## 文档说明与简要计划

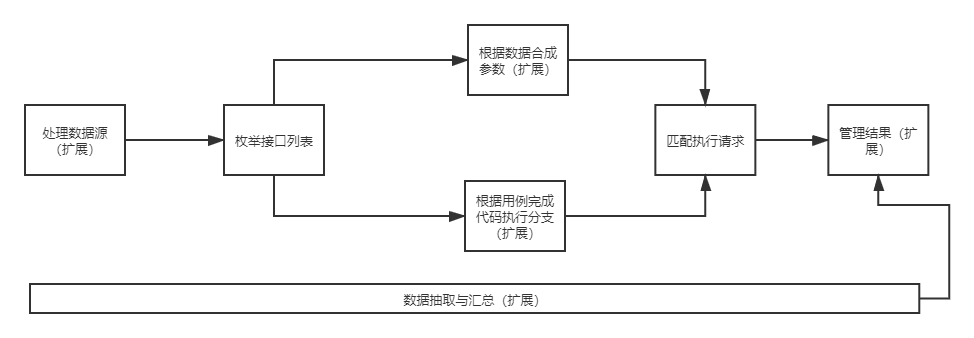
## 原始需求

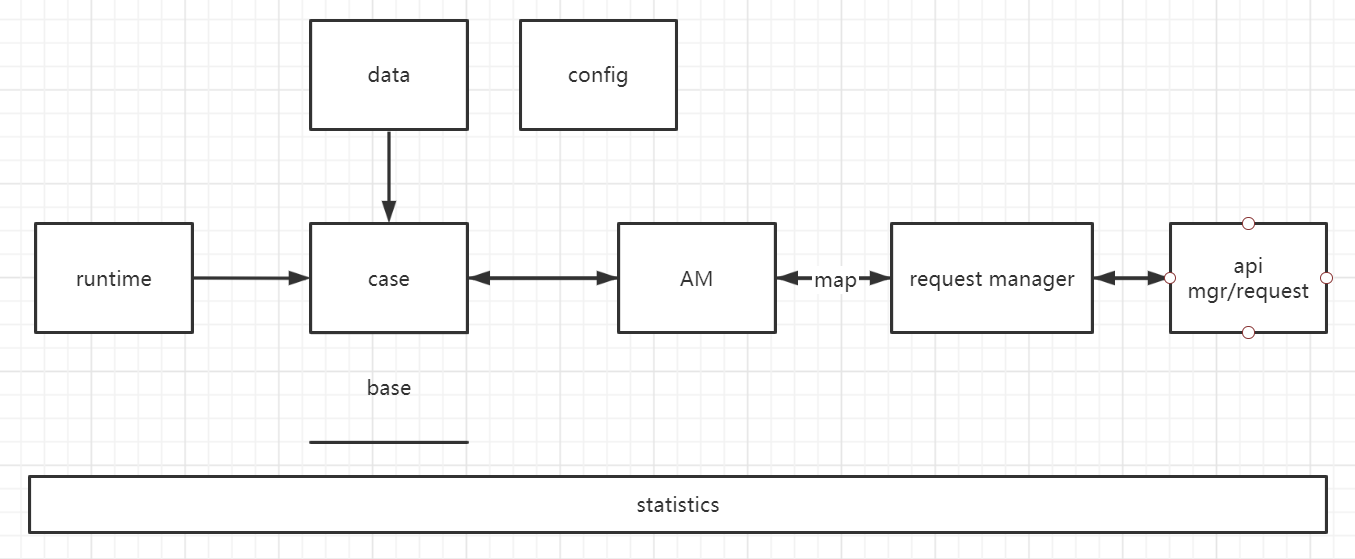
随着测试组的变动、公司对自动化的重视提升，自动化的目标、推行方式、资源投入得到更强的支持，由此我们需要符合以下的开发技术支持以进行更高效的测试自动化工作：

1. 工程设计灵活，能够适应IN3变动【5】
2. 便于 用例与代码管理【1、2】、测试执行使用【1、4】、测试过程信息采集【4、5】
3. 日常使用与工程开发工作量较少【1、3】

## 需求内容

1. 工程执行框架基于Pytest，拆分工作角色重心，分为 测试数据+用例逻辑（文本）+轻业务代码、重代码支持框架与业务功能基础封装
2. 对测试数据+用例的载体内容格式便于表达：定流程&多种平行变化测试数据场景（预期不一）、变流程&较多场景分支依赖&场景流程较长 的两种主要情况
3. 尽可能大的复用现有资源：数据库、接口文档等，以减少迭代变更时的维护成本，以及规范化。在兼容性工作上多方努力平衡工作量
4. 一般而言，代码侧的大致任务如下，其中标注扩展的内容参考【5】





5、典型的扩展需求举例（扩展性）：

|  |  |  |
| --- | --- | --- |
| 扩展功能项 | 功能描述 | 典型扩展修改 |
| 处理数据源 | 支持数据的多种来源与格式的读取，以及向下游传递 | Jira API/数据库/其他格式的文本等 |
| 数据源包含之前不存在不利用的新型信息 |
| 数据源内容兼容（如后文） |
| 数据合成参数 | 将用例当中以IN3前端文本描述的数据映射到接口参数上，可能带有其他的较复杂情况 | 合成依据（如Code-Def表、前端工程内容等） |
| 支持的in3业务功能 |
| 必填校验 |
| 数据中的自定义变量等 |
| 用例转代码 | 用例的执行代码 | in3业务操作修改逻辑与期望（IN3升级迭代） |
| 数据抽取与汇总 | 抽取或者记录测试过程信息（如指标、日志等） | 增加执行覆盖、响应时间、用例统计等新的统计指标 |
| 管理结果 | 形成结果报告、数据输出 | 结果报表内容与字段对接、与Jira & Jenkins等的呈现对接 |
| 其他 |  | 上下游环节发生变动时的支持 |

## 原型设计

几种可能的示例，注意这里的示例是测试样例单元，不一定单个文件就要这样，最好单个数据文件的长度能自由安排、均衡适中

1、

**Data文件，参考YAML格式，也可能引自Jira API等其他地方**

*用例层级结构*：

AM/PC端/主数据-资产管理/IN3-4294主数据-资产管理-资产管理tab的查询

*数据实体*：

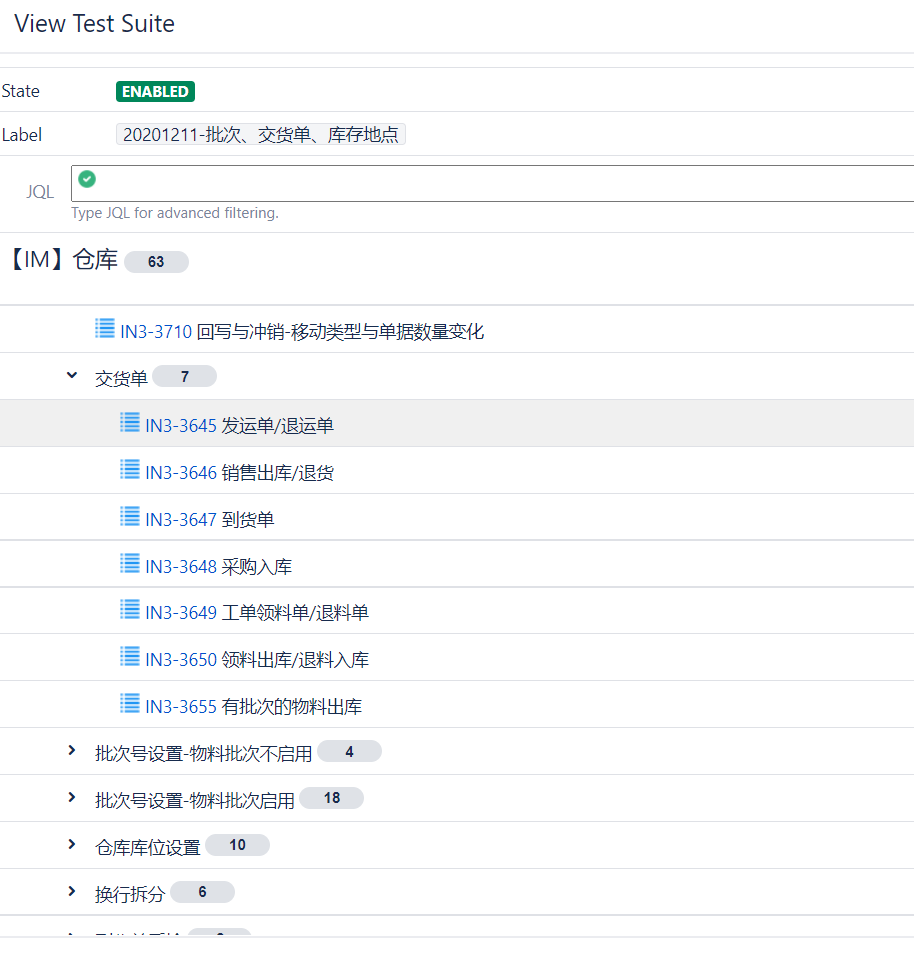
|  |
| --- |
| **id**: 用于用例索引管理？  **描述**: 这是xxxx  **tag**:  **用例步骤s**:  **-步骤1**: 输入资产名称，点击查询  **-数据1**:  **资产名称**: 电子母板  **-期望1**: 可以按查询条件正常筛选出列表内容（查询结果全部符合特征且数量完全）  **-步骤2**: 选择资产类别、启用状态，点击查询  **-期望2**: 可以按查询条件正常筛选出列表内容  **-步骤3**: 输入查询条件，点击重置  **-期望3**: 查询条件回复默认值 |

**用例扩展中，代码侧所需编写的上层结构**

*代码实体：*

|  |
| --- |
| **class** **AM:**#testBaseFunc  **class** **capital:**  **def** query**(**inputData**):**  **return** api operations  **class** **commonUtils:**  **def** factorMapping**()**  **def** logging**()**  **def** recordStandardize**()**  **def** **...** #其他框架工具，也可以考虑独立成类,此处作为函数仅供功能示意  **class** **testCase(** ymlCaseData**:** Jira API**/**DB**/**File **,**preData**):**    baseData**=** casePreAction**(**preData**)**    actualSteps**=** testCaseFunc**&** testBaseFunc **(**ymlCaseData**,** baseData**)->**  **[{** api**:** apiInfoProvider**(),**  originCaseData**:** ymlText**,**  requestFactors**:** requestProvider**(** factorMapping**(** ymlDataKeys**))},**  **{},{},{}]**  #这个地方把请求等信息集中暴露出来是不是能增加拓展性？  result**=**recordStandardize**(**run**(**actualSteps**))**    **class** **newCase1(**testCase**):**#testCaseFunc  **def** \_\_init\_\_**(** ymlData**,** preData**)**  #AM/PC端/主数据-资产管理/IN3-4294主数据-资产管理-资产管理tab的查询  **def** testCaseFunc**():**  **return** preActions**+**AM**.**query**(**Data**)**  **class** **pytest:**  **def** testMethod**():**#测试最上层执行  preData**=**runtimePreAction**()**  newCase1**(**caseData**,** preData**).**result**()** |

2、其他用例示意：以jira情况为例，可能会影响用例与data对应关系以及case代码，面对层级信息不匹配的情况如何处理





## 概要设计

按照前文的需求，进行结构设计

### 数据设计

细节见 数据设计.json

#### BDD支持：

##### 用例：

1、无需特别设计，由现有用例结构与代码复用支持

2、但是以测试过程角度入手的例子还是要带步骤，期望比较多

3、但是再细节的一条用例、一个step，也会包含多个接口行为

##### Yaml：

1、对步骤、用例、用例集、每字段描述的标识

#### DDT支持：

##### 用例：

1. 这一块在文本用例里面多半还是拆条写的，比如入库x，出库y，这个过程随着x&y的数量关系组合的不一样会有y<x的限制
2. 从属于同一条用例，但是有不同的数据与对应期望

##### Yaml：

1. 增加用例所属关系标识、增加操作、数据&期望对应关联关系标识
2. 或者将ddt的平行数据视为一个用例的多个步骤，其他就是多个步骤的单独期望
3. 如果是同一个期望，那就直接做数据list好了
4. **换个形式，对最外层list包装的数据、期望进行匹配就好了**
5. 层级关系设计，参考jira导入的字段

#### 复用性：

1. 接口文档直接参与工程
2. 数据库config、合成依据code-def等

#### 扩展性：

1. 对存在扩展的功能对象使用 功能对象逻辑控制类provider（复用处定义，调用处实现）+专属处理类（复用处定义，复用处调用） 进行封装
2. 上下游数据环节传递统一使用字典 or list包装

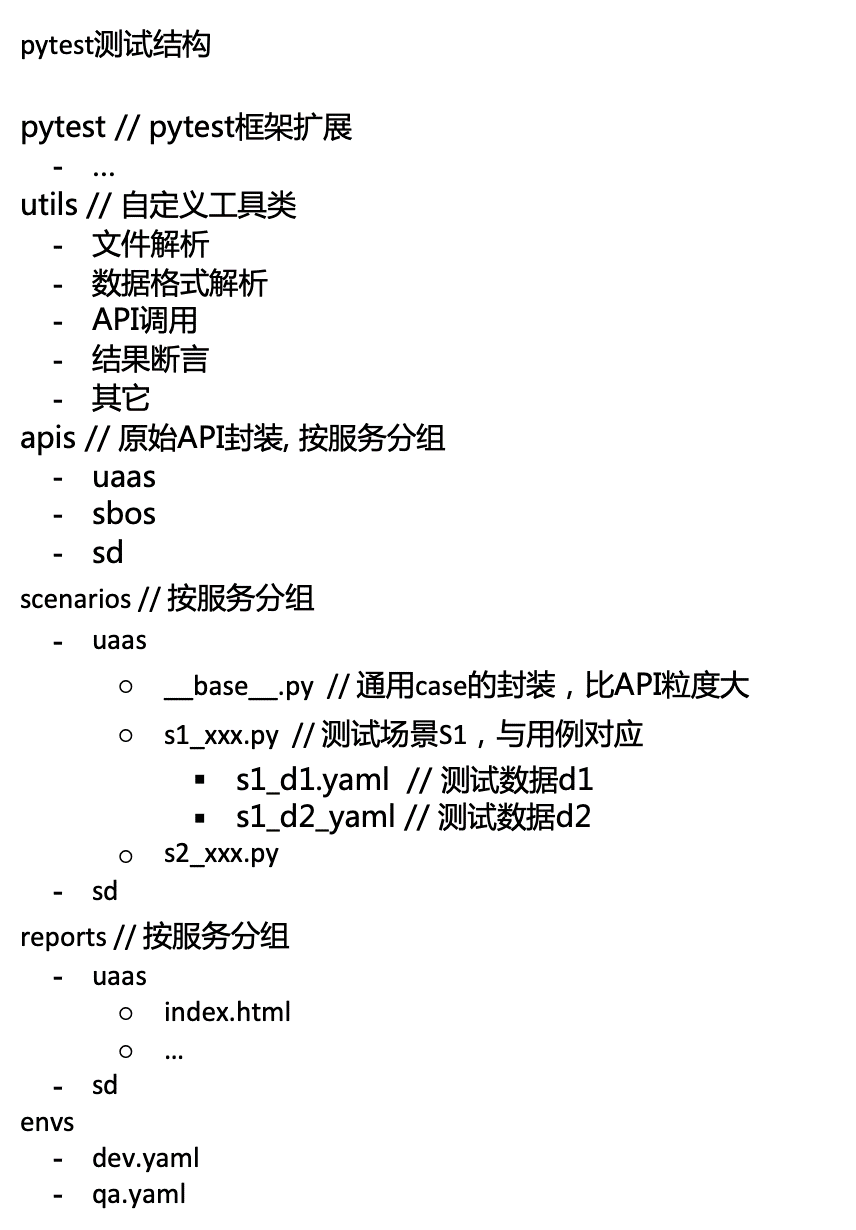
### 代码设计

#### 层级设计

代码侧存在不同的一个层级关系

但是又存在前面的一个流程设计，整体设计调用主要在base层

先镇个楼



|  |  |
| --- | --- |
| **功能项** | **层级描述** |
| 处理数据源 | Util层复用，base调用 |
|  |
|  |
| 枚举接口 | Case调用，base复用 |  |
| 数据合成参数 | Util层复用，base调用 |  |
|  |
|  |
|  |
| 用例转代码 | Case复用，runtime调用 |  |
| 匹配执行请求 | Base复用&调用 |  |
| 数据抽取与汇总 | Util层复用，base调用 |  |
| 管理结果 | Util层复用，base调用 |  |
|  |  |  |
| 业务封装-library | library层复用，case层调用：业务模型需要抽象，自主复用 |  |
| api处理 | api层复用，base调用 |  |
| 测试环境与系统配置-conf | ？还没设计好，这是扩展性需求 |  |
| Case | 自主复用（比如集中的query和match） |  |

|  |  |  |
| --- | --- | --- |
| report | | |
| util | runtime | config |
| case |
| library |
| base |
| api |
|  |
| data | | |

#### 过程模块设计-excel