目录

[1 整体结构图 1](#_Toc477777200)

[V1 2](#_Toc477777201)

[Revision V2 2](#_Toc477777202)

[Revision V3 3](#_Toc477777203)

[Revision V4 补充 4](#_Toc477777204)

[Revision V5 4](#_Toc477777205)

[Revision V6 5](#_Toc477777206)

[Revision V7 6](#_Toc477777207)

[2 计划周期 6](#_Toc477777208)

[1 一期 6](#_Toc477777209)

[2 二期 7](#_Toc477777210)

[3 三期 7](#_Toc477777211)

[4 四期 7](#_Toc477777212)

[5 五期 7](#_Toc477777213)

[6 六期 7](#_Toc477777214)

[3 功能简述 7](#_Toc477777215)

[1 资产管理 7](#_Toc477777216)

[2 IAM 8](#_Toc477777217)

[3 信息收集监控 8](#_Toc477777218)

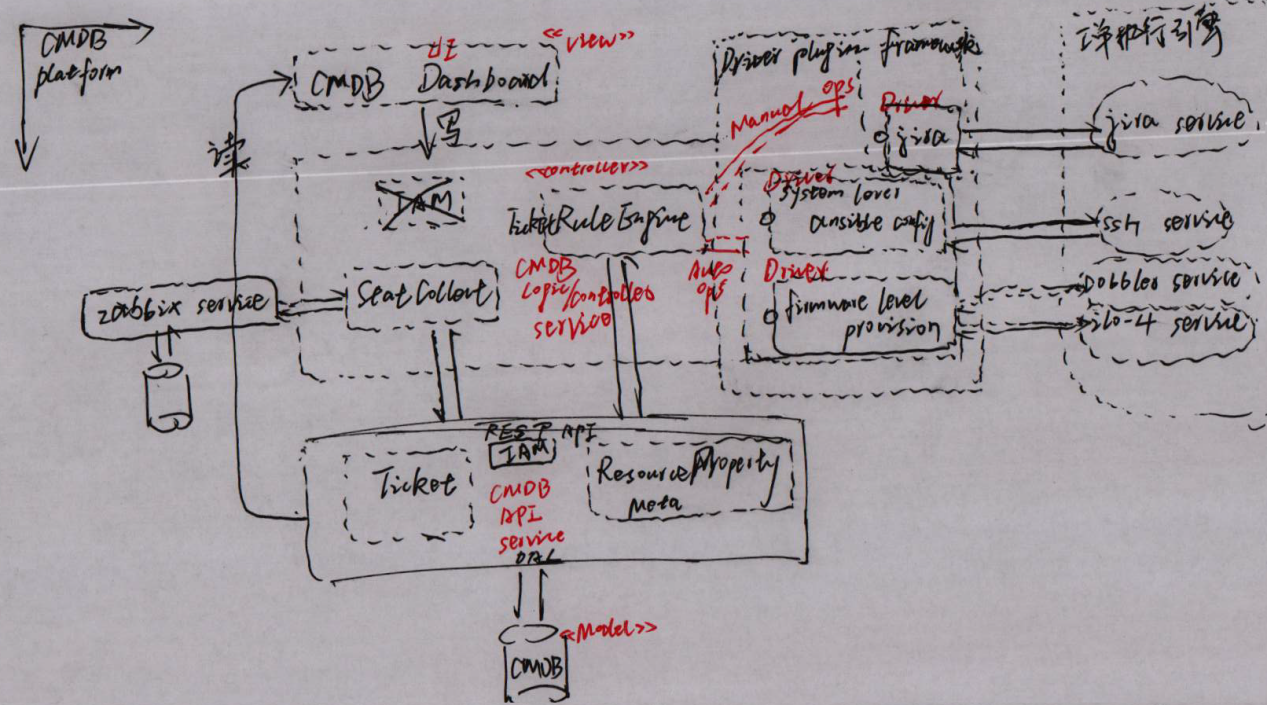
[4 工单管理 9](#_Toc477777219)

[5 UI 10](#_Toc477777220)

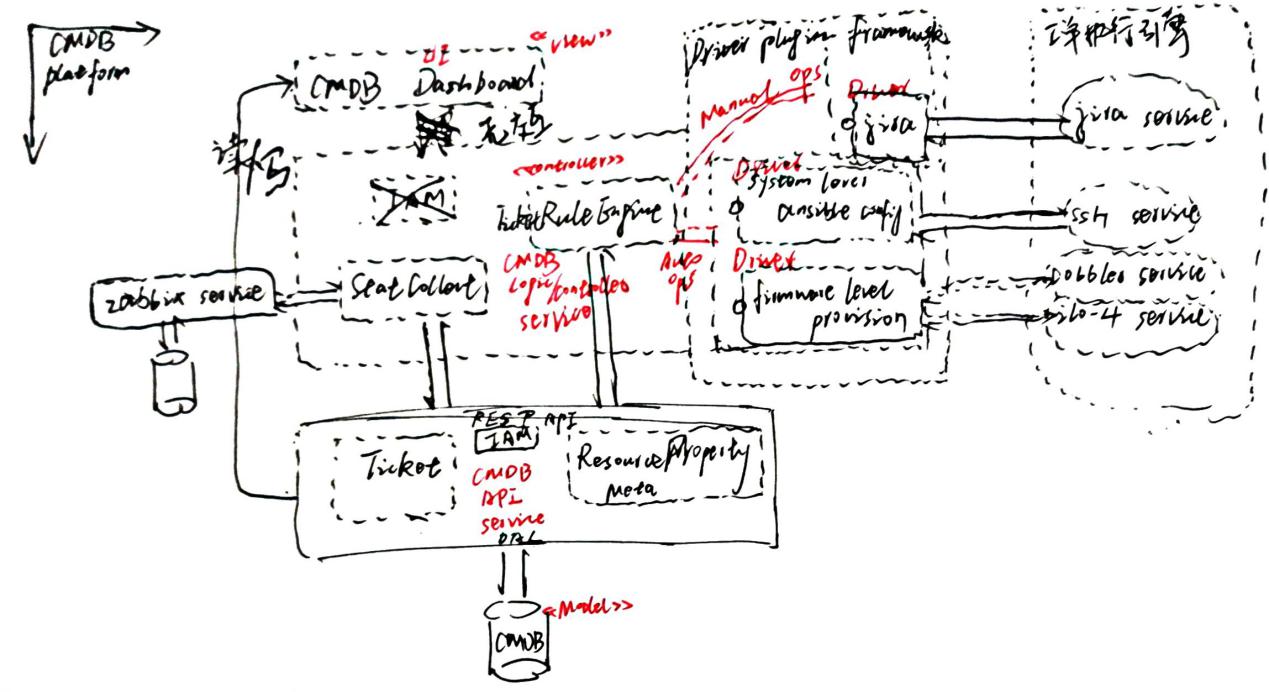
# 1 整体结构图

需求、用例、系统原型、功能流程拆分、接口及数据模型

## V1

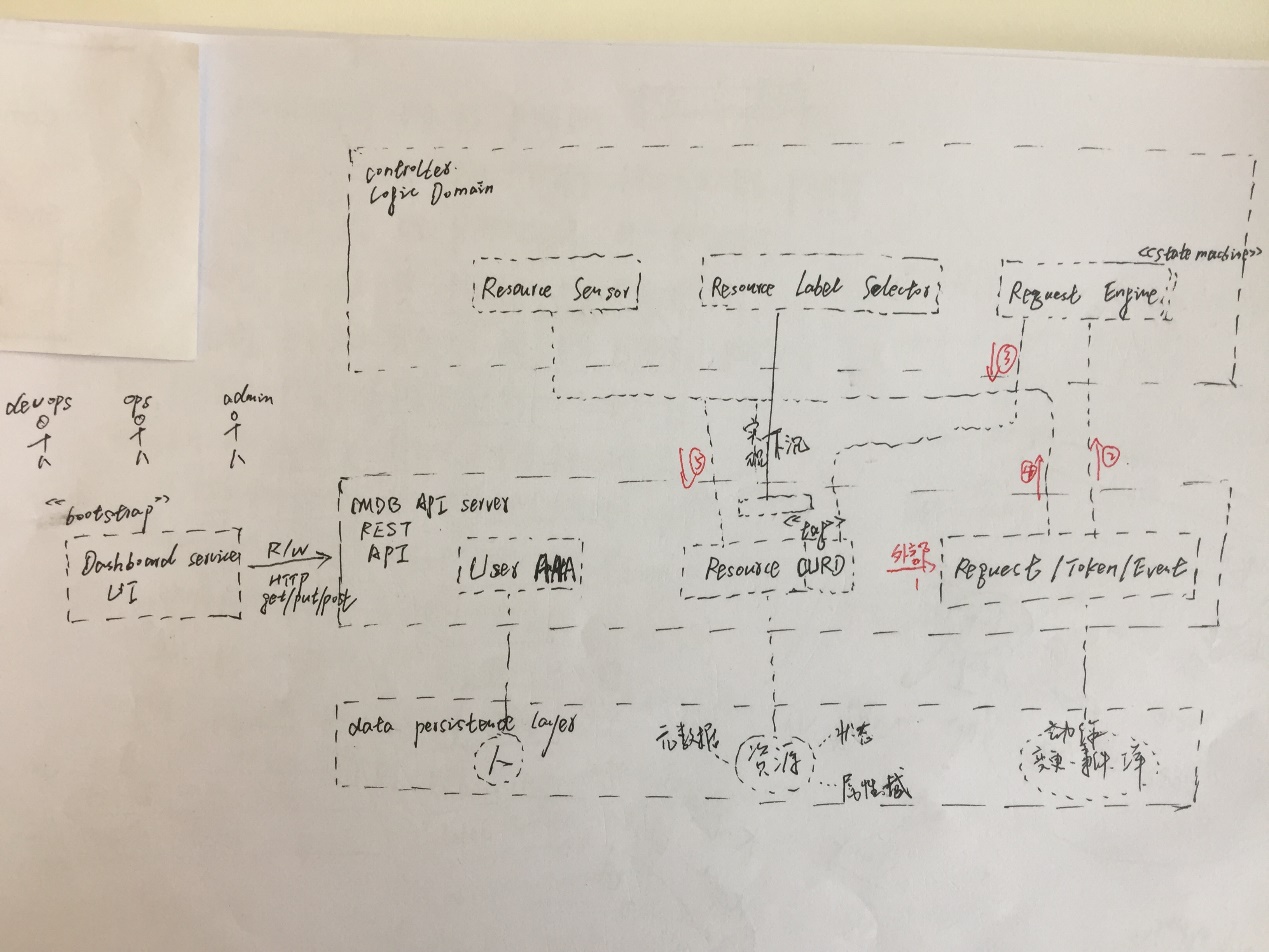


## Revision V2



* CMDB Dashboard UI 借助bootstrap框架及现有模板，实现html , CSS及 JS的特性融合
* CMDB Dashboard 用户注册登录，针对某种资源的任务创建、查看及任务历史等R/W操作均与CMDB api service 交互，针对资源本身可条件过滤、标签匹配、关键词搜索等
* CMDB logic controller仅与api service交互，不对外暴露。负责检索待执行的任务并执行，将任务状态的变更记录通过api service回写至数据库，方便后期归档查看。

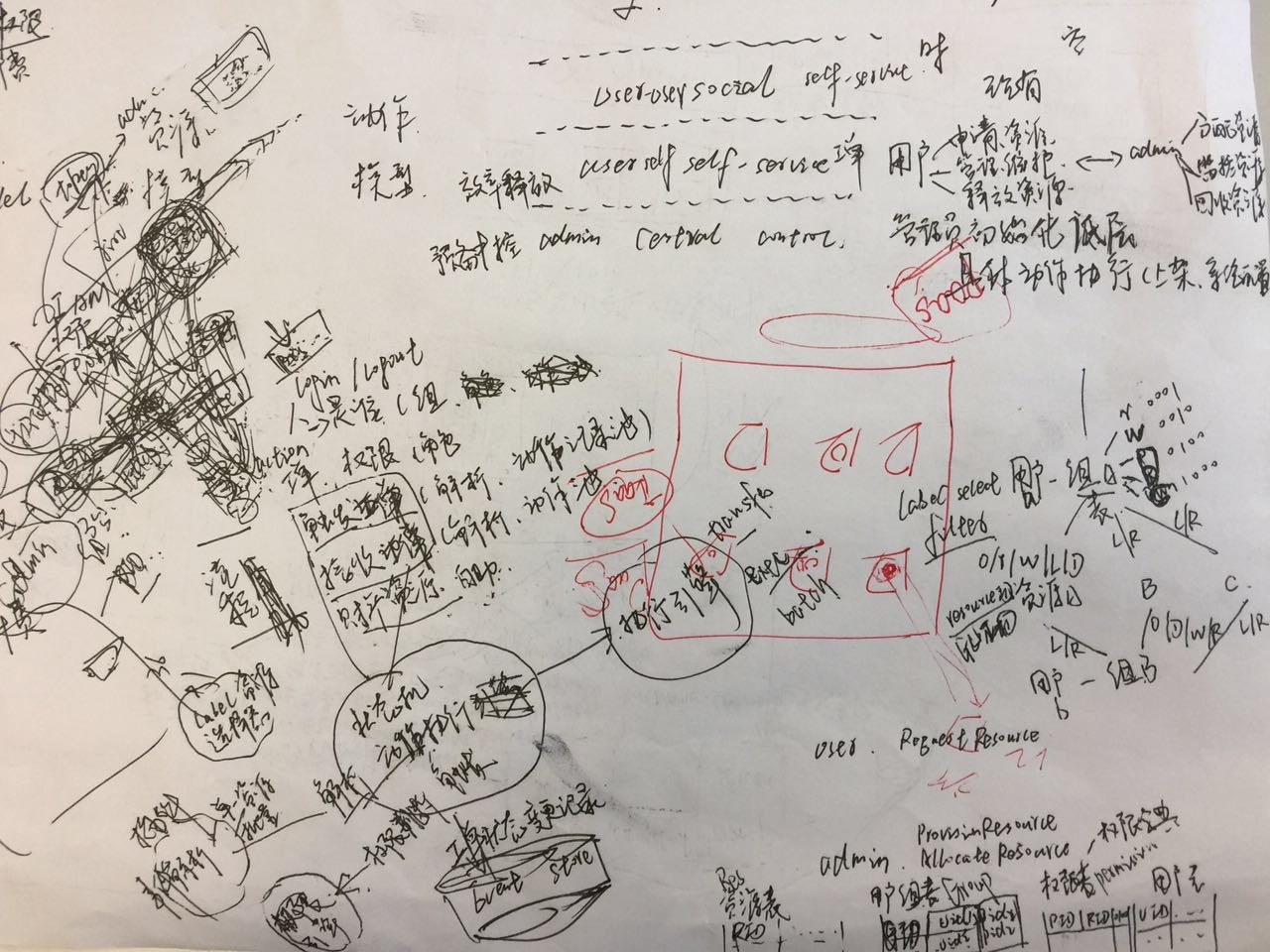
## Revision V3



架构演进：

* 将资源本身的CRUD，与针对资源发起的工单，以及资源变更引发的event 路径解耦分离 --- 正常流程是 用户发起ticket/token，由request engine解析动作，并丢给权限规则引擎判断，基于返回结果，由request engine 触发jira审批权限escalation或通知执行引擎即可执行。
* 引入针对具体资源的label selector，动态灵活的针对同质资源进行管控
* 数据模型垂直拆分为人、资源、动作三个维度。人可分组，可认证，可关联授权，可操纵资源；资源有公共属性，自有特定域的属性以及状态描述数据；资源变更相关的action，手动创建的工单ticket，status event自动产生，处理均归一至Request Engine。
  + 目前系统设计中token/ticket management全部依赖外部的out of box jira service ( ticket state machine by button transmission, ticket filtering by priority/creation time, pick a ticket from ticket queue, ticket escalation and change history, ticket trigger to notify ticket handling, ticket board for view and analyzing team ticket load, multi-channels support like jabber integration for email notification…)

## Revision V4 补充



* 架构调整，突出区分admin与用户用例，将工单系统与资源相关的处理执行引擎 （子module包括请求资源选择器、事件解析器、权限管理、及具体执行引擎）逻辑上拆分，实现上解耦，并再细化各子model的rest接口定义（目前未采用swagger )
* 用户登录认证及操作资源的权限管控, 依旧在api server中。
* 确立用户、组、资源、权限（权限字典）具体数据模型

## Revision V5

Provision layer 资源初始化 针对管理员

IT运维上架计算设备，（万信、邮件）通知cmdb平台admin

admin 通过api server登录平台，基于配置模板在平台资源配置界面填写远控ip及密码，然后提交资源初始化任务，

cmdb后台任务管理器controller监测任务库，并行执行具体工作，如安装系统、磁盘分区、配置基础服务如防火墙、ip、路由、hostname、Yum配置源

后台任务异步执行期，cmdb admin可通过任务界面查看任务状态，通过资源界面查看可用的计算资源。

self-service layer 资源自给自足 针对用户

用户通过api server注册并登录平台，可提交资源工单，如申请、释放、修改更新等。

cmdb后台任务管理器controller监测任务库，对用户提交的任务，并发执行基于任务类型对应的处理逻辑

草拟如下：

if (ticket\_opt\_type == 申请资源)

if (ticket\_status == assigned)

if ( ticket\_create\_time > ticket\_expired\_time )

修改 ticket\_status == escalation

if ( ticket\_status == approved) 查看工单 实现主逻辑path

修改资源ownership为uid；

修改资源状态为used;

若存在 对应的紧急通知任务，则修改任务状态为completed

if (ticket\_status == escalation)

查看对应 紧急通知任务 是否存在，不存在则创建

邮件或短信紧急通知平台admin 由于超时 工单优先级被提升

if (ticket\_status == completed )

将此已完成的任务 写入 completed\_job\_table 并从 pending\_job\_table中删除

else if (ticket\_opt\_type == notification)

if (ticket\_status == completed)

归档并删除

if (ticket\_status == started )

使用定时器，判断是否需要再次推送通知

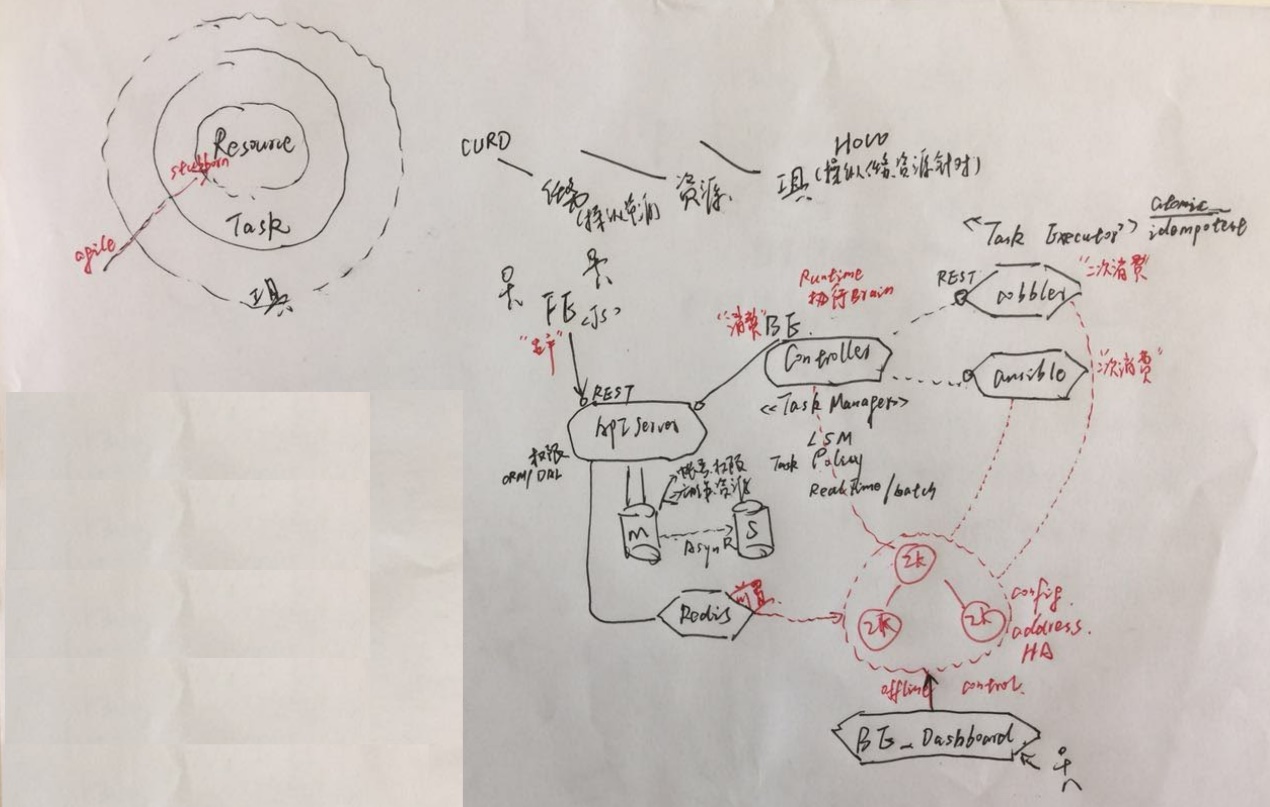
else if (ticket\_opt\_type == 修改、更新或释放 资源)

通过api\_server 判断 ticket\_owner 是否有权限 操作资源

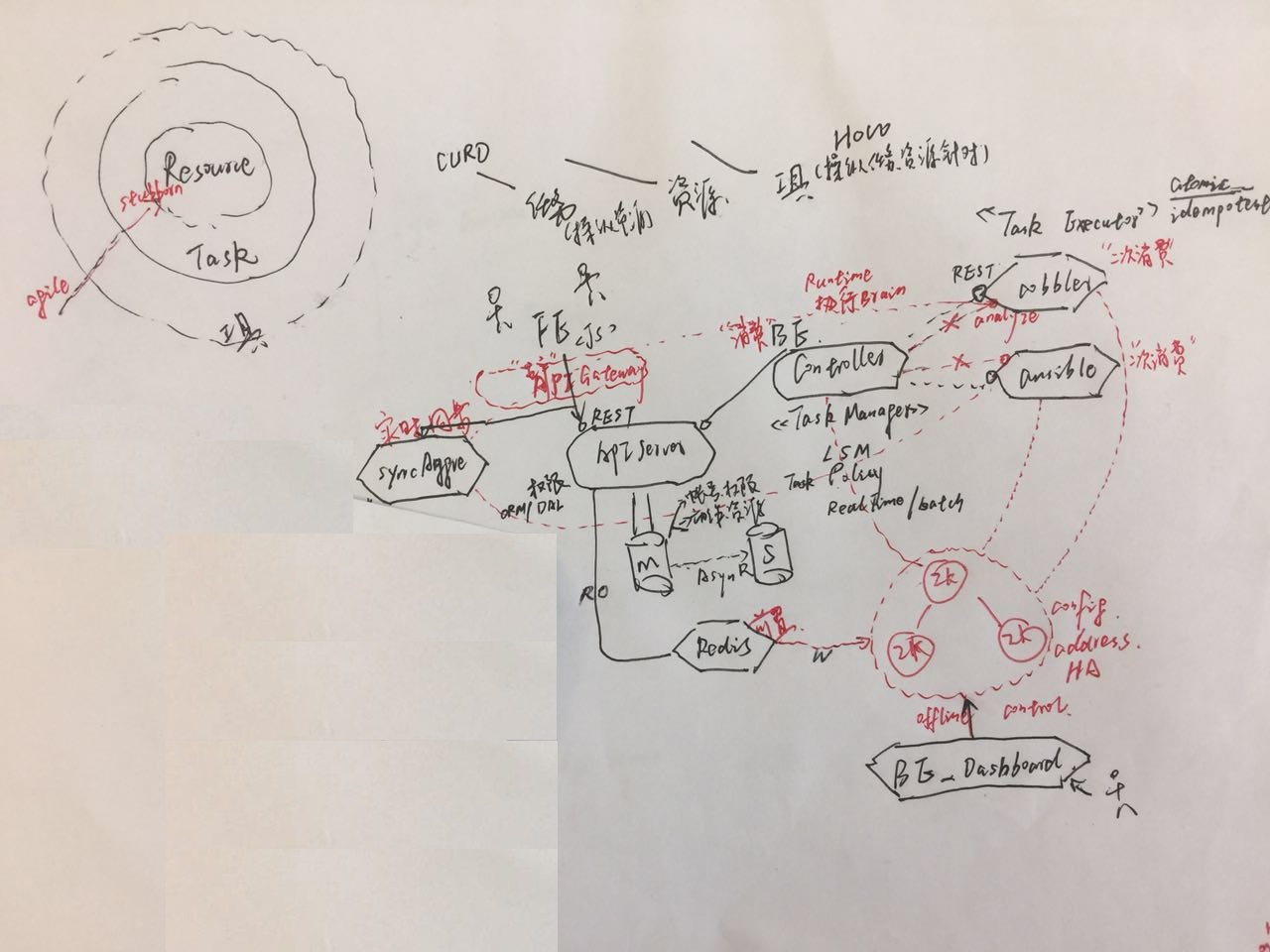
若acl true， 则自动执行 对应operation

若acl false, 则ticket\_status=no\_privilege，归档并删除

## Revision V6



## Revision V7



目前 预置给前端的变化较少的配置信息 是放入redis直接从api server读取，后续如系统layer的特定实例的资源状态信息，（如是否已配置指定的Yum源、具体是否已安装 软件包、及安装的软件包的版本号等，及其它后续需实时获取的当下时间点的监控信息、状态信息、 会拆分出一个syncProvider service（原因是：无法约束 用户必须通过CMDB平台来配置管理软件包，持久化特定实例的已配置的源及已安装的包信息 会导致不一致，若通过定时任务去抓取存储 有延迟 还是可能导致不一致） （实时同步的组合型服务，实现为细粒度的接口）与TaskController走的异步化流程式的任务处理 完全独立开。

# 2 计划周期

## 1 一期

1. Controller Server, API service setup 选型，建立数据模型
2. 硬件资产录入，变更，查询管理
3. 系统供给（针对录入服务器的资源）
4. Driver plugin framework setup
5. API service setup

## 2 二期

1. Stat collect实现与zabbix集成
2. Ansible config模块实现与Ansible api集成

## 3 三期

1、Ticket Rule Engine 模块实现（配置解析，规则执行）

2、IAM实现（用户，资源，权限，组管理）

## 4 四期

1. jira intergration
2. Driver plugin framework

## 5 五期

1. Ticket Rule Engine 与 driver plugin集成
2. Ticket Rule Engine 与 State Collection集成（控制闭环）

## 6 六期

1. Dash board UI设计
2. UI与API service集成联调
3. UI与logic service集成调用

# 3 功能简述

## 1 资产管理

1. 新增资产
2. 删除资产
3. 修改资产
4. 查询资产

基础配置项各项信息

1. 硬件信息

服务器，虚拟机，网络设备等

1. 系统信息

操作系统，系统版本，MAC地址，IP地址等

1. 网络信息

DNS服务器，DHCP服务器及其IP池，NTP服务器等

1. 应用服务信息

略

## 2 IAM

1 人员信息

1 新增人员信息

2 删除人员信息

3 修改人员信息

4 查询人员信息

2 权限管理

1. 权限组
   1. 新增组
   2. 删除组
   3. 修改组
   4. 查询组
   5. 添加人员到组
   6. 删除组内人员
2. 个人权限
   1. 新增
   2. 删除
   3. 修改
   4. 查询

## 3 信息收集监控

1 服务器硬件信息收集

通过服务器远控端提供的rest api获取服务器硬件信息

2 服务器系统信息收集（与zabbix集成）

通过zabbix提供的rest api接口获取系统的各项信息

3 报警

1. 设定报警项
2. 设定报警阀值
3. 邮件报警

# 4 工单管理

与jira进行集成，工单管理流程在jira中处理

硬件

1. 机器申请

功能 ：

1. 录入基本信息

1. 服务器厂商，型号，配置
2. 所属部门
3. 负责组
4. 机器上架

功能：

1. 配置服务器系统信息

配置主机名，远控IP地址，业务IP地址，选择服务器系统安装模版

2.启动主机安装系统

3.安装状态监控反馈

3. 机器维护

功能：

1. 新增资源

配置新增资源项，进度反馈，验收

1. 回收资源

配置回收资源项，进度反馈，验收

3.更换配件

更换配置信息，更换原因，进度反馈，验收

1. 机器下架
2. 配置下架信息

下架原因

下架时间

2.机器下架

进度反馈，验收

系统

1. 基础服务安装，卸载，更新

可通过Yum,Apt,Zypper进行维护的服务

1. 配置服务项，服务器信息
2. 安装指定服务
3. 安装结果反馈

2.基础配置项更改

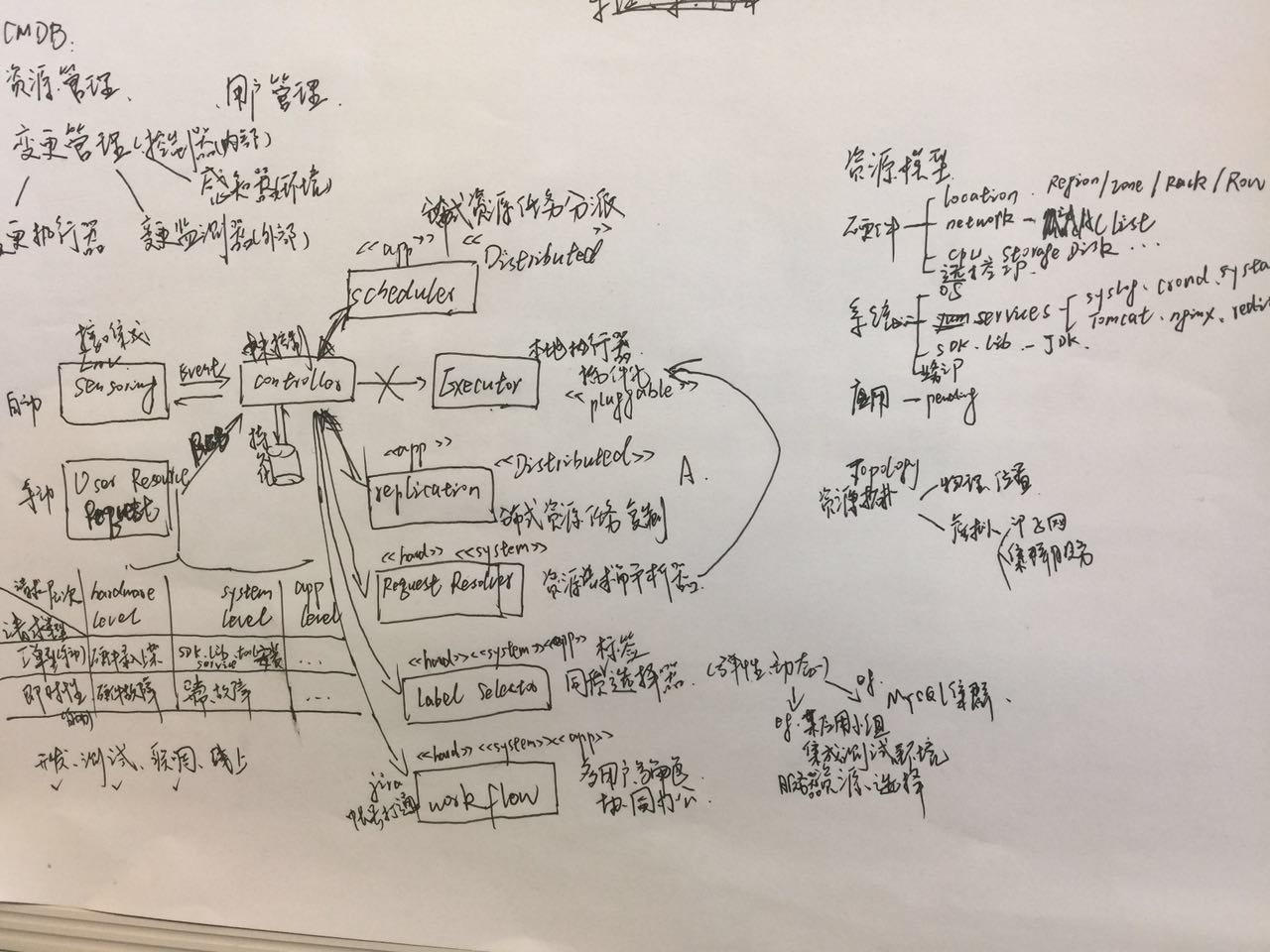
系统基础配置项包括，IP地址，主机名，磁盘等

1. 配置基础配置项
2. 修改基础配置项
3. 结果反馈

# 5 UI

采用bootstrap，分离UI部分，废弃采用开源的WEB UI框架与django进行集成

##### （废弃）CMDB controller 组件功能拆分图



CMDB平台设计宗旨，将各种层次的资源在平台内沉淀，将对资源的各种动作以事件流的方式在平台内流动，多维度视觉审视资源拓扑关系及抽象集合，对资源控制全面且极大灵活。

CMDB平台组件，主要模块细粒度功能描述如下：

#### control center standalone api server:

user authenticate, resource authorize, also function as only entrance for data persistence

#### Sensor:

integrated Zabbix’s monitoring http service for hardware/system level’s live resource status

#### Label selector:

dynamic and flexible resource aggregator targeting certain resource set

#### Request engine --- Request handling framework

~~Request queue~~

~~Request queue for storing request entity in memory, persistent to disk~~

Request state machine (AO)

Tracking request lifecycle until request’s status is finished

Request resolver:

Hardware/system/app level resource request adaptor

Request executor ( idempotent )

Through underline tcp level rpc service, or http/ssh service to manipulate target resource

Request status checker

By-pass checker, whether the request executor function as expected