上海双地信息系统有限公司

上海图书馆  
众包服务平台和基础服务平台

概要设计方案

目 录

[第一章 众包服务平台设计 7](#_Toc466553766)

[1. 概述 7](#_Toc466553767)

[1.1. 项目背景 7](#_Toc466553768)

[1.2. 项目目标和内容 7](#_Toc466553769)

[2. 系统架构 8](#_Toc466553770)

[2.1. 整体架构(B/S统一架构) 8](#_Toc466553771)

[2.2. 系统实现技术 10](#_Toc466553772)

[3系统设计 0](#_Toc466553773)

[3.1. 众包工作流程主要模块 0](#_Toc466553774)

[3.2. 众包工作流程 0](#_Toc466553775)

[3.3. 众包项目策划 2](#_Toc466553776)

[3.4. 任务发布 2](#_Toc466553777)

[3.4.1. 任务发布方式 2](#_Toc466553778)

[3.4.2. 项目组织和发布 3](#_Toc466553779)

[3.4.3. 任务显示界面 4](#_Toc466553780)

[3.5. 任务加工 6](#_Toc466553781)

[3.5.1. 加工原则 6](#_Toc466553782)

[3.5.2. 资格判定 7](#_Toc466553783)

[3.5.3. 加工功能 8](#_Toc466553784)

[3.5.4. 系统功能 9](#_Toc466553785)

[3.6. 审核 9](#_Toc466553786)

[3.6.1. 系统合法性校验 9](#_Toc466553787)

[3.6.2. 管理员审核 10](#_Toc466553788)

[3.7. 其他 11](#_Toc466553789)

[3.7.1. 用户管理机制 11](#_Toc466553790)

[3.7.2. 激励机制 11](#_Toc466553791)

[3.7.3. 用户反馈 11](#_Toc466553792)

[3.7.4. 其它界面 11](#_Toc466553793)

[3.8. 加工工具 12](#_Toc466553794)

[3.8.1. 书籍文献数据录入 12](#_Toc466553795)

[3.8.2. 期刊文献数据录入 13](#_Toc466553796)

[3.8.3. 期刊论文数据录入 14](#_Toc466553797)

[3.8.4. 科技文献数据录入 15](#_Toc466553798)

[3.8.5. 书籍、期刊、科技文献的子章节录入 15](#_Toc466553799)

[3.8.6. 图像文件的数据录入 15](#_Toc466553800)

[3.8.7. Mechanism、Person、Interactiv animation数据录入 16](#_Toc466553801)

[3.8.8. Softwear、Videos、Cax-files 数据录入 16](#_Toc466553802)

[3.8.9. 数据查询、修改、删除 16](#_Toc466553803)

[第二章 基础服务平台 18](#_Toc466553804)

[1. 整体设计 18](#_Toc466553805)

[1.1. 核心架构设计要素 18](#_Toc466553806)

[1.1.1. 性能 18](#_Toc466553807)

[1.1.2. 可用性 22](#_Toc466553808)

[1.1.3. 伸缩性 22](#_Toc466553809)

[1.1.4. 扩展性 23](#_Toc466553810)

[1.1.5. 安全性 23](#_Toc466553811)

[1.2. 架构模式 23](#_Toc466553812)

[1.2.1. 分层（逻辑上） 23](#_Toc466553813)

[1.2.2. 分割 24](#_Toc466553814)

[1.2.3. 分布式 24](#_Toc466553815)

[1.2.4. 集群 25](#_Toc466553816)

[1.2.5. 缓存 25](#_Toc466553817)

[1.2.6. 异步 25](#_Toc466553818)

[1.2.7. 冗余 25](#_Toc466553819)

[1.2.8. 自动化 25](#_Toc466553820)

[1.2.9. 安全 26](#_Toc466553821)

[2. 服务平台架构 27](#_Toc466553822)

[2.1. 服务平台整体架构 27](#_Toc466553823)

[2.2. 架构分层图 28](#_Toc466553824)

[2.3. 整体架构说明 28](#_Toc466553825)

[2.4. 未来发展需求支撑说明 30](#_Toc466553826)

[3. 数据框架设计 32](#_Toc466553827)

[3.1. 业务数据架构 32](#_Toc466553828)

[3.2. 数据流程 32](#_Toc466553829)

[3.2.1. 流程A 32](#_Toc466553830)

[3.2.2. 流程B 33](#_Toc466553831)

[3.3. 服务平台数据库集群 34](#_Toc466553832)

[3.3.1. 数据库集群架构 34](#_Toc466553833)

[3.3.2. 硬件环境 34](#_Toc466553834)

[3.4. 搜索引擎架构 35](#_Toc466553835)

[3.4.1. 规模定义 35](#_Toc466553836)

[3.4.2. 服务平台Solr架构 35](#_Toc466553837)

[3.4.3. 中文分词 36](#_Toc466553838)

[3.4.4. 搜索引擎测试标准 36](#_Toc466553839)

[4. 应用框架设计 38](#_Toc466553840)

[4.1. 应用架构 38](#_Toc466553841)

[4.2. 技术详解 38](#_Toc466553842)

[4.2.1. 基础框架——Spring 3.2.4 （MVC） 38](#_Toc466553843)

[4.2.2. ORM——Hibernate 4.3.7 40](#_Toc466553844)

[4.2.3. 任务调度——Quartz2.2.1 40](#_Toc466553845)

[4.2.4. 日志——Log4j2 40](#_Toc466553846)

[5. 日志框架设计 41](#_Toc466553847)

[5.1. 日志架构 41](#_Toc466553848)

[5.2. 日志存储 41](#_Toc466553849)

[5.2.1. 日志级别 41](#_Toc466553850)

[5.2.2. 日志类型 41](#_Toc466553851)

[5.2.3. 统一日志内容 42](#_Toc466553852)

[6. 安全设计 43](#_Toc466553853)

[6.1. 安全要素 43](#_Toc466553854)

[6.2. 信息安全 43](#_Toc466553855)

[6.3. 应用安全 44](#_Toc466553856)

[6.3.1. 安全威胁 44](#_Toc466553857)

[6.3.2. 应用安全防火墙 ModSecurity 2.9 44](#_Toc466553858)

[6.4. 系统安全 44](#_Toc466553859)

[6.5. 物理安全 45](#_Toc466553860)

[7. 性能测试设计 46](#_Toc466553861)

[7.1. 测试工具 HP LoadRunner v11.0 46](#_Toc466553862)

[7.2. 性能测试方法 46](#_Toc466553863)

[7.2.1. 性能测试 46](#_Toc466553864)

[7.2.2. 负载测试 46](#_Toc466553865)

[7.2.3. 压力测试 46](#_Toc466553866)

[7.2.4. 稳定性测试 46](#_Toc466553867)

[8. 自动化功能测试设计 47](#_Toc466553868)

[8.1. 测试工具 (HP QTP v11.0) 47](#_Toc466553869)

[8.2. 测试流程 47](#_Toc466553870)

[9. 自动化安全测试 49](#_Toc466553871)

[9.1. 测试工具 49](#_Toc466553872)

[9.2. 测试流程 49](#_Toc466553873)

[第三章 镜像站服务 50](#_Toc466553874)

[1. 整体设计 50](#_Toc466553875)

[1.1. 整体架构 50](#_Toc466553876)

[1.2. 加密基础库 50](#_Toc466553877)

[1.3. 镜像站安全管理 51](#_Toc466553878)

[1.3.1. 镜像站分布点管理 51](#_Toc466553879)

[1.3.2. 镜像站更新包管理 51](#_Toc466553880)

[1.3.3. 镜像站更新包安全下发工具 51](#_Toc466553881)

[1.3.4. 关于镜像站版本更新 51](#_Toc466553882)

[1.4. 安全镜像站服务 52](#_Toc466553883)

[1.4.1. 镜像站嵌入式WEB应用服务 52](#_Toc466553884)

[1.4.2. 镜像站嵌入式全文索引 54](#_Toc466553885)

[1.4.3. 镜像站嵌入式数据库 54](#_Toc466553886)

[1.4.4. 镜像站嵌入式资源库 54](#_Toc466553887)

[1.4.5. 镜像站更新包接收工具 55](#_Toc466553888)

[2. 技术选型和说明 56](#_Toc466553889)

[2.1. 安全应用服务 （Lighttpd） 56](#_Toc466553890)

[2.2. 安全检索服务（CLucene） 56](#_Toc466553891)

[2.3. 安全数据库（Sqlite） 56](#_Toc466553892)

[2.4. 安全的资源存储（S-VFS） 57](#_Toc466553893)

# 众包服务平台设计

# 概述

## 项目背景

上海图书馆众包服务平台，利用现代化信息技术和服务手段，联合海内外专家，在近代文献等珍贵文献抢救和发掘方面提供的共建共享平台。该平台促进了图书馆数字资源建设的优势互补，提升知识服务的能力与共享水平，丰富数字资源，实现资源共享的全面性、多样性、互动性和个性化，为广大读者用户提供良好的信息参考和决策咨询服务，也为海内外研究国学的人士获取信息提供更多的便利。

系统建成后，随着数据库技术和网络技术的成熟,以及新业务的拓展和管理方式的改变等因素，系统平台需要根据业务和管理的变化进行相应的频繁调整，使之满足业务的开展。为保证系统能始终保持稳定的运行，任何系统变更都必须绝对正确，这给系统的管理和安全运营带来了极大的挑战。

本系统平台的主要做互联网上的知识加工和管理。共建系统最终目标是建成了网络环境下集原数据采集、印本扫描、数据加工、审核校对、数据规范、引文归一、查询检索和统计分析等功能为一体的数据加工和管理系统，同时，对各加工人员进行了明确的角色权限管理和任务的自动分配、监控。系统涉及了图书馆的加工任务管理员、扫描人员、数据采集人员、数据加工人员和数据审核人员等工作人员，通过任务识别、分配、监控的方式实现各工作人员的任务管理，通过扫描、OCR识别的方式实现数据的数字化，加工人员通过标引、加条等进行数据的加工、审核、校对，并最终提交到成品库中。

元数据是解释数据的数据，关于数据的数据（data about data）。在图书馆与信息界，元数据被定义为：提供关于信息资源或数据的一种结构化的数据,是对信息资源的结构化的描述。其作用为：描述信息资源或数据本身的特征和属性,规定数字化信息的组织,具有定位、发现、证明、评估、选择等功能。为了使现有的篇名数据加工系统更好的满足上图文献加工平台对高质量元数据的需求，提出了对现有系统进行改造与升级的项目。

## 项目目标和内容

项目建设内容如下：

* 书籍、期刊、图像、其它类型数据的系统录入
* 系统中图像数据的二次编辑，保存为pdf文件
* 系统录入数据的检索、修改、删除

# 系统架构

## 整体架构(B/S统一架构)

结合上海图书馆众包服务平台和双地信息系统公司长期维护上图项目的经验，考虑到上海图书馆众包服务平台的中长期发展规划，综合系统的网络结构、应用、管理、系统性能、系统稳定性等各个方面，提出了文献索引数据生产系统改造与升级的项目方案。

现有篇名数据加工系统是一个B/S架构的系统项目，以多种系统集群的方式提供强大的可伸缩性和可扩展性。系统架构总体上为三层结构，如下图所示：



图1. 系统架构图

* 数字化加工客户端

用于提交扫描文件，提交元数据文献索引信息的客户端程序。

* 数字化加工管理端

用于管理平台，配置人员，定义项目及其工作流程，统计项目等。

* 应用服务器

为客户端提供系统的接入接口，后台工作流引擎保证项目状态的正常流转。

* 数据库服务器

保存应用服务器中的数据如（人员，权限，工作流引擎等等）

* 文件共享服务器

保存项目中共享的扫描件图片，OCR文档，校验文档等文件

本项目的主要功能是可以支持多种类型文献的知识加工，根据不同文献的元数据，以及一些配置选项，系统可以自动生成相应的知识加工流水线。

操作流程如下：



以期刊为例，系统建立3条流水线，3条流水线如下：



其中浅绿色底色是刊简介的知识加工流程；褐色底色是分篇著录的加工流程；黄色底色是已有著录信息的文献分篇著录的加工流程。每个加工流程（即流水线）由多步工序组成。

## 系统实现技术

框架： GWT Smart Client，SpringMVC，Hibernate

开发语言： Java

部署环境： Linux Server

数据库： SQL Server 2008（或2005，数据库还原是向下兼容，因此版本不能低于分校的数据库版本）



# 3系统设计

## 众包工作流程主要模块

众包工作流程分为四个主要流程：众包项目策划、任务包发布、用户加工任务、任务审核。

任务审核

任务

加工

任务

发布

众包项目

策划

1. 

## 众包工作流程

具体流程如图1所示，各环节具体功能详见文档对应内容。



**图1：众包具体流程**

## 众包项目策划

众包项目的策划包括以下几个步骤和流程，确定众包内容、相关规则、项目期限、项目目标等。

## 任务发布

#### 任务发布方式

发包方通常是企业或者拥有工作任务需要解决的个人。对于发包方来说，其发布任务的方式有两种：一种是在公司网站上直接发布，以悬赏的方式吸引众多的[网民](http://wiki.mbalib.com/wiki/%E7%BD%91%E6%B0%91)来参与解决问题，这种方式避开了中介机构从而使问题解决的成本更低。

还有另外一种方式，就是通过中介机构，通常是网站社区协会等，发包方与新型网络签订合约，合约中包括了需要解决的问题、价格和[售后服务](http://wiki.mbalib.com/wiki/%E5%94%AE%E5%90%8E%E6%9C%8D%E5%8A%A1)条款等。

#### 项目组织和发布

1. 项目建立：
   1. 项目管理员：系统管理员负责建立项目并指定项目管理员，其余活动由项目管理员进行。
   2. 建立项目内容：建立或导入项目元数据模板、批量导入对应项目数据（mdb或图片）、导入建立项目说明及使用帮助、配置基本校验规则、设定项目是否需要资格审核，设定训练页内容；
   3. 数据导入功能：将线下建立好的元数据模板、或已加工完成的民国期刊篇名库、报纸相关数据导入；
   4. 元数据模板直接在数据库中建立；
   5. 元数据模板进行修改，包括新增字段或改变字段类型，修改后的元数据模板可以另存为其它名称的模板；
   6. 能够将导入的元数据和图片（jpg、pdf）自动进行勾连；
   7. 项目说明及使用帮助在线进行编辑；
   8. 资格审核以训练页的形式展开，项目管理员设定该项目是否需要资格审核。
   9. 训练页：训练页功能同项目正式任务功能。
2. 任务发布：由项目管理员操作。
   1. 任务按最小单位进行发布，如：民国期刊最小单位为篇，广告为单条广告（不涉及到群组广告和跨页广告）；
   2. 项目管理员可设定发布任务的条数，由系统随机抽取数据并发布；项目管理员也可手工批量选择任务数据并发布；
   3. 系统记录每次任务发放的时间、数量（条）、发布者（账号）以及对应任务数据的id；记录形式可以为txt、excel等；提供记录查询功能，能够按时间或账号进行查询；项目管理员只能查询本账号的任务发布记录，系统管理员可以查看所有。

#### 任务显示界面

1. 列表的形式分别展示不同的项目、项目说明；点击某项目，则进入该项目任务显示界面
2. 缩略图的形式分别展示不同的任务，包括任务对应的文献（图片或pdf）首页的缩略图，文献信息、任务状态及最新修改记录；
   1. 缩略图默认显示任务中的第一张图片，显示不出则用系统默认图片；
   2. 文献信息包括文献名称、作者等文献基本信息；显示的信息个数以人工设定，默认为3个；信息字段长度固定，不能显示的部分以……代替；
   3. 任务状态包括“新任务”、“加工中”和“已完成”。“新任务”即最新发布的任务，“加工中”即任务正在进行，“已完成”即已关闭的任务；
   4. 任务状态由审核人员或项目管理员批量进行修改；若该任务目前有用户领取，则不能修改任务状态；
   5. 需在系统设置强制关闭任务或项目的权限，任务强制关闭权限赋予项目管理员，项目强制关闭权限赋予系统管理员；
   6. 能够按任务状态对任务进行排序或筛选；
   7. 最新修改记录格式为：XXX（用户名）于XXX（时间）修改。修改记录取最近通过审核的修改，时间为用户加工通过审核的时间；
3. 提供了检索功能，按文献名称、作者查询；
4. 任务总量：任务显示界面显示任务区共有多少任务；
5. 在任务加工界面高亮显示使用帮助、训练页按钮；

## 任务加工

众包项目的任务加工包括以下几个步骤和流程，加工原则、资格判定、加工功能、系统功能。

#### 加工原则

1. 项目管理员可更改单个任务可以被多少个用户接受（被一定数量的用户更改后该项目结束），这里需要做好每个用户的版本保存；
2. 项目管理员设定单个用户最多可以同时接受多少个任务（任务被审核之后即结束）；
3. 若项目需要资格审查，用户接受任务时首先需要进行资格审查，用户只有通过资格审查才可以领取任务（训练任务）；
4. 用户没有通过资格审查，则在用户领取任务时，以弹窗通知用户需要资格审查，可在弹窗中附上“训练页”按钮，用户点击此按钮进入训练页；用户也可在任务显示界面点击“训练页”按键进入；
5. 所有弹窗中的提示信息均可由系统管理员修改（下同）；
6. 用户只能接受状态为“新任务”的任务；用户接受任务后（所谓的接受任务，是指用户点击加工界面中的修改按钮），该任务状态自动变为“加工中”；
7. 当前任务若有人接受，系统以弹窗形式提示其余用户不能接受任务；
8. 用户只能在线进行加工，不提供文献下载功能；
9. 项目管理员可以设置用户一段时间内浏览任务量，时间和浏览量均可改变；如果用户浏览量超出限制，以弹窗形式提示用户不能够再浏览；
10. 用户对任务的加工具有时间限制，时间限制可以由项目管理员配置，接近时限时系统会提醒用户。若在时间范围内没有提交其结果，则认为本次任务失效，重新发布到任务区，状态为“新任务”；
11. 用户可以保存其正在加工的任务，以便继续加工；
12. 用户加工的数据以最小单位进行提交；
13. 提交的数据需进行审核，审核由审核人员进行；
14. 审核人员或是项目管理员，或是由项目管理员指定（也可能是用户）；

#### 资格判定

1. 资格判定以训练页的形式开展；
2. 训练页界面、功能同具体加工界面
3. 用户提交的结果经审核通过后方具备项目加工权限；
4. 训练页的内容，在训练后，不做保存，可以重复训练使用；

#### 加工功能

1. 显示功能：图像可按比例缩放；图像可拖动；
2. 编辑功能：用户点击按钮，文本显示区域变为可编辑状态；
3. 列表选择功能：根据项目初始预设的列表供用户选择；可以进行多选；
4. 保存功能：用户可对未加工完成的记录进行保存，供以后继续加工；
5. 提交功能：用户对加工完成的任务进行提交，供审核人员审核；
6. 上传功能：用户能上传文献到系统中，附件提交可能有安全性问题，功能先开发，暂不开放给用户使用；
7. 使用帮助功能：任务的使用帮助，主要是任务的操作说明以及加工中可能遇到的问题等；
8. 联系项目管理人员或审核人员的功能；

#### 系统功能

1. 统计功能：
2. 项目统计：共参与项目的用户数、项目创建时间、项目关闭时间、项目周期、项目任务数、项目进度（以完成的项目/项目任务数）、项目管理员、质量管理人员；
3. 任务统计：任务发布时间、任务关闭时间、任务周期、参与加工用户数、通过审核的用户加工数，未通过审核的用户加工数；
4. 统计结果以列表形式展开，可导出为excel表；
5. 历史记录保存：系统需记录任务关闭前的用户提交的加工记录；
6. 权限设定：系统需针对每个项目划分相应的权限，权限需灵活可配；权限包括项目建立权限，项目管理权限、项目浏览权限、项目加工权限、项目审核权限、项目关闭权限。
7. 系统安全：需考虑系统数据的安全性；

## 审核

#### 系统合法性校验

1. 系统合法性校验是一些简单的校验功能，具有自动校验功能；
2. 通过系统合法性校验的加工操作，系统将相关信息传递给系统管理员审核；
3. 没有通过系统合法性校验的加工操作，系统在审核后弹出提示框，包含其错误原因；

#### 管理员审核

1. 审核
2. 管理员权限：管理员分为系统管理员、项目管理员和审核人员。系统管理员具备所有的众包工作管理权限，项目管理员可对一个或多个项目进行管理，审核人员仅能够浏览任务及审核用户加工结果；所有管理员均不具有项目加工权限；
3. 可赋予满足一定贡献度的用户审核权限；
4. 系统按顺序分配审核任务给审核人员，审核人员对通过系统校验的数据进行审核，并可修改任务状态；
5. 项目管理员可查看任务的历史修改记录；
6. 通过审核的任务，系统用修改后的数据替换之前的数据，并且经过质量评估，若审核人员认为其质量已满足要求,则将任务状态变为"已完成",若认为尚不满足要求,还有再次加工的必要，则该任务继续发放,任务状态变为"新任务"，重新开放任务供所有用户加工；
7. 对于没通过审核的数据，系统以email或站内信形式通知用户，并记录其加工记录；
8. 审核人员对用户的加工记录进行审核，查看是否属于恶意修改，可警告用户或通知系统管理员将用户加入黑名单禁止其再接受任务；
9. 系统需记录审核人员的审核记录，并以列表形式展现，内容包括账号、用户记录提交时间、审核时间、被审核用户账号、审核结果；审核记录可以excel形式导出。
10. 项目关闭
11. 系统管理员可关闭项目；
12. 关闭的项目，可根据实际需求由系统管理员在任务区将项目全部数据移除；
13. 任务完成后系统需提供数据导出功能，导出数据包括用户的加工记录、加工后的项目任务数据；

## 其他

#### 用户管理机制

1. 用户注册
2. 用户已服务平台上的个人用户作为用户来源。
3. 在众包平台的页面上点击注册直接转入平台个人用户注册。
4. 用户权限设定
5. 游客仅有浏览任务列表的权限，不能查看具体任务；
6. 用户具体权限的设定；

#### 激励机制

以排名贡献的形式对用户的加工任务数量做统计排序；

#### 用户反馈

用户具有留言反馈的功能，用户可以已email等形式反馈意见和建议。

#### 其它界面

1. 用户界面：显示内容包含用户基本信息、我的任务、任务记录、站内信等，具有审核权限的用户还需显示其待审核的任务；
2. 用户基本信息：显示用户注册时的信息，用户可在此修改基本信息；
3. 我的任务：指用户保存的任务；
4. 任务记录：指用户参与过的任务列表，列表按项目排序，显示内容包括项目名称、任务基本信息、任务当前状态、参与时间、修改内容、审核结果。
5. 审核人员界面：分为已审核任务、待审核任务、用户反馈、任务管理；已审核记录参照2.1中点8所述；待审核任务以列表形式展示其需要审核的用户加工记录；用户反馈以列表形式展示用户反馈意见，含用户账号、反馈时间、反馈信息；任务管理界面以列表形式展示项目任务及其状态，审核人员可在此修改任务状态。
6. 项目管理员界面：包含其所属项目基本信息、项目管理、用户反馈意见、任务状态管理、用户管理、数据导入导出、统计功能、若项目管理员同时担负审核人员角色，则还需添加审核界面；
7. 系统管理员界面：包含所有项目基本信息、用户列表、项目管理员列表、审核人员列表、权限管理、统计功能等。

## 加工工具

#### 书籍文献数据录入

书籍文献数据著录流程为：

上图首先对需要著录的书籍文献进行扫面和非专业性著录，生成PDF和MDB（PDF的存储路径，以及MDB的数据格式要求采用系统可识别的统一模板，保证系统能将MDB中存储的书籍信息和PDF做勾连。下同），上传至系统，系统处理上图整理的书籍信息，处理成元数据保存至数据库。上图可以在系统中对系统中书籍信息进行二次专业性著录。

|  |  |
| --- | --- |
| 元素名称 | 元素修饰词 |
| 题名 | 题名、交替题名 |
| 主要责任者 | 主要责任者、责任方式 |
| 分类号 | 按中图分类法(书籍上印刷的分类号) |
| 主题词 | 上图书籍可以从marc格式导入主题词 |
| 网页分类 |  |
| 出版者 | 出版单位 |
| 其他责任者 | 其他责者、责任方式 |
| 资源描述 | 目录（文字形式） |
| 摘要 |
| 附注 |
| 章节 | 列出所有章节及其章节名称 |
| 日期 | 出版日期 |
| 资源类型 | 图书 |
| ISBN |
| 语种 |  |
| 相关资源 | 包含、包含于、参照、被参照、等相关资源的URI、ISBN、DOI等（预留项） |
| 时空范围 | 出版地点 |
| 元素名称 | 元素修饰词 |
| 版本 |  |
| 原文链接 | 整本、各章节的PDF路径 |
| 数据提供商 |  |

#### 期刊文献数据录入

期刊文献的著录流程为：

上图首先对需要著录的期刊文献进行扫面和非专业性著录，生成PDF和MDB，上传至系统，系统分析上图整理的期刊文献信息，处理成数据保存至数据库。上图可以对系统中期刊文献信息进行二次专业性著录。

|  |  |
| --- | --- |
| 元素名称 | 元素修饰词 |
| 题名 | 刊名 |
| 简体刊名 |
| 网页分类 | 机械设计 |
| 机械制造 |
| 出版者 | 主办单位 |
| 资源描述 | 出版地 |
| 期刊 |
| 日期 | 创刊日期 |
| 停刊日期 |
| 资源类型 | 期刊 |
| 资源标识 | ISSN |
| CN |
| 语种 | 中文/外文 |
| 相关资源 | 曾用名、改名 |
| 覆盖范围 | 年卷期范围 |

#### 期刊论文数据录入

期刊论文的著录流程为：

上图首先对期刊论文进行扫描和非专业性著录，生成PDF和MDB文件，上传至系统，系统分析上图已经加工的文献信息，保存至数据库。上图可以对系统中期刊论文信息进行二次专业性著录。

|  |  |
| --- | --- |
| 元素名称 | 元素修饰词 |
| 题名 | 题名 |
| 交替题名 |
| 主要责任者 | 主要责任著者 |
| 责任方式 |
| 机构 |
| 网页分类 |  |
| 其它责任者 | 其它责者、责任方式 |
| 主题 | 主题词（根据专业词表选择）+关键字（自由输入） |
| 分类号 | 科图分类法 |
| 资源描述 | 摘要 |
| 页码 |
| 日期 | 论文发表时间 |
| 资源类型 | 期刊论文 |
| 格式 | PDF |
| 来源 | 期刊信息 |
| 年卷期 | 论文所在期刊年卷期 |
| 语种 | 语种 |
| 相关资源 | 参考文献、译自、改编自 |
| 个别元素 | |
| 原文链接 | PDF存储路径 |
| 数据供应商 | 自动生成 |

#### 科技文献数据录入

科技文献的著录流程：

上图首先对科技文献进行扫描和非专业性著录，生成PDF和MDB文件，上传至系统，系统处理上图已经加工的文献信息。上图可以在系统中对系统中科技文献信息进行二次专业性著录。

#### 书籍、期刊、科技文献的子章节录入

子元数据录入方式有两种：

1、单独扫描，生成PDF和MDB文件录入系统。子章节录入时必须关联父元数据，如章节信息单独录入系统时，必须先录入章节的书籍信息。

2、在书籍、期刊、科技文献的数据基础上新建子章节。如对于书的章节信息的录入，可以在书的PDF基础上切割，生成章节对应的PDF文件，填写章节的数据信息，实现章节录入。

#### 图像文件的数据录入

图像文件著录流程分为两种：

1、对已经著录的书籍或期刊文献中的图像进行著录。系统提供将图片单独抠图的界面，用户可以对抠出的新图进行二次录著，保存著录信息。

2、对系统为收集来的其它图片进行著录。系统提供平台外的图片著录的接口，上图可以上传单独的图片，并对它进行录著，保存著录信息。

|  |  |
| --- | --- |
| 元素名称 | 元素修饰词 |
| 题名 | 图片名称 |
| 主题 | 主题词+关键字 |
| 创作者 | 图片创作者 |
| 网页分类 |  |
| 资源描述 | 图片内容介绍 |
| 资源类型 | 图片/图纸/表格/模型… |
| 资源标识 | ISSN |
| CN |
| PDF存放路劲 |
| 语种 | 中文/英文/… |
| 来源 | 期刊或图书信息 |
| 相关资源 | 参考文献 |
| 日期 | 图片出版、拍摄、制作时间 |

#### Mechanism、Person、Interactiv animation数据录入

Mechanism、Person、Interactiv animation数据著录流程为：

上图首先在系统中输入Mechanism、Person、Interactiv animation数据信息，在从图片库中关联一个或者多个图像数据，生成Mechanism、Person、Interactiv animation数据o

#### Softwear、Videos、Cax-files 数据录入

Softwear、Videos、Cax-files数据录入流程：

系统提供以Softwear、Videos、Cax-files数据录入界面，同时提供三种数据附件上传接口，保存数据录入功能。

#### 数据查询、修改、删除

所有录入数据都可以被查询。

数据查询机制：在对应的元数据类别下，查询元数据对应的关键字。其中文字类型的数据采用模糊查询的方式进行查询，日期类型、数字的数据采用范围查询的方式查询。

录入系统的数据可以进行再次修改。

录入系统的数据可以删除。

# 基础服务平台

# 整体设计

## 核心架构设计要素

设计架构首先要明确目标，明确目标前我们先要了解评判目标的标准，因此，我们专门设立一章就设计要素和评判标准进行整理。

### 性能

性能是整个服务平台非常重要的一个环节，我们架构设计的目标就是通过合理的技术规划和设计来提供最佳的服务性能，为用户提高效率和用户体验。我们需要处理的是海量知识数据，如何加快知识检索、知识导航等等最常用的业务操作是我们需要妥善解决的最紧迫的问题。

性能指标如下所示：

#### 服务端性能指标

* TPS(Transaction Per Second) 每秒事务数
  + 每秒钟系统能够处理的交易或事务的数量，它是衡量系统处 理能力的重要指标。TPS 是 LoadRunner 中重要的性能参数指标。
* PV(Page View) 页面访问数
  + 用户通过浏览器访问页面，对应用服务器产生的每一次请求， 记为一个 PV。
* Concurrency 并发数
  + 狭义的并发，即所有的用户在同一时刻做同一件事情或操作，这种操作一般针对同一类型的业务，或者所有用户进行完全一样的操作，目的是测试数据库和程序对并发操作的处理。狭义并发强调对系统的请求操作是完全相同的，多适用于性能测试、负载测试、压力测试、稳定性测试场景；
  + 广义的并发，即多个用户对系统发出了请求或者进行了操作，但是这些请求或操作可以是不同的。对整个系统而言，仍然有很多用户同时进行操作。广义并发不限制对系统的请求操作，多适用于混合场景、稳定性测试 场景。
* ResponseTime 响应时间
  + 响应时间是指从客户端发一个请求开始计时，到客户端接收到从服务器端返回的响应结果结束所经历的时间，响应时间组成部分：
    - 请求发送时间
    - 网络传输时间
    - 服务器处理时间
* CPU资源
  + CPU 资源是指性能测试场景运行的这个时间段内，应用服务系统的 CPU 资源占用率。CPU 资源是判断系统处理能力以及应用运行是否稳定的重要参数。应用服务系统可以包括 应用服务器、Web 服务器、数据库服务器等。
* Load负载
  + 系统平均负载，被定义为在特定时间间隔内运行队列中的平均进程数。
  + 如果一个进程满 足以下条件则其就会位于运行队列中：
    - 它没有在等待 I/O 操作的结果；
    - 它没有主动进入等待状态（也就是没有调用“wait”）
    - 没有被停止（例如：等待终止）
* Std. Deviation标准差
  + 该标准差根据数理统计的概念得来，标准差越小，说明波动越小，系统越稳定，反之， 标准差越大，说明波动越大，系统越不稳定

#### 客户端性能评级

YSlow 是Yahoo发布的一款基于FireFox的插件。

YSlow可以对网站的页面进行分析，并告诉你为了提高网站性能，如何基于某些规则而进行优化。YSlow可以分析任何网站，并为每一个规则产生一个整体报告，如果页面可以进行优化，则YSlow会列出具体的修改意见。

YSlow设定了页面23规则，每一条规则从F-A进行等级评分，A最佳，F最差，一般定义C级和C级以上通过测试，C级以下需要根据规则进行调整。

YSlow二十三条规则：

* 减少HTTP请求次数
  + 合并图片、CSS、JS，改进首次访问用户等待时间。
* 使用CDN
  + 就近缓存==>智能路由==>负载均衡==>WSA全站动态加速
* 避免空的src和href
  + link标签的href属性为空、script标签的src属性为空的时候，浏览器渲染的时候会把当前页面的URL作为它们的属性值，从而把页面的内容加载进来作为它们的值。
* 为文件头指定Expires
  + 使内容具有缓存性。避免了接下来的页面访问中不必要的HTTP请求。
* 使用gzip压缩内容
  + 压缩任何一个文本类型的响应，包括XML和JSON，都是值得的
* 把CSS放到顶部
* 把JS放到底部
  + 防止js加载对之后资源造成阻塞。
* 避免使用CSS表达式
* 将CSS和JS放到外部文件中
  + 目的是缓存，但有时候为了减少请求，也会直接写到页面里，需根据PV和IP的比例权衡。
* 权衡DNS查找次数
  + 减少主机名可以节省响应时间。但同时，需要注意，减少主机会减少页面中并行下载的数量。
* 精简CSS和JS
* 避免跳转
  + 同域：注意避免反斜杠 “/” 的跳转；
  + 跨域：使用Alias或者mod\_rewirte建立CNAME（保存域名与域名之间关系的DNS记录）
* 删除重复的JS和CSS
  + 重复调用脚本，除了增加额外的HTTP请求外，多次运算也会浪费时间。在IE和Firefox中不管脚本是否可缓存，它们都存在重复运算JavaScript的问题。
* 配置ETags
  + 它用来判断浏览器缓存里的元素是否和原来服务器上的一致。比last-modified date更具有弹性，例如某个文件在1秒内修改了10次，Etag可以综合Inode(文件的索引节点(inode)数)，MTime(修改时间)和 Size来精准的进行判断，避开UNIX记录MTime只能精确到秒的问题。 服务器集群使用，可取后两个参数。使用ETags减少Web应用带宽和负载。
* 可缓存的AJAX
  + “异步”并不意味着“即时”：Ajax并不能保证用户不会在等待异步的JavaScript和XML响应上花费时间。
* 使用GET来完成AJAX请求
  + 当使用XMLHttpRequest时，浏览器中的POST方法是一个“两步走”的过程：首先发送文件头，然后才发送数据。因此使用GET获取数据时更加有意义。
* 减少DOM元素数量
  + 是否存在一个是更贴切的标签可以使用？人生不仅仅是DIV+CSS
* 避免404
  + 有些站点把404错误响应页面改为“你是不是要找\*\*\*”，这虽然改进了用户体验但是同样也会浪费服务器资源（如数据库等）。最糟糕的情况是指向外部 JavaScript的链接出现问题并返回404代码。首先，这种加载会破坏并行加载；其次浏览器会把试图在返回的404响应内容中找到可能有用的部分当 作JavaScript代码来执行。
* 减少Cookie的大小
* 使用无cookie的域
  + 比如图片 CSS 等，静态文件都在主域名以外，客户端请求静态文件的时候，减少了 Cookie 的反复传输对主域名的影响。
* 不要使用滤镜
  + png24的在IE6半透明那种东西，不要乱用，需要切成PNG8+jpg
* 不要在HTML中缩放图片
* 缩小favicon.ico并缓存

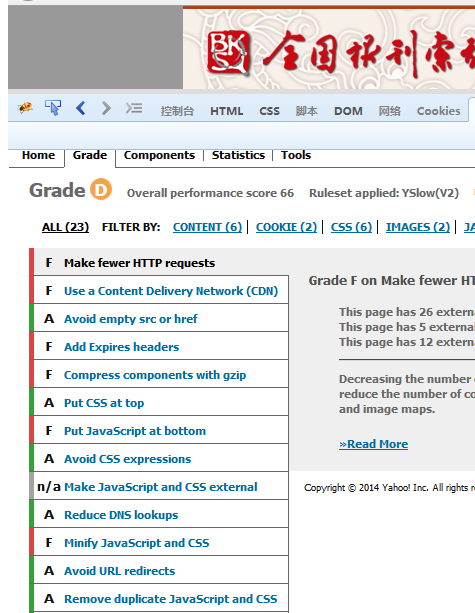


图1 当前网站使用yslow分析的结果。

### 可用性

|  |  |  |
| --- | --- | --- |
| 可用性 | 可用性指标 | 全年故障时间（365天） |
| 一般 | 99% | 87.6小时 |
| 较高 | 99.9% | 8.76小时 |
| 高 | 99.99% | 52.5分钟 |

表1 可用性指标评判基本标准

### 伸缩性

衡量架构伸缩性的主要标准是是否可以用多台服务器构建集群，是否容易向集群中添加新的服务器。加入新的服务器后是否可以提供和原来的服务器无差别的服务。集群中可容纳的总的服务器数量是否有限制。

### 扩展性

对于我们服务平台，扩展性主要是针对数据的扩展。评判标准是是否能够方便扩展并且支持新的文献类型，新的文章类型数据，新加入的数据能够达到和原有数据无差别的检索服务性能。

### 安全性

互联网是开放的，任何人在任何地方都可以访问。网站的安全架构是保护网站不受恶意访问和攻击，保护网站重要数据不被窃取。

衡量网站安全架构的标准是针对现存的和潜在的各种攻击和窃密手段，是否有可靠的应对策略。

资源安全，病毒扫描，对于系统扩展的资源需要进行病毒扫描，保障安全。

## 架构模式

架构设计发展到今天，已经经过了多次进化，从初始阶段的单服务器承载应用、数据和资源，到后来的应用和数据分离，到缓存的引入改善网站的性能，到集群的使用改善网站的并发处理能力，再到进一步的数据库的读写分离，到使用CDN加速网站响应，到分布式文件系统和分布式数据库的应用，到NoSQL使用和搜索引擎，到业务拆分，到分布式服务。

我们将架构典型模式做一个罗列和简介，我们的架构设计也是这些模式的组合应用，寻找最适合服务平台的一种组合方式。

### 分层（逻辑上）

分层是企业应用系统最常见的架构模式，将系统在横向维度上切分成几个部分，每一个部分负责一部分相对比较单一的职责，然后通过上层对下层的依赖和调用组成一个完整的系统。

典型的三层分层结构如下图所示：

|  |  |
| --- | --- |
| 应用层 | 负责具体业务和视图展示，如网站首页及检索输入和结果呈现 |
| 服务层 | 为应用层提供服务，如用户管理和购物车服务等等 |
| 数据层 | 提供数据存储访问服务，如数据库、缓存、文件、搜索引擎等等。 |

更多的分层往往就是这主流三层的每一层的进一步细分。例如，应用层可以细分为视图层（美工和前端负责），业务逻辑层（工程师）；服务层可以细分为数据接口层（适配各种输入和输出的数据格式）和逻辑处理层等等。

### 分割

分层是软件横向方面的切分，分割则是纵向方面对软件进行切分。

网站越大，功能越复杂，服务和数据处理种类越来越多，可以将这些不同的功能和服务分割开来，包装成高内聚、低耦合的模块单元；一方面有助于软件的开发和维护；另一方面，便于不同模块的分布式部署，提高网站的并发处理能力和功能扩展能力。一般来讲，这种分割在不同层次可以按照不同的规则进行分割，例如应用层可以根据业务分割，数据层可以根据数据相关业务或数据分类进行分割。

### 分布式

分层和分割让系统更加适应于分布式的部署。分布式意味着可以使用更多的计算机完成同样的功能，计算机越多，CPU、内存、存储资源等等也就越多，能够处理的并发访问和数据量就越大，进而能够为更多的用户提供服务。

分布式方案：

* 分布式应用和服务 —— 将业务和服务模块进行分布式部署
* 分布式静态资源 —— 网站的静态资源进行独立分布式部署，采用独立的域名，也就是常说的“动静分离“。
* 分布式数据和存储 —— 可以使用关系数据的分布式部署，还可以使用NoSQL产品。
* 分布式计算 —— 使用分布式计算框架。

### 集群

服务器集群采用多个服务器组合起来提供同样的服务，具备更加优良的并发特性。集群出了带来更好的并发性能外，还可以应对服务器故障，提供失效转移，保障服务的可用性。

### 缓存

缓存是将需要的数据放到最近的位置，缓存形式如下：

* CDN
* 反向代理
* 本地缓存
* 分布式缓存

### 异步

异步是通过共享队列或者消息中间件实现的业务解耦合，即将一个业务操作分为多个阶段，每个阶段通过共享数据方式异步执行。异步可以：

* 提高系统可用性
* 加快网站响应速度
* 消除并发访问高峰

### 冗余

冗余设计是为了保障系统的数据安全，冗余设计分为两部分：

* 热备 （采用集群方式）
* 冷备 （采用定期备份方式）

### 自动化

* 自动化代码管理
  + 自动化代码版本控制，主干分支合并；
* 自动化测试
  + 代码开发完成，提交后自动部署到测试环境进行自动化用例测试，向相关人员发送测试报告，向系统反馈测试结果。自动化进行回归性测试；
* 自动化安全测试
  + 安全检测工具通过对代码进行安全扫描，网站安全工具对网站进行安全扫描，评估安全性。
* 自动化部署
  + 发布部署过程自动化，尽量减少人工操作的情况；
  + 自动将通过测试的功能部署到生产环境中
* 自动化监控
  + 生产环境自动化监控，进行心跳检测，监控各项性能指标和应用程序的关键数据指标。
* 自动化报警
  + 发现异常，进行自动化报警，警告故障发生的位置和可能的原因。
* 自动化失效转移
  + 系统发生故障，会自动化失效转移，将失效的服务器从集群中隔离出去，保障数据的一致性。
* 自动化降级
  + 遇到访问高峰，超出网路最大处理能力，允许自动化降级，拒绝部分请求，将系统负载降低到一个安全水平。
* 自动化分配资源
  + 允许为重要服务，扩大部署规模，进行自动化资源分配。

### 安全

安全扫描工具使用，通过多重安全扫描工具对常见的网络安全攻击进行扫描，排除软件漏洞。通过杀毒软件的使用，对资源进行扫描，确保网站提供的资源都是无毒的。通过加密技术的使用确保系统的账号安全和数据安全，防止数据丢失。做好系统安全措施，防止黑客攻击造成服务瘫痪。

# 服务平台架构

## 服务平台整体架构



## 架构分层图



## 整体架构说明

* 数据层
  + Oracle数据库集群
    - Oracle集群存储了整个服务平台拥有的文献数据和业务数据，Oracle集群规模暂定为两台。
    - 文献数据的来源是基于信息中心的生产数据库导入进来的数据。
  + Solr搜索引擎
    - Solr搜索引擎是对文献数据进行反向索引，提供检索服务，采用SolrCloud集群进行部署，目前暂定采用4台主机组成集群。
  + 资源存储
    - 资源存储是文献资源的存储，同时还有部分上传资源进行存储（会议这块的内容）。
  + 缓存
    - 数据层还有一块为了访问提速而在服务器内存中设置的缓存区域，通过将常用数据进行缓存（再次访问直接从内存读取而不需要从数据库操作），提供系统的响应速度。
    - 缓存采用Memcached进行，主要由于Memcached的特性非常适合进行进一步的扩展，对分布式缓存提供非常好的支持。（暂时用虚线表示本项目内不采用分布式缓存，以后如果有需要提供可以平滑扩展）。
* 服务层
  + 统一数据访问层

统一数据访问层是将所有的对于数据层的数据访问合并为一个功能层次，进行统一的管理和维护。

* + - 关系数据库访问ORM：采用Hibernate作为ORM技术，提供关系数据库访问的功能，主要是对于一些业务数据的处理。
    - Solr Client : SolrCloud检索引擎的访问接口，提供了构建Solr查询语句调用Solr服务器查询功能获取查询结果的公用接口。
    - Storage Access： 存储访问接口，我们提供了支持多路径优化寻址的资源访问，可以通过此接口透明访问需要的文献资源。当前对上传资源也一样支持。为了安全起见，对上传资源和文献资源进行逻辑或者物理上的隔离。
    - Memcached Client：缓存访问客户端，是对缓存区进行访问的接口，可以自由的进行缓存的读取和写入，控制缓存失效机制等等。
  + 通用服务层（或成为基础服务层）

我们定义的通用通用服务层是非业务相关的服务，主要为业务提供一些基础性的服务。主要包含了：

* + - 数据框架管理，对通用数据框架进行管理，提供元数据管理支撑。
    - 规则引擎，提供对数据约束规则的引擎支撑，可以指定数据约束，数据转换中进行约束控制。数据约束就是一种数据严格。
    - 数据导入，提供数据导入功能，可以根据数据框架的定义和规则引擎的支持，进行严格的数据导入。
    - 在线支付，提供通用的银联支付的支撑。
    - 定时任务，服务平台将会由很多的定时或触发任务需要调度
    - 邮件服务，提供邮件发送支持，可以向用户发送邮件
    - 数据检索，通用的数据检索服务，减少业务层数据检索的功能耦合。
* 应用层
  + 业务逻辑层

业务逻辑层是和业务密切相关的处理层，所有视图需要的数据和业务处理都是业务逻辑层进行提供的。

* + - 前台业务

一般检索、高级检索、专业检索、原文索取、文献导航、会议中心、新闻等等

* + - 后台业务

数据管理、用户管理、产品管理、订单管理、会议管理、统计报表等等

* + 视图层
    - Web视图

目前只需要支持Web视图，采用目前流行的Bootstrap+Jquery+Html方式进行呈现，大量采用Ajax技术提升用户体验，部分采用HTML5技术，同时考虑兼容IE8以上的浏览器。

## 未来发展需求支撑说明



严格的分层机制，松散耦合的模块化业务模块、业务管理模块和表现层页面的模块化设计，可以方便的集成新的模块进入整个架构，未来需要扩展支持的积分机制、众包服务、讨论社区都可以通过模块化的开发方式集成到现有的系统中。

# 数据框架设计

## 业务数据架构



图1.1.- 数据库业务架构示意图

## 数据流程

### 流程A



### 流程B



## 服务平台数据库集群

### 数据库集群架构



图1.2.- Oracel RAC架构

### 硬件环境

|  |  |  |
| --- | --- | --- |
| 名称 | 参数 | 数量 |
| 节点数据库服务器 | CPU：2颗 8核 2.9GHZ  Memory：32G DDR3-1600  NIC：2块千兆网卡 | 2 |
| 交换机 | 8口千兆交换机1台（私有）  24口千兆交换机1台（公共） | 2 |
| 共享数据存储 | 支持共享存储的存储设备，可以使用ISCSI、NAS、FC SAN等等 | 1 |

## 搜索引擎架构

### 规模定义

### 服务平台Solr架构



Solr架构模块简介

* 数据转换传输服务：是之前数据业务框架中的数据中间件；支持从生产数据库提取数据进行转换和传输，同时传输到关系数据库和Solr中。
* Solr索引管理：集成了Solr配置管理、词典集中管理、数据缓冲的功能，集成了Solr DIH功能(全量索引更新、增量索引更新、重载配置等等)
* 工作队列：一个高性能磁盘队列，对发送过来的数据进行缓冲处理，防止直接提交Solr失败。
* QueueReader是一个定时任务，定时从工作队列中将数据读取出来，发送给Solr，加入一些重试处理和失败处理等等操作。

好的SolrCloud 实践

* Schema 或config 通过Zookeeper进行集中管理，能够自动进行配置修改的下发；
* 词典文件配置到Zookeeper中，通过插件支持对字典的监控和重载，实现自动化词典更新，不需要重载Core即可即时生效，带来业务上的方便。
* 数据导入，采用javabin格式提供更加高效的数据导入，放弃json这种仅可读性具备优势的数据格式。采用单线程读、多线程发的方式发挥集群优势。

### 中文分词

主流中文分词工具如下：

* IK Analyzer 2012 FF
* mmseg4j-Solr-2.2

根据网上的评测，两种分词器的实际分词效率和效果基本接近，主要还是需要看词库的配置。需要进一步测试分词的效率和效果，以及对Solr4.10.2的支持情况，来最终选择使用的版本。

至于其他的一些分词方法，例如庖丁解牛因为早就停止更新所以基本没有使用的必要，不过部分词库可以借鉴使用。

### 搜索引擎测试标准

* 查全率

查全率是指搜索引擎搜索出的与该查询关键字相关的信息的数量与搜索引擎索引数据库中实际与该查询关键字相关的信息总量的比值。比如该搜索引擎的索引数据库中有100条关于A的信息，而搜索引擎只搜索出其中的90条信息，那么查全率就是90%。

查全率不能以某一次查询的结果作为评价标准，而是以相当多数据的查询结果的平均值作为其查全率。这个比值越大则说明该搜索引擎查询的相关信息就越多，它的使用价值自然就越大。

* 查准率

查准率是指搜索引擎搜索出的结果中实际与该查询关键字相关的信息的数量与搜索引擎搜索出的信息总量的比值。比如该搜索引擎的索引数据库中有100条关于A的信息，而搜索引擎只搜索出90条信息，而这90条信息中只有80条与A有关，那么其查准率为80%。这个比值越大查准率就越高，则说明该搜索引擎搜索信息就越准确。

* 索引更新频率

索引库的更新频率会直接影响到搜索引擎的查全率和查准率。因为信息是变化更新的，因此搜索引擎的索引库也要适时进行更新。首先，新增的索引信息应能及时地出现在索引库中。其次，信息源文档已经不存在的索引要及时删除，否则，将会增加检索的响应时间，以致于影响整个搜索引擎的性能。一般情况下，索引的更新采用两种方法，一种是大批量的索引重建，也叫全量索引更新，另外一种是小批量的索引扩展，也叫增量式索引更新。大批量的索引更新一般是针对数据量在10G以下的信息资料库，可以定期对所有文档进行索引重建。但对于那些经常处于动态更新并且数据量庞大的信息资料库而言，增量式索引更新更适合且效率更高。

* 检索结果排序

搜索引擎对检索结果的排序的好坏也是评价一个搜索引擎优劣与否的一个重要因素。81.7%的用户不会浏览第三页之后的搜索结果，而52.2%的用户只会关注第一页的搜索结果。如何将最重要的和用户最关注的搜索结果放在最靠前的位置就成为了提高搜索引擎性能的一个重要指标。

* 响应时间

搜索引擎的响应时间是指从用户提交搜索请求到搜索引擎返回搜索结果所花的时间，它包括进入搜索引擎的等待时间和得到搜索结果的等待时间。一个搜索引擎无论它的查全率和查准率如何的高，如果其响应时间超出了用户能够忍受的程度，都会被用户弃之不用。因此，响应时间对用户是否选择使用某个搜索引擎显得至关重要。另外，响应时间不仅仅取决于搜索引擎本身的响应速度，在很大程度上还取决于用户的网络设备、网络状况等外部因素。

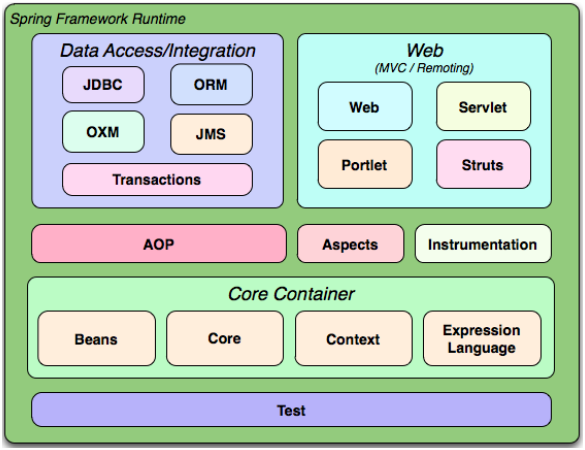
# 应用框架设计

## 应用架构

参见平台层次架构

## 技术详解

### 基础框架——Spring 3.2.4 （MVC）



核心容器(Core Container) 包括Core、Beans、Context、EL模块。

1：Core和Beans模块提供了Spring最基础的功能，提供IoC和依赖注入特性。这里的基础概念是BeanFactory，它提供对Factory模式的经典实现来消除对程序性单例模式的需要，并真正地允许你从程序逻辑中分离出依赖关系和配置。

2：Context模块基于Core和Beans来构建，它提供了用一种框架风格的方式来访问对象，有些像JNDI注册表。Context封装包继承了beans包的功能，还增加了国际化（I18N）,事件传播，资源装载，以及透明创建上下文，例如通过servlet容器，以及对大量JavaEE特性的支持，如EJB、JMX。核心接口是ApplicationContext。

3：Expression Language，表达式语言模块，提供了在运行期间查询和操作对象图的强大能力。支持访问和修改属性值，方法调用，支持访问及修改数组、容器和索引器，命名变量，支持算数和逻辑运算，支持从Spring 容器获取Bean，它也支持列表投影、选择和一般的列表聚合等。

数据访问/集成部分(Data Access/Integration)

1：JDBC模块，提供对JDBC的抽象，它可消除冗长的JDBC编码和解析数据库厂商特有的错误代码。

2：ORM模块，提供了常用的"对象/关系"映射APIs的集成层。 其中包括JPA、JDO、Hibernate 和 iBatis 。利用ORM封装包，可以混合使用所有Spring提供的特性进行"对象/关系"映射，如简单声明性 事务管理 。

3：OXM模块，提供一个支持Object和XML进行映射的抽象层，其中包括JAXB、Castor、XMLBeans、JiBX和XStream。

4：JMS模块，提供一套"消息生产者、消费者"模板用于更加简单的使用JMS，JMS用于用于在两个应用程序之间，或分布式系统中发送消息，进行异步通信。

5：Transaction模块，支持程序通过简单声明性 事务管理，只要是Spring管理对象都能得到Spring管理事务的好处，即使是POJO，也可以为他们提供事务。

Web

1：Web模块，提供了基础的web功能。例如多文件上传、集成IoC容器、远程过程访问、以及Web Service支持，并提供一个RestTemplate类来提供方便的Restful services访问

2：Web-Servlet模块，提供了Web应用的Model-View-Controller（MVC）实现。Spring MVC框架提供了基于注解的请求资源注入、更简单的数据绑定、数据验证等及一套非常易用的JSP标签，完全无缝与Spring其他技术协作。

3：Web-Struts模块， 提供了对Struts集成的支持，这个功能在Spring3.0里面已经不推荐了，建议你迁移应用到使用Struts2.0或Spring的MVC。

4：Web-Portlet模块，提供了在Portlet环境下的MVC实现

AOP

1：AOP模块，提供了符合AOP 联盟规范的面向方面的编程实现，让你可以定义如方法拦截器和切入点，从逻辑上讲，可以减弱代码的功能耦合，清晰的被分离开。而且，利用源码级的元数据功能，还可以将各种行为信息合并到你的代码中 。

2：Aspects模块，提供了对AspectJ的集成。

3：Instrumentation模块， 提供一些类级的工具支持和ClassLoader级的实现，可以在一些特定的应用服务器中使用

Test

1：Test模块，提供对使用JUnit和TestNG来测试Spring组件的支持，它提供一致的ApplicationContexts并缓存这些上下文，它还能提供一些mock对象，使得你可以独立的测试代码。

### ORM——Hibernate 4.3.7

Hibernate是轻量级ORM框架，它对JDBC进行了非常轻量级的对象封装，使得Java程序员可以随心所欲的使用对象编程思维来操纵数据库。

### 任务调度——Quartz2.2.1

Quartz是一个由James House创立的开源项目，是一个功能强大的作业调度工具，可以计划的执行任务，定时、循环或在某一个时间来执行我们需要做的事，这可以给我们工作上带来很大的帮助。例如，你的程序中需要每个月的一号导出报表、定时发送邮件或程序需要每隔一段执行某一任务……等等，都可以用Quartz来解决。

Quartz大致可分为三个主要的核心：

1、调度器Scheduler:是一个计划调度器容器，容器里面可以盛放众多的JobDetail和Trigger，当容器启动后，里面的每个JobDetail都会根据Trigger按部就班自动去执行.

2、任务Job：要执行的具体内容。JobDetail:具体的可执行的调度程序，包含了这个任务调度的方案和策略。

3、触发器Trigger：调度参数的配置，什么时候去执行调度。

Spring官方支持和Quartz的集成。

### 日志——Log4j2

Log4J已经发布了2.0版本，增加很多优良的特性，更加方便开发者使用，而且带来了很强的性能提升而且带来了新的插件系统，以及配置设置方面的很多改善。

# 日志框架设计

## 日志架构



## 日志存储

### 日志级别

|  |  |
| --- | --- |
| 级别编码 | 说明 |
| INFO | 一般信息日志 |
| DEBUG | 调试日志 |
| WARN | 警告日志 |
| ERROR | 错误日志 |
| FATAL | 致命错误日志 |

### 日志类型

|  |  |
| --- | --- |
| 类型 | 说明 |
| 业务日志 （BIZLOG） | 和业务紧密相关日志 |
| 系统日志 （SYSLOG） | 业务无关的日志 |

### 统一日志内容

|  |  |  |
| --- | --- | --- |
| 列名 | 编码 | 说明 |
| 日志时间 | CreateTime | 日志记录时间 |
| 日志类型 | LogTypee | BIZLOG、 SYSLOG |
| 日志级别 | LogLevel | 日志级别 |
| 关联用户信息 | UserInfo | UserInfo包含了：  UserId：用户ID  UserName：用户名  UserType：用户类型  UserIP：IP信息 |
| 关联组织信息 | OrgInfo | OrgInfo包含了：  OrgId：组织ID  OrgName：组织名称  OrgManageId：组织管理员ID |
| 日志页面 | PageInfo | PageInfo包含了：  Controller：页面控制器  Action：页面动作  Params：参数表 |
| 日志分类 | LogClass | 日志分类 |
| 日志数据 | Data | 关联数据信息，根据LogClass进行Json存储 |

# 安全设计

## 安全要素

* 信息安全：泛指服务平台的文献数据、业务数据、文献资源等等数据和信息的安全；信息安全面临的风险是信息的保密性、信息的完整性、信息的可用性等方面的威胁；
* 应用安全：应用安全泛指服务平台应用架构的安全性，我们的应用架构是否存在安全隐患，是否容易遭受到外来攻击，是否能够保障服务的可用性（防范DDOS）；
* 系统安全：承载应用平台的操作系统是否安全，是否存在已知的漏洞未修复容易造成系统脆弱、容易被入侵。
* 物理安全：特指我们承载系统的硬件设备是否物理安全，管理上是否能够保证设备的正常运行。

## 信息安全

关于信息安全，我们首先要明确信息边界。我们将会对每一类（细分类）角色进行信息权限判断。例如，后台管理员我们可以进一步的细分为系统管理员、业务管理员。他们的权限是有很大区别的。系统管理员可以管理架构方面的、数据框架方面的功能；业务管理员专门针对前台生成的业务进行处理。这是两类需要明确区分的类型。

敏感信息需要进行严格安全，我们建议采用高级散列算法，而且是加上Salt的散列算法，可以更加彻底的保障诸如账户密码的信息安全。严格控制密码重置的功能。

信息安全还涉及到的就是下载安全，首先需要确保具备权限的用户可以进行资源下载；但同时需要防范恶意下载；所谓恶意下载是指通过网络爬虫之类的工具进行信息的自动抓取，无论是下载还是页面访问，都需要进行控制。初步的控制规则如下：

* 单个IP允许开启的Session数量，个人用户进行限定，组织用户有组织上限
* 单一Session的页面浏览频率和资源下载频率必须控制在一定范围内
* 对单日浏览量和下载量进行监控，超过规定则进行限定。

信息安全还需要保障资源是安全可靠的，所有的资源无论来源只要进入服务平台就需要保障其安全无毒的。

## 应用安全

### 安全威胁

应用安全是指服务平台web应用可能遭受的安全威胁，主要有以下几种：

* XSS攻击，称之为跨站脚本攻击，通过篡改网页、诸如恶意脚本，控制用户浏览器进行恶意操作的一种攻击方式。XSS攻击是古老的攻击方式，但是确实最常见的，需要进行XSS消毒操作。
* 注入攻击，注入形式分为SQL 注入和OS注入。SQL注入是用户输入恶意脚本，一般可以通过参数绑定方式防范此类攻击；
* CSRF攻击（跨站点请求伪造），攻击者通过跨站请求，以合法的用户身份进行非法操作。跨站攻击往往利用客户端Cookie或服务端Session的策略，盗取用户身份。防范主要手段是：表单Token，通过增加随机数作为表单参数，让伪造者无从伪造；验证码，更加有效方便防止用户伪造攻击；Referer检查，HTTP请求中的Referer域中记录请求来源，可以通过检查请求来源，验证是否合法。（通过此种方法可以防止图片盗链、资源盗链等等）。
* ERROR CODE、HTTP注释、上传控制、路径遍历等等都是可能存在的不合理应用发布方式，需要做出相应的处理保障应用的安全。

### 应用安全防火墙 ModSecurity 2.9

ModSecurity是开源的Web应用防火墙，但是功能非常强大，可以有效保障Web应用的安全，可以和Apache webserver的一个模块进行使用。

## 系统安全

虽然说，我们服务平台采用的Linux系统相对于Windows系统安全性有优势，但是，系统安全还是需要进一步的加强的。系统安全两个方面：

1. 权限控制，严格控制服务器设备的账户权限和控制；
2. 防火墙开启，注意开启防DDOS的选项。
3. 杀毒软件启用

## 物理安全

这个只有通过进一步的加强管理制度来保障，不是本项目可控范围。

# 性能测试设计

## 测试工具 HP LoadRunner v11.0

LoadRunner，是一种预测系统行为和性能的负载测试工具。通过以模拟上千万用户实施并发负载及实时性能监测的方式来确认和查找问题，LoadRunner能够对整个企业架构进行测试。通过使用 LoadRunner，企业能最大限度地缩短测试时间，优化性能和加速应用系统的发布周期。 LoadRunner是一种适用于各种体系架构的自动负载测试工具，它能预测系统行为并评估系统性能，很实用的一种工具。

## 性能测试方法

### 性能测试

以系统设计规划的性能指标为预期目标，对系统不断施加压力，验证系统在资源可接受范围内，是否能到达性能预期。

### 负载测试

对系统不断地增加并发请求以增加系统压力，直到系统的某项或多项性能指标达到安全临界值，如某种资源已经呈现饱和。

### 压力测试

超出安全负载情况下，对系统继续加大压力，测试系统崩溃到不能处理任何请求，获得系统的最大压力承受能力。

### 稳定性测试

长时间保持一定的业务压力，测试系统提供服务的稳定性。

# 自动化功能测试设计

## 测试工具 (HP QTP v11.0)

QTP是quicktest Professional的简称，是一种自动测试工具。使用QTP的目的是想用它来执行重复的手动测试，主要是用于回归测试和测试同一软件的新版本。因此你在测试前要考虑好如何对应用程序进行测试，例如要测试哪些功能、操作步骤、输入数据和期望的输出数据等

## 测试流程

1）准备TestCase

- 在进行自动化之前，将测试内容进行文档化，不建议直接录制脚本

- 在录制脚本之前设计好脚本，便于录制过程的流畅

- 由于测试用例设计和脚本开发可能不是同一个人完成，便于团队合作

- 便于后期的维护

- 文档化的方式：TD或者文档

2）配置QTP

QTP支持不同的开发环境，在正式录制之前，需要根据被测程序的开发环境，选择合适的Add-In，并进行加载。

3）录制脚本

启动QTP的录制功能，按照Test Case的操作步骤描述执行，QTP自动记录每一步操作，并自动生成VBScript脚本。

4）修改增强脚本

刚刚录制好的脚本可能包含错误，或者没有达到预期的目的，这就需要在录制脚本的基础上，进行修改增强

- 删除录制过程中多余的以及错误的操作，以最少的脚本完成任务

- 如果前面操作的输出是后面操作的输入，则需要使用变量或者输出值来进行替换

- 不是所有的操作都可以通过录制产生的，有些需要通过手工编码实现这些功能

- 录制产生的脚本是线性的，可以加入条件、循环控制语句，实现更复杂的流程

- 对脚本进行结构化

- 加入注释，便于阅读和维护

5）调试脚本

- 回放通过的脚本，不一定是正确的，也可能会包含错误

- 在测试脚本正式使用之前，要保证其本身的正确性

- 避免测试脚本故障和被测程序故障搅在一起，不容易定位

6）回放脚本

- 对于回放的错误，不要急于马上提交Bug，首先要判断是脚本本身的错误还是程序的错误，确认后再提交。

7）脚本维护

- 随着工作的不断推进，脚本量会越来越多

- 被测试程序的不断更新，也需要更新相应的测试脚本

- 采用版本管理工具保存脚本，如CVS、VSS，可以随时获取历史版本

- 采用统一的脚本架构

- 采用统一的命名规范

- 添加充分的注释，避免时间久了，自己都不能马上读懂脚本。

# 自动化安全测试

## 测试工具

Acunetix Web Vulnerability Scanner 9.5 专业版

IBM Security AppScan V9.0 标准版

## 测试流程

* 安全扫描需要进行一定的脚本录制，让软件能够遍历到系统的每一个页面
* 通过两种工具的扫描对比，判断系统的安全状况

# 镜像站服务

# 整体设计

## 整体架构

安全是需要一个完整的体系来支撑的，需要技术和管理相互结合，以安全技术为基石，以严格的管理作为安全保障，共同打造一个安全环境。

为了支持整个镜像站的安全体系，我们设计了以下的安全架构来提供支撑：



## 加密基础库

加密基础库是我们整个安全体系的核心基础，所有的安全加密的核心算法是基于这个基础组件的。加密基础库包含很多对称和非对称公开的加密算法的标准实现方式。

我们知道，针对标准的安全加密算法，我们的安全性完全建立在密码的安全性上，一个是密码需要保密，另外就是密码需要足够复杂，不能够被暴力破解等等。

为了，进一步的增强安全性，我们在安全加密体系中加入了私有加密算法，这个算法将不会对外公开任何细节，采用特有的加密机制和流程，进而保障镜像站的核心数据，除了安全密码的保护能力外，额外加上一道加密算法的防火墙。

私有加密算法，是结合了对称加密算法和非对称加密算法机制，采用多重加密机制，兼顾加密效率的基础上，私有安全定制的加密算法。

## 镜像站安全管理

### 镜像站分布点管理

镜像站分布点管理，主要管理了镜像站的站点分布组成，我们需要为各个镜像站建立独有的安全认证机制，因为，如果采用公共安全认证机制，可能会发生镜像站用户之间的数据资源的非法共享和使用。

我们的镜像站的初始化安装包，需要在镜像站分布点配置完成安全秘钥后，才能生成打包完成，将安全认证过的镜像站安装包发送给镜像站用户，用户按照提示安装到自有的服务器上。镜像站的安装包包含了镜像站所在用户的信息，而且不允许篡改，因此，镜像站的安装包泄露，将可以追查到泄露者的责任。

镜像站分布点的密钥采用多组密钥方式，从机制方面，加大安全破解的难度。

### 镜像站更新包管理

镜像站更新包管理，是对于数据版本发生变化或产品发生修改后，需要下发资源到镜像站用户，我们需要对下发资源进行抽取，然后进行加密的。采用的安全加密密钥就是镜像站分布点管理的密钥组中按照选择规则进行选取。

镜像站更新包包含了数据框架、版本变化的全文索引数据、变化的数据资源等等，以及需要删除的数据脚本，这些内容全部进行加密存储，整个文件包再通过镜像点密钥加密存储。

### 镜像站更新包安全下发工具

镜像站更新包安全下发工具，将提供加密链路，提供个镜像站进行远程下载，当然，也支持手动下载，手动更新工具包。

镜像站更新包下发工具，采用的HTTPS协议，SSL加密通讯机制，保障更新包不存在被非法篡改的可能。

更新包内通过目录组织不同的更新数据，程序版本更新放到“program”目录，数据框架放到“framework”目录，索引数据放到“data”目录，资源数据放到“resource”目录，产品数据变化放到“product”目录。

### 关于镜像站版本更新

我们的更新数据包，可以支持镜像站的版本更新，如果发布新版本程序，会将版本更新涉及的文件打包，放入的更新包，一起下发到镜像站。

## 安全镜像站服务

### 镜像站嵌入式WEB应用服务

WEB应用服务器是整个镜像站的核心，通过WEB应用服务器来提供镜像站的所有服务，采用公共的WEB应用服务器将会导致以下问题：

1. 服务器应用程序开放，可能被篡改；
2. 服务器配置开放，可以修改；
3. 服务器安全漏洞公开，可能被攻击；

因此，镜像站需要安全，首先需要改变的就是WEB应用服务器，我们决定采用嵌入式的WEB应用服务，采用C++为语言开发，将服务应用和服务器本身糅合在一起，整个服务无任何配置项公开，全部采用封闭的方式提供服务。

WEB服务器理论来讲，其实就是一个网络服务程序，采用的是HTTP协议，我们定制的WEB服务器默认采用HTTPS协议，仅支持GET、POST方法，支持的路由仅支持需要的开放的路由，仅公开需要公开的信息。

WEB应用的静态资源和HTML页面全部采用加密资源形式嵌入到服务器中，不允许任何非授权的访问和篡改，从而杜绝服务端的任何破坏性活动。

应用服务安全我们将会重点保障：

#### 验证用户及文件授权

Web登录屏幕就像一个关卡，强迫用户提供一个有用的用户名和密码来证明他们的身份。验证用户，确认用户的身份非常重要。使用表单验证(DynamicForms)只适用于由web管理的资源。如果使用表单验证未授权的用户访问web管理的资源，则会转向到登录页面，可以通过设置非web资源的映射，对非web资源进行保护。一般来说，对于SQL数据库、其它数据库文件，使用自定义验证时，选用表单验证最为安全。

#### 管理用户会话

在进行数据交互时，往往通过会话来实现。会话跟踪客户端和服务器之间的完整交互，表现出客户端和服务器之间的无缝连接。

如果使用会话cookie，可以通过cookie的安全属性(domain、path、expires、secure、value)来限制攻击范围，加强会话安全。

#### 编写安全代码

恶意用户通过web应用程序本身的代码的缺陷操纵程序输入，使服务器泄露数据或允许访问未授权的文件。这些用户还可以控制服务器执行预先设定的程序代码。我们可以通过一下措施提高代码的安全性：

(1)防御恶意代码的输入：用户输入的数据一般都存储在变量里，为了安全，首先要把输入的数据存于变量通过过滤函数进行检测，以确定可以安全使用。确保每次输入都要存入没有被更改的变量，永远不要用原始的用户输入进行程序操作。变量的使用要明确变量的类型，通过类型在一定的程度上可以限制用户的输入。变量的引用要显式引用，例如：如果要引用表单usename，在程序里又有查询字符串usename，那么对于request(“usename”)，程序会先引用查询字符串中的usename，因为，对于request对象，web依次查找queryString，form，cookies，clientCertificate和serverVariabes集合已发现匹配，所以，在引用代码中，使用显式非常重要。

(2)验证所有用户的输入：代码中有很多可能被用户输入影响的操作，如数据类型的不匹配可能导致代码无法编译运行；输入shell命令执行系统命令等等。因此，我们可以通过检查输入值的数据类型、字符串长度、格式：还可以通过模式匹配，如正则表达式匹配合格输入，阻塞有害字符或字符串等方法来验证用户的输入。

(3)强大的错误处理功能可以阻止攻击者非法的程序操作。例如执行一条数据库查询，查找特定用户的纪录，期望匹配的是一条数据，但是如果返回的查询记录不止一条，就有应该怀疑该操作的合法性。我们可以对用户给出异常提示并转向特定的错误页面后返回限定的信息。

#### 编写安全的数据访问程序

在放置数据库文件时，要考虑数据库存放的位置。一般来说，web服务器应该与数据库服务器放在不同的主机上。由于某些原因，不得不把两者放到同一主机时，我们应该审查网络拓扑和网络安全，设计合理的防火墙布局，对数据库文件的访问提供最少权限，限制非法用户的访问。

(1)连接到数据源：我们采用嵌入数据库，不存在安全风险。

(2) 防止SQL注入：SQL注入就是用户通过输入sql命令导致数据库运行破坏或者危险。可以通过转义字符或者使用存储过程来防止sql注入。建议使用存储过程和SQLPrameter集合，因为，存储过程比结构化Sql语句执行速度快，而且它本身就有安全上的优势，与SQLPrameter集合的结合使用，把输入的字符自动转为文本字符，还可以通过SQLPrameter集合中的类型要求和类型长度来防止用户的非法输入即使恶意用户知道了sql参数，却无法通过存储过程知道表名或通过存储过程来修改字段名。

(3)安全的sql代码：通过编写有安全意识的代码，意在减少或消除代码漏洞，从数据库获取需要的最少数据，防止泄露过多数据信息。尽量使用“select字段名 from”而不要使用“select \* from”语句，因为“\*”是通配符，存在潜在危险。检查查询返回的结果集的期望属性，通过输入值和返回值的匹配检查是否正确操作。

### 镜像站嵌入式全文索引

全文索引服务是整个镜像站文献检索的支撑，传统的全文检索服务器，例如SOLR等，都是开源开放的，所以，索引数据的安全存在很大的隐患。而且SOLR是基于JAVA的，很难做到安全控制。

和应用服务的处理一样，我们需要将全文检索封闭起来，整个检索服务不需要对外开放任何检索接口，只需要支持应用服务的检索调用，因此，我们将会采用基于C++语言开发的全文检索服务来提供内部检索服务，关闭全部的外部接口。

另外，由于将全文检索的代码结合到整个应用中，我们将会修改全文检索的文件存储体系，将所有索引文件进行加密存储，读取的时候进行解密。这样既可以保障全文检索服务的安全，也可以保障全文检索服务产生的数据的安全性。

### 镜像站嵌入式数据库

业务数据是存储在数据库中的，我们的镜像站需要关系数据库的支持，采用开放的数据库必然导致数据存在泄漏风险，因为，数据库配置文件显然是公开可见的。因此，我们的嵌入式数据库将会采用嵌入式数据库。

我们将采用的嵌入式数据库为SQLite，SQLite的安全加密方式有两种：

1. 数据加密后存储到数据库
2. 将整个数据库文件进行加密

我们将会采用加密整个数据库文件的方式，这样可以保障整个业务数据不存在数据泄露风险。

另外，对于业务敏感数据我们将会全部采用加密方式进行存储。杜绝任何可能的安全风险。

### 镜像站嵌入式资源库

文献资源，是最宝贵的数字资源，需要进行高级别的安全保障。镜像站对的资源库我们将会采用嵌入式资源库进行存储。我们将会采用自有格式进行存储，划分一个大的文件块，采用定制的逻辑组织文件，所有资源文件加密存储到文件块。这样做，有两个好处：

1. 文献资源不可见，整个文献资源不存在文件路径中，无法直接复制
2. 文献资源加密存储，文献名等等信息全部隐藏；

### 镜像站更新包接收工具

镜像站更新包接收工具，是需要解密导入的更新包，从更新包获取数据，应用修改镜像站的数据和资源。具体的更新顺序如下：

1. 更新镜像站程序版本，可能会涉及到程序版本更新；
2. 更新数据框架版本，整个系统的元数据定义体系；
3. 更新全文索引的模式，元数据变化必然导致发生变化；
4. 更新全文索引数据；
5. 更新产品数据定义；
6. 更新资源库；

为了能够支持镜像站应用程序版本更新，我们更新程序是一个独立于主体程序外的执行程序。

# 技术选型和说明

## 安全应用服务 （Lighttpd）

安全应用服务器我们通过大量选型对比，注重选择了一个轻量级的应用服务器Lighttpd。Lighttpd 是一个德国人领导的开源Web服务器软件，其根本的目的是提供一个专门针对高性能网站，安全、快速、兼容性好并且灵活的web server环境。具有非常低的内存开销、cpu占用率低、效能好以及丰富的模块等特点。Lighttpd是众多OpenSource轻量级的web server中较为优秀的一个。支持FastCGI，CGI，Auth，输出压缩(output compress)，URL重写，Alias等重要功能；而Apache之所以流行，很大程度也是因为功能丰富，在lighttpd上很多功能都有相应的实现了，这点对于apache的用户是非常重要的，因为迁移到lighttpd就必须面对这些问题。lighttpd采用了Multiplex技术，代码经过优化，体积非常小，资源占用很低，而且反应速度相当快。

我们是基于Lighttpd基础上进行了调整，我们对镜像站不需要的服务器功能全部屏蔽，仅保留必须的技术，最大化提高服务的性能。另外Lighttpd支持部分高级特性，异步IO和FastCGI等等技术的使用可以提升镜像站的服务能力，最终目标是降低镜像站的服务器配置要求。

## 安全检索服务（CLucene）

全文检索服务，由于我们服务器的全文检索服务是基于Solr的，采用的是Lucene的技术体系，因此，我们的镜像站最佳的选择也是采用这个框架体系的检索技术。目前Lucene的c++版本仅有CLucene是相对比较成熟的，我们将会基于CLucene进行安全检索服务的集成，通过对存储的修改调整，达到文件级别的安全，改造外部服务为内部，关闭数据泄露的通道。

## 安全数据库（Sqlite）

安全数据库我们初步选择的是Sqlite数据库，Sqlite是一个嵌入式的数据库产品，使用范围非常广泛，项目非常活跃。经过了若干年的发展，目前的技术比较成熟，而且性能也极大提升，能够满足我们镜像站对单个组织的服务要求。我们将会对Sqlite进行内部集成，通过代码级别的调整，纳入到整个镜像站的安全体系中，全面控制镜像站的数据泄露的风险。

我们的数据库仅仅存储了业务数据和日志数据，不需要存储资源数据，保持数据库的高效。

目前Sqlite3已经支持了安全服务，有成熟的安全体系来加密数据库文件，采用256bit的AES加密机制，保障数据不会恶意修改。另外，Sqlite3可以方便的进行数据分拆，业务数据是加密库，另外日志数据需要采用普通库。

## 安全的资源存储（S-VFS）

安全的资源存储主要技术是采用VFS技术，即虚拟文件系统技术，我们通过一层虚拟的文件系统，来讲加密的文件块解密提供给内部的应用系统进行使用，这样所有的上层应用不需要做特别多的调整，就可以使用一个安全的存储体系。

安全的虚拟文件系统也承担了我们的资源服务的任务，加密的资源下发到镜像站，将会加入到虚拟文件系统中，对内部应用提供资源下载、浏览等等功能。