

[19] 中华人民共和国国家知识产权局

[51] Int. Cl.
G06F 17/30 (2006.01)



[12] 发 明 专 利 说 明 书

专利号 ZL 200410055008.4

[45] 授权公告日 2009 年 9 月 2 日

[11] 授权公告号 CN 100535894C

[22] 申请日 2004.6.4

[21] 申请号 200410055008.4

[30] 优先权

[32] 2003. 6. 6 [33] US [31] 10/456,139

[73] 专利权人 微软公司

地址 美国华盛顿州

[72] 发明人 D·松金 M·沃里斯

[56] 参考文献

CN1323424A 2001.11.21

US6061515A 2000.5.9

US5797137A 1998.8.18

CN1271439A 2000.10.25

审查员 王 骞

[74] 专利代理机构 上海专利商标事务所有限公司
代理人 陈 斌

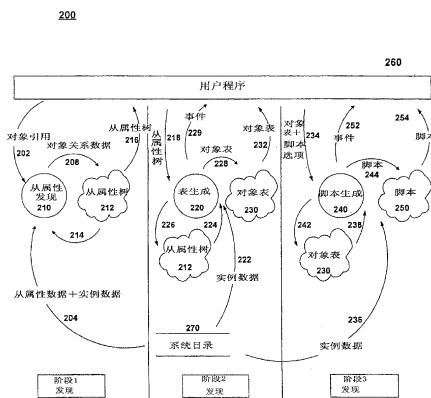
权利要求书 3 页 说明书 17 页 附图 6 页

[54] 发明名称

数据库对象脚本生成方法和系统

[57] 摘要

在关系数据库中为不同复杂程度的对象生成脚本的方法包括从通过系统的单一或多个指向对象的指针生成一个分级对象树。在分级树的构成中将重复对象引用删除。从分级树中生成从属性列表。所述从属性列表代表一个对象的线性表，所述对象以其将被生成以满足相关限定的顺序排列。从从属性列表中生成一个脚本，所述从属性列表允许调度不同复杂程度的对象到目标数据库中。通过实例化从属性列表并且调用一个事先构建的对象脚本方法生成脚本。每一个阶段都能够单独操作。



- 1.一种在关系数据库中为多个对象引用生成SQL脚本的方法，所述方法包括：
接收关系数据库中将被脚本化的至少一个对象引用；
侦测所述至少一个对象引用的关系从属性；
构建一个包括所述至少一个对象引用的分级对象树；
基于侦测到的关系从属性从所述分级对象树中派生出一从属性列表，其中所述从属性列表可通过用户程序编辑，并且所述从属性列表包括表示将被创建的对象顺序的线性列表；以及
响应所述从属性列表生成所述SQL脚本，其中所述SQL脚本使用在一目标关系数据库中的关系数据库对象。
- 2.如权利要求1所述的方法，其中构建过程包括删去重复对象引用和输入非重复对象引用及相关元数据到一个分级对象树。
- 3.如权利要求1所述的方法，其中构建过程包括触发能够表明进度并且提供可选择的对象操作的事件。
- 4.如权利要求3所述的方法，其中可选择的对象操作包括过滤所述至少一个对象引用，其中被一个或多个用户和/或一个或多个程序选择的一个对象引用能够被删除。
- 5.如权利要求4所述的方法，其中可选择的对象操作包括删除一个已选对象引用以及所有依赖于已选对象引用的后续引用。
- 6.如权利要求1所述的方法，其中从属性列表包括所述线性列表，所述线性列表描述了为了满足从属性限定而生成的对象的顺序。
- 7.如权利要求1中所述的方法，其中所述派生步骤递归地遍历较低层对象引用的从属性，并且提供可选择的对象操作，其中较低层从属性能被删除。
- 8.如权利要求1所述的方法，其中所述生成包括应用脚本选项以改变所述SQL脚本。
- 9.如权利要求1所述的方法，其中关系数据库是一个SQL数据库。
- 10.如权利要求1所述的方法，其中关系从属性是父亲—儿子、儿子—孙子和孙子—曾孙中的一个或多个。
- 11.如权利要求1所述的方法，其中至少一个对象引用包括一个使用包括服务器

/数据库/表格的格式的统一资源名称。

12.如权利要求1的方法,其中所述构建过程进一步包括触发一个或多个允许进展监控和对象操作中的至少一个的事件。

13.如权利要求12的方法,其中对象操作包括加入、修改和删除对象引用中的至少一个。

14.一种在关系数据库中为一个或多个对象引用生成SQL脚本的计算机系统,所述计算机系统包括:

用于接收关系数据库中将被脚本化的至少一个对象引用的装置;

用于侦测所述至少一个对象引用的关系从属性的装置;

用于构建一个包括所述至少一个对象引用的分级对象树的装置;

用于基于侦测到的关系从属性从所述分级对象树中派生出一个从属性列表的装置,其中所述从属性列表可通过用户程序编辑,并且所述从属性列表包括表示将被创建的对象顺序的线性列表;并且

用于响应所述从属性列表生成所述SQL脚本的装置,其中所述SQL脚本使用在一目标关系数据库内的关系数据库对象。

15.如权利要求14所述的计算机系统,其中用于构建的装置包括用于删去重复对象引用和输入一个非重复对象引用及相关的元数据到一个分级对象树的装置。

16.如权利要求14所述的计算机系统,其中用于构建的装置包括用于触发表明进度并且提供可选择的对象操作的事件的装置。

17.如权利要求16所述的计算机系统,其中可选择的对象操作包括过滤所述至少一个对象引用,其中所述对象引用能被删除。

18.如权利要求17所述的计算机系统,其中可选择的对象操作包括删除一个对象引用以及所有依赖于所删除的对象引用的后续指针。

19.如权利要求14所述的计算机系统,其中从属性列表包括所述线性列表,所述线性列表描述了为了满足从属性限定而生成的对象的顺序。

20.如权利要求14所述的计算机系统,其中所述派生步骤递归地遍历较低层对象引用的从属性,并且提供可选择的对象操作,其中较低层从属性能被删除。

21.如权利要求14所述的计算机系统,其中所述用于生成的装置包括用于应

用脚本选项以改变所述SQL脚本的装置。

22.如权利要求14所述的计算机系统，其中所述至少一个对象引用包括一个使用包括服务器/数据库/表格的格式的统一资源名称。

数据库对象脚本生成方法和系统

技术领域

本发明通常涉及计算机数据库领域并且更加特别的涉及在关系数据库中为复杂对象自动生成脚本的领域。

背景技术

结构化查询语言是美国国家标准学院(ANSI)用来与关系数据库进行通信的标准。SQL是关系数据库管理系统的标准语言。SQL语句被用来执行诸如从关系数据库中刷新数据或检索数据的任务。虽然很多数据库系统使用SQL,其中的很多数据库仍旧具有他们自己附加独有的扩展,所述扩展通常只能在其自己的系统中使用。但是,标准SQL命令,例如"Select","Insert","Update","Delete","Create","Drop"仍旧能够被用来实现获得人们希望从关系数据库中获得的业务。

关系数据库通常被构造成包含行和列的表。行和列交点或交点单元的数据的条目通常与一系列对数据的限制保持一致,所述限制被用于定义交点单元所允许的数据类型。与这种数据条目相伴随的一个长期的问题是对那些人们希望放入数据库的大且复杂的对象缺少定义。对已使用的SQL数据类型的限制通常限制了那些能够被输入到关系数据库例如SQL中的数据的类型。一个大的对象,如果放入SQL数据库,将会使数据库处理变慢或者使用例如存储器和处理器时间的系统资源以获得一个或多个庞大又复杂的对象。

脚本查询和在诸如SQL关系数据库中利用对象是既浪费时间又需要有较强的专业知识。这些脚本的作者为了在目标关系数据库中恰当地与正确的元数据一起用具体例子说明(instantiate)对象,必须知道与其他对象相关联的所述新且复杂对象的依赖关系。例如,这种脚本的专业知识可能会超出那些希望利用其关系数据库以追踪和查询与其企业(business)相关的复杂对象的一般用户的经验。另一方面,系统管理员可以利用脚本来帮助维护关系数据库。这些脚本动作需要时间和管理去准备和运行。目前,为涉及复杂对象的关系数据库自动生成脚本,并不是很容易实现的。

于是,对那些希望被引用并对SQL数据库有所动作的对象需要有一个统一的标记(reference)。另外,对应用和系统维护的工作来说,需要一种生成脚本的机制以方便在诸如SQL关系数据库内配置对象。本发明提出了上述已经提到的需求并且利用多种系统、方法和技术解决了他们,所述多种系统、方法和技术不需要知道关系数据库元数据的复杂结构就能生成复杂的关联树和表并且修改他们以生成脚本。

发明内容

本发明包括关系数据库的一个脚本器(scripter),所述脚本器能够特别地应用于一个SQL数据库管理系统。多种独立的软件模块或多种功能的串连输入对象引用并输出脚本。本发明的一个示例性部分或模块从所传入的一个或多个对象引用(OBJECT REFERENCE)生成分级对象树。利用对象引用的复杂对象,可以使用统一资源名称表示。所述模块排除重复的对象标记并产生一个关联树。所述模块还能在树被建立及完成之后提供机会编辑所述树。

另外的示例性模块或部分从用户或之前的模块输入一个等级关联树,并且生成一个从属性列表(dependency list)。从属性列表是一个线性表,所述线性表表达了为满足相关限制而利用的对象的生成顺序。这个模块还能在树被建立及完成之后提供编辑从属性列表的机会。

另外的示例性模块或部分从从属性列表中生成脚本。所述从属性列表可以是用户生成的,或者他也可以是从前面的模块输入的。所述模块在从属性列表中例示(instantiate)对象,并且调用与对象相关的脚本方法。所述模块提供给用户或控制程序广泛的灵活性以当脚本正被产生及完成之后编辑脚本。

附图说明

前面的发明概述,以及后面对较佳实施例的详细描述,当与附图相结合阅读时将被更好的理解。为了阐述本发明的目的,在图中示出了本发明的示例性结构;但是本发明不应限于所述的特定方法及其中所公开的手段。

图1是显示了本发明能够被实施的一个方面的示例性计算环境的方块图;

图2描述了本发明能够被实施的一个方面示例性、全面的流程图;

图3描述了与本发明第一模块或状态相适应的示例性流程图;

图4描述了与本发明第二模块或状态相适应的示例性流程图;

图5描述了与本发明第三模块或状态相适应的示例性流程图;

图6描述了本发明一个实现的示例性静态结构图。

具体实施方式

概述

本发明提供了在关系数据库中为多种复杂对象自动生成脚本。提供了一些技术，其中从属性列表被输入到处理中并且反应了多种对象间关系的从属性列表或分级树能够被产生。从属性列表或分级树可以然后被输入到处理的其他部分，其中分级树被处理成一个有序的从属性列表。从属性列表之后可以被处理成一个能够用来配置在一个特定目标关系数据库中的关系数据库对象的脚本。本发明的多种状态和模块可以被分别执行或以串连顺序执行。

示范性计算装置

图1和下述讨论，将用来提供一个对本发明所实施的合适的计算环境的简要的总的描述。然而，应该明白，可以预料到各种计算对象和手持式、便携式和其他计算装置，能够结合本发明一起使用。因此，虽然在下面描述了一种通用计算机，但其并不限于一个例子，并且本发明还可以用其他计算装置来实施，诸如具有网络/总线互操作性和交互性的客户端。因此，本发明可以在一种网络主机服务的环境中实施，其中牵涉到很少或最小的客户端资源，例如，一种客户装置仅仅起到网络/总线接口的作用的网络环境，诸如一个置于设备中的对象，或者其他计算装置和对象。其实，根据本发明，无论数据存于何处或者对哪个数据进行检索，都是合意的、或合适的操作环境。

虽然不是必需的，本发明可以通过一个操作系统来实施，以供装置或对象的服务的开发者使用，和/或可以包括在根据本发明操作的应用软件中。在计算机可执行指令的通常情况下，软件可被描述成，诸如程序模块，其可由一个或多个诸如客户端工作站、服务器或其他装置这样的计算机执行。通常，程序模块包括例行程序、程序、对象、组成部分（component），数据结构等，执行特定任务或实施特定的抽象数据类型。典型地，可以如在各种实施例中所描述的那样，合并或分散程序模块的功能。而且，本领域中的技术人员将会意识到，本发明可以用其他计算机配置来实施。适于本发明使用的其他公知的计算系统、环境、和/或配置包括，但不限于，个人计算机（PC）、自动柜员机、服务器计算机、手持式或膝上型装置、多处理器系统、基于微处理器的系统、可编程的客户设备、网络PC、仪表、照明器材、环境控制元件、小型计算机、大型计算

机等等。本发明还可以在分布式计算环境中实施，其中所述任务由远程处理装置来执行，该远程处理装置经由通信网络/总线或其他传输介质而被连接。在分布式计算环境中，程序模块可以存放于包括存储器存储装置的本地和远程计算机存储介质上、和可以轮流起服务器节点作用的客户端节点。

图1，如此示出了一个能实施本发明的适于计算系统环境100的例子，虽然如上所阐明的，计算系统环境100只是适于计算系统的一个例子，并不是想要暗示关于本发明的功能或使用的范围。也不应把计算环境100解释成对关于任一示例性操作环境100中所示出的部件或其组合的具备依赖性和相关要求。

参照图1，用于实施本发明的一个示例性系统，包括一个作为计算机系统110的通用计算装置。计算机系统110的部件包括，但是不限于，一个处理器单元120、一个系统存储器130、和一个各种系统部件的系统总线121，其中该总线包括把系统存储器连至处理器单元120。所述系统总线121可以是下述几种总线结构中的一种，包括利用各种总线体系结构中的任一种的局部总线和存储器总线或存储器控制器、外围总线。作为例子，而不限于，这种体系结构包括工业标准体系结构（ISA）总线、微通道体系结构（MCA）总线、增强型ISA（EISA）总线、视频电子标准协会（VESA）局部总线、和周边元件扩展接口（PCI）总线（还通称为Mezzanine总线）。

计算机系统110通常包括各种计算机可读介质。计算可读介质可以是任一可用介质，其能够被计算机系统110访问并包括易失性和非易失性介质、可移动和不可移动介质。作为例子，而不限于，计算机可读介质可以由计算机存储介质和通信介质组成。计算机存储介质包括，以任一方法或技术为诸如计算机可读指令、数据结构、程序模块或其他数据这样的信息存储所提供的，易失性和非易失性、可移动和不可移动的介质。计算机存储介质包括，而不限于，随机存储器（RAM）、只读存储器（ROM）、电可擦可编程只读存储器（EEPROM）、闪速存储器或其他存储器技术、只读光盘（CDROM）、可重写光盘（CDRW）、数字通用光盘（DVD）、或其他光盘存储介质、磁带盒（magnetic cassette）、磁带、磁盘介质或其他磁性存储装置、或任一其他能被用来存储想要的信息和能被计算机系统110访问的介质。通信介质，通常包括计算机可读指令、数据结构、程序模块、或其他诸如载波这样的已调数据信号的数据，或其他传输装置，并包括任一信息传递介质。所述术语“已调数据信号”，指一种这样的信号，即，

其具有一个或多个已按照该信号中的编码信息的这种方式而被改变或设定的特征。作为例子，而不限于，通信介质包括诸如有线网络或直接的有线连接这样的有线介质，和诸如声音、RF（射频）、红外和其他无线介质这样的无线介质。任一上述的组合还应被包括在计算机可读介质的范围中。

系统存储器130包括作为易失性和/或非易失性存储器的计算机存储介质，诸如只读存储器（ROM）131和随机存储器（RAM）132。基本输入/输出系统133（BIOS），通常存于ROM131中，其含有有助于在计算机系统110内的元件之间传输信息的基本例程序，诸如在启动期间。RAM132通常含有被直接访问和/或马上通过处理单元120而其作用的数据和/或程序模块。作为例子，而不限于，图1示出了操作系统134、应用程序135、其他程序模块136、和程序数据137。

计算机系统110还可以包括其他可移动/不可移动的、易失性/非易失性计算机存储器介质。只为了举例，图1示出了对不可移动、非易失性磁介质进行读或写的硬盘驱动器141，对可移动、非易失性磁盘152进行读或写的磁盘驱动器151，对诸如CD ROM、CDRW、DVD或其他光介质这样的可移动、非易失性光盘156进行读或写的光盘驱动器155。其他能用于示例性操作环境的可移动/不可移动、易失性/非易失性计算机存储器介质包括，而不限于，盒式磁带、闪存存储卡、数字通用光盘、数字视频磁带、固态RAM、固态ROM、等等。所述硬盘驱动器141通常被经由诸如接口140这样的不可移动的存储器接口连至系统总线121，以及磁盘驱动器151和光盘驱动器155通常由诸如接口150这样的可移动的存储器接口连至系统总线121。

所述驱动器和其在上面讨论过并在图1中示出的相连的计算机存储介质，为计算机系统110提供了机器可读指令、数据结构、程序模块和其他数据的存储。在图1中，例如，硬盘驱动器141被示出为存储有操作系统144、应用程序145、其他程序模块146、和程序数据147。注意到，这些部件能够和操作系统134、应用程序135、其他程序模块136、以及程序数据137相同或不相同。此处，给出了不同标号的操作系统144、应用程序145、其他程序模块146、以及程序数据147，以最小限度地说明他们是不同的副本。用户可以通过诸如键盘162和指示装置161这样的输入装置，把命令和信息输入到计算机系统110，其中所述指示装置161通常被称为鼠标、轨迹球、或触摸板。其他输入装置（未示出）可以包括一个麦克风、操纵杆、游戏垫、圆盘式卫星电视天线（satellite dish）、扫描仪、

或类似的装置。这些和其他输入装置，通常经由连至系统总线121的用户输入接口160而被连至处理单元120，但是也可被诸如并行端口、游戏端口或通用串行总线（USB）这样的其他接口和总线结构连接。监视器191或其他类型的显示装置，也被经由一诸如视频接口190这样的接口连至系统总线121上，所述视频接口190可以反过来与视频存储器（未示出）进行通信。除了监视器191之外，计算机系统还可以包括其他外部输出装置，诸如能够通过一个输出外设接口195而被连接的扬声器197和打印机196。

计算机系统110，可以利用到一个或多个诸如远程计算机180这样的远程计算机上的逻辑连接，在网络或分布式环境中运行。所述远程计算机180可以是个人计算机、服务器、路由器、网络PC、等同装置或其他公用网络节点，并典型地包括很多或全部在上面关于计算机系统110而描述的元件，虽然在图1中仅仅示出了存储器存储装置181。在图1中所描绘的逻辑连接，包括局域网（LAN）171和广域网（WAN）173，但是还可以包括其他网络/总线。这种网络环境在家里、办公室、企业范围的计算机网络、内部网和因特网内是常见的。

一旦在LAN网络环境中被采用，计算机系统110就被通过一网络接口或适配器170连至LAN171。一旦在WAN网络环境中被采用，计算机系统110就会典型地包括调制解调器172或其他用于在诸如因特网这样的WAN173上建立通信的装置。所述调制解调器172，可以是外部的也可以是内部的，其经由用户输入接口160或其他合适的装置而被连至系统总线121上。在网络环境中，在关于计算机系统110所描述的程序模块、或其部分，可以存于远程存储器存储装置内。作为例子，而不限于，图1示出了作为驻留在存储器装置181中的远程应用程序185。应该意识到，所示出的网络连接，是示例性的，还可以采用其他在计算机之间建立通信连接的装置。

已经并且正在根据个人计算和因特网的集中，研发各种分布式计算结构。对个人和相似的商业用户，提供了用于应用程序和计算装置的无缝地能共同操作的和Web启用接口，这使面向Web浏览器或网络的计算活动增加了。

例如，MICROSOFT®的.NET™平台，可从微软公司购买（地址：华盛顿州雷蒙德市—微软路（One Microsoft Way, Redmond, Washington）邮编98052），包括服务器，模块构建服务（building—block service），诸如基于网页的数据存储，和可下载装置软件。虽然在此结合了驻留在计算装置上的软件对示例性实施例进

行了描述时，但是本发明的一个或多个部分，还可以通过操作系统、应用程序接口（API）或在协处理器、显示装置和请求对象之间的“中间人”对象来实施，以便根据本发明的操作可以被其执行，由所有的.NET™语言和服务支持或访问，并且也在其他分布式计算结构中。

本发明的示范性实施例

关系数据库管理对象脚本器为对象生成脚本。在一个实施例中，对象在SQL数据库中被实现。本发明可以按照其一般适用性被描述也可以根据所述SQL环境的实施例被描述。所述脚本的一个例子是Transact-SQL脚本。脚本编写允许系统管理作业的自动操作包括SQL数据库中的对象。脚本器对象（scripter object）是存在于所述SQL管理对象的名字空间中的对象，但是没有依赖于使用脚本器的SQL管理对象。脚本器可以利用两个操作：建立（creat）和移除（drop）操作，所述操作由每个实例类执行。因此，为了给建立或删除SQL数据库中的所述对象生成脚本文本，生成实例对象仅需对其自身的实例脚本编写负责。作为结果，脚本编辑器对剩余的功能性操作负责，所述功能性操作包括寻找相关性，输出脚本到存储器，文件或显示器，调用所述实例对象以对上述对象编写脚本，和控制编写脚本操作的环境及进程。

一个脚本器对象模型可以部分的用于本发明中。脚本器对象模型是编写脚本的一个进入点。所述对象模块持有编写脚本操作的语境。所述对象模型可以利用统一资源名称被唯一的识别出来。例如，一个拥有数据库名称“pub”的SQL管理对象可以被同一资源名称唯一地指示，其中所述数据库包含有被“dbo”拥有的表“作者（authors）”：

Server/Database[@Name='pubs']/table[@Name='authors' and schema='dbo']

对统一资源名称的利用提供了一种唯一识别对象的标准格式并且与其他利用可允许的灵活度的标准相一致并且在本发明中的应用中更进一步扩充。

脚本器在大量独特阶段进行操作（phase），所述阶段允许对底层数据结构进行中间层的操作。这些阶段（phase）可以整体地执行，允许最小限度的操作；或可以被分别执行以给修订底层数据结构最大的自由。脚本器执行的示例性阶段发现其从属性（dependency），生成从属性列表，并从从属性列表中生成脚本。

图2是一个结构图，显示了本发明示例性阶段之间的关系以及这些阶段数据的流动和用户或其他控制程序。用户程序260能够与本发明的示例性且独特的阶

段互相影响:发现,生成表,并且脚本化生成。在每一个阶段用户程序提供对特定操作的输入。在每一个阶段返回一个结果,所属结果能够被操作并被输入到处理的下一个阶段。在每一个阶段事件被发送到用户过程,如果其对这些事件赞成,就提供此处理中过程的相关信息。这个事件机制,通过允许用户在这些事件上对处理作出反应并与处理相互影响,来允许用户改变脚本器对每一个对象的行为。这个脚本选项,通过脚本生成的每个阶段,为脚本输出提供完全的控制。

希望每一个阶段都是断开的,并且每个阶段的输入不必要求由脚本器生成。用户程序根据其自身的内部算法能够生成其自身的数据结构,并且将这些数据结构提供给脚本器作为输入。于是,脚本器阶段能够被看作有三个独立的部分。

返回到图2,发现阶段从用户程序260获得对象引用202来对关联发现机制210进行初始化。对象可以使用统一资源名称被标记。关联数据和实例数据(元数据)204从系统中被调回以帮助所述发现处理。特别地,关联数据可以通过在服务器和系统目录(system catalog)270中的关联检索被定位。应该注意,关联的粒度(granularity)也决定了从属性发现操作(dependency discovery operation)的粒度。一旦使用关联查找和发现或侦测算法发现两个或多个对象间的关连,对象关系数据208就进入一个相关树212。所述处理返回214到发现机制210来继续为相关树212生成条目。在所述处理的最后,一个相关树212被生成,并且对用户程序260是可用的。

关联查找及侦测技术将以其操作的形式被描述。另外,该算法描述为生成父对象图表,生成原始对象图表,并随后生成子对象图表。所述生成图表实质上就是相关树的结构。最初,关联发现机制210看到对象引用并且将对象作为枚举对象调回。在此处并不需要真正的实例,因此对象本身并没有被调回;只有对象的标记被激活。所述对象被加入已知的对象表。关联发现机制为加入的对象寻找每一个不在已知对象表中的父亲。关联发现机制之后可以调回父对象并且将其加入已知的对象表。当最高行的父亲被侦测到时,最高行的父亲就被加入到相关树中,所述相关树是分级树。关联发现机制随后在已知的对象表中向下寻找所有的子对象。上述过程向相关树中添加原始对象,同时添加已找到的父亲的儿子,原始对象,以及儿子和孙子对象。此递归算法生成正确的相关顺序。

图2中的阶段2包括了从属性列表的生成。从属性树可以从阶段1 (216) 得到或其可以通过面向用户程序的接口218获得。无论怎样, 从属性树218被接收并且表生成机制220生成对象表230的任务被激活, 该对象列表也被视为从属性列表230。表生成机制220为从属性树212和实例数据 (元数据) 222提供接口224, 226以生成一个对象表230的条目228。一旦对象和其相关数据被侦测到, 就可产生对象表条目。所述处理返回以从从属性树读出附加项目 (item) 并且不断生成对象表直到从属性树元素都被遍历完。当诸如对象表条目之类的事件被发现, 事件229就被生成为用户程序以编辑对象表的方式来操作所述处理提供一个机会。在所述处理的最后, 对象表或从属性列表230被生成并且使232对用户程序260来说是可以得到的, 同时可以被用户程序进行全面编辑。

图2的第3阶段通过脚本生成机制240生成脚本 (例如: Transact-SQL脚本), 所述脚本生成机制利用对象表和脚本选项234。对象表可能与早前的阶段232生成的对象表相同, 或者可以是通过用户程序260的接口234接收到的。无论怎样, 脚本生成机制240获得被用户程序选择或默认的脚本选项234以及实例数据 (元数据) 236。当对象表230被处理238, 242时, 脚本生成机制240可以生成脚本条目244。当脚本250被生成时, 诸如新条目和编辑已生成脚本的事件252就能够被呈现给用户程序260。最后, 对象表230被完全处理并且生成了完全脚本250。为了后续处理, 所述处理随后能够发送脚本254到用户程序或发送给用户或一个等价的接口。

如图2所示, 对脚本的全部处理可看作是以获得对象引用202开始并且以生成脚本254结束的一个过程。另外, 所述处理也可以被看作是能够被串连的多个独立的处理。图3、4、和5分别描述了阶段1、2和3的单独处理。

图3是本发明关联发现阶段的方法300的一个示例性流程图。发现关联阶段从通过所述处理的单个或多个对象引用中生成一个分级对象树 (图graph)。所述处理以获得一个或多个与对象相关的对象引用310开始, 所述对象是希望在诸如SQL数据库的关连数据库中被用到的。所述处理随后转移为为侦测相关于对象引用的相关性而进行的查询。关连数据被定义成涉及对象间相关性的信息。例如, 对SQL的观察 (view) 依赖于所存在的从属性列表。反过来说, 从属性列表对于SQL观察具备依赖。作为附加, 描述对象实例的实例数据, 例如, 一个实例或元数据表, 可以作为320中揭示的关系相关数据的一部分。

当侦测到相关性时, 在330生成一个相关或分级树, 代表了相关数据和关联

的对象。由于对复杂等级编写脚本时引用（reference）同一个对象是可能的，因此，用来生成分级树或图的算法避免了重复 340。这些被一次或多次参考的对象可以生成脚本。相关树或分级树可以包括父亲、第一儿子、下一个兄弟、孙子、和曾孙类型的相关结构，例如。当相关性被侦测时，所述处理检测相关树是否完成。如果没有，处理可以允许编辑对象360。如果不需要编辑360，所述处理通过转向下一个对象引用390而继续。如果需要编辑360，就能够编辑树370。例如，对象的删除不仅仅能导致删除一个对象，还能导致删除所有在其关系分支上的后继的子对象。这允许用户在对象被加入到最终的图中之前，处理或过滤对象。当完成编辑之后，所述处理转向下一个对象引用380，并且继续查找相关数据320。

如果既没有控制过程也没有用户需要对象编辑，关联发现300继续下一个对象引用并且寻找相关数据320摘要（resume）。最后，从属性树或分级树被完成350并且从属性树作为一个输出399对后续过程来说是可得到的或作为用户输出。

图4是本发明的示例性相关发现方法400的步进阶段的流程图。所述从属性步进阶段从已进入所述处理的从属性树生成从属性的线性表格。所述处理由接收一个从属性树410所谓处理的输入而开始。这一输入可以从诸如图3中的输出399获得，或者可以由用户或一个处理程序分别输入。

返回图4，从属性发现步进方法400继续对从从属性树中导出的关系从属性进行侦测。当从属性被侦测到，所述处理生成一个从属性列表条目430。从属性列表较佳地是一个线性表，所述线性表列出了为了满足从属性限定而生成的对象的顺序。例如，一个用户定义的数据类型必须作为对象，可能是元数据，的一部分存在，这优先于依赖所述对象或用户数据类型。

假设所述处理第一次通过440时并没有完成，所述处理400允许进行编辑来生成一个表。如果所述表将被编辑450，那么所述编辑就能够在460被允许，其中从属性列表中被标记的对象将被移除或改变。当编辑完成后，所述处理继续至从属性树470的下一个对象而对关系从属性的侦测继续至420。如果并不需要一个编辑，450，那么就通知所述处理过程480并且程序对下一个树对象作索引并且继续侦测关系从属性420。

最后，反复进行所述处理直到所述树被完全遍历并且对从属性列表条目的生成过程完成440。从属性列表可以随后被获得或被一个后续过程或一个给用户的输出获

得499。

图5本发明示例性脚本阶段方法500的流程图。所述脚本阶段从输入到所述过程的从属性列表生成一个脚本，例如Transact-SQL脚本。所述处理以获得从属性列表作为处理的输入而开始。这一输入可能从一个诸如图4的输出中获得，也可以从分别由用户或一个处理过程输入。返回到图4，脚本阶段500根据从属性列表对象的标记实例520继续进行。接着，根据对象引用530作出一个关于脚本的对象引用调用。根据调用的结果返回的脚本，当所述处理反复进行的时候，所述脚本也进行积累（accumulate）。

假设所述处理没有完成550，处理500允许对脚本进行编辑560。如果需要编辑，用户或其他控制过程可以编辑脚本570并且然后继续进行从属性列表580的下一个对象，允许下一个对象引用520的下一个实例。如果不需要编辑560，一个处理通知被通知给控制处理或用户590并且处理500继续到下一个对象引用的实例。

最后，所述处理遍历所有在从属性列表中的对象并且所述处理完成550。之后，被积累的脚本对于后续过程是可以获得的或作为给用户的输出也是可以获得的。

应该注意，不偏离本发明意图，图3、4和5中的处理是可以修改成包括或不包括在任何一点的通知处理或在处理的任何一点进行编辑。例如，在图5中，不偏离本发明，一个脚本过程控制点可以被放在脚本积累器（script accumulator）写之后，或者当完成了检查550之后，或者在编辑570之后，或者在可获得脚本595之后。这样不超过本发明精神的可选择控制点的改变与图3、4、5中的流程图的应用是相等的。

当前发明的一个实施例以图6中的一个统一模型化语言（uniform modeling language（UML））表示的结构来说明。UML图表600用示意图描述了在对象脚本编写器模式中的每一个类。这个实施例假设其被应用在一个SQL数据库中。从属性遍历器（dependency walker）610提供了能够发现在SQL服务器数据库中的类之间的相关性及关系的功能。对于脚本器类这是一个基本的类。相关遍历器610使用过滤器代表（filter delegate）612和一个进展报告代表（progress report delegate）614。过滤器代表612是一个事件，所述事件允许对在发现阶段找到的类进行改变和/拒绝。进展报告代表614是一个事件，所述事件提供有关相关性和/或将本编写阶段过程的处理信息。

脚本器620是对终端用户来说是揭示脚本编写功能的主要的脚本编写类。脚

本器620使用过程报告代表614和错误事件代表(error event delegate) 624。脚本选择622是一个允许改变脚本器620的动作用的类。脚本选择622作为脚本器620的类被揭示。但是,脚本选择类622能够独立的作为实例,这允许这一类作为一个独立类的脚本编写方法的一个变量通过。

相关节点类630包括URN作为对象引用。相关节点630类对所有从属性树和表类来说是一个基础类。相关树节点类632是一个含有父亲和儿子节点信息的类。其扩大了相关节点类630。相关树类634包括脚本器关系。这一类可以扩充相关树节点类632。所述相关树类634作为脚本器620性质被揭示。

数组表640是.NET 结构的一个标准类,所述表能够在线性表的0—n类中保存标记。数组表类是作为从属性列表类642基础类的。相关标类在线性表中642保存有从属性列表节点644的类。他被用于从属性列表642并且作为脚本器类620的一个属性被揭示。从属性列表节点633包括有对象是否是原始列表(例如,根对象)的信息。它扩展了相关节点类630。

根据本发明的一个实施例,脚本器可以有在操作和应用上的灵活性。图2的处理也可以使用一个过滤器来完成。在第2和第3阶段可以调用所述过滤器,在这些阶段允许对象被转移或改变。所述脚本生成器可以执行一个能够被利用排除对象(包括儿子或更低层次的对象)的性质。此外,在脚本阶段被更改的对象也可以包括这些更改而被编写脚本。

如果在对象中可以的话,通过更改名称属性可以改变对象的名字。所述对象并不必须维持这一目的。如果需要,所述图通过设置图的属性就能够被更改,如果在对象中可以的话。可以设置一个脚本错误事件来接收在脚本生成阶段发生的错误。通过允许一个当错误发生时继续的选项来期待当错误发生时(例如,当一个对象在服务器中不能被找到)继续。

系统对象可以被编写脚本以允许用户基于系统对象的结构生成新的(非系统的)对象。下列的脚本规则可以适用于系统对象。

- (1) 用户应该被允许传递一个系统对象到脚本器并且生成一个脚本。
- (2) 由于服务器可能不允许生成系统对象,因此没有事先修改的脚本是不能够在服务器上执行的;并且
- (3) 如果用户传递了多个对象并且其中一个对象是系统对象,所属系统对象是被允许的。如果在发现阶段侦测到一个系统对象,可以选择一个选项来停

止脚本编写。当对一个完整的数据库或者包括相关性的大量的对象编写脚本时是很重要的。

每一次，都利用不同的脚本操作对一个或多个对象编写脚本。可选择的，一个过滤功能能够被传递以利用其来过滤统一资源名称，所述统一资源名称被用于唯一的识别对象。在发现阶段（例如图2中阶段1）当一个对象被加入相关树时，所述过滤器被调用。对于用户化目的来说这是很有用的。一个过滤对象和所有其关联都不会被编写脚本。

在另一个实施例，提供脚本选项作为本发明的一部分。这些选项允许用户或控制程序，例如一个执行程序或一个应用程序，来对一个脚本过程行使控制。下面说明了一些选项：

脚本输出格式选项

脚本器选项	描述
-------	----

Append To File	附加在标明的输出文件上。缺省的，脚本方法重写已知文件。
----------------	-----------------------------

ANSI To File	利用多字节特征生成脚本文件
--------------	---------------

Drops	生成Transact-SQL来移走已经标记的内容。用于在移除内容之前是否存在进行脚本测试
-------	--

Encrypt PWD	与脚本一起编码口令
-------------	-----------

Include Headers	生成的脚本拥有包括生成日期和时间及其他描述信息的头
-----------------	---------------------------

Include If Not Exists	Transact-SQL生成的内容包括检查存在的前缀。当脚本被执行时，仅仅当已命名内容的拷贝不存在时生成一个内容。
-----------------------	---

No Command term	利用特定连接命令终止符（connection-specific command terminator）在脚本中独立的Transact-SQL语句不被划界。缺省的，独立的Transact-SQL语句不被划界。
-----------------	---

Schema Qualify/ Transact-SQL中生成的能够移走一个对象的对象名称被标记对象的拥有者限制。生成的为了产生标记对象的Transact-SQL利用当前对象拥有者限定对象名称。

Schema Qualify Foreign Keys	为外来字符(key)约束的图限定表（schema qualify table）标记
-----------------------------	---

Timestamp To Binary	当对象脚本编写生成一个表或用户定义的数据类型时，将时间戳数据类型的说明转换为二进制。
---------------------	--

To File Only 多数SQL对象脚本编写方法指定一个返回值和一个可选输出文件。当使用时，并且一个输出文件被指定时，所述方法并不将脚本返回给调用者，而是仅仅将脚本写入输出文件。由于编写出相关性脚本潜在性的会产生出很大数量的字符串，因此只要不需要字符串输出，就指定这一选项。

Unicode File 缺省的生成unicode输出

Login SID 包括登陆脚本的安全识别

DDL Header Only 仅为对象的DDL头编写脚本，所述对象具有诸如存储式程序（Stored Procedure）的脚本文本。缺省是脚本化输出全部DDL。

DDL Body Only 仅为对象的DDL脚本编写脚本，述对象具有诸如存储式程序（Stored Procedure）的脚本文本。缺省是脚本化输出全部DDL。

脚本相关性选项

脚本器选项	描述
With Dependencies	对输出的脚本表进行扩展，包括所有相关的对象
Database	生成Transact-SQL数据库优先定义脚本。数据库允许或不允许语句的执行权力
Permissions	利用一个OR逻辑运算符使聚类索引，非聚类索引和DRI索引结合起来。既适用于表格也适用于查看对象
Scrip Indexes	利用一个OR逻辑运算符使SQL SMO脚本对象允许和SQL SMO脚本数据库允许结合起来。
Primary Object	生成能够产生相关内容的Transact-SQL
Extended Property	包括扩展作为对象脚本一部分的脚本属性
XML Namespaces	包括作为对象脚本一部分的XML名空间
Full Text Vatalogs	批处理命令，包括能够生成查找全文目录的Transact-SQL语句

交叉服务器层脚本生成

脚本器选项	描述
No Collation	缺省为生成校对。说明表对象。
Full Text Index	包括定义查询全文索引的语句的批处理命令
Bindings	生成sp-缺省聚合 (sp-bindefault) 和sp-聚合规则 (sp-bindrule) 语句。仅当脚本编写涉及一个SQL服务器表时适用。
Clustered Indexes	生成定义聚合索引的Transact-SQL。仅当脚本编写涉及一个SQL服务器表时适用。
DRI-All	使用逻辑运算符OR定义所有值为DRI组合
DRI-All Constraints	使用OR逻辑运算符结合DRI一检查, DRI一缺省, DRI外来密钥, DRI初级密钥, DRI单件密钥, DRI XML密钥
DRI-All Keys	使用OR逻辑运算符结合DRI外来密钥, DRI初级密钥, DRI单件密钥, DRI XML密钥
XML Indexes	生成的脚本产生XML索引
DRI-Checks	生成的脚本产生指定行的检查限制。当宣布的引用完整性建立了关系从属性时指向脚本。仅当脚本涉及一个SQL服务器表时应用。
DRI-Clustered	生成的脚本产生聚合索引。当宣布的引用完整性建立了关系从属性时指向脚本。仅当脚本涉及一个SQL服务器表时应用。
DRI-Defaults	生成的脚本包括指定行的缺省。当宣布的引用完整性建立了关系从属性时指向脚本。仅当脚本涉及一个SQL服务器表时应用。
DRI-Foreign keys	生成的脚本产生外来密钥限制。当宣布的引用完整性建立了关系从属性时指向脚本。仅当脚本涉及一个SQL服务器表时应用。
DRI-Non Clustered	生成的脚本产生非聚合索引。当宣布的引用完整性建立了关系从属性时指向脚本。仅当脚本涉及一个SQL服务器表时应用。
DRI-Primary Key	生成的脚本产生原始密钥限定。当宣布的引用完整性建立了关系从属性时指向脚本。仅当脚本涉及一个SQL服务器表时应用。
DRI-Unique Key	生成的脚本产生候选密钥, 所述密钥被单一索引定义。当宣布的引用完整性建立了关系从属性时指向脚本。仅当脚本涉及一个SQL服务器表时应用。
DRI Indexes	利用单一索引脚本化原始密钥限制以完成宣布的引用完整性。仅当脚本涉及一个SQL服务器表时应用。
DRI With No Check	当使用DRI检查, DRI外来密钥时生成的脚本包括WITH NO CHECK子句以最佳化限制的生成。仅当脚本涉及一个SQL服务器表时应用。
Non Identity	生成的Transact-SQL语句不包括对同一属性、种子和增量的定义。仅当脚本涉及一个SQL服务器表时应用。
Non Clustered Indexes	生成Transact-SQL定义非聚合索引。仅当脚本涉及一个SQL服务器表时应用。
Object Permissions	在脚本化数据库对象时, 包括Transact-SQL优先权定义语句
Triggers	生成Transact-SQL定义触发。仅当脚本涉及一个SQL服务器表时应用。
User Types To Base Type	将用户定义数据类型的说明转换为SQL服务器的基础数据类型。仅当脚本涉及一个SQL服务器表时使用
No File Groups	不包括'ON <filegroup>'子句的批处理命令, 所述子句指向文件组用法

其他

脚本选项	描述
Allow System Objects(Boolean)	允许脚本化系统对象。如果没有说明系统对象将被过滤出去。
Agent Alert Job	生成Transact-SQL脚本，产生SQL服务器代理服务作业和警告
Agent Notify	当脚本化一个警报时，生成脚本以产生所述警告的通知。
ANSI Padding	批处理命令包括Transact-SQL语句SET ANSI PADDING ON和SET ANSI PADDING OFF语句，在生成的脚本的CTEATE TABLE语句之前或之后
No What If Index	批处理命令不包括CREATE STATISTICS语句
NoTablePartitioningSchemes	批处理命令不包括表对象的部分图表
NoIndexPartitioningSchemes	批处理命令不包括索引对象的部分图表
No Assemblies	批处理命令不包括Assemblies
No View Columns	为了查看对象不脚本化指定列。查看列在查看产生时被记录，可以由特别指定上述内容指定也可以由选择语句定义
Include Database Context	在脚本头加入一个USE[database]语句。所述[database]包括被脚本化对象的数据库名称

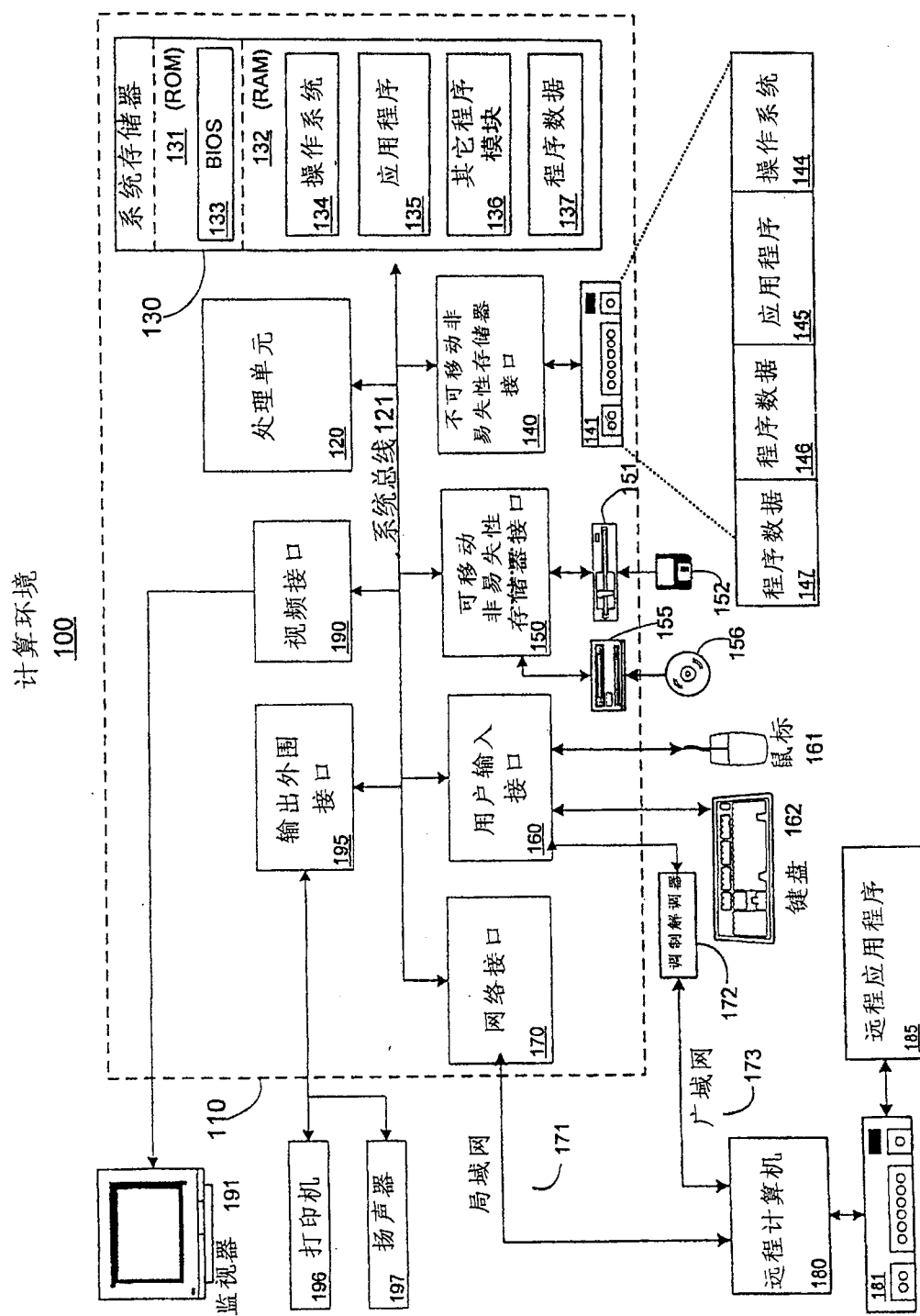
如上所提到的，尽管已经结合了各种计算装置和网络体系结构，描述了本发明的示例性具体实施例，但是所述基础概念可以被用于任何想要在其中实施自动任务生成器的计算装置或系统。因此，本发明的方法或系统，可以被用于各种应用和装置。尽管在此选出了示例性编程语言、名称和例子作为各种选择中的代表，但并不是想要以这些语言、名称和例子进行限制。本领域的技术人员如果能意识到，存在大量提供对象代码的方法，其可实现本发明所完成的同样、类似或等效的系统和方法。

可以结合硬件或软件或如果合适的话用两者的组合，来实施此处所描述的各种技术。因此，本发明的方法和装置、或其部分的某一方面，可以表现为包括在有形的介质中程序代码（即，指令）的形式，所述介质可以是诸如软盘、CD-ROM、硬盘、或任一其他机器可读存储介质，其中当所述程序代码被诸如计算机这样的机器载入并执行时，所述机器就变成了实施本发明的装置。若是在可编程计算机上的程序代码，所述计算装置将通常包括处理器、处理器可读的存储介质（包括易失和非易失性存储器和/或存储器元件）、至少一个输入装

置、和至少一个输出装置。可以利用处理本发明的服务程序(service)的信号的一个或多个程序,例如通过使用数据处理API或类似的,被优选地以高级程序性的或面向对象的编程语言来实施,以于计算机进行通信。然而,如果想要的话,所述程序(s)能够以汇编或机器语言来实施。无论如何,所述语言都可以被编译或解释成语言,并与硬件的实施相结合。

本发明的方法和装置,还可以通过通信来实施,以程序代码的形式被具体化的通信,通过某些传输介质而被传输,诸如通过电线或电缆,通过光纤,或通过任一其他形式的传输,其中,当所述程序代码被机器收到和载入并执行时,诸如EPROM、门阵列、可编程逻辑装置(PLD)、客户端计算机、录像机或类似的、或具有如在上面的示例性实施例作描述的信号处理能力的接收机成了实施本发明的装置。当在通用处理器上实施本发明时,所述程序代码就与处理器相结合以提供一唯一运行以调用本发明所描述的功能的装置。此外,结合本发明所用的存储技术,总是硬件和软件的结合。

虽然已经结合各种附图的优选实施例详细描述了本发明,但是应该明白其他类似具体实施例可以被使用,或可以为了执行本发明的同样的功能而对所描述的具体实施例进行修改和添加而不脱离本发明。而且,应该强调,各种计算机平台,包括手持式装置的操作系统和其他应用特定的操作系统都是预期中的,尤其是在无线连网装置的数目仍在增加时。因此,本发明不应被限制成单个具体实施例,而是应该根据所附的权利要求被广义地解释。



200

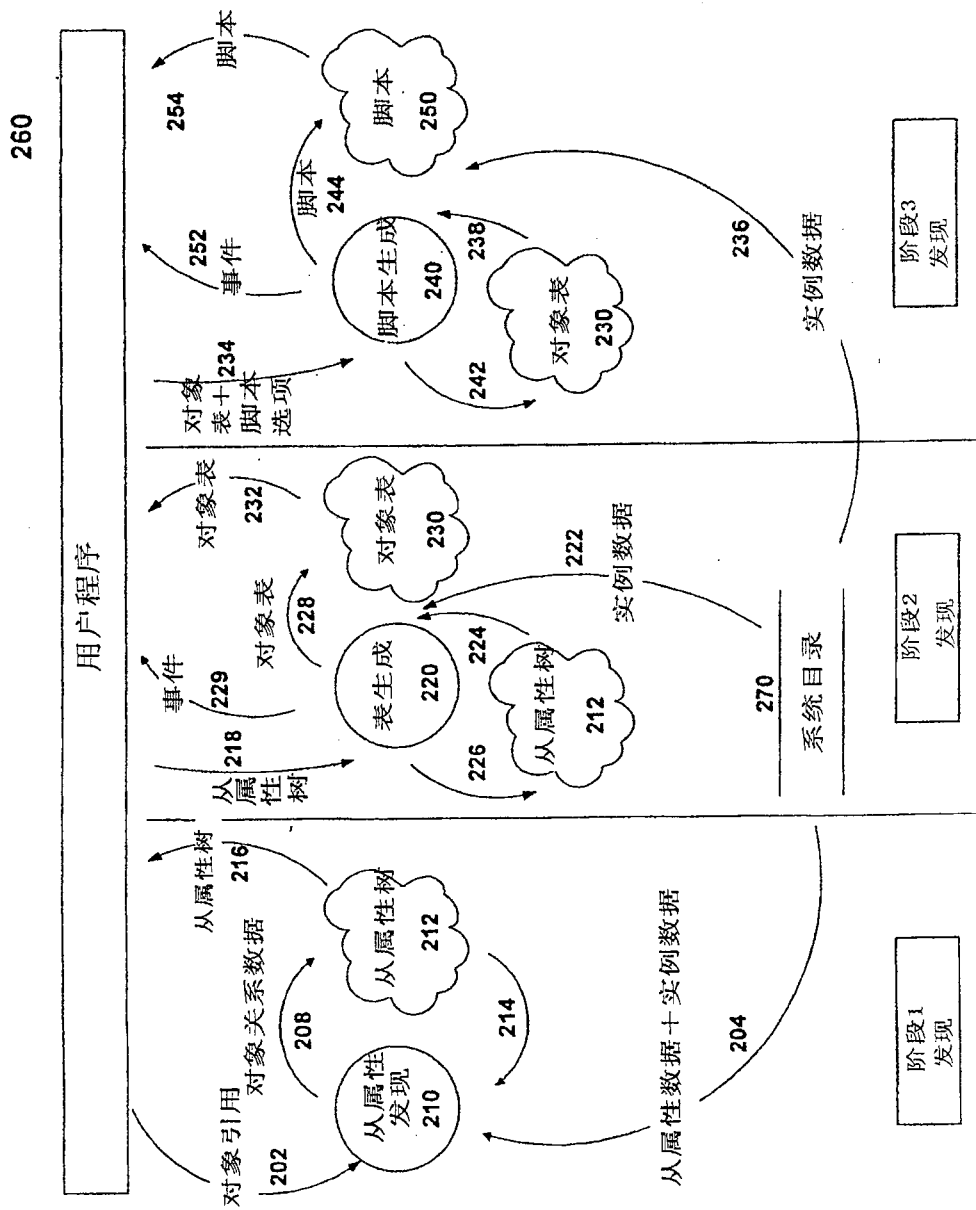


图 2

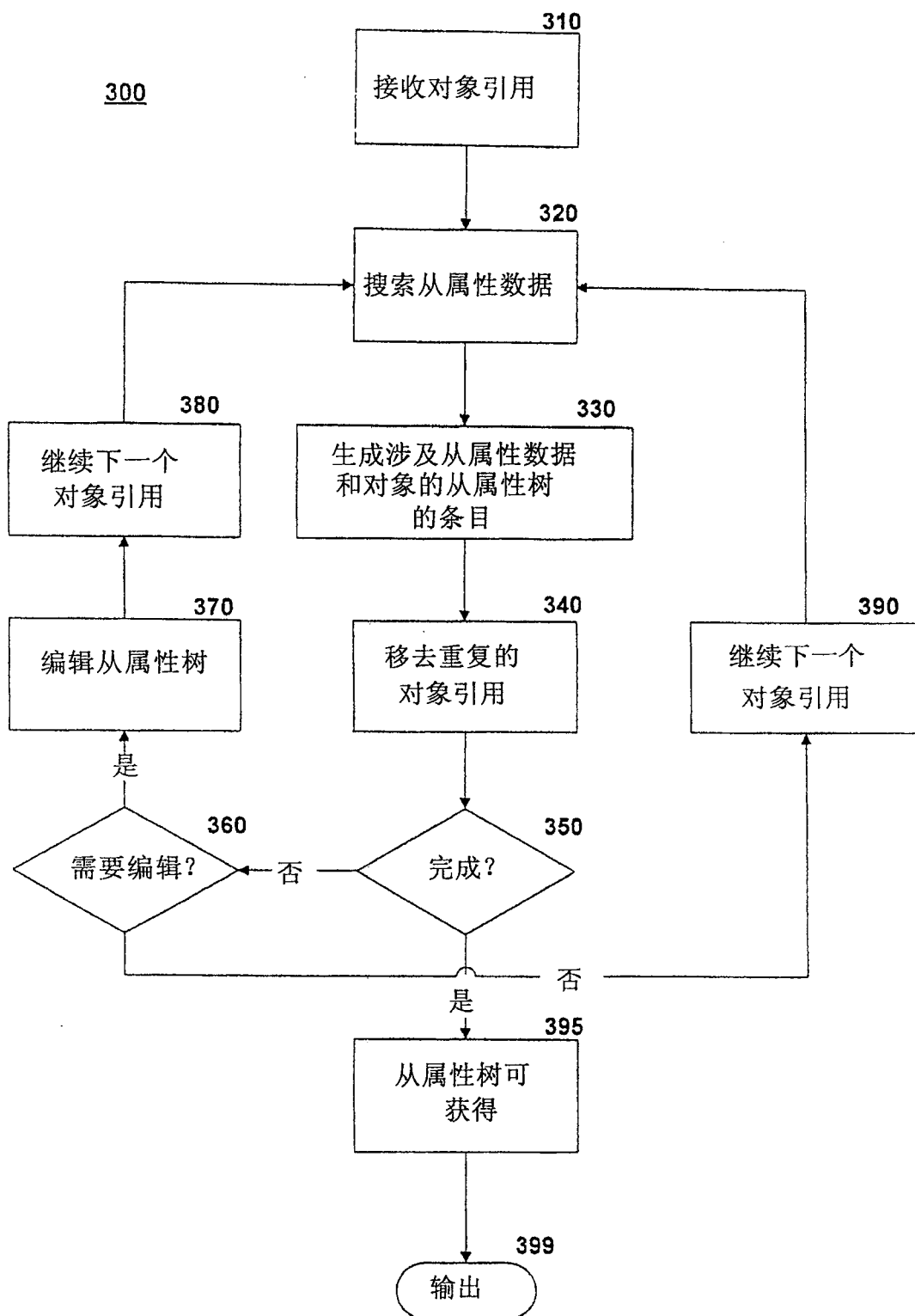


图 3

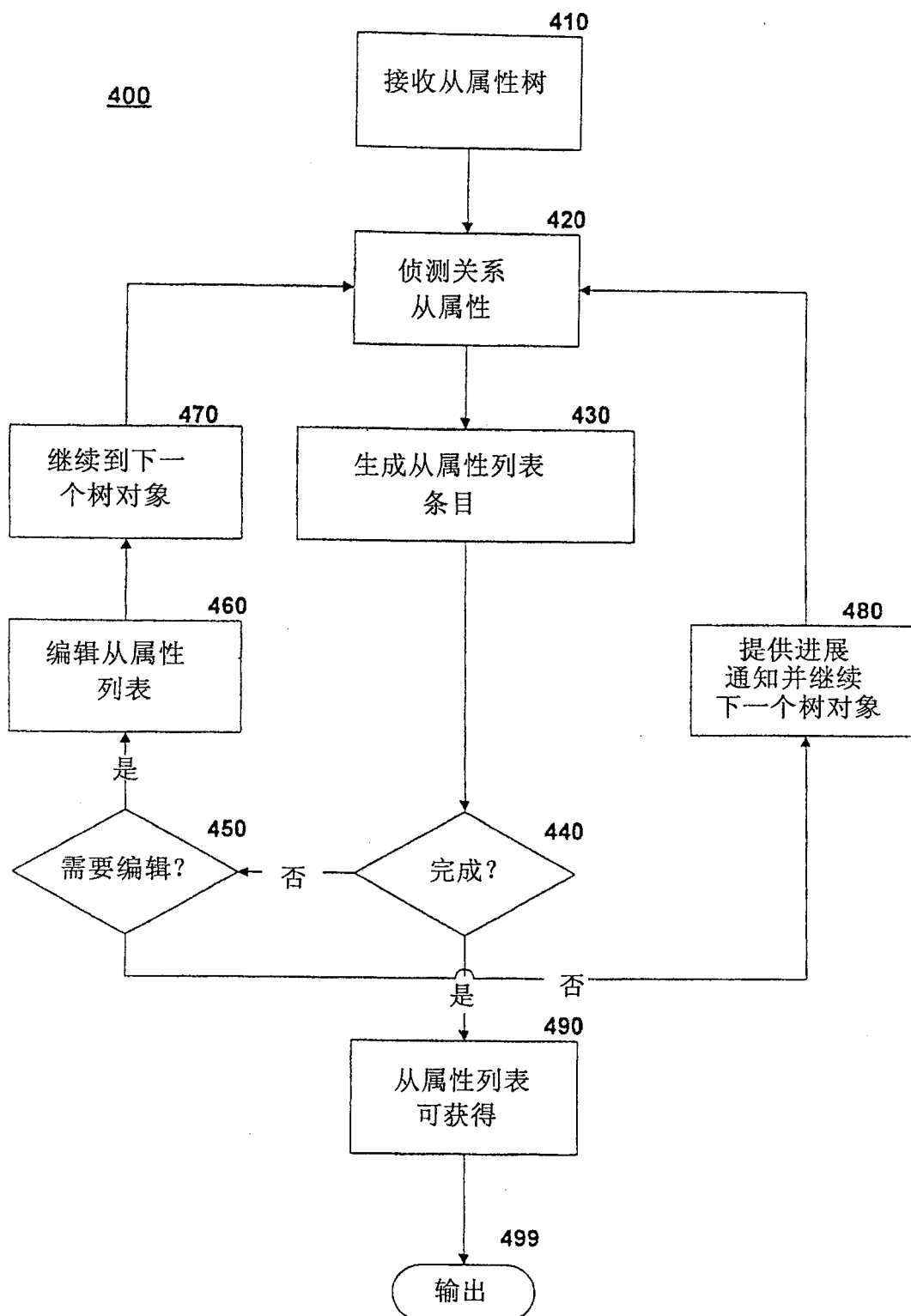


图 4

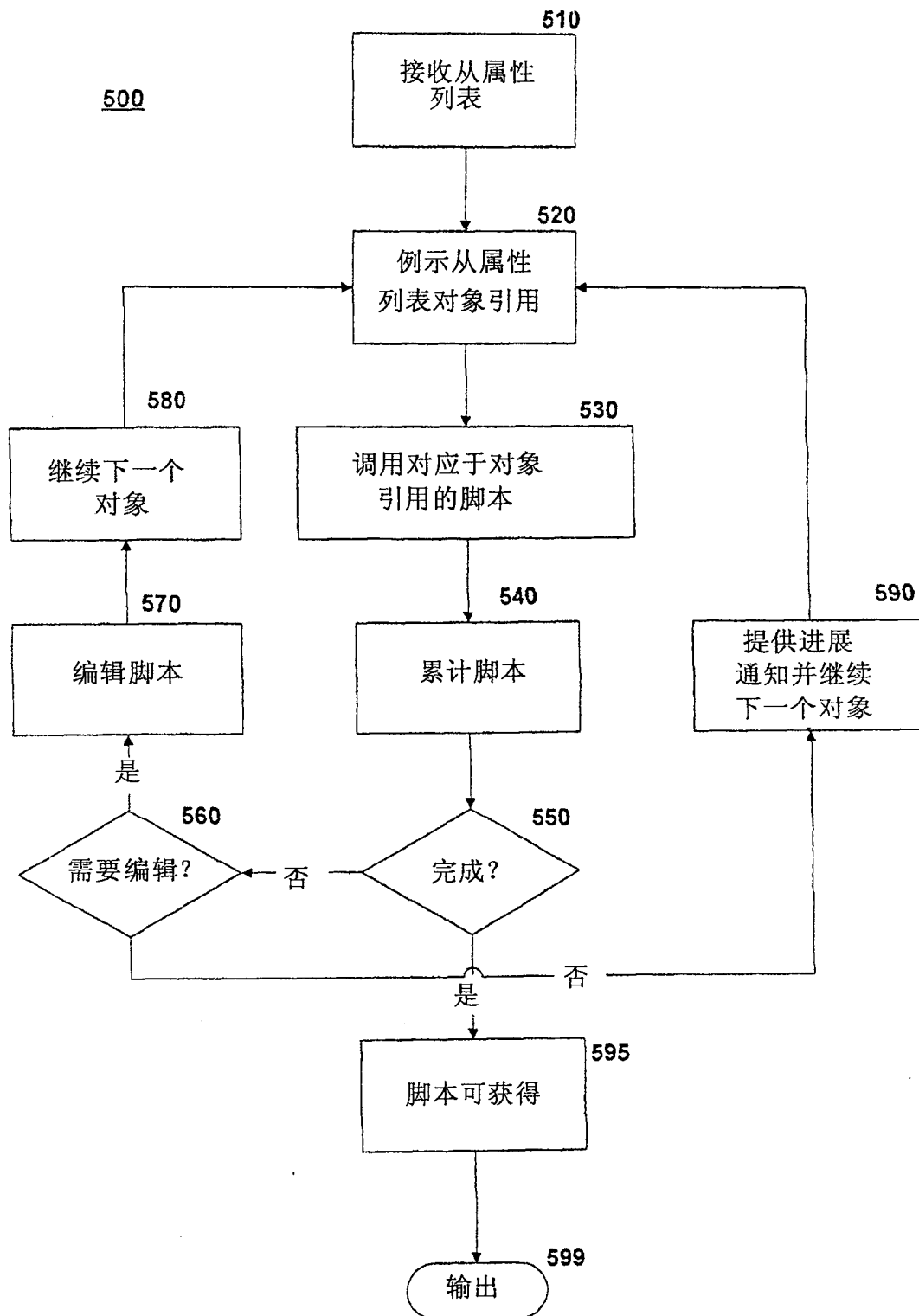


图 5

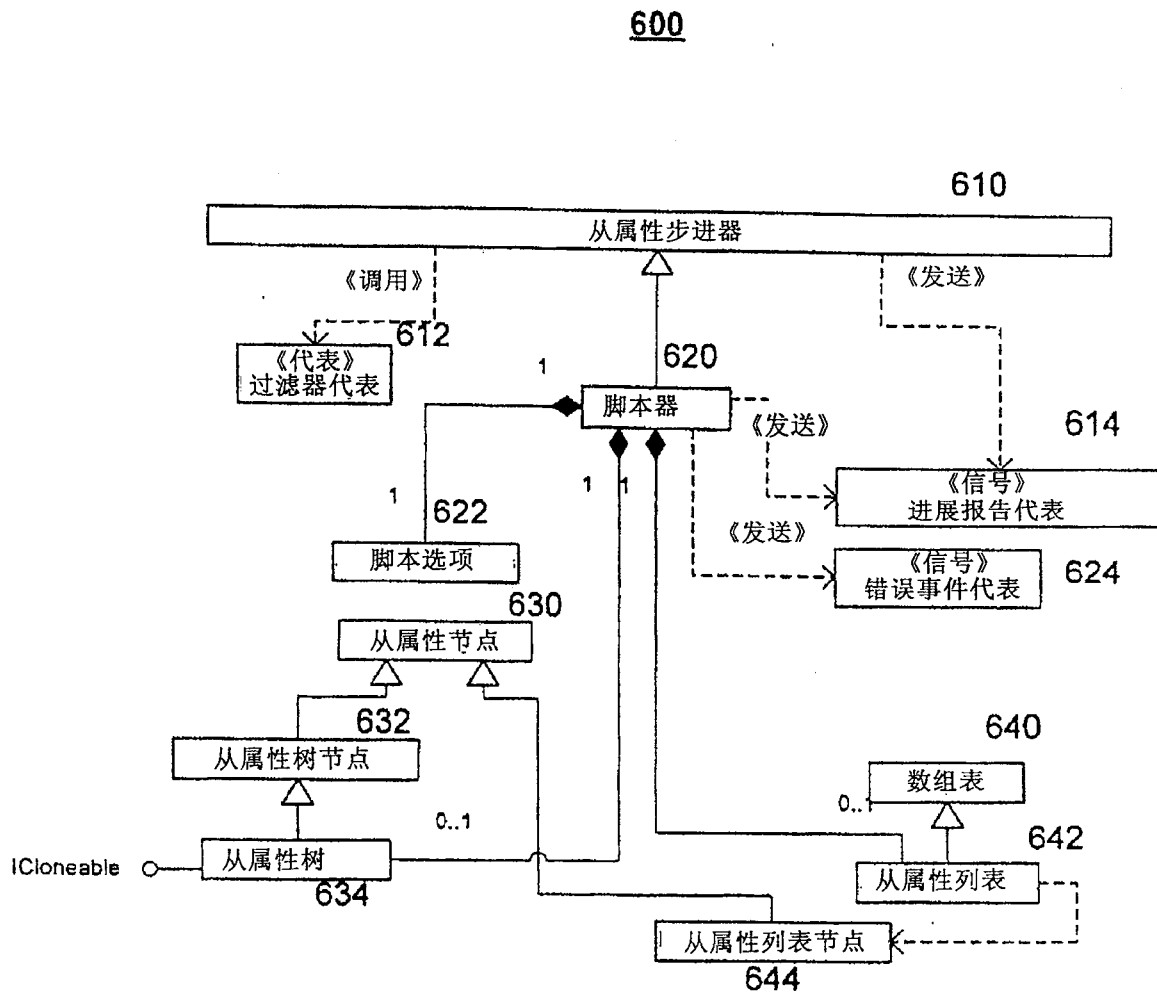


图 6