发明名称
用于管理祖先树的方法和系统

摘要
公开了一种用于管理祖先树的计算机实施的方法、装置和计算机程序产品。接收制造产品的订单。创建祖先树。祖先树包括一组节点，其中每个节点代表产品的一个部件。接收确定包含特定部件的更高级子部件的请求。在祖先树中定位该特定部件的节点。返回在该特定部件的节点中的祖先指针。祖先指针指示包含该部件的更高级子部件。
1. 一种用于管理祖先树的计算机实施的方法，该计算机实施的方法包括:

响应于接收到制造产品的订单，创建所述祖先树，其中所述祖先树包括一组节点，并且该组节点中的每个节点代表所述产品的一个部件；

响应于接收到确定包含所述产品的特定部件的更高级子配件的请求，在所述祖先树中定位所述特定部件的节点；以及

返回在所述特定部件的节点中的祖先指针，其中所述祖先指针指示包含所述特定部件的更高级子配件。

2. 如权利要求 1 所述的计算机实施的方法，其中该组节点中的每个节点包含指向所述节点的父亲的父亲指针、指向所述节点的孩子的一组孩子指针以及指向包含所述产品的该部件的更高级子配件的祖先指针。

3. 如权利要求 2 所述的计算机实施的方法，其中所述父亲指针、所述祖先指针和所述一组孩子指针中的一一个孩子指针当中至少有一个为空。

4. 如权利要求 1 所述的计算机实施的方法，还包括:

响应于接收到向所述祖先树添加新节点的请求，根据所述请求识别所述新节点的父亲节点；

在所述父亲节点中添加指向所述新节点的孩子指针；以及

在所述新节点中添加指向所述父亲节点的父亲指针。

5. 如权利要求 4 所述的计算机实施的方法，还包括:

根据所述父亲节点中的祖先指针确定更高级祖先；以及

将所述新节点中的祖先指针变为指向所述更高级祖先。

6. 如权利要求 4 所述的计算机实施的方法，还包括:

响应于确定所述新部件是子配件，根据所述新节点中的一组孩子指针确定所述新节点的一个或多个孩子；以及
在所述新节点的一个或多个孩子的每一个中将祖先指针变为所述更高级祖先。

7. 如权利要求1所述的计算机实施的方法，还包括：
相应于接收到从所述祖先树中去除当前节点的请求，根据所述当前节点中的父亲指针对识别父亲节点；
在所述父亲节点中去除指向所述当前节点的孩子指针；
在所述当前节点中去除指向所述父亲节点的父亲指针；以及
在所述当前节点中将所述祖先指针变为指向所述当前节点。
8. 如权利要求7所述的计算机实施的方法，还包括：
相应于确定所述当前节点是子配件，根据所述当前节点中的一组孩子指针识别一个或多个孩子；以及
在所述当前节点的一组孩子指针所指向的一个或多个孩子中，将所述祖先指针变为指向所述当前节点。
9. 如权利要求7所述的计算机实施的方法，还包括：
相应于去除所述孩子指针，将所述孩子指针设为空；以及
相应于去除所述父亲指针，将所述父亲指针设为空。
10. 如权利要求1所述的计算机实施的方法，其中所述更高级子配件是所述祖先树中的最高级子配件。
11. 一种用于管理祖先树的数据处理系统，该数据处理系统包括：
用于接收制造产品的订单的接收装置；
相应于接收到制造产品的订单，用于创建所述祖先树的创建装置，其中所述祖先树包括一组节点，并且该组节点中的每个节点代表所述产品的一个部件；
用于接收确定包含所述产品的特定部件的更高级子配件的请求的接收装置；
相应于接收到确定包含所述产品的特定部件的更高级子配件的请求，在所述祖先树中定位所述特定部件的节点的定位装置；以及
用于返回在所述特定部件的节点中的祖先指针的返回装置，其中
所述祖先指针指示包含所述特定部件的更高级子配件。

12. 如权利要求 11 所述的数据处理系统，其中该组节点中的每个节点包含指向所述节点的父亲的指针，指向所述节点的孩子的一组子指针以及指向包含所述产品的该部件的更高等级子配件的祖先指针。

13. 如权利要求 11 所述的数据处理系统，还包括：

响应于接收到向所述祖先树添加新节点的请求，根据所述请求识别所述新节点的父亲节点的识别装置；以及

用于在所述父亲节点中添加指向所述新节点的孩子指针，亦在所述新节点中添加指向所述父亲节点的孩子指针的添加装置。

14. 如权利要求 13 所述的数据处理系统，还包括：

用于根据所述父亲节点中的祖先指针确定更高等级祖先的识别装置；以及

用于将所述新节点中的祖先指针变为指向所述更高等级祖先的变化装置。

15. 如权利要求 13 所述的数据处理系统，还包括：

响应于确定所述当前节点是子配件，根据所述当前节点中的一组孩子指针识别一个或多个孩子的识别装置；以及

用于在所述当前节点的一组孩子指针所指向的一个或多个孩子中，将所述祖先指针变为指向所述当前节点的改变装置。

16. 如权利要求 11 所述的数据处理系统，还包括：

响应于接收到从所述祖先树中去除当前节点的请求，根据所述当前节点中的父亲指针识别父亲节点的识别装置；

用于在所述当前节点中去除指向所述当前节点的孩子指针，并在所述当前节点中去除指向所述父亲节点的父亲指针的去除装置；以及

用于在所述当前节点中将所述祖先指针变为指向所述当前节点的改变装置。

17. 如权利要求 16 所述的数据处理系统，还包括：

用于确定所述当前节点是否为子配件的确定装置；
响应于确定所述当前节点是子配件，根据所述当前节点中的一组
孩子指针识别一个或多个孩子的识别装置；以及
用于在所述当前节点的一组孩子指针所指向的一个或多个孩子
中，将所述祖先指针变为指向所述当前节点的改变装置。
18. 如权利要求 16 所述的数据处理系统，还包括：
响应于去除所述孩子指针，将所述孩子指针设为空的设置装置；
以及
响应于去除所述父亲指针，将所述父亲指针设为空的设置装置。
19. 如权利要求 11 所述的数据处理系统，其中所述更高级子配
件是所述祖先树中的最高级子配件。
20. 一种包括计算机可用介质的计算机程序产品，该计算机可用
介质包括用于管理祖先树的计算机可用程序代码，该计算机程序产品
包括用于实现根据权利要求 1-10 中的任何一个的方法的步骤的代码。
用于管理祖先树的方法和系统

技术领域

本发明总地涉及数据处理系统，具体而言涉及订单管理系统。更具体地说，本发明涉及用于对包含嵌套部件的子配件进行实时识别的计算机实施的方法、装置和计算机程序产品。

背景技术

现代制造经常涉及多级装配工艺。首先，一组部件（part）被装配在一起形成子配件（sub-assembly）。其次，这些子配件被进一步装配成更复杂的子配件。利用子配件来建立更大的子配件的过程一直持续，直到生产出成品为止。因而，成品可以是由多个嵌套的子配件构成的复杂结构。例如，硬盘驱动器可以是驱动器托盘子配件的一个部件，驱动器托盘子配件可以是抽屉子配件的一个部件，而抽屉子配件可以是完整的服务器的一个部件。

当多个部件被深度嵌套在子配件或成品中时，能够找到包含特定部件的最顶级子配件是很重要的。例如，当一个特定部件被发现是有缺陷的时，能够找到包含该有缺陷部件的所有最高级子部件以便替换有缺陷部件是有用的。

当前，软件应用被用于执行生产后分析以确定哪些产品包含特定部件。生产后分析递归地分析子配件以确定包含特定部件的最顶级产品。由于分析是在生产后而不是实时执行的，因此当部件已被装配到成品中时才执行分析。

发明内容

这里描述的不同实施例提供了一种用于管理祖先树的计算机实施的方法、装置和计算机程序产品。接收制造产品的订单。创建祖先
树。祖先树包括一组节点，其中每个节点代表产品的一个部件。接收确定包含特定部件的更高级子配件的请求。在祖先树中定位特定部件的节点。返回在特定部件的节点中的祖先指针。祖先指针指示包含该部件的更高级子配件。

附图说明
在所附权利要求中给出了被认为是本发明的特性的新颖特征。但是，结合附图参考下面对说明性实施例的详细描述将最好地理解本发明自身及其优选使用方式、其他目的和优点，在附图中：

图 1 示出了根据说明性实施例的数据处理系统的网络的图解表示；
图 2 是可以实现说明性实施例的数据处理系统的框图；
图 3 是根据说明性实施例的祖先树（ancestry tree）的图；
图 4 是根据说明性实施例具有父亲和孩子指针的祖先树的框图；
图 5 是根据说明性实施例包含有缺陷部件的祖先树的框图；
图 6 是根据说明性实施例包含三级部件的祖先树的框图；
图 7 是根据说明性实施例其中一个部件被发现是有缺陷的并且变为孤儿的祖先树的框图；
图 8A 和 8B 一起是根据说明性实施例示出了在一个新的部件被添加到祖先树中之前该新部件的祖先树的框图；
图 9A 和 9B 一起是根据说明性实施例示出了在一个新的部件被添加到祖先树中之后该新部件的祖先树的框图；
图 10 是根据说明性实施例的简化制造体系结构的框图；
图 11 是根据说明性实施例用于创建并维护祖先树的过程的流程图；
图 12 是根据说明性实施例用于从祖先树中去除部件的过程的流程图；以及
图 13 是根据说明性实施例用于向祖先树添加部件的过程的流程图。
具体实施方式

现在参考附图，具体而言参考图1和2，图1和2提供了可以实现说明性实施例的数据处理系统的示例图。应当意识到，图1和2仅是示例性的，而并不是要表明或暗示对可以实现不同实施例的环境的任何限制。可以对图示环境进行许多修改。

图1示出了可以实现说明性实施例的数据处理系统的网络的图解表示。网络数据处理系统100是可以实现实施例的计算机网络。网络数据处理系统100包含网络102，网络102是用于提供在网络数据处理系统100内连接在一起的各种设备和计算机之间的通信链路的介质。网络102可以包括多种连接，例如有线线路、无线通信链路或光缆。

在图示的例子中，服务器104和服务器106与存储单元108一起连接到网络102。另外，客户端110、112和114连接到网络102。客户端110、112和114例如可以是个人计算机或网络计算机。在图示的例子中，服务器104向客户端110、112和114提供数据，例如引导文件、操作系统映像和应用程序。在该例子中，客户端110、112和114是服务器104的客户端。网络数据处理系统100可以包括未示出的其它服务器、客户端和其他设备。

在图示的例子中，网络数据处理系统100是因特网，网络102代表使用传输控制协议/因特网协议（TCP/IP）协议组来相互通信的网络和网关的全球集合。因特网的核心是主要节点或主机之间的高速数据通信线路的主干线，而所述主要节点或主机由路由器数据和消息的千万个商业、政府、教育和其他计算机系统组成。当然，网络数据处理系统100也可以被实现为多种不同类型的网络，例如内联网、局域网（LAN）或广域网（WAN）。图1仅是举例说明，并不是对不同实施例的体系结构的限制。

现在参考图2，图2示出了可以实现说明性实施例的数据处理系统的框图。数据处理系统200是诸如图1中的服务器104或客户端110
之类的计算机的例子，其中可以存储有实施用于说明性实施例的不同处理过程的计算机可用代码或指令。

在图示的例子中，数据处理系统 200 采用集线器体系结构，包括北桥和存储器控制器集线器（MCH）202 以及南桥和输入/输出（I/O）控制器集线器（ICH）204。处理单元 206、主存储器 208 和图形处理器 210 耦合到北桥和存储器控制器集线器 202。处理单元 206 可以包含一个或多个处理器，甚至可以利用一个或多个异构处理器系统实现。图形处理器 210 可以通过例如加速图形端口（AGP）耦合到 MCH。

在图示的例子中，局域网（LAN）适配器 212 耦合到南桥和 I/O 控制器集线器 204，并且音频适配器 216、键盘和鼠标适配器 220、调制解调器 222、只读存储器（ROM）224、通用串行总线（USB）端口和其他通信端口 232、以及 PCI/PCIe 设备 234 通过总线 238 耦合到南桥和 I/O 控制器集线器 204，并且硬盘驱动器（HDD）226 和 CD-ROM 驱动器 230 通过总线 240 耦合到南桥和 I/O 控制器集线器 204。PCI/PCIe 设备例如可以包括以太网适配器、插入（add-in）卡和用于笔记本电脑的 PC 卡。PCI 使用卡总线控制器，而 PCIe 并不使用。ROM 224 可以是例如闪存二进制输入/输出系统（BIOS）。硬盘驱动器 226 和 CD-ROM 驱动器 230 可以使用例如集成驱动电子设备（IDE）或串行高级技术附件（SATA）接口。超级 I/O（SIO）设备 236 可被耦合到南桥和 I/O 控制器集线器 204。

操作系统运行在处理单元 206 上并且对图 2 的数据处理系统 200 中的各种组件进行协调并提供控制。该操作系统可以是商业上可获得的操作系统，比如 Microsoft® 的 Windows® XP。（Microsoft 和 Windows 是在美国、其他国家或两者都有的微软公司的商标）。诸如 Java™ 程序设计系统等面向对象的程序设计系统可与操作系统一同运行并从运行在数据处理系统 200 上的 Java 程序或应用提供对操作系统的调用。（Java 和所有基于 Java 的商标都是在美国、其他国家或两者都有的 Sun Microsystems 公司的商标）。

用于操作系统、面向对象的程序设计系统和应用或程序的指令位
于诸如硬盘驱动器 226 之类的存储设备上，并且可以加载到主存储器 208 中以供处理单元 206 执行。说明性实施例的处理过程可以由处理单元 206 利用计算机实施的指令来执行，计算机实施的指令可以位于诸如主存储器 208、只读存储器 224 之类的存储器中，或者位于一个或多个外围设备中。

图 1 和 2 中的硬件可以根据实施方案而变。可以使用其他内部硬件或外围设备（例如闪存、等效的非易失性存储器或光盘驱动器等）作为对图 1 和 2 中所示的硬件的补充或替换。另外，说明性实施例的处理过程可以应用于多处理器数据处理系统。

在某些说明性的例子中，数据处理系统 200 可以是个人数字助理（PDA），其一般配置有闪存以提供用于存储操作系统文件和/或用户生成数据的非易失性存储器。总线系统可以由一条或多条总线构成，例如系统总线、I/O 总线和 PCI 总线。当然，总线系统可以利用任何类型的通信架构或体系结构实现，这种通信架构或体系结构实现了附接到该架构或体系结构的不同组件或设备之间的数据传输。通信单元可以包括用于发送和接收数据的一个或多个设备，例如调制解调器或网络适配器。存储器可以例如是主存储器 208 或诸如在北桥和存储器控制器集线器 202 中找到的缓存。处理单元可以包括一个或多个处理器或 CPU。图 1 和 2 中的图示的例子以及上述例子并不是要暗示对体系结构的限制。例如，数据处理系统 200 除了采用 PDA 形式以外，也可以是平板计算机、膝上型计算机或电话设备。

现代制造一般涉及这样一种过程，其中多个部件最初被装配在一起形成子配件，这些子配件与附加的部件一起被进一步装配成更复杂的子配件。装配子配件以产生更多子配件的过程持续，直到生产出成品为止。

成品是制造商生产的最高级别的子配件。因而，成品可以包含多个嵌套的子配件。在下面给出的例子中，术语“子配件”应当理解为包括成品。

由一个制造商生产的成品可以被不同制造商用作部件或子配件。
例如，一个制造商可以将各种部件装配在一起产生硬盘驱动器。第二制造商可以将硬盘驱动器用作驱动器托盘子配件的一个部件，而驱动器托盘子配件又是完整的服务器的一个部件。完整的服务器可以是第二制造商的成品，但是又可以被第三制造商用作一个部件来生产另一成品，例如容错事务处理器。

制造商经常使用材料表（BOM）来描述产品的子配件和基本部件。材料表一般用于描述制造商重复制造的成品或子配件。

当制造商在许多不同产品中或者在同一产品的不同位置中使用一个特定的子配件时，制造商通常向该子配件分配唯一的工作单元（work unit）来区分每种使用。例如，服务器在一个架子上可能具有三个硬盘驱动器子配件，一个硬盘驱动器子配件在左边，一个在中间，一个在右边。每个硬盘驱动器子配件具有相同的材料表。但是通过给每个硬盘驱动器子配件分配不同的工作单元而使每个硬盘驱动器子配件相互区分。

当多个部件被深度嵌套在子配件中时，能够实时地找到包含特定部件的最高级别子配件是很重要的。例如，当一个部件已被识别为有缺陷的时，能够快速找到包含有缺陷部件的所有子配件从而可以替换有缺陷部件是很重要的。

在传统系统中，软件应用执行生产后分析以构造祖先树并识别哪些产品包含哪些部件。在生产后分析中，子配件被递归地分析以识别包含特定部件的最高级产品。递归意味着成品被反复分析并分解为部件和子配件。例如，成品可能由多个部件和几个子配件组成。每个子配件随后被进一步分析并分解为部件和子配件，直到成品中的所有子配件都已被分解为组成部件为止。

由于分析是在生产后进行的，因此该分析通常是在部件已被装配到成品或复杂子配件中之后进行的。因此，当在生产后进行分析时，一般而言，需要更多的努力来替换有缺陷部件，这是因为该部件已经结合到更高级的配件或成品中。

本发明的实施例认识到，如果分析是实时进行的，则可以在子配
件被结合到成品中之前较早地在子组件中发现组件。有缺陷组件一般更易于在较小的包含该组件的子组件中替换，而不是在该组件已经结合到几个子组件中并随后结合到成品中之后。

这里描述的不同实施例提供了一种用于管理祖先树的计算机实施的方法。装置和计算机程序产品。接收制造产品的订单。创建祖先树。祖先树包括一组节点，其中每个节点代表产品的组件。接收确定包含特定组件的更高级子组件的请求。在祖先树中定位特定组件的节点。返回在特定组件的节点中的祖先指针。祖先指针指示包含该组件的更高级子组件。

许多子组件通常是由提供商装配并提供的。为了供制造商实时地跟踪祖先树，如果提供商提供一个子组件，则提供商还向制造商提供该子组件的材料表。

祖先树由含有一个或多个节点的一组节点构成，其中每个节点都代表组件。组件可以指不能被进一步分解的个体组件，或者组件可以指子组件。每个节点的数据结构通常包含指向更高级子组件的指针和指向父亲子组件的指针。更高级子组件可以是家庭子组件、祖父子组件或者包含该组件的任何更高级组件。

每个节点的数据结构还可以包含指向该组件的零个或更多个孩子的一组指针。如果一个节点是由其他组件构成的子组件，则该节点的数据结构一般包含一个或多个孩子指针。另一方面，如果一个节点是不能被进一步分解的个体组件，则该节点的数据结构将具有零个孩子指针。

本领域技术人员将意识到，祖先树中每个节点的数据结构可以包含额外的字段和信息。例如，该数据结构可以包含附加信息，例如该组件在何处制造、制造该组件所需的时间或者该组件的其他制造商。

图 3 是根据说明性实施例的祖先树。祖先树 300 图示了成品是如何使用多个子组件和组件产生的。组件 302、304 和 306 被用于产生子组件 308。组件 312、312 和 314 被用于产生子组件 316。子组件 308 和 316 被用作产生子组件 318 的组件。子组件 318 与组件 320 和 322
一起被用于产生子配件 324。
子配件 324 与零件 326、328、330 和 332 一起被用于产生子配件 334。部件 336 和 338 被用于产生子配件 340。子配件 340 与部件 342 相结合以产生子配件 344。子配件 334 和 344 以及部件 346、348 和 350 被用于产生成部件 352。

在该例中，五级嵌套子配件和部件构成了成品 352。部件 302、304、306、310、312 和 314 构成第一级。子配件 308 和 316 构成第二级。子配件 318 与部件 320、322、336 和 338 构成第三级。子配件 324 和 340 与部件 326、328、330、332 和 342 构成第四级。子配件 334 和 344 与部件 346、348 和 350 构成第五级。

这个例子图示了诸如成品 352 之类的产品是如何可以包含多个子配件的，其中每个子配件可以包含附加的子配件。如果处于较低级的部件被识别为有缺陷的，则重要的是识别包含有缺陷部件的所有更高级子配件从而可以替换有缺陷部件。图 3 是可以被构造并用于识别包含特定部件的所有更高级子配件的祖先树的例子。

通过保存一个复杂成品的祖先树（其中根节点是该成品），可以实时地保存每个子配件或部件与根节点之间的关系。当装配计划被首次发送到制造环节时，产品树被初始化。随着产品的制造，在每次添加子配件时，与子配件相对应的子树就被添加到产品的祖先树。

如果发现有缺陷的部件，则识别包含有缺陷部件的子配件。与包含有缺陷部件的子配件相对应的每个子树被从树中去除。与替换子配件相对应的子树被插入到树中，以替换包含有缺陷部件的子配件。向祖先树中添加部件和子树并从祖先树中去除部件和子树的过程在下面的例子中将更详细地描述。

图 4 是根据说明性实施例的具有父节点和孩子节点的祖先树。具有父节点和孩子节点的祖先树 400 图示了诸如图 3 中的子配件 308 之类的子配件是如何利用配件 402、404 和 406 产生的。

在这个例子中，部件 402、404 和 406 被用于产生子配件 408。在这个例子中使用的祖先树数据结构具有四个字段：指向最高级祖先
的指针，详细数据字段、指向父亲的指针和指向所有孩子的（一个或多个）指针。指针是其数值直接标引（“指向”）存储在别处的另一值的数据。在这个例子中，部件 402、404 和 406 是孩子，而父亲是子配件 408。

在部件 402 的数据结构中，字段 410 包含指向最高级祖先的指针，字段 412 包含关于该部件的详细数据，字段 414 包含父亲指针，字段 416 包含（一个或多个）孩子指针。部件 402 不是子配件，不可以被进一步分解为更多的部件。因此，字段 416 具有空指针，指示部件 402 没有孩子。部件 402 是子配件 408 的部件。因此，字段 414 包含指向父亲子配件（部件 408）的指针。部件 408 也是部件 402 的最高级祖先，因此，字段 410 指向部件 408。

在部件 404 的数据结构中，字段 418 包含指向最高级祖先的指针，字段 420 包含关于该部件的详细数据，字段 422 包含父亲指针，字段 424 包含（一个或多个）孩子指针。字段 420 中的详细数据提供了关于该部件的信息，例如部件号、该部件被制造的位置以及可以替换部件 404 的等效部件的列表。本领域技术人员将意识到，在字段 420 中可以提供各种类型的信息，作为对这里给出的例子的补充或替代。

部件 404 不是子配件，因此，字段 424 具有空指针，指示部件 404 没有孩子。部件 404 是子配件 408 的部件。因此，字段 422 包含指向父亲子配件（部件 408）的指针。部件 408 也是部件 404 的最高级祖先，因此，字段 418 指向部件 408。

在部件 406 的数据结构中，字段 426 包含指向最高级祖先的指针，字段 428 包含关于该部件的详细数据，字段 430 包含父亲指针，字段 432 包含（一个或多个）孩子指针。部件 406 不是子配件，因此，字段 432 具有空指针，指示部件 406 没有孩子。部件 406 是子配件 408 的部件。因此，字段 430 包含指向父亲子配件（部件 408）的指针。部件 408 也是部件 406 的最高级祖先，因此，字段 426 指向部件 408。

在部件 408 的数据结构中，字段 434 包含指向最高级祖先的指针，字段 436 包含关于该部件的详细数据，字段 438 包含父亲指针，字段
440 包含（一个或多个）孩子指针。部件 408 是子配件，因此，字段 440 具有指向孩子（部件 402、404 和 406）的三个指针。在装配过程中的这一点处的部件 408 不是子配件的部件。因此，字段 438 包含空指针，指示部件 408 没有父亲子配件。部件 408 也是部件 408 的高级祖先，因此，字段 434 指向部件 408。

图 5 是说明性实施例的包含有缺陷部件的祖先树。包含有缺陷部件的祖先树 500 图示了当一个部件被确定为有缺陷时，诸如下图 3 中的子配件 308 之类的子配件所发生的事情。

在这个例子中，图 4 中的部件 406 假定是有缺陷的，并且被部件 502 替换。这个例子图示了当发现有缺陷部件时祖先指针、父亲指针和孩子指针是如何动态更新的。在图 5 中，部件 402 和 404 与图 4 没有变化。

但是，一旦部件 406 被确定为有缺陷的，部件 406 就变为孤儿，并且指针字段被相应更新。字段 426 被更新使得指向最高级祖先的指针指向部件 406 而不是部件 408。由于部件 406 不再有父亲，因此字段 430 中的父亲指针被设为空。

部件 502 用于替换有缺陷部件 406，并且适当的指针字段被更新。在子配件 408 中，字段 440 中指向部件 406 的孩子指针被替换为指向部件 502 的指针。在部件 502 中，字段 504 被更新以使得指向最高级祖先的指针指向部件 408。字段 506 包含关于部件 502 的详细数据。字段 508 中的父亲指针被更新为指向部件 408。字段 510 中的孩子指针保持为空，因为部件 502 没有孩子部件。

图 6、7、8A、8B、9A 和 9B 一起图示了当一个部件是子配件时在祖先树中是如何替换该部件的。描述祖先树，确定一个部件是有缺陷的，引入新的部件，并且随后利用新的部件来替换祖先树中的有缺陷部件。

图 6 是根据说明性实施例的包含三级部件的祖先树。包含三级部件的祖先树 600 图示了诸如图 3 中的子配件 318 之类的子配件是如何由其他子配件构成的。为了简化，只讨论每个部件的数据结构的相关
字段，但是应当理解，每个数据结构可以具有附加的字段，例如前述的那些字段。

在这个例子中，部件 602 是通过组合部件 604 和部件 606 而构成的子配件。部件 604 和 606 各自也是子配件。部件 604 由部件 608、610 和 612 构成。部件 606 由部件 614、616 和 618 构成。

在部件 602 的数据结构中，最高级祖先指针（字段 620）被设为指向部件 602。由于部件 602 当前并不用作任何其他子配件的部件，因此字段 622 将父亲指针设为空。由于部件 602 具有两个孩子，因此字段 624 将孩子指针设为指向部件 604 和部件 606。

在部件 604 的数据结构中，最高级祖先指针（字段 626）被设为指向部件 602。由于部件 604 是子配件 602 的部件，因此字段 628 将父亲指针设为部件号 602。由于部件 604 具有三个孩子，因此字段 630 具有指向孩子（部件 608、610 和 612）的指针。

在部件 608 的数据结构中，最高级祖先指针（字段 632）被设为指向部件 602。由于部件 608 是子配件 604 的部件，因此字段 634 将父亲指针设为部件号 604。由于部件 608 没有孩子，因此字段 636 为空。

部件 610 和 612 的数据结构中的字段类似于部件 608 的字段。例如，字段 638 和 640 中指向最高级祖先的指针被设为指向部件 602。部件 606、614、616 和 618 的字段分别类似于部件 604、608、610 和 612 的部件字段。由于子配件 602 是图 6 中所有部件的最高级祖先，因此部件 602-618 的祖先指针被设为指向部件 602。部件 608-618 并不由其他部件构成，因此部件 608-618 的孩子指针为空。

如果发现有缺陷部件，则包含有缺陷部件的子配件被从树中去除，并且与替换子配件相对应的子树被插入到树中，以替换包含有缺陷部件的子配件。例如，如果子配件 604 是有缺陷的，则子配件 604 可以被另一子配件替换。

图 7 是根据说明性的实施例，其中一个部件被发现是有缺陷的并且变为孤儿的祖先树。图 7 是当一个部件被确定为有缺陷时如何从祖
先树中去除子配件的例子。

在其中一个部件被发现是有缺陷的并且变为孤儿的祖先树 700 中，确定子配件 604 是有缺陷的。替换有缺陷子配件的第一步是去除子配件 604。当配件 604 被从子配件 602 中去除时，对各个指针进行下面的改变。

在子配件 602 中，指向部件 604 的指针被从字段 624 中去除以指示部件 604 不再是子配件 602 的子配件。在子配件 604 中，字段 628 中的指针变为空以指示子配件 604 不再是子配件 602 的部件。在子配件 604 中，最高级祖先从部件 602 变为部件 604 以指示部件 604 不再在子配件 602 中，因此最高级祖先为配件 604。字段 632、638 和 640 中的最高级祖先指针也发生改变以指示部件 608、610 和 612 的最高级祖先分别是子配件 604。

图 8A 和 8B 一起是根据说明性实施细则的一个祖先树，其中示出了在一个新的部件被添加到祖先树中之前的该新部件。祖先树 800 示出了在新部件被添加到祖先树中之前该新部件和祖先树中的指针。新部件被添加用来替换有缺陷部件。

在这个例子中，新部件是一个子配件，并且有缺陷部件也是子配件。本领域技术人员将意识到，这里描述的技术可以用在新部件是具体部件或子配件时。

在图 8A 中，子配件 802 是代表将替换子配件 602 中的子配件 604 的新部件的数据结构。子配件 802 由部件 804、806 和 808 构成。图 8A 显示了子配件 802 中的指针值。在子配件 802 中，字段 810 中的最高级祖先指针最初被设为指向子配件 802。在被添加到子配件 602 的祖先树中之前，字段 812 中的父亲指针被设为空。字段 814 中的孩子指针指向部件 804、806 和 808。

在部件 804、806 和 808 中，字段 816、822 和 828 中的最高级祖先指针被设为指向子配件 802。字段 818、824 和 830 中的父亲指针也被设为指向子配件 802。在部件 804、806 和 808 中，字段 820、826 和 832 中的孩子指针分别被设为空。图 8B 表明子配件 604 是孤儿并
且不再是子配件 602 的孩子。

图 9A 和 9B 一起是根据说明性实施例的一个祖先树，其中示出了在一个新的部件被添加到祖先树中之后的该新部件。祖先树 900 示出了在新部件被添加到祖先树中以替换有缺陷部件之后该新部件和祖先树中的指针。

在这个例子中，部件 802 替换子配件 602 的祖先树中的部件 604，并且部件 602、802、804、806 和 808 中的指针字段按如下方式更新。在图 9A 中，字段 812 中的父亲指针被从空变为指向部件 602，这是因为部件 802 现在是子配件 602 的部件。祖先字段 810、816、822 和 828 被更新以指示部件 602 现在是最高级祖先。

在图 9B 中，字段 624 中的孩子指针通过添加指向部件 802 的指针来进行修改，这是因为部件 802 现在是子配件 602 的部件。被确定为有缺陷的子配件 604 仍然是子配件 602 的孤儿。

这样就构造了祖先树，其中该树的每个节点代表一个部件。部件可以是个体组件或子配件。每个节点是包含有关该节点的信息以及有关该节点与其他节点的关系的信息的数据结构。

每个节点可以包含不同类型的指针，例如祖先指针、父亲指针和指向孩子节点的指针。节点中的祖先指针指向最高级子配件，其中该节点是最高级子配件的一个部件。如果该节点是子配件的一个部件，则该节点包含指向父配件的指针。如果该节点是子配件，则该节点包含指向构成子配件的每个子配件的指针。关于该节点的附加信息也可以被包含在该节点的数据结构中。

对祖先树中的节点可以执行各种操作。可以创建新的节点，可以破坏现有的节点。节点与该节点的孩子一起可以被添加到祖先树中作为另一节点的孩子。作为另一节点的孩子的节点可以变为孤儿，使得该节点不再是任何节点的孩子。

当对祖先树中的节点执行操作时，祖先指针、父亲指针或者（一个或多个）子配件中中的至少一个被修改。具体而言，每个节点总是知道该节点所属的最高级部件，从而每个部件的最高级祖先可以被快
速识别。无论何时，每个节点都只有单个祖先节点。

图 10 示出了根据说明性实施例的简化的制造体系结构。简化的制造体系结构 1000 图示了制造环境中的基本组成部分，并且示出了创建并维护祖先树的软件可以被放置在制造环境中的何处。

在这个例子中，客户 1002、1004 和 1006 分别发出订单 1008、1010 和 1212。订单 1008、1010 和 1212 被订单管理系统（OMS）1014 接收。订单管理系统 1014 将订单发送到制造资源规划（MRP）或企业资源规划（ERP）系统 1016 中。制造资源规划系统 1016 连接到其他系统，包括采购 1018、库存管理 1020、运输 1022 和车间控制器 1024。

车间控制器 1024 一般是被用于管理制造产品的各种设备的软件和硬件。车间控制器 1024 可以通过网络（例如网络 1026）连接。网络 1026 可以是有线网络、无线网络或同时使用有线和无线技术提供连接到网络 1026 的各种网络组件之间的通信的网络。例如，网络 1026 可以使用诸如铜双绞线、以太网、通用仪器总线（GPIB）和光纤之类的有线技术以及诸如 IEEE 802.11 和蓝牙之类的无线技术。

各种工作站连接到网络 1026 以控制制造设备。在这个例子中，工作站 1028、1030 和 1032 连接到网络 1026。工作站 1028 控制设备 1034，工作站 1030 控制设备 1036，工作站 1032 控制设备 1038 和 1040。

本领域技术人员将意识到，图 10 中所示的工作站和设备的数目仅是用于说明目的。在制造环境中，多个工作站可以附接到网络 1026，并且每个工作站可以控制多个设备。

工作站 1042 附接到车间控制器 1024。软件 1044 运行在工作站 1042 上。软件 1044 利用车间控制器 1024 创建并维护制造的货物的祖先树。在这个例子中，工作站 1042 被示为附接到车间控制器 1024。但是，在另一实施例中，工作站 1042 可以利用网络 1026 连接到制造设备。

图 11 是根据说明性实施例用于创建并维护祖先树的过程的流程图。图 11 中所述的过程由软件过程（例如图 10 中的软件 1044）执行。该过程开始于接收到制造产品的订单时（步骤 1102）。该订单
可以是例如订单配置（CTO）或订单生产（BTP）。
创建并初始化祖先树（步骤 1104）。可选地，如果用于生产产品的子配件由提供商提供，则该过程从提供商接收关于子配件的信息（步骤 1106）。提供商提供关于子配件的足够信息以构造子配件的祖先树，例如关于各个组件的信息，以及各个组件之间的父亲和孩子关系。

该过程接收关于用于构成产品的一个部件已被确定为是有缺陷的信息（步骤 1108）。该过程使用包含在祖先树的每个节点的数据结构中的祖先指针来识别包含有缺陷部件的所有最高级祖先部件（步骤 1110）。该过程从祖先树中去除包含有缺陷部件的最高级祖先部件（步骤 1112）。在祖先树中，对于包含有缺陷部件的最高级祖先部件，更新指向最高级祖先、父亲和孩子的指针（步骤 1114）。

对于包含有缺陷部件的每个最高级祖先部件识别替换部件（步骤 1116）。将一个或多个替换部件添加到祖先树中（步骤 1118）。在祖先树中更新指向最高级祖先、父亲和孩子的指针（步骤 1120），然后该过程结束。

图 12 是根据说明性实施例用于从祖先树中去除部件的过程的流程图。图 12 中所述的过程由软件过程（例如图 10 中的软件 1044）执行。

该过程开始于接收到从祖先树中去除部件的请求时（步骤 1202）。根据部件中的父亲指针识别该部件的父亲节点（步骤 1204）。在父亲节点中，去除指向该部件的孩子指针（步骤 1206）。在该部件中，去除指向父亲节点的父亲指针（步骤 1208）。在该部件中，祖先指针被设为指向该部件（步骤 1210）。

如果该部件具有孩子指针，则在每个孩子中，将祖先指针变为指向该部件的指针（步骤 1212），然后该过程结束。步骤 1212 是可选的，并且只在该部件具有孩子指针时执行。

图 13 是根据说明性实施例用于向祖先树添加部件的过程的流程图。图 13 中所述的过程由软件过程（例如图 10 中的软件 1044）执行。

该过程开始于接收到向祖先树添加部件的请求时（步骤 1302）。
一般根据请求中的信息识别将成为该部件的父亲节点的节点（步骤1304）。在父亲节点中，添加指向该部件的孩子指针（步骤1306）。在该部件中，添加指向父亲节点的父亲指针（步骤1308）。

利用父亲节点中的祖先指针识别父亲节点的最高级祖先（步骤1310）。在该部件中，祖先指针被变为先前识别的最高级祖先（步骤1312）。

如果该组件具有孩子指针，则每个孩子中的祖先指针被设为指向最高级祖先的指针（步骤1314），然后该过程结束。步骤1314 是可选的，并且只在该部件具有孩子指针时执行。

不同的实施例提供了一种用于管理祖先树的计算机实施的方法、装置和计算机程序产品。接收制造产品的订单。创建祖先树。祖先树包括一组节点，其中每个节点代表产品的一个部件。接收确定包含特定部件的最高级子部件的请求。在祖先树中定位特定部件的节点。返回在特定部件的节点中的祖先指针。该祖先指针指示包含该部件的更高级子部件。

图中的流程图和框图图示了根据各种实施例的系统、方法和计算机程序产品的某些可能实现方式的体系结构、功能和操作。为此，流程图或框图中的每个块可以代表一个模块、代码段或代码的一部分，其包括用于实现指定逻辑功能的一个或多个可执行指令。还应当注意到，在某些替换实现方式中，块中所示的功能可以在没有图中所示的订单的情况下发生。例如，被示为连续的两个块事实上可以基本同时执行，或者这些块取决于涉及的功能有时可以按相反顺序执行。

本发明可以采用完全硬件的实施方式、完全软件的实施方式或包含硬件和软件成分的实施方式。在优选实施例中，本发明以软件实现，其包括但不限于固件、常驻软件、微代码等。

此外，本发明可以采用计算机程序产品的形式，该计算机程序产品可以从提供程序代码供计算机或任何指令执行系统使用或与之结合使用的计算机可用介质或计算机可读介质来访问。对于本说明书，计算机可用介质或计算机可读介质可以是可包含、存储、传送、传播
或输送程序供计算机或任何指令执行系统、装置或设备使用或与之结合使用的任何装备。

介质可以是电的、磁的、光的、电磁的、红外或半导体系统(或设备或器件)或传播媒体。计算机可读介质的例子包括半导体或固态存储器、磁带、可拆卸的计算机硬盘、随机访问存储器(RAM)、只读存储器(ROM)、硬磁盘和光盘。光盘的当前的例子包括压缩盘-只读存储器(CD-ROM)、压缩盘-读/写(CD-R/W)和DVD。

适于存储和/或执行程序代码的数据处理系统将包括至少一个通过系统总线直接或间接耦合到存储器单元的处理器。存储器单元可包括在程序代码执行期间所使用的局部存储器、大容量存储装置和缓存，缓存提供至少某些程序代码的临时存储，以使减小代码在执行期间必须从大容量存储装置取回的次数。

输入/输出或者说 I/O 设备(包括但不限于键盘、显示器、指针设备等等)可直接或通过居中的 I/O 控制器被耦合到系统。

网络适配器也可以被耦合到系统，使得数据处理系统能够通过居中的专用网或公共网被耦合到其它数据处理系统或远程打印机或存储装置。调制解调器、有线调制解调器和以太网卡仅仅是几种当前可用类型的网络适配器。

本发明的描述是为了解释和说明的目的而给出的，不希望是穷举的或限于所公开的形式的发明。许多修改和变型对于本领域技术人员是显而易见的。选择和描述一些实施例是为了最好地解释本发明的原理和实际应用，并使本领域技术人员能够理解本发明的多种实施方式，这些实施方式为适于所设想的具体使用而作出多种修改。
图8A