**SparrowData平台方案**

华南理工大学 Micerlabs

# 1简介

社会生产与生活产生大量数据，需要由计算机信息系统进行组织与管理。这些数据，既包括结构化数据，比如对某一事物的属性描述，和非结构化数据，如电子文件（文本文件（如doc文件）、图片文件（如jpeg文件）、音频文件和视频文件等）。

SparrowData是一个数据管理基础平台，它通过HTTP REST API接口形式，为客户管理海量结构化与非结构化数据，提供高效存取、多维检索、智能推荐和安全访问等核心服务。

# 2理念与概念

数据的产生及管理与企业业务活动过程密切相关。企业通常形成树形层次的组织管理结构（如公司、部门和科室的三级层次机构），并在业务活动中，每级机构通常围绕某个事件或主题产生相关属性数据和文件数据，如某个会议作为主题，包括会议属性，会议纪要文档和会议图片等。同时，对这些数据的访问须满足企业业务安全规则，如科室人员未经授权无法查看所属部门等上层机构人员产生的文件等。如下示例1，

公司[c1]设置两个部门[d1]和[d2]，及其属下的科室或办公室[o1]、[o2]和[o3]。为了有效进行数据管理，该公司设定管理员用户[c1root]，和其他人员用户[d1user]和[o1user]。在业务安全规则方面，用户[c1root]作为公司管理员用户，对该公司辖下资源拥有完全操作权限，并可创建其他系统用户。用户[d1user]和[o1user]频繁参与企业日常数据管理活动。除此外，用户[d1user]是部门[d1]的代表用户，代表部门[d1]进行数据管理活动，并对部门[d1]辖下数据拥有完全操作权限；用户[o1user]则是科室[o1]的代表用户。同时，经系统授权，用户[d1user]和[o1user]可对非其代表机构的数据进行操作，如用户[o1user]对科室[o2]下的数据拥有读取权限等。另外，企业人员还希望有个人数据空间，存储初稿、个人文件等不公开的文件等。

## 2.1 基本理念

SparrowData遵循如下设计理念：

* **数据（Data）**，是一个抽象的概念，由SparrowData所管理的全部数据对象（data node）构成。
* **数据对象（Data Node）**，SparrowData中可以唯一寻址与访问的对象；可分为结构化/非文件对象，表现为一组描述性属性，和非结构化对象/文件对象，表现为文件自身及其描述性属性。
* **数据分类**：依据数据操作频率与业务安全规则等分类。

1. **业务类数据**，业务活动中频繁操作的数据；此类数据默认企业全体人员均可访问（同质权限，全体人员）；
2. **机构类数据**，组织机构管辖的数据，如发布通知等；业务人员需依据既定业务安全规则进行访问（异质权限，需结合组织结构进行安全访问管理）；
3. **个人数据**，业务人员暂不想公开的数据，仅拥有者可以访问（拥有者权限）。

* **数据组织**：

1. 采用目录结构（层次、树形结构）对数据对象进行有效组织；每个数据对象作为目录结构中的一个节点；
2. 目录结构支持与组织机构层次映射，如一个机构对应一个目录节点（机构目录）；
3. 目录结构有利于实现组织层次结构的安全访问控制；
4. 目录结构支持多租户划分，结合安全访问机制，确保租户数据隔离；
5. 目录支持事件、主题等集合管理概念，有效满足业务活动特点；
6. 为不同业务场景提供适宜的术语场景，比如部门、工作区等。

* **访问控制**

1. 面向资源许可的安全访问机制；
2. 支持细粒度访问控制策略，确保高质量的安全访问；
3. 支持粗粒度访问控制策略，确保高效率安全访问；
4. 支持租户、用户和目录的细粒度关闭与开放，确保灵活的安全访问；

## 2.2 核心概念

SparrowData对数据管理行为进行高度抽象，设计了如下核心概念与机制，包括数据组织方面的目录（Directory）、文档（Document）和文件（File）、和安全访问方面的用户（User）、群组（Group）、操作（Operation）和许可（Permission）和其他管理概念，包括工作区（Workplace）。等。

* **系统管理员（Sparrow Root）**：Sparrow初始化后，默认存在一个全系统域root用户，拥有整个系统资源的全部权限，并在业务过程中，创建其他租户和用户。
* **租户（Customer）**：租户根据服务协议使用Sparrow各项服务；Sparrow系统支持多租户。

### 2.2.1 数据组织

本节结合前述示例1对SparrowData 的目录、文档和文件概念进行说明。SparrowData为公司[c1]构建如下图1所示树状目录结构。

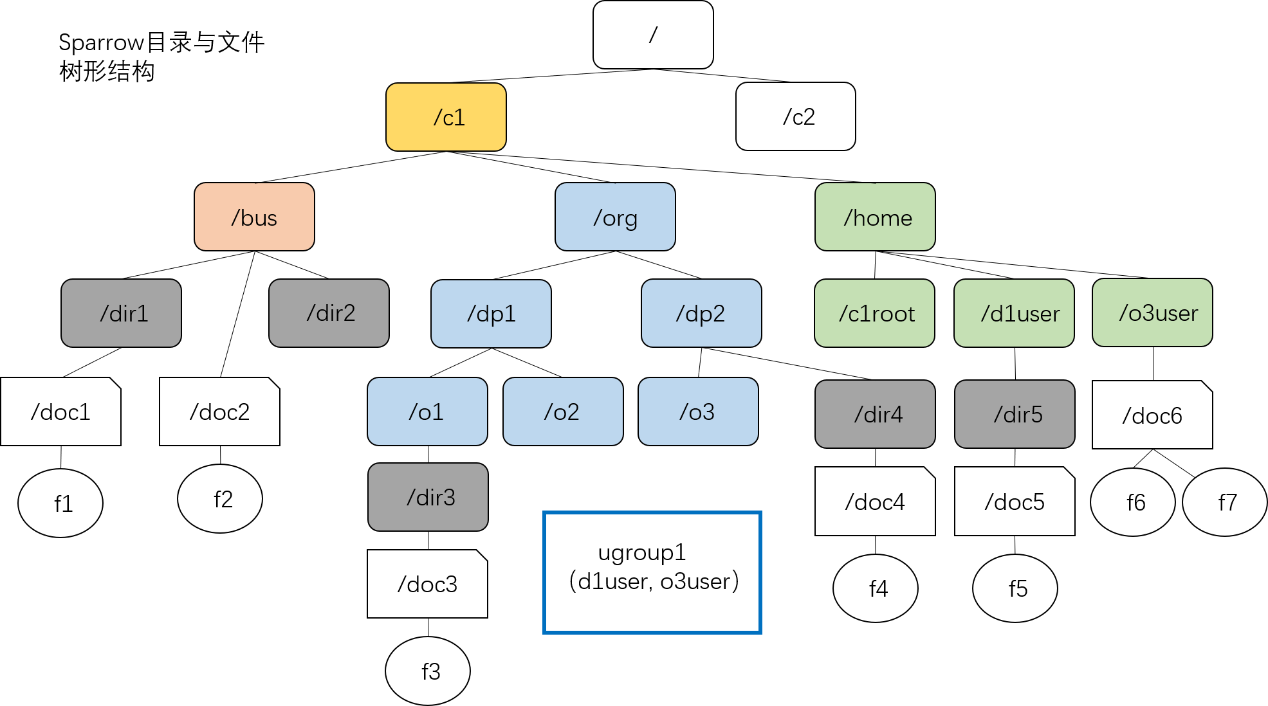


图1 目录与文件组织结构（示例1）

* **普通目录（General Dir，GDir）对象，用于构建数据对象树状组织结构。**

目录指一个装有数据对象系统的虚拟“容器”。它可保存一组数据对象和其它一些目录（又称子目录）。这些目录（及其子目录）就构成了层次（hierarchy），或树形结构。

在SparrowData当前版本中，仅支持一个目录树，即所有子目录从系统根目录“/”[sparrow root]派生。

示例图1中，圆角四边形为目录对象，如[dir1]，其路径为[/c1/bus/dir1]。

系统为租户-公司[c1]创建了租户根目录[/c1]（customer root dir），并在此租户根目录下创建了3个固定根目录：

* **业务数据根目录**（Business Root Dir, bus）：[/c1/bus]，由系统创建，固定不变。
* **机构数据根目录**（Organization Root Dir, org）：[/c1/org]，由系统创建，稳定不变。

租户管理员用户[c1root]可根据实际组织结构创建其他子机构的根目录，并形成相应的机构子目录树，如

1. [/c1/org/d1]和[/c1/org/d2]：为相应部门根目录；
2. [/c1/org/d1/o1]、[/c1/org/d1/o2]和[/c1/org/d2/o3]：为相应科室的根目录。

* **用户数据根目录**（Home Dir, home）：由系统创建，固定不变。

1. [/c1/home/c1root]：公司管理员用户[c1root]的默认home目录；
2. [/c1/home/d1user]：用户[d1user]的默认home目录；
3. [/c1/home/o1user]：用户[o1user]的默认home目录。
4. 回收站根目录（Recycle Bin Dir）：？？

* **文档（Document Dir）对象，特殊的数据集合，对应业务活动中的事件或主题等。**

文档是一种特殊的目录，又称为文档目录。文档仅允许包含多个文件数据对象（见下文），不允许包含子目录。

示例图1中，缺右上角四边形为文档对象，如[doc1]，其路径为[/c1/bus/dir1/doc1]。

* **文件（File），指具体文件数据对象，如一张图片、一份doc文件等。**

文件指具体文件数据对象，比如一张图片等，可从属于一个普通目录或者文档目录。

示例图1中，红色圆形为文件数据对象，如[f3]，其路径为[/c1/d2/doc3/f3]。

在数据组织上，目录、文档和文件都是目录树的节点。其中，结构化/非文件对象通常可组织对应为一个普通目录节点；而非结构化/文件对象通常可组织对应为一个文件节点；如果需要对一组文件对象进行集中描述与管理，通常可将该组文件的父节点组织对应为同一个文档目录节点。

### 2.2.2安全访问

对目录、文档和文件等数据对象资源，SparrowData采用面向资源许可访问控制策略（Resource-oriented and Permission based），相关概念说明如下：

* **用户，指系统使用者。**

用户类型：

1. Sparrow Root 系统管理员，具有整个Sparrow的完全权限；
2. System User 系统用户，由Sparrow Root创建与管理，并依据授权进行系统访问。
3. Customer Root 租户管理员，具有租户所辖范围的完全权限；
4. Customer User 租户用户，由Customer Root创建与管理，并依据授权进行系统访问。

示例图1中，为租户[c1]共创建了如下用户：

1. [c1root]：租户管理员；
2. [d1user]：租户用户[d1user]；
3. [o1user]：租户用户[o1user]。

* **群组，指一组用户的集合，便于授权管理。**

SparrowData可授予群组（多个用户）对一个资源相同的操作许可。

示例图1中，群组[ugroup1]包含用户[d1user]和[o1user]，对文档[doc4]有读取权限，群组中的用户继承群组的权限。

* **操作，对资源的操作。**

1. **目录（文档）/文件操作：**

采用16个比特的二进制数符（字符串）进行表达，如下：

|  |  |  |
| --- | --- | --- |
| 序号 | 操作 | 比特系列（8个） |
| 1 | 新建（子目录）  /node/new/dir | 0000 0000 0000 0001 |
| 2 | 新建（子文件）  /node/new/file | 0000 0000 0000 0010 |
| 3 | 读取（本节点 SPA META）  /node/read/smeta | 0000 0000 0000 0100 |
| 读取（本节点Extend META）  /node/read/emeta |
| 读取（本节点 META）  /node/read/meta |
| 4 | 更新（本节点SPA META）  /node/update/smeta | 0000 0000 0000 1000 |
| 更新（本节点Extend META）  /node/update/emeta |
| 更新（本节点META）  /node/update/meta |
| 5 | 删除（本节点）  （文件节点或子节点为空）  /node/delete | 0000 0000 0001 0000 |
| 6 | 检索（按条件查询SPA META）  /node/list/smeta | 0000 0000 0010 0000 |
| 检索（按条件查询Extend META）  /node/list/emeta |
| 检索（按条件查询META）  /node/list/meta |
| 7 | 检索（Index子节点，索引数据库ES，多维检索）  /node/list/index | 0000 0000 0100 0000 |
| 8 | 下载（单个文件）  /node/download/file | 0000 0000 1000 0000 |
| 下载（整个文档目录打包下载）  /node/download/dir |
| 9 | 预览（文件）  /node/preview/file | 0000 0001 0000 0000 |
| 10 | 执行（文件）  /node/exec/file | 0000 0010 0000 0000 |
| 11 | 移动（更换父目录）  /node/link | 0000 0100 0000 0000 |
| 12 |  | 0000 1000 000 0000 |
| 13 |  | 0001 0000 000 0000 |
| 14 |  | 0010 0000 000 0000 |
| 15 |  | 0100 0000 000 0000 |
| 16 |  | 1000 0000 000 0000 |

* 操作组合，如CRUD：0000 0000 0000 1111；所有操作都是单一进行的，即用户每次请求仅能指定一个操作。
* **许可，群组（用户）拥有对资源执行操作的准许。**

许可是群组（主体）、资源（数据对象）与操作的三元组；创建许可过程，可视为资源创建一个群组，并为群组绑定具体操作权限，又称为授权。

示例图1中，用户[o1user]对文档[doc4]拥有读取操作许可。

* **授权粒度**：SparrowData针对目录对象进行授权，不针对每个文件对象进行授权。当用户具有A目录的“预览（文件）”操作许可，那么该用户具有对A目录下所有文件的预览操作许可。

### 2.2.3其他概念

SparrowData提供两种资源管理视图：

1. 树形目录视图，用户在目录视图中进行数据对象管理。主要针对SparrowData管理员和租户管理员用户。
2. 工作区（Workplace，WP）：业务工作区（Business WP， BWP）、组织机构工作区（Organization WP）和个人工作区（User WP， UWP）。

# 3系统设计

Sparrow采用模块化和层次结构，基于面向资源的服务架构进行设计，依据业务与数据路由机制，实现多租户与分布式部署。

## 3.1总体结构

Sparrow主要包含系统管理、系统安全、数据存储和核心业务等模块，和HTTP REST API接口模块，如下图2所示。

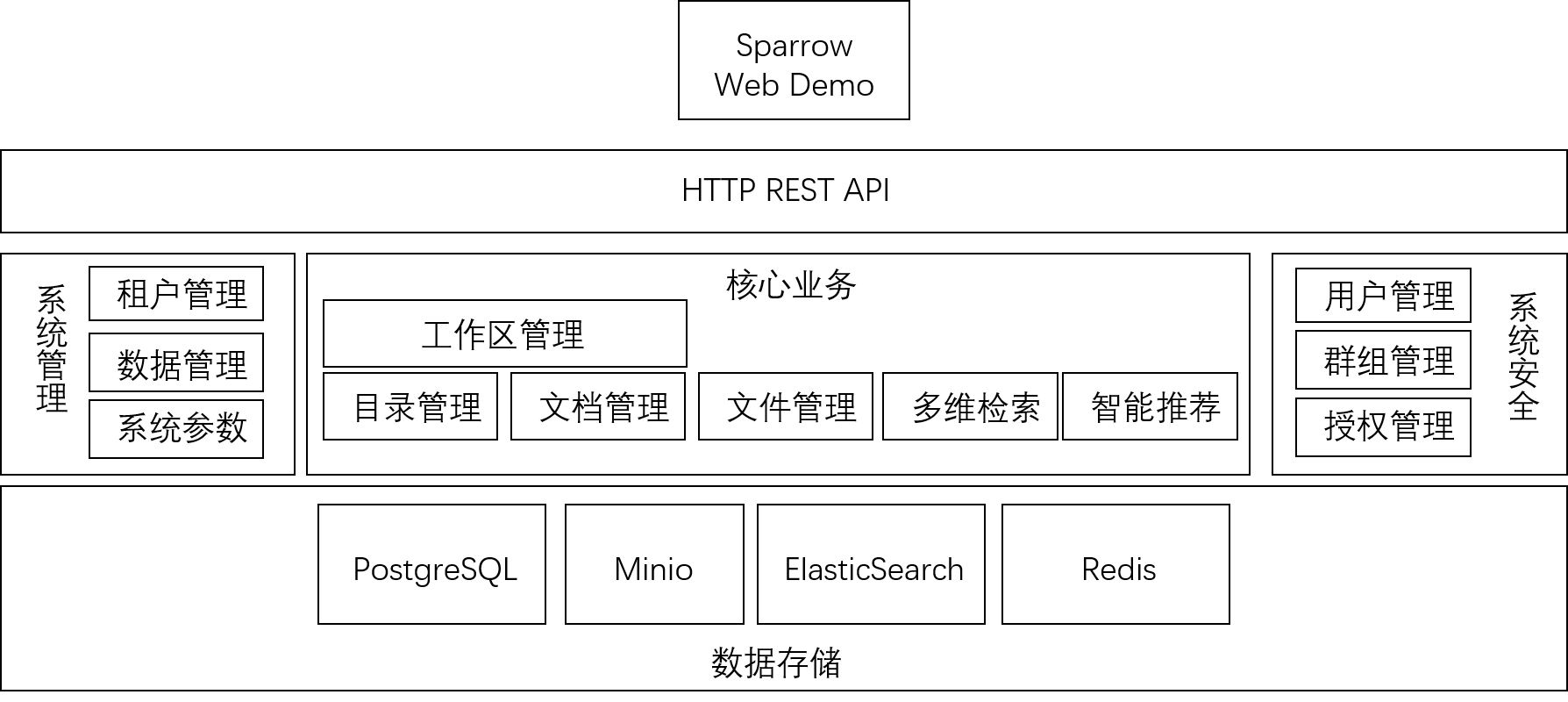


图2 系统结构

**系统管理**，包括1）系统参数配置管理；2）数据管理，支持用户定义目录、文档和文件的元数据模型、多维检索及智能推荐模型；3）租户管理，支持多租户。

**系统安全**，包含1）用户管理，2）许可管理。

**核心业务**，作为文件存储、检索与推荐的核心业务，包含1）目录管理，2）文档管理，3）文件管理，4）工作区管理，5）多维检索和6）智能推荐。

**数据存储**，业务数据的持久化，综合采用了关系型数据库PostgreSQL，面向对象的存储Minio，全文检索引擎ElasticSearch和缓存系统（待设计）。

1. PostgreSQL，存储数据对象的结构化属性部分、租户、用户、许可等数据，提供高效访问与一致性服务。
2. Minio，为海量非结构化/文件对象提供高效存储与安全访问服务。
3. ElasticSearch，为数据对象属性数据及文件对象的全文内容、类目、标签等提供检索服务。
4. 应该在PostgreSQL与应用程序之间增加一个缓存层，实现热点数据缓存，因为大量频繁操作在目录与许可的访问。（具体机制，待设计）

其中，系统管理、系统安全和核心业务模块，通过HTTP REST API接口提供服务。

### 3.1.2数据模型

SparrowData的数据模型包涵如下三个部分：

1. SPA META，模型在SparrowData中的固定属性，对不同业务系统具有通用性。
2. Extend META，模型的业务系统相关扩展属性。
3. IndexId，前述META均由PostgreSQL关系型数据库提供存储与访问服务；系统支持META在ElasticSearch索引数据库中存储，提供多维检索服务。

SPA META和Extend META完全描述一个数据对象的结构化属性。数据对象的非结构化部分，即文件本身，存储在系统中。

* **租户**：由PostgreSQL支持实现，其二维表结构如下。

表 1 租户数据模型

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 名称 | 列名 | 约束 | 说明 |
| 0 | Id | 行Id | 自增 | 便于运维（？） |
| 1 | CustId | 租户Id | 字符串{36} | UUID |
| 2 | Name | 名称 | 字符串{25} | 用户定义 |
| 3 | Type | 类型 | 字符串{8} | {vip，normal，test} |
| 4 | CreateAt | 创建时间 | 长整数 | 时间戳的长整数 |
| 5 | Access | 访问状态 | 整数 | {0（关闭）, 1（正常）} |
| 6 | RootDir | 租户根目录 | 字符串{8} | 表 4 DirId |
| 7 | ExtendType | 用户扩展类型 | 字符串{16} | 用户扩展数据类型 |
| 8 | ExtendData | 用户扩展字段 | JSON | 支持用户自定义扩展，PostgreSQL支持JSON数据类型的CRUD与基本检索 |
| 9 | IndexId | 外部数据库Id | 字符串{\*\*} | ExtendData中的部分字段可扩展到非本SQL数据库系统进行索引，如ElasticSearch |

* **（系统）用户**：由PostgreSQL支持实现，其二维表结构如下。

表 2 （系统）用户数据模型

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 名称 | 列名 | 约束 | 说明 |
| 0 | Id | 行Id | 自增 | 便于运维（？） |
| 1 | UserId | 用户Id | 字符串{36} | UUID |
| 2 | Name | 登录名 | 字符串{25} | 用户定义 |
| 3 | Type | 类型 | 字符串{8} | {sysroot，sysuser，custroot，custuser } |
| 4 | CreateAt | 创建时间 | 长整数 | 时间戳的长整数 |
| 5 | Access | 访问状态 | 整数 | {0（关闭）, 1（正常）} |
| 6 | HomeDir | 用户文件根目录 | 字符串{36} | 见表 4 DirId |
| 7 | CustId | 租户Id | 字符串{36}  见表1 CustId | 多租户支持  当type = sparoot和sysuser，CustId = system |
| 8 | ExtendType | 用户扩展类型 | 字符串{16} | 用户扩展数据类型 |
| 9 | ExtendData | 用户扩展字段 | JSON | 支持用户自定义扩展，PostgreSQL支持JSON数据类型的CRUD与基本检索 |
| 10 | IndexId | 外部数据库Id | 字符串{\*\*} | ExtendData中的部分字段可扩展到非本SQL数据库系统进行索引，如ElasticSearch |

* **许可**：由PostgreSQL支持实现，其二维表结构如下。

(许可是群组（主体）、资源（目录）与操作的三元组)

表 3 许可数据模型

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 名称 | 列名 | 约束 | 说明 |
| 0 | Id | 行Id | 自增 | 便于运维（？） |
| 1 | PermissionId | 主键 | 字符串{36} | UUID |
| 2 | Name | 名称 | 字符串{25} | 用户定义 |
| 3 | Type | 类型 | 字符串{8} | {simple（细粒度权限），supper（层次权限继承）} |
| 4 | Setting | 成员设置（主体） | 字符串{8} | {positive（默认包含全部用户），negative（默认不包含全部用户）} |
| 5 | Members | 成员列表 | 字符串{1000} | 99个用户Id列表；JSON格式  当Setting为positive时，Members表示exclude这些用户；  当Setting为negative时，Members表示include这些用户； |
| 6 | DirId | 资源（目录）Id | 字符串{36}  见表 4 DirId | 一个群组仅对应一个资源（数据对象） |
| 7 | PathRegex | 目录授权正则表达式 | 字符串{255} | 当type=simple时，DirPathRegex为DirId的DirPath；  当type=supper时，DirPathRegex为DirId的DirPath\*； |
| 8 | Operations | 操作许可 | 二进制字符串 | 见：**2.2.2节中的操作** |
| 9 | Access | 访问状态 | 整数 | {0（关闭）, 1（正常）} |
| 10 | CreatorId | 创建者 | 字符串{36} | 表 2 UserId |
| 11 | CreateAt | 创建时间 | 长整数 | 时间戳的长整数 |
| 12 | Invalid\_at | 有效期至 | 长整数 | 时间戳的长整数  长期有效:  2999年12月31日23时29分 |
| 13 | CustId | 租户Id | 字符串{36}  见表1 CustId | 多租户支持{System|CustId} |
| 14 | ExtendType | 用户扩展类型 | 字符串{16} | 用户扩展数据类型 |
| 15 | ExtendData | 用户扩展字段 | JSON | 支持用户自定义扩展，PostgreSQL支持JSON数据类型的CRUD与基本检索 |
| 16 | IndexId | 外部数据库Id | 字符串{\*\*} | 当type为docdir或file时，描述该文档或文件的全文检索数据模型在ElasticSearch中存储的id |

(PostgreSQL, 正则表达式检索: )

9.7. Pattern Matching

<https://www.postgresql.org/docs/9.3/functions-matching.html>

: UserId，DirId（DirPath），Operation？准许

: DirId（/c1/bus/dir1: /d3eh4kel/wm3o2yd/a3dni78d）匹配DirPathRegex：/c1/bus\*: /d3eh4kel/wm3o2yd\*

: Operation（???? ???1）匹配Permission（0000 0001等）:

: （UserSetting=positive and UserId not in UserList）or （UserSetting=negative and UserId in UserList）

* **目录（文档、文件）**：由PostgreSQL支持实现，其二维表结构如下。

表4 目录（文档、文件）数据模型（二维表结构）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 名称 | 列名 | 约束 | 说明 |
| 0 | Id | 行Id | 自增 | 便于运维（？） |
| 1 | DirId | 主键 | 字符串{36} | UUID |
| 2 | ReferDirId | 关联目录Id | 字符串{36} | 在访问控制上，访问本节点，等同于访问ReferDirId指向的节点 |
| 3 | Name | 名称 | 字符串{25} | 用户定义 |
| 4 | Type | 类型 | 字符串{8} | 目录类型{sysroot（系统根目录），custroot（租户根目录），busdir（业务类文件根目录），orgdir（机构类型根目录），homedir（个人文件根目录），nordir（普通目录），docdir（文档目录）、file（文件）} |
| 5 | MineType | 文件类型 | 字符串{8} | 文件的多媒体类型  {dir or mine, e.g. jpeg} |
| 6 | Thumbnail | 缩略图路径 | 字符串{256} | 缩略图路径？？ |
| 8 | CreatorId | 创建者 | 字符串{36} | 表 2 UserId |
| 9 | CreateAt | 创建时间 | 长整数 | 时间戳的长整数 |
| 10 | ModifyAt | 最新修改时间 | 长整数 | 时间戳的长整数 |
| 11 | Access | 访问状态 | 整数 | {0（关闭）, 1（正常）} |
| 12 | FileId | 外部  Minio  或者  阿里云OSS Id | 字符串{\*\*} | 当type为file时，其需要在Mino或者阿里云OSS中进行存储，OSS文档Id |
| 13 | CustId | 租户Id | 字符串{36}  见表1 CustId | 多租户支持{System|CustId} |
| 14 | ExtendType | 用户扩展类型 | 字符串{16} | 用户扩展数据类型 |
| 15 | ExtendData | 用户扩展字段 | JSON | 支持用户自定义扩展，PostgreSQL支持JSON数据类型的CRUD与基本检索 |
| 16 | IndexId | 外部  ElasticSearch Id | 字符串{\*\*} |  |

* MASTER\_SLAVES，树形目录关系

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 名称 | 列名 | 约束 | 说明 |
| 0 | Id | 行Id | 自增 | 便于运维（？） |
| 1 | MasterId | 父节点id | 字符串{36}  见表 4 DirId | UUID |
| 2 | SlaveId | 子节点id | 字符串{36} | UUID |

**工作区**：由PostgreSQL支持实现，其二维表结构如下。（暂不考虑）

## 3.2核心服务

SparrowData提供用户管理、高效存取、多维检索、智能推荐和安全访问等核心服务。

### 3.2.1用户管理

Sparrow系统用户包含如下类型：

1. System Root：系统管理员；Sparrow初始化时，默认存在。对整个系统资源具有全部访问权限。
2. Customer Root：租户管理员（如c1root）；由System Root依据服务协议创建，在创建时，同时创建其租户根目录[/c1]、业务根目录[/c1/bus]、组织根目录[/c1/rog]、租户管理员用户个人根目录[/c1/home/c1root]。

表 \*\* 租户操作

|  |  |  |
| --- | --- | --- |
| 序号 | 操作 | 说明 |
| 1 | 新建  /custroot/new |  |
| 2 | 读取（META）  /custroot/read/meta |  |
| 3 | 检索/列表（META）  /custroot/list/meta |  |
| 4 | 检索/列表  /custroot/list/index | 见多维检索  （Index，索引数据库ES，多维检索） |
| 5 | 更新（META）  /custroot/update/meta |  |
| 6 | 删除  /custroot/delete | 不支持 |
| 7 | 使能  /custroot/access | Access = 0 or 1 |
| 8 |  |  |
| 9 |  |  |

1. System User: 系统普通用户，有System Root创建，并授予相关资源访问权限。
2. Customer User：租户普通用户，由Customer Root创建，并授予相关资源访问权限。

表 \*\* 业务用户

|  |  |  |
| --- | --- | --- |
| 序号 | 操作 | 说明 |
| 1 | 新建  /user/new |  |
| 2 | 读取（META）  /user/read/meta |  |
| 3 | 检索/列表（META）  /user/list/meta |  |
| 4 | 检索/列表  /user/list/index | 见多维检索  （Index，索引数据库ES，多维检索） |
| 5 | 更新（META）  /user/update/meta |  |
| 6 | 删除  /user/delete |  |
| 7 | 使能  /user/access | Access = 0 or 1 |
| 8 |  |  |
| 9 |  |  |

### 3.2.2安全访问

安全访问指防止未授权用户访问受保护的系统资源，如目录、文档和文件等。

假设租户[c1]下的系统用户[A]对目录[/c1/bus/dir1]进行读取操作。这涉及如下三个概念：

1） 业务操作的主体：系统用户，在sparrow中，普通系统用户总是从属一个租户。

2） 业务操作：对目录进行读取操作。

3） 业务操作的客体：目录（资源）。

对上述资源访问行为，Sparrow提供面向资源与许可的访问控制服务，并同时支持两种具体机制：1）扩展Linux权限管理策略，提供细粒度的访问控制机制，和2）层次权限继承管理策略，提供粗粒度的访问控制机制。

Sparrow将海量的文件存储在Minio对象存储服务中。Minio为文件的CRUD提供基于秘钥及安全Token的访问控制机制，即Minio为用户的访问行为提供授权Token，用户在Token有效期内凭借该Token实施被授权的访问。这是文件安全访问的基础。

Sparrow的安全访问控制策略实施在目录（和文档）上，即就用户对目录（和文档）的访问行为（执行、删除、创建/更新和读取）进行授权管理。值得注意的是，由于文件由其父级文档统一管理，因此，用户对文档的权限传递给该文档管理的文件。

* 扩展Linux权限模型

租户Root用户和资源拥有者，可为对应资源（目录或文档）创建若干普通用户群组（Regular Group，RG），并授予该群组对该资源的一组访问操作许可（RG Permission，RGP）。普通用户群组可包含0或多个用户，并仅对应一个资源，形成资源、用户与操作（许可）的三元组。RGP的操作许可，仅对对应目录与文档有效。

在创建目录和文档时，系统默认创建一个默认RGP。该RGP被授予全部操作细节，并将创建者作为其的唯一用户，即拥有者（Owner）。

此访问控制策略，具有细粒度和灵活的特点，但是授权过程复杂，成本高。

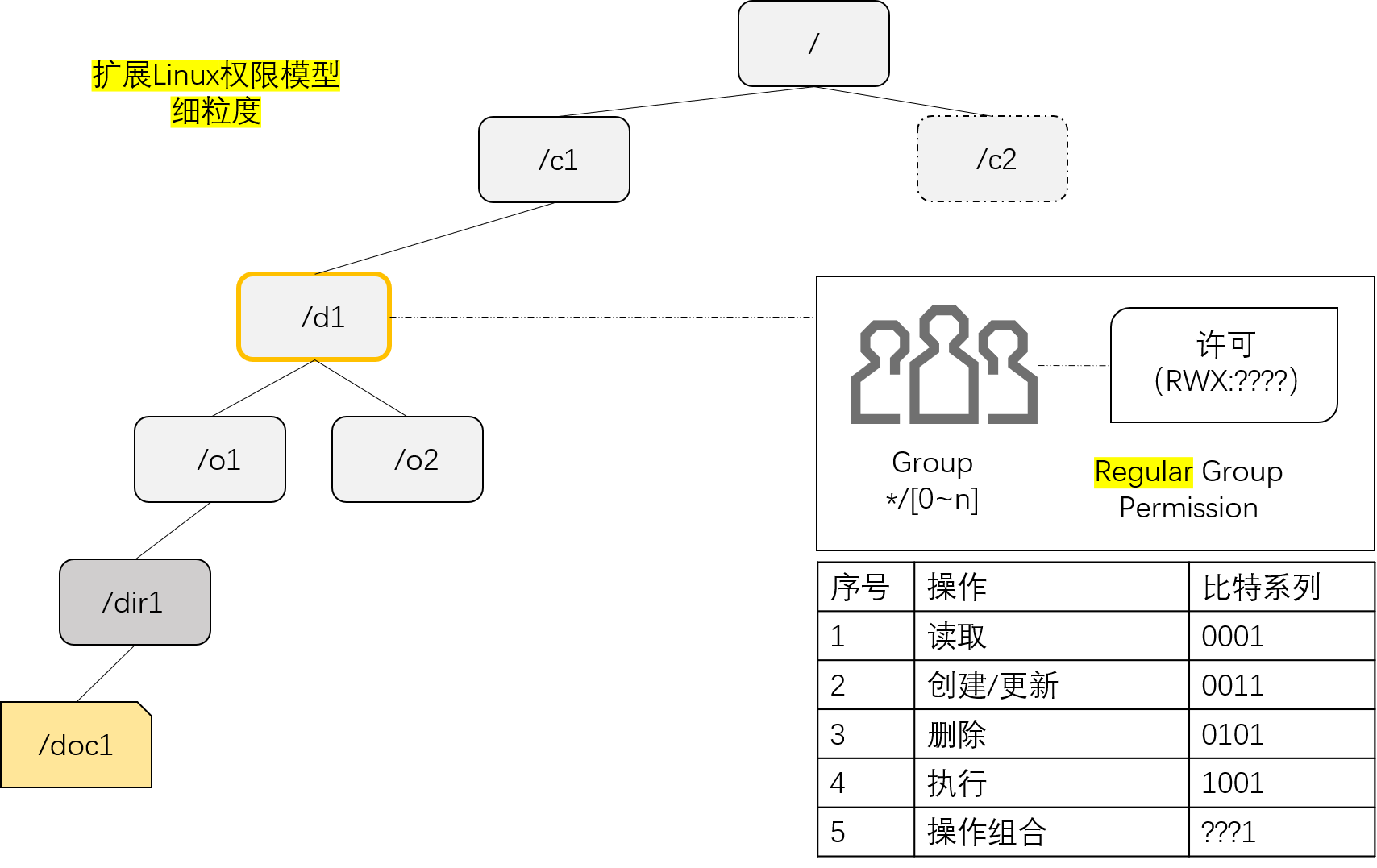


图 3 扩展Linux文件权限模型

* 层次权限继承模型

Root用户可为资源（仅目录）创建若干超级用户群组（Supper Group，SG），并授予该群组对该资源的一组访问操作许可（SG Permission，SGP）。超级用户群组可包含0或多个用户，并仅对应一个资源，形成资源、用户与操作（许可）的三元组。SGP的操作许可，对该目录及其任何深度的子目录（和文档）有效。

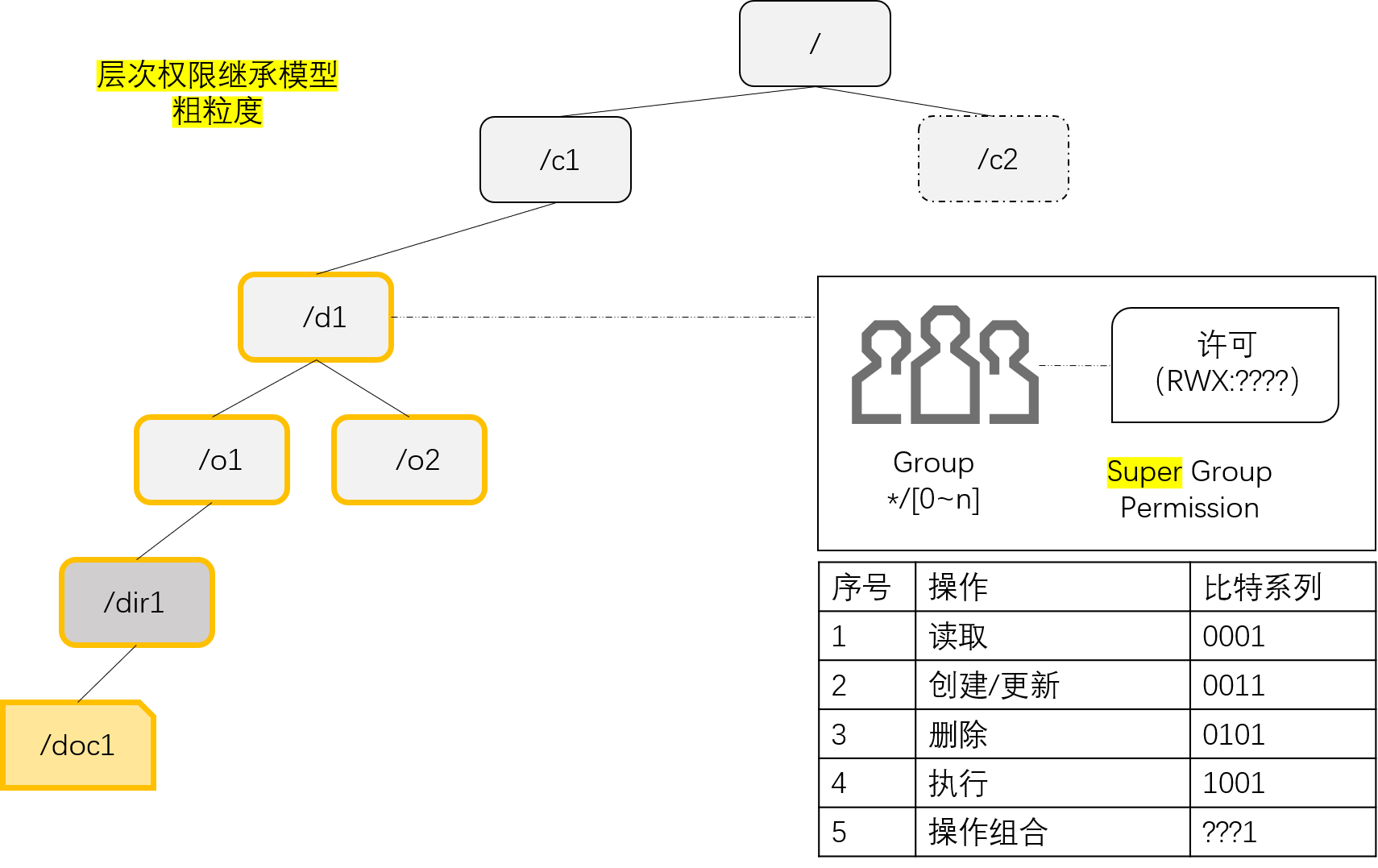


图 4层次权限继承模型

除前述面向资源与操作许可的访问控制策略，Sparrow还提供租户/用户与前述根目录（租户、业务、机构和用户）的开放/关闭访问管理。开放与关闭管理是一种非常粗放式的访问控制策略，通常用在应急响应情况。

* **租户开放**，指该租户下所有系统用户依据前述面向资源的操作许可机制进行系统资源访问；**租户关闭**，拒绝该租户下所有系统用户登录。相应，用户开放与关闭的控制策略仅对该用户有效。仅Sparrow root系统管理员用户具有开闭租户的权限。
* 租户/业务/机构和用户**根目录开放**，系统用户依据前述面向资源的操作许可机制对根目录的进行访问；业务/机构和用户**根目录关闭**，拒绝系统用户对该根目录的访问。Sparrow root系统管理员用户和customer root租户管理员用户具有开闭根目录的权限。

对已登录用户访问资源（目录、文档和文件）进行许可决策的过程如下：

1. 用户是否具有对该资源进行目标操作的许可；
2. 该资源所属的根目录是否开放；

表 \*\* 许可

|  |  |  |
| --- | --- | --- |
| 序号 | 操作 | 说明 |
| 1 | 新建  /permission/new |  |
| 2 | 读取（META）  /permission/read/meta |  |
| 3 | 检索/列表（META）  /permission/list/meta |  |
| 4 | 检索/列表  /permission/list/index | 见多维检索  （Index，索引数据库ES，多维检索） |
| 5 | 更新（META）  /permission/update/meta |  |
| 6 | 删除  /permission/delete |  |
| 7 | 使能  /permission/access | Access = 0 or 1 |
| 8 |  |  |
| 9 |  |  |

### 3.2.3高效存取

高效存取涉及数据对象（目录、文档和文件）的CRUD操作和检索等操作。

Sparrow基于PostgreSQL实现目录组织（郑铠锋），Minio实现海量文件存储（陈晓滨、梁宏达），ElasticSearch存储文档和文件的数据模型数据（元数据，Meta Data）（陈绿佳）以提供多维检索功能及基于内容推荐。

在上传文件后，Sparrow支持文件缩略图生成；如果是视频文件，支持截取若干帧图片，采用base64编码存储（不建ElasticSearch反向索引），后续支持图片检索。对文本类型文件，如doc，转换为pdf后，再生成缩略图，采用base64编码存储（不建ElasticSearch反向索引）。

文档是为文件高效管理而设计的（见\*\*\*）。与普通目录可能包含较多子目录或文档（文件）不同，文档目录通常仅包含较少（<10）的文件，可有效支持整个文档目录的下载、备份等操作；但是，

### 3.2.4多维检索

从检索对象角度，Sparrow支持检索目录、文档和文件。

目录检索：假设某目录包含50多个子目录或文档，系统支持在目录下检索名称包含检索关键字的目录（或文档）；可采用客户端，如Web浏览器，进行字符串模糊检索实现；

文档检索：构建文档数据模型的ElasticSearch索引，提供多维结构化检索与全文检索。

文件检索：构建文件数据模型的ElasticSearch索引，提供多维结构化检索与全文检索。

为支持多维检索，Sparrow构建了4个ElasticSearch索引库（Index），图示如下：

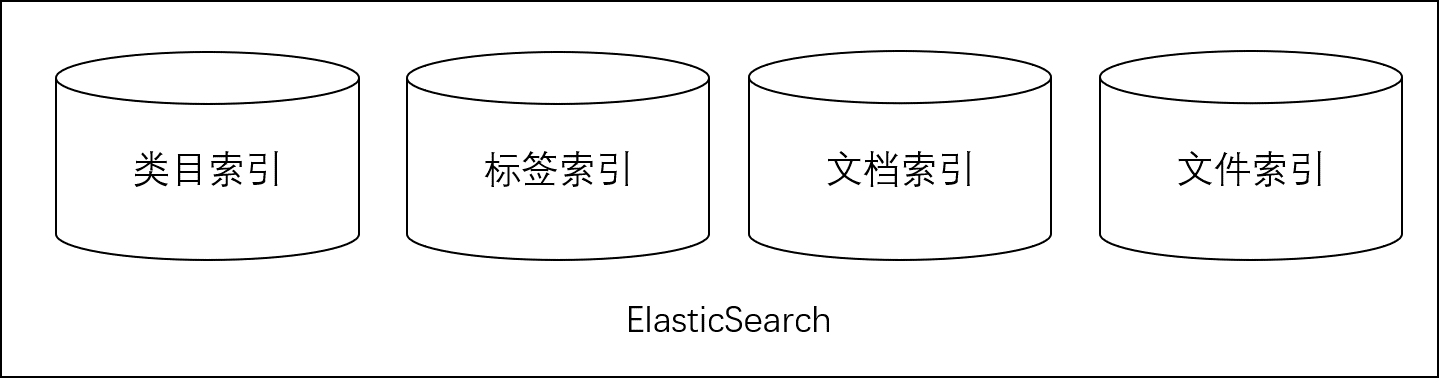


图 3 ElasticSearch索引

文档检索与文件检索，待分析陈绿佳的设计。

### 3.2.5智能推荐

待设计与整合！！基于内容的推荐，和协同推荐！待设计！（苏恒）

# 4 SparrowData Web Admin

Sparrow Web Admin是 … …。（蔡东林）

涉及技术：

|  |  |  |
| --- | --- | --- |
| 序号 | 技术 | 说明 |
| 1 | Java Servlet | Java平台开发Web应用的基础； |
| 2 | Spring IOC/Bean | IOC：控制反转与依赖注入的容器机制 |
| 3 | Spring Boot /Web MVC | 对Java Servlet的封装，一种快速开发Web应用/微服务应用的框架 |
| 4 | Spring AOP | 面向切面的编程，在不修改已有业务代码情况下，增加系统功能 |
| 5 | Spring Security | 用户访问控制（认证与授权），可扩展！ |
| 6 | PostgreSQL | 关系型数据库，支持JSON字段的高性能处理 |
| 7 | Minio | S3规范的对象存储系统，用来存储海量小文件 |
| 8 | 阿里云OSS | 对象存储系统，用来存储海量小文件 |
| 9 | ElasticSearch | 高性能索引数据系统 |
| 10 | Redis、或者其他缓存 | 基于内存的缓存系统，为PostgreSQL提供缓存服务 |
| 11 | Thumbor | 处理处理服务，裁剪、缩放与旋转等 |
| 12 | Nginx | HTTP反向代理与负载均衡 |
| \* | ThingsBoard | 一个物联网平台，学习它对代码的模块划分、整合；技术的综合运用 |

基于Minio和Thumbor搭建独立图片服务

https://hearrain.com/ji-yu-minio-he-thumbor-da-jian-du-li-tu-pian-fu-wu