

一种业务处理方法及装置

技术领域

本申请涉及计算机技术领域，尤其涉及一种业务处理方法及装置。

背景技术

在线系统（如：网站）在其后台的业务系统的支持下，可向用户提供丰富的业务服务。其中的某些业务可能需要不同的业务系统共同完成。

现有技术中，由用户通过诸如浏览器、应用（Application，APP）等客户端所发起的业务请求，那么，对上述的业务请求的处理过程具体可为：按照业务流程的顺序，先对该业务请求进行处理的业务系统会生成相应的处理结果（可称为中间结果），返回给客户端，客户端根据中间结果，重定向至业务流程中的下一业务系统，并发起请求（可称为中间请求），使得下一业务系统进行后续处理，以此类推，直到完成整个业务流程为止。

例如：假设客户端所发出的某业务请求需要业务系统 A 和 B 共同完成。按照业务流程，该业务请求首先由业务系统 A 进行处理，生成中间结果 a，此时，业务系统 A 会将中间结果 a 返回给客户端，该客户端基于中间结果 a 重定向至业务系统 B，并进一步向业务系统 B 发出请求，从而，业务系统 B 会对该请求处理，生成业务结果 b，并返回给客户端。

但是，在现有技术的方式中，各业务系统与客户端之间通过互联网进行交互，而互联网的网络环境的稳定性较差，可能由于网络环境的影响，产生网络延时，导致客户端根据中间业务结果向下一业务系统发送中间请求的过程需要耗费较长的时间；此外，上述的方式中，业务系统只会在接收到了客户端的请求后，才进行相应的处理，而在线系统通常会面对大量的用户访问，造成各业务系统的工作负荷较高，导致请求在业务系统的处理队列中产生等待延时。显然，在请求的传输阶段和处理阶段都可能存在一定的延时，这必然会造成客户

端的等待时间较长，使得业务处理过程的时效性较差。

发明内容

本申请实施例提供一种业务处理方法及装置，用以解决现有的业务处理过程的时效性较差的问题。

本申请实施例提供的一种业务处理方法，包括：

第二服务器接收第一服务器发送的第一业务结果，其中，所述第一业务结果是所述第一服务器针对客户端发送的第一业务请求进行处理后生成的；

确定所述第一业务结果对应的类别；

根据所述类别以及所述第一业务结果，生成与该类别相匹配的第二业务结果；

当所述第二服务器接收到所述客户端发送的第二业务请求后，将已生成的第二业务结果反馈给所述客户端。

本申请实施例还提供的一种业务处理方法，包括：

第一服务器接收客户端发送的第一业务请求；

根据所述第一业务请求，生成对应于该第一服务器自身类别的第一业务结果；

将所述第一业务结果发送给所述第二服务器，以使得所述第二服务器确定所述第一业务结果对应的类别，根据第一业务结果及类别，生成与该类别相匹配的第二业务结果，并当所述第二服务器接收到客户端根据所述第一业务结果生成的第二业务请求后，将已生成的所述第二业务结果发送给客户端。

本申请实施例提供的一种业务处理装置，包括：

接收模块，接收第一服务器发送的第一业务结果，其中，所述第一业务结果是所述第一服务器针对客户端发送的第一业务请求进行处理后生成的；

类别确定模块，确定所述第一业务结果对应的类别；

生成模块，根据所述类别以及所述第一业务结果，生成与该类别相匹配的

第二业务结果；

反馈模块，当所述第二服务器接收到所述客户端发送的第二业务请求后，将已生成的第二业务结果反馈给所述客户端。

本申请实施例另提供一种业务处理装置，包括：

接收模块，接收客户端发送的第一业务请求；

生成模块，根据所述第一业务请求，生成对应于该第一服务器自身类别的第一业务结果；

发送模块，将所述第一业务结果发送给所述客户端、以及第二服务器，以使得所述第二服务器确定所述第一业务结果对应的类别，根据所述第一业务结果及类别，生成第二业务结果，并当该第二服务器接收到客户端根据所述第一业务结果生成的第二业务请求后，将已生成的所述第二业务结果发送给客户端。

本申请实施例提供一种业务处理方法及装置，通过该方法，第一服务器接收到了客户端所发出的第一业务请求后，将对该第一业务请求进行处理，并生成相应的第一业务结果，此时，第一服务器除了将第一业务结果返回给客户端之外，还会按照业务流程，将该第一业务结果发送给第二服务器，第二服务器会确定第一业务结果对应的类别，这样一来，第二服务器便可以该第一业务结果进行处理，提前生成与该类别相匹配的第二业务结果。正是采用了这样的方式，使得第二服务器在接收到来自客户端所发送的第二业务请求之前，便可以生成第二业务结果，当第二服务器接收到了客户端实际发送的第二请求后，便可以将已经生成的第二业务结果直接返回给客户端，而且，所生成的第二业务结果符合第一业务结果的业务类型。显然，相较于现有技术中的方式而言，第二服务器在接收到客户端的第二业务请求之前，就可以根据第一业务结果预先生成第二业务结果，从而有效节省了客户端的等待时长，也提升了处理业务请求的时效性。

附图说明

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：

图 1a 为本申请实施例提供的业务处理过程所基于的架构示意图；

图 1b 为本申请实施例提供的基于第二服务器侧的业务处理过程；

图 2 为本申请实施例提供的基于第一服务器侧的业务处理过程；

图 3 为本申请实施例提供的应用实例中的业务处理过程的示意图；

图 4 为本申请实施例提供的基于第二服务器侧的业务处理装置结构示意图；

图 5 为本申请实施例提供的基于第一服务器侧的业务处理装置结构示意图。

具体实施方式

为使本申请的目的、技术方案和优点更加清楚，下面将结合本申请具体实施例及相应的附图对本申请技术方案进行清楚、完整地描述。显然，所描述的实施例仅是本申请一部分实施例，而不是全部的实施例。基于本申请中的实施例，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他实施例，都属于本申请保护的范围。

正如前述，用户通过客户端从在线系统中获取需要由多个业务系统共同完成的业务服务时，客户端端通常需要接收不同业务系统所返回的中间业务结果，并基于中间业务结果进行重定向后，向下一业务系统发出请求，直到完成完整的业务流程为止，但在该过程中，客户端在向业务系统发送请求时，容易受到网络环境的影响，产生传输延时，且业务系统只能够在接收到客户端所发出的业务请求后，才会进行处理，在业务系统工作负荷较高的情况下，业务系统处理该业务请求时也会产生一定的处理延时，这样一来，对于客户端而言，将受到传输延时和处理延时的双重影响，使得客户端的等待耗时较长，显然，这会对业务请求的时效性造成严重影响。

基于此，就需要一种能够减少客户端等待时间的业务处理方式，也即，在

本申请实施例中，供一种业务处理方法。

需要说明的是，在本申请实施例中，由于用户通过客户端，从在线系统内所获取的业务服务往往需要多个业务系统共同完成，故客户端会按照业务流程，针对不同的业务系统会发起不同的业务请求，但可以认为，客户端所发起的不同的业务请求，是为了完成同一业务。

本申请实施例中所述的业务处理方法，基于如图 1a 所示的架构，在图 1 中可见，该架构中包含用户所使用的业务客户端（以下简称为：客户端）、第一服务器以及第二服务器，其中，所述的客户端，可以是运行在终端中的浏览器、APP 应用等，第一服务器和第二服务器具有不同的业务功能，第一服务器和第二服务器具体可以是在业务流程中相邻的两个业务服务器，所述的第二服务器，可以是在业务流程中处理顺序位于第一服务器之后的业务服务器，例如：假设现有 3 个业务服务器 A、B、C，某种业务服务需要按照业务服务器 A 至 B 再至 C 的顺序完成，那么，针对业务服务器 A、B 而言，第一服务器就可以是业务服务器 A，第二服务器就可以是业务服务器 B；针对业务服务器 B、C 而言，第一服务器就可以是业务服务器 B，第二服务器就可以是业务服务器 C。换言之，本申请中的第一服务器和第二服务器并非只限于 2 个服务器的场景。

下面将结合如图 1a 所示的架构，对本申请中的业务处理方法进行说明，具体而言，如图 1b 所示的业务处理过程包括如下步骤：

步骤 S101：第二服务器接收第一服务器发送的第一业务结果。

其中，所述第一业务结果是所述第一服务器针对客户端发送的第一业务请求进行处理后生成的。

在实际应用场景下，当用户使用客户端获取业务服务时，通常需要向相应的业务服务器发出业务请求，例如：用户针对某商品下单后，此时，客户端就会向下单服务器发送下单请求，在该示例中，下单服务器就是第一服务器，下单请求就是第一业务请求。

其中，第一业务请求，可以由第一服务器所接收到的业务请求，该第一

业务请求既可以由客户端根据用户的操作指令所发出的业务请求，也可以是客户端接收到了业务流程中的前一业务服务器所反馈的中间业务结果后所发出的业务请求。当然，这里并不构成对本申请的限定。

第一业务结果，可认为是第一服务器对第一业务请求进行了相应处理后所生成的中间结果。

考虑到实际应用中，客户端向第二服务器发送第二业务请求的过程可能会受到网络环境的影响而产生延时，第二服务器在接收到第二业务请求并进行处理时，也可能产生处理延时，显然，这两种情况均会造成客户端的等待时间较长，因此，为了减少客户端的等待时长，在本申请实施例中，第一服务器除了将第一业务结果发送给客户端之外，还会直接向第二服务器发送该第一业务结果，那么，第二服务器便可根据第一业务结果提前进行处理。

S102：确定所述第一业务结果对应的类别。

本申请实施例中，第一业务结果所对应的类别，可认为是第一服务器所对应的类别，应理解，在实际应用中，对于不同业务提供方的第一服务器而言，其类别（如：所提供的业务类别、业务提供方自身的类别）通常存在差异，相应地，对第二业务结果的类别也会存在不同的要求。

例如：电信运营商（此时，第一服务器就是电信运营商服务器）和银行（此时，第一服务器就是银行服务器）分别是不同的业务提供方，显然，电信运营商服务器所生成的第一业务结果通常与电信业务相关，而银行服务器所生成的第一业务结果通常与金融业务相关，可以理解，以上两个业务提供方对后续生成的第二业务结果的类别应该也存在差异。

因此，在本申请实施例中，将确定出第一业务结果所对应的类别，以便于执行后续操作。

S103：根据所述类别以及所述第一业务结果，生成与该类别相匹配的第二业务结果。

当第二服务器接收到了第一业务结果后，便可以提前进行处理，生成与该

类别相匹配的第二业务结果。

S104: 当所述第二服务器接收到所述客户端发送的第二业务请求后, 将已生成的第二业务结果反馈给所述客户端。

第二服务器接收到了第二业务请求后, 便可以将提前生成的第二结果直接反馈给客户端, 从而减少或避免了客户端的等待耗时。

通过上述步骤, 第一服务器接收到了客户端所发出的第一业务请求后, 对该第一业务请求进行处理, 并生成相应的第一业务结果, 此时, 第一服务器除了将第一业务结果返回给客户端之外, 还会按照业务流程, 将该第一业务结果发送给第二服务器, 第二服务器会确定第一业务结果对应的类别, 这样一来, 第二服务器便可以该第一业务结果进行处理, 提前生成与该类别相匹配的第二业务结果。正是采用了这样的方式, 使得第二服务器在接收到来自客户端所发送的第二业务请求之前, 便可以生成第二业务结果, 当第二服务器接收到了客户端实际发送的第二请求后, 便可以将已经生成的第二业务结果直接返回给客户端, 而且, 所生成的第二业务结果符合第一业务结果的业务类型。显然, 相较于现有技术中的方式而言, 第二服务器在接收到客户端的第二业务请求之前, 就可以根据第一业务结果预先生成第二业务结果, 从而有效节省了客户端的等待时长, 也提升了处理业务请求的时效性。

为了清楚的阐述本申请的上述业务处理方法, 现以实际的支付业务场景进行详细说明:

假设某用户使用某商品网站的客户端, 在该商品网站中购买某商品。可以认为, 用户通过客户端针对某商品的购买操作涉及下单和支付两种业务, 而支付业务提供方(如: 支付平台)可在该过程中, 提供支付服务(即, 下单业务由该商品网站后台的下单服务器完成, 而支付业务由支付平台后台的支付服务器完成)。那么, 在该场景中, 第一业务请求为下单请求, 第一服务器包括下单服务器, 第一业务结果包括订单信息, 第二服务器包括支付服务器, 第二业务结果包括支付界面或异常页面。

在此场景下，上述步骤 S101 中，第二服务器接收第一服务器发送的第一业务结果，具体包括：支付服务器接收下单服务器发送的订单数据。其中，所述的订单数据中可包含诸如订单流水号、第一服务器的类别标识、支付状态数据、账户信息、商品信息、金额信息等。这里并不作具体限定。

上述步骤 S102 中，确定所述第一业务结果对应的类别，具体包括：确定所述订单数据中包含的下单服务器的类别标识。

这里，所述的类别标识，实际上反映了第一服务器的类别标识，在本申请实施例中，所述的类别标识可以包括：第一服务器所属的业务提供方的名称、行业类别信息、第一服务器自身的名称、设备号等等，这里并不构成对本申请的限定。

根据所述类别以及所述第一业务结果，生成与该类别相匹配的第二业务结果，具体包括：根据预先建立的类别标识与页面模板的对应关系，确定所述订单数据所对应的页面模板，根据确定出的所述页面模板，在所述订单数据中标记匹配于该页面模板的订单数据，根据标记的订单数据渲染并构建支付页面。

需要说明的是，在实际的支付场景下，针对不同的第一服务器，即下单服务器（如：商品网站服务器、电信运营商服务器等），所生成的支付页面通常也是不同的，故支付服务器中通常会存储有不同的支付页面模板，以对应于不同的下单服务器，而下单服务器发送的订单数据中，也会包含有下单服务器的类别标识，从而可确定相应的支付页面模板。

针对不同的支付页面模板，其所需渲染的订单数据也可能不同，例如：针对某商品网站的支付页面来说，该支付页面上需要展示用户的订单号、支付金额、邮寄地址等数据，而针对手机卡充值的支付页面来说，该支付页面上需要展示单号、待充值手机号、金额等数据。显然，对于这两类支付页面，所需渲染的订单数据并不相同。

此外，在实际支付场景下，支付页面可能存在两种情况：一种情况是成功渲染支付页面，另一种情况是支付过程出现问题而显示异常页面。所以，支付

服务器在进行渲染页面的操作之前，还会判断是否能够成功的支付，也即，根据标记的订单数据渲染并构建支付页面，具体包括：根据所述订单数据判断支付状态，当判断出所述支付状态为可进行支付时，则根据标记的订单数据渲染并构建支付页面。

这里需要说明的是，作为本申请实施例中的一种方式，支付服务器根据订单数据判断支付状态的过程，可以基于用户预先注册的账户实现，该方式下分为两种情况：

一、如果下单服务器和支付服务器均属于同一业务提供方，那么，用户在该业务提供方处所注册的账户通用于下单服务器和支付服务器，换言之，支付服务器有权限查询用户的账户余额，从而判断本次支付是否能够成功。

二、下单服务器和支付服务器分别属于不同的业务提供方，那么，可以理解，用户在两个业务提供方处所注册的账户应是相互关联的，这样，支付服务器仍然有权限查询用户在下单服务器中注册的账户的余额，并基于此判断支付是否能成功。

当然，上述的两种情况并不构成对本申请的限定。

而如果当确定所述通知消息所表征的支付状态为不可进行支付时，则所述方法还包括：删除对所述订单数据的标记，渲染并构建异常页面。

作为本申请实施例中的一种可能方式，在生成了相应的页面（包括前述的支付页面或异常页面）后，还可针对成功进行渲染的数据进行标记，用以后续对系统进行优化，这里并不进行详细赘述。

当然，本申请实施例中的所述订单数据中，还携带有客户端标识，基于此，所述方法还包括：建立所述客户端标识与所述支付页面或异常页面之间的对应关系。这样一来，将已生成的第二业务结果反馈给所述客户端，具体包括：确定所述第二业务请求中的客户端标识，将已生成的且与该客户端标识对应的支付页面或异常页面反馈给所述客户端。

通过上述内容，当支付服务器接收到客户端发送的支付请求后，便可以将

提前渲染并构建的支付页面或异常页面反馈给所述客户端。

可见，采用上述的方式，可以节省客户端完成支付流程所消耗的时间。

以上是针对第二服务器侧的描述，在本申请实施例中，基于第一服务器侧，还提供一种业务处理方法，如图 2 所示，具体包括以下步骤：

步骤 S201：第一服务器接收客户端发送的第一业务请求。

步骤 S202，根据所述第一业务请求，生成对应于该第一服务器自身类别的第一业务结果。

当第一服务器接收到了第一业务请求后，便会针对该第一业务请求进行处理，从而生成相应的业务结果，即第一业务结果，这里的第一业务结果通常是一种中间业务结果。

当然，在本申请实施例中的一种方式下，第一业务请求中往往会携带客户端的标识信息，那么，第一服务器所生成的第一业务结果中也会包含客户端的标识信息。

步骤 S203，将所述第一业务结果发送给所述第二服务器，以使得所述第二服务器确定所述第一业务结果对应的类别，根据第一业务结果及类别，生成与该类别相匹配的第二业务结果，并当该第二服务器接收到客户端根据所述第一业务结果生成的第二业务请求后，将已生成的所述第二业务结果发送给客户端。

实际应用中，如图 3 所示，示出了上述场景下的业务流程：

步骤 S301：用户向客户端发出下单操作指令。

步骤 S302：客户端根据下单操作指令，向下单服务器发送下单请求。

步骤 S303：下单服务器针对接收到的下单请求进行处理，生成订单信息。

步骤 S304：下单服务器将订单信息发送给支付服务器。以使得支付服务器执行步骤 S306。

步骤 S305：下单服务器将订单信息发送给客户端。以使得客户端执行步骤

S307。

步骤 S306: 支付服务器根据订单信息, 确定相应的支付页面模板, 并进一步生成支付页面。

步骤 S307: 客户端根据订单信息, 重定向至支付服务器, 并向支付服务器发出支付页面请求。

步骤 S308: 支付服务器向客户端返回支付页面。

从上述内容及示例中可见, 在实际支付场景中, 一旦订单服务器生成了订单信息后, 将立即发送给支付服务器进行处理, 以生成相应的支付页面, 那么, 当客户端向支付服务器发出支付页面请求后, 支付服务器可以立即向客户端返回已经生成的支付页面。显然, 这样的方式将有效减少客户端等待生成支付页面的时间, 进一步也使得用户可更快捷地浏览到支付页面, 以便在支付页面中完成支付业务。

以上为本申请提供的业务处理方法的几种实施例, 基于同样的思路, 本申请还提供了业务处理装置的实施例, 如图 4 所示。图 4 中的业务处理装置, 所述装置可以设置于第二服务器中, 该装置包括:

接收模块 401, 接收第一服务器发送的第一业务结果, 其中, 所述第一业务结果是所述第一服务器针对客户端发送的第一业务请求进行处理后生成的;

类别确定模块 402, 确定所述第一业务结果对应的类别;

生成模块 403, 根据所述类别以及所述第一业务结果, 生成与该类别相匹配的第二业务结果;

反馈模块 404, 当所述第二服务器接收到所述客户端发送的第二业务请求后, 将已生成的第二业务结果反馈给所述客户端。

在本申请实施例中的一种场景下, 所述第一业务请求为下单请求; 所述第一服务器包括下单服务器; 所述第一业务结果包括订单数据; 所述第二服务器包括支付服务器; 所述第二业务结果包括支付界面或异常页面。

基于此,所述订单数据中包含下单服务器的类别标识;所述类别确定模块 402,确定所述订单数据中包含的下单服务器的类别标识;所述生成模块 403,根据预先建立的类别标识与页面模板的对应关系,确定所述订单数据所对应的页面模板,根据确定出的所述界面模板,在所述订单数据中标记匹配于该页面模板的订单数据,根据标记的订单数据渲染并构建支付页面。

所述生成模块 403,根据所述订单数据判断支付状态,当判断出所述支付状态为可进行支付时,则根据标记的订单数据渲染并构建支付页面。。

当所述生成模块 403 判断出所述支付状态为不可进行支付时,删除对所述订单数据的标记,渲染并构建异常页面。。

所述订单数据中携带有客户端标识,所述装置还包括:关系建立模块 405,建立所述客户端标识与所述支付页面或异常页面之间的对应关系。所述反馈模块 404,确定所述第二业务请求中的客户端标识,将已生成的且与该客户端标识对应的支付页面或异常页面反馈给所述客户端。

在第一服务器侧,本申请还提供一种业务处理装置,如图 5 所示,该装置可以设置在第一服务器中,包括:

接收模块 501,接收客户端发送的第一业务请求;

生成模块 502,根据所述第一业务请求,生成对应于该第一服务器自身类别的第一业务结果;

发送模块 503,将所述第一业务结果发送给所述客户端、以及第二服务器,以使得所述第二服务器确定所述第一业务结果对应的类别,根据所述第一业务结果及类别,生成第二业务结果,并当该第二服务器接收到客户端根据所述第一业务结果生成的第二业务请求后,将已生成的所述第二业务结果发送给客户端。

在 20 世纪 90 年代,对于一个技术的改进可以很明显地区分是硬件上的改

进(例如,对二极管、晶体管、开关等电路结构的改进)还是软件上的改进(对于方法流程的改进)。然而,随着技术的发展,当今的很多方法流程的改进已经可以视为硬件电路结构的直接改进。设计人员几乎都通过将改进的方法流程编程到硬件电路中来得到相应的硬件电路结构。因此,不能说一个方法流程的改进就不能用硬件实体模块来实现。例如,可编程逻辑器件(Programmable Logic Device, PLD)(例如现场可编程门阵列(Field Programmable Gate Array, FPGA))就是这样一种集成电路,其逻辑功能由用户对器件编程来确定。由设计人员自行编程来把一个数字系统“集成”在一片 PLD 上,而不需要请芯片制造厂商来设计和制作专用的集成电路芯片。而且,如今,取代手工地制作集成电路芯片,这种编程也多半改用“逻辑编译器(logic compiler)”软件来实现,它与程序开发撰写时所用的软件编译器相类似,而要编译之前的原始代码也得用特定的编程语言来撰写,此称之为硬件描述语言(Hardware Description Language, HDL),而 HDL 也并非仅有一种,而是有许多种,如 ABEL(Advanced Boolean Expression Language)、AHDL(Altera Hardware Description Language)、Confluence、CUPL(Cornell University Programming Language)、HDCal、JHDL(Java Hardware Description Language)、Lava、Lola、MyHDL、PALASM、RHDL(Ruby Hardware Description Language)等,目前最普遍使用的是 VHDL(Very-High-Speed Integrated Circuit Hardware Description Language)与 Verilog。本领域技术人员也应该清楚,只需要将方法流程用上述几种硬件描述语言稍作逻辑编程并编程到集成电路中,就可以很容易得到实现该逻辑方法流程的硬件电路。

控制器可以按任何适当的方式实现,例如,控制器可以采取例如微处理器或处理器以及存储可由该(微)处理器执行的计算机可读程序代码(例如软件或固件)的计算机可读介质、逻辑门、开关、专用集成电路(Application Specific Integrated Circuit, ASIC)、可编程逻辑控制器和嵌入微控制器的形式,控制器的例子包括但不限于以下微控制器:ARC 625D、Atmel AT91SAM、Microchip

PIC18F26K20 以及 Silicone Labs C8051F320, 存储器控制器还可以被实现为存储器的控制逻辑的一部分。本领域技术人员也知道, 除了以纯计算机可读程序代码方式实现控制器以外, 完全可以通过将方法步骤进行逻辑编程来使得控制器以逻辑门、开关、专用集成电路、可编程逻辑控制器和嵌入微控制器等的形式来实现相同功能。因此这种控制器可以被认为是一种硬件部件, 而对其内包括的用于实现各种功能的装置也可以视为硬件部件内的结构。或者甚至, 可以将用于实现各种功能的装置视为既可以是实现方法的软件模块又可以是硬件部件内的结构。

上述实施例阐明的系统、装置、模块或单元, 具体可以由计算机芯片或实体实现, 或者由具有某种功能的产品来实现。一种典型的实现设备为计算机。具体的, 计算机例如可以为个人计算机、膝上型计算机、蜂窝电话、相机电话、智能电话、个人数字助理、媒体播放器、导航设备、电子邮件设备、游戏控制台、平板计算机、可穿戴设备或者这些设备中的任何设备的组合。

为了描述的方便, 描述以上装置时以功能分为各种单元分别描述。当然, 在实施本申请时可以把各单元的功能在同一个或多个软件和/或硬件中实现。

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、系统、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。

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。

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。

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。

在一个典型的配置中,计算设备包括一个或多个处理器(CPU)、输入/输出接口、网络接口和内存。

内存可能包括计算机可读介质中的非永久性存储器,随机存取存储器(RAM)和/或非易失性内存等形式,如只读存储器(ROM)或闪存(flash RAM)。内存是计算机可读介质的示例。

计算机可读介质包括永久性和非永久性、可移动和非可移动媒体可以由任何方法或技术来实现信息存储。信息可以是计算机可读指令、数据结构、程序的模块或其他数据。计算机的存储介质的例子包括,但不限于相变内存(PRAM)、静态随机存取存储器(SRAM)、动态随机存取存储器(DRAM)、其他类型的随机存取存储器(RAM)、只读存储器(ROM)、电可擦除可编程只读存储器(EEPROM)、快闪记忆体或其他内存技术、只读光盘只读存储器(CD-ROM)、数字多功能光盘(DVD)或其他光学存储、磁盒式磁带,磁带磁磁盘存储或其他磁性存储设备或任何其他非传输介质,可用于存储可以被计算设备访问的信息。按照本文中的界定,计算机可读介质不包括暂存电脑可读媒体(transitory media),如调制的数据信号和载波。

还需要说明的是,术语“包括”、“包含”或者其任何其他变体意在涵盖非排他性的包含,从而使得包括一系列要素的过程、方法、商品或者设备不仅包括

那些要素，而且还包括没有明确列出的其他要素，或者是还包括为这种过程、方法、商品或者设备所固有的要素。在没有更多限制的情况下，由语句“包括一个……”限定的要素，并不排除在包括所述要素的过程、方法、商品或者设备中还存在另外的相同要素。

本领域技术人员应明白，本申请的实施例可提供为方法、系统或计算机程序产品。因此，本申请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，本申请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。

本申请可以在由计算机执行的计算机可执行指令的一般上下文中描述，例如程序模块。一般地，程序模块包括执行特定事务或实现特定抽象数据类型的例程、程序、对象、组件、数据结构等等。也可以在分布式计算环境中实践本申请，在这些分布式计算环境中，由通过通信网络而被连接的远程处理设备来执行事务。在分布式计算环境中，程序模块可以位于包括存储设备在内的本地和远程计算机存储介质中。

本说明书中的各个实施例均采用递进的方式描述，各个实施例之间相同相似的部分互相参见即可，每个实施例重点说明的都是与其他实施例的不同之处。尤其，对于系统实施例而言，由于其基本相似于方法实施例，所以描述的比较简单，相关之处参见方法实施例的部分说明即可。

以上所述仅为本申请的实施例而已，并不用于限制本申请。对于本领域技术人员来说，本申请可以有各种更改和变化。凡在本申请的精神和原理之内所作的任何修改、等同替换、改进等，均应包含在本申请的权利要求范围之内。