XML 或 JSOM 类型的视频 Feed 文件，使得视频播放平台可以流畅高清的播放各种视频。

### 4.1.2 系统管理平台的功能结构

根据具体的需求分析和总体结构分析，可知基于 HTML5 的视频系统的管理平台的功能结构如下图所示：

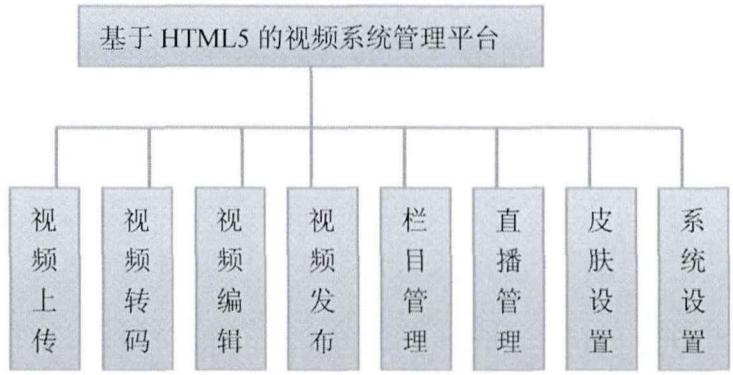


**图 4-2 管理平台总体功能图**

上图中显示了管理平台的总体功能图，通过管理平台对用户视频进行上传，转码，编目，发布等操作，可使用户在播放平台观看和共享视频，同时可对系统进行皮肤设置，广告设置，系统管理等功能。各功能模块介绍如下：

1）登录验证功能

（1）实现用户的登录和验证；

（2）验证成功后的用户可进入系统。

2）视频上传功能

（1）实现用户将本地视频上传到转码服务系统；

（2）对正在上传的视频实时显示上传进度，根据实时网速显示上传速度；

（3）对于待上传的视频进行排队等待，用户可通过拖拉动作对排队顺序进行调整。

3）视频转码功能

（1）对上传完毕的视频调用转码服务进行转码；

（2）对待转码的视频通过拖拉动作改变排队顺序；

（3）转码中及待转码视频可进行删除、暂停等操作。

4）视频发布功能

将转码好的视频进行发布，这样可以在视频播放平台用户就可以对此视频进行播放和共享。

5）视频列表功能

（1）将系统中的所有视频通过分页列表形式显示出来，包括视频的状态（正在转码、未转码、待发布、已发布）、视频标题、视频基本信息、视频缩略图等；

（2）通过可选操作对视频进行删除、编辑和快速发布等操作；

（3）在编辑页面中，可编辑视频的基本信息，包括：视频标题，视频内容，视频标签，视频播放平台等；

（4）在快速发布页面中，可预览视频的基本信息，确认无误后可将视频

快速发布到播放页面，供用户观看和共享；

（5）提供多条件查询功能，管理员可通过关键字快速查询所需视频信息。

6）栏目管理功能

（1）对视频进行分类，通过树形结构显示视频栏目的层次结构；

（2）可对栏目进行查找、插入、删除、修改等基本操作，来实现对栏目层次结构的管理；

（3）可对特定视频的栏目进行修改，便于视频的属性管理。

7）皮肤设置功能

（1）支持更改视频门户的风格样式，提供两套皮肤供切换；

（2）支持更换视频门户的 logo 图片。

8）广告设置功能

（1）支持更换视频门户页面底部的三个宣传广告位的图片；

（2）支持设置广告超链接。

9）系统设置功能

（1）支持更改视频服务器的外网 IP 地址或者域名；

（2）支持修改管理员的用户名和密码；

（3）支持显示服务器运行时间，并提供重启、关闭服务器的功能；

（4）支持显示系统的日志，如服务器何时启动、管理员何时登录等。

### 4.1.3 系统管理平台的流程图

根据系统的需求分析中的划分的系统功能可知，在基于 HTML5 的视频系统中管理平台主要提供九部分功能，主要对视频的上传、转码、编目、信息管理等提供一站式全面管理，在管理平台的整体数据流程图中主要描述视频数据在管理平台中的走向，通过视频信息的编辑和管理，实现对用户视频的系统化和模块化，使得用户在播放平台可以很方便的对自己的视频进行高清播放和共享，其整体的数据流程图如下所示：

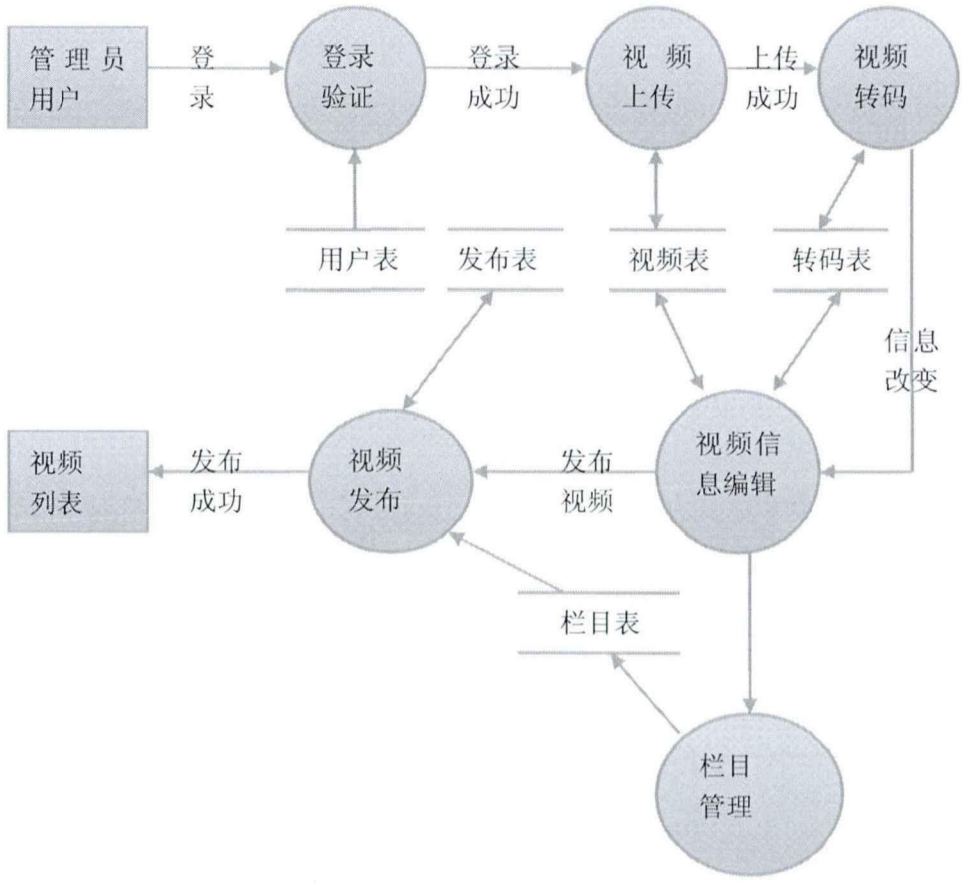


**图 4-3 管理平台数据流程图**

由上述数据流程图可知，在基于 HTML5 视频系统中，管理平台通过对视频信息的一系列编辑和操作实现了对视频从上传、转码到视频发布的一站式管理，便于用户在系统播放平台中实现对视频快捷高清播放。

## 4.2 系统播放平台的概要设计

本节在系统播放平台需求分析的基础上提出概要设计，从播放平台功能结构对视频系统播放平台进行概要设计，为 HTML5 视频系统的播放平台的详细设计提供依据。

### 4.2.1 系统播放平台的功能结构

视频播放客户端主要采用 HTML5、CSS、Javascript 与 WebService 相结合来实现对视频的播放，为用户提供高清，便捷的视频播放体验。

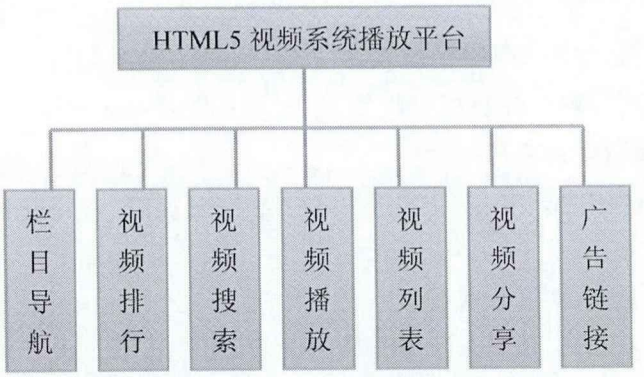


**图 4-4 播放平台功能结构**

传统视频播放采用FLASH插件，在此系统中采用HTML5的音视频标签video和audio标签，同时视频管理平台中经过转码，编目，编辑等一系列操作后的视频数据转化成xml格式通过使用 WebService 传递到客户端，供用户播放，完成读取视频、控制视频播放（一般的播放准备完成、播放、停止、完成播放事件的编程）和响应用户操作。

## 4.3 系统管理平台的详细设计

本节在系统管理平台概要设计的基础上，提出更全面的详细设计，分别从管理平台功能模块设计和数据结构设计来对视频系统管理平台进行详细设计，为HTML5视频系统管理平台的具体实现提供依据。

### 4.3.1 系统管理平台功能模块设计

基于HTML5视频系统的管理平台实现系统的核心功能，为使用户可在播放平台上享受跨平台、多设备高清播放、清晰流畅的视频观看体验，要实现对视频的基本管理，因此系统管理平台功能模块在实现基本功能操作的同时还要满足以下详细设计需求，描述如下：

（1）视频上传：上传视频模块对视频的格式选择需要过滤，并且显示正确的上传进度，包括百分比以及视频大小，当取消上传时可正确处理；

（2）视频转码：上传好的视频自动进入转码队列，在后台系统中有条不紊的处理转码进度，正确显示转码状态，转码过程中可删除，转码队列可排序等；

（3）视频列表：在视频列表中可对视频进行编辑，预览，发布，选择栏目等操作。实现这些快速编辑时需要快速准确，并实时保存更新状态；

（4）视频发布：发布的视频可直接在视频门户中直接点击观看，发布的视频也可取消发布成未发布状态；

（5）栏目管理：栏目管理可实时添加修改，实现树形结构模式。可在视频处选择栏目添加，也可在栏目块添加视频。同一栏目下的视频可移动排序；

（6）广告设置：按照指定大小上传广告图片，同时正确设置广告链接地址；

（7）皮肤设置：按照指定配置文件格式设置皮肤图片和皮肤路径，按照指定大小上传LOGO图片；

（8）系统管理：正确设置系统IP，系统配置信息，同时指定日志文件路径。

### 4.3.2 系统管理平台数据设计

在HTMl5视频系统的管理平台中，有多个数据表参与，因此本节中对管理平台中的用到的数据表对输入项的特性：包括名称、标识、数据的类型和格式、数据值的有效范围、输入的方式进行量度，表的详细信息描述如下：

1） 用户表：存放各级用户管理员信息，包括用户的基本信息，操作权限，所属分组，操作角色等信息。

**表4-1 视频表t_user**

| 名称 | 标识符 | 数据类型 | 数据长度 | 备注 |
|---|---|---|---|---|
| ID 号 | USER_ID | integer | 20 | 自增长 |
| 名称 | USERNAME | varchar | 100 | NOT NULL |
| 密码 | USER_PASSWORD | varchar | 100 | NOT NULL |
| 邮箱 | USER_EMAIL | varchar | 100 | Email 格式 |
| 登录日期 | LOGIN_DATE | Date | | yyyy-mm-dd:hh mm ss |
| 登录状态 | LOGIN_STATUS | varchar | 100 | 记录登录状态 |
| 角色 | ROLE_ID | varchar | 100 | 外键 |

2） 角色表：存放用户角色信息，包括：角色名称，角色类型，角色描述等信息。

**表4-2 角色表role_type**

| 名称 | 标识符 | 数据类型 | 数据长度 | 备注 |
|---|---|---|---|---|
| ID | ROLE_ID | integer | | NOT NULL |
| 类型 | ROLE_TYPE | varchar | 100 | |
| 名称 | ROLE_NAME | varchar | 100 | |
| 描述 | ROLE_DESC | varchar | 1000 | |

| | | | | |
|---|---|---|---|---|
| 优先级 | DEFAULT_VALUE | integer | | 用数值表示 |
| 权限 | PERMISSION | varchar | | 不同权限 |

3） 视频表，用于存放视频信息，包括视频编号，视频标题，视频内容，视频详细信息。视频时长。视频大小，视频上传日期，视频转码状态，视频发布状态等信息。

**表4-3 视频表t_program**

| 名称 | 标识符 | 数据类型 | 数据长度 | 备注 |
|---|---|---|---|---|
| ID 号 | ID | integer | | NOT NULL |
| 名称 | PROG_NAME | varchar | 100 | NOT NULL |
| 类别 | PROG_TYPE | varchar | 100 | FORMATE |
| 详细信息 | PROG_CONTENTt | varchar | 1000 | |
| 时长 | PROG_LENGTH | varchar | 100 | |
| 大小 | PROG_SIZE | double | 100 | MB |
| 上传日期 | PROG_DATE | date | | |
| 路径 | PROG_PATH | varchar | 200 | NOT NULL |
| 是否发布 | PROG_PUBLISH | boolen | | |

4） 栏目表，存放栏目的信息，包括栏目ID，栏目标题，关键字，栏目层级等信息。

**表4-4 栏目表category**

| 名称 | 标识符 | 数据类型 | 数据长度 | 备注 |
|---|---|---|---|---|
| ID | SUBJECT_ID | integer | | NOT NULL |
| 标题 | SEO_TITLE | varchar | 100 | |
| 描述 | SEO_DEC | varchar | 1000 | |
| 关键字 | SEO_KEYWORDS | varchar | 100 | |
| 栏目排名 | RANK | integer | | 栏目层级 |

5） 发布表：存放发布视频信息，包括：视频ID，发布时间，发布状态，发布者等信息。

**表4-5 发布表t_publish**

| 名称 | 标识符 | 数据类型 | 数据长度 | 备注 |
|---|---|---|---|---|
| ID | PROG_ID | integer | | NOT NULL |
| 平台 | PLAFFORM_TYPE | varchar | 100 | |
| 发布日期 | PUB_DATE | Date | | |
| 角色 | ROLE_ID | integer | 100 | |
| 发布者 | USER_ID | integer | | |
| 栏目 | SUBJECT_ID | integer | | |

6） 计数表，统计视频播放的次数，按照各时间段统计。

**表4-6 计数表v_video_count**

| 名称 | 标识符 | 数据类型 | 数据长度 | 备注 |
|---|---|---|---|---|
| 视频 ID | id | integer | | |
| 视频计数 | Count | integer | | 总播放次数 |
| 按天计数 | Count_day | integer | | 次数/天 |
| 按周计数 | Count_week | integer | | 次数/周 |
| 按月计数 | Count_month | integer | | 次数/月 |
| 更新日期 | Update_time | date | | |

## 4.4 系统播放平台的详细设计

本节在系统播放平台概要设计的基础上，提出更全面的详细设计，分别从播放平台功能模块设计和数据结构来对视频系统播放平台进行详细设计，为 HTML5 视频系统的播放平台实现提供依据。

### 4.4.1 系统播放平台功能模块设计

视频播放客户端主要采用 HTML5、CSS、Javascript 与 WebService 相结合来

实现对视频的播放，为用户提供高清，便捷的视频播放体验。

传统视频播放采用 FLASH 插件，在此系统中采用 HTML5 的音视频标签 video 和 audio 标签，同时视频管理平台中经过转码，编目，编辑等一系列操作后的视频数据转化成 xml 格式通过使用 WebService 传递到客户端，供用户播放，完成读取视频、控制视频播放（一般的播放准备完成、播放、停止、完成播放事件的编程）和响应用户操作。

### 4.4.2 系统播放平台数据结构

采用 WebService 将系统的视频信息以 XML 和 JSON 类型传输到播放平台页面，以供用户观看和分享视频，因此视频播放平台数据结构为 XML 和 JSON 类型，如下图所示：

```
<?xml version="1.0" encoding="UTF-8" ?>
- <Latest xmlns:atom="http://www.w3.org/2005/Atom" xmlns:georss="http://www.georss.org/georss" xmlns:media="http://search.yahoo.com/mrss/"
  xmlns:nl="http://www.neulion.com/iptv/pc/rss" xmlns:twitter="http://api.twitter.com">
  - <News>
    - <NewsItem BigHref="http://www.nba.de/photos/mobile_0310-erik-spoelstra-608.jpg" SmallHref="http://www.nba.de/photos/mobile_0310-erik-spoel
      300.jpg">
      - <Identification>
        - <NewsIdentifier>
          <ProviderId>NBA.com</ProviderId>
          <DateId>20110310</DateId>
          <NewsItemId>/2011/news/features/fran_blinebury/03/10/erik-spoelstra-miami-heat-future/index.html</NewsItemId>
          <RevisionId PreviousRevision="0" Update="N">1</RevisionId>
```

**图 4-5 播放平台数据结构 XML 类型**

```
{ "adaptive" : true,
  "available" : true,
  "bigImage" : "http://localhost:80/image/43_eb.jpg",
  "clicks" : 0,
  "countDay" : 0,
  "countMonth" : 0,
  "countWeek" : 0,
  "currency" : "USD",
  "description" : "请输入描述文字",
  "duration" : 30,
  "episodeId" : 39,
  "fileName" : "",
  "flvFormat" : 1,
  "groupId" : 1000,
  "image" : "http://localhost:80/image/43_es.jpg",
  "mediaType" : "flvas",
  "multiChnDesc" : "",
  "multiEngDesc" : "",
  "name" : "2.wmv",
  "operId" : 0,
  "path" : "adaptive://localhost:80/vod/upload/20120725/neuvision/d450374ba42649fba2cb3af610b0e7a2_pc.mp4",
  "pathIpad" : "http://localhost:80/vod/upload/20120725/neuvision/d450374ba42649fba2cb3af610b0e7a2_ipad.mp4.m3u8",
  "pathIphone" : "http://localhost:80/vod/upload/20120725/neuvision/d450374ba42649fba2cb3af610b0e7a2_iphone.mp4.m3u8",
  "platformType" : "PC",
  "progDate" : "2012-07-25T11:25:00+0800",
  "progEpisodes" : 1,
  "progPrice" : 0,
  "progType" : "VOD",
  "programId" : 43,
  "refer" : true,
  "seq" : 1,
  "shareFlag" : true,
  "starLevel" : 0,
  "tags" : "",
  "uniqueId" : "3ff22920-9650-3ddc-0c9b-a339ac1002b3",
  "updateTime" : "2012-07-31T22:08:54+0800"
}
```

**图 4-6 播放平台数据结构 JSON 类型**

在 HTML5 视频系统播放平台中，XML 和 JSOM 类型的视频信息通过 WebService 传输到页面，用户通过简单操作就可观看和分享视频。

## 4.5 本章小结

本章节是介绍基于 HTML5 视频系统的概要设计和详细设计，分别从系统的管理平台和播放平台两部分来分别介绍，然后从系统的功能结构和系统的流程图角度来介绍系统所要实现的功能的大致设计思路，这章的概要设计和详细为下一章的系统实现奠定了基础。

# 第五章 HTML5 视频系统的实现

在本章中，在 HTML5 视频系统需求分析和概要设计的基础上提出系统的具体实现，从系统的管理平台和播放平台分别介绍系统的具体功能实现，同时详细描述实现过程遇到的问题以及解决办法。

## 5.1 HTML5 视频系统管理平台的实现

HTML5 视频系统管理平台是系统中核心的部分，主要采用 JAVA/JSP 结合 MySQL 数据库实现管理平台视频上传、视频转码、视频列表、视频发布、栏目管理、广告和皮肤设置、系统设置等一站式视频管理，使用户可以在播放平台实现流畅、清晰的视频播放体验。

### 5.1.1 用户登录模块实现

在用户登录模块，提供管理员登录界面，只有登录后才能进入视频管理系统。同时此管理平台支持三个管理员帐号，一期不区分三个管理员的权限。用户可以根据不同功能需求选择不同账号进行登录，对视频进行集中管理。实现界面如下所示：

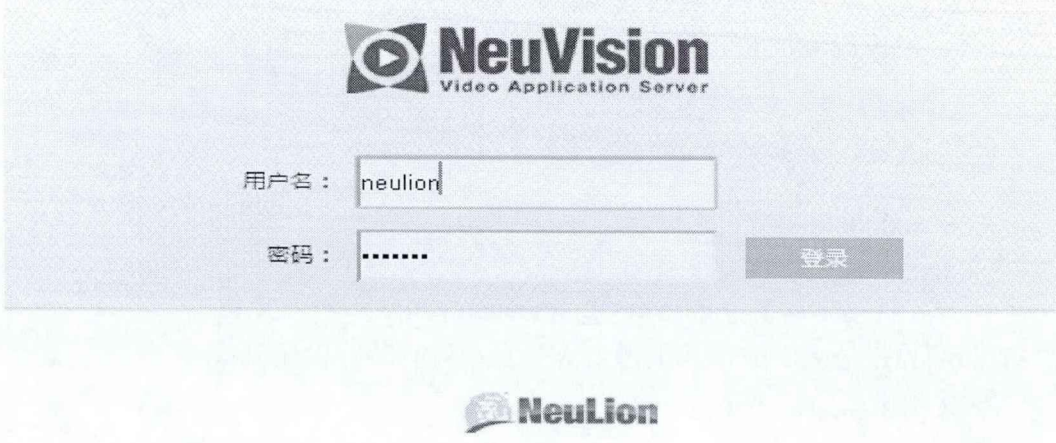


**图 5-1 登录页面**

具体代码实现如下图所示：

```
public static UserLogin login(HttpServletRequest request, HttpServletResponse response)
{
    UserLogin ret = null;
    BeanManager bm = BeanManagerFactory.getBeanManager();
    AuthenticationService service = (AuthenticationService)bm.getBean("service.Authentication");
    try
    {
        Map<String, String> m = new HashMap<String, String>();
        m.put(AuthenticationService.CONTEXT_CLIENT_IP_PARAM, request.getRemoteAddr());
        String username = request.getParameter("username");
        String password = request.getParameter("password");
        SecurityContext ctx = service.authenticate(username, password, m);
        if(ctx==null)
        {
            request.setAttribute(Constants.RETURN_CODE, UserConstants.LOGIN_FAILED);
        }
        else
        {
            ret = (UserLogin)ctx.getPrincipal();
            PartyService pservice = (PartyService)bm.getBean("service.Party");
            getEmployeeCompany(pservice,ret.getPartyId().intValue());
            request.setAttribute(Constants.RETURN_CODE, UserConstants.LOGIN_SUCCESS);

        }
    }
    catch(AccountDisabledException ex)
    {
        request.setAttribute(Constants.RETURN_CODE, UserConstants.LOGIN_DISABLED);
    }
```

**图 5-2 登录代码**

### 5.1.2 视频上传模块实现

视频上传模块支持批量上传视频文件，正在上传的文件显示上传的进度，等待上传的文件支持排序和取消上传，同时支持 avi、wmv、mp4、mkv 等主流视频格式；上传成功的视频自动进入转码状态，并被转码成支持 PC、iPad、iPhone 三种设备的多码率视频文件。具体实现界面如图所示：


**图 5-3 视频上传**

视频上传：使用$("#uploader").juploader({})；实现插入 flex 编写的上传 swf 文件 juploader.js 来完成视频上传功能和自动排队等待上传功能。

具体代码实现如下所示：

```
$(document).ready(function(e) {
  $(".ui-widget-overlay").hide();
  $(".dialog").hide();
  $("#uploader").juploader({
     debug:true,
     width:60,
     height:50,
     uploadType:dectectUploadType(),
     serverUrl: "${serverUrl}",
     swfUploadUrl:"${context}/static/ui/images/Uploader.swf",
     params:{
      "cpId":"Neuvision",
      "cpKey":"v3"
     },
     uploadImage:'${context}/static/ui/images/btn_upload.png',
     uploadImageHover:'${context}/static/ui/images/btn_upload_mo.png',
     onNewUpload:function(params){
       var devices=[];
       devices.push("pc");
       devices.push("ipad");
       devices.push("iphone");
       devices.push("android");
       params["devices"]= devices;
       params["format"]="json";
       return params;
     },
```

**图 5-4 实现代码**

### 5.1.3 视频列表模块实现

视频列表模块中主要以分页列表的形式列出系统中的所有视频节目的状态（已发布、待发布、正在转码、未转码）、缩略图、标题等信息，同时针对列表中选中的视频可进行编辑、删除和快速发布操作，在编辑页面中可编辑视频的标题、描述等文本信息；可用打钩的方式设置节目的标签，并可以对便签列表进行维护管理；可设置视频所属的栏目，支持一个视频同时属于多个栏目；可设置视频节目的缩略图、支持从视频中截图和手动上传图片；可以勾选是否立即发布，同时可删除选中的视频节目，针对选中的视频，可以在视频列表右侧选择快速发布，发布之前可以快速编辑视频的标题和描述。


**图 5-5 视频列表**

在视频列表中，以分页列表形式显示系统中的视频，同时可显示视频的标题，视频状态（转码，待转码，发布，待发布），视频内容，更新日期，同时可对视频进行快速编辑，快速发布，快速删除操作，当点击编辑按钮时候，可跳转到编辑页面，对视频信息进行详尽的编辑和修改，具体代码实现如下图所示：

```
//默认显示第一个视频的信息，快速发布后默认显示当前发布的视频
if (index == 0)
{
    if ($.trim($("#programId").val()).length == 0)
        $("#programId").val($(this).children("td:last").html());

    programId = $("#programId").val();
    getProgramInfo(programId);
    selectTR(programId);
    //getProgramInfo(programId);
}
.hover(function()
{   var get_status = $(this).children("td:eq(5)").html();
    $(this).find(".video_button").show();
    if($(this).children("td:eq(0)").attr('class')=="ui-table-td")
    {
        if(get_status=="Published")
        {row_sel_click=true;
           $(this).find("td:first").find("div").removeClass("status_published");
           $(this).find("td:first").find("div").addClass("status_published_1");
        }
        if(get_status=="Encoded")
        {
            row_sel_click=true;
            $(this).children("td:first").find("div").removeClass("status_encoded");
```

**图 5-6 视频列表代码实现**

### 5.1.4 栏目管理模块实现

栏目管理模块实现管理视频的栏目，可方便的创建、修改和删除视频栏目，同时支持三级栏目，管理员可根据需求对栏目进行各种管理。

具体界面实现如下图所示：


**图 5-7 栏目管理**

在栏目管理模块中，栏目的层级结构通过树形插件来实现，来简洁明了的显示系统中视频分类情况，同时可对栏目层级结构进行快速编辑，可进行简单的查找、删除、插入、修改等操作，同时可快速对视频进行分类，点击分类后，增加视频，即可实现将视频分类的操作，具体代码实现如下图所示：

```
fnShowProgress();
setTimeout(function(){
    var t = Math.random();
    var request = $.ajax({
        type:"get",
        url:"categoryaction?t="+t,
        data:dataMap,
        dataType:"html",
        success:function(data)
        {
            hideProgress();
            $(".program_list").html(data);
            var pageSize = $("#pageSize").val();
            var pageIndex = $("#pageIndex").val();
            var totalCount = $("#totalCount").val();
            var pageHtml= page({"indexPage":pageIndex,"pageSize":pageSize,totalCou
            $("#pageInfo").html(pageHtml);
        },
        error:function(xhRequest, errorText, thrownError)
        {
            //alert(xhRequest.status + "\n" + errorText);
            alert(document.getElementById("categorytree_show_failed").innerHTML);
            hideProgress();
        }
});
}, 500);
```

**图 5-8 栏目管理层级结构实现**

### 5.1.5 视频发布模块实现

视频发布模块中可以显示待发布视频列表和已发布视频列表，同时提供批量

勾选视频节目进行发布和取消发，并且提供搜索功能，管理员可设置多条件查询所需视频，具体界面实现如下图所示：

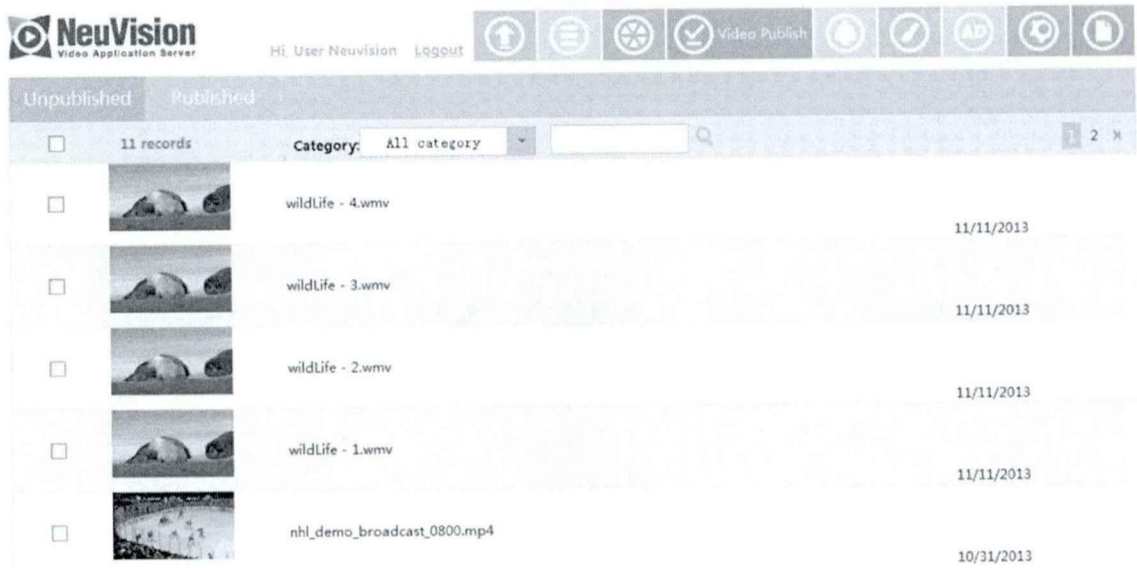


**图 5-9 视频发布**

在视频发布模块中，当视频发布后，在播放平台中，用户才可以观看和分享此视频，具体代码实现如下所示：

```
var zNodes = ${categories};

function onClick(e, treeId, treeNode) {
    $("#citySel").val(treeNode.name);
    $("#categoryId1").val(treeNode.id);

    hideMenu();
    window.location = "?action=list&clearSession=true&categoryId1=" + tr
}
function onClick2(e, treeId, treeNode) {
    $("#citySel2").val(treeNode.name);
    $("#categoryId2").val(treeNode.id);

    hideMenu2();
    window.location = "?action=list&clearSession=true&categoryId2=" + tr
}

var shown = false;
function showMenu() {
    if(!shown)
    {
        var cityObj = $("#citySel");
        var cityOffset = $("#citySel").offset();
        $("#menuContent").css({left:cityOffset.left + "px", top:cityOffs
        $("body").bind("mousedown", onBodyDown);
        shown = true;
```

**图 5-10 视频发布代码实现**

### 5.1.6 皮肤和广告设置模块实现

在皮肤和广告设置模块中支持更改视频门户的风格样式，提供两套皮肤供切换。同时支持更换视频门户的 logo 图片。并且支持更换视频门户页面底部的三个宣传广告位的图片，并设置超链接


**图 5-11 广告和皮肤设置**

在广告和皮肤设置模块中，对播放平台的背景和三个宣传广告位进行图片和链接设置。具体代码实现如下图所示：

```
List<Map<String, String>> themes =SkinGet.getThemes(domain);
request.setAttribute("themes", themes);//skin's name
request.setAttribute("imagePath", imagePath);
request.setAttribute("previewPath", previewPath);
request.setAttribute("attributes", getAllWebsiteAttributes(groupId));

List type=ServiceFactory.getGrabImageService().getFileExtents();
String imageType="*."+(String) type.get(0);
for(int i=1;i<type.size();i++)
    imageType=imageType+";*."+(String) type.get(i);
request.setAttribute("imageType", imageType);

UserService userService = ServiceFactory.getUserService();;
BeanManager bm = BeanManagerFactory.getBeanManager();
UserLogin login = getUser(bm);
request.setAttribute("password", userService.decryptUserPassword(login.getPassword()));
request.setAttribute("username", login.getName());

return JSP_LOCATION + "components/ad.jsp";
```

**图 5-12 广告和皮肤设置代码实现**

在服务器硬盘中存储皮肤图片和配置文件，代码中读取正确的皮肤文件路径，实现皮肤设置，管理员按照指定大小上传广告图片，并设置正确的链接地址，实现播放平台广告宣传位的设置。

### 5.1.7 系统设置和系统状态日志模块实现

在系统设置和系统状态日志模块中管理员可更改NeuVision视频服务器的外网 IP 地址或者域名，同时可修改管理员的用户名和密码，在日志中可以设置显

示服务器运行了多长时间，并提供重启、关闭服务器的功能。


**图 5-13 系统设置**

具体代码实现如下图所示：

```
ccuri = TextUtil.isEmpty(request.getParameter("cc")) ? config.getServiceUrl() + ccuri :
StringBuffer sb = new StringBuffer("<?xml version=\"1.0\" encoding=\"UTF-8\"?>");
sb.append("<configDomain>");
sb.append("<cpId>" + config.getCpId() + "</cpId>");
sb.append("<cpKey>" + config.getCpKey() + "</cpKey>");
sb.append("<domain>" + domain + ":" + SERVER_PORT + "</domain>");
sb.append("<foramt>xml</foramt>");
sb.append("</configDomain>");

String postString = sb.toString();
try
{
    (new HttpUtils()).postUrlContent(ccuri + "?", postString);
    //Document doc = XmlUtil.createDocumentByContent(html);
    //cc_result = XmlUtil.getInnerText(XmlUtil.getNode(doc, "result.code"));
} catch (Exception e)
{
    _LOGGER.error("Call CC 'configDomain' failed. URI=" + ccuri + "?" + postString);
    _LOGGER.error(e);
    }
    request.setAttribute("result", result ? "success" : "");
if(request.getParameterValues("cdnserver[]")!=null)
{
    String[] cdnserver =new String[request.getParameterValues("cdnserver[]").length];
    cdnserver = request.getParameterValues("cdnserver[]");
    String [] cdnprefix =new String[request.getParameterValues("cdnprefix[]").length];
```

**图 5-14 系统设置代码实现**

在系统设置模块中，管理员可以设置系统运行的 IP 地址，同时可快速获得系统当前的运行状态，通过查看日志文件可知系统各个功能模块当前正在执行的任务。

## 5.2 HTML5 视频系统播放平台的实现

视频播放客户端主要采用 HTML5、CSS、Javascript 与 WebService 相结合来实现对视频的播放，为用户提供高清，便捷的视频播放体验。

传统视频播放采用 FLASH 插件，在此系统中采用 HTML5 的音视频标签 video 和 audio 标签，同时视频管理平台中经过转码，编目，编辑等一系列操作后的视频数据转化成 xml 格式通过使用 WebService 传递到客户端，供用户播放，完成读取视频、控制视频播放（一般的播放准备完成、播放、停止、完成播放事件的编程）和响应用户操作，如下所示：


**图 5-15 基于 HTML5 的视频系统播放平台**

视频播放客户端主要实现以下功能：

1）登录认证

支持 LDAP 帐户的登录认证。

2）栏目导航

（1）显示在视频管理系统中创建的视频栏目，并作为网页的一级导航；

（2）一期最多支持三级栏目。

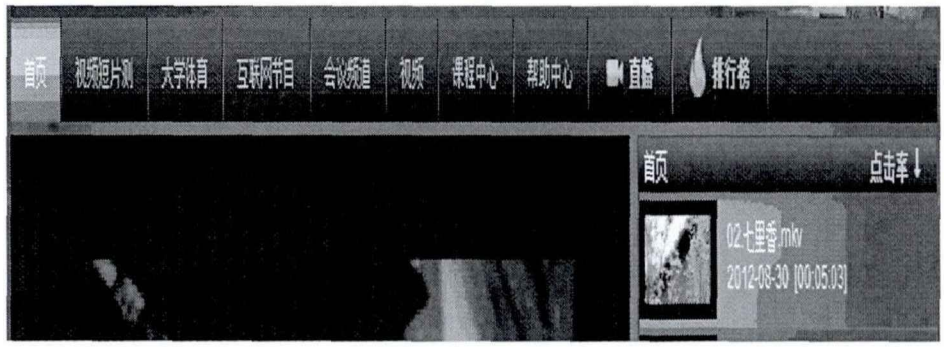


**图 5-16 栏目导航**

3）视频搜索

可根据关键字搜索视频，在视频列表中显示搜索结果。

4）视频列表和视频排行

（1）用户可按照最新发布和最热门两种方式列出前 100 个视频；

（2）支持按点击量、按发布时间、按标题正序或者倒序排列，默认按时间倒序排列；

（3）点击列表中的视频，记录和统计该视频的点击量。


**图 5-17 视频列表和视频排行**

5）视频播放器

（1）采用 HTML5 的多媒体元素：video 和 audio 标签实现视频的高清流畅播放；

（2）支持基本的视频播放控制、音量控制、全屏播放等视频播放器功能。

**图 5-18 视频播放器**

6）视频信息

（1）显示正在播放的视频的标题、描述、发布时间等信息；

（2）显示正在播放的视频的标签，并且点击标签上的超链接，可以按该标签的关键字进行搜索；

（3）显示视频的点击量；

（4）可生成视频分享 HTML 代码，用于将其嵌入其他网页，并在这些网页中生成视频播放器，可生成视频 URL（Deep Link）并自动复制到剪贴板，供用户将该 URL 通过各种渠道分享给自己的好友。

7）宣传广告位

显示视频管理系统中设置的三个宣传图片，并可点击图片链接到指定的网址。

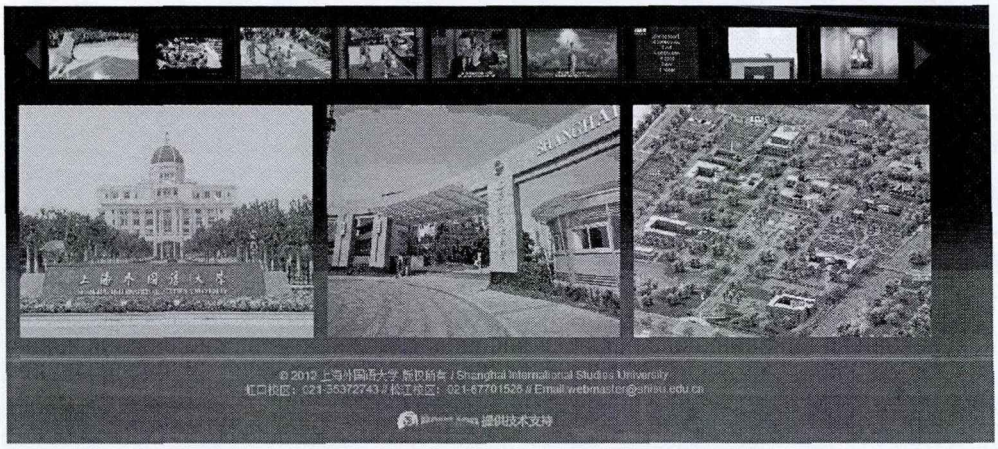

**图 5-19 视频广告位**

使用 Web Service 获得视频播放器数据，代码如下：

```
<%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8" errorPage="error.jsp"%><%@
include file="_common.jsp"%><%@
page import="net.sf.json.*,java.io.*,java.net.*,java.util.*,org.apache.commons.logging.*,java.text.SimpleDateFormat"%><%!

private final Log log = LogFactory.getLog(getClass());
//data provider url //relarvie self
String rootUrl = "http://127.0.0.1";
private String configUrl = rootUrl+"/video-portal/resources/VideoPortalRS/getPortalConfig?client=NeuVision&format=json";
//this is for simple cache
private static JSONObject _configuration = null;

public  JSONObject initConfiguration(HttpServletRequest request){
   if (request.getParameter("_clearCache")!=null){
      _configuration = null;
   }
   _configuration = null;
   if (_configuration==null){
     try {
        String url = configUrl+"&time="+System.currentTimeMillis();
        if (url.indexOf("http://")!=0){
           url = "http://"+request.getHeader("HOST")+"/"+url;
        }
        String content = loadUrlContent(url,request);
        _configuration = JSONObject.fromObject(content);

     } catch (Exception e) {
         System.out.println("can't load config from url=" + configUrl);
         _configuration = null;
     }
   }
   return _configuration;
}
```

图 5-20 Web Service 传输数据

在视频播放平台采用 Web Service 获得视频播放器数据，以供用户播放视频，数据的传输格式为 XML，在传输过程可保证安全性和快速传输

## 5.3 本章小结

在本章中对系统的管理平台和播放平台实现做了详细介绍，并通过系统截图和代码展示详细的展现了系统的实现过程，并详细描述了系统实现过程中遇到的问题，不断更新完善系统的功能。

# 第六章 FFMPEG转码在视频系统中的应用

本章节主要讨论将 FFMPEG 转码技术应用到 HTML5 视频系统中，首先对 FFMPEG 技术做了详细介绍，在 FFMPEG 源码深入分析的基础上，具体研究使用 FFMPEG 技术对视频进行编解码操作，并将此功能应用到 HTML5 视频系统中。

## 6.1 FFMPEG 技术

FFmpeg 是一套可以用来记录、转换数字音频、视频，并能将其转化为流的开源计算机程序。它包括了目前领先的音/视频编码库 libavcodec。 FFmpeg 是在 Linux 下开发出来的，但它可以在包括 Windows 在内的大多数操作系统中编译。可以轻易地实现多种视频格式之间的相互转换，FFmpeg 支持 MPEG, DivX, MPEG4, AC3, DV, FLV 等 40 多种编码，AVI, MPEG, OGG, Matroska, ASF 等 90 多种解码。FFmpeg 具有强大的音视频转码功能，而且是开源的免费软件，因此，可以在 FFmpeg 基础上进行相应的二次开发，对多种视频格式进行转码[4]。FFmpeg 的基本框架如下图所示：

**表 6-1 FFMPEG 基本框架**

| 基本模块 | 功能描述 | 其他属性 |
| --- | --- | --- |
| FFmpeg | 视频文件转换命令行工具 | 也支持经过实时电视卡抓取和编码成视频文件 |
| ffserver | 基于 HTTP, RTSP 用于实时广播的多媒体服务器 | 也支持时间平移 |
| ffplay | 一个简单的媒体播放器 | 用 SDL 和 FFmpeg 库开发的 |

| | | |
|---|---|---|
| libavcodec | 包含所有FFMPEG音视频编解码器的库 | 为了保证最优性能和高可复用性，大多数编解码器从头开发的 |
| libavutil | 存放常用算法工具函数库 | 包含一些公共的工具函数 |
| libavformat | 包含所有的普通音视格式的解析器和产生器的库 | 用于各种音视频封装格式的生成和解析，包括获取解码所需信息以生成解码上下文结构和读取音视频帧等功能。 |

## 6.2 搭建 Windows 编译环境

由于FFmpeg是基于Linux开发的开源项目，源代码和Windows下最常见的Visual Studio提供的C/C++编译器不兼容，因此它不能使用MSVC++编译。要想使用FFmpeg，最先要解决的问题就是在Windows下配置一个类似Linux的编译环境，将FFmpeg编译为二进制库以后，再利用其进行进一步开发。MSVC++并不严格的遵循C标准，所以整个编译过程必须使用MSys+MinGW系统来完成[8]。

1）需求：

（1）Windows XP，虽然FFMPEG最早在Linux系统下开发，但Windows版本已出现，可供在市场占有率更高的Windows系统进行开发；

（2）可用网络，需要使用NPM在线安装所需的库，所以网络为必需。

2）安装MinGW+MSYS：

（1）从官方网站下载文件：mingw，点击文件，进行MinGW安装操作；

（2）选择安装路径为C:\MinGW，在安装过程中选择同时安装MSYS Basic System，可同时安装MSYS，则MSYS不需另外安装。

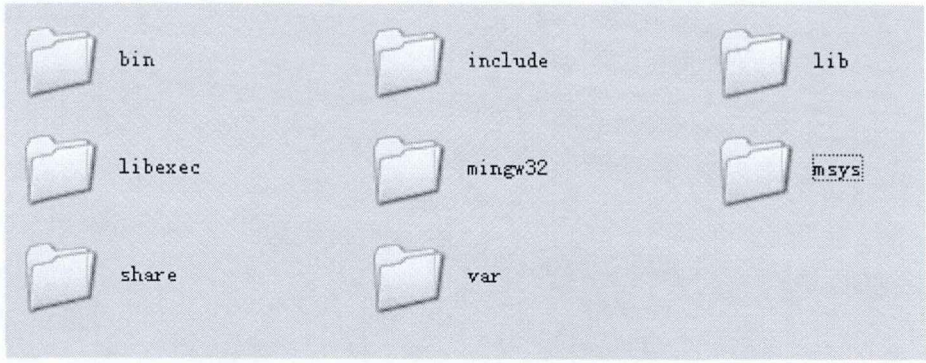


**图 6-1 MinGW 和 MSYS 安装结果**

3）配置 FFMPEG 所需汇编编译器 yasm.exe

从官网下载 FFMPEG 编译所需的汇编编译器 yasm-1.1.0-win32.exe，为简化环境变量设置，将文件名改为 yasm.exe 放到系统文件路径下。

4）配置媒体开发库 SDL

（1）从官网下载 SDL-1.2.14.tar.gz - GPG signed；

（2）编译 SDL：双击 C:\MinGW\msys\1.0\msys.bat 启动“MinGW32”，切换到 SDL 目录后执行编译命令./configure –prefix=/user 回车后执行 make 和 make install 命令后在 C:\MinGW\msys\1.0 的 bin、include 和 lib 文件夹下可以看到编译结果。

5）配置 FFMPEG

（1）从官网下载 FFmpeg，下载完成后解压到某处待命；

（2）通过桌面或开始菜单的快捷方式，进入 MinGW 的命令行，进入 FFmpeg 的解压目录。FFmpeg 的目录结构如下：

a. doc:存放相关文档

b. ffpresets:参数设置

c. libavcodec:存放 encoder/decoder 模块

d. libavdeice:对输出输入设备的支持，声卡或视频采集卡

e. libavfilter:过滤器，主要加入水印功能

f. libavformat:存放 mux/demux 模块对音频视频格式的解析

g. libavutil:存放常用算法工具函数库

h. libpostproc:对于视频做前处理的函数库

i. libswscale:视频场景比例缩放、色彩映射转换

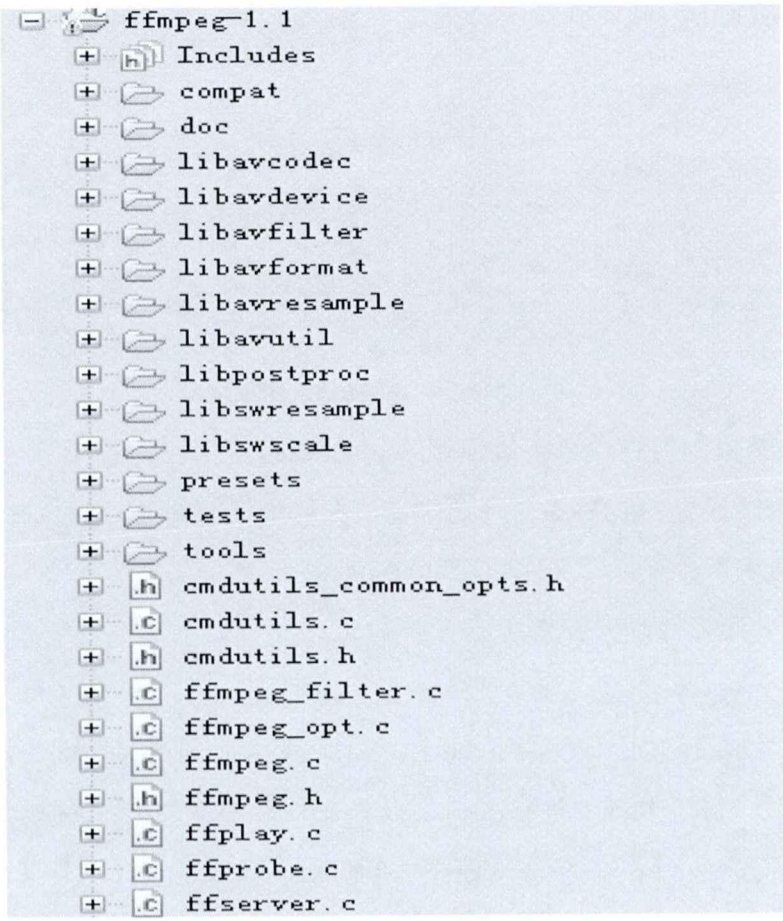


**图 6-2 FFMPEG 目录结构**

## 6.3 FFMPEG 数据结构分析

由FFMPEG的代码结构可知，实现视频编解码功能的主要函数存放在libavcodec库中， FFMPEG主要通过以下数据接口来实现视频的编解码：

1）AVFormat是一个包含了所有的普通的音视频格式的解析器和产生器的库。包括以下结构：

（1）AVInputFormat: 解码格式，主要处理解码输入，

提供主要接口：

int (*read_probe)(...)判断输入码流是否与选择的format是否一致；

int (*read_header)(...);处理一些初始化工作；

int (*read_packet)(...);解析packet获取有效数据；

int (*read_close)(...)。

（2）AVOutputFormat: 编码格式，主要处理编码输出，

提供主要接口：

int (*write header)( ); 写入数据头；

int (*write_packet)(…);写入packet数据，也就是编码后数据；

int (*write_trailer)(…);编结束调用，写入缓存中的数据。

（3）AVFormatContext:在编解码中它是一个贯穿始终的数据结构，包含了一个视频流的格式内容。

2）AVCodec：code-decode的联合体。从库的名称来看这是一个集编码，解码于一体的库，包括以下结构：

（1）AVCodec：编解码器的全局变量，存放编解码的一些公共接口；

（2）AVStream：保存与数据流相关编解码器和数据段的信息；

（3）AVCodecContext：保存AVCodec指针和与codec相关数据；

（4）AVPacket：保存一个或多个完整的编码或解码的帧；

（5）AVCodecParser：解码分析器；

（6）AVCodecParserContext：解码分析器上下文。

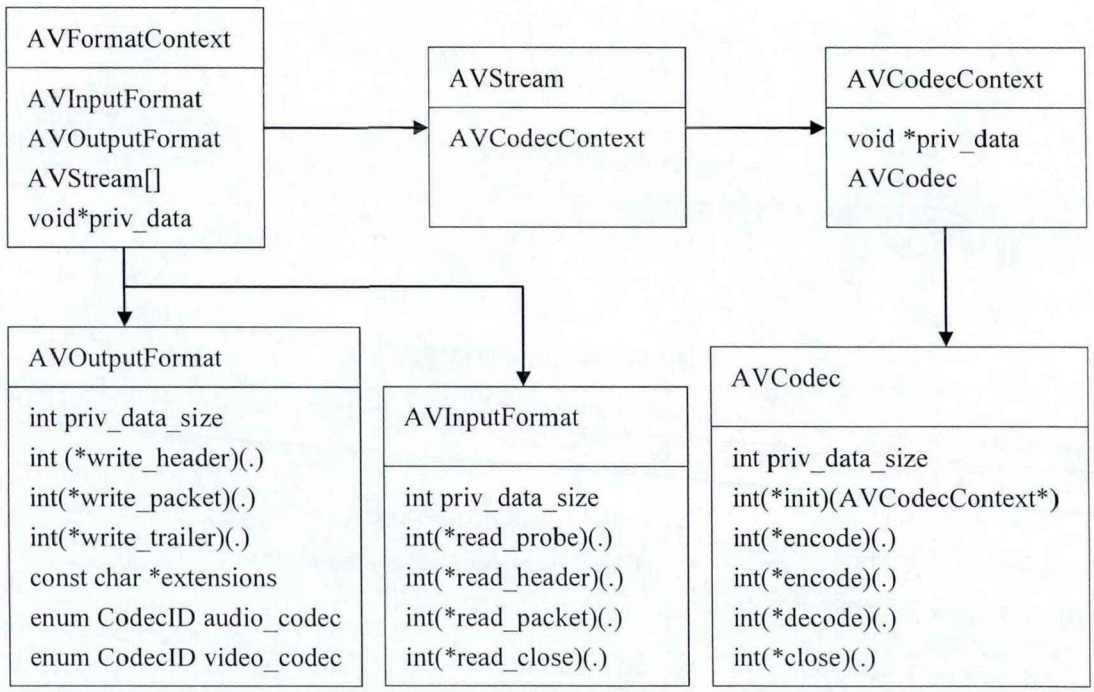


**图 6-2 FFMPEG 数据结构**

## 6.4 FFMPEG 视频解码过程

使用 FFmpeg 对视频进行解码首先需要对 FFmpeg 库进行初始化，注册库中

含有的所有可用的文件格式和解码器，这样当打开一个文件时，它们才能够自动选择相应的文件格式和解码器[7]。然后，打开视频文件，通过简单地指明 NULL 或 0 告诉 libavformat 去自动探测文件格式并且使用默认的缓冲区大小，取出包含在文件中的流信息，同时用有效的信息把 AVFormatContext 的流域（streams field）填满，从中得到视频流编码上下文的指针，并通过此指针找到视频流的解码器，打开编码器，循环调用 GetNextFrame 将视频流转换成一帧一帧的 RGB 格式，然后，释放分配的资源关闭解码器，关闭原始文件，完成视频解码。解码流程如图所示：

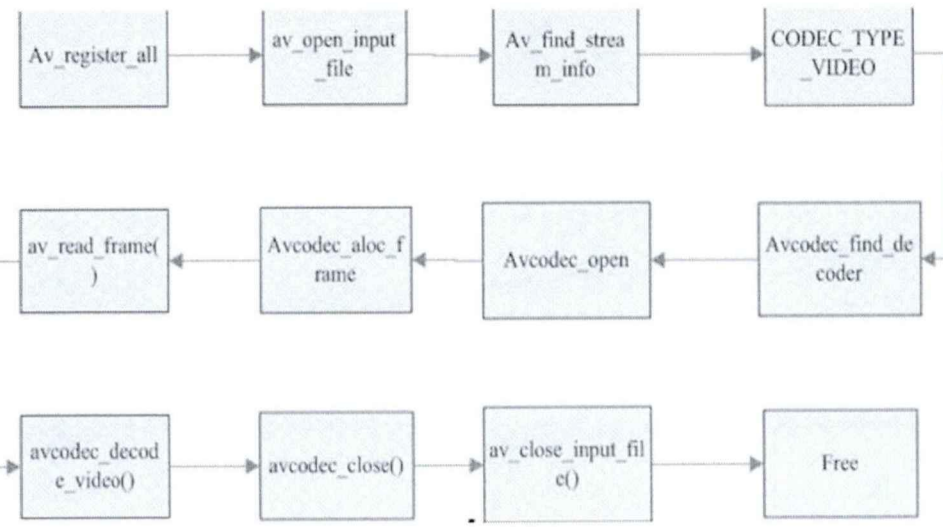


**图 6-3 FFmpeg 解码过程**

（1）注册所有容器格式和 CODEC: av_register_all()注册库中含有的所有可用的文件格式和编码器，同时初始化 libavformat/libavcodec；

（2）打开文件: av_open_input_file(&pFormatCtx, filename, NULL, 0, NULL) 打开视频文件，获取视频流信息；

（3）从文件中提取流信息: av_find_stream_info(pFormatCtx) 取出包含在文件中的流信息，以便将其转成一帧；

（4）查找对应的解码器: avcodec_find_decoder(pCodecCtx->codec_id) 穷举所有的流，查找其中种类为 CODEC_TYPE_VIDEO 的编解码器；

（5）打开编解码器: avcodec_open(pCodecCtx, pCodec)；

（6）为解码帧分配内存: avcodec_alloc_frame()，给视频帧分配空间以便存储解码后的图片；

（7）停地从码流中提取出帧数据: av_read_frame(pFormatCtx, &packet) 读取包

来读取整个视频流，然后把它解码成帧，最后转换格式并且保存；

（8）判断帧的类型，对于视频帧调用: avcodec_decode_video(pCodecCtx, pFrame, &frameFinished, packet.data, packet.size)；

（9）解码完后，释放解码器: avcodec_close()；

（10）关闭输入文件: avformat_close_input_file()。

## 6.5 FFMPEG 视频编码过程

由于 FFmpeg 没有实现 H.264 的编码，需要调用 x264 的编码库并对其进行封装来完成的，首先都需要对 FFmpeg 库进行初始化，注册库中含有的所有可用的文件格式和编码码。调用 x264_param_default( x264_param_t *param ) 对编码器进行参数设定，通过设置描述编解码器上下文的数据结构 AVCodecContext 来设置编码器需要的各种参数信息：码率、帧速率、编码像素格式等，然后寻找编码器 xxx 并打开，打开编码器后，按照参数设置对视频流进行编码，循环调用编码的核心函数 avcodec_decode_video()来对视频流的每一帧进行 H.264 编码,最后得到 H.264 视频流。FFmpeg 编码流程如下所示：

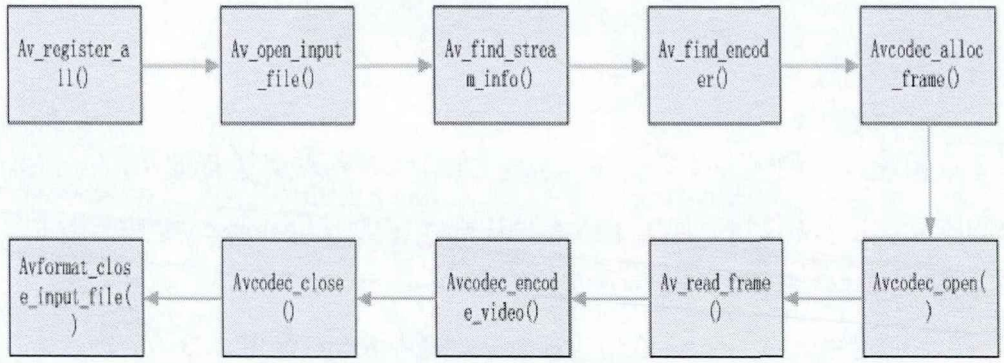


**图 6-4 FFmpeg 编码过程**

（1）注册库中含有所有文件格式和编解码器：调用 av_register_all()进行文件格式和编解码注册，当打开一个文件时，它们才能够自动选择相应的文件格式和编码器。av_register_all()只需调用一次，所以，要放在初始化代码中。也可以仅仅注册个人的文件格式和编码；

（2）打开所有视频文件：av_open_input_file（&pFormatCtx, filename, NULL, 0, NULL）函数中参数为视频文件上下文和名称，以及视频格式等基本信息，最后三个参数描述了文件格式，缓冲区大小（size）和格式参数；

（3）寻找视频流：av_find_stream_info(m_pFormatCtx_dec)从包含文件信息的上下中找到视频流信息；

（4）寻找编码器：av_find_encoder(pCodecCtx->codec_id)在注册的所有文件格式编解码器的链表中找到与视频流格式相匹配的视频编码器；

（5）为编码帧分配内存：avcodec_alloc_frame()给视频帧分配空间以便存储解码后的图片；

（6）打开编码器：avcodec_open(pCodecCtx, pCodec)；

（7）从视频流中读取一帧视频数据：av_read_frame(pFormatCtx, &packet) 通过读取包来读取整个视频流，然后把它编码保存；

（8）编码一帧视频数据：avcodec_encode_video(pCodecCtx, pFrame, &frameFinished, packet.data, packet.size)对每一帧的视频流进行编码操作；

（9）关闭编码器：avcodec_close()；

（10）关闭视频文件：avformat_close_input_file()。

## 6.6 FFMPEG转码技术在系统中的应用

在对FFMPEG视频编解码的工作原理进行研究之后，对FFMPEG的源代码进行修改，使之可以支持HTML5 video标签指定的三种视频格式进行编解码，同时将此功能应用HTML5视频系统中，使用户上传的视频通过转码后，可发布到视频播放平台进行清晰流畅的播放。

### 6.6.1 编译FFMPEG

使用配置好的MinGW编译环境对FFMPEG源代码进行编译，得到转码模块的静动态链接库，然后使用JAVA调用编译好的ffmpeg.exe实现对用户上传视频的转码，具体操作流程如下所示：

（1）通过桌面或开始菜单的快捷方式，进入MinGW的命令行，进入FFmpeg的解压目录；

（2）使用configure命令配置FFmpeg的编译参数。配置的参数如下：

```
./configure --enable-shared --enable-gpl --enable-version3 --disable-doc
--disable-ffmpeg --disable-ffplay --disable-ffprobe --disable-ffserver
--enable-w32threads --disable-network --enable-memalign-hack
--disable-everything --enable-decoder=h264 --enable-decoder=mpeg2video
--enable-parser=h264 --enable-encoder=mpeg2video --disable-debug;
```

（3） 输入make进行编译；

（4）编译完成后，输入make install进行部署；

生成的动态库在“msys 目录\local\bin”目录下；链接库文件在“msys 目录\local\lib”目录下的，*.dll.a 为动态库需要的链接文件，*.a 为纯静态库（不依赖 dll）；修改扩展名为.lib 就可以用了[8]。需要的头文件在“msys 目录\local\include”目录下。

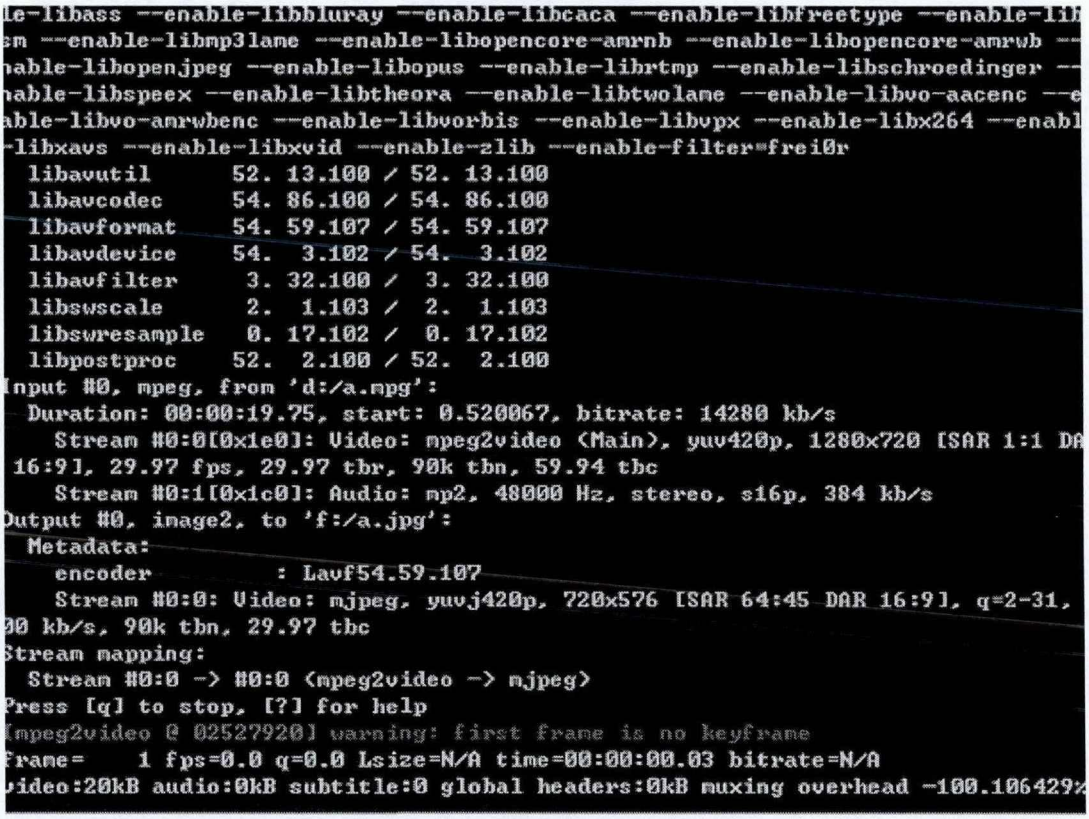


**图 6-5 FFMPEG 编译结果**

### 6.6.2 HTML5 视频系统调用 FFMPEG

首先将 FFMPEG 编译后的可执行文件和动态链接库放到一个目录中，可使 JAVA 后台可以成功调用 FFMPEG，视频转码命令如下所示：

（1）转码成 ogv (Theora / Vorbis):

ffmpeg -i a.avi -b 1500k -vcodec libtheora -acodec libvorbis -ab 160000 -g 30 sample1.ogv

（2）转码成 webm (VP8 / Vorbis):

ffmpeg -i a.avi -b 1500k -vcodec libvpx -acodec libvorbis -ab 160000 -f webm -g 30 sample2.webm

（3）转码 mp4 (H.264 / ACC):

ffmpeg -i a.avi -b 1500k -vcodec libx264 -vpre slow -vpre baseline -g 30 sample3.mp4

代码如下所示：

```
/*
 * 视频转码
 * @param ffmpegPath      转码工具存放路径
 * @param inputFilePath   要转换格式的视频源文件
 * @param outputFilePath  转换格式后的文件保存路径
 * @return
 * @throw Exception
 */
public boolean transfer(String ffmpegPath,String inputFilePath,String outputFilePath) throws Exception{
    //创建List集合来保存视频转码命令
    List<String> converToogv = new ArrayList<String>();
    List<String> converTowebm = new ArrayList<String>();
    List<String> converTomp4 = new ArrayList<String>();
    converToogv.add(ffmpegPath);//添加转码工具路径
    converToogv.add("i");//添加参数"i"，该参数指定要转码的视频文件
    converToogv.add(inputFilePath);//添加要转码的视频文件路径
    converToogv.add("-qscale");//指定转码质量
    converToogv.add("6");
    converToogv.add("-vcodec");//指定视频格式
    converToogv.add("libtheora");
    converToogv.add("-acodec");//指定音频格式
    converToogv.add("libvorbis");
    converToogv.add("-ab");//设置音频码率
    converToogv.add("64");
    converToogv.add("-ac");//设置声道数
    converToogv.add("22050");
    converToogv.add("-r");//设置帧数
    converToogv.add("24");
    converToogv.add("-y");//设置是否覆盖已存在文件
    converToogv.add(outputFilePath);//设置要转码后的输出视频文件路径
    converTowebm.add(ffmpegPath);//添加转码工具路径
    converTowebm.add("i");//添加参数"i"，该参数指定要转码的视频文件
    converTowebm.add(inputFilePath);//添加要转码的视频文件路径
    converTowebm.add("-qscale");//指定转码质量
    converTowebm.add("6");
    converTowebm.add("-vcodec");//指定视频格式
```

**图 6-6 Java 调用 FFMPEG 实现转码**

应用到 HTML5 视频系统中的界面如图所示：


**图 6-7 FFMPEG 转码界面**

FFMPEG转码在HTML5视频系统中实现用户上传视频的编解码功能，可将视频转码成HTML5 video标签可播放的三种高清视频格式，以供用户在播放平台可更好的享受HTML5多媒体元素的跨平台，流畅清晰的播放体验。

## 6.7 本章小结

本章节对FFMPEG技术做了详细介绍，对FFMPEG的数据结构和流程图做了深入分析，在此基础上，提出了使用FFMPEG技术对视频进行编解码操作，并将此功能成果植入到HTML5视频系统中，并成功应用，结果显示具有可行性和一定优势。

# 第七章 总结与展望

本章节首先对课题中所作的工作进行总结，并对课题中遇到的问题以及解决办法进行概括，提出课题的实用性和现实意义，在此基础上提出对课题项目的展望，更加全面的对课题内容进行评价并提出新的期望，来更好的完善课题内容。

## 7.1 项目总结

本次课题在基于 HTML5 的视频系统的基础上，提出使用 FFMPEG 进行视频转码的研究与实现，充分利用 HTML5 的多媒体新特性，为用户提供网络视频的高清和多屏享受。主要完成了以下工作：

（1）研究了当前网络视频系统的发展背景和视频转码技术的研究现状，同时对 HTML 新标准 HTML5 的各种新特性进行了深入研究，并结合 CSS3 的优势，提出了构建 HTML5 视频系统从背景和技术上的可行性；

（2）对广泛使用的开源多媒体处理软件 FFMPEG 从发展历史，核心架构，主要数据接口和视频转码操作的主要流程进行了深入的研究与分析，提出了使用 FFMPEG 技术构建视频转码模块的可行性；

（3）对 HTML5 视频系统进行需求分析和总体设计，并在此基础上使用 HTML5、CSS5、JAVA、SSH 框架和 Mysql 完成了系统的实现；

（4）构建 FFMPEG 在 Windows 环境下的编译平台，对 FFMPEG 进行二次开发和编译后，以 HTML5 视频系统为背景平台，将 FFMPEG 视频转码功能成果植入系统中，实现了将用户上传视频转码为 HTML5 video 标签可支持的三种视频格式，为用户提供流畅、高清的播放体验。

## 7.2 项目展望

本次课题中实现的在基于 HTML5 视频系统平台上实现 FFMPEG 转码技术的研究与应用拥有有好的用户体验，使得用户可以更好的享受 HTML5 多媒体元素带来的跨平台，轻负载，低内耗，高清播放等体验，但是，该系统还是存在的一些需要完善的地方：

（1）从对HTML5和CSS的新特性新技术在各种浏览器和各种平台的研究和应用可知，目前这两种技术已经处于被多个厂商竞相学习和尝试阶段，但仍处于起步阶段，仍需继续研究和完善；

（2）本次课题的重点研究方向即 FFMPEG 转码技术的研究过程中可知，FFMPEG 对于视频转码有着无法比拟的优势和便捷性，虽然 FFMPEG 视频采集功能非常强大，能支持多种视频格式的转码，但仍然存在一些无法解析的格式如 FFMPEG 转码时格式的支持问题 wmv9， rmvb 等，这就需要在使用 FFMPEG 进行转码时首先使用其他工具将不能支持的视频格式转换为 FFMPEG 能解析的格式，然后在对视频进行相应需求的转码。为用户提供视频播放需求。

# 参考文献

[1] 维基百科 HTML5 http://zh.wikipedia.org/wiki/HTML5

[2] 百度百科 HTML5
http://baike.baidu.com/link?url=TnByEcdkkVHex_WDm47YlHAj0huyKqmJHy7sSbFSRfGm8_PIE6rfPN7-ZZe40AIk

[3] 永不假死_HTML5 干掉 Flash 的最大法宝[J].电脑爱好者.2010 年第 09 期

[4] 维基百科 CSS3 http://zh.wikipedia.org/wiki/CSS3#CSS3

[5] FFmpeg on Windows［Z/OL］.［2012-05-30］

[6] 百度百科 Web Service http://baike.baidu.com/view/67105.htm?fromId=837392

[7] 在 Windows 下编译 ffmpeg 完全手册［Z/OL］.[2007-03-18]
http://www.cnblogs.com/sunlin886/archive/2007/03/18/678863.html

[8] 邓蕾蕾，张里荃.基于 FFmpeg 技术的媒体资产管理系统设计［J］.吉林大学学报:理学版，2012，50(6)：1218-1222.

[9] 王彤.基于 FFmpeg 的 H.264 解码器实现[D].大连理工大学;2011 年

[10]龙奇. 新一代网络技术标准 HTML5 的研究[J]. 科技信息, 2011, (10).

[11]乐天,有了 HTML5 还要 Flash 吗？计算机世界.2009 年 6 月 29 日第 034 版

[12]刘斌.HTML5 未来网络应用的核心技术研究,自动化与仪器仪表,201104 期.

[13]维基百.CSS[EB/OL].(2011-11-09) http://zh.wikipedia.org/wiki/Css.

[14]覃艳.基于 FFMPEG 的视频格式转换技术研究[J].电脑知识与技术.2011(12)

[15]刘建敏,杨斌.嵌入式 Linux 下基于 FFmpeg 的视频硬件编解码[J].单片机与嵌入式系统应用.2011(06)

[16]马洪堂.基于 FFMPEG 的视频转换系统的模块研究[J].电脑知识与技术.2009(13)

[17]高睿鹏,刘佳玲.基于 FFMPEG 的通用视频插件[J].电脑知识与技术.2010(10)

[18]吴张顺,张珣.基于 FFmpeg 的视频编码存储研究与实现[J].杭州电子科技大学学报.2006(03)

[19]单海涛,方向忠.基于 FFMpeg 的高清数字电影软件编码系统的设计[J].信息技术.2007(01)

[20]李少春.基于 FFMPEG 的嵌入式视频监控系统[J].电子技术.2007(03)

[21]马小铁,马子彦.基于 Mencoder 编码器的 AVI 视频转换软件的设计[J].北京服装学院学报(自然科学版).2008(01)

[22]肖友能,薛向阳,曾玮.视频转码技术回顾[J].通信学报.2002(08)

[23]Asheesh Bhardwaj.视频转码技术与系统要求相匹配可显著提升新一代视频应用性能[J].今日电子.2010(01)
[24]FFMpeg.H264 解码开发环境搭建--For Windows
http://bbs.chinaunix.net/linux/ffmpeg.shtml
[25]维基百科 FFMPEG http://zh.wikipedia.org/zh-cn/FFmpeg
[26]FFMpeg 框架代码阅读 http://bbs.chinaunix.net/linux/ffmpeg.shtml
[27]windows 下提取 ffmpeg 中的 h264decodec
http://bbs.chinaunix.net/linux/ffmpeg.shtml
[28]Ffmpeg.Lambert M Surhone,Mariam T Tennoe,Susan F Henssonow, Betascript Publishing (2010 年 11 月 15 日)
[29]FFmpeg SDK 开发手册
http://wenku.baidu.com/view/a4710a12a21614791711284a.html
[30]FFmpeg Documentation, http://ffmpeg.org/
[31]胡聪,周甜,唐璐丹.基于 FFMPEG 的跨平台视频编解码研究[J].武汉理工大学学报.2011(11)
[32]马洪堂.基于 FFMPEG 的视频转换系统的模块研究[J].电脑知识与技术.2009(13)
[33]辛长春,娄小平,吕乃光.基于 FFmpeg 的远程视频监控系统编解码[J].电子技术，2013,(01):3-5.
[34]丁峰,刁鸣.FFmpeg 的音视频格式转换设计[J].应用科技，2013,(02):11-14.
[35]刘马飞,曾学文,倪宏.Windows 平台下应用 FFMPEG 实现 H.264 视频回放[J].微计算机应用，2008，（11）:61-65

# 攻读学位期间的研究成果目录

1. 已发表或已录用的主要学术论文

[1]赵淑漫，夏小玲，乐嘉锦. A REAL-TIME WEB APPLICATION SOLUTION BASED ON NODE.JS AND WEBSOCKET. The 2013 6th International Conference on Advanced Computer Theory and Engineering (ICACTE 2013)已录用.

# 致谢

随着研究生毕业论文的即将完成，我的研究生生涯也接近尾声，我六年多的东华学生生涯也即将结束，回顾在东华的四年本科生涯和两年半的研究生生涯，我感触良多，我由衷感谢学院领导、老师在各方面给予我的指导与帮助。在学习上，是他们在学术、科研项目上给予我悉心指导，在生活中给我无微不至的关心和照顾，在工作方面给我鼓励和支持，才使我能够顺利的完成在东华的学习和科研任务。

自从 2007 年进入东华大学计算机科学与技术学院学习以来，在这六年多的时间里，我完成了本科和研究生阶段的学习，我非常有幸能认识乐嘉锦老师和夏小玲老师，我的每一步成长和进步，无不凝聚着两位老师的心血，乐老师以他渊博的学术知识，严谨的治学态度，刻苦的钻研精神和勤恳的工作作风时时刻刻感染着我，使我在学术方面和工作方面精益求精，一丝不苟，夏老师对待学生和蔼可亲，在我遇到困难瓶颈的时候对我循循善诱，谆谆教诲，不断鼓励我，支持我，对我产生了非常深刻的影响。与两位老师相处的时光将成我今后学习生活工作中宝贵的人生财富，在此我要向乐老师和夏老师表示我最崇高的敬意和最诚挚的感谢，同时我要向新诗信息科技（上海）有限公司的总经理蔡昭华先生和我实习期间给以我极大帮助的罗曼、甘泉等新诗公司同仁表示我诚挚的谢意。

我还要向王嘉廉东华大学新媒体实验室和同寝室的好友们表示诚挚的谢意，感谢他们与我一起度过东华大学这段人生中的美好时光，特别感谢同实验室的曹丹丹、王云林、张鑫、郭永鑫同学对我的帮助和支持，在研究过程中，是他们的帮助使我克服了很多困难，相信在今后的岁月中，我们的友谊会更加深厚和长久。

我更要感谢我的父母和家人，是他们在我整个成长过程中给予我无微不至的照料和关怀，是他们的支持和鼓励，我才有了今天的成绩。

感谢所有关心和帮助过我的领导、老师、家人、朋友们。

最后，由衷地感谢各位专家老师的评议和指导。

作者：赵淑漫

2013 年 12 月