# 供应链金融

## 供应链金融



### 什么是供应链金融

供应链金融就是供应链中上下游中小微企业通过自身的真实交易和应收账款和订单等信息，来获取平台的授信，通过资金方获取资金，解决自身融资问题

### 系统角色

#### 资产方(卖方)

在供应链金融中处于弱势地位，因为要垫付资金导致资金周转有问题，需要通过应收账款或者订单或者货物或者信用等抵押发起融资

#### 核心企业(买方)

处于优势位置，但是通过平台获取资金，可以提高资金周转率，相当于变相增加了收入

#### 平台(技术和数据和资金服务等)

参与供应链贸易场景，提供技术支持数据服务资金服务等，主要是供应链金融技术和数据服务平台

#### 资金方

提供资金支持(包含自有资金和三方资金)等

### 业务流程

#### 平台入驻

供应链上下游企业入驻平台(注册登录，创建用户信息，提供客户信息和企业信息，创建账户)

#### 上传资料

供应链上下游企业上传资料(企业资料管理) ，尽调，授信，核定授信额度(授信管理)

#### 展业

供应链上下游企业选择对应的业务模式，金融产品(金融产品管理) ，开展业务

#### 下单

根据合作厂家的商品(商品管理) ，签署订单，产生订单凭证

#### 审核

融资方向平台申请融资服务(流程管理) ，平台进行核实

#### 合同

基于真实交易，签署对应的贸易服务合同(合同管理)

#### 资金管理

基于资金渠道(资金路由管理) ，提供对应的融资方案(资金管理)

#### 账单

根据支付凭证，产生对应的账单(账单管理)和票据(发票管理)和账务管理(账务管理)

#### 贷后

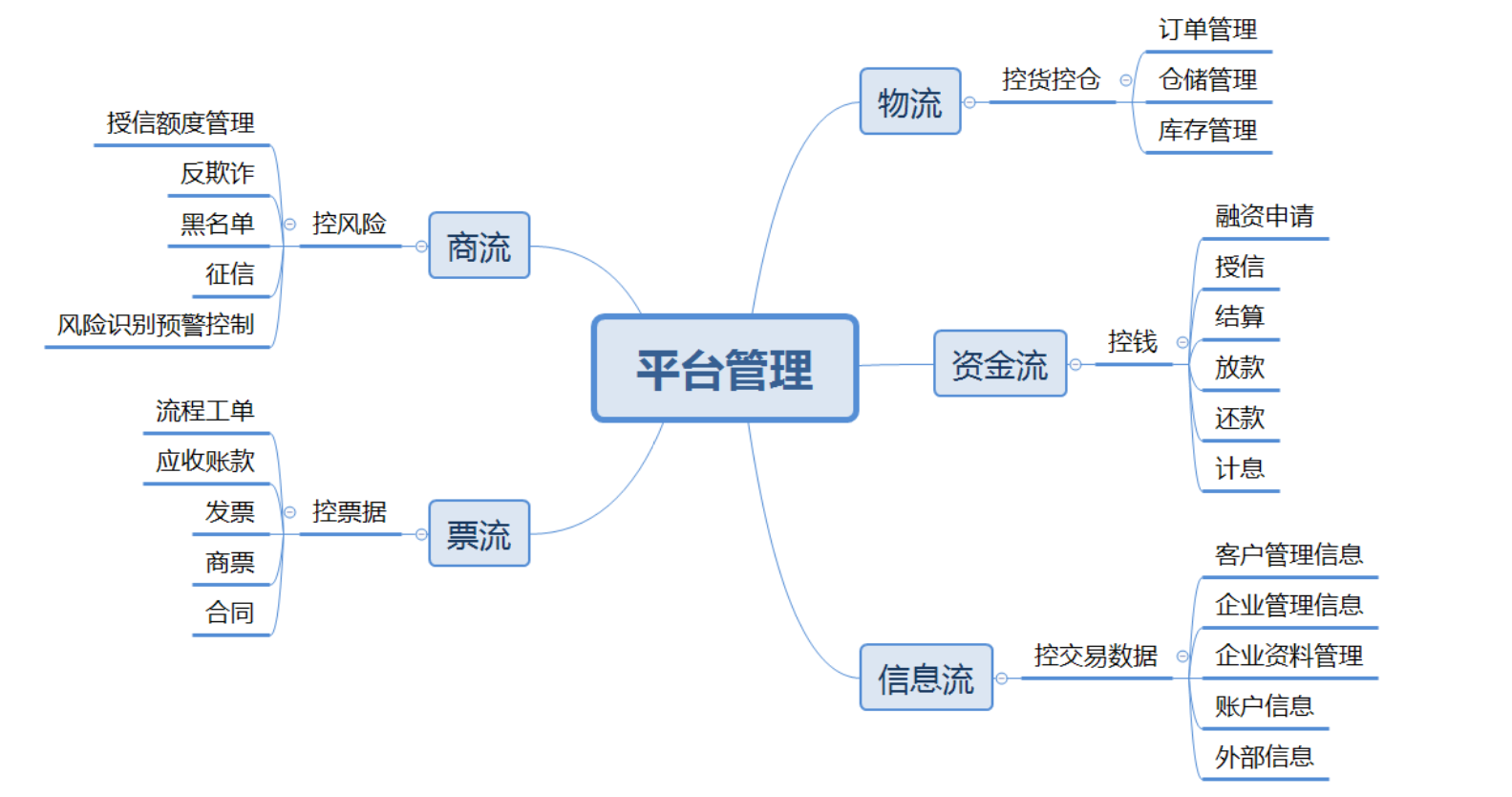
平台向融资方或者担保方进行催收，管理账款(贷后管理)(清结算服务)

#### 风控

风控贯穿全生命周期，提供风险识别，监控等服务

### 供应链金融平台

要从供应链金融管理的角度，搭建平台，要做到资金流，物流，信息流，商流，票据等的控制管理

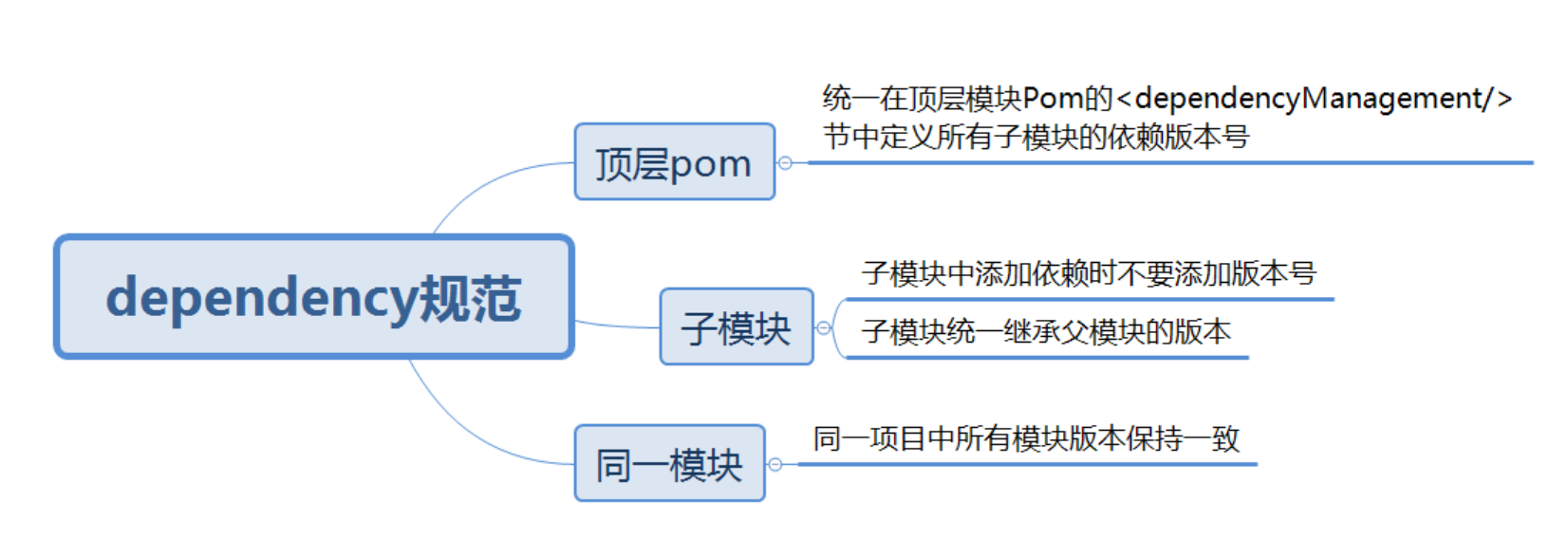


## 供应链金融规范设计

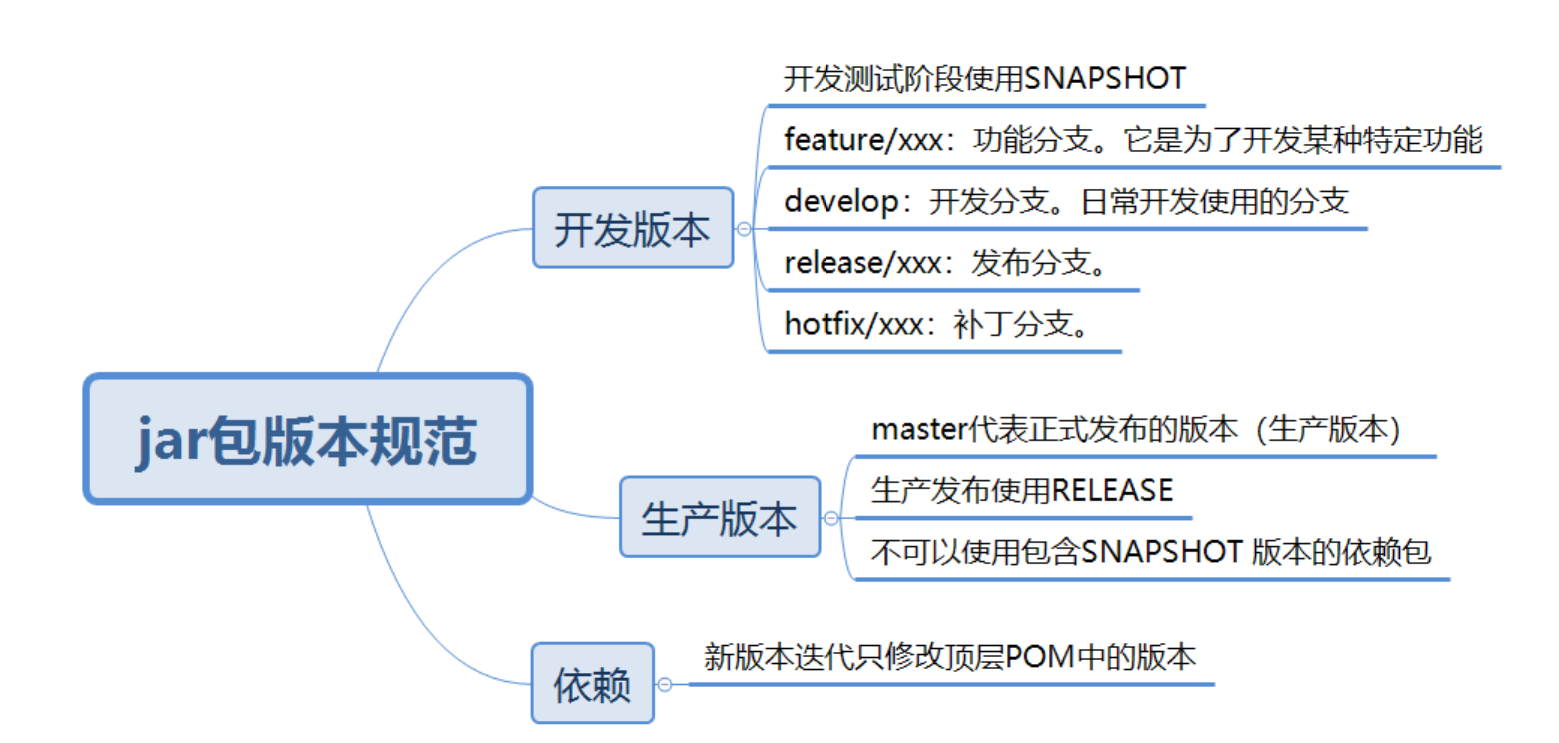
规范设计可以减少系统的复杂度，架构本身就是一种规范。

### 版本规范

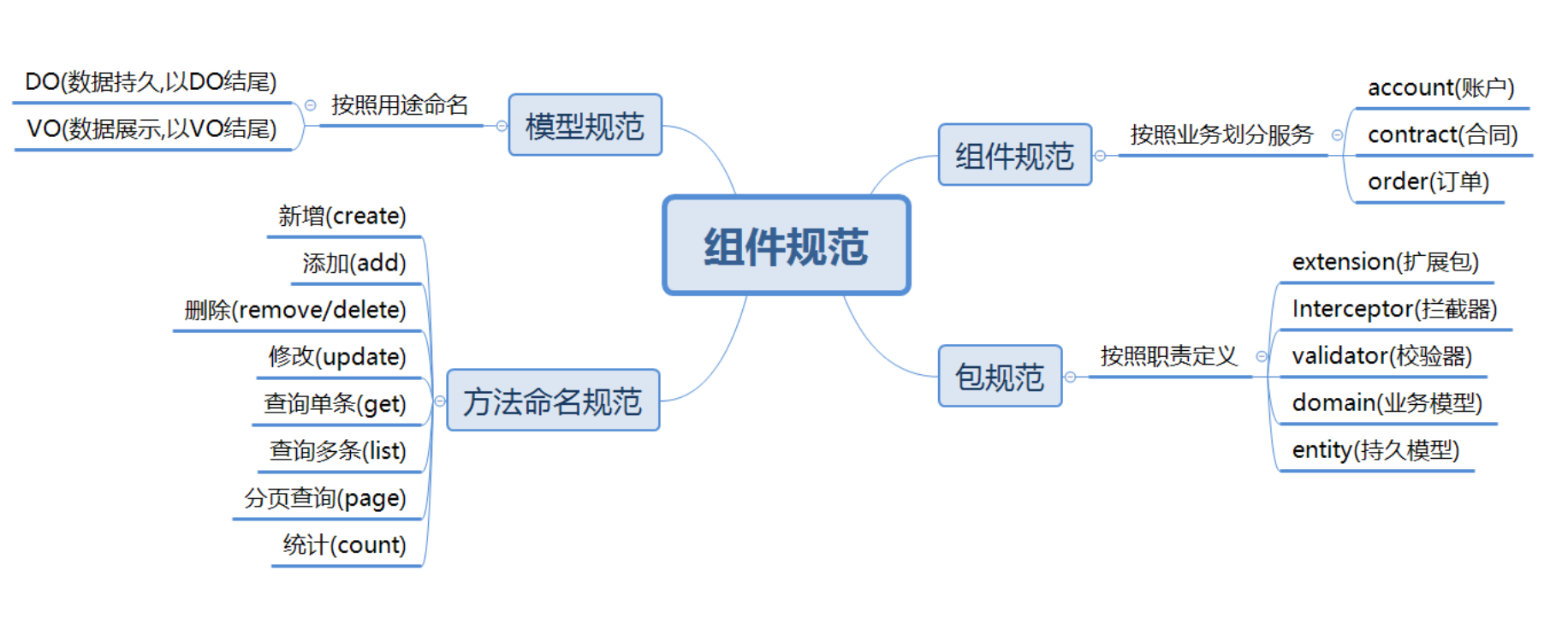
#### dependency版本规范



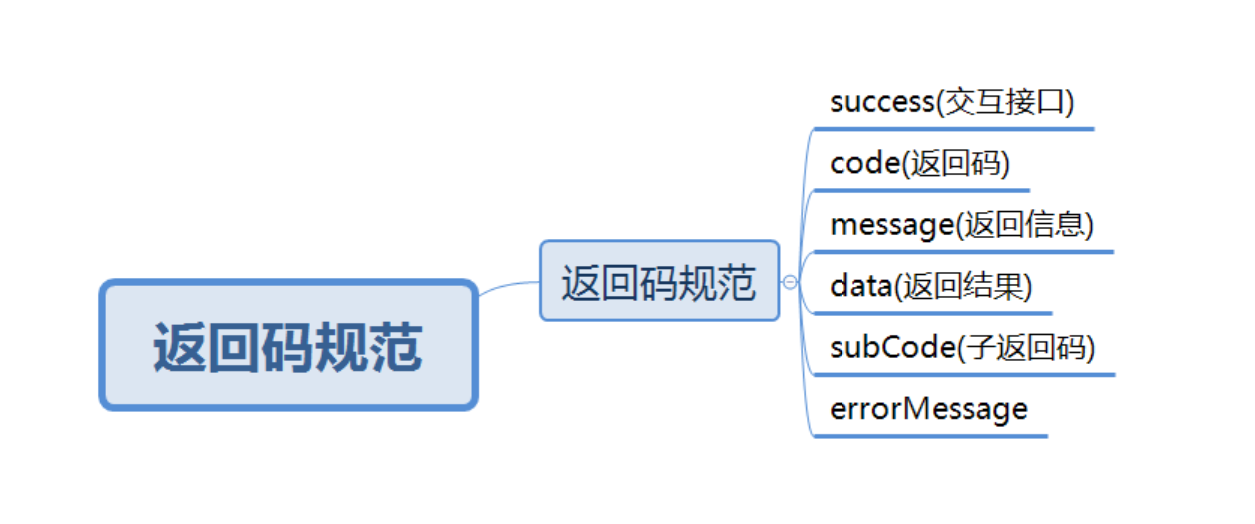
#### jar包规范



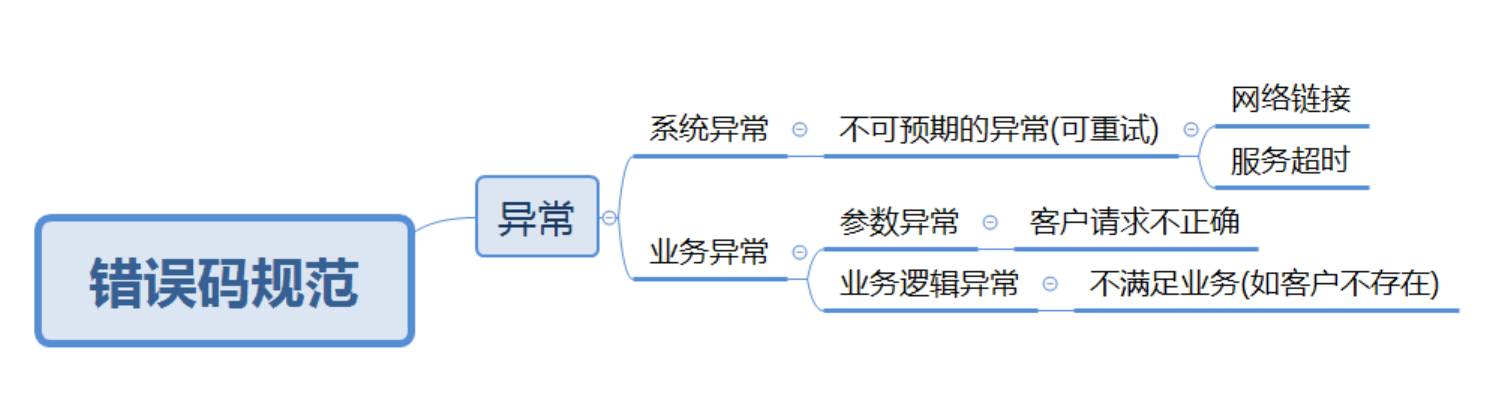
### 组件规范



### 返回码规范

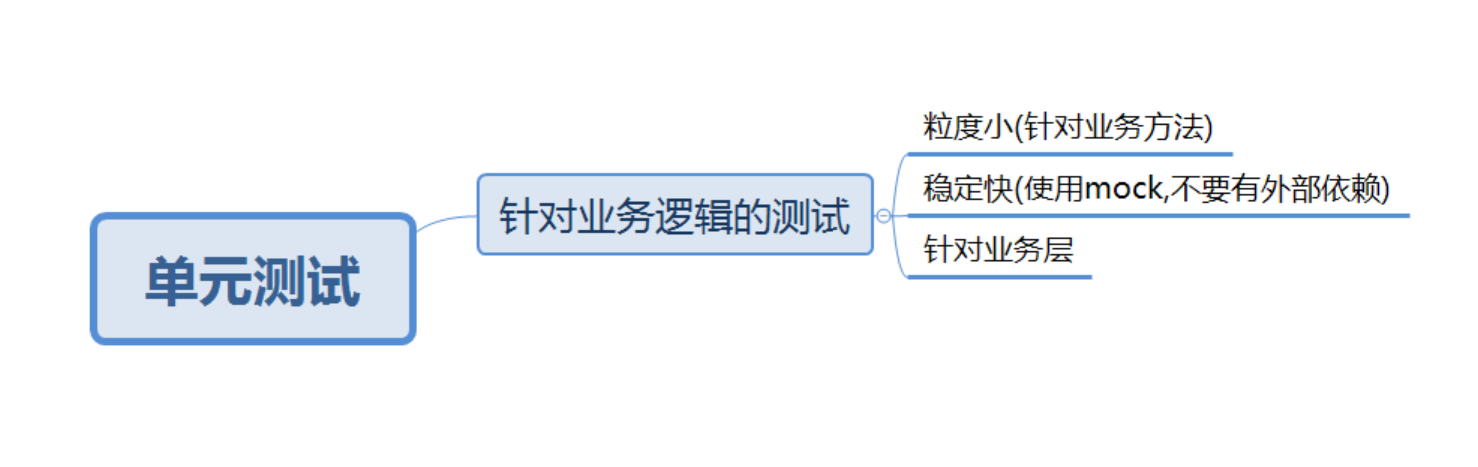


### 错误码规范



### 测试规范

#### 单元测试



#### 集成测试



## 供应链金融系统

### 基础服务

|  |  |
| --- | --- |
| id-service | ID生成器 |
| file-service | 文件服务器 |
| enum-service | 枚举服务 |
| sms-service | 信息发送服务 |
| activiti-service | 流程引擎 |
| rule-service | 规则引擎 |
| config-service | 参数配置 |
| mq-service | 消息配置 |
| es-service | 查询管理 |
| report-service | 报表管理 |
| log | 日志追踪系统 |

### 业务服务系统

|  |  |
| --- | --- |
| sso-service | 单点登录系统 |
| login-service | 用户登录管理系统 |
| oauth-service | 权限管理系统 |
| corp-service | 企业管理系统 |
| document-service | 企业档案管理系统 |
| antifraud-service | 反欺诈系统 |
| data-service | 三方数据系统 |
| risk-service | 风控系统 |
| account-service | 账户管理系统 |
| credit-service | 授信管理 |
| product-service | 金融产品系统 |
| item-service | 商品管理系统 |
| order-service | 订单管理系统 |
| work-service | 工单系统 |
| bill-service | 账单系统 |
| transaction-service | 账务系统 |
| fund-service | 资金管理系统 |
| route-service | 渠道路由系统 |
| payment-service | 支付系统 |
| invoice-service | 开票系统 |
| fee-service | 费用系统 |
| after-service | 贷后系统 |
| alert-service | 告警系统 |
| vip-service | 会员管理系统 |
| marketing-service | 营销管理系统 |
| crm-service | 客户关系管理系统 |
| contract-servie | 合同协议管理系统 |
| portrait-service | 用户画像服务 |

### 外围服务系统

|  |  |
| --- | --- |
|  | 同盾数据 |
|  | 魔蝎数据 |
|  | 企查查数据 |
|  | 法海数据 |
|  | e签宝 |
|  | 鹏元征信 |
|  | 工商银行等银行 |

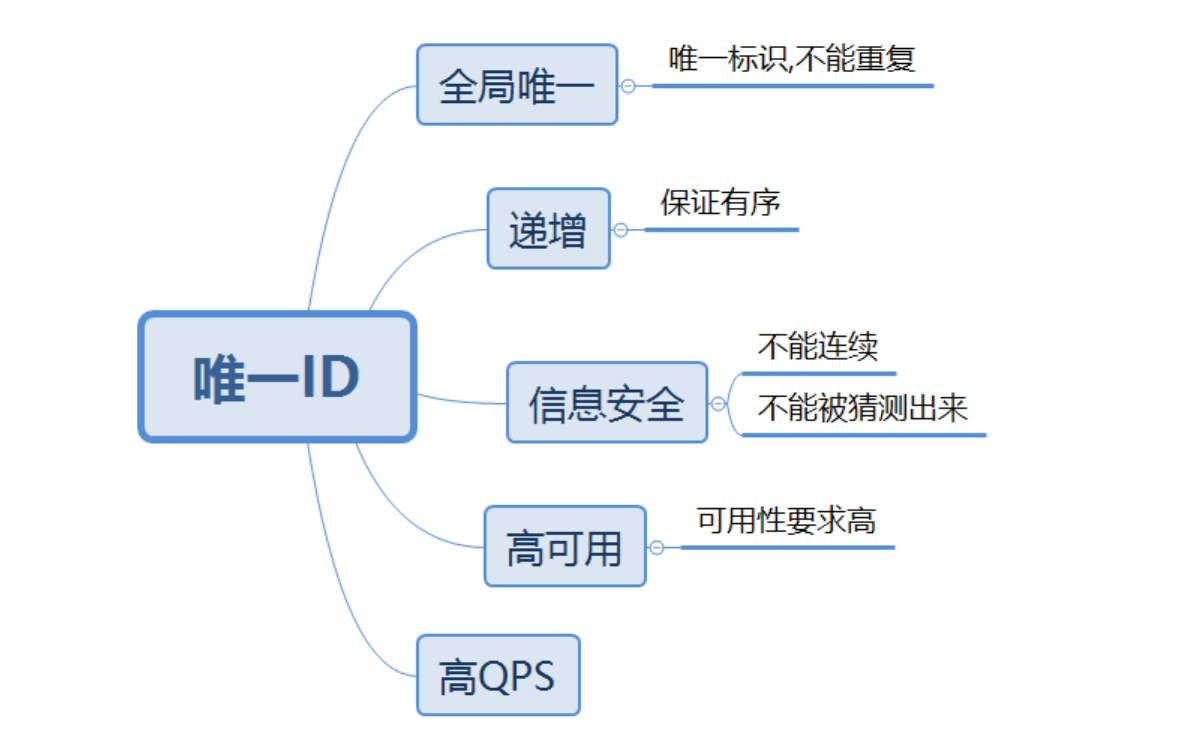
## 基础服务详情

### ID生成器(id-service)

#### 背景

在复杂的分布式系统中，需要对数据或者消息进行唯一标识，主要为了分库分表或者进行消息幂等等

#### 需求

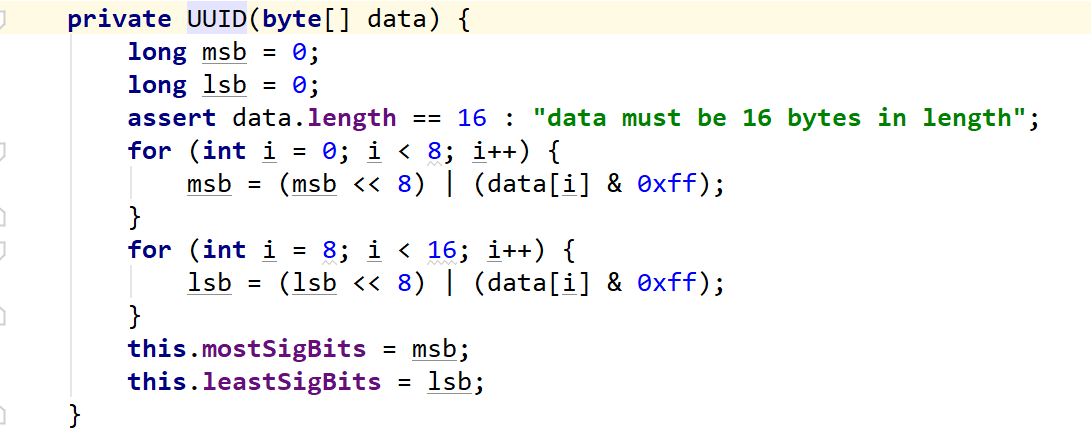


#### 实现

##### UUID

UUID的标准形式包含32个16进制数字，形式如下

550e8400-e29b-41d4-a716-446655440000



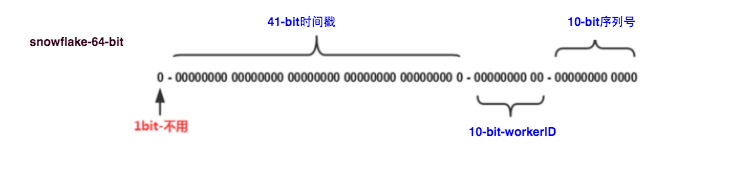
优点：

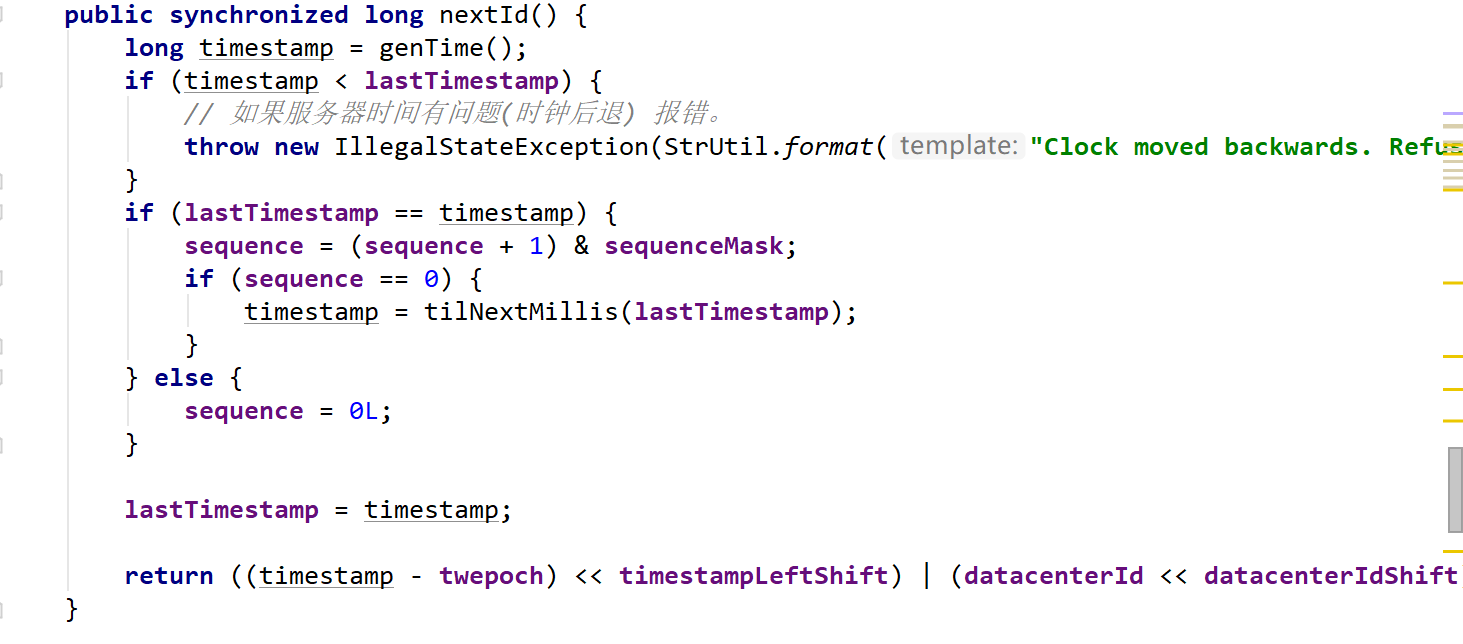
性能高

缺点：

不易存储：作为数据库主键不适合

##### 类snowflake



它给每台机器分配一个唯一标识，然后通过时间戳+标识+自增实现全局唯一ID。

Snowflake算法采用41bit毫秒时间戳，加上10bit机器ID，加上12bit序列号，理论上最多支持1024台机器每秒生成4096000个序列号

优点：

毫秒数在高位，自增序列在低位，整个ID都是趋势递增的

可以根据自身业务特性分配bit位

不依赖数据库等第三方系统，以服务的方式部署，稳定性更高

缺点:

强依赖机器时钟，如果机器上时钟回拨，会导致发号重复或者服务会处于不可用状态。

应用举例：Mongdb objectID

通过“时间+机器码+pid+inc”共12个字节，通过4+3+2+3的方式最终标识成一个24长度的十六进制字符。

##### 数据库

|  |
| --- |
| begin;  REPLACE INTO Tickets64 (stub) VALUES ('a');  SELECT LAST\_INSERT\_ID();  commit; |

优点：

非常简单，利用现有数据库系统的功能实现，成本小

ID号单调自增

缺点：

强依赖DB，当DB异常时整个系统不可用，属于致命问题

ID发号性能瓶颈限制在单台MySQL的读写性能

##### 如何实现id-service的高可用

使用Zookeeper持久顺序节点(PERSISTENT\_SEQUENTIAL)的特性完成对snowflake节点配置wokerID

1. 启动id-service,连接ZK,检查是否在id-service节点下注册
2. 如果注册过,取回自己的workerID,启动服务
3. 如果没注册过,如果在id-service节点下注册成功,取顺序号作为workerID

### ****文件服务器(file-service)****

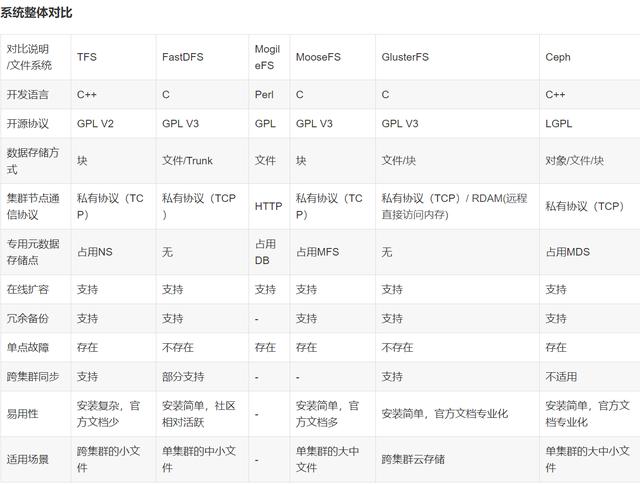
#### 背景

数据是数据服务型公司的最有价值的资源，分布式文件存储是应对数据爆炸的最好解决方案

#### 需求

那么该如何处理大数据的存储，计算分析呢？

#### 实现



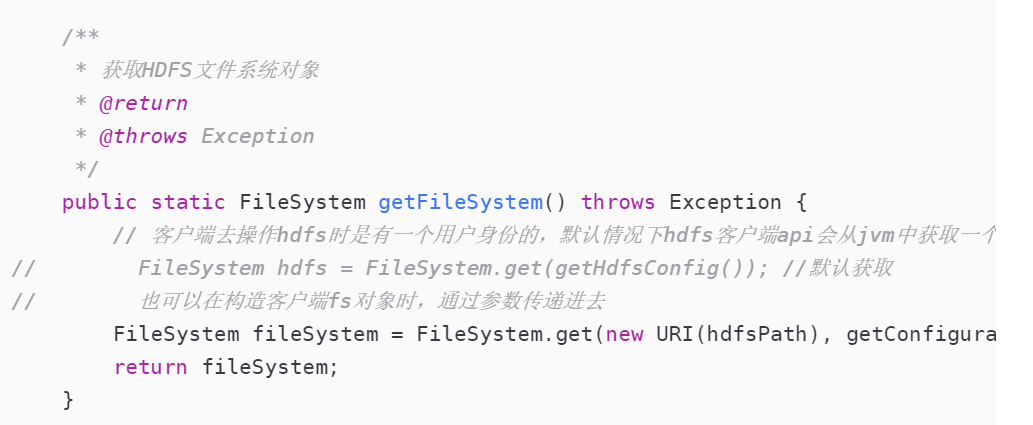
##### GFS（Google File System）

Google公司为了满足本公司需求而开发的基于Linux的专有分布式文件系统。没有开源。

##### HDFS

Hadoop 实现了一个分布式文件系统（Hadoop Distributed File System），简称HDFS。 Hadoop是Apache Lucene创始人Doug Cutting开发的使用广泛的文本搜索库。它起源于Apache Nutch，

后者是一个开源的网络搜索引擎，本身也是Luene项目的一部分。Aapche Hadoop架构是MapReduce算法的一种开源应用，是Google开创其帝国的重要基石。



##### TFS(Taobao FileSystem)

TFS（Taobao FileSystem）是一个高可扩展、高可用、高性能、面向互联网服务的分布式文件系统，主要针对海量的非结构化数据，它构筑在普通的Linux机器 集群上，可为外部提供高可靠，好长时间没有更新了

##### OSS

OSS 是一个分布式的对象存储服务，提供的是一个 Key-Value 对形式的对象存储服务。用户可以根据 Object 的名称（Key）唯一的获取该Object的内容。



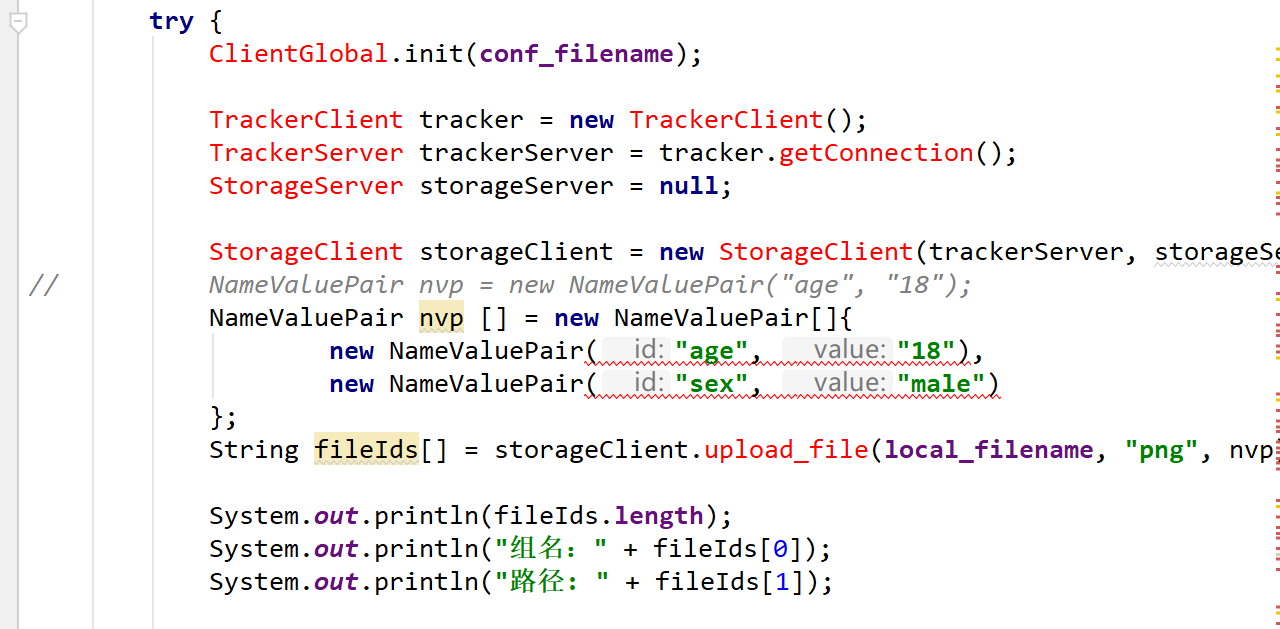
##### GridFS

具有多机器存储备份，可以突破一般文件系统对file的限制，分段存储，不像普通file system是整个存储的。这样读取大型文件时就不会占用大量的内存。



##### FastDFS

FastDFS是一个开源的轻量级分布式文件系统，它对文件进行管理，功能包括：文件存储、文件同步、文件访问（文件上传、文件下载）等，解决了大容量存储和负载均衡的问题。



### 信息服务系统(sms-service)

消息发送

(短信) (邮箱)(钉钉消息)等第三方消息

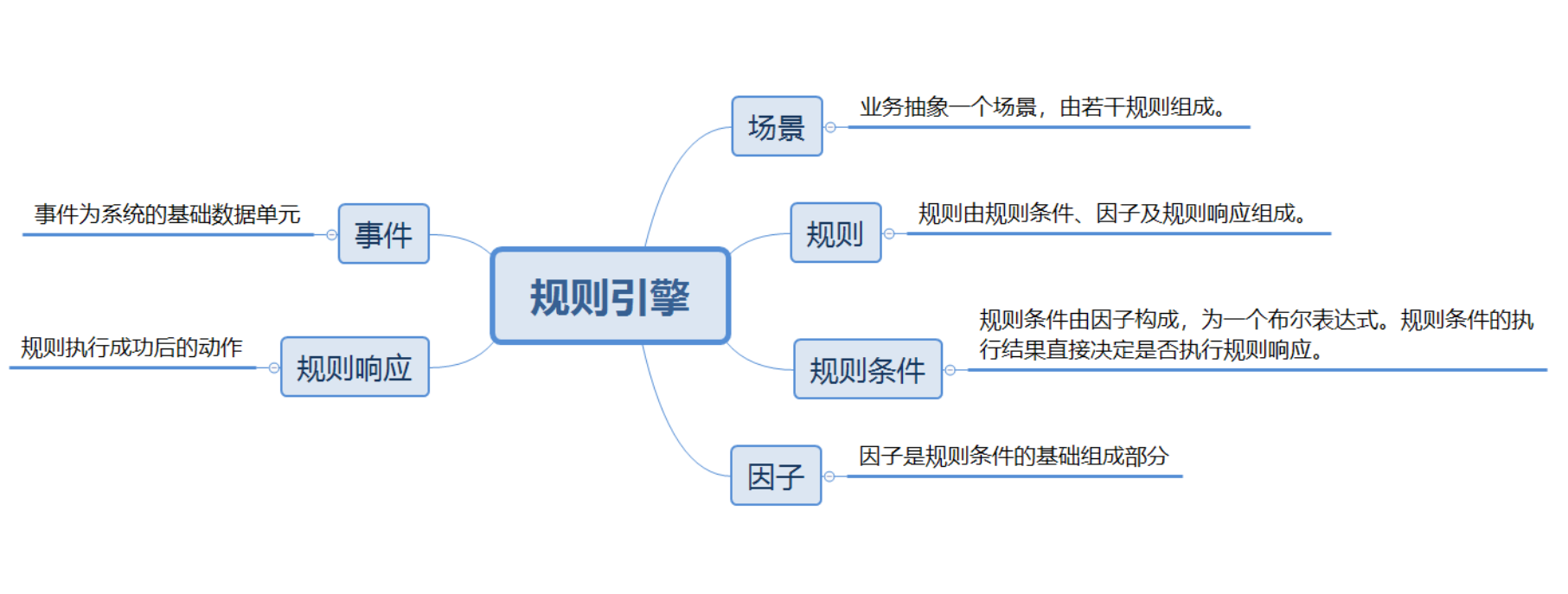
### 流程引擎(activiti-service)

### 规则引擎(rule-service)

##### 背景

作为风控系统的风控规则，每种业务评分规则会经常变动，为适应这样的业务需求，可以采用 drools 开源规则引擎。

##### 规则模型



根据业务场景配置规则(如果规则不同，需要根据业务场景将规则进行分组，形成规则集) ，不同的规则需要配置规则条件，规则条件由因子构成，为一个布尔表达式。规则条件执行后，需要调用规则相应，执行后续操作

##### 实现



### 参数配置(config-service)

### 消息配置(mq-service)

### 查询管理(es-service)

### 报表管理(report-service)

### 日志追踪(log)

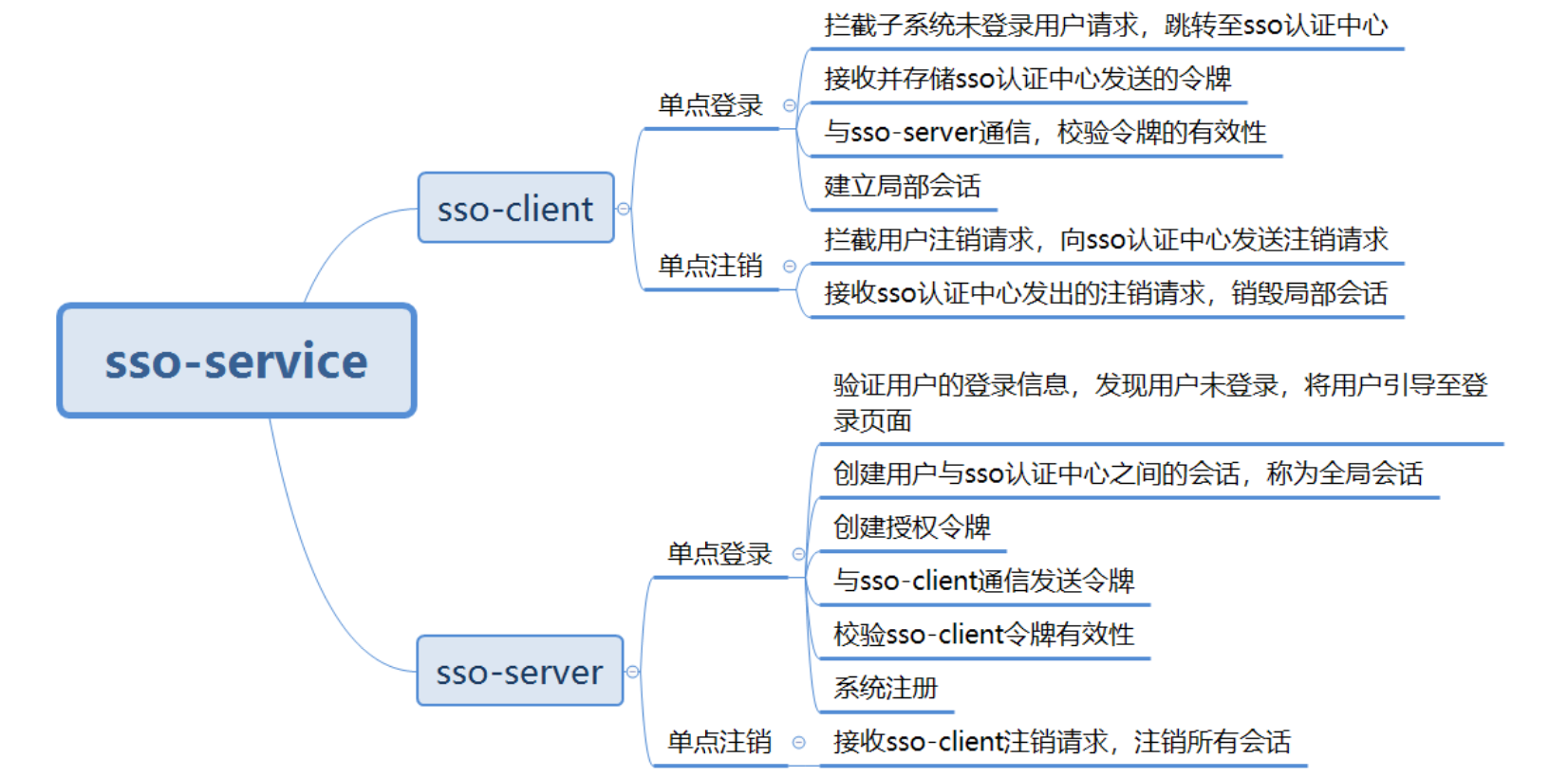
## 业务服务系统详情

### 单点登录系统(sso-service)

##### 背景

单点登录全称Single Sign On（以下简称SSO），是指在多系统应用群中登录一个系统，便可在其他所有系统中得到授权而无需再次登录，包括单点登录与单点注销

##### 需求



##### 实现



##### 如何实现服务高可用

保存用户会话信息使用Redis，支持集群部署

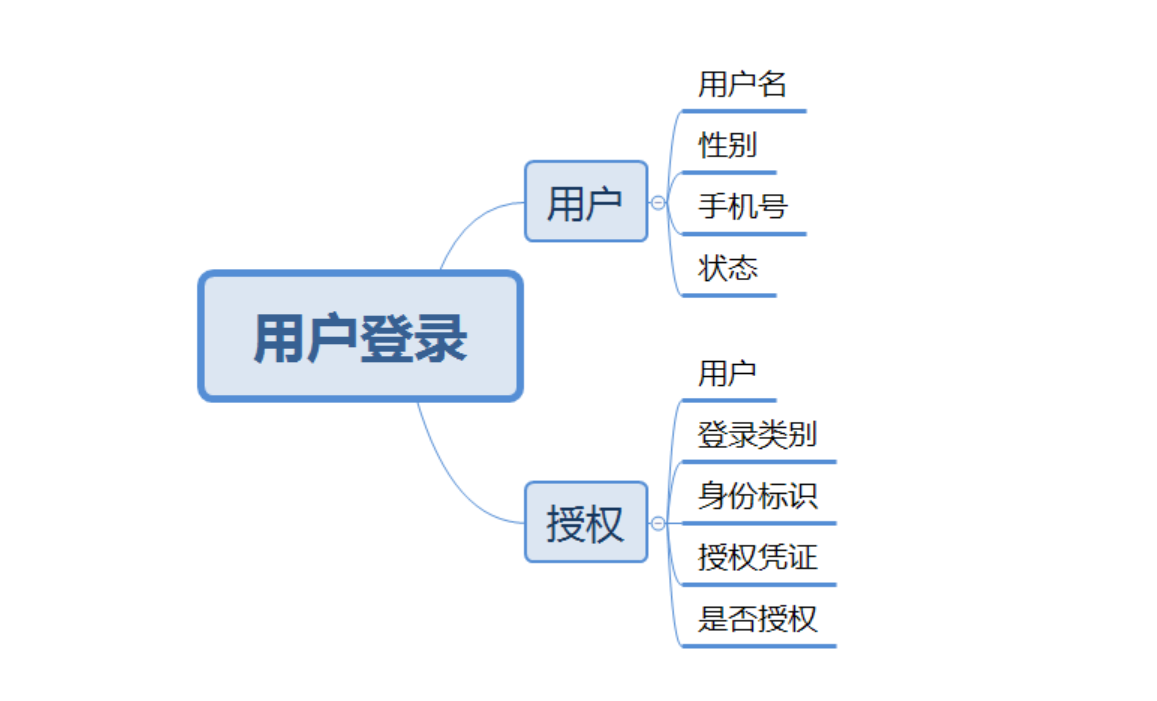
原则上可以无限扩展sso-client和sso-server

### 用户登录管理系统(login-service)

##### 背景

用户最基本的登录方式是站内账号登录，但是往往需要更多的登录方式，例如小程序登录，手机号登录，邮箱登录等

##### 模型



##### 如何实现百万级别的用户登录

使用Hash算法分库存储用户信息

### 用户权限管理系统(oauth-service)

##### 背景

系统的使用者通常称为用户，用户登录后，系统通过各种标识赋予用户操作和查看权限。

##### 需求

**如何统一用户身份**

用户分为外部用户和内部用户，平台运维公司的员工称为内部用户，其他用户称为外部用户。

统一身份有利于对用户统一进行鉴权，进行角色分配

**如何设置用户角色**

用户角色主要用来操作用的相关权限信息

角色可以被继承，例如赋予某企业外部用户admin权限，该企业其它其用户只能从admin中选择权限

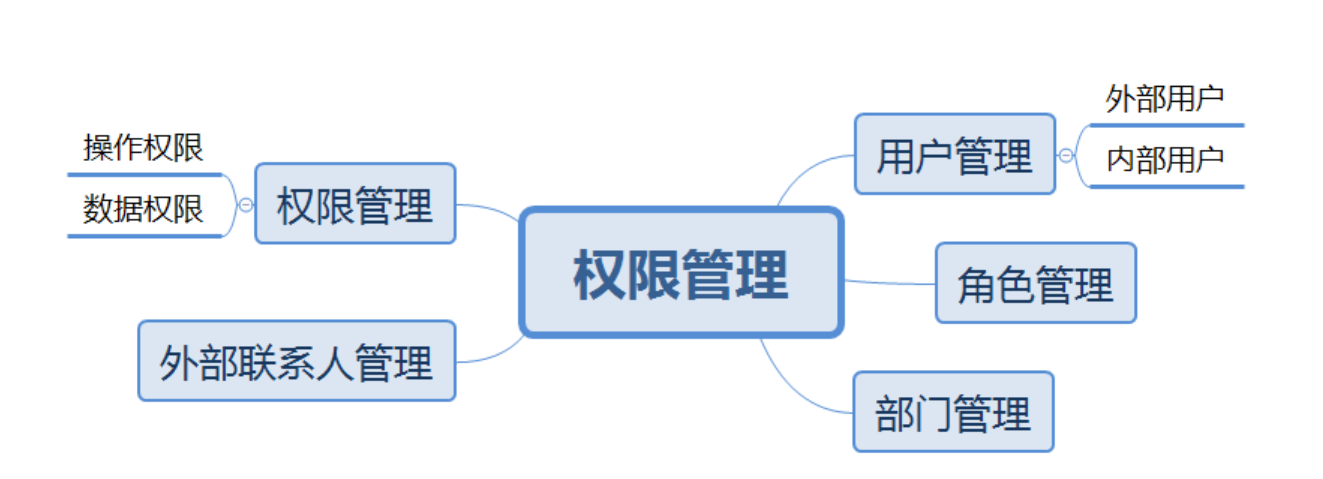
**如何配置用户权限**

用户权限可以分为操作权限和数据权限

**如何配置用户组织架构管理**

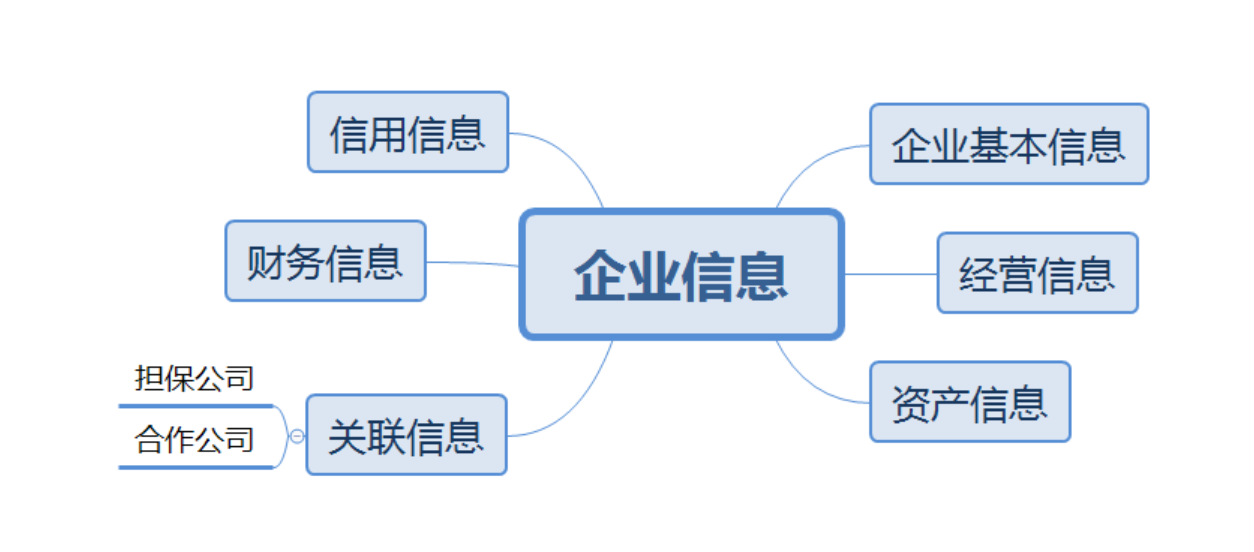
按照公司业务发展和组织架构进行调整

##### 实现



### 企业管理系统(corp-service)

#### 企业信息管理

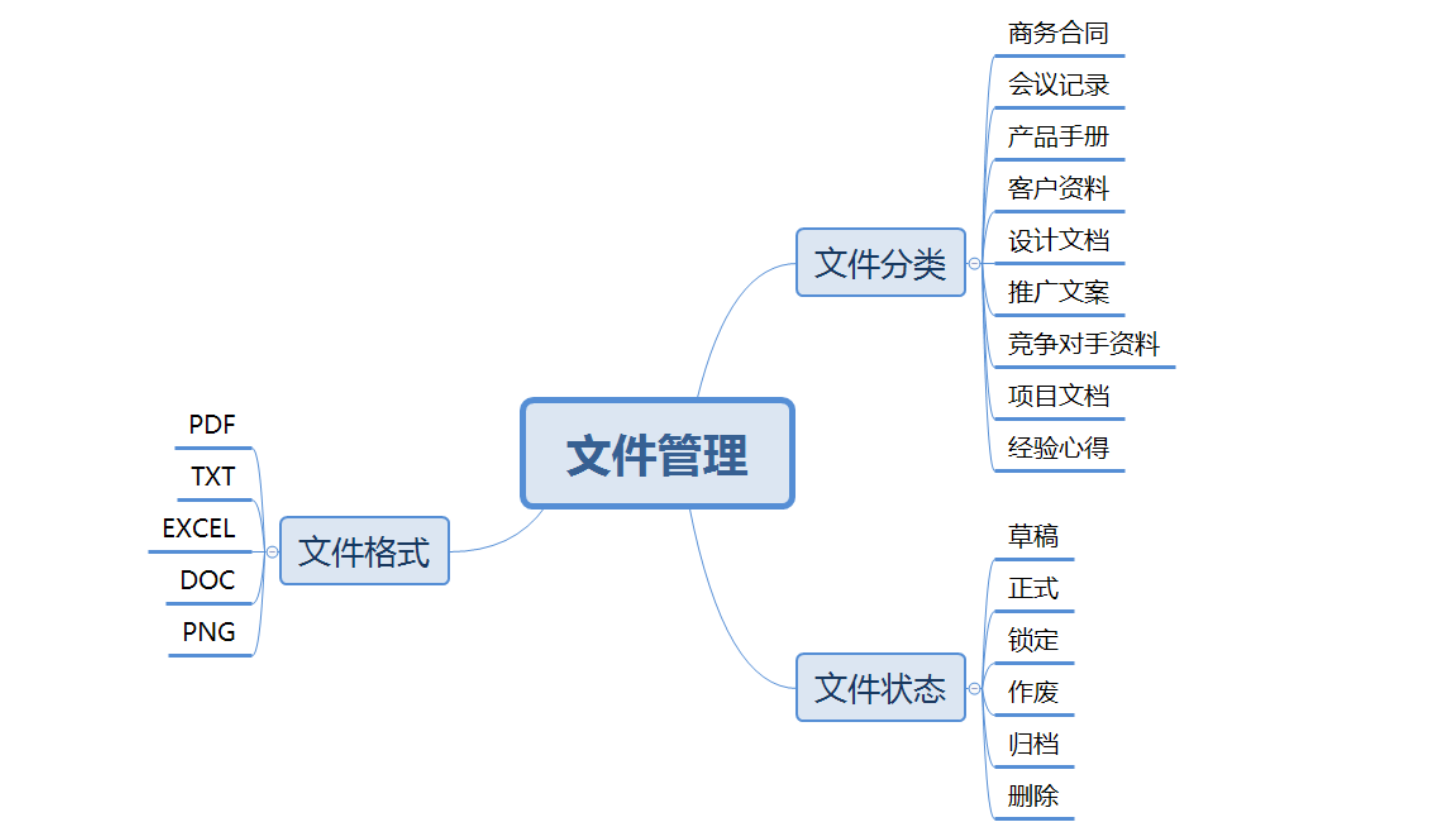


### 企业档案管理系统(document-service)

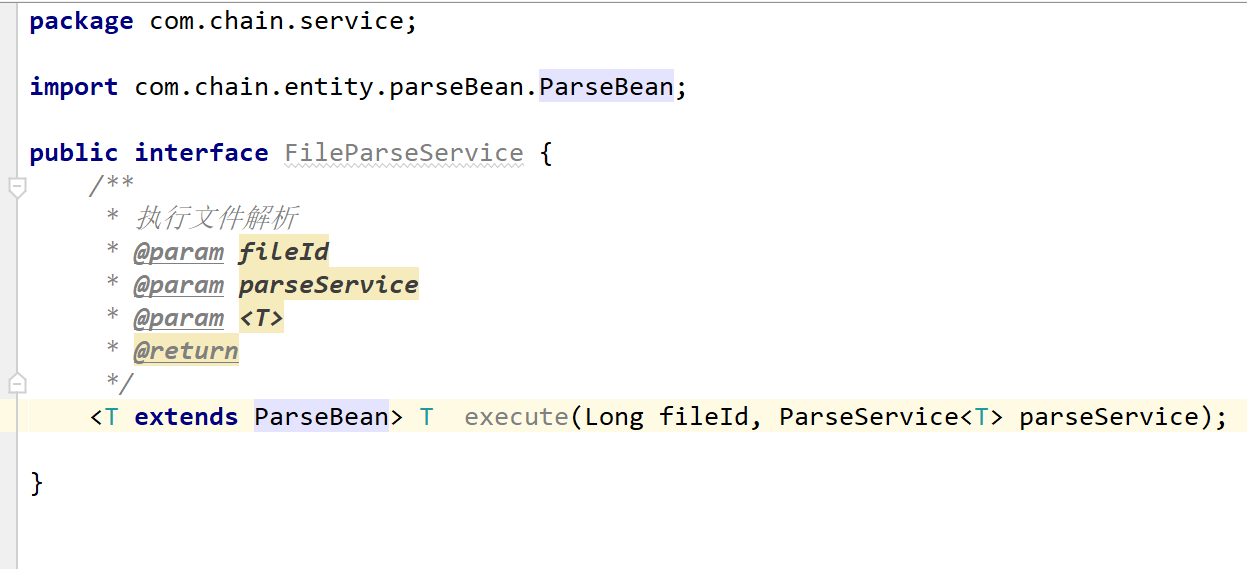
#### 何为企业档案管理

文件管理就是指文档、电子表格、图形和影象扫描文档的存储、分类和检索。文档管理的关键问题就是解决文档的存储、文档的安全管理、文档的查找、文档的在线查看、文档的协作编写及发布控制等问题。

##### 设计



##### 实现

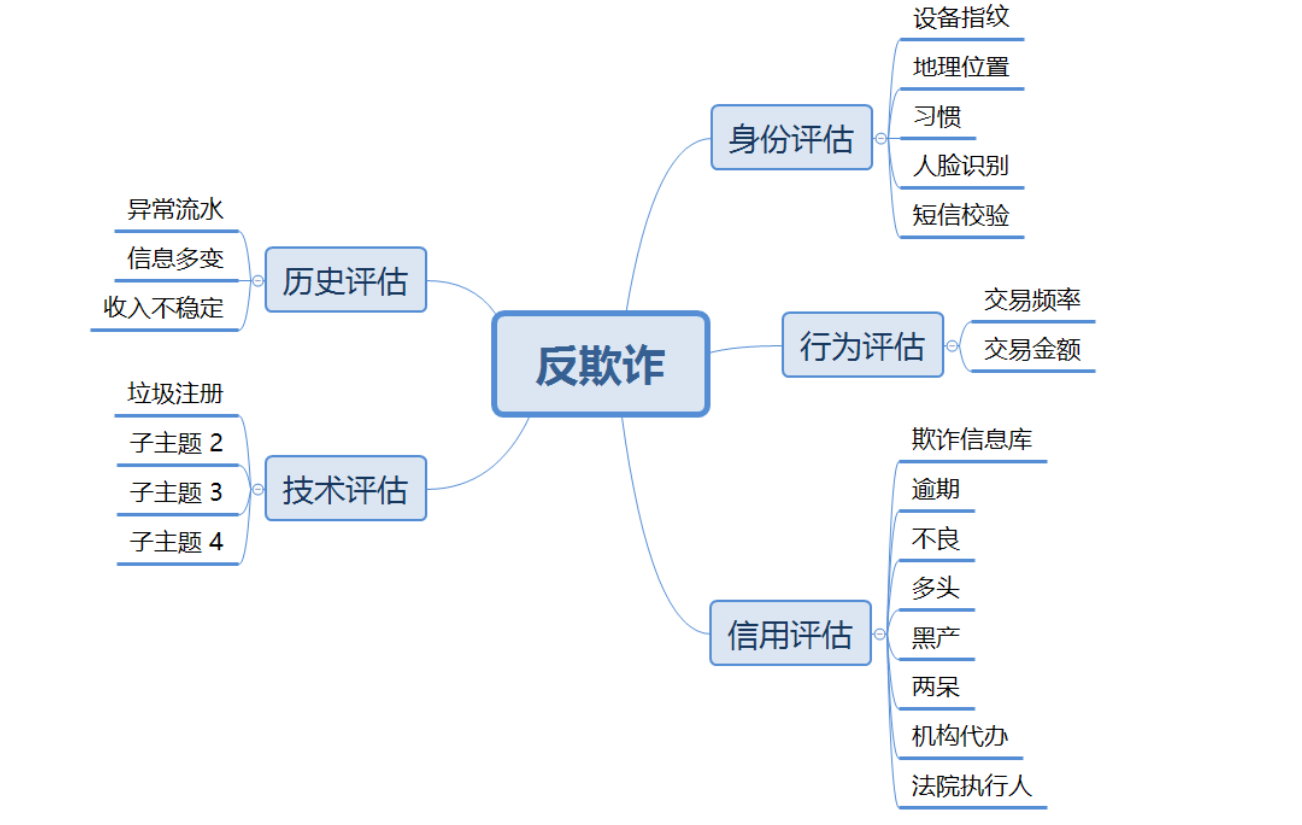


### 反欺诈系统(antifraud-service)

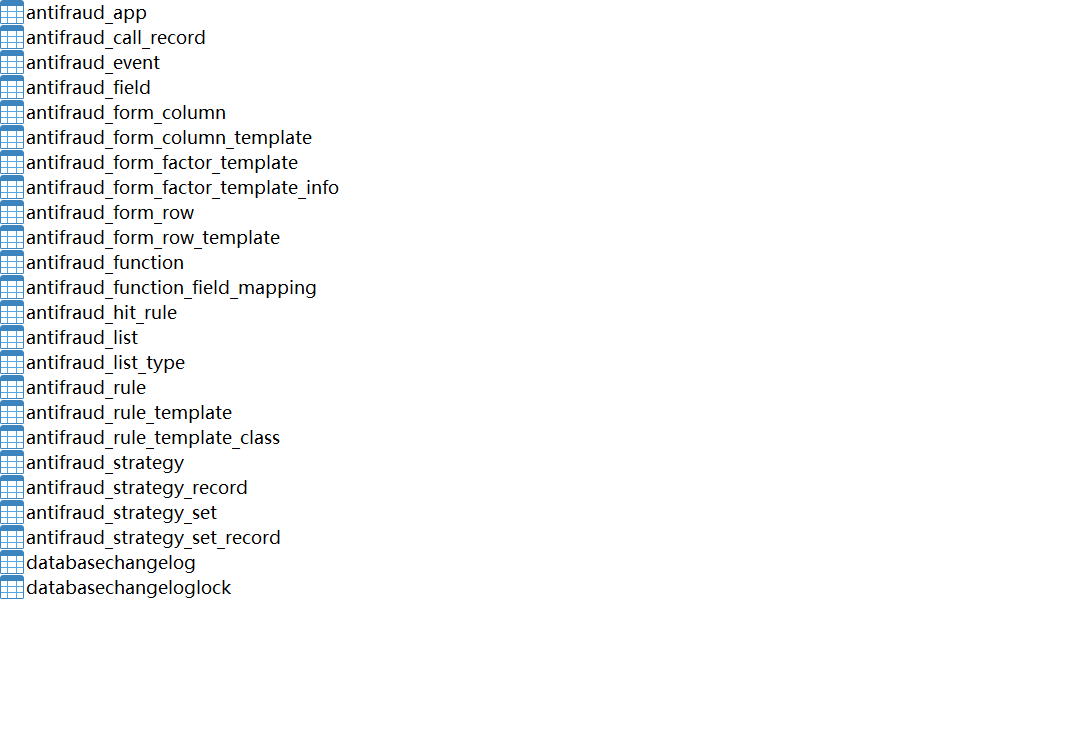
##### 背景

通过反欺诈模型、用户行为分析、风险信息库和反欺诈规则库判定客户的信用度。

##### 需求



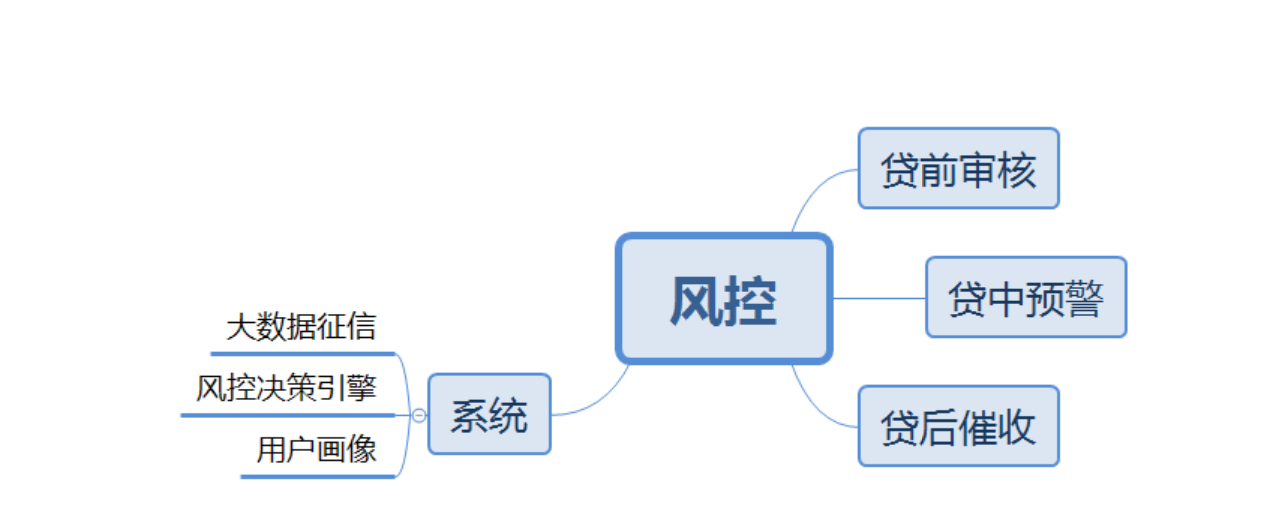
##### 实现



### 三方数据系统(data-service)

### 风控系统(risk-service)

##### 设计



### 账户管理系统(account-service)

#### 何为账户

账户体系设计首先要区分两个概念，支付账户和登录账号。 这是两个不同业务领域的概念。 **支付账户** 指用户在支付系统中用于交易的资金所有者权益的凭证。 **登录账号** 指用户在系统中的登录的凭证和个人信息。 一个用户可以有多个登录账户，一个登录账户可以有多个支付账户，比如零钱账户，储值卡账户等。 一般来说，支付账户不会在多个登录账户之间共用。如果没有特殊说明，下文中的账户，都默认指支付账户。

#### 账户的需求设计

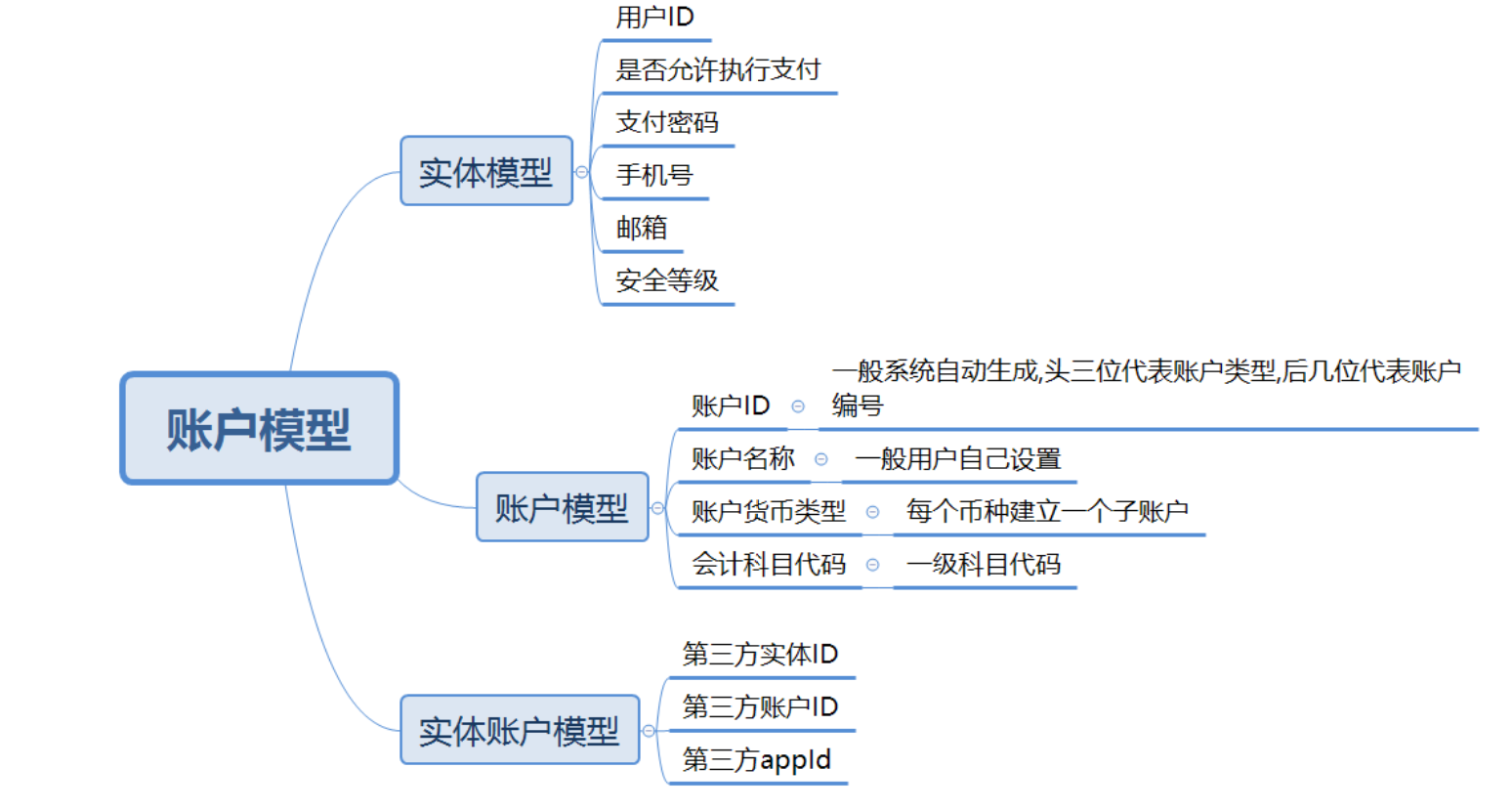


首先账户分为用户账户和业务账户。业务账户初始值为0，但是可以不断透支。一个用户完成某个动作，产生的一条从对应的业务账户到该用户对应账户的交易记录。至于具体是哪个账户，可以按照需求增加，比如积分账户、红包账户、人民币账户、提现账户等等。同理，交易记录也可分为红包、充值、转账、提现等等。

而交易记录根据是否需要审核，决定是直接生成流水记录，还是通过审核记录来生成流水记录。

#### 账户建模

账户建模本身包括三大核心模型：实体模型、账户模型和交易模型。 从交易模型中可以衍生出针对各个角色的账户流水，即明细模型，用于支持对账。



#### 交易设计

交易记录，交易流水，账户流水，交易台账，这三个容易混淆的概念，从数据上来说，却并不复杂，它们的核心是交易流水，账户流水是从账户视角的交易流水。那对一笔交易，涉及到的方方面面内容很多，有哪些需要记录的呢？考虑到交易记录将被用于风控和信用分析，能收集到的信息是越全面越好。



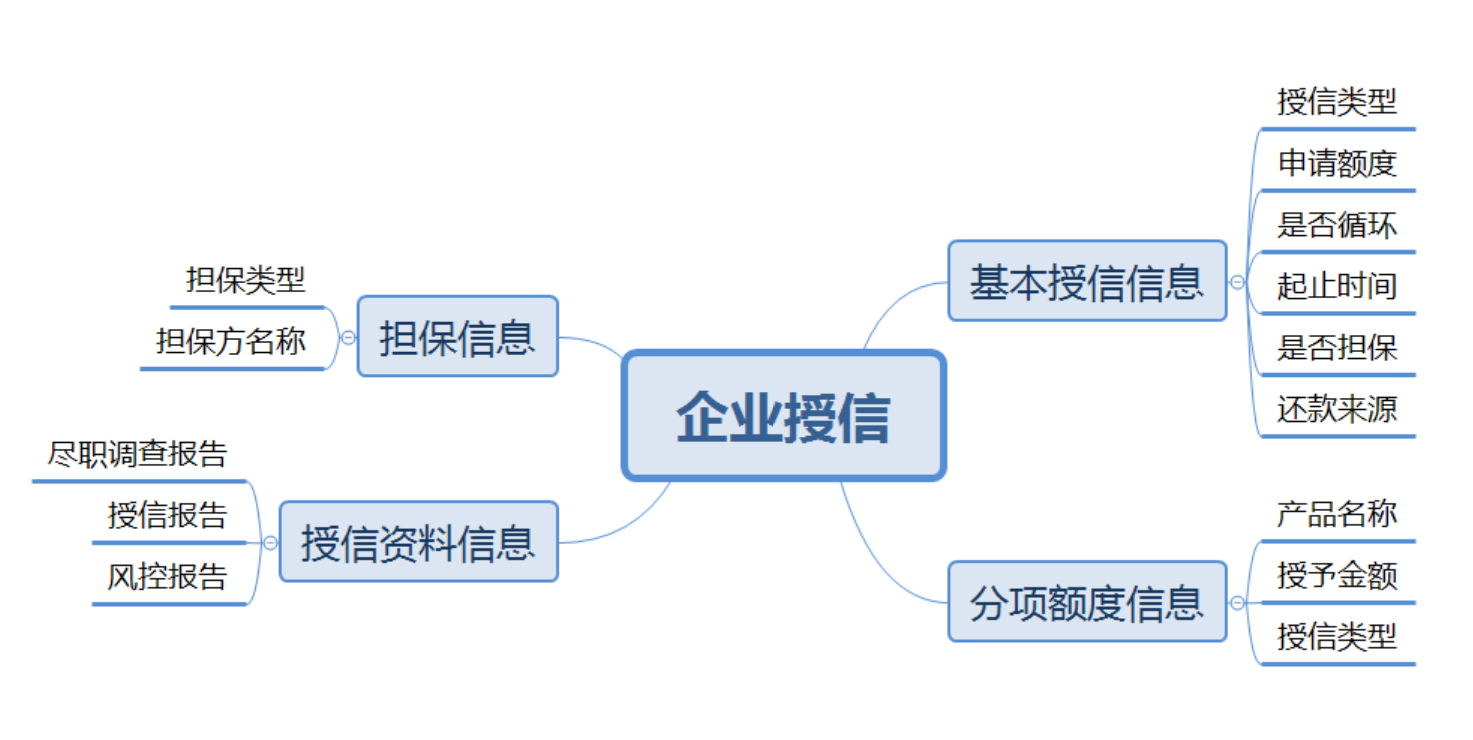
### 授信管理(credit-service)

#### 何为企业授信

简单来说，企业授信是指资金方(资金来源)向资产方(资金使用方)直接提供资金支持，授信不能等同于贷款，授信是一种风险控制的总的概念。当融资企业需要贷款时，贷款金额不能超过其企业的授信金额。

#### 授信内容概述

企业授信涉及到单个企业授信和集团企业授信。



#### 授信内容详解

##### 基本授信信息

**额度类型**:

新贷:企业第一次办理授信

增贷:本年内业务变更需要提额

年审:每年重新发起一次审核

变更:条件变更需要修改额度信息

**申请的额度**:

本次授信的金额(一般根据尽职调查报告等授信资料获取额度)

**是否循环**:

额度是否可以循环使用

循环授信与非循环授信的区别：

**循环授信：**授信500W，用信200W后，剩余额度为300W，当核心企业还款后，剩余额度变为500W。

**非循环授信：**授信500W，用信200W后，剩余额度为300W，当核心企业还款后，剩余额度为300W。

**额度的起止时间**:额度生效时间与额度到期时间

**是否担保**:授信过程中可以不提供担保企业或者担保人

**还款来源**:描述信息

##### 分项额度信息

分项额度指资金方针对每一项基础产品来对企业授信，一般情况下，资金方会有多种产品组合。

**产品名称**:基础产品

**授予金额**:该项产品的授予金额

**额度类型**:额度共享和额度独立

**额度共享：**指不同的产品授予金额可以小于等于企业的申请额度。

**额度独立：**指所有的产品授予总金额不能大于企业的申请额度。

企业申请额度为100W，额度共享指所有产品都可以是100W.额度独立指的是一个产品如果是50W，那么其他额度共享的产品只能是50W,了。相当于从100W中单独分配了50W给这个额度独立的产品

##### 担保信息

若申请信息有担保，则担保信息必填；若申请信息无担保，则担保信息可不填。

**担保类型**：企业担保/自然人担保

**担保方名称**:企业名称/法人名称

##### 授信资料

授信资料直接影响了整个业务环节的授信情况。

#### 授信功能设计

### 金融产品系统(product-service)

### 商品管理系统(item-service)

### 订单管理系统(order-service)

### 工单系统(work-service)

### 账单系统(bill-service)

### 账务系统(transaction-service)

### 资金管理系统(fund-service)

### 渠道路由系统(route-service)

### 支付系统(payment-service)

### 开票系统(invoice-service)

### 费用系统(fee-service)

### 贷后系统(afterloan-service)

### 告警系统(alert-service)

### 会员管理系统(vip-service)

### 营销管理系统(marketing-service)

### 客户关系管理系统(CRM-service)

### 合同协议管理系统(contract-service)

### 用户画像服务(portrait-service)

## 系统改造

## 进阶