(19)中华人民共和国国家知识产权局

(12)发明专利

(10)授权公告号 CN 108845794 B
(45)授权公告日 2021.09.10

(21)申请号 201810466470.5
(22)申请日 2018.05.16
(65)同一申请的已公布的文献号
申请公布号 CN 108845794 A
(43)申请公布日 2018.11.20
(73)专利权人 浪潮软件科技有限公司
地址 250100 山东省济南市高新区浪潮路1036号浪潮科技园
(72)发明人 宫若瑜 程林 杨培强
(74)专利代理机构 济南信达专利事务所有限公司 37100
代理人 李世喆
(51)Int.Cl.
G06F 8/30(2018.01)

审查员 雷雯静

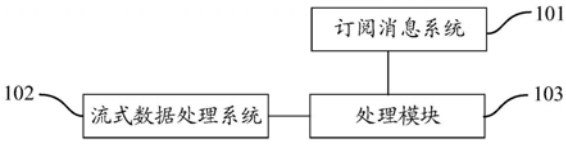
权利要求书2页 说明书7页 附图3页

(54)发明名称

一种流式运算系统、方法、可读介质和存储控制器

(57)摘要

本发明提供了一种流式运算框架、方法、可读介质和存储控制器，该流式运算框架包括：订阅消息系统、流式数据处理系统和处理模块；其中，所述处理模块，用于将所述订阅消息系统与所述流式数据处理系统进行封装，生成数据转换工具；当接收到所述订阅消息系统发送的至少一个数据流时，利用所述数据转换工具对所述至少一个数据流的格式进行标准化处理，并将标准化处理后的所述至少一个数据流发送给所述流式数据处理系统；所述订阅消息系统，用于从外部的至少一个数据源中抽取至少一个数据流，并将所述至少一个数据流发送给所述处理模块；所述流式数据处理系统，用于对标准化处理后的所述至少一个数据流进行业务处理。本方案能提高数据处理效率。



CN 108845794 B

1.一种流式运算系统,其特征在于,包括:订阅消息系统、流式数据处理系统和处理模块;其中所述处理模块,用于将所述订阅消息系统与所述流式数据处理系统进行封装,生成数据转换工具;当接收到所述订阅消息系统发送的至少一个数据流时,利用所述数据转换工具对所述至少一个数据流的格式进行标准化处理,并将标准化处理后的所述至少一个数据流发送给所述流式数据处理系统;

所述流式数据处理系统,用于对标准化处理后的所述至少一个数据流进行业务处理;

所述流式数据处理系统,用于从各个所述数据流中选择当前数据流,为所述当前数据流设置已处理标签;对所述当前数据流进行业务处理,当在业务处理过程中出现数据异常时,确定所述数据异常出现的次数是否大于预设阈值,如果大于预设阈值,将当前数据流标记为异常数据流;如果小于预设阈值,删除所述已处理标签,并从未处理的各个数据流中选择出当前数据流,直至各个数据流均被处理;

所述流式数据处理系统,进一步用于将业务处理后的所述至少一个数据流发送给所述处理模块;

所述处理模块,进一步用于根据外部输入的调用指令,确定每一个所述数据流对应的数据输出向;根据所述数据输出向对应的数据格式,分别对各个所述数据流进行格式转换,并将转换后的所述数据流发送给所述订阅消息系统;

所述订阅消息系统,用于从外部的至少一个数据源中抽取至少一个数据流,并将所述至少一个数据流发送给所述处理模块;

所述订阅消息系统,用于根据预先设置的订阅模式,将格式转换后的各个所述数据流分别输出给对应的数据输出向。

2.根据权利要求1所述的流式运算系统,其特征在于,

所述订阅消息系统包括:Kafka;

和/或,

所述流式数据处理系统包括:Storm Trident。

3.一种流式运算方法,其特征在于,包括:

将订阅消息系统与流式数据处理系统进行封装,生成数据转换工具;

还包括:

从外部的至少一个数据源中抽取至少一个数据流;

利用所述数据转换工具对所述至少一个数据流的格式进行标准化处理;

对标准化处理后的所述至少一个数据流进行业务处理;

在所述对标准化处理后的所述至少一个数据流进行业务处理之后,进一步包括:

根据外部输入的调用指令,确定每一个所述数据流对应的数据输出向;

根据所述数据输出向对应的数据格式,分别对各个所述数据流进行格式转换;

根据预先设置的订阅模式,将格式转换后的各个所述数据流分别输出给对应的数据输出向;

所述对标准化处理后的所述至少一个数据流进行业务处理,包括:

循环执行以下步骤,直至各个数据流均被处理:

A1:从未处理的各个数据流中选择当前数据流,并为所述当前数据流设置已处理标签;

A2:对所述当前数据流进行业务处理;

A3:当在业务处理过程中出现数据异常时,确定所述数据异常出现的次数是否大于预设阈值,如果大于预设阈值,将当前数据流标记为异常数据流;如果小于预设阈值,删除所述已处理标签,删除所述已处理标签,并执行A1。

4.一种可读介质,包括执行指令,当存储控制器的处理器执行所述执行指令时,所述存储控制器执行上述权利要求3所述的方法。

5.一种存储控制器,包括:处理器、存储器和总线;所述存储器用于存储执行指令,所述处理器与所述存储器通过所述总线连接,当所述存储控制器运行时,所述处理器执行所述存储器存储的所述执行指令,以使所述存储控制器执行上述权利要求3所述的方法。

一种流式运算系统、方法、可读介质和存储控制器

技术领域

[0001] 本发明涉及计算机技术领域，特别涉及一种流式运算系统、方法、可读介质和存储控制器。

背景技术

[0002] Trident是基于Storm进行实时流处理的高级抽象，提供了对实时流的聚集、投影和过滤等功能，从而大大减少了开发Storm程序的工作量。随着大数据技术的飞速发展，Trident可用于处理不同数据源的数据流。

[0003] Trident在处理不同数据源的数据流时，根据不同数据流的数据来源，确定不同的数据格式，并进一步根据不同的数据格式，开发不同的数据流处理流程，然后利用对应的处理流程对不同数据源的数据流进行处理。

[0004] 在此过程中，由于Trident在处理数据流时，需确定不同数据源的数据流的数据格式，并根据数据格式开发对应的处理流程，导致数据处理效率较低。

发明内容

[0005] 本发明实施例提供了一种流式运算系统、方法、可读介质和存储控制器，能提高数据处理效率。

[0006] 第一方面，本发明实施例提供了一种流式运算系统，包括：订阅消息系统、流式数据处理系统和处理模块；其中，

[0007] 所述处理模块，用于将所述订阅消息系统与所述流式数据处理系统进行封装，生成数据转换工具；当接收到所述订阅消息系统发送的至少一个数据流时，利用所述数据转换工具对所述至少一个数据流的格式进行标准化处理，并将标准化处理后的所述至少一个数据流发送给所述流式数据处理系统；

[0008] 所述订阅消息系统，用于从外部的至少一个数据源中抽取至少一个数据流，并将所述至少一个数据流发送给所述处理模块；

[0009] 所述流式数据处理系统，用于对标准化处理后的所述至少一个数据流进行业务处理。

[0010] 优选地，

[0011] 所述流式数据处理系统，进一步用于将业务处理后的所述至少一个数据流发送给所述处理模块；

[0012] 所述处理模块，进一步用于根据外部输入的调用指令，确定每一个所述数据流对应的数据输出向；根据所述数据输出向对应的数据格式，分别对各个所述数据流进行格式转换，并将转换后的所述数据流发送给所述订阅消息系统；

[0013] 所述订阅消息系统，用于根据预先设置的订阅模式，将格式转换后的各个所述数据流分别输出给对应的数据输出向。

[0014] 优选地，

[0015] 所述流式数据处理系统，用于从各个所述数据流中选择当前数据流，为所述当前数据流设置已处理标签；对所述当前数据流进行业务处理，当在所述业务处理过程中出现数据异常时，确定所述数据异常出现的次数是否大于预设阈值，如果是，将所述当前数据流标记为异常数据流；否则，删除所述已处理标签，并从未处理的各个数据流中选择出当前数据流，直至各个所述数据流均被处理。

[0016] 优选地，

[0017] 所述订阅消息系统包括：Kafka；

[0018] 优选地，

[0019] 所述流式数据处理系统包括：Storm Trident。

[0020] 第二方面，本发明实施例提供了一种流式运算方法，包括：

[0021] 将订阅消息系统与流式数据处理系统进行封装，生成数据转换工具；

[0022] 还包括：

[0023] 从外部的至少一个数据源中抽取至少一个数据流；

[0024] 利用所述数据转换工具对所述至少一个数据流的格式进行标准化处理；

[0025] 对标准化处理后的所述至少一个数据流进行业务处理。

[0026] 优选地，

[0027] 在所述对标准化处理后的所述至少一个数据流进行业务处理之后，进一步包括：

[0028] 根据外部输入的调用指令，确定每一个所述数据流对应的数据输出向；

[0029] 根据所述数据输出向对应的数据格式，分别对各个所述数据流进行格式转换；

[0030] 根据预先设置的订阅模式，将格式转换后的各个所述数据流分别输出给对应的数据输出向。

[0031] 优选地，

[0032] 所述对标准化处理后的所述至少一个数据流进行业务处理，包括：

[0033] 循环执行以下步骤，直至各个所述数据流均被处理：

[0034] A1：从未处理的各个所述数据流中选择当前数据流，并为所述当前数据流设置已处理标签；

[0035] A2：对所述当前数据流进行业务处理；

[0036] A3：当在所述业务处理过程中出现数据异常时，确定所述数据异常出现的次数是否大于预设阈值，如果是，将所述当前数据流标记为异常数据流；否则，删除所述已处理标签，并执行A1。

[0037] 第三方面，本发明实施例提供了一种可读介质，包括执行指令，当存储控制器的处理器执行所述执行指令时，所述存储控制器执行本发明上述任一实施例提供的方法。

[0038] 第四方面，本发明实施例提供了一种存储控制器，包括：处理器、存储器和总线；所述存储器用于存储执行指令，所述处理器与所述存储器通过所述总线连接，当所述存储控制器运行时，所述处理器执行所述存储器存储的所述执行指令，以使所述存储控制器执行本发明上述任一实施例提供的方法。

[0039] 本发明实施例提供了一种流式运算系统、方法、可读介质和存储控制器，首先将订阅消息系统(Kafka)与流式数据处理系统(Storm Trident)进行封装，生成数据转换工具(KafkaTrident)。然后利用Kafka作为消息中间件，获取外部不同数据源的数据流，然后

KafkaTrident对各个数据流的格式进行标准化处理，并将处理后的数据流发送给Storm Trident，Storm Trident即可直接对标准化处理后的各个数据流进行业务处理，而无需根据不同数据流的数据格式，开发相应的处理流程，再利用开发出的处理流程处理相应数据格式的数据流，由此提高了数据处理效率。

附图说明

[0040] 为了更清楚地说明本发明实施例或现有技术中的技术方案，下面将对实施例或现有技术描述中所需要使用的附图作简单地介绍，显而易见地，下面描述中的附图是本发明的一些实施例，对于本领域普通技术人员来讲，在不付出创造性劳动的前提下，还可以根据这些附图获得其他的附图。

[0041] 图1是本发明一个实施例提供的一种流式运算系统的结构示意图；

[0042] 图2是本发明一个实施例提供的一种流式运算方法的流程图；

[0043] 图3是本发明另一个实施例提供的一种流式运算方法的流程图；

[0044] 图4是本发明又一个实施例提供的一种流式运算方法的流程图。

具体实施方式

[0045] 为使本发明实施例的目的、技术方案和优点更加清楚，下面将结合本发明实施例中的附图，对本发明实施例中的技术方案进行清楚、完整地描述，显然，所描述的实施例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的前提下所获得的所有其他实施例，都属于本发明保护的范围。

[0046] 如图1所示，本发明实施例提供了一种流式运算系统，该流式运算系统可以包括：订阅消息系统101、流式数据处理系统102和处理模块103；其中，

[0047] 所述处理模块103，用于将所述订阅消息系统101与所述流式数据处理系统102进行封装，生成数据转换工具；当接收到所述订阅消息系统101发送的至少一个数据流时，利用所述数据转换工具对所述至少一个数据流的格式进行标准化处理，并将标准化处理后的所述至少一个数据流发送给所述流式数据处理系统102；

[0048] 所述订阅消息系统101，用于从外部的至少一个数据源中抽取至少一个数据流，并将所述至少一个数据流发送给所述处理模块103；

[0049] 所述流式数据处理系统102，用于对标准化处理后的所述至少一个数据流进行业务处理。

[0050] 上述实施例中，首先将订阅消息系统(Kafka)与流式数据处理系统(Storm Trident)进行封装，生成数据转换工具，为了便于描述，下文将数据转换工具称为KafkaTrident。然后利用Kafka作为消息中间件，获取外部不同数据源的数据流，然后KafkaTrident对各个数据流的格式进行标准化处理，并将处理后的数据流发送给Storm Trident，Storm Trident即可直接对标准化处理后的各个数据流进行业务处理，而无需根据不同数据流的数据格式，开发相应的处理流程，再利用开发出的处理流程处理相应数据格式的数据流，由此提高了数据处理效率。

[0051] 本发明一个实施例中，所述流式数据处理系统102，进一步用于将业务处理后的所述至少一个数据流发送给所述处理模块101；

[0052] 所述处理模块101，进一步用于根据外部输入的调用指令，确定每一个所述数据流对应的数据输出向；根据所述数据输出向对应的数据格式，分别对各个所述数据流进行格式转换，并将转换后的所述数据流发送给所述订阅消息系统101；

[0053] 所述订阅消息系统101，用于根据预先设置的订阅模式，将格式转换后的各个所述数据流分别输出给对应的数据输出向。

[0054] 在这里，当Storm Trident对数据流进行业务处理后，将其发送给KafkaTrident，KafkaTrident可根据外部的调用指令，确定各个数据流对应的数据输出向，例如，可根据调用指令确定出数据输出向为文件系统A和存储系统B，则可对数据流进行相应的格式转换，并将转换后的数据流发送给Kafka。由于Kafka本身具有订阅模式，使得Kafka可将转换后的数据流输出给对应的数据输出向。由此，将业务处理后的各个数据流输出给对应的数据输出向时，也对数据进行相应的格式转换，使得接收各个数据流的数据输出向可直接对数据流进行读取或存储等操作，即数据流的业务处理结果可被其他业务程序(文件系统A或存储系统B)直接取用，省去了数据库的交互时间，从而进一步提高了数据处理效率。

[0055] 本发明一个实施例中，所述流式数据处理系统102，用于从各个所述数据流中选择当前数据流，为所述当前数据流设置已处理标签；对所述当前数据流进行业务处理，当在所述业务处理过程中出现数据异常时，确定所述数据异常出现的次数是否大于预设阈值，如果是，将所述当前数据流标记为异常数据流；否则，删除所述已处理标签，并从未处理的各个数据流中选择出当前数据流，直至各个所述数据流均被处理。

[0056] 当Storm Trident对数据流进行业务处理时，是从KafkaTrident中逐个调取标准化处理后的数据流，每调取一个数据流，则为调取的当前数据流设置已处理标签，然后对当前数据流进行业务处理，当在业务处理过程中出现数据异常时，确定该当前数据流出现异常的次数是否大于预设阈值，如果是，则将该当前数据流作为异常数据流，后期可不再对该异常数据流进行处理，同时可发出异常提示，以提示用户进行相应的异常恢复操作。若当前数据流出现异常的次数不大于预设阈值，则将记录的该当前数据流的异常次数加1，并删除当前数据流的已处理标签，将其发回KafkaTrident，然后再从KafkaTrident中调取另一个标准化处理后的数据流作为当前数据流，直至KafkaTrident中的各个数据流均被处理，由此，当业务处理过程出现数据异常时，将相应数据流返回KafkaTrident重新处理，或者将其标记为异常数据流，从而保证了数据处理的整体事务性。

[0057] 如图2所示，本发明一个实施例提供了一种流式运算方法，包括：

[0058] 步骤201：将订阅消息系统与流式数据处理系统进行封装，生成数据转换工具；

[0059] 步骤202：从外部的至少一个数据源中抽取至少一个数据流；

[0060] 步骤203：利用所述数据转换工具对所述至少一个数据流的格式进行标准化处理；

[0061] 步骤204：对标准化处理后的所述至少一个数据流进行业务处理。

[0062] 本发明一个实施例中，在步骤204之后，进一步包括：

[0063] 根据外部输入的调用指令，确定每一个所述数据流对应的数据输出向；

[0064] 根据所述数据输出向对应的数据格式，分别对各个所述数据流进行格式转换；

[0065] 根据预先设置的订阅模式，将格式转换后的各个所述数据流分别输出给对应的数据输出向。

[0066] 本发明一个实施例中，步骤204的具体实施方式，可以包括：

[0067] 循环执行以下步骤,直至各个所述数据流均被处理:

[0068] A1:从未处理的各个所述数据流中选择当前数据流,并为所述当前数据流设置已处理标签;

[0069] A2:对所述当前数据流进行业务处理;

[0070] A3:当在所述业务处理过程中出现数据异常时,确定所述数据异常出现的次数是否大于预设阈值,如果是,将所述当前数据流标记为异常数据流;否则,删除所述已处理标签,并执行A1。

[0071] 上述方法中的各步骤之间的信息交互、执行过程等内容,由于与本发明系统实施例基于同一构思,具体内容可参见本发明系统实施例中的叙述,此处不再赘述。

[0072] 下面以订阅消息系统为Kafka,流式数据处理系统为Storm Trident为例,对本发明实施例提供的流式运算方法进行详细说明,如图3所示,该方法可以包括以下步骤:

[0073] 步骤301:将Kafka和Storm Trident交互的代码进行封装,生成KafkaTrident。

[0074] 步骤302:利用Kafka从外部的至少一个数据源中抽取至少一个数据流。

[0075] 步骤303:利用KafkaTrident将所述至少一个数据流的格式进行标准化处理。

[0076] 以上步骤至少可通过以下程序语言实现:

[0077] Stream source zsJks=Kafka FormatedStream.getKafkaStream(topology,“source_zsjks”,

[0078] “topic_trident_ZsJks”,new Fields(ZsJks.getFieldList()),“topic_zsJks_consumer_transactionaltridenttotal”);

[0079] /**

[0080] * @author gongruoyu

[0081] * @param topology TridentTopology

[0082] * @paramtNameTridentTopologyName

[0083] * @param topic KafkaTopicName

[0084] * @paramfields FormatTridentFields

[0085] * @paramgroupIDKafkaGroupID

[0086] * @ return TridentKafkaStream

[0087] */

[0088] Public static Stream getKafkaSteam(TridentTopologytopology,StringtName,Stringtopic,Fieldsfields,stringgroupId)

[0089] 步骤304:利用Storm Trident从KafkaTrident中选择标准化处理后的数据流作为当前数据流,并为所述当前数据流设置已处理标签。

[0090] 步骤305:利用Storm Trident对当前数据流进行业务处理。

[0091] 步骤306:当在业务处理处理过程中出现数据异常时,判断所述数据异常出现的次数是否大于预设阈值,如果是,执行步骤307,否则执行步骤308。

[0092] 步骤307:将所述当前数据流标记为异常数据流,并执行步骤309。

[0093] 步骤308:删除所述已处理标签。

[0094] 步骤309:判断KafkaTrident中是否存在未处理数据流,如果是,执行步骤304,否则执行步骤310。

[0095] 步骤310:Storm Trident将业务处理后的数据流发送给KafkaTrident。

[0096] 步骤311:KafkaTrident根据外部输入的调用指令,确定每一个所述数据流对应的数据输出向,并根据所述数据输出向对应的数据格式,分别对各个所述数据流进行格式转换,将转换后的数据流发送给Kafka。

[0097] 上述步骤至少可以通过以下程序语言实现:

[0098] Steam.partitionPersist(new KafkaFactory("Result_topic"),tmp_zs_waz_pl_f,newMyKafkaProducer());

[0099] 步骤312:Kafka根据预先设置的订阅模式,将格式转换后的各个所述数据流分别输出给对应的数据输出向。

[0100] 综上所述,利用Kafka作为消息中间件,作为StormTrident的消息入/出口,减少流式运算中的数据库交互,同时做到了数据库操作和流式计算解耦,并且因为Kafka的存在,可以极大的提高数据并行度。Kafka的加入,使得不同数据源的数据只要按照统一标准存入Kafka中,就可以被已经设计好的StormTrident程序处理,不会因数据源不同,导致StormTrident程序需要修改。同时,也避免了StormTrident针对不同数据源的数据,要进一步开发以转化为可处理的流的过程。这样同时又将数据抽取同StormTrident程序本身分离开,极大提高了程序构建及运行速度。而Kafka本身的订阅模式,又使得Kafka内的数据可以被多个不同的StormTrident程序或者其他业务程序同时使用,极大的提高了数据使用效率。其中,数据流在处理过程中的流向示意图如图4所示,其中的DB、文件系统以及其他存储或业务系统为不同的数据源和数据输出向,数据流按顺序1-6被处理。可以理解的是,StormTrident在处理数据的过程中全部数据都是按照流的形式在程序中流转,因此,默认输入输出结果都是流的形式。

[0101] 本发明实施例还提供了一种可读介质,包括执行指令,当存储控制器的处理器执行所述执行指令时,所述存储控制器执行本发明上述任一实施例提供的方法。

[0102] 本发明实施例还提供了一种存储控制器,包括:处理器、存储器和总线;所述存储器用于存储执行指令,所述处理器与所述存储器通过所述总线连接,当所述存储控制器运行时,所述处理器执行所述存储器存储的所述执行指令,以使所述存储控制器执行本发明上述任一实施例提供的方法。

[0103] 综上所述,本发明以上各个实施例至少具有如下有益效果:

[0104] 1、在本发明实施例中,首先将订阅消息系统(Kafka)与流式数据处理系统(Storm Trident)进行封装,生成数据转换工具(Kafka Trident)。然后利用Kafka作为消息中间件,获取外部不同数据源的数据流,然后Kafka Trident对各个数据流的格式进行标准化处理,并将处理后的数据流发送给Storm Trident,Storm Trident即可直接对标准化处理后的各个数据流进行业务处理,而无需根据不同数据流的数据格式,开发相应的处理流程,再利用开发出的处理流程处理相应数据格式的数据流,由此提高了数据处理效率。

[0105] 2、在本发明实施例中,将业务处理后的各个数据流输出给对应的数据输出向时,也对数据进行相应的格式转换,使得数据流的业务处理结果可被其他业务程序直接取用,省去了数据库的交互时间,从而进一步提高了数据处理效率。

[0106] 3、在本发明实施例中,Storm Trident对数据流进行业务处理时,当业务处理过程出现数据异常时,将相应数据流返回Kafka Trident重新处理,或者将其标记为异常数据

流，从而保证了数据处理的整体事务性。

[0107] 4、在本发明实施例中，将数据抽取同StormTrident程序本身分离开，极大提高了程序构建及运行速度。而Kafka本身的订阅模式，又使得Kafka内的数据可以被多个不同的StormTrident程序或者其他业务程序同时使用，极大的提高了数据使用效率。

[0108] 需要说明的是，在本文中，诸如第一和第二之类的关系术语仅仅用来将一个实体或者操作与另一个实体或操作区分开来，而不一定要求或者暗示这些实体或操作之间存在任何这种实际的关系或者顺序。而且，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、物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、物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、物品或者设备中还存在另外的相同因素。

[0109] 本领域普通技术人员可以理解：实现上述方法实施例的全部或部分步骤可以通过程序指令相关的硬件来完成，前述的程序可以存储在计算机可读取的存储介质中，该程序在执行时，执行包括上述方法实施例的步骤；而前述的存储介质包括：ROM、RAM、磁碟或者光盘等各种可以存储程序代码的介质中。

[0110] 最后需要说明的是：以上所述仅为本发明的较佳实施例，仅用于说明本发明的技术方案，并非用于限定本发明的保护范围。凡在本发明的精神和原则之内所做的任何修改、等同替换、改进等，均包含在本发明的保护范围内。

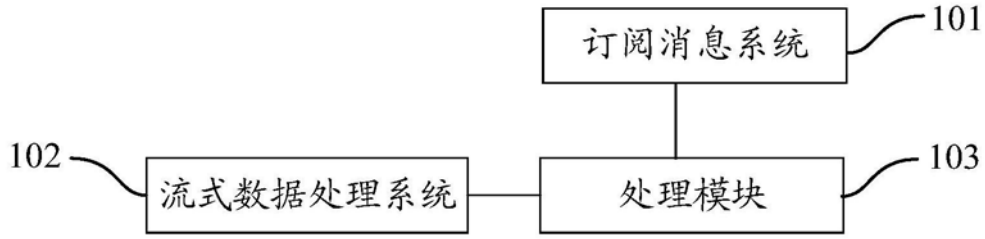

订阅消息系统
101
102
流式数据处理系统
处理模块
103

图1

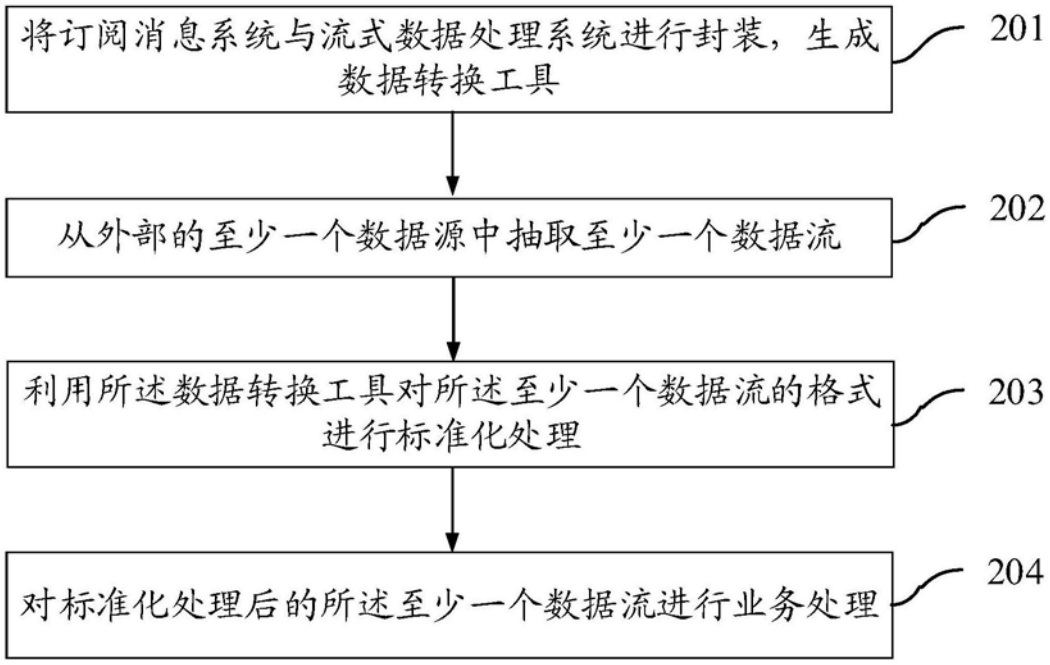

将订阅消息系统与流式数据处理系统进行封装，生成数据转换工具
201
从外部的至少一个数据源中抽取至少一个数据流
202
利用所述数据转换工具对所述至少一个数据流的格式进行标准化处理
203
对标准化处理后的所述至少一个数据流进行业务处理
204

图2

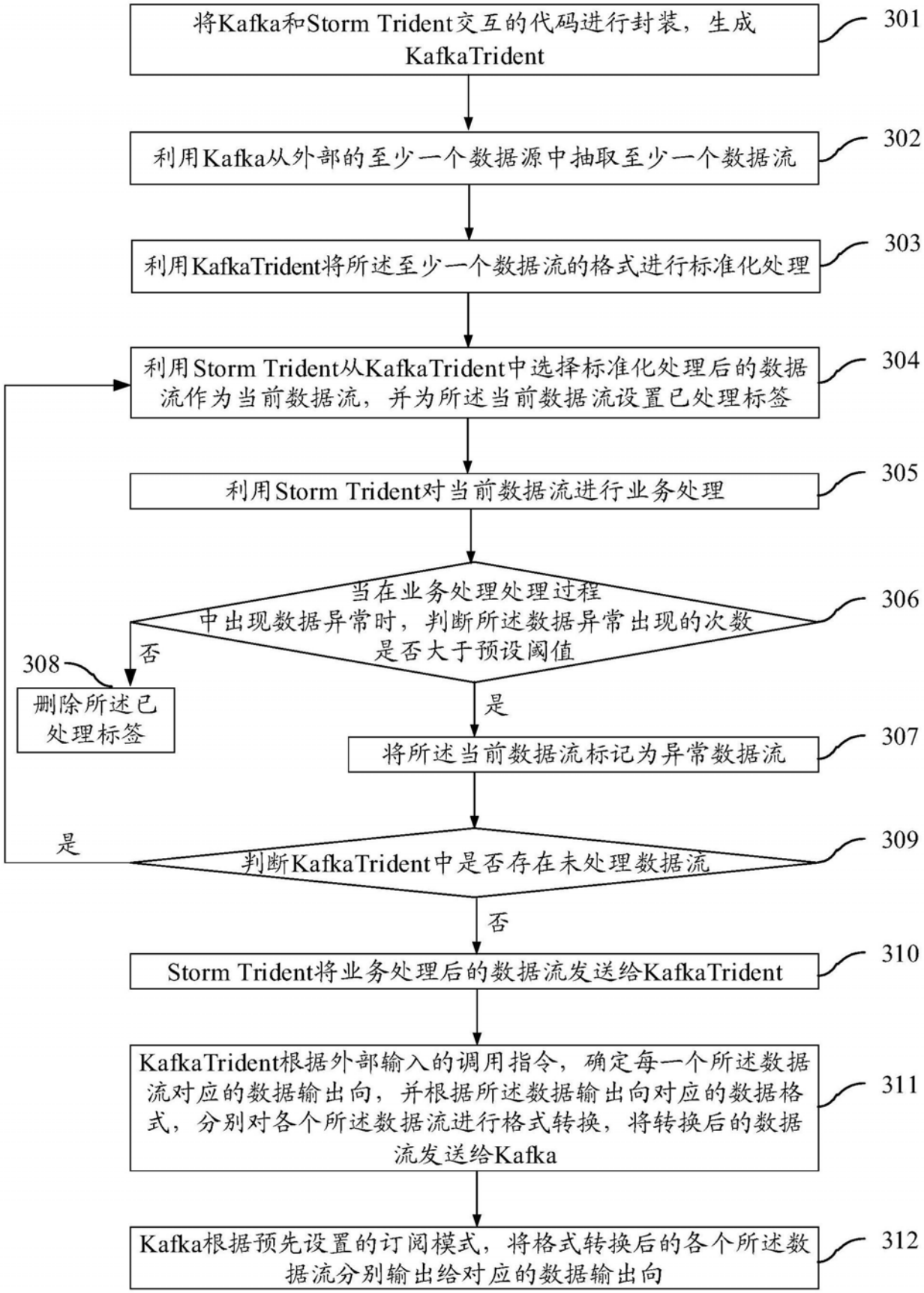

将Kafka和Storm Trident交互的代码进行封装，生成KafkaTrident
301
利用Kafka从外部的至少一个数据源中抽取至少一个数据流
302
利用KafkaTrident将所述至少一个数据流的格式进行标准化处理
303
利用Storm Trident从KafkaTrident中选择标准化处理后的数据流作为当前数据流，并为所述当前数据流设置已处理标签
304
利用Storm Trident对当前数据流进行业务处理
305
当在业务处理处理过程中出现数据异常时，判断所述数据异常出现的次数是否大于预设阈值
306
否
308
删除所述已处理标签
是
将所述当前数据流标记为异常数据流
307
是
判断KafkaTrident中是否存在未处理数据流
309
否
Storm Trident将业务处理后的数据流发送给KafkaTrident
310
KafkaTrident根据外部输入的调用指令，确定每一个所述数据流对应的数据输出向，并根据所述数据输出向对应的数据格式，分别对各个所述数据流进行格式转换，将转换后的数据流发送给Kafka
311
Kafka根据预先设置的订阅模式，将格式转换后的各个所述数据流分别输出给对应的数据输出向
312

图3

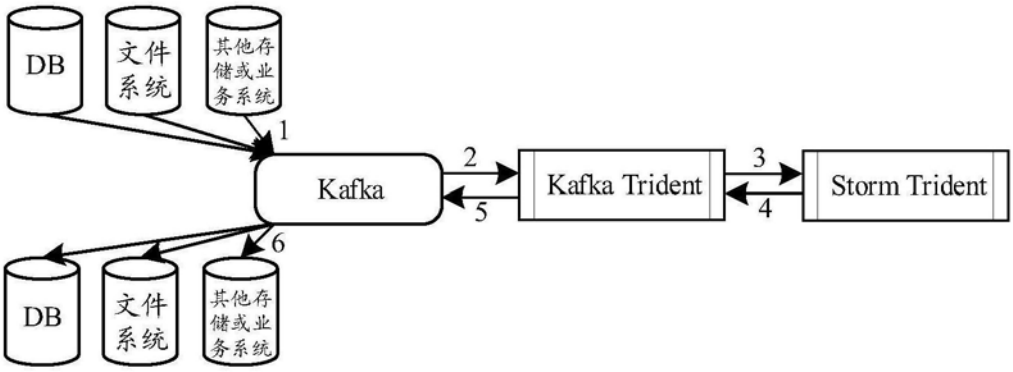

DB
文件系统
其他存储或业务系统
1
Kafka
2
5
Kafka Trident
3
4
Storm Trident
6
DB
文件系统
其他存储或业务系统

图4