(19)国家知识产权局

(12)发明专利

(10)授权公告号 CN 115686888 B
(45)授权公告日 2023.03.21

(21)申请号 202211714595.8
(22)申请日 2022.12.30
(65)同一申请的已公布的文献号
申请公布号 CN 115686888 A
(43)申请公布日 2023.02.03
(73)专利权人 浙江城云数字科技有限公司
地址 310000 浙江省杭州市萧山区北干街道博学路618号萧山科创中心
(72)发明人 刘仿 秦可 吴桐
(74)专利代理机构 杭州汇和信专利代理有限公司 33475
专利代理师 陈江
(51)Int.Cl.
G06F 9/54(2006.01)
G06F 9/448(2018.01)
G06F 8/36(2018.01)
(56)对比文件
CN 102523308 A,2012.06.27
CN 1573688 A,2005.02.02
审查员 金媛

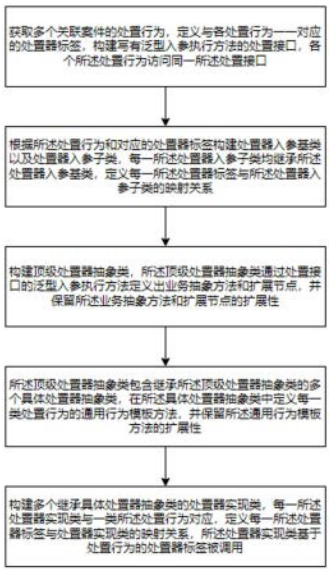
权利要求书2页 说明书11页 附图7页

(54)发明名称

一种基于规则的处置行为流程编排方法、装置及应用

(57)摘要

本申请提出了一种基于规则的处置行为流程编排方法、装置及应用,包括以下步骤:获取多个关联案件的处置行为,定义与各处置行为一一对应的处置器标签,根据所述处置器标签构建处置器入参基类以及处置器入参子类,每一所述处置器入参子类均继承所述处置器入参基类;构建一个顶级处置器抽象类,所述顶级处置器抽象类包含多个具体处置器抽象类,每一具体处置器抽象类包含多个处置器实现类;构建策略工厂,在策略工厂中定义处置器标签分别与处置器入参子类和处置器实现类的映射关系以用来实现对应的处置行为。本方案可以提高接口以及处置行为流程的扩展性,实现代码的高复用性,提高扩展开发性,且还可实现编排多个处置行为进行自动执行。



CN 115686888 B

1.一种基于规则的处置行为流程编排方法,其特征在于,包括以下步骤:

获取多个关联案件的处置行为,定义与各处置行为一一对应的处置器标签,构建写有泛型入参执行方法的处置接口,各个所述处置行为访问同一所述处置接口;

根据所述处置行为和对应的处置器标签构建处置器入参基类以及处置器入参子类,每一所述处置器入参子类均继承所述处置器入参基类,定义每一所述处置器标签与所述处置器入参子类的映射关系;

构建顶级处置器抽象类,所述顶级处置器抽象类通过处置接口的泛型入参执行方法定义出业务抽象方法和扩展节点,并保留所述业务抽象方法和扩展节点的扩展性;

所述顶级处置器抽象类包含继承所述顶级处置器抽象类的多个具体处置器抽象类,在所述具体处置器抽象类中定义每一类处置行为的通用行为模板方法,并保留所述通用行为模板方法的扩展性;

构建多个继承具体处置器抽象类的处置器实现类,每一所述处置器实现类与一类所述处置行为对应,定义每一所述处置器标签与处置器实现类的映射关系,所述处置器实现类基于处置行为的处置器标签被调用。

2.基于权利要求1所述的一种基于规则的处置行为流程编排方法,其特征在于,所述处置器标签具有唯一性,所述处置接口采用Map结构来接收处置行为的参数。

3.基于权利要求1所述的一种基于规则的处置行为流程编排方法,其特征在于,所述处置器入参基类抽取所述处置行为的公共参数,所述公共参数包含处置器标签以及当前处置行为归属案件的案件编号,每一所述处置器入参子类包含一类处置行为特有的参数,所述处置器入参基类有且只有一个,所述处置器入参子类的数量与所述处置行为的数量相同。

4.基于权利要求1所述的一种基于规则的处置行为流程编排方法,其特征在于,所述顶级处置器抽象类根据所述泛型入参执行方法定义处置器执行流程,所述处置器执行流程包括执行处置行为的前置执行、业务抽象方法和后置执行,并保留每一所述处置器执行流程节点的扩展性。

5.基于权利要求1所述的一种基于规则的处置行为流程编排方法,其特征在于,在所述顶级处置器抽象类中定义一个标签变量,所述标签变量用来存取所述处置器标签,当所述顶级处置器抽象类中定义的业务抽象方法执行完毕后,判断所述标签变量中是否存在下一处置行为所需的处置器标签,若存在则根据所述处置器标签执行对应处置行为。

6.基于权利要求1所述的一种基于规则的处置行为流程编排方法,其特征在于,在“在所述具体处置器抽象类中定义每一类处置行为的通用行为模板方法”步骤中,所述具体处置器抽象类继承所述顶级处置器抽象类中的业务抽象方法,并重写顶级处置器抽象类中的业务抽象方法得到多个处置行为的通用行为模板方法,所述通用行为模板方法包括多个处置行为的共同点。

7.基于权利要求1所述的一种基于规则的处置行为流程编排方法,其特征在于,在“构建多个继承具体处置器抽象类的处置器实现类,每一所述处置器实现类与一类所述处置行为对应”步骤中,所述处置器实现类根据具体处置器抽象类的扩展性进行扩展得到,在所述处置器实现类中定义了对应处置行为的至少一处置方法。

8.基于权利要求1所述的一种基于规则的处置行为流程编排方法,其特征在于,定义一个处置器策略工厂,所述处置器策略工厂使用两个Map结构分别维护所述处置器标签与处

置器入参子类的映射关系以及所述处置器标签与所述处置器实现类的映射关系，并将所述映射关系保存到内存中。

9.一种对基于规则的处置行为流程进行扩展的方法，其特征在于，包括：

当权利要求1-8所述的一种基于规则的处置行为流程编排方法需要扩展新的处置行为时，定义所述处置行为的处置器实现类，将所述处置器实现类继承于符合其通用行为的具体处置器抽象类，通过实现和重写所述具体处置器抽象类中定义的通用行为模板方法来实现所述处置行为的业务，完成新的处置行为的扩展。

10.一种基于规则的处置行为流程编排装置，其特征在于，包括：

获取模块：获取多个关联案件的处置行为，定义与各处置行为一一对应的处置器标签，构建写有泛型入参执行方法的处置接口，各个所述处置行为访问同一所述处置接口；

第一构建模块：根据所述处置行为和对应的处置器标签构建处置器入参基类以及处置器入参子类，每一所述处置器入参子类均继承所述处置器入参基类，定义每一所述处置器标签与所述处置器入参子类的映射关系；

第二构建模块：构建顶级处置器抽象类，所述顶级处置器抽象类通过处置接口的泛型入参执行方法定义出业务抽象方法和扩展节点，并保留所述业务抽象方法和扩展节点的扩展性；

定义模块：所述顶级处置器抽象类包含继承所述顶级处置器抽象类的多个具体处置器抽象类，在所述具体处置器抽象类中定义每一类处置行为的通用行为模板方法，并保留所述通用行为模板方法的扩展性；

执行模块：构建多个继承具体处置器抽象类的处置器实现类，每一所述处置器实现类与一类所述处置行为对应，定义每一所述处置器标签与处置器实现类的映射关系，所述处置器实现类基于处置行为的处置器标签被调用。

11.一种电子装置，包括存储器和处理器，其特征在于，所述存储器中存储有计算机程序，所述处理器被设置为运行所述计算机程序以执行权利要求1-8任一所述的一种基于规则的处置行为流程编排方法或权利要求9所述的一种对基于规则的处置行为流程进行扩展的方法。

12.一种可读存储介质，其特征在于，所述可读存储介质中存储有计算机程序，所述计算机程序包括用于控制过程以执行过程的程序代码，所述过程包括权利要求1-8任一所述的一种基于规则的处置行为流程编排方法或权利要求9所述的一种对基于规则的处置行为流程进行扩展的方法。

一种基于规则的处置行为流程编排方法、装置及应用

技术领域

[0001] 本申请涉及计算机软件开发领域，特别是涉及一种基于规则的处置行为流程编排方法、装置及应用。

背景技术

[0002] 城市管理场景中的案件执法流程复杂多变，且需要跨部门协作，完整的完成一个案件的流程需要耗费大量的人力和时间，在实际的场景中，当存在对业务理解不熟或操作不规范的执法人员时可能会出现案件的错误处理情况，从而造成下一环节职责部门的审核、退回等操作，导致一个案件的人力资源消耗十分严重。因此，通过算法来进行案件执法流程的智能化成为降低人力资源消耗和提升用户体验的重要方式。

[0003] 基于城市管理场景中的案件执法流程存在上报、受理、派遣部门、派遣个人、处置、审核、复核、办结等诸多环节，各环节又对应多种处置行为。对于使用人员来说，会根据当地的行政法规和人员职责情况，通过多个职责部门和执法人员的相互协作来完成一个案件的执法流程；对于研发人员来说，在项目落地时会根据当地的行政法规和人员职责情况进行案件执法流程的定制化，再根据实际的案件执法流程进行定制开发。

[0004] 对于研发人员来说，在针对不同的处置执行接口进行研发时，前后端人员需要逐个对处置行为接口进行联调对接，接口无法统一从而导致研发成本增加；而且当需要增加处置行为时，需要研发新的处置行为接口并再次进行联调对接，处置行为接口没有扩展性；当其中一处置行为的需求发生变更时，仅仅分析辨识出本次修改在全局中对其他模块的影响性就会耗费大量时间，而且受到影响的每个处置行为接口都需要进行对应修改并与前端重新联调，这是由于整个执法流程的扩展性不足所导致的；当满足一定规则时需要编排派遣部门、派遣个人处置行为执行时，必须添加满足特定需求的处置行为接口才可以，无法进行自动化编排。

[0005] 综上所述，可以发现目前的针对处置行为流程的编排上存在不同处置行为的接口不统一、处置行为接口没有扩展性、执法流程代码复用度低、扩展性差、执法处理流程无法编排多个处置行为自动执行等诸多问题。

发明内容

[0006] 本申请实施例提供了一种基于规则的处置行为流程编排方法、装置及应用，可以将多个处置行为的接口进行统一，且提高接口以及处置行为流程的扩展性，实现代码的高复用性，提高扩展开发性，且还可实现编排多个处置行为进行自动执行。

[0007] 第一方面，本申请实施例提供了一种基于规则的处置行为流程编排方法，所述方法包括：

[0008] 获取多个关联案件的处置行为，定义与各处置行为一一对应的处置器标签，构建写有泛型入参执行方法的处置接口，各个所述处置行为访问同一所述处置接口；

[0009] 根据所述处置行为和对应的处置器标签构建处置器入参基类以及处置器入参子

类,每一所述处置器入参子类均继承所述处置器入参基类,定义每一所述处置器标签与所述处置器入参子类的映射关系;

[0010] 构建顶级处置器抽象类,所述顶级处置器抽象类通过处置接口的泛型入参执行方法定义出业务抽象方法和扩展节点,并保留所述业务抽象方法和扩展节点的扩展性;

[0011] 所述顶级处置器抽象类包含继承所述顶级处置器抽象类的多个具体处置器抽象类,在所述具体处置器抽象类中定义每一类处置行为的通用行为模板方法,并保留所述通用行为模板方法的扩展性;

[0012] 构建多个继承具体处置器抽象类的处置器实现类,每一所述处置器实现类与一类所述处置行为对应,定义每一所述处置器标签与处置器实现类的映射关系,所述处置器实现类基于处置行为的处置器标签被调用。

[0013] 第二方面,本申请实施例提供了一种基于规则的处置行为流程编排装置,包括:

[0014] 获取模块:获取多个关联案件的处置行为,定义与各处置行为一一对应的处置器标签,构建写有泛型入参执行方法的处置接口,各个所述处置行为访问同一所述处置接口;

[0015] 第一构建模块:根据所述处置行为和对应的处置器标签构建处置器入参基类以及处置器入参子类,每一所述处置器入参子类均继承所述处置器入参基类,定义每一所述处置器标签与所述处置器入参子类的映射关系;

[0016] 第二构建模块:构建顶级处置器抽象类,所述顶级处置器抽象类通过处置接口的泛型入参执行方法定义出业务抽象方法和扩展节点,并保留所述业务抽象方法和扩展节点的扩展性;

[0017] 定义模块:所述顶级处置器抽象类包含继承所述顶级处置器抽象类的多个具体处置器抽象类,在所述具体处置器抽象类中定义每一类处置行为的通用行为模板方法,并保留所述通用行为模板方法的扩展性;

[0018] 执行模块:构建多个继承具体处置器抽象类的处置器实现类,每一所述处置器实现类与一类所述处置行为对应,定义每一所述处置器标签与处置器实现类的映射关系,所述处置器实现类基于处置行为的处置器标签被调用。

[0019] 第三方面,本申请实施例提供了一种对基于规则的处置行为流程进行扩展的方法,包括:

[0020] 当编排好一个完整的处置流程,且需要扩展新的处置行为时,定义所述处置行为的处置器实现类,将所述处置器实现类继承于符合其通用行为的具体处置器抽象类,通过实现和重写所述具体处置器抽象类中定义的通用行为模板方法来实现所述处置行为的业务,完成新的处置行为的扩展。

[0021] 第四方面,本申请实施例提供了一种电子装置,包括存储器和处理器,所述存储器中存储有计算机程序,所述处理器被设置为运行所述计算机程序以执行一种基于规则的处置行为流程编排方法或一种对基于规则的处置行为流程进行扩展的方法。

[0022] 第五方面,本申请实施例提供了一种可读存储介质,所述可读存储介质中存储有计算机程序,所述计算机程序包括用于控制过程以执行过程的程序代码,所述过程包括一种基于规则的处置行为流程编排方法或一种对基于规则的处置行为流程进行扩展的方法。

[0023] 本发明的主要贡献和创新点如下:本方案采用接口泛化的调用方式,将多个处置行为接口统一通过一个入口进行接口调用,简化了调用方法,且采用map结构来接收参数使

用户无需关心入参类的实体接口定义，只需将需要的参数传入即可；本方案通过在顶级处置器抽象类中定义处置行为的抽象方法，再由具体处置器抽象类来对所述抽象方法以通用行为模板方法形式进行编写，并保留其扩展性，在有新的抽象方法或现有的抽象方法进行改动时可以直接进行扩展，在现有的抽象方法进行变更时需要修改通用行为模板方法的内部逻辑，当添加新的抽象方法时也只需找到合适的扩展点进行开发，扩展开放且代码复用性高；本方案在顶级处置器抽象类中定义一个标签变量，所述标签变量用来存取每一处置行为对应的处置器标签，当前的处置行为执行完毕后，若判断所述标签变量中存在下一个处置行为的处置器标签时则自动执行下一个处置行为，依次类推可以形成一条责任链，这样就可以做到通过已有的处置行为组合出复杂多变的执法流程从而进行自动化完成。

[0024] 本申请的一个或多个实施例的细节在以下附图和描述中提出，以使本申请的其他特征、目的和优点更加简明易懂。

附图说明

[0025] 此处所说明的附图用来提供对本申请的进一步理解，构成本申请的一部分，本申请的示意性实施例及其说明用于解释本申请，并不构成对本申请的不当限定。在附图中：

[0026] 图1是根据本申请实施例的一种基于规则的处置行为流程编排方法的流程图；

[0027] 图2是根据本申请实施例的一种某地执法流程的流程示意图；

[0028] 图3是根据本申请实施例中的将处置行为接口进行统一的示意图；

[0029] 图4是根据本申请实施例中的通过泛化调用的方式将处置器标签实例化为处置器入参子类的流程示意图；

[0030] 图5是根据本申请实施例中的处置器入参基类与处置器入参子类的包含参数的结构示意图；

[0031] 图6是根据本申请实施例中的处置器入参基类与处置器入参子类的结构示意图；

[0032] 图7是根据本申请实施例中的顶级处置器抽象类的结构示意图；

[0033] 图8是根据本申请实施例中的顶级处置器抽象类与具体处置器抽象类之间关系的示意图；

[0034] 图9是根据本申请实施例的一种基于规则编排的执行处置装置的结构框图；

[0035] 图10是根据本申请实施例的电子装置的硬件结构示意图。

具体实施方式

[0036] 这里将详细地对示例性实施例进行说明，其示例表示在附图中。下面的描述涉及附图时，除非另有表示，不同附图中的相同数字表示相同或相似的要素。以下示例性实施例中所描述的实施方式并不代表与本说明书一个或多个实施例相一致的所有实施方式。相反，它们仅是与如所附权利要求书中所详述的、本说明书一个或多个实施例的一些方面相一致的装置和方法的例子。

[0037] 需要说明的是：在其他实施例中并不一定按照本说明书示出和描述的顺序来执行相应方法的步骤。在一些其他实施例中，其方法所包括的步骤可以比本说明书所描述的更多或更少。此外，本说明书中所描述的单个步骤，在其他实施例中可能被分解为多个步骤进行描述；而本说明书中所描述的多个步骤，在其他实施例中也可能被合并为单个步骤进行

描述。

[0038] 实施例一

[0039] 本申请实施例提供了一种基于规则的处置行为流程编排方法,参考图1,所述方法包括:

[0040] 获取多个关联案件的处置行为,定义与各处置行为一一对应的处置器标签,构建写有泛型入参执行方法的处置接口,各个所述处置行为访问同一所述处置接口;

[0041] 根据所述处置行为和对应的处置器标签构建处置器入参基类以及处置器入参子类,每一所述处置器入参子类均继承所述处置器入参基类,定义每一所述处置器标签与所述处置器入参子类的映射关系;

[0042] 构建顶级处置器抽象类,所述顶级处置器抽象类通过处置接口的泛型入参执行方法定义出业务抽象方法和扩展节点,并保留所述业务抽象方法和扩展节点的扩展性;

[0043] 所述顶级处置器抽象类包含继承所述顶级处置器抽象类的多个具体处置器抽象类,在所述具体处置器抽象类中定义每一类处置行为的通用行为模板方法,并保留所述通用行为模板方法的扩展性;

[0044] 构建多个继承具体处置器抽象类的处置器实现类,每一所述处置器实现类与一类所述处置行为对应,定义每一所述处置器标签与处置器实现类的映射关系,所述处置器实现类基于处置行为的处置器标签被调用。

[0045] 在一些实施例中,所述处置器标签具有唯一性,所述处置接口采用Map结构来接收处置行为的参数。

[0046] 具体的,将所述处置接口改为泛化调用方式后,可以将各个处置行为接口统一为一个接口请求地址,简化了调用方式,且本方案采用Map结构来接收参数,使得客户端无需关心入参类的实体结构定义,只需将需要的参数传入即可。

[0047] 具体的,对所有处置行为的接口进行统一可以避免在传统模式下需要针对不同的处置结果定制化开发多个处置接口的问题。

[0048] 在一些具体实施例中,某地的执法流程分为上报、处理、审核、结案四个处置行为,如图2所示,为每一处置行为定义一个标签tag,上报行为的tag为REPORT,处理行为的tag为HANDLE,审核行为的tag为CHECK,结案行为的tag为CLOSE,并将四个处置行为的接口统一为一个接口地址“http(s)://ip:port/context/handle”,如图3所示。

[0049] 具体的,在前端对所述处置接口进行请求时传入对应的tag入参。

[0050] 具体的,所述泛型入参执行方法为将参数类型进行参数化,当在实际调用时在确定该参数是什么具体类型,当使用泛型入参执行方法可以让多种不同的数据类型执行相同的代码,使代码的扩展性更强。

[0051] 在一些实施例中,所述处置器入参基类抽取所述处置行为的公共参数,所述公共参数包含处置器标签以及当前处置行为归属案件的案件编号,每一所述处置器入参子类包含一类处置行为特有的参数,所述处置器入参基类有且只有一个,所述处置器入参子类的数量与所述处置行为的数量相同。

[0052] 具体的,所述公共参数除了包含所述处置器标签以及当前处置行为归属案件的案件编号外,可根据案件的实际情况进行设定,本方案在此不进行具体限定。

[0053] 具体的,如图4所示,通过泛化调用的方式将入参的处置器标签根据所述处置器入

参基类中的公共参数实例化为所述处置器入参子类。

[0054] 在一些具体实施例中，如图5所示，BaseEventHandleCmd为所述处置器入参基类，所述处置器入参基类中的公共参数为tag(处置器标签)以及caseID(案件编号)，ReportCmd为上报行为的处置器入参子类，其特有的参数为reportAddr(上报地址)，HandleCmd为处理行为的处置器入参子类，其特有的参数为handleResult(处理结果)，CheckCmd为审核行为的处置器入参子类，其特有的参数为checkResult(审核结果)，CloseCmd为结案行为的处置器入参子类，其特有的参数为closeReason(结案原因)。

[0055] 具体的，所述处置器入参子类直接或间接继承所述处置器入参基类，每一处置行为所对应的处置器入参子类可包含更低维度的处置器入参子类，低维度的处置器入参子类继承至高维度的处置器入参子类，如图6所示，上报行为的处置器入参子类包含更低维度的AI上报行为的处置器入参子类以及移动上报行为的处置器入参子类。

[0056] 在一些实施例中，所述顶级处置器抽象类根据所述泛型入参执行方法定义处置器执行流程，所述处置器执行流程包括执行处置行为的前置执行、业务抽象方法和后置执行，并保留每一所述处置器执行流程节点的扩展性。

[0057] 在一些具体实施例中，如图7所示，定义顶级处置器抽象类AbstractEventHandler，其通过实现处置接口中的执行方法handle(T parameter)定义出处置器执行流程为：前置执行beforeExecutor()、执行方法executor(T parameter)，后置执行afterExecutor()，另外，所述顶级处置器抽象类具备私有方法executeSuccessor()和clear()且持有EVENT_CONTEXT及successorTag属性。

[0058] 具体的，execute(T parameter)为执行处置行为的业务抽象方法，beforeExecutor()以及afterExecutor()为扩展节点，便于进行业务扩展时使用，例如在所述具体处置器抽象类执行业务抽象方法之前通过beforeExecutor()来进行数据的预处理，在所述具体处置器抽象类执行业务抽象方法之后通过afterExecutor()来进行日志的记录。beforeExecutor()和afterExecutor()的内容可以根据不同的业务抽象方法进行相应的扩展。

[0059] 具体的，EVENT_CONTEXT为ThreadLocal线程副本对象，用于解决一次处置器执行流程中各流程节点间的数据传递问题，ThreadLocal为是Thread的局部变量。

[0060] 具体的，当一次处置器执行流程结束后所述顶级处置器抽象类自动调用私有方法clear()放资源，避免资源的浪费。

[0061] 具体的，successorTag为自动执行的处置器标签，使用successorTag来判断当一次处置器执行流程结束后是否需要根据处置器标签自动执行下一个处置行为，以实现自动化功能。

[0062] 在一些实施例中，在所述顶级处置器抽象类中定义一个标签变量，所述标签变量用来存取所述处置器标签，当所述顶级处置器抽象类中定义的业务抽象方法执行完毕后，判断所述标签变量中是否存在下一处置行为所需的处置器标签，若存在则根据所述处置器标签执行对应处置行为。

[0063] 具体的，当前的处置行为执行完毕后，若判断存在下一个处置行为的标签时则自动执行对应的处置行为，依次类推可以形成一条责任链以达到编排处置行为的目的，可以保证有序的完成案件的多个处置行为。

[0064] 在一些实施例中,在“在所述具体处置器抽象类中定义每一类处置行为的通用行为模板方法”步骤中,所述具体处置器抽象类继承所述顶级处置器抽象类中的业务抽象方法,并重写顶级处置器抽象类中的业务抽象方法得到多个处置行为的通用行为模板方法,所述通用行为模板方法包括多个处置行为的共同点。

[0065] 具体的,定义多个处置行为的通用行为模板方法,可以做到在多个处置器之间将公共业务抽象给通用行为模板方法来实现,并可以根据扩展点重写需要变更的通用行为和根据扩展点实现各处置器自身特有的业务行为,当原处置行为的业务发生变更时,如果是共同点发生变更,只需要修改通用行为模板方法的内部逻辑,如果是某个处置行为特有的业务发生变更,只需要修改该处置器特有的扩展点方法,其次,存在新的处置行为需要开发时,只需选择合适的扩展点进行开发,修改封闭且复用度高。

[0066] 在一些具体实施例中,如图8所示,定义具体处置器抽象类AbstractAcceptHandler和AbstractProcessHandler,其继承自顶级处置器抽象类AbstractEventHandler,所述具体处置器抽象类的特征在于,一些同为处置器执行流程中的不同处置行为具备共同点,但又不完全一致,所以将不同处置行为的共同点通过重写业务抽象方法得到通用行为模板。

[0067] 例如,web端上报以及AI上报具备通用的操作记录、附件记录等业务逻辑,但由于上报终端不同存在部分业务差异,于是使用所述具体处置器抽象类定义web端上报以及AI上报的通用业务逻辑得到一个具体处置器抽象类。

[0068] 以图8中的处置器抽象类AbstractProcessHandler为例:

[0069] AbstractProcessHandler为流程处置器抽象类,在其中重写了顶级处置器抽象类AbstractEventHandler中的业务抽象方法execute(T parameter),并在所述流程处置器抽象类中细化定义了流程处置业务所需要的通用行为,具体如下所示:

[0070] 1.公共校验commonCheck(T parameter),该方法作为流程处置器执行的首个执行方法,用于参数、业务的校验。其又分为三个方法执行,分别是:入参校验checkParameter(T parameter)、业务合理性校验checkLegality(T parameter),业务校验checkBiz(T parameter)。checkParameter(T parameter)和checkLegality(T parameter)通过流程处置器抽象类的模版方法实现,为子类处置器提供了通用能力。而checkBiz(T parameter)为抽象方法,交由子类重写,用于子类处置器对应处置行为特有的业务校验。

[0071] 2.执行业务executeBiz(T parameter),交由子类重写,用于子类处置器实现类对应处置行为特有的业务能力。

[0072] 3.执行操作记录executeOperationRecord(T parameter),流程处置器抽象类模版方法实现,用于对流程相关执法业务做通用的操作日志记录。

[0073] 4.执行附件操作executeFileOperation(T parameter),流程处置器抽象类模版方法实现,用于执行流程相关执法业务通用的附件信息记录。

[0074] 5.执行工作流集成executeFlow(T parameter),流程处置器抽象类模版方法实现,用于流程相关执法业务通用的集成工作流完成节点、分配待办人、设置流程参数等相关操作。

[0075] 6.执行收尾工作executeConcludingWork(T parameter),流程处置器抽象类模版方法实现,用于流程相关执法业务通用的在执行完工作流集成后,通过工作流返回的节点

信息做节点状态记录、案件信息更新等操作。

[0076] 7.执行扩展业务executeExtBiz(T parameter),交由子类重写,当流程处置器抽象类模版方法无法满足具体处置行为的业务执行时,通过重写该方法进行能力扩展。

[0077] 8.通知消息notifyMsg(),流程处置器抽象类模版方法实现,用于流程相关执法业务执行完毕后发送消息通知,便于进行可能存在的跨服务的能力扩展。

[0078] 9.执行责任链executeSuccessor(T paramter),交由子类重写,用于子类处置器根据其特有业务的需要,集成算法平台或其他业务规则,基于规则的结果编排需要智能流转的处置行为,通过维护顶级处置器抽象类AbstractEventHandler中所持有的属性successorTag,进行责任链执行。

[0079] 具体的,本方案通过在顶级处置器抽象类中将执行流程的业务抽象方法定义完毕,再由具体处置器抽象类实现执行流程中的业务抽象方法,并在该业务抽象方法实现中将多个处置行为的共同点写成通用行为模板,并保留其扩展性和设计新的抽象方法扩展点,以达到通用行为可重写和业务抽象方法可扩展的目的。

[0080] 在一些实施例中,在“构建多个继承具体处置器抽象类的处置器实现类,每一所述处置器实现类与一类所述处置行为对应”步骤中,所述处置器实现类根据具体处置器抽象类的扩展性进行扩展得到,在所述处置器实现类中定义了对应处置行为的至少一处置方法。

[0081] 在一些具体实施例中,以具体处置器抽象类AbstractProcessHandler为例,其包含HandleProcessHandler、CheckProcessHandler、CloseProcessHandler三个子类,HandleProcessHandler对应的处置行为为“处理”,在HandleProcessHandler中使用所述处置器实现类重写抽象方法,通过所述处置器实现类得到的处置操作如下:

[0082] 1.处理业务校验checkBiz(T parameter),进行处理操作特有的校验,如:校验该案件信息是否齐全,是否满足处理条件等。

[0083] 2.处理业务执行executeBiz(T parameter),进行处理操作特有的业务,如:记录处理部门、处理人、处理时间、处理现场照片、处理结果等。

[0084] 3.责任链执行executeSuccessor(T paramter),进行智能化场景集成,如:集成自动审核规则,通过规则获取是否满足自动审核条件,若条件满足,通过维护其父类AbstractEventHandler中的的successorTag字段为“CHECK”,编排需自动执行的审核处置器CheckProcessHandler,实现满足规则时完成处理操作后自动进行审核的业务效果。

[0085] CheckProcessHandler对应的处置行为为“审核”,使用所述处置器抽象类在CheckProcessHandler中重写抽象方法,得到的审核操作如下:

[0086] 1.审核业务校验checkBiz(T parameter),进行审核操作特有的校验,如:该人员是否具有审核权限等。

[0087] 2.审核业务执行executeBiz(T parameter),进行审核操作特有的业务,如:记录审核结果、审核意见、审核时间等。

[0088] 3.责任链执行executeSuccessor(T paramter),进行智能化场景集成,如:集成自动结案规则,若满足自动结案条件,通过维护其父类AbstractEventHandler中的的successorTag字段为“CLOSE”,编排需自动执行的结案处置器CloseProcessHandler,实现满足规则时完成审核操作后自动进行结案的业务效果。

[0089] CloseProcessHandler对应的处置行为为“结案”,使用所述处置器抽象类在CloseProcessHandler中重写抽象方法,得到的结案操作如下:

[0090] 1.结案业务校验checkBiz(T parameter),进行结案操作特有的校验,如:案件信息是否满足结案标准,该人员是否具有结案权限等。

[0091] 2.结案业务执行executeBiz(T parameter),进行审核行为特有的业务,如:记录结案原因等。

[0092] 3.通知消息notifyMsg(),进行结案后的消息通知,如:结案后发送APP消息推送或短信至管理人员移动终端,发送MQ消息便于第三方跨服务获取通知进行业务扩展。

[0093] 具体的,除了业务校验checkBiz和业务执行executeBiz这两个必须实现的方法外,可以通过重写任一方法实现业务扩展,例如,在CloseProcessHandler中可以重写executeOpeartionRecord(T parameter)来变更原有的操作记录逻辑,或通过重写executeExtBiz(T parameter)实现一些诸如案件结案数统计的扩展性业务、

[0094] 在一些实施例中,定义一个策略工厂,所述处置器策略工厂使用两个Map结构分别维护所述处置器标签与处置器入参子类的映射关系以及所述处置器标签与所述处置器实现类的映射关系,并将所述映射关系保存到内存中。

[0095] 具体的,将所述映射关系保存到内存中以便于实时进行调用。

[0096] 在一些具体实施例中,可以使用STRATEGY_HANDLER_MAP和STRATEGY_PARAMETER_MAP两个map结构来维护映射关系,并对外提供以下四个方法进行映射:

[0097] 1.addStrategy(String tag, AbstractEventHandler<?> handler, Class<? extends BaseEventHandleCmd> parameterClz, boolean isOverride),新增策略映射关系方法。在新增业务或业务变更的情况下,处置器标签Tag对应的处置器入参和处置器实现类需要替换或新增,所以在该处置器策略工厂中,需要提供增加映射关系的方法,并根据入参中的isOverride参数为true或false来决定当该处置器标签Tag存在时,是否覆写原有映射关系。

[0098] 2.getStrategyHandler(String tag),通过处置器标签Tag获取对应处置器的方法。

[0099] 3.parseParameter(String tag, Map parameterMap),通过处置器标签Tag获取对应处置器入参类并解析泛化调用入参parameterMap的方法。

[0100] 4.execute(String tag, Map parameterMap),策略执行方法,该方法内会调用getStrategyHandler(String tag)和parseParameter(String tag, Map parameterMap)方法,得到处置器标签tag映射的具体处置器和该处置器的入参,执行其实现的处置器接口EventHandler的handle(T parameter)方法完成执法流程的运行。

[0101] 在一些实施例中,当调用方传入处置器标签时,所述处置器策略工厂根据所述处置器标签调用对应的处置器入参子类以及处置器实现类,使用所述处置器入参子类进行入参以执行对应的处置行为。

[0102] 当需要添加一种新的处置行为时,采用一种对基于规则的处置行为流程进行扩展的方法,其步骤如下所示:

[0103] 当编排好一个完整的处置流程,且需要扩展新的处置行为时,定义所述处置行为的处置器实现类,将所述处置器实现类继承于符合其通用行为的具体处置器抽象类,通过

实现和重写所述具体处置器抽象类中定义的通用行为模板方法来实现所述处置行为的业务,完成新的处置行为的扩展。

[0104] 实施例二

[0105] 基于相同的构思,参考图9,本申请还提出了一种基于规则的处置行为流程编排装置,包括:

[0106] 获取模块:获取多个关联案件的处置行为,定义与各处置行为一一对应的处置器标签,构建写有泛型入参执行方法的处置接口,各个所述处置行为访问同一所述处置接口;

[0107] 第一构建模块:根据所述处置行为和对应的处置器标签构建处置器入参基类以及处置器入参子类,每一所述处置器入参子类均继承所述处置器入参基类,定义每一所述处置器标签与所述处置器入参子类的映射关系;

[0108] 第二构建模块:构建顶级处置器抽象类,所述顶级处置器抽象类通过处置接口的泛型入参执行方法定义出业务抽象方法和扩展节点,并保留所述业务抽象方法和扩展节点的扩展性;

[0109] 定义模块:所述顶级处置器抽象类包含继承所述顶级处置器抽象类的多个具体处置器抽象类,在所述具体处置器抽象类中定义每一类处置行为的通用行为模板方法,并保留所述通用行为模板方法的扩展性;

[0110] 执行模块:构建多个继承具体处置器抽象类的处置器实现类,每一所述处置器实现类与一类所述处置行为对应,定义每一所述处置器标签与处置器实现类的映射关系,所述处置器实现类基于处置行为的处置器标签被调用。

[0111] 实施例三

[0112] 本实施例还提供了一种电子装置,参考图10,包括存储器404和处理器402,该存储器404中存储有计算机程序,该处理器402被设置为运行计算机程序以执行上述任一项方法实施例中的步骤。

[0113] 具体地,上述处理器402可以包括中央处理器(CPU),或者特定集成电路(ApplicationSpecificIntegratedCircuit,简称为ASIC),或者可以被配置成实施本申请实施例的一个或多个集成电路。

[0114] 其中,存储器404可以包括用于数据或指令的大容量存储器404。举例来说而非限制,存储器404可包括硬盘驱动器(HardDiskDrive,简称为HDD)、软盘驱动器、固态驱动器(SolidStateDrive,简称为SSD)、闪存、光盘、磁光盘、磁带或通用串行总线(UniversalSerialBus,简称为USB)驱动器或者两个或更多个以上这些的组合。在合适的情况下,存储器404可包括可移除或不可移除(或固定)的介质。在合适的情况下,存储器404可在数据处理装置的内部或外部。在特定实施例中,存储器404是非易失性(Non-Volatile)存储器。在特定实施例中,存储器404包括只读存储器(Read-OnlyMemory,简称为ROM)和随机存取存储器(RandomAccessMemory,简称为RAM)。在合适的情况下,该ROM可以是掩模编程的ROM、可编程ROM(ProgrammableRead-OnlyMemory,简称为PROM)、可擦除PROM(ErasableProgrammableRead-OnlyMemory,简称为EPROM)、电可擦除PROM(ElectricallyErasableProgrammableRead-OnlyMemory,简称为EEPROM)、电可改写ROM(ElectricallyAlterableRead-OnlyMemory,简称为EAROM)或闪存(FLASH)或者两个或更多个以上这些的组合。在合适的情况下,该RAM可以是静态随机存取存储器(StaticRandom-

AccessMemory,简称为SRAM)或动态随机存取存储器(DynamicRandomAccessMemory,简称为DRAM),其中,DRAM可以是快速页模式动态随机存取存储器404(FastPageModeDynamicRandomAccessMemory,简称为FPMDRAM)、扩展数据输出动态随机存取存储器(ExtendedDateOutDynamicRandomAccessMemory,简称为EDODRAM)、同步动态随机存取内存(SynchronousDynamicRandom-AccessMemory,简称SDRAM)等。

[0115] 存储器404可以用来存储或者缓存需要处理和/或通信使用的各种数据文件,以及处理器402所执行的可能的计算机程序指令。

[0116] 处理器402通过读取并执行存储器404中存储的计算机程序指令,以实现上述实施例中的任意一种基于规则编排的执行处置方法。

[0117] 可选地,上述电子装置还可以包括传输设备406以及输入输出设备408,其中,该传输设备406和上述处理器402连接,该输入输出设备408和上述处理器402连接。

[0118] 传输设备406可以用来经由一个网络接收或者发送数据。上述的网络具体实例可包括电子装置的通信供应商提供的有线或无线网络。在一个实例中,传输设备包括一个网络适配器(Network Interface Controller,简称为NIC),其可通过基站与其他网络设备相连从而可与互联网进行通讯。在一个实例中,传输设备406可以为射频(Radio Frequency,简称为RF)模块,其用于通过无线方式与互联网进行通讯。

[0119] 输入输出设备408用于输入或输出信息。在本实施例中,输入的信息可以是各种处置行为、处置器参数等,输出的信息可以是处置行为的处置结果等。

[0120] 可选地,在本实施例中,上述处理器402可以被设置为通过计算机程序执行以下步骤:

[0121] S101、获取多个关联案件的处置行为,定义与各处置行为一一对应的处置器标签,构建写有泛型入参执行方法的处置接口,各个所述处置行为访问同一所述处置接口;

[0122] S102、根据所述处置行为和对应的处置器标签构建处置器入参基类以及处置器入参子类,每一所述处置器入参子类均继承所述处置器入参基类,定义每一所述处置器标签与所述处置器入参子类的映射关系;

[0123] S103、构建顶级处置器抽象类,所述顶级处置器抽象类通过处置接口的泛型入参执行方法定义出业务抽象方法和扩展节点,并保留所述业务抽象方法和扩展节点的扩展性;

[0124] S104、所述顶级处置器抽象类包含继承所述顶级处置器抽象类的多个具体处置器抽象类,在所述具体处置器抽象类中定义每一类处置行为的通用行为模板方法,并保留所述通用行为模板方法的扩展性;

[0125] S105、构建多个继承具体处置器抽象类的处置器实现类,每一所述处置器实现类与一类所述处置行为对应,定义每一所述处置器标签与处置器实现类的映射关系,所述处置器实现类基于处置行为的处置器标签被调用。

[0126] 需要说明的是,本实施例中的具体示例可以参考上述实施例及可选实施方式中所描述的示例,本实施例在此不再赘述。

[0127] 通常,各种实施例可以以硬件或专用电路、软件、逻辑或其任何组合来实现。本发明的一些方面可以以硬件来实现,而其他方面可以以可以由控制器、微处理器或其他计算设备执行的固件或软件来实现,但是本发明不限于此。尽管本发明的各个方面可以被示出

和描述为框图、流程图或使用一些其他图形表示，但是应当理解，作为非限制性示例，本文中描述的这些框、装置、系统、技术或方法可以以硬件、软件、固件、专用电路或逻辑、通用硬件或控制器或其他计算设备或其某种组合来实现。

[0128] 本发明的实施例可以由计算机软件来实现，该计算机软件由移动设备的数据处理器诸如在处理器实体中可执行，或者由硬件来实现，或者由软件和硬件的组合来实现。包括软件例程、小程序和/或宏的计算机软件或程序(也称为程序产品)可以存储在任何装置可读数据存储介质中，并且它们包括用于执行特定任务的程序指令。计算机程序产品可以包括当程序运行时被配置为执行实施例的一个或多个计算机可执行组件。一个或多个计算机可执行组件可以是至少一个软件代码或其一部分。另外，在这一点上，应当注意，如图10中的逻辑流程的任何框可以表示程序步骤、或者互连的逻辑电路、框和功能、或者程序步骤和逻辑电路、框和功能的组合。软件可以存储在诸如存储器芯片或在处理器内实现的存储块等物理介质、诸如硬盘或软盘等磁性介质、以及诸如例如DVD及其数据变体、CD等光学介质上。物理介质是非瞬态介质。

[0129] 本领域的技术人员应该明白，以上实施例的各技术特征可以进行任意的组合，为使描述简洁，未对上述实施例中的各个技术特征所有可能的组合都进行描述，然而，只要这些技术特征的组合不存在矛盾，都应当认为是本说明书记载的范围。

[0130] 以上实施例仅表达了本申请的几种实施方式，其描述较为具体和详细，但并不能因此而理解为对本申请范围的限制。应当指出的是，对于本领域的普通技术人员来说，在不脱离本申请构思的前提下，还可以做出若干变形和改进，这些都属于本申请的保护范围。因此，本申请的保护范围应以所附权利要求为准。

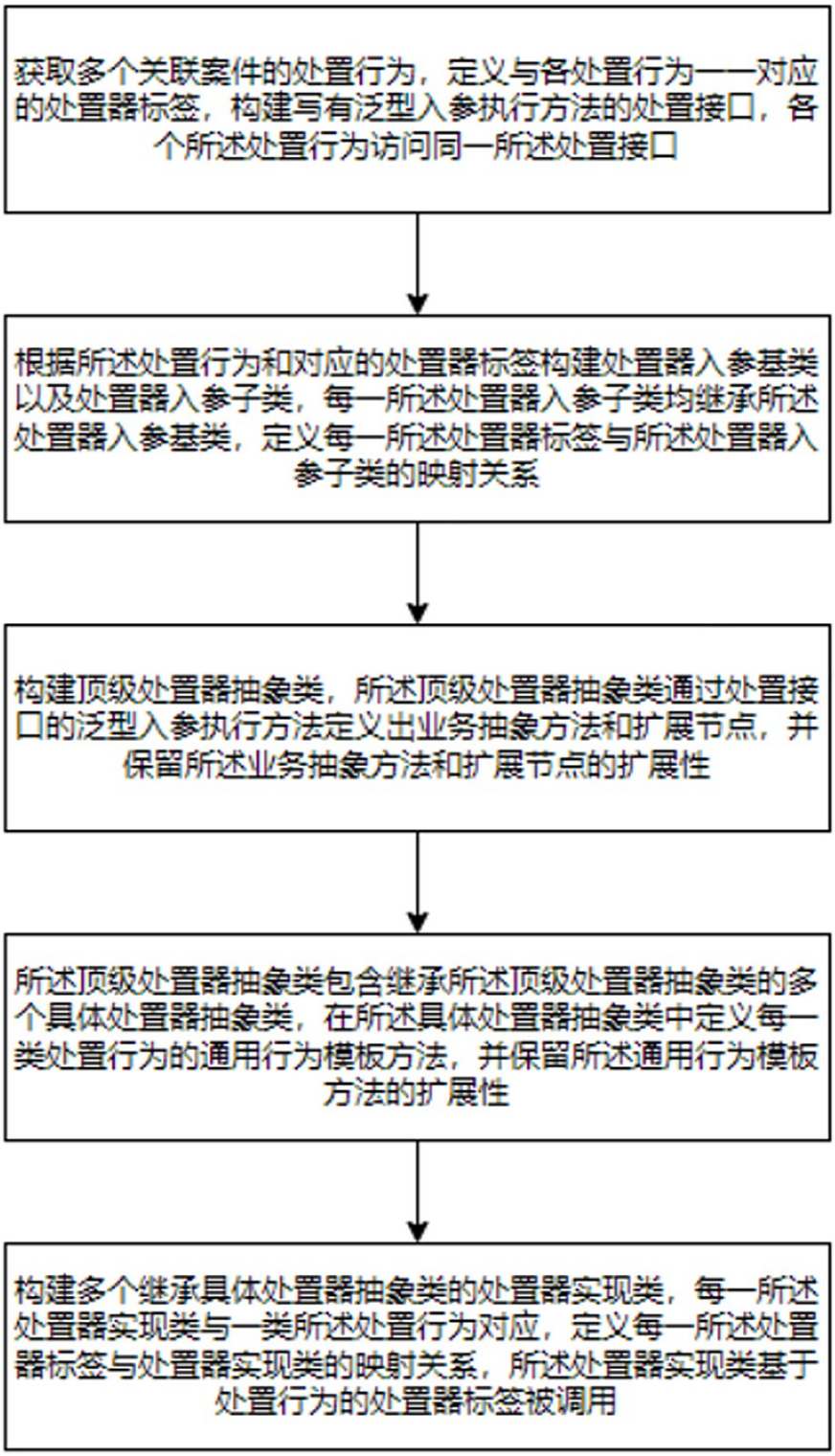

获取多个关联案件的处置行为，定义与各处置行为一一对应的处置器标签，构建写有泛型入参执行方法的处置接口，各个所述处置行为访问同一所述处置接口
根据所述处置行为和对应的处置器标签构建处置器入参基类以及处置器入参子类，每一所述处置器入参子类均继承所述处置器入参基类，定义每一所述处置器标签与所述处置器入参子类的映射关系
构建顶级处置器抽象类，所述顶级处置器抽象类通过处置接口的泛型入参执行方法定义出业务抽象方法和扩展节点，并保留所述业务抽象方法和扩展节点的扩展性
所述顶级处置器抽象类包含继承所述顶级处置器抽象类的多个具体处置器抽象类，在所述具体处置器抽象类中定义每一类处置行为的通用行为模板方法，并保留所述通用行为模板方法的扩展性
构建多个继承具体处置器抽象类的处置器实现类，每一所述处置器实现类与一类所述处置行为对应，定义每一所述处置器标签与处置器实现类的映射关系，所述处置器实现类基于处置行为的处置器标签被调用

图1

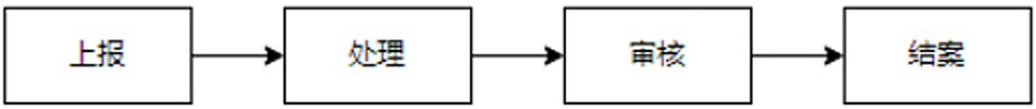

上报
处理
审核
结案

图2

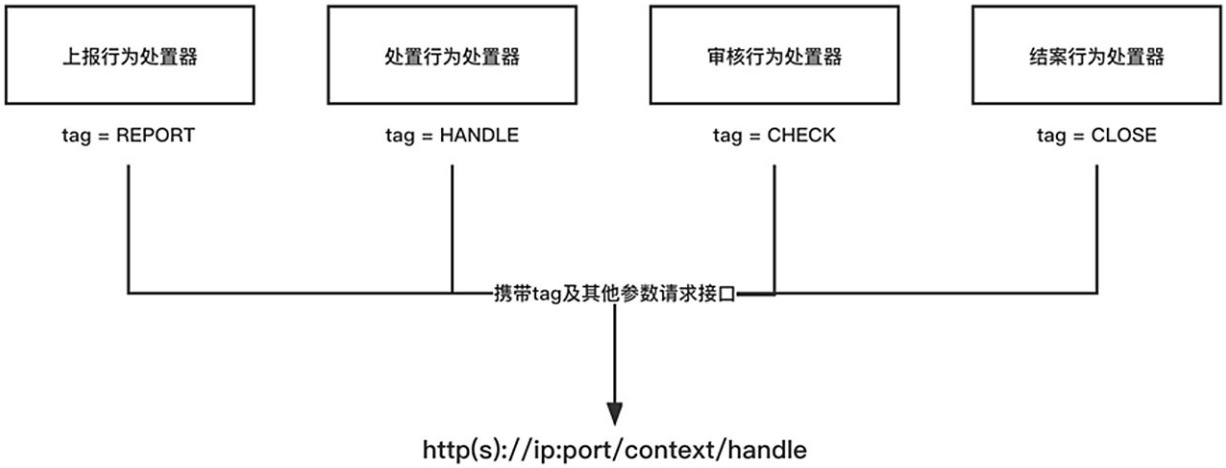

上报行为处置器
处置行为处置器
审核行为处置器
结案行为处置器
tag = REPORT
tag = HANDLE
tag = CHECK
tag = CLOSE
携带tag及其他参数请求接口
http(s)://ip:port/context/handle

图3

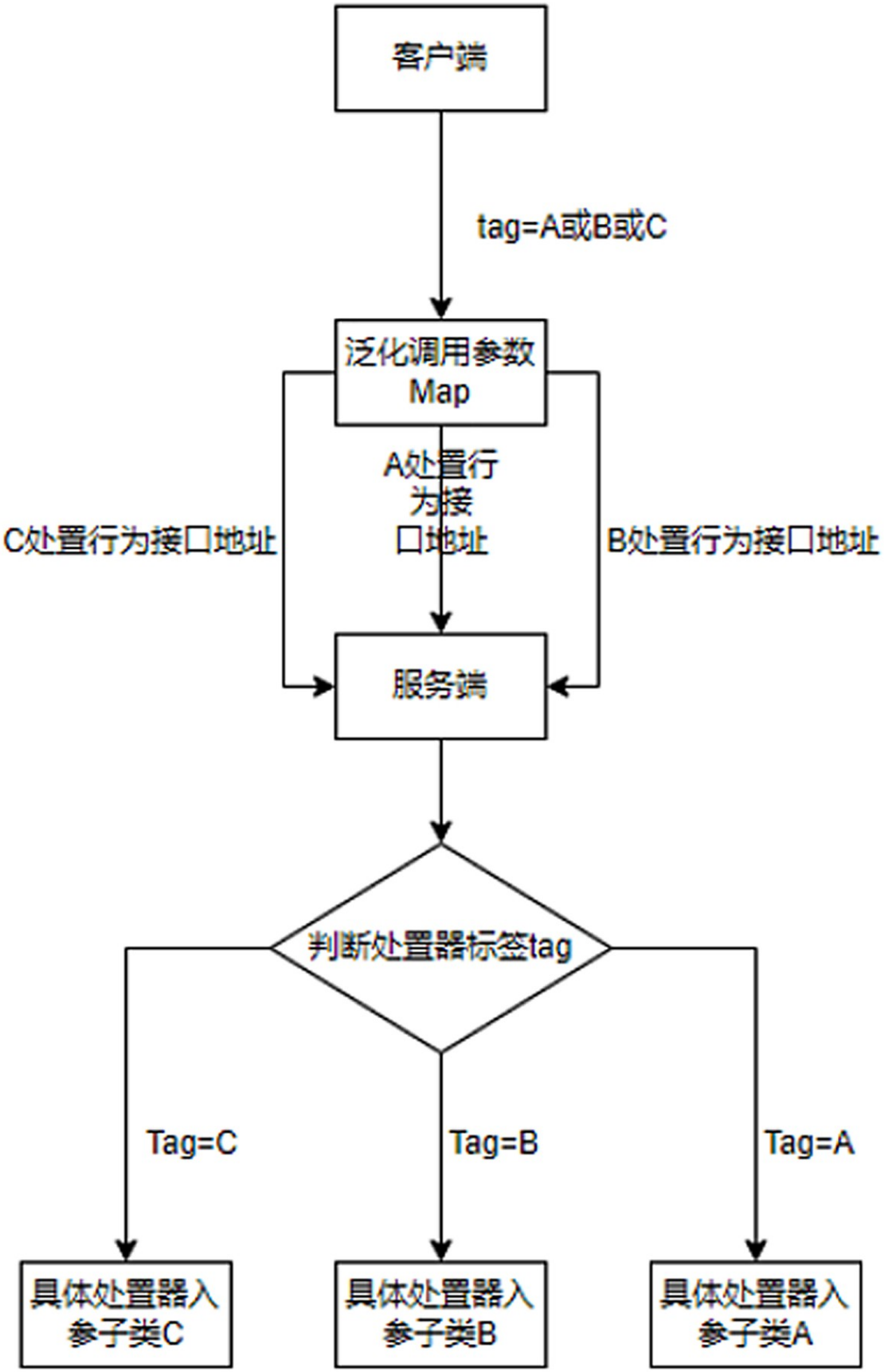

客户端
tag=A或B或C
泛化调用参数
Map
A处置行
为接
口地址
C处置行为接口地址
B处置行为接口地址
服务端
判断处置器标签tag
Tag=C
Tag=B
Tag=A
具体处置器入
参子类C
具体处置器入
参子类B
具体处置器入
参子类A

图4

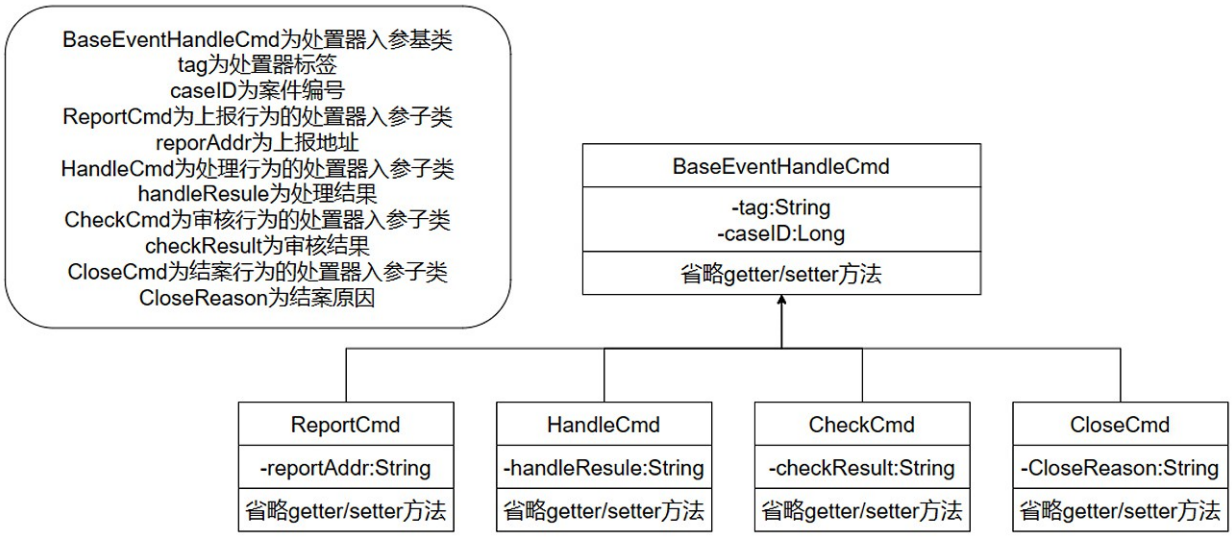

BaseEventHandleCmd为处置器入参基类
tag为处置器标签
caseID为案件编号
ReportCmd为上报行为的处置器入参子类
reporAddr为上报地址
HandleCmd为处理行为的处置器入参子类
handleResule为处理结果
CheckCmd为审核行为的处置器入参子类
checkResult为审核结果
CloseCmd为结案行为的处置器入参子类
CloseReason为结案原因
BaseEventHandleCmd
-tag:String
-caseID:Long
省略getter/setter方法
ReportCmd
-reportAddr:String
省略getter/setter方法
HandleCmd
-handleResule:String
省略getter/setter方法
CheckCmd
-checkResult:String
省略getter/setter方法
CloseCmd
-CloseReason:String
省略getter/setter方法

图5

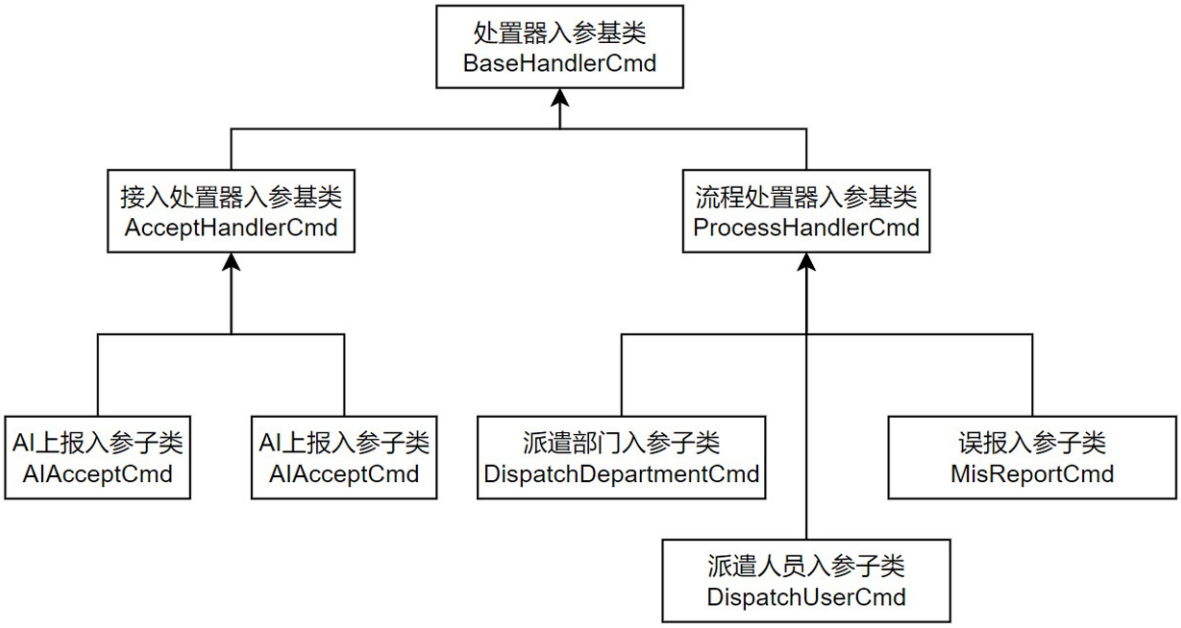

处置器入参基类
BaseHandlerCmd
接入处置器入参基类
AcceptHandlerCmd
流程处置器入参基类
ProcessHandlerCmd
AI上报入参子类
AIAcceptCmd
AI上报入参子类
AIAcceptCmd
派遣部门入参子类
DispatchDepartmentCmd
误报入参子类
MisReportCmd
派遣人员入参子类
DispatchUserCmd

图6

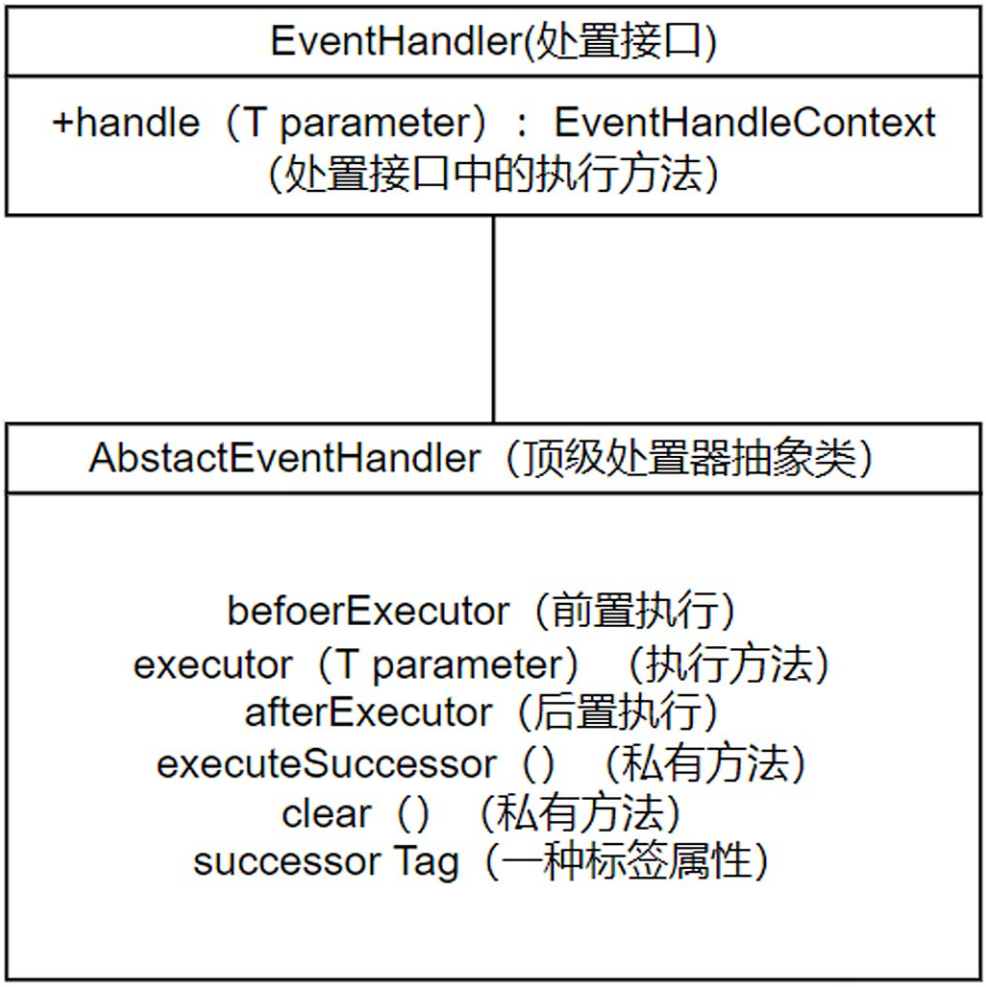

EventHandler(处置接口)
+handle（T parameter）：EventHandleContext
(处置接口中的执行方法)
AbstactEventHandler（顶级处置器抽象类）
befoerExecutor（前置执行）
executor（T parameter）（执行方法）
afterExecutor（后置执行）
executeSuccessor（）（私有方法）
clear（）（私有方法）
successor Tag（一种标签属性）

图7

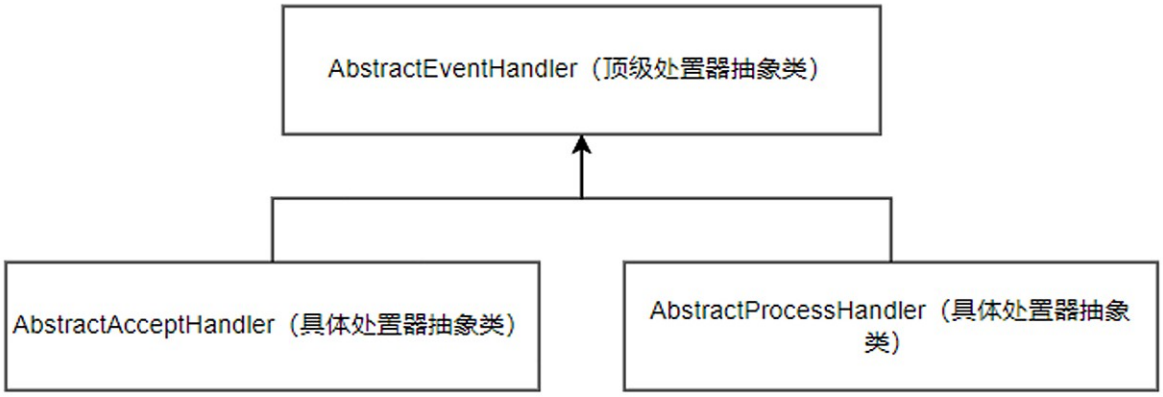

AbstractEventHandler（顶级处置器抽象类）
AbstractAcceptHandler（具体处置器抽象类）
AbstractProcessHandler（具体处置器抽象类）

图8

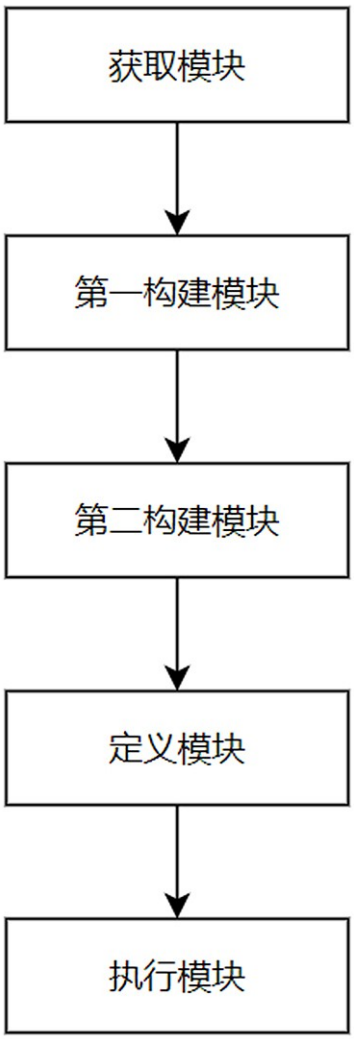


图9

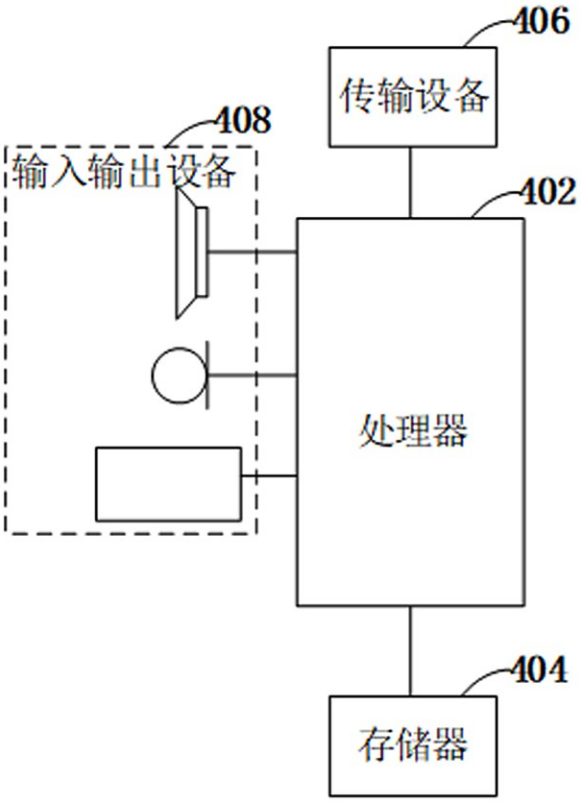


图10