(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 106663035 B
(45)授权公告日 2020.11.03

(21)申请号 201580035413.6
(22)申请日 2015.07.10
(65)同一申请的已公布的文献号
申请公布号 CN 106663035 A
(43)申请公布日 2017.05.10
(30)优先权数据
62/023,076 2014.07.10 US
62/054,901 2014.09.24 US
14/795,427 2015.07.09 US
(85)PCT国际申请进入国家阶段日
2016.12.29
(86)PCT国际申请的申请数据
PCT/US2015/040048 2015.07.10
(87)PCT国际申请的公布数据
WO2016/007922 EN 2016.01.14
(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 S·赛亚加拉詹 J·拉穆
K·萨克塞纳 R·斯里瓦斯塔瓦
L·菲根 N·梅塔
P·萨布拉马尼安
(74)专利代理机构 中国贸促会专利商标事务所有限公司 11038
代理人 李晓芳
(51)Int.Cl.
G06F 9/50(2006.01)
(56)对比文件
CN 102170457 A,2011.08.31
US 2009183168 A1,2009.07.16
CN 103368767 A,2013.10.23
CN 102681899 A,2012.09.19
审查员 王佳

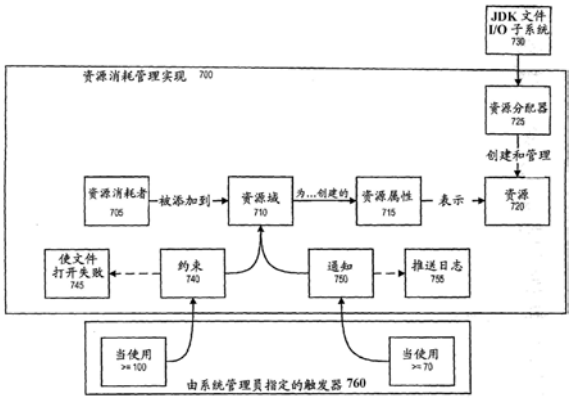
权利要求书3页 说明书22页 附图11页

(54)发明名称

用于资源隔离和消耗的系统、方法、介质和装置

(57)摘要

根据实施例，本文描述的是用于应用服务器环境中的资源隔离和消耗的系统和方法。该系统可以在包括在其上执行的应用服务器环境的一个或多个计算机处提供可以在应用服务器环境内使用的多个资源以及一个或多个分区，其中每个分区提供域的管理和运行时细分。该系统还可以配置资源消耗管理模块，以监控每个分区对所述多个资源的使用。资源消耗管理模块可以包括包含资源预留、资源约束和资源通知的组中的至少一个成员。



CN 106663035 B

1.一种用于应用服务器环境中的资源隔离和消耗的系统,包括:

一个或多个计算机,所述一个或多个计算机包括实现软件应用的部署和执行的应用服务器,其中所述应用服务器与域配置、一个或多个分区以及能够在所述应用服务器环境内使用的多个资源相关联,所述域配置被用于在运行时定义用于软件应用的执行的域,

其中所述一个或多个分区中的每个分区与分区配置相关联,并且其中每个分区提供所述域的细分并且包括一个或多个资源组和部署在该分区中的一个或多个软件应用,其中所述一个或多个资源组由一个或多个资源组模板定义;以及

可配置资源消耗管理模块,所述可配置资源消耗管理模块包括包含资源预留、资源约束和资源通知的组中的至少一个成员;

其中所述资源消耗管理模块被配置为监控每个分区对所述多个资源的使用,每个分区对所述多个资源的使用与在每个分区内部署的所述一个或多个软件应用相关联。

2.如权利要求1所述的系统,其中所述多个资源包括共享资源,并且其中所述资源消耗管理模块还被配置为接收指令来配置包含所述资源预留、所述资源约束和所述资源通知的组中的至少一个成员。

3.如权利要求2所述的系统,其中所述资源约束被配置为在所述一个或多个分区中的分区使用多于预定义量的所述共享资源时,执行约束动作。

4.如权利要求3所述的系统,其中所述约束动作从包含减慢、失败和关闭的组中选择。

5.如权利要求2至4中的任何一项权利要求所述的系统,其中所述资源通知被配置为在所述一个或多个分区中的分区使用多于预定义量的所述共享资源时,提供通知动作。

6.如权利要求2所述的系统,其中所述资源约束被配置为在所述一个或多个分区中的分区使用多于公平份额值的共享资源时,执行约束动作,所述约束动作包括减缓,其中所述减缓降低所述一个或多个分区中的所述分区的所述公平份额值,并且其中所述公平份额值与分配给所述分区的工作实例的数量相关联。

7.如权利要求1-4和6中的任何一项权利要求所述的系统,其中所述应用服务器环境包括多租户应用服务器环境,并且所述一个或多个分区中的每一个与多个预定义服务级别中的一个相关联,所述多个预定义服务级别中的每一个被配置为用于由所述资源消耗管理模块使用,以确定所述分区对资源的管理。

8.一种用于应用服务器环境中的资源隔离和消耗的方法,包括:

在包括实现软件应用的部署和执行的应用服务器的一个或多个计算机处提供以下各项,其中所述应用服务器与域配置相关联,所述域配置被用于在运行时定义用于软件应用的执行的域:

一个或多个分区以及能够在所述应用服务器环境内使用的多个资源,

其中所述一个或多个分区中的每个分区与分区配置相关联,并且其中每个分区提供所述域的细分并且包括一个或多个资源组和部署在该分区中的一个或多个软件应用,其中所述一个或多个资源组由一个或多个资源组模板定义;以及

配置资源消耗管理模块,以监控每个分区对所述多个资源的使用,所述资源消耗管理模块包括包含资源预留、资源约束和资源通知的组中的至少一个成员,每个分区对所述多个资源的使用与在每个分区内部署的所述一个或多个软件应用相关联。

9.如权利要求8所述的方法,其中所述多个资源包括共享资源,并且其中所述资源消耗

管理模块还被配置为接收指令来配置包含所述资源预留、所述资源约束和所述资源通知的组中的至少一个成员。

10.如权利要求8或9所述的方法，还包括：

当所述一个或多个分区中的分区使用多于预定义量的共享资源时，配置所述资源约束以执行约束动作。

11.如权利要求10所述的方法，其中所述约束动作从包含减慢、失败和关闭的组中选择。

12.如权利要求8至9和11中的任何一项权利要求所述的方法，还包括：

配置所述资源通知，以在所述一个或多个分区中的分区使用多于预定义量的所述资源时，提供通知动作。

13.如权利要求12所述的方法，其中所述通知动作推送日志事件文件，所述日志事件文件能够由系统管理员访问。

14.如权利要求8至9、11和13中的任何一项权利要求所述的方法，其中所述一个或多个分区中的每一个与多个预定义服务级别中的一个相关联，所述多个预定义服务级别中的每一个被配置为由所述资源消耗管理模块使用，以确定所述分区对资源的管理。

15.一种用于应用服务器环境中的资源隔离和消耗的装置，所述装置包括用于执行如权利要求8至14中的任何一项权利要求所述的方法的部件。

16.一种非暂态计算机可读存储介质，包括存储在其上的用于应用服务器环境中的资源隔离和消耗的指令，当所述指令被一个或多个计算机读取和执行时，使得所述一个或多个计算机执行步骤，包括：

在包括实现软件应用的部署和执行的应用服务器的一个或多个计算机处提供以下各项，其中所述应用服务器与域配置相关联，所述域配置被用于在运行时定义用于软件应用的执行的域：

一个或多个分区以及能够在所述应用服务器环境内使用的多个资源，

其中所述一个或多个分区中的每个分区与分区配置相关联，并且其中每个分区提供所述域的细分并且包括一个或多个资源组和部署在该分区中的一个或多个软件应用，其中所述一个或多个资源组由一个或多个资源组模板定义；以及

配置资源消耗管理模块，以监控每个分区对所述多个资源的使用，所述资源消耗管理模块包括包含资源预留、资源约束和资源通知的组中的至少一个成员，每个分区对所述多个资源的使用与在每个分区内部署的所述一个或多个软件应用相关联。

17.如权利要求16所述的非暂态计算机可读存储介质，其中所述多个资源包括共享资源，并且其中所述资源消耗管理模块还被配置为接收指令来配置包含所述资源预留、所述资源约束和所述资源通知的组中的至少一个成员。

18.如权利要求17所述的非暂态计算机可读存储介质，所述步骤包括：

当所述一个或多个分区中的分区使用多于预定义量的所述共享资源时，配置所述资源约束以执行约束动作。

19.如权利要求18所述的非暂态计算机可读存储介质，其中所述约束动作从包含减慢、失败和关闭的组中选择。

20.如权利要求17至19中的任何一项权利要求所述的非暂态计算机可读存储介质，所

述步骤还包括：

配置所述资源通知，以在所述一个或多个分区中的分区使用多于预定义量的所述共享资源时，提供通知动作。

21.如权利要求20所述的非暂态计算机可读存储介质，其中所述通知动作推送日志事件文件，所述日志事件文件能够由系统管理员访问。

22.如权利要求16至19和21中的任何一项权利要求所述的非暂态计算机可读存储介质，其中所述一个或多个分区中的每一个与多个预定义服务级别中的一个相关联，所述多个预定义服务级别中的每一个被配置为由所述资源消耗管理模块使用，以确定所述分区对资源的管理。

用于资源隔离和消耗的系统、方法、介质和装置

[0001] 版权声明

[0002] 本专利文档的公开内容的一部分包含受版权保护的素材。版权拥有者不反对任何人对专利文档或专利公开按照在专利商标局的专利文件或记录中出现的那样进行传真复制，但是除此之外在任何情况下都保留所有版权。

技术领域

[0003] 本发明的实施例一般涉及应用服务器和云环境，并且特别涉及用于多租户应用服务器环境中的资源隔离和消耗的系统和方法。

背景技术

[0004] 其示例包括Oracle WebLogic Server(WLS)和Glassfish的软件应用服务器一般提供用于运行企业软件应用的受管理环境。近来，技术还被开发以用于在云环境中使用，这允许用户或租户在云环境内开发和运行他们的应用，并且利用由环境提供的分布式资源。

发明内容

[0005] 根据实施例，本文描述的是用于应用服务器环境中的资源隔离和消耗的系统和方法。该系统可以在包括在其上执行的应用服务器环境的一个或多个计算机处提供可以在应用服务器环境内使用的多个资源以及一个或多个分区，其中每个分区提供域的管理和运行时细分。该系统还可以配置资源消耗管理模块，以监控每个分区对所述多个资源的使用。资源消耗管理模块可以包括包含资源预留、资源约束和资源通知的组中的至少一个成员。

附图说明

[0006] 图1示出了根据实施例的、用于在应用服务器、云或其它环境中支持多租赁的系统。

[0007] 图2进一步示出了根据实施例的、用于在应用服务器、云或其它环境中支持多租赁的系统。

[0008] 图3进一步示出了根据实施例的、用于在应用服务器、云或其它环境中支持多租赁的系统。

[0009] 图4示出了根据实施例的、用于与示例性多租户环境一起使用的域配置。

[0010] 图5进一步示出了根据实施例的示例性多租户环境。

[0011] 图6示出了根据实施例的、应用服务器环境中的资源隔离和消耗。

[0012] 图7示出了根据实施例的资源消耗管理实现。

[0013] 图8示出了根据实施例的、示出资源消耗管理集成中的交互的序列图。

[0014] 图9示出了根据实施例的资源消耗管理实现。

[0015] 图10示出了根据实施例的、应用服务器环境中的资源隔离和消耗。

[0016] 图11描绘了根据实施例的、用于应用服务器环境中的资源隔离和消耗的方法的流

程图。

具体实施方式

[0017] 根据实施例，本文描述的是用于应用服务器环境中的资源隔离和消耗的系统和方法。该系统可以在包括在其上执行的应用服务器环境的一个或多个计算机处提供可以在应用服务器环境内使用的多个资源以及一个或多个分区，其中每个分区提供域的管理和运行时细分。该系统还可以配置资源消耗管理模块，以监控每个分区对所述多个资源的使用。资源消耗管理模块可以包括包含资源预留、资源约束和资源通知的组中的至少一个成员。

[0018] 应用服务器(例如，多租户，MT)环境

[0019] 图1示出了根据实施例的用于在应用服务器、云或其它环境中支持多租赁的系统。

[0020] 如图1中所示，根据实施例，实现软件应用的部署和执行的应用服务器(例如，多租户，MT)环境100或其它计算环境可被配置为包括在运行时被用来定义应用服务器域的域102配置并且根据该域102配置来操作。

[0021] 根据实施例，应用服务器可以包括被定义以用于在运行时使用的一个或多个分区104。每个分区可以与全局唯一的分区标识符(ID)和分区配置相关联，并且还可以包括一个或多个资源组124，连同对资源组模板的引用126和/或分区特定的应用或资源128。域级资源组、应用和/或资源140也可以在域级被定义，可选地具有对资源组模板的引用。

[0022] 每个资源组模板160可以定义一个或多个应用A 162、B 164、资源A 166、B 168和/或其它可部署的应用或资源170，并且可以由资源组来引用。例如，如图1中所示，分区104中的资源组124可以引用190资源组模板160。

[0023] 一般而言，系统管理员可以定义分区、域级资源组和资源组模板以及安全领域；而分区管理员可以例如通过创建分区级资源组、将应用部署到分区或者引用用于分区的具体领域来定义其自己的分区的方面。

[0024] 图2进一步示出了根据实施例的、用于在应用服务器、云或其它环境中支持多租赁的系统。

[0025] 如图2中所示，根据实施例，分区202可以包括例如资源组205，资源组205包括对资源组模板210的引用206、虚拟目标(例如，虚拟主机)信息207以及可插入数据库(PDB)信息208。资源组模板(例如，210)可以定义例如多个应用A 211和B 212，连同诸如Java消息服务器(JMS)服务器213、存储转发(SAF)代理215、邮件会话组件216或Java数据库连接(JDBC)资源217之类的资源。

[0026] 图2中所示的资源组模板通过示例的方式被提供；根据其它实施例，可以提供不同类型的资源组模板和元素。

[0027] 根据实施例，当分区(例如，202)内的资源组引用220特定的资源组模板(例如，210)时，与特定分区相关联的信息可以与所引用的资源组模板结合使用，以指示分区特定信息230，例如分区特定的PDB信息。然后，分区特定信息可以由应用服务器用来配置资源(例如PDB资源)以供分区使用。例如，与分区202关联的分区特定的PDB信息可以由应用服务器用来利用适当的PDB 238配置232容器数据库(CDB)236，以供该分区使用。

[0028] 类似地，根据实施例，与特定分区相关联的虚拟目标信息可被用来定义239分区特定的虚拟目标240(例如，baylandurgentcare.com)，以供该分区使用，然后可以使该分区特

定的虚拟目标240经由统一资源定位符(URL)(例如,http://baylandurgentcare.com)可访问。

[0029] 图3进一步示出了根据实施例的用于在应用服务器、云或其它环境中支持多租赁的系统。

[0030] 根据实施例,诸如config.xml配置文件之类的系统配置被用来定义分区,该系统配置包括用于与该分区相关联的资源组的配置元素,和/或其它分区特性。可以使用特性名称/值对来每分区地指定值。

[0031] 根据实施例,多个分区可以在可以提供对CDB 243的访问并且经由web层244可访问的受管理的服务器/集群242或者类似环境内执行。这允许例如域或分区与(CDB的)PDB中的一个或多个PDB相关联。

[0032] 根据实施例,多个分区中的每个分区(在该示例中为分区A 250和分区B 260)可被配置为包括与该分区相关联的多个资源。例如,分区A可被配置为包括资源组251,资源组251包含应用A1252、应用A2254和JMS A 256,连同与PDB A 259相关联的数据源A 257,其中该分区可经由虚拟目标A 258访问。类似地,分区B 260可被配置为包括资源组261,资源组261包含应用B1 262、应用B2 264和JMS B 266,连同与PDB B 269相关联的数据源B 267,其中该分区可经由虚拟目标B 268访问。

[0033] 虽然上面的示例中的几个示例示出了CDB和PDB的使用,但是根据其它实施例,可以支持其它类型的多租户或非多租户数据库,其中可以例如通过模式的使用或不同数据库的使用来为每个分区提供特定的配置。

[0034] 资源

[0035] 根据实施例,资源是可被部署到环境的域的系统资源、应用或者其它资源或对象。例如,根据实施例,资源可以是可被部署到服务器、集群或其它应用服务器目标的应用、JMS、JDBC、JavaMail、WLDF、数据源或者其它系统资源或其它类型的对象。

[0036] 分区

[0037] 根据实施例,分区是可以与分区标识符(1D)和配置相关联、并且可以通过资源组和资源组模板的使用包含应用和/或参考域范围的资源的域的运行时和管理细分或切片。

[0038] 一般而言,分区可以包含其自己的应用、经由资源组模板参考域范围的应用,并且具有其自己的配置。可分区的实体可以包括资源,例如JMS、JDBC、JavaMail、WLDF资源,以及其它组件,诸如JNDI命名空间、网络业务、工作管理器以及安全策略和领域。在多租户环境的上下文中,系统可被配置为提供对于与租户相关联的分区的管理和运行时方面的租户访问。

[0039] 根据实施例,分区内的每个资源组可以可选地引用资源组模板。分区可以具有多个资源组,并且这些资源组中的每个资源组可以引用资源组模板。每个分区可以定义用于在该分区的资源组所引用的资源组模板中未指定的配置数据的特性。这使得分区能够充当在资源组模板中定义的可部署资源到用于与该分区一起使用的具体值的绑定。在一些情况下,分区可以重写(override)由资源组模板指定的配置信息。

[0040] 根据实施例,例如由config.xml配置文件定义的分区配置可以包括多个配置元素,例如:“partition(分区)”,其包含定义分区的属性和子元素;“resource-group(资源组)”,其包含被部署到分区的应用和资源;“resource-group-template(资源组模板)”,其

包含由那个模板定义的应用和资源;“jdbc-system-resource-override(JDBC系统资源重写)”,其包含数据库特定的服务名称、用户名和密码;以及“partition-properties(分区特性)”,其包含可用于资源组模板中的宏替换的特性键值。

[0041] 在启动时,系统可以使用由配置文件提供的信息以从资源组模板为每个资源生成分区特定的配置元素。

[0042] 资源组

[0043] 根据实施例,资源组是可以在域或分区级被定义并且可以引用资源组模板的可部署资源的命名的、完全限定的集合。资源组中的资源被认为是完全限定的,是因为管理员已提供了启动或连接到那些资源所需的信息中的所有信息,例如用于连接到数据源的凭据或者用于应用的定向信息。

[0044] 系统管理员可以在域级或者在分区级声明资源组。在域级,资源组提供了分组相关资源的方便的方式。系统可以与未分组的资源相同地管理在域级资源组中声明的资源,以使得资源可以在系统启动期间被启动,并且在系统关闭期间被停止。管理员还可以单独地停止、启动或移除组中的资源,并且可以通过对组的操作来隐式地对组中的所有资源执行动作。例如,停止资源组停止该组中尚未停止的资源中的所有资源;启动资源组启动该组中尚未启动的任何资源;而移除资源组移除该组中包含的资源中的所有资源。

[0045] 在分区级,受任何安全限制的影响,系统或分区管理员可以在分区中配置零个或更多个资源组。例如,在SaaS用例中,各种分区级资源组可以参考域级资源组模板;而在PaaS用例中,不参考资源组模板的分区级资源组可以被创建,但是替代地表示将仅在该分区内可用的应用及其相关资源。

[0046] 根据实施例,资源分组可被用来将应用和它们使用的资源分组在一起作为域内的不同管理单元。例如,在下面描述的医疗记录(MedRec)应用中,资源分组定义MedRec应用及其资源。多个分区可以运行相同的MedRec资源组,每个分区利用分区特定的配置信息,以使得作为每个MedRec实例的一部分的应用特定于每个分区。

[0047] 资源组模板

[0048] 根据实施例,资源组模板是在域级定义的、可以从资源组引用的可部署资源的集合,并且激活其资源所需的信息中的一些信息可以不被存储为模板本身的一部分,以使得其支持分区级配置的规范。域可以包含任何数量的资源组模板,这些资源组模板中的每个资源组模板可以包括例如一个或多个相关的Java应用和这些应用所依赖的资源。关于这些资源的信息中的一些信息可以是跨所有分区相同的,而其它信息可以依分区而不同。并非所有配置都需要在域级指定——分区级配置可以替代地通过宏或特性名称/值对的使用在资源组模板中指定。

[0049] 根据实施例,特定的资源组模板可以由一个或多个资源组引用。一般而言,在任何给定的分区内,资源组模板可以一次由一个资源组引用,即,不由同一分区内的多个资源组同时引用;然而,它可以由不同分区中的另一资源组同时引用。包含资源组的对象(例如域或分区)可以使用特性名称/值指派来设置资源组模板中任何标记(token)的值。当系统使用引用的资源组激活资源组模板时,它可以用在资源组的包含对象中设置的值替换这些标记。在一些情况下,系统还可以使用静态配置的资源组模板和分区来为每个分区/模板组合生成运行时配置。

[0050] 例如,在SaaS用例中,系统可以多次激活相同的应用和资源,包括一次为了将使用它们的每个分区的激活。当管理员定义资源组模板时,他们可以使用标记来表示将在其它地方提供的信息。例如,在连接到CRM相关的数据资源中使用的用户名可以在资源组模板中被指示为\${CRMDataUsername}。

[0051] 租户

[0052] 根据实施例,在诸如多租户(MT)应用服务器环境之类的多租户环境中,租户是可以由一个或多个分区和/或一个或多个租户感知的应用表示或以其它方式与一个或多个分区和/或一个或多个租户感知的应用相关联的实体。

[0053] 例如,租户可以表示诸如不同的外部公司或特定企业内的不同部门(例如,HR部门和财务部门)之类的不同的用户组织,这些用户组织中的每个用户组织可以与不同的分区相关联。租户全局唯一身份(租户ID)是特定用户在特定时刻与特定租户的关联。系统可以从用户身份例如通过参考用户身份仓库(store)来导出特定用户属于哪个租户。用户身份使系统能够实施用户被授权执行的那些动作,包括但不限于用户可以属于哪个租户。

[0054] 根据实施例,系统实现将不同租户的管理和运行时彼此隔离。例如,租户可以配置他们的应用的一些行为,以及他们可以访问的资源。系统可以确保特定的租户不能管理属于另一租户的工件(artifact);并且在运行时代表特定租户工作的应用仅参考与那个租户相关联的资源,而不参考与其他租户相关联的资源。

[0055] 根据实施例,非租户感知的应用是不包含显示处理租户的逻辑的应用,以使得不管哪个用户提交了该应用正在对其进行响应的请求,该应用使用的任何资源都可以是可访问的。相比之下,租户感知的应用包括显式处理租户的逻辑。例如,基于用户的身份,应用可以导出用户所属的租户并且使用该信息来访问租户特定的资源。

[0056] 根据实施例,系统使用户能够部署被显式写为租户感知的的应用,以使得应用开发者可以获得当前租户的租户ID。然后,租户感知的应用可以使用租户ID来处理正在使用应用的单个实例的多个租户。

[0057] 例如,支持单个医生的办公室或医院的MedRec应用可以暴露给两个不同的分区或租户(例如,湾地紧急护理(Bayland Urgent Care)租户和山谷健康(Valley Health)租户),这些不同的分区或租户中的每个能够访问诸如分开的PDB之类的分开的租户特定的资源,而无需改变底层应用代码。

[0058] 示例性域配置和多租户环境

[0059] 根据实施例,应用可以在域级被部署到资源组模板,或者被部署到范围为分区或范围为域的资源组。可以使用每应用或每分区指定的部署计划重写应用配置。部署计划也可以指定为资源组的一部分。

[0060] 图4示出了根据实施例的、用于与示例性多租户环境一起使用的域配置。

[0061] 根据实施例,当系统启动分区时,它根据所提供的配置创建到各自的数据库实例的虚拟目标(例如,虚拟主机)和连接池,这包括为每个分区创建一个虚拟目标和连接池。

[0062] 通常,每个资源组模板可以包括一个或多个相关的应用和那些应用所依赖的资源。通过提供资源组模板中的可部署资源到与分区相关联的具体值的绑定,每个分区可以提供在它所参考的资源组模板中没有指定的配置数据;在一些情况下,这包括重写由资源组模板指定的某些配置信息。这使得系统能够使用每个分区已定义的特性值为每个分区以

不同方式激活由资源组模板表示的应用。

[0063] 在一些实例中，分区可以包含不参考资源组模板或直接定义它们自己的分区范围的可部署资源的资源组。在分区内定义的应用和数据源一般仅对于该分区是可用的。资源可以被部署以使得可以使用分区：＜分区名称＞/＜资源JNDI名称＞或者域：＜资源JNDI名称＞从跨分区来访问它们。

[0064] 例如，MedRec应用可以包括多个Java应用、数据源、JMS服务器和邮件会话。为了为多个租户运行MedRec应用，系统管理员可以定义单个MedRec资源组模板286，在该模板中声明那些可部署资源。

[0065] 与域级的可部署资源相比，在资源组模板中声明的可部署资源可能未在模板中完全配置，或者不能按原样被激活，因为它们缺少一些配置信息。

[0066] 例如，MedRec资源组模板可以声明由应用使用的数据源，但是它可以不指定用于连接到数据库的URL。与不同租户相关联的分区(例如分区BUC-A 290(湾地紧急护理，BUC)和分区VH-A 292(山谷健康，VH))可以通过各自包括引用296、297 MedRec资源组模板的MedRec资源组293、294来引用一个或多个资源组模板。然后，该引用可被用来创建302、306用于每个租户的虚拟目标/虚拟主机，包括与BUC-A分区相关联的虚拟主机baylandurgentcare.com304，以供湾地紧急护理租户使用；以及与VH-A分区相关联的虚拟主机valleyhealth.com 308，以供山谷健康租户使用。

[0067] 图5进一步示出了根据实施例的示例性多租户环境。如图5中所示，并且继续其中两个分区引用MedRec资源组模板的来自上文的示例，根据实施例，服务器端小程序(servlet)引擎310可被用来支持多个租户环境，在这个示例中是湾地紧急护理医师(Bayland Urgent Care Physician)租户环境320和山谷健康医师(Valley Health Physician)租户环境330。

[0068] 根据实施例，每个分区321、331可以定义在其上接受用于该租户环境的传入业务的不同虚拟目标，以及用于连接到该分区及其资源324、334的不同URL 322、332，在这个示例中分别包括湾地紧急护理数据库或山谷健康数据库。数据库实例可以使用兼容模式，因为相同的应用代码将对这两个数据库都执行。当系统启动分区时，它可以创建到各自的数据库实例的虚拟目标和连接池。

[0069] 资源隔离和消耗

[0070] 根据实施例，通过允许多个租户共享单个域实现了客户的计算基础设施的密度提高和利用率增加。然而，当多个租户或分区驻留在同一域内时，可以有必要确定、管理、隔离和监控在域运行时各个分区对资源的访问和使用，以便于促进那些资源的分配中的公平性、防止对共享资源访问的竞争和/或干扰、以及为多个分区和/或相关联的租户提供一致的性能。

[0071] 根据实施例，本系统利用由Java开发工具包(例如，JDK 8u40)提供的资源管理支持，以便于允许系统管理员指定关于JDK受管理资源的资源消耗管理策略(例如，指定约束、追索(recourse)动作和通知)。这些资源可以包括例如CPU、堆、文件和网络。本文描述的方法和系统还可以实现轻量级的、基于标准的、全面且可扩展的资源消耗管理(RCM)框架，其可以由应用服务器环境内的容器和组件使用，以用于管理共享资源的消耗。

[0072] 根据实施例，在应用由应用服务器环境内的不同分区和/或不同租户部署的情况

下,当应用试图使用共享资源(例如,诸如CPU、网络和存储之类的低级资源、或者诸如数据源、JMS连接、日志之类的较高级资源等)时,一般有两个问题会出现。这两个问题包括分配期间的竞争/不公平,以及变化的性能和潜在的服务级别协议(SLA)违反。当存在对共享资源的多个请求时,分配期间的竞争和不公平可以产生,并且这导致竞争和干扰。由于良性原因(例如,高流量、DDoS攻击)、行为不当或有错误的应用、恶意代码等,异常资源消耗请求也可以发生。这样的异常请求可以导致共享资源的容量过载,从而阻止另一应用对资源的访问。变化的性能可以导致潜在的SLA违反。竞争和不公平可以导致对于不同应用/消耗者的不同性能,并且可能导致SLA违反。

[0073] 虽然在讨论与支持资源隔离和消耗相关联的问题时,本文描述的各种系统和方法使用多分区/多租户应用服务器环境,但是这些系统和方法适用于其中域中的多个共驻留应用(与在域内的分区内的应用不同)彼此争夺(compete)并且它们对共享资源的访问导致竞争(contention)和不可预测的性能的传统部署。

[0074] 根据实施例,向诸如系统管理员之类的管理员提供用于监控对共享资源(例如,诸如CPU、堆、文件和网络之类的JDK资源)的访问的机制,并且实施关于消耗者对这些资源的消耗的策略。例如,在多分区系统中,系统管理员可以配置它,以使得消耗者不会过度消耗共享资源(多个共享资源)并且从而使另一消耗者难以访问该共享资源。

[0075] 根据实施例,诸如系统管理员之类的管理员可以向不同的消耗者/客户提供分层/差异化的服务级别。附加地,管理员可以可选地配置业务特定的策略(例如,基于每天的时间(time-of-day)对于特定分区应用不同的策略,以容纳将在分区中运行的特定的或预测的工作负载),并且使策略在共享资源的消耗时被实施。

[0076] 根据实施例,向管理员提供灵活的、可扩展的和通用的资源消耗框架,其可以被用于协助应用服务器环境的容器和组件管理由不同消耗者访问和/或利用的共享资源。该框架可以附加地允许用于RCM框架的不同用户的在不同级别的细粒度策略的规范和实施。

[0077] 根据实施例,当描述或解释用于资源隔离和消耗的方法和系统时,可以使用以下定义。

[0078] 根据实施例,并且在资源消耗管理的上下文中,术语资源是指诸如应用服务器环境之类的系统内的如下实体:其表示在消耗者、分区和/或租户之间共享的并且以有限数量存在的东西,并且它的缺少会导致资源消耗者的性能变化。

[0079] 根据实施例,术语资源消耗者可以是其资源使用由资源消耗管理(RCM)API控制的可执行实体。例如,多租户应用服务器环境内的资源消耗者可以包括分区。

[0080] 根据实施例,资源域可以将一组资源消耗者绑定到资源,并且对该资源的资源消耗者组施加公共资源管理策略。

[0081] 根据实施例,资源管理策略可以定义预留、约束和通知,并且可以限制由资源消耗者消耗的资源的数量或者调节(throttle)消耗的速率。

[0082] 根据实施例,约束可以是当对资源做出资源消耗请求时被调用的回调(callback)。回调可以为该资源实现确定可以由请求消耗的资源单元的数量的策略。

[0083] 根据实施例,通知可以是紧跟在资源消耗请求已被处理之后被调用的回调。通知可以被用于实现指定在资源消耗之后要执行的动作(例如,记账、日志记录等)的资源管理策略。

[0084] 根据实施例，资源属性描述资源的特性或属性。附加地，资源分配器(dispenser)是指创建资源的代码/容器/逻辑。此外，对某物启用RM可以指修改资源分配器以使得RCM框架可以管理该资源的消耗的行为。

[0085] 根据实施例，追索动作可以是当资源消耗者的资源消耗达到某个状态时执行的动作。追索动作可以是通知或采取预防动作，以便防止资源的过度使用或尝试处理过度使用。追索动作可以在与做出资源消耗请求的线程不同的线程中执行。

[0086] 图6示出了根据实施例的、应用服务器环境中的资源隔离和消耗。图6描绘了包含域610的应用服务器环境600。系统管理员620可以配置域内的资源管理策略630。资源管理策略630可以被配置为至少包含资源预留631、资源约束632和资源通知633。域附加地分别包含分区A 640和分区B 650。分区A和分区B分别包含分别与虚拟目标A 645和虚拟目标B 655相关联的资源组642和652。如图6中所示，当例如分区运行需要使用共享资源的应用时，分区A或分区B还可以访问共享资源660。共享资源660可以包括但不限于JDK资源，诸如CPU、堆、网络和文件。

[0087] 根据实施例，资源管理策略可以由系统管理员或另一个具有足够许可的人员配置，以在分区A或分区B关于共享资源满足由资源管理策略设置的已配置条件时执行行动或任务。这些行动或任务可以包括但不限于约束或通知。约束的示例包括减慢或停止分区对共享资源的使用。通知的示例包括但不限于日志记录或向管理员提供通知。

[0088] 根据实施例，图6中描绘的应用服务器环境可以是Oracle WebLogic服务器。WebLogic服务器可以包括基于类加载器(ClassLoader)的隔离(不属于应用的共享库的类被彼此隔离)特征和WebLogic工作管理器特征，其可以允许管理员通过配置一组调度准则来配置应用如何确定它的工作的执行的优先级。然而，这两个特征在提供给应用的隔离方面相当有限，并且在WLS-MT部署中，期望将诸如所有JDK资源之类的所有资源分区并且管理它们的消耗以向分区保证性能隔离。

[0089] 软隔离和硬隔离

[0090] 根据实施例，隔离的执行环境和共享的执行环境之间的区别不是只能二者择其一的(binary)，在共享基础设施并且完全隔离的执行环境与非隔离的执行环境之间可以存在连续的选择。

[0091] 根据实施例，对于供应来自同一应用服务器环境中的不同分区中的非信任租户的代码的系统管理员，系统可以提供对于隔离的执行环境的支持(例如，MVM/隔离保护、隔离的VM运行时-[GC、堆、线程、系统类]、调试和诊断等)，以使得系统可以使分区中发生的操作彼此沙盒化(sandbox)-这可以被称为执行隔离。这可以被认为是硬隔离的一种形式。

[0092] 通过来自由JDK 8u40暴露的RM API的支持，系统还可以提供隔离的较软形式，其中对某些共享JDK资源的访问被隔离，并且通过追索动作来维护它们的隔离-这可以被称为性能隔离。

[0093] 与工作管理器(Work Manager，WM)的关系

[0094] 根据实施例，应用服务器(例如，WebLogic服务器)工作管理器(WM)可以允许管理员通过配置一组调度准则(Max/Min和容量约束、响应时间(ResponseTime)请求类)来配置应用如何确定其工作的执行的优先级。这可以允许WM管理分配给应用的线程、管理向那些线程调度工作实例、以及帮助维护协定。

[0095] 根据实施例,WM为它的调度策略使用为工作实例的执行分配的时间。工作实例的经历时间(elapsed time)可以是CPU时间和非CPU时间(等待I/O、系统调用本地调用等)的组合,并且对于某些用例(例如,I/O约束的(I/Obound)工作负载)来说是用来针对其指定策略的较好衡量(measure)。

[0096] 通过资源消耗管理(RCM)的增强

[0097] 根据实施例,WM分区公平份额可以在公平份额(fair-share)计算中使用用于执行分区的工作实例的经历时间。在诸如WebLogic服务器之类的应用服务器环境中的并置分区的情况下,确保在CPU级的性能隔离也是重要的。

[0098] 例如,在其中分区P1具有I/O为主的工作负载,并且P2具有CPU密集的工作负载并且P1和P2都具有相同的WM公平份额的情况下。假定工作实例由P1提交并且同时完成,则P1和P2将由WM以类似的方式调度(例如,将为P1和P2调度和执行相同数量的工作实例)。然而,由于P2的工作负载是CPU密集的,因此它对共享CPU的使用大于P1。这可能不是所期望的。

[0099] 根据实施例,由诸如JDK RM API之类的API提供的CPU时间使用通知考虑了积累到资源上下文的每线程的CPU时间。这为系统管理员提供了在CPU级指定策略并且在并置分区之间实现性能隔离的不同的正交方式。对于CPU时间资源来说可用的WM分区公平份额和公平份额策略二者可以由系统管理员有意义地采用以实现不同的结果。对于I/O相关的性能隔离,在文件和网络资源级的公平份额策略通过RCM特征也是可用的。

[0100] 根据实施例,WM可以仅对其管理的线程实施其策略。可以存在作为资源上下文的一部分的、不由WM管理的线程(例如,驻留在分区中的、线程产生的应用),并且可以使用RCM策略来管理它们。通过JDK RM API支持,除了卡住线程的情况之外,系统可以确定和标记失控线程和死锁线程,并且允许系统管理员对它们采取动作。

[0101] 与基于规则的弹性以及监视/通知的关系

[0102] 根据实施例,诸如WebLogic Diagnostics Framework(WLDF)之类的诊断框架的监视和通知组件允许系统管理员监控服务器和应用状态并且基于标准发送通知。这些监视器在资源被消耗之后从事监控关于系统的状态的度量。RCM策略允许在消耗请求期间实施策略。RCM策略帮助管理员精确地调整(shape)资源消耗者对它们的共享资源的使用。

[0103] 根据实施例,基于规则的弹性特征可以建立在WLDF监视/通知特征上,以允许系统管理员构建监控跨集群的资源使用的复杂规则以做出缩放管理动作。基于规则的弹性可以关注“宏观(macro)”级别上(例如,WLS集群范围或跨服务)的历史趋势的使用,以监控和执行在微观级别上影响多个工作的动作,从而关注单独消耗者对共享组件的单独资源消耗请求、以及通常导致对该资源的特定进一步使用(例如,仅影响当前VM的动作)的改变的策略(追索动作)。

[0104] 根据实施例,JDK RM API可以在每资源上下文基础(例如,分区)上提供对于JDK管理的资源的、关于某些资源消耗度量的信息。这些信息可以被传递到分区范围的监控特征,以使得它们可以被显现为对应的PartitionRuntimeMBean上的属性。

[0105] 与分区生命周期事件以及启动/停止支持的关系

[0106] 根据实施例,RCM框架可以建立线程中的正确的资源上下文,以使得在该线程的上下文中消耗的所有资源由实现针对该资源上下文适当地考虑。RCM框架可以使用通过诸如WebLogic Server事件特征之类的应用服务器特征提供的分区启动/停止(以及启用/禁用

(如果可用的话))事件来知道新分区何时被启动或停止、以及创建新的资源上下文或删除它们。

[0107] 根据实施例,由于分区文件系统可以不实现虚拟文件系统,因此对分区的文件系统的访问可以通过java.[n]io,并且应用服务器可能不能拦截对它的文件I/O操作。然而,RCM框架可以直接向系统管理员暴露JDK文件描述符和文件相关资源。

[0108] 与协同存储器管理(CMM)的关系

[0109] 根据实施例,协同存储器管理(Co-operative Memory Management,CMM)特征可以被用于通过使系统的不同部分对存储器压力做出反应来处理低存储器情况。存储器压力可以从外部确定。然后驻留在系统中的应用、中间件和JVM可以对存储器压力水平做出反应。例如,在高存储器压力情况下,可以使最少使用的应用空闲/休眠,JDBC/JMS和Coherence子系统可以减少它们的高速缓存大小,而JVM可以减少它的堆大小等。

[0110] 根据实施例,CMM特征可以被用于在宏观(机器)级别处理存储器压力场景并且在高存储器情况发生之后做出反应。机器中的高存储器压力可以是外部错误的JVM或机器中的进程的结果,并且在接近(分配的和保持的存储器大小)时不需要采取动作来控制存储器压力的“源”以及控制异常存储器使用。策略可以在“微观”(WLS服务器JVM)级别操作,并且可以主动控制错误的“资源消耗者”对堆的使用。因此,CMM特征和系统管理员应用RCM策略在本质上是互补的。

[0111] 资源管理支持

[0112] 根据实施例,诸如JDK 8u40之类的开发工具包中提供的资源管理器API可以将资源策略实施与资源检测(instrumentation)和资源消耗的通知分开。RM API可以允许诸如资源消耗管理器之类的外部管理器将应用域中的资源消耗者线程与资源上下文相关联。资源上下文可以为开发工具包提供针对其考虑资源消耗请求的上下文。RM API还可以允许诸如资源消耗管理器之类的外部管理器注册对于通过资源计量器(resourcemeter)获得关于资源上下文对(通过resourceIds识别的)特定资源的消耗的信息的兴趣。

[0113] 根据实施例,由诸如JDK之类的开发工具包提供的标准资源计量器实现是:简单计量器(用于资源上下文的成功资源分配的简单计数);有界计量器(利用上界的实施的计数);通知计量器(对于配置粒度县(granular county)的资源计数和请求批准、在资源分配之前形成资源管理器);以及调节计量器(throttlemeter)(将消耗限制为每秒指定的速率)。

[0114] 根据实施例,诸如简单计量器和有界计量器之类的某些计量器以拉取(pull)的方式获得信息。诸如通知计量器之类的其它计量器可以向资源管理器发回预消耗通知调用,以便资源管理器实施接受、减慢或拒绝资源消耗请求的策略。

[0115] 启用RM的JDK资源

[0116] 根据实施例,可以是启用RM的(RM-enabled)、从而允许系统管理员指定对它们的资源管理策略的JDK管理资源的示例包括打开文件描述(Open File Descriptions)、堆、线程、CPU时间等。附加地,可以使以下资源可用作资源消耗度量:文件描述符、文件(例如,打开文件计数、发送的字节、接收的字节、总数)、套接字和数据报、堆、线程、活动线程计数、CPU时间。

[0117] 根据实施例,CPU时间测量可以由诸如JDK之类的开发工具包定期地执行,从而监

控可以对每个资源上下文中活动的线程进行采样并且向计量表发送更新的线程。

[0118] 根据实施例,G1垃圾收集器可以是基于区域的分配器。因为垃圾收集被执行并且对象被复制,所以垃圾收集器可以确保对象被复制到具有相同资源上下文的区域。

[0119] 资源消耗管理-资源,触发器

[0120] 根据实施例,系统管理员可以在每资源消耗者的基础上围绕启用RM的资源指定RCM策略。对于资源,系统管理员可以指定一个或多个触发器(例如,资源的最大使用被限制为400个单位),以及当触发器达到值时必须执行的动作。这些动作可以导致在做出资源消耗请求的同一线程中发生的(同步)、或者可以在与做出资源消耗请求的线程不同的线程中执行的(异步)活动。触发器/动作组合可以允许系统管理员调整、控制和限制资源消耗者对资源的使用。

[0121] 根据实施例,存在系统管理员可以指定的若干动作。这些动作包括但不限于通知、减慢、失败和关闭。通知动作向系统管理员提供通知作为信息更新。减慢动作可以调节(例如,减缓)资源被消耗的速率。失败动作可以对于一个或多个资源消耗请求失败。一旦资源的使用落入期望的限制以下,则失败动作还可以终止,从而再次允许资源消耗。关闭动作可以是SIGTERM等价物(例如,发送到进程以请求其终止的信号),并且试图停止资源消耗,同时允许消耗者能够进行清除。

[0122] 根据实施例,RCM是基于JSR-284 API的WebLogic Server RCM框架,并且提供用于由WebLogic容器、组件和工具使用的轻量级RCM SPI(服务提供者接口)和API。

[0123] 现在参考图7,其示出了根据实施例的资源消耗管理实现。如所描绘的,资源消耗管理实现700可以包括资源消耗者705、资源域710、资源属性715、资源720、资源分配器725、约束740、通知750、触发器760和JDK文件I/O子系统730。如图7中所示,触发器760可以由系统管理员例如在如图6中所示的资源消耗管理策略630内设置。

[0124] 当资源消耗者705(例如,分区)开始消耗资源时,那些资源的使用被监控。回想例如消耗的资源可以包括CPU、堆、文件和网络,资源消耗管理实现监控消耗的资源并且将消耗的资源与系统管理员设置/配置/指定的触发器760进行比较。

[0125] 例如,如图7中所示,由系统管理员设置的触发器是用于在资源的使用大于或等于70(例如,CPU时间的70%)时要日志记录/向系统管理员显示的通知。一旦对于资源消耗者来说CPU时间超过70%,则资源消耗管理策略可以将日志755推送到文件,该日志文件指示资源消耗者已超过触发器内由系统管理员设置的阈值。

[0126] 根据实施例,当系统管理员将“通知”指定为触发器中的追索动作类型时,WLS RCM框架可以日志记录关于触发器中指定的使用已达到的标准消息。它日志记录消息,并且将所需的补充属性(例如,当前使用、先前使用、对于分区达到的通知限额)作为消息的一部分。系统管理员可以部署WLDF系统模块来配置监视规则,以监听标准日志消息,并且使用框架中的通知功能来发送通知。

[0127] 此外,如图7中所描绘的,当资源消耗者对资源的使用大于或等于100(例如,100个打开文件描述符)时,约束被抛出。例如,假定资源消耗者(例如,分区)超过100的限制。然后,基于由系统管理员设置的触发器,约束将被调用,并且异常可以被抛出,例如,无法打开所请求的文件745。

[0128] 根据实施例,约束的另一个示例是减缓(slow-down)。减缓可以涉及减少分区工作

管理器的公平份额,从而工作管理器将为分区调度较少数量的工作实例。附加地,工作管理器还可以减少分区的最大线程约束,以便于减少分区的并发活动工作实例的最大数量。减缓还可以降低应用创建的线程的优先级。减缓动作/约束可以被多次调用,以便于进一步减缓分区。这可以导致分区工作管理器的公平份额、最大线程约束和应用线程的优先级的值进一步减小。这可以例如在公平份额策略支持的情况下完成。类似地,减慢条件可以被撤销,这将导致使得分区工作管理器公平份额和最大线程约束回到原始配置的值,并且还恢复应用线程的正常优先级。

[0129] 根据实施例,关闭分区动作可以被指定为触发器已被满足之后的追索动作。当系统管理员将关闭指定为追索动作之一时,其资源消耗违背指定限额的分区将被关闭。

[0130] 根据实施例,在分区关闭时,将进行适当的清除,并且可以关闭对应的JDK资源上下文。一旦分区被关闭,则分区及其应用将不可访问。

[0131] 资源可插入性

[0132] 根据实施例,资源实现者可以使用SPI在资源消耗管理框架中注册它们的资源属性。管理员可以定义关于这些资源属性的资源管理策略。通过资源属性,资源实现者可以描述资源的各种特性(可丢弃(disposable)、有界、可保留、粒度、测量单位等)。资源实现者还可以确定资源的粒度和默认最大测量延迟以及速率管理周期。资源实现者可以选择这些值,以便于控制测量的性能影响。

[0133] 根据实施例,资源实现者可以将consume()调用插入到RCM框架,以检查是否可以消耗一定量的资源。这允许RCM框架为请求对资源的访问的资源消耗者检查和实施关于资源的资源管理策略。对于可丢弃资源,资源实现者可以将relinquish()调用插入到RCM框架,以指示资源不再由资源容器使用,因此它可用于重用。

[0134] 根据实施例,RCM框架可以咨询适当的资源域及其相关联的策略,以检查是否可以向资源消耗者提供指定的数量。

[0135] 资源消耗者可插λ性

[0136] 根据实施例,RCM框架可以允许通过SPI插入新的资源消耗者类型。资源消耗者可以是例如分区。资源分配器可以将一个资源消耗者的资源使用与另一资源消耗者进行确定、隔离和区分。例如,如果资源消耗者类型是分区,并且被访问的资源是共享数据源上的打开连接的数量,则数据源容器可以将来自某一分区的连接请求(以及该请求的后续满意度)与另一个分区的连接请求区分开。如果资源的使用不能被明确地指派给唯一的资源消耗者,则可能无法保证隔离和公平。

[0137] Java开发工具包的资源管理器API

[0138] 根据实施例,可以通过JSR-284资源属性(ResourceAttribute)来表示上文描述的启用RM的JDK资源。表示资源的代理资源分配器(ResourceDispenser)可以被实现,并且可以与JDK RM API对接(interface)。

[0139] 根据实施例,WLS RCM实现可以监听分区启动事件,并且为启动的分区创建资源上下文(ResourceContext)。可以基于为该分区指定的RCM策略配置为需要被监控的每个JDK资源创建JDK资源计量器(ResourceMeters)。然后,这些JDK资源计量器可以与该分区的JDK资源上下文相关联。

[0140] 根据实施例,WLS RCM实现可以向每个JDK资源计量器(例如通知计量器)注册JDK

资源批准器(ResourceApprover)(监听器)。这个监听器可以由JDK在每个预消耗分配请求时进行回调,然后监听器可以将其委派给RCM框架中的适当的JSR-284资源域(ResourceDomain)。如在第4.2.8节中所描述的,RCM框架为该资源上下文实现资源消耗管理策略。

[0141] 现在参考图8,图8示出了根据实施例的示出资源消耗管理集成中的交互的序列图。图8中描绘的序列从应用805执行将消耗资源的操作开始。然后JDK 810请求数量。通知计数器815执行资源请求,然后资源批准器820执行消耗操作。资源域825执行预消耗(preConsume),其还可以包括与分区当前使用和建议使用相关的指示符。然后,如由系统管理员配置的约束830可以执行策略。策略835将根据其设置或者返回由应用请求的资源量或者通过返回零资源来拒绝对资源的请求。该确定基于可以由系统管理员配置的策略。

[0142] 在线程中建立正确的上下文

[0143] 根据实施例,由于JDK RM实现考虑在线程中对与线程相关联的资源上下文发生的资源消耗,因此系统可能需要在线程上建立正确的资源上下文,以防止资源的错误计算。可以是WLS RCM的RCM可以使用当前线程的组件调用上下文(Component Invocation Context,CIC)来确定要用于资源上下文的分区。WLS RCM实现可以注册ComponentInvocationContextChangeListener(组件调用上下文改变监听器),以当CIC在线程中改变时接收通知。当该CIC改变通知被接收到并且CIC改变是由于分区切换(例如,交叉分区(cross-partition)或到/来自全局转换的分区)时,WLS RCM实现可以将当前资源上下文从线程解绑,并且绑定与新分区相关的资源上下文。

[0144] 现在参考图9,其示出了根据实施例的资源消耗管理实现。如所描绘的,资源消耗管理实现700可以包括资源消耗者705、资源域710、资源属性715、资源720、资源分配器725、约束740、通知750、触发器760和JDK文件I/O子系统730。如所示出的,触发器760可以由系统管理员例如在如图6中所示的资源消耗管理策略630内设置。附加地,资源消耗管理实现与资源上下文910相关联,资源上下文910可以通过参考当前线程的组件调用上下文920来设置。这可以确保约束或通知在被触发时与活动资源消耗相关联。

[0145] 根据实施例,当线程产生新线程时,JDK RM API可以自动继承在父线程中建立的资源上下文。

[0146] JDK资源度量

[0147] 根据实施例,各种组件可以得到分区特定的资源使用度量。RCM实现可以暴露用于监控的API,以便通过API获得分区特定的资源消耗度量。

[0148] CPU利用率

[0149] 根据实施例,JDK资源管理API支持测量由资源上下文的资源消耗者消耗的CPU时间。在多分区环境中,每个分区可以映射到资源上下文,并且线程将被绑定到资源上下文,以代表分区进行工作。利用CPU时间度量,RCM框架可以测量由分区利用的CPU时间。然而,通过绝对数字(例如,1小时、30分钟)将CPU使用表示为资源对系统管理员来说将不是有用的,因为CPU是无界(无限)的资源,并且对系统管理员来说在绝对CPU使用时间方面为分区指定限制/限额实际上价值有限(或者是非直观的)。因此,可以从CPU使用导出的CPU利用率资源类型可以通过百分比值(1-100)来表示。CPU利用率百分比指示相对于对系统进程可用的CPU的、由分区利用的CPU的百分比。使用分区的CPU消耗百分比和CPU负载因子来计算该值。

[0150] 根据实施例,CPU消耗百分比可以基于分区对CPU的消耗比上WLS过程对CPU的总消耗来计算。然而,考虑相对于WLS进程的总CPU消耗的、分区的CPU利用率的份额将不总是产生对系统管理员来说在设置策略时参考和/或引用的有用的衡量。

[0151] 例如,如果在WLS进程中仅有一个分区是活动的参与者,并且WLS进程正在其中运行的机器是轻负载的,则将分区的CPU使用相对于总WLS进程的CPU使用的简单比率视为该分区的CPU利用率将导致过度表示(该分区的非常高的CPU利用率值),并且不必要地惩罚该分区。

[0152] 作为附加的示例,假设在域中存在两个分区。分区-1的CPU消耗百分比为95,而分区-2的CPU消耗百分比为4,并且WLS进程正在其中运行的机器是轻负载的。即使在这种情况下,考虑分区的CPU使用相对于总WLS进程的CPU使用的简单比率将导致不必要地惩罚分区-1。

[0153] 根据实施例,为了将其中系统没有严重负载的上面的情景与其中系统由于应用服务器对CPU的大量使用或外部进程对CPU的大量使用而严重负载的其它情景区分开,可以对CPU消耗百分比应用附加的负载因子。这个附加的因子是CPU负载并且可以使用一个或多个API来计算。

[0154] 根据实施例,在服务器中仅有一个分区活动的情况下,负载因子可以基于ProcessCPULoad和CONTENTION_THRESHOLD。CONTENTION_THRESHOLD值可以是例如0.8(或CPU的80%)。

[0155] 根据实施例,在服务器中有多于一个分区活动的情况下,如果ProcessCPULoad值大于80(CONTENTION_THRESHOLD),则RCM可以使用ProcessCPULoad值来计算负载因子。这指示系统(例如,WLS)进程利用该周期的多于80%的CPU。

[0156] 根据实施例,在服务器中有多于一个分区活动的情况下,并且在ProcessCPULoad值小于0.8(或CPU的80%)的情况下,这可以指示系统(例如,WLS)进程不是CPU密集的。在这样的情况下,RCM可以检查SystemCPULoad是否大于0.8(或CPU的80%)。如果它大于0.8(CPU的80%),则SystemCPULoad可以被用来计算负载因子。这还指示系统中存在CPU密集的其它进程并且JVM进程被提供了有限的CPU。

[0157] 根据实施例,在服务器中有多于一个分区活动的上面的情况中的任一情况下,如果CPU使用大于CONTENTION_THRESHOLD,则它指示系统被充分负载,并且相对于对JVM进程可用的CPU而言该分区可能正在以牺牲其它分区为代价消耗CPU(例如,存在资源竞争)。

[0158] 根据实施例,在服务器中有多于一个分区活动的情况下,RCM可以基于ProcessCPULoad或SystemCPULoad是否已超过它们的CONTENTION_THRESHOLD来将ProcessCPULoad或SystemCPULoad考虑为负载因子。

[0159] 根据实施例,在服务器中有多于一个分区活动的情况下,当ProcessCPULoad和SystemCPULoad二者都小于它们的CONTENTION_THRESHOLD时,负载因子将基于ProcessCPULoad和CONTENTION_THRESHOLD。

[0160] 根据实施例,CPU负载因子可以协助确定对CPU的竞争,从而进一步量化基于JDK资源管理API导出的CPU消耗度量值。

[0161] 根据实施例,CPU利用率值可以被用于配置各种策略和追索,例如,通知、减慢、关闭和公平份额。由于来自JDK的关于CPU时间消耗的通知是消耗后的调用,因此对于CPU利用

率资源类型支持每请求的失败追索动作是不可能的。

[0162] 根据实施例，当CPU利用率值的衰减(decayed)平均值对于某一时间段始终高于阈值时，采取诸如通知、减慢、关闭等之类的追索动作。基于通用轮询机制的算法可以由保持的堆(Heap Retained)资源类型和CPU利用率(CPU Utilization)资源类型二者使用，这也有助于避免/忽略尖峰(spike)(零散的消耗激增/急降)。

[0163] 根据实施例，对CPU利用率资源类型支持公平份额策略，该公平份额策略基于服务器中所有分区的公平份额值工作。

[0164] 保持的堆

[0165] 根据实施例，API(例如，JDK资源管理API)可以支持测量特定资源上下文中的堆分配和保持的堆。用于这两种资源类型的回调是消耗后的。因此，不能主动执行动作来控制堆消耗。

[0166] 根据实施例，在堆分配的情况下，JDK RM提供用于为特定资源上下文运行堆分配的总数的回调。来自JDK的堆分配通知可以单调增加，并且通常不反映由该资源上下文使用的实际堆。WLDF规则可以被配置为通过使用堆分配分区范围度量来监控堆分配。

[0167] 根据实施例，在保持的堆的情况下，回调可以作为垃圾收集器活动的结果源自JDK。保持的堆的通知可以具有相关联的准确度级别，例如，置信因子。堆使用的高置信度测量可以在完全垃圾收集之后进行，从而产生高准确度通知。在那之后，测量堆使用的置信度降低。G1垃圾收集并发标记事件完成可以产生高置信度值，从而产生高准确度通知。较小的垃圾收集可以给出关于测量值的低置信度，从而产生较低准确度通知。直到下一个垃圾收集周期，由JDK报告的用于堆使用的值将具有非线性降低的置信因子。保持的堆的使用信息的准确度可以是或者精确(在用于资源上下文的垃圾收集之后标记留下的所有对象并且计算大小的总和)且昂贵的，或者较粗糙且较便宜的(计算分配给资源消耗者的G1区域)。垃圾收集周期之间的置信因子可以基于过去的分配历史并且假定稳态分配。

[0168] 根据实施例，在多分区环境中，每个分区可以映射到资源上下文，并且线程在代表分区执行工作时将被绑定到适当的资源上下文。这可以允许保持的堆读取的资源消耗对于该资源上下文被正确地考虑。利用来自JDK的保持的堆的回调，RCM框架可以测量由分区保持的堆的数量。如上文所描述的，具有高准确度的通知可以被用于对保持的堆的资源类型实施RCM策略。例如通知、减慢、关闭等的追索动作不是必须基于来自JDK的单个通知来调用。然而，当JDK的保持的堆的通知的衰减平均值对于某一时间段始终高于配置的阈值时，可以调用那些动作。设置默认时间段是可能的。例如，用于基于这种机制触发动作的默认时间段为100秒。基于通用轮询机制的算法可以由保持的堆(Heap Retained)资源类型和CPU利用率(CPU Utilization)资源类型二者使用，这也有助于避免/忽略尖峰(零散的消耗激增/急降)。

[0169] 根据实施例，对保持的堆资源类型支持公平份额策略，该公平份额策略基于服务器中的所有分区的公平份额值工作。

[0170] 策略评估、追索动作/通知

[0171] 根据实施例，分区(例如，多分区环境内的分区)可以与资源管理器相关联。当没有资源管理器与分区相关联时，那么该分区不受任何资源消耗管理策略的约束，并且其资源的使用是不受约束的。

[0172] 根据实施例,对于为分区定义的每个资源管理器,RCM框架可以为表示ResourceId的适当的资源属性(ResourceAttribute)实例化JSR-284资源域。JSR 284约束和通知基于用户/系统管理员指定的触发器针对该资源域而被注册。

[0173] 根据实施例,系统管理员可以指定一个或多个触发器来管理资源的使用。每个触发器可以包含名称、值对和动作。触发器可以由系统管理员指定,以在资源消耗级别达到由系统管理员设置的值时实现一系列动作。一旦使用达到给定值,则WLS RCM框架执行相关联的动作。

[0174] 根据实施例,通过动作元素指定的追索动作类型可以是以下四种类型之一:通知、减慢、失败、关闭。

[0175] 根据实施例,由WLS RCM框架执行以实现由系统管理员指定的追索动作类型的活动的具体集合可以是用户定义的或从预先存在的矩阵中选择的。

[0176] 例如,根据实施例,系统管理员可以希望分区P1仅使用2000个文件描述符。一旦使用达到指定值,则系统管理员希望阻止该分区使用任何更多的文件描述符。这样的触发器可以看上去像这样:

[0177] ＜trigger＞

[0178] ＜value＞2000＜/value＞

[0179] ＜action＞fail＜/action＞

[0180] ＜/trigger＞

[0181] 指定的失败动作可以映射为向文件打开请求抛出IOException。WLS RCM框架可以通过在表示用于分区P1的FILE_OPEN资源类型的资源属性的资源域中注册约束来拦截文件打开请求。

[0182] 根据实施例,由系统执行的追索动作中的一些追索动作可以在消耗行为期间发生(例如,为打开文件描述符资源设置的FAIL触发器导致在新文件的创建期间抛出IOException)。附加地,一些动作可以在当前消耗请求之后执行(例如,关于CPU和堆使用的通知是异步的,并且例如,用于基于那些资源的触发器的“关闭”追索动作(其导致相关分区的关闭)被异步执行)。

[0183] 根据实施例,同步动作可以通过JSR-284约束来实现,而异步动作可以通过JSR-284通知来实现。

[0184] 公平份额策略实现

[0185] 根据实施例,对于系统管理员来说在质量上和数量上确保向多租户应用服务器环境中的不同资源消耗者(例如,多分区环境中的不同分区)“公平”分配启用RM的共享资源是所期望的。

[0186] 根据实施例,如果对具有相同公平份额值的两个资源域的资源分配不相等/不均匀,或者换句话说,相比于另一个域偏向一个域的消耗者中的一些消耗者,则对这两个资源域的资源分配被认为是不公平的。这两个资源域可以具有变化的资源请求模式,并且始终保证瞬时的公平份额可以是困难的。然而,尤其当两个资源域争夺/竞争资源,从而将跨分区的资源使用推向对于该资源的系统最大限制时,仍然期望将资源分配维持在其配置的公平份额的比率中。

[0187] 根据实施例,资源分配中的公平性可以在某一时间段内计算,以考虑来自各个资

源域的资源消耗请求的变化。因为公平性可以随时间计算和实施，所以公平性不一定意味着在特定时间窗中的分配相等(当公平份额相等时)。公平份额策略实现在意识到特定资源消耗者尚未使用窗口中的它们的资源份额时，可以暂时分配大于该消耗者的指定公平份额的资源份额。然而，随着时间的推移，分配可以被调整以使得它们与用户指定的公平份额一致。

[0188] 公平份额策略可以暂时过度分配的程度还可以基于资源的类型。例如，在诸如CPU利用率之类的“无界”资源的情况下，公平份额实现可以使最近非常活跃的资源消耗者完全缺乏资源，以向其份额落后的消耗者提供资源。然而，在诸如堆分配/保持的堆之类的“有界”、“可丢弃”和“不可撤销”资源的情况下，公平份额策略实现可能不允许来自其份额已落后的资源消耗者的所有堆分配请求(因为任何准予的分配以后不能被撤销)。

[0189] 根据实施例，公平份额策略可以向系统管理员提供以下保证。当存在对于资源的“竞争”并且“在某一时间段内”存在两个资源域的“均匀负载”时，分配给两个域的资源的份额“大致”按照系统管理员为那两个域配置的公平份额。

[0190] 根据实施例，对于CPU利用率和保持的堆资源类型支持公平份额策略。对于这两种资源类型，来自JDK的通知可以在消耗后提供。当这种情况发生时，公平份额策略不能确切地在对资源消耗的请求的时间处被应用。在保持的堆的情况下，来自JDK的通知可以被链接到垃圾收集(GC)周期，并且可以在进行堆消耗请求的时间点之后发生。在CPU利用率的情况下，利用率可以在固定时间窗之后确定。

[0191] 根据实施例，对资源消耗的请求可以间接地链接到由分区的工作管理器为分区进行的工作。因此，如果可以基于分区的资源消耗控制为分区执行的工作实例，则仍然可以确保该资源的公平份额。

[0192] 根据实施例，在保持的堆的情况下，由于之前已分配的任何堆不能由系统强制撤销，因此公平份额策略通过控制分区的工作管理器公平份额来控制堆的分配。

[0193] RCM配置

[0194] 根据实施例，资源消耗管理器(RCM)策略和追索动作可以由诸如系统管理员之类的管理员配置。系统管理员还可以创建可重用的定义，以及创建为分区定制的策略。

[0195] 根据实施例，系统管理员可以创建资源管理器定义，然后该资源管理器定义可以跨多个分区被重用。例如，在SaaS用例中，管理域的系统管理员可以具有他们想要应用到他们为客户的特定“类”(例如，铜类、银类、金类)创建的每个新分区的一组RCM策略。例如，铜客户的分区可以具有关于堆和CPU使用的某些受管理资源策略(例如，堆被限制为2GB)。当新的铜客户被加入(on-board)并且分区P2以他们的名义被创建时，系统管理员有可能在域级创建铜资源管理策略，并且在铜分区的创建期间指向该策略。然后，在铜资源管理器中包含的所有策略配置被应用到P2的资源管理策略。资源管理器被添加到的每个分区得到在该配置中定义的受管理资源策略的“拷贝”。因此，例如，铜资源管理器被应用到P2和P3分区。P2和P3二者各自被允许2GB堆使用。

[0196] 根据实施例，系统管理员可以创建为分区定制的策略。例如，在整合(consolidation)用例中，系统管理员可以为为特定部门创建的分区指定唯一的资源消耗管理策略。为该分区创建新的分区范围的资源管理策略是可能的。例如，系统管理员为公司的CEO创建分区CEO。然后，系统管理员还可以定义仅适用于与系统中定义的其它分区不同

的那个分区的具体的受管理资源策略(例如,给予该分区相对高的公平份额)。

[0197] 图10示出了根据实施例的应用服务器环境中的资源隔离和消耗。图10描绘了包含域610的应用服务器环境600。域可以包含分区A640和分区B 650。分区A和分区B分别包含资源组642和652,资源组642和652分别与虚拟目标A 645和虚拟目标B 655相关联。如图6中所示,当例如分区运行需要使用共享资源的应用时,分区A或分区B还可以访问共享资源660。共享资源660可以包括但不限于JDK资源,诸如CPU、堆、网络和文件。

[0198] 如上文所描述的,系统管理员可以分别定义分区特定的资源消耗管理策略1005和1015。这些分区特定的资源消耗管理策略可以被配置为分别包含分区特定的资源预留1006和1016、分别包含分区特定的资源预留1007和1017、以及分别包含分区特定的资源通知1008和1018。

[0199] 根据实施例,分区特定的资源消耗管理策略可以由系统管理员或具有足够许可的另一个人员来配置,以在分区A或分区B相对于共享资源满足由资源管理策略设置的已配置条件时执行行动或任务。这些行动或任务可以包括但不限于约束或通知。约束的示例包括减慢或停止分区对共享资源的使用。通知的示例包括但不限于日志记录或向管理员提供通知。

[0200] 以下是用于指定分区的资源管理策略的定义的说明性示例:

```
<domain>
...
    <!--Define RCM Configuration -->
```

[0201]

```
    <resource-management>
        <resource-manager>
            <name>Gold</name>
```

```
<file-open>
    <trigger>
        <name>Gold2000</name>
        <value>2000</value><!-- in units-->
        <action>shutdown</action>
    </trigger>
    <trigger>
        <name>Gold1700</name>
        <value>1700</value>
        <action>slow</action>
    </trigger>
```

[0202]

```
    <trigger>
        <name>Gold1500</name>
        <value>1500</value>
        <action>notify</action>
    </trigger>
</file-open>
<heap-retained>
    <trigger>
        <name>Gold2GB</name>
        <value>2097152</value>
        <action>shutdown</action>
```

```
            </trigger>
            <fair-share-constraint>
                <name>FS-GoldShare</name>
                <value>60</value>
            </fair-share-constraint>
        </heap-retained>
    </resource-manager>
    <resource-manager>
        <name>Silver</name>
        <file-open>
            <trigger>
```

[0203]

```
                <name>Silver1000</name>
                <value>1000</value><!-- in units-->
                <action>shutdown</action>
            </trigger>
            <trigger>
                <name>Silver700</name>
                <value>700</value>
                <action>slow</action>
            </trigger>
            <trigger>
                <name>Silver500</name>
```

```
                    <value>500</value>
                    <action>notify</action>
                </trigger>
            </file-open>
        </resource-manager>
    </resource-management>
    <partition>
        <name>Partition-0</name>
        <resource-group>
            <name>ResourceTemplate-0_group</name>
            <resource-group-template>ResourceTemplate-
```

[0204]

```
0</resource-group-template>
        </resource-group>
        ...
        <partition-id>1741ad19-8ca7-4339-b6d3-
78e56d8a5858</partition-id>
        <!-- RCM Managers are then targeted to Partitions during
partition creation time or later by system administrators -->
        <resource-manager-ref>Gold</resource-manager-ref>
    ...
    </partition>
...
```

[0205] </domain>

[0206] 根据实施例,可以支持所有RCM元素的动态重新配置。在某些情况下,对于重新配置资源消耗管理策略来说不需要重新启动分区和服务器。

[0207] 资源隔离和消耗

[0208] 现在参考图11,其描绘了根据实施例的、用于应用服务器环境中的资源隔离和消

耗的方法的流程图。示例性方法可以在步骤1110处开始，其中在包括在其上执行的应用服务器环境的一个或多个计算机处提供可以在应用服务器环境内使用的多个资源、一个或多个分区，其中每个分区提供域的管理和运行时细分。该方法可以在步骤1120处继续，其中配置资源消耗管理模块，以监控每个分区对所述多个资源的使用，资源消耗管理模块包括包含资源预留、资源约束和资源通知的组中的至少一个成员。

[0209] 可以使用包括根据本公开的教导编程的一个或多个处理器、存储器和/或计算机可读存储媒介的一个或多个常规的通用或专用数字计算机、计算设备、机器或微处理器来方便地实现本发明。如对软件领域的技术人员将明显的，基于本公开的教导，熟练的程序员可以容易地准备适当的软件编码。

[0210] 在一些实施例中，本发明包括计算机程序产品，该计算机程序产品是具有可以被用于对计算机进行编程以执行本发明的过程中的任何过程的存储在其上/其中的指令的非暂态存储介质或计算机可读介质(媒介)。存储介质可以包括但不限于任何类型的盘，包括软盘、光盘、DVD、CD-ROM、微型驱动器以及磁光盘、ROM、RAM、EPROM、EEPROM、DRAM、VRAM、闪速存储器设备、磁卡或光卡、纳米系统(包括分子存储器IC)，或适合于存储指令和/或数据的任何类型的媒介或设备。

[0211] 出于说明和描述的目的提供了本发明的上述描述。它并不旨在是详尽的或者将本发明限制到所公开的精确形式。许多修改和变化对于本领域技术人员来说将是明显的。实施例被选择和描述以便于最佳地解释本发明的原理及其实践应用，从而使得本领域其他技术人员能够对于各种实施例以及利用适合所设想的特定使用的各种修改来理解本发明。旨在由以下权利要求和它们的等价物来限定本发明的范围。

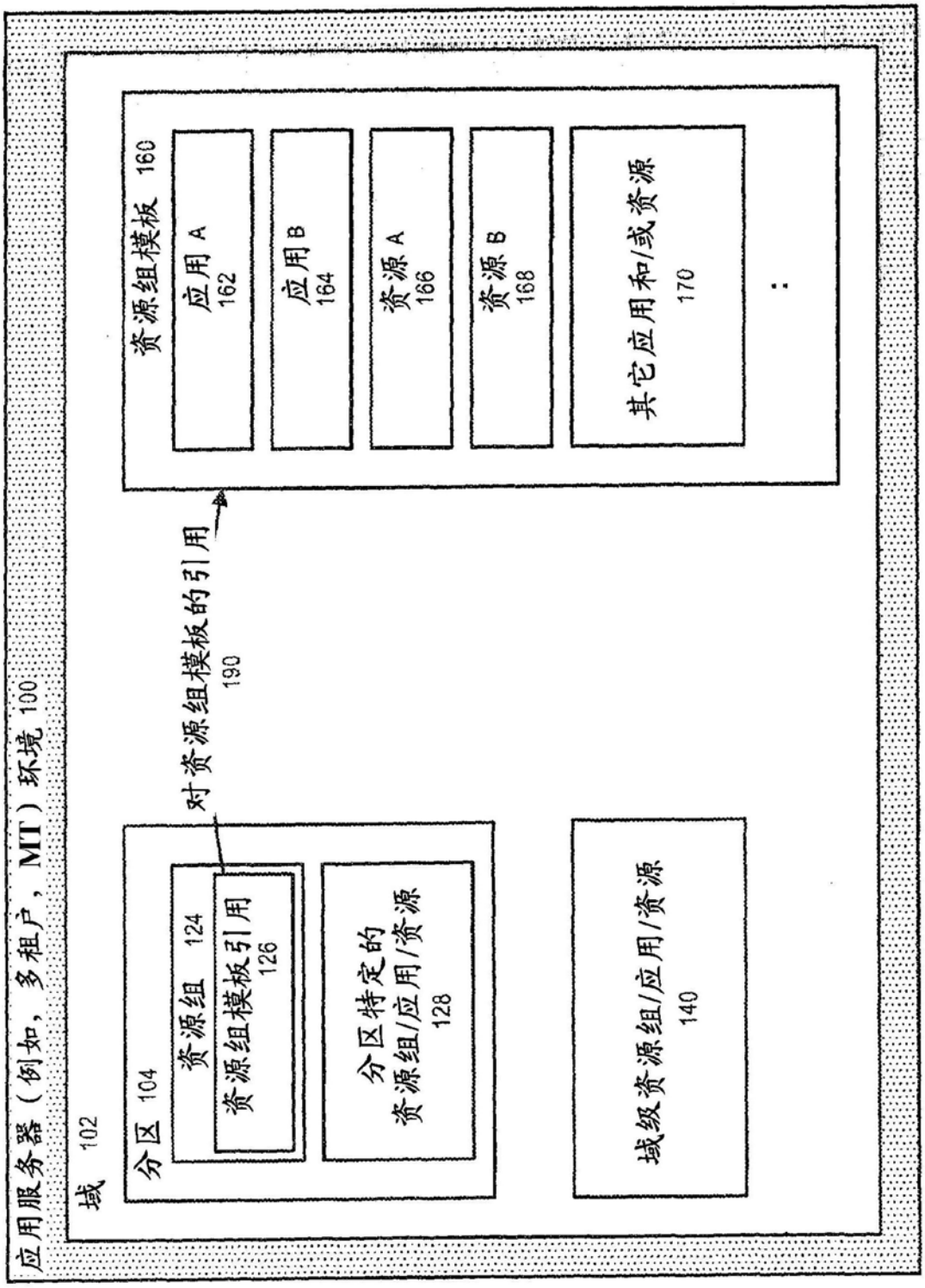

应用服务器（例如，多租户，MT）环境 100
域 102
分区 104
资源组 124
资源组模板引用 126
分区特定的资源组/应用/资源 128
域级资源组/应用/资源 140
对资源组模板的引用 190
资源组模板 160
应用 A 162
应用 B 164
资源 A 166
资源 B 168
其它应用和/或资源 170

图1

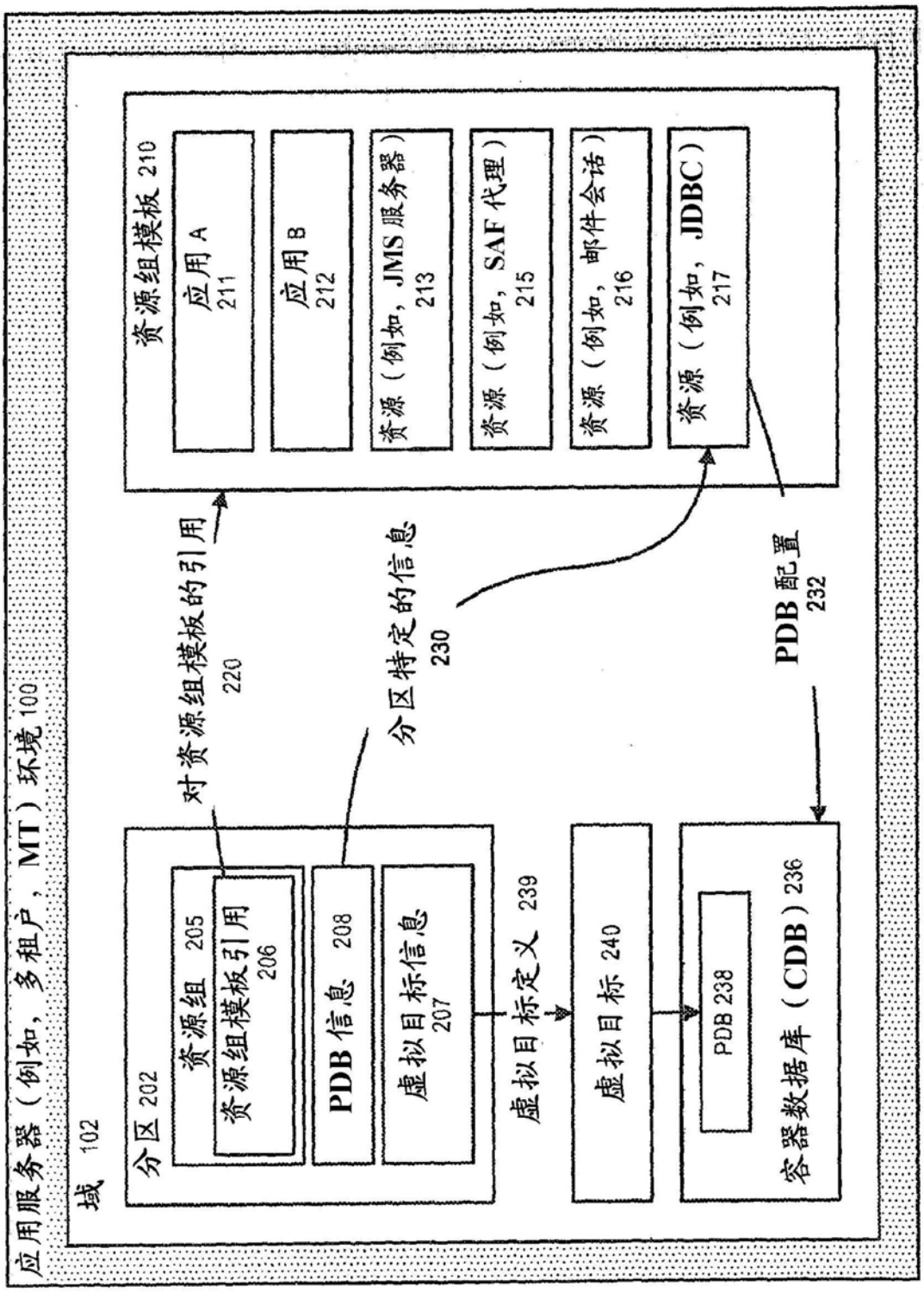

应用服务器（例如，多租户，MT）环境 100
域 102
分区 202
资源组 205
资源组模板引用 206
PDB 信息 208
虚拟目标信息 207
虚拟目标定义 239
虚拟目标 240
PDB 238
容器数据库（CDB）236
对资源组模板的引用 220
分区特定的信息 230
PDB 配置 232
资源组模板 210
应用 A 211
应用 B 212
资源（例如，JMS 服务器）213
资源（例如，SAF 代理）215
资源（例如，邮件会话）216
资源（例如，JDBC）217

图2

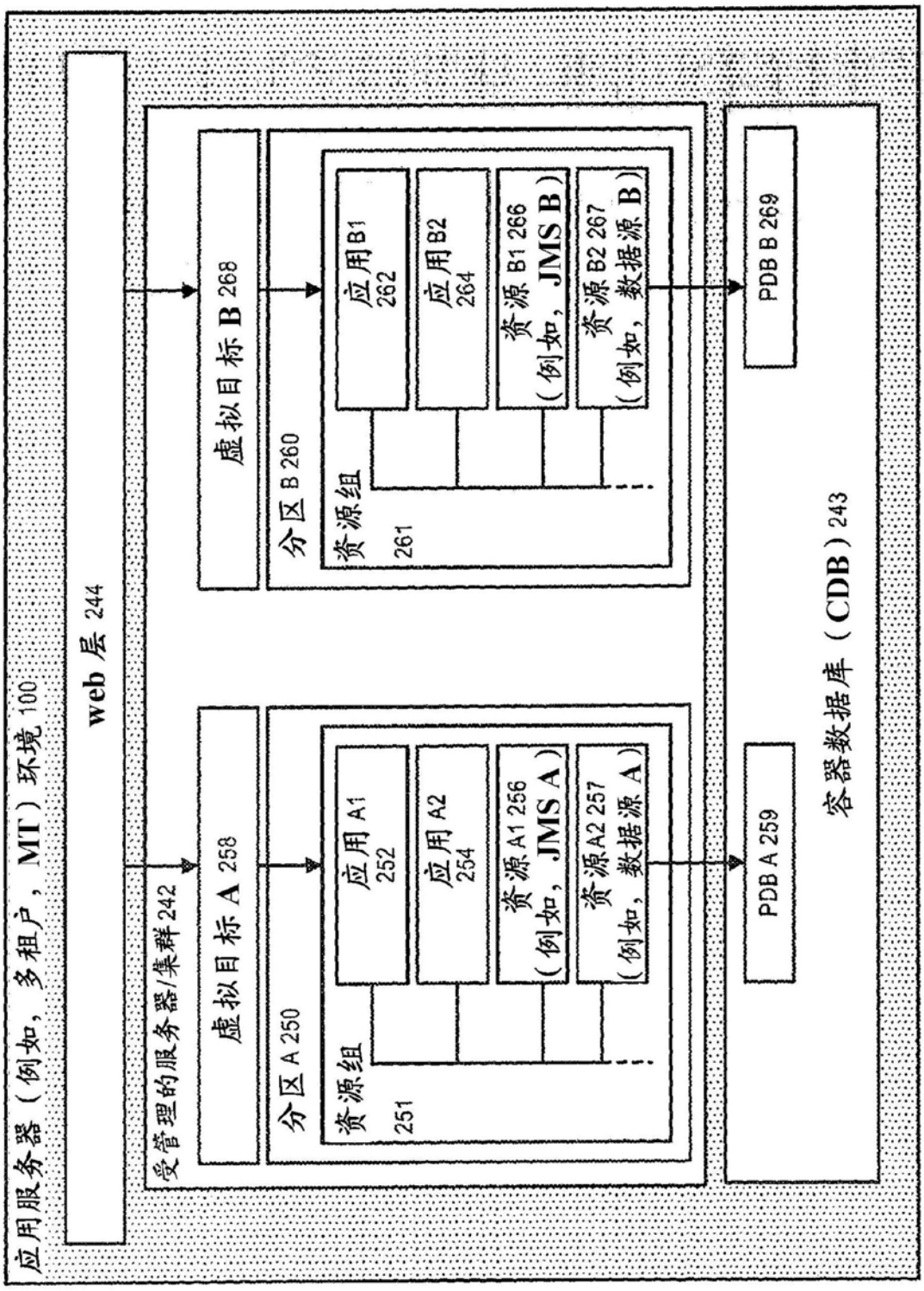

应用服务器（例如，多租户，MT）环境 100
web 层 244
受管理的服务器/集群 242
虚拟目标 A 258
虚拟目标 B 268
分区 A 250
分区 B 260
资源组 251
资源组 261
应用 A1 252
应用 A2 254
资源 A1 256 （例如，JMS A）
资源 A2 257 （例如，数据源 A）
应用 B1 262
应用 B2 264
资源 B1 266 （例如，JMS B）
资源 B2 267 （例如，数据源 B）
PDB A 259
PDB B 269
容器数据库（CDB）243

图3

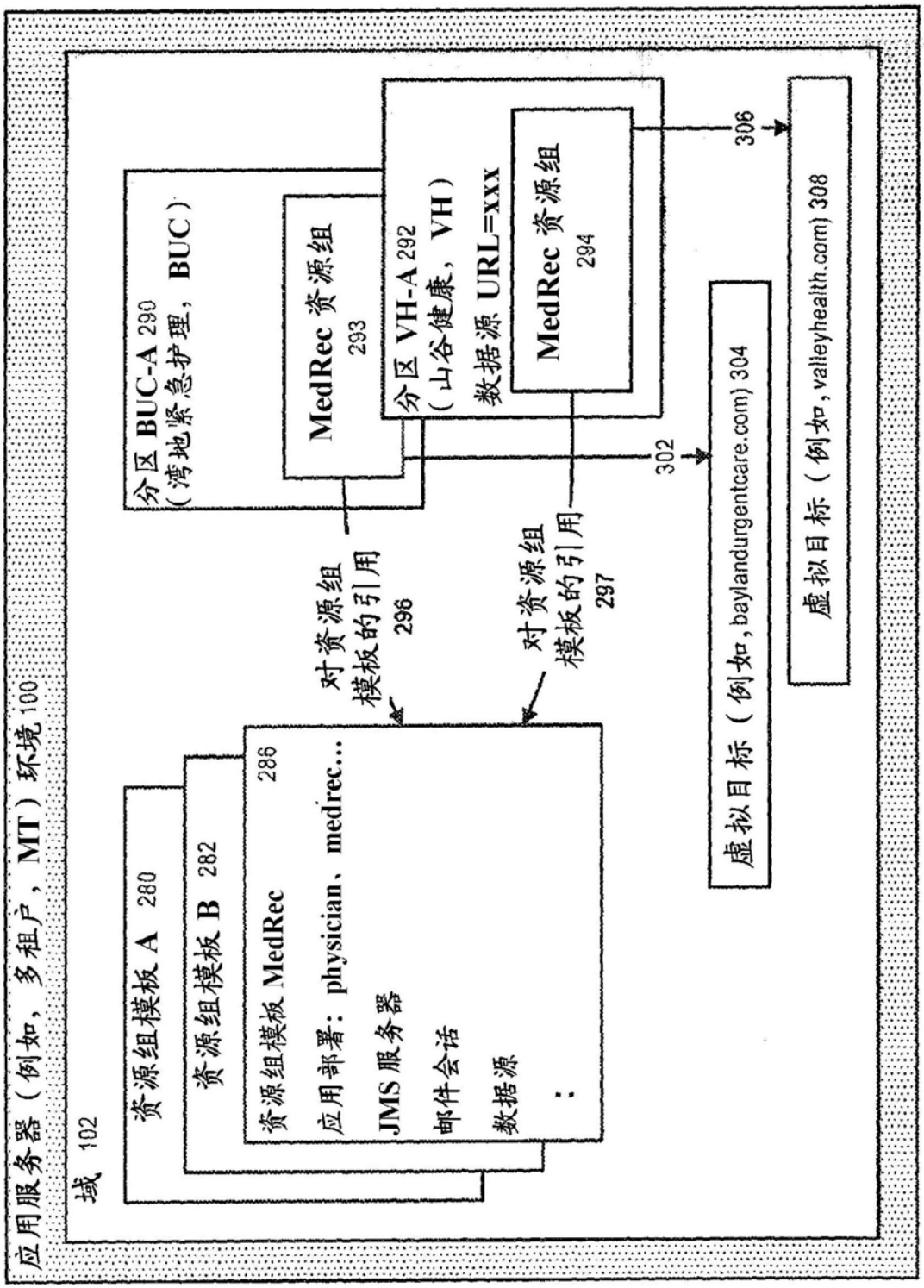

应用服务器（例如，多租户，MT）环境 100
域 102
资源组模板 A 280
资源组模板 B 282
资源组模板 MedRec 286
应用部署：physician、medrec...
JMS 服务器
邮件会话
数据源
:
对资源组模板的引用 296
对资源组模板的引用 297
分区 BUC-A 290
（湾地紧急护理，BUC）
MedRec 资源组 293
分区 VH-A 292
（山谷健康，VH）
数据源 URL=XXX
MedRec 资源组 294
302
虚拟目标（例如，baylandurgentcare.com）304
306
虚拟目标（例如，valleyhealth.com）308

图4

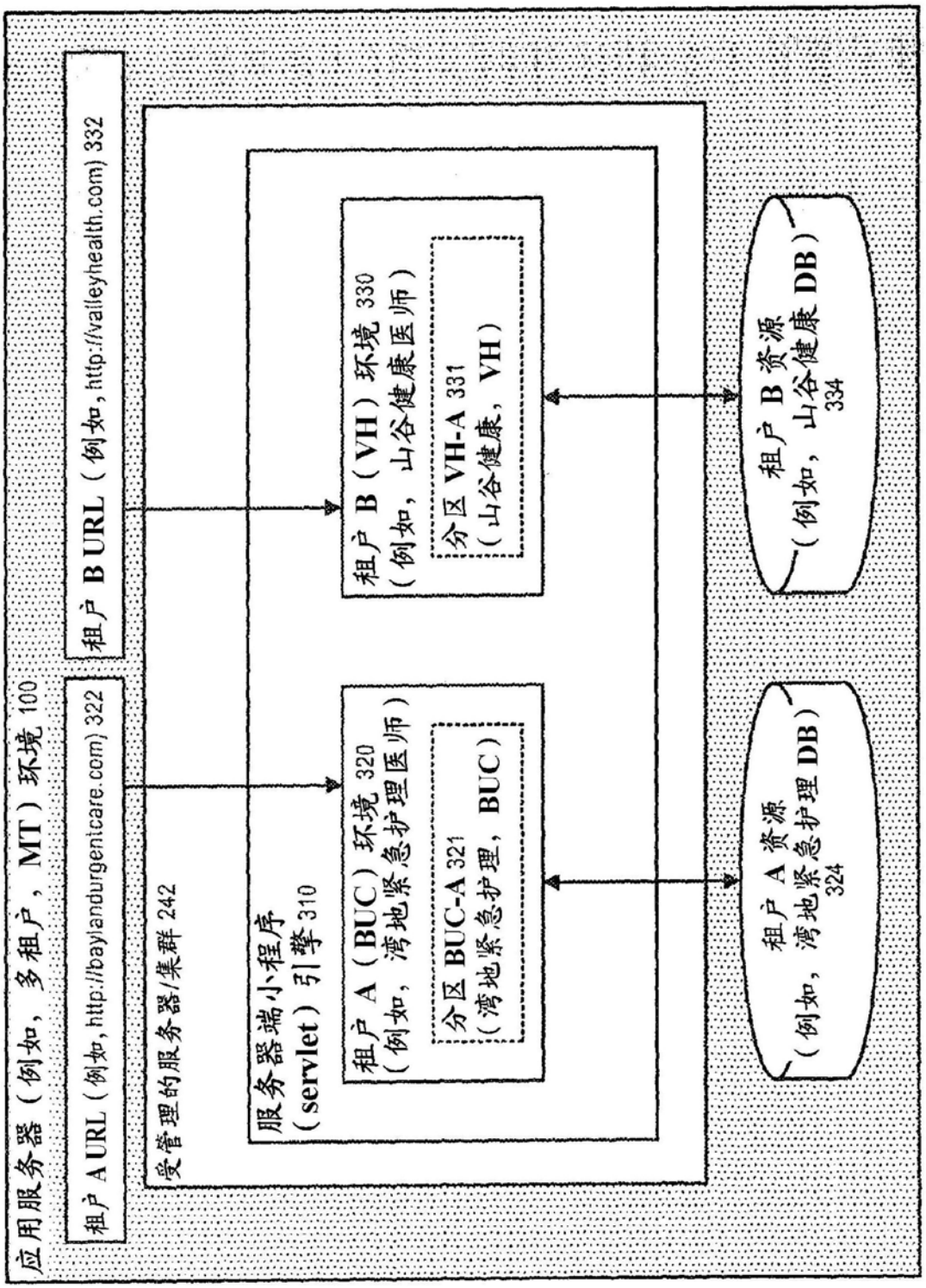

应用服务器（例如，多租户，MT）环境 100
租户 A URL（例如，http://baylandurgenicare.com）322
租户 B URL（例如，http://valleyhealth.com）332
受管理的服务器/集群 242
服务器端小程序
（servlet）引擎 310
租户 A（BUC）环境 320
（例如，湾地紧急护理医师）
分区 BUC-A 321
（湾地紧急护理，BUC）
租户 B（VH）环境 330
（例如，山谷健康医师）
分区 VH-A 331
（山谷健康，VH）
租户 A 资源
（例如，湾地紧急护理 DB）
324
租户 B 资源
（例如，山谷健康 DB）
334

图5

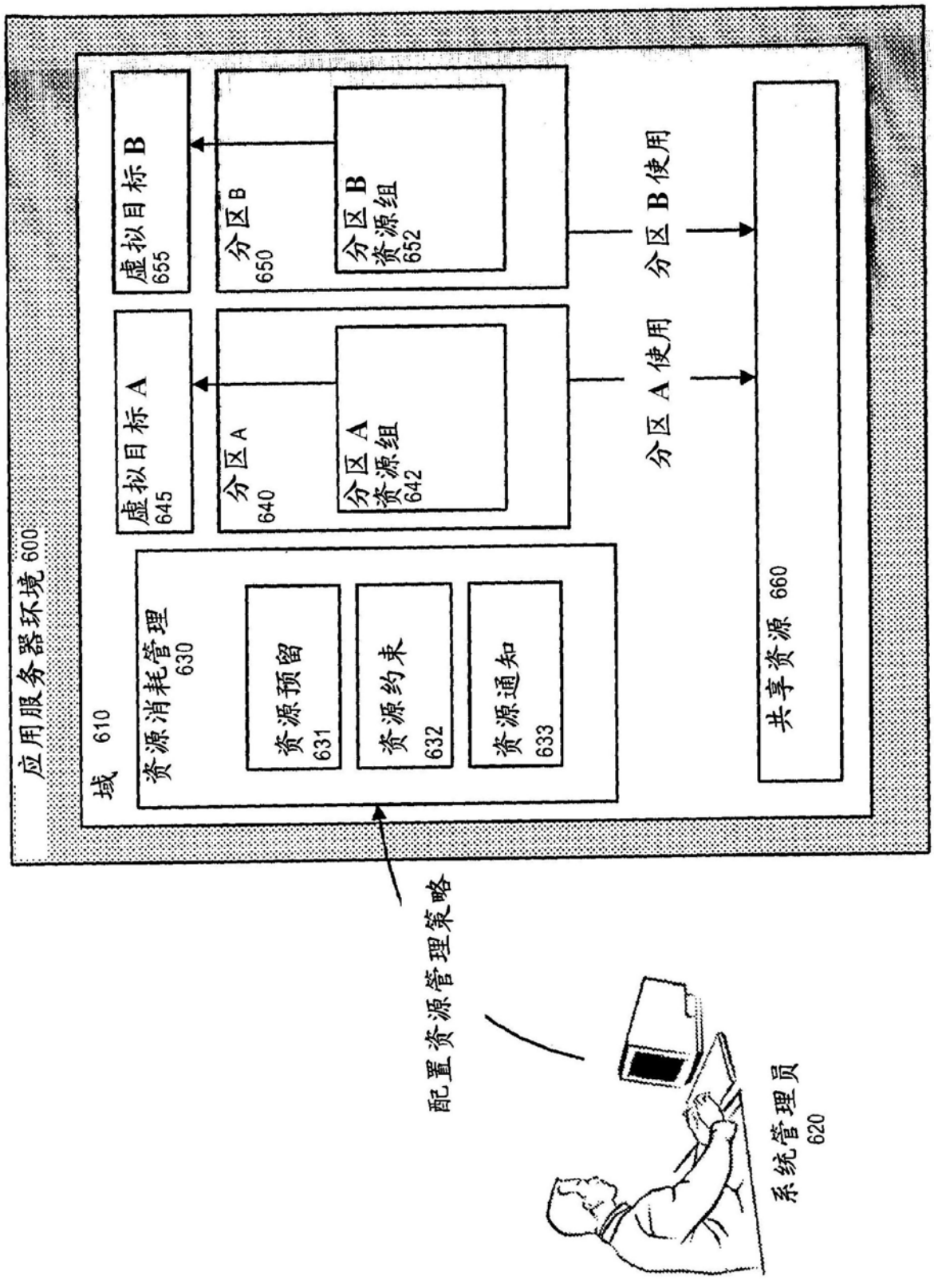

应用服务器环境 600
域 610
资源消耗管理 630
资源预留 631
资源约束 632
资源通知 633
虚拟目标 A 645
分区 A 640
分区 A 资源组 642
虚拟目标 B 655
分区 B 650
分区 B 资源组 652
分区 A 使用
分区 B 使用
共享资源 660
配置资源管理策略
系统管理员 620

图6

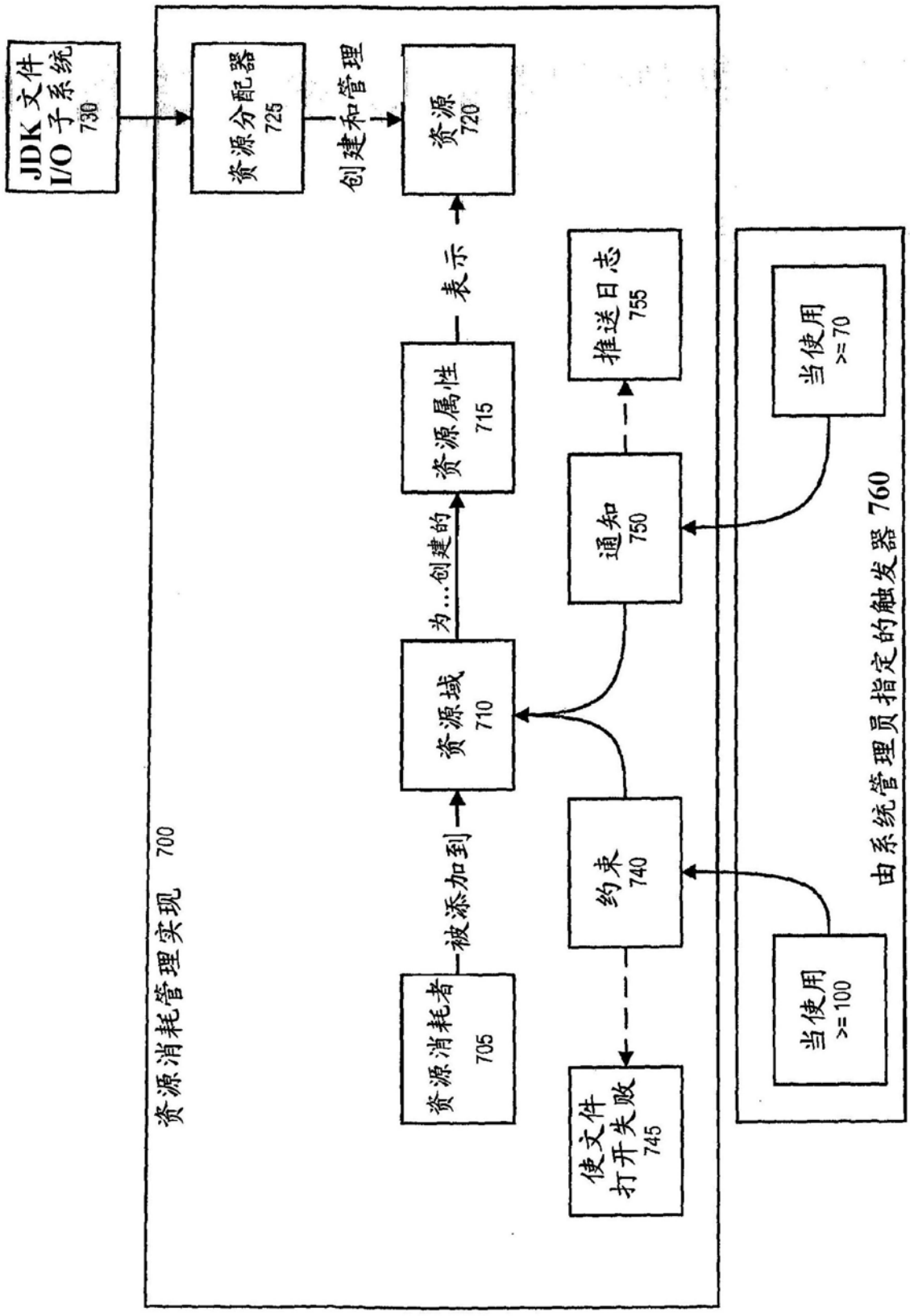

JDK 文件 I/O 子系统 730
资源分配器 725
创建和管理
资源 720
资源消耗管理实现 700
资源消耗者 705
被添加到
资源域 710
为...创建的
资源属性 715
表示
约束 740
通知 750
推送日志 755
使文件打开失败 745
当使用 >= 100
当使用 >= 70
由系统管理员指定的触发器 760

图7

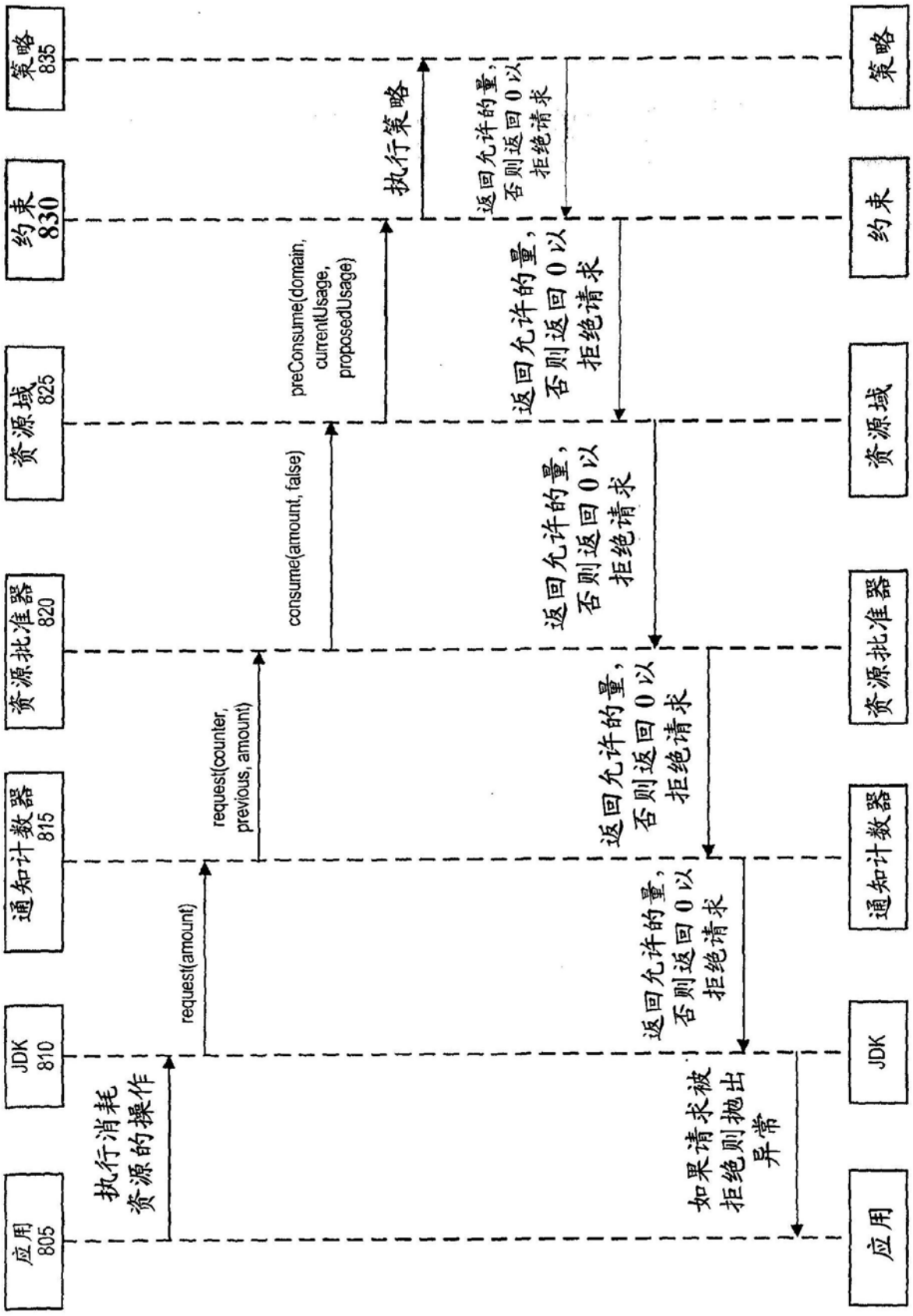

应用 805
JDK 810
通知计数器 815
资源批准器 820
资源域 825
约束 830
策略 835
执行消耗资源的操作
request(amount)
request(counter, previous, amount)
consume(amount, false)
preConsume(domain, currentUsage, proposedUsage)
执行策略
返回允许的量，否则返回0以拒绝请求
返回允许的量，否则返回0以拒绝请求
返回允许的量，否则返回0以拒绝请求
返回允许的量，否则返回0以拒绝请求
如果请求被拒绝则抛出异常

图8

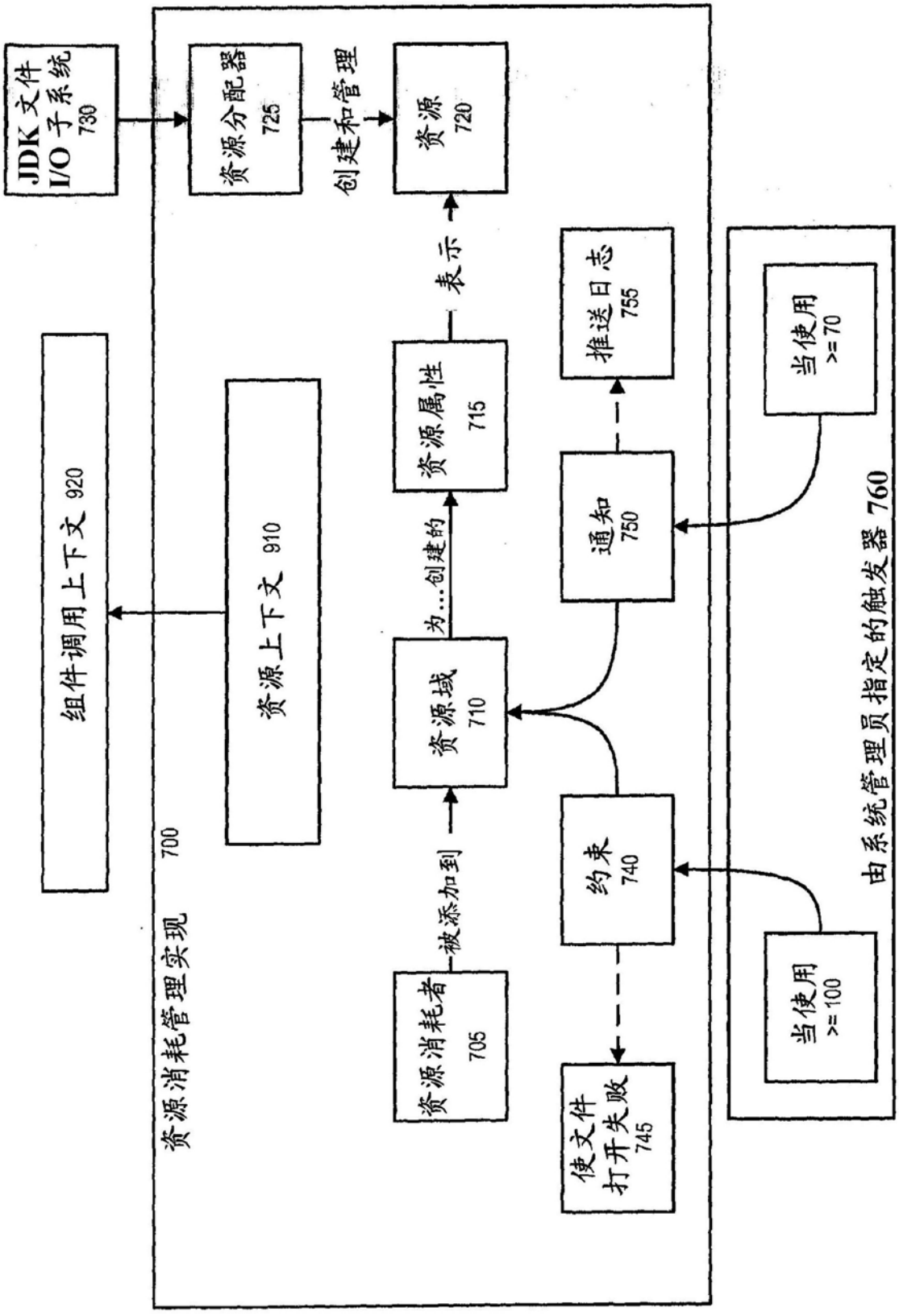

JDK 文件 I/O 子系统 730
资源分配器 725
创建和管理
资源 720
表示
资源属性 715
为...创建的
资源域 710
被添加到
资源消耗者 705
组件调用上下文 920
资源上下文 910
资源消耗管理实现 700
推送日志 755
通知 750
约束 740
使文件打开失败 745
当使用 >= 70
当使用 >= 100
由系统管理员指定的触发器 760

图9

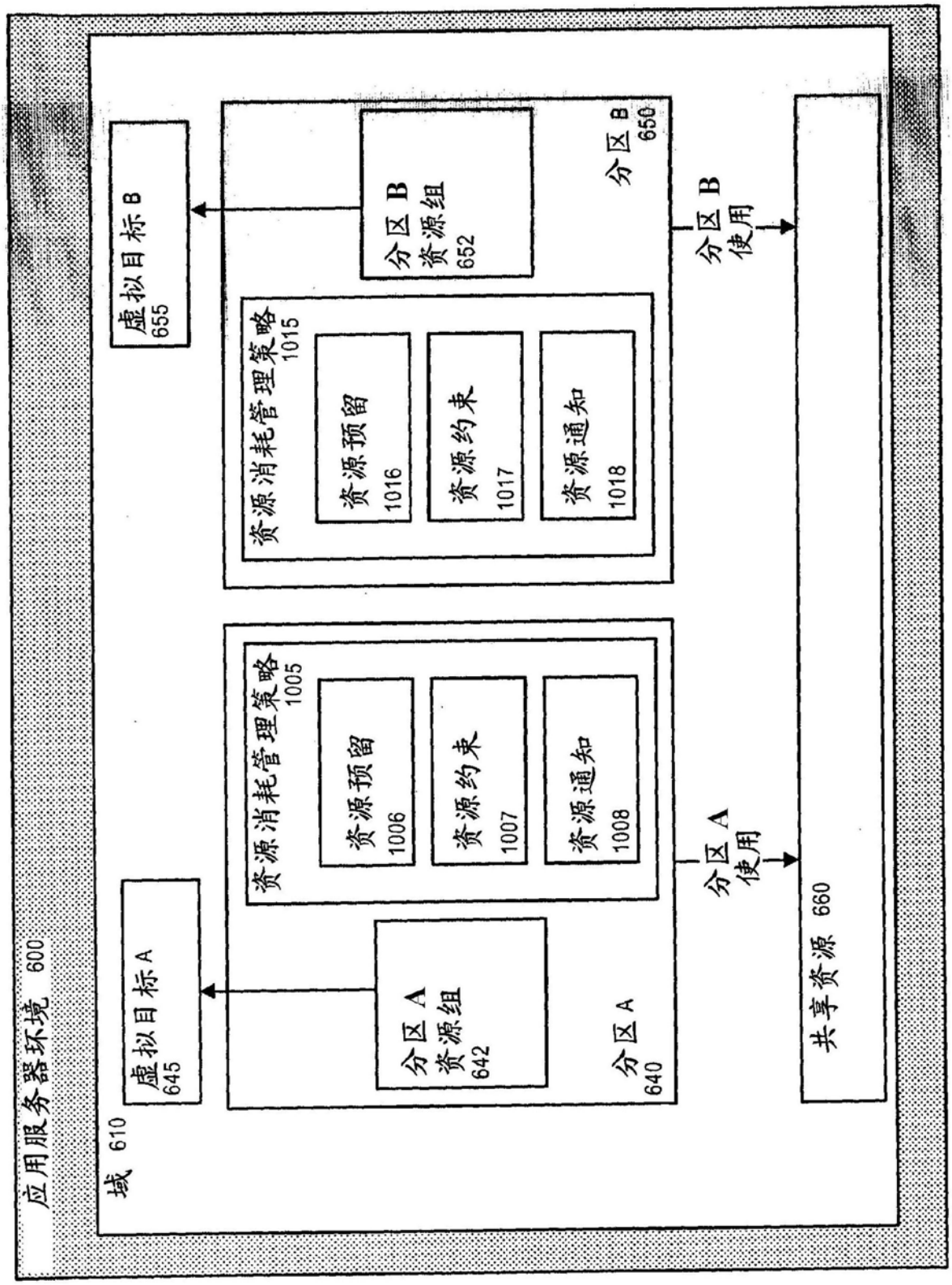

应用服务器环境 600
域 610
虚拟目标 A 645
虚拟目标 B 655
分区 A 资源组 642
分区 B 资源组 652
分区 A 640
分区 B 650
资源消耗管理策略 1005
资源预留 1006
资源约束 1007
资源通知 1008
资源消耗管理策略 1015
资源预留 1016
资源约束 1017
资源通知 1018
分区 A 使用
分区 B 使用
共享资源 660

图10

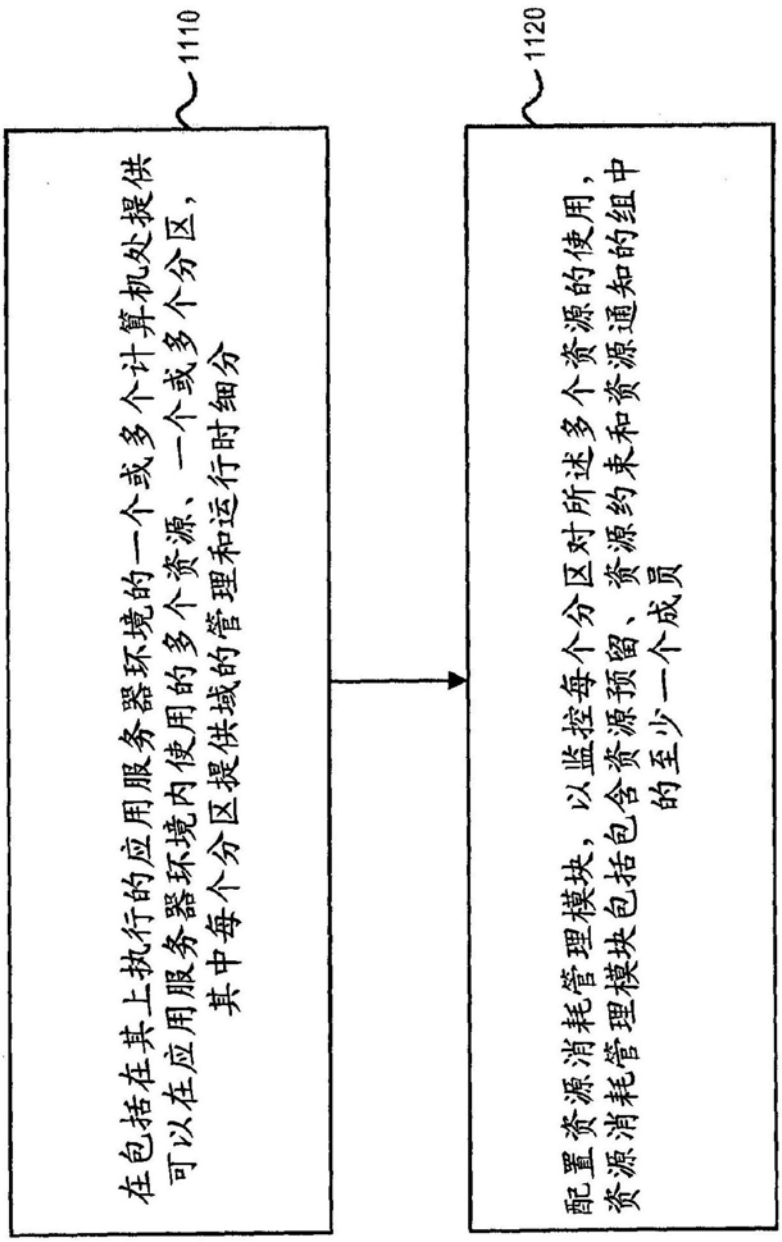

1110
在包括在其上执行的应用服务器环境的一个或多个计算机处提供可以在应用服务器环境内使用的多个资源、一个或多个分区，其中每个分区提供域的管理和运行时细分
1120
配置资源消耗管理模块，以监控每个分区对所述多个资源的使用，资源消耗管理模块包括包含资源预留、资源约束和资源通知的组中的至少一个成员

图11