[19] 中华人民共和国国家知识产权局

[51] Int. Cl.
G06Q 10/00 (2006.01)
G06F 9/44 (2006.01)

[12] 发明专利申请公布说明书

[21] 申请号 200710090499.X

[43] 公开日 2008 年 10 月 15 日

[11] 公开号 CN 101286212A

[22] 申请日 2007.4.12
[21] 申请号 200710090499.X
[71] 申请人 国际商业机器公司
地址 美国纽约
[72] 发明人 刘 英 朱 俊 谭 伟

[74] 专利代理机构 北京市中咨律师事务所
代理人 李 峥 刘瑞东

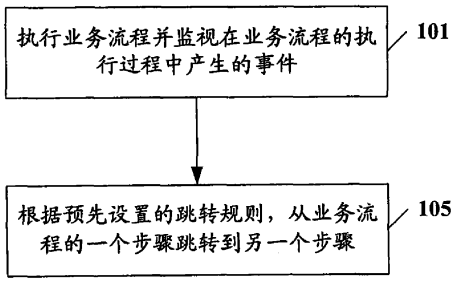
权利要求书 3 页 说明书 18 页 附图 6 页

[54] 发明名称

业务流程执行方法、业务流程引擎及其部署方法

[57] 摘要

本发明提供了业务流程的执行方法，业务流程引擎，部署业务流程引擎的方法，以及计算机程序产品。 根据本发明的一个方面，提供了一种业务流程的执行方法，其中，该业务流程是利用业务流程描述语言预先定义的，所述方法包括：执行上述业务流程并监视在上述业务流程的执行过程中产生的事件；以及根据预先设置的跳转规则，从上述业务流程的一个步骤跳转到另一个步骤，从而改变上述业务流程的预先定义的处理过程。



1. 一种业务流程的执行方法，其中，该业务流程利用业务流程描述语言预先定义，所述方法包括:

执行上述业务流程并监视在上述业务流程的执行过程中产生的事件;以及

根据预先设置的跳转规则，从上述业务流程的一个步骤跳转到另一个步骤，从而改变上述业务流程的预先定义的处理过程。

2. 根据权利要求 1 所述的业务流程的执行方法，其特征在于，所述跳转规则至少包括: 触发事件、跳转条件、跳转目的。

3. 根据权利要求 2 所述的业务流程的执行方法，其特征在于，所述从上述业务流程的一个步骤跳转到另一个步骤的步骤，包括:

判断监视到的上述跳转规则的触发事件是否满足该跳转规则的跳转条件; 以及

如果满足该跳转规则的跳转条件，则从上述业务流程的当前执行的步骤跳转到该跳转规则的跳转目的的步骤。

4. 根据权利要求 3 所述的业务流程的执行方法，其特征在于，还包括:

在所述从上述业务流程的当前执行的步骤跳转到该跳转规则的跳转目的的步骤之前，保存上述业务流程的当前的执行环境。

5. 根据权利要求 4 所述的业务流程的执行方法，其特征在于，所述跳转规则的跳转目的是利用偏移量设置的，从上述业务流程的当前执行的步骤跳转到该跳转规则的跳转目的的步骤包括:

确定上述业务流程的当前执行的步骤;

根据上述业务流程的当前执行的步骤和所述跳转规则的跳转目的的偏移量，计算跳转目的的步骤。

6. 根据权利要求 1－5 任意一项所述的业务流程的执行方法，其特征在于，所述预先设置的跳转规则包括多条跳转规则。

7. 根据权利要求 1-6 任意一项所述的业务流程的执行方法，其特征在于，针对一个服务组合设置一组跳转规则。

8. 根据权利要求 1-7 的任意一项所述的业务流程的执行方法，其特征在于，所述跳转规则是利用上述业务流程描述语言设置的。

9. 根据权利要求 7 或 8 所述的业务流程的执行方法，其特征在于，所述跳转规则是在服务组合创建过程中设置的。

10. 一种业务流程引擎，用于执行利用业务流程描述语言预先定义的业务流程，包括:

事件监视单元，用于监视在上述业务流程的执行过程中产生的事件;以及

跳转单元，用于根据预先设置的跳转规则，将上述业务流程从上述业务流程的一个步骤跳转到另一个步骤，从而改变上述业务流程的预先定义的处理过程。

11. 根据权利要求 10 所述的业务流程引擎，其特征在于，所述跳转规则至少包括: 触发事件、跳转条件、跳转目的。

12. 根据权利要求 11 所述的业务流程引擎，其特征在于，还包括:

条件判断单元，判断监视到的上述跳转规则的触发事件是否满足该跳转规则的跳转条件;

其中，如果满足该跳转规则的跳转条件，上述跳转单元将上述业务流程从上述业务流程的当前执行的步骤跳转到该跳转规则的跳转目的的步骤。

13. 根据权利要求 12 所述的业务流程引擎，其特征在于，还包括:

环境保存单元，用于在所述从上述业务流程的当前执行的步骤跳转到该跳转规则的跳转目的的步骤之前，保存上述业务流程的当前的执行环境。

14. 根据权利要求 13 所述的业务流程引擎，其特征在于，所述跳转规则的跳转目的是利用偏移量设置的; 上述跳转单元被设定为确定上述业务流程的当前执行的步骤，并且根据上述业务流程的当前执行的步骤和所述跳转规则的跳转目的的偏移量，计算跳转目的的步骤。

15. 根据权利要求 10－14 任意一项所述的业务流程引擎，其特征在于，所述预先设置的跳转规则包括多条跳转规则。

16. 根据权利要求 10－15 任意一项所述的业务流程引擎，其特征在于，针对一个服务组合设置一组跳转规则。

17. 根据权利要求 10－16 的任意一项所述的业务流程引擎，其特征在于，所述跳转规则是利用上述业务流程描述语言设置的。

18. 根据权利要求 16 或 17 所述的业务流程引擎，其特征在于，所述跳转规则是在服务组合创建过程中设置的。

19. 一种部署业务流程引擎的方法，该业务流程引擎用于执行利用业务流程描述语言预先定义的业务流程，并包括：

事件监视单元，用于监视在上述业务流程的执行过程中产生的事件；以及

跳转单元，用于根据预先设置的跳转规则，将上述业务流程从上述业务流程的一个步骤跳转到另一个步骤，从而改变上述业务流程的预先定义的处理过程。

业务流程执行方法、业务流程引擎及其部署方法

技术领域

本发明涉及数据处理技术，特别涉及解决业务流程之间的兼容性问题的技术。

背景技术

在当前的服务环境中，通过业务流程执行语言（Business Process Execution Language，BPEL）引擎，可以获得许多服务。目前，BPEL 引擎的产品有很多，例如包括 IBM（国际商业机器）公司的 WebSphere™ 业务流程服务器、Active Endpoints 公司的 ActiveBPEL™ 引擎、IBM 公司的 BPWS4J 产品、Oracle 公司的 BPEL 流程管理器、Bexee、Cape Clear 公司的 Orchestrator™、以及 Parasoft 公司的 BPEL Maestro™ 等等。通过这些 BPEL 引擎，可以获得各种各样服务。

但是，在实际的应用环境中，通常无法找到完全匹配的服务。通常的情况是用户可以找到一些所需的服务，但是这些服务包括一些不兼容的功能。

不管是在服务组合创建阶段（service composition build-time）还是在服务组合运行阶段（service composition run-time），服务不兼容都是一个极具挑战的问题。下面给出两个服务不兼容的例子。

（1）在服务组合创建阶段，由于存在不兼容的功能，一些服务不能直接与其它服务组合。然而，修改服务的成本又很高。

（2）在动态服务环境中，各个服务供应商通常会修改或升级他们的服务，从而会影响服务的用户。此外，在服务运行阶段，服务可能出现一些错误或异常问题，因此如何避免服务用户的损失就成为一个亟待解决的问

题。然而，由于修改、升级或异常而停止服务对于服务用户的损失巨大。

目前，除了由软件开发人员对应用程序进行修改以外，还没有在服务组合创建阶段不对服务进行修改而解决服务不兼容问题的方法。在服务组合运行阶段，大多数解决服务修改、升级或异常的方法是停止冲突服务。然而，修改服务或停止服务成本太高并影响太多服务用户。

发明内容

为了解决上述现有技术中存在的问题，本发明提供了业务流程的执行方法，业务流程引擎，部署业务流程引擎的方法，以及计算机程序产品。

根据本发明的一个方面，提供了一种业务流程的执行方法，其中，该业务流程利用业务流程描述语言预先定义，所述方法包括：执行上述业务流程并监视在上述业务流程的执行过程中（during the execution of the service workflow）产生的事件；以及根据预先设置的跳转规则，从上述业务流程的一个步骤跳转到另一个步骤，从而改变上述业务流程的预先定义的处理过程（process）。

根据本发明的另一个方面，提供了一种业务流程引擎，用于执行利用业务流程描述语言预先定义的业务流程，包括：事件监视单元，用于监视在上述业务流程的执行过程中产生的事件；以及跳转单元，用于根据预先设置的跳转规则，将上述业务流程从上述业务流程的一个步骤跳转到另一个步骤，从而改变上述业务流程的预先定义的处理过程。

根据本发明的另一个方面，提供了一种部署业务流程引擎的方法，该业务流程引擎用于执行利用业务流程描述语言预先定义的业务流程，并包括：事件监视单元，用于监视在上述业务流程的执行过程中产生的事件；以及跳转单元，用于根据预先设置的跳转规则，将上述业务流程从上述业务流程的一个步骤跳转到另一个步骤，从而改变上述业务流程的预先定义的处理过程。

根据本发明的另一个方面，提供了一种计算机程序产品，存储在计算机可用介质上，包括：计算机可读程序代码，用于使计算装置执行上述业

务流程的执行方法，其中，该业务流程是利用业务流程描述语言预先定义的，所述方法包括：执行上述业务流程并监视在上述业务流程的执行过程中产生的事件；以及根据预先设置的跳转规则，从上述业务流程的一个步骤跳转到另一个步骤，从而改变上述业务流程的预先定义的处理过程。

附图说明

相信通过以下结合附图对本发明具体实施方式的说明，能够使人们更好地了解本发明上述的特点、优点和目的。

图 1A 是根据本发明的一个实施例的业务流程的执行方法的流程图；

图 1B 是根据本发明的实施例的业务流程的执行方法的详细流程图；

图 2 是根据本发明的实施例的跳转过程的一个实例；

图 3 是根据本发明的实施例的跳转过程的另一个实例；

图 4 是根据本发明的实施例的跳转过程的另一个实例；

图 5 是根据本发明的另一个实施例的业务流程引擎的方框图。

具体实施方式

下面就结合附图对本发明的各个优选实施例进行详细的说明。

业务流程的执行方法

图 1A 是根据本发明的一个实施例的业务流程的执行方法的流程图。如图 1A 所示，首先，在步骤 101，执行业务流程并监视在业务流程的执行过程中产生的事件。其中，在该步骤执行的业务流程是利用例如业务流程描述语言（Business Process Specification Language，BPSL），又或称为业务流程建模语言（Business Process Modeling Language，BPML）预先定义的，然而，本实施例的业务流程也可以利用本领域的技术人员公知的其它语言或未来开发的语言定义，本发明对此没有任何限制。

在本实施例中，在业务流程的执行过程中产生的事件是指该业务的执行过程中发生的可以由业务流程引擎监视到的各种事件，例如订票服务发送中“接收用户信息”的事件，下面将对其进行详细描述。

例如，在采用 IBM 公司的 WebSphere™ 业务流程服务器的情况下，可以利用通用事件基础构架（Common Event Infrastructure，CEI）来监视在业务流程的执行过程中产生的事件。当然，本发明并不限于此，例如可以采用 Active Endpoints 公司的 ActiveBPEL™ 引擎、IBM 公司的 BPWS4J 产品、Oracle 公司的 BPEL 流程管理器、Bexee、Cape Clear 公司的 Orchestrator™、以及 Parasoft 公司的 BPEL Maestro™ 等等，已经它们相应的监视工具，只要是可以执行业务流程并监视在业务流程的执行过程中产生的事件即可。

最后，在步骤 105，根据预先设置的跳转规则，从上述业务流程的一个步骤跳转到另一个步骤。其中，预先设置的跳转规则利用上述业务流程描述语言根据事件-条件-动作（Event-Condition-Action，ECA）规则设置，然而，应该理解，可以利用本领域的技术人员公知的任何语言和规则设置跳转规则，只要能够清楚地描述跳转条件和跳转目的即可，本发明对此没有任何限制。

通常，业务流程服务器或引擎都具有服务调用功能，例如，在采用 IBM 公司的 WebSphere™ 业务流程服务器的情况下，可以利用适配动作管理器（Adaptive Action Manager）来实现上述跳转。本发明并不限于此，例如可以采用 Active Endpoints 公司的 ActiveBPEL™ 引擎、IBM 公司的 BPWS4J 产品、Oracle 公司的 BPEL 流程管理器、Bexee、Cape Clear 公司的 Orchestrator™、以及 Parasoft 公司的 BPEL Maestro™ 等等各自已有的服务调用功能；当然，也可以采用另外的程序来实现，总之，只要在需要的时候能够实现上述跳转功能即可，本发明对此没有任何限制。

本实施例的跳转规则主要包括以下三个方面：

事件（Event）用于定义该跳转规则的触发事件，也就是说，如果在上述步骤 101 中监视到在业务流程的执行过程中产生的事件与该触发事件一致时，则进一步判断是否满足该跳转规则的跳转条件；

条件（Condition）用于定义该跳转规则的跳转条件，如果满足该条件，则执行该跳转规则的动作，如果不满足条件，则执行该业务流程的当前步

骤;

动作（Action）用于定义该跳转规则的跳转目的，即应该跳转到的流程中的活动，也就是说，如果满足该跳转规则的跳转条件，则将上述业务流程从当前执行的步骤跳转到该跳转规则的跳转目的的步骤。

并表示为如下形式:

On_EVENT EventID IF ConditionExp DO ActionExp

其中 EventID 表示触发事件，ConditionExp 表示跳转条件，ActionExp 表示跳转目的。应该理解，这里给出跳转规则的只是一种示例性表示形式，可以利用本领域的技术人员公知的任何方式表示跳转规则，只要能够利用其实现跳转功能即可。

具体地，图 1B 示出了根据上述实施例的一个优选方式执行业务流程方法的详细流程图。在步骤 101 通过业务流程引擎（例如 IBM 公司的 Websphere Process Server）执行业务流程并通过事件监视器（例如 IBM 公司的 Wehsphere Business Monitor）监视在业务流程的执行过程中产生的事件之后，在步骤 1051，判断监视到的事件是否是上述跳转规则中定义的触发事件。如果在步骤 1051 监视到上述跳转规则中定义的触发事件，则进行到步骤 1052。否则，返回到步骤 101。

接着，在步骤 1052，判断是否满足该跳转规则中定义的跳转条件。如果在步骤 1052 判断满足该跳转规则的跳转条件，则进行到步骤 1053。否则，返回到步骤 101。

接着，在步骤 1053，确定上述业务流程的当前执行的步骤。

接着，在步骤 1054，根据上述业务流程的当前执行的步骤和该跳转规则的偏移量，计算跳转目的，其中设置偏移量的具体细节将在下面参考实例 2 和 3 进行详细描述。此外，应该理解，设置偏移量只是确定跳转目的的一种可选方式，本发明也可以不通过偏移量，而直接将跳转目的设置为上述业务流程的某个步骤，具体细节将在下面参考实例 1-3 进行描述。

最后，在步骤 1055，从上述业务流程的当前执行的步骤跳转到该跳转规则的跳转目的，从而实现本实施例的跳转功能。

下面通过实例 1-3 详细描述在步骤 105 中进行的跳转过程。

实例 1

图 2 是根据本发明的实施例的业务流程的执行方法中跳转过程的一个实例。在图 2 中，标号 201 表示旅行社的订票业务流程，标号 205 表示航空公司的业务流程。在航空公司的业务流程 205 中，为了给 VIP 用户和非 VIP 用户提供不同的服务，在步骤 2051（接收用户信息）和步骤 2053（接收旅行路线）之间引入一个判断步骤 2052，即在步骤 2052 中判断用户是否为 VIP 用户，如果不是，则执行步骤 2053，如果是 VIP 用户，则在另一分支为 VIP 用户提供一站式服务。顺便提及，在引入该判断步骤之后，在订票业务流程 201 和航空公司的业务流程 205 之间出现了不兼容。

为了解决二者之间的不兼容，在实例 1 中设置了如下跳转规则：

On_EVENT Accept_customer_information IF MsgFrom (Booking service) DO Jump (Accept_customer_information, Accept_itinerary)

其中：

触发事件（EventID）= Accept_customer_information（接收用户信息）

跳转条件（ConditionExp）= MsgFrom (Booking service)（消息是否来自于订票服务）

跳转目的（ActionExp）= Jump (Accept_customer_information, Accept_itinerary)（从接收用户信息跳转到接收旅行路线）

也就是说，本实施例的业务流程的执行方法在执行业务流程的过程中监视在业务流程的执行过程中产生的事件，如果监视到上述触发事件，即在业务流程 205 执行到步骤 2051 时，监视到 “接收用户信息”事件的发生，则判断跳转条件“消息是否来自于订票服务”。如果消息不是来自于订票业务流程 201，则业务流程 205 进行到步骤 2052。如果消息来自于订票业务流程 201，则执行该跳转规则的跳转步骤，即从业务流程 205 的当前步骤 2052 跳转到步骤 2053（接收旅行路线）。从而，通过本实施例的业务流程的执行方法避免了订票业务流程 201 和航空公司的业务流程 205 之间的不兼容。

实例 2

图 3 是根据本发明的实施例的业务流程的执行方法中跳转过程的另一个实例。在图 3 中，标号 301 表示没有设置跳转规则之前的订票业务流程，标号 305 表示设置跳转规则之后的订票业务流程。在订票业务流程 301 中，只有携带信用卡的用户可以使用该业务流程，为了让没有携带信用卡的用户使用订票业务流程 301，在实例 2 中设置了如下跳转规则：

On_EVENT Accept_customer_information DO Jump (Accept_customer_information, Ticket_service)

其中：

触发事件（EventID）= Accept_customer_information（接收用户信息）

跳转条件（ConditionExp）= 缺省（即只要监视到触发事件即执行跳转步骤）

跳转目的（ActionExp）= Jump (Accept_customer_information, Ticket_service)（从接收用户信息跳转到机票服务）

也就是说，本实施例的业务流程的执行方法在执行业务流程 301 的过程中监视在业务流程的执行过程中产生的事件，如果监视到上述触发事件，即在业务流程 301 执行到步骤 3011 时，监视到 “接收用户信息” 事件的发生，则执行该跳转规则的跳转步骤，即从业务流程 301 的当前步骤 3011 跳转到步骤 3013（机票服务）。从而，通过本实施例的业务流程的执行方法可以在没有携带信用卡时通过用户输入来使用业务流程 301。

实例 3

图 4 是根据本发明的实施例的业务流程的执行方法中跳转过程的另一个实例。在图 4 中，标号 401 表示没有设置跳转规则之前的订票业务流程，标号 405 表示设置跳转规则之后的订票业务流程。在订票业务流程 401 中，在机票服务之后检查付款情况，为了在机票服务之前检查付款，在实例 3 中设置了如下两个跳转规则：

跳转规则 1

On_EVENT Accept_customer_information DO Jump

(Accept_customer_information, Check_payment)

其中:

触发事件(EventID)=Accept_customer_information(接收用户信息)

跳转条件(ConditionExp)= 缺省(即只要监视到触发事件即执行跳转步骤)

跳转目的(ActionExp)= Jump (Accept_customer_information, Check_payment)(从接收用户信息跳转到检查付款)

跳转规则 2

On_EVENT Check_payment DO Jump (Check_payment, Ticket_service)

其中:

触发事件(EventID)=Check_payment(检查付款)

跳转条件(ConditionExp)= 缺省(即只要监视到触发事件即执行跳转步骤)

跳转目的(ActionExp)=Jump (Check_payment, Ticket_service)(从检查付款跳转到机票服务)

也就是说,本实施例的业务流程的执行方法在执行业务流程 401 的过程中监视在业务流程的执行过程中产生的事件,如果监视到上述跳转规则 1 的触发事件,即在业务流程 401 执行到步骤 4011 时,监视到 "接收用户信息" 事件的发生,则执行该跳转规则的跳转步骤,即从业务流程 401 的当前步骤 4011 跳转到步骤 4013(检查付款)。此外,如果监视到上述跳转规则 2 的触发事件,即在业务流程 401 执行到步骤 4013 时,监视到 "检查付款" 事件的发生,则执行该跳转规则的跳转步骤,即从业务流程 401 的当前步骤 4013 跳转到步骤 4012(机票服务)。从而,通过本实施例的业务流程的执行方法可以改变业务流程 401 中步骤 4012 和步骤 4013 的顺序。

在本实施例中,在从上述业务流程的当前执行的步骤跳转到该跳转规则的跳转目的的步骤之前,保存上述业务流程的当前的执行环境,例如在

上述实例 1 中，在从步骤 2051 跳转到步骤 2053 之前，保存包括在步骤 2051 处接收到的用户信息在内的与服务执行有关的全部环境变量，以在后续步骤中使用。

此外，在本实施例中，可以在服务组合创建阶段设置一个或多个上述跳转规则以避免两个或两个以上的服务之间的不兼容问题，也可以在服务运行阶段设置一个或多个上述跳转规则以解决服务可能出现的错误或异常问题。此外，可以针对一个服务或一个服务组合设置一个或一组跳转规则。然而，应该理解，只要需要，可以为任何服务、在服务的任何阶段设置任何跳转规则，可以在不修改可执行的服务本身的情况下，解决服务不兼容和冲突而不修改或停止服务，本发明对此没有任何限制。

可选地，在本实施例中，上述跳转规则的跳转目的可以利用偏移量设置，也就是说，根据上述业务流程的当前执行的步骤和所述跳转规则的跳转目的的偏移量，计算跳转目的的步骤。例如，在上述实例 2 中，将跳转规则的偏移量设置为向下 2 个步骤，在执行跳转步骤时，从业务流程 301 的当前步骤 3011 向下跳转 2 个步骤，即跳转到步骤 3013。又例如，在上述实例 3 中，将跳转规则 2 的偏移量可以设置为向上 1 个步骤，在执行跳转规则 2 的跳转步骤时，从业务流程 401 的当前步骤 4013 向上跳转 1 个步骤，即跳转到步骤 4011。

通过使用本实施例的业务流程的执行方法，在部分不兼容的服务进行组合时通过设置跳转规则，不需要对可执行的服务本身进行修改，从而极大地降低了服务开发和再利用的成本。

此外，通过使用本实施例的业务流程的执行方法，服务供应商可以在相对稳定的服务中设置跳转规则，从而不需要对可执行的服务本身进行修改即可为不同的用户提供不同的服务。

此外，通过使用本实施例的业务流程的执行方法，可以通过修改跳转规则方便地对服务进行升级和修改，而不需要升级和修改服务本身。

此外，通过使用本实施例的业务流程的执行方法，可以通过设置跳转规则解决服务错误和异常情况，从而极大地降低了停止服务带来的损失。

业务流程引擎

在同一发明构思下，图 5 是根据本发明的另一个实施例的业务流程引擎的方框图。下面就结合该图，对本实施例进行描述。对于那些与前面实施例相同的部分，适当省略其说明。

如图 5 所示，本实施例的业务流程引擎 500 用于执行利用业务流程描述语言预先定义的业务流程，包括：事件监视单元 501，用于监视在上述业务流程的执行过程中产生的事件；条件判断单元 505，当上述事件监视单元 501 监视到跳转规则的触发事件发生时，判断是否满足该跳转规则的跳转条件；以及跳转单元 510，用于根据预先设置的跳转规则，将上述业务流程从上述业务流程的一个步骤跳转到另一个步骤，从而改变上述业务流程的处理过程。

可选地，本实施例的业务流程引擎 500 还可以包括现有的业务流程引擎，例如 BPEL 引擎 525。也就是说，本实施例的业务流程引擎 500 可以基于 BPEL 引擎 525 进行扩展或改进而获得。BPEL 引擎 525，例如，包括 IBM 公司的 WebSphere™ 业务流程服务器、Active Endpoints 公司的 ActiveBPEL™ 引擎、IBM 公司的 BPWS4J 产品、Oracle 公司的 BPEL 流程管理器、Bexee、Cape Clear 公司的 Orchestrator™、以及 Parasoft 公司的 BPEL Maestro™ 等等。除了 BPEL 引擎 525，也可以基于本领域的技术人员公知的其它业务流程引擎进行扩展或改进，本发明对此没有任何限制。

可选地，本实施例的业务流程引擎 500 还可以包括现有的其它公知部件，例如通用事件基础构架（Common Event Infrastructure，CEI）515，它用于捕获在 BPEL 引擎 525 执行业务流程的过程中产生的事件。除了 CEI 515，也可以使用本领域的技术人员公知的其它产品实现此功能，例如 BEA Weblogic 公司的 Tuxedo™ 事件代理/监视器（Broker/Monitor），以及 Oracle 公司的业务活动监视器等等，本发明对此没有任何限制。

可选地，本实施例的业务流程引擎 500 还可以包括现有的其它公知部件，例如通用执行逻辑控制业务单元 520，它用于在跳转单元 510 和 BPEL

引擎525之间进行交互，从而实现跳转功能。

业务流程引擎500的其它公知部件的具体细节参见Dongsoo Han和Jaeyong Shim的文献“A Framework Supporting Dynamic Workflow Interoperation and Enterprise Application Integration”, Proceedings of the 35th Hawaii International Conference on System Sciences，2002，在此通过参考引入其整个内容（下文中称为文献1）。

在本实施例中，业务流程引擎500执行的业务流程是利用例如业务流程描述语言（Business Process Specification Language，BPSL），又或称为业务流程建模语言（Business Process Modeling Language，BPML）预先定义的，然而，本实施例的业务流程也可以利用本领域的技术人员公知的其它语言或未来开发的语言定义，本发明对此没有任何限制。

在本实施例中，事件监视单元501监视的在业务流程的执行过程中产生的事件是指该业务的执行过程中发生的可以由业务流程引擎监视到的各种事件，例如订票服务发送中“接收用户信息”的事件，下面将对其进行详细描述。

在本实施例中，预先设置的跳转规则利用上述业务流程描述语言根据事件-条件-动作（Event-Condition-Action，ECA）规则设置，然而，应该理解，可以利用本领域的技术人员公知的任何语言和规则设置跳转规则，只要在需要的时候能够实现跳转功能即可，本发明对此没有任何限制。

本实施例的跳转规则主要包括以下三个方面：

事件（Event）用于定义该跳转规则的触发事件，也就是说，如果在上述步骤101中监视到在业务流程的执行过程中产生的事件与该触发事件一致时，并进一步判断是否满足该跳转规则的跳转条件；

条件（Condition）用于定义该跳转规则的跳转条件，如果满足该条件，则执行该跳转规则的动作，如果不满足条件，则执行该业务流程的当前步骤；

动作（Action）用于定义该跳转规则的跳转目的，即应该跳转到的流程中的活动，也就是说，如果满足该跳转规则的跳转条件，则将上述业务

流程从当前执行的步骤跳转到该跳转规则的跳转目的的步骤。

并表示为如下形式:

On_EVENT EventID IF ConditionExp DO ActionExp

其中 EventID 表示触发事件，ConditionExp 表示跳转条件，ActionExp 表示跳转目的。应该理解，这里给出跳转规则的只是一种示例性表示形式，可以利用本领域的技术人员公知的任何方式表示跳转规则，只要能够利用其实现跳转功能即可。

下面通过实例 1-3 详细描述跳转单元 510 进行的跳转过程。

实例 1

图 2 是根据本发明的实施例的业务流程引擎 500 的跳转过程的一个实例。在图 2 中，标号 201 表示旅行社的订票业务流程，标号 205 表示航空公司的业务流程。在航空公司的业务流程 205 中，为了给 VIP 用户和非 VIP 用户提供不同的服务，在步骤 2051（接收用户信息）和步骤 2053（接收旅行路线）之间引入一个判断步骤 2052，即在步骤 2052 中判断用户是否为 VIP 用户，如果不是，则执行步骤 2053，如果是 VIP 用户，则在另一分支为 VIP 用户提供一站式服务。顺便提及，在引入该判断步骤之后，在订票业务流程 201 和航空公司的业务流程 205 之间出现了不兼容。

为了解决二者之间的不兼容，在实例 1 中设置了如下跳转规则:

On_EVENT Accept_customer_information IF MsgFrom (Booking service) DO Jump (Accept_customer_information, Accept_itinerary)

其中:

触发事件（EventID）= Accept_customer_information（接收用户信息）

跳转条件（ConditionExp）= MsgFrom (Booking service)（消息是否来自于订票服务）

跳转目的（ActionExp）= Jump (Accept_customer_information, Accept_itinerary)（从接收用户信息跳转到接收旅行路线）

也就是说，本实施例的业务流程引擎 500 在执行业务流程的过程中利用事件监视单元 501 监视在业务流程的执行过程中产生的事件，并利用条

件判断单元505判断当上述事件监视单元501监视到上述触发事件，即在业务流程205执行到步骤2051时，监视到“接收用户信息”事件的发生时，是否满足跳转条件“消息是否来自于订票服务”。如果消息不是来自于订票业务流程201，则业务流程205进行到步骤2052。如果消息来自于订票业务流程201，则利用跳转单元510执行该跳转规则的跳转步骤，即从业务流程205的当前步骤2052跳转到步骤2053（接收旅行路线）。从而，通过本实施例的业务流程引擎500避免了订票业务流程201和航空公司的业务流程205之间的不兼容。

实例2

图3是根据本发明的实施例的业务流程引擎500的跳转过程的另一个实例。在图3中，标号301表示没有设置跳转规则之前的订票业务流程，标号305表示设置跳转规则之后的订票业务流程。在订票业务流程301中，只有携带信用卡的用户可以使用该业务流程，为了让没有携带信用卡的用户使用订票业务流程301，在实例2中设置了如下跳转规则：

On_EVENT Accept_customer_information DO Jump (Accept_customer_information, Ticket_service)

其中：

触发事件（EventID）= Accept_customer_information（接收用户信息）

跳转条件（ConditionExp）= 缺省（即只要监视到触发事件即执行跳转步骤）

跳转目的（ActionExp）= Jump (Accept_customer_information, Ticket_service)（从接收用户信息跳转到机票服务）

也就是说，本实施例的业务流程引擎500在执行业务流程301的过程中利用事件监视单元501监视在业务流程的执行过程中产生的事件，并利用条件判断单元505判断当上述事件监视单元501监视到上述触发事件，即在业务流程301执行到步骤3011时，监视到“接收用户信息”事件的发生时，利用跳转单元510执行该跳转规则的跳转步骤，即从业务流程301的当前步骤3011跳转到步骤3013（机票服务）。从而，通过本实施例的

业务流程引擎 500 可以在没有携带信用卡时通过用户输入来使用业务流程 301。

实例 3

图 4 是根据本发明的实施例的业务流程引擎 500 的跳转过程的另一个实例。在图 4 中，标号 401 表示没有设置跳转规则之前的订票业务流程，标号 405 表示设置跳转规则之后的订票业务流程。在订票业务流程 401 中，在机票服务之后检查付款情况，为了在机票服务之前检查付款，在实例 3 中设置了如下两个跳转规则：

跳转规则 1

On_EVENT Accept_customer_information DO Jump (Accept_customer_information, Check_payment)

其中：

触发事件（EventID）= Accept_customer_information（接收用户信息）

跳转条件（ConditionExp）= 缺省（即只要监视到触发事件即执行跳转步骤）

跳转目的（ActionExp）= Jump (Accept_customer_information, Check_payment)（从接收用户信息跳转到检查付款）

跳转规则 2

On_EVENT Check_payment DO Jump (Check_payment, Ticket_service)

其中：

触发事件（EventID）= Check_payment（检查付款）

跳转条件（ConditionExp）= 缺省（即只要监视到触发事件即执行跳转步骤）

跳转目的（ActionExp）= Jump (Check_payment, Ticket_service)（从检查付款跳转到机票服务）

也就是说，本实施例的业务流程引擎 500 在执行业务流程 401 的过程中利用事件监视单元 501 监视在业务流程的执行过程中产生的事件，并利

用条件判断单元 505 判断当上述事件监视单元 501 监视到上述跳转规则 1 的触发事件，即在业务流程 401 执行到步骤 4011 时，监视到 “接收用户信息”事件的发生时，利用跳转单元 510 执行该跳转规则的跳转步骤，即从业务流程 401 的当前步骤 4011 跳转到步骤 4013（检查付款）。此外，利用条件判断单元 505 判断当上述事件监视单元 501 监视到上述跳转规则 2 的触发事件，即在业务流程 401 执行到步骤 4013 时，监视到 “检查付款”事件的发生时，利用跳转单元 510 执行该跳转规则的跳转步骤，即从业务流程 401 的当前步骤 4013 跳转到步骤 4012（机票服务）。从而，通过本实施例的业务流程引擎 500 可以改变业务流程 401 中步骤 4012 和步骤 4013 的顺序。

在本实施例中，业务流程引擎 500 还包括环境保存单元，用于在上述跳转单元 510 进行从上述业务流程的当前执行的步骤跳转到该跳转规则的跳转目的流程中的步骤之前，保存上述业务流程的当前的执行环境，例如在上述实例 1 中，在从步骤 2051 跳转到步骤 2053 之前，保存包括在步骤 2051 处接收到的用户信息在内的与服务执行有关的全部环境变量，以在后续步骤中使用。

此外，在本实施例中，业务流程引擎 500 可以在服务组合创建阶段设置一个或多个上述跳转规则以避免两个或两个以上的服务之间的不兼容问题，也可以在服务运行阶段设置一个或多个上述跳转规则以解决服务可能出现的错误或异常问题。此外，业务流程引擎 500 可以针对一个服务或一个服务组合设置一个或一组跳转规则。然而，应该理解，只要需要，业务流程引擎 500 可以为任何服务、在服务的任何阶段设置任何跳转规则，可以在不修改可执行的服务本身的情况下，解决服务不兼容和冲突而不修改或停止服务，本发明对此没有任何限制。

可选地，在本实施例中，上述跳转规则的跳转目的可以利用偏移量设置，也就是说，根据上述业务流程的当前执行的步骤和所述跳转规则的跳转目的的偏移量，计算上述跳转单元 510 的跳转目的。例如，在上述实例 2 中，将跳转规则的偏移量设置为向下 2 个步骤，在执行跳转步骤时，从

业务流程 301 的当前步骤 3011 向下跳转 2 个步骤，即跳转到步骤 3013。又例如，在上述实例 3 中，将跳转规则 2 的偏移量可以设置为向上 1 个步骤，在执行跳转规则 2 的跳转步骤时，从业务流程 401 的当前步骤 4013 向上跳转 1 个步骤，即跳转到步骤 4011。

通过使用本实施例的业务流程引擎 500，在部分不兼容的服务进行组合时通过设置跳转规则，不需要对可执行的服务本身进行修改，从而极大地降低了服务开发和再利用的成本。

此外，通过使用本实施例的业务流程引擎 500，服务供应商可以在相对稳定的服务中设置跳转规则，从而不需要对可执行的服务本身进行修改即可为不同的用户提供不同的服务。

此外，通过使用本实施例的业务流程引擎 500，可以通过修改跳转规则方便地对服务进行升级和修改，而不需要升级和修改服务本身。

此外，通过使用本实施例的业务流程引擎 500，可以通过设置跳转规则解决服务错误和异常情况，从而极大地降低了停止服务带来的损失。

部署业务流程引擎的方法

在同一发明构思下，根据本发明的另一个实施例提供了一种部署业务流程引擎的方法。下面就对本实施例进行描述，对于那些与前面实施例相同的部分，适当省略其说明。

在本实施例的部署业务流程引擎的方法中，业务流程引擎用于执行利用业务流程描述语言预先定义的业务流程，并包括：事件监视单元，用于监视在上述业务流程的执行过程中产生的事件；以及跳转单元，用于根据预先设置的跳转规则，将上述业务流程从上述业务流程的一个步骤跳转到另一个步骤，从而改变上述业务流程的处理过程。此外，本实施例的方法部署的业务流程引擎还可以是上述参考图 5 的实施例中描述的业务流程引擎 500。

本实施例的部署业务流程引擎的方法的具体细节参见上述 Dongsoo Han 和 Jaeyong Shim 的文献 1，在此不再赘述。

通过使用本实施例的部署业务流程引擎的方法，在部分不兼容的服务

进行组合时通过设置跳转规则，不需要对可执行的服务本身进行修改，从而极大地降低了服务开发和再利用的成本。

此外，通过使用本实施例的部署业务流程引擎的方法，服务供应商可以在相对稳定的服务中设置跳转规则，从而不需要对可执行的服务本身进行修改即可为不同的用户提供不同的服务。

此外，通过使用本实施例的部署业务流程引擎的方法，可以通过修改跳转规则方便地对服务进行升级和修改，而不需要升级和修改服务本身。

此外，通过使用本实施例的部署业务流程引擎的方法，可以通过设置跳转规则解决服务错误和异常情况，从而极大地降低了停止服务带来的损失。

计算机程序产品

在同一发明构思下，根据本发明的另一个实施例提供了一种计算机程序产品。下面就对本实施例进行描述，对于那些与前面实施例相同的部分，适当省略其说明。

本实施例的计算机程序产品，存储在计算机可用介质上，包括：计算机可读程序代码，用于使计算装置执行业务流程的执行方法，其中，该业务流程是利用业务流程描述语言预先定义的，所述方法包括：执行上述业务流程并监视在上述业务流程的执行过程中产生的事件；以及根据预先设置的跳转规则，从上述业务流程的一个步骤跳转到另一个步骤，从而改变上述业务流程的处理过程。此外，本实施例的计算机程序产品的计算机程序代码也可以用于使计算装置执行上述参考图 1A 的实施例中描述的业务流程的执行方法。

通过使用本实施例的计算机程序产品，在部分不兼容的服务进行组合时通过设置跳转规则，不需要对可执行的服务本身进行修改，从而极大地降低了服务开发和再利用的成本。

此外，通过使用本实施例的计算机程序产品，服务供应商可以在相对稳定的服务中设置跳转规则，从而不需要对可执行的服务本身进行修改即可为不同的用户提供不同的服务。

此外，通过使用本实施例的计算机程序产品，可以通过修改跳转规则方便地对服务进行升级和修改，而不需要升级和修改服务本身。

此外，通过使用本实施例的计算机程序产品，可以通过设置跳转规则解决服务错误和异常情况，从而极大地降低了停止服务带来的损失。

以上虽然通过一些示例性的实施例对本发明的业务流程的执行方法，业务流程引擎，部署业务流程引擎的方法，以及计算机程序产品进行了详细的描述，但是以上这些实施例并不是穷举的，本领域技术人员可以在本发明的精神和范围内实现各种变化和修改。因此，本发明并不限于这些实施例，本发明的范围仅由所附权利要求为准。

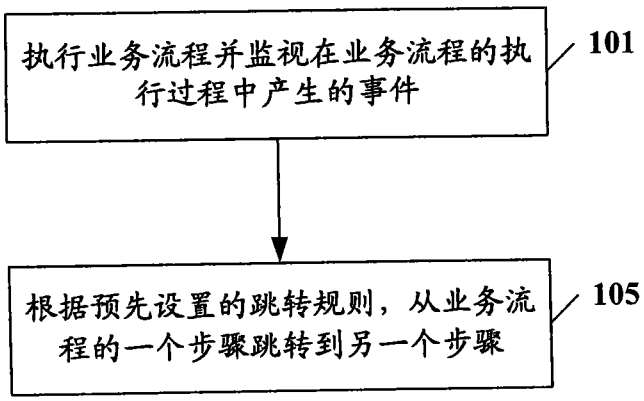

执行业务流程并监视在业务流程的执行过程中产生的事件
101
根据预先设置的跳转规则，从业务流程的一个步骤跳转到另一个步骤
105

图1A

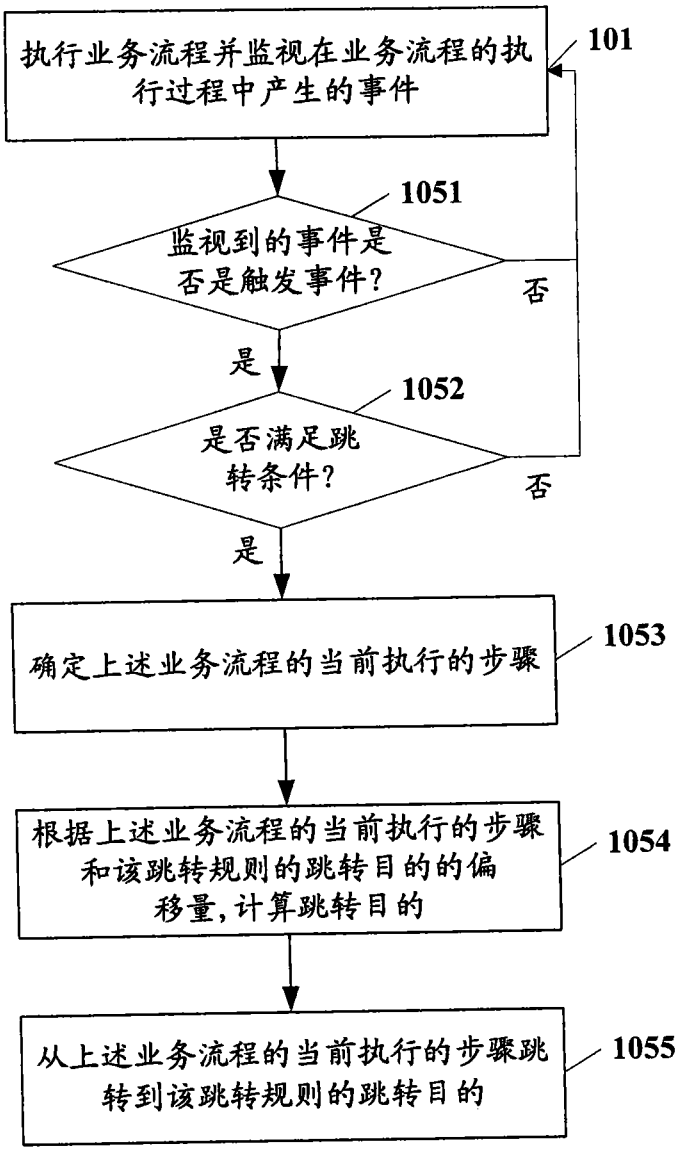

执行业务流程并监视在业务流程的执行过程中产生的事件
101
1051
监视到的事件是否是触发事件?
否
是
1052
是否满足跳转条件?
否
是
确定上述业务流程的当前执行的步骤
1053
根据上述业务流程的当前执行的步骤和该跳转规则的跳转目的的偏移量,计算跳转目的
1054
从上述业务流程的当前执行的步骤跳转到该跳转规则的跳转目的
1055

图1B

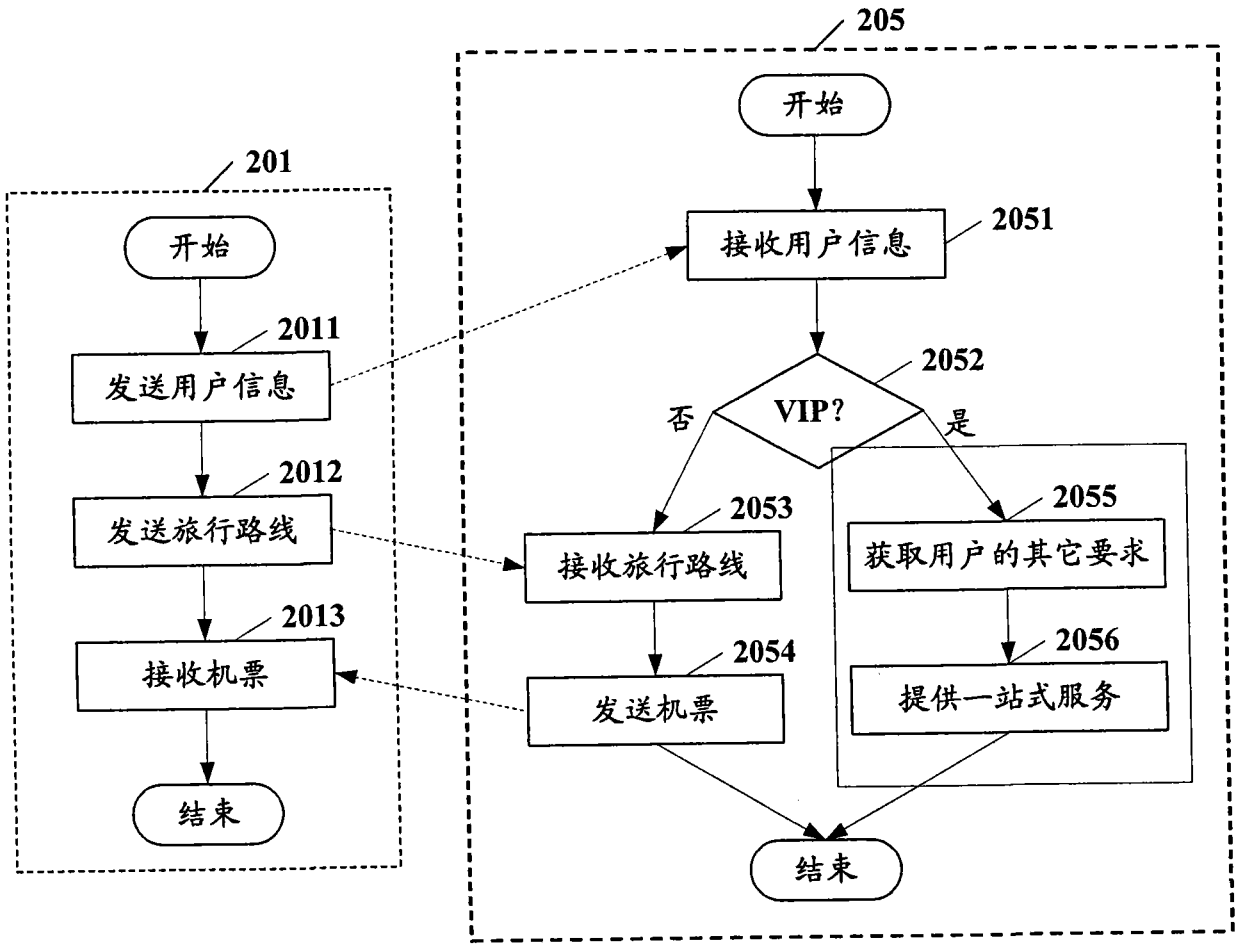

205
开始
201
开始
接收用户信息
2051
2011
发送用户信息
2052
VIP?
否
是
2012
发送旅行路线
2053
接收旅行路线
2055
获取用户的其它要求
2013
接收机票
2054
发送机票
2056
提供一站式服务
结束
结束

图2

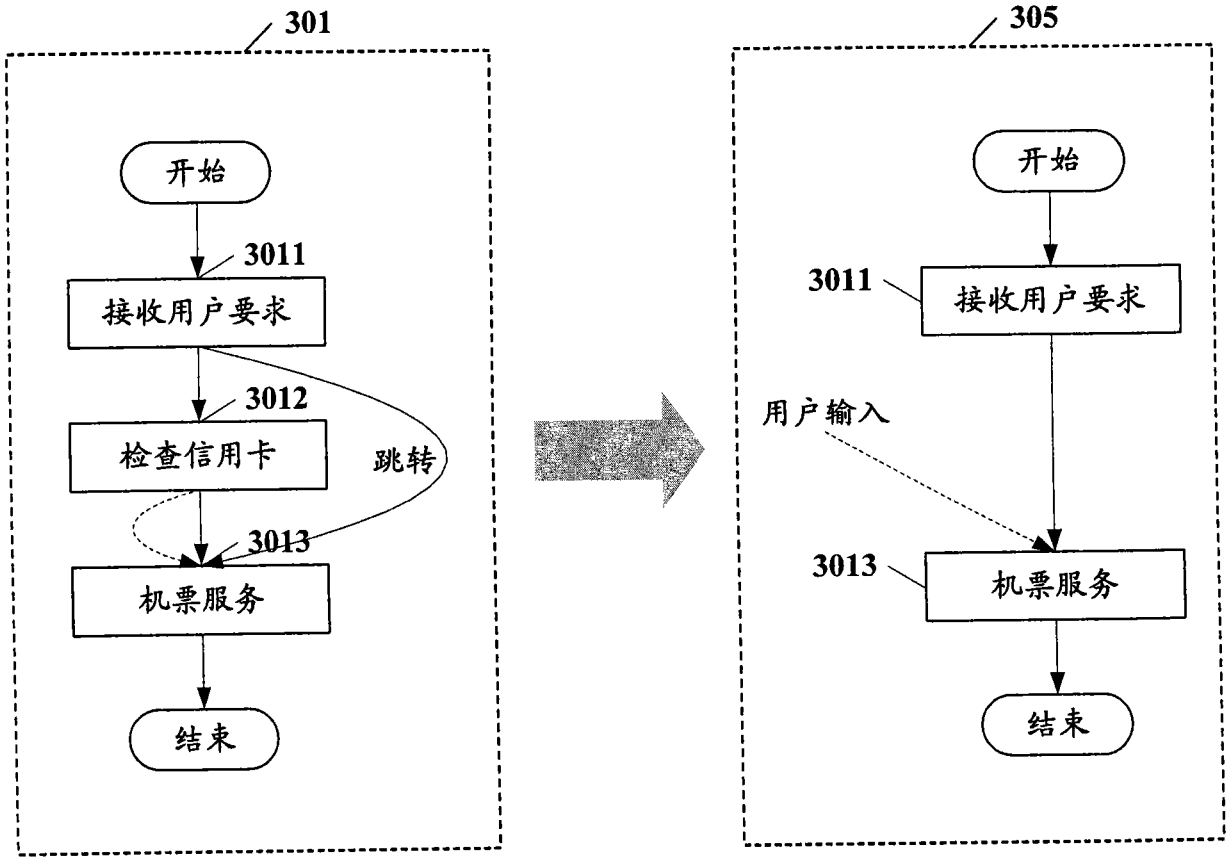

301
开始
3011
接收用户要求
3012
检查信用卡
跳转
3013
机票服务
结束
305
开始
3011
接收用户要求
用户输入
3013
机票服务
结束

图3

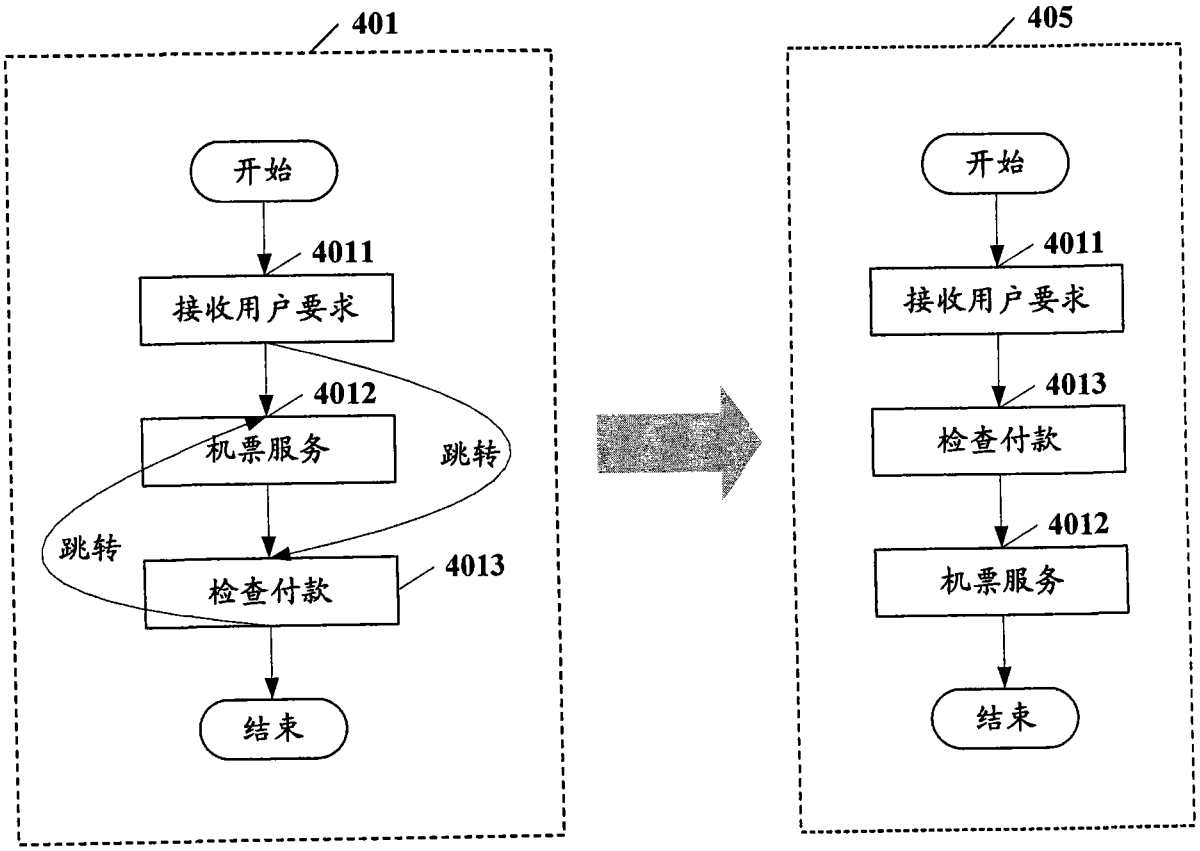

401
开始
4011
接收用户要求
4012
机票服务
跳转
跳转
检查付款
4013
结束
405
开始
4011
接收用户要求
4013
检查付款
4012
机票服务
结束

图4

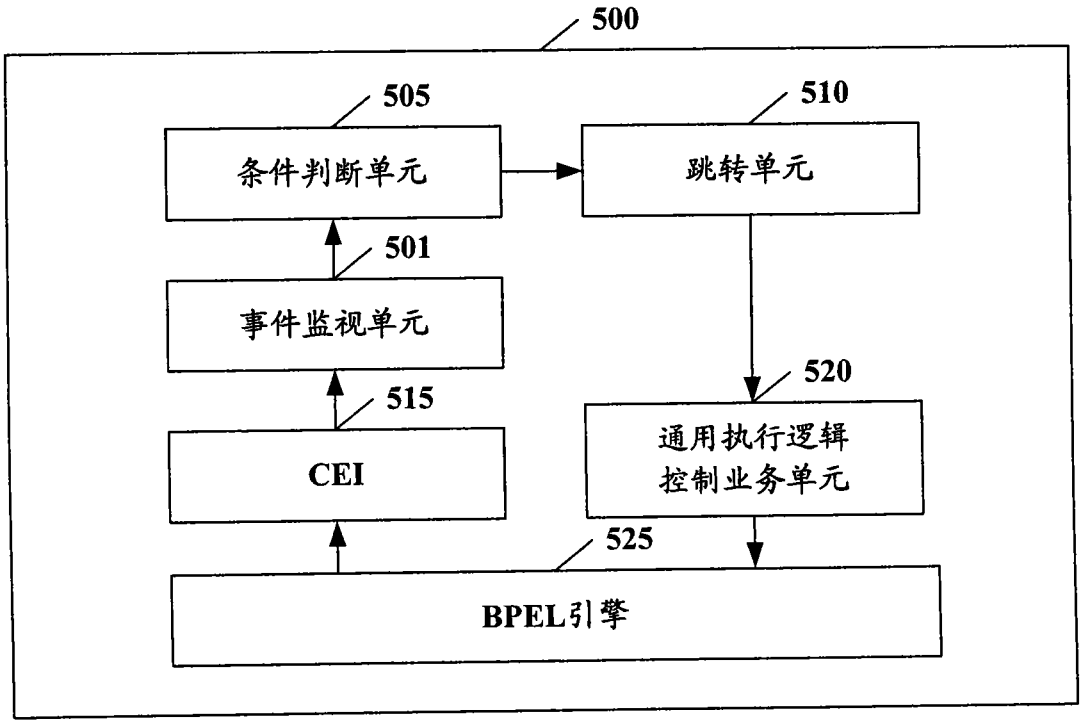

500
505
条件判断单元
510
跳转单元
501
事件监视单元
515
CEI
520
通用执行逻辑
控制业务单元
525
BPEL引擎

图5