



(12)发明专利

(10)授权公告号 CN 106909436 B

(45)授权公告日 2020.07.21

(21)申请号 201510974631.8

(22)申请日 2015.12.23

(65)同一申请的已公布的文献号
申请公布号 CN 106909436 A

(43)申请公布日 2017.06.30

(73)专利权人 财团法人工业技术研究院
地址 中国台湾新竹县

(72)发明人 阙志克 林浩澄

(74)专利代理机构 北京市柳沈律师事务所
11105

代理人 王珊珊

(51)Int.Cl.

G06F 9/455(2006.01)

G06F 9/54(2006.01)

(56)对比文件

US 2012158867 A1,2012.06.21,

US 2013159999 A1,2013.06.20,

CN 104253860 A,2014.12.31,

CN 104715201 A,2015.06.17,

CN 1790270 A,2006.06.21,

CN 103647695 A,2014.03.19,

审查员 刘津

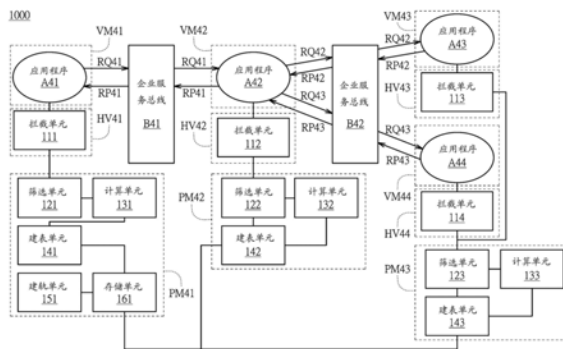
权利要求书2页 说明书6页 附图6页

(54)发明名称

产生虚拟机消息队列应用程序的相关关系的方法与系统

(57)摘要

本发明提供一种产生虚拟机消息队列应用程序的相关关系的方法与系统。此方法包括以下步骤。拦截多个应用程序通过至少一企业服务总线传递的至少一传递消息。此至少一传递消息包括至少一请求消息。筛选此至少一传递消息,以筛选出此至少一请求消息,并获得各个请求消息的客户端及服务器;计算各个请求消息的校验和,以获得各个请求消息的消息编号。将各个请求消息的客户端、服务器及消息编号记录于相关关系表。



1. 一种产生虚拟机消息队列应用程序的相关关系的方法,包括:
拦截多个应用程序通过至少一企业服务总线传递的至少一传递消息,该至少一传递消息为一请求消息或一回复消息;
判断拦截的各该传递消息是该请求消息或该回复消息;
筛选掉该回复消息,并保留该请求消息,并分别获得该请求消息的客户端及服务器;
分别计算该请求消息的校验和,以分别获得该请求消息的消息编号;以及
分别将该请求消息的该客户端、该服务器及该消息编号记录于相关关系表。
2. 如权利要求1所述的方法,还包括:
依据该相关关系表,建立应用程序轨迹。
3. 如权利要求2所述的方法,其中在依据该相关关系表,建立该应用程序轨迹的步骤中,该应用程序轨迹以树状结构建立。
4. 如权利要求3所述的方法,其中在依据该相关关系表,建立该应用程序轨迹的步骤中,该树状结构以深度优先搜寻法或广度优先搜寻法建立。
5. 如权利要求1所述的方法,其中在拦截所述应用程序通过该至少一企业服务总线传递的该至少一传递消息的步骤中,该至少一请求消息通过虚拟机管理器的拦截单元进行拦截。
6. 如权利要求1所述的方法,其中在拦截所述应用程序通过该至少一企业服务总线传递的该至少一传递消息的步骤中,该至少一请求消息通过监控多个线程的方式进行拦截。
7. 如权利要求1所述的方法,其中在筛选掉该回复消息,并保留该请求消息的步骤中,通过该至少一传递消息的至少一时间信息,筛选掉该回复消息,并保留该请求消息。
8. 如权利要求1所述的方法,其中在分别计算该请求消息的校验和的步骤中,该请求消息分别包括固定信息及变动信息,该请求消息分别以该固定信息进行计算,以分别获得该校验和。
9. 如权利要求1所述的方法,其中在分别将该请求消息的该客户端、该服务器及该消息编号记录于该相关关系表的步骤中,所有的该至少一请求消息均记录于同一相关关系表中。
10. 如权利要求1所述的方法,其中在分别将该请求消息的该客户端、该服务器及该消息编号记录于该相关关系表的步骤中,该相关关系表还包括该至少一请求消息的至少一时间信息。
11. 如权利要求1所述的方法,其中
在分别将该请求消息的该客户端、该服务器及该消息编号记录于该相关关系表的步骤中,还分别将该回复消息的时间信息及消息大小记录于该相关关系表。
12. 一种产生虚拟机消息队列应用程序的相关关系的系统,包括:
拦截单元,拦截多个应用程序通过至少一企业服务总线传递的至少一传递消息,该至少一传递消息为一请求消息或一回复消息;
筛选单元,判断拦截的各该传递消息是该请求消息或该回复消息,筛选该回复消息,并保留该请求消息,并分别获得该请求消息的客户端及服务器;
计算单元,分别计算该请求消息的校验和,以分别获得该请求消息的消息编号;以及
建表单元,分别将该请求消息的该客户端、该服务器及该消息编号记录于相关关系表。

13. 如权利要求12所述的系统,还包括:
建轨单元,依据该相关关系表,建立应用程序轨迹。
14. 如权利要求13所述的系统,其中该建轨单元以树状结构建立该应用程序轨迹。
15. 如权利要求14所述的系统,其中该建轨单元以深度优先搜寻法或广度优先搜寻法建立该树状结构。
16. 如权利要求12所述的系统,其中该拦截单元设置于虚拟机管理器。
17. 如权利要求12所述的系统,其中该拦截单元通过监控多个线程的方式拦截该至少一请求消息。
18. 如权利要求12所述的系统,其中该筛选单元通过该至少一传递消息的至少一时间信息,筛选掉该回复消息,并保留该请求消息。
19. 如权利要求12所述的系统,其中该请求消息分别包括固定信息及变动信息,该计算单元分别以该固定信息进行计算,以分别获得该校验和。
20. 如权利要求12所述的系统,其中该建表单元将所有的该至少一请求消息均记录于同一该相关关系表中。
21. 如权利要求12所述的系统,其中该建表单元还记录该至少一请求消息的至少一时间信息于该相关关系表。
22. 如权利要求12所述的系统,其中该拦截单元、该筛选单元、该计算单元及该建表单元设置于实体机器中。
23. 如权利要求12所述的系统,其中该建表单元还分别将该至少一回复消息的时间信息及消息大小记录于该相关关系表。

产生虚拟机消息队列应用程序的相关关系的方法与系统

技术领域

[0001] 本发明涉及一种产生虚拟机应用程序的相关关系的方法与系统,且特别是涉及一种产生虚拟机消息队列应用程序的相关关系的方法与系统。

背景技术

[0002] 企业服务通常是由大量的服务器(service server)及网络所组成。随着网络成长与虚拟机技术的发展,更多的用户级的应用程序(user application)因为云端计算(cloud computing),从个人计算机转移到虚拟数据中心(virtual data center,VDC),终端用户(end-user)通过请求数据中心的服务器所提供的服务,来使用远程的应用程序(remote application)。

[0003] 此外,分布式的应用程序变得性能更强也更复杂。单一服务或单一网络组件的失效(failure)或性能降低(performance degradation)可能降低企业网络服务的质量及客户满意度。由于负载平衡及高利用性(high availability)的设计要求,应用程序的相关性在分布式系统中会动态改变,要找出这些失效的问题点对人们来说是相当困难的。所以,在多个应用程序之间如何自动找出相关性已成为相当重要的议题。

[0004] 许多企业使用消息队列(queue based)应用程序来提供服务。其中部分企业使用企业服务总线(enterprise service bus,ESB)架构来传递消息。

[0005] 随着消息队列应用程序已广泛应用于虚拟机环境,企业需要一个有系统、低成本、简单的方法来管理性能并进行除错(trouble-shooting)。

发明内容

[0006] 本发明涉及一种产生虚拟机消息队列应用程序的相关关系的方法与系统。

[0007] 根据本发明公开内容的一实施例,提出一种产生虚拟机(virtual machine)消息队列(queue based)应用程序的相关关系的方法。此方法包括以下步骤。拦截多个应用程序通过至少一企业服务总线(enterprise service bus)传递的至少一传递消息(transmitting message)。此至少一传递消息包括至少一请求消息(request message)。筛选此至少一传递消息,以筛选出此至少一请求消息,并获得各个请求消息的客户端(client)及服务器(server)。计算各个请求消息的校验和(checksum),以获得各个请求消息的消息编号。将各个请求消息的客户端、服务器及消息编号记录于相关关系表(dependencies table)。

[0008] 根据本发明公开内容的另一实施例,提出一种产生虚拟机(virtual machine)消息队列(queue based)应用程序的相关关系的系统。此系统包括拦截单元、筛选单元、计算单元及建表单元。拦截单元拦截多个应用程序通过至少一企业服务总线(enterprise service bus)传递的至少一传递消息(transmitting message)。此至少一传递消息包括至少一请求消息(request message)。筛选单元筛选此至少一传递消息,以筛选出此至少一请求消息,并获得各个请求消息的客户端(client)及服务器(server)。计算单元计算各个请

求消息的校验和 (checksum), 以获得各个请求消息的消息编号。建表单元将各个请求消息的客户端、服务器及消息编号记录于相关关系表 (dependencies table)。

附图说明

[0009] 为了对本发明的上述及其他方面有更佳的了解, 下文特举若干实施例, 并配合附图, 作详细说明如下:

[0010] 图1绘示一实施例的1:1的应用程序与企业服务总线 (enterprise service bus, ESB) 的关系图。

[0011] 图2绘示一实施例的1:1:1的应用程序与企业服务总线的关系图。

[0012] 图3绘示一实施例的1:2的应用程序与企业服务总线的关系图。

[0013] 图4A~4B绘示产生虚拟机消息队列应用程序的相关关系的系统的一实施例示意图。

[0014] 图5绘示产生虚拟机消息队列应用程序的相关关系的方法的一实施例流程图。

[0015] 图6绘示图4A~4B实施例的应用程序轨迹的树状结构的示意图。

[0016] 图7绘示应用程序的一实施例示意图。

[0017] 图8绘示图7实施例的应用程序轨迹的树状结构的示意图。

[0018] 附图标记说明

[0019] 1000、2000: 系统

[0020] 111、112、113、114: 拦截单元

[0021] 121、122、123: 筛选单元

[0022] 131、132、133: 计算单元

[0023] 141、142、143: 建表单元

[0024] 151: 建轨单元

[0025] 161: 存储单元

[0026] A11、A12、A21、A22、A23、A31、A32、A33、A41、A42、A43、A44、A71、A72、A73、A74、A75: 应用程序

[0027] B11、B21、B22、B31、B41、B42、B71、B72: 企业服务总线

[0028] HV41~HV44: 虚拟机管理器

[0029] PM40、PM41、PM41'、PM42、PM43: 实体机器

[0030] RQ11、RQ21、RQ22、RQ31、RQ32、RQ41、RQ42、RQ43、RQ71、RQ72: 请求消息

[0031] RP11、RP21、RP22、RP31、RP32、RP41、RP42、RP43: 回复消息

[0032] S501、S502、S503、S504、S505: 流程步骤

[0033] VM41、VM42、VM43、VM44: 虚拟机

具体实施方式

[0034] 在本发明中, 通过拦截请求消息 (request message) 及计算请求消息的校验和 (checksum) 的方式, 产生虚拟机 (virtual machine) 消息队列 (queue based) 应用程序 (applications) 的相关关系, 使得系统能够有效进行性能管理与除错。

[0035] 请参照图1, 其绘示一实施例的1:1的应用程序A11、A12与企业服务总线

(enterprise service bus,ESB)B11的关系图。应用程序A11发出请求消息RQ11至企业服务总线B11后,应用程序A12自企业服务总线B11取得此请求消息RQ11。对请求消息RQ11来说,应用程序A11为客户端(client),应用程序A12为服务器(server)。另外,应用程序A12回传回复消息RP11至企业服务总线B11后,应用程序A11自企业服务总线B11取得此回复消息RP11。

[0036] 请参照图2,其绘示一实施例的1:1:1的应用程序A21、A22、A23与企业服务总线B21、B22的关系图。应用程序A21发出请求消息RQ21至企业服务总线B21后,应用程序A22自企业服务总线B21取得此请求消息RQ21。对请求消息RQ21来说,应用程序A21为客户端,应用程序A22为服务器。应用程序A22发出请求消息RQ22至企业服务总线B22后,应用程序A23自企业服务总线B22取得此请求消息RQ22。对请求消息RQ22来说,应用程序A22为客户端,应用程序A23为服务器。另外,应用程序A23回传回复消息RP22至企业服务总线B22后,应用程序A22自企业服务总线B22取得此回复消息RP22。应用程序A22回传回复消息RP21至企业服务总线B21后,应用程序A21自企业服务总线B21取得此回复消息RP21。在另一实施例中,图2也可继续扩展为1:1:1:⋯:1。

[0037] 请参照图3,其绘示一实施例的1:2的应用程序A31、A32、A33与企业服务总线B31的关系图。应用程序A31发出请求消息RQ31及请求消息RQ32至企业服务总线B31后,应用程序A32自企业服务总线B31取得请求消息RQ31,应用程序A33自企业服务总线B31取得请求消息RQ32。对请求消息RQ31来说,应用程序A31为客户端,应用程序A32为服务器。对请求消息RQ32来说,应用程序A32为客户端,应用程序A33为服务器。另外,应用程序A32回传回复消息RP31至企业服务总线B31后,应用程序A31自企业服务总线B31取得此回复消息RP31。应用程序A33回传回复消息RP32至企业服务总线B31后,应用程序A31自企业服务总线B31取得此回复消息RP32。在另一实施例中,图3也可继续扩展为1:n.n为自然数。

[0038] 以上三种关系可能交错应用,而形成相当复杂的网络。例如是1:1:n。如此一来,这些应用程序的相关关系变的相当复杂,而不容易追踪。

[0039] 请参照图4A~4B,其绘示产生虚拟机消息队列应用程序的相关关系的系统1000、2000的一实施例示意图。在图4A的一实施例中,系统1000包括多个虚拟机VM41~VM44、多个虚拟机管理器(hypervisor)HV41~HV44及多个实体机器PM41~PM43。多个应用程序A41~A44分别架构于虚拟机VM41~VM44上。多个拦截单元111~114分别架构于虚拟机管理器HV41~HV44上。实体机器PM41包括筛选单元121、计算单元131、建表单元141、建轨单元151及存储单元161。实体机器PM42包括筛选单元122、计算单元132及建表单元142。实体机器PM43包括筛选单元123、计算单元133及建表单元143。

[0040] 拦截单元111~114可以拦截消息。筛选单元121~123可以筛选消息,计算单元131~133可以进行各种计算程序,建表单元141~143可以建立或更新表格。建轨单元151可以追踪轨迹。存储单元161可以存储各种数据。拦截单元111~114、筛选单元121~123、计算单元131~133、建表单元141~143及建轨单元151例如是(但不限于)电路、芯片、电路板或存储阵列程序代码的记录媒体。存储单元161例如是(但不限于)硬盘、存储器、便携式存储媒体、或云端存储中心。

[0041] 在图4A的实施例中,多个虚拟机VM41~VM44架构于多个实体机器PM41~PM43上。在另一实施例中,多个虚拟机可以架构于单实体机器上。

[0042] 此外,在图4A的实施例中,存储单元161及建轨单元151架构于实体机器PM41中,其余的实体机器PM42~PM43联机至实体机器PM41,以联系存储单元161及建轨单元151。在图4B的实施例中,系统2000包括实体机器PM40、PM41'、PM42、PM43。存储单元161及建轨单元151架构于作为存储服务器的实体机器PM40上。所有的实体机器PM41'、PM42、PM43联机至实体机器PM40,以联系存储单元161及建轨单元151。

[0043] 请参照图5,其绘示产生虚拟机消息队列应用程序的相关关系的方法的一实施流程图。以下流程图的说明以图4A的系统1000为例。在步骤S501中,拦截单元拦截多个应用程序通过至少一企业服务总线传递的至少一传递消息(transmitting message)。举例来说,拦截单元111~114拦截应用程序A41~A44通过企业服务总线B41~B42传递的多个传递消息。这些传递消息包括请求消息(request message)RQ41~RQ43及回复消息(reply message)RP41~RP43。在此步骤中,拦截单元111~114并未分辨哪一个是请求消息哪一个是回复消息,而是直接将所有传递消息均拦截下来,并记录时间信息。

[0044] 在此步骤中,此请求消息RQ41~RQ43及回复消息RP41~RP43通过监控多个线程(running thread)的方式进行拦截。如此一来,可以精准地追踪出服务层级协议(Service Level Agreement,SLA)失效的根本原因(root cause)。

[0045] 在步骤S502中,筛选单元筛选此至少一传递消息,以筛选出此至少一请求消息,并获得各个请求消息的客户端(client)及服务器(server)。举例来说,筛选单元121~123筛选传递消息,以筛选出请求消息RQ41~RQ43,并获得各个请求消息RQ41~RQ43的客户端及服务器。在此步骤中,筛选单元121~123观察通过传递消息的时间信息。在两个应用程序之间一来一往的两笔传递消息中,将较早传输的传递消息视为请求消息,将较晚传输的传输消息视为回复消息。

[0046] 筛选单元121~123分辨出请求消息RQ41~RQ43后,在后续步骤则忽略回复消息RP41~RP43。

[0047] 接着,在步骤S503中,计算单元计算各个请求消息的校验和(checksum),以获得各个消息的消息编号。举例来说,计算单元131~133计算各个请求消息RQ41~RQ43的校验和(checksum),以获得各个请求消息RQ41~RQ43的消息编号(例如是“RQ41key~RQ43key”)。在此步骤中,各个请求消息RQ41~RQ43包括固定信息及变动信息,计算单元131~133以各个固定信息进行计算,以获得各个校验和。如此一来,即使变动信息改变,同一请求消息仍可被计算出一致的校验和。在一实施例中,校验和的计算方式可以是加、减、乘、除、异或(XOR)等运算,或是以上任二者或二者以上的组合运算等。

[0048] 接着,在步骤S504中,建表单元将各个请求消息的客户端、服务器及消息编号记录于相关关系表(dependencies table)。举例来说,建表单元141~143将各个请求消息RQ41~RQ43的客户端、服务器及消息编号记录于相关关系表。请参照表一,其绘示相关关系表的实施例。在请求消息RQ41中,应用程序A41为客户端,应用程序A42为服务器,因此请求消息RQ41的消息编号RQ41key记录于应用程序A41的传送字段及应用程序A42的接收字段。在请求消息RQ42中,应用程序A42为客户端,应用程序A43为服务器,因此请求消息RQ42的消息编号RQ42key记录于应用程序A42的传送字段及应用程序A43的接收字段。在请求消息RQ43中,应用程序A42为客户端,应用程序A44为服务器,请求消息RQ43的消息编号RQ43key记录于应用程序A42的传送字段及应用程序A44的接收字段。

	接收	应用程序	传送	时间信息
[0049]		A41	RQ41key	...
	RQ41key	A42	RQ42key RQ43key	...
[0050]	RQ42key	A43		...
	RQ43key	A44		...

[0051] 表一

[0052] 在一实施例中,建表单元141~143将所有的请求消息RQ41~RQ43均记录于同一相关关系表中。在一实施例中,此相关关系表可以存储于如图4A的实体机器PM41的存储单元161中,实体机器PM42的建表单元142及实体机器PM43的建表单元143连接至实体机器PM41的存储单元161来更新相关关系表。在另一实施例中,此相关关系表可以存储于如图4B的实体机器PM40的存储单元161中。

[0053] 在一实施例中,在此步骤中,相关关系表还可以包括此至少一请求消息的至少一时间信息。如此一来,可以通过时间信息了解运算的瓶颈,以利进行改善。

[0054] 然后,在步骤S505中,建轨单元依据相关关系表,建立应用程序轨迹(application trajectory)。举例来说,建轨单元151依据相关关系表,建立应用程序轨迹。请参照图6,其绘示图4A~4B实施例的应用程序轨迹的树状结构的示意图。建轨单元151可利用深度优先搜寻法(Depth-first search,DFS)或广度优先搜寻法(Breadth-first Search,BFS),以树状结构建立应用程序轨迹。如此一来,方便用户清楚了解应用程序的相关关系。

[0055] 以下还以另一实施例说明相关关系表及树状结构的建立。请参照第7~8图,图7绘示应用程序A71~A75的一实施例示意图,图8绘示图7实施例的应用程序轨迹的树状结构的示意图。如图7所示,应用程序A71发出请求消息RQ71至企业服务总线B71后,应用程序A72自企业服务总线B71取得请求消息RQ71。对请求消息RQ71来说,应用程序A71为客户端,应用程序A72为服务器。应用程序A72发出请求消息RQ72至企业服务总线B72后,应用程序A73、应用程序A74、应用程序A75自企业服务总线B72取得请求消息RQ72。对请求消息RQ72来说,应用程序A72为客户端,各个应用程序A73、应用程序A74及应用程序A75为服务器。

[0056] 请参照表二,其绘示另一相关关系表的实施例。在请求消息RQ71中,应用程序A71为客户端,应用程序A72为服务器,因此请求消息RQ71的消息编号RQ71key记录于应用程序A71的传送字段及应用程序A72的接收字段。在请求消息RQ72中,应用程序A72为客户端,应用程序A73、应用程序A74、应用程序A75为服务器,因此请求消息RQ72的消息编号RQ72key记录于应用程序A72的传送字段及应用程序A73、应用程序A74、应用程序A75的接收字段。

[0057]

接收	应用程序	传送	时间信息
	A71	RQ71key	...
RQ71key	A72	RQ72key	...
RQ72key	A73		...
RQ72key	A74		...
RQ72key	A75		...

[0058] 表二

[0059] 如图8所示,上述表二的应用程序轨迹可以树状结构来表示,方便用户清楚了解应用程序的相关关系。

[0060] 在一实施例中,在步骤S502中,筛选单元121、122、123可以进一步筛选出回复消息RP41、RP42、RP43。并且,在步骤S504中,建表单元141、142、143将各个回复消息RP41、RP42、RP43的时间信息及消息大小等相关信息记录于相关关系表。

[0061] 综上所述,虽然已以实施例公开如上,然其并非用以限定本发明。本领域技术人员在不脱离本发明的精神和范围的情况下,可作各种的更动与润饰。因此,本发明的保护范围以所附的权利要求为准。

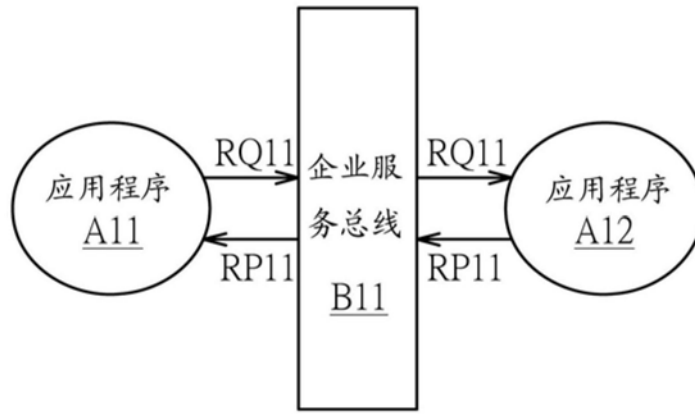


图1

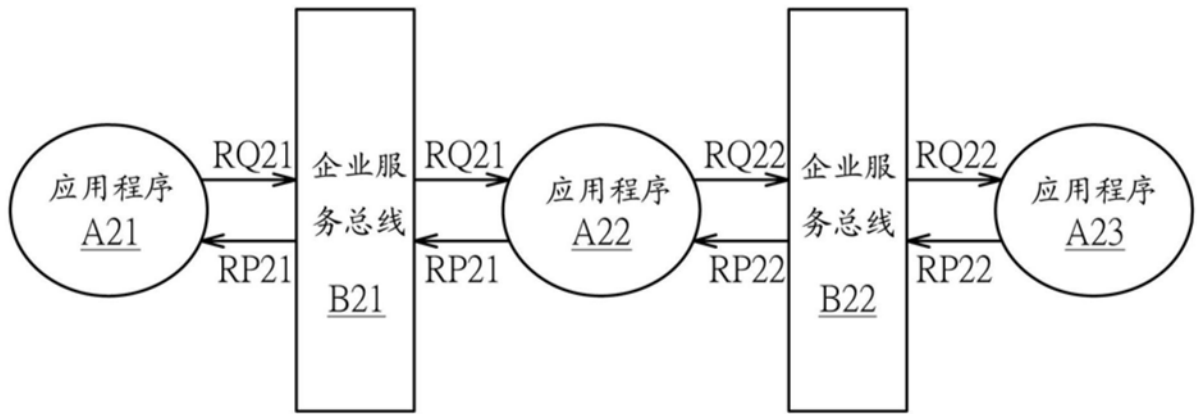


图2

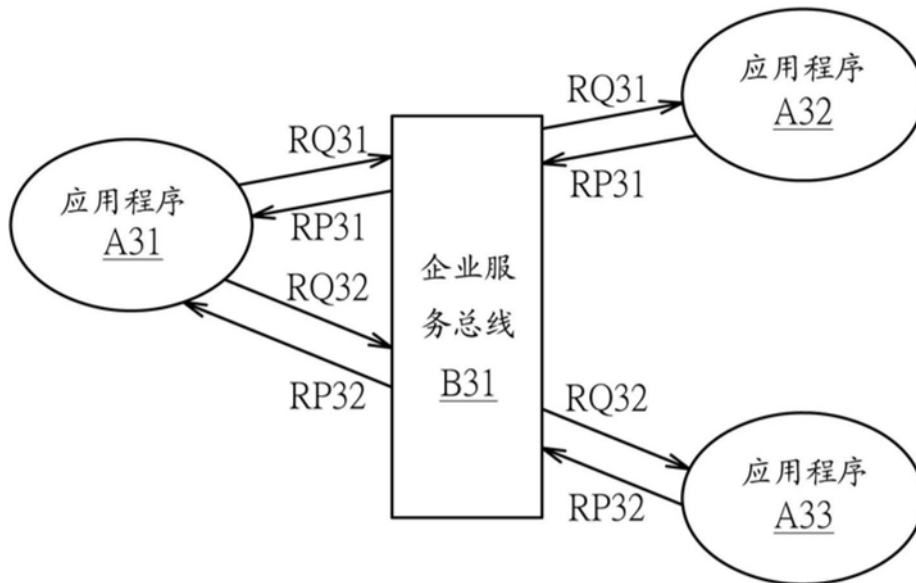
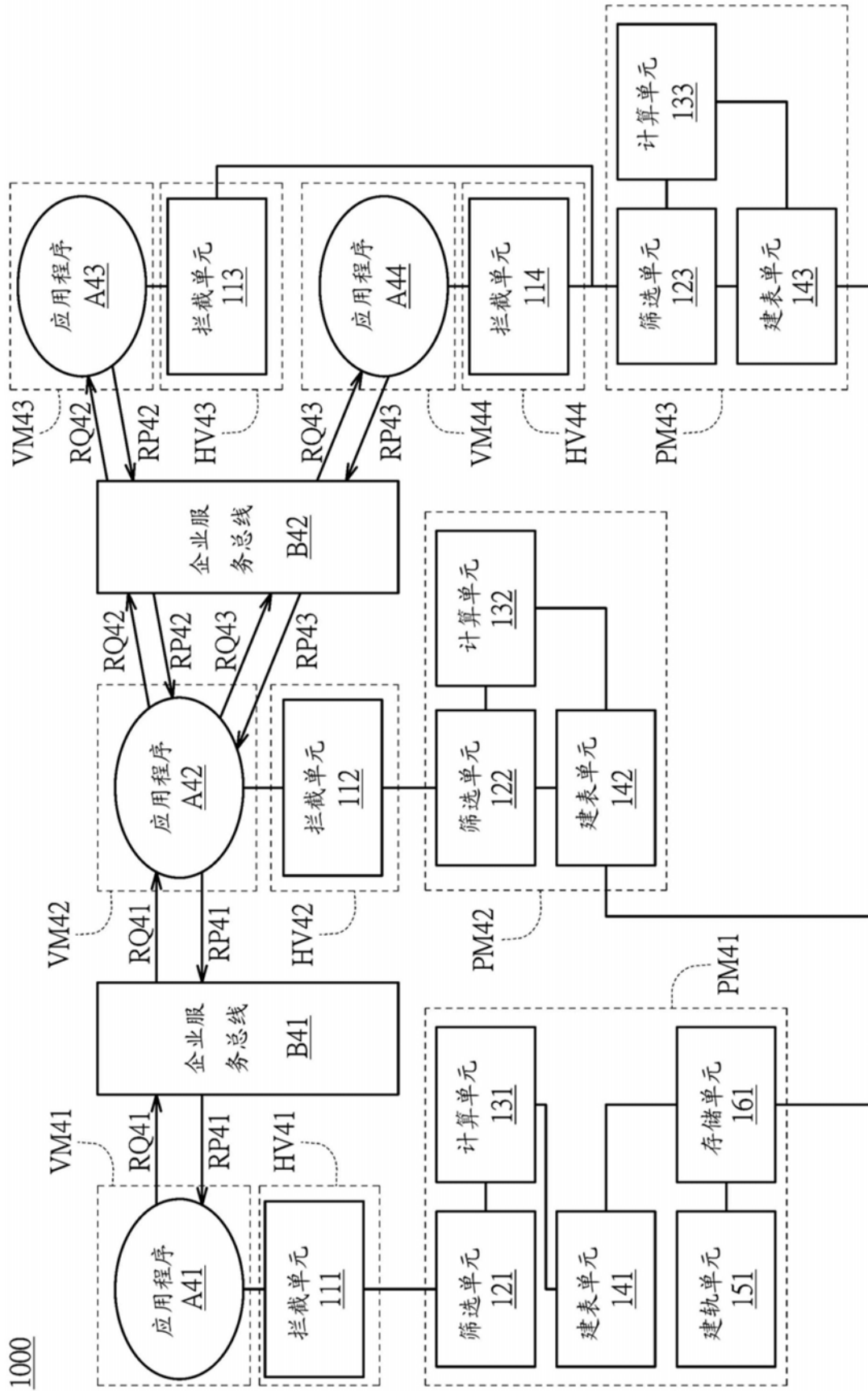
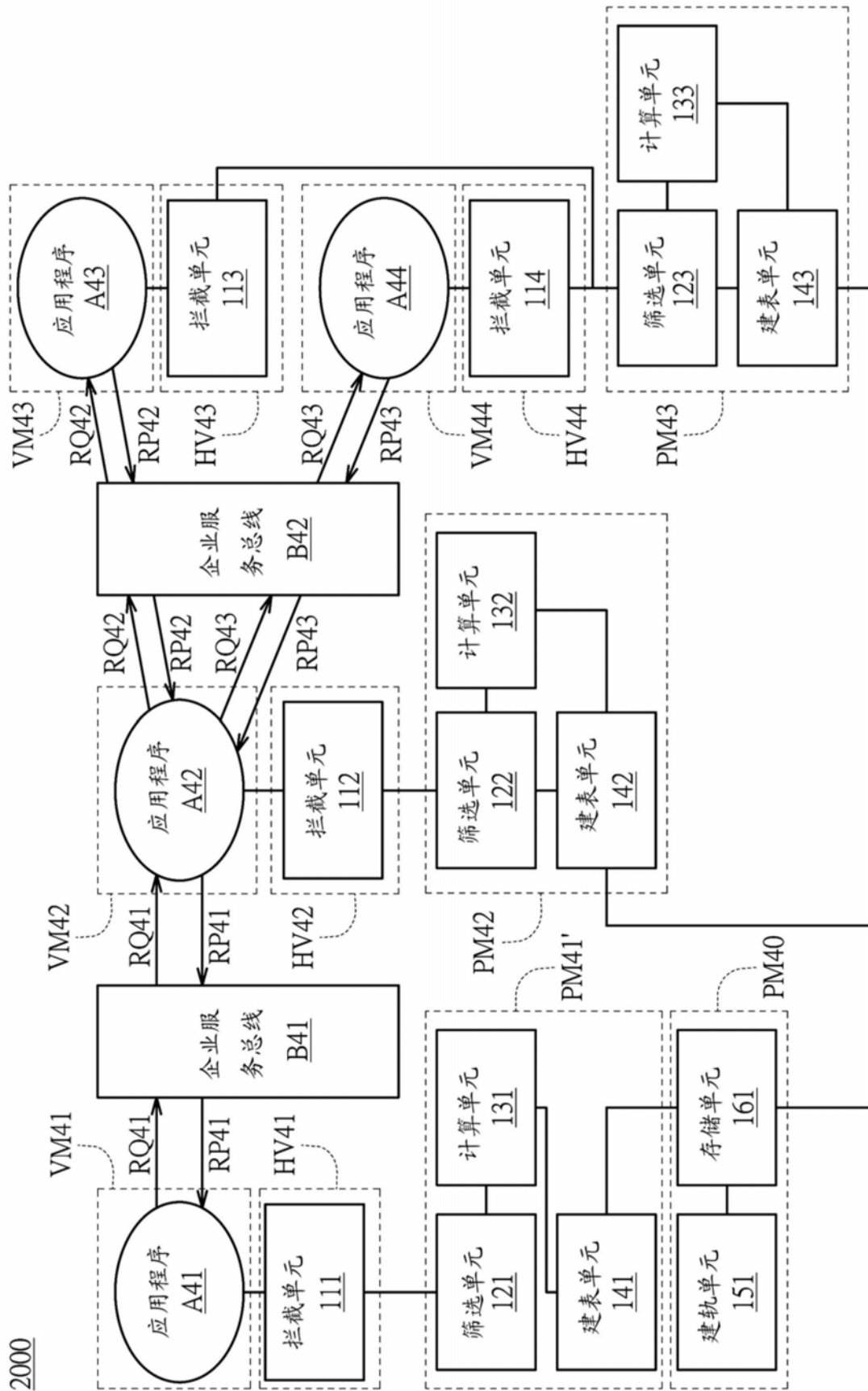


图3



1000

图4A



2000

图4B

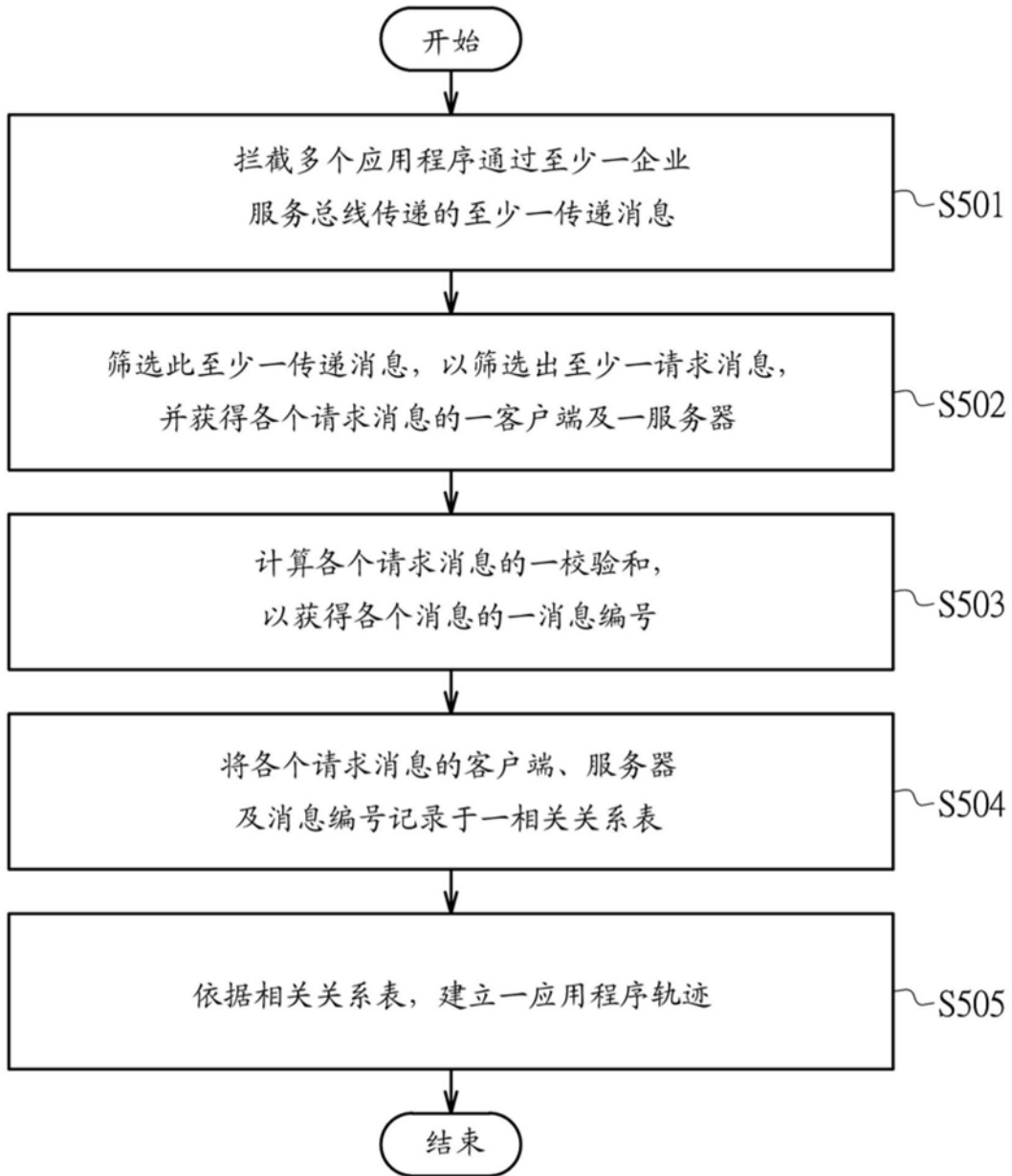


图5

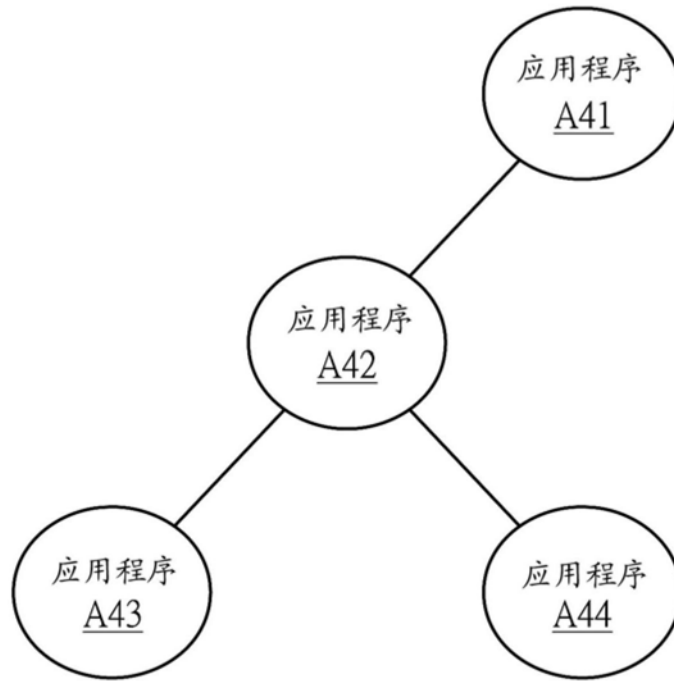


图6

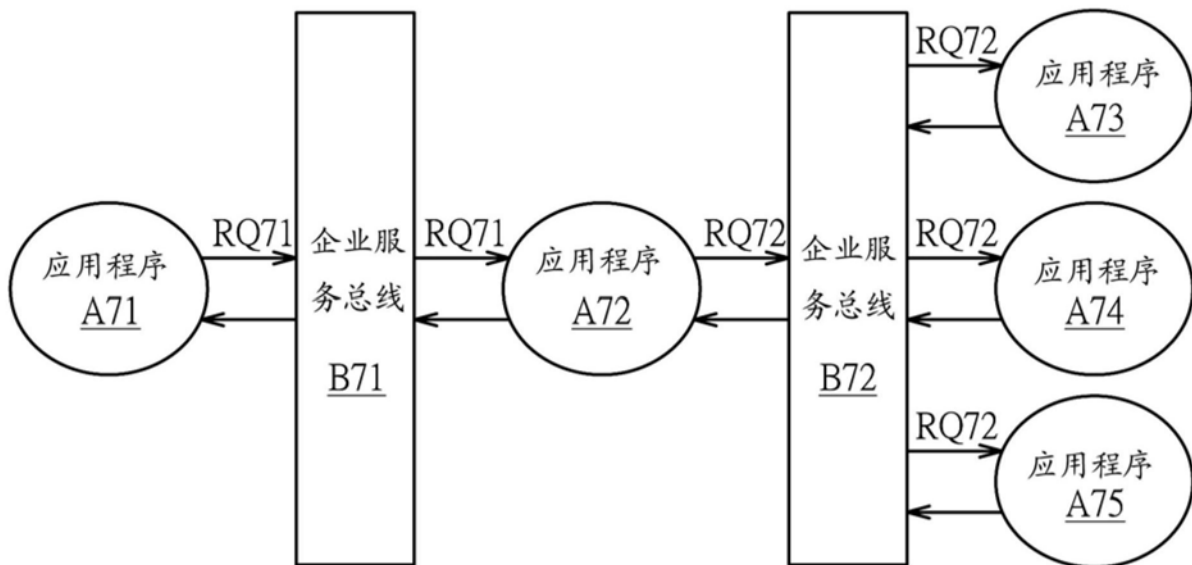


图7

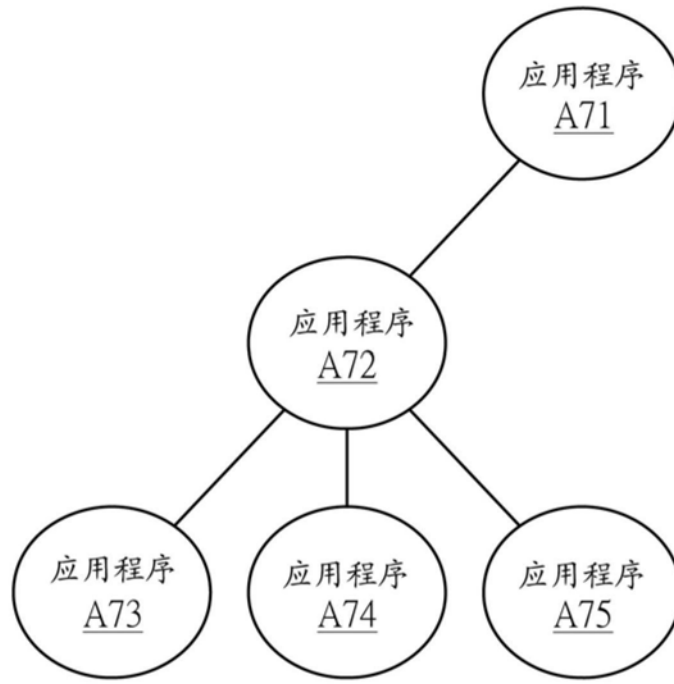


图8