文本复制检测报告单(全文标明引文)

№:ADBD2018R_2018053015312720180530154850440174335082 检测时间:2018-05-30 15:48:50

检测文献: 53141518_谭深_物联网工程_基于CS结构的视频管理平台的设计与实现

作者: 谭深

检测范围: 中国学术期刊网络出版总库
中国博士学位论文全文数据库/中国优秀硕士学位论文全文数据库
中国重要会议论文全文数据库
中国重要报纸全文数据库
中国专利全文数据库
图书资源
优先出版文献库
大学生论文联合比对库
互联网资源(包含贴吧等论坛资源)
英文数据库(涵盖期刊、博硕、会议的英文数据以及德国Springer、英国Taylor&Francis 期刊数据库等)
港澳台学术文献库
互联网文档资源
CNKI大成编客-原创作品库
个人比对库

时间范围: 1900-01-01至2018-05-30

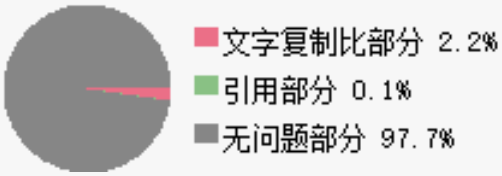
检测结果

总文字复制比：2.3% 跨语言检测结果：0%

去除引用文献复制比：2.2% 去除本人已发表文献复制比：2.3%

单篇最大文字复制比：0.4%

重复字数：	[602]	总段落数：	[7]
总字数：	[26235]	疑似段落数：	[5]
单篇最大重复字数：	[104]	前部重合字数：	[69]
疑似段落最大重合字数：	[453]	后部重合字数：	[533]
疑似段落最小重合字数：	[34]		



指 标：☐疑似剽窃观点 ☑疑似剽窃文字表述 ☐疑似自我剽窃 ☐疑似整体剽窃 ☐过度引用

表格： 0 公式： 0 疑似文字的图片： 0 脚注与尾注：0

0%（0）	中英文摘要等（总2724字）
2.1%（39）	第1章绪论（总1817字）
4.9%（453）	第2章开发环境与技术依赖_第1部分（总9279字）
1.3%（38）	第2章开发环境与技术依赖_第2部分（总2815字）
0.8%（38）	第3章视频管理平台系统设计（总4939字）
0%（0）	第4章视频管理平台系统实现（总3181字）
2.3%（34）	第5章总结与展望（总1480字）

（注释：无问题部分 文字复制比部分 引用部分）

1. 中英文摘要等 总字数：2724

相似文献列表 文字复制比：0%(0) 疑似剽窃观点：(0)

原文内容 红色文字表示存在文字复制现象的内容; 绿色文字表示其中标明了引用的内容

吉林大学学士学位论文（设计）承诺书

本人郑重承诺：所呈交的学士学位毕业论文（设计），是本人在指导教师的指导下，独立进行实验、设计、调研等工作基础上取得的成果。除文中已经注明引用的内容外，本论文（设计）不包含任何其他个人或集体已经发表或撰写的作品成果。

对本人实验或设计中做出重要贡献的个人或集体，均已在文中以明确的方式注明。本人完全意识到本承诺书的法律结果由本人承担。

学士学位论文（设计）作者签名：

2018年5月20日

摘要

基于C/S结构的视频管理平台的设计与实现

随着访问网络成本的降低和智能设备的普及，越来越多的人们开始不满足于通过枯燥乏味的文字等传统媒体方式，而是更加青睐于通过生动快捷的视频等新媒体方式来满足各方面的信息需求。据统计，有84.4%的人群更乐于通过视频等新媒体方式获取关注的信息，其中有64.5%的人为重度网络视频用户。在这种千载难逢的机遇下，视频业务难免需要面对两方面的挑战：其一是如何在保证各方面性能的情况下大批量生产视频，另一方面是如何保障视频的质量和在多样设备上良好的兼容性。大批量的生产除了对视频转码任务的各方面性能有较高的要求，还需要制定视频的编码格式、帧率等参数的规范以适应日渐复杂的视频需求。

论文以视频平台业务需求和C/S架构开发现状为依据，详细介绍了基于C/S结构的视频管理平台的整套功能的设计思路和具体实现过程，重点讲解了能较好解决上述两个挑战并具有良好的并行处理能力和稳定的异常重试机制的视频转码流程，简单分析了针对视频管理平台的存储、计算、分发等强依赖因素能够采用的大规模生产技术方案。该视频管理平台分为四个相互独立的模块：视频生产模块、视频审核模块、视频管理模块和用户管理模块，旨在建立一个简便、准确、可靠的生态化视频管理平台。

关键字：视频管理平台，C/S模式，视频转码流程，异常重试

Abstract

Design and Implementation of Video Management Platform Based on C/S Structure

With the decrease in the cost of accessing the Internet and the popularity of smart devices, more and more people are beginning to dissatisfied with traditional media such as boring words, but prefer to meet the information needs of various aspects through new media such as video. According to statistics, 84.4% of the people are more willing to obtain information of interest through new media such as video, of which 64.5% are heavily network video users. Under this rare opportunity, video business unavoidably need to face the challenges of the two aspects: one is how to ensure that all aspects of performance under the condition of mass production video, on the other hand is how to guarantee the quality of the video and on various equipment compatibility. High-volume production not only has higher requirements on the performance of various aspects of video transcoding tasks, but also requires the specification of video coding formats, frame rates, and other parameters to meet the increasingly complex video requirements.

Based on the business requirements of the video platform and the development status of the C/S architecture, this paper describes in detail the design ideas and specific implementation process of the whole set of functions of the video management platform based on the C/S structure, and focuses on solving the above two challenges better. With the video transcoding process with good parallel processing capability and stable abnormal retry mechanism, the large-scale production technology solution that can be adopted for the strong dependency factors such as storage, calculation, and distribution of the video management platform is simply analyzed. The video management platform is divided into four independent modules: video production module, video management module, video distribution module and user management module, aiming to establish a simple, accurate and reliable ecological video management platform.

Keywords：Video management platform, C/S architecture, video transcoding process, abnormal retry

目录

2. 第1章绪论

总字数：1817

相似文献列表　文字复制比：2.1%(39)　疑似剽窃观点：(0)

1	汽轮机DEH系统快控方法和故障诊断及容错控制的研究 李潇潇(导师：王宣银) -《浙江大学硕士论文》- 2009-04-01	1.8%（33） 是否引证：否
2	基于IGE的NAT穿越技术的研究与实现 姜帆(导师：张国强) -《南京师范大学博士论文》- 2015-03-24	1.8%（33） 是否引证：否
3	面向树木科普知识的三维游戏设计 胡逊之(导师：王忠芝) -《北京林业大学硕士论文》- 2010-04-01	1.7%（31） 是否引证：否

原文内容　红色文字表示存在文字复制现象的内容; 绿色文字表示其中标明了引用的内容

第1章绪论

1.1 研究背景及意义

随着网络接入技术的飞速发展，无线网络成为了人们生活中必不可少的重要组成部分，越来越多的家庭、商场、餐厅甚至交通设备都开始大规模引入无线网络设备以满足人们的日常需求。作为网络的应用者，智能设备（手机、平板和电脑）也成为人们生活中不可或缺的重要组成因素。近几年，智能设备软硬件等各个方面日新月异的发展使人们清楚的意识到如今的智能设备已经不再是传统意义上的通信设备和计算设备，而是一个可以解决生活中大部分需求并和世界连通的窗口。人们将越来越多的时间来通过智能设备获取感兴趣的信息或实现自身的相关诉求。据调查，在2015年中国成年人群每天在媒体上花费的平均时间为6小时8分，其中花费在数字媒体上的时间为3小时5分，占比从2011年的35.8%上升到2015年的50.4%[1]。导致中国成年人群每天在数字媒体上花费时间飙升的主力因素之一是网络视频。据合理推测，2年后，中国成年人群会将日常网络时间的三分之二用来观看网络视频。

视频逐渐成为互联网的主导内容形态之一，在很大程度上也逐渐代替图文类型的消费，越来越多的人不再满足于通过枯燥乏味的文字等传统方式，而是渴望通过更加省心快捷的视频等新颖方式来获取感兴趣的信息，而且视频具有得天独厚的吸引力和表现力，鲜明的感官体验更是让视频渗透到了生活的方方面面，甚至很多用户在使用视频平台观看视频的时候丝毫觉察不到时间的流逝。除此之外，体验优秀的视频平台具有无与伦比的用户留存率，因为互联网具有后置盈利的特性，直观表现利润是从未来所提供的服务中产生的，所以用户留存率可以作为一个产品竞争力和整体影响力的重要指标。

生活中各个方面的视频缺口导致了视频需求的急速增长，市场的刺激使得各大互联网厂商争相研发视频平台通过实行各种策略和优惠活动来抢夺视频资源和用户人群。在这样的大潮流下，我认为设计和实现一个基于C/S结构的视频管理平台会在自身成长和未来发展上会有很大帮助。

然而视频本身具有视频类型格式丰富、占用带宽资源大和播放终端参数多样化，而且视频需求较大，转码任务繁重。这就要求需要一个视频转码流程并行独立，结果是通过转码产出多种规格格式视频的高性能转码流程来提升转码处理效率、提升用户上传视频的体验。构建一个具有并行对视频转码进行分阶段处理且转码发散成多种格式的视频供不同终端设备正常消费的可靠视频转码流程的视频管理平台就显得更加重要和迫切。

总的来说，视频管理平台的设计和开发主要目的是在现有主流技术的基础上打造一个具有独立视频转码流程的生态化视频管理分发平台。给那些想分享视频的人一个可靠的平台，把促进社会进步和国家发展的视频分享给更多的人，为人类的美好生活作出贡献。

1.2 论文工作介绍

论文是在实现C/S结构视频管理平台系统的基础上撰写的，主要工作是叙述设计和实现C/S结构视频管理平台系统的过程以及感悟。

首先论文主要介绍了开发过程选择使用哪些主要技术及其原因，包括前后端开发框架、工程构建工具、代理服务器、消息队列、视频存储服务、视频转码服务、内容网络分发和多种数据存储技术等主要技术和服务依赖。然后在合理使用这些技术和服务的基础上，论文介绍了C/S结构的视频管理平台的架构设计、转码流程设计和数据存储设计。其次论文介绍四大模块的具体实现与功能开发：视频生产模块、视频审核模块、视频管理模块和用户管理模块，主要功能点包括视频上传、视频审核、视频发布、视频删除、视频恢复、视频编辑、添加用户以及编辑用户等，其中开发重点为视频转码流程和异常重试机制。最后论文总结了设计和开发视频管理平台的全部工作，介绍了这一过程中收获和对未来视频管理平台应该怎么做的展望。

1.3 论文组织架构

总体来说，本论文相关内容组织结构如下所述：

第一章阐述了与本课题相关的研究背景及意义，调查当今社会对视频的需求和视频管理平台的发展前景提出了本课题的主要目标。

第二章汇总视频管理平台的开发过程所需要的技术依赖和相关支持，简单介绍技术的主要功能和显著优点。

第三章阐述视频管理平台总体架构和功能模块联系模型，重点介绍视频转码和数据存储的设计方案。

第四章根据总体架构和设计方案整体实现视频管理平台系统，简单介绍各个功能模块的操作界面和核心代码。

第五章总结设计与实现基于C/S结构的视频管理平台的工作和收获，并对未来视频管理平台的研究进行展望。

第2章开发环境与技术依赖

2.1 开发环境和相关技术

2.1.1 Fis3工程构建工具

目前主流的构建工具包括Fis3、Gulp和Grunt。相对而言，Grunt虽然插件功能比较完善但是构建时间会随着项目的增大而指数型增加，很大程度上会拖延开发速度；如果说Grunt是工程构建工具的一个重量级极端的话，那么Gulp就是另一个轻量级极端，因为Gulp的插件输了非常少，在很大概率上会出现在开发过程中不得不自己实现某些插件功能的情况；相比于前两者，Fis3更注重性能优化的方向，直观表现在它不像Gulp那么轻量，插件基本可以满足需求的正常开发，又不像Grunt那样重量地构建，构建时间能够让人接受。最值得一提的是，Fis3优秀的部署机器功能可以快速部署前后端代码文件，我们可以通过Fis3实现各种文件的快速部署，有效节省开发时间，极大提高开发效率。

Fis3是基于文件对象进行构建的设计原则是“*基于依赖关系表的静态资源管理系统与模块化框架设计*”的面向前端的工程构建工具[2]。

Fis3在执行编译的过程中扫描所有文件，并自动生成一个详细记录了项目内的静态资源Id、发布后的线上路径、资源类型以及依赖关系和资源打包等信息的静态资源关系表；Fis3在运行时会根据静态资源关系表按需加载资源。Fis3有三大重要特征：资源定位、内容嵌入和依赖声明：

1. 资源定位

资源定位可以有效的分离开发时的文件路径与部署后的文件路径的关系。如图2-1所示，借助资源定位我们可以不再关注部署的文件路径，我们可以在开发的时候很方便的使用开发环境下文件的相对路径，Fis3会在部署的时候把所有引用文件相对路径的地方替换为部署环境下的绝对路径。在部分文件对于部署路径有特殊要求的情况下，我们也可以通过配置文件实现控制。资源定位的存在极大地提升了代码的可移植性，有效解决了长久以来各环境下的文件部署路径不一致的问题，有效提高工作效率。

2. 内容嵌入

内容嵌入的主要作用是把一个文本文件的文本内容或一个图片文件的Base64编码嵌入到另一个文件中。Fis3在执编译的过程中，会在使用内容嵌入的地方直接嵌入所需要文本文件的文本内容或图片文件的Base64编码。HTTP请求的数量在一定程度上决定了页面的响应速度和整体性能，所以适当的资源嵌入可以能够使这些资源不再通过HTTP进行请求获取，能够有效地减少HTTP请求数，进而极大地提高响应速度并提升工程的可维护性。

3. 依赖声明

依赖声明为整个项目的模块化开发预留了接口。正如上面提到的那样，Fis3会在编译的过程把所有带有依赖标记的关系记录下来从而组成静态资源关系表从而实现按需加载资源。依赖声明的存在使模块化开发变为可能，而且由于静态资源关系表的存在，在线上运行时不需要再进行依赖分析，这是传统模块化开发框架不能做到的。模块化开发一方面可以分工合作提升工作效率，另一方面有利于拆分组织结构来有效提升工程可维护性。

图2-1 Fis3资源定位示例

Fis3有几千个比较使用的插件，这里重点提及一下实现对Smarty模板解决方案的工程构建工具支持的Fis3-smarty。Fis3-smarty用PHP写成的能够很好的支持模块化拆分，利用PHP脚本实现Smarty的解析，而且具有很好的目录规范，不依赖后端真正实现静态资源的管理的可靠插件。除此之外，Fis3-smarty默认已经依赖了所有需要的插件，默认情况下，你不需要再安装任何其他的Fis3插件。

2.1.2 Vue.js框架

目前的主流前端框架有Angular.js、React以及Vue.js。相比于Angular.js，Vue.js拥有非常生态的中文API文档和官方教程，在一定程度上帮助我们快速地学习并上手使用Vue.js；同时比起React + Redux更加繁琐复杂的架构，Vue.js具有更加简单实用的基础架构。据调查显示，Vue.js的性能远好于Augular.js，且稍优于React。

Vue.js是一款非常友好的、多用途且高性能的渐进式JavaScript框架，它可以帮你创建具有更强可维护性和可测试性的代

码库。简单来说，如果你已经有一个现成的服务端应用那么你就可以把Vue.js当做一个现成的应用嵌入其中用以带来更加丰富的交互体验，而且Vue.js的生态系统可以允许你将更多的业务逻辑放到前端来实现。而渐进式就表现在Vue.js没有强主张，通俗一点讲就是你可以在原有系统的上，自定义可复用组件，把Vue.js当作jQuery来用；你也利用Vue的扩展功能（像Vue-rounte、Vuex等）把Vue.js当作Angular.js用；你也可以用它的视图，自定义搭配并在整个下层使用；甚至你可以在底层数据逻辑的地方用面向对象编程、设计模式或者函数式编程的那套理念进行开发。而Vue.js就只是一个轻量级视图而已，只做了自己该做的事。

Vue.js是以数据驱动和组件化思想构建的自底向上增量开发的MVVM（Model - View- VideoModel）架构模式的JavaScript框架。Vue.js初始的设计核心是围绕视图的，它可以自主划分可复用视图组件，组件化是有效提高了代码可维护性和可复用性。举个例子，我们可以把搜索框做成一个组件，供项目里的所有需要添加搜索框的页面使用，这样就不用每次使用搜索框都现写，而且易于修改。[3]如图2-2页面组件架构树所示，页面实际上就是一棵嵌套的组件树组织的。而且MVVM模式下，工程系统的可维护性、可扩展性和开发效率都会有一定程度的提高。如图2-3 MVVM架构模式所示，MVVM的中View和ViewModel之间、ViewModel和Model之间都是双向绑定的， 在这种架构下，任意一个部分的数据发生改变，那么就会立即呈现在其他部分，也即是操作了ViewModel中的数据（当然只能是监控属性），就会同步到DOM，我们透过DOM事件监控用户对DOM的改动，也会同步到ViewModel[4]。

图2-2 页面组件架构树

图2-3 MVVM架构模式

由于需要打造一个单页模式的视频管理平台用以优化用户使用体验，我们这里需要简单介绍一个Vue-router。Vue-router是在不依赖后端就能实现页面跳转的路由组件，利用Vue-router可以很高效地打造一个具有良好用户体验的单页模式应用。单页模式是指只加载单个HTML页面并在用户与应用程序交互时动态更新该页面而不是重新刷新。Vue-router实现路由的两种方式分别是通过HTML5的API接口（history）和页面Hash。通常情况下，代码通过监听URL Hash的DOM事件进而形成典型的观察者模式[18]，Vue-router通过创建和挂载router实例，在页面Hash发生改变的时候会重定向路由，继而把新的组件加载到挂载的元素上，从而实现了无刷新变更数据的单页体验。

2.1.3 ODP框架

ODP（Online Develop Platform，在线业务开发平台）是百度开发的包括开发执行环节、核心基础库、数据交互层级规范、核心框架在内的用于规范各大产品线的技术标准和提高产品稳定性与安全性以及开发人员工作效率的一整套体系架构。ODP提供了标准的webserver环境、标准PHP环境、AP框架、SAF社区业务框架、基础库、RAL资源访问层、KSARCH通用服务等组件，统一业务的逻辑和部署结构，为测试、运维等提供一致的视图，利用ODP这套完善的框架可以帮助解决各大产品线之间开发环境各异、开发技术良莠不齐的问题，防止出现个人私自选择开发技术出现的服务不稳定与数据不安全的问题，同时也将所有的开发依赖都植入进来，为开发人员提供了极大的便利。[5]

1. AP框架

AP框架是基于PHP扩展开发，且完全由C语言写成（相关内容请参考PHP扩展开发的知识）的PHP框架，因此，AP框架的的性能比由纯PHP语言编写的开发框架有更好的性能。AP框架不仅制定了路由协议，还规定了开发的分层结构和只能向后依赖的层级关系。AP框架规定的分层功能明确、要求严格，显著的提高了开发人员的开发效率和代码的可维护性。

2. SAF框架

SAF框架包含了控制器组件、通用业务组件（参数处理、session处理、日志打印等）以及通用配置组件，并把开发工程中业务逻辑的共性问题给提取了出来，并提供了统一的解决方案的业务层框架。简单而言，SAF框架是一个能够为我们提供了很多的通用功能的工具库，比如校验用户的登录信息、接受用户提交的数据、更改用户的信息和记录用户的操作行为等。

3. RAL框架

RAL（Resource Access Layer，资源访问层）就是ODP提供的用于资源访问的RPC（Remote Procedure Call，远程过程调用）接口。RAL框架很好的实现了对后端服务端交互的支持，用户在添加相应服务配置后，即可使用RAL一站式的接口与后端服务进行交互，而不需要再关注数据格式处理与协议交互的过程，为后端交互提供简单可依赖的支持，有效提升代码可复用性和工作人员的开发效率的PHP扩展。RAL具有以下五个特点：

（1）支持多种交互协议和数据打包格式。RAL框架将整个交互过程分为数据打包解包和实际交互两个步骤，且易于扩充。

（2）支持资源定位，统一配置。RAL框架不仅支持传统的在本地文件中直接配置IP + Port的方式获取服务，也支持通过Galileo和 Webfoot（BNS，Baidu Naming Service）方式来获取服务配置。

（3）支持本机配置健康检查和负载均衡策略。RAL框架支持多种均衡策略: 一致哈希、随机、轮询。推荐的负载均衡参数, 可以直接使用配置套餐，负载均衡需要用户以参数形式，指定本次交互的 balance key。

（4）服务自动加载。RAL框架支持服务配置自动加载，对于上述提到的三种资源定位的配置方式，修改配置文件或资源定位的数据发生变化时，RAL框架会动态加载最新配置加载时支持数据合法行校验，通过后再更新本地备份。

（5）性能优秀，使用简单。RAL框架高度封装了交互过程，极大地简化上游调用方的开发成本。

2.1.4 Nginx服务器

Nginx是一个轻量级高性能的HTTP服务器和反向代理服务器，也是IMAP/POP3/SMTP代理服务器。Nginx是通过事件进

行驱动而且具有非常高效的反向代理能力和负载均衡能力的服务器。由于Nginx是以性能作为最重要的衡量指标且在实现上非常注重效率，所以Nginx的稳定性首屈一指。有调查显示，Nginx可以同时支持高达50000个并发连接数。Nginx的稳定性不仅表现在同时并发连接请求上，还表现在CPU与内存的超低占用率上。Nginx采用了分阶段资源分配技术，其官方表示，在保持10000个没有活动的连接不被中断的情况下，Nginx只占用了2.5M的内存，在根本层面上解决了DOS攻击。除了具备非常可靠的稳定性和低消耗性，采用模块化开发的Nginx具有非常优秀的可扩展性和可维护性，

Nginx模块从功能上可以分为Handlers处理器模块、Filters过滤器模块和Proxy代理类模块，如图2-4 Nginx响应HTTP请求的过程可以看到各个模块之间的工作顺序和依赖关系。

简单来说，Nginx具有两个主要功能：动静态资源分离和负载均衡。动静态资源分离分离是指Nginx允许借助正则表达式以区分资源属于动态资源还是静态资源，然后作出不同的处理；负载均衡是指Nginx在把动态资源转发给代理服务器群的时候会根据一定的算法选择合适的代理服务器，从而缓解请求压力。

如图2-5 Nginx基本架构所示，由事件驱动的Nginx主要依赖一个Master主进程和多个Work工作进程响应请求。[6]

图2-4 Nginx响应HTTP请求的过程

图2-5 Nginx基本架构

2.1.5 Nmq消息队列

Nmq（New Message Queue）是一个通用的分布式消息队列系统，常见的使用场景一般是不需要立即获得结果同时需要控制并发量，消息队列主要具有了异步并发、应用解耦、异步处理和流量控制等特点，进而实现高性能、高复用、高扩展和高维护的目的。

Nmq本质上就是分布式应用间交换信息的一种技术，本质是作为一个中间容器把消息暂时存储起来以备消息接受者进行消费。如图2-6 Nmq系统架构所示，Nmq分为三层：代理层（Proxy）、存储层（Topic）和分发层（Pusher）。代理层（Proxy）是Nmq整个集群的唯一入口，无状态且无单点，允许接收多种格式的消息数据，然后负责把接收的消息数据发送给存储层（Topic）；存储层（Topic）主要用保存消息队列，存储各个未处理的消息，主要负责级联与备份；分发层（Pusher）主要用于下发消息给消息接受者。除此之外，不同的消息下发方式对应着不同的业务场景，Nmq可以设置消息的下发方式为为拉模式（下游向分发层Pusher索要消息进行消费）或推模式（分发层Pusher负责推送消息给下游进行消费），用于标识消息下发的具体方式。

消息队列一般分为两种模式：点对点模式和发布订阅模式。[7]

点对点模式架构如图2-7所示，消息发送者生产消息发送到消息队列中间件中，然后消息接收者从消息队列中间件中取出并且消费消息。消息被消费以后或者被下发以后，消息队列会把该消息从队列中删除，防止同一个消息被多次消费或者下发。点对点模式对于一条消息只有一个消息接收者，发送消息者和接收消息者之间不存在任何的依赖关系（即解耦），而且在接收消息者成功接收消息之后需要向消息队列中间件应答接受成功，目的是实现消息队列中间件从消息队列中删除已被接收的消息。

发布订阅模式架构如图2-8所示，发送消息者将消息放松给消息队列，消息队列系统将这个消息传递给多个消息订阅者。发布订阅模式下，一条消息可以被多个订阅者接收，消息会被分发给提前订阅该主题消息的订阅者们，所以相比于点对点模式，发布订阅模式存在时间关系上的依赖，消息订阅者必须在消息产生并分发之前订阅该主题消息。

图2-7 Nmq系统架构

图2-7 Nmq点对点模式架构

图2-8 Nmq发布订阅模式架构

以向数据库写入数据为例，无论是Nmq的哪种工作模式，用户请求的数据都不再直接写入数据库，而是直接把消息发送给消息中间件就立即返回了，此时消息发送者已经不需要再关注消息后续处理系统怎么处理，处理结果怎样，Nmq会负责把消息下发为正确的消息消费者，如果消息处理成功，那么消息消费者会告知Nmq已经处理成功，此时Nmq会记录成功状态并把该消息从消息队列中删除，如果消息处理失败，那么Nmq会根据业务需求的相关设置进行重试多次甚至一直重试。因此消息队列的存在不仅在高并发的情况下高效实现限流削峰的作用进而有效提升了系统稳定性和系统响应速度，而且还成功实现了应用解耦，各功能应用各司其职、相互独立、互不依赖。

2.2 百度云平台

通过借助百度云平台提供了云端对象存储（BOS）、音视频转码（MCT）、内容网络分发（CDN）等优质服务，而且各个服务相互依存，我们可以使用对象存储（BOS）中的视频作为视频转码的主体，也可以把水印图片作为音视频转码模板的原始文件控制转码。通过百度云平台提供的优质服务，我们可以更加高效的进行视频转码、视频存储和视频分发，帮助我们忽略视频转码、分布式存储、网络节点分发等底层的技术难题，从而让我们有更多的精力去思考视频转码流程这一核心的设计与实现。

1.1.1 百度对象存储（BOS）

如图2-9 BOS整体架构所示，百度对象存储BOS（Baidu Object Storage）是提供稳定、安全、高效以及高扩展存储服务。

对于视频等静态资源的存储，我们首先需要考察的功能点在于是否可以有效防盗链。常规的防盗链技术多利用HTTP请求头中的Referer字段配合白名单机制实现，但是配置安全协议（HTPPS）的网站请求没有配置安全协议（HTTP）的网站时，

HTTP协议允许不在HTTP请求头中添加Referer字段，那么此时被请求网站是不知道这个请求是谁发出的，同样不能判断这个请求者是否是在白名单中，这种情况下，单纯的返回正确信息的逻辑和不反悔正确信息的逻辑都是有问题的；另外对于很多具备基本相关知识的人而言，伪装HTTP请求成本并不高，他们很容易就可以利用各种插件伪装HTTP的所有字段（包括Referer）来发送HTTP请求。所以通过HTTP请求的Referer字段的手段进行防盗链就不能再发挥作用。然而百度对象存储（BOS）不仅支持常规的白名单防盗链技术，还使用了非常可靠的URL鉴权进行资源防盗，基本杜绝了静态资源的被盗链的情况发生。除此之外，百度对象存储具有良好的多语言SDK和API文档，并且移入STS（Security Token Service）实现临时访问授权机制，使得数据存储的安全可靠更上一层楼。

图2-9 BOS整体架构

1.1.2 音视频转码（MCT）

音视频转码MCT（Multimedia Cloud Transcode）是在百度对象存储服务的基础上为用户和开发人员提供转码相关的扩展服务。音视频转码服务（MCT）可以支持主流的全部转码格式，我们可以预先设置多个转码模板，勾选相应的转码规格进行支持多码流多格式的转码操作。音视频转码服务（MCT）具有性能非常强大的转码集群，我们可以通过建立多个转码队列，将繁重的转码任务利用不同的转码队列进行转码，从而使转码任务分发到不同的转码集群上，为转码服务的稳定性护驾保航。

如图2-10 MCT工作模式所示，我们对BOS中存储的音视频文件进行转码操作，并把转码完成后的目标文件存储到BOS的指定目录中。除此之外，很多转码操作需要依赖文件的存在，比如给视频添加水印，我们可以事先把水印图片存储在百度对象存储（BOS）之中，建立转码模板指明依赖文件在BOS中的存储路径即可，这为我们的转码需求提供了极大的便利。和百度对象存储（BOS）一样，音视频转码（MCT）具有良好的多语言SDK和API文档，能够友好的支持功能接入和代码封装。

图2-10 MCT工作模式

1.1.3 内容分发网络（CDN）

内容分发网络（Content Delivery Network）是一种按照一定策略进行分配网络分发节点进而提升网站响应速度和网站访问成功率的一种技术手段。如图2-11 CDN工作架构所示，百度云平台所提供的内容分发网络是基于百度在全国各地的网络分布的基础上根据用户请求信息进行合理的策略计算实现提供可靠网络节点的加速服务。如图2-12 CDN加速原理所示，配置内容分发网络之后，我们会把源站的资源数据（主要是静态资源）拷贝到各个网络节点中，当用户请求资源数据的时候，内容分发网络（CDN）会根据一定的算法分析请求者的地域信息，得到距离该请求者最近、网络状态最佳且所请求的资源数据信息存在的网络节点，并把用户的请求分发到该网络节点上，网络节点经过正常的处理操作，把所请求的资源内容返回给请求者，完成整个请求与响应的操作。内容分发网络（CDN）并不是单拆的选择最近的网络节点，而是根据现在的节点请求数等负载状态、距离用户的距离等综合因素做出节点的最终选择。百度具有分布全国各地的可靠网络节点，通过百度云平台提供的内容分发网络（CDN），能够使我们的页面基本满足绝大多数用户的访问要求。

图2-11 CDN工作架构

图2-12 CDN加速原理

2.3 数据存储技术

随着网络数据产生量的急剧增加和数据存储成本的逐渐下降，人们越来越方便地存储生活中产生的各种数据，与此同时，也出现了很多优秀的数据存储技术，它们以互补的姿态功能组成了现在的数据存储现状。在允许牺牲较大性能的前提下，单一数据存储模式确实可以解决很大一部分的需求，但这不是我们想要的。

这里简单介绍一下视频管理系统需要使用到的主要存储技术：MySQL、Redis和ElasticSearch。

2.3.1 MySQL

MySQL是当今最流行的关系型数据库管理系统（RDBMS，Relational Database Management System），具有存储、截取、安全保障、备份等基本功能，用户可以通过MySQL实现对大数据量进行新增、删除、编辑等管理操作。

关系型数据库技术最早出现在1970年，几十年来，它通过出色的高度一致性事务一直是结构化存储的事实标准。本质上来说，关系型数据库就是建立在关系模型基础上的数据库，通过集合代数等数学概念和数学方法处理大规模的视频；简单而言，关系型数据库会把数据根据特定的规则存储在不同的表中，而且表与表之间可以通过各种特殊的键实现关联。MySQL有四种常见的基础架构：[8]

1. MySQL单实例架构

MySQL单实例就是在服务器上部署一个MySQL实例并对外提供服务，这是最初接触MySQL的常见的学习模型，甚至有的时候，MySQL数据库和应用程序会在同一个服务器上。如图2-13所示，单实例架构是非常简单并且直观的架构模型，通常不需要依赖任何其他第三方工具，具有良好的引导新手学习的特点，但是单实例架构由于其惨不忍睹的可用性和容灾性，所以一般MySQL单实例架构不用在业务开发中。

2. MySQL多主架构

MySQL多主架构是在MySQL单实例架构的基础上，添加多个数据库提供服务。这在一定程度上可以解决单实例单机出现故障之后服务立即终止的问题。如图2-15 MySQL多主架构所示，MySQL多主架构是通过多个可以进行操作的数据库节点存储数据，多个数据库节点是具有相同地位的读写数据库，当一个节点数据发生变更的时候，会立马发起同步，使另一个节点的数据同步为最新数据。虽然MySQL多主架构在一定程度上缓解了节点故障导致服务不稳定的问题，但是即使在低并发的情况下

，频繁的数据同步操作也是非常浪费资源和影响服务性能的操作，更何况是在高并发的情况下，我们不可能把那么多的资源和时间用于数据的同步操作上。除了浪费资源这个问题，MySQL多主架构还会导致各种同步冲突的出现，也就是在高并发情况下常见的问题，同一个时间点对多个数据库节点进行了写入数据的操作，那么下次各个数据节点再进行数据同步的时候很可能会出现合并冲突，此时数据节点一般会选择进行数据合并，但对于同时修改一条数据的情况，数据合并出现错误的概率极高，这对于数据存储的稳定性是一个很大的挑战。

图2-13 MySQL单实例架构

图2-14 MySQL多主架构

3. MySQL master-slave主从架构

MySQL master-slave主从架构是非常常见的一种MySQL数据架构，它在MySQL单实例架构的基础上提高了MySQL数据库的性能、可用性和可扩展性。如图2-15 MySQL主从架构所示，MySQL master-slave主从架构把MySQL进行了全库备份，然后拷贝出一个或多个MySQL实例，并保持多个MySQL实例数据保持完全一致。作为备份源的数据库节点被称为主数据节点master，需要借助主节点的备份进行更新数据的数据库节点被称为从数据节点slave。

显而易见，通过MySQL master-slave主从架构有效使线上数据库的有了多份数据，具有很好的数据备份功能。该架构从节点slave一般只对外部提供数据读取的服务并没有权限进行数据写入，这样可以高效解决读需求多而写需求少的业务的吞吐量和处理效率。通俗点讲就是，主节点master既用于提供读服务也会提供写服务，而从节点slave只提供读服务。通过这种设计模式，基本实现了数据库信息读写的有效分离。我们之所以需要实现读写分离，就是为了防止发生像MySQL多主架构那样的频繁同步和难以控制的合并冲突的异常情况出现。虽然MySQL数据库的master-salve主从架构实现了读写分离在一定程度上防止出现频繁地数据同步导致的资源浪费和同时写入数据导致的异常合并冲突的问题，但是master-salve主从架构始终无法扩展主数据节点master，换句话说，一旦主节点瘫痪，那么整个数据存储系统将没有办法继续提供写入数据服务。

4. MySQL MHA高可用架构

MySQL MHA高可用架构是目前数据存储架构中最完善的高可用性架构。如图2-16 MySQL MHA架构所示，MySQL MHA高可用架构实际上就是在MySQL master-slave主从架构的基础上，把master主节点的节点IP与业务联系起来，把master主节点从一个物理IP变成一个虚拟IP（VIP）或者域名。业务通过联系定位VIP访问数据，进行读写操作。如果出现主从架构不能解决的master节点出现故障问题的情况，那么MHA高可用组件会检测到这个故障，并将VIP根据一定的算法切换到另一个slave从库的VIP，即把新的这个节点当做主节点进行后续服务处理。换句话说，MySQL MHA高可用架构下的任何节点都可能是一个master主节点，但在一个时段内有且只能有一个master主节点。

图2-15 MySQL主从架构

图2-15 MySQL MHA架构

2.3.2 Redis

Redis是能够保存五种不同类型值之间映射的异常快速的非关系型（NoSQL）内存型键值数据库。其中五种类型数据类型包括：字符串（String）、列表（List）、集合（Sets）、有序集合（Sorted sets）、散列表（Map）。Redis内置了复制、LUA脚本、LRU驱动事件、事务和不同级别的磁盘持久化。Redis支持使用复制来扩展读性能、分片来扩展写性能并且可以为键设置过期时间，当键过期的时候自动删除该键等扩展功能。[9]

Redis的数据库完全存在于操作速度非常快的内存之中，并且可以借助把数据转存到磁盘来实现数据持久化。Redis提供了两种数据持久化的方式：AOF方式数据持久化和RDB Snapshotting方式数据数据持久化（默认）。

Redis支持主从复制，即可以把主服务器上的数据复制到任意数量从服务器中，从服务器甚至可以级联地把数据再复制到其他从服务器，来实现类似上文介绍的MySQL master-slave主从功能的目的，我可以通过多个从服务器实现读服务和数据持

久化工作，甚至可以禁止主服务器数据持久化，而且配置非常简单。[10]

然而对于一个网站很大一部分数据实际上使用频率并不高，倘若Redis把所有的数据都存储在内存中势必会导致内存空间紧张。基于这一现状，Redis引用了自身实现的虚拟内存机制来有效缓解这一问题。其主要思路是通过一定的算法把相对来说不常用的数据从内存移动到磁盘中，并通过一定的算法把常用的数据分离到多个服务器中，从而有效缓解单台服务器压力和整体内存紧张的问题。

2.3.3 ElasticSearch

ElasticSearch是一个基于Apache Lucene（TM）的开源的分布式搜索和分析引擎，它实际上就是封装了Lucene之后提供了良好的RESET API的操作接口。由于Lucene是由Java开发的成熟搜索引擎系统，所以ElasticSearch需要依赖正确的Java环境。ElasticSearch本质上是一个分布式的数据库，它不仅可以分布式地实时存储文件、分析搜索文件，甚至还具有扩展到上百台服务器的高扩展性，进而快速的实现海量数据的存储、搜索和分析。[11]

传统关系型数据库（MySQL）的存储方式是通过建立库和表格，根据表格类型存入各个字段实现存储，然而ElasticSearch是面向文档进行存储的数据库，会把JSON序列化格式的文件作为一个数据。ElasticSearch还会根据所存储的文档的内容建立对应的索引。因此通过索引搜索得到的不是传统关系型数据库那样的一行数据或者字段信息，而是整个文档信息，这就是全文索引。

ElasticSearch采用比传统关系型数据库B-Tree索引更快的倒序索引加速全文索引。在ElasticSearch为文档数据建立全文索引之前会先分析文档的文本，根据文本建立一个倒序索引。[12]

从根本出发，ElasticSearch就是借助各种有效的压缩算法尽量实现数据存储在内存中，以减少磁盘的随机读写次数。

2.4 本章小结

本章主要介绍了视频管理平台在开发阶段主要依赖的关键技术和支持。本章从三个部分拆分了关键技术依赖，第一个部分主要介绍了视频管理平台的开发过程中需要依赖的开发技术和基础环境，讲解了所依赖的前端和后端框架基本特征和选择原因、所依赖的服务器的基础知识和选择原因、所依赖消息队列的基础知识和选择原因，总体奠定了开发的环境和平台走向；第二个部分主要介绍了提供可靠云计算服务的百度云平台，通过借助百度云平台的技术手段支持解决视频转码流程过程中出现的视频存储在哪、视频怎么转码、如何内容分发等问题，一遍全身心投入到视频转码流程的开发当中；通过搭建一个MySQL + Redis + ElasticSearch的存储架构,可以成功打造一个支持高并发和具有良好稳定性的视频管理平台，所以第三个分布主要介绍了数据存储技术的相关知识。本章总体从实现开发的角度思考了各个部分需要采用哪些技术依赖，为后续开发打下坚实基础。

第3章视频管理平台系统设计

3.1 视频管理平台总体架构设计

如图3-1 视频管理平台系统框架所示，视频管理平台总体可以分为四个独立的模块：视频生产模块、视频审核模块、视频管理模块和用户管理模块。

图3-1 视频管理平台系统架构

3.1.1 模块简介

1. 视频生产模块：用户视频上传阶段的页面展现和视频上传、视频转码的功能开发。视频生产模块的页面展现提示用户录入囊括视频文件本身、视频封面图信息、视频转码信息、视频标题简介信息等重要信息。该模块工作重点和工作难点是如何设计和实现能解决第一章所说通病的视频转码流程。

2. 视频审核模块：用户上传视频之后管理审核的页面展现和审核视频的功能开发。用户在上传视频之后，需要通过管理员进行审核视频的各方面信息是否符合要求，如果视频文件本身存在问题导致不能通过转码，那么视频审核模块需要告知审核员文件不合法的原因，以便审核人员通知视频上传者修改视频重新上传；如果视频已经转码通过，但视频内容或视频附属信息不符合规定，审核人员可以通过视频审核模块拒绝视频申请通过并上线的要求。

3. 视频管理模块：对审核通过的视频进行线上管理的页面展现、重新编辑和上下线功能开发。视频管理模块主要是对审

核通过（即成功上线）的视频进行后续管理，如果视频需要重新编辑来更新部分信息，可以通过视频管理模块实现跳转到视频生产模块；如果视频需要下线或上线的需求，则也可以通过视频管理模块进行下线或者重新上线就行二次管理，视频管理模块是用于管理视频审核通过之后的后续管理操作。

4. 用户管理模块：添加用户、删除用户的页面展现以及系统权限逻辑的功能开发。视频管理平台系统只对部分有权限的人员开放，对于没有权限的人员不能访问。用户管理模块可以添加具有视频上传权限的用户，也可以编辑现有的已经赋予权限的用户，进而防止恶意操作。

3.1.2 模块目标

视频管理平台系统各个模块的主要目标是：

1. 通过技术依赖实现系统基本功能，搭建能够符合预期的视频管理平台系统

2. 合理利用UI库，设计并实现一个美观且大方、用户体验更好的用户界面。

3. 引入较为成熟的技术着力打造一个高性能视频管理系统

3.1.3 模块关系

如图3-2模块关系所示，视频管理平台系统的各模块之间虽然功能独立但也通过业务需求紧密联系。我们在视频生产模块进行视频上传与转码，视频符合要求且转码通过的视频会被添加到视频审核模块的视频审核列表中，允许管理人员进行视频人工审核，如果视频符合既定要求那么视频会被添加到视频管理模块的视频列表之中，以便方便进行视频上线、下线、编辑和视频信息的直观查询，如果视频不符合既定要求，视频平台管理人员会拒绝通过该视频，此时视频需要在修改视频文件以符合要求之后再通过视频生产模块进行视频上传，重新走一遍上述流程。视频在通过审核之后我们可以通过视频管理模块的视频编辑再次进行视频生产模块，重新编辑并提交之后重新走一遍删除流程。用户管理模块的权限控制是整个视频管理平台系统的基础，所有的页面都需要建立在视频管理平台的管理用户登录才能访问到，而且所有的视频生产者都需要在视频管理平台系统中录入相关信息才被允许借助视频管理平台的管理用户来上传视频。

总的来看，视频管理平台的四大模块是按照业务功能划分并根据功能联系起来的相互独立的视频管理单元，是一个成熟并行得通的平台架构。

图3-2 视频管理平台模块关系

3.2 视频管理平台转码流程设计

像绪论提到的那样，视频具有视频类型格式丰富、占用带宽资源大和播放终端参数多样化的特点，而且视频平台的视频生产需求大、转码任务复杂且繁重，很容易导致各种不稳定和性能低下的问题，所以视频转码流程的设计是整个视频管理平台系统的重中之重。

视频格式转换可以通过借助FFMPEG实现。FFMPEG是一个非常强大的视频处理软件，可以在Windows、Mac和Linux中使用。FFMPEG提供了一个用于解决录制、转换和播放音视频的完整解决方案[19]。

所谓视频转码流程，无非就是一个视频源文件经过多次技术处理产出能够满足预期的一个或多个视频文件的过程。视频转码流程的设计不仅需要考虑如何解决视频需求大导致的视频转码任务繁重的系统不稳定性，还需要考虑如何给出一个能够解决日渐丰富的播放终端参数多样化和支持多种视频类型分发的设计方案，甚至需要考虑如何进行转码重试机制和预留未来的业务需求等问题。

如图3-3视频转码流程架构所示，源视频文件在经过多次转码等技术处理之后获得最终的可用于分发的视频文件。

归一化视频文件是指把源视频文件按照既定参数规则进行转码得到的标准文件。视频管理平台的所有视频文件都分别有一个归一化视频文件与之对应，其目的是解决用户生产的视频的各方面规格参数不一导致的视频转码流程异常情况太多的问题。归一化视频文件的存在可以让视频转码流程不再关注源视频文件本身的参数是什么样的，源视频文件在上传阶段实际上就会对视频文件是否符合既定标准进行过校验，也就是对于满足视频上传标准能够进入视频转码流程的视频而言，视频文件本身是具备转码成为归一化视频文件的能力的，整个后续的视频转码流程都是建立在归一化视频文件的基础上进行的，下文我们称这一过程为归一化过程，所以在某种意义上，视频转码流程得到了质的优化，成功解耦视频转码的具体流程与视频上传源文件，使视频上传和视频转码相互独立透明，更不需要再浪费太多的精力去思考如何兼容不同规格视频转码流程的异常问题。

临时视频转码文件是处于视频转码流程中与业务紧密相关的临时视频文件。简单来说，临时视频转码文件在归一化视频文件的基础上根据上传视频的转码要求进行视频转码而产生的一系列临时文件，也就是通过转存一个归一化视频文件作为起点根据后续业务需求进行相关操作。如果视频上传的时候告知了需要添加水印和拼接片头片尾的话，临时视频转码流程就需要先对视频添加视频水印，然后再对视频进行拼接片头片尾，此时临时视频转码文件就会包含转存的归一化视频文件、转码得到的视频水印片和拼接得到的视频水印片头片尾片（简称为视频水印头尾片）；如果用户上传时只选择添加片头片尾，那么临时视频转码文件就包含转存的归一化视频文件和拼接得到的视频头尾片；如果用户上传时只选择对视频添加水印，那么视频视频转码文件就包含转存的归一化视频文件和转码得到的视频水印片；如果对视频文件没有任何拼接片头片尾或添加水印的需求，那么临时视频转码流程实际上就不会做转码或拼接处理，临时视频文件就只包括转存得到的归一化视频文件。临时视频转码文件是视频转码流程的关键，需要整合视频上传视频时是否添加片头片尾、添加水印模块等信息，自主选择如何进行转码、拼接等处理。临时视频转码是视频转码流程中与业务需求紧密相连的高可扩展性核心，也是代码实现中的难点与重点。

标准视频文件是按照上传信息进行转码之后的最终视频文件的转存而来的用于视频播放和分发的视频文件。从本质来看，标准视频文件只是一个转折点，用于划分多样化的播放最终态视频与临时转码的中间态视频文件，目的是为了把视频上传时

用户自定义决定的转码过程和每个视频都需要做的视频多样化的转码过程完全解耦，进而提高视频上传时转码的扩展性。举个例子，假设我们现在需要对一个视频添加两次片头，那么我们只需要把临时视频转码模块进行改写就可以了，我们无需再关注归一化过程和发布多样化过程。

可用于发布和播放的视频文件是对标准视频文件进行多次转码形成的不同规格的视频文件群，我们可以称这一转码过程为发布多样化。发布多样化的目的是根据既定规则生成不同规格视频文件（如1080P MP4、1080P Flv等），以解决终端播放需求不一的问题，我们可以保证每一个正确转码的视频都有多种视频规格来满足用户的各种需求。

从根本来看，归一化过程和发布多样化过程是属于通用过程，也就是所有的视频转码都需要按照同一个标准来做，然而临时转码过程却是根据用户上传时自主选择决定的如何进行视频转码的自定义过程。这种拆分通用过程和自定义过程的设计方案能够在很大程度上提升了转码流程的可扩展性和可维护性，同时减轻了开发负担并提高了系统稳定性。

图3-3 视频转码流程架构

3.3 视频管理平台数据存储设计

由于视频文件一般比较大，自己搭建存储视频的服务器显然是一个不小的成本，而且随着所保存的视频的快速增加和服务请求的急剧增加我们无法保证自己搭建的服务器是否可以有一个良好的扩展性和稳定性。在这种情况下，通过市面上较为成熟的云端存储可以很好的解决这个问题，比如第二章所介绍的百度云BOS。所以对于视频管理平台数据存储的设计，我们不仅仅需要考虑视频管理平台所产生和依赖的数据如何进行存储，还需要考虑复杂视频转码流程下视频文件本身应该如何在BOS上进行合理存储。

视频管理平台在数据存储上可以划分为以下几个表：

1. 视频表（Video），主要用于存储视频管理列表所需要的大部分信息，比如视频播放量、视频点赞量、视频封面图、视频上下线状态、视频作者、视频上线时间等信息。

2. 审核表（Audit），主要用于存储视频审核列表所需要的大部分信息，比如视频封面图、视频审核状态、视频作者、视频上传时间等信息。

3. 用户表（User），主要用于存储用户的相关信息。

4. 视频文件表（Video-file），主要用于存储视频通用转码流程中产生视频文件的具体信息，比如各个归一化视频和可播放的发布视频等相关信息。

5. 临时文件表（Temp-file），主要用于存储视频自定义转码流程中产生视频文件的具体信息，比如视频水印片和视频头尾片文件等相关信息。

6. 脚本任务表（task），主要用于存储视频转码时各个脚本需要扫描的转码信息，比如脚本类型、想要透传给脚本的数据等。

视频管理平台的视频文件在BOS的存储流程如图3-4 BOS存储流程所示，转码流程中的一系列视频文件在BOS上的存储流程也是根据视频转码流程按需进行。我们在BOS上创建了三个Bucket分别用于不同状态下的视频存储，upload bucket是用于存储用户上传阶段源视频的文件，original bucket是用于存储视频转码流程中非最终状态的视频文件，publish bucket是用于存储视频转码流程中的用于播放和发布的最终状态视频文件。这种隔离式的划分可以很好的契合视频管理平台的需求，因为大多数情况下，我们可以简单的认定upload bucket中存储的是所有上传的视频文件，其中也包含了不符合既定规则的一部分违规视频文件，而这类违规视频文件是不会在original bucket中被找到的，我们通过单独存放源视频文件可以很好的隔离不符合上传视频要求的部分视频；除此之外，大多数情况下我们很少对中间状态的视频文件进行相关操作，我们不需要关注中间状态的视频是什么样的，因为我们更关心的是最终用于对外播放的视频应该是什么样的，所以一方面存储最终状态的bucket肯定会背负比其它两个bucket更重的带宽负担和请求压力，我们可以通过为publish bucket配置内容分发网络（CDN）的方式来中有效缓解高并发下的请求负担，提升响应速度和系统性能；另一个方面，由于对外我们只会暴露publish bucket的相关内容，upload bucket和original bucket的内容对于用户是透明的，这样可以的设计方案可以有效地防止源视频或中间状态视频外流，起到了一个很好的保护作用。

图3-4 BOS存储流程

3.4 本章小结

本章根据视频管理平台的功能特点和业务需求总体给出了视频管理平台的设计架构单个分析了视频转码流程和视频管理平台的数据存储的具体设计方案。一个好的设计可以让整个开发过程和后续的维护过程事半功倍，本章从根本出发，总结视频管理平台的功能和业务，深入思考视频管理平台各个模块之间的联系以及模块内重要流程，细致拆分各个模块，进而给出较为合理的设计方案，为下一章的实际开发奠定了坚实基础。

本章从视频管理平台的四个模块分别介绍视频管理平台系统的具体实现。

4.1 视频生产模块的实现

视频管理平台的视频生产模块是整个系统的关键，如图4-1视频生产页面所示，管理者在上传视频时需要选择要上传的视频文件并填写视频作者的用户id、视频标题、视频封面、视频需要自定义的片头片尾和水印模板、视频标签、视频关键字和视频描述等信息。

图4-1 视频生产页面

视频生产模块会在上传视频的时候请求一个videostart接口，该接口的作用是把视频文件本身的相关信息传递给后端，后端接受消息后返还一个FileId作为该文件的唯一凭证和BOS的各种凭证信息等，前端利用接收到返回的凭证信息请求BOS，并用返回的FileId作为文件名把视频源文件上传到upload bucket中，当上传完成之后，视频生产模块请求一个videofinish的接口，该接口用于通知后端该FileId对应的源视频文件已经上传完成，此时后端会返回给前端一个VideoId作为视频生产提交的唯一标识，并且后端此时会在video-file表中创建源文件记录，和添加task记录进行归一化视频转码。当管理用户填写完其他信息之后，点击提交按钮，前端会把用户填写的信息和刚刚上传成功返回的VideoId作为参数向后端请求create接口，该接口会创建一条待审核的审核记录，同时也是整个临时转码极其后续转码的流程开始。

视频转码的实现思路是通过开启扫描脚本实现的。扫描脚本是指在线上机器运行的、用于时刻扫描task表中相对应任务类型的记录信息，通过task记录的状态和透传信息判断是否数据是否正确并且是否需要执行该条记录对应的视频转码的操作。简单而言，视频转码流程的整个设计是基于task表中与各个脚本对应的记录推动的，每个脚本都只负责一个类型任务的后续操作，当脚本扫描task表中自己负责类型的数据记录时，根据逻辑判断和task表中透传给脚本的数据，可以很稳定的推动该脚本正确执行代码预先设定的操作。对于这么复杂的转码流程，单独通过task进行推动是一个很繁琐而且很容易出错的事情，所以我们通过crontab指令永不停息地执行两个扫描全表的脚本，一个用于检测父级转码是否已经正确完成，另一个用于执行现在需要转码操作的相关记录，这两个脚本不再是通过task来推动，而是本身去加快推进其他转码的快速进行。以create接口的下游脚本为例，create下游有一个Plan.php的脚本，该脚本会在线上时刻扫描task表中的类型为1的记录，循环执行下面的操作：如果该记录的状态是已执行或者执行中，那么脚本跳过该记录判断下一条的记录；如果该记录的状态是未执行，那么该脚本会校验该记录对应的转码流程是否处于这一环节而且各方面数据均正确无误，此时该脚本会把task记录的状态改为执行中并经过一定计算地向数据库中的多个表预插入数据记录，同时向task中插入接下来需要执行的脚本CopyToTemp.php对应的任务类型的记录插入到task表之中。整个流程就是通过这种思想来整体实现第三章所设计的视频转码流程的。

需要重点提到的是重试机制也是基于上述大思路下实现的。假设我们在视频转码过程中MCT发生了未知错误导致了转码失败，此时转码系统发现MCT的返回值不正确或者超时，那么此时该脚本会把导致错误的任务类型的状态重新改回未执行并在task表中记录重试次数和重试原因（改写task表的数据需要根据实际情况来判断，有时需要task中透传的数据等），那么下次与该任务记录对应的脚本在扫描时会发现此条，那么就会重新执行后续操作。

4.2 视频审核模块的实现

视频管理平台的视频审核模块主要工作是视频审核列表的展现和对视频审核操作的功能开发。视频审核列表需要根据视频的状态进行不同的展现。视频审核列表支持通过ElasticSearch来进行检索和排序，检索条件有标题、作者id、视频id、审核记录的状态、审核人员id、创建时间和审核时间等，排序的依据有创建视频、审核时间和审核记录的状态。我们通过ElasticSearch考验快速的进行检索和排序，快速显示页面信息。

如图4-2 视频审核列表页面所示，在视频上传的时候点击提交时就会在视频审核列表页中添加一条状态为转码中的数据记录，当视频处于转码中或者转码失败的时候视频并不能被管理人员审核，视频本身也无法被观看，管理人员所能知悉的就是视频标题、视频作者、视频上传提交时间和视频的当前状态。视频在转码完成之后会点击视频封面会弹出播放栏，并允许管理人员根据视频信息进行审核的“通过”和“不通过”按钮进行视频审核，此时也允许通过“编辑”按钮重新编辑视频标题。对于审核通过的视频会在视频管理页面中呈现，审核拒绝的视频则需要勾选或输入拒绝的理由，并且允许通过已经被拒绝的视频，防止出现误拒之后还需要重新上传的情况。无论是审核通过还是审核拒绝，只是视频对应审核记录的状态发送了改变而已，视频审核的数据信息仍然一直会在视频审核列表中存在，可以通过勾选对应的状态就可以检索。

图4-2 视频审核列表页面

4.3 视频管理模块的实现

视频管理平台的视频管理模块主要工作是对审核通过的视频进行管理。视频管理列表同样支持通过ElasticSearch来进行检索和排序，检索排序的条件在审核列表审核排序的基础上添加了发布时间和删除时间。

如图4-3 视频管理列表页面所示，审核通过视频默认处于上线状态，上线状态下的视频可以被外界所观看，我们可以通过视频管理列表页查看视频的总播放量、总点赞数和上下线状态。如果我们需要下线一个视频，即不想让外界继续观看，那么我们可以点击“删除”按钮并填入删除原因，此时视频就会被成功下线并在视频管理列表中显示下线原因；如果我们需要重新上线一个视频，即重新想让外界继续观看，那么我们可以点击“发布”按钮直接上线视频；如果我们需要对线上视频进行二次编辑，那么我们可以通过“编辑”按钮进入视频编辑页进行重新编辑，此时若修改视频文件本身，那么该视频会重新发起转码流程，相反若只是修改一些附带信息而视频文件没有发生改变，那么只需要通过修改数据库即可实现，不再需要发起转码。

视频管理列表页和视频审核列表页可以通过点击视频封面进行平台内播放，这个功能与审核列表转码完成之后点击封面播放的功能是共用一个组件。 如图4-4 视频管理平台内播放所示，该播放器基本实现了视频播放的需求，可以自动播放、拖拽

进度、暂停播放、音量调节、全屏显示以及下载按钮等功能。

图4-3 视频管理列表页面

图4-4 视频管理平台内播放页面

4.4 用户管理模块的实现

视频管理平台的用户管理模块的主要工作是添加与管理用户信息。如图4-5 用户管理列表页所示，我们可以很直观的查看现有已开通用户有哪些，并提供添加用户和编辑用户的功能。

通过用户管理系统，我们可以很直观的给能够上传视频的用户分配相应的权限，所以在上传视频必须要填入用户的id来表明视频的生产者是否拥有上传权限，如果没有或者用户不存在，那么视频将无法上传。用户管理系统的存在极大的方便了用户校验环节的工作量，而且我们可以通过用户管理系统非常方便地进行后续的用户功能扩展，比如结算分红等。

需要重点提到的是权限控制方面，用户管理模块需要依靠百度的Passport服务的支持。百度Passport服务指面向普通用户的百度账号服务，用于提供诸如账号注册、用户个人信息维护、统一登录、用户在线状态查询等服务。在视频管理平台进行操作的用户需要登录Passport，然后后端配合上文提到的SAF框架得到当前访问页面的用户的相关信息，进而实现权限校验和后续处理。

图4-5 用户管理列表页

7. 第5章总结与展望

总字数：1480

相似文献列表　文字复制比：2.3%(34)　疑似剽窃观点：(0)

1	我国食品安全法律保护研究 黄艳华(导师：漆丹) -《江西师范大学博士论文》- 2010-06-01	2.3%（34） 是否引证：否

原文内容　红色文字表示存在文字复制现象的内容; 绿色文字表示其中标明了引用的内容

第5章总结与展望

5.1 总结

到目前为止，基于C/S结构的视频管理平台的设计与实现已基本实现。

首先分析了整个视频平台的现状，发现了很多其他类型网站所不具备的技术挑战。通过分析可能出现的技术挑战和需要实现的基本功能之后，我开始调研挑战的解决方案和可能依赖的主流技术。往往一个问题会有多个不同的技术解决方案，对比与选择一个更加适合的尤为重要。然后我就开始了视频管理平台的设计过程，通过拆分视频管理平台的功能模块和前置的技术调研，我优先选择设计视频转码流程的实现方案，通过多次推翻与重构，最终确立了一个任务推动的脚本转码流的大方案，紧接着我设计了数据存储的设计方案，选择了通过MySQL + Redis + ElasticSearch的适用于高并发的数据存储方案，根据功能需求和业务逻辑多次设计关系型数据库的字段结构和表格关系。最后我根据划分的模块关系和设计思路开始了按模块进行开发的工作，无论是前端还是后端的开发，随着功能实现的增加，我发现越来越多的地方出现了代码重复和框架混乱，此时我不得不重新考虑修改设计方案和重构代码，就这样经过了多次重构和返回设计，最终基本实现了各个功能的开发工作。最终成功实现了一个能够解决高并发下系统性能不稳定、终端播放多样化、转码任务繁重不可靠等问题的视频管理平台。

5.2 展望

视频管理平台还有许多功能需要继续迭代和优化。

从运转模式的角度考虑，单纯通过平台管理用户进行视频的上传是一件很繁琐的事情，我们不可能每次都先从视频生产用户那里把视频要过来，然后再通过管理用户进行上传，这是低效而且很病态的运行模式。从本质来看，视频生产和视频上传是视频生产商需要做的，平台管理用户不需要再关注视频是如何获取和上传的，而是应该把更多的精力用于视频的审核和线上管理。所以从这个问题来看，我们需要修改现有逻辑，让视频生产和上传由用户自己来做，视频审核和线上管理让管理用户参与但对用户透明。我们可以重新打造一个用户使用的视频生产平台，用户可以通过这个平台自己上传视频并且查看自己上传的视频状态和线上的相关信息但无权操作。

从平台使用者的角度考虑，我们需要一个页面更加友善的落地页用于视频播放和视频推荐，现状是一个视频只能通过特定链接进行观看，而且只能观看一个视频，我们需要一个能够留住用户的推荐算法，把平台里类似这种视频的视频们挑选出来推荐给用户，所以我们需要一个良好的推荐算法和落地交互。

总而言之，我需要对自己的系统设计能力和开发实现能力提出新的要求，不断完善和改进视频管理平台，努力学习更加成熟使用的软甲开发技巧，为以后的发展奠定坚实基础。

参考文献

[1] 张海英,任君宁. 用“互联网思维”报道和策划新闻[J].新闻研究导刊,2016,7(24):211.

[2] TAT.yana． fis3初步学习体验[EB/OL]．http://www.alloyteam.com/2016/01/fis3-preliminary-learning-experiences/?hmsr=toutiao.io&utm_medium=toutiao.io&utm_source=toutiao.io,2016-01-05/2018-05-12．

[3] 昌维． Vue.js新手入门指南[EB/OL]．https://zhuanlan.zhihu.com/p/25659025,2017-05-25/2018-05-12．

[4] 毕安． 谈谈MVVM框架[EB/OL]．https://www.jianshu.com/p/ccea87433caf,2016-09-23/2018-05-12．

[5] 青年马士豆． 自己用过的框架[EB/OL]．https://www.jianshu.com/p/9e0429228132,2017-08-01/2018-05-12．
[6] 彭东稳. Nginx特性及原理介绍[EB/OL]. http://www.ywnds.com/?p=4040,2016-04-16/2018-05-12.
[7] 曾令武. 消息队列及常见消息队列介绍[EB/OL]. https://cloud.tencent.com/developer/article/1006035,2017-09-29/2018-05-13.
[8] 赵飞祥． 纲举目张：打通MySQL架构和业务的任督二脉[EB/OL]．https://mp.weixin.qq.com/s/wS6bRSAPplhK6tRO3pb5xg,2018-01-22/2018-05-13．
[9] 徐刘根. Redis简介以及和其他缓存数据库的区别[EB/OL]. https://blog.csdn.net/xlgen157387/article/details/60761232,2017-03-07/2018-05-13.
[10] 唐诚. Redis数据库在微博系统中的实践[J].厦门城市职业学院学报,2012,14(03):55-59.
[11] 孟方园. ElasticSearch分布式搜索引擎在导航大数据检索中的应用[A]. 中国卫星导航系统管理办公室学术交流中心.第八届中国卫星导航学术年会论文集——S01卫星导航应用技术[C].中国卫星导航系统管理办公室学术交流中心:,2017:4.
[12] 沙恒． ElasticSearch学习总结—原理篇 [EB/OL]．http://www.shaheng.me/blog/2015/06/elasticsearch--.html,2015-06-03/2018-05-14．
[13] 赵纪伟. 基于云平台的视频监控和存储系统研究与实现[D].内蒙古大学,2017.
[14] 江泓,金蓓弘.支持可扩展性的设计模式协作研究[J].计算机工程与设计,2007(06):1244-1247.
[15] 刘逻. 软件可靠性设计技术应用研究[D].中国科学院研究生院（长春光学精密机械与物理研究所）,2013.
[16] 李云云.浅析B/S和C/S体系结构[J].科学之友,2011(01):6-8.
[17] 林越,王翠珍.浅谈面向对象开发思想与软件设计架构分析[J].信息通信,2016(03):152-154.
[18] MU Huaxin International School Beijing University of Posts and Telecommunications Beijing,PRC JIANG Shuai School of Software Engineering Beijing University of Posts and Telecommunications Beijing,PRC. Design Patterns in Software Development[A]. IEEE Beijing Section、IEEE China Council、Beijing University of Technology、Beijing University of Posts and Telecommunications、The Hong Kong Polytechnic University、Andhra University.Proceedings of 2011 IEEE 2nd International Conference on Software Engineering and Service Science(ICSESS 2011)[C].IEEE Beijing Section、IEEE China Council、Beijing University of Technology、Beijing University of Posts and Telecommunications、The Hong Kong Polytechnic University、Andhra University:,2011:4.
[19] Yungeng Xu. Design and Implementation of a Multi Video Transcoding Queue Based on MySQL and FFMPEG[A]. IEEE Beijing Section.Proceedings of 2015 IEEE 6th International Conference on Software Engineering and Service Science(ICSESS 2015)[C].IEEE Beijing Section:,2015:4.
[20] 余圣发,陈曾平,庄钊文.针对网络视频应用的视频转码技术综述[J].通信学报,2007(01):111-118.
[21] 尚书林,杜清秀,卢汉清,唐小军.视频转码技术研究现状与最新进展[J].自动化学报,2007(12):1233-1241.
[22] 邹艳玲,宋晓辉,常征.基于NGINX的Web集中管理架构设计与实现[J].中国教育网络,2017(12):48-50.
[23] 吴锋. 基于Hadoop平台的视频转码系统设计与实现[D].上海交通大学,2014.
[24] 刘炳均. 基于云平台的分布式视频转码系统的设计与实现[D].中山大学,2015.
[25] 李亚飞. 分布式视频转码系统的设计与实现[D].哈尔滨工业大学,2014.
[26] 黄锡波,杨浪,钟建坤.迭代开发模式运用于软件综合实训的探究[J].中国科教创新导刊,2011(28):201-202.

致谢

光阴似箭，日月如梭，一转眼我四年的本科生活即将告一段落，在此临别之际我想要由衷的感谢兢兢业业教书育人的老师、包容理解我的父母和尽心尽力帮助我的同学们，是你们让我的大学生活变得丰富多彩、变得简单而快乐。

首先特别鸣谢我的论文指导老师王康平讲师，王老师严谨负责的态度与因材施教的教学手段深深地影响着我，让我更加全面和系统的思考问题，并对这篇论文的撰写提供了莫大的帮助。

其次非常感谢生我养我的父母，在我迷茫不安的时候，他们总会用爱温暖我并照亮前方的路，让我明白不管何时何地，家和父母一直在那里售后着我。

同时非常感谢身边的同学们，在我大学四年中他们不求回报地为我提供了很多或大或小的帮助，和他们在一起让我的每天都过得很充实、很快乐。

最后郑重感谢我的母校，是它为我提供了一个较高的平台，让我有机会与那么多优秀的人结交，并为我在未来的求职和发展中奠定了良好基础。

4.单篇最大文字复制比：被检测文献与所有相似文献比对后，重合字数占总字数的比例最大的那一篇文献的文字复制比

5.指标是由系统根据《学术论文不端行为的界定标准》自动生成的

6.红色文字表示文字复制部分;绿色文字表示引用部分

7.本报告单仅对您所选择比对资源范围内检测结果负责

amlc@cnki.net

http://check.cnki.net/

http://e.weibo.com/u/3194559873/