(19)中华人民共和国国家知识产权局

(12)发明专利申请

(10)申请公布号 CN 110990278 A
(43)申请公布日 2020.04.10

(21)申请号 201911207343.4

(22)申请日 2019.11.29

(71)申请人 深圳前海微众银行股份有限公司
地址 518027 广东省深圳市前海深港合作区前湾一路1号A栋201室

(72)发明人 夏竞博 李鸿哲 江和智

(74)专利代理机构 北京同达信恒知识产权代理有限公司 11291
代理人 侯林林

(51)Int.Cl.
G06F 11/36(2006.01)

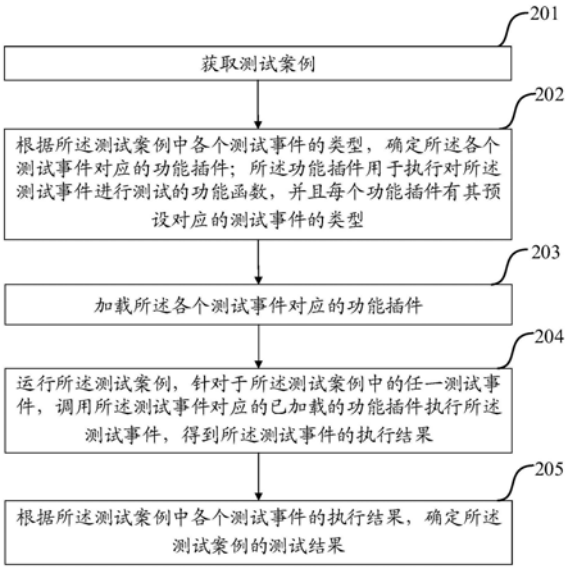
权利要求书2页 说明书13页 附图4页

(54)发明名称

一种测试方法及装置

(57)摘要

本发明实施例公开了一种测试方法及装置，获取测试案例，并根据测试案例中各个测试事件的类型确定各个测试事件对应的功能插件，加载各个测试事件对应的功能插件，运行测试案例，针对于测试案例中的任一测试事件，调用该测试事件对应的已加载的功能插件执行测试事件，得到测试事件的执行结果，根据各个测试事件的执行结果，确定测试案例的测试结果。通过预先为每个功能插件设置对应的测试事件的类型，使得在测试任意的测试案例时，能够仅加载该测试案例中的测试事件的类型所对应的功能插件，而无需加载测试案例不需要的功能插件，从而可以提高测试的灵活性，还能节省测试所需的系统资源，提高测试的效率。



CN 110990278 A

1.一种测试方法,其特征在于,所述方法包括:

获取测试案例;

根据所述测试案例中各个测试事件的类型,确定所述各个测试事件对应的功能插件;所述功能插件用于执行对所述测试事件进行测试的功能类函数,并且每个功能插件有其预设对应的测试事件的类型;

加载所述各个测试事件对应的功能插件;

运行所述测试案例,针对于所述测试案例中的任一测试事件,调用所述测试事件对应的已加载的功能插件执行所述测试事件,得到所述测试事件的执行结果;

根据所述测试案例中各个测试事件的执行结果,确定所述测试案例的测试结果。

2.根据权利要求1所述的方法,其特征在于,所述调用所述测试事件对应的已加载的功能插件执行所述测试事件,包括:

若所述测试事件对应的已加载的功能插件包括多个,则按照所述多个功能插件的优先级执行所述测试事件;所述多个功能插件的优先级用于标识执行所述测试事件的顺序。

3.根据权利要求1所述的方法,其特征在于,所述调用所述测试事件对应的已加载的功能插件执行所述测试事件,包括:

若确定采用同步处理方式执行所述测试事件,则调用所述已加载的功能插件执行所述测试事件,若确定采用异步处理方式执行所述测试事件,则生成异步线程,并使用所述异步线程调用所述已加载的功能插件执行所述测试事件。

4.根据权利要求1至3中任一项所述的方法,其特征在于,所述调用所述测试事件对应的已加载的功能插件执行所述测试事件,得到所述测试事件的执行结果,包括:

若所述已加载的功能插件为外部接口调用插件,则根据所述测试事件所需调用的接口,确定是否存在所述测试事件对应的报文模板;

若存在所述测试事件对应的报文模板,则从所述测试事件中提取得到测试场景,并根据所述测试场景和所述测试事件对应的报文模板,得到所述测试事件对应的执行结果;若不存在所述测试事件对应的报文模板,则将所述测试事件发送给所述测试事件所需调用的外部接口,并获取所述外部接口执行所述测试事件得到的执行结果。

5.根据权利要求4所述的方法,其特征在于,所述根据所述测试场景和所述测试事件对应的报文模板,得到所述测试事件对应的执行结果,包括:

根据所述测试场景从所述测试事件对应的报文模板中获取所述测试事件对应的初始报文结果;

若所述初始报文结果中包括设定占位符,则从所述测试事件中提取得到所述设定占位符对应的测试内容,并使用所述测试内容替换所述初始报文结果中的设定占位符,得到所述测试事件对应的执行结果;若所述初始报文结果中不包括所述设定占位符,则将所述初始报文结果作为所述测试事件对应的执行结果。

6.一种测试装置,其特征在于,所述装置包括:

获取模块,用于获取测试案例;

确定模块,用于根据所述测试案例中各个测试事件的类型,确定所述各个测试事件对应的功能插件;所述功能插件用于执行对所述测试事件进行测试的功能类函数,并且每个功能插件有其预设对应的测试事件的类型;

加载模块，用于加载所述各个测试事件对应的功能插件；

执行模块，用于运行所述测试案例，针对于所述测试案例中的任一测试事件，调用所述测试事件对应的已加载的功能插件执行所述测试事件，得到所述测试事件的执行结果；

所述确定模块，还用于根据所述测试案例中各个测试事件的执行结果，确定所述测试案例的测试结果。

7.根据权利要求6所述的装置，其特征在于，所述执行模块具体用于：

若所述测试事件对应的已加载的功能插件包括多个，则按照所述多个功能插件的优先级执行所述测试事件；所述多个功能插件的优先级用于标识执行所述测试事件的顺序。

8.根据权利要求6所述的装置，其特征在于，所述执行模块具体用于：

若确定采用同步处理方式执行所述测试事件，则调用所述已加载的功能插件执行所述测试事件，若确定采用异步处理方式执行所述测试事件，则生成异步线程，并使用所述异步线程调用所述已加载的功能插件执行所述测试事件。

9.根据权利要求6至8中任一项所述的装置，其特征在于，所述执行模块具体用于：

若所述已加载的功能插件为外部接口调用插件，则根据所述测试事件所需调用的接口，确定是否存在所述测试事件对应的报文模板；

若存在所述测试事件对应的报文模板，则从所述测试事件中提取得到测试场景，并根据所述测试场景和所述测试事件对应的报文模板，得到所述测试事件对应的执行结果；若不存在所述测试事件对应的报文模板，则将所述测试事件发送给所述测试事件所需调用的外部接口，并获取所述外部接口执行所述测试事件得到的执行结果。

10.根据权利要求9所述的装置，其特征在于，所述执行模块具体用于：

根据所述测试场景从所述测试事件对应的报文模板中获取所述测试事件对应的初始报文结果；

若所述初始报文结果中包括设定占位符，则从所述测试事件中提取得到所述设定占位符对应的测试内容，并使用所述测试内容替换所述初始报文结果中的设定占位符，得到所述测试事件对应的执行结果；若所述初始报文结果中不包括所述设定占位符，则将所述初始报文结果作为所述测试事件对应的执行结果。

11.一种计算设备，其特征在于，包括至少一个处理器以及至少一个存储器，其中，所述存储器存储有计算机程序，当所述程序被所述处理器执行时，使得所述处理器执行权利要求1～5任一权利要求所述的方法。

12.一种计算机可读存储介质，其特征在于，其存储有可由计算设备执行的计算机程序，当所述程序在所述计算设备上运行时，使得所述计算设备执行权利要求1～5任一权利要求所述的方法。

一种测试方法及装置

技术领域

[0001] 本发明涉及金融科技(Fintech)技术领域,尤其涉及一种测试方法及装置。

背景技术

[0002] 随着计算机技术的发展,越来越多的技术应用在金融领域,传统金融业正在逐步向金融科技(Fintech)转变,但由于金融行业的安全性、实时性要求,也对技术提出了更高的要求。

[0003] 对于金融行业的任一业务系统来说,测试是在开发该业务系统的功能时不可缺少的一环,然而,现阶段的测试流程通常是设定好的,任意的测试案例均需要按照相同的测试流程来进行测试,从而导致测试的灵活性较差。

[0004] 综上,目前亟需一种测试方法,用以解决现有技术按照设定好的测试流程执行测试所导致的测试的灵活性较差的技术问题。

发明内容

[0005] 本发明实施例提供一种测试方法及装置,用以解决现有技术按照设定好的测试流程执行测试所导致的测试的灵活性较差的技术问题。

[0006] 第一方面,本发明实施例提供的一种测试方法,包括:

[0007] 获取测试案例,根据所述测试案例中各个测试事件的类型,确定所述各个测试事件对应的功能插件;所述功能插件用于执行对所述测试事件进行测试的功能类函数,并且每个功能插件有其预设对应的测试事件的类型;进一步地,加载所述各个测试事件对应的功能插件,运行所述测试案例,针对于所述测试案例中的任一测试事件,调用所述测试事件对应的已加载的功能插件执行所述测试事件,得到所述测试事件的执行结果,根据所述测试案例中各个测试事件的执行结果,确定所述测试案例的测试结果。

[0008] 本发明实施例中,通过预先为每个功能插件设置对应的测试事件的类型,使得在测试任意的测试案例时,能够仅加载该测试案例中所包括的测试事件的类型对应的功能插件,而无需加载测试案例不需要的功能插件,从而可以提高测试的灵活性,还能节省测试所需的系统资源,提高测试的效率。

[0009] 在一种可能的实现方式中,所述调用所述测试事件对应的已加载的功能插件执行所述测试事件,包括:若所述测试事件对应的已加载的功能插件包括多个,则按照所述多个功能插件的优先级执行所述测试事件;所述多个功能插件的优先级用于标识执行所述测试事件的顺序。

[0010] 在上述实现方式中,响应同一类型的测试事件的各个功能插件的优先级能够标识各个功能插件的依赖关系,因此当某一测试事件对应多个功能插件时,通过按照多个功能插件的优先级依次执行测试事件,使得测试事件能够按照设定的依赖关系被准确地执行,从而提高测试的准确性。

[0011] 在一种可能的实现方式中,所述调用所述测试事件对应的已加载的功能插件执行

所述测试事件，包括：若确定采用同步处理方式执行所述测试事件，则调用所述已加载的功能插件执行所述测试事件，若确定采用异步处理方式执行所述测试事件，则生成异步线程，并使用所述异步线程调用所述已加载的功能插件执行所述测试事件。

[0012] 在上述实现方式中，通过判断采用何种处理方式执行测试事件，使得处理方式能够更加符合系统的性能损耗情况，从而降低对系统的影响，避免系统卡顿，提高测试的准确性。

[0013] 在一种可能的实现方式中，所述调用所述测试事件对应的已加载的功能插件执行所述测试事件，得到所述测试事件的执行结果，包括：若所述已加载的功能插件为外部接口调用插件，则根据所述测试事件所需调用的接口，确定是否存在所述测试事件对应的报文模板；若存在所述测试事件对应的报文模板，则从所述测试事件中提取得到测试场景，并根据所述测试场景和所述测试事件对应的报文模板，得到所述测试事件对应的执行结果；若不存在所述测试事件对应的报文模板，则将所述测试事件发送给所述测试事件所需调用的外部接口，并获取所述外部接口执行所述测试事件得到的执行结果。

[0014] 在上述实现方式中，通过预先设置各个接口对应的报文模板，使得执行接口调用的测试事件能够直接根据调用的接口确定出测试结果，而无需人为地针对于每个测试事件修改测试程序，从而可以提高测试的效率；且，通过预设报文模板的形式来执行外部接口调用，还能避免将已存在报文模板的测试事件发送给外部接口，从而可以保证在外部接口不可用时仍能得到准确的测试结果，或者避免无用的操作，保证测试的顺利进行。

[0015] 在一种可能的实现方式中，所述根据所述测试场景和所述测试事件对应的报文模板，得到所述测试事件对应的执行结果，包括：根据所述测试场景从所述测试事件对应的报文模板中获取所述测试事件对应的初始报文结果，若所述初始报文结果中包括设定占位符，则从所述测试事件中提取得到所述设定占位符对应的测试内容，并使用所述测试内容替换所述初始报文结果中的设定占位符，得到所述测试事件对应的执行结果；若所述初始报文结果中不包括所述设定占位符，则将所述初始报文结果作为所述测试事件对应的执行结果。

[0016] 在上述实现方式中，通过使用测试数据中设定占位符对应的测试内容替换初始报文结果中的设定占位符，使得报文结果能够更加贴合测试事件的测试内容，如此，将报文结果作为测试事件的测试结果，可以使得测试结果更加清晰明确，且准确性更好。

[0017] 第二方面，本发明实施例提供的一种测试装置，所述装置包括：

[0018] 获取模块，用于获取测试案例；

[0019] 确定模块，用于根据所述测试案例中各个测试事件的类型，确定所述各个测试事件对应的功能插件；所述功能插件用于执行对所述测试事件进行测试的功能类函数，并且每个功能插件有其预设对应的测试事件的类型；

[0020] 加载模块，用于加载所述各个测试事件对应的功能插件；

[0021] 执行模块，用于运行所述测试案例，针对于所述测试案例中的任一测试事件，调用所述测试事件对应的已加载的功能插件执行所述测试事件，得到所述测试事件的执行结果；

[0022] 所述确定模块，还用于根据所述测试案例中各个测试事件的执行结果，确定所述测试案例的测试结果。

[0023] 在一种可能的实现方式中，所述执行模块具体用于：若所述测试事件对应的已加载的功能插件包括多个，则按照所述多个功能插件的优先级执行所述测试事件；所述多个功能插件的优先级用于标识执行所述测试事件的顺序。

[0024] 在一种可能的实现方式中，所述执行模块具体用于：若确定采用同步处理方式执行所述测试事件，则调用所述已加载的功能插件执行所述测试事件，若确定采用异步处理方式执行所述测试事件，则生成异步线程，并使用所述异步线程调用所述已加载的功能插件执行所述测试事件。

[0025] 在一种可能的实现方式中，所述执行模块具体用于：若所述已加载的功能插件为外部接口调用插件，则根据所述测试事件所需调用的接口，确定是否存在所述测试事件对应的报文模板；若存在所述测试事件对应的报文模板，则从所述测试事件中提取得到测试场景，并根据所述测试场景和所述测试事件对应的报文模板，得到所述测试事件对应的执行结果；若不存在所述测试事件对应的报文模板，则将所述测试事件发送给所述测试事件所需调用的外部接口，并获取所述外部接口执行所述测试事件得到的执行结果。

[0026] 在一种可能的实现方式中，所述执行模块具体用于：根据所述测试场景从所述测试事件对应的报文模板中获取所述测试事件对应的初始报文结果，若所述初始报文结果中包括设定占位符，则从所述测试事件中提取得到所述设定占位符对应的测试内容，并使用所述测试内容替换所述初始报文结果中的设定占位符，得到所述测试事件对应的执行结果；若所述初始报文结果中不包括所述设定占位符，则将所述初始报文结果作为所述测试事件对应的执行结果。

[0027] 第三方面，本发明实施例提供的一种计算设备，包括至少一个处理器以及至少一个存储器，其中，所述存储器存储有计算机程序，当所述程序被所述处理器执行时，使得所述处理器执行上述第一方面任意所述的测试方法。

[0028] 第四方面，本发明实施例提供的一种计算机可读存储介质，其存储有可由计算设备执行的计算机程序，当所述程序在所述计算设备上运行时，使得所述计算设备执行上述第一方面任意所述的测试方法。

[0029] 本发明的这些方面或其他方面在以下实施例的描述中会更加简明易懂。

附图说明

[0030] 为了更清楚地说明本发明实施例中的技术方案，下面将对实施例描述中所需要使用的附图作简要介绍，显而易见地，下面描述中的附图仅仅是本发明的一些实施例，对于本领域的普通技术人员来讲，在不付出创造性劳动性的前提下，还可以根据这些附图获得其他的附图。

[0031] 图1为本发明实施例提供的一种可能的系统架构示意图；

[0032] 图2为本发明实施例提供的一种测试方法对应的流程示意图；

[0033] 图3为本发明实施例提供的一种报文模板的存储形式示意图；

[0034] 图4为本发明实施例提供的一种测试装置的结构示意图；

[0035] 图5为本发明实施例提供的一种计算设备的结构示意图。

具体实施方式

[0036] 为了使本发明的目的、技术方案和优点更加清楚，下面将结合附图对本发明作进一步地详细描述，显然，所描述的实施例仅仅是本发明一部分实施例，而不是全部的实施例。基于本发明中的实施例，本领域普通技术人员在没有做出创造性劳动前提下所获得的所有其它实施例，都属于本发明保护的范围。

[0037] 图1为本发明实施例提供的一种可能的系统架构示意图，如图1所示，该系统中可以设置有至少一个业务系统，比如业务系统110、业务系统120和业务系统130；其中，业务系统可以为金融科技领域中的任意业务系统，比如订单系统、支付系统、购物车系统等。

[0038] 本发明实施例中，每个业务系统中可以设置有开发节点、测试节点和业务节点，开发节点可以分别与测试节点和业务节点连接，比如可以通过有线方式连接，或者也可以通过无线方式连接，具体不作限定。

[0039] 具体实施中，针对于任一业务系统，当该业务系统的开发人员在开发节点上开发了功能程序后，可以将功能程序发送给测试节点进行测试。在测试时，测试节点可以根据预先设置的测试需求生成各个测试案例，每个测试案例用于对功能程序中的一个或多个功能类函数进行测试，如此，当测试节点完成对各个测试案例的测试后，可以将各个测试案例的测试结果回传给开发节点。相应地，开发节点可以根据各个测试案例的测试结果确定功能程序中的各个功能类函数是否正常，若均正常，则可以将功能程序发送给业务节点，以使业务节点对功能程序进行上线。

[0040] 在一个示例中，任一业务系统的测试节点还可以连接其它业务系统的业务节点，如此，当某个测试案例存在对其它业务系统的业务接口的调用需求时，该业务系统的测试节点可以通过该连接关系调用其它业务系统的业务节点来完成对该测试案例的测试过程。举例来说，如图1所示，业务系统110中的测试节点112可以分别连接业务系统120中的业务节点123和业务系统130中的业务节点133，假设业务系统110为支付系统，业务系统120为购物车系统，测试节点112当前测试的测试案例为购买一部手机，则由于购买手机的流程包括将手机添加至购物车和对购物车中的手机付款这两个流程，因此在测试节点112使用开发节点111提供的支付程序对购物车中的手机付款之前，需要先调用购物车系统120将手机添加至购物车；因此，在使用开发节点111提供的支付程序测试测试案例时，测试节点112可以先调用业务节点123将手机添加至购物车，再使用开发节点111提供的支付程序对购物车中的手机进行付款，从而完成对测试案例的测试。

[0041] 需要说明的是，本发明实施例中的各节点可以是指服务器，或者也可以为服务器中的进程，具体不作限定。

[0042] 基于图1所示意的系统架构，图2为本发明实施例提供的一种测试方法对应的流程示意图，该方法应用于任一业务系统中的测试节点，该方法包括：

[0043] 步骤201，获取测试案例。

[0044] 本发明实施例中，测试案例用于对开发节点提供的待测试的功能程序(简称为待测试程序)进行测试，测试案例可以由测试人员根据待测试程序所能实现的功能进行设置，待测试程序中可以包含大量的功能类函数，而每个测试案例可以用于对待测试程序中的一个或多个功能类函数进行测试。

[0045] 举例来说，支付系统所能实现的功能包括调用其它业务系统的接口、与银行系统

的交互通信、支付操作、生成流水号、提供全球广域网(World Wide Web、web)服务等，因此支付系统的待测试程序中可以相应地包括外部接口调用类函数、下游系统信息收发类函数、支付类函数、流水号生成类函数和网络服务类函数，如此，测试节点可以根据外部接口调用类函数、支付类函数和流水号生成类函数生成一个测试案例，也可以根据接口调用类函数和下游系统信息收发类函数生成一个测试案例，还可以根据支付类函数、流水号生成类函数和网络服务类函数生成一个测试案例，等等。

[0046] 需要说明的是，本发明实施例中，由于各个业务系统之间具有相互依赖的关系(比如各个业务系统共同完成一个项目)，因此测试案例除了可以包括对该业务系统的待测试程序中的测试类函数的调用，还可以包括对该业务系统的已测试程序中的测试类函数的调用，或者还可以包括对其它业务系统的待测试程序中的测试类函数和/或已测试程序中的测试类函数的调用，具体不作限定。

[0047] 本发明实施例中，测试案例中可以包括一个或多个测试事件和一个或多个测试事件的执行流程，为了便于理解，下面示意出一个可能的测试案例：

[0048] 测试案例A：

[0049] {测试事件1：加购手机；

[0050] 测试事件2：加购手表；

[0051] 测试事件3：支付购物车中的手机和手表；}

[0052] 由此可知，在该示例中，测试案例A中设置有测试事件1、测试事件2和测试事件3，由于测试事件1为将手机添加到购物车，测试事件2为将手表添加到购物车，测试事件3为支付购物车中的手机和手表，因此，测试事件1和测试事件2均需要使用到加购类函数，测试事件3需要使用到支付类函数。相应地，这三个测试事件的执行流程为：先执行测试事件1，再执行测试事件2，最后执行测试事件3。

[0053] 需要说明的是，该测试案例仅是一种示例性的说明，并不构成对本方案的限定，具体实施中，测试案例中的每个测试事件也可以直接由功能类函数来表示，比如测试事件1在测试案例A中的表示形式为加购类函数和对应的参数(即手机)，测试事件2在测试案例A中的表示形式为加购类函数和对应的参数(即手表)，测试事件3在测试案例A中的表示形式为支付类函数和对应的参数(即手机和手表)。

[0054] 步骤202，根据所述测试案例中各个测试事件的类型，确定所述各个测试事件对应的功能插件；所述功能插件用于执行对所述测试事件进行测试的功能类函数，并且每个功能插件有其预设对应的测试事件的类型。

[0055] 本发明实施例中，各个业务系统的开发节点可以联合开发同一项目，每个开发节点可以开发该项目中的部分功能程序，并在调用对应的测试节点对各自开发的功能程序进行测试后，由各自的业务节点进行项目上线。相应地，各个业务系统的测试节点也可以基于各个业务系统开发的全部功能程序综合执行各自的测试工作，测试节点的测试工作可以基于测试框架来实现，测试框架中可以存储有各个业务系统在该项目中的全部项目代码，项目的全部项目代码包括每个业务系统在该项目中开发的功能程序。

[0056] 其中，测试框架可以为开源测试框架，比如Spring测试框架，或者也可以为各个业务系统联合设置的测试框架，具体不作限定。

[0057] 在一种可能的实现方式中，测试节点可以预先从各个业务系统的全部功能程序中

提取出各个功能类函数，并为各个功能类函数设置对应的功能插件，每个功能插件具有预先设置的测试事件的类型，每个功能插件用于实现其对应的功能类函数所独有的功能，不同的功能插件的功能可以不同。其中，功能插件可以由插件属性和插件方法来定义，插件属性用于标识该功能插件的属性信息，比如可以包括功能插件所需的XML配置文件的地址、功能插件所需的Properties配置文件的地址、功能插件能响应的测试事件的类型、功能插件的优先级等，插件方法用于标识使用该功能插件的方式，比如可以包括功能插件的加载方法、功能插件的优先级的获取方法、测试事件的执行方法等。

[0058] 具体实施中，测试节点提取得到各个功能插件后，可以将各个功能插件集成在测试框架的公共框架层中，公共框架层用于管理各个功能插件，比如控制每个功能插件的加载和/或断开、管理测试事件与功能插件之间的通讯、管理响应同一类型的测试事件的各个功能插件之间的依赖关系等。

[0059] 表1为一种各个功能插件的可能集成方式的示意表。

[0060] 表1:一种各个功能插件的集成方式的示意

[0061]

业务系统	测试事件的类型	功能插件集合	优先级
业务系统 110	支付类型	支付插件	0.7
		生成流水号插件	0.3
业务系统 120	加购类型	购物车插件	1

[0062] 如表1所示，功能插件在公共框架层中按照业务系统、测试事件的类型、功能插件集合的层级关系进行设置的，针对于任一测试事件的类型，该类型对应的功能插件集合中可以包括能够响应该类型的测试事件的全部功能插件，相应地，功能插件的优先级用于标识该功能插件与所在的功能插件集合中的其它功能插件的依赖关系。举例来说，支付类型的测试事件对应的功能插件集合中包含支付插件和生成流水号插件，由于支付插件的优先级0.7>生成流水号插件的优先级0.3，因此支付类型的测试事件被触发时，测试节点可以先调用支付插件执行测试事件得到支付插件的执行结果，然后再根据支付插件的执行结果调用生成流水号插件执行测试事件得到生成流水号插件的执行结果，其中，生成流水号插件的执行结果即为测试事件对应的执行结果。

[0063] 在一个示例中，在获取测试案例后，可以先确定出测试案例中的每个测试事件的类型，然后根据公共框架层中测试事件的类型与功能插件的对应关系，确定出测试案例中的每个测试事件对应的功能插件，如此，测试案例中的各个测试事件对应的功能插件即为执行该测试案例所需的全部功能插件。举例来说，若测试案例为测试案例A，则由于测试案例A中的测试事件1和测试事件2需要调用加购类函数，测试事件3需要调用支付类函数，因此，测试事件1和测试事件2属于加购类型的测试事件，对应的功能插件为购物车插件，测试事件3属于支付类型的测试事件，对应的功能插件为支付插件。

[0064] 以Spring测试框架为例，Spring测试框架中预先向用户提供了beforeTest Class方法类，beforeTest Class方法类在初始状态下为空，其支持用户在该方法类中自定义Spring测试框架启动之前的操作。如此，开发人员可以通过修改beforeTest Class方法类，使得Spring测试框架在执行测试案例之前，自动匹配出测试案例中的各个测试事件对应的

功能插件。

[0065] 在一个示例中，公共框架层中还可以设置有广播插件，当测试节点获取测试案例后，可以调用广播插件将测试案例中的每个测试事件广播给Spring测试框架。相应地，在Spring测试框架启动之前，修改后的beforeTest Class方法类可以获取广播插件广播的各个测试事件，并确定出各个测试事件的类型，进而可以对各个测试事件的类型中的重合类型进行剔除，得到测试事件的类型清单；进一步地，修改后的beforeTest Class方法类还可以从公共框架层中确定出与测试事件的类型清单中的每个测试事件类型匹配的功能插件，从而得到测试案例对应的功能插件清单。

[0066] 步骤203，加载所述各个测试事件对应的功能插件。

[0067] 具体实施中，可以从公共框架库中获取每个测试事件对应的功能插件的配置信息，然后将每个测试事件对应的功能插件的配置信息添加到测试框架的配置信息库中，如此，当测试框架启动时，测试框架可以根据配置信息库中的各个功能插件的配置信息加载各个功能插件。其中，功能插件的配置信息可以包括功能插件所需的XML配置文件地址、功能插件所需的Properties配置文件地址等；相应地，测试框架的配置信息库中还可以包括其它配置信息，比如必备中间件的配置信息、必备服务的配置信息等。

[0068] 举例来说，在得到测试案例对应的功能插件清单后，针对于功能插件清单中的每个功能插件，修改后的beforeTest Class方法类可以从公共框架库中获取该功能插件的配置信息，然后将该功能插件的配置信息添加到Spring测试框架的配置信息库中；如此，当修改后的beforeTest Class方法类执行完成后，Spring测试框架可以启动，在启动的过程中加载测试案例中的各个测试事件对应的功能插件。

[0069] 步骤204，运行所述测试案例，针对于所述测试案例中的任一测试事件，调用所述测试事件对应的已加载的功能插件执行所述测试事件，得到所述测试事件的执行结果。

[0070] 在一个示例中，当各个测试事件对应的功能插件加载完成后，针对于任一测试事件，测试节点可以先根据公共框架层中测试事件的类型与功能插件的对应关系，确定出该测试事件对应的功能插件，然后调用该测试事件对应的功能插件执行该测试事件。通过查询测试事件的类型与功能插件的对应关系，能够快速地确定出每个测试事件对应的功能插件，从而可以提高测试的效率。

[0071] 在另一个示例中，当各个测试事件对应的功能插件加载完成后，针对于任一测试事件，测试节点也可以调用广播插件向已加载的各个功能插件广播该测试事件。具体实施中，测试节点可以按照测试案例中各个测试事件的执行流程依次广播各个测试事件，比如若存在多个测试事件的执行流程为并行处理，则测试节点可以依次广播各个并行的测试事件，并在得到每个测试事件的执行结果之后再广播下一个并行的测试事件，或者也可以同时广播各个并行的测试事件，具体不作限定。

[0072] 相应地，已加载的每个功能插件在接收到任一测试事件的广播消息后，可以根据该测试事件的类型和该功能插件能响应的测试事件的类型，判断该功能插件是否可响应该测试事件，若不能，则可以不作处理，若能，则可以向测试节点发送成功响应的响应消息。

[0073] 在上述示例中，通过采用广播的方式确定测试事件对应的功能插件，能够避免未被加载的功能插件执行测试事件，还可以校验功能插件加载的准确性，提高测试的准确性。

[0074] 在一种可能的实现方式中，若测试事件对应的功能插件只有一个，则测试节点可

以直接调用该功能插件执行测试事件，并可以获取该功能插件执行测试事件得到的执行结果。若测试事件对应的功能插件存在多个，则测试节点可以按照多个功能插件的优先级依次调用多个功能插件执行测试事件；比如，由于测试案例A中的测试事件3同时对应支付插件和生成流水号插件，支付插件的优先级大于生成流水号插件的优先级，因此测试节点可以先调用支付插件执行测试事件1(即支付购物车中的手机和手表)，再根据支付插件的执行结果调用生成流水号插件执行测试事件1(即生成流水号)，从而得到测试事件3的执行结果。

[0075] 在上述实现方式中，响应同一类型的测试事件的各个功能插件的优先级能够标识各个功能插件的依赖关系，因此当某一测试事件对应多个功能插件时，通过按照多个功能插件的优先级依次执行测试事件，使得测试事件能够按照设定的依赖关系被准确的执行，从而提高测试的准确性。

[0076] 作为一种示例，测试节点在调用功能插件执行测试事件之前，还可以判断采用何种处理方式执行测试事件，若确定采用同步处理方式执行该测试事件，则可以直接调用该功能插件执行测试事件，并在得到测试事件的执行结果后，调用下一个功能插件执行该测试事件或下一个测试事件；若确定采用异步处理方式执行该测试事件，则可以创建异步进程，使用异步进程调用该功能插件执行测试事件，并使用原进程调用下一个功能插件执行该测试事件或下一个测试事件。

[0077] 其中，可以通过多种方式判断执行测试事件的处理方式，比如可以分析测试节点当前的性能损耗，若性能损耗较小，则可以采用异步处理方式执行测试事件，若性能损耗较大，则可以采用同步处理方式处理测试事件；或者也可以确定测试节点当前空闲的中央处理器(central processing unit，CPU)核数，若空闲的CPU核数大于设定核数，则可以采用异步处理方式执行测试事件，若空闲的CPU核数小于或等于设定核数，则可以采用同步处理方式处理测试事件；或者还可以统计测试节点当前的内存占用量，若内存占用量大于设定内存占用量，则可以采用异步处理方式执行测试事件，若内存占用量小于或等于设定的内存占用量，则可以采用同步处理方式处理测试事件，等等。

[0078] 在上述示例中，通过判断采用何种处理方式执行测试事件，使得处理方式能够更加符合系统的性能损耗情况，从而降低对系统的影响，避免系统卡顿，提高测试的准确性。

[0079] 本发明实施例中，测试节点可以通过功能插件的执行方法来调用功能插件，比如，测试节点可以先从测试事件中提取得到测试数据，然后将测试数据作为功能插件的执行方法的参数，从而调用功能插件的执行方法执行测试事件。举例来说，由于测试事件1为将手机添加至购物车，因此，测试节点可以获取加购插件中的加购方法，然后将手机作为加购方法的参数，从而完成对测试事件1的执行。

[0080] 在一种可能的实现方式中，若测试事件对应的功能插件为外部接口调用类插件，则测试节点可以先确定测试事件所需调用的接口，然后根据接口与报文模板的对应关系，确定是否存在测试事件对应的报文模板，若不存在，则可以将测试事件发送给对应的外部接口，并等待接收外部接口返回的执行结果；若存在，则可以使用测试事件对应的报文模板确定执行结果。其中，每个报文模板中可以设置有至少一个场景对应的初始报文结果，如此，若确定存在测试事件对应的报文模板，则测试节点可以先从测试事件的测试数据中提取得到测试场景，然后从测试事件对应的报文模板中获取测试场景对应的初始报文结果。

[0081] 本发明实施例中，接口与报文模板的对应关系可以为基于各个业务系统中各个接口的接口文档得到的，每个接口的接口文档中包括该接口的各个场景对应的初始报文结果。举例来说，针对于下单系统的下单接口，由于下单接口的场景包括下单成功场景、下单失败场景和下单超时场景，因此下单接口对应的报文模板中可以包括下单成功场景对应的初始报文结果、下单失败场景对应的初始报文结果和下单超时场景对应的初始报文结果。

[0082] 图3为发明实施例提供的一种报文模板的存储形式示意图，如图3所示，各个接口对应的报文模板可以按照接口-场景-初始报文结果的层级形式存储在测试节点的内存中，每个接口对应的报文模板可以为json类型的数据文件，报文模板的文件名中包括模板名称和接口标识，报文模板的文件内容中可以包括该接口的各个场景对应的初始报文结果。

[0083] 举例来说，针对于文件名为BFA_RSP0200466000.json的报文模板，其文件名中的BFA_RSP为模板名称，0200466000为接口标识，其文件内容中设置有“AUTH_LOAN_SUCC”场景对应的初始报文结果和“DEFAULT”场景对应的初始报文结果。

[0084] 本发明实施例中，通过预先根据各个接口文档设置各个接口对应的报文模板，使得执行接口调用的测试事件能够直接根据调用的接口确定出测试结果，而无需人为地针对于每个测试事件修改测试程序，从而可以提高测试的效率；且，通过预设报文模板的形式来执行外部接口调用，还能避免将已存在报文模板的测试事件发送给外部接口，从而可以保证在外部接口不可用时仍能得到准确的测试结果，或者避免无用的操作，保证测试的顺利进行。

[0085] 在一个示例中，在确定测试事件对应的初始报文结果后，可以检测初始报文结果中是否包含设定替换符，若不存在，则测试节点可以直接将测试事件对应的初始报文结果作为测试事件对应的执行结果，若存在，则测试节点可以从测试事件的测试数据中提取得到设定替换符对应的测试内容，然后使用设定替换符对应的测试内容替换初始报文结果中的设定替换符，得到报文结果，并将报文结果作为测试事件对应的执行结果。

[0086] 举例来说，表2为一种设定替换符对应的测试内容的可能的提取规则的示意表。

[0087] 表2：一种设定替换符对应的测试内容的提取规则的示意

[0088]

设定替换符	提取规则
@REQ.XXX	从测试数据中提取XXX对应的内容
@NOW	获取当前时间，默认格式yyyyMMdd；其他格式可以预先在设定替换符中指明，比如@NOW(yyyyMMddHHmmss)
@MOCK_SYS_ID	模拟生成新的流水号

[0089] 比如，当测试数据以键值对的形式存储时，若设定替换符为“@REQ.商品名称”，则可以从测试数据中提取键为“商品名称”的值，并使用该值替换测试事件对应的初始报文结果中的设定替换符“@REQ.商品名称”，针对于测试事件1，则使用“手机”替换初始报文结果中的“@REQ.商品名称”，针对于测试事件2，则使用“手表”替换初始报文结果中的“@REQ.商品名称”；若设定替换符为“@NOW”，则可以使用当前时间替换测试事件对应的初始报文结果中的设定替换符“@NOW”，比如若当前时间为2019年11月21日，则可以使用“20191121”替换

测试事件对应的初始报文结果中的设定替换符“@NOW”;若设定替换符为“@MOCK_SYS_ID”,则可以先模拟生成一个流水号,再使用该流水号替换测试事件对应的初始报文结果中的设定替换符“@MOCK_SYS_ID”。

[0090] 在上述示例中,通过使用测试数据中设定占位符对应的测试内容替换初始报文结果中的设定占位符,使得报文结果能够更加贴合测试事件的测试内容,如此,将报文结果作为测试事件的测试结果,可以使得测试结果更加清晰明确,且准确性更好。

[0091] 步骤205,根据测试案例中各个测试事件的测试结果,确定测试案例的测试结果。

[0092] 本发明实施例中,测试节点可以先根据各个测试事件的测试结果得到测试案例的测试结果,然后分析测试案例的测试结果的正确率,若正确率大于或等于预设正确率,则确定至少基于待测试程序所抽离的各个功能插件是准确的,若正确率小于预设正确率,则确定至少基于待测试程序所抽离的各个功能插件不准确,如此,测试节点可以根据测试案例的测试结果和正确率生成告警信息,并可以发送给开发节点,或者通过钉钉、微信、qq、邮箱、短信推动给开发人员。

[0093] 本发明的上述实施例中,获取测试案例,根据所述测试案例中各个测试事件的类型,确定所述各个测试事件对应的功能插件;所述功能插件用于执行对所述测试事件进行测试的功能类函数,并且每个功能插件有其预设对应的测试事件的类型;进一步地,加载所述各个测试事件对应的功能插件,运行所述测试案例,针对于所述测试案例中的任一测试事件,调用所述测试事件对应的已加载的功能插件执行所述测试事件,得到所述测试事件的执行结果,根据所述测试案例中各个测试事件的执行结果,确定所述测试案例的测试结果。本发明实施例中,通过预先为每个功能插件设置对应的测试事件的类型,使得在测试任意的测试案例时,能够仅加载该测试案例中所包括的测试事件的类型对应的功能插件,而无需加载测试案例不需要的功能插件,从而可以提高测试的灵活性,还能节省测试所需的系统资源,提高测试的效率。

[0094] 针对上述方法流程,本发明实施例还提供一种测试装置,该装置的具体内容可以参照上述方法实施。

[0095] 图4为本发明实施例提供的一种测试装置的结构示意图,包括:

[0096] 获取模块401,用于获取测试案例;

[0097] 确定模块402,用于根据所述测试案例中各个测试事件的类型,确定所述各个测试事件对应的功能插件;所述功能插件用于执行对所述测试事件进行测试的功能类函数,并且每个功能插件有其预设对应的测试事件的类型;

[0098] 加载模块403,用于加载所述各个测试事件对应的功能插件;

[0099] 执行模块404,用于运行所述测试案例,针对于所述测试案例中的任一测试事件,调用所述测试事件对应的已加载的功能插件执行所述测试事件,得到所述测试事件的执行结果;

[0100] 所述确定模块402,还用于根据所述测试案例中各个测试事件的执行结果,确定所述测试案例的测试结果。

[0101] 可选地,所述执行模块404具体用于:

[0102] 若所述测试事件对应的已加载的功能插件包括多个,则按照所述多个功能插件的优先级执行所述测试事件;所述多个功能插件的优先级用于标识执行所述测试事件的顺

序。

[0103] 可选地，所述执行模块404具体用于：

[0104] 若确定采用同步处理方式执行所述测试事件，则调用所述已加载的功能插件执行所述测试事件，若确定采用异步处理方式执行所述测试事件，则生成异步线程，并使用所述异步线程调用所述已加载的功能插件执行所述测试事件。

[0105] 可选地，所述执行模块404具体用于：

[0106] 若所述已加载的功能插件为外部接口调用插件，则根据所述测试事件所需调用的接口，确定是否存在所述测试事件对应的报文模板；

[0107] 若存在所述测试事件对应的报文模板，则从所述测试事件中提取得到测试场景，并根据所述测试场景和所述测试事件对应的报文模板，得到所述测试事件对应的执行结果；若不存在所述测试事件对应的报文模板，则将所述测试事件发送给所述测试事件所需调用的外部接口，并获取所述外部接口执行所述测试事件得到的执行结果。

[0108] 可选地，所述执行模块404具体用于：

[0109] 根据所述测试场景从所述测试事件对应的报文模板中获取所述测试事件对应的初始报文结果；

[0110] 若所述初始报文结果中包括设定占位符，则从所述测试事件中提取得到所述设定占位符对应的测试内容，并使用所述测试内容替换所述初始报文结果中的设定占位符，得到所述测试事件对应的执行结果；若所述初始报文结果中不包括所述设定占位符，则将所述初始报文结果作为所述测试事件对应的执行结果。

[0111] 从上述内容可以看出：本发明的上述实施例中，获取测试案例，根据所述测试案例中各个测试事件的类型，确定所述各个测试事件对应的功能插件；所述功能插件用于执行对所述测试事件进行测试的功能类函数，并且每个功能插件有其预设对应的测试事件的类型；进一步地，加载所述各个测试事件对应的功能插件，运行所述测试案例，针对于所述测试案例中的任一测试事件，调用所述测试事件对应的已加载的功能插件执行所述测试事件，得到所述测试事件的执行结果，根据所述测试案例中各个测试事件的执行结果，确定所述测试案例的测试结果。本发明实施例中，通过预先为每个功能插件设置对应的测试事件的类型，使得在测试任意的测试案例时，能够仅加载该测试案例中所包括的测试事件的类型对应的功能插件，而无需加载测试案例不需要的功能插件，从而可以提高测试的灵活性，还能节省测试所需的系统资源，提高测试的效率。

[0112] 基于同一发明构思，本发明实施例还提供一种计算设备，如图5所示，包括至少一个处理器501，以及与至少一个处理器连接的存储器502，本发明实施例中不限定处理器501与存储器502之间的具体连接介质，图5中处理器501和存储器502之间通过总线连接为例。总线可以分为地址总线、数据总线、控制总线等。

[0113] 在本发明实施例中，存储器502存储有可被至少一个处理器501执行的指令，至少一个处理器501通过执行存储器502存储的指令，可以执行前述的测试方法中所包括的步骤。

[0114] 其中，处理器501是计算设备的控制中心，可以利用各种接口和线路连接计算设备的各个部分，通过运行或执行存储在存储器502内的指令以及调用存储在存储器502内的数据，从而实现数据处理。可选的，处理器501可包括一个或多个处理单元，处理器501可集成

应用处理器和调制解调处理器，其中，应用处理器主要处理操作系统、用户界面和应用程序等，调制解调处理器主要处理下发指令。可以理解的是，上述调制解调处理器也可以不集成到处理器501中。在一些实施例中，处理器501和存储器502可以在同一芯片上实现，在一些实施例中，它们也可以在独立的芯片上分别实现。

[0115] 处理器501可以是通用处理器，例如中央处理器(CPU)、数字信号处理器、专用集成电路(Application Specific Integrated Circuit，ASIC)、现场可编程门阵列或者其他可编程逻辑器件、分立门或者晶体管逻辑器件、分立硬件组件，可以实现或者执行本发明实施例中公开的各方法、步骤及逻辑框图。通用处理器可以是微处理器或者任何常规的处理器等。结合测试实施例所公开的方法的步骤可以直接体现为硬件处理器执行完成，或者用处理器中的硬件及软件模块组合执行完成。

[0116] 存储器502作为一种非易失性计算机可读存储介质，可用于存储非易失性软件程序、非易失性计算机可执行程序以及模块。存储器502可以包括至少一种类型的存储介质，例如可以包括闪存、硬盘、多媒体卡、卡型存储器、随机访问存储器(Random Access Memory，RAM)、静态随机访问存储器(Static Random Access Memory，SRAM)、可编程只读存储器(Programmable Read Only Memory，PROM)、只读存储器(Read Only Memory，ROM)、带电可擦除可编程只读存储器(Electrically Erasable Programmable Read-Only Memory，EEPROM)、磁性存储器、磁盘、光盘等等。存储器502是能够用于携带或存储具有指令或数据结构形式的期望的程序代码并能够由计算机存取的任何其他介质，但不限于此。本发明实施例中的存储器502还可以是电路或者其它任意能够实现存储功能的装置，用于存储程序指令和/或数据。

[0117] 基于同一发明构思，本发明实施例还提供了一种计算机可读存储介质，其存储有可由计算设备执行的计算机程序，当所述程序在所述计算设备上运行时，使得所述计算设备执行图2任意所述的测试方法。

[0118] 本领域内的技术人员应明白，本发明的实施例可提供为方法、或计算机程序产品。因此，本发明可采用完全硬件实施例、完全软件实施例、或结合软件和硬件方面的实施例的形式。而且，本发明可采用在一个或多个其中包含有计算机可用程序代码的计算机可用存储介质(包括但不限于磁盘存储器、CD-ROM、光学存储器等)上实施的计算机程序产品的形式。

[0119] 本发明是参照根据本发明实施例的方法、设备(系统)、和计算机程序产品的流程图和/或方框图来描述的。应理解可由计算机程序指令实现流程图和/或方框图中的每一流程和/或方框、以及流程图和/或方框图中的流程和/或方框的结合。可提供这些计算机程序指令到通用计算机、专用计算机、嵌入式处理机或其他可编程数据处理设备的处理器以产生一个机器，使得通过计算机或其他可编程数据处理设备的处理器执行的指令产生用于实现在流程图一个流程或多个流程和/或方框图一个方框或多个方框中指定的功能的装置。

[0120] 这些计算机程序指令也可存储在能引导计算机或其他可编程数据处理设备以特定方式工作的计算机可读存储器中，使得存储在该计算机可读存储器中的指令产生包括指令装置的制造品，该指令装置实现在流程图一个流程或多个流程和/或方框图一个方框或多个方框中指定的功能。

[0121] 这些计算机程序指令也可装载到计算机或其他可编程数据处理设备上，使得在计

算机或其他可编程设备上执行一系列操作步骤以产生计算机实现的处理，从而在计算机或其他可编程设备上执行的指令提供用于实现在流程图一个流程或多个流程和/或方框图一个方框或多个方框中指定的功能的步骤。

[0122] 尽管已描述了本发明的优选实施例，但本领域内的技术人员一旦得知了基本创造性概念，则可对这些实施例作出另外的变更和修改。所以，所附权利要求意欲解释为包括优选实施例以及落入本发明范围的所有变更和修改。

[0123] 显然，本领域的技术人员可以对本发明进行各种改动和变型而不脱离本发明的精神和范围。这样，倘若本发明的这些修改和变型属于本发明权利要求及其等同技术的范围之内，则本发明也意图包含这些改动和变型在内。

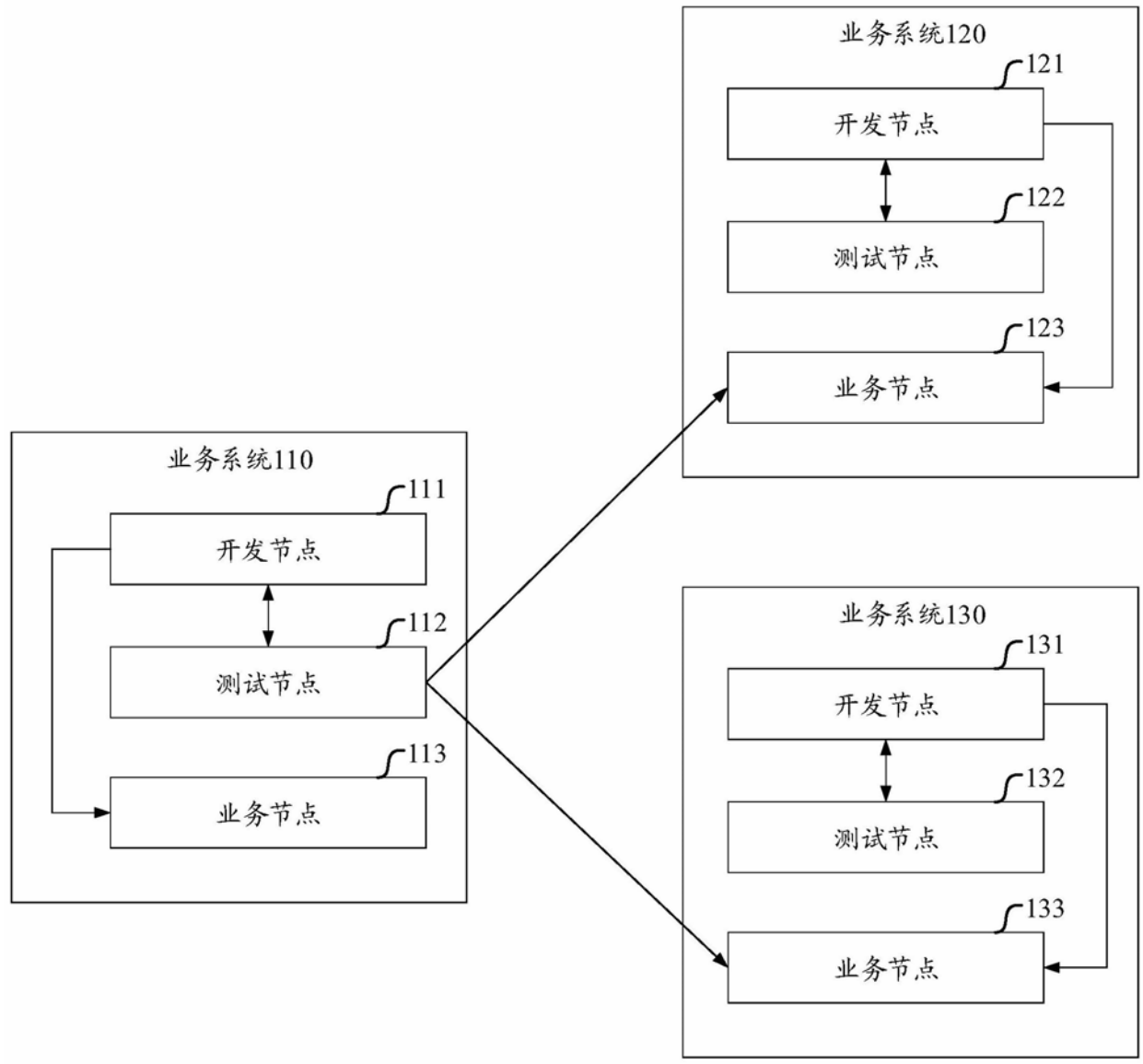

业务系统120
121
开发节点
122
测试节点
123
业务节点
业务系统110
111
开发节点
112
测试节点
113
业务节点
业务系统130
131
开发节点
132
测试节点
133
业务节点

图1

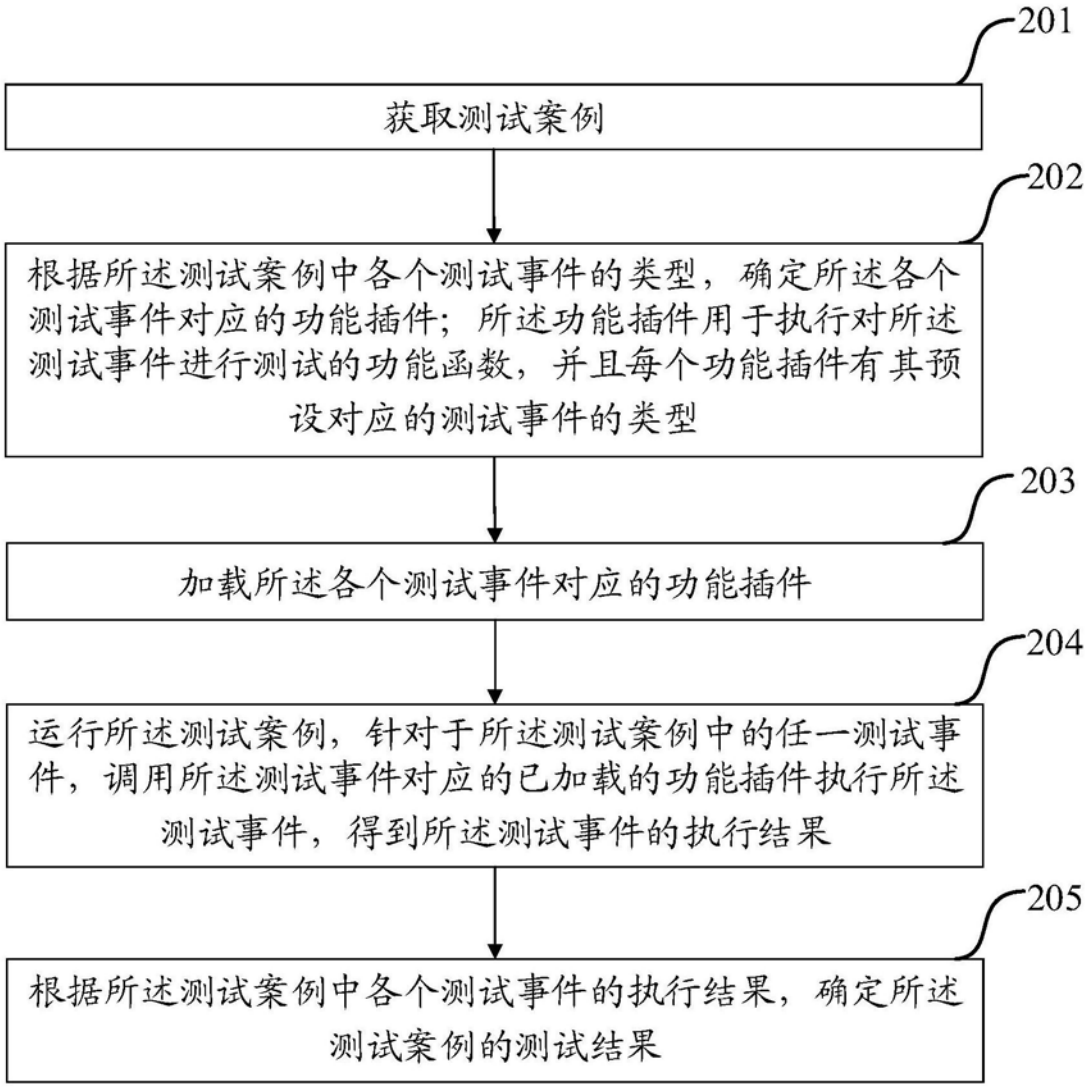

201
获取测试案例
202
根据所述测试案例中各个测试事件的类型，确定所述各个测试事件对应的功能插件；所述功能插件用于执行对所述测试事件进行测试的功能函数，并且每个功能插件有其预设对应的测试事件的类型
203
加载所述各个测试事件对应的功能插件
204
运行所述测试案例，针对于所述测试案例中的任一测试事件，调用所述测试事件对应的已加载的功能插件执行所述测试事件，得到所述测试事件的执行结果
205
根据所述测试案例中各个测试事件的执行结果，确定所述测试案例的测试结果

图2

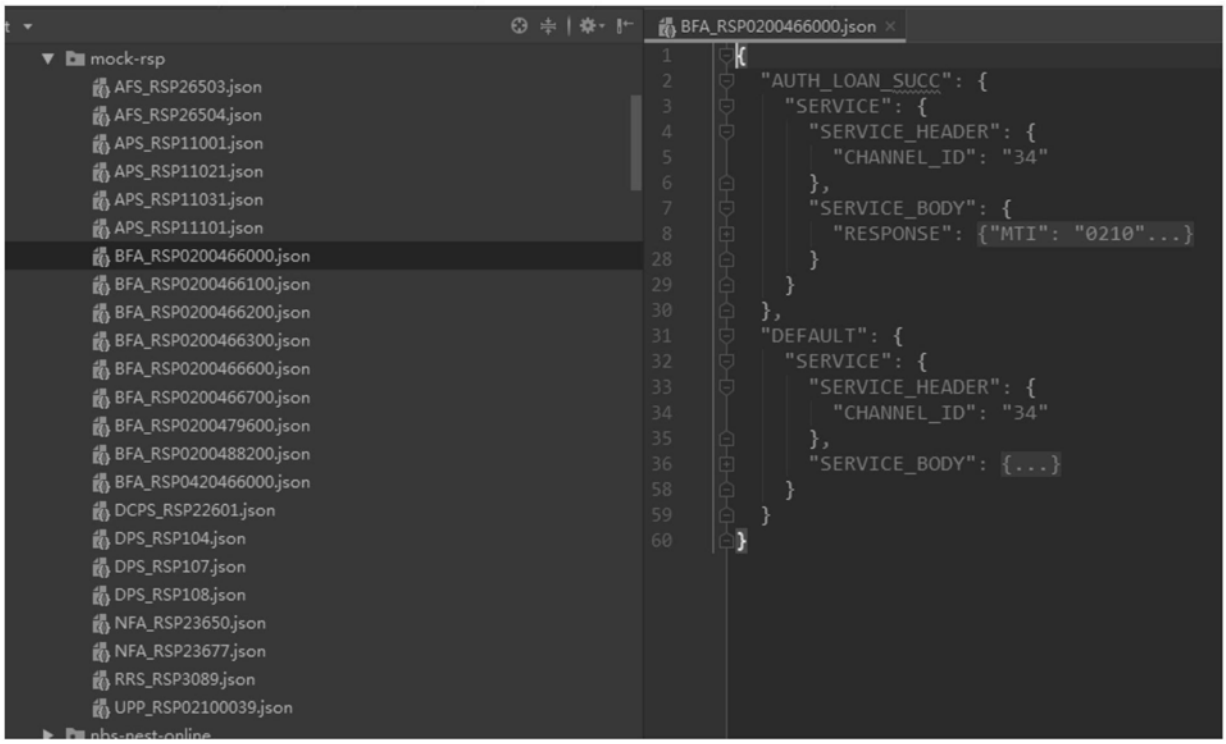

mock-rsp
AFS_RSP26503.json
AFS_RSP26504.json
APS_RSP11001.json
APS_RSP11021.json
APS_RSP11031.json
APS_RSP11101.json
BFA_RSP0200466000.json
BFA_RSP0200466100.json
BFA_RSP0200466200.json
BFA_RSP0200466300.json
BFA_RSP0200466600.json
BFA_RSP0200466700.json
BFA_RSP0200479600.json
BFA_RSP0200488200.json
BFA_RSP0420466000.json
DCPS_RSP22601.json
DPS_RSP104.json
DPS_RSP107.json
DPS_RSP108.json
NFA_RSP23650.json
NFA_RSP23677.json
RRS_RSP3089.json
UPP_RSP02100039.json
BFA_RSP0200466000.json
{
"AUTH_LOAN_SUCC": {
"SERVICE": {
"SERVICE_HEADER": {
"CHANNEL_ID": "34"
},
"SERVICE_BODY": {
"RESPONSE": {"MTI": "0210"...}
}
}
},
"DEFAULT": {
"SERVICE": {
"SERVICE_HEADER": {
"CHANNEL_ID": "34"
},
"SERVICE_BODY": {...}
}
}
}

图3

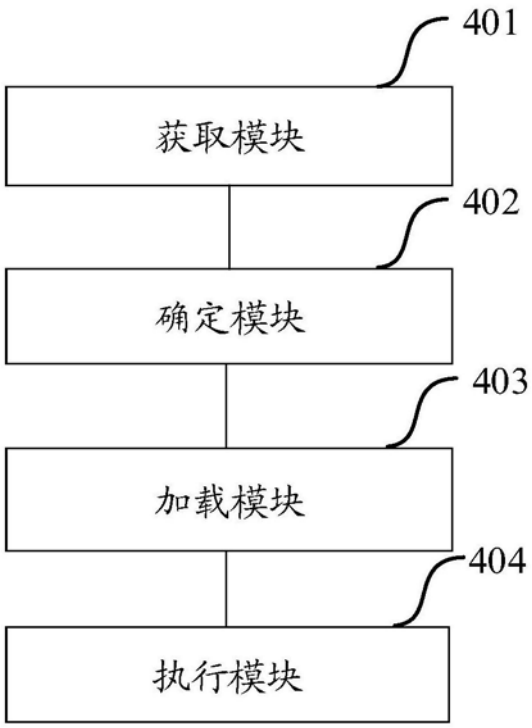

401
获取模块
402
确定模块
403
加载模块
404
执行模块

图4

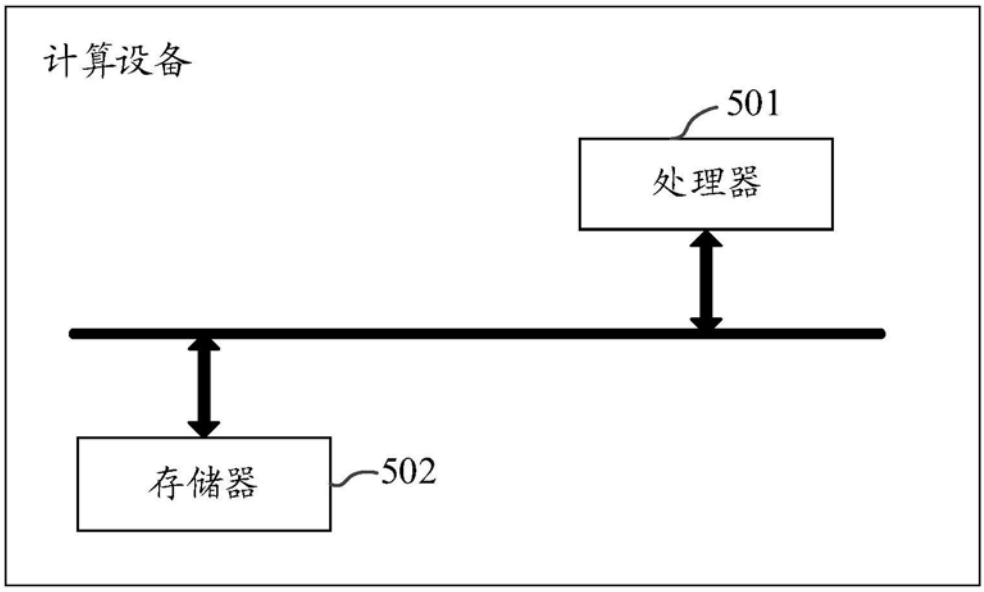


图5