(19)国家知识产权局

(12)发明专利

(10)授权公告号 CN 109716322 B
(45)授权公告日 2023.03.21

(21)申请号 201780057260.4
(22)申请日 2017.09.15
(65)同一申请的已公布的文献号
申请公布号 CN 109716322 A
(43)申请公布日 2019.05.03
(30)优先权数据
201641031479 2016.09.15 IN
(85)PCT国际申请进入国家阶段日
2019.03.12
(86)PCT国际申请的申请数据
PCT/US2017/051887 2017.09.15
(87)PCT国际申请的公布数据
WO2018/053338 EN 2018.03.22
(73)专利权人 甲骨文国际公司
地址 美国加利福尼亚
(72)发明人 H·帕克 S·比施诺伊
P·图卡拉姆 S·库马
P·阿德瓦尼 K·穆拉伊
J·图里昂
(74)专利代理机构 中国贸促会专利商标事务所有限公司 11038
专利代理师 刘玉洁
(51)Int.Cl.
G06F 16/2453(2019.01)
G06F 16/2455(2019.01)
(56)对比文件
CN 104756111 A,2015.07.01
CN 105308592 A,2016.02.03
CN 104756112 A,2015.07.01
CN 104471572 A,2015.03.25
US 2014095444 A1,2014.04.03
Loic Salmon.et.“Design principles of a stream-based framework for mobility analysis”.《Springer》.2016,
Sanket Chintapalli.et.“Benchmarking Streaming Computation Engines: Storm, Flink and Spark Streaming”.《2016 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)》.2016,
审查员 齐丽静

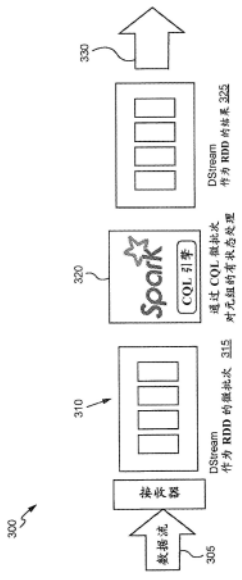
权利要求书2页 说明书28页 附图12页

(54)发明名称

用于微批次流传输的复杂事件处理

(57)摘要

公开了一种用于处理事件流中的事件的事件处理系统。该系统可以执行指令以:接收连续查询,对连续查询应用变换以生成用于连续查询的查询计划,使用变换算法来变换查询计划以生成经变换的查询计划,接收与应用相关的输入事件的微批次流,至少部分地基于经变换的查询计划来处理微批次流的输入事件以生成与应用相关的输出事件集合,以及将与应用相关的输出事件集合存储在输出队列中。



CN 109716322 B

1.一种用于处理微批处理流以支持完全有状态查询处理的方法,包括:

由计算设备接收连续查询;

由计算设备对连续查询应用变换,以生成用于连续查询的查询计划;

由计算设备使用变换算法来变换查询计划,以生成经变换的查询计划;

由计算设备接收与应用相关的输入事件的微批次流;

由计算设备至少部分地基于经变换的查询计划来处理微批次流的输入事件,以生成与应用相关的输出事件集合;以及

由计算设备将与应用相关的输出事件集合存储在输出队列中,

其中所述变换算法是连续查询语言CQL变换,所述处理是使用连续查询处理引擎来执行的,其中所述连续查询处理引擎被配置为维护所述微批次流的所述连续查询的状态,并且所述处理包括增量地处理所述输入事件中的每个输入事件以生成输出事件。

2.如权利要求1所述的方法,其中变换是有向无环图DAG变换。

3.如权利要求2所述的方法,还包括:当已经处理完所有输入事件时,由计算设备发送输出队列中的输出事件。

4.如权利要求3所述的方法,其中所述微批次流包括弹性分布式数据集RDD或数据的微批次,并且DAG变换是顶点和边的集合,其中顶点表示RDD并且边表示要应用于RDD的操作。

5.如权利要求4所述的方法,其中处理所述输入事件中的每个输入事件包括至少部分地基于经变换的查询计划对所述输入事件中的每个输入事件执行计算。

6.如权利要求5所述的方法,其中所述连续查询包括模式匹配,并且CQL变换被添加到DAG变换,以便支持包括模式匹配的完全有状态查询处理。

7.一种系统,包括:

存储器,被配置为存储计算机可执行指令;以及

处理器,被配置为访问所述存储器并且执行所述计算机可执行指令,以:

接收连续查询;

对连续查询应用变换,以生成用于连续查询的查询计划;

使用变换算法来变换查询计划,以生成经变换的查询计划;

接收与应用相关的输入事件的微批次流;

至少部分地基于经变换的查询计划来处理微批次流的输入事件,以生成与应用相关的输出事件集合;以及

将与应用相关的输出事件集合存储在输出队列中,

其中所述变换算法是连续查询语言CQL变换,所述处理是使用连续查询处理引擎来执行的,其中所述连续查询处理引擎被配置为维护所述微批次流的所述连续查询的状态,并且所述处理包括增量地处理所述输入事件中的每个输入事件以生成输出事件。

8.如权利要求7所述的系统,其中变换是有向无环图DAG变换。

9.如权利要求8所述的系统,其中所述计算机可执行指令进一步可执行,以便在已经处理完所有输入事件时发送输出队列中的输出事件。

10.如权利要求9所述的系统,其中所述微批次流包括弹性分布式数据集RDD或数据的微批次,并且DAG变换是顶点和边的集合,其中顶点表示RDD并且边表示要应用于RDD的操作。

11.如权利要求10所述的系统,其中处理所述输入事件中的每个输入事件包括至少部分地基于经变换的查询计划对所述输入事件中的每个输入事件执行计算。

12.如权利要求11所述的系统,其中所述连续查询包括模式匹配,并且CQL变换被添加到DAG变换,以便支持包括模式匹配的完全有状态查询处理。

13.一种存储计算机可执行代码的计算机可读介质,所述计算机可执行代码在由处理器执行时使所述处理器执行包括以下各项的操作:

接收连续查询;

将有向无环图DAG变换应用于连续查询,以生成用于连续查询的DAG查询计划;

使用变换算法来变换DAG查询计划,以生成经变换的查询计划;

接收与应用相关的输入事件的微批次流;

至少部分地基于经变换的查询计划来处理微批次流的输入事件,以生成与应用相关的输出事件集合;以及

将与应用相关的输出事件集合存储在输出队列中,

其中所述变换算法是连续查询语言CQL变换,所述处理是使用连续查询处理引擎来执行的,其中所述连续查询处理引擎被配置为维护所述微批次流的所述连续查询的状态,并且所述处理包括增量地处理所述输入事件中的每个输入事件以生成输出事件。

14.如权利要求13所述的计算机可读介质,其中所述操作还包括当已经处理完所有输入事件时发送输出队列中的输出事件。

15.如权利要求14所述的计算机可读介质,其中所述微批次流包括弹性分布式数据集RDD或数据的微批次,并且DAG变换是顶点和边的集合,其中顶点表示RDD并且边表示要应用于RDD的操作。

16.如权利要求15所述的计算机可读介质,其中处理所述输入事件中的每个输入事件包括至少部分地基于经变换的查询计划对所述输入事件中的每个输入事件执行计算。

17.如权利要求16所述的计算机可读介质,其中所述连续查询包括模式匹配,并且CQL变换被添加到DAG变换,以便支持包括模式匹配的完全有状态查询处理。

18.一种包括用于执行如权利要求1-6中任一项所述的方法的操作的部件的装置。

用于微批次流传输的复杂事件处理

[0001] 对相关申请的交叉引用

[0002] 本申请要求于2016年9月15日提交的标题为“MANAGING SNAPSHOTS AND STATE WITH MICRO-BATCHING”的印度临时申请No.201641031479的优先权和权益，该印度临时申请的全部内容通过引用并入本文，用于所有目的。

背景技术

[0003] 在传统的数据库系统中，数据通常以表的形式存储在一个或多个数据库中。然后使用诸如结构化查询语言(SQL)之类的数据管理语言来查询和操纵所存储的数据。例如，可以定义并执行SQL查询，以从存储在数据库中的数据中识别相关数据。因此，SQL查询是对存储在数据库中的有限数据集执行的。另外，当执行SQL查询时，它在有限数据集上执行一次并且产生有限静态结果。因此，数据库最佳地被配备为在有限存储数据集上运行查询。

[0004] 但是，多个现代应用和系统以连续数据或事件流而不是有限数据集的形式生成数据。此类应用的示例包括但不限于传感器数据应用、金融报价机、网络性能测量工具(例如，网络监视和流量管理应用)、点击流分析工具、汽车交通监视等。这些应用引起了对可以处理数据流的新型应用的需求。例如，温度传感器可以被配置为发出温度读数。

[0005] 管理和处理用于这些类型的基于事件流的应用的数据涉及构建具有强时间焦点的数据管理和查询能力。需要不同类型的查询机制，该查询机制包括对连续无界数据集的长时间运行的查询。虽然一些供应商现在提供面向事件流处理的产品套件，但是这些产品提供物(offering)仍然缺乏处理当今事件处理需求所需的处理灵活性。

发明内容

[0006] 提供了用于处理事件流的事件的技术(例如，方法、系统、存储可由一个或多个处理器执行的代码或指令的非瞬态计算机可读介质)。在实施例中，公开了一种事件处理系统。一个或多个计算机的系统可以被配置为凭借在系统上安装在操作时使系统执行动作的软件、固件、硬件或它们的组合来执行特定的操作或动作。一个或多个计算机程序可以被配置为凭借包括当由数据处理装置执行时使装置执行动作的指令来执行特定的操作或动作。一个一般方面包括一种用于处理微批处理流以支持完全有状态查询处理的方法，该方法包括：由计算设备接收连续查询。该方法还包括由计算设备向连续查询应用变换，以生成用于连续查询的查询计划。该方法还包括由计算设备使用变换算法来变换查询计划，以生成经变换的查询计划。该方法还包括由计算设备接收与应用相关的输入事件的微批次流。该方法还包括由计算设备至少部分地基于经变换的查询计划来处理微批次流的输入事件，以生成与应用相关的输出事件集合。该方法还包括由计算设备在输出队列中存储与应用相关的输出事件集合。该方面的其它实施例包括各自被配置为执行方法的动作的对应计算机系统、装置和记录在一个或多个计算机存储设备上的计算机程序。

[0007] 实施方式可以包括以下特征中的一个或多个。该方法其中变换算法是连续查询语言(CQL)变换，使用连续查询处理引擎执行处理，并且处理包括增量地处理输入事件中的每

个输入事件以生成输出事件。该方法其中变换是有向无环图(dag)变换。该方法还包括:当已经处理完所有输入事件时,由计算设备发送输出队列中的输出事件。该方法其中微批次流包括数据或弹性分布式数据集(RDD)的微批次,并且dag变换是顶点和边的集合,其中顶点表示RDD并且边表示要应用于RDD的操作。该方法其中处理输入事件中的每个输入事件包括至少部分地基于经变换的查询计划对输入中的每个输入执行计算。该方法其中连续查询包括模式匹配,并且将CQL变换添加到dag变换,以便支持包括模式匹配的完全有状态查询处理。所描述的技术的实现可以包括硬件、方法或过程或计算机可访问介质上的计算机软件。

[0008] 一个一般方面包括一种系统,包括:存储器,被配置为存储计算机可执行指令;以及处理器,被配置为访问存储器并执行计算机可执行指令。该系统还包括接收连续查询。该系统还包括对连续查询应用变换,以生成用于连续查询的查询计划。该系统还包括使用变换算法来变换查询计划,以生成经变换的查询计划。该系统还包括接收与应用相关的输入事件的微批次流。该系统还包括至少部分地基于经变换的查询计划来处理微批次流的输入事件,以生成与应用相关的输出事件集合。该系统还包括将与应用相关的输出事件集合存储在输出队列中。该方面的其它实施例包括各自被配置为执行方法的动作的对应计算机系统、装置和记录在一个或多个计算机存储设备上的计算机程序。

[0009] 实施方式可以包括以下特征中的一个或多个。该系统其中变换算法是连续查询语言(CQL)变换,使用连续查询处理引擎执行处理,并且处理包括增量地处理输入事件中的每个输入事件以生成输出事件。该系统其中变换是有向无环图(dag)变换。该系统其中计算机可执行指令进一步可执行以在已经处理完所有输入事件时发送输出队列中的输出事件。该系统其中微批次流包括数据或弹性分布式数据集(RDD)的微批次,并且dag变换是顶点和边的集合,其中顶点表示RDD并且边表示要应用于RDD的操作。该系统其中处理输入事件中的每个输入事件包括至少部分地基于经变换的查询计划对输入中的每个输入执行计算。该系统其中连续查询包括模式匹配,并且将CQL变换添加到dag变换,以便支持包括模式匹配的完全有状态查询处理。所描述的技术的实现可以包括硬件、方法或过程或计算机可访问介质上的计算机软件。

[0010] 一个一般方面包括存储计算机可执行代码的计算机可读介质,该计算机可执行代码在由处理器执行时使处理器执行包括以下各项的操作:接收连续查询。计算机可读介质还包括将有向无环图(dag)变换应用于连续查询,以生成用于连续查询的dag查询计划。计算机可读介质还包括使用变换算法来变换dag查询计划,以生成经变换的查询计划。计算机可读介质还包括接收与应用相关的输入事件的微批次流。计算机可读介质还包括至少部分地基于经变换的查询计划来处理微批次流的输入事件,以生成与应用相关的输出事件集合。计算机可读介质还包括将与应用相关的输出事件集合存储在输出队列中。该方面的其它实施例包括各自被配置为执行方法的动作的对应计算机系统、装置和记录在一个或多个计算机存储设备上的计算机程序。

[0011] 实施方式可以包括以下特征中的一个或多个。计算机可读介质其中变换算法是连续查询语言(CQL)变换,使用连续查询处理引擎执行处理,并且处理包括增量地处理输入事件中的每个输入事件以生成输出事件。计算机可读介质其中操作还包括当已经处理完所有输入事件时发送输出队列中的输出事件。计算机可读介质其中微批次流包括数据或弹性分

布式数据集(RDD)的微批次，并且dag变换是顶点和边的集合，其中顶点表示RDD，并且边表示要应用于RDD的操作。计算机可读介质其中处理输入事件中的每个输入事件包括至少部分地基于经变换的查询计划对输入中的每个输入执行计算。计算机可读介质其中连续查询包括模式匹配，并且将CQL变换添加到dag变换，以便支持包括模式匹配的完全有状态查询处理。所描述的技术的实现可以包括硬件、方法或过程或计算机可访问介质上的计算机软件。

[0012] 上文和下文描述的技术可以以多种方式并且在多种上下文中实现。参考以下附图提供若干示例实现和上下文，如下文更详细地描述的。但是，以下实现和上下文只是其中的一小部分。

附图说明

[0013] 图1是根据本公开实施例的事件处理网络的图形表示。

[0014] 图2描绘了根据本公开实施例的事件处理系统的简化高级图。

[0015] 图3是根据本公开实施例的示例系统或体系架构，其中可以实现被配置为用于利用微批处理流处理进行有状态处理的流处理应用。

[0016] 图4示出了图示根据本公开实施例的微批处理流的处理的流程图。

[0017] 图5是根据本公开实施例的其中实现CQL引擎跟踪器的示例系统或体系架构。

[0018] 图6A是根据本公开实施例实现的Map目录结构的示例性数据结构。

[0019] 图6B是根据本公开实施例实现的Map目录结构的示例性数据结构。

[0020] 图7示出了图示根据本公开实施例的微批处理流的处理的流程图。

[0021] 图8示出了图示根据本公开实施例的微批处理的处理的流程图。

[0022] 图9描绘了用于实现本公开实施例的分布式系统的简化图。

[0023] 图10是根据本公开实施例的系统环境的一个或多个部件的简化框图，由实施例系统的一个或多个部件提供的服务可以通过该系统环境作为云服务提供。

[0024] 图11示出了可以用于实现本公开实施例的示例计算机系统。

具体实施方式

[0025] 在以下描述中，将描述各种实施例。出于解释的目的，阐述具体配置和细节以便提供对实施例的透彻理解。但是，对于本领域技术人员来说明显的是，可以在没有具体细节的情况下实践这些实施例。此外，可以省略或简化众所周知的特征，以免模糊所描述的实施例。

[0026] 复杂事件处理(CEP)的概述

[0027] 复杂事件处理(CEP)提供用于基于事件驱动的体系架构来构建应用的模块化平台。CEP平台的核心是连续查询语言(CQL)，CQL允许应用使用声明性的类似SQL的语言对数据流进行过滤、查询以及执行模式匹配操作。开发人员可以将CQL与轻量级Java编程模型结合使用来编写应用。其它平台模块包括特征丰富的IDE、管理控制台、聚类(clustering)、分布式高速缓存、事件储存库和监视等。

[0028] 随着事件驱动的体系架构和复杂事件处理已成为企业计算领域的突出特征，越来越多的企业已开始使用CEP技术来构建任务关键应用。如今，任务关键CEP应用可以在许多

不同的行业中找到。例如，CEP技术正在电力行业中使用，以通过允许公用设施立即对电力需求的变化作出反应而使公用设施更高效。CEP技术正在信用卡行业中使用，以便在潜在欺诈性交易发生时实时地检测它们。任务关键CEP应用的列表继续增长。使用CEP技术构建任务关键应用导致需要使CEP应用具有高可用性和容错性。

[0029] 当今的信息技术(IT)环境为每样事物生成连续的数据流，从监视金融市场和网络性能，到业务处理执行和跟踪RFID标记的资产。CEP为开发事件处理应用提供丰富的声明性环境，以提高业务操作的有效性。CEP可以处理多个事件流，以实时检测模式和趋势，并为企业提供必要的可见性，以利用新出现的机会或减轻发展风险。

[0030] 连续数据流(也称为事件流)可以包括数据流或事件流，其本质上可以是连续的或无界的，并且没有明确的结束。在逻辑上，事件流或数据流可以是数据元素(也称为事件)的序列，每个数据元素具有相关联的时间戳。连续事件流可以在逻辑上表示为元素(s,T)的包(bag)或集合，其中“s”表示数据部分，并且“T”在时域中。“s”部分一般被称为元组或事件。因此，事件流可以是带时间戳的元组或事件的序列。

[0031] 在一些方面，与流中的事件相关联的时间戳可以等于时钟时间。但是，在其它示例中，与事件流中的事件相关联的时间可以由应用域定义，并且可以不与时钟时间对应，而是可以例如代替地由序列号表示。因而，与事件流中的事件相关联的时间信息可以由数字、时间戳或表示时间概念的任何其它信息来表示。对于接收输入事件流的系统，事件以增大的时间戳的次序到达系统。可以存在具有相同时间戳的多于一个事件。

[0032] 在一些示例中，事件流中的事件可以表示某个世间事件的发生(例如，当温度传感器的值改变为新值时，当股票代码的价格改变时)，并且与事件相关联的时间信息可以指示由数据流事件表示的世间事件何时发生。

[0033] 对于经由事件流接收的事件，可以使用与事件相关联的时间信息来确保事件流中的事件以增大的时间戳值的次序到达。这可以使得在事件流中接收的事件能够基于其相关联的时间信息而被排序。为了实现这种排序，时间戳可以以非递减方式与事件流中的事件相关联，使得较晚生成的事件具有与较早生成的事件相比较晚的时间戳。作为另一个示例，如果序列号被用作时间信息，那么与较晚生成的事件相关联的序列号可以大于与较早生成的事件相关联的序列号。在一些示例中，多个事件可以与相同的时间戳或序列号相关联，例如，当由数据流事件表示的世间事件同时发生时。属于同一事件流的事件一般可以按照由相关联的时间信息强加于事件的次序进行处理，其中较早的事件在较晚的事件之前被处理。

[0034] 与事件流中的事件相关联的时间信息(例如，时间戳)可以由流的源设置，或者可替代地可以由接收流的系统设置。例如，在某些实施例中，可以在接收事件流的系统上维持心跳，并且与事件相关联的时间可以基于由心跳测量的事件到达系统的时间。事件流中的两个事件有可能具有相同的时间信息。要注意的是，虽然时间戳排序要求特定于一个事件流，但是不同流的事件可以任意交织。

[0035] 事件流具有相关联的模式“S”，该模式包括时间信息以及一个或多个命名属性的集合。属于特定事件流的所有事件都符合与该特定事件流相关联的模式。因而，对于事件流(s,T)，事件流可以具有模式“S”，如(<time_stamp(时间戳)>,<attribute(s)(属性)>)，其中<attributes(属性)>表示模式的数据部分并且可以包括一个或多个属性。例如，用于股

票报价机事件流的模式可以包括属性<股票代码>和<股票价格>。经由这种流接收的每个事件将具有时间戳和两个属性。例如，股票报价机事件流可能会收到以下事件和相关联的时间戳：

[0036] …

[0037] (<timestamp_N>,<NVDA,4>)

[0038] (<timestamp_N+1>,<ORCL,62>)

[0039] (<timestamp_N+2>,<PCAR,38>)

[0040] (<timestamp_N+3>,<SPOT,53>)

[0041] (<timestamp_N+4>,<PDCO,44>)

[0042] (<timestamp_N+5>,<PTEN,50>)

[0043] …

[0044] 在上述流中，对于流元素(<timestamp_N+1>,<ORCL,62>)，事件是具有属性“stock_symbol(股票代码)”和“stock_value(股票价格)”的<ORCL,62>。与流元素相关联的时间戳是“timestamp_N+1”。因此，连续事件流是事件流，每个事件具有相同的属性系列。

[0045] 如上所述，流可以是CQL查询可以对其作用的主要数据源。流S可以是元素(s,T)的包(也称为“多集”)，其中“s”在S的模式中，并且“T”在时域中。此外，流元素可以是元组-时间戳对，元组-时间戳对可以被表示为带时间戳的元组插入的序列。换句话说，流可以是带时间戳的元组的序列。在一些情况下，可以存在具有相同时间戳的多于一个元组。并且，可以请求输入流的元组以增大的时间戳的次序到达系统。可替代地，关系(也称为“时变关系”，并且不与“关系数据”混淆，“关系数据”可以包括来自关系数据库的数据)可以是从时域到模式R的元组的无界包的映射。在一些示例中，关系可以是无序的、时变元组包(即，瞬时关系)。在一些情况下，在每个时间实例，关系可以是有界集。它还可以被表示为带时间戳的元组的序列，其可以包括插入、删除和/或更新以捕获关系的改变的状态。与流类似，关系可以具有关系的每个元组可以符合的固定模式。另外，如本文所使用的，连续查询一般可能能够处理流(即，对流进行查询)和/或关系的数据。此外，关系可以引用流的数据。

[0046] 在一些方面，CQL引擎可以包括充分发展的查询语言。照此，用户可以根据查询指定计算。此外，CQL引擎可以被设计用于优化存储器、利用查询语言特征、运算符共享、丰富模式匹配、丰富语言构造等。此外，在一些示例中，CQL引擎可以处理历史数据和流传输数据两者。例如，用户可以设置查询，以在加州销售额超过某个目标时发送警报。因此，在一些示例中，警报可以至少部分地基于历史销售数据以及传入的实况(即，实时)销售数据。

[0047] 在一些示例中，CQL引擎或下面描述的概念的其它特征可以被配置为以实时方式将历史上下文(即，仓库数据)与传入的数据组合。因此，在一些情况下，本公开可以描述数据库存储的信息和在途信息的边界。数据库存储的信息和在途信息都可以包括BI数据。照此，在一些示例中，数据库可以是BI服务器，或者它可以是任何类型的数据库。另外，在一些示例中，本公开的特征可以使得能够实现上述特征而无需用户知道如何编程或以其它方式编写代码。换句话说，可以以特征丰富的用户接口(UI)或允许非开发人员实现历史数据与实时数据的组合的其它方式来提供特征。

[0048] 在一些示例中，可以利用上述概念来利用与复杂事件处理相关联的丰富的实时和连续事件处理能力。可以支持若干特征，诸如但不限于存档关系。照此，为了利用这些特征

(例如,丰富的、实时的和连续的事件处理),系统可以被配置为透明地处理关系数据的启动状态和运行时状态。换句话说,系统可以被配置为管理在查询创建的瞬间非空的查询(即,存档关系)。

[0049] 在一些示例中,可以使用存档关系。照此,例如,当CQL引擎看到指示查询基于存档关系的查询时,该存档关系还可以指示存在它可以调用以查询历史上下文的某些实体。在一些示例中,数据定义语言(DDL)可以指示关于存档关系的注释,诸如但不限于如何进行查询、表中的重要列是什么、和/或在何处发送其余的数据。在一些示例中,一旦在CQL引擎中构建了查询(例如,作为图),系统就可以分析查询图。此外,在一些方面,存在某些有状态的运算符,如“distinct”、“group aggr”、“pattern”和/或“group by”。但是,无状态运算符可以只接受输入并将其发送给父亲(例如,下游运算符)。因此,一种方法是将整个表存储在这里。但是,通过利用存档关系,系统可以分析查询图并决定可以它使用哪个最低有状态运算符来查询存档。在一些示例中,系统(或一个或多个计算机实现的方法)可以在遍历图形的同时检索所到达的最低有状态运算符处的状态。例如,可以从源以拓扑次序分析查询图。至少部分地基于该第一有状态运算符,CQL引擎然后可以确定要取回的最佳数据量,以便初始化针对存档关系定义的查询的运算符的状态。

[0050] 在至少一个非限制性示例中,比如关系和/或源的源运算符可以在拓扑遍历中首先出现,并且查询输出和/或根最后出现。例如,如果CQL查询看上去像这样:select sum(c1)from R1where c2>c25,那么用于这个查询的计划可以看上去像这样:RelationSource→SELECT→GroupAggr。因此,遵循拓扑次序,并且由于RelationSource和SELECT都是无状态的,因此最低有状态运算符可以是GroupAggr。以这种方式,查询的有状态运算符(在这个示例中为GroupAggr)可以使得查询引擎能够在接收流传输数据之前用来自数据存储区的历史数据填充查询引擎。这可以至少部分地基于查询正在分析存档关系并且存档关系已经被如此指示的事实来实现。

[0051] 在一些示例中,可以由用户指定给定存档关系的窗口尺寸。在一些方面,与存档关系相关的窗口可以包括查询图中的节点,该节点分析或以其它方式评估传入的流传输内容。换句话说,窗口可以定义由查询引擎分析和/或处理的流传输内容的量和/或将包括在存档关系中的历史数据的量。

[0052] 在高级别,一旦窗口被应用于流,它就变成了关系,然后可以应用常规关系逻辑,如同关系数据库一样。当元组到达和离开窗口时,正在考虑的关系会随着针对它编译的查询而改变,同时发出结果。CQL可以支持RANGE(高达纳秒粒度)、ROWS、PARTITION BY和可扩展窗口。这些窗口是流到关系运算符的示例。另一方面,ISTREAM(即,插入流)、DSTREAM(即,删除流)和RSTREAM(即,关系流)是关系到流运算符。在一些示例中,用户、开发人员和/或管理者可以设置由查询引擎或者操作或托管查询引擎的一个或多个计算系统(例如,经由UI)提供的窗口尺寸。在一些示例中,流上的窗口可以是基于时间的范围窗口。例如,可以使用窗口尺寸和在其上计算窗口的属性来指定存档关系上的可配置值窗口。当在存档关系之上指定的可配置值窗口存在时,可以计算快照查询并且可以输出窗口限制内的快照元组。此外,在状态初始化之后,值窗口可以应用于传入的活动数据。在一些示例中,仅传入的活动数据将被插入如下窗口:该窗口的窗口属性的值与当前事件时间的差异小于窗口尺寸。

[0053] 此外,在一些示例中,本公开的特征还可以利用CQL引擎和/或CEP引擎的连续查询

处理能力来支持实时数据分析。在一些方面,CQL引擎和/或CEP引擎传统上可以是面向流的分析引擎;但是,它可以被增强,以支持由耐久存储区支持的面向流的数据(例如,上述存档关系)。例如,本公开描述了可以支持数据对象(DO)的概念的特征,数据对象是耐久存储区(数据库和/或表)。对DO所做的修改可以使改变通知被广播给感兴趣的听众,从而实际上创建数据流。这个数据流可以由CQL引擎和/或CEP引擎消费,以支持任何正在运行的查询;但是,CQL引擎和/或CEP引擎可能未被设计为考虑DO后备存储库中的现有数据。例如,CQL引擎和/或CEP引擎可以请求在CQL引擎和/或CEP引擎中运行的查询的初始状态反映DO的当前状态,该当前状态包括当前在DO后备存储库中的所有数据。一旦该查询被这样初始化,CQL引擎和/或CEP引擎就只需要以传统的面向流的方式关注从该点开始的DO改变通知流。

[0054] 在一些方面,CQL引擎和/或CEP引擎可以按传统方式处理流或非存档关系,因此可能没有初始状态。例如,可以加载查询,其中该查询可以开始运行并监听改变等。在一些情况下,如果用户按状态、在条形图中请求销售额,然后某人进行新的销售,那么表可以被更新并且用户可以预期看到推送给他们的图中的改变。但是,如果他们关闭控制面板并在一周后返回并且带来一些销售额,那么用户可以预期根据总计销售数据表获得销售总额。换句话说,查询可能需要将查询带到存档的状态,然后监听活动的改变。

[0055] 在一些方面,例如,可以利用存档数据来预初始化CQL引擎。一旦被初始化,CQL引擎就可以监听Java消息传递服务(JMS)或其它信使,以(例如,至少部分地基于用于插入、删除等来自存档的数据的API调用)获得改变通知。因此,服务可以监听,并且如果JMS在监听服务正在监听的相同主题上发布,那么它可以接收数据。这些服务不必知道谁在发布,或者它们是否正在发布。监听服务可以只是监听,并且如果发生了什么,监听服务可以听到它。在一些示例中,这是持久性如何例如与其消费者解耦的方式。此外,在一些示例中,警报引擎可以基于警报引擎听到的内容发出警报,并且潜在地进一步,基于可能正在监听与监听器相关的处理查询的SQL引擎发出警报。

[0056] 在一些示例中,查询可以在CQL、SQL和/或CEP引擎中开始,并且指令可以被配置为获取存档数据(例如,以做好准备),然后开始监听这些JMS消息。但是,对于大量插入、删除等,这可以包括大量信息。此外,在监听器听到消息之前可以存在滞后时间,并且在一些示例中,监听可以跳入、查询存档、返回并开始监听。因此,存在丢失事件和/或对事件重复计数的可能性。

[0057] 此外,如果引擎仅运行查询,那么在它正在运行查询的同时,事物(things)可以进入JMS并在引擎未监听的地方发布。因此,引擎可以被配置为首先设置监听器、运行存档查询,然后返回并实际开始拉出队列,以便它不会错过任何内容。因此,JMS可以将事物排队,并且如果事物备份,那么在引擎进行查询时是可以的,因为引擎可以在以后赶上并且不必担心它是否同步。如果它不在这里监听,那么它不会错过它,它只是排队,直到引擎回来,只要引擎已经建立了它的监听器即可。

[0058] 此外,在一些示例中,可以将系统列添加到用户的数据。这个系统列可以用于指示事务ID以尝试处理重复计数和/或错过的操作问题。但是,在其它示例中,系统可以提供或以其它方式生成事务上下文表。此外,可以有两个附加列TRANSACTION_CID和TRANSACTION_TID。上下文表可以始终由持久性服务维护,以便知道最后提交的事务ID的线程(上下文)。对于线程(上下文),可以保证事务ID以升序提交。例如,当服务器启动时,它可以运行持久

性服务。每个持久性服务可以分配上下文ID和事务ID的集合，用于确定预先初始化的信息的数据是否包括已经通过JMS的所有数据。此外，在一些情况下，可以使用多个输出服务器(以符合JTA和/或实现高可用性(HA))，其中每个服务器可以管理与由其它服务器管理的其它表完全分离的单个上下文/事务表集合。

[0059] 在一些实施例中，当连续(例如，CQL)查询被创建或注册时，它可以经受解析和语义分析，在解析和语义分析结束时创建逻辑查询计划。当开始CQL查询时，例如，通过发出“alter query<queryname>start”DDL，可以将逻辑查询计划转换成物理查询计划。在一个示例中，物理查询计划可以被表示为物理运算符的有向无环图(DAG)。然后，可以将物理运算符转换成执行运算符，以得到用于该CQL查询的最终查询计划。到CQL引擎的传入事件到达(一个或多个)源运算符，并最终向下游移动，其中途中的运算符对那些事件执行它们的处理并产生适当的输出事件。

[0060] 事件处理应用

[0061] 在IT环境中，原始基础设施和业务事件的数量和速度都呈指数增长。无论是用于金融服务的流传输股票数据、用于军事的流传输卫星数据还是用于运输和物流业务的实时车辆位置数据，多个行业中的公司都必须实时处理大量复杂数据。此外，移动设备的爆炸式增长和无处不在的高速连接性增加了移动数据的爆炸式增长。与此同时，对业务处理敏捷性和执行的需求也在增长。这两个趋势给组织带来了增加组织的能力以支持事件驱动的体系架构实现模式的压力。实时事件处理要求基础设施和应用开发环境都按事件处理要求执行。这些要求常常包括从日常使用案例扩展到极高速度的数据和事件吞吐量的需要，可能还有以微秒测量的延迟而不是几秒的响应时间。此外，事件处理应用必须常常检测这些事件的流中的复杂模式。

[0062] Oracle Stream Analytics(Oracle流分析)平台面向众多行业和功能领域。以下是一些用例：

[0063] 电信：执行实时呼叫细节(CDR)记录监视和分布式拒绝服务攻击检测的能力。

[0064] 金融服务：利用存在于毫秒或微秒窗口中的套利机会的能力。执行金融证券交易的实时风险分析、监视和报告以及计算外汇价格的能力。

[0065] 运输：在由于当地或目的地城市天气、地勤人员操作、机场安全等引起飞行差异的情况下创建乘客警报并检测行李位置的能力。

[0066] 公共部门/军事：检测分散的地理敌人信息、抽象它并解读敌人攻击的高概率的能力。提醒最合适的资源以应对紧急情况的能力。

[0067] 保险：学习和检测潜在欺诈性索赔的能力。

[0068] IT系统：实时检测故障应用或服务器并触发纠正措施的能力。

[0069] 供应链和物流：实时跟踪货物并检测和报告到达的潜在延迟的能力。

[0070] 实时流传输和事件处理分析

[0071] 随着来自增加数量的连接设备的数据激增，大量动态变化的数据增加；不仅有在组织内部移动的数据，而且还有防火墙外的数据。高速数据带来高价值，尤其是对于多变的业务处理。但是，这种数据中的一些在短时间内失去其操作价值。大数据允许有充裕的时间处理以获得可行动的见解。另一方面，快数据要求从高度动态和战略性数据中提取最大价值。它要求更快地处理，并且促进与生成的数据尽可能接近地采取及时的行动。Oracle

Stream Analytics平台提供对于快数据的响应性。Oracle Edge Analytics将处理推送到网络边缘,从而实时地关联、过滤并分析数据,以供可行动的见解。

[0072] Oracle Stream Analytics平台提供将传入的流传输事件与持久数据联接的能力,从而提供上下文感知的过滤、关联、聚合和模式匹配。它为常见的事件源提供轻量级的开箱即用适配器。它还为定制的适配器开发提供易于使用的适配器框架。利用这个平台,组织可以识别和预见机会,以及由看似无关的事件表示的威胁。它的增量处理范式可以使用提供极低延迟处理的最少量资源来处理事件。它还允许它创建非常及时的警报,并立即检测丢失或延迟的事件,诸如以下:

[0073] 相关连的事件:如果事件A发生,那么事件B几乎总是在事件A的2秒内发生。

[0074] 丢失或失序的事件:事件A、B、C应当按次序发生。在A之后立即看到C,而没有B。

[0075] 因果事件:制造的物品的重量缓慢地趋于变低或读数落在可接受的范围之外。这标志着潜在的问题或未来的维护需求。

[0076] 除了实时事件源之外,Oracle Stream Analytics平台设计环境和运行时执行还支持跨事件流和持久数据存储区(如数据库和高性能数据网格)的基于标准的连续查询执行。这使得平台能够充当需要以微秒或分钟为单位进行回答以辨别否则将被忽视的模式和趋势的系统的智能的核心。事件处理用例要求存储器内处理的速度以及标准数据库SQL的数学准确性和可靠性。该平台查询监听传入的事件流,并对每个事件在存储器中连续执行注册的查询,利用先进的自动化算法进行查询优化。虽然基于存储器中执行模型,但是该平台利用标准的ANSI SQL语法进行查询开发,从而确保查询构造的准确性和可扩展性。这个平台完全符合ANSI SQL'99标准,并且是业界首批支持ANSI SQL审查的用于实时连续查询模式匹配的标准SQL扩展的产品之一。CQL引擎优化处理器内查询的执行,从而使开发人员更多地关注业务逻辑而不是优化。

[0077] Oracle Stream Analytics平台允许组合SQL和Java代码以提供健壮的事件处理应用。通过利用标准行业术语来描述事件源、处理器和事件输出或汇点(sink),这个平台提供元数据驱动的方法,以定义和操纵应用内的事件。它的开发人员使用可视化的有向图画布和调色板进行应用设计,以快速勾勒事件流以及跨事件源和数据源的处理的轮廓。通过拖放建模和配置向导来开发流程,开发人员然后可以输入适当的元数据定义以将设计与实现连接起来。在必要时或优选地,只需一次点击,开发人员就能够进入自定义Java代码开发或直接使用**Spring®**框架将高级概念编码到他们的应用中。

[0078] 事件驱动的应用的特征常常在于在处理极高速率的流传输输入数据时提供低的和确定性的延迟的需要。Oracle Stream Analytics平台的基础是基于**OSGi®**背板的轻量级Java容器。它包含来自WebLogic JEE应用服务器的成熟部件,诸如安全性、日志记录和工作管理算法,但是在实时事件处理环境中利用这些服务。集成的实时内核提供独特的服务来优化由JMX框架支持的线程和存储器管理,从而为了性能和配置而实现与容器的交互。Web 2.0丰富互联网应用可以使用HTTP发布和订阅服务与平台进行通信,这使得它们能够订阅应用频道并将事件推送到客户端。这个平台占用空间小,是提供较快的生产时间以及较低的总体拥有成本的基于Java的轻量级容器。

[0079] Oracle Stream Analytics平台具有在标准的商用硬件上或者最佳地使用Oracle Exalogic及它的其它工程系统的组合以微秒级的处理延迟来每秒处理数百万个事件的能

力。这是通过完整的“自上而下”分层解决方案实现的，该解决方案不仅将设计重点放在高性能事件处理用例上，而且还与企业级实时处理基础设施部件紧密集成。面向性能的服务器集群的平台体系架构专注于可靠性、容错和极高的灵活性并且与Oracle Coherence技术紧密集成，并使得企业能够可预测地跨数据网格扩展任务关键应用，从而确保持续的数据可用性和事务完整性。

[0080] 此外，这个平台允许确定性处理，这意味着可以以不同的速率将相同的事件馈送到多个服务器或同一个服务器中，并且每次都实现相同的结果。与仅依赖正在运行的服务器的系统时钟的系统相比，这实现难以置信的优势。

[0081] 上文和下文描述的技术可以以多种方式并在多种上下文中实现。参考以下附图提供了若干示例实现和上下文，如下面更详细地描述的。但是，以下实现和上下文只是其中的一小部分。

[0082] 在基于微批次的流处理系统中逐个事件处理的框架

[0083] 近年来，已经开发出了数据流管理系统(DSM)，DSM可以针对潜在地无界的实时数据流以连续的方式执行查询。在新的DSM当中，这些系统一般采用基于微批处理的流处理，以便从单个框架提供批处理和流处理的组合。此类系统的示例是在Spark®平台上运行的Spark Streaming应用。由于系统设计的本质，在系统中有状态处理可以是复杂的，因此微批处理流处理具有一些缺点。一个这样的缺点是无法执行“模式匹配”操作。模式匹配是期望流处理系统应当支持的重要特征，并且模式匹配需要高度有状态的处理，以便运行状态机以检测来自无界事件流的模式。

[0084] 通过使用上述Oracle Stream Analytics(流分析)平台，所提出的解决方案将有状态处理与微批处理流处理相结合。本质上，该解决方案组合复杂事件处理(CEP)和微批处理流处理。有状态处理由CQL引擎处理，CQL引擎是用连续查询语言(CQL)编写的连续查询处理引擎。为了支持完全有状态的查询处理，在一个实施例中，将CQL查询引擎添加到微批处理流处理中。

[0085] 在实施例中，公开了可以添加到有向无环图(DAG)变换的CQL变换算法。在某些实施例中，变换算法可以如下实现：(i)来自流处理应用的驱动程序将CQL引擎作为永不返回的长期运行任务启动到执行器中的一个或多个执行器；(ii)CQL引擎保持运行并维护查询状态；(iii)在每个微匹配作业上，CQL变换作为微批次作业的一部分运行；(iv)当执行CQL变换时，微批次的输入事件被发送到CQL引擎；CQL引擎逐个事件处理微批次中的每个事件，执行查询的增量计算，并创建输出事件；(v)在处理微批次中的事件的同时，在队列中捕获输出事件；(vi)在利用CQL引擎完成微批次中的每个事件之后，结果队列中的输出事件作为CQL变换的结果被返回；(vii)CQL变换的下一次变换可以消费输出事件而不需要附加的变换。

[0086] 所公开的CQL变换算法/处理提供了在一般流处理系统中添加CQL变换以处理CQL的能力。此外，通过使用CQL引擎，可以组合功能处理和有状态处理。所公开的处理通过添加复杂事件处理来解决基于微批处理的流处理的若干缺点。而且，通过使用CEP技术的增量计算，可以更高效地执行分析中的一些分析。

[0087] 图1是可以结合本公开实施例的事件处理网络(EPN)的图形表示。如图1中所示，EPN 100可以由若干阶段组成，每个阶段在对事件流中的事件的处理中起不同的作用。根据

定义,事件是基于时间的,因此流感知事件的自然条件。它是事件数据到达Oracle事件处理(Event Processing)应用的方式。为了利用Oracle事件处理来处理事件,构建核心是EPN(诸如EPN 100)的应用。EPN 100由各个阶段组成,每个阶段在处理事件时起着不同的作用,从接收事件数据到查询数据到基于关于事件所发现的内容执行逻辑。应用接收原始事件数据,将数据绑定到事件类型,然后将事件从一个阶段路由到另一个阶段以进行处理。EPN中连接的阶段提供了针对通过EPN的事件执行不同类型的代码的方式。各种阶段可以包括适配器、处理器和bean。更具体而言,在各种实施例中,EPN 100包括接收事件的事件源105、连接阶段的信道110、诸如包含以连续查询语言(CQL)的查询代码的CQL处理器之类的处理器115,以及bean 120、代码125和/或执行通用处理逻辑的汇点130。如本文所述,事件流按时间顺序次序-一个接一个。

[0088] 在一些实施例中,事件源105包括但不限于适配器(例如,JMS、HTTP和文件)、信道、处理器、表、高速缓存等。例如,事件源105可以包括一个或多个适配器。一个或多个适配器可以直接与输入和输出流以及关系源和汇点对接。一个或多个适配器可以被配置为理解输入和输出流协议,并且负责将事件数据转换成可以由应用处理器查询的规格化形式。例如,适配器可以接收事件数据并将其绑定到事件类型实例,然后将事件传递给处理器115。可以为各种数据源和汇点定义一个或多个适配器。信道110充当事件处理端点。除其它以外,信道110还负责对事件数据进行排队,直到事件处理代理可以对事件数据进行动作。处理器115可以是被配置为对事件数据执行动作(诸如对事件数据的查询的执行)的事件处理代理。在某些实施例中,处理器115包括CQL处理器,CQL处理器可以与对输入信道(例如,信道110)提供的事件进行操作的一个或多个CQL查询相关联。例如,处理器的CQL代码可以查询事件(当SQL代码查询数据库行时),从而在数据流经EPN 100时查找数据中的特定模式。CQL处理器可以连接到向其写入查询结果的输出信道(例如,信道110)。例如,满足模式标准的事件可以被传递到(例如,用Java编写的)bean 120或代码125,其中数据可以用在利用从外部源检索出的数据的计算中。另一个下游bean 120或代码125可以使用计算结果来执行使用外部部件的处理。bean 120或代码125可以被注册以监听输出信道,并且通过将新事件插入到输出信道中而被触发。在一些实施例中,bean 120的处理逻辑可以用诸如Java或普通老式Java对象(POJO)之类的编程语言编写。在一些实施例中,处理逻辑可以使用Oracle CEP事件bean API,以便可以由Oracle CEP管理bean。被设计为在EPN 100中接收或发送事件的任何部件(诸如EPN阶段)可以专门实现为这样做。能够接收事件的部件被称为事件汇点130,而发送事件的部件被称为事件源105。单个部件可以既是事件源又是汇点。Oracle事件处理中包括的所描述的阶段部件(诸如CQL处理器所基于的部件和适配器)已经支持所需的功能。开发人员可以通过从OEP API实现接口来为bean、新适配器以及他们编写的其它代码添加事件汇点和源支持。

[0089] 图2描绘了可以结合本公开实施例的事件处理系统200的简化高级图。事件处理系统200可以包括一个或多个事件源(204、206、208)、被配置为提供用于处理事件流的环境的事件处理服务(EPS)202(也称为CQ服务202)以及一个或多个事件汇点(210、212)。事件源生成由EPS 202接收的事件流。EPS 202可以从一个或多个事件源接收一个或多个事件流。例如,如图2中所示,EPS 202从事件源204接收第一输入事件流214,从事件源206接收第二输入事件流216,以及从事件源208接收第三事件流218。一个或多个事件处理应用(220、222和

224)可以部署在EPS 202上并由EPS 202执行。由EPS 202执行的事件处理应用可以被配置为监听一个或多个输入事件流，基于从输入事件流中选择一个或多个事件作为显著事件的处理逻辑来处理经由一个或多个事件流接收的事件。然后可以以一个或多个输出事件流的形式将显著事件发送到一个或多个事件汇点(210、212)。例如，在图2中，EPS 202将第一输出事件流226输出到事件汇点210，并将第二输出事件流228输出到事件汇点212。在某些实施例中，事件源、事件处理应用和事件汇点彼此解耦，使得可以添加或移除这些部件中的任何部件而不会导致对其它部件的改变。

[0090] 在一个实施例中，EPS 202可以被实现为包括轻量级Java应用容器的Java服务器，诸如基于Equinox OSGi、具有共享服务的容器。在一些实施例中，EPS 202可以例如通过使用JRockit Real Time而支持用于处理事件的超高吞吐量和微秒延迟。EPS 202还可以提供开发平台(例如，完整的实时端到端Java事件驱动体系架构(EDA)开发平台)，该开发平台包括用于开发事件处理应用的工具(例如，Oracle CEP Visualizer和Oracle CEP IDE)。

[0091] 事件处理应用被配置为监听一个或多个输入事件流，执行用于从一个或多个输入事件流中选择一个或多个显著事件的逻辑(例如，查询)，并经由一个或多个输出事件流将所选择的显著事件输出到一个或多个事件源。图2提供了用于一个这样的事件处理应用220的深入分析。如图2中所示，事件处理应用220被配置为监听输入事件流218，执行包括用于从输入事件流218中选择一个或多个显著事件的逻辑的连续查询230，并经由输出事件流228将所选择的显著事件输出到事件汇点212。事件源的示例包括但不限于适配器(例如，JMS、HTTP和文件)、信道、处理器、表、高速缓存等。事件汇点的示例包括但不限于适配器(例如，JMS、HTTP和文件)、信道、处理器、高速缓存等。

[0092] 虽然图2中的事件处理应用220被示为监听一个输入流并经由一个输出流输出所选择的事件，但是这不旨在是限制性的。在替代实施例中，事件处理应用可以被配置为监听从一个或多个事件源接收的多个输入流，从被监视的流中选择事件，并经由一个或多个输出事件流将所选择的事件输出到一个或多个事件汇点。相同的查询可以与多于一个事件汇点并且与不同类型的事件汇点相关联。

[0093] 由于其无界性质，经由事件流接收的数据量一般非常大。因此，为了查询目的而存储或存档所有数据一般是不切实际且不期望的。事件流的处理需要在事件由EPS 202接收时实时处理事件，而不必存储所有接收到的事件数据。因而，EPS 202提供特殊的查询机制，该查询机制使得能够在事件由EPS 202接收时执行事件的处理，而不必存储所有接收到的事件。

[0094] 事件驱动的应用是规则驱动的，并且这些规则可以以用于处理输入流的连续查询的形式表达。连续查询可以包括指令(例如，业务逻辑)，该指令识别要对接收到的事件执行的处理，包括什么事件将要被选择为显著事件并且作为查询处理的结果输出。连续查询可以持久保存到数据存储区并且用于处理事件的输入流以及生成事件的输出流。连续查询通常执行过滤和聚合功能，以从输入事件流中发现和提取显著事件。因此，输出事件流中的出站事件的数量一般远低于从中选择事件的输入事件流中的事件的数量。

[0095] 与在有限数据集上运行一次的SQL查询不同，每次在特定事件流中接收到事件时，可以执行已经由具有EPS 202的应用针对该事件流注册的连续查询。作为连续查询执行的一部分，EPS 202基于由连续查询指定的指令来评估接收到的事件，以确定是否一个或多个

事件要被选择为显著事件,并且作为连续查询执行的结果输出。

[0096] 可以使用不同语言对连续查询进行编程。在某些实施例中,可以使用由Oracle公司提供并且由Oracle的复杂事件处理(CEP)产品提供物使用的CQL来配置连续查询。Oracle的CQL是一种声明性语言,其可以用于对可以针对事件流执行的查询(称为CQL查询)进行编程。在某些实施例中,CQL基于具有支持流事件数据处理的添加的构造的SQL。

[0097] 在一个实施例中,事件处理应用可以包括以下部件类型:

[0098] (1)一个或多个适配器,它们直接与输入和输出流以及关系源和汇点对接。适配器被配置为理解输入和输出流协议,并负责将事件数据转换成可由应用处理器查询的规格化形式。适配器可以将规格化的事件数据转发到信道或输出流和关系汇点中。可以为各种数据源和汇点定义事件适配器。

[0099] (2)充当事件处理端点的一个或多个信道。除其它之外,信道还负责对事件数据进行排队,直到事件处理代理可以对其进行动作。

[0100] (3)一个或多个应用处理器(或事件处理代理)被配置为消费来自信道的规格化事件数据,使用查询来处理它以选择显著事件,并将所选择的显著事件转发(或复制)到输出信道。

[0101] (4)一个或多个bean被配置为监听输出信道,并且通过将新事件插入到输出信道中而被触发。在一些实施例中,这个用户代码是普通老式Java对象(POJO)。用户应用可以使用外部服务的集合(诸如JMS、Web服务和文件编写器)来将生成的事件转发到外部事件汇点。

[0102] (5)事件bean可以被注册以监听输出信道,并且通过将新事件插入到输出信道中而被触发。在一些实施例中,这个用户代码可以使用Oracle CEP事件bean API,以便可以由Oracle CEP管理bean。

[0103] 在一个实施例中,事件适配器将事件数据提供给输入信道。输入信道连接到与一个或多个CQL查询相关联的CQL处理器,该一个或多个CQL查询对由输入信道提供的事件进行操作。CQL处理器连接到向其写入查询结果的输出信道。

[0104] 在一些实施例中,可以为事件处理应用提供组装(assembly)文件,该组装文件描述事件处理应用的各个部件、部件如何连接在一起,由应用处理的事件类型。可以提供分开的文件,以用于指定用于选择事件的连续查询或业务逻辑。

[0105] 应当认识到的是,图2中描绘的系统200可以具有除图2中描绘的部件之外的其它部件。另外,图2中示出的实施例仅仅是可以结合本公开实施例的系统的一个示例。在一些其它实施例中,系统200可以具有比图2中所示的更多或更少的部件,可以组合两个或更多个部件,或者可以具有不同的部件配置或布置。系统200可以是各种类型,包括图1中描述的服务提供商计算机106、个人计算机、便携式设备(例如,移动电话或设备)、工作站、网络计算机、大型机、信息亭、服务器或任何其它数据处理系统。在一些其它实施例中,系统200可以被配置为分布式系统,其中系统200的一个或多个部件分布在云中的一个或多个网络上。

[0106] 图2中描绘的部件中的一个或多个部件可以用软件、硬件或其组合来实现。在一些实施例中,软件可以存储在存储器(例如,非瞬态计算机可读介质)中、存储在存储器设备或某种其它物理存储器上,并且可以由一个或多个处理单元(例如,一个或多个处理器、一个或多个处理器核心、一个或多个GPU等)执行。

[0107] 图3是其中可以实现被配置为用于利用微批处理流处理进行有状态处理的流处理应用300的示例系统或体系架构。在各种实施例中，流处理应用300包括一个或多个数据流305。数据流305表示常常仅通过插入新元素而不断变化的数据。与数据集相反，许多类型的应用生成数据流305，包括传感器数据应用、金融报价机、网络性能测量工具、网络监视和流量管理应用以及点击流分析工具。在一些实施例中，Spark Streaming是用于Spark的增量微批处理流处理框架310，并且Spark Streaming提供称为离散流(Dstream)315的数据抽象，其隐藏处理连续数据流的复杂性并使得处理连续数据流对于程序员而言像一次使用一个RDD那样简单。DStream基本上是弹性分布式数据集(RDD)的流，其中元素是从批次的输入流接收的数据(可能由加窗或有状态的运算符在范围上进行扩展)。RDD是Spark的基本数据结构。它是不可变的分布式对象集合。RDD中的每个数据集被划分为逻辑分区，逻辑分区可以在集群的不同节点上计算。RDD可以包含任何类型的Python、Java或Scala对象，包括用户定义的类。在由流处理框架310执行的微批处理中，批次是基本上一次一个RDD。因而，不是一次一个记录地处理数据流305，而是Spark Streaming的接收器并行接受数据并将其缓冲在Spark的工作节点的存储器中。然后，延迟优化的Spark引擎运行短任务(数十毫秒)以处理批次并将结果输出到其它系统。

[0108] 流处理应用300还包括CQL引擎320。检测来自诸如在离散流315内的无界事件流的模式所需的有状态处理由CQL引擎320处理，其中CQL引擎320是用CQL编写的连续查询处理引擎。为了支持完全有状态的查询处理，在一个实施例中，CQL引擎320被添加到用于离散流315内的事件或数据的微批处理流处理中。CQL引擎320使用可以添加到DAG变换的CQL变换算法优化在诸如CQL处理器之类的处理器内的有状态查询处理的执行。变换算法将离散流315作为输入，并与CQL处理器结合以帮助生成结果325。如应当理解的，有两种类型的操作可以针对RDD完成以获得结果：(i)变换，比如导致另一个RDD 325的映射、过滤，以及(ii)导致输出的动作，比如计数。spark作业通常包括对RDD执行变换和动作的任务的DAG。CQL处理器代码可以查询离散流315内的事件，在数据流过EPN(例如，如关于图1所描述的EPN 100)时查找数据中的特定模式。满足模式标准的事件可以作为结果325传递给bean、代码或汇点，其中结果325最终可以作为输入330被传递，以用在利用从外部源检索到的数据的计算中。

[0109] 图4、图7和图8示出了根据一些实施例的用于处理微批处理流以支持完全有状态查询处理的技术。各个实施例可以被描述为被描绘为流程图、流程图表、数据流程图、结构图或框图的处理。虽然流程图可以将操作描述为顺序处理，但是操作中的许多操作可以并行或并发执行。此外，可以重新安排操作的次序。处理在其操作完成时终止，但可以有未包含在图中的附加步骤。处理可以与方法、函数、过程、子例程、子程序等对应。当处理与函数对应时，其终止可以与函数返回到调用函数或主函数对应。

[0110] 图4、图7和图8中描绘的处理和/或操作可以由一个或多个处理单元(例如，处理器核心)执行的软件(例如，代码、指令、程序)、硬件或其组合来实现。软件可以存储在存储器中(例如，存储在存储器设备上、存储在非瞬态计算机可读存储介质上)。图4、图7和图8中的处理步骤的特定系列不旨在是限制性的。根据替代实施例，还可以执行其它步骤序列。例如，在替代实施例中，可以以不同次序执行上面概述的步骤。而且，图4、图7和图8中所示的各个步骤可以包括多个子步骤，该多个子步骤可以按照适合于各个步骤的各种顺序执行。

此外，取决于特定应用，可以添加或移除附加步骤。本领域普通技术人员将认识到许多变化、修改和替代。

[0111] 图4示出了图示由本公开实施例实现的微批处理流的处理以支持完全有状态查询处理的流程图400。在一些实施例中，流程图400中描绘的处理可以由图1、图2和图3的事件处理系统实现。在步骤405处，启动以连续查询语言编写的连续查询处理引擎。在一些实施例中，来自流处理应用的驱动器将CQL引擎作为永不返回的长期运行任务启动到执行器中的一个或多个执行器。CQL引擎保持运行并维护微批次流的查询状态。在步骤410处，接收连续查询。在一些实施例中，查询包括模式识别。例如，以CQL的MATCH_RECOGNIZE子句及其子子句可以被调用，以在CQL查询中执行模式识别。在步骤415处，将操作应用于连续查询，以生成用于连续查询的查询计划。查询计划(或查询执行计划)是用于访问数据的有序步骤集，例如，在SQL关系数据库管理系统中，用于处理查询或连续查询。在一些实施例中，操作是DAG变换并且查询计划是DAG查询计划。DAG变换是顶点和边的集合，其中顶点表示RDD并且边表示要应用于RDD的操作。在步骤420处，使用变换算法来变换查询计划，以生成经变换的查询计划。在各种实施例中，变换算法是CQL变换。例如，在操作微批次或RDD以生成查询计划的每个实例中，执行CQL变换。在一些实施例中，将CQL变换添加到DAG变换，以生成经变换的查询计划。在步骤425处，接收与应用相关的输入事件的微批次流。在一些实施例中，Spark Streaming可以将连续的数据流离散化为微小的亚秒级(sub-second)微批次或微批次流。在步骤430处，至少部分地基于经变换的查询计划来处理输入事件，以生成与应用相关的输出事件集合。在一些实施例中，使用连续查询处理引擎来执行处理，并且处理包括增量地处理每个输入事件以生成输出事件。例如，当执行诸如CQL变换之类的变换算法时，微批次的输入事件被发送到CQL引擎。CQL引擎逐个事件地处理微批次中的每个输入事件，至少部分地基于经变换的查询计划对微批次中的每个输入事件执行用于查询的增量计算，并为微批次中的每个输入事件创建输出事件。照此，有状态处理由CQL引擎执行。在步骤435处，将与应用相关的输出事件集存储在输出队列中。在一些实施例中，输出事件在输出队列中被捕获，而微批次中的剩余事件由CQL引擎处理。在步骤440处，在处理微批次中的每个事件之后，输出队列中的输出事件作为连续查询的结果被返回和/或发送。

[0112] 在基于微批次的事件处理系统中管理快照和应用状态

[0113] 近年来，已经开发了数据流管理系统(DSM)，DSM可以针对潜在无界的实时数据流以连续的方式执行查询。在新的DSM当中，这些系统一般采用基于微批处理的流处理，以便从单个框架提供批处理和流处理的组合。这种系统的示例是在**Spark®**平台上运行的Spark Streaming应用。由于系统设计的性质，在系统中有状态处理可以是复杂的，因此微批处理流处理具有一些缺点。一个这样的缺点是无法执行“模式匹配”操作。模式匹配是期望流处理系统应当支持的重要特征，并且模式匹配需要高度有状态的处理，以便运行状态机来检测来自无界事件流的模式。

[0114] 为了支持完全有状态的查询处理，在一个实施例中，如本文所述，将CQL查询引擎添加到微批处理流处理中。该解决方案基本上组合了复杂事件处理(CEP)和微批处理流处理。有状态处理由CQL引擎处理，CQL引擎是用连续查询语言(CQL)编写的连续查询处理引擎。

[0115] 在某些情况下，集群中可能存在多于一个CQL引擎，并且每个引擎将需要单独创建

用于检查点的状态快照。照此，需要协调快照生成和管理快照，诸如在完成微批次处理之后保留快照。

[0116] 图5是根据本公开实施例的示例系统或体系架构500，在示例系统或体系架构500中可以实现CQL引擎跟踪器605以用于协调快照生成和管理快照。在各种实施例中，系统或体系架构500包括与监听器510通信的CQL引擎跟踪器505。CQL引擎跟踪器505和监听器510可以布置在驱动器515中。CQL引擎跟踪器505被配置为管理从CQL引擎520创建的快照，CQL引擎520可以在集群中的一个或多个执行器525上。在某些实施例中，CQL引擎跟踪器505可以使用两个目录结构，Snapshot Map(快照映射)和Map(映射)来管理快照。Snapshot Map目录结构可以用于直接访问来自给定queryid、分区和时间的快照信息以及映射。Snapshot Map可以用于查找要恢复或清理的快照。图6A示出了Snapshot Map目录结构的示例性数据结构。Map目录结构：(queryId,partitionId,time)->mark.Map:(queryId,partitionId)->List of Snapshot(time,mark,fullFlag)，以相反的次序。Map目录结构的示例性数据结构在图6B中示出。来自CQL引擎520的快照和来自CQL引擎跟踪器505的元数据可以被写入快照存储装置530中。

[0117] 在实施例中，CQL引擎跟踪器结合CQL引擎可以实现快照管理算法。在一些实施例中，快照管理算法可以包括添加快照的处理、获取快照的处理和清理快照的处理。在一些实施例中，AddSnapshot处理包括以下操作：(i)为了管理快照，主结构以相反的次序使用PartitionKey(queryId,partitionId)到Snapshot(time,mark,full_flag)的列表的映射；(ii)CQL引擎在完成计算后向CQLEngineTracker调用addSnapshot RPC，并利用queryId、partitionId、time(时间)和快照标记信息以及full_flag创建快照；(iii)调用AddSnapshot；(iv)利用queryId和partitionId创建PartitionKey对象；(v)如果映射中没有用于partitionKey的列表，那么创建新列表，否则使用现有列表；以及(vi)利用时间、标记和full_flag创建Snapshot对象。

[0118] 在一些实施例中，GetSnapshots处理包括以下操作：(i)CQL RDD(弹性分布式数据集)在开始计算之前利用queryId、partitionId和时间向CQLEngineTracker调用getSnapshot RPC(远程过程调用)，以获取快照的列表以恢复状态；(ii)利用queryId和partitionId创建GetSnapshots PartitionKey；(iiii)利用partitionKey从快照映射中查找快照；(iv)如果没有存储的快照映射，那么返回empty[Snapshot]；(v)在逆序表中为每个快照创建Stack[Snapshot]；以及(vi)如果快照时间小于(时间-batchDuration)，那么将其添加到堆栈，并将堆栈变换成列表并返回。

[0119] 在一些实施例中，CleanSnapshots处理包括以下操作：(i)一旦批次完成，移除快照可能是安全的；(ii)当批处理完成时，从作业调度器调用onEndBatch，它利用批次时间调用EndOfBatch RPC调用；(iii)算法是在给定批次时间之前移除除了完整快照之外的所有快照；(iv)对于快照映射中的每个条目并且对于快照列表中的每个快照，清理快照；以及(v)如果(快照时间小于批次时间)，那么将其从映射中移除，并且还从快照存储装置中移除。

[0120] 图7示出了流程图700，流程图700示出了由本公开实施例实现的微批处理流的处理以支持完全有状态的查询处理。在一些实施例中，流程图700中描绘的处理可以由图1、图2、图3和图5的事件处理系统实现。在步骤705处，接收与应用相关的输入事件的微批次流。

在一些实施例中，Spark Streaming可以将连续的数据流离散化为微小的亚秒级微批次或微批次流。在步骤710处，使用连续查询处理引擎处理输入事件，以生成与应用相关的输出事件集合。在一些实施例中，处理包括增量地处理每个输入事件以生成输出事件。例如，当执行诸如CQL变换之类的变换算法时，微批次的输入事件被发送到CQL引擎。CQL引擎逐个事件地处理微批次中的每个输入事件，至少部分地基于经变换的查询计划对微批次中的每个输入事件执行增量计算，并为微批次中的每个输入事件创建输出事件。照此，有状态处理由CQL引擎执行。在步骤715处，至少部分地基于与应用相关的事件的输出事件集合来生成系统的当前状态的快照。在一些实施例中，使用由CQL引擎实现的快照管理算法生成快照。在某些实施例中，快照管理算法可以包括添加快照的处理、获取快照的处理和清理快照的处理。在步骤720处，生成第一目录结构，以访问与系统的当前状态的快照相关联的快照信息。在一些实施例中，第一目录结构是快照映射目录结构。在步骤725处，生成第二目录结构，以生成与系统的当前状态相关联的快照的列表。在一些实施例中，第二目录结构是Map(映射)目录结构。在步骤730处，至少部分地基于快照管理算法确定处理，以生成、添加或清理与系统的当前状态有关的快照列表。在一些实施例中，当快照管理算法包括添加快照的处理时，该处理被确定为添加与系统的当前状态有关的快照的列表。在一些实施例中，当快照管理算法包括获取快照的处理时，该处理被确定为获得与系统的当前状态有关的快照列表。在一些实施例中，当快照管理算法包括清理快照的处理时，该处理被确定为清理与系统的当前状态有关的快照列表。应当理解的是，该处理还可以包括关于图4描述的步骤，例如，启动连续查询处理引擎、将操作应用于连续查询以生成用于连续查询的查询计划、变换查询计划以生成经变换的查询计划、至少部分地基于经变换的查询计划来处理输入事件以生成输出事件集合、将与应用相关的输出事件集合存储在输出队列中，并且在处理微批次中的每个事件之后，输出队列中的输出事件可以作为连续查询的结果被返回和/或发送。

[0121] 本公开的实施例提供有状态部件，这些有状态部件维护Spark Streaming系统中的运行状态，在微批处理流处理内提供完全有状态的CQL引擎，管理从分布式CQL引擎创建的快照，并提供用于处理增量快照的保留算法。即使在将逐个事件的CEP处理添加到基于微批处理的流处理之后，所公开的技术也允许高可用性。

[0122] 非侵入式监视Spark Streaming中的阶段的输出

[0123] 近年来，已经开发了数据流管理系统(DSM)，DSM可以针对潜在无界的实时数据流以连续的方式执行查询。在新的DSM当中，这些系统一般采用基于微批处理的流处理，以便从单个框架提供批处理和流处理的组合。这种系统的示例是在**Spark®**平台上运行的Spark Streaming应用。

[0124] DSMS中的典型应用被设计为操作或变换的DAG(有向无环图)形状的“拓扑”。拓扑充当数据变换流水线。大多数流处理系统(例如，Spark Streaming系统)提供了将用于应用的拓扑快速部署到机器集群并且能够立即查看结果的方式。此类部署的快速周转周期对于对应用进行改变是重要的。如果周转周期足够快，那么用户可以在无需等待部署延迟的情况下查看结果。这被称为“流探索”。

[0125] 在流探索模式下，客户一般通过将新部件添加到现有拓扑或数据变换流水线来增量地开发业务应用。在这种探索模式下，重要的是看到来自改变的即时输出以及来自流水线中的每个阶段的中间输出。

[0126] 在诸如Spark®Streaming或Apache®Flink之类的当前DSMS中，使用诸如Java、Scala或Closure之类的编程语言来编写拓扑。因此，当应用开发人员想要监视来自一个变换的中间输出时，开发人员必须改变程序并添加输出操作。这不仅麻烦而且具有侵入性，因为所有输出操作通常都会成为一些系统(比如Spark Streaming)的附加作业。使情况更复杂的是，目前没有机制在将其被放入应用之后在应用运行的时候进行输出监视。

[0127] 在实施例中，公开了一种监视变换处理，其具有以下特征：(i)直通变换，其在向指定目的地发送输出的同时生成到下一流水线的输出而不添加任何变换、(ii)监视输出在应用中被配置，以及(iii)监视输出可以在运行应用时被关闭或改变。

[0128] 在实施例中，可以使用以下示例来实现以上特征：

[0129] val s1＝cc.cql(inputs,"select*from stream")

[0130] val producerConfig＝KafkaMonitorConfig(outputTopic,brokerList)

[0131] val s1output＝s1.monitor(KafkaMonitorOutput(producerConfig))；

[0132] val s2output＝cc.cql(s1output,"select*from s1")

[0133] 以上示例的流程可以描述如下：(i)通过向“s1”DStream调用“monitor(监视器)”，在“s1”的CQLDStream之后将MonitorDStream添加到DAG。MonitorDStream通过配置携带有关KafkaMonitorOutput的信息；(ii)作业生成步骤从MonitorDStream创建MonitorRDD；(iii)当作业运行时，调用MonitorRDD.compute；以及(iv)pathThroughIterator将输出写入配置的监视器输出，同时将元组返回到下一个流水线。

[0134] 关闭监视输出或更新配置的流程可以如下实现：(i)从应用运行REST服务以获得更新；(ii)为应用和阶段生成的MonitorDStream实例存储在应用中，并且可以使用appname和stagename作为键找到它；(iii)REST请求，诸如’/monitoroutput/<appname>/<stagename>/off上的PUT操作或’/monitoroutput/<appname>/<stagename>/configure上的POST操作，其中新配置将被委托给“MonitorOutputManager”部件；(iv)MonitorOutputManager将改变MonitorDStream对象实例的设置或配置；以及(v)由作业运行器运行的下一个作业将受到改变的影响。

[0135] 图8示出了流程图800，流程图800示出了由本公开实施例实现的微批处理流的处理以支持完全有状态查询处理。在一些实施例中，流程图800中描绘的处理可以由图1、图2和图3的事件处理系统实现。在步骤805处，接收连续查询。在一些实施例中，查询包括模式识别。例如，可以调用按CQL的MATCH_RECOGNIZE子句及其子子句，以在CQL查询中执行模式识别。在步骤810处，对连续查询应用操作，以生成用于连续查询的查询计划。查询计划(或查询执行计划)是用于访问数据的有序步骤集，例如，在SQL关系数据库管理系统中，用于处理查询或连续查询。在一些实施例中，操作是DAG变换，并且查询计划是DAG查询计划。DAG变换是顶点和边的集合，其中顶点表示RDD并且边表示要应用于RDD的操作。在步骤815处，使用监视变换处理来监视连续查询。例如，监视变换处理可以具有以下特征：(i)直通变换，其在将输出发送到指定目的地的同时生成到下一个流水线的输出而不添加任何变换、(ii)监视输出在应用中被配置，以及(iii)在运行应用时监视输出可以被关闭或改变。在步骤820处，接收与应用相关的输入事件的微批次流。在一些实施例中，Spark Streaming可以将连续的数据流离散化为微小的亚秒级微批次或微批次流。在步骤825处，至少部分地基于监视变换处理来处理输入事件，以生成与应用相关的输出事件集合。在一些实施例中，使用连续

查询处理引擎来执行处理，并且处理包括增量地处理每个输入事件以生成输出事件。例如，当执行监视变换处理时，微批次的输入事件被发送到CQL引擎。CQL引擎逐个事件地处理微批次中的每个输入事件，至少部分地基于监视变换处理对微批次中的每个输入事件执行针对查询的增量计算，并为微批次中的每个输入事件创建输出事件。照此，有状态处理由CQL引擎执行。与应用相关的输出事件集合存储在输出队列中。在一些实施例中，输出事件在输出队列中被捕获，而微批次中的剩余事件由CQL引擎处理。如应当理解的，该处理还可以包括关于图4描述的步骤，例如，启动连续查询处理引擎，并且在处理微批次中的每个事件之后，输出队列中的输出事件可以作为连续查询的结果被返回和/或发送。

[0136] 本公开的实施例提供了利用Spark Streaming的非侵入式输出监视技术以及用于打开/关闭来自正在运行的Spark Streaming应用的中间输出的技术。此外，所公开的技术使得能够添加用于流探测的中间输出监视以及改变输出。

[0137] 说明性系统

[0138] 图9-7示出了用于实现根据各种实施例的本公开的方面的示例环境的各方面。图9描绘了用于实现本公开实施例的分布式系统900的简化图。在所示实施例中，分布式系统900包括一个或多个客户端计算设备902、904、906和908，该一个或多个客户端计算设备被配置为通过(一个或多个)网络910执行和操作客户端应用，诸如web浏览器、专有客户端(例如，Oracle Forms)等等。服务器912可以经由网络910与远程客户端计算设备902、904、906和908通信地耦合。

[0139] 在各种实施例中，服务器912可以适于运行一个或多个服务或软件应用，诸如提供身份管理服务的服务和应用。在某些实施例中，服务器912还可以提供其它服务，或者软件应用可以包括非虚拟和虚拟环境。在一些实施例中，这些服务可以作为基于web的服务或云服务或者在软件即服务(SaaS)模型下提供给客户端计算设备902、904、906和/或908的用户。操作客户端计算设备902、904、906和/或908的用户又可以利用一个或多个客户端应用与服务器912交互以利用由这些部件提供的服务。

[0140] 在图9描绘的配置中，系统900的软件部件918、920和922被示为在服务器912上实现。在其它实施例中，系统900的部件中的一个或多个部件和/或由这些部件提供的服务还可以由客户端计算设备902、904、906和/或908中的一个或多个来实现。然后，操作客户端计算设备的用户可以利用一个或多个客户端应用来使用由这些部件提供的服务。这些部件可以用硬件、固件、软件或其组合来实现。应该理解的是，各种不同的系统配置是可能的，其可以与分布式系统900不同。因此，图9中所示的实施例是用于实现实施例系统的分布式系统的一个示例，并且不旨在是限制性的。

[0141] 客户端计算设备902、904、906和/或908可以包括各种类型的计算系统。例如，客户端设备可以包括便携式手持设备(例如，**iPhone®**、蜂窝电话、**iPad®**、计算平板、个人数字助理(PDA))或可穿戴设备(例如，Google **Glass®**头戴式显示器)，其运行诸如Microsoft Windows **Mobile®**之类的软件和/或诸如iOS、Windows Phone、Android、BlackBerry 10、Palm OS等之类的各种移动操作系统。设备可以支持各种应用(诸如各种互联网相关的应用、电子邮件、短消息服务(SMS)应用)，并且可以使用各种其它通信协议。客户端计算设备还可以包括通用个人计算机，作为示例包括运行各种版本的Microsoft **Windows®**、Apple **Macintosh®**和/或Linux操作系统的个人计算机和/或膝上型计算机。客户端计算

设备可以是运行各种商用的**UNIX®**或类UNIX操作系统(包括但不限于诸如像Google Chrome OS之类的各种GNU/Linux操作系统)中的任何操作系统的工作站计算机。客户端计算设备还可以包括能够通过(一个或多个)网络910通信的电子设备,诸如瘦客户端计算机、启用互联网的游戏系统(例如,具有或不具有**Kinect®**姿势输入设备的Microsoft Xbox游戏控制台)和/或个人消息传送设备。

[0142] 虽然图9中的分布式系统900被示为具有四个客户端计算设备,但是可以支持任何数量的客户端计算设备。其它设备(诸如具有传感器的设备等)可以与服务器912交互。

[0143] 分布式系统900中的(一个或多个)网络910可以是对本领域技术人员熟悉的可以利用各种可用协议中的任何协议支持数据通信的任何类型的网络,其中各种可用协议包括但不限于TCP/IP(传输控制协议/互联网协议)、SNA(系统网络体系架构)、IPX(互联网分组交换)、AppleTalk等。仅仅作为示例,(一个或多个)网络910可以是局域网(LAN)、基于以太网的网络、令牌环、广域网、互联网、虚拟网络、虚拟专用网络(VPN)、内联网、外联网、公共交换电话网络(PSTN)、红外网络、无线网络(例如,在电气和电子协会(IEEE)1002.11协议套件中的任何协议、**Bluetooth®**、和/或任何其它无线协议下操作的网络)和/或这些网络和/或其它网络的任意组合。

[0144] 服务器912可以由一个或多个通用计算机、专用服务器计算机(作为示例,包括PC(个人计算机)服务器、**UNIX®**服务器、中型服务器、大型计算机、机架安装的服务器等)、服务器场、服务器集群或任何其它适当的布置和/或组合组成。服务器912可以包括运行虚拟操作系统的一个或多个虚拟机,或涉及虚拟化的其它计算体系架构。一个或多个灵活的逻辑存储设备池可以被虚拟化,以维护用于服务器的虚拟存储设备。虚拟网络可以由服务器912利用软件定义的联网来控制。在各种实施例中,服务器912可以适于运行在前述公开内容中描述的一个或多个服务或软件应用。例如,服务器912可以与根据本公开的实施例的用于执行如上所述的处理的服务器对应。

[0145] 服务器912可以运行包括以上讨论的操作系统中的任何操作系统的操作系统,以及任何商用的服务器操作系统。服务器912还可以运行各种附加的服务器应用和/或中间层应用中的任何一个,包括HTTP(超文本传输协议)服务器、FTP(文件传输协议)服务器、CGI(公共网关接口)服务器、**JAVA®**服务器、数据库服务器等。示例性数据库服务器包括但不限于来自Oracle、Microsoft、Sybase、IBM(国际商业机器)等的商用数据库服务器。

[0146] 在一些实现中,服务器912可以包括一个或多个应用,以分析和整合从客户端计算设备902、904、906和908的用户接收到的数据馈送和/或事件更新。作为示例,数据馈送和/或事件更新可以包括但不限于从一个或多个第三方信息源和持续数据流接收到的**Twitter®**馈送、**Facebook®**更新或实时更新,其可以包括与传感器数据应用、金融报价机、网络性能测量工具(例如,网络监视和流量管理应用)、点击流分析工具、汽车交通监视等相关的实时事件。服务器912还可以包括经由客户端计算设备902、904、906和908的一个或多个显示设备显示数据馈送和/或实时事件的一个或多个应用。

[0147] 分布式系统900还可以包括一个或多个数据库914和916。这些数据库可以提供用于存储信息(诸如用户身份信息和由本公开实施例使用的其它信息)的机制。数据库914和916可以驻留在各种位置中。举例来说,数据库914和916中的一个或多个可以驻留在服务器912本地(和/或驻留在服务器912中)的非瞬态存储介质上。可替代地,数据库914和916可以

远离服务器912并经由基于网络的或专用的连接与服务器912通信。在一组实施例中，数据库914和916可以驻留在存储区域网络(SAN)中。类似地，用于执行归属于服务器912的功能的任何必要文件可以适当地本地存储在服务器912上和/或远程存储。在一组实施例中，数据库914和916可以包括适于响应于SQL格式的命令来存储、更新和检索数据的关系数据库，诸如由Oracle提供的数据库。

[0148] 图10示出了可以用于实现本公开实施例的示例性计算机系统1000。在一些实施例中，计算机系统1000可以用于实现上述各种服务器和计算机系统中的任何服务器和计算机系统。如图10中所示，计算机系统1000包括包含处理子系统1004的各种子系统，处理子系统1004经由总线子系统1002与多个外围子系统通信。这些外围子系统可以包括处理加速单元1006、I/O子系统1008、存储子系统1018和通信子系统1024。存储子系统1018可以包括有形计算机可读存储介质1022和系统存储器1010。

[0149] 总线子系统1002提供用于使计算机系统1000的各种部件和子系统按预期彼此通信的机制。虽然总线子系统1002被示意性地示为单个总线，但总线子系统的替代实施例可以使用多个总线。总线子系统1002可以是若干类型的总线结构中的任何一种，包括使用各种总线体系架构中的任何总线体系架构的存储器总线或存储器控制器、外围总线和本地总线。例如，这样的体系架构可以包括工业标准体系架构(ISA)总线、微通道体系架构(MCA)总线、增强型ISA(EISA)总线、视频电子标准协会(VESA)本地总线和可以实现为根据IEEE P1386.1标准制造的夹层(Mezzanine)总线的外围部件互连(PCI)总线，等等。

[0150] 处理子系统1004控制计算机系统1000的操作，并且可以包括一个或多个处理单元1032、1034等。处理单元可以包括一个或多个处理器，包括单核或多核处理器、处理器的一个或多个核心或其组合。在一些实施例中，处理子系统1004可以包括一个或多个专用协处理器，诸如图形处理器、数字信号处理器(DSP)等。在一些实施例中，处理子系统1004的处理单元中的一些或全部可以使用定制电路来实现，定制电路诸如专用集成电路(ASIC)或现场可编程门阵列(FPGA)。

[0151] 在一些实施例中，处理子系统1004中的处理单元可以执行存储在系统存储器1010中或计算机可读存储介质1022上的指令。在各种实施例中，处理单元可以执行各种程序或代码指令并且可以维护多个并发执行的程序或进程。在任何给定时间，要执行的程序代码中的一些或全部可以驻留在系统存储器1010中和/或驻留在计算机可读存储介质1010上，包括潜在地在一个或多个存储设备上。通过适当的编程，处理子系统1004可以提供上述各种功能，以用于响应于使用模式而动态地修改文档(例如，网页)。

[0152] 在某些实施例中，可以提供处理加速单元1006，用于执行定制处理或用于卸载由处理子系统1004执行的处理中的一些处理，以便加速由计算机系统1000执行的整体处理。

[0153] I/O子系统1008可以包括用于向计算机系统1000输入信息和/或用于从计算机系统1000或经由计算机系统1000输出信息的设备和机制。一般而言，术语“输入设备”的使用旨在包括用于向计算机系统1000输入信息的所有可能的类型的设备和机制。用户接口输入设备可以包括例如键盘、诸如鼠标或轨迹球之类的定点设备、结合到显示器中的触摸板或触摸屏、滚轮、点击轮、拨号盘、按钮、开关、小键盘、带语音命令识别系统的音频输入设备、麦克风和其它类型的输入设备。用户接口输入设备还可以包括使用户能够控制输入设备并与输入设备交互的诸如Microsoft **Kinect®**运动传感器之类的运动感测和/或姿势识别设

备、Microsoft Xbox® 360游戏控制器、提供用于接收利用姿势和口语命令的输入的接口的设备。用户接口输入设备还可以包括眼睛姿势识别设备，诸如从用户检测眼睛活动(例如，当拍摄图片和/或进行菜单选择时的“眨眼”)并将眼睛姿势变换为到输入设备(例如，Google Glass®)中的输入的Google Glass®眨眼检测器。此外，用户接口输入设备可以包括使用户能够通过语音命令与语音识别系统(例如，Siri®导航器)交互的语音识别感测设备。

[0154] 用户接口输入设备的其它示例包括但不限于三维(3D)鼠标、操纵杆或指点杆、游戏手柄和图形输入板，以及诸如扬声器、数码相机、数码摄像机、便携式媒体播放器、网络摄像头、图像扫描仪、指纹扫描仪、条形码阅读器3D扫描仪、3D打印机、激光测距仪和视线跟踪设备之类的音频/视觉设备。此外，用户接口输入设备可以包括，例如，医疗成像输入设备，诸如计算机断层摄影、磁共振成像、位置发射断层摄影、医疗超声检查设备。用户接口输入设备还可以包括，例如，音频输入设备，诸如MIDI键盘、数字乐器等。

[0155] 用户接口输出设备可以包括显示子系统、指示器灯或诸如音频输出设备之类的非可视显示器等。显示子系统可以是阴极射线管(CRT)、诸如使用液晶显示器(LCD)或等离子体显示器的平板设备之类的平板设备、投影设备、触摸屏等。一般而言，术语“输出设备”的使用旨在包括用于从计算机系统1000向用户或其它计算机输出信息的所有可能类型的设备和机制。例如，用户接口输出设备可以包括但不限于可视地传达文本、图形和音频/视频信息的各种显示设备，诸如监视器、打印机、扬声器、耳机、汽车导航系统、绘图仪、语音输出设备和调制解调器。

[0156] 存储子系统1018提供用于存储由计算机系统1000使用的信息的储存库或数据存储库。存储子系统1018提供有形非瞬态计算机可读存储介质，用于存储提供一些实施例的功能的基本编程和数据结构。当由处理子系统1004执行时提供上述功能的软件(程序、代码模块、指令)可以存储在存储子系统1018中。软件可以由处理子系统1004的一个或多个处理单元执行。存储子系统1018还可以提供用于存储根据本公开使用的数据的储存库。

[0157] 存储子系统1018可以包括一个或多个非瞬态存储器设备，包括易失性和非易失性存储器设备。如图10中所示，存储子系统1018包括系统存储器1010和计算机可读存储介质1022。系统存储器1010可以包括多个存储器，该多个存储器包括用于在程序执行期间存储指令和数据的易失性主随机存取存储器(RAM)和其中存储固定指令的非易失性只读存储器(ROM)或闪存存储器。在一些实现中，包含帮助诸如在启动期间在计算机系统1000内的元件之间传送信息的基本例程的基本输入/输出系统(BIOS)通常可以存储在ROM中。RAM通常包含当前由处理子系统1004操作和执行的数据和/或程序模块。在一些实现中，系统存储器1010可以包括多个不同类型的存储器，诸如静态随机存取存储器(SRAM)或动态随机存取存储器(DRAM)。

[0158] 作为示例而非限制，如在图10中所描绘的，系统存储器1010可以存储应用程序1012、程序数据1014和操作系统1016，其中应用程序1012可以包括客户端应用、Web浏览器、中间层应用、关系数据库管理系统(RDBMS)等。作为示例，操作系统1016可以包括各种版本的Microsoft Windows®、Apple Macintosh®和/或Linux操作系统、各种商用UNIX®或类UNIX操作系统(包括但不限于各种GNU/Linux操作系统、Google Chrome® OS等)和/或诸如iOS、Windows® Phone、Android® OS、BlackBerry® 10OS和Palm® OS操作系统之类

的移动操作系统。

[0159] 计算机可读存储介质1022可以存储提供一些实施例的功能的编程和数据结构。当由处理子系统1004执行时使处理器提供上述功能的软件(程序、代码模块、指令)可以存储在存储子系统1018中。作为示例,计算机可读存储介质1022可以包括非易失性存储器,诸如硬盘驱动器、磁盘驱动器、诸如CD ROM、DVD、**Blu-Ray®**(蓝光)盘或其它光学介质之类的光盘驱动器。计算机可读存储介质1022可以包括但不限于**Zip®**驱动器、闪存存储器卡、通用串行总线(USB)闪存驱动器、安全数字(SD)卡、DVD盘、数字视频带等。计算机可读存储介质1022还可以包括基于非易失性存储器的固态驱动器(SSD)(诸如基于闪存存储器的SSD、企业闪存驱动器、固态ROM等)、基于易失性存储器的SSD(诸如基于固态RAM、动态RAM、静态RAM、DRAM的SSD、磁阻RAM(MRAM)SSD)、以及使用基于DRAM的SSD和基于闪存存储器的SSD的组合的混合SSD。计算机可读介质1022可以为计算机系统1000提供计算机可读指令、数据结构、程序模块和其它数据的存储。

[0160] 在某些实施例中,存储子系统1000还可以包括计算机可读存储介质读取器1020,计算机可读存储介质读取器1020可以进一步连接到计算机可读存储介质1022。与系统存储器1010一起并且可选地组合,计算机可读存储介质1022可以全面地表示远程存储设备、本地存储设备、固定存储设备和/或可移动存储设备加上用于存储计算机可读信息的存储介质。

[0161] 在某些实施例中,计算机系统1000可以提供对执行一个或多个虚拟机的支持。计算机系统1000可以执行诸如管理程序之类的程序,以便促进虚拟机的配置和管理。每个虚拟机可以被分配存储器、计算资源(例如,处理器、内核)、I/O资源和联网资源。每个虚拟机通常运行其自己的操作系统,该操作系统可以与由计算机系统1000执行的其它虚拟机执行的操作系统相同或不同。相应地,多个操作系统可以潜在地由计算机系统1000并发地运行。每个虚拟机一般独立于其它虚拟机运行。

[0162] 通信子系统1024提供到其它计算机系统和网络的接口。通信子系统1024用作用于从计算机系统1000接收数据和从计算机系统1000向其他系统发送数据的接口。例如,通信子系统1024可以使计算机系统1000能够经由互联网建立到一个或多个客户端设备的通信信道,用于从客户端设备接收信息和发送信息到客户端设备。此外,通信子系统1024可以用于从特权账户管理器向发出请求的用户传送成功登录的通知或重新输入密码的通知。

[0163] 通信子系统1024可以支持有线通信协议和/或无线通信协议两者。例如,在某些实施例中,通信子系统1024可以包括用于(例如,使用蜂窝电话技术、高级数据网络技术(诸如3G、4G或EDGE(全球演进的增强数据速率)、WiFi(IEEE 802.11族标准)、或其它移动通信技术、或其任意组合)接入无线语音和/或数据网络的射频(RF)收发器部件、全球定位系统(GPS)接收器部件和/或其它部件。在一些实施例中,作为无线接口的附加或替代,通信子系统1024可以提供有线网络连接(例如,以太网)。

[0164] 通信子系统1024可以以各种形式接收和发送数据。例如,在一些实施例中,通信子系统1024可以接收以结构化和/或非结构化的数据馈送1026、事件流1028、事件更新1030等形式的输入通信。例如,通信子系统1024可以被配置为实时地从社交媒体网络的用户和/或诸如**Twitter®**馈送、**Facebook®**更新、诸如丰富站点摘要(RSS)馈送之类的web馈送之类的其它通信服务接收(或发送)数据馈送1026,和/或来自一个或多个第三方信息源的实时

更新。

[0165] 在某些实施例中，通信子系统1024可以被配置为接收以连续数据流的形式的本质上可能是连续的或无界的没有明确结束的数据，其中连续数据流可以包括实时事件的事件流1028和/或事件更新1030。生成连续数据的应用的示例可以包括例如传感器数据应用、金融报价机、网络性能测量工具(例如网络监视和流量管理应用)、点击流分析工具、汽车交通监视等。

[0166] 通信子系统1024还可以被配置为向一个或多个数据库输出结构化和/或非结构化的数据馈送1026、事件流1028、事件更新1030等，其中该一个或多个数据库可以与耦合到计算机系统1000的一个或多个流传输数据源计算机通信。

[0167] 计算机系统1000可以是各种类型中的一种，包括手持便携式设备(例如，**iPhone®**蜂窝电话、**iPad®**计算平板、PDA)、可穿戴设备(例如，Google **Glass®**头戴式显示器)、个人计算机、工作站、大型机、信息站、服务器机架或任何其它数据处理系统。

[0168] 由于计算机和网络不断变化的性质，对图10中描绘的计算机系统1000的描述旨在仅仅作为具体示例。具有比图10中所描绘的系统更多或更少的部件的许多其它配置是可能的。基于本文所提供的公开内容和教导，本领域普通技术人员将理解实现各种实施例的其它方式和/或方法。

[0169] 附图中的一些附图中描绘的系统可以以各种配置提供。在一些实施例中，系统可以被配置为分布式系统，其中系统的一个或多个部件跨一个或多个网络分布在一个或多个云基础设施系统中。

[0170] 云基础设施系统是一个或多个服务器计算设备、网络设备和/或存储设备的集合。这些资源可以由云服务提供商划分，并以某种方式分配给其客户。例如，云服务提供商(诸如加利福尼亚州红木海岸的Oracle公司)可以提供各种类型的云服务，包括但不限于在软件即服务(SaaS)类别下提供的一个或多个服务、在平台即服务(PaaS)类别下提供的服务、在基础设施即服务(IaaS)类别下提供的服务，或包括混合服务的其它类别的服务。SaaS服务的示例包括但不限于构建和递送一套按需应用(诸如Oracle Fusion应用)的能力。SaaS服务使客户能够利用在云基础设施系统上执行的应用，而无需客户购买用于应用的软件。PaaS服务的示例包括但不限于使组织(诸如Oracle)能够在共享的公共体系架构上整合现有应用的服务，以及构建利用平台提供的共享服务(诸如Oracle Java云服务(JCS)、Oracle数据库云服务(DBCS)等)的新应用的能力。IaaS服务通常有助于管理和控制底层计算资源(诸如存储、网络和其它基本计算资源)，以用于让客户利用SaaS平台和PaaS平台提供的服务。

[0171] 图11是根据本公开实施例的系统环境1100的一个或多个部件的简化框图，通过该系统环境1100，可以提供由实施例系统的一个或多个部件提供的服务作为云服务。在所示实施例中，系统环境1100包括可以由用户用于与提供云服务的云基础设施系统1102交互的一个或多个客户端计算设备1104、1106和1108。客户端计算设备可以被配置为操作可以由客户端计算设备的用户用来与云基础设施系统1102交互以使用由云基础设施系统1102提供的服务的客户端应用，诸如web浏览器、专有客户端应用(例如，Oracle Forms)或某个其它应用。

[0172] 应当认识到的是，图中描绘的云基础设施系统1102可以具有除所描绘的部件之外

的其它部件。另外，图中所示的实施例仅仅是可以结合本公开实施例的云基础设施系统的一个示例。在一些其它实施例中，云基础设施系统1102可以具有比图中所示的更多或更少的部件、可以组合两个或更多个部件、或者可以具有不同的部件配置或布置。

[0173] 客户端计算设备1104、1106和1108可以是与上面针对902、904、906和908描述的设备类似的设备。

[0174] 虽然示出了具有三个客户端计算设备的示例性系统环境1100，但是可以支持任何数量的客户端计算设备。诸如具有传感器的设备等之类的其它设备可以与云基础设施系统1102交互。

[0175] (一个或多个)网络1110可以促进客户端1104、1106和1108与云基础设施系统1102之间的数据通信和交换。每个网络可以是本领域技术人员熟悉的、可以使用各种商用协议(包括上面针对(一个或多个)网络910描述的协议)中的任何一种来支持数据通信的任何类型的网络。

[0176] 云基础设施系统1102可以包括一个或多个计算机和/或服务器，该一个或多个计算机和/或服务器可以包括上面针对服务器912描述的计算机和/或服务器。

[0177] 在某些实施例中，由云基础设施系统提供的服务可以包括按需对云基础设施系统的用户可用的大量服务，诸如在线数据存储和备份解决方案、基于Web的电子邮件服务、托管的办公套件和文档协作服务、数据库处理、受管理的技术支持服务等。由云基础架构系统提供的服务可以动态扩展以满足其用户的需求。由云基础设施系统提供的服务的具体实例化在本文中被称为“服务实例”。一般而言，经由通信网络(诸如互联网)从云服务提供商的系统对用户可用的任何服务被称为“云服务”。通常，在公共云环境中，构成云服务提供商的系统的服务器和系统与客户自己的本地服务器和系统不同。例如，云服务提供商的系统可以托管应用，并且用户可以经由诸如互联网之类的通信网络按需订购和使用应用。

[0178] 在一些示例中，计算机网络云基础设施中的服务可以包括对存储装置、托管的数据库、托管的web服务器、软件应用或者由云供应商向用户提供或者如本领域中已知的其他方式提供的其它服务的受保护的计算机网络访问。例如，服务可以包括通过互联网对云上的远程存储装置的受密码保护的访问。作为另一个示例，服务可以包括基于web服务的托管的关系数据库和脚本语言中间件引擎，用于由联网的开发人员私人使用。作为另一个示例，服务可以包括对在云供应商的网站上托管的电子邮件软件应用的访问。

[0179] 在某些实施例中，云基础设施系统1102可以包括以自助服务、基于订阅、弹性可扩展、可靠、高度可用和安全的方式交付给客户的应用套件、中间件和数据库服务提供物。这种云基础设施系统的示例是由本受让人提供的Oracle Public Cloud(Oracle公共云)。

[0180] 在各种实施例中，云基础设施系统1102可以适于自动地供应、管理和跟踪客户对由云基础设施系统1102提供的服务的订阅。云基础设施系统1102可以经由不同的部署模型提供云服务。例如，服务可以在公共云模型下提供，其中云基础设施系统1102由销售云服务的组织拥有(例如，由Oracle拥有)并且服务对一般公众或不同的行业企业可用。作为另一个示例，服务可以在私有云模型下提供，其中云基础设施系统1102仅针对单个组织操作，并且可以为组织内的一个或多个实体提供服务。云服务还可以在社区云模型下提供，其中云基础设施系统1102和由云基础设施系统1102提供的服务由相关社区中的若干个组织共享。云服务还可以在混合云模型下提供，混合云模型是两个或更多个不同模型的组合。

[0181] 在一些实施例中，由云基础设施系统1102提供的服务可以包括在软件即服务(SaaS)类别、平台即服务(PaaS)类别、基础设施即服务(IaaS)类别、或包括混合服务的其它服务类别下提供的一个或多个服务。客户经由订阅订单可以订购由云基础设施系统1102提供的一个或多个服务。云基础设施系统1102然后执行处理，以提供客户的订阅订单中的服务。

[0182] 在一些实施例中，由云基础设施系统1102提供的服务可以包括但不限于应用服务、平台服务和基础设施服务。在一些示例中，应用服务可以由云基础设施系统经由SaaS平台提供。SaaS平台可以被配置为提供属于SaaS类别的云服务。例如，SaaS平台可以提供在集成的开发和部署平台上构建和交付按需应用套件的能力。SaaS平台可以管理和控制用于提供SaaS服务的底层软件和基础设施。通过利用由SaaS平台提供的服务，客户可以利用在云基础设施系统上执行的应用。客户可以获取应用服务，而无需客户购买分开的许可和支持。可以提供各种不同的SaaS服务。示例包括但不限于为大型组织提供的用于销售业绩管理、企业集成和业务灵活性的解决方案的服务。

[0183] 在一些实施例中，平台服务可以由云基础设施系统经由PaaS平台提供。PaaS平台可以被配置为提供属于PaaS类别的云服务。平台服务的示例可以包括但不限于使组织(诸如Oracle)能够在共享的共同体系架构上整合现有应用的服务，以及利用由平台提供的共享服务构建新应用的能力。PaaS平台可以管理和控制用于提供PaaS服务的底层软件和基础设施。客户可以获取由云基础设施系统提供的PaaS服务，而无需客户购买分开的许可和支持。平台服务的示例包括但不限于Oracle Java云服务(JCS)、Oracle数据库云服务(DBCS)以及其它。

[0184] 通过利用由PaaS平台提供的服务，客户可以采用由云基础设施系统支持的编程语言和工具，并且还可以控制所部署的服务。在一些实施例中，由云基础设施系统提供的平台服务可以包括数据库云服务、中间件云服务(例如，Oracle Fusion Middleware服务)和Java云服务。在一个实施例中，数据库云服务可以支持共享服务部署模型，其使得组织能够汇集数据库资源并且以数据库云的形式向客户提供数据库即服务。中间件云服务可以为客户提供开发和部署各种业务应用的平台，并且Java云服务可以在云基础设施系统中为客户提供部署Java应用的平台。

[0185] 可以由云基础设施系统中的IaaS平台提供各种不同的基础设施服务。基础设施服务促进对底层计算资源(诸如存储装置、网络和其它基本计算资源)的管理和控制，以便客户利用由SaaS平台和PaaS平台提供的服务。

[0186] 在某些实施例中，云基础设施系统1102还可以包括基础设施资源1130，用于提供用来向云基础设施系统的客户提供各种服务的资源。在一个实施例中，基础设施资源1130可以包括执行由PaaS平台和SaaS平台提供的服务的硬件(诸如服务器、存储装置和联网资源)的预先集成和优化的组合。

[0187] 在一些实施例中，云基础设施系统1102中的资源可以由多个用户共享并且按需动态地重新分配。此外，资源可以分配给在不同时区中的用户。例如，云基础设施系统1130可以使第一时区内的第一用户集合能够利用云基础设施系统的资源指定的小时数，然后使得能够将相同资源重新分配给位于不同时区中的另一用户集合，从而最大化资源的利用率。

[0188] 在某些实施例中，可以提供由云基础设施系统1102的不同部件或模块以及由云基

础设施系统1102提供的服务共享的多个内部共享服务1132。这些内部共享服务可以包括，但不限于，安全和身份服务、集成服务、企业储存库服务、企业管理器服务、病毒扫描和白名单服务、高可用性、备份和恢复服务、用于启用云支持的服务、电子邮件服务、通知服务、文件传输服务等。

[0189] 在某些实施例中，云基础设施系统1102可以在云基础设施系统中提供云服务(例如，SaaS、PaaS和IaaS服务)的综合管理。在一个实施例中，云管理功能可以包括用于供应、管理和跟踪由云基础设施系统1102接收到的客户的订阅的能力等。

[0190] 在一个实施例中，如图中所描绘的，云管理功能可以由诸如订单管理模块1120、订单编排模块1122、订单供应模块1124、订单管理和监视模块1126以及身份管理模块1128之类的一个或多个模块提供。这些模块可以包括一个或多个计算机和/或服务器或可以利用一个或多个计算机和/或服务器提供，该一个或多个计算机和/或服务器可以是通用计算机、专用服务器计算机、服务器场、服务器集群或任何其它适当的布置和/或组合。

[0191] 在示例性操作1134中，使用客户端设备(诸如客户端设备1104、1106或1108)的客户可以通过请求由云基础设施系统1102提供的一个或多个服务并且对由云基础设施系统1102提供的一个或多个服务的订阅下订单来与云基础设施系统1102交互。在某些实施例中，客户可以访问云用户界面(UI)(云UI 1112、云UI 1114和/或云UI 1116)并经由这些UI下订阅订单。响应于客户下订单而由云基础设施系统1102接收到的订单信息可以包括识别客户和客户打算订阅的由云基础设施系统1102提供的一个或多个服务的信息。

[0192] 在客户下订单之后，经由云UI 1112、1114和/或1116，接收订单信息。

[0193] 在操作1136处，订单存储在订单数据库1118中。订单数据库1118可以是由云基础设施系统1118操作并且与其它系统元件一起操作的若干数据库之一。

[0194] 在操作1138处，订单信息被转发到订单管理模块1120。在一些实例中，订单管理模块1120可以被配置为执行与订单相关的计费和记帐功能，诸如验证订单，并且在通过验证后，预订订单。

[0195] 在操作1140处，将关于订单的信息传送到订单编排模块1122。订单编排模块1122可以利用订单信息来协调用于由客户下的订单的服务和资源的供应。在一些实例中，订单编排模块1122可以使用订单供应模块1124的服务来编排资源的供应以支持订阅的服务。

[0196] 在某些实施例中，订单编排模块1122使得能够管理与每个订单相关联的业务流程并应用业务逻辑，以确定订单是否应当继续供应。在操作1142处，在接收到对新订阅的订单后，订单编排模块1122向订单供应模块1124发送分配资源并配置完成订阅订单所需的那些资源的请求。订单供应模块1124使得能够为客户订购的服务分配资源。订单供应模块1124提供由云基础设施系统1100提供的云服务与用于供应用于提供请求的服务的资源的物理实现层之间的抽象级别。因此，订单编排模块1122可以与实现细节隔离，实现细节诸如服务和资源实际上是实时供应的还是预先供应并且仅在请求时才被分配/指派。

[0197] 在操作1144处，一旦供应了服务和资源，就可以通过云基础设施系统1102的订单供应模块1124向客户端设备1104、1106和/或1108上的客户发送提供的服务的通知。在操作1146处，客户的订阅订单可以由订单管理和监视模块1126管理和跟踪。在一些情况下，订单管理和监视模块1126可以被配置为收集订阅订单中的服务的使用统计数据，诸如所使用的存储装置的量、传送的数据量、用户的数量、系统运行时间和系统停机时间。

[0198] 在某些实施例中，云基础设施系统1100可以包括身份管理模块1128。身份管理模块1128可以被配置为在云基础设施系统1100中提供身份服务，诸如访问管理和授权服务。在一些实施例中，身份管理模块1128可以控制关于希望利用由云基础设施系统1102提供的服务的客户的信息。这种信息可以包括认证这些客户的身份的信息和描述那些客户被授权相对于各种系统资源(例如，文件、目录、应用、通信端口、存储器段等)执行的动作的信息。身份管理模块1128还可以包括对关于每个客户的描述性信息以及关于如何和由谁来访问和修改该描述性信息的管理。

[0199] 虽然已经描述了本公开的具体实施例，但是各种修改、变更、替代构造和等同物也包含在本公开的范围内。本公开的实施例不限于在某些特定数据处理环境内的操作，而是可以在多个数据处理环境内自由操作。此外，虽然已使用特定系列的事务和步骤描述了本公开的实施例，但是，对本领域技术人员来说明显的是，本公开的范围不限于所描述的系列的事务和步骤。上述实施例的各种特征和方面可以被单独或结合使用。

[0200] 另外，虽然已经使用硬件和软件的特定组合描述了本公开的实施例，但是应该认识到的是，硬件和软件的其它组合也在本公开的范围内。本公开的实施例可以只用硬件、或只用软件、或利用硬件和软件的组合来实现。本文描述的各种进程可以在同一处理器上实现或以任何组合在不同处理器上实现。相应地，在部件或模块被描述为被配置为执行某些操作的情况下，这种配置可以例如通过设计电子电路来执行操作、通过对可编程电子电路(诸如微处理器)进行编程来执行操作、或其任意组合来实现。进程可以利用包括但不限于用于进程间通信的常规技术的各种技术来通信，并且不同的进程对可以使用不同的技术，或者同一对进程可以在不同时间使用不同的技术。

[0201] 因而，说明书和附图应当在说明性而不是限制性的意义上考虑。但是，将明显的是，在不背离权利要求中阐述的更广泛的精神和范围的情况下，可以对其进行添加、减少、删除和其它修改和改变。因此，虽然已描述了具体的公开实施例，但是这些实施例不旨在进行限制。各种修改和等同物都在以下权利要求的范围之内。

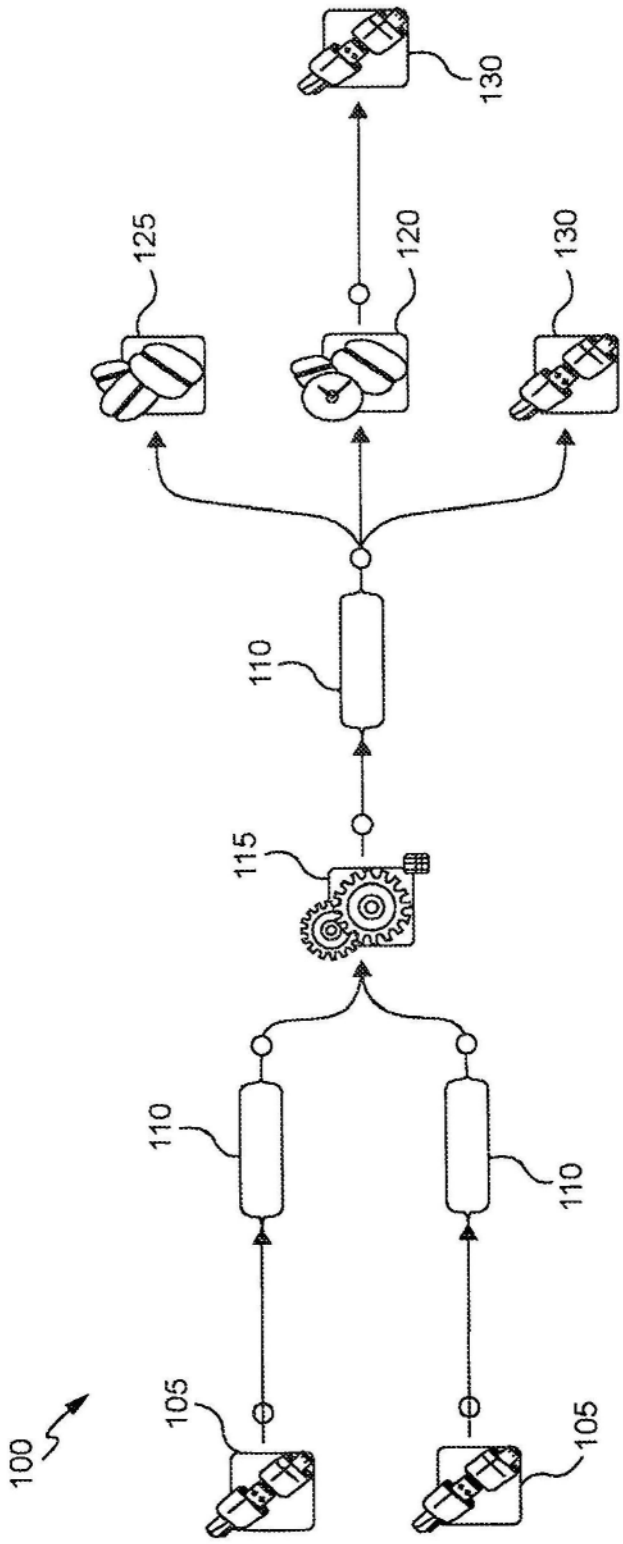


图1

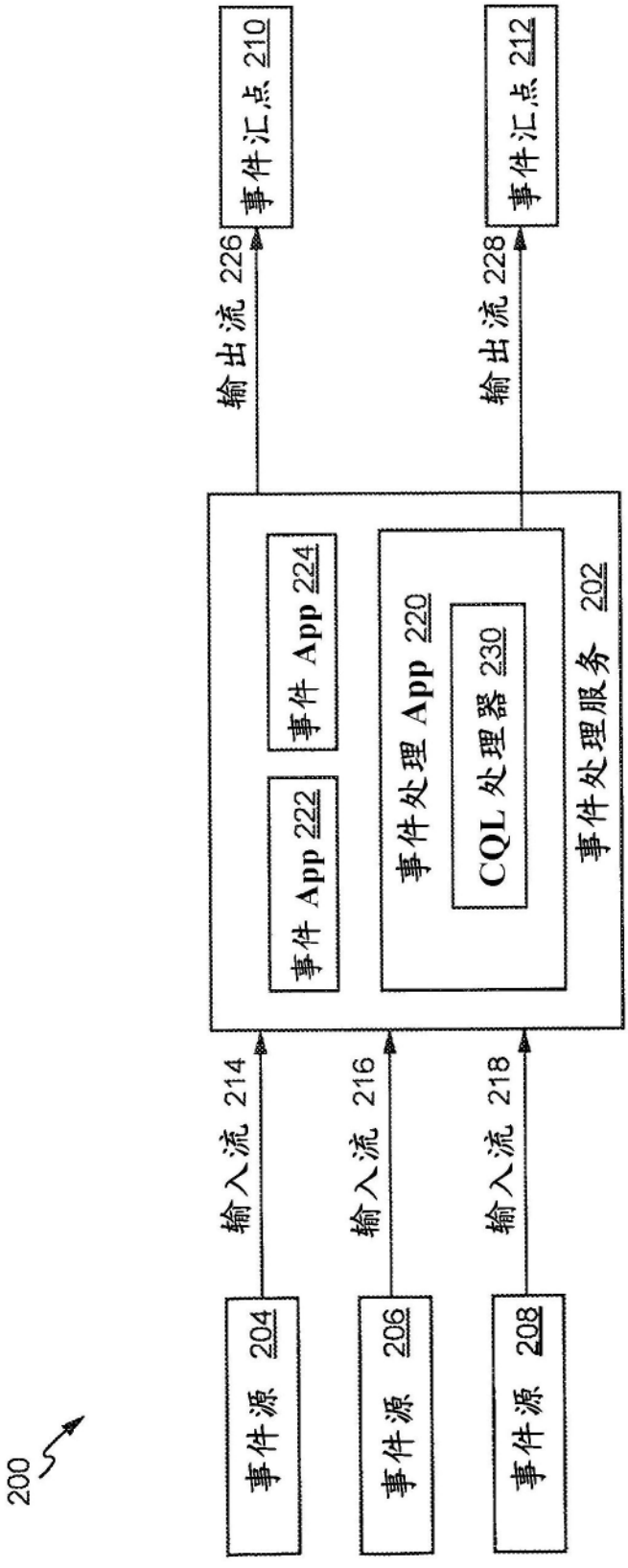

200
事件源 204
事件源 206
事件源 208
输入流 214
输入流 216
输入流 218
事件处理服务 202
事件 App 222
事件 App 224
事件处理 App 220
CQL 处理器 230
输出流 226
输出流 228
事件汇点 210
事件汇点 212

图2

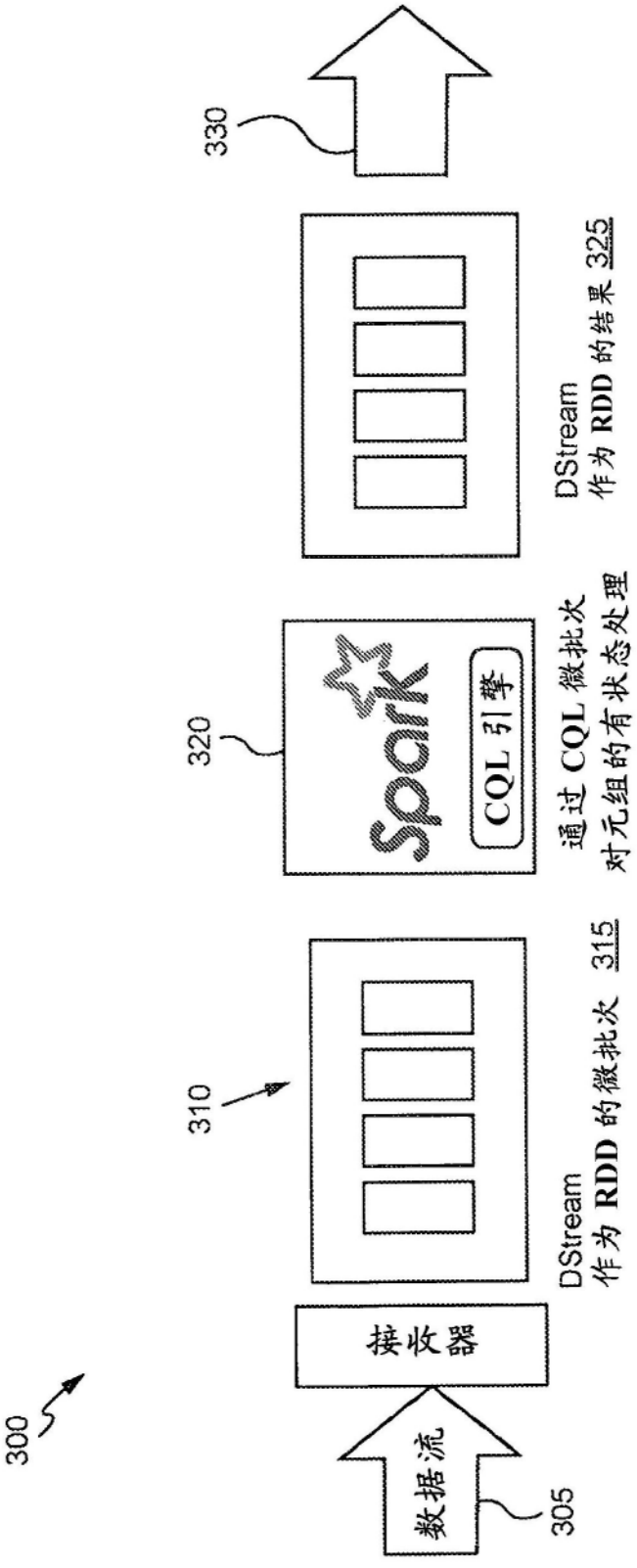

300
数据流
305
接收器
310
DStream
作为 RDD 的微批次 315
320
Spark
CQL 引擎
通过 CQL 微批次
对元组的有状态处理
DStream
作为 RDD 的结果 325
330

图3

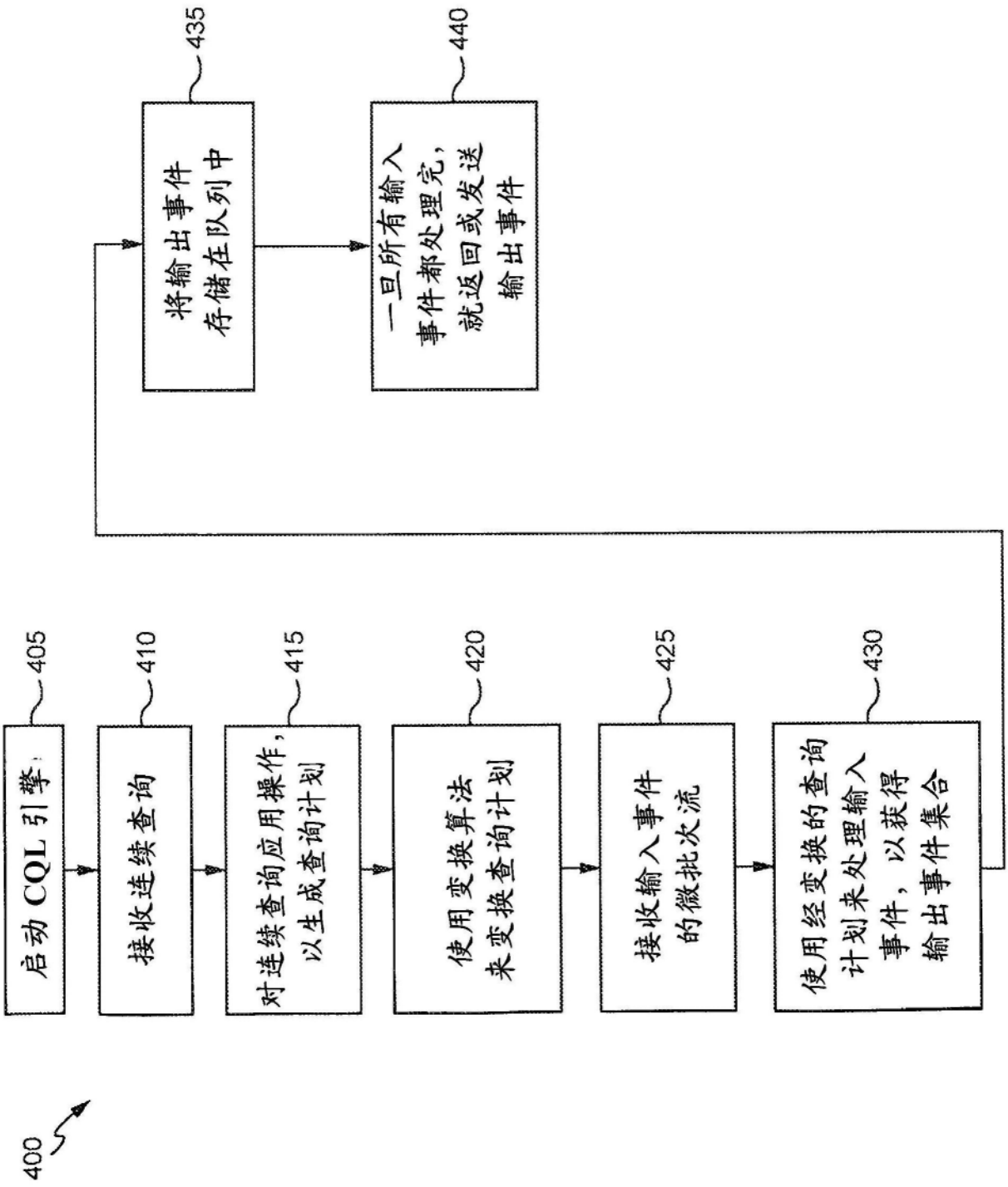

400
405
启动 CQL 引擎
410
接收连续查询
415
对连续查询应用操作，以生成查询计划
420
使用变换算法来变换查询计划
425
接收输入事件的微批次流
430
使用经变换的查询计划来处理输入事件，以获得输出事件集合
435
将输出事件存储在队列中
440
一旦所有输入事件都处理完，就返回或发送输出事件

图4

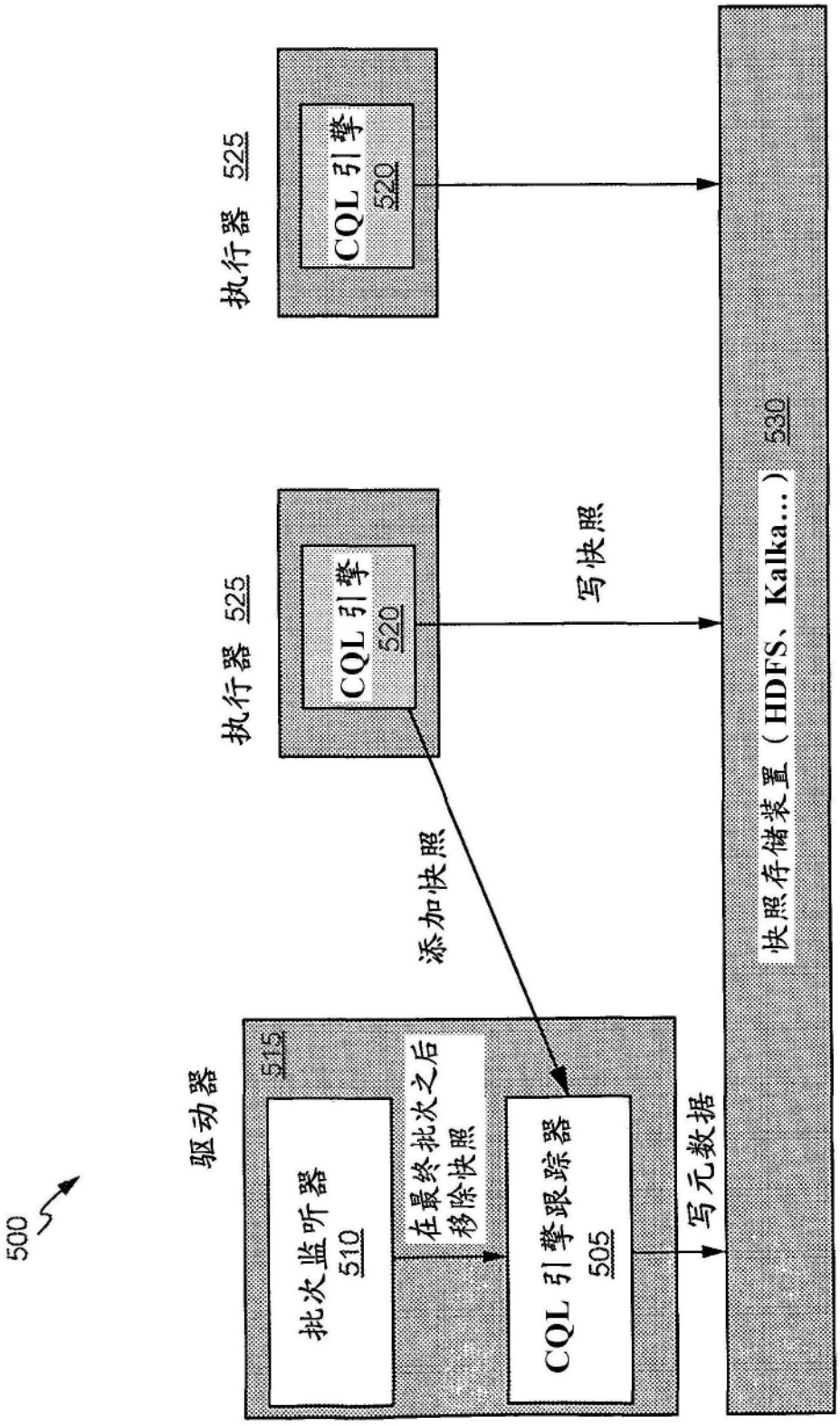

500
驱动器
515
批次监听器
510
在最终批次之后
移除快照
CQL 引擎跟踪器
505
添加快照
执行器 525
CQL 引擎
520
执行器 525
CQL 引擎
520
写快照
写元数据
快照存储装置（HDFS、Kalka...）
530

图5

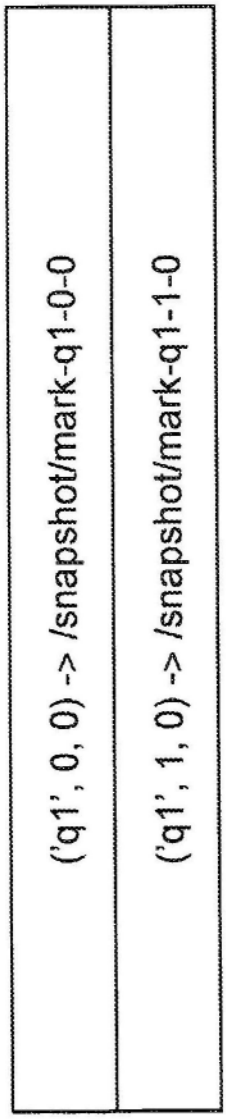


图6A

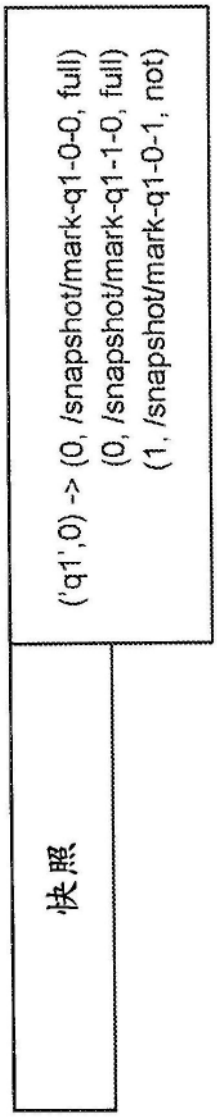

快照
('q1',0) -> (0, /snapshot/mark-q1-0-0, full)
(0, /snapshot/mark-q1-1-0, full)
(1, /snapshot/mark-q1-0-1, not)

图6B

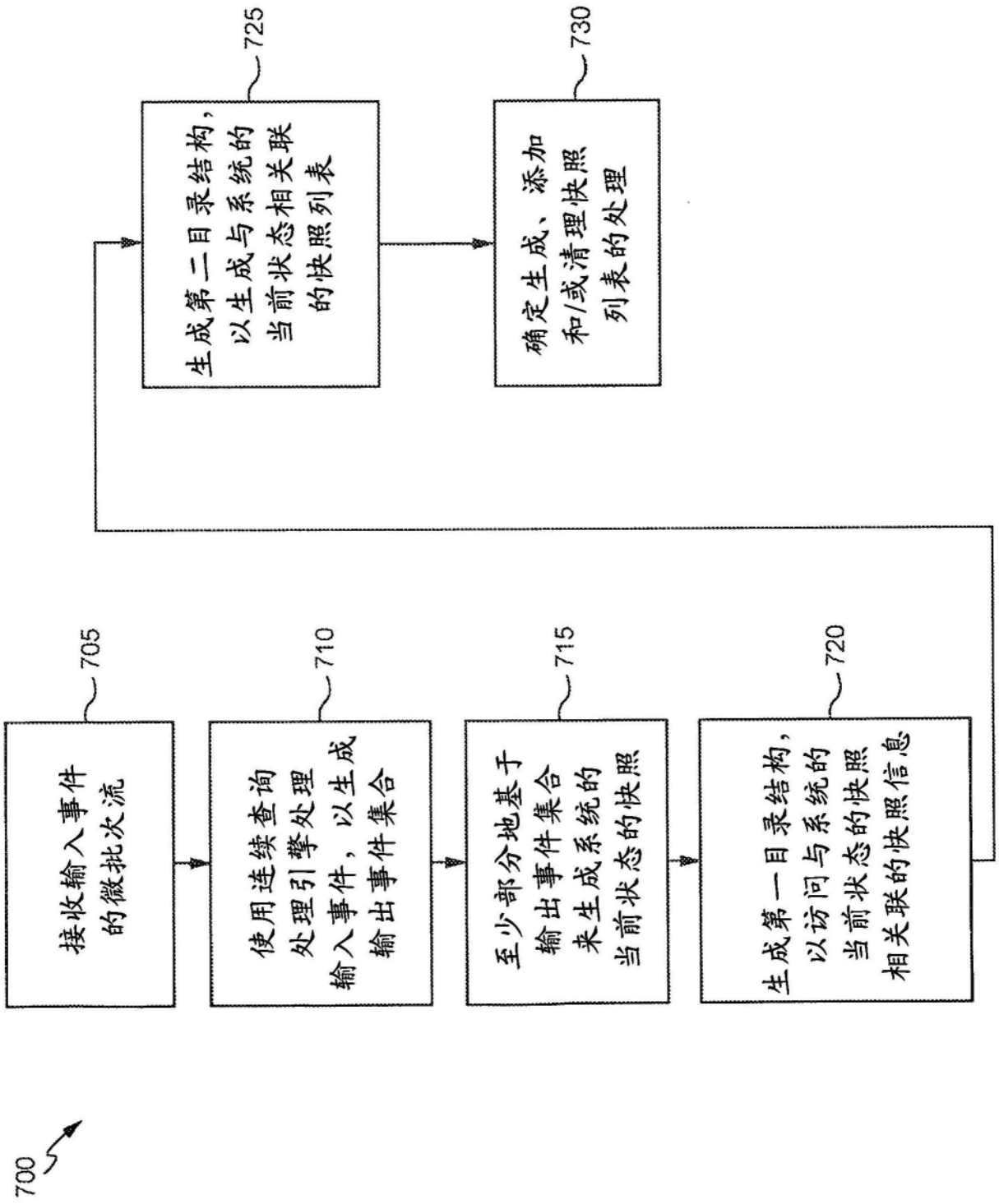

700
705
接收输入事件的微批次流
710
使用连续查询处理引擎处理输入事件，以生成输出事件集合
715
至少部分地基于输出事件集合来生成系统的当前状态的快照
720
生成第一目录结构，以访问与系统的当前状态的快照相关联的快照信息
725
生成第二目录结构，以生成与系统的当前状态相关联的快照列表
730
确定生成、添加和/或清理快照列表的处理

图7

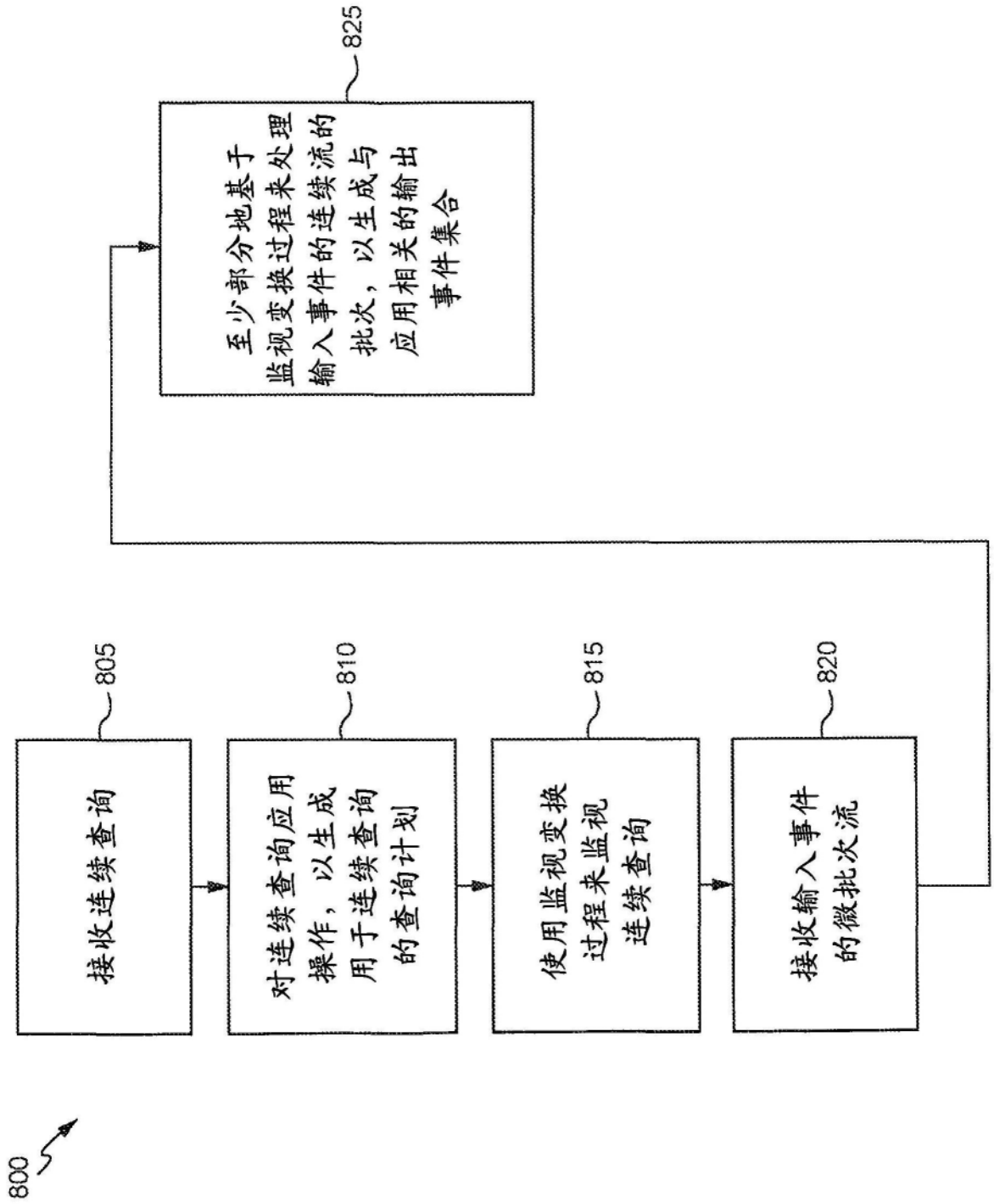

800
805
接收连续查询
810
对连续查询应用操作，以生成用于连续查询的查询计划
815
使用监视变换过程来监视连续查询
820
接收输入事件的微批次流
825
至少部分地基于监视变换过程来处理输入事件的连续流的批次，以生成与应用相关的输出事件集合

图8

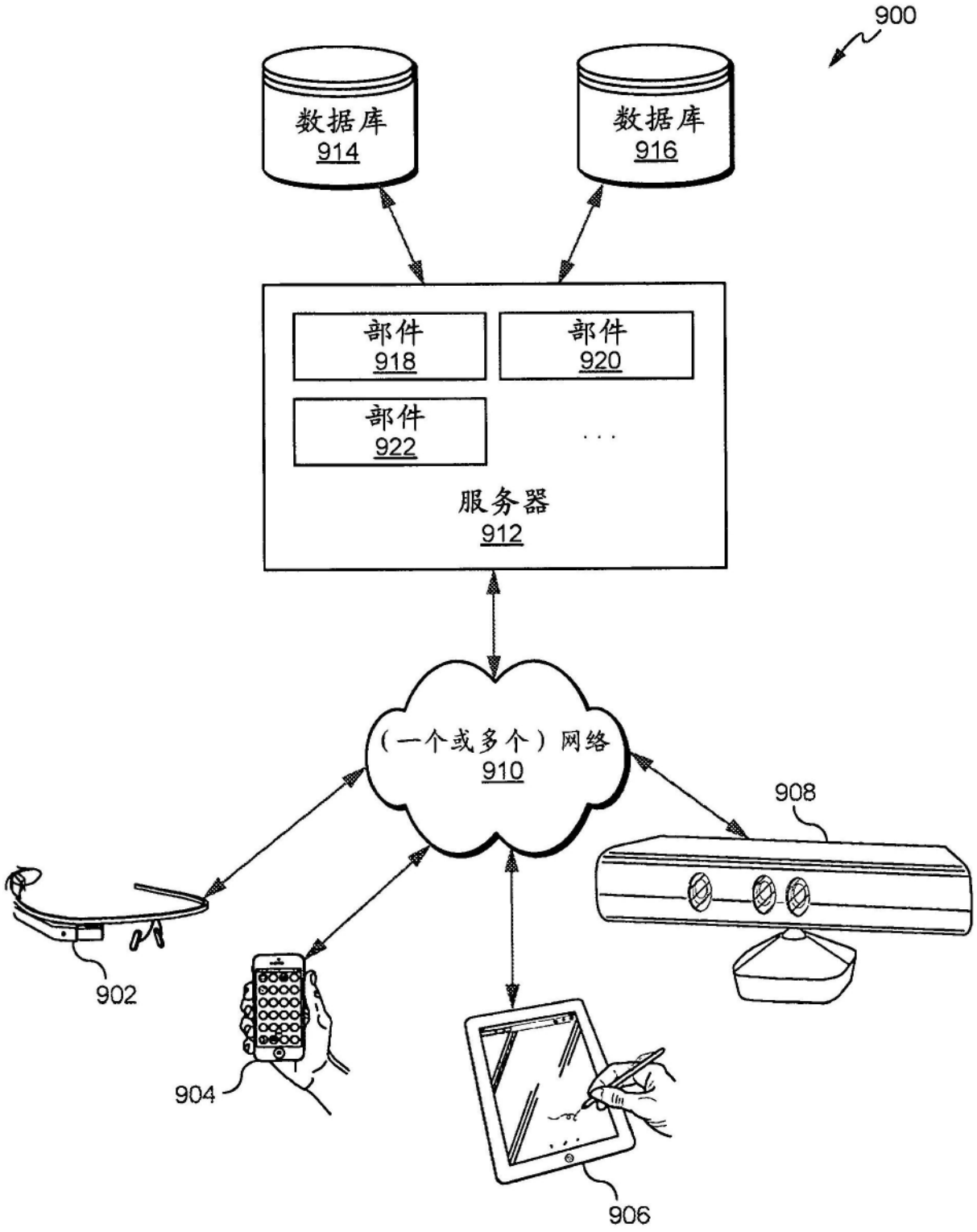

900
数据库
914
数据库
916
部件
918
部件
920
部件
922
. . .
服务器
912
（一个或多个）网络
910
908
902
904
906

图9

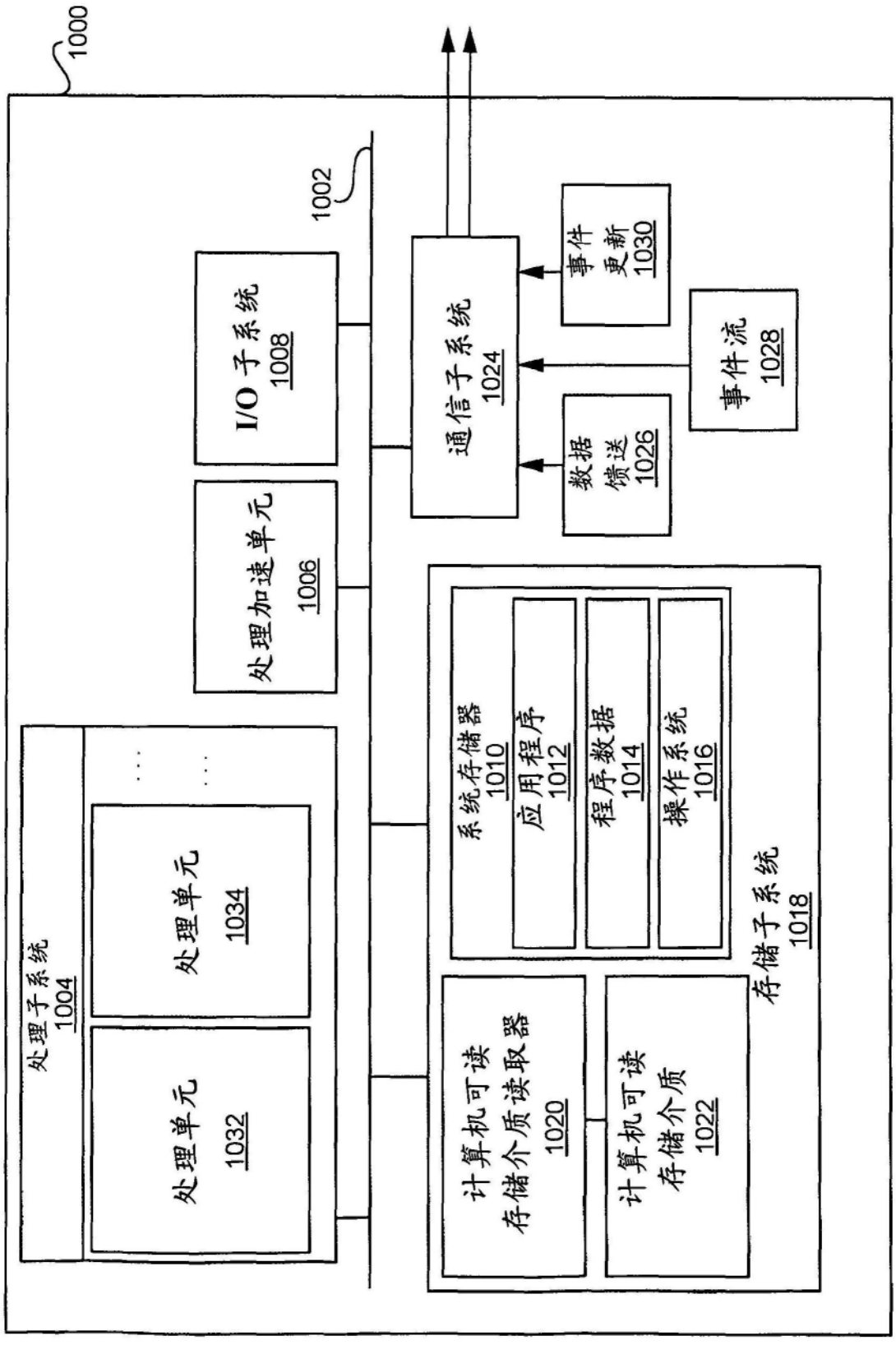

1000
1002
处理子系统
1004
处理单元
1032
处理单元
1034
...
处理加速单元
1006
I/O 子系统
1008
通信子系统
1024
数据馈送
1026
事件更新
1030
事件流
1028
存储子系统
1018
系统存储器
1010
应用程序
1012
程序数据
1014
操作系统
1016
计算机可读
存储介质读取器
1020
计算机可读
存储介质
1022

图10

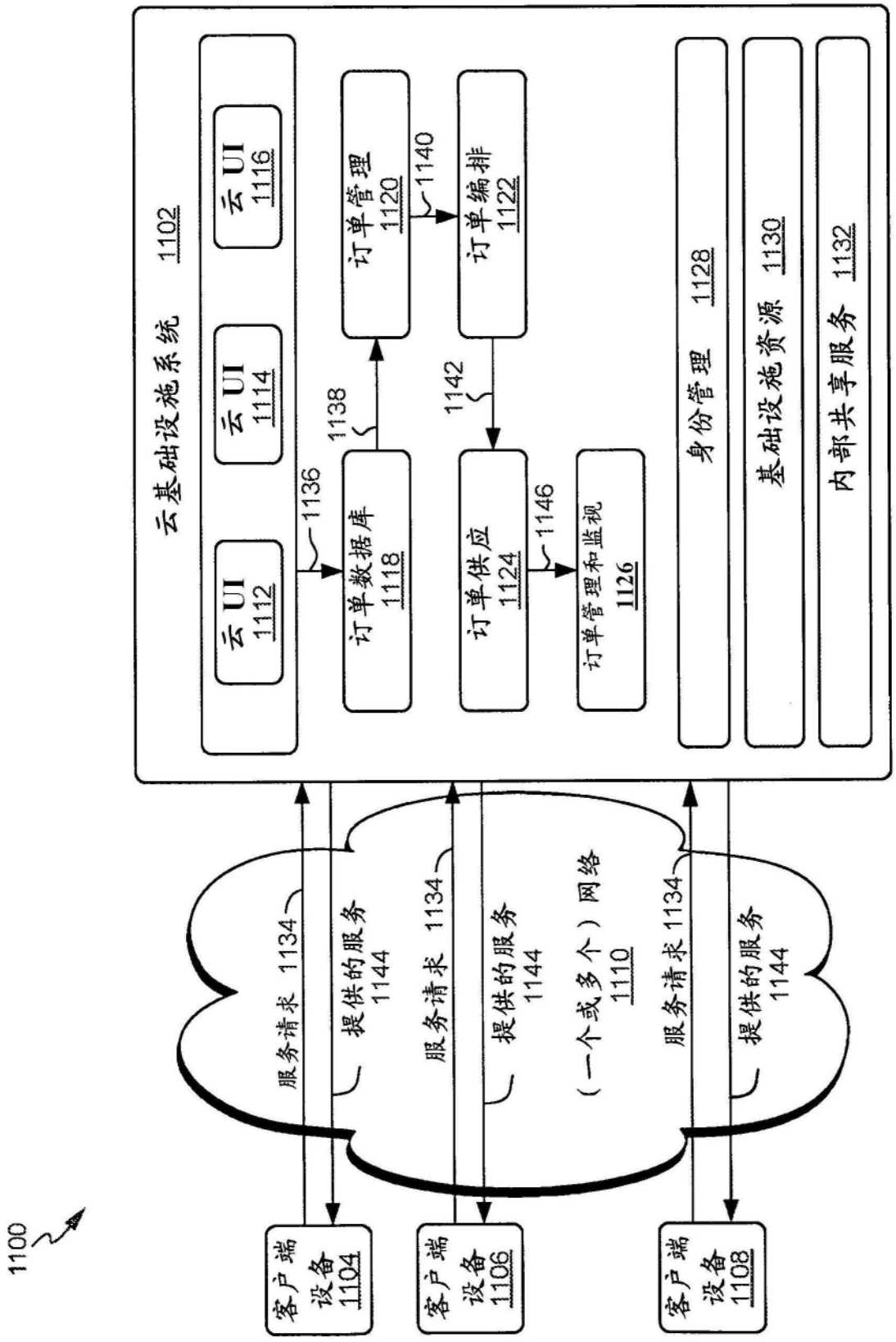

1100
客户端设备 1104
客户端设备 1106
客户端设备 1108
（一个或多个）网络 1110
服务请求 1134
提供的服务 1144
服务请求 1134
提供的服务 1144
服务请求 1134
提供的服务 1144
云基础设施系统 1102
云 UI 1112
云 UI 1114
云 UI 1116
订单数据库 1118
订单管理 1120
订单编排 1122
订单供应 1124
订单管理和监视 1126
身份管理 1128
基础设施资源 1130
内部共享服务 1132
1136
1138
1140
1142
1146

图11