(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 101416180 B
(45)授权公告日 2015.11.25

(21)申请号 200780012239.9
(22)申请日 2007.02.21
(30)优先权数据
11/395,624 2006.03.31 US
(85)PCT国际申请进入国家阶段日
2008.10.06
(86)PCT国际申请的申请数据
PCT/US2007/004455 2007.02.21
(87)PCT国际申请的公布数据
WO2007/120388 EN 2007.10.25
(73)专利权人 微软技术许可有限责任公司
地址 美国华盛顿州
(72)发明人 J·泽曼
(74)专利代理机构 上海专利商标事务所有限公司 31100
代理人 顾嘉运
(51)Int.Cl.
G06F 17/21(2006.01)
(56)对比文件
US 6907417 B2,2005.06.14,
US 2004239683 A1,2004.12.02,
US 6532471 B1,2003.03.11,
US 2004205726 A1,2004.10.14,
US 6931625 B1,2005.08.16,
审查员 刘浩然

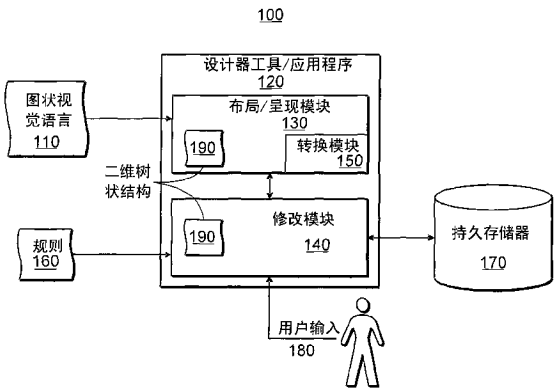
权利要求书4页 说明书13页 附图6页

(54)发明名称

用于编辑图状示图的二维树

(57)摘要

基于图状视觉语言自动将图形示图动态地布局成二维树状结构以便于用户交互与最优显示。显示包括在树状结构的根开始的至少一个分支的图形树状结构，该分支包括被配置成在垂直方向和水平方向上修改的一个或多个子部分。该子部分按照一个或多个对象、属性、操作和关联的功能关系和内部行为来表示整体对象模型。接收修改除了端部分之外的一个或多个子部分的用户输入。基于该用户输入，水平地、垂直地、或两者兼有地修改一个或多个子部分以便于用户交互、优化图形树状结构的显示或两者。



CN 101416180 B

1. 一种在计算系统 (100) 中的用于将示图动态地布局成二维树状结构 (200) 以允许不论该树状结构有多大都可查看该树状结构的所需部分、并与此同时保存示图的流程的方法，所述计算系统 (100) 使用图状视觉语言 (110) 来为诸如系统、服务或过程等事物的图形示图建模，该模型表示整体对象系统，且通常需要手动修改和重新缩放，其中所述计算系统包括设计工具，所述设计工具用于将图状视觉语言显示成二维树状结构，所述方法包括 ：

通过所述设计工具的布局模块，接收图状视觉语言 ；

通过所述布局模块的转换模块，将图状视觉语言转换成图形树状结构 ；

通过所述设计工具的布局模块，显示 (901) 包括至少一个分支的图形树状结构 (200)，所述分支从所述图形树状结构 (200) 的根 (201) 开始并且包括用于按照一个或多个对象、属性、操作、关联的功能关系和内部行为来表示整体对象模型的多个子部分 (231-237)，其中所述多个子部分 (231-237) 被配置成在垂直方向和水平方向上修改，其中所述多个子部分中的每一个包括相关联的用户交互元素和形状，所述相关联的用户交互元素允许对所述子部分进行折叠或展开 ；

通过所述设计工具的修改模块，接收 (902) 选择所述图形树状结构的所述多个子部分中的第一子部分的用户交互 (180)，其中第二子部分连接到所述第一子部分 ；以及

基于所述用户交互，所述修改模块动态地并自动地通过仅显示所述第一用户交互元素来替换所述第一子部分并同时依然显示所述第二子部分来折叠 (903) 所述第一子部分以允许易于用户交互、优化所述图形树状结构 (200) 的显示、或两者。

2. 如权利要求 1 所述的方法，其特征在于，还包括 ：

动态地修改具有耦合到所述第一子部分的至少一个节点的所述第二子部分 ；以及

在修改所述第二子部分之后动态地修改所述第一子部分以使得对所述第二子部分的修改在所述第一子部分被修改时被隐藏，并且其中所述隐藏的对第二子部分的修改能够在接收到附加用户交互时显示。

3. 如权利要求 1 所述的方法，其特征在于，新子部分通过将所述新子部分定位于所述图形树状结构的所需位置来添加到所述图形树状结构，并且其中所述新子部分被自动添加到所述图形树状结构。

4. 如权利要求 1 所述的方法，其特征在于，所述图状视觉语言包括强调在正被建模的系统、服务或过程中有什么东西的结构图 ；强调在正被建模的系统、服务或过程中发生什么事情的行为图 ；以及强调在正被建模的系统、服务或过程中的东西之间的控制和数据流的交互图中的一个或多个。

5. 如权利要求 1 所述的方法，其特征在于，优化所述图形树状结构的显示包括将一个或多个子部分从水平方向自动切换为垂直方向或将一个或多个子部分从垂直方向自动切换为水平方向以针对显示设备或窗口的高宽比进行优化。

6. 一种在计算系统中的用于将示图动态地布局成二维树状结构以允许不论该树状结构有多大都可查看该树状结构的所需部分、并与此同时保存示图的流程的系统，所述计算系统使用图状视觉语言来为诸如系统、服务或过程等事物的图形示图建模，该模型表示整体对象系统，且通常需要手动修改和重新缩放，其中所述计算系统包括设计工具，所述设计工具用于将图状视觉语言显示成二维树状结构，所述系统包括 ：

用于通过所述设计工具的布局模块，接收图状视觉语言的装置 ；

用于通过所述布局模块的转换模块，将图状视觉语言转换成图形树状结构的装置；

用于通过所述设计工具的布局模块，显示包括至少一个分支的图形树状结构的装置，所述分支从所述图形树状结构的根开始并且包括用于按照一个或多个对象、属性、操作、关联的功能关系和内部行为来表示整体对象模型的多个子部分，其中所述多个子部分被配置成在垂直方向和水平方向上修改，其中所述多个子部分中的每一个包括相关联的用户交互元素和形状，所述相关联的用户交互元素允许对所述子部分进行折叠或展开；

用于通过所述设计工具的修改模块，接收选择所述图形树状结构的所述多个子部分中的第一子部分的用户交互的装置，其中第二子部分连接到所述第一子部分；以及

用于基于所述用户交互，所述修改模块垂直地、水平地、或两者兼有地动态地并自动地通过仅显示所述第一用户交互元素来替换所述第一子部分并同时依然显示所述第二子部分来折叠所述第一子部分以允许易于用户交互、优化所述图形树状结构的显示、或两者的装置。

7. 一种在计算系统 (100) 中的用于将示图动态地布局成二维树状结构 (300) 以允许不论该树状结构有多大都可查看该树状结构的所需部分、并与此同时保存示图的流程的方法，所述计算系统 (100) 使用图状视觉语言 (110) 来为诸如系统、服务、或过程等事物的图形示图建模，该模型表示整体对象系统，且通常需要手动修改和重新缩放，其中所述计算系统包括设计工具，所述设计工具用于将图状视觉语言显示成二维树状结构，所述方法包括：

通过所述设计工具的布局模块，接收图状视觉语言；

通过所述布局模块的转换模块，将图状视觉语言转换成图形树状结构；

通过所述设计工具的布局模块，显示 (601) 包括至少一个分支的图形树状结构 (300)，所述分支从所述图形树状结构 (300) 的根 (301) 开始并且包括用于按照一个或多个对象、属性、操作、关联的功能关系和内部行为来表示整体对象模型的多个子部分 (350-361)，并且其中所述子部分 (350-361) 包括至少一个端部分 (361)，其中所述多个子部分中的每一个包括相关联的用户交互元素和形状，所述相关联的用户交互元素允许对所述子部分进行折叠或展开；

通过所述设计工具的修改模块，接收 (602) 选择所述图形树状结构中除了所述至少一个端部分 (361) 之外的所述多个子部分中的第一子部分的用户交互元素的用户交互 (180)，其中第二子部分连接到所述第一子部分；以及

基于所述用户交互，所述修改模块动态地并自动地通过仅显示所述第一用户交互元素来替换所述第一子部分并同时依然显示所述第二子部分来折叠所述第一子部分以允许易于用户交互、优化所述图形树状结构 (300) 的显示、或两者，而无需修改所述至少一个端部分。

8. 如权利要求 7 所述的方法，其特征在于，所述多个子部分能够垂直地、水平地、对角地、或以其任何组合修改。

9. 如权利要求 7 所述的方法，其特征在于，还包括：

接收指定所述树状结构将如何被修改和显示的一条或多条预定义规则；以及

基于所述一条或多条规则以及所接收到的用户交互，动态地缩放、展开或折叠除了所述至少一个端部分之外的子部分。

10. 如权利要求 7 所述的方法，其特征在于，还包括将一维图形树状结构或不可修改图形树状结构中的一个自动转换成二维图形树状结构。

11. 如权利要求 7 所述的方法，其特征在于，还包括将两个或多个子部分的连接线的段组合成单个连接线。

12. 如权利要求 7 所述的方法，其特征在于，接收用户交互包括 ：

选择一个或多个子部分以供缩放、折叠或展开。

13. 如权利要求 7 所述的方法，其特征在于，折叠包括 ：

将所述子部分中的至少一个自动格式化为单个图形对象，所述单个图形对象被配置成可在接收到附加用户交互时展开回到所述至少一个子部分。

14. 如权利要求 7 所述的方法，其特征在于，展开包括 ：

将可展开图形对象自动格式化为至少一个子部分。

15. 一种在计算系统中的用于将示图动态地布局成二维树状结构以允许不论该树状结构有多大都可查看该树状结构的所需部分、并与此同时保存示图的流程的系统，所述计算系统使用图状视觉语言来为诸如系统、服务、或过程等事物的图形示图建模，该模型表示整体对象系统，且通常需要手动修改和重新缩放，其中所述计算系统包括设计工具，所述设计工具用于将图状视觉语言显示成二维树状结构，所述系统包括 ：

用于通过所述设计工具的布局模块，接收图状视觉语言的装置 ；

用于通过所述布局模块的转换模块，将图状视觉语言转换成图形树状结构的装置 ；

用于通过所述设计工具的布局模块，显示包括至少一个分支的图形树状结构的装置，所述分支从所述图形树状结构的根开始并且包括用于按照一个或多个对象、属性、操作、关联的功能关系和内部行为来表示整体对象模型的多个子部分，并且其中所述子部分包括至少一个端部分，其中所述多个子部分中的每一个包括相关联的用户交互元素和形状，所述相关联的用户交互元素允许对所述子部分进行折叠或展开 ；

用于通过所述设计工具的修改模块，接收选择所述图形树状结构中除了所述至少一个端部分之外的所述多个子部分中的第一子部分的用户交互元素的用户交互的装置，其中第二子部分连接到所述第一子部分 ；以及

用于基于所述用户交互，所述修改模块通过缩放、展开或折叠中的一个或多个来动态地并自动地通过仅显示所述第一用户交互元素来替换所述第一子部分并同时依然显示所述第二子部分来折叠所述第一子部分以允许易于用户交互、优化所述图形树状结构的显示、或两者，而无需修改所述至少一个端部分。

16. 一种在计算系统 (100) 中的方法，所述方法用于将示图动态地布局成二维树状结构以允许不论该树状结构有多大都可查看该树状结构的所需部分、并与此同时保存示图的流程以及用于将对视觉树状结构 (200) 中的一个或多个子部分 (231-237) 作出的修改持久保留在存储器 (170) 中，以使得由用户对所述视觉树状结构 (200) 的显示做出的改变甚至在用于修改所述视觉树状结构 (200) 的应用程序 (120) 被用户退出时也被保持，所述计算系统 (100) 包括用于为诸如系统、服务或过程等事物的示图建模的图状视觉语言 (110)，该模型表示被显示为所述视觉树状结构 (200) 的整体对象系统，所述方法包括 ：

激活 (701) 显示包括至少一个分支的图形树状结构 (200) 的应用程序 (120)，所述分支从所述图形树状结构 (200) 的根开始并且包括按照一个或多个对象、属性、操作、关联的功

能关系和内部行为来表示整体对象模型的多个子分支 (231-237)，其中所述多个子分支中的每一个包括相关联的用户交互元素和形状，所述相关联的用户交互元素允许对所述子分支进行折叠或展开；

所述应用程序的修改模块接收 (702) 选择所述多个子分支 (231-237) 中的第一子分支的用户交互元素的用户输入 (180)，所述第一子分支连接到第二子分支；

基于用户交互，所述修改模块动态地并自动地通过仅显示所述第一用户交互元素来替换所述第一子分支并同时依然显示所述第二子分支来折叠所述第一子分支来折叠所述第一子分支以允许易于用户交互、优化所述图形树状结构 (200) 的显示、或两者；以及

在持久存储器 (170) 中保持 (703) 所述折叠，以使得在用户已退出所述应用程序后的稍后时间重新激活所述应用程序时，所述图形树状结构 (200) 被显示为在所述应用程序 (120) 退出之前它出现的样子。

17. 如权利要求 16 所述的方法，其特征在于，还包括：

接收执行对具有耦合到所述第一子分支的至少一个节点的所述第二子分支的修改的用户输入；

在修改所述第一子分支之前修改所述第二子分支以使得对所述第二子分支的修改在所述第一子分支被修改时被隐藏；以及

在退出所述应用程序时在所述持久存储器中保存所述隐藏的所述第二子分支的已修改状态，以使得在稍后运行所述应用程序时所述隐藏的对所述第二子分支的修改可在接收到附加用户交互时被显示。

用于编辑图状示图的二维树

[0001] 背景

[0002] 计算系统已经彻底改变了人们工作和游戏的方式。计算系统有各种各样形式，包括膝上型计算机、台式计算机、个人数字助理、电话、甚至传统上不与计算系统相关联的设备，诸如例如冰箱和汽车等。计算系统甚至可包括经由网络互连的多个组成的计算系统。因此，某些计算系统可能足够小以适合手掌，而其他计算系统遍布全球。

[0003] 已变得日益流行的计算技术的一个领域是图形建模，其允许用户在视觉上显示诸如众多系统、服务、或过程的流程图等图状示图(graph-like diagram)。示图的示例包括工作流程图、数据流图、E-R图、业务流程图等等。更具体地，这些图状示图允许用户在视觉上传达不同计算或其他系统、商业或项目模型等的各种组件或元素的功能、行为方面、以及关系；因此允许经常是复杂的系统或过程的高效传达。

[0004] 例如，图状示图可以可视化销售交易的流程。该流程图的元素和组件可以表示诸如选择所需产品、下定单、安排递送和支付等事件。该示图还可以可视化执行以上所列出的任务的实体。对于涉及众多实体的非常复杂的销售交易，图形化地查看该交易可以比以其他形式更容易理解。

[0005] 虽然图状示图在可视化系统和过程中很有用，但是存在对其有用性的限制。例如，随着示图变得更大且更复杂，由于显示设备的大小、分辨率、以及其他能力可能只是没有足够的空间来显示所有组件和元素。换言之，可用显示空间和能力限制同时能够查看多少组件和对象。对于用户想在其上显示过程的开始和结束两者的示图而言这尤其成问题。

[0006] 一种解决方案将是使用更大的显示设备。然而，由于这将需要的成本，对于大多数用户这是不切实际的。此外，示图可能变得太大以使得没有实际上的显示设备能够显示该示图的所有元素。

[0007] 通常使用的对于该显示限制的另一种解决方案是让用户手动地重新缩放该示图。例如，用户将经常在新元素被添加到示图的任何时候“放大”。虽然该解决方案可能对于只有少量元素的图有效，但是对于较大的示图却不然。这些示图在被放大若干次后经常变得不可阅读。

[0008] 另一种常见解决方案是制作一起包含可读大小的所有所需示图元素的多个示图。然而，这经常是费时的。此外，使用多个示图经常丢失示图的流程或“思维地图(mental map)”，这对于用户是令人沮丧的。

[0009] 另一个解决方案是创建自动对示图的元素布局的工具以试图将示图缩放为可读形式。然而，这些工具实现复杂的算法，其计算复杂度使得处理它们非常缓慢。在新元素被添加到该示图的任何时候，这些工具必须重新处理该算法并且创建新示图，使得这些工具对于交互式编辑器是不切实际的。此外，这些算法中的大多数在添加新元素时重新定位该示图的每个元素。结果，用户经常丢失他或她的该示图的“思维地图”，这是恼人的。

[0010] 简要概述

[0011] 以上所标识的现有示图生成和显示方法的缺陷和缺点通过此处所公开的示例实施例来克服。注意，提供本概述以便用简化的形式介绍将在以下详细描述中进一步描述的

一些概念。该概述不旨在标识所要求保护的主题的关键特征或必要特征，也不旨在用于帮助确定所要求保护的主题的范围。

[0012] 一个示例实施例提供基于图状视觉语言自动将图形示图布局成二维树状结构以便于用户交互与最优显示。在该实施例中，显示图形树状结构。该图形树状结构包括至少一个分支，该分支从该图形树状结构的根开始并且包括被配置成在垂直方向和水平方向上修改的一个或多个子部分。该子部分按照一个或多个对象、属性、操作和关联的功能关系和内部行为来表示总对象模型。然后接收修改一个或多个子部分的用户输入。基于该用户输入，水平地、垂直地、或两者兼有地动态修改一个或多个子部分以便于用户交互、优化图形树状结构的显示或两者。

[0013] 另一个示例实施例还能够基于图状视觉语言动态地将流程图布局成二维树状结构以便于用户交互与最优显示。在该实施例中，显示图形树状结构。该图形树状结构包括从该树状结构的根开始并且包括多个子部分的至少一个分支。子部分中的至少一个也是端部分。该子部分按照一个或多个对象、属性、操作和关联的功能关系和内部行为来表示总对象模型。然后接收修改除了端部分之外的一个或多个子部分的用户输入。基于该用户输入，缩放、展开、或折叠除了端部分之外的一个或多个子部分而无需也修改该端部分。该修改允许易于用户交互、优化图形树状结构的显示或两者。

[0014] 另一个示例实施例能够在显示图形树状结构的应用程序退出时将对该图形树状结构的一个或多个子分支作出的修改持久保留在存储器中。在该实施例中，激活显示图形树状结构的应用程序。该图形树状结构包括从该树状结构的根开始的至少一个分支，该分支包括按照一个或多个对象、属性、操作和关联的功能关系和内部行为来对系统、服务、或过程建模的一个或多个子分支。接收修改至少一个子分支的用户输入。基于该用户输入，将该修改保持在持久存储器中。在用户退出应用程序之后的稍后时间重新激活该应用程序后，图形树状结构被显示为在该应用程序退出之前其出现的样子。

[0015] 本发明的附加特征和优点将在以下描述中叙述，且其一部分根据本描述将是显而易见的，或可通过对本发明的实践来获知。本发明的特征和优点可通过在所附权利要求书中特别指出的手段和组合来实现和获得。本发明的这些和其他特征将通过以下描述和所附权利要求书而变得更加完全明显，或可通过对下文中所述的本发明的实践来获知。

[0016] 附图简述

[0017] 为了描述可获得本发明的上述和其他优点特征的方式，将通过引用附图中示出的本发明的具体实施例来呈现以上简要描述的本发明的更具体描述。可以理解，这些附图仅描述本发明的典型实施例，从而不因此被认为是对其范围的限制，本发明将通过使用附图用附加的特征和细节来描述和解释，附图中：

[0018] 图 1 示出用于动态地将图状视觉语言显示为图形二维树状结构的计算系统；

[0019] 图 2A 示出图状视觉语言的示例；

[0020] 图 2B 示出根据各示例实施例被显示为图形二维树状结构的图 2A 的图状视觉语言；

[0021] 图 3A 示出根据各示例实施例的示例图形二维树状结构；

[0022] 图 3B 示出对图 3A 的图形二维树状结构所作出的修改；

[0023] 图 3C 示出对图 3A 的图形二维树状结构所作出的进一步修改；

[0024] 图 4A 示出根据各示例实施例的另一个示例图形二维树状结构；

[0025] 图 4B 示出对图 4A 的图形二维树状结构所作出的修改；

[0026] 图 4C 示出对图 4A 的图形二维树状结构所作出的进一步修改；

[0027] 图 5 示出图形一维树状结构；

[0028] 图 6 示出根据各示例实施例用于将图状视觉语言的流程图自动布局成图形二维树状结构的方法的流程图；

[0029] 图 7 示出根据各示例实施例用于将对图形树状的一个或多个子分支作出的修改持久保留在存储器中的方法的流程图；

[0030] 图 8 示出根据各示例实施例用于将图形树状结构转换为图形二维树状结构的方法的流程图；以及

[0031] 图 9 示出根据各示例实施例用于将图状视觉语言的流程图自动布局成图形二维树状结构的方法的流程图。

[0032] 详细描述

[0033] 本发明涉及用于将图状视觉语言自动布局成二维树状结构以允许易于用户交互和最优显示的各种方法、系统和计算机程序产品。本发明的实施例可以包括含有各种计算机硬件或模块的专用或通用计算机，这将在以下做出进一步讨论。

[0034] 示例实施例提供被配置成克服现有工具的各种缺陷、用于以允许动态交互和最优显示的方式来布局图状视觉语言的若干机制。例如，此处一个实施例提供一种用于将流程图动态布局成二维树状结构的机制。该图状视觉语言被显示为二维图形树状结构。该图形树状结构包括从该树状结构的根开始并且包括子部分的至少一个分支。该子部分用于按照该树状结构的一个或多个对象、属性、操作和关联的功能关系和内部行为来表示总对象模型。该子部分还可包括至少一个端部分，尽管这对于所有实施例并不是必需的。

[0035] 该树状结构以有助于用户交互的方式来显示。例如，可以显示控制子部分的修改的用户交互式 (UI) 元素。接收诸如选择 UI 元素等的修改不是端部分的一个或多个子部分的用户交互。这些修改包括缩放、折叠、以及展开子部分。

[0036] 基于该用户交互，动态修改二维树状结构。例如，在某些实施例中，除了端部分之外的子部分可以被缩放以优化树状结构在可用显示空间中的显示，或者子部分可以被折叠或展开以允许不论该树状结构有多大都可查看该树状结构的所需部分。在其他实施例中，可以水平地、垂直地、或两者兼有地修改子部分。

[0037] 其他实施例允许甚至在运行该树状结构的应用程序退出时持久保留对树状结构作出的修改。例如，激活显示图形树状结构的应用程序并且接收以先前示例的方式修改图形树状结构的用户输入。然而，基于该用户输入，将对该树状结构的修改保持在持久存储器中。此外，因对其他子部分的后续修改而变为隐藏的对一个子部分的修改也可被保持在持久存储器中。在用户退出了应用程序之后的稍后时间重新激活该应用程序后，图形树状结构被显示为在该应用程序退出之前它出现的样子。对可见和隐藏的修改的保持允许随着时间的推移显示图形树状结构而不必在应用程序退出和重启时一直重做修改。此外，保持隐藏的状态保证在应用程序退出时没有子部分丢失。

[0038] 其他实施例允许不能被修改的图状视觉语言和树状结构转换成能够被动态修改的二维树状结构。这些实施例还允许包括一个分支并包括端部分的一维树状结构的转换，

该分支从该树状结构的根开始并且包括对系统、服务和过程建模的多个子部分。该一维树状结构的子部分不能够在也不修改端部分的情况下修改，这与其子部分能够在无需也修改端部分的情况下修改的二维树状结构相反。将这些结构转换为二维树状结构给予用户利用该动态修改并且对该二维树状结构进行显示优化的能力。此外，这允许通过按需在垂直和水平之间自动切换子部分的方向来自动优化到可用显示屏幕或窗口的高宽比。

[0039] 虽然如下将参考附图对有利特征的更具体叙述进行更为详尽的描述，但是本发明范围内的各实施例还包括用于承载或在其上存储计算机可执行指令或数据结构的计算机可读介质。这样的计算机可读介质可以是可由通用或专用计算机访问的任何可用介质。作为示例而非限制，这样的计算机可读介质可包括 RAM、ROM、EEPROM、CD-ROM 或其它光盘存储、磁盘存储或其它磁存储设备、或可用于承载或存储计算机可执行指令或数据结构形式的所需程序代码装置且可由通用或专用计算机访问的任何其它介质。当信息通过网络或另一通信连接（硬连线、无线或硬连线或无线的组合）传输或提供给计算机时，该计算机将该连接完全视为计算机可读介质。因此，任何这样的连接被适当地称为计算机可读介质。以上的组合也应包括在计算机可读介质的范围之内。

[0040] 计算机可执行指令包括例如，使通用计算机、专用计算机、或专用处理设备执行某一功能或某组功能的指令和数据。尽管用对结构特征和 / 或方法动作专用的语言描述了本主题，但可以理解，所附权利要求书中定义的主题不必限于上述具体特征或动作。相反，上述具体特征和动作是作为实现权利要求的示例形式公开的。

[0041] 如此处所使用的，术语“模块”或“组件”可以指在计算系统上执行的软件对象或例程。此处描述的不同的组件、模块、引擎和服务可被实现为在计算系统上执行的对象或进程（例如，作为分开的线程）。尽管此处描述的系统和方法较佳地可用软件来实现，但用硬件或软件和硬件的组合的实现也是可能的且已被想到。在本说明书中，“计算实体”可以是如上文定义的任何计算系统，或者是在计算系统上运行的任何模块或模块的组合。

[0042] 现在参考附图，图 1 示出可用于实现此处所公开的各种实施例的计算系统 100。计算系统 100 描绘了能够在实现各实施例时使用的各种模块和组件。注意，计算系统 100 只是作为示例而示出，并且不应用于限制所附权利要求书或此处所公开的各实施例中的任一个的范围。换言之，存在可用于实现此处所描述的各实施例的众多其他计算系统和组件配置。

[0043] 计算系统 100 包括可用于对系统、服务和过程的流程图建模的图状视觉语言 110。图状视觉语言的常见示例包括强调在所建模的系统、服务或过程中通常有什么东西的结构图；强调在所建模的系统、服务或过程中一般发生什么事情的行为图；或强调在所建模的系统、服务或过程中的东西之间的控制和数据流的交互图。其他常见图状视觉语言包括各种示图类型的统一建模语言 (UML)、数据库领域中的实体关系图、以及用于业务流程设计的业务流程图。一般地，此处所描述的各实施例应用于可以是任何通用图状视觉语言的图状视觉语言 110。因此，在本说明书中对特定图状视觉语言的任何引用只是为了说明并且不应用于限制此处所公开的各实施例的范围。

[0044] 计算系统 100 还包括帮助将图状视觉语言 110 显示成二维树状结构 190 的设计工具 / 应用程序 120。设计工具 / 应用程序 120 还帮助动态修改所显示的二维树状结构 190。设计工具 / 应用程序 120 及其各种模块和 / 或组件中的任一个都可被实现为软件、硬件、或

两者的任何组合并且可被主存在单个本地或远程计算设备上或在分布式计算环境中。

[0045] 设计工具 120 包括将图状视觉语言 110 显示成二维树状结构 190 的布局 / 呈现模块 130。在某些实施例中，布局 / 呈现模块 130 显示由用户输入到设计器工具 / 应用程序 120 中的元素。设计器工具 / 应用程序 120 然后根据由所实现的图状视觉语言 110 定义的关系在各元素之间自动绘出互连线。

[0046] 在其他实施例中，布局模块 130 接收或以其他方式访问在接收到的时间之前存在的图状视觉语言 110。现有的图状视觉语言 110 可以从主存设计工具 120 的计算机或从通过任何类型的网络连接到计算系统 100 的某一其他计算机接收。在这些实施例中，图状视觉语言 110 可作为具有不能被修改的子部分的树状结构来接收，或者它可作为如将在以下更详细描述的具有只可被一维地修改的子部分的一维树状结构来接收。然后可以将现有的图状视觉语言提供给转换模块 150 以便转换成如也将在以下更详细解释的二维树状结构。

[0047] 不论图状语言 110 是预先存在的还是由用户创建的，布局 / 呈现模块 130 都自动将该图状语言显示成二维树状结构 190。树状结构 190 可包括从该树状结构的根开始的至少一个分支并且包括多个子部分或子分支。子部分中的一个也可以是端部分。该子部分按照所建模的系统、服务或过程的对象、属性、操作和关联的功能关系和内部行为来对由图状语言 110 显示的系统、服务或过程建模。

[0048] 此外，该二维树状结构 190 由布局 / 呈现模块 130 以允许该树状结构的子部分由设计器工具 / 应用程序 120 垂直地、水平地、对角地或三者兼有地动态及自动地修改的方式来构造或转换。在某些实施例中，也是端部分的子部分不被修改。二维树状结构 190 的具体示例将在下文中给出。

[0049] 设计器工具 / 应用程序 120 还包括修改模块 140。修改模块 140 从用户接收修改树状结构 190 的子部分的用户输入 180。例如，该用户输入可以选择可修改的子部分以供折叠、展开或缩放。响应于该用户输入，修改模块 140 动态及自动地修改正由布局模块 130 显示的树状结构的子部分中的一个或多个。

[0050] 例如，该子部分可以被缩放以优化可用显示空间，或者可被折叠或展开以允许树状结构 190 的所选子部分的聚焦显示。有利的是，树状结构 190 的配置允许修改给定子部分而不必也修改从该给定子部分分叉出来的任何或所有子部分，因此允许用户在运行中动态地显示所需子部分。可以水平地、垂直地、对角地（可被认为是垂直或水平的组合）、或用三者的任何组合来修改该子部分。

[0051] 此外，所修改的子部分被动态地定位以使得所显示的子部分优化可用显示空间，这有利地允许流程图维持最优外观。在某些情况下，可以垂直地或对角地重新定位水平显示的子部分以平衡显示。对垂直和对角显示的子部分的改变也可被改变。以上优点将在以下对二维树状结构的具体示例的讨论中变得更加显而易见。

[0052] 在某些实施例中，修改模块 140 也接收指定树状结构 190 将被如何修改和 / 或显示的预定义规则 160。在这些实施例中，修改模块 160 基于已经讨论过的用户交互以及所接收到的规则 160 动态地缩放、展开或折叠除了至少一个端部分之外的子部分。一个示例规则 160 可引导布局模块 130 在流程图中将第二图形元素放置在第一图形元素的右侧并且将所有附加对象放置在该第二对象的下面。

[0053] 在其他实施例中，设计器工具 / 应用程序 120 可以访问持久存储器 170。持久存储

器 170 可以是任何类型的合理持久存储器并且可驻留在主存设计器工具 / 应用程序 120 的计算机上，或者可通过任何类型的网络连接来访问。在这些实施例中，使用持久存储器 170 以在用户退出设计器工具 / 应用程序 120 时保持对树状结构 190 的子部分所作出的任何修改。当该用户重新激活设计器工具 / 应用程序 120 时，如将在以下更详细解释的，树状结构 190 被显示成在设计器工具 / 应用程序 120 退出之前它出现的样子。

[0054] 参考图 2A，描绘了一具体的图状视觉语言流程图 200。流程图 200 是示出行动者可以如何使用软件来下定单的用例图 (UML 的一部分)。该示图中所示出的是不同的对象和连接器类型。例如，形状 201 是行动者类型。其他形状表示该行动者所做的事情或由该过程完成的事情。在各种形状之间的线是表示各形状之间的关系的连接器。在图 2A 中，连接器可以是例如 < 使用 > 或 < 扩展 > 类型以及其他连接器类型。

[0055] 例如，用户 (行动者)201 可以下订单 208。如连接器 211 所示的，下定单可展开到请求商品目录 204。下定单使用提供动作顾客数据 202、定购产品 203 以及安排支付 205，如连接器 212-214 所示的。支付类型可以是现金 206 或可安排信用卡 207，如连接器 215 和 216 所示的。

[0056] 如上所述，设计器工具 / 应用程序 120 可以接收流程图 200 并且将该流程图 200 转换为二维树状结构，其包括可通过如上所述在垂直和水平方向上并且可能在对角方向上缩放、折叠和展开来修改的子部分 231-237。现在参考图 2B，流程图 200 被示为在经历了转换和显示过程之后它可能出现在设计器工具 / 应用程序 120 中的样子。注意，可与树状结构 190 相对应的示图 200 尽管不是必需的，但仍旧包括先前描述的对象 201-208。还要注意，用户交互式 (UI) 元素 221-229 也已由设计器工具 / 应用程序 120 添加到该树状结构。该 UI 元素用于允许用户动态地修改树状结构 200 的子部分。

[0057] 在图 2B 中，连接器 212、213、和 214 已基于其作为 < 使用 > 类型连接器的共同性被转换为单个连接分支。以同样的方式，连接器 215 和 216 基于其共同的连接器类型被转换为单个连接分支。这两者都是多连接器的示例，其中在向二维树状结构的转换期间，各子部分之间的两个或多个连接被组合成表示组合连接器的连接器。当然，如果情况保证，则不同类型的连接器能够被组合成单个连接器也是可能的。例如，在图 2 中，一般对应于子部分 237 的垂直线将会是多连接器，而对应于子部分 233 和 234 的水平线将会是特定连接器。在这些实施例中，不同类型的连接器已被组合成单个连接器的指示可被放置在对应于子部分 233 和 234 的水平连接器旁边。

[0058] 在该图中，用户 201 充当该树状结构的根。注意，可能存在不止一个根。例如，“下定单”208 可以是从其分叉出的两个分支的根，而“安排支付”205 也可用作从其分叉出的分支的根。树状结构 200 还包括用作分支的末尾的若干端部分。例如，“请求商品目录”204 可以是端部分，因为它是分支的末尾。以同样的方式，“提供顾客数据”202、“定购产品”203、“安排支付”207、以及“支付现金”206 也可被认为是端部分，因为它们是分支的末尾。在某些实施例中，可对除了端部分之外的子部分作出修改。在其他实施例中，可对所有子部分作出修改。

[0059] 如上所述，设计器工具 / 应用程序 120 将 UI 元素 221-229 添加到二维树状结构 200。这些 UI 元素允许用户水平及垂直地动态地修改各子部分 231-237。该修改通常是缩放、折叠或展开各子部分中的一个。现在将描述若干示例。注意，尽管示例通常使用折叠作

为所示出的修改，但是展开和 / 或缩放也可甚至在没有显式提到时执行。

[0060] 例如，可以选择 UI222，使得在垂直方向上折叠包括对象 204 的子部分 232。同时，可选择 UI223，使得折叠具有其相对应对象的子部分 233-237。因此，二维树状结构 200 具有水平或者垂直修改的能力。尽管图 2B 未示出，但是如将在以下参考图 3 描述的，二维树状机构还可在对角方向上修改。

[0061] 此外，子部分也可在水平和垂直两个方向上修改。例如，可以选择 UI226，使得在垂直方向上折叠对象 206 和 207。相反，可以选择 UI227 和 / 或 228，使得水平折叠对象 206 和 / 或 207。以同样的方式，子部分 233、234、和 237 在选择 UI223 的情况下可被垂直修改并且在分别选择 UI224、225、和 229 的情况下也可被水平修改。

[0062] 在某些实施例中，二维树状结构 200 也具有对该树状结构 200 的子部分作出修改而无需也对从被修改的子部分分叉出的子部分自动作出修改的能力。例如，可以折叠 UI225。这将折叠包括对象 205 的子部分 234。然而，子部分 235 和 236 将不会也被自动折叠。它们可以简单地重新缩放以优化可用查看空间。当然，用户可能只需要显示对象 206，在这一情况下对象 207 在接收到对于该端点的用户交互之后也可被水平折叠。

[0063] 作为又一个示例，包括对象 208 的子部分 231 可通过选择 UI221 来折叠。在许多传统的一维树状结构中，这将自动折叠该树状结构的剩余部分。在二维树中，仍然显示其余子部分直到用户交互指定对它们的修改，尽管在某些实施例中它们可以被自动缩放以允许优化显示。因此，用户可以动态地修改树状结构 200 以便只显示用户所期望的子部分。

[0064] 注意，在以下的图 2B 以及图 3 和 4 中，为每个对象或子部分只示出了一个 UI 元素。所示 UI 元素允许折叠和展开子部分。可以理解，也可以采用更大数量的 UI 元素。可以使用这些附加 UI 元素来展开和折叠特性标志、用户注释、对象和 / 或子部分之间的附加关系等等。因此，此处所公开的各实施例构想了实现任何数量的 UI 元素的二维树状结构。

[0065] 为了进一步示出由设计工具 / 应用程序 120 提供的动态修改能力，图 3A 描绘了可与图 1 的树状结构 190 相对应的通用二维树状结构 300。图 3A 包括各种形状 301-313。注意，这些形状包括或描述各种对象、属性、操作、关联等等。形状 301-313 的确切特性对于此处所描述的各实施例并不重要，并且被示为各种形状以反应这一点。树状结构 300 还包括 UI 元素 321-333，这些元素放置于该树状结构中以允许通过展开 / 折叠以及缩放机制来动态修改该树状结构的子部分。注意，树状结构 300 的总体布局只是为了说明并且不应用于限制此处所公开的各实施例。

[0066] 形状 301 可充当该树状结构的根，尽管其他元素也可以是该树状结构的根。从该根分叉出的是各个子部分 350-361。注意，某些子部分是水平方向的，某些是垂直方向的，而某些是对角方向的。因为树状结构 300 是二维的，所以如果情况保证则它可以在这些方向的任一个上修改。如上所述，修改通常是缩放、折叠、或展开。

[0067] 参考图 3B，描绘了在某些子分支被修改之后的树状结构 300。例如，用户可能期望修改对角子部分 350。用户能够提供将选择 UI 元素 321 的输入，因此在对角方向上折叠包括形状 302 的子部分 350。在某些实施例中，当选择 UI321 时子部分 351 和 352 也可被对角折叠。在其他实施例中，二维树状结构 300 只折叠对角子部分 350 而仍旧显示子部分 351 和 352。在一维树状结构中，子部分 351 和 352 通常始终将被折叠。稍后，可以选择 UI321 以动态地展开对角子部分 350，如果这是所期望的。

[0068] 以同样的方式，用户也可能期望修改垂直子部分 360。用户将提供将选择 UI 元素 331 的输入，因此在垂直方向上折叠包括形状 312 的子部分 360。在一维树状结构中，子部分 360 和 352 通常始终也将被折叠。然而，在某些实施例中，二维树状结构 300 只折叠垂直子部分 360 而仍旧显示子部分 361。稍后，可以选择 UI331 以动态地展开垂直子部分 360，如果这是所期望的。

[0069] 类似地，用户也可能期望修改水平子部分 354。用户将提供将选择 UI 元素 326 的输入，因此在水平方向上折叠包括形状 306 的子部分 354。如上所述，在一维树状结构中，子部分 355 和 356 通常也将被折叠。在某些实施例中，二维树状结构 300 只折叠水平子部分 306 而仍旧显示子部分 307 和 308。如同对角和垂直修改，稍后，可以选择 UI326 以动态地展开水平子部分 354，如果这是所期望的。

[0070] 此外，图 3B 也示出包括形状 314 和 UI 元素 334 的新子部分 362 已被添加到二维树状结构 300。设计器工具 / 应用程序 120 允许用户选择期望向其添加新子部分的树状结构 300 的一部分。例如，用户可以使用用户界面，该用户界面允许用户选择并拖拽一新图形子部分或对象并将其放置在树状结构上所期望的位置。设计器工具 / 应用程序 120 然后可以自动添加该新子部分或对象；且树状结构 300 被缩放以优化可用显示空间。因此，该用户可以动态地将新子部分或对象添加到树状结构 300。

[0071] 参考图 3C，示出了对二维树状结构 300 的进一步修改。在该描绘中，示出了根形状 301 以及包括形状 311 的端部分 359。也示出了表示图 3A 的子部分 350 以及超出子部分 350 的所有子部分的框 380。类似地，框 390 表示图 3A 的子部分 353 以及超出子部分 353 的所有子部分。注意，使用框 380 和 390 只是为了方便，并且并不意味着暗示设计器工具 120 的任何实际显示。

[0072] 图 3C 示出二维树状结构 300 修改任何子部分而不必也修改从被修改的部分分叉出的子部分的能力。尽管只有三个示例修改在树状结构 300 中示出，但是图 3A 所示的其他子部分中的任一个也可按需水平地、垂直地和 / 或对角地缩放、折叠、或扩展。用户因此具有动态地展开框 380 和 390 的各部分以显示被认为是该用户所期望的那些部分的能力。这有利地允许该用户聚焦于最感兴趣的子部分。

[0073] 此外，图 3C 示出树状结构 300 具有如果需要折叠除了根形状和端部分之外的所有元素的能力。这对于具有几百个子部分的树状结构而期望只显示例如过程的开头和结尾而言是有利的。注意，当在图 3B 和 3C 中作出修改时（尽管未示出），设计器工具 120 也动态地缩放树状结构 300 以最优地适合可用显示空间。

[0074] 以上已提到过很多次，此处所公开的某些实施例与具有各子部分的二维树状相关，这些子部分可被修改而不必也修改端部分或从正被修改的子部分分叉出的子部分。这与一维树状结构通常如何被修改不同。这些一维树状结构一般被显示为具有根以及包括从主分支分叉出的各个子部分和 / 或其他子部分的一个或多个分支。一维树状结构的示例包括公知的计算机操作系统的文件和文件夹目录。一维树状结构允许用户折叠和展开各个子部分。然而，与上述二维树状结构的某些实施例截然相反的是，充当端部分的子部分通常在修改其上方的任何其他子部分的任何时候也被修改。

[0075] 例如，图 5 示出了典型的一维树状结构 500。树状结构 500 包括可以是任何图形对象的形状 501-504。第二子部分 522 从第一子部分 521 分叉出并且也充当该树状结构中的

端部分的第三子部分 523 从该第二子部分 522 分叉出。如果用户折叠该第一子部分 521，则包括形状 502、503 和 504 的该第二和第三子部分 522 和 523 也将自动折叠。以同样的方式，折叠该第二子部分 522 将自动折叠包括形状 503 和 504 的该第三子部分 523。用户仅仅是不能够在仍然显示该第三子部分 523 的同时折叠该第二子部分 522，或在仍然显示该第二和 / 或第三子部分 522 和 523 的同时折叠该第一子部分 521。以同样的方式，展开子部分 521 和 / 或 522 也自动展开子部分 523。在以上所有场景中，对端部分 523 以上的子部分的任何修改都会导致端部分 523 也被修改。

[0076] 现在参考图 4A-4C，示出了二维树状结构的某些实施例的其它修改方面的又一个示例。在图 4A 中，用形状 401-405 来示出可与图 1 的树状结构 190 相对应的二维树状结构 400。如同先前的示例，形状 401-405 的确切特性并不重要。在该示例中，形状 401 表示也包括子部分 410-413 的树状结构 400 的根。如上所述，可接收修改子部分 411 的用户输入。

[0077] 参考图 4B，修改子部分 411 得到表示已被折叠的形状 402 和 403 的 UI 元素 460。树状结构 400 现在包括形状 401、404 和 405。此外，然后创建包括子部分 410、412 以及 UI 元素 460 的子部分 430。

[0078] 现在参考图 4C，示出了用户已修改子部分 430，这得到表示已被折叠的形状 401 和 404 的 UI 元素 461 以及表示先前折叠的形状的 UI 元素 460。注意，尽管 UI 元素 460 在图 4C 中被示为在 UI 元素 461 内部，但是这不是一个必要的特征。例如，在某些实施例中，单个 UI 元素在被用户选择时可使得 UI 元素 460 和 461 的所有折叠的子部分展开。在其他实施例中，UI 元素 460 可以在 UI 元素 461 中隐藏并且可以直到 UI 元素 461 被展开才可见。

[0079] 在其他实施例中，图 4C 示出设计器工具 120 能够记住隐藏子部分的状态。例如，如果 UI 元素 461 稍后被用户展开，则将显示形状 401 和 404，但是形状 402 和 403 将仍然被折叠并由 UI 元素 460 来表示直到 UI 元素 460 稍后被展开。

[0080] 在其他实施例中，如上所述，计算系统 100 包括可用于保持对二维树状结构所作出的修改的持久存储器 170。例如，回头参考图 4A-4C，当子部分 411 被折叠成 UI 元素 460 时，该修改可被保持在持久存储器中。如果用户要退出设计器工具 / 应用程序 120，则该修改将不会丢失。稍后，当该用户重新激活设计器工具 / 应用程序 120 时，将如图 4B 所示地显示树状图 400。

[0081] 类似地，如果用户在退出设计器工具 / 应用程序 120 之前将子部分 430 折叠成 UI 元素 461，则对子部分 430 作出的修改以及可被隐藏的对子部分 411 作出的修改被保持在持久存储器 170 中。稍后，当该用户重新激活设计器工具 / 应用程序 120 时，将如图 4C 所示显示树状图 400。有利地，由 UI 元素 460 表示的修改，无论隐藏与否，都被保留以使得如果用户展开 UI 元素 461 则将如图 4B 所示地显示 UI 元素 460。如果 UI 元素 460 然后被展开，则将如图 4A 所示地显示树状结构 400。因此，甚至在退出运行该树状结构的应用程序时也持久存储隐藏和非隐藏的修改的能力给予用户修改而不必跟踪在退出该应用程序时所作出的所有修改的能力。这对于包括几百个嵌套子部分的树状结构而言尤其有用。如果该修改没有被持久存储，则用户将不得不在每次应用程序启动时重做修改。

[0082] 本发明还可按照包括功能性步骤和 / 或非功能性动作的方法来描述。下文是可被执行以实践本发明的步骤和 / 或动作的描述。通常，功能性步骤按照可以实现的结果来描述本发明，而非功能性动作描述实现特定结果的更为具体的动作。虽然功能性步骤和 / 或

非功能性动作可以按特定次序描述或声明，但是本发明不一定要受限于步骤和 / 或动作的任何特定次序或组合。此外，在权利要求的叙述中以及在以下对图 6-9 的流程图的描述中使用步骤和 / 或动作以指示对这些术语的期望的特定使用。

[0083] 如上所述，图 6-9 示出了用于本发明的各示例性实施例的流程图。以下对图 6-9 的描述将不时参考图 1-4 中的对应元素。虽然可以做出对这些附图中特定元素的参考，但是这些参考只用于说明性目的，而非旨在限制或以其他方式缩小所述实施例的范围，除非明确声明。

[0084] 首先转向图 6，示出了方法 600 的流程图，该方法用于将图状视觉语言的流程图动态地布局成允许易于用户交互与最优显示的二维树状结构。方法 600 包括显示 601 包括至少一个分支的图形树状结构，该分支从该树状结构的根开始并且包括多个子部分，其中子部分包括至少一个端部分。该子部分按照一个或多个对象、属性、操作和关联的功能关系和内部行为来对系统、服务或过程建模。

[0085] 例如，设计器工具 / 应用程序 120 的布局 / 呈现模块 130 可以显示二维树状结构 190。树状结构 200、300 和 400 是满足以上定义的二维树状结构的示例。

[0086] 方法 600 还包括接收 602 修改该图形树状结构中除了至少一个端部分之外的一个或多个子部分的用户交互。例如，设计器工具 / 应用程序 120 的修改模块 140 可以接收修改树状结构 190 的子部分的用户输入 180。在某些实施例中，用户交互可以选择一个或多个子部分以供缩放、折叠或展开。这可以通过激活与将被修改的子部分相关联的 UI 元素来完成。

[0087] 在某些实施例中，设计器工具 / 应用程序 120 的修改模块 140 也接收指定树状结构 190 将如何被修改和显示的一个或多个预定义规则 160。基于该规则和接收到的用户交互，动态地修改树状结构 190。

[0088] 方法 600 还包括基于该用户交互，通过缩放、展开或折叠中的一个来动态地修改 603 除了端部分之外的一个或多个子部分以便于用户交互、优化显示、或两者而无需修改端部分。例如，设计器工具 / 应用程序 120 的修改模块 140 可以动态地修改树状结构 190，其然后可由布局模块 130 来优化以供显示。可以垂直地、水平地、对角地、或以其任何组合修改子部分。

[0089] 例如，在某些实施例中，可以垂直地、水平地、对角地、或以其任何组合折叠一个或多个子部分而无需折叠端部分，如图 3B 所示的。在这些实施例中，折叠可包括将子部分中的至少一个自动格式化为诸如以上所描述的 UI 元素等单个图形对象。该单个图形对象一般将被配置成可在接收到附加用户交互之后展开回到该至少一个子部分。该附加用户交互可以与用户展开该 UI 元素一样简单。

[0090] 在其他实施例中，具有耦合到第一子部分的至少一个节点的第二子部分可被动态地缩放、折叠或展开。该第一子部分然后可在该第二子部分后被动态地修改而无需修改端部分。这允许如图 4A-4C 所示的树状数据结构中的隐藏修改。

[0091] 在其他实施例中，也可垂直地、水平地、或以其任何组合展开一个或多个子部分而无需展开端部分。在这些实施例中，展开可包括将诸如 UI 元素等可展开图形对象自动格式化为至少一个子部分。

[0092] 在另外的实施例中，动态修改可包括将新子部分添加到树状结构。例如，包括形状

314 的子部分 362 被添加到二维树状结构 300。该新子部分可通过选择树状结构中的所需位置并且使设计器工具 / 应用程序 120 自动定位该新子部分同时还自动平衡该树状结构来添加。

[0093] 现在转向图 7，示出了方法 700 的流程图，该方法用于将对视觉树状结构的一个或多个子分支作出的修改持久保留在存储器中以使得对该树状结构的显示所作出的改变甚至在用户退出用于修改该树状结构的应用程序时也被保持。方法 700 包括激活 701 显示树状结构的应用程序，该树状结构包括从该树状结构的根开始并且包括多个子分支的至少一个分支。这些子分支用于按照一个或多个对象、属性、操作和关联的功能关系和内部行为来对系统、服务或过程建模。

[0094] 例如，可激活设计器工具 / 应用程序 120 以显示树状结构 190，其示例在图 2-4 中示出。在某些实施例中，树状结构也可以是一维树状结构。

[0095] 方法 700 还包括接收 702 修改多个子分支中的至少一个的用户输入。例如，修改模块 140 接收修改树状结构 190 的一个或多个子分支的用户输入 140。在某些实施例中，该用户输入可以选择展开或折叠该子分支的 UI 元素。修改可包括缩放、折叠和扩展正被修改的子分支。

[0096] 方法 700 还包括基于该用户输入，在持久存储器中保持 703 对该子分支所作出的修改以使得在用户已退出该应用程序之后的稍后时间重新激活该应用程序时，该树状结构可与其在应用程序被退出时所出现的完全一样地显示。例如，对树状结构 190 所作出的修改可以被保持在持久存储器 170 中。当设计器工具 / 应用程序 120 退出并然后稍后被重新激活时，树状结构 190 的显示在设计器工具 / 应用程序 120 退出时再一次被显示。

[0097] 在某些实施例中，接收修改具有耦合到第一子分支的节点的第二子分支的用户输入。该第二子分支被修改。在该修改之后，修改第一子分支以使得第二子分支的修改被隐藏。第二子分支的隐藏的已修改状态然后可以被保存在持久存储器中。当该应用程序稍后运行时，诸如选择 UI 元素的用户输入的接收允许该隐藏的第二子分支被显示。

[0098] 例如，参考图 4A-4C，子部分 411 可被折叠成 UI 元素 411。在这之后，如以上所解释的，子部分 430 可被折叠成隐藏 UI 元素 411 的 UI 元素 461。将包括 UI 元素 461 的树状结构 400 保存在持久存储器 170 中允许如上所述当应用程序 120 被重新激活时恢复子部分 411。

[0099] 参考图 8，示出了用于将视觉树状结构转换为二维树状结构的方法 800。方法 800 包括接收 801 包括不能够被修改的多个子部分的第一树状结构或图状视觉语言。在替换方案中，接收第二树状结构，该第二树状结构包括从该树状结构的根开始并且包括多个子部分的至少一个分支。这些子部分按照一个或多个对象、属性、操作和关联的功能关系和内部行为来对系统、服务或过程建模。这些子部分包括只要一个或多个子部分被修改就被修改的至少一个端部分。

[0100] 例如，设计器工具 / 应用程序 120 的布局模块 130 可以接收图状视觉语言 110，该图状视觉语言可以是不能够被修改的预先存在的树状结构、只能够在端部分被修改的情况下被修改的预先存在的一维树状结构、或预先存在的任意图状视觉语言。图状视觉语言 110 也可以是如上所述由用户创建的这三个中的任一个。该任意图状视觉语言可包括强调在正在建模的系统、服务或过程中通常必须有什么东西的结构图；强调在正在建模的系统、服务

或过程中必须发生什么事情的行为图；以及强调在正在建模的系统、服务或过程中的东西之间的控制和数据流的交互图。它还可包括各种示图类型的统一建模语言 (UML)、数据库领域中的实体关系图、以及用于业务流程设计的业务流程图。

[0101] 方法 800 还包括将接收到的第一或第二树状结构或任意图状视觉语言转换 802 为二维树状结构。二维树状结构包括至少一个分支，该分支从该二维树状结构的根开始并且包括包含端部分的多个子部分，其中除了端部分之外的子部分可通过缩放、折叠或展开来修改而无需也修改该端部分。

[0102] 例如，转换模块 150 可以将接收到的树状结构或图状视觉语言转换为二维树状结构 190。如上所述，可垂直地、水平地、对角地、或用三者的任何组合来修改该二维树状结构。已转换的二维树状结构也允许先前对二维树状结构所描述的与此处所描述的其他方法和实施例相关的所有功能。

[0103] 现在转向图 9，示出了方法 900 的流程图，该方法用于将图状视觉语言的示图动态地布局成允许易于用户交互与最优显示的二维树状结构。方法 900 包括显示 901 包括至少一个分支的图形树状结构，该分支从该图形树状结构的根开始并且包括被配置成在垂直方向和水平方向上修改的多个子部分。这些子部分按照一个或多个对象、属性、操作和关联的功能关系和内部行为来对系统、服务或过程建模。子部分也可被配置成对角地修改。

[0104] 例如，设计器工具 / 应用程序 120 的布局 / 呈现模块 130 可以显示二维树状结构 190。树状结构 200、300、和 400 是满足以上定义的二维树状结构的示例。

[0105] 方法 900 还包括接收 902 修改除了该图形树状结构中至少一个端部分之外的一个或多个子部分的用户交互。例如，设计器工具 / 应用程序 120 的修改模块 140 可以接收修改树状结构 190 的子部分的用户输入 180。在某些实施例中，用户交互可以选择一个或多个子部分以供垂直地、水平地以及可能对角地缩放、折叠或展开。这可以通过激活与将被修改的子部分相关联的 UI 元素来完成。

[0106] 方法 900 还包括基于该用户交互，水平地、垂直地、或两者兼有地动态修改 903，以允许易于用户交互、以及显示优化、或两者。例如，设计器工具 / 应用程序 120 的修改模块 140 可以动态地修改树状结构 190，其然后可由布局模块 130 来优化以供显示。子部分可以被缩放、展开或折叠。

[0107] 在其他实施例中，具有耦合到第一子部分的至少一个节点的第二子部分可被动态地修改。第一子部分然后可以在第二子部分之后以隐藏对该第二子部分的修改的方式来动态地修改。这允许如图 4A-4C 所示的树状数据结构中的隐藏修改。该隐藏的修改稍后可以在接收到附加用户输入时显示。

[0108] 在其他实施例中，动态修改可包括将新子部分添加到树状结构。例如，包括形状 314 的子部分 362 被添加到二维树状结构 300。该新子部分可通过选择树状结构中的所需位置并且使设计器工具 / 应用程序 120 自动定位该新子部分同时还自动平衡该树状结构来添加。也可实现诸如将多个连接器组合成一个连接器等其他修改技术。

[0109] 在另外的实施例中，优化图形树状结构的显示包括将一个或多个子部分从水平方向自动切换为垂直方向或将一个或多个子部分从垂直方向自动切换为水平方向以针对显示设备或窗口的高宽比进行优化。例如，在某些情况下，图形树状结构的一个或多个分支可能使得该图形树状结构的显示对于显示设备的高宽比并未被优化。例如，该图形树状结构

可能在垂直方向上过长，而显示设备在水平方向上具有未使用的显示空间。在这种情况下，设计器工具 / 应用程序 120 可以将一个或多个垂直子部分自动切换为水平显示。这些子部分可以在诸如当一新元素被添加到需要切换来保持显示优化的图形树状结构时等情况保证的稍后时刻切换回到垂直显示。

[0110] 例如，参考图 3A，在某些实施例中，设计器工具 / 应用程序 120 可以自动切换垂直子部分 360 和 361 的显示以将其显示为从元素 310 水平分叉出以便针对显示设备的高宽比进行优化。以同样的方式，设计器工具 / 应用程序 120 可以自动切换水平子部分 354-356（或这些子部分的任何子集）的显示以在情况保证的情况下将其显示为从元素 305 垂直分叉出。注意，设计器工具 / 应用程序 120 可在必要时自动切换图 3A 和 3B 的元素中的任一个的显示方向以针对显示设备或窗口的高宽比进行优化。

[0111] 本发明可具体化为其它具体形式而不背离其精神或本质特征。所述实施例在所有方面都应被认为仅是说明性而非限制性的。从而，本发明的范围由所附权利要求书而非前述描述指示。落入权利要求书的等效方式的含义和范围内的所有改变应被权利要求书的范围涵盖。

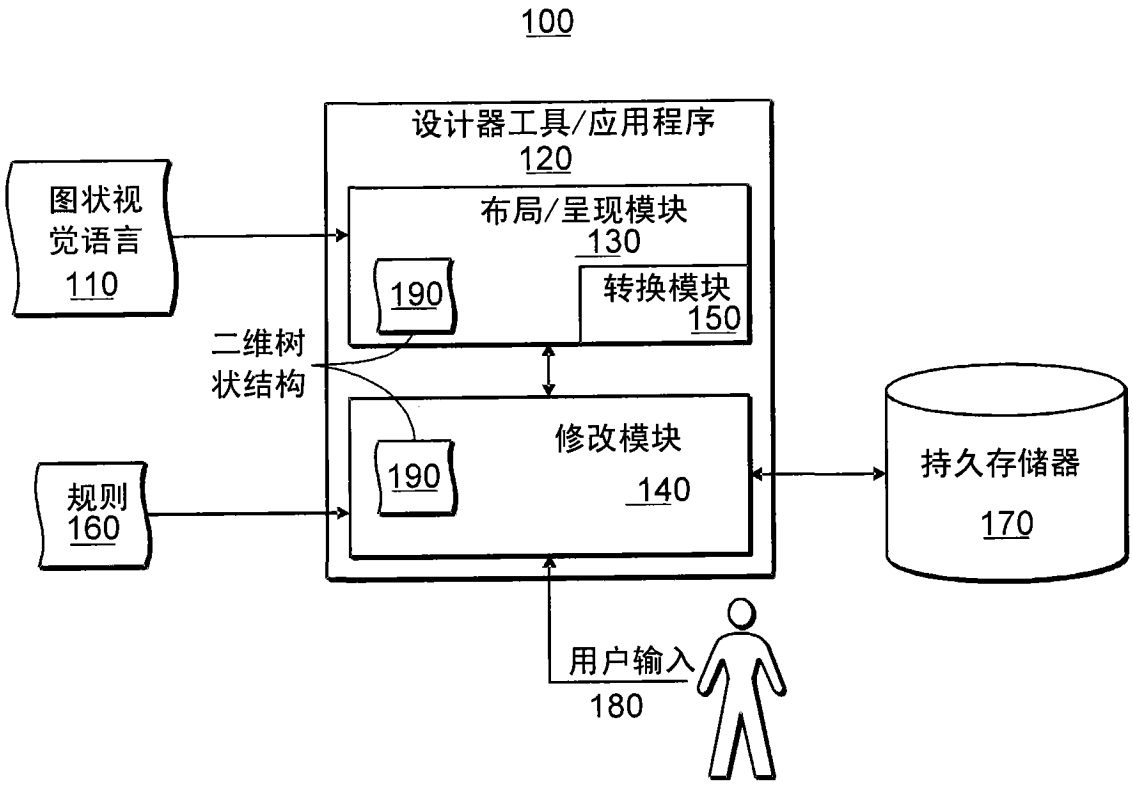

100
设计器工具/应用程序
120
图状视觉语言
110
布局/呈现模块
130
190
转换模块
150
二维树状结构
修改模块
140
190
规则
160
持久存储器
170
用户输入
180

图 1

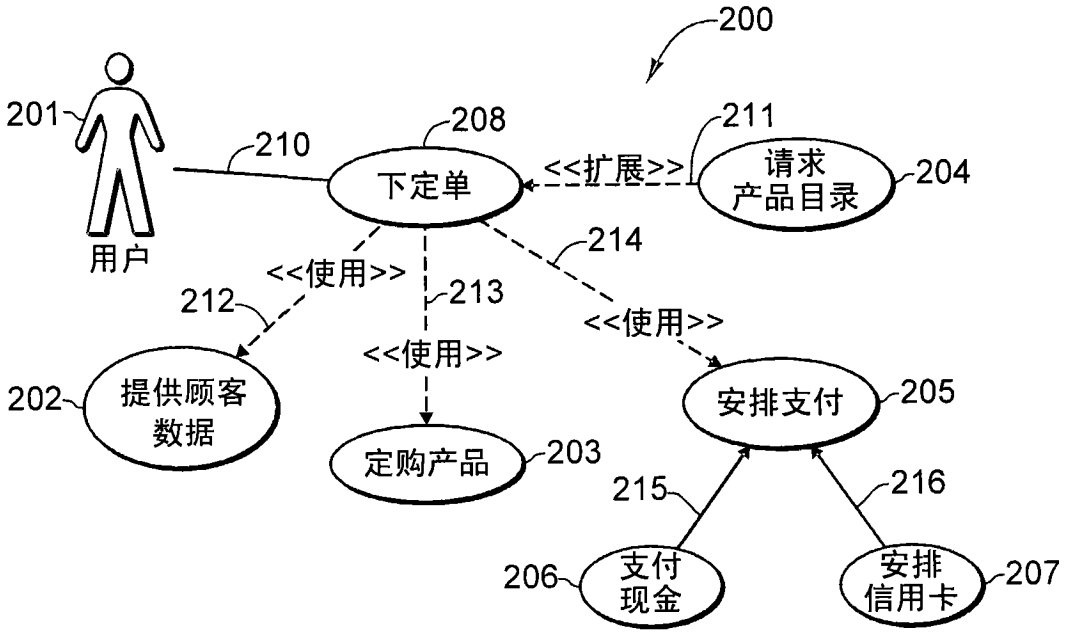

200
201
用户
210
208
下定单
<<扩展>>
211
请求产品目录
204
214
<<使用>>
212
213
<<使用>>
<<使用>>
202
提供顾客数据
定购产品
203
安排支付
205
215
216
206
支付现金
安排信用卡
207

图 2A

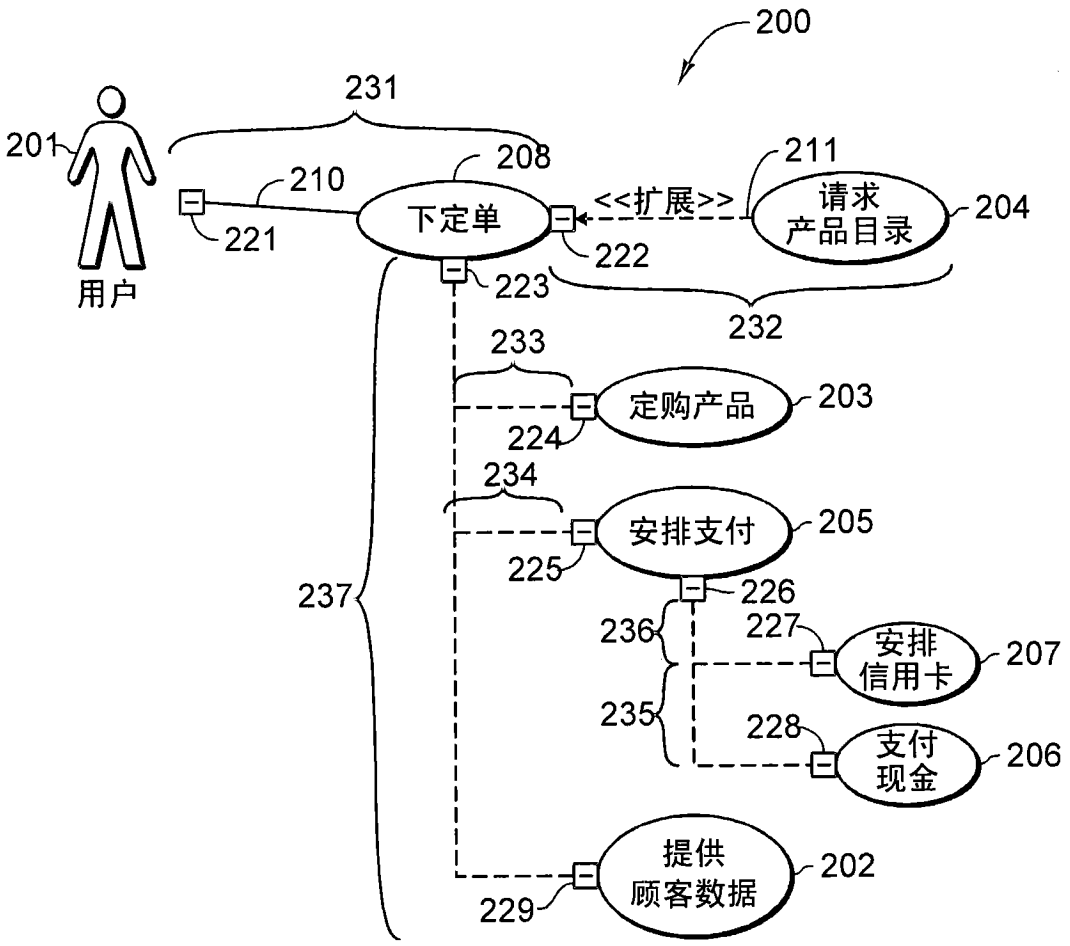

200
231
201
208
211
210
下定单
<<扩展>>
请求
产品目录
204
221
222
用户
223
232
233
定购产品
203
224
234
安排支付
205
225
237
226
236
227
安排
信用卡
207
235
228
支付
现金
206
提供
顾客数据
202
229

图 2B

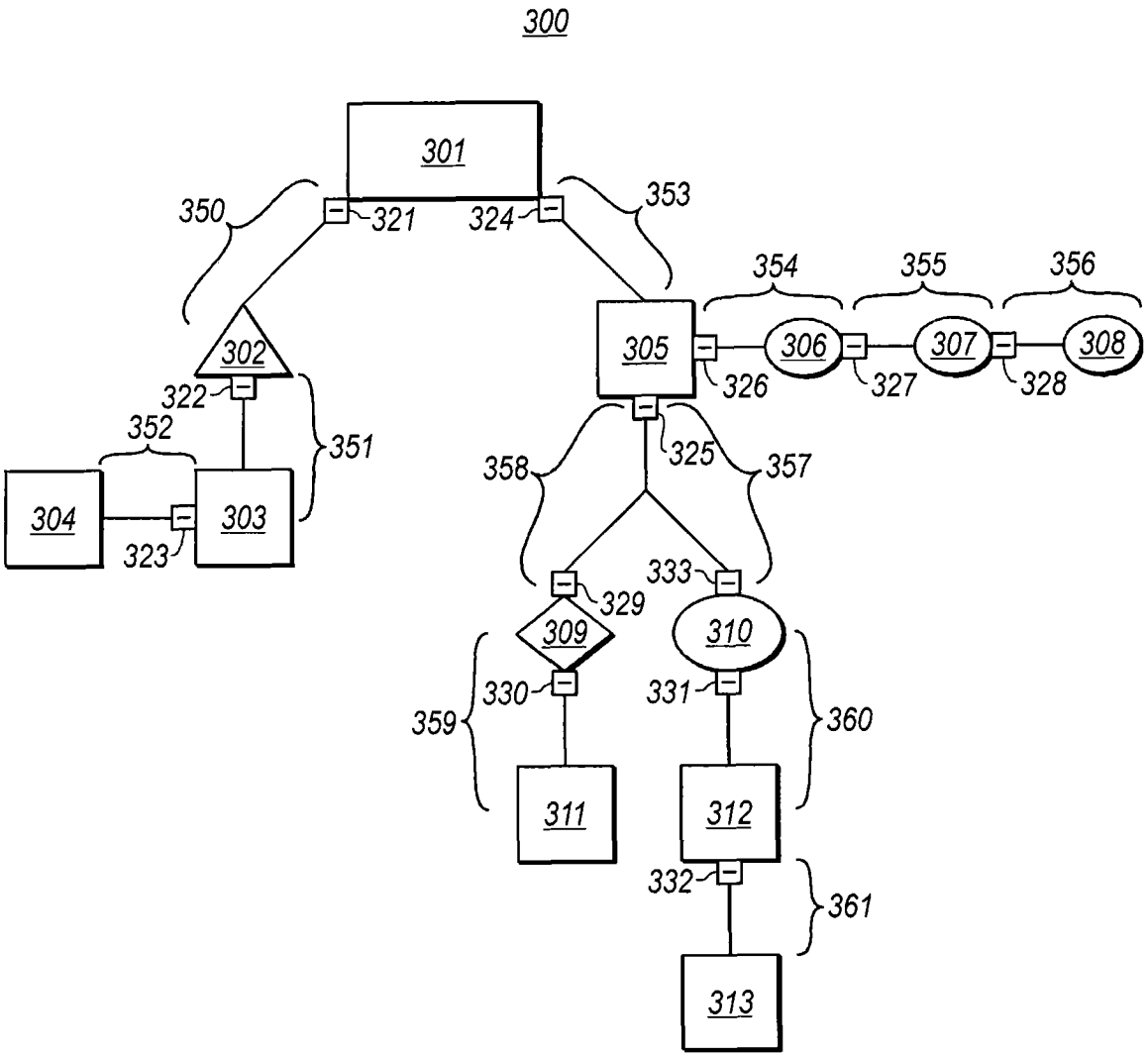

300
301
350
321
324
353
354
355
356
302
305
306
307
308
322
326
327
328
352
351
325
358
357
304
303
323
333
329
309
310
330
331
359
360
311
312
332
361
313

图 3A

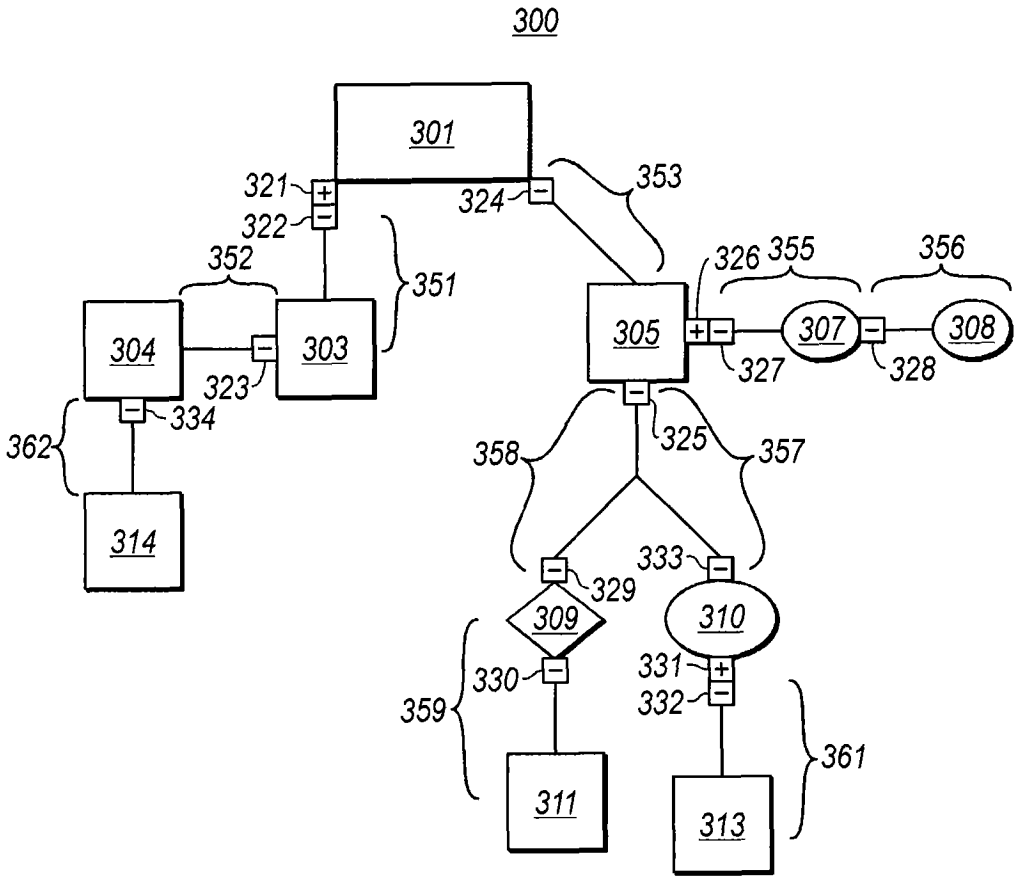

300
301
321
322
324
353
352
351
326
355
356
304
303
305
307
308
323
327
328
334
362
325
358
357
314
333
329
309
310
330
331
332
359
361
311
313

图 3B

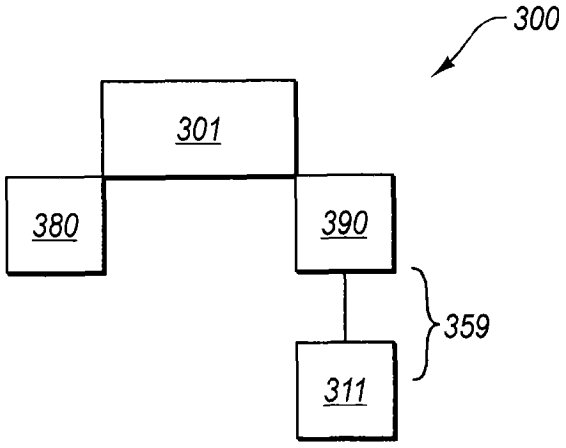

300
301
380
390
359
311

图 3C

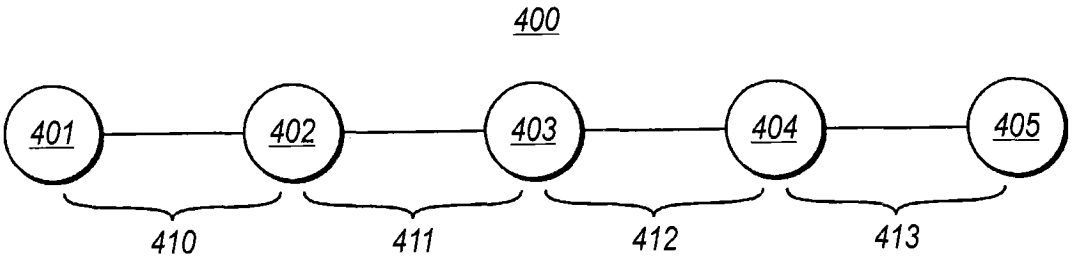


图 4A

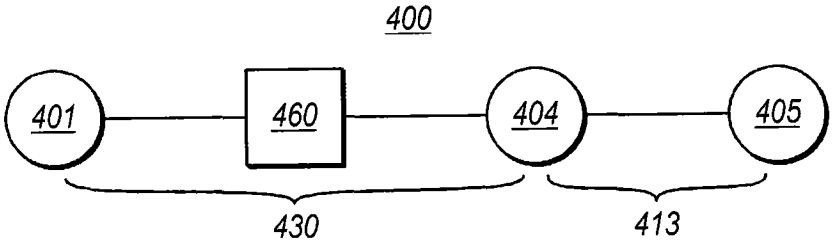


图 4B

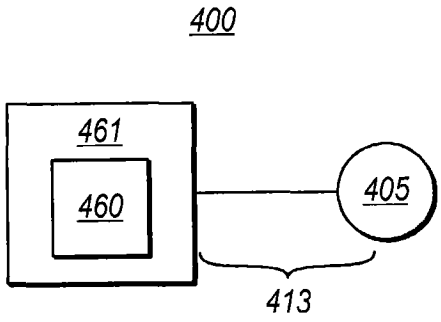


图 4C

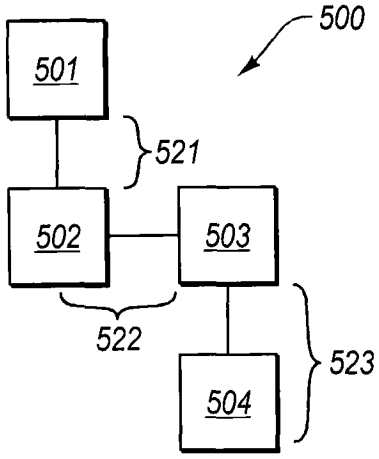


图 5

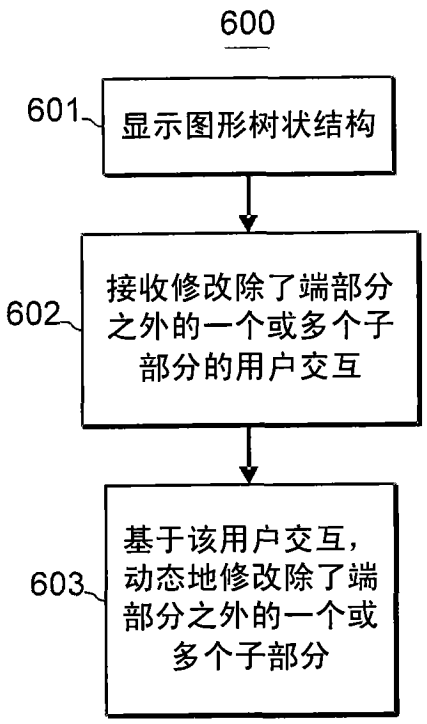

600
601
显示图形树状结构
602
接收修改除了端部分之外的一个或多个子部分的用户交互
603
基于该用户交互，动态地修改除了端部分之外的一个或多个子部分

图 6

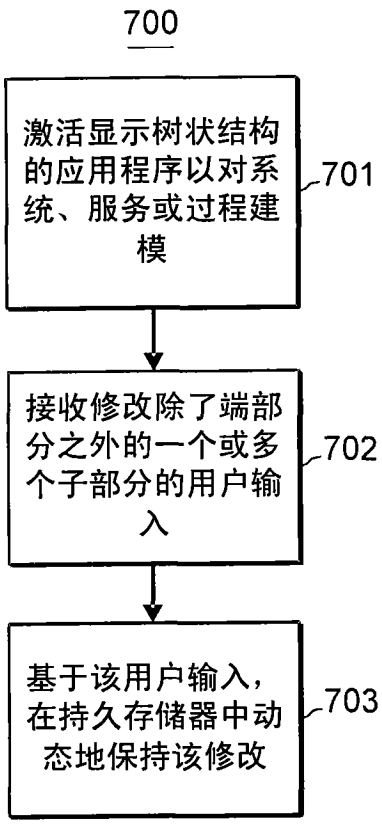

700
激活显示树状结构的应用程序以对系统、服务或过程建模
701
接收修改除了端部分之外的一个或多个子部分的用户输入
702
基于该用户输入，在持久存储器中动态地保持该修改
703

图 7

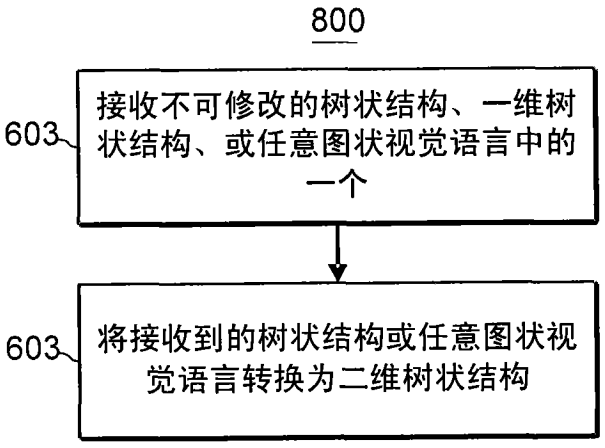

800
603
接收不可修改的树状结构、一维树状结构、或任意图状视觉语言中的一个
603
将接收到的树状结构或任意图状视觉语言转换为二维树状结构

图 8

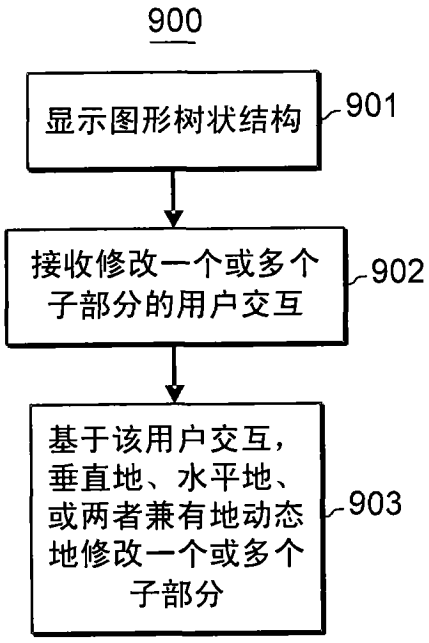

900
显示图形树状结构
901
接收修改一个或多个子部分的用户交互
902
基于该用户交互，垂直地、水平地、或两者兼有地动态地修改一个或多个子部分
903

图 9