|  |  |  |  |
| --- | --- | --- | --- |
|  | **文 档 编 号** | **产品版本** | **密级** |
|  |  |  |
|  | | **共 页** |

详细设计说明书

**北京瑞金麟网络技术服务有限公司**

**Rkylin Network Technology & Service Co. Ltd.**

**二○一五年四月**

改 版 履 历 表

|  |  |  |  |
| --- | --- | --- | --- |
| **版数** | **发行年月** | **作成者** | **变 更 内 容 概 要** |
| V0.0 | 2015-09-17 | 孙锐斌 | 初版做成 |

目录

[1．引言 8](#_Toc432169311)

[1.1编写目的 8](#_Toc432169312)

[1.2项目背景 8](#_Toc432169313)

[1.3定义 8](#_Toc432169314)

[2．总体设计 10](#_Toc432169315)

[2.1需求概述 10](#_Toc432169316)

[2.2软件结构 10](#_Toc432169317)

[3．程序描述 11](#_Toc432169318)

[3.1功能 11](#_Toc432169319)

[3.1.1清算 11](#_Toc432169320)

[3.1.1.1获取账户系统交易信息 11](#_Toc432169321)

[3.1.1.1.1实时接受交易信息 11](#_Toc432169322)

[3.1.1.1.2主动获取交易信息 11](#_Toc432169323)

[3.1.1.4冲正处理 11](#_Toc432169324)

[3.1.1.5交易后退款处理 12](#_Toc432169325)

[3.1.1.6清分 12](#_Toc432169326)

[3.1.1.7分润(结算) 12](#_Toc432169327)

[3.1.2对账 13](#_Toc432169328)

[3.1.2.1上游对账文件 13](#_Toc432169329)

[3.1.2.2读取上游对账文件信息 13](#_Toc432169330)

[3.1.2.3获取下游对账交易 13](#_Toc432169331)

[3.1.2.4进行对账 13](#_Toc432169332)

[3.1.2.5上传下游对账文件 14](#_Toc432169333)

[3.1.3系统管理页面 14](#_Toc432169334)

[3.1.3.1文件信息管理 14](#_Toc432169335)

[3.1.3.2文件列信息管理 14](#_Toc432169336)

[3.1.3.3对账规则管理 15](#_Toc432169337)

[3.1.3.4分润规则管理 16](#_Toc432169338)

[3.1.3.5分润规则明细管理 16](#_Toc432169339)

[3.1.3.6交易明细管理 17](#_Toc432169340)

[3.1.3.7上游交易管理 17](#_Toc432169341)

[3.1.3.8分润结果管理 18](#_Toc432169342)

[3.1.3.9挂账信息管理 18](#_Toc432169343)

[3.1.4业务相关画面 19](#_Toc432169344)

[3.1.4.1 清分 19](#_Toc432169345)

[3.1.4.2 结算 19](#_Toc432169346)

[3.1.4.3 冲正 19](#_Toc432169347)

[3.1.4.4 抹账 || 交易后退款 19](#_Toc432169348)

[3.1.4.5 对账 19](#_Toc432169349)

[3.2性能 20](#_Toc432169350)

[3.3输入项目 20](#_Toc432169351)

[3.3.1上游对账文件格式 20](#_Toc432169352)

[3.4输出项目 20](#_Toc432169353)

[3.4.1下游对账文件格式 20](#_Toc432169354)

[3.5算法 21](#_Toc432169355)

[3.6程序逻辑 22](#_Toc432169356)

[3.6.1 清算流程图 22](#_Toc432169357)

[3.6.1.1 获取分润规则 23](#_Toc432169358)

[3.6.1.2 判断是否分润 24](#_Toc432169359)

[3.6.1.3 匹配分润规则（逻辑） 25](#_Toc432169360)

[3.6.1.4分润操作（逻辑） 26](#_Toc432169361)

[3.6.1.5挂账 27](#_Toc432169362)

[3.6.1.6冲正 28](#_Toc432169363)

[3.6.1.7结算 29](#_Toc432169365)

[3.6.2 对账 30](#_Toc432169366)

[3.6.2.1 对账（逻辑） 31](#_Toc432169367)

[3.6.2.2 对账规则 32](#_Toc432169368)

[3.6.2.3 对账结构 33](#_Toc432169369)

[3.6.2.4 对账业务逻辑 34](#_Toc432169370)

[3.6.2.4 对账结果插入 35](#_Toc432169371)

[3.6.3 上传下游对账文件 36](#_Toc432169372)

[3.6.3.1 查询清算交易信息 36](#_Toc432169374)

[3.6.3.2 查询文件属性信息 38](#_Toc432169375)

[3.6.3.3 封装文件信息结构体 39](#_Toc432169376)

[3.6.4 下载上游对账文件 40](#_Toc432169377)

[3.6.4.1 读取对账文件并入库 41](#_Toc432169378)

[3.6.4.2 获取文件属性信息 42](#_Toc432169379)

[3.6.4.3 获取上游对账交易信息 43](#_Toc432169380)

[3.6.4.4 文件信息结构体 44](#_Toc432169381)

[3.6.4 读取多渠道交易信息 45](#_Toc432169382)

[3.7接口 46](#_Toc432169383)

[3.7.1存储账户系统交易信息接口 46](#_Toc432169384)

[3.8存储分配 46](#_Toc432169385)

[3.9限制条件 46](#_Toc432169386)

[3.10测试要点 46](#_Toc432169387)

[3.10.1清算 46](#_Toc432169388)

[3.10.1.1获取账户系统交易信息 46](#_Toc432169389)

[3.10.1.2冲正处理 46](#_Toc432169390)

[3.10.1.5挂账处理 46](#_Toc432169391)

[3.10.1.6清分 47](#_Toc432169392)

[3.10.1.7分润(结算) 47](#_Toc432169393)

[3.10.2对账 47](#_Toc432169394)

[3.10.2.1读取上游对账文件信息 47](#_Toc432169395)

[3.10.2.2获取下游对账交易 47](#_Toc432169396)

[3.10.2.3进行对账 47](#_Toc432169397)

[3.10.2.4上传下游对账文件 48](#_Toc432169398)

# 1．引言

## 1.1编写目的

本详细设计是为了测试组和与清洁算系统相关各组能详细了解清洁算1期完成的功能，以能更好的完成测试工作，更好的与相关系统完成对应功能

## 1.2项目背景

现有清洁算功能集成在账户系统，无法应对未来的大量上游渠道和大量下游商户的接入，1期清洁算系统旨在完全所有必要清洁算后端处理，部分差错处理，部分数据管理

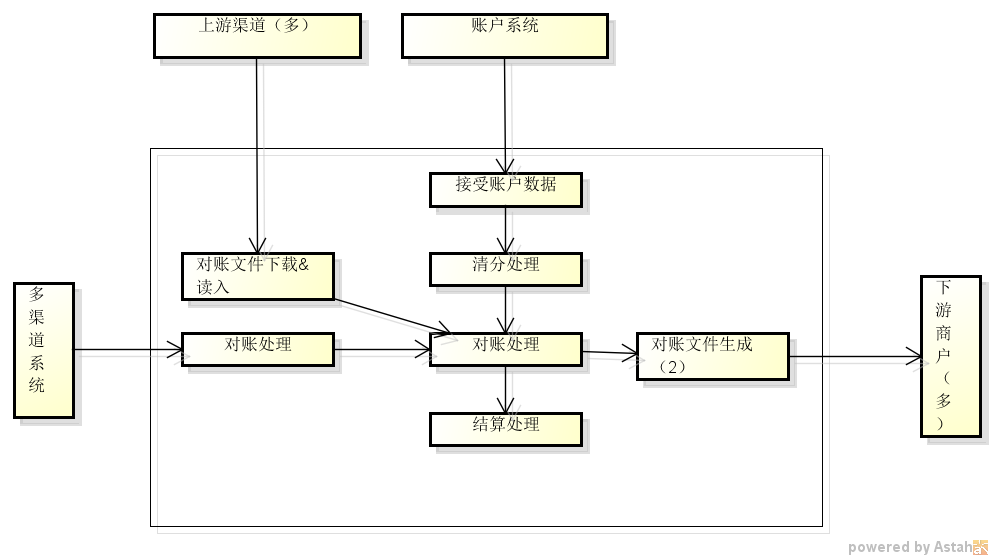
## 1.3定义

|  |  |  |
| --- | --- | --- |
| 序号 | 定义 | 解释说明 |
| 1 | 清算 | 资金清分和结算的总称。是交易数据的清分、计费、轧差、汇总的过程。作为最终记账，对前期完成的一笔或多笔交易所进行的资金划拨。 |
| 2 | 清分 | 清分与交易数据相关，清分是指交易记录汇总分类并计算出相互结算金额的过程。 |
| 3 | 结算 | 结算与交易金额相关，产生支付文件、清算凭证的过程。 |
| 4 | 清算数据 | 是指根据交易记录进行清分、计算得到作为资金划拨和对账依据的数据。 |
| 5 | 清算机构 | 是指在金融产品中心平台中清结算机构的总称；一般分为支付机构和受理机构。 |
| 6 | 支付机构 | 支付机构是指资金的来源方，包括银行、第三方支付机构、行业卡发卡机构。 |
| 7 | 受理机构 | 交易的受理主体即为与商户签约引入金融产品中心的机构。 |
| 8 | 自主清分 | 瑞金麟金融中心账户交易的清算模式，其交易数据以瑞金麟金融中心账户数据为主。 |
| 9 | 非自主清分 | 指与银行、第三方支付机构、行业卡等的通道交类交易的清算模式。其交易数据以银行和第三方支付机构为主。 |
| 10 | 对账 | 两个机构（受理方机构、支付机构）、商户之间的交易总额核对，已达到数据的一致性。 |
| 11 | 错账 | 我方与渠道参与对账的金额不一致的交易 |
| 12 | 长款 | 我方与渠道对账发生交易不一致的情况,我方不会因此交易支出额外款项的情况(需查明原因)  例如:充值交易A,我方没有A交易,但是渠道有A交易,实际上充值金额已经入到我方账户 |
| 13 | 短款 | 我方与渠道对账发生交易不一致的情况,我方会因此交易支出额外款项的情况(严重情况,需查明原因)  例如:充值交易A,我方有A交易,但是渠道没有A交易,我方已经把充值金额加入充值者账户到实际上充值金额并没有入到我方账户 |
| 14 | 平账 | 我方与渠道对账交易一致 |
| 15 | 一级清算 | 机构间的清算，即支付机构、金融产品中心和受理机构之间的数据清结算 |
| 16 | 二级清算 | 受理机构与商户以及各分润主体之间的清结算。 |
|  |  |  |

# 2．总体设计

## 2.1需求概述

## 2.2软件结构



# 3．程序描述

## 3.1功能

#### 3.1.1清算

一期“清算”的业务主要指“分润”，既将需要分润的交易做分润处理获取分润结果信息并存入数据库，然后根据分润结果信息更新账户系统相应的账户信息。

##### 3.1.1.1获取账户系统交易信息

“账户系统”是“清算系统”交易信息的来源之一，需要分润的交易信息目前都由“清算系统”提供。获取交易的方式如下两种，（1）由“账户系统”实时推送。（2）由“清结算”系统主动从“账户DB”，获取信息存入SETTLE\_TRANS\_DETAIL表中“STATUS\_ID为1[成功] || 0[失败]”，“READ\_STATUS\_ID为账户系统交易状态”

清算交易状态STATUS\_ID说明:

交易失败：0；交易成功:1；交易不明:2；

清分失败：10；清分成功：11；

异常：20；平账：21；错账：22；长款：23；短款：24

清算失败：40；清算成功：41；

唯一标示：ORDER\_NO + MERCHANT\_CODE

###### 3.1.1.1.1实时接受交易信息

“清结算系统”提供接口TradeService，接口中暴露saveAccountTrans方法入参为“清算”交易信息实体Bean，返回值为提示信息Map，账户系统获取到任何交易信息都会立即通过dubbo调用此方法将交易信息存入“清算DB”。

###### 3**.1.1.1.2主动获取交易信息**

当发生例如网络中断或电力系统故障等特殊情况时，“账户系统”实时推送交易信息未能成功，清算系统可以通过页面调用接口SettleWebInterface中暴露的getAccountTransOrderInfos方法主动获取“账户系统”交易，入参为交易的开始时间和结束时间[匹配REQUEST\_TIME]。

##### 3.1.1.4冲正处理

添加一笔交易B用于对T日进行的交易（未结算交易）A进行冲正处理，此次行为中B为冲正交易，A为被冲正的交易。当“清算系统”的SETTLE\_TRANS\_DETAIL【交易信息表】中存在未处理的冲正交易时，会对此交易做如下操作，判断此交易对应的被冲正交易是否需要分润，如需分润则将被冲正交易对应分润结果的状态更改为2。

##### 3.1.1.5交易后退款处理

添加一笔交易B用于对T - N日进行的交易（已结算交易）A进行交易后退款处理，如果A交易为需要分润的交易将依据A交易的分润结果创建退款信息并存入“清结算DB”的SETTLE\_TRANS\_BILL表中。

唯一标示：ORDER\_NO + ROOT\_INST\_CD + USER\_ID

##### 3.1.1.6清分

对交易进行清分处理。如果次交易不需要清分则将此交易的状态update为已清分STATUS\_ID=11，否则进入清分逻辑进行处理详情如下。

* + - * 匹配对应分润规则

“清结算DB”中SETTLE\_PROFIT\_KEY为分润规则表（一方），SETTLE\_PROFIT\_RULE为分润规则明细表（多方）两张表以‘一对多’向关联。“清算系统”使用这两张表管理分润规则。

正常情况使用funcCode和marchantCode匹配唯一分润规则，特殊情况使用KEY\_NAME和KEY\_VALUE字段配置匹配映射关系。

* + - * 清算处理

将交易信息按照此信息对用的分润规则，计算分润金额和匹配分润账户信息，既得到“分润结果信息”。

* + - * 清算结果入库

将清算处理后的“分润结果信息”存入SETTLE\_TRANS\_PROFIT【分润结果表】，STATUS\_ID为0未进行结算。

如果表中存在此交易的分润结果，做update否则做insert。

唯一标示：ORDER\_NO + MERCHANT\_CODE + ROOT\_INST\_CD + USER\_ID + STATUS\_ID==0

##### 3.1.1.7分润(结算)

* 获取未分润的清算结果

从SETTLE\_TRANS\_PROFIT和SETTLE\_TRANS\_DETAIL表中关联查询‘对账成功’并‘未结算’【STATUS\_ID==0】的分润结果信息。

* 分润处理

从SETTLE\_TRANS\_BILL表中查询所有未处理的挂账信息，和分润结果信息进行匹配和计算获得最终的分润金额，通过dubbo调用‘账户系统’的分润接口实现分润处理。STATUS\_ID为1【已进行】结算

#### 3.1.2对账

将上游渠道的交易信息和下游机构的交易信息进行匹配并核对交易总金额，验证交易信息的一致性，如果上下游两笔交易对平既“平账”，否则根据不同情况分为“长款”，“短款”，“错账”。需要根据具体情况进行分析和处理。

##### 3.1.2.1上游对账文件

下载上游参加对账的交易信息文件，既“对账文件”，调用FileBaseLogic类中的readUpFile方法，入参为settleFile清算文件格式对象和accountDate账期读入账户。返回值List<Map>为从文件中读取到的交易信息结构体。

文件对象的唯一标示:MERCHANT\_CODE + READ\_TYPE + PAY\_CHANNEL\_ID + FILE\_ACTIVE

##### 3.1.2.2读取上游对账文件信息

在ReadFileLogic类中的readCollateFile方法，将从文件中读取到的交易信息结构体，匹配对应的文件格式信息，并按照文件格式匹配对应字段存入“清结算DB”SETTLE\_TRANS\_ACCOUNT表。此表中交易状态STATUS\_ID为1。上游交易信息，已存入“清结算DB”

##### 3.1.2.3获取下游对账交易

调用CollateLogic类中的getTransDetailFromMultiGate方法读取下游交易，入参为accountDate[匹配UPDATED\_TIME]账期如果调用重载的无参方法，既默认使用当前账期进行获取操作。目前需要对账的交易只从“多渠道系统”进行读取，读取过程中通过SETTLE\_PARAMETER\_INFO的配置对应funcCode并存入“清结算DB”SETTLE\_TRANS\_DETAIL，交易状态STATUS\_ID默认为11。

##### 3.1.2.4进行对账

* + - * 获取上游对账信息

从SETTLE\_TRANS\_ACCOUNT表中读取STATUS\_ID为1的全部交易，既未对账的上游交易。

* + - * 获取下游对账信息

从SETTLE\_TRANS\_DETAIL表中读取STATUS\_ID为11的并且ORDER\_TYPE为1的全部交易，既未对账的下游交易。

* + - * 获取并匹配对账规则进行对账

获取全部对账规则，上（下）游交易通过相应的规则比配对应的下（上）游交易信息然后通过规则中配置对账的项目，进行对账处理，生成对象结果信息，既交易结果Bean准备存入SETTLE\_BALANCE\_ENTRY表STATUS\_ID为对账结果，此时SETTLE\_TRANS\_DETAIL（下游交易）和SETTLE\_TRANS\_ACCOUNT(上游交易)的STATUS\_ID也update为对账结果

（异常:20；平账:21；错账:22；长款:23；短款:24）。

* + - * 存储对账结果

将生成的对账结果信息存入SETTLE\_BALANCE\_ENTRY表STATUS\_ID为对账结果，如果表中存在此条对账结果既做update操作否则insert。

唯一标示：ORDER\_NO + OBLIGATE1(对账key)

##### 3.1.2.5上传下游对账文件

通过WriteFileLogic中提供的uploadCollateFile方法进行上传下游对账文件操作，步骤如下：

* 获取文件格式信息

通过merchantCode机构号和funcCode业务编码获取文件格式信息和相关联的明细信息。

* 获取对账交易信息

通过文件明细的信息查询SETTLE\_TRANS\_DETAIL表获取即将写入对账文件的对账交易信息。

* 生成对账文件并上传ROP服务器

将交易信息按照匹配到的文件格式明细中描述的格式生成下游对账文件，调用SettlementUtil提供的setFileFromROP方法将文件上传至ROP服务器。

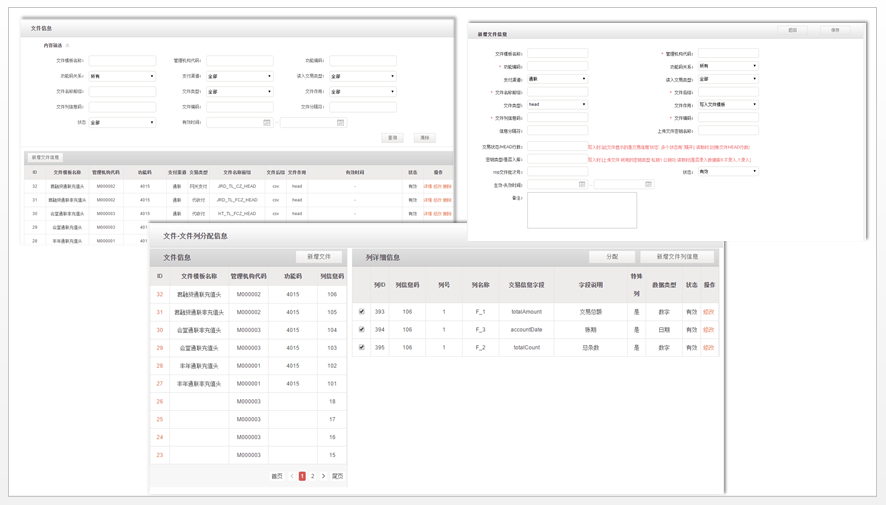
#### 3.1.3系统管理页面

##### 3.1.3.1文件信息管理

下载上游对账文件入库和上传下游对账文件至ROP服务器业务中，因为上游有N个渠道，下游有N个商户并且各个渠道和各个商户的文件格式不尽相同，考虑到为实现代码的复用性和可维护性，所以将文件的格式信息维护到DB中既“文件信息管理”相关DB中的SETTLE\_FILE表。

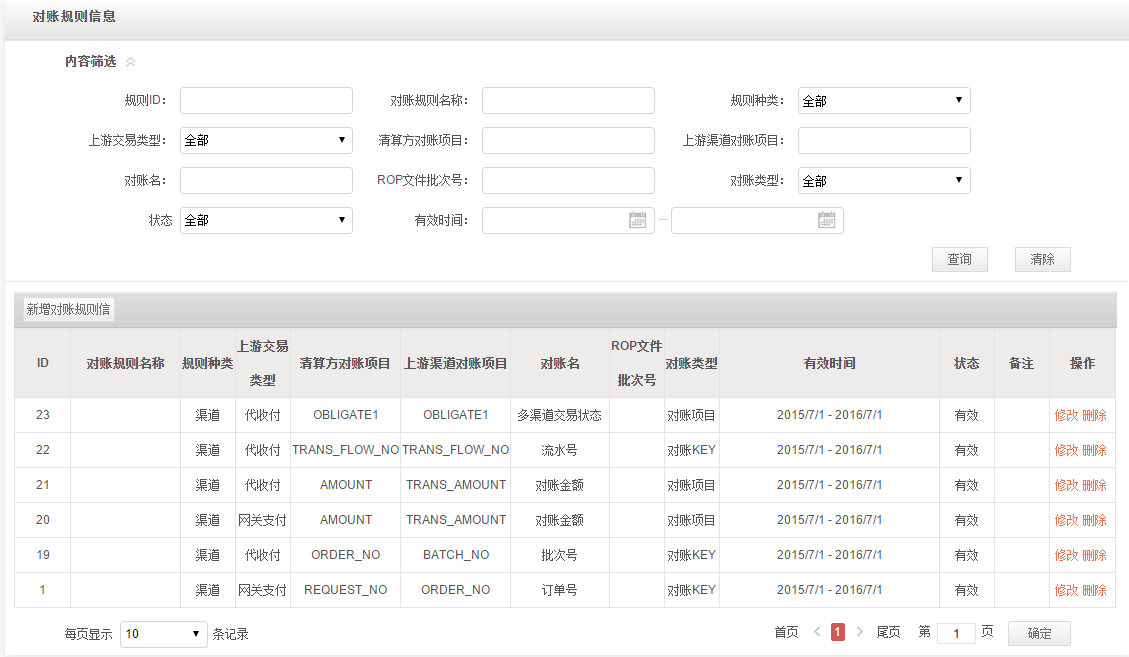
##### 3.1.3.2文件列信息管理

文件列与表中字段名的映射关系。例如FILE\_COLUMN\_KEY\_CODE存储的是文件中所需要的交易表字段名或者特殊列的约定名称。



##### 3.1.3.3对账规则管理

对账业务需要将上游和下游描述同一笔交易的信息匹配到一起，然后进行对账。匹配交易所依据的KEY，和对账时需对照的数据项等信息配置到对账规则表里既SETTLE\_RULE表中。

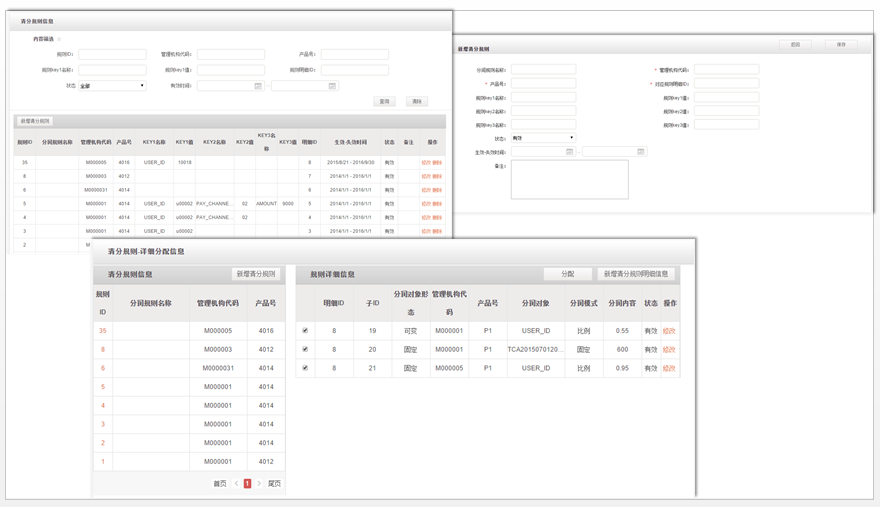


##### 3.1.3.4分润规则管理

当对交易信息进行清分处理时，需根据交易信息匹配相应切唯一的分润规则，普通情况下采用funcCode和merchantCode就可以匹配到唯一的分润规则，但如有特殊情况，例如同一机构，同一业务但商家不同并且有各自对应的规则需在此表中使用KEY\_NAME(交易表字段名)和KEY\_VALUE(交易表字段值)这组字段进行配置。（SETTLE\_PROFIT\_KEY预留3组KEY\_NAME(交易表字段名)和KEY\_VALUE(交易表字段值)字段，可以满足5层条件的 匹配。）

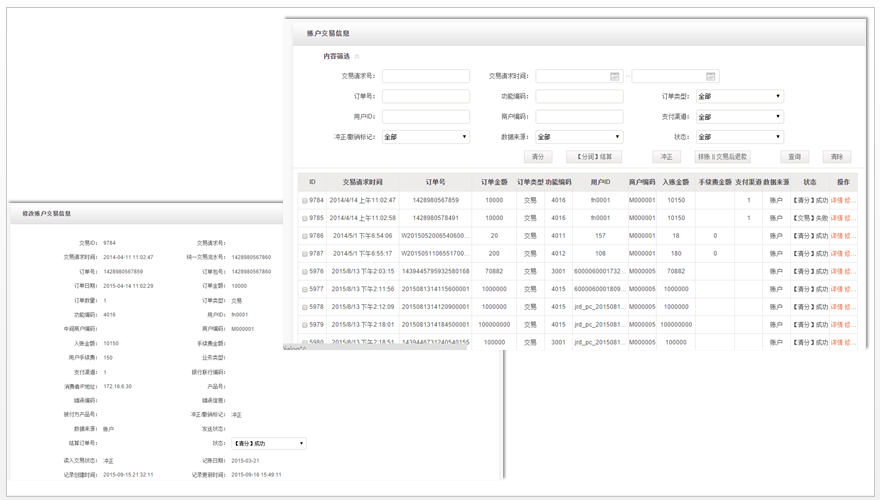
##### 3.1.3.5分润规则明细管理

分润规则明细，用来描述交易金额具体的清分方式，例如 情分类型，清分形式：百分比or固定金额，封顶值，封底值等信息。



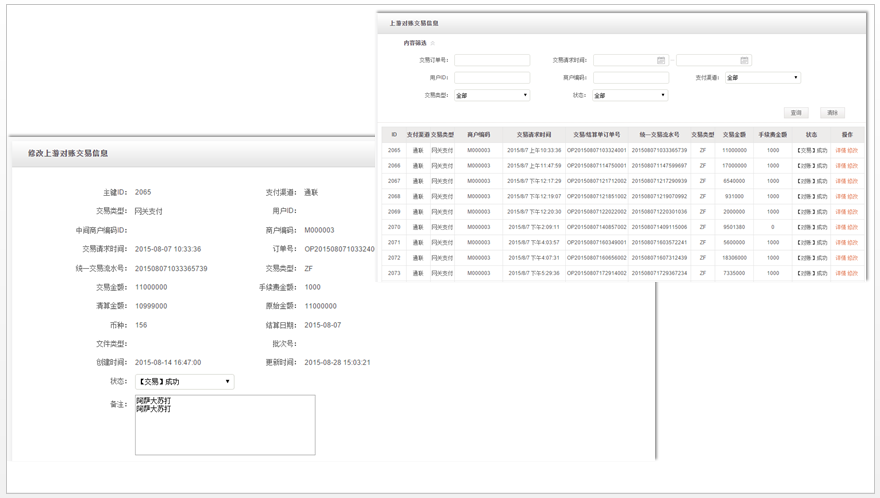
##### 3.1.3.6交易明细管理

SETTLE\_TRANS\_DETAIL表的管理画面



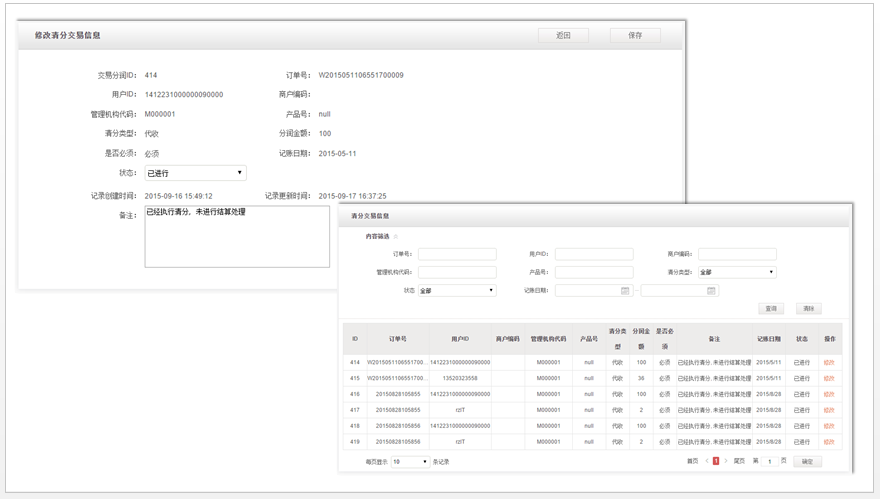
##### 3.1.3.7上游交易管理

SETTLE\_TRANS\_ACCOUNT表的管理画面



##### 3.1.3.8分润结果管理

SETTLE\_BALANCE\_ENTRY表的管理画面



##### 3.1.3.9挂账信息管理

SETTLE\_TRANS\_BILL表的管理画面



#### 3.1.4业务相关画面

##### 3.1.4.1 清分

详见上文 3.1.1.6

##### 3.1.4.2 结算

详见上文 3.1.1.7

##### 3.1.4.3 冲正

详见上文 3.1.1.4

##### 3.1.4.4 抹账 || 交易后退款

详见上文 3.1.1.5

##### 3.1.4.5 对账

详见上文 3.1.2

## 3.2性能

略

## 3.3输入项目

#### 3.3.1上游对账文件格式

第一行为汇总信息：

结算日期|批次号|交易笔数|成功笔数|交易金额|退款笔数|退款金额|手续费|清算金额

第二行开始为交易明细：

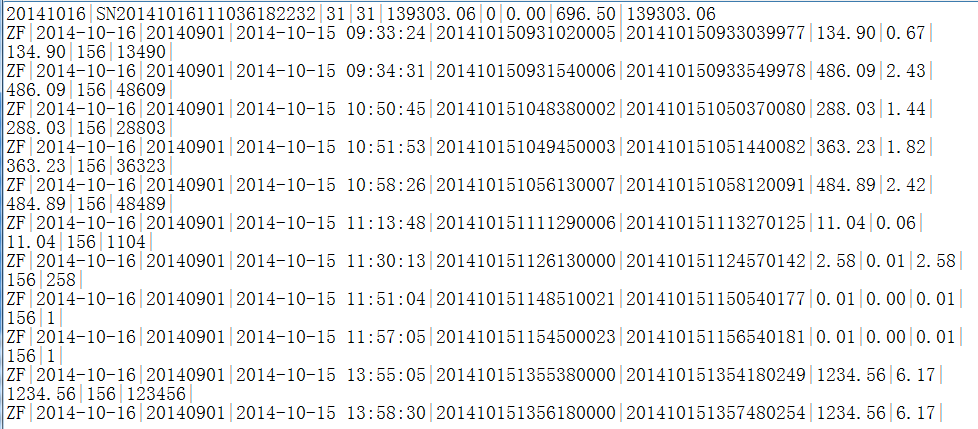
交易类型|结算日期|商户号|交易时间|商户订单号|通联流水号|交易金额|手续费|清算金额|币种|商户原始订单金额(分)

备注：币种采用国标形式（3位数字，比如人民币156 美元840 等）

文件最后为签名信息，与交易明细间有一空行。

请参考对账文件范例如下：

汇总和交易明细明文：



## 3.4输出项目

#### 3.4.1下游对账文件格式

第一行为汇总信息：

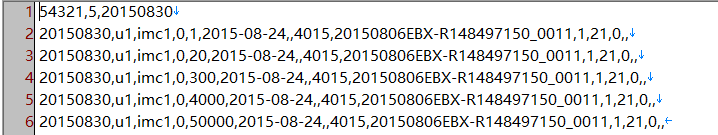
交易金额,总条数,账期

第二行开始为交易明细：

账期,用户ID,商户号,交易流水号,交易金额,交易日期,交易时间,功能编码,订单号,订单数量,交易状态,手续费1,手续费2,备注,预留字段1, 预留字段2ssssssss

请参考对账文件范例如下：

汇总和交易明细明文：

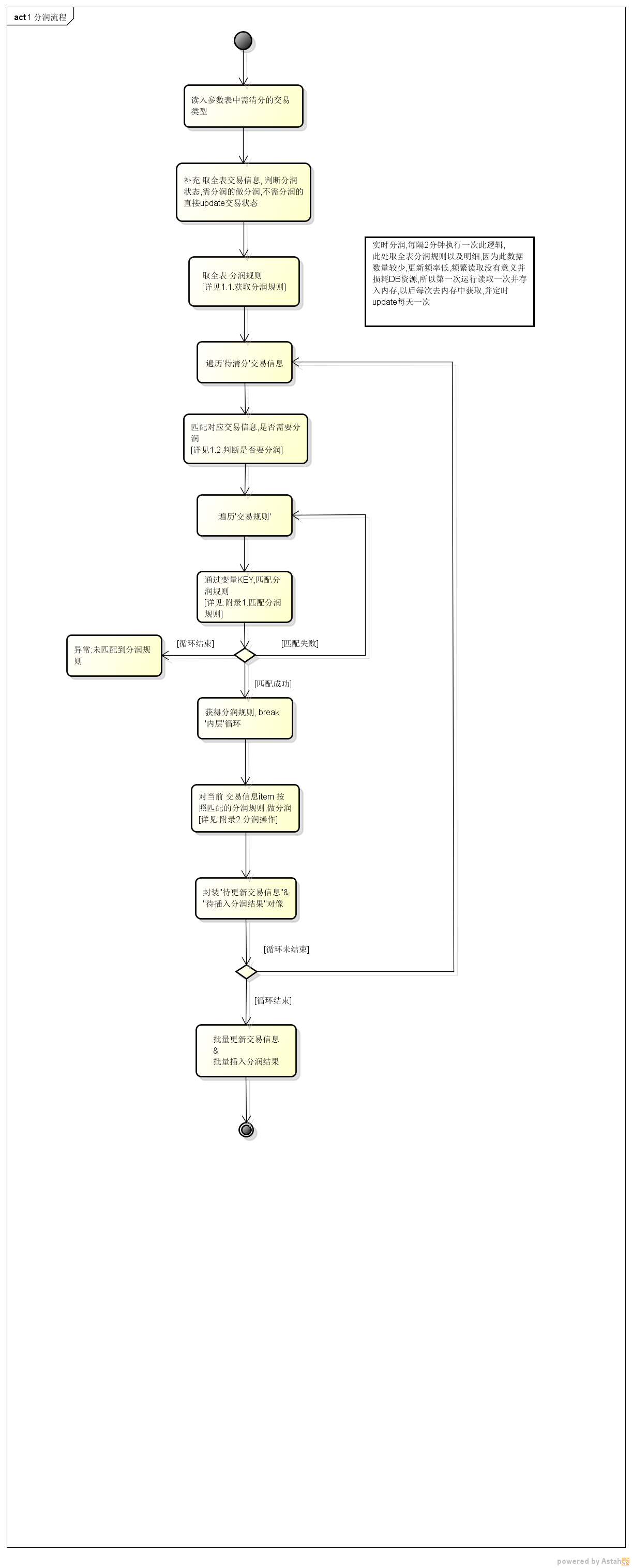


## 3.5算法

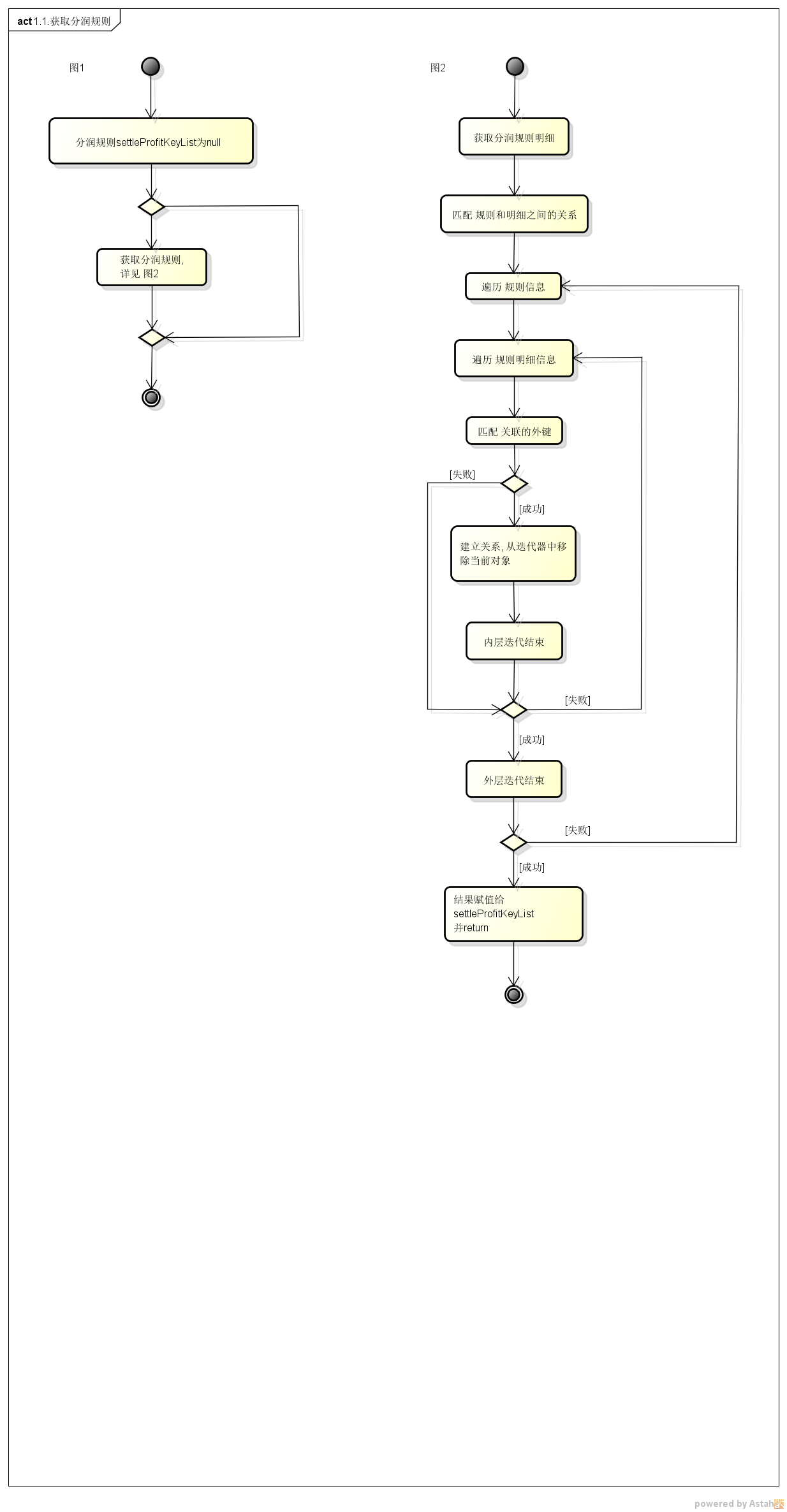
略

## 3.6程序逻辑

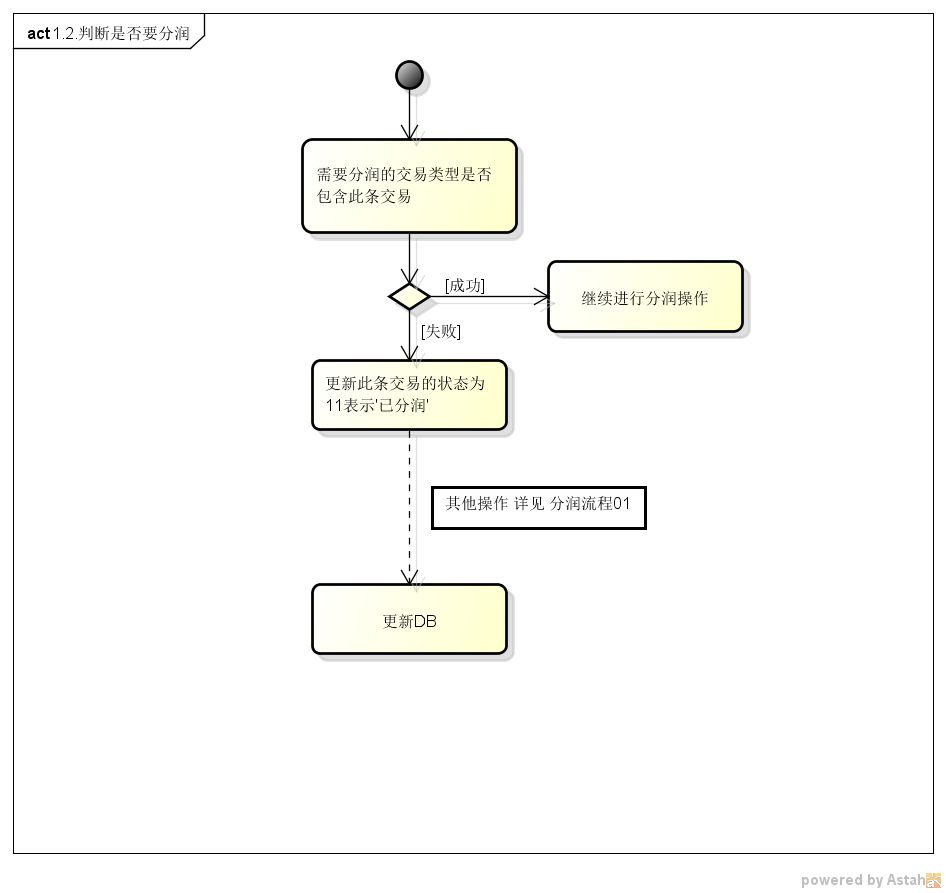
#### 3.6.1 清算流程图



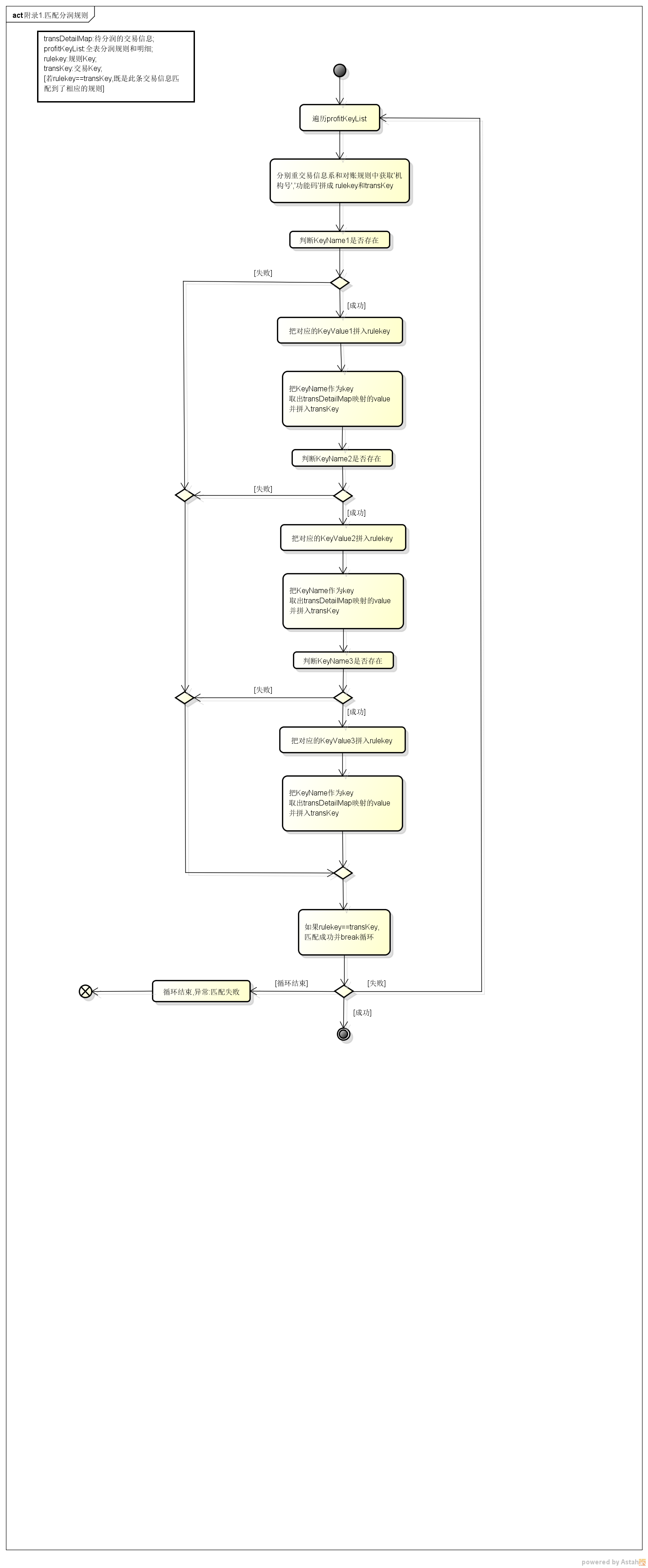
##### 3.6.1.1 获取分润规则



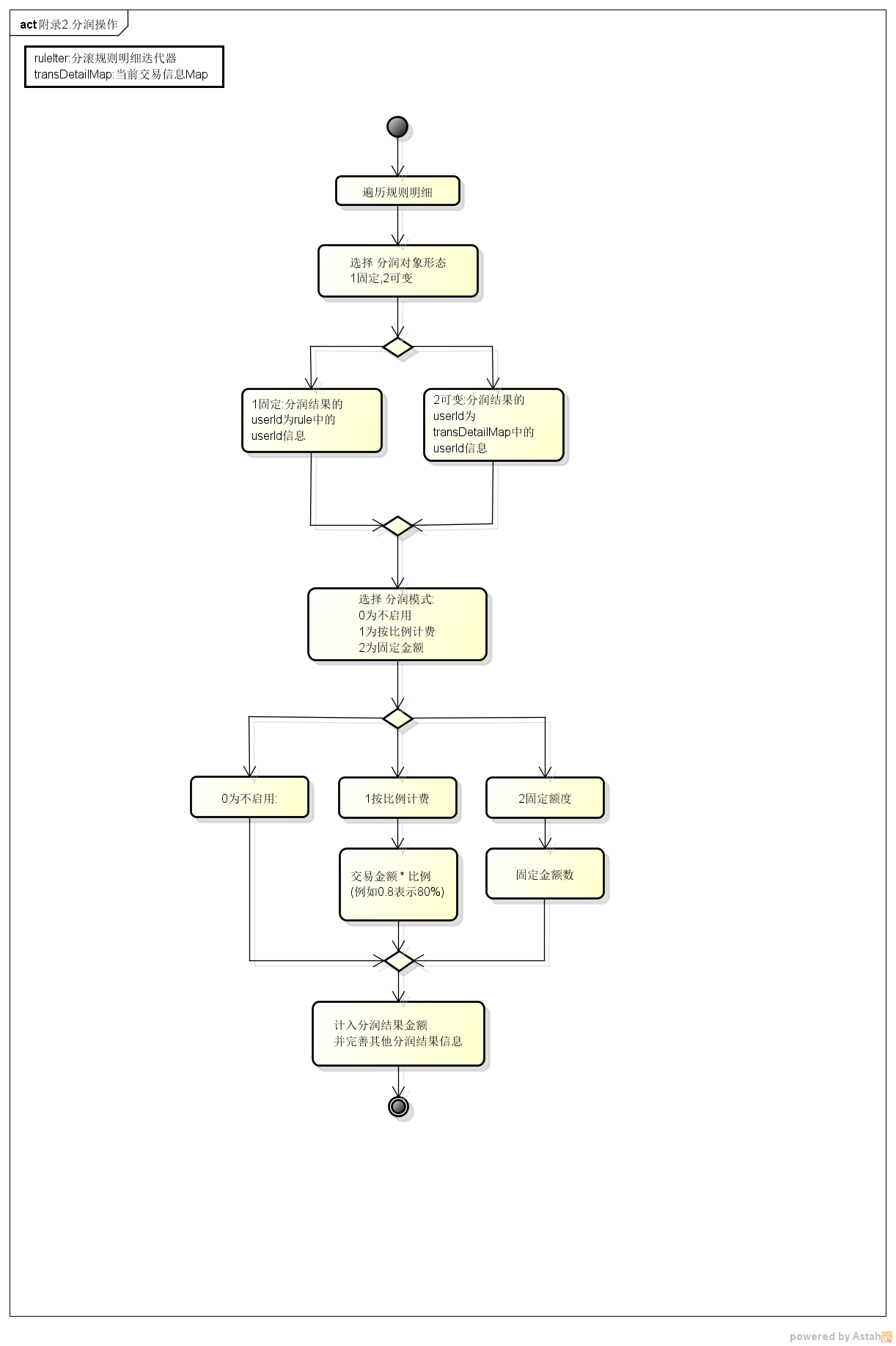
##### 3.6.1.2 判断是否分润



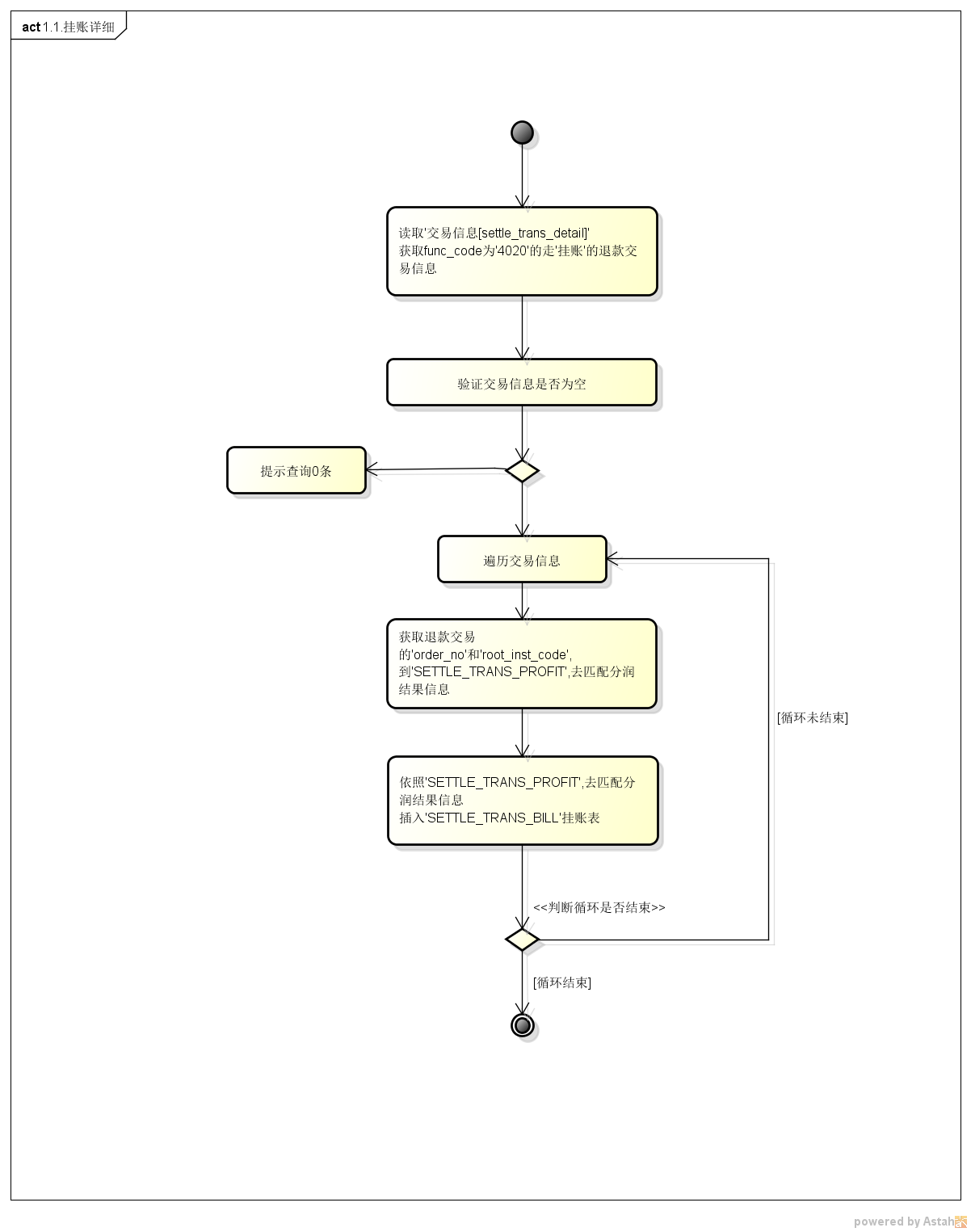
##### 3.6.1.3 匹配分润规则（逻辑）



##### 3.6.1.4分润操作（逻辑）

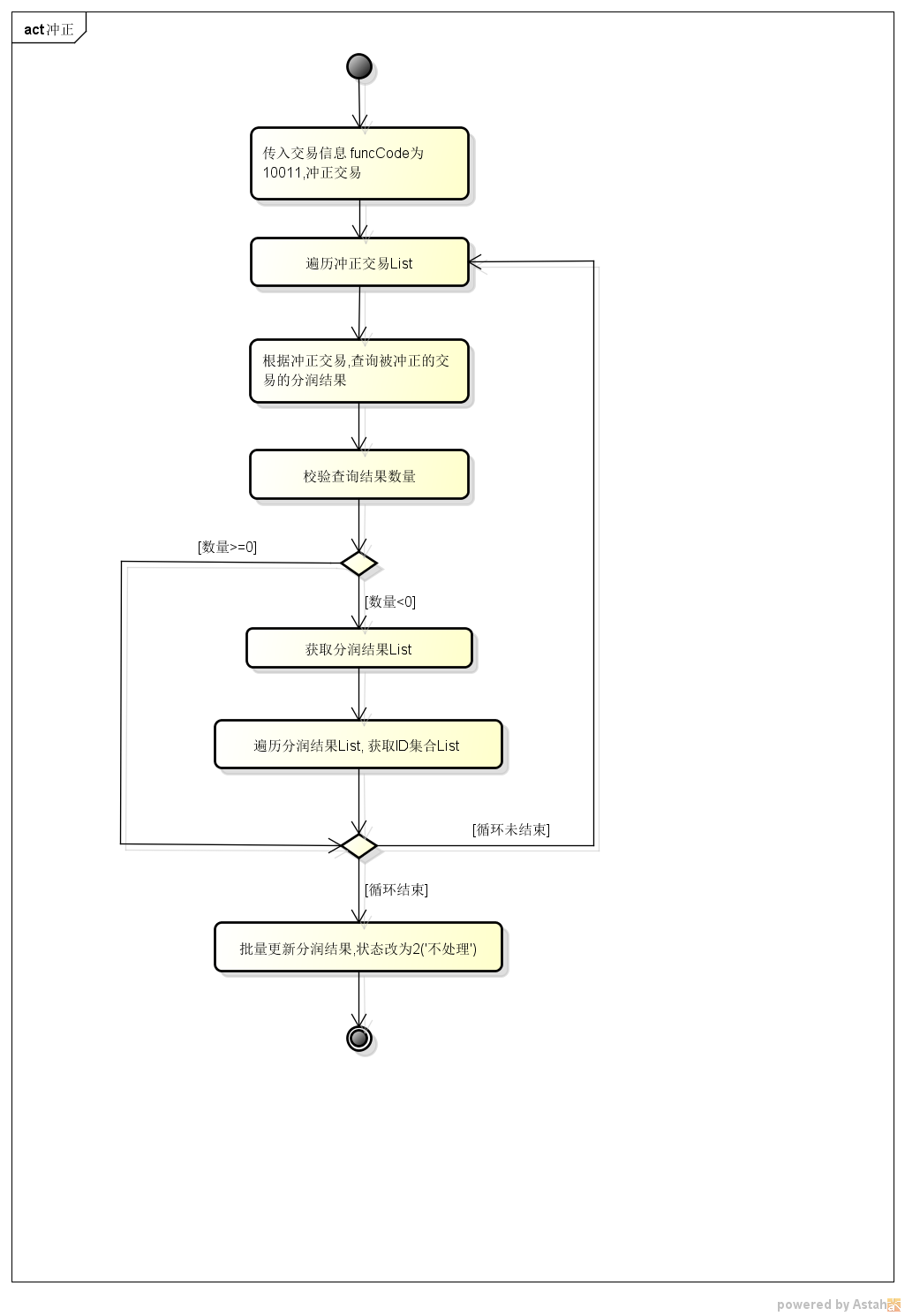


##### 3.6.1.5挂账

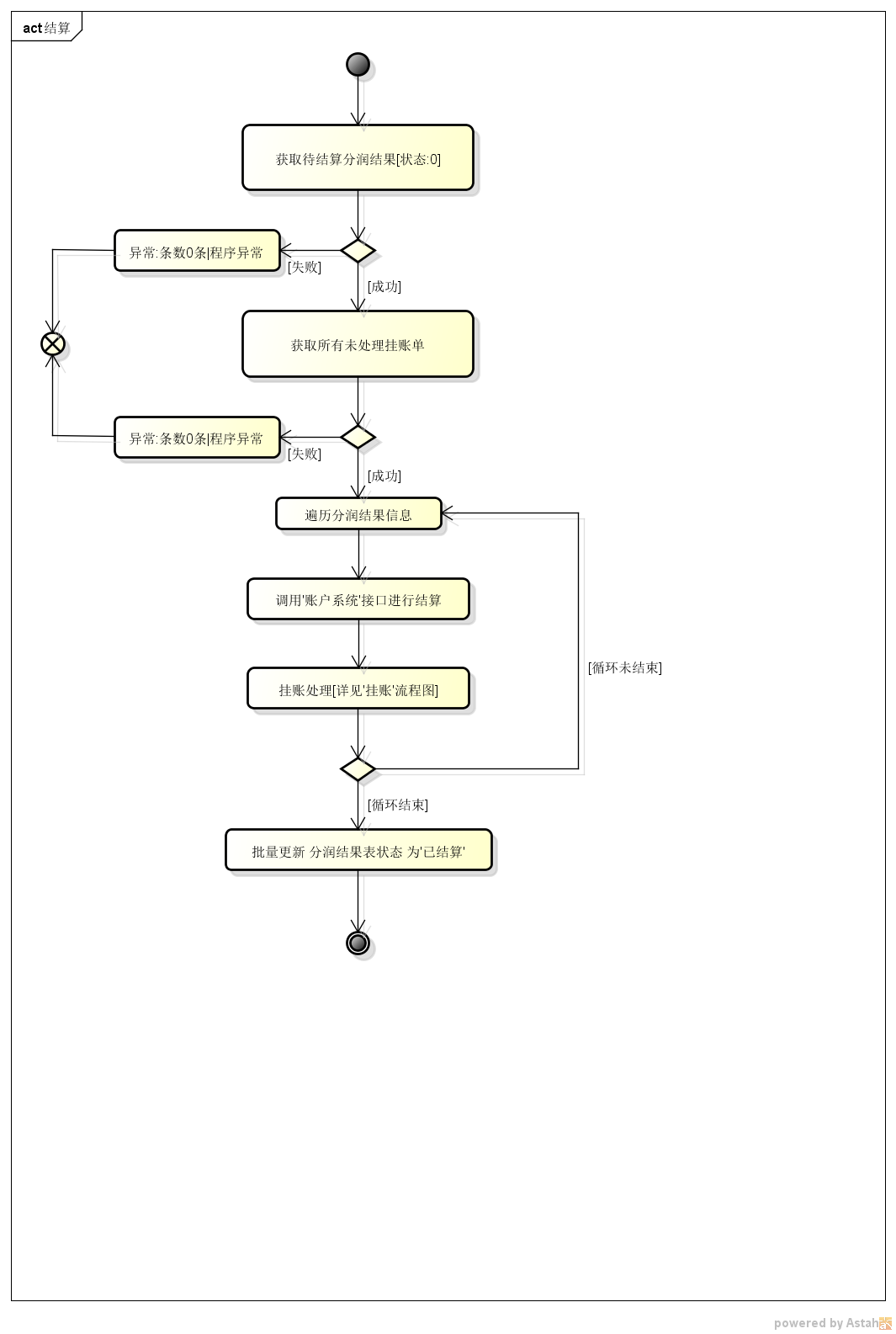


##### 3.6.1.6冲正

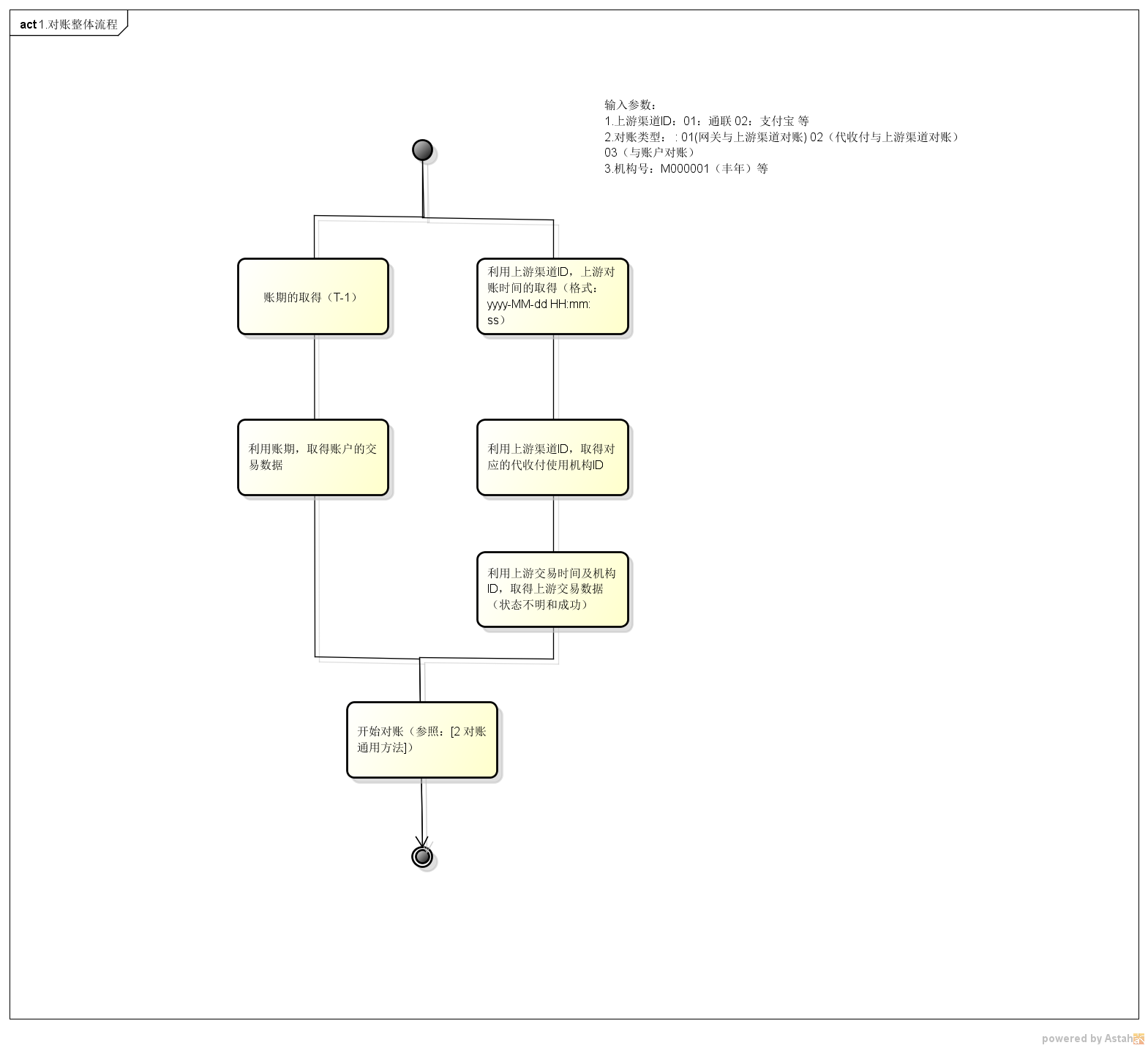
##### 



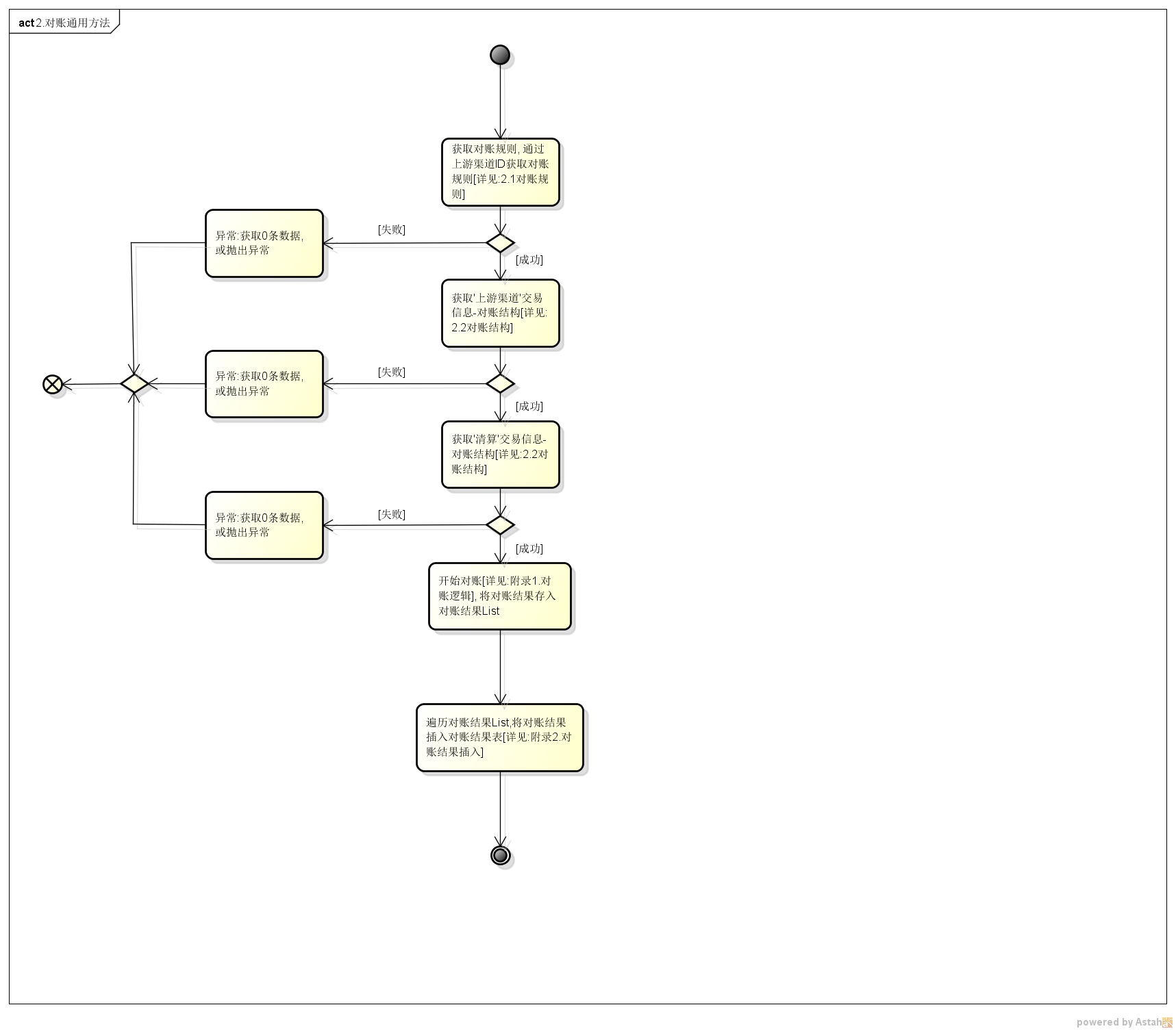
##### 3.6.1.7结算



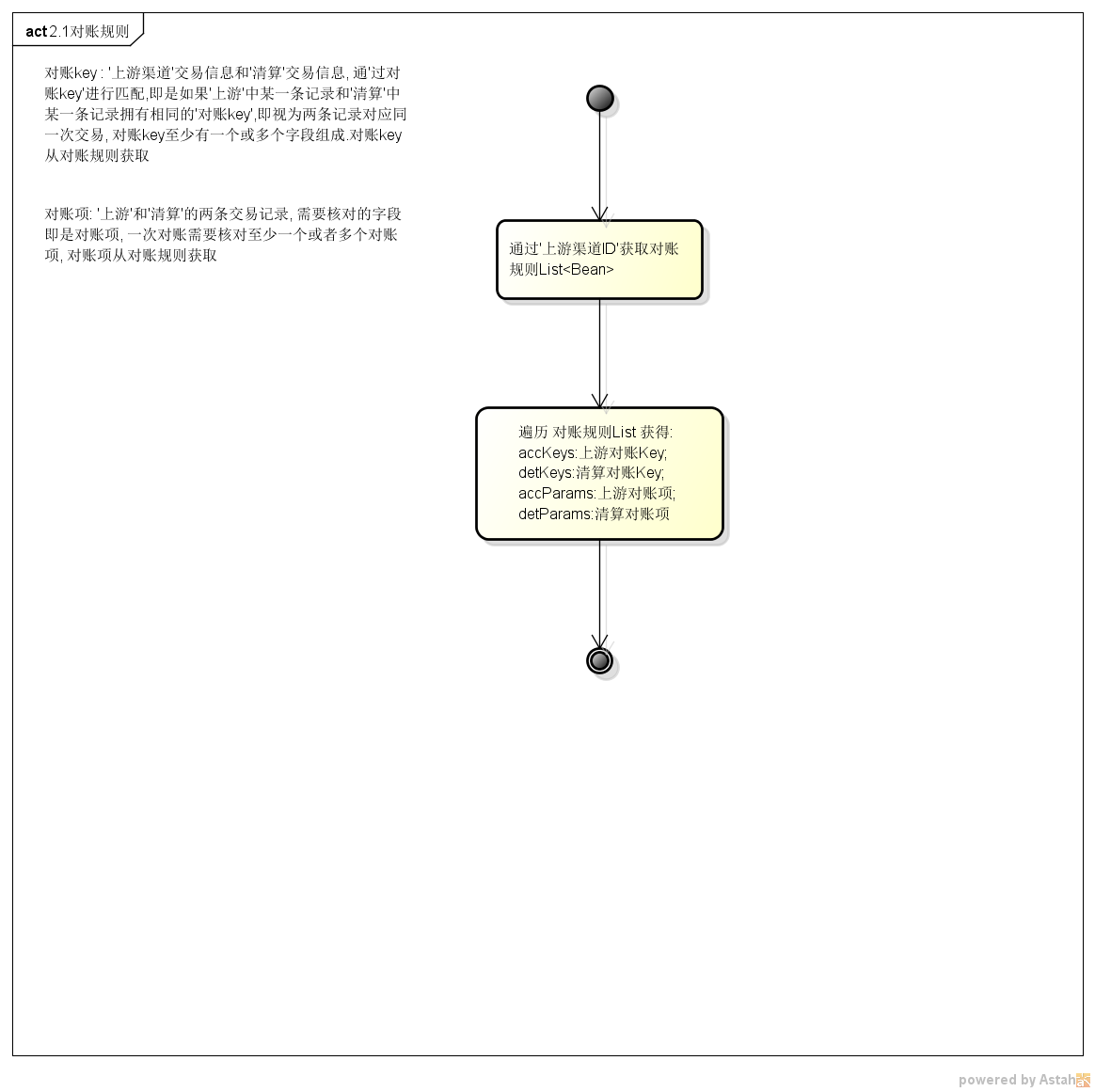
#### 3.6.2 对账



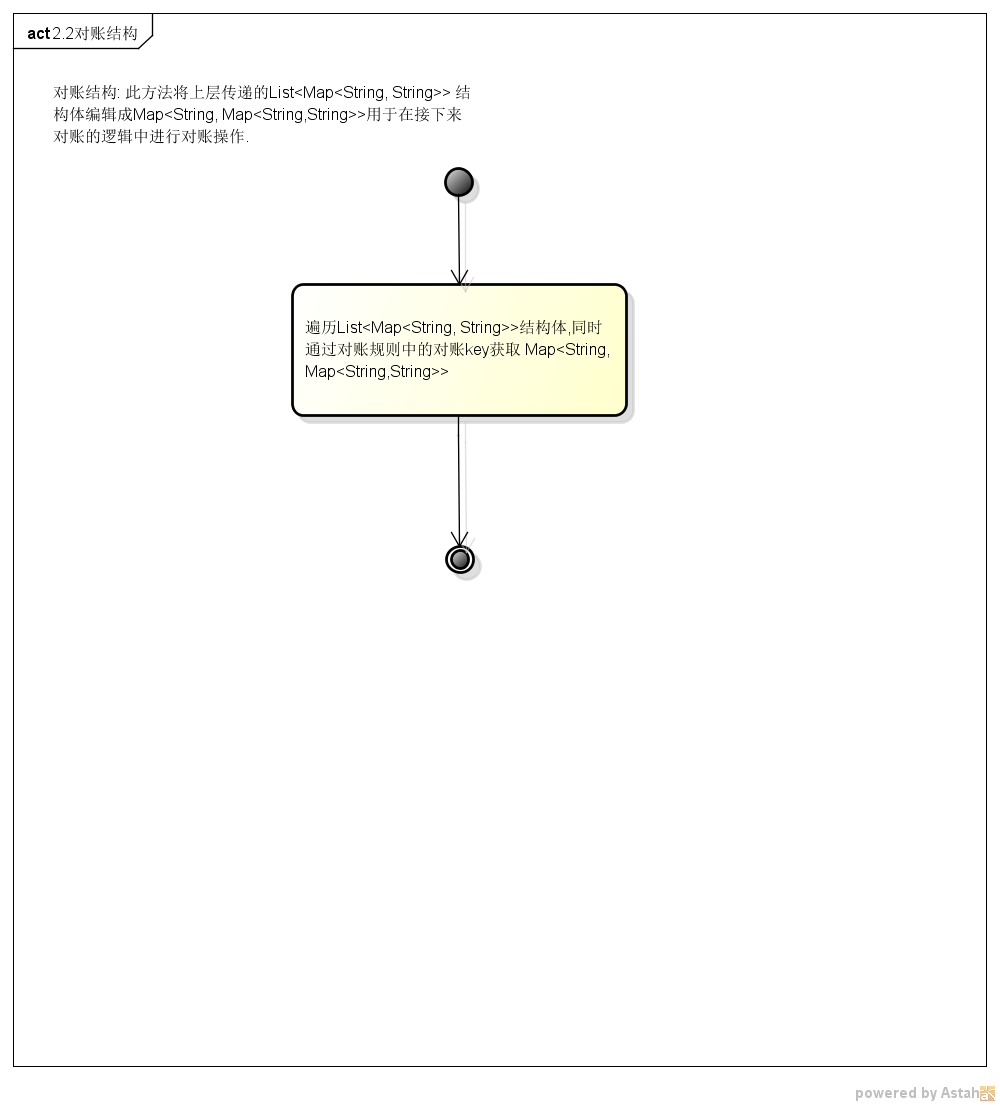
##### 3.6.2.1 对账（逻辑）



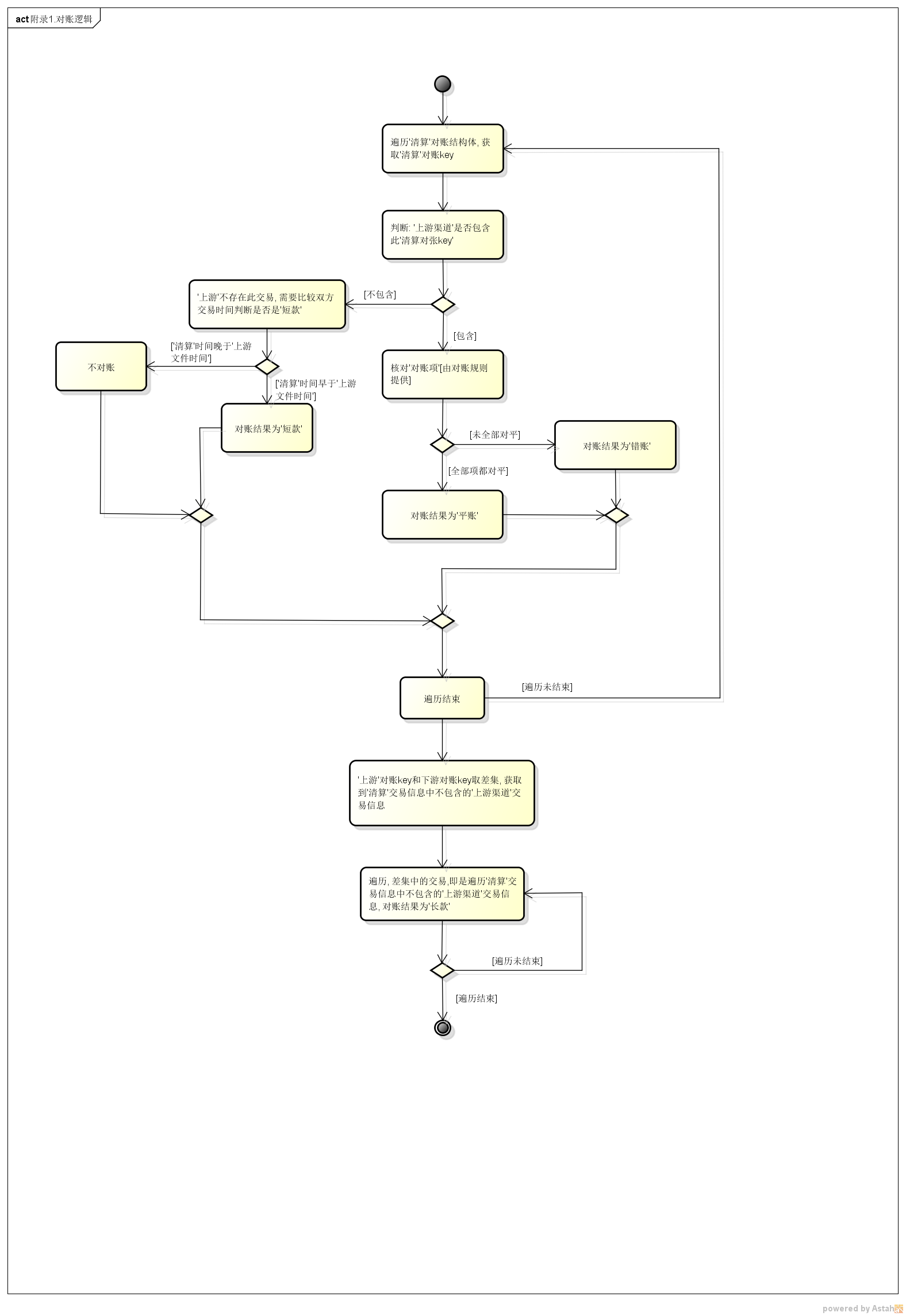
##### 3.6.2.2 对账规则



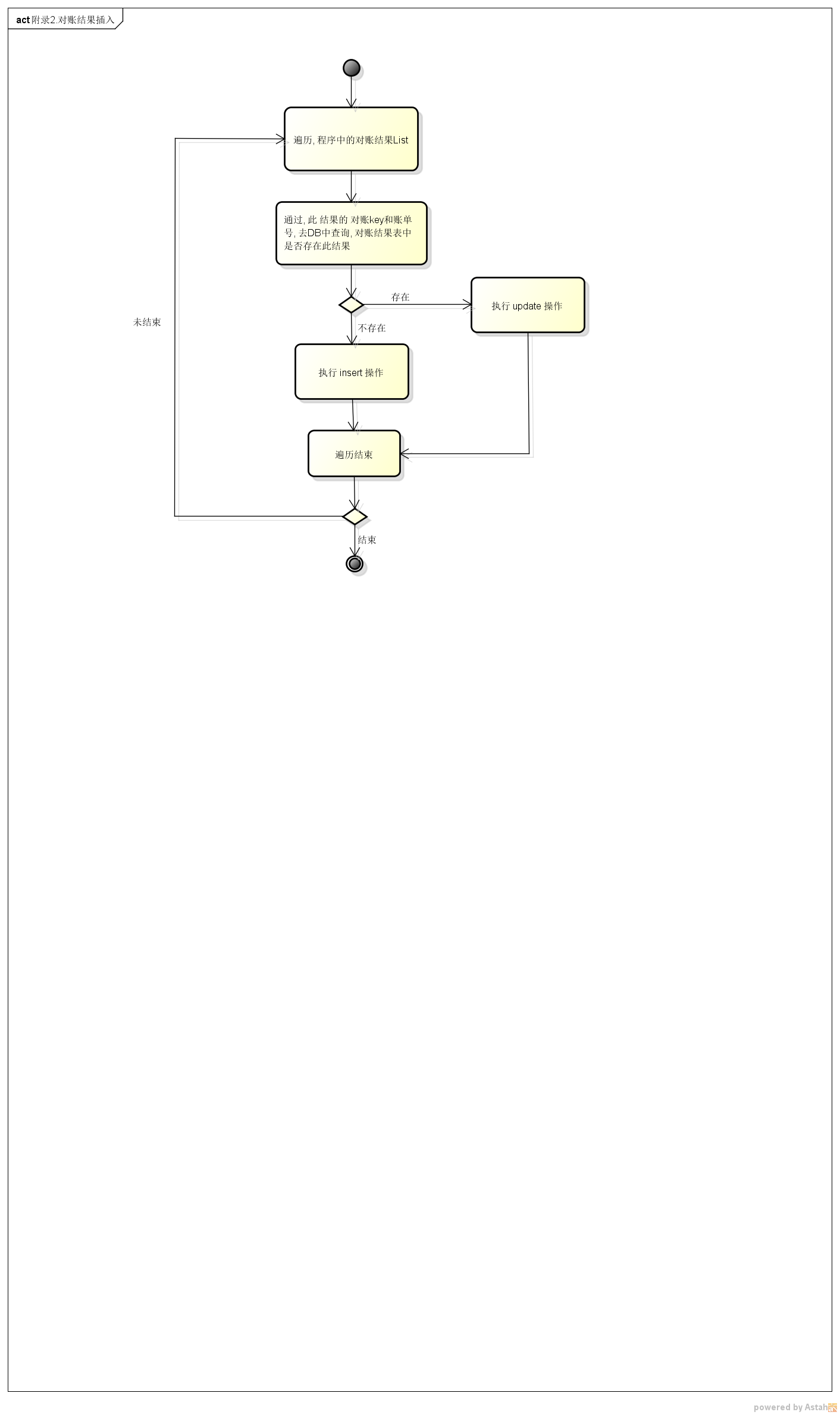
##### 3.6.2.3 对账结构



##### 3.6.2.4 对账业务逻辑

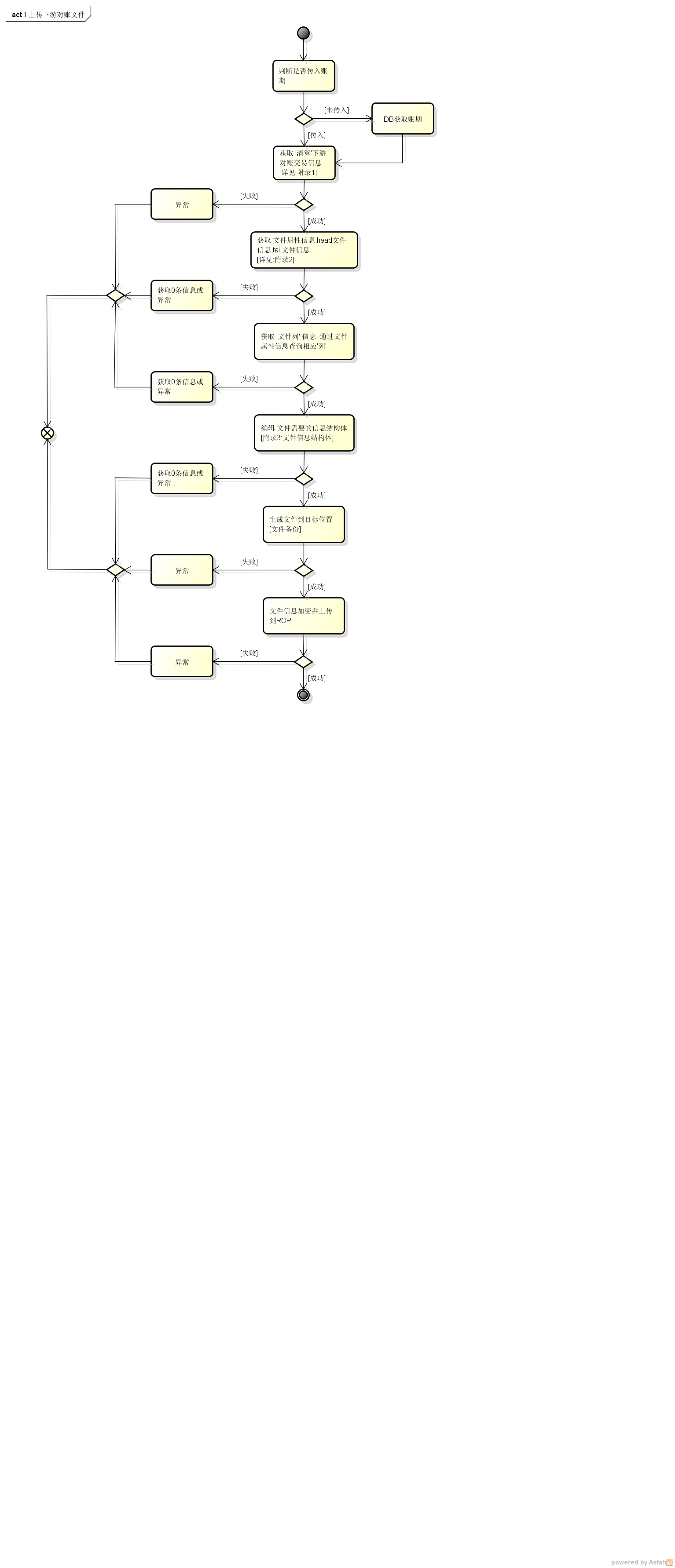


##### 3.6.2.4 对账结果插入

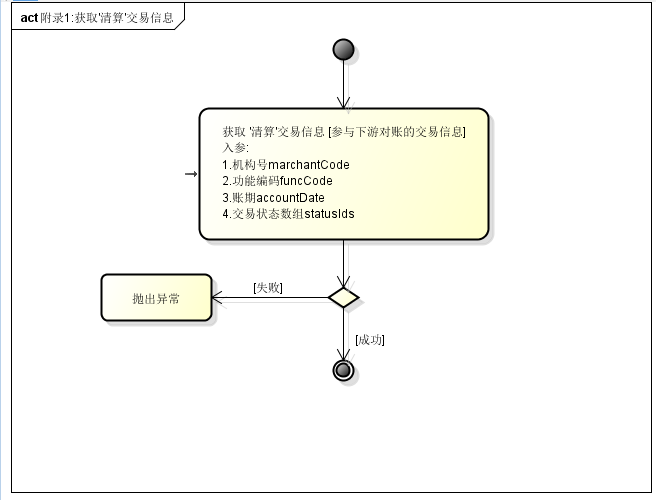


#### 3.6.3 上传下游对账文件

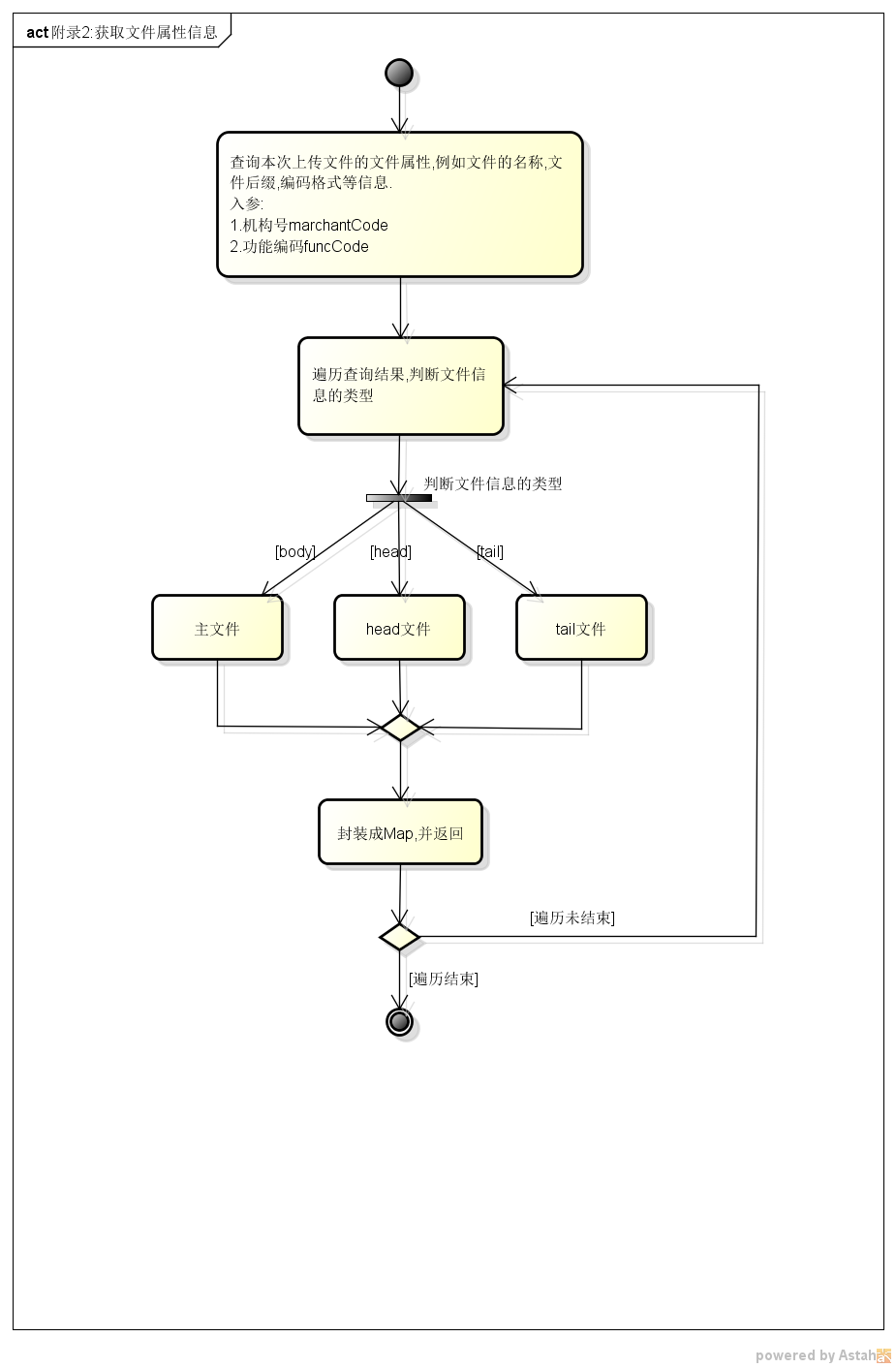
#### 



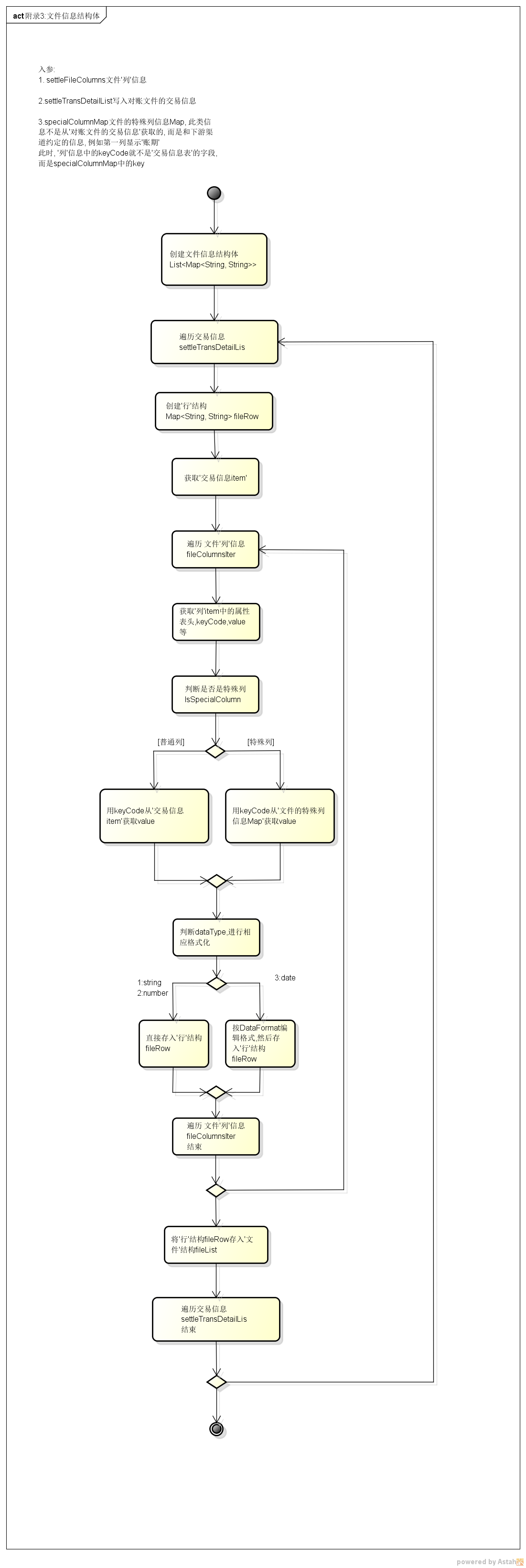
##### 3.6.3.1 查询清算交易信息



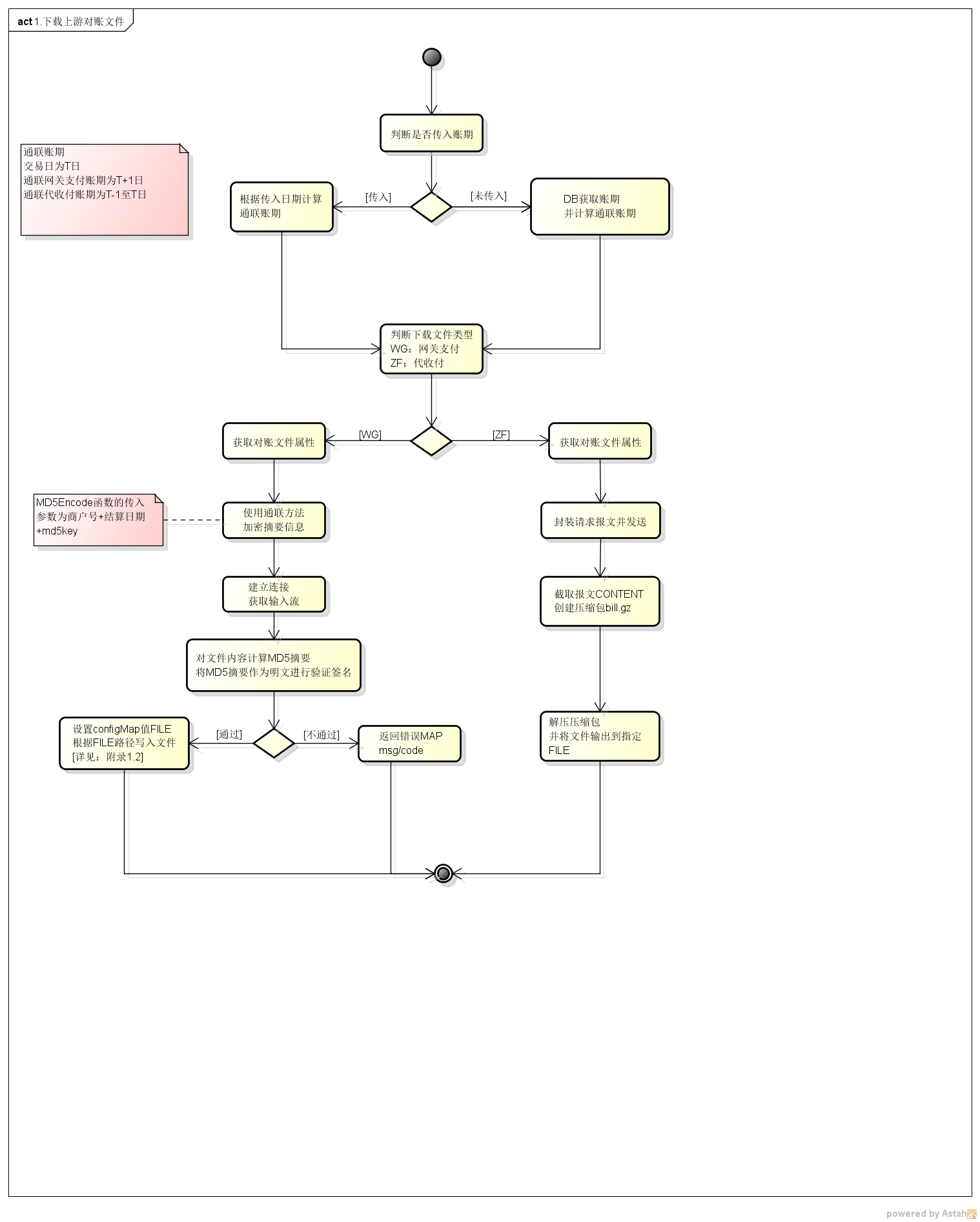
##### 3.6.3.2 查询文件属性信息



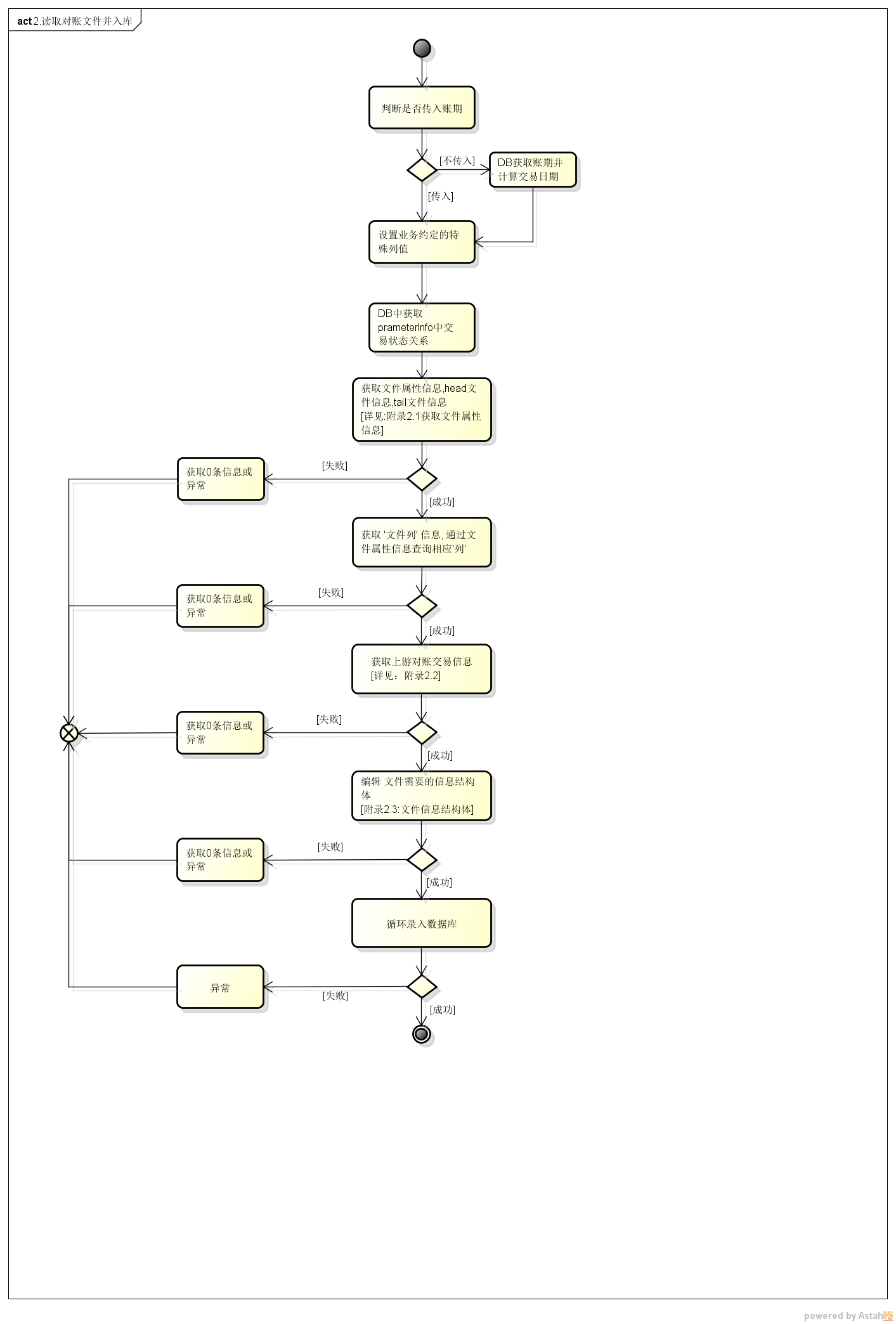
##### 3.6.3.3 封装文件信息结构体



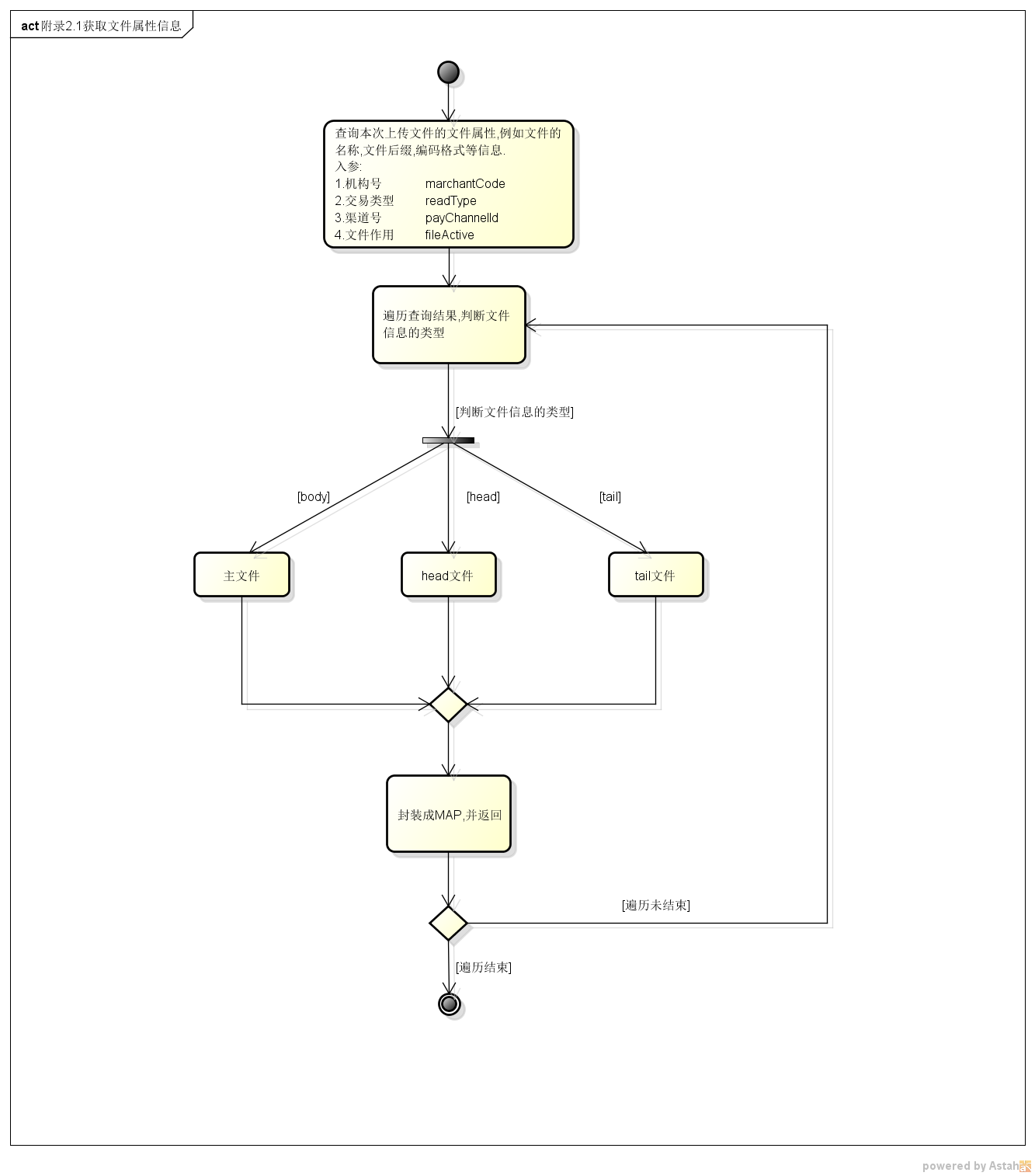
#### 3.6.4 下载上游对账文件



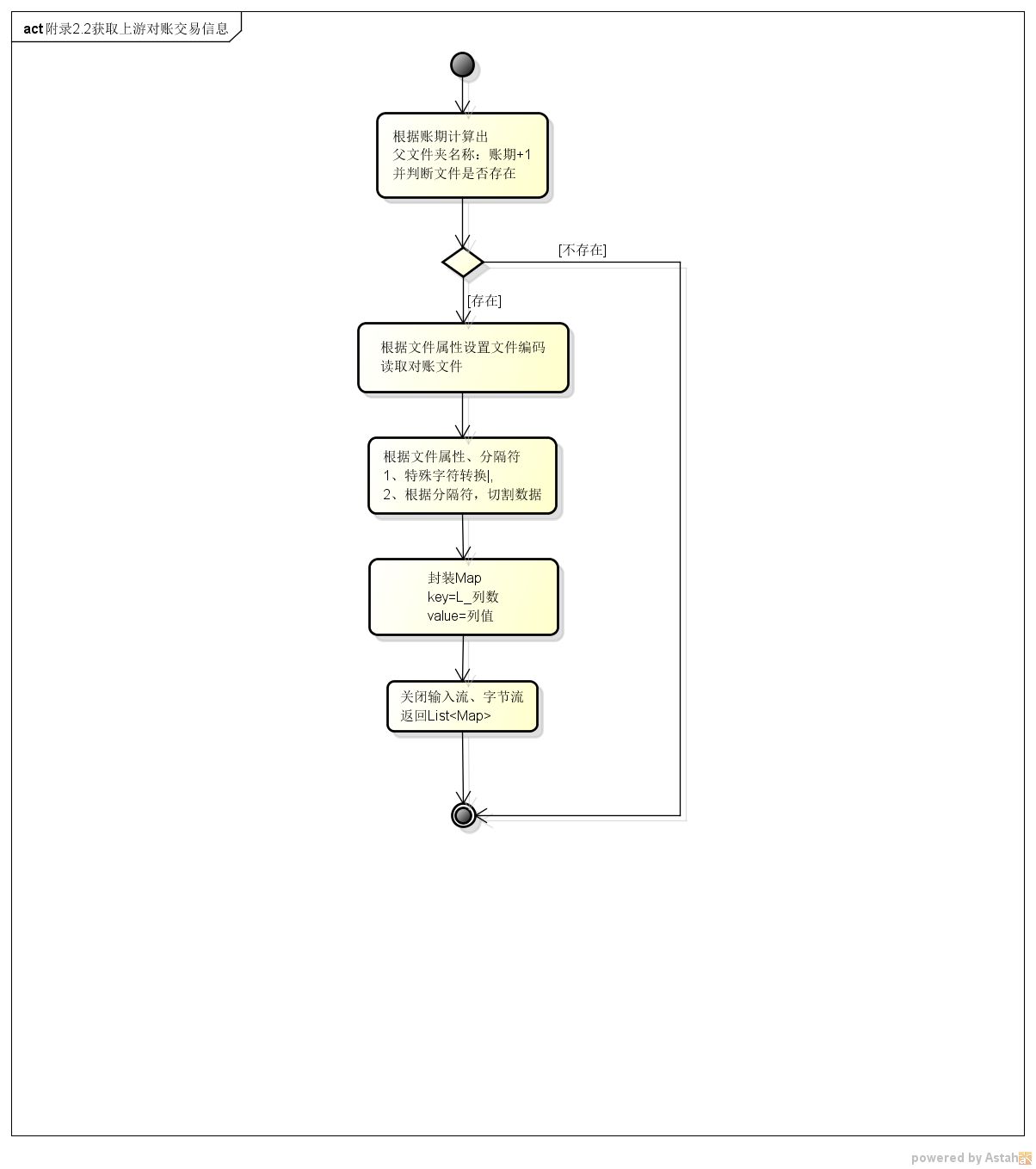
##### 3.6.4.1 读取对账文件并入库



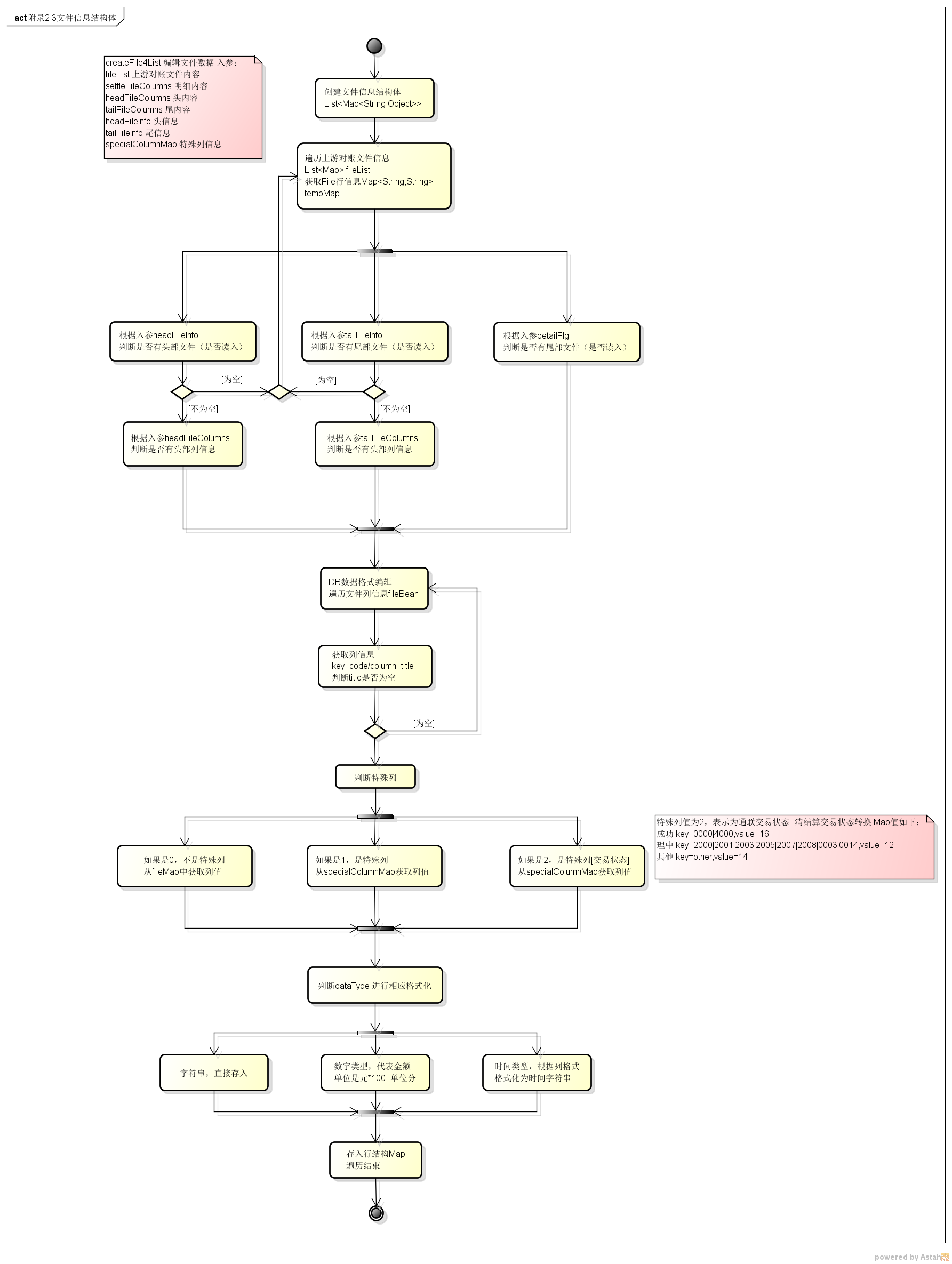
##### 3.6.4.2 获取文件属性信息



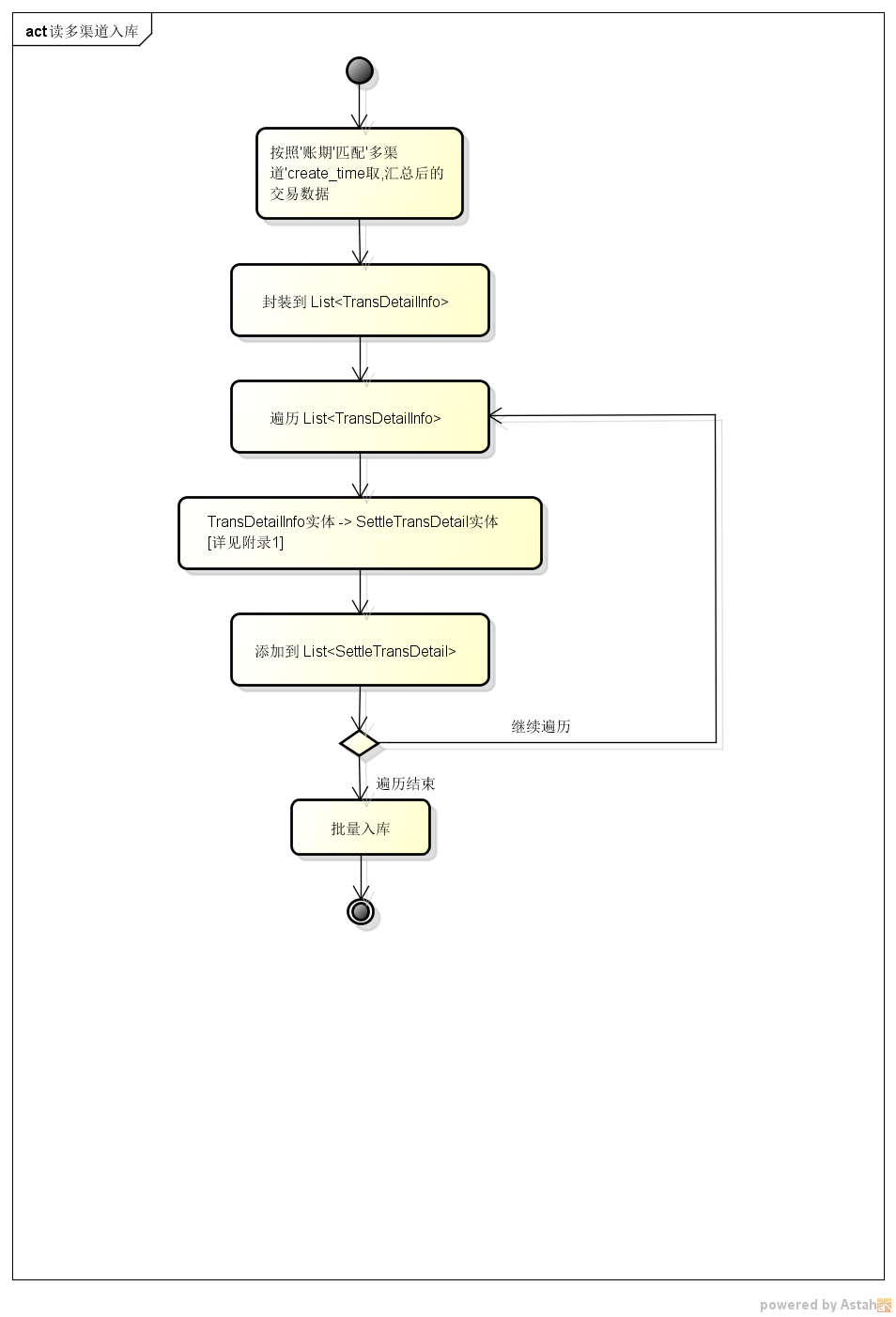
##### 3.6.4.3 获取上游对账交易信息



##### 3.6.4.4 文件信息结构体



#### 3.6.4 读取多渠道交易信息



## 3.7接口

#### 3.7.1存储账户系统交易信息接口

接口名: com.rkylin.settle.service.TradeService

方法: saveAccountTrans

入参: SettleTransDetail清算交易明细实体Bean

返回值: Map<String,String>运行结果

## 3.8存储分配

略

## 3.9限制条件

略

## 3.10测试要点

#### 3.10.1清算

##### [3.10.1.1获取账户系统交易信息](#_Toc430789170)

* **账户系统交易信息和清算系统交易信息的字段是否一一对应**
* **订单号位置对用关系(逻辑)**
* **金额位置对应关系(逻辑)**

##### [3.10.1.2冲正处理](#_Toc430789173)

* **冲正后，被冲正交易所关联的分润结果的状态应为2（不分润）**

##### [3.10.1.5挂账处理](#_Toc430789174)

* **挂账后，被挂账交易关联的分润结果和所产生的挂账信息数据需要一致**

##### [3.10.1.6清分](#_Toc430789175)

* **获取待清分的交易信息的funcCode必须是“字典表”指定的业务编码**
* **获取全部分润规则并存入内存，分润规则和分润明细必须是一对多对应关系**
* **清算结束后相关数据的状态位**

##### [3.10.1.7分润(结算)](#_Toc430789176)

* **结算后清算系统数据状态位**
* **账户系统更新是否正确**

#### 3.10.2对账

##### [3.10.2.1读取上游对账文件信息](#_Toc430789179)

* **读取上游对账文件,必须匹配到唯一的文件信息**
* **上游对账文件信息入库后字段的对应情况**

##### [3.10.2.2获取下游对账交易](#_Toc430789180)

* **下游对账文件信息入库后字段的对应情况**
* **funcCode 匹配情况**
* **statusId 匹配情况**

##### [3.10.2.3进行对账](#_Toc430789181)

* **对账的交易需要匹配唯一的对账规则**
* **参与对账的数据项必须全部对平，否则视为错账**
* **SETTLE\_TRANS\_ACCOUNT有，SETTLE\_TRANS\_DETAIL无，视为短款**
* **SETTLE\_TRANS\_ACCOUNT无，SETTLE\_TRANS\_DETAIL有，视为长款**
* **相关数据的更新情况**

##### [3.10.2.4上传下游对账文件](#_Toc430789182)

* **上传下游对账文件,必须匹配到唯一的文件信息**
* **文件中准备写入的交易信息必须是文件信息中配置的交易信息**
* **生成文件后，内容和格式是否正确**
* **上传文件后ROP服务器是否存在此文件**