(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 1555530 B
(45)授权公告日 2010.05.05

(21)申请号 02817983.8

(22)申请日 2002.07.15

(30)优先权数据
60/305,978 2001.07.16 US
60/305,986 2001.07.16 US
09/975,587 2001.10.11 US
09/975,590 2001.10.11 US

(85)PCT申请进入国家阶段日
2004.03.15

(86)PCT申请的申请数据
PCT/US2002/022366 2002.07.15

(87)PCT申请的公布数据
WO03/009092 EN 2003.01.30

(73)专利权人 BEA 系统公司
地址 美国加利福尼亚州

(72)发明人 迪安·B·雅各布斯 雷托·克雷默
阿南萨恩·B·斯里尼瓦桑

(74)专利代理机构 北京市柳沈律师事务所 11105
代理人 郭定辉 黄小临

(51)Int.Cl.
G06F 15/16(2006.01)
G06F 7/00(2006.01)
G06F 17/30(2006.01)

(56)对比文件
US 5920867 A,1999.07.06, 全文.
US 6088694 A,2000.07.11, 全文.

审查员 李琼

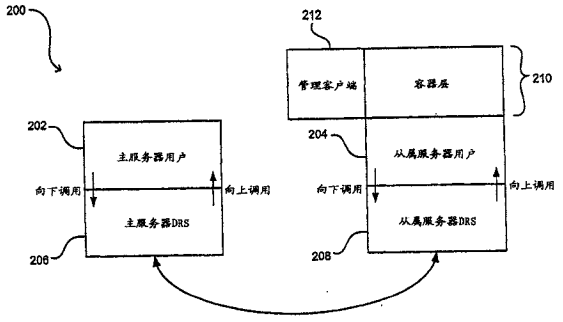
权利要求书 4 页 说明书 10 页 附图 7 页

(54)发明名称

通过网络复制数据的方法和系统

(57)摘要

可以使用一或两阶段方法通过网络复制数据。对于一阶段方法,包含数据原始拷贝的主服务器向网络上每一个从属服务器发送该数据当前状态的版本号,从而每一个从属服务器可以从主服务器请求增量。所请求的增量包括将该从属服务器更新到该数据适当版本所需的数据。对于两阶段方法,主服务器向每一从属服务器发送信息包。由该从属服务器委托(commit)该信息包,如果每个从属服务器都能够处理该委托的话。



CN 1555530 B

1. 一种用来通过网络从主服务器向从属服务器复制数据的方法，该方法包括以下步骤：

从主服务器向从属服务器发送版本号，该版本号有关于存贮在该主服务器上的数据的当前状态；

使该从属服务器能够确定该从属服务器是否已经被更新以反应对应于发送自该主服务器的版本号的该数据的当前状态；以及

如果在该从属服务器不对应由该主服务器发送的版本号，则请求从主服务器向从属服务器发送增量，该增量包含更新该从属服务器所需的信息。

2. 如权利要求 1 所述的方法，进一步包括：

从主服务器向从属服务器发送该增量。

3. 如权利要求 1 所述的方法，进一步包括：

向从属服务器提交该增量。

4. 如权利要求 1 所述的方法，进一步包括：

在提交该增量之后，更新从属服务器的版本号。

5. 如权利要求 1 所述的方法，进一步包括：

从主服务器向从属服务器周期性地发送版本号。

6. 如权利要求 1 所述的方法，进一步包括：

向从属服务器发送版本号直到该从属服务器确认收到该版本号。

7. 一种用来通过包括主服务器与至少一个从属服务器的网络复制数据的方法，该方法包括以下步骤：

从主服务器向网络上每一从属服务器发送信息包，该信息包有关于存贮在该主服务器上的数据的改动，并且包含用于该数据当前状态的当前版本号，该信息包进一步有关于该数据的先前改动以及用于每次先前改动的版本号；

使每一从属服务器都能够确定该从属服务器是否已经被更新以对应该当前版本号；

如果该从属服务器没有错过先前的改动，则允许每一从属服务器提交该信息包；以及

在该从属服务器提交该信息包之前，允许每一错过了先前改动的从属服务器请求从主服务器向该从属服务器发送先前的改动。

8. 一种用来通过包括主服务器与至少一个从属服务器的网络复制数据的方法，该方法包括以下步骤：

从主服务器向网络上每一从属服务器发送信息包，该信息包有关于存贮在该主服务器上的数据的改动，并且包含用于该数据在先状态的在先版本号以及用于该数据新状态的新版本号，该信息包进一步有关于该数据的先前改动以及用于每次先前改动的版本号；

使每一从属服务器都能够确定该从属服务器上的数据是否对应包含在该包中的该在先版本号；

如果在该从属服务器上的该数据对应包含在该包中的该在先版本号，则允许每一从属服务器提交该信息包，该提交还将该从属服务器的版本更新到该新版本号；以及

在该从属服务器提交该信息包之前，允许每一不对应该在先版本号的从属服务器请求从主服务器发送增量，该增量包含将该从属服务器的版本更新到该在先版本号所需的信息包。

9. 一种用来通过包括主服务器与至少一个从属服务器的网络复制数据的方法，该方法包括以下步骤：

从主服务器向网络上每一从属服务器发送信息包，该信息包有关于存贮在该主服务器上的数据的改动，并且包含用于该数据在先状态的版本号以及用于该数据新状态的版本号，该信息包进一步有关于该数据的先前改动以及用于每次先前改动的版本号；

使每一从属服务器都能够确定该从属服务器上的数据是否对应包含在该包中的该在先版本号；

如果在该从属服务器上的该数据对应包含在该包中的该在先版本号，则允许每一从属服务器提交该信息包，该提交还将该从属服务器的版本更新到该新版本号；以及

允许每一不对应该在先版本号的从属服务器请求从主服务器发送增量，该增量包含将该从属服务器更新到该新版本号所需的信息包。

10. 一种用来通过网络从主服务器向至少一个从属服务器复制数据的方法，该方法包括以下步骤：

从主服务器向从属服务器发送信息包，该信息包有关于存贮在该主服务器上的数据的改动，并且包含该数据当前状态的版本号；

接收给从属服务器的信息包；

使该从属服务器能够确定在该从属服务器是否已经被更新以对应包含在该包中的版本号，并且进一步确定如果需要更新以对应包含在该包中的版本号，该从属服务器是否能够处理该信息包；

从该从属服务器向主服务器发送信号，该信号表示该从属服务器是否需要被更新，以及该从属服务器是否能够处理该更新；以及

从主服务器向从属服务器发送响应信号，该信号表示该从属服务器是否应该对包含在该包中的信息进行提交；以及

如果该响应信号表示应该如此，则向该从属服务器提交该信息包。

11. 如权利要求 10 所述的方法，进一步包括：

只要当该至少一个从属服务器中任何一个不能处理该提交，就中止该数据。

12. 如权利要求 10 所述的方法，进一步包括：

向那些能够处理该提交的从属服务器提交该数据。

13. 如权利要求 10 所述的方法，进一步包括：

向该至少一个从属服务器中任何一个不能处理该提交的从属服务器多播该数据。

14. 如权利要求 10 所述的方法，进一步包括：

向该至少一个从属服务器中任何一个不能处理该提交的从属服务器心跳该新版本号。

15. 如权利要求 10 所述的方法，进一步包括：

请求向不能处理该提交的从属服务器发送增量。

16. 一种用来通过网络复制数据的方法，该方法包括以下步骤：

(a) 确定该复制应该用一阶段还是两阶段方法完成；

(b) 通过以下方法发送用一阶段方法复制的信息：

从主服务器向从属服务器发送该数据当前状态的版本号；

如果在该从属服务器上的该数据不对应该版本号，则请求从主服务器向从属服务器发

送增量；以及

(c) 通过以下方法发送用两阶段方法复制的信息：

从主服务器向从属服务器发送信息包；

确定该从属服务器是否能够处理该信息包；以及

如果该从属服务器能够处理该信息包，则向该从属服务器提交该信息包。

17. 一种用来在网络上从主机向多个从属复制数据的方法，该方法包括以下步骤：

(a) 确定该复制应该用一阶段还是两阶段方法完成；

(b) 通过以下方法发送用一阶段方法复制的信息：

从主机向每一个从属发送该数据当前状态的版本号；

请求从主机向每一包含不对应该版本号的数据的从属发送增量；

(c) 通过以下方法发送用两阶段方法复制的信息：

从主机向从属发送信息包；以及

如果该多个从属中的每一个都能够处理该信息包，则向该从属提交该信息包。

18. 一种用来使用一及两阶段方法在网络上从主机向多个从属复制数据的方法，该方法包括以下步骤：

(a) 通过以下方法发送用一阶段方法复制的信息：从主机向每一个从属发送该数据当前状态的版本号，从而每一个从属可以请求从主机向该从属发送增量以更新该从属上的该数据；以及

(b) 通过以下方法发送用两阶段方法复制的信息：从主机向每一个从属发送信息包，如果每一个从属都能够提交该信息包，则每一个从属提交该信息包。

19. 一种用来通过网络复制数据的系统，包括：

a. 包含该数据原始拷贝的主服务器，所述主服务器包括：

i. 主服务器用户层，用来通过调用开始方法来开始数据复制过程，该主服务器用户层进一步用来发送有关于该数据原始拷贝的信息；

ii. 主服务器服务层，包括开始方法，并且用来接收来自主服务器用户层的调用以及有关于该数据原始拷贝的信息，该主服务器服务层进一步用来产生并且发送数据复制包，该包包含至少一些有关于该数据原始拷贝的信息；

b. 从属服务器，用来存储来自该主服务器的数据的拷贝，该从属服务器包括：

i. 从属服务器服务层，用来接收来自主服务器服务层的数据复制包，并且处理该数据复制包，该从属服务器服务层进一步用来发送有关于该数据复制包的信息；

ii. 从属服务器用户层，用来接收来自从属服务器服务层的有关于该数据复制包的信息，该从属服务器用户层进一步用来存储在该数据复制包中的信息。

20. 如权利要求 19 所述的系统，其中所述主服务器用户层用来以增量的形式发送有关于该数据原始拷贝的信息，该增量包含有关于该数据原始拷贝的当前状态与先前状态之间的改动的信息。

21. 如权利要求 19 所述的系统，其中所述主服务器用户层用来发生退回消息，该消息表示对该数据原始拷贝的改动不应该在从属服务器上复制。

22. 如权利要求 19 所述的系统，其中所述主服务器用户层用来为该复制设置超时值。

23. 如权利要求 19 所述的系统，其中所述主服务器用户层用来产生该数据原始拷贝的

当前状态与该数据原始拷贝的在先状态之间的增量。

24. 如权利要求 19 所述的系统，其中所述主服务器用户层用来产生该数据原始拷贝的当前状态与该数据原始拷贝的先前状态之间的增量。

25. 如权利要求 19 所述的系统，其中所述主服务器用户层用来为该数据原始拷贝的每一个状态生成唯一的版本号。

26. 如权利要求 19 所述的系统，其中所述主服务器服务层用来多播该数据复制包。

27. 如权利要求 19 所述的系统，其中所述主服务器服务层用来心跳该数据复制包。

28. 如权利要求 19 所述的系统，其中所述主服务器服务层用来在该数据复制包内包括版本号。

通过网络复制数据的方法和系统

[0001] 优先权声明

[0002] 本发明要求以下优先权：美国临时专利申请60/305986，2001年7月16日提交，名为“数据复制协议”；美国临时专利申请60/305978，2001年7月16日提交，名为“用于数据复制的分层体系结构”；美国专利申请09/975590，2001年10月11日提交，名为“数据复制协议”；美国专利申请09/975578，2001年10月11日提交，名为“用于数据复制的分层体系结构”，此处融入以上文件内容作为参考。

[0003] 版权声明

[0004] 本专利文件公开的一部分包含受版权保护的材料。版权所有人不反对任何人精确复制本专利文件或者专利公开出版物，如其在专利与商标局的专利文件或记录中一样，但保留其他所有版权权利。

技术领域

[0005] 一般地，本发明涉及用于传递数据的系统。更具体地讲，本发明涉及用于通过网络复制数据的系统与方法。

背景技术

[0006] 存在几种类型的分布式处理系统。一般地，分布式处理系统包括多个处理设备，诸如通过通信介质连接的两台计算机。一种分布式处理系统为客户端/服务器网络。客户端/服务器网络包括至少两个处理设备，一般为中央处理器与客户端。其他客户端可能连接至该中央服务器，可能有多个服务器，或者该网络可能只包括通过通信介质连接的多个服务器。

[0007] 在这样的网络环境中，经常希望从中央服务器向许多个工作站和/或其他服务器发送应用程序或信息。这常常可能涉及在各个工作站上的独立的安装，或者可能涉及从该中央服务器向每个单个的工作站和/或服务器分别推送新的信息库。这些方法可能很耗时，并且资源的使用效率不高。在各个工作站或服务器上分别安装应用程序也引入了其他潜在的错误来源。

[0008] 理想状况下，信息的发送既应该对故障可靠，也应该可扩展，以便该处理有效使用网络。现有的解决方案一般不能达到这些目标的一个或全部。一个简单的方法是让主服务器分别联系各个从属，并且通过点到点链接（诸如TCP/IP连接）传递数据。如果一个或更多个从属暂时不可达到，或者如果从属在处理更新时出现错误，则这种方法将导致数据的不一致的副本。复杂分布式协定协议是另一个极端，其要求在从属之间进行大量会话，以确保数据的所有副本一致。

发明内容

[0009] 本发明包括一种方法，用来从主服务器向至少一个从属或被管理服务器复制数据，例如这可以通过网络完成。在该方法中，可能要确定该复制应该用一阶段还是两阶段方

法完成。如果要用一阶段方法完成该复制，则可以发送对应主服务器上数据当前状态的版本号。该版本号可以被送往网络上的每一个从属服务器，或者只送往部分从属服务器。然后，收到该版本号的从属服务器可能请求从主服务器发送增量(delta)。该增量可以包含更新该从属服务器上数据以对应当前版本号所需的数据。

[0010] 如果要用两阶段方法完成该复制，则从主服务器向每一个从属服务器或者从属服务器的子集发送信息包。然后，这些从属服务器可以对主服务器作出响应，表示它们是否能够提交该信息包。如果至少有一些从属服务器能够提交该数据，则主服务器可以向这些从属服务器发送信号，表示它们应该处理该提交(commit)。在处理该提交之后，这些从属服务器可以更新到当前版本号。如果任一从属服务器不能处理该提交，则可以中止该提交。

附图说明

[0011] 图 1 为根据本发明的一个实施例的域结构的图；

[0012] 图 2 为根据本发明的一个实施例的分层体系结构的图；

[0013] 图 3 为根据本发明的一个实施例的群集域结构的图；

[0014] 图 4 为根据本发明的一个实施例的用于分层体系结构的一阶段处理的图；

[0015] 图 5 为根据本发明的一个实施例的用于分层体系结构的两阶段处理的图；

[0016] 图 6 为根据本发明的一个实施例的一阶段处理的流程图；

[0017] 图 7 为根据本发明的一个实施例的两阶段处理的流程图；

具体实施方式

[0018] 本发明支持将数据或诸如来自从主服务器或“管理员”服务器（“管理服务器”）之类的其他信息复制到例如一组从属服务器或“被管理”服务器。这种复制可以通过任何适当的网络发生，例如常规的局域网或以太网。在一个实施例是，主服务器拥有网络上所有数据的原始记录，任何更新都应用该原始记录。该数据的副本与更新（在其发生时）一起可以被发送到每个从属服务器。一个示例应用为从管理服务器向一组被管理服务器分发配置信息。

[0019] 在根据本发明的一个系统中，可能需要一种诸如数据复制服务(DRS)之类的服务来从管理服务器向适当域中的被管理服务器分发配置与部署信息。大数据项可以通过点到点连接（例如传输控制协议(“TCP”)）分发，这是因为多播协议如用户数据报协议(“UDP”)没有流控制，并且可能使系统崩溃。远程方法调用(RMI)、超文本传送协议(HTTP)或者类似协议可以用于点到点连接。

[0020] 被管理服务器也可以在本地磁盘上持久缓存数据。如果没有这种缓存，可能需要无法接受的时间量来传送必须的数据。被管理服务器的缓存能力至观重要，这是因为这种能力通过减少需传送的启动数据量而提高了启动速度。如果管理服务器不可达到，还允许启动和/或重启缓存。重启可能是更吸引人的选择，并且可能的情况是：管理服务器指示服务器启动。然而，重启可能提供在管理服务器不可用时启动域的能力。

[0021] 如图 1 的域结构 100 所示，管理服务器 102 和至少一个被管理服务器 104 可以构成域 106。这个域 106 可以是启动与关闭的管理单位。在一个实施例中，浏览器 108 或者其他用户应用程序或设备命令管理服务器 102 启动。然后，管理服务器 102 命令域 106 中的

所有被管理服务器 104 启动，并且发送适当的配置信息。如果在被管理服务器 104 已经启动之后一台服务器出了故障，则可能希望该服务器自动重启，而不管管理服务器 102 是否可用。缓存数据可以用于此目的。

[0022] 对管理服务器上数据的更新可以作为版本之间的增量增量打包。该增量可能包含要改变的配置和 / 或其他信息。可能希望在该域运行时更新配置，这是因为可能不希望将系统脱机。在一个实施例中，配置更改动态发生，因为这些修改由管理服务器推送。只有对配置的修改以增量传送，这是因为可能不需要每次都发送全部配置，并且可能造成不必要的麻烦。

[0023] 根据本发明的协议集成了更新分发的两种方法，尽管可以相应地使用其他适当方法。这些分发方法可以称为一阶段方法与两阶段方法，并且提供了一致性与可扩展性之间的平衡。在一阶段方法中，其可能对可扩展性有利，每个从属服务器可以按各自的节奏取得并处理更新。从属服务器可能在不同的时间从主服务器取得更新，但可以在接到数据后马上提交。从属服务器在处理更新时可能遇到错误，但在一阶段方法中这不会阻止其他从属服务器处理该更新。

[0024] 在根据本发明的两阶段方法中，其可能对一致性有利，分发可能是“原子”的，即全部从属服务器或者无任何从属服务器成功处理该数据。可能有独立的阶段（例如准备与提交阶段），这考虑了中止的可能性。在准备阶段，主服务器可以确定是否每一从属服务器都可能采用该更新。如果所有从属服务器都表示能够接受该更新，则可以将新数据送往从属服务器，以在提交阶段提交。如果从属服务器中的至少一个不能采用该更新，则可以中止该更新，并且不会有提交。在这种情况下，可以通知被管理服务器应该退回该准备，并且不做任何改变。根据本发明的这种协议是可靠的，这是因为 :在任一方法中，在提交更新时不可达到的从属服务器最终会得到该更新。

[0025] 根据本发明的系统也可以确保暂时不可用的服务器最终会接收到所有更新。例如，服务器可能会暂时从网络中隔离，然后返回网络而不用重启。因为服务器不重启，所以该服务器一般将不检查更新。可以通过以下方法处理重返网络的服务器 :让服务器定期检查新更新，或者让主服务器定期查看这些服务器是否已经接收到了这些更新。

[0026] 在一个实施例中，主服务器定期发送多播 (multicast) 的“心跳 (heartbeat)”给从属服务器。因为多播方法可能不可靠，所以从属服务器可能会错过任意序列的心跳。例如，由于网络分区，从属服务器可能暂时从网络中断开，或者从属服务器可能暂时对网络不可用，从而错过心跳。因此，心跳可能包含有关最近更新的信息窗口。这样的有关先前更新的信息可以用来降低网络业务量，如下所述。

[0027] 在每一主服务器与每一从属服务器中可以有至少两层 :用户层与系统层（或者 DRS 层）。用户层可以相应于该数据复制系统的用户。DRS 层可以相应于该数据复制系统自身的实现。有关各方与层的交互如图 2 所示。

[0028] 如图 2 的启动图 200 所示，本实施例中，主服务器用户 202 与从属服务器用户 204 层分别向下调用主服务器 DRS 206 与从属服务器 DRS 208 层。这些向下调用可以采用以下形式（例如）：

[0029] registerMaster(DID, verNum, listener)

[0030] registerSlave(DID, verNum, listener)

[0031] 其中 DID 为得自公知 DID 知识的标识符并且指向感兴趣的对象 ;verNum 得自本地持久的存储,作为该用户的当前版本号 ;listener 为将处理来自 DRS 层的向上调用 (upcall) 的对象。该向上调用可以调用该 listener 对象的方法。然后主服务器可以开始发送心跳或者周期性增量以及当前版本号。显示了容器层 210,其可以包含适合于从从属服务器用户 204 取得信息的容器。可能的容器例子包括企业版 Java beans,万维网界面以及 J2EE(Java 2 平台企业版) 应用程序。其他应用程序和 / 或组件可以插入容器层 210(例如管理客户端 212)。在用户与 DRS 层之间更新消息往来的例子在图 4 的一阶段方法以及图 5 的两阶段方法中示出。

[0032] 图 4 显示了一个基本过程 400,其可用于根据本发明的分层体系结构中的一阶段分发方法。在这一过程中,主服务器用户层 402 对主服务器 DRS 层 406 进行向下调用 (downcall)404,以开始一阶段分发。该调用可以是针对该系统中所有从属服务器的,也可以是只针对从属服务器的一个子集。如果该调用针对子集,则主服务器用户层 402 可以确定该更新的范围,或者哪些从属服务器应该接收到该更新。

[0033] 主服务器 DRS 层开始向从属服务器 DRS 层 410 多播心跳 408,其包含了主服务器上数据的当前版本号。从属服务器 DRS 层 410 从从属服务器用户层 414 请求用于该从属服务器的当前版本号。然后,从属服务器用户层 414 以从属服务器版本号响应 416 从属服务器 DRS 层 416。如果该从属服务器同步,或者已经是当前版本号,则不再做进一步的请求,直到下一次更新。如果该从属服务器不同步,并且该从属服务器在该更新范围内,则从属服务器 DRS 层 410 可以从主服务器 DRS 层 406 请求增量 420,以将该从属服务器更新到主服务器上数据的当前版本号。主服务器 DRS 层 406 请求 422 主用户层 402 创建增量以更新从属服务器。然后主服务器用户层 402 发送增量 424 到主服务器 DRS 层 406,主服务器 DRS 层 406 将增量 426 与主服务器的当前版本号转发到从属服务器 DRS 层 410,从属服务器 DRS 层 410 将增量 426 发送到从属服务器用户以备提交。当前版本号与增量一起发送,以防止从属服务器接到心跳 408 后主服务器进行了更新。

[0034] 主服务器 DRS 层 406 可能继续定期向 (多个) 从属服务器发送包含版本号 408 的多播心跳。这就使不可用或者不能接受并处理增量的任何从属服务器能够确定其不在数据的当前版本号上并且在以后请求增量 420,例如当该从属服务器返回该系统时。

[0035] 图 5 显示了一个基本过程 500,其可用于根据本发明的分层体系结构中的两阶段分发方法。在这一过程中,主服务器用户层 502 向下调用 504 进入主服务器 DRS 层 506,以开始两阶段分发。主服务器用户层 502 也可能需要确定该更新的范围,并且可以为该更新过程设置“超时”值。

[0036] 主服务器 DRS 层 506 向从属服务器 DRS 层 510 发送新增量 508。从属服务器 DRS 层 510 为该新增量向从属服务器用户层 514 发送准备请求 512。然后,从属服务器用户层 514 向从属服务器 DRS 层 510 响应该从属服务器是否能够处理该新增量。从属服务器 DRS 层将响应 518 转发给主服务器 DRS 层 506。如果该从属服务器因为其不同步不能处理该请求,则主服务器 DRS 层 506 对主服务器用户层 502 进行向上调用 520,以生成将使该从属服务器同步从而提交的增量。主服务器用户层 502 向主服务器 DRS 层发送同步增量 522,主服务器 DRS 层将同步增量 524 转发给从属服务器 DRS 层 510。如果该从属服务器能够处理该同步增量,则从属服务器 DRS 层 510 将向主服务器 DRS 层 506 发送同步响应 526,表示该

从属服务器现在可以处理该新增量。如果该从属服务器不能处理该同步增量，则从属服务器 DRS 层 510 将向主服务器 DRS 层 506 发送适当的同步响应 526。然后，根据从属服务器是否响应其能够处理该新增量，主服务器 DRS 层 506 向从属服务器 DRS 层 510 发送心跳提交 / 中止消息 528。如果所有从属服务器都能够准备该增量，则例如主服务器可以发送心跳提交信号。否则，主服务器可以发送心跳中止信号。心跳也可以包含更新的范围，以使从属服务器知晓其是否应该处理包含在该心跳中的信息。

[0037] 从属服务器 DRS 层将此命令 530 转发给从属服务器用户层 514，然后从属服务器用户层 514 为提交或中止对该新增量的更新。如果在由主服务器用户层 502 所设置的超时 (timeout) 值之内没有完成准备阶段，则主服务器 DRS 层 506 可以自动向所有从属服务器发送心跳中止 528。例如，当主服务器 DRS 层 506 不能联系从属服务器中的至少一个以确定该从属服务器是否能够处理提交时，这就可能发生。可以这样设置超时值，使主服务器 DRS 层 506 在中止更新前的一段特定时间尝试联系该从属服务器。

[0038] 对于一阶段方法中的更新，这些心跳可能引起每一从属服务器请求从该从属服务器的数据当前版本开始的增量。这一过程在图 6 的流程图中显示。在该基本过程 600 中，其可能使用或不使用根据本发明的分层体系结构，从主服务器向从属服务器发送主服务器上当前数据的版本号 602。从属服务器确定自己是否已经被更新到该当前版本号 604。如果从属服务器不在当前版本上，则它将请求从主服务器发送增量以更新该从属服务器 606。当增量被发送到从属服务器时，该从属服务器将处理该增量以将从属服务器数据更新到当前版本 608。然后，从属服务器将自己的版本号更新到单前版本号 610。

[0039] 对于两阶段方法中的更新，主服务器可以开始于准备阶段，在该阶段中主服务器主动向每一从属服务器发送从紧接上一版本的增量。这一过程在图 7 的流程图中显示。在基本过程 700 中，其可能使用或不使用根据本发明的分层体系结构，从主服务器向（多个）从属服务器发送信息包 702。每个收到该包的从属服务器确定自己是否能够处理该包并且更新到该当前版本 704。每个收到该包的从属服务器向主服务器响应，指示该从属服务器是否能够处理该包。如果所有从属服务器（向它们发送了增量）在某个超时时段内确认成功处理了增量，则主服务器可能决定提交该更新。否则，主服务器可能决定中止该更新。一旦作出了该决定，则主服务器向（多个）从属服务器发送消息，表示应该提交或中止该更新 708。如果该决定是提交，则每一服务器处理提交 710。可以进一步使用心跳来表示提交或中止是否发生了，以防止从属服务器之一错过该命令。

[0040] 可以配置从属服务器以使用缓存数据来立即启动和 / 或重启，而不首先从主服务器获得当前版本号。如上所述，根据本发明的一种协议允许从属服务器在本地磁盘上持久缓存数据。该缓存降低了系统启动所需的时间，并且通过降低需要传送的数据量而增强了可扩展性。通过允许从属服务器在主服务器不可达到时能够启动和 / 或重启，该协议可以增强可靠性，并且可以进一步允许将更新打包为版本之间的增量。如果不存在缓存的数据，则从属服务器可以等待主服务器或者可以自己下拉数据。如果从属服务器有缓存，则其仍然可能不希望失去同步。如果从属服务器知道等待，则可以降低启动时间。

[0041] 该协议可以是双边的，即根据环境，主服务器或者从属服务器可以开始传输数据。例如，在域启动期间，从属服务器可以从主服务器下拉增量。当从属服务器确定自己与该增量所要更新的版本不同时，该从属服务器可以请求从其当前版本到当前系统版本的增量。

在一阶段分发期间，从属服务器也可以下拉增量。此处，该系统可以读取心跳，确定自己已经错过了更新，并且请求适当的增量。

[0042] 当需要从例外情况恢复时，从属服务器也可以下拉增量。例如，当该系统的组件失去同步时，就可能存在以外情况。当从属服务器下拉增量时，该增量可以是数据任意版本之间的。换而言之，该增量可以是该从属服务器的当前版本与该系统（或域）的当前版本之间的，而不管这些版本之间可能有多少代。在此实施例中，心跳的可用性以及接收增量的功能可以提供该系统的同步。

[0043] 除了从属服务器可以下拉增量，主服务器也可以在两阶段分发期间向从属服务器推送增量。在一个实施例中，这些增量总是在数据的连续版本之间的。该两阶段分发方法可以最小化有关方面之间不一致的可能性。只要可能，从属服务器用户就可以处理准备，而不会向客户端公开该更新或者使该更新无法退回。这可以包括诸如检查服务器已防冲突的任务。如果任何一个从属服务器发出出错信号，例如通过发送“磁盘满”或者“不一致配置”信息，则可以将该更新统一退回。

[0044] 然而，仍然有可能出现不一致。例如，在处理提交时可能出错，原因是诸如不能打开套接字。服务器也可以在不同的时间提交并公开更新。因为数据不能刚好在同一时间到达每一个被管理的服务器，所以会有一些波动效应。使用多播可以支持小的时间窗口，以最小化该波动效应。在一个实施例中，如果错过了提交，不管是错过了该信号，还是主服务器崩溃等等，则已做准备的从属服务器会中止，

[0045] 多播的尽力型方法可以使从属服务器错过提交信号。如果主服务器在提交阶段中途崩溃，则可能会没有日志，或者没有恢复的方法。对于主服务器可能没有方法命令剩余的从属服务器它们需要提交。一旦中止，如果版本没有被正确地退回，则一些从属服务器可能会终止提交该数据。在一个实施例中，剩余从属服务器可以使用一阶段分发获得更新。例如，这可以发生在被管理服务器响应于接收自管理服务器的心跳而下拉增量时。这种方法可以保持系统可扩展性，而如果为了避免任何提交或者版本错，系统限制(tie down)分发，则可能失去可扩展性。

[0046] 系统所管理的每一数据项都可以被构造成具有唯一的长寿命的域标识符 (DID)，该域标识符在整个域上是公知的。数据项可以是大型复杂对象，包括许多组件，每一组件都与该域内服务器的某一子集有关。因为这些对象可以是一致性的单位，所以希望有几个大型对象，而不是多个小型对象，作为示例，单独一个数据项或对象就可以代表系统的所有配置信息，包括诸如 config.xml 文件或应用程序 EAR(application-EAR) 文件的代码文件。数据项内的给定组件，例如，可以有关于单个服务器的线程数目，有关于族的被部署服务，或者有关于整个域的安全证书。两个版本之间的增量可以包括所有或者一些这些组件的新值。例如，这些组件可以包括部署在域成员上的企业版 Java Beans。增量可以只包括对这些 Java Beans 的子集的改动。

[0047] 增量的“范围”可以指在该增量中具有相关组件的所有服务器的集合。根据本发明的管理服务器可能能够截收配置改动，以确定该增量的范围。主服务器上的 DRS 系统可能需要知道该范围，以将数据发送给适当的从属服务器。当主服务器可能在每次更新中只需要联系服务器的一个子集时，向每一服务器发送每一配置更新可能是对时间与资源的浪费。

[0048] 为了控制分发，主服务器用户可以与连续版本之间的增量一起提供每一更新的范围。范围可以表示为一组指向服务器和/或族的名称，其可以取自域内的同一命名空间。在一个实施例中，DRS 用户使用解析器模块将名称映射到地址。族名称可以映射到在该族内的所有服务器的地址的集合。这些地址可能是相对的，例如相对于虚拟机器。如本领域已知并且使用的，该解析器可以确定是否存在中间防火墙，并且根据该服务器是否“在防火墙里面”而返回“里面”或着“外面”的地址。管理服务器或者其他服务器可以用配置数据初始化相应的解析器。

[0049] 与每一被管理的数据项的唯一的长寿命的标识符 (DID) 一起，数据项的每一个版本也可以具有长寿命的版本号。每一版本号对更新企图可以是唯一的，以使服务器不会因为对正确版本的混淆而错误地更新或者不更新。类似地，被中止的两阶段分发的版本号也不能复用。只要给定了版本号，主服务器就可以产生两个任意版本之间的增量。如果主服务器不能产生这样的增量，则可以提供数据或应用程序的完整拷贝。

[0050] 可能希望该数据复制服务尽可能地通用。因此可能对该系统的用户有几个假定。例如，该系统可能依赖于三个主要假定：

[0051] 该系统可能包括增加版本号的方法。

[0052] 该系统可能将版本号持久存贮在主服务器以及从属服务器上。

[0053] 该系统可能包括比较版本号并且确定相等的方法。

[0054] 满足这些假定可以通过用户级的 DRS 接口实现，例如“VersionNumber”接口。这样的接口使用户能够提供版本号抽象的特定概念与实现，同时保证该系统可以访问版本号属性。例如，在 Java 中 VersionNumber 接口可以以下实现：

[0055]
```
package weblogic.drs;
```

[0056]
```
public interface VersionNumber extends Serializable{
```

[0057]
```
  VersionNumber increment();
```

[0058]
```
  void persist()throws Exception;
```

[0059]
```
  boolean equals(VersionNumber anotherVN);
```

[0060]
```
  boolean strictlyGreaterThan(VersionNumber anotherVN);
```

[0061]
```
}
```

[0062] 用户可以向系统提供的这一抽象的简单化的实现可以是大的正整数。该实现也可以确保该系统可以将增量信息借助网络从主服务器传送到从属服务器，这在本领域中被称为“可序列化”。

[0063] 如果使用上述的抽象，则在用户级上从增量的详细内容的概念上进行抽象可能是有用的。系统可能不需要知道增量信息结构，并且实际上系统根本就不能确定该结构。增量的实现也可能是可序列化的，这就确保了系统能够借助网络将增量版本信息从主服务器传送到从属服务器。

[0064] 可能希望主服务器与适当的 DID 和版本号一起持久存储每一数据项的记录的副本。在开始两阶段分发之前，主服务器可以持久存储所提出的新版本号，以确保万一主服务器出故障时，该版本号不会被复用。从属服务器可以与适当的 DID 和版本号一起持久存储每一相关数据项的最近副本。从属服务器也可以被配置为进行必须的缓存，以使该从属服务器可能必须每一次都取得数据或协议。这可能不是对所有情况都希望的，但可以提供来

处理可能发生的某些情况。

[0065] 根据本发明的系统可以进一步包括并发限制。例如，在更新的两阶段分发期间，对于给定域上的给定 DID，可能不允许某些操作。这些操作可能包括一或两阶段更新，例如在同一 DID 上，将范围成员身份修改到非空交集范围上。

[0066] 在至少一个实施例中，主服务器向域内的每一服务器上的从属服务器 DRS 定期多播心跳或者信息包。对于每一 DID，心跳可以包括：有关（多个）最近更新的信息窗口，包括每一更新版本号；相对于先前版本的增量的范围，以及是否提交或者中止该更新。也可能包括有关当前版本的信息。也可以使用有关较老版本的信息，以最小化返回主服务器的业务量，并且不是用于正确性或者鲜活性。

[0067] 在增量中包括较老版本信息的情况下，从属服务器可以提交自己在准备时所期望的那一部分更新，并且请求新的增量以处理更近的更新。至少对于某些固定的可配置数目的心跳，可以包括有关给定版本的信息，尽管快速的更新可能使窗口增大到不可接受的尺寸。在另一实施例中，一旦主服务器确定所有从属服务器都已经接收了更新，则可以抛弃有关较老版本的信息。

[0068] 多播的心跳可以顾及到多个属性。这些心跳可以是异步的或“单向”的。结果，到从属服务器响应心跳的时间，主服务器可能已经升级到新的状态。另外，并非所有的从属服务器可以在刚好同一时间响应。这样一来，主服务器可以假定从属服务器不知道它的状态，并且可以包括增量要更新的状态。这些心跳也可以是不可靠的，因为从属服务器可能错过任意序列的心跳。这又可以导致在心跳中包括较老版本信息。在一个实施例中，从属服务器以心跳发送的顺序接收心跳。例如，从属服务器不能调节版本七，直到它提交了版本六。服务器可以等待直到它收到六，或者它可以简单地抛弃六而提交七。这种顺序可以消除由于版本回退而产生混乱的可能性。

[0069] 如上所述，域也可能使用族 (cluster)，如图 3 所示（多播心跳滑动属性）。此实施例的一般网络拓扑为多播岛集合，这些多播岛连接到包含主服务器的集线器岛。多播业务可以从集线器向外点到点地转发。可能在一阶段方法中分发的小增量可以直接通过多播发送。在其他情况下，可以在毂辐式多播结构上加上点到点转发方案，以减少主服务器处的瓶颈。

[0070] 在图 3 的域图 300 中，可以将一个或更多个被管理服务器 302 组合到多播岛内，也称为族 304。域 308 的管理服务器 306 作为集线器岛 312 的主服务器，并且是到该域的入口点，例如通过浏览器 310。管理服务器 306 联系族中的一个被管理服务器，也被称为族主服务器。在此实施例中，管理服务器可以向每一族主服务器多播增量或者消息，然后每一族主服务器通过多播向该族内的其他被管理服务器转发该增量或者消息。族主服务器不能拥有任何配置信息，而是从管理服务器接收该信息。在族主服务器下线或者崩溃的情况下，该域中的另外一个被管理服务器可以顶替作为族主服务器。在这种情况下，可以设置一种机制来防止下线服务器返回该族作为第二族主服务器。这可以通过族或系统基础结构处理。

[0071] 也可能有多于一个的域。在这种情况下，可以有嵌套域或者“合成域”。通过直接联系每一个域主服务器，可以将信息发布给域主服务器，这是因为每一域主服务器可以具有将信息推到其他域主服务器的功能。然而，可能不希望向域主服务器进行多播。

[0072] 在一阶段分发中，域主服务器可以进行向下调用以触发更新的分发。这样的向下

调用可以采用以下形式：

[0073] startOnePhase(DID,newVerNum,scope)

[0074] 其中 DID 为被更新的数据项或者对象的 ID，newVerNum 为该对象的新的版本号，scope 为应用该更新的范围。主服务器 DRS 可以通过以下方式响应：升级到新的版本号，将新号写入磁盘，并且在随后的心跳中包括该信息。

[0075] 当从属服务器 DRS 接收心跳时，通过分析与最近更新有关的信息窗口，它可以确定自己是否需要下拉。如果从属服务器的当前版本号在该窗口之内，并且该从属服务器不在任何随后的被提交更新的范围之内，则它可以简单地升级到最近的版本号，而不下拉任何数据。该过程可以包括从属服务器为最新的平凡情况。否则，从属服务器 DRS 可以进行点到点调用，要求从主服务器 DRS 的增量，或者另一类似请求，其形式可能为：

[0076] createDelta(DID,curVerNum)

[0077] 其中 curVerNum 为从属服务器的当前号，这将被送回到域主服务器或者族主服务器。为了处理该请求，主服务器 DRS 可以进行向上调用，例如 create(curVerNum)。该向上调用可以通过适当的监听器进行，以取得增量与新的版本号，并且将这些返回到从属服务器 DRS。应该包括新的版本号，这是因为在从属服务器最后一次收到心跳之后版本号可能已经改变了。增量可以只相当于最近提交的更新。任何进行着的两阶段更新可以通过独立的机制进行处理。然后，从属服务器 DRS 可以向从属服务器用户进行向上调用，例如 commitOnePhase(newVerNum,delta)，然后升级到新版本号。

[0078] 为了触发两阶段分发，主服务器用户可以进行向下调用，例如 startTwoPhase(DID,oldVerNum,newVerNum,delta,scope,timeout)，其中 DID 为待更新的数据项或者对象，oldVerNum 为先前版本号，newVerNum 为新版本号（距先前版本号一步），delta 为待推送的连续版本之间的增量，scope 为更新的范围，而 timeout 为该任务的最大生存时间。因为“准备”与“提交”是同步的，所以可能希望为该任务设置特定的时间限制。可以包括先前的版本号，以使不同版本号上的服务器不会采用该增量。

[0079] 在一个实施例中，主服务器 DRS 遍历范围中的所有服务器，并且对每一从属服务器 DRS 进行点到点调用，例如 prepareTwoPhase(DID, oldVerNum, newVerNum, delta, timeout)。然后，从属服务器取得适当的超时值。在增量大时，例如包含二进制代码的增量，可以使用点到点协议。小一些的更新（例如，可能只包括诸如缓存大小的修改之类的次要配置改动）可以使用一阶段方法进行。可以使用该方法是因为：像应用程序增加的大改动以一致的方式到达服务器可能更加重要。可替换地，主服务器可能求助于族主服务器（如果存在的话），并且让族主服务器进行调用。让主服务器代理族主服务器可以提高系统可扩展性。

[0080] 在一个实施例中，每次向从属服务器或者族主服务器的调用产生四个响应中的一个，例如“不可达到”、“不同步”、“未确认”和“确认”，它们由主服务器 DRS 处理。如果响应为“不可达到”，则当前服务器不能达到，并且可能排队重试。如果响应为“不同步”，则该服务器可能排队重试。同时，该服务器将通过使用从主服务器的下拉而试图使自己同步，从而在重试时该服务器可以接收增量。如果响应为“未确认”，则中止该任务。当该服务器不能接受该任务时，可能给出该响应。如果响应是“确认”，则不采取行动。

[0081] 为了准备从属服务器，主服务器 DRS 可以调用诸如 prepareTwoPhase 的方法。当

从主服务器 DRS 收到“准备”请求时，从属服务器 DRS 可以先检查自己的当前版本号是否等于待更新的老版本号。如果不等，则该从属服务器可以返回“不同步”的响应。然后，该从属服务器可以从主服务器 DRS 下拉增量，就好像它刚刚收到心跳一样。最终，主服务器 DRS 可以重试 prepareTwoPhase。与让主服务器推送增量相比，该方法可以更加简单，但需要主服务器的仔细配置。可能需要主服务器的配置，因为等待响应时间太长可能使任务超时。另外，等待时间不足可能导致取得“不同步”响应的其他请求。更好地可能是在完成来自从属服务器的下拉请求时，触发重试。

[0082] 如果从属服务器同步，则在从属服务器一侧，该从属服务器可以向客户层尽可能深入该服务器地进行向上调用，例如 prepareTwoPhase(newVerNum,delta)。然后，将返回的结果“确认”或者“未确认”送到主服务器 DRS。如果响应为“确认”，则从属服务器可以进入特殊的已准备状态。如果响应为“未确认”，则从属服务器可以冲掉该更新的所有记录。如果以后由于某种原因要提交该更新，则该从属服务器可以以一阶段分发取得该更新，然后这可能失败。

[0083] 如果在超时期内主服务器 DRS 从每一服务器都取得了“确认”，则主服务器 DRS 可以进行提交向上调用，例如 twoPhaseSucceeded(newVerNum)，并且升级到新的版本号。如果主服务器 DRS 从任一服务器收到“未确认”，或者如果超时期过期，则主服务器 DRS 可以进行中止向上调用，例如 twoPhaseFailed(newVerNum,reason)，并且不改变版本号。此处，reason 为例外，包含所有“未确认”响应的累积。在这两种情况下，在随后的心跳中都可能包括中止 / 提交信息。

[0084] 在任意时间上，主服务器 DRS 都可以进行取消向下调用，例如 cancelTwoPhase(newVerNum)。然后，主服务器 DRS 可以通过以下方式处理带调用 :如果该任务没有正在进行，则抛出例外，或者好象要发生中止一样动作。

[0085] 如果已准备的从属服务器 DRS 获得表示新版本已经被提交的心跳，则从属服务器 DRS 可以进行向上调用，例如 commitTwophase(newVerNum)，并且升级到新的版本号。如果已准备的从属服务器 DRS 获得表示新版本已经被中止的心跳，则从属服务器 DRS 可以中止该任务。当从属服务器获得其中窗口已经超过新版本、从属服务器获得对同一数据项上的新 prepareTwoPhase 调用或者从属服务器对任务超时的心跳时，从属服务器也可以中止任务。在这样的情况下，从属服务器可以进行向上调用，例如 abortTwophase(newVerNum)，并且不改变版本号。对于（例如）在从属服务器已经准备之后，但在从属服务器提交之前主服务器出故障的情况下，这是一种确保正确处理的方法。

[0086] 提供本发明实施例的上述描述是为了展示与描绘的目的。这不是穷尽的也不是用来将本发明局限于所公开的精确形式。对于本领域的技术人员来讲，显然可以作出许多修改与变动。选择并描述这些实施例的目的是更好地解释本发明的原理与实际用途，由此使本领域技术人员能够理解本发明的种种实施例与适合于所考虑的特定用途的种种修改。本发明的范围由权利要求及其等价物所界定。

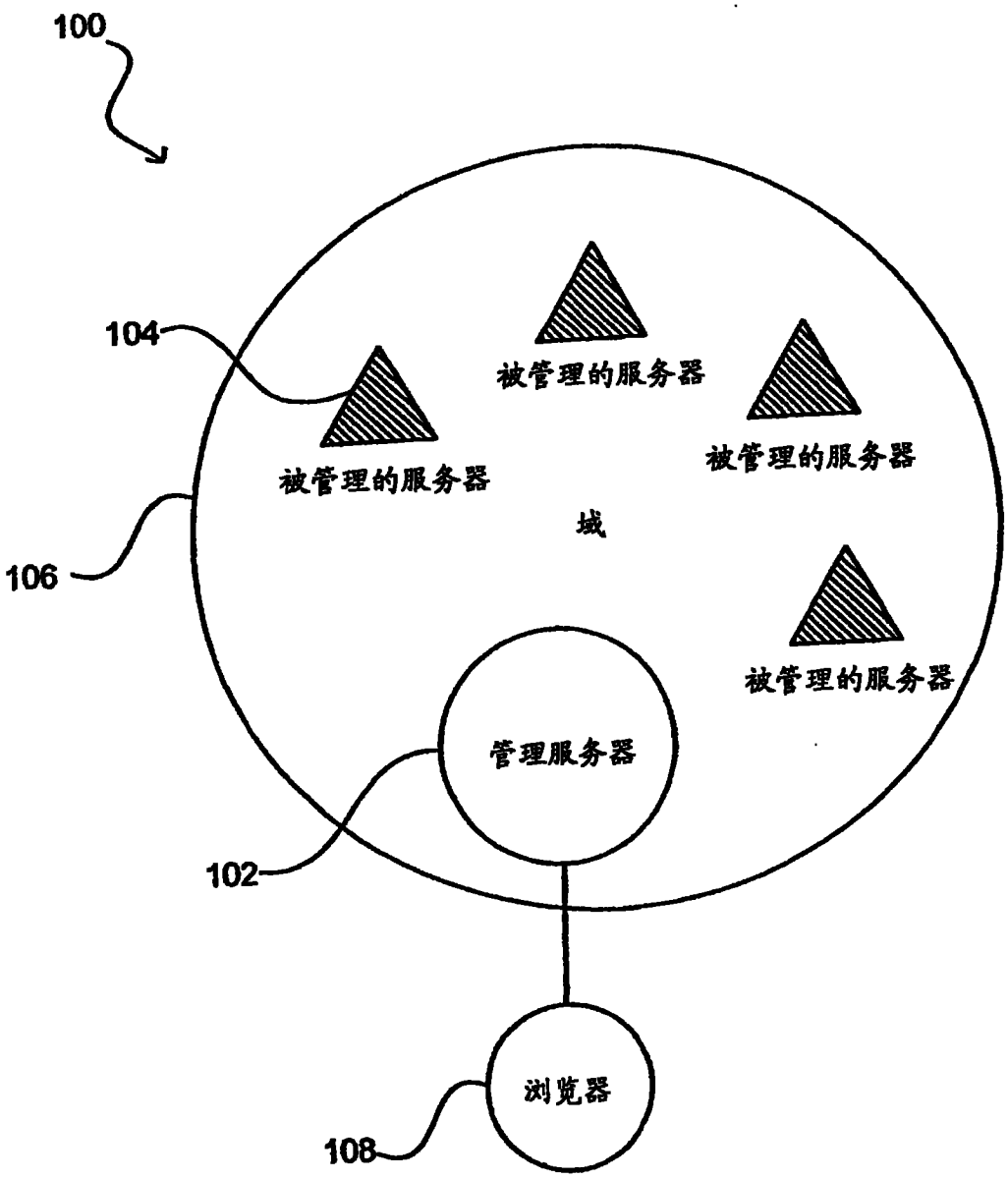

100
104
被管理的服务器
被管理的服务器
被管理的服务器
域
106
被管理的服务器
管理服务器
102
浏览器
108

图 1

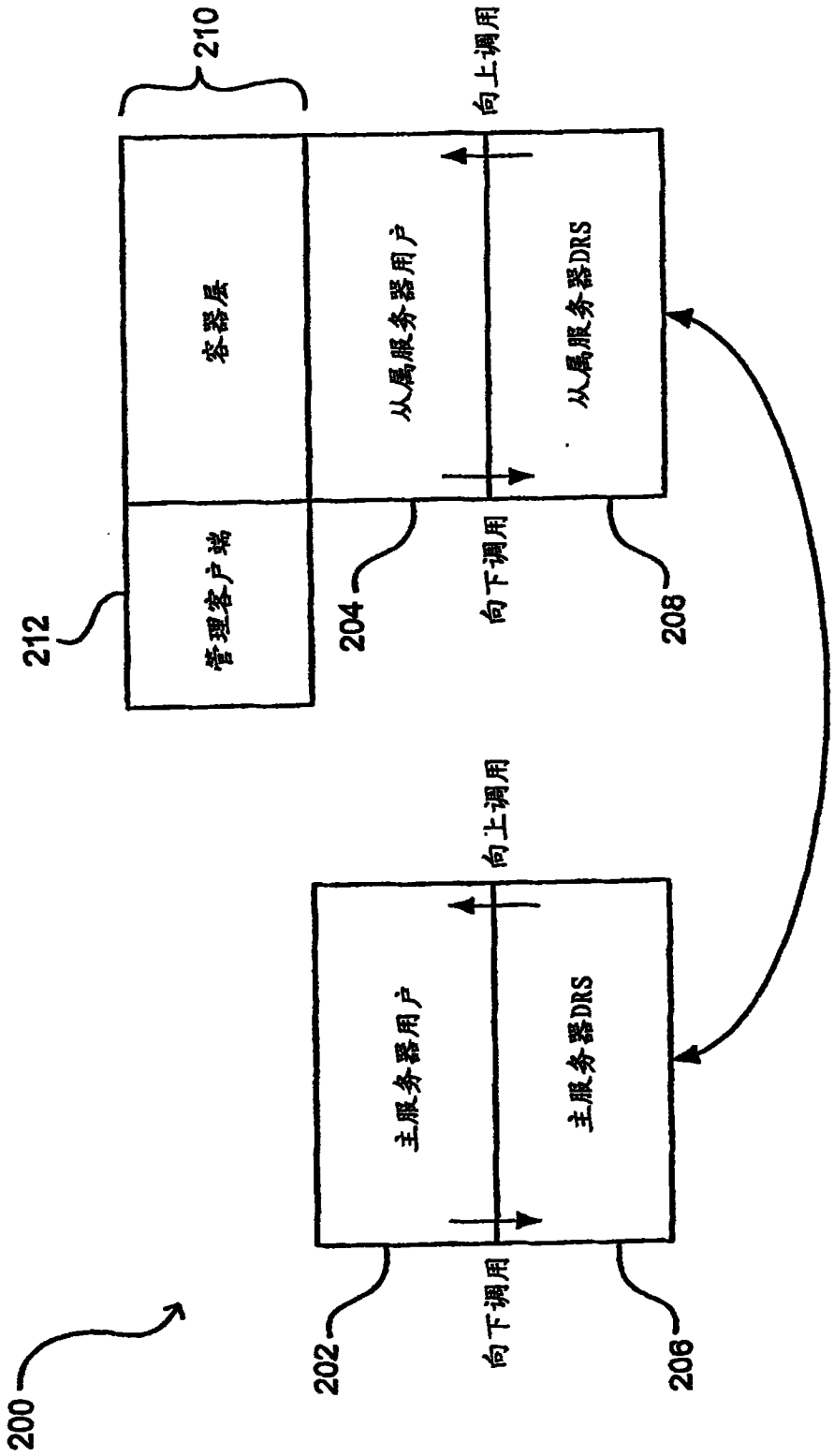

200
202
204
206
208
210
212
主服务器用户
主服务器DRS
从属服务器用户
从属服务器DRS
容器层
管理客户端
向上调用
向下调用
向上调用
向下调用

图 2

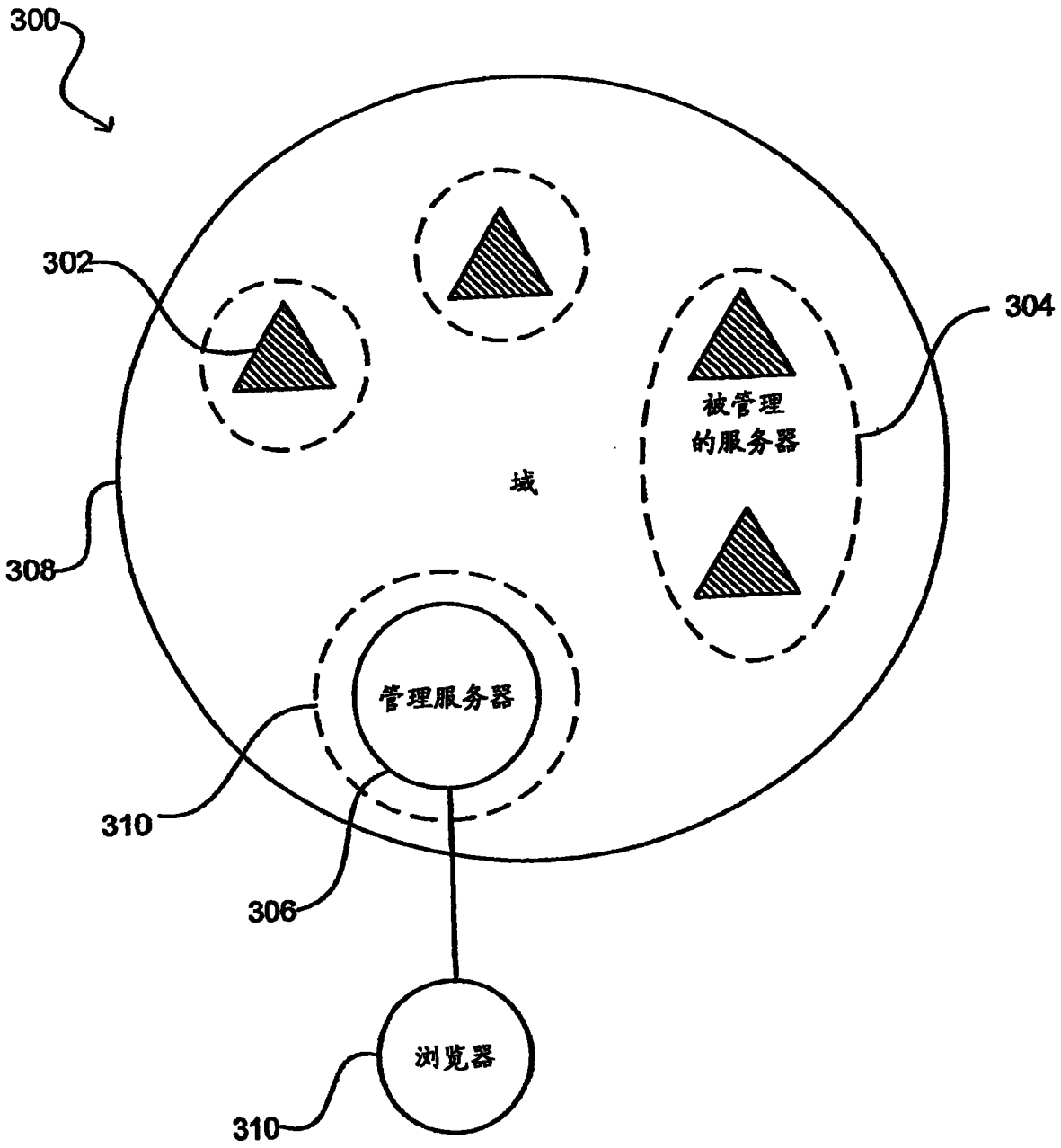


图 3

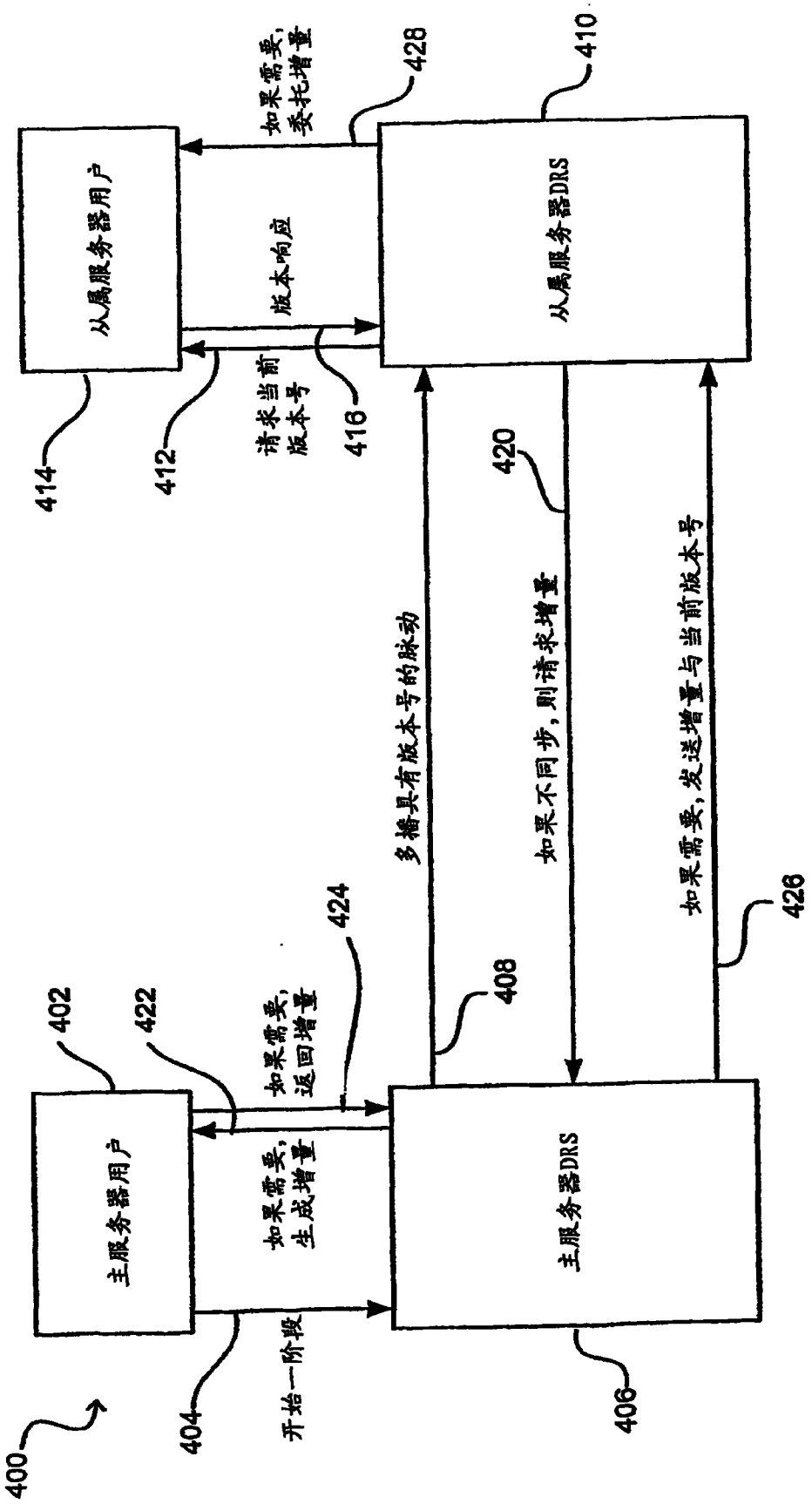

从属服务器用户
从属服务器DRS
主服务器用户
主服务器DRS
版本响应
如果需要，委托增量
请求当前版本号
多播具有版本号的脉动
如果不同步，则请求增量
如果需要，发送增量与当前版本号
如果需要，返回增量
如果需要，生成增量
开始一阶段
400
402
404
406
408
410
412
414
416
420
422
424
426
428

图 4

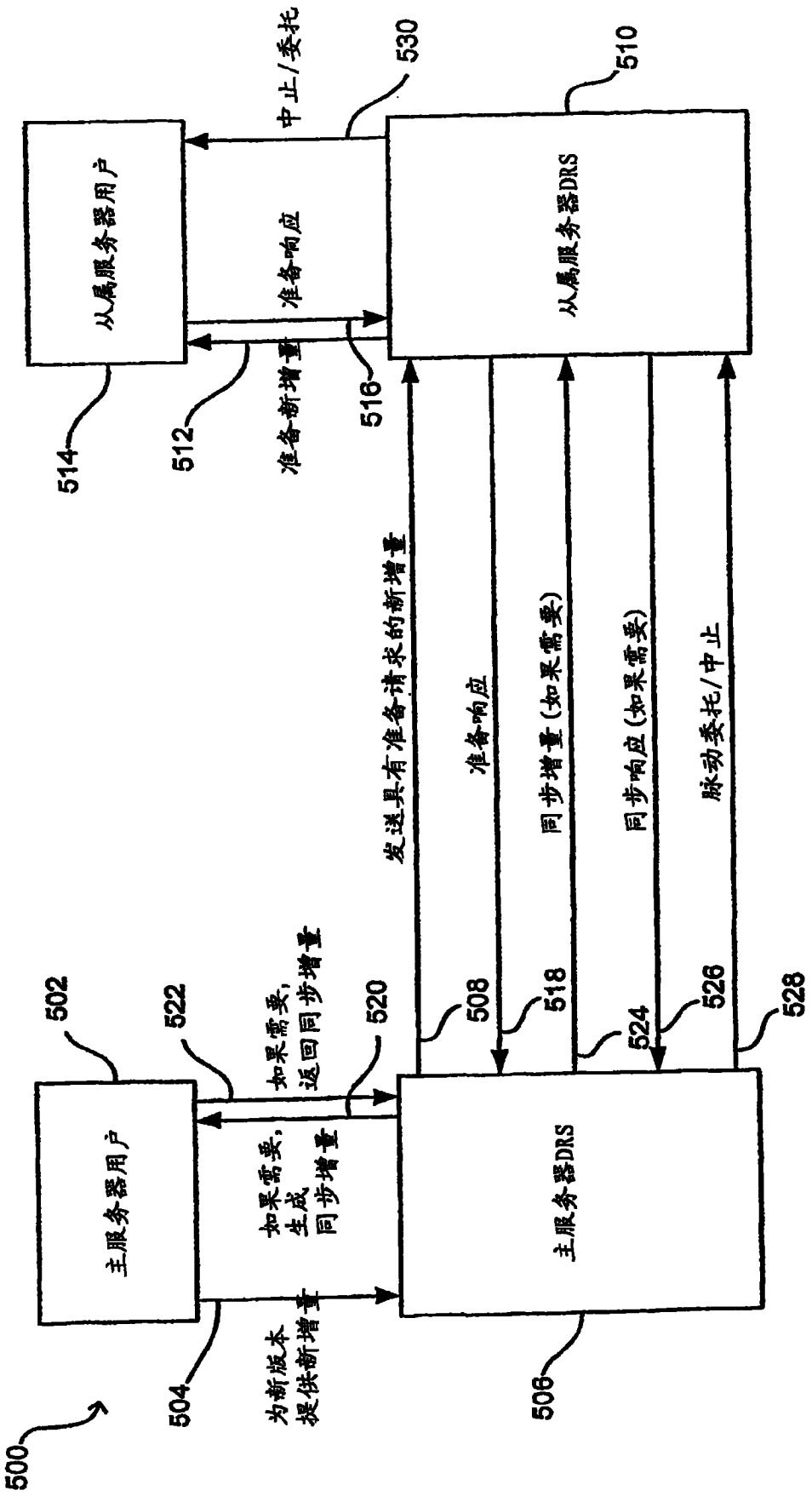

从属服务器用户
中止/委托
530
510
从属服务器DRS
准备响应
准备新增量
514
512
516
发送具有准备请求的新增量
准备响应
同步增量(如果需要)
同步响应(如果需要)
脉动委托/中止
502
522
如果需要,
返回同步增量
520
508
518
524
526
528
主服务器用户
如果需要,
生成
同步增量
主服务器DRS
为新版本
提供新增量
504
506
500

图 5

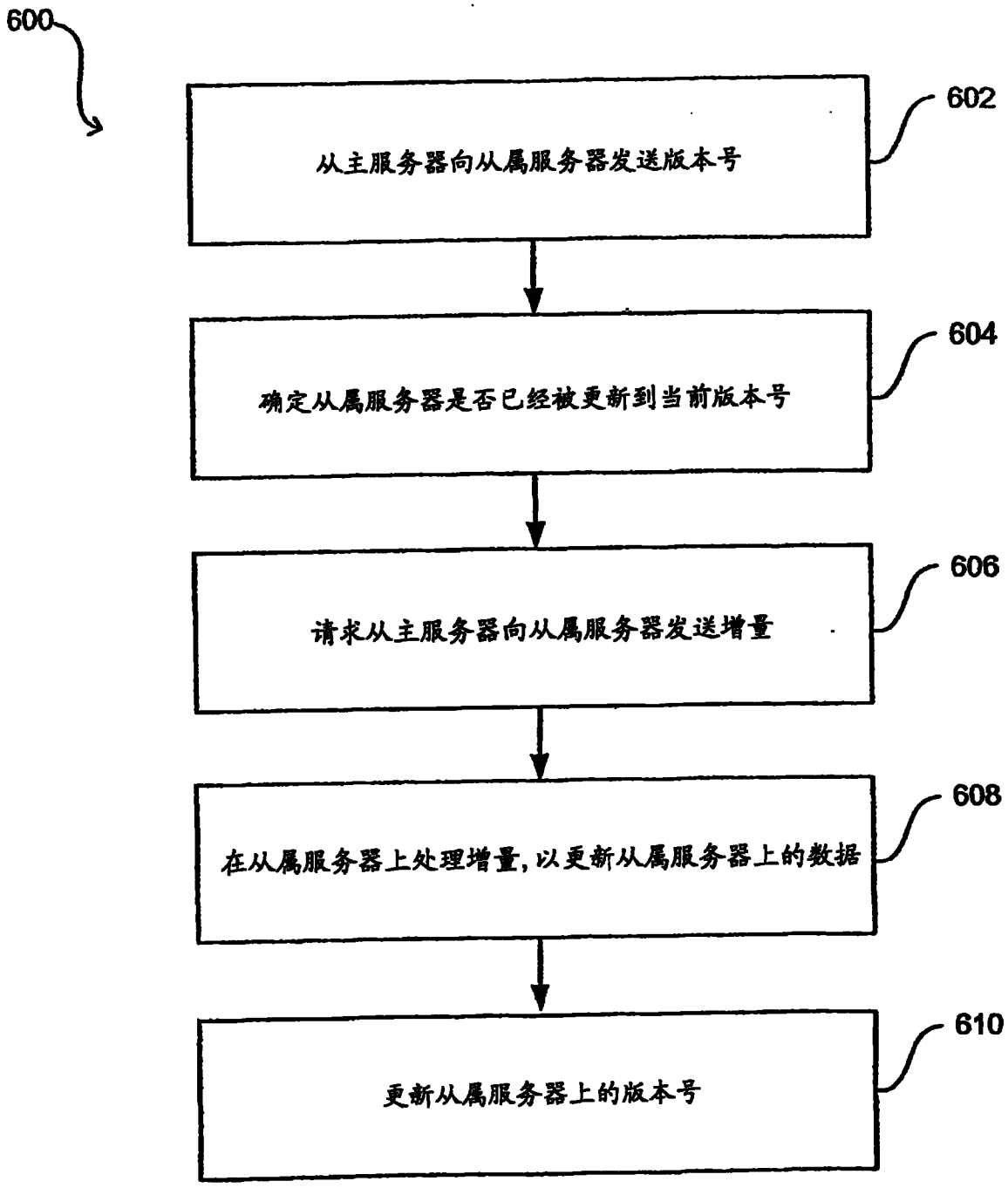

600
从主服务器向从属服务器发送版本号
602
确定从属服务器是否已经被更新到当前版本号
604
请求从主服务器向从属服务器发送增量
606
在从属服务器上处理增量，以更新从属服务器上的数据
608
更新从属服务器上的版本号
610

图 6

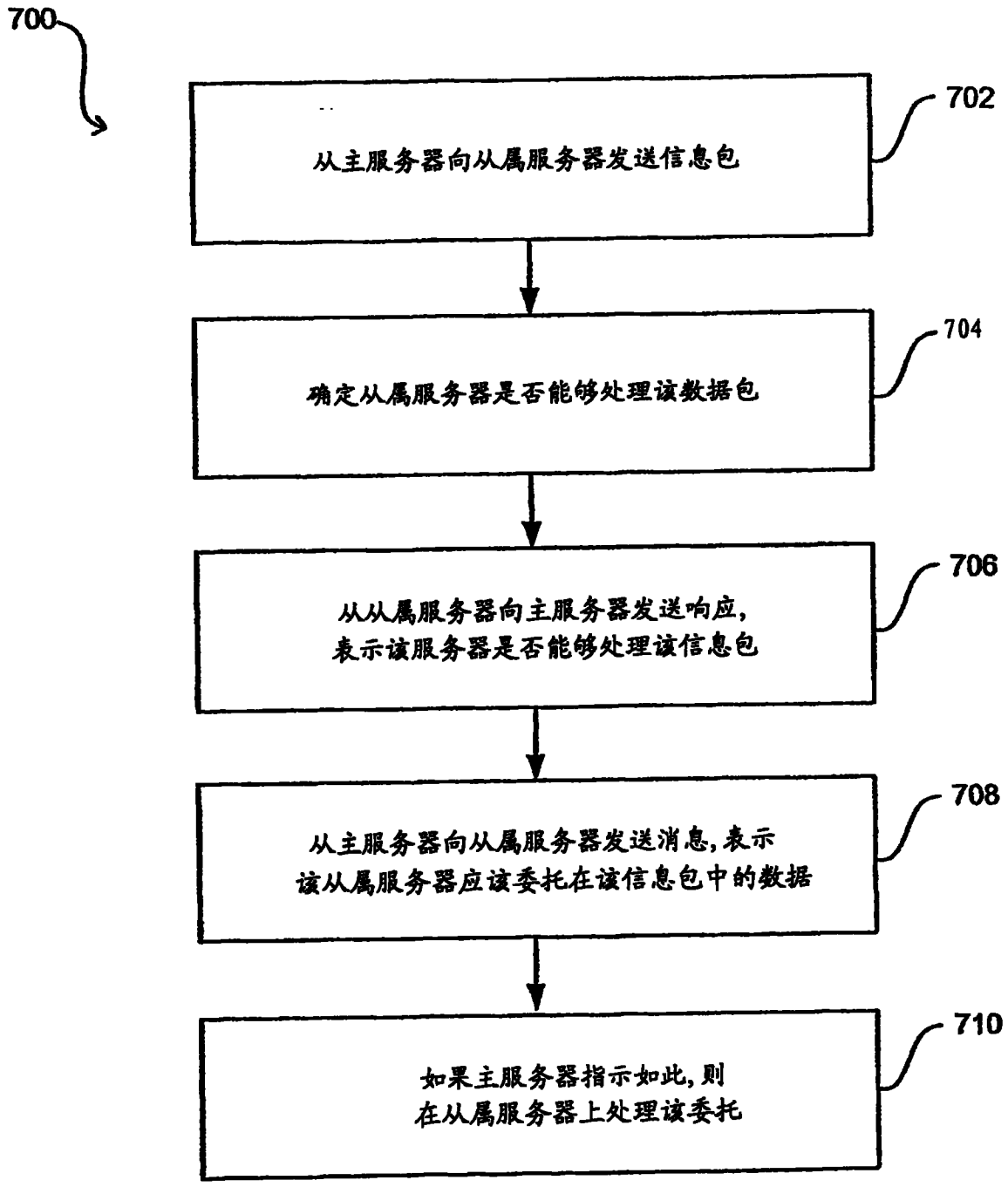

700
从主服务器向从属服务器发送信息包
702
确定从属服务器是否能够处理该数据包
704
从从属服务器向主服务器发送响应，
表示该服务器是否能够处理该信息包
706
从主服务器向从属服务器发送消息，表示
该从属服务器应该委托在该信息包中的数据
708
如果主服务器指示如此，则
在从属服务器上处理该委托
710

图 7