行业中台设计文档

战法引擎

2019年06月12日

修订、签收记录

变更记录

|  |  |  |  |
| --- | --- | --- | --- |
| 日期 | 作者 | 版本 | 变更参考 |
| 2019-06-12 | 陈远超 | v.01 | 初始化版本 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

评审记录

|  |  |  |  |
| --- | --- | --- | --- |
| 日期 | 评审人 | 版本 | 评审/审批备注 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

目录

[1 概述 4](#_Toc11915993)

[1.1 目的 4](#_Toc11915994)

[1.2 范围 4](#_Toc11915995)

[1.3 提供方式 4](#_Toc11915996)

[2 功能规格 4](#_Toc11915997)

[3 概要设计 5](#_Toc11915998)

[3.1 总体架构 5](#_Toc11915999)

[3.1.1 上下文定义 5](#_Toc11916000)

[3.1.2 设计思路 5](#_Toc11916001)

[3.1.3 设计可选方案 6](#_Toc11916002)

[3.1.4 设计原则与约束 6](#_Toc11916003)

[3.2 逻辑设计 6](#_Toc11916004)

[3.2.1 领域模型 6](#_Toc11916005)

[3.2.2 模块设计 7](#_Toc11916006)

[3.2.3 逻辑架构 7](#_Toc11916007)

[3.2.4 业务流程 7](#_Toc11916008)

[3.3 数据库设计 9](#_Toc11916009)

[3.3.1 数据定义 9](#_Toc11916010)

[3.3.2 实体关系图（E-R图） 9](#_Toc11916011)

[3.4 接口设计 9](#_Toc11916012)

[3.5 原型设计 10](#_Toc11916013)

[4 非功能性需求 10](#_Toc11916014)

[4.1 设计目标 10](#_Toc11916015)

[4.2 设计方案 10](#_Toc11916016)

[4.3 集成需求 11](#_Toc11916017)

[5 假设与依赖 11](#_Toc11916018)

[图表 1：模块分解示意 5](#_Toc10712291)

[图表 2：逻辑架构示意 5](#_Toc10712292)

[图表 3：涉驾查稽时序图示例 6](#_Toc10712293)

[图表 4：E-R图示意 7](#_Toc10712294)

[表格 1：技术选型列表 4](#_Toc10712286)

[表格 2：设计目标示例 10](#_Toc10712287)

1. 概述

复杂的问题会在任何领域都出现，但是解决它们的总体策略通常是一样的：分而治之。我们会将问题拆分为更容易解决的子问题，然后这些方案再按照与分解相反的方式组合在一起形成整体的解决方案。通过观察会发现这样的问题是经常发生的；借助于经验，能够识别出最优的方案。这种集成模式同样适用于安防行业。

我们面临多行业多场景使用多战法协同工作以解决特性问题的业务需求，传统方式是各行业为固化的流程采用硬编码的方式串联所需的技战法，这并不利于提升产品开发效率和客户实战价值，战法引擎赋予的战法管理、战法路由和数据驱动会契合这个问题的解决方案。

* 1. 目的

战法引擎的设计目标是为前台提供平台化的底层支持，所谓平台化，就是对业务开发能以扩展，隔离的方式推进，驱动业务快速支持，根据业务需要灵活配置技战法，快速响应行业前台业务需要，加速行业产品研发效能，优化业务流程，提高用户业务效率。战法引擎达到以下设计目标：

1. 业务和引擎分离：战法引擎将不关心具体的业务，只通过抽象统一的模型去完成业务逻辑的串联和数据驱动。
2. 变化和实现分离：在复杂前台系统中，业务变动需求频繁，因此我们需要把业务变化通过战法引擎管理起来，实现变化与实现分离，通过引擎的规则配置来快速响应需求变化而不是硬编码实现，从而提高业务服务能力和系统稳定性。
   1. 范围

战法引擎适用于相对固化的、自动化的业务流程串联，不适用于需要人工干预较多的场景。

* 1. 提供方式

战法引擎以独立运行的平台（工具）提供。

1. 功能规格

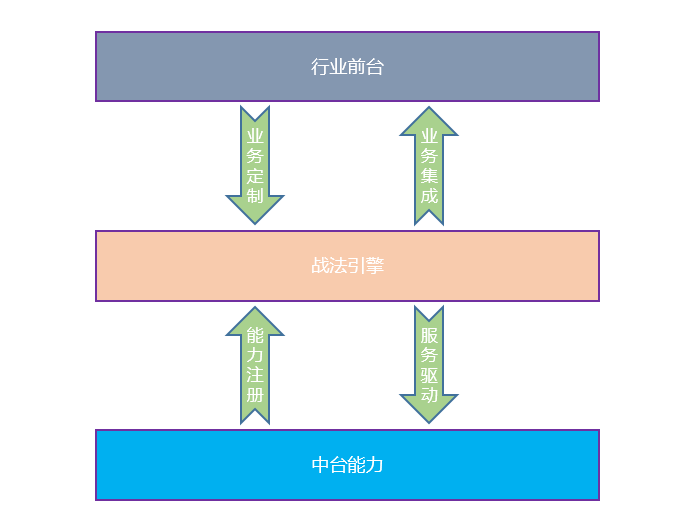
描述该服务/组件的功能规格，要能与用户故事相对应。Todo: 把需求规格补充进来

1. 概要设计
   1. 总体架构

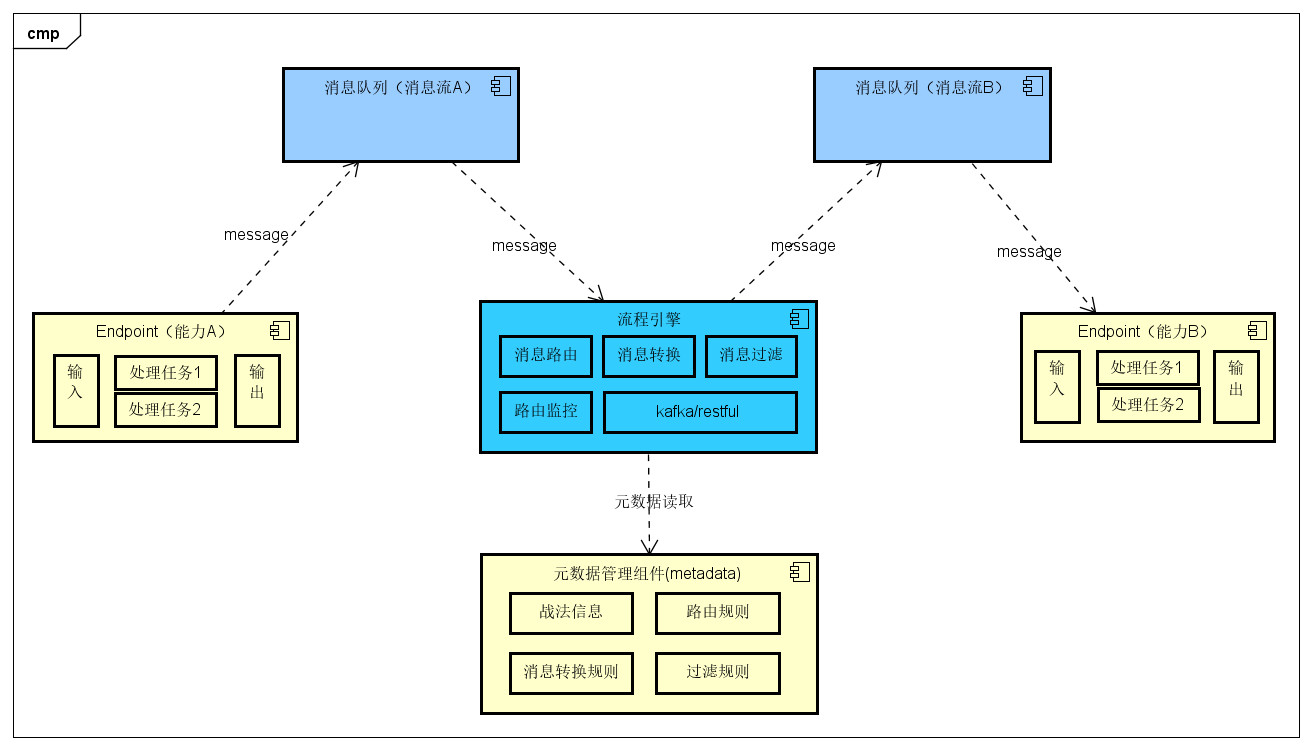


图表 1：战法引擎总体架构

* + 1. 上下文定义



图表 2：战法引擎上下文定义



战法引擎主要用于各个独立的中台战法服务的快速集成，串联出一个完整的业务流程。战法引擎依托于两个以上的可运行的战法服务，并依赖战法商店所提供的各类战法的元数据定义。战法引擎为行业前台的赋能包括以下两点：1、能够以配置的方式串联各个战法，覆盖部分应用场景。2、解耦各个战法服务的数据传输和接口调用。

* + 1. 设计思路

战法引擎设计思路借鉴了流程引擎（BPM）和企业服务总线（ESB），这使得战法引擎在设计上可以支持快速配置、串联和驱动技战法的目的，但它也在一定程度上违背了中台/微服务去IOE原则。

|  |  |  |
| --- | --- | --- |
| 组件 | 选型 | 选择原因说明 |
| 路由 | Camel | 实现了EIP的大部分模式，提供了大量Component(组件)，允许用户定义灵活的路由规则 |
| 数据映射 | JOLT | json转json的工具 |
| 消息通道 | kafka | 高吞吐量的分布式发布订阅消息系统 |
| 调度 | Quartz | 是可靠的可拓展的调度框架 |
| 数据库 | PGSQL | 稳定且功能完善的开源数据库 |

表格 1：技术选型列表

* + 1. 设计可选方案

备选方案分析。

方案一：

选择Apache Camel作为核心组件，基于其已实现的消息路由、协议转换等功能进行战法引擎的开发。通过Quartz实现流程调度。通过JOLT进行完成json数据的映射，json schema做消息校验。

方案二：

选择一款成熟的ESB产品做二次开发，实现战法服务的集成和串联。

分析：

基于现有资源、需求以及技术栈的考虑，本产品选择方案一作为最终方案。

* + 1. 设计原则与约束

描述服务/组件设计过程的遵循标准、硬件限制、技术限制等。

战法引擎遵循前后端分离的设计原则，各个组件的拆分基于AKF拆分原则，组件间以及前后端通信基于Restful风格。战法引擎串联的战法服务消息传输依赖于kafka。

* 1. 逻辑设计

从逻辑视图描述服务/组件领域模型，模块分解，逻辑架构和流程。

战法引擎包含消息路由组件、消息过滤组件、消息映射组件。消息路由组件主要由Camel实现，配合kafka或者其他消息队列完成消息的转发。消息过滤组件完成消息转发过程中的数据过滤功能。消息映射组件完成需要传输的数据对象的属性映射，以实现战法服务之间的数据对接。提供数据的战法服务将数据推入kafka消息队列，战法引擎的消息路由组件消费该数据，并将数据通过消息过滤组件和消息映射组件的处理推入kafka，为需要该数据的战法服务提供数据。

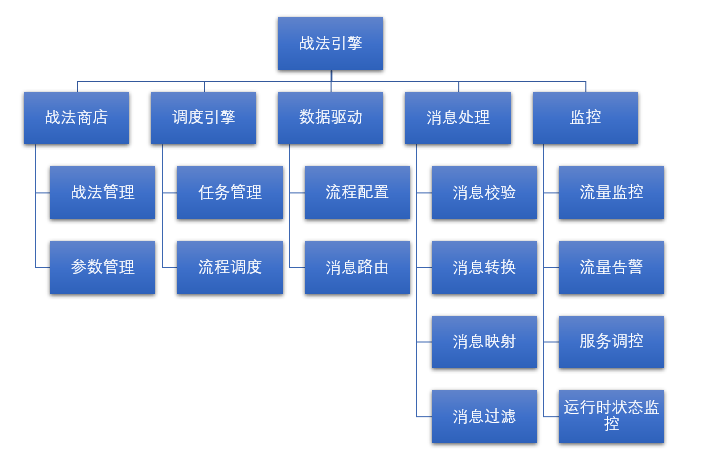
* + 1. 领域模型

说明系统的核心（顶级）业务流程和核心领域对象。

战法引擎的核心领域对象包括战法服务、战法功能、接口参数、过滤器、映射器等。其中战法服务功能接口属性相关的参数由战法商店提供。战法引擎根据战法的输入输出定义通过过滤器及映射器等组件完成战法的集成和串联。

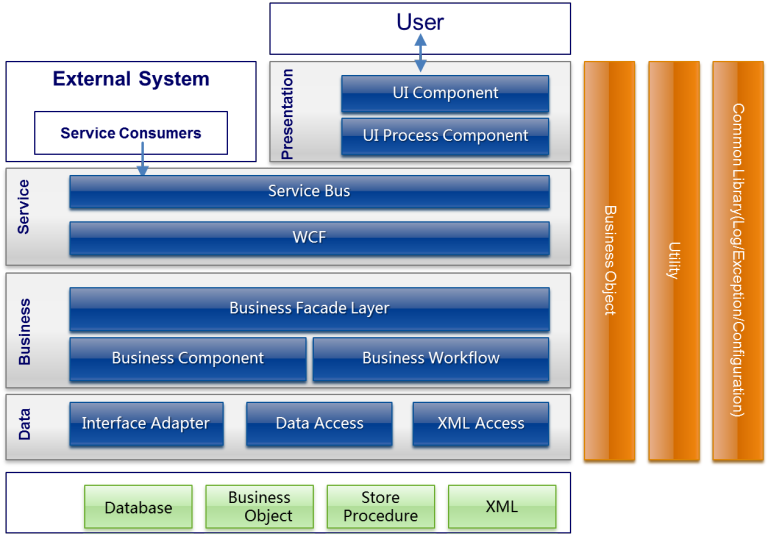
* + 1. 模块设计

说明模块分解设计。



图表 3：模块分解示意

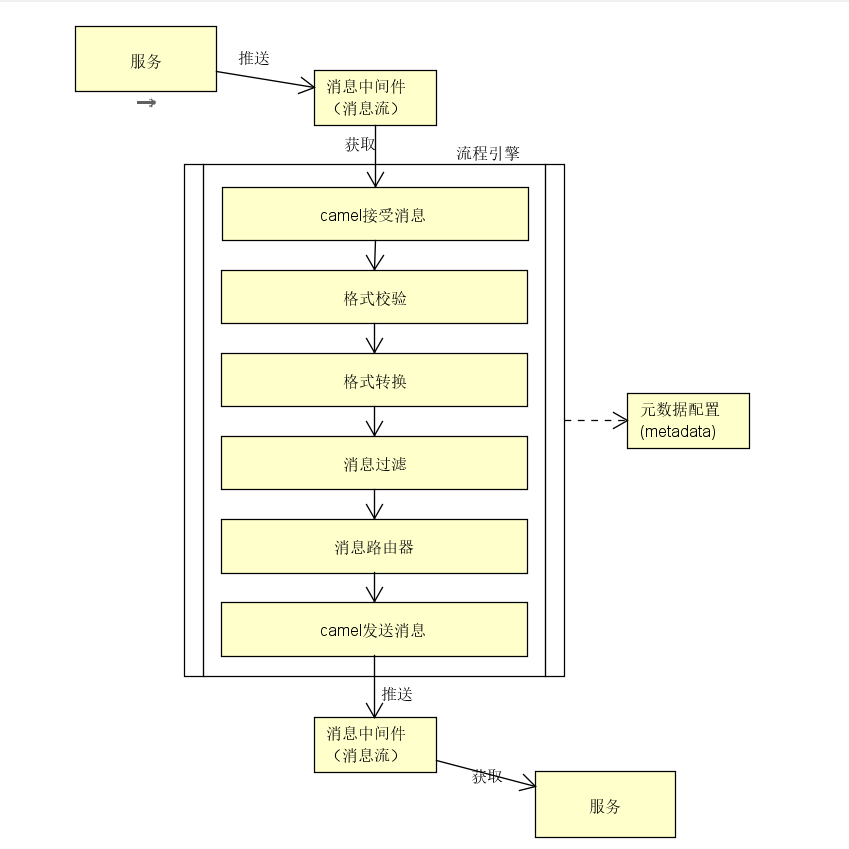
* + 1. 逻辑架构



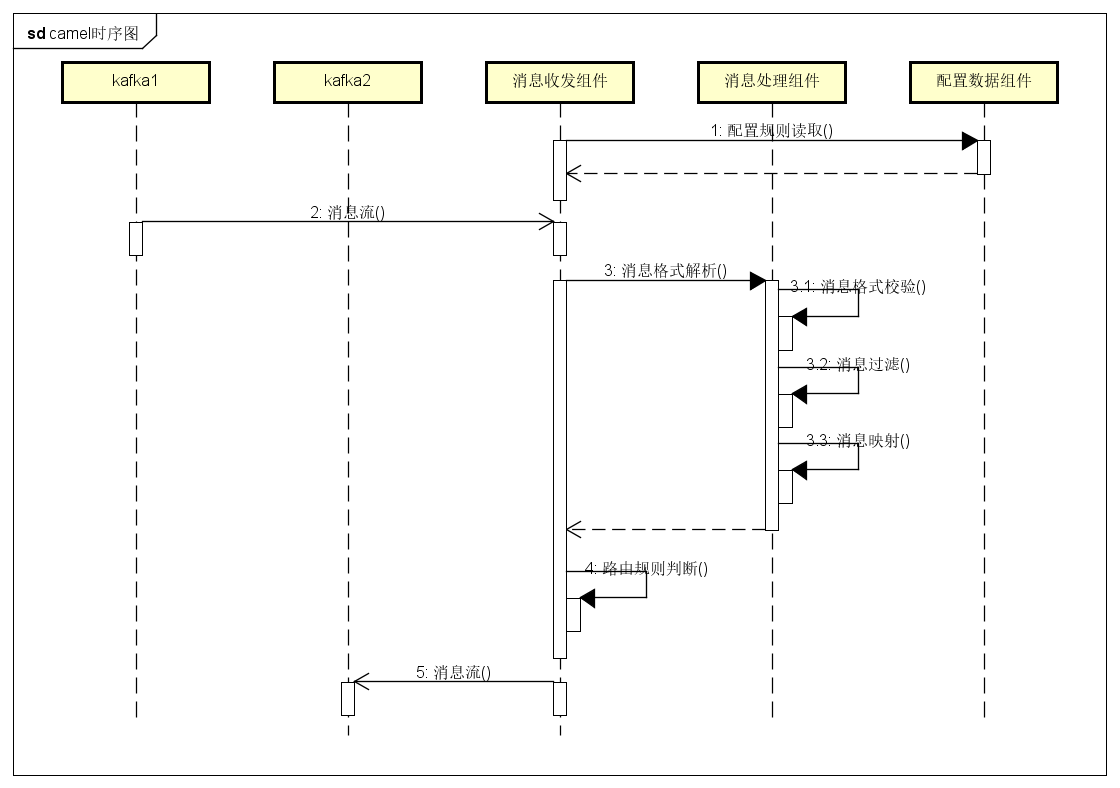
图表 4：逻辑架构示意

* + 1. 业务流程

流程引擎流程图：

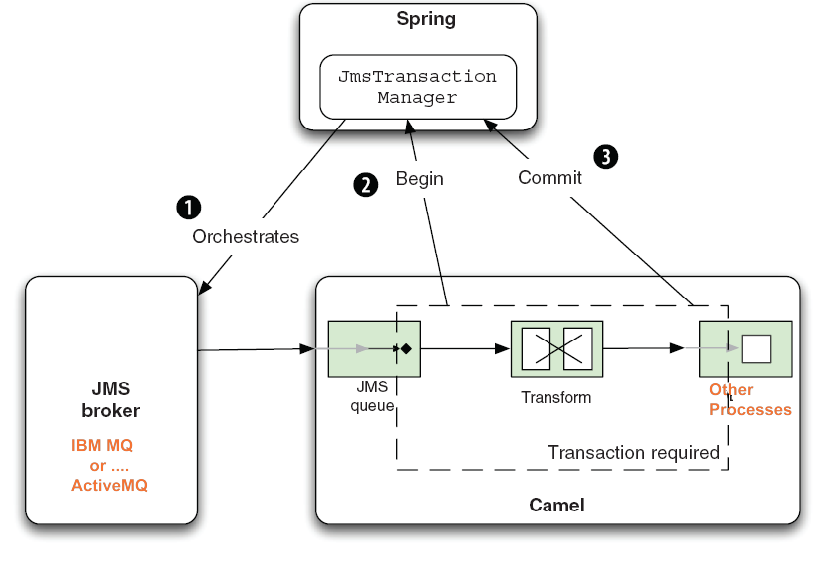


流程引擎时序图：



* + 1. 局部事务

整个route只有一个事务参与者,"局部事务",这里用JMS的例子,后台的MQ为ActiveMQ,示例图如下:



* + 1. 消息异常通知
  1. 数据库设计
     1. 数据定义

services

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 字段类型 | 字段长度 | 描述 |
| id | varchar | 36 | 主键 |
| name | varchar | 32 | 服务名称 |
| icon | varchar | 255 | 服务图标 |
| type | varchar | 32 | 服务类型 |
| description | varchar | 255 | 流程描述 |
| create\_time | int | 8 | 创建时间 |
| update\_time | int | 8 | 更新时间 |

functions

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 字段类型 | 字段长度 | 描述 |
| id | varchar | 36 | 主键 |
| name | varchar | 32 | 功能名称 |
| description | varchar | 255 | 功能描述 |
| service\_id | varchar | 36 | 服务编号 |
| create\_time | int | 8 | 创建时间 |
| update\_time | int | 8 | 更新时间 |

process\_nodes

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 字段类型 | 字段长度 | 描述 |
| id | varchar | 36 | 主键 |
| process\_id | varchar | 36 | 流程编号 |
| create\_time | int | 8 | 创建时间 |
| update\_time | int | 8 | 更新时间 |
| fun\_id | varchar | 36 | 功能编号 |
| name | varchar | 32 | 节点名称 |
| position | varchar | 255 | 节点位置 |
| type | varchar | 32 | 节点类型 |

line

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 字段类型 | 字段长度 | 描述 |
| id | varchar | 36 | 主键 |
| status | varchar | 32 | 连线状态 |
| source\_node\_id | varchar | 36 | 起始节点编号 |
| target\_node\_id | varchar | 36 | 目标节点编号 |
| Start\_param\_id | varchar | 36 | 起始参数编号 |
| End\_param\_id | varchar | 36 | 截止参数编号 |
| Create\_time | int | 8 | 创建时间 |

process

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 字段类型 | 字段长度 | 描述 |
| id | varchar | 36 | 主键 |
| name | varchar | 32 | 节点名称 |
| status | varchar | 16 | 流程状态 |
| description | varchar | 255 | 流程描述 |
| create\_time | int | 8 | 创建时间 |
| update\_time | int | 8 | 更新时间 |
| creator | varchar | 32 | 流程创建者 |

task

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 字段类型 | 字段长度 | 描述 |
| id | varchar | 36 | 主键 |
| status | varchar | 32 | 任务状态 |
| process\_id | varchar | 36 | 流程编号 |
| create\_time | int | 8 | 创建时间 |
| update\_time | int | 8 | 更新时间 |

fun\_params

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 字段类型 | 字段长度 | 描述 |
| id | varchar | 36 | 主键 |
| param\_type | varchar | 32 | 参数类型 |
| fun\_id | varchar | 36 | 功能编号 |
| param\_id | varchar | 36 | 参数编号 |
| create\_time | int | 8 | 创建时间 |
| update\_time | int | 8 | 更新时间 |

parameters

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 字段类型 | 字段长度 | 描述 |
| id | varchar | 36 | 主键 |
| Json\_schema | text | NA | Json参数描述 |
| create\_time | int | 8 | 创建时间 |
| update\_time | int | 8 | 更新时间 |

paramter\_items

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 字段类型 | 字段长度 | 描述 |
| id | varchar | 36 | 主键 |
| field\_type | varchar | 32 | 字段类型 |
| name | varchar | 255 | 字段名称 |
| description | varchar | 255 | 字段描述 |
| parent | varchar | 255 | 父级字段编号 |
| param\_path | varchar | 255 | 字段路径 |
| default\_value | varchar | 255 | 默认值 |
| param\_id | varchar | 36 | 参数标号 |
| create\_time | int | 8 | 创建时间 |

protocols

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 字段类型 | 字段长度 | 描述 |
| id | varchar | 36 | 主键 |
| key | varchar | 255 | 协议key |
| value | varchar | 255 | 协议值 |
| fun\_param\_id | varchar | 36 | 功能参数编号 |
| create\_time | int | 8 | 创建时间 |
| update\_time | int | 8 | 更新时间 |

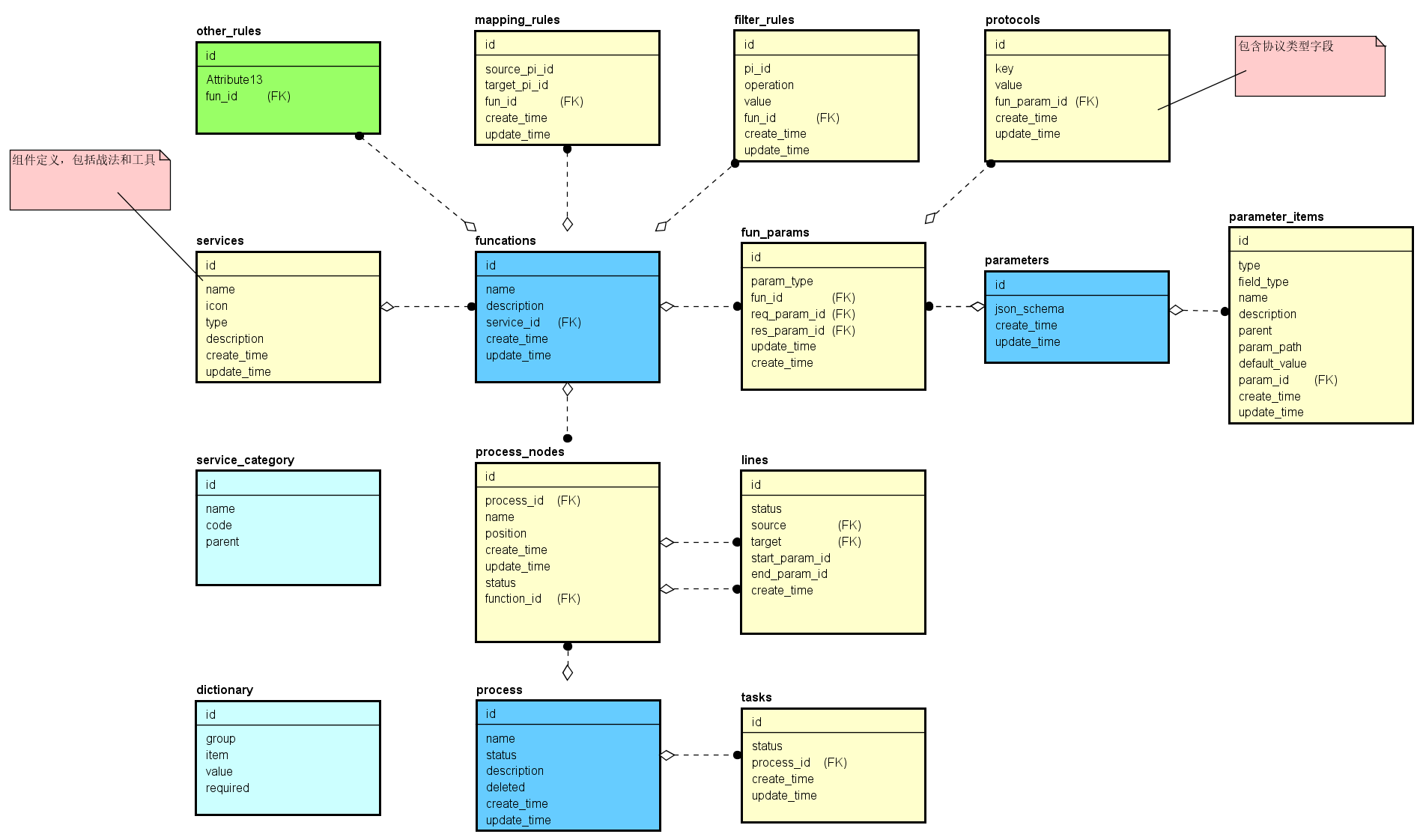
filter\_rules

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 字段类型 | 字段长度 | 描述 |
| id | varchar | 36 | 主键 |
| field\_id | varchar | 32 | 字段编号 |
| operation | varchar | 16 | 过滤规则操作 |
| value | varchar | 255 | 对比值 |
| fun\_id | varchar | 36 | 功能编号 |
| create\_time | int | 8 | 创建时间 |
| update\_time | int | 8 | 更新时间 |

mapping\_rules

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 字段类型 | 字段长度 | 描述 |
| id | varchar | 36 | 主键 |
| source\_field\_id | varchar | 36 | 来源字段编号 |
| target\_field\_id | varchar | 36 | 目标字段编号 |
| fun\_id | varchar | 36 | 功能编号 |
| create\_time | int | 8 | 创建时间 |
| update\_time | int | 8 | 更新时间 |

* + 1. 实体关系图（E-R图）



图表 5：战法引擎E-R图

* 1. 接口设计
  2. 原型设计

提供服务/组件的原型设计效果图，例如多维轨迹展示形式可以是2D或3D切换可选。此处原型设计稿和高保真设计稿都可以接受。

1. 非功能性需求

详细非功能性需求可参考《华智应用非功能需求规格表0321.xlsx》

* 1. 设计目标

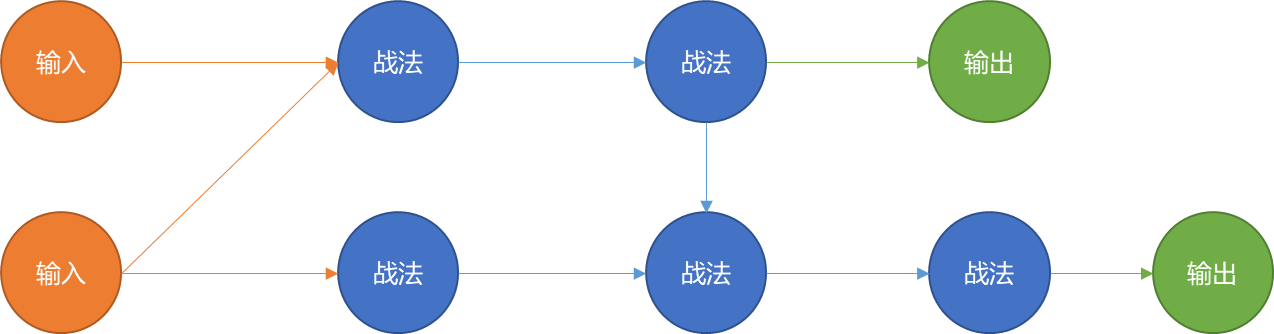
|  |  |
| --- | --- |
| 质量属性 | 要求 |
| 性能和容量 | • QPS/TPS指标  • 高峰并发用户数  • 系统响应时间要求  • 最大数据容量规格 |
| 安全 | • 使用环境（内外网），用户群体  • 安全合规要求  • 数据加密要求等 |
| 可扩展性 | • 高可用指标(例如99.9%） |
| 互操作 | • 需要支持的系统间集成协议  • 需要交互的外部系统 |
| 兼容性 | • 需要支持的浏览器版本，  • 需要兼容的特殊软件版本 |
| 数据保留与归档 | • 数据线上保留时间，业务量等  • 归档策略 |
| 备份与恢复 | • 数据如何备份恢复 |
| 可靠性 | • 业务可靠性  • 数据可靠性  • 传输可靠性等 |

表格 2：设计目标示例

* 1. 设计方案

针对如何达到上述目标的设计策略和方案，可分为多个章节。

* 1. 集成需求



图表 6：战法引擎输入输出示意图

战法引擎核心的意义在于各种技战法的串联和驱动，战法引擎的输出是一条完整业务流程的配置规则。业务系统集成战法引擎只需关注该完整业务流程的输入和输出。而

1. 集成输入：
   1. 如果该战法流程的输入需要用户界面进行，由于战法本身会提供UI界面（如果需要），因为业务系统只需通过多产品融合的方式访问启动战法的界面即可；
   2. 如果该战法流程的输入无需UI交互，业务系统集成过程则遵循传统的战法调用过程。
2. 集成输出：
   1. 如果该战法流程的输出具有UI界面，则通过多产品融合的方式直接由战法本身提供输出展现；
   2. 如果该战法流程的输出不需UI界面，业务系统集成过程则遵循传统的战法调用过程。
3. 假设与依赖
4. 战法引擎实现数据驱动依赖于消息队列，目前支持通过Kafka和Restful接口调用的方式来流转数据，因此对技战法有以下约束条件：

* 每个技战法的输入必须以Restful接口或者Kafka的方式；
* 每个技战法的输出必须输出到Kafka；

1. 战法引擎数据驱动过程中如果某一战法发生异常，整个流程链路对该异常的感知和事务回滚机制依赖对每个技战法有以下依赖：

* 每个技战法支持异常回调接口；
* 每个技战法提供事务回滚接口；