[19] 中华人民共和国国家知识产权局

[51] Int. Cl.
G06F 3/048 (2006.01)

[12] 发明专利申请公布说明书

[21] 申请号 200480021303.6

[43] 公开日 2007 年 9 月 19 日

[11] 公开号 CN 101040249A

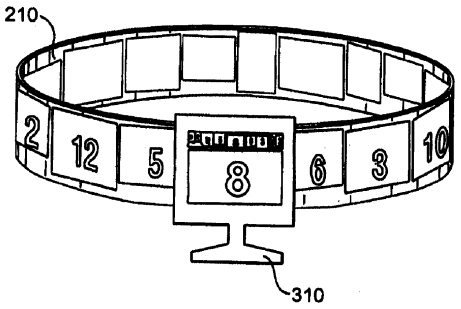
[22] 申请日 2004.6.25
[21] 申请号 200480021303.6
[30] 优先权
[32] 2003. 6.27 [33] US [31] 60/483,304
[86] 国际申请 PCT/US2004/020460 2004.6.25
[87] 国际公布 WO2005/001658 英 2005.1.6
[85] 进入国家阶段日期 2006.1.23
[71] 申请人 索夫特斯扣普有限公司
地址 美国纽约州
[72] 发明人 I·卡茨 E·帕拉特尼克
G·莱德尔曼
[74] 专利代理机构 上海专利商标事务所有限公司
代理人 钱慰民
权利要求书 2 页 说明书 29 页 附图 6 页

[54] 发明名称

虚拟桌面—元－组织和控制系统

[57] 摘要

一种虚拟桌面—元－组织和控制系统，其中具有至少一个动态的基本循环的电子数据结构的可实时访问的存储器媒体(通过分别定期地转变在环境或其预定部分中运行的多个进程中的每个进程的进行中的算法动作)与相关联图形表示相关联，并逻辑地将该表示分配到数据结构中的某一位置上—从而对该数据结构，图形用户界面便于查看分配给至少一个数据结构的表示，并便于组织该至少一个数据结构。



1. 一种在计算机处理环境中使用的虚拟桌面一元-组织和控制系统，所述计算机处理环境具有至少一个具有相应操作系统的处理单元，且所述虚拟桌面系统包括：

A. 在可实时访问的存储器媒体中的至少一个动态的基本循环的电子数据结构；

B. 与每个所述数据结构相关联的进行中的算法动作，所述进行中的算法动作分别定期地：

I. 基于相应的操作系统数据访问，将在所述环境或其预定部分中运行的多个进程中的每个进程转变为相关联的图形表示，以及

II. 逻辑地将该表示分配到数据结构中的位置上；

C. 与每个所述数据结构相关联的图形用户界面，所述图形用户界面便于：

I. 在显示设备上查看分配给至少一个数据结构或其一部分的表示，以及

II. 组织该至少一个数据结构。

2. 如权利要求 1 所述的虚拟桌面一元-组织和控制系统，其特征在于，所述至少一个动态的基本循环的电子数据结构包括具有指向该循环电子数据结构中各位置的指针的分辨率下降的元数据结构。

3. 如权利要求 1 所述的虚拟桌面一元-组织和控制系统，其特征在于，所述进行中的算法动作包括实质上如上所描述和说明的以及从以下列表中选择的至少一个程序：UIManager (UI)、MapManager (MAP)、AnimatorManager (ANIM)、SystemHookManager (SYSHOOK)、ScrollManager (SCROLLER)、可执行代码核心算法组(ECCAG)。

4. 如权利要求 1 所述的虚拟桌面一元-组织和控制系统，其特征在于，图形用户界面包括实质上如上所描述和说明的以及从以下列表中选择的至少一个程序功能：窗口分组、3D 支持、粘滞窗口、多监视器支持、环的紧致化、多环支持、增大的最大窗口大小。

5. 如权利要求 1 所述的虚拟桌面一元-组织和控制系统，其特征在于，相关联的图形表示是从以下列表中选择的：进程的 GUI 的高分辨率快照、进程的 GUI 的低分辨率快照、进程的符号图形显示、进程的 GUI 的高分辨率数据流、进程的 GUI

的低分辨率数据流、进程状态的符号图形表示数据流。

6. 如权利要求 1 所述的虚拟桌面一元-组织和控制系统，其特征在于，所述多个进程是从以下列表中选择的：

A. 从以下的组中选择出的至少两个程序：电子邮件、字处理、流媒体、网络电台、网络电视、网络视频、web 浏览器、聊天室、电子消息传送、图形应用程序包、PowerPoint、建筑支持程序、室内设计支持程序、CAD/CAM、财务处理支持程序、电子制表程序；

B. 从以下的组中选择出的至少两个程序：实时金融数据流演示程序、交易事件确认程序、交易事件程序的综合分析、集体交易管理支持程序、金融分析警告程序、金融分析警报程序、日间交易者交互程序、经纪人管理指示程序；

C. 从以下的组中选择出的至少两个程序：项目管理程序、供应链程序、调度程序、财务处理程序、项目协调程序、资源分配程序；

D. 从以下的组中选择出的至少两个程序：ECG 监控程序、EEG 监控程序、生理监控程序、病史报告程序、药物相互作用程序、医疗专家系统程序、生理监控程序的相关性、医疗条件警告程序、医疗条件警报程序、医疗信息系统程序；

E. 从以下的组中选择出的至少两个程序：染色体组数据碱基序列显示程序、局部查找染色体组片断标识计算程序、与已知有机化合物标识程序的相关性、染色体组计算策略比较程序；

F. 从以下的组中选择出的至少两个程序：作品编排协议、管弦乐编曲程序、电影制作管理程序、动画程序、音响特效程序、视觉特效程序、多媒体播放事件程序、电影编辑程序、音响编辑程序、混音程序、图像序列混合和排序程序。

G. 从以下的组中选择出的至少两个程序：交互式命令控制设备程序、观察监控程序、状态程序的被动查看、警告启动程序、警报启动程序；以及

H. 从任一前述的组中选择出的第一程序、从任一前述的组中选择出的第二程序、以及使来自第一程序的数据内容与来自第二程序的数据内容相互关连的第三程序。

7. 实质上如前所述或所示的虚拟桌面一元-组织和控制系统，其特征在于，具有至少一个循环数据结构，以及与其相关联的小型-映象模块和操作系统界面。

虚拟桌面一元-组织和控制系统

本申请要求对“虚拟桌面（桌面环）”（VIRTUAL DESKTOP (“DeskLoops”)）一2003 年 6 月 27 日提交的美国临时申请序列号 60/483,304 的优先权，该申请通过被引用而结合在此。

技术领域

本发明一般涉及计算机系统的图形用户界面（GUI）；包括通用个人计算机、膝上型计算机、个人数据助理、具有内置计算应用程序的高级无线通信设备、“瘦”客户机前端系统（依赖于基于服务器的 CPU 引擎）、分布式计算系统、并行处理系统、同步和异步的无定形计算团体（例如 SETI-宇宙智慧生物搜索）等的图形用户界面。

更具体地，本发明涉及对在进程中访问多个应用程序的用户的视觉组织和意识促进；所述进程包括独立进程和相关进程一在一些应用程序中有时称为“虚拟桌面”。

从进程中计算机化应用程序本身的角度看，本发明涉及：交互式命令-控制工具和/或观测监控（例如状态的被动查看、警告激活、警报激活等）；包括实时的金融数据流交易交互、项目管理和协作、生理监控和医学信息系统的集成、染色体组计算策略、艺术作品编排协议（例如管弦乐编曲、电影制作管理、多媒体播放事件等）等。

背景技术

今天，从普通计算机用户的角度看，多个基本同时发生的功能的使用仅仅是平凡的日常动作而已。例如，一个具有代表性的用户会打开应用程序用于：电子邮

件、字处理、流媒体（网络电台）、web 浏览器、聊天室（电子消息传送）、图形应用程序包（例如 PowerPoint、或 Architecture（建筑）、或 CAD/CAM）、以及财务处理程序包（例如电子制表软件）。通过在它们之间前后“切换”（running），他分散其注意力并完成越来越多的不同任务。

现在使常规用户能将其焦点从一个应用程序切换到另一个应用程序的 GUI，通常实际上同时将用户转变成顺序任务操作者－尽管用户更喜欢一些人机工程方法来扩展其便利性并参与多个动作。在比该晦暗示例更明显的情形中，我们发现许多管理员盼望能对所有辅助活动和交易进行主动和交互式的监控。然而，向常规用户返回顺序模式的同一“拖走”（plod along）GUI 系统同样地将管理员的视野和活动范围限制于类似的顺序模式。尽管对扩展查看和交互性的持久需要是长期所探索的，并且以可想像的最大代价对专门实例化的军用应用程序进行了开发，但一般用户和普通管理员仍然在继续等待使他们能对各个兴趣点上的同一级别响应性进行命令控制的一般界面。

这并非是不合理的期望，特别是因为（从用户角度而言）运行同步的机载程序进程的复杂性显现为显然类似于将那些进程扩展为包括与计算机通信互连组的其它成员的组件（group-ware）交互；这进而显现为明显类似于包括对任一进程集合的运营、管理和交易促进。此外，用户日益增加地同时参与多种半自动动作，这些动作涉及他的个人生活、他的工作、以及他的可能与其个人和职业生活平行的或可能－以明显危害社会的行为、无道德约束的猎奇、“实验性的”兴趣、和瞬时“调查”的现实化－不当补充的因特网社会信息“环境”。在每个这些正常的公共和专门研究中，用户逐渐发展新技能并进行自学，以便于执行迄今为止通过专门的人工辅助代理来进行的任务和交易，诸如旅行订票、医学了解、日间交易、电子商务、约会、市场调查等。

因而，根据实际上由一般低水平消费者期望所推动的对便利多程序组织和交互性控制的长期需要，没有提供一般的可直观管理的软件系统来用于公用，这从历史上来看是令人惊讶的。然而，一些复杂的尝试被记录成文档并通过引用结合于此，包括：PCT/US96/11765、PCT/US99/08669、PCT/IL00/00504（本发明的发明人之一）、PCT/FI03/00315、US-6,308,199、US-6,687,878 和 PCT/US00/28319。每一个这些现有技术对比文件，尽管尝试构建一些用于计算机活动进程等的通用元-组织

系统，但未能得到一种充分直观的界面格式来使不同背景的普通用户都能获益－其原因是勉强地配置了不便利的人机交互格式（包括 GUI）。此外，没有明显的认知步骤可使（本领域）普通（技术）人员能组合这些对比文件并得到一种便利的方案－相反看起来组合这些对比文件将产生一个更为混乱的用户界面。然而，根据我们的最佳了解，这些是普通的可访问的文献中最为接近的对比文件－并且本发明将展现出（单独地或总体地）优于每个对比文件的合理进步。

共享虚拟桌面综合性应用系统 PCT/US96/11765 (WO 97/04383)的摘要："包括处理器、输入设备和输出设备并运行支持执行的操作系统的计算机系统，用来执行第一和第二应用程序集。操作系统包括可通过输出驱动器耦合到输出设备的图形用户界面，以及包括可通过输入驱动器耦合到输入设备的输入队列的输入界面。处理器还执行环境管理器程序。该程序包括第二应用程序集的第三列表，以及对应于应用程序第二列表的应用程序窗口的第四列表。环境管理器程序的执行提供了将环境管理器程序包括在第一和第二集中，并用操作系统来选择性地交换一三列表和二四列表，以在第一和第二应用程序集的执行之间切换。"分析：显然该实施例看起来需要专门的用户培训和特定的用户智力来变得便利；因而它们看来并未上升到完成本领域长期需要的水平。

用于提供虚拟桌面系统体系结构的方法和装置 PCT/US99/08669 (WO 99/54804)的摘要："本发明提供用于计算的中央办公室比喻（metaphor），其中各个特征和功能由一个或多个服务器提供，并通过网络传送给设备终端。数据提供者被定义为"服务"并由一个或多个处理资源提供。各个服务通过诸如以太网的网络向显示终端通信。各个终端被配置成显示数据，并通过网络向处理服务器发送键盘、光标、音频和视频数据。功能进行划分，从而数据库、服务器和图形用户界面功能由服务提供，而终端提供用户界面功能。从各个服务与各终端的通信通过将不同输出转换成同一协议来完成。向每个服务提供适当的驱动器，以允许协议转换。多个终端与网络耦合。用户可通过将"智能卡"插入读卡器在任一终端上启用其唯一的会话。移除该卡则停止该会话。再将该卡插入同一或任何其它终端可再次启用该会话。"分析：显然该实施例看起来"怀旧地"期望过时较长时间的远程通信体的界面体系结构能符合当今用户的实时复杂性期望－虽然并没有再次发明打孔纸带；但类似地未能上升到完成本领域长期需要的水平。

用于改进虚拟桌面的方法和装置 PCT/IL00/00504 (WO 01/14956)的摘要："一种改进的虚拟桌面涉及一种用于改进人机之间工作界面的质量的方法和装置，以及一种便利的命令控制系统，包括（A）具有至少一种演示格式的计算机工作站，其中存储多个图标对象用于在相关联图形显示设备上显示，并且工作站的动态图标对象管理软件将演示格式图标对象配置成图标对象的虚拟桌面；以及（B）与工作站互连的命令控制装置，用于将控制命令传送给工作站，其中控制命令的至少之一引发动态图标对象管理软件的动作。"分析：在此，像其它现有技术对比文件一样，即使本发明的发明人也未能看到并察觉到简化复杂元-组织所需的简单用户界面和现行计算机进程管理的结合；因而也未能上升到完成本领域长期需要的水平。

图形用户界面和用于在图形用户界面中导航的方法和电子设备 PCT/FI03/00315 (WO 03/091867)的摘要："本发明描述一种方法、图形用户界面和电子设备，用于形成至少包括显示器和导航装置的电子设备的图形用户界面的引导线（guiding line），其中所述图形用户界面的虚拟桌面区域的一部分在同时可在显示器上看到。在该方法中，数字材料被置于虚拟桌面区域上。该方法还包括在数字材料内确定原点，并限定至少两点，通过该两点绘制引导线，引导线表示到原点的距离和/或方向。然后引导线连同数字材料显示在显示器上。"分析：显然该实施例看起来有些察觉到对用于元-组织进程监控和控制的逻辑简单 GUI 的一般需要；然而该特定例示并未回应本领域中的长期需要。

用于管理窗口显示的协作工作支持系统 US-6,308, 199 的摘要："在多个用户可使用延伸到多个应用程序窗口的各条信息之间的关系进行讨论的应用程序共享系统中，提供了为每个用户选择要显示的窗口和要隐藏的窗口的能力。在该系统中，包括在一个计算机中的应用程序由通过网络连接的多个计算机共享，且由该应用程序产生的显示屏幕也得到共享。在具有共享应用程序的计算机中，对每个用户而言要显示还是隐藏在其它计算机共享的屏幕显示期间所显示的窗口由显示控制单元进行控制。提供了用户信息管理单元，它管理表示对每个用户而言是要显示还是要隐藏每个窗口的用户信息。该显示控制单元使用用户信息管理单元中的用户信息来确定对每个用户显示或隐藏。"分析：显然该实施例看起来忽略了没有直观元-组织 GUI 可用的事实－因而将现有的有缺陷的管理界面提高为（跨服务器链接的）大组件体系结构。

用笔记数据库中（组件版本控制协议）其客户机先前所作标注来同步/更新本地客户机笔记的US-6,687,878的摘要："诸如图像或文本文档的文档存储在中央笔记服务器上的笔记数据库中。各文档和关联标注彼此独立地进行处理，从而对文档和关联标注创建独立的数据结构。服务器侧的 web 服务器用来捕捉来自一个或多个笔记客户机应用程序的对创建、存储和检索与存储在笔记服务器上的特定文档相关的标注的请求。在客户机侧，笔记客户机用来显示用户需要标注的文档，并提供允许用户创建、标记、输出、检索和存储标注所必须的工具。同步进程将用户产生的标注从笔记客户机传送到笔记服务器。作为响应，笔记服务器将确认与从上次执行同步起其它笔记客户机已公布的任何新笔录一起传送回去，从而使多个笔记客户机能彼此异步地标注一文档。当标注由笔记客户机分布给笔记服务器时，标注数据库的状态被同步成所有其它笔记客户机能检索与文档相关联的当前最新标注。"分析：显然该实施例看起来察觉到版本控制协议可有助于解决支配组件环境的混乱情况一但未能解决基本的GUI元-组织问题。

知识工程协议序列 PCT/US00/28319 (WO 01/33501)的摘要："提供了一种知识工程协议序列，它通常包括用于搜索-空间组织确认的方法和系统、装置以及所使用的组件。该协议序列包括一种搜索-空间组织确认方法，用于协同组合不同分辨率数据集的知识库，诸如通过较低分辨率的基于专家经验的类模型模板和较高分辨率的根据实验的数据捕捉密集量化搜索-空间。此外，从可选技术优势来看，该序列涉及这样的情形：该协同组合在诸如控制系统、命令控制系统、命令控制通信系统、与前述相关联的计算装置中有利地实现，并涉及所使用的量化建模和测量工具。"分析：显然该实施例看起来察觉到有对元-组织、进程协调和智能控制机制的需要一然而，像所有前述对比文件一样，它们未能找到一种强健的直观GUI，允许普通用户界面有极为复杂的处理环境。

归纳现有技术可发现，富有经验的资金到位的计算机系统用户还有未得到满足的对提供元-组织和适当控制功能的简单GUI界面系统的需要。因而，普通消费用户会期望等待较长时间直到其规范不同的实例能找到便利方案，这看来是合理的。简言之，用户并不关心进程是否在他的机器上或通过某些数据通信协议在其它地方运行一他想要一种能看到在进行什么、能交互他认为适当的内容、并使各进程也能彼此交互的一般直观方法；并且这是未得到逐步实现的长久需要一因为各进程

和拓朴变得更加复杂了。

发明内容

前述长久需要通过本发明各实施例得到实质解决，本发明具体涉及虚拟桌面一元-组织和控制系统。本发明系统在存在对一般直观方式的需要的人机交互中特别有用，其中用户能同时看到多进程环境中所发生的事并按需与这些进程进行最广泛地交互；并且最好使这些进程也能够互相交互。

参看本发明的主要实施例，它们涉及一种在计算机处理环境中使用的虚拟桌面一元-组织和控制系统，该计算机处理环境具有至少一个具有相应操作系统的处理单元，且虚拟桌面系统包括：A. 在可实时访问的存储器媒体中的至少一个动态的基本循环的电子数据结构（参见图 1：210、220、230）；B. 与每个所述数据结构相关联的进行中的算法动作，所述进行中的算法动作分别定期地（I.）基于相应的操作系统数据访问，将在所述环境或其预定部分中运行的多个进程中的每个进程转变为相关联的图形表示（参见图 2：310、312、314、316、318），以及（II.）逻辑地将该表示分配到数据结构中的位置上；C. 与每个所述数据结构相关联的图形用户界面，所述图形用户界面便于（I.）在显示设备（参见图 3：310）上查看分配给至少一个数据结构或其一部分的表示，并（II.）组织（参见图 4：410）该至少一个数据结构。

更具体地，本发明的主要实施例涉及一种应用于具有至少一个处理单元的计算机处理环境的虚拟桌面-组织和控制系统，且该系统包括三个元件 A、B 和 C一立即在后面描述。

A. 在可实时访问的存储器媒体中的至少一个动态的基本循环的电子数据结构一且该结构“基本上”循环是因为应用于极长的数据序列（例如金融数据序列、染色体组序列等）使得形成端对端链接微不足道一当使用本发明来组织进程和图形表示时，循环特征立即提供一种直观的元-组织元件（在“模拟世界”中通过 Flexible Volume Rolodex organizers（可变容量的罗拉代克斯组织者）的示例而众所周知，而在“数字世界”中通过 LISP GUI一从 McCarthy 的列表处理语言中“进化”而来的示例而众所周知）。该数据结构的实现可以是简单的链表；或者甚至是稀疏数组（诸如通过行程编码有效压缩来允许连续部分的呈现），然后（在必要时）通过在

末端设置指回开始处的（以及双向的）指针来人工地使其循环。

B. 与所述数据结构相关联的进行中的算法动作，该进行中的算法动作分别定期地（I.）基于各个操作系统数据访问，将在所述环境或其预定部分中运行的多个进程中的每个进程转变为相关联的图形表示，以及（II.）逻辑地将该表示分配到数据结构中的某一位置上一且转变可以像来自进程的当前GUI显示的快照(snapshot)一样直接，或者转变可以是进程的聚集的元-表示，或者转变可以是可应用于一个本发明循环结构的一个或多个进程之间关系的递归快照，其中一个或多个进程可应用于另一本发明的循环结构（多层实施例）。相应的操作系统访问是本发明实施例通过一些协议互兼容的数据通信拓朴跨越多个计算机的情形；而对于核心技术实施例，只有下层计算机的操作系统本身。

C. 与每个所述数据结构相关联的图形用户界面，该图形用户界面便于（I.）在显示设备上查看分配给至少一个数据结构或其一部分的表示，以及（II.）组织该至少一个数据结构一且该表示可以是静态的写实图像、静态的符号图像、动态的写实图像、条件激活的图像。对于允许音频的计算机处理环境，基本上一些“图像”可以是音频原声摘要播出、音频数据流等。然而组织的目的是允许每个数据结构的表示按照当前优选顺序排序。

现在简言之，从本发明核心技术实施例（在“具体实施例”小节中详述）的用户角度而言，用户显示器的至少一个水平横截面被分配用于演示智能显像。这些显像可以是用户机器上运行的用户应用程序进程，或互连服务器体系结构的用户分配虚拟地址空间中的用户相关进程，或在一些公共或专用的信息多维空间中用户感兴趣的进程，或在与非用户机器互连的服务器上运行的用户感兴趣的应用程序进程，或驻留在任一互连的前述或数据通信进程中的任何数字系统进程。

因为用户是支撑该横截面的基本循环数据结构的组织者，许多便利和直观的操作使用户能（因为结构是循环的所以是无限地）双向地按序查看这些显示，在横截面上的显示和它们分别监控的进程之间切换，重新组织横截面上各个显示的排序等一所有这些都是任务栏、静态菜单、控制代码和定制应用程序集的特定管理软件显像工具上的建议。本发明的便利方面所固有的是一种能力，来重新组织对循环数据结构的各个显示的相对位置，以选择这些表示的哪个子集可在屏幕上查看、将可查看表示用作输入（调用）与关联于表示的进程的交互的快速界面。

本发明最简单的直接应用是针对代表性的同时具有很多在他的机器上运行的进程的个人计算机用户的一诸如电子邮件、字处理、流媒体（网络电台）、web 浏览器、聊天室（电子消息传送）、图形应用程序包（例如 PowerPoint、或 Architecture（建筑）、或 CAD/CAM）、以及财务处理程序包。实际上，当在文本文档上工作时，他可使多个文本文档甚至多个 web 浏览器打开来完成他的文档编写任务。因此，使用本发明的核心技术实施例向用户提供当前静态层叠任务栏的视觉化智能选择。

考虑到根据本发明的应用特定实施例过多，有两种基本情形进程一一种在离散操作程序上运行（在以下“具体实施方式”小节中详述的核心技术实施例），而另一种则在下层的极长静态数据序列或基本上进行中的动态的数据序列上运行。

离散操作程序基本情形

在该变体中，本发明实施例最好将用户应用程序视为具有在基本循环的电子数据结构的图形表示上显示的静态或动态快照的进程。这种静态表示的示例包括 acrobat reader、 Microsoft word、 Norton Anti-Virus、 Microsoft Internet Explorer、Excel for Windows 等，而动态表示的示例包括视频流、RealOne Player、QuickTime Player 等。因此在该基本情形中，“至少一个动态的基本循环的电子数据结构”最好实质上是单个独立结构。

该情形的这种较佳选择的一个有趣例外在于用户保存结构配置一从而他能同时重新启用多个进程并从停止之处继续其动作。例如，用户使预先配置结构具有多个进程，用于个人游戏和娱乐、或个人通信、或商业活动、或组件项目工具和数据共享（通过诸如因特网、LAN、WAN 等的数据通信拓朴）。

该较佳选择的另一有趣例外是将循环数据结构用作参与分组活动一项目开发、学习或重建一的多个用户的共用 C3 引用；且该共用引用通过服务器体系结构组件（因特网等）提供或作为网站的主页 URL一因为显像表示是实质上无数进程（例如网页）的易于重新配置的索引，或递归地作为多个其它基本循环的数据结构的元-组织（循环数据结构的进程等）。

数据序列基本情形

在该变体中，本发明各实施例最好将用户应用程序视为具有在基本循环的电子数据结构的图形表示上显示的预定长度的序列数据或活动数据流的进程。静态表示的示例包括：历史性的计量经济学或统计数据、预定染色体组中的碱基对、音乐

作品中的音符、工业过程中的步骤、受控规划项目中的阶段等；而活动数据流的示例包括股市价格、货币市场价格、来自生理监控或地震监控或天气监控装置的测量等。参看图 14（1400），有六种基本并行的循环数据结构（在该示例中视为具有同步的共用时间标度）。从底部开始，有两个数据序列，具有在它们上面显示的相关函数产生的数据。再上面，是在相关函数产生在预定阈值（或条件）外结果的任何时候的警报流。多个前后紧接的警报将触发警告事件，该事件将允许决策者进行干预。该示例的一个应用涉及底部的股票流处理，它最终导致经理向其下级经纪人发出“干预”特定类型金融交易的指示。该示例的另一应用从两个示例监控开始，以医务人员干预动作结束。在医疗信息系统的情形中，判定可用 SOAP 记录保存格式递归地进行归纳一作为各个基本循环的数据结构上的四个并行“事件”等等。

现在，对于这些类型的数据序列基本情形，有三种类型的本发明实施例提供发明性进步的长期需要一甚至是按照以上列示的现有技术对比文件以及在此列示的引用！第一类是对经改进的数据显像、增量式图形显像、以及地域-型数据一“环境”显像的需要。第二类是对数据密度、细节密度和数据索引分层呈现的经改进的元-组织的需要。第三类涉及对经改进的服务器侧组件（与单个共用环或多个互连环链接）和因特网入口的内容访问组织（使用本发明实施例来格式化“主页”URL 等）的需要。

此外，根据本发明另一实施例的虚拟桌面一元-组织和控制系统，至少一个动态的基本循环的电子数据结构包括具有指向循环电子数据结构中各位置的指针的分辨率下降的（reduced resolution）元数据结构。

类似地，根据本发明又一实施例的虚拟桌面一元-组织和控制系统，进行中的算法动作包括实质上如下描述和说明的并且从以下列表中选择的至少一个程序：UIManager (UI)、 MapManager (MAP)、 AnimatorManager (ANIM)、SystemHook Manager (SYSHOOK)、ScrollManager (SCROLLER)、可执行代码核心算法组 (ECCAG)。

此外，根据本发明的再一实施例的虚拟桌面一元-组织和控制系统，相关联的图形表示是从以下列表中选择的：进程的 GUI 的高分辨率快照、进程的 GUI 的低分辨率快照、进程的符号图形显示、进程的 GUI 的高分辨率数据流、进程的 GUI 的低分辨率数据流、进程状态的符号图形表示数据流。

现在，根据本发明一些极为有趣的实施例的虚拟桌面一元-组织和控制系统，从以下的列表中选择多个进程：

A. 从以下的组中选择出的至少两个程序：电子邮件、字处理、流媒体、网络电台、网络电视、网络视频、web 浏览器、聊天室、电子消息传送、图形应用程序包、PowerPoint、建筑支持程序、室内设计支持程序、CAD/CAM、财务处理支持程序、电子制表程序；

B. 从以下的组中选择出的至少两个程序：实时金融数据流演示程序、交易事件确认程序、交易事件程序的综合分析、集体交易管理支持程序、金融分析警告程序、金融分析警报程序、日间交易者交互程序、经纪人管理指示程序；

C. 从以下的组中选择出的至少两个程序：项目管理程序、供应链程序、调度程序、财务处理程序、项目协调程序、资源分配程序；

D. 从以下的组中选择出的至少两个程序：ECG 监控程序、EEG 监控程序、生理监控程序、病史报告程序、药物相互作用程序、医疗专家系统程序、生理监控程序的相关性、医疗条件警告程序、医疗条件警报程序、医疗信息系统程序；

E. 从以下的组中选择出的至少两个程序：染色体组数据碱基序列显示程序、局部查找染色体组片断标识计算程序、与已知有机化合物标识程序的相关性、染色体组计算策略比较程序；

F. 从以下的组中选择出的至少两个程序：作品编排协议、管弦乐编曲程序、电影制作管理程序、动画程序、音响特效程序、视觉特效程序、多媒体播放事件程序、电影编辑程序、音响编辑程序、混音程序、图像序列混合和排序程序。

G. 从以下的组中选择出的至少两个程序：交互式命令控制设备程序、观察监控程序、状态程序的被动查看、警告启动程序、警报启动程序；以及

H. 从任一前述的组中选择出的第一程序、从任一前述的组中选择出的第二程序、以及使来自第一程序的数据内容与来自第二程序的数据内容相互关连的第三程序。

因而，从进程中计算机化应用程序的角度而言，实质上本发明的数据流和/或数据序列实施例涉及交互式命令控制装置和/或观察监控（例如状态的被动观察、警告启动、警报启动等）；包括实时金融数据流交易的交互、项目管理和协作、生理监控和医疗信息系统的集成、染色体组计算策略、作品编排协议（例如管弦乐编

曲、电影制作管理、多媒体播放事件等）等。

然而，作为最简单的表述，本发明涉及基本上如在此所述或所示的虚拟桌面一元组织和控制系统，并表征为具有至少一个循环数据结构，以及与其相关联的小型-映象（Mini-Map）模块和操作系统界面。

本发明的核心技术实施例使用户能保存当前环（窗口及其在环上的位置），并因此使得能在其它事件继续完全相同的工作环境。用户可具有分类排列的“模板环”（例如具有大学材料应用程序、工作应用程序的环、新闻环、嗜好环等）以及在计算机工作期间某一时刻保存的“快照环”。对于手-眼协作补偿方法，这是用极简单和直观的方法来启用的一以旋转环。此外，“小型-映象”（Mini-map）包括环上窗口的实际显示，但其尺寸较小且可配置。用户通过上下（往上将增加所显示窗口的数量，而往下则减少其数量）拖动小型-映象的底部，可定义所显示窗口的尺寸和数量。用小型-映象导航与导航实际环相似。最后，系统可使用个人计算机、PDA、手机、机顶盒、TV 等。

本实施例便于最一般的元-组织和控制模式，通过该模式来表示、导航并操纵任意图标对象集的动态改变的“虚拟显示”。

图标对象一般被定义为应用程序（操作系统应用、HTML 页面、web 应用程序等）的任意显示。对该技术的实施例的特定当前观点陈述如下。然而，此处的定义旨在包括该特定当前观点的全部应用和变化，以及全部组分。

以软件实现的本系统由两个主要部分组成一新的显示管理器、以及辅助的“小型-映象”桌面导航工具。目的是提供直观和可以管理的虚拟桌面。该目的通过创建新的“环”的虚拟概念来获得，其中应用程序窗口彼此相邻地放置，而不是彼此重叠。

本系统的实施例沿着虚拟环延伸桌面，从而使用户能平稳并连续地旋转桌面，获得像用户想要限定的大小一样的有效桌面尺寸。该环是环形的；因此，如果用户往右滚动，显示将最终返回到其原始起点。这样用户在环的环境中将永远不会“迷路”。当打开新的应用程序窗口时（例如 Microsoft Explorer），它自动地或通过在托盘（tray）图标上双击（或者可能通过键盘/鼠标的组合）结合到“环”中。该窗口可在用户在屏幕上看到的当前应用程序窗口之间被结合到环中，或环的“右端”。在第一种情形中，桌面可延伸到侧面并在屏幕上显示的当前应用程序窗口之间打开一个“空白空间”，即将放置新窗口的空间。该放置操作可动态地或立即进行。当

用户关闭一应用程序时，该环将闭合空缺空间，保持整齐有序的桌面。显示管理器与用于操纵应用程序的操作系统内嵌方法完全兼容。

如果用户想要移动窗口的位置，然后将其放回“环”中，可在任何时候通过双击托盘图标或使用鼠标/键盘组合来进行。

导航工具

系统的自动窗口位置预测器（AWPP）特征被设计成改进用户的手-眼协作。在用户开始环的移动的任何时候，AWPP 投入动作。它计算当前移动的最可能结束位置，因此能补偿用户较小的不精确。例如，当用户将环向全尺寸窗口移动并结束移动时上述窗口未准确地与显示器边界对齐，则 AWPP 将假设用户实际上想要得到的是将窗口与显示器边界对齐，因此无缝地校正用户输入来获得该目的。

当用户按下 alt-tab 键或使用任务栏来切换应用程序时，该环将自动旋转以将选定应用程序带入视图中。此外，该平台使用户能以非常简便的方式在应用程序之间进行拖放（Microsoft Windows OS 特征）。用户将对象拖到屏幕末端直到目标应用程序显示，或者只是将对象放入该应用程序。

“后退-向前”：当用户改变其活动窗口时，系统将保存活动窗口的历史，并向用户提供在活动窗口历史上浏览的两个箭头（向前和后退）的界面。当用户在窗口历史中前后移动时，将旋转整个环以将所需窗口置于显示屏幕中央。

“环着色”：为了改进用户的方向感，该环被分成四个部分，每个 1/4 部分背景的着色都不同。

“环图标”：

1. 环的平面图标表示，分成四个部分，每一个都根据预先定义的“环着色”来进行“着色”。点击该四个部分之一将使该环旋转到该部分的开始处。

2. 环的适当着色的 3D 图像，呈现了整个当前 1/4 部分的缩略图。像第一个图标一样，用户可点击 3D 环图标的任一部分，并且该环将相应地旋转。

“历史”：小型-映象将打开用户曾经访问的 HTML 页面的历史。每个页面都伴有 HTML 页面的较小缩略图。

“最近文件”：最近在该环上使用的但现在并未打开的文件列表。该列表将以相似方式呈现给“活动环”小型-映象（即每个最近文件都伴有缩略图）。

显示特征

“窗口分组”：该特征根据窗口内容（及应用程序的类型）将窗口归入环的特定位置。例如，一种策略是全部 Word 窗口将组成一组，从而在打开新的 Word 窗

口的任何时候，它毗邻于环中另一个打开的 Word 窗口。

“3D 支持”：近来，3D 显示器已引入市场。这些监视器支持 3D 幻象（类似于 iMax 影院）。在这种硬件上，该系统将便于超出标准硬件上可能的屏幕尺寸地来显示该环的较大部分。这将通过在显示屏的 3D 空间中呈现毗邻于前景窗口的环的一部分来实现，从而该环将显现为沿该屏幕弯曲。该用户将获得他位于该环中央的感觉。

‘粘滞窗口’特征使用户能定义当卷动桌面时在屏幕上保持其位置的应用程序窗口。该特征可用于诸如音乐播放器的应用程序；或者用户想要极快速进入同时使其窗口尺寸保持相对较小的任何应用程序。

“多监视器支持”：该软件支持多监视器显示。屏幕尺寸将计算为全部监视器的总屏幕尺寸。

“环的紧致化”：通常，网页被设计成以“独立”模式由用户查看。即，单个全尺寸窗口。因此，许多网页包括在共享显示环境中被视为浪费的较大边界。在面向环的显示中，这些边界不再是必要的，因此在启用时该特征将自动地调整尺寸或以其它方式改变 Web 浏览器，来消除浪费边界，从而使用户能以紧凑和有效的方式来查看更多的信息。

“多环支持”：该软件的基本操作是使用一个虚拟环。此外，该软件还支持多个环的同时存在。用户可单独添加、移除、命名并配置每个环的属性，并从一个环到另一个环地重新定位窗口。

该软件提供使用户能在各环之间导航的界面。

“增大的最大窗口大小”：在操作系统的正常操作中，很难理解允许窗口大于屏幕尺寸。然而，面向环的显示使该选项变成可行。因此该软件支持尺寸大于屏幕的窗口的存在。例如，集成化开发环境（诸如 Microsoft .NET IDE）可受益于该选项。

界面

该界面是简单并且直观的，可基于鼠标、或任何其它定点或滚动设备、和/或键盘。

当用户将鼠标指针指向显示器的右端时，桌面自动向右滚动，露出该环的“隐藏”部分。滚动继续，且滚动的速度可根据鼠标指针在显示屏上的相对高度来进行控制。

滚动通过按压鼠标的中间键使鼠标移向桌面应滚动的方向来执行。例如，如

果用户按压中间键并将鼠标移向右，则桌面将滚向右边。桌面的滚动速度取决于鼠标从用户按下中间键时的点移动开始与鼠标的当前位置的距离。当距离越大时，桌面移动就越快。

离散滚动 a)： 键盘组合将允许用户能使该环一次旋转一个窗口。该移动是动画的，且在环上的下一窗口移到显示屏中央时结束。

离散滚动 b)：键盘/鼠标组合将用动画或不用动画，使该环旋转预先配置的距离。例如，‘shift’+鼠标滚轮对该滚轮的每次“点击”旋转一个窗口。

环的保存和加载

该软件使用户能保存整个的环构造。用户后来可在几秒钟内将整个环载入桌面，从而创建预定义的工作环境。当保存一个环时，该软件将把用户当前使用的环的全部内容保存到一文件中。当用户想要从现有文件中载入环时，软件将执行相关应用程序和文档，并以保存它们的完全相同方法排列它们。当选择要载入的文件时，将向用户提供要载入的可用环的缩略图，它非常像“小型-映象”上所显示的图（参见以下内容）。

小型-映象模块

参看图 5 和 6，软件包括桌面（530、630）的可调节尺寸的小型-映象（510、610）显示。该小型-映象是整个虚拟桌面的缩略图，并且实际上是桌面的准确和完整的按比例缩小版本。用户能在按比例缩小版本中看到像它们显现在屏幕上一样的每个窗口或应用程序的内容。

对当前未显示在屏幕上的环的“隐藏”部分中桌面的改变仍然在小型映射显示中得到反映。另一方面，对小型映射的改变转而反映到桌面上。例如，用户可使用鼠标来将小型-映象上的微缩窗口拖到新的位置，并且桌面上的实际窗口将相应地移动。小型-映象可用来简便地导航该环－当用户点击映射上的窗口时，该环将自动旋转到准确位置，以便于将该窗口置于显示屏中央。该小型-映象能被配置为自动隐藏，且其显示屏可以是半透明的，从而还能看到其下显示屏的内容。为便于对小型-映象的用户定向，它将根据“环着色”方案进行着色。

附图说明

为了理解本发明并查看实际上可如何实现它，现在将不仅仅通过非限制示例，而是参照附图来描述包括较佳实施例的各个实施例。此外，对本发明及其优点的更全面理解可通过参阅附图和以下描述来获得，在附图中相同标号表示相同特征，且

其中：

图 1 示出三个并行循环数据结构的示意图；

图 2 示出与按序关联于循环数据结构上各个位置的进程相对应的图项的示意图；

图 3 示出其上能看到与循环数据结构相关联的一个图项的显示屏的示意图；

图 4 示出其上能看到与循环数据结构相关联的两个部分图项的显示屏的示意图；

图 5 和 6 示出与具有本发明最佳实现模式的计算机相关联的显示器的实际屏幕截图；

图 7 示出本发明的核心技术实施例的基本功能组织的示意图；

图 8-13 示出本发明各个较佳实施例的示意图；

图 14 示出在多个数据序列、事件等中看到的基本上并行的循环数据结构的示意图。

具体实施方式

参看图 7，本发明的主要较佳实施例涉及一种在计算机处理环境中使用的虚拟桌面一元-组织和控制系统，该计算机处理环境具有至少一个具有相应操作系统的处理单元，且虚拟桌面“软件”系统包括：A. 在可实时访问的存储器媒体中的至少一个动态的基本循环的电子数据结构（130）；B. 与每个所述数据结构相关联的进行中的算法动作（140），该进行中的算法动作分别定期地（I.）基于相应的操作系统（100）数据访问，将在环境或其预定部分中运行的多个进程中的每个进程转变（142）为相关联的图形表示，以及（II.）逻辑地将该表示分配（144）到数据结构中的位置上；C. 与每个所述数据结构相关联的图形用户界面（150），该图形用户界面便于（I.）在显示设备上（190）查看（152）分配给至少一个数据结构或其一部分的表示，以及（II.）组织（154）该至少一个数据结构。

包括与 OS 相连的有些复杂界面的软件（以上 B.I 元件）的当前最佳启用模式有两个部分，都需要使软件的小型-映象模块（以上 C.I 元件）能显示应用窗口的微缩表示。出现困难是因为（多数时候）共用面向环的桌面的应用程序窗口并不直接在屏幕上显示。尽管捕捉屏幕上窗口的图形内容（然后简单地缩小它）是简单和普通的动作，但是却没有捕捉屏幕下窗口的图形内容的直接方法。该问题有两个版

本，因此有两种解决方案：

有些应用程序使捕捉其图形内容更加简单，比如 Microsoft Internet Explorer。IE 应用程序实际上是具有一标准界面的 IE 浏览器“组件”的包装器。该界面的方法之一是具体地设计成允许捕捉浏览器的图形内容。然而，该界面仅可用于拖移应用程序，而我们却需要捕捉不同应用程序的内容。为此，我们使用一特定注入技术（有些类似“黑客”但有良好文档的技术）来将一代理程序（agent）插入 IE 进程。该代理程序能捕捉图形内容，然后将其通过标准 IPC（进程间通信）传送给主要应用程序，然后缩小之并显示在小型-映象上。

对于没有诸如在 IE 中呈现的预先设计界面的“哑”应用程序，问题就更严重了。在该情形中，必须使用更复杂的技术。再一次，我们将代理程序插入主应用程序中。然而，这一次我们使用前述的“API 钩连”技巧，以便于钩连所有的标准 OS 涂画界面，然后我们从中进行干预并诱使应用程序将它自己画入存储器“屏幕对象”（同时认为它已画入实际屏幕）。然后代理程序随后通过标准 IPC 像前面一样将该存储器木对象发送给主应用程序。

核心技术实施例

基本环（循环数据结构）导航可用鼠标来完成。有两种缺省界面。这些界面可从设置面板进行配置。

A. 通过相关鼠标移动来滚动－通过按下<CTRL>键+鼠标右键或按住鼠标中键来激活；当左右移动鼠标时，滚动速度直接与鼠标的水平移动成比例。

B. 通过鼠标在屏幕边缘滚动－视图通过将鼠标置于屏幕侧边来滚动。滚动速度与鼠标的竖直位置成比例：在顶部快一些，而在底部则慢一些。

C. 选项菜单－通过在任务栏右角中的预定“DeskLoops”图标上点击鼠标右键来从托盘图标上激活，或者从映象上激活。双击“Deskloops”激活“排列”选项，该选项展开该环上的全部打开窗口。

D. 导航图－当鼠标置于屏幕顶部时显现，用来查看整个环或环的一部分；通过点击它跳到一窗口；通过鼠标右键关闭窗口；通过鼠标右键将窗口设置为粘滞窗口（总在视图中）；且该图可通过拖动该图窗口的下边缘来调整大小。

核心技术实施例－可执行代码结构

软件中主要模块的列表及其功能：

1. UIManager（UI）。该模块负责收集所有系统范围的用户输入。它分辨用户何时按压或移动鼠标、映象是否必须显示或隐藏、用户滚动是否启动或终止等。

2. MapManager（MAP）。该模块负责显示小型-映象并在小型-映象本身处理用户输入。

3. AnimatorManager（ANIM）。该模块负责将所有在不同的异步时间上发生的可能滚动和移动命令组合成一个流畅的和（30 毫秒）同步的移动。它从系统的所有其它模块中接收所有移动命令，并将它们“切”成多个馈入 ScrollManager 的 30 毫秒的块。

4. SystemHookManager（SYSHOOK）。该模块负责通过捕获 OS 事件来与 OS 通信。它检测窗口何时打开或关闭、移动、调整尺寸、或作图形更新。它还检测焦点的变化，并从其它窗口接收图形内容。SystemHookManager 模块涉及本实现的一些更为技巧的技术方面：

a. 事件的捕获用系统范围的钩连完成，这些钩连必须置入与主要可执行文件不同的 DLL 中。该方法将 DLL 载入系统中每个其它进程的地址空间，因此将信息传回主应用程序进程就有困难了。它可使用标准的进程间通信（事件、共享存储器等）来完成。

b. 其它应用程序的图形内容的捕获在技术上也是有些复杂的。有些应用程序支持 COM 界面来接收图形内容（像 Internet Explorer），但是许多应用程序并不支持。捕捉这些应用程序的内容可通过钩连到操作系统的绘制功能来完成。在 Windows OS 上，这可通过繁难但现在有良好文档的 API 钩连来执行。

5. ScrollManager（SCROLLER）：滚动引擎执行单个的移动命令。为了使单个滚动移动更快，并不更新整个屏幕，而是仅更新看得见的相关部分。这样各个应用程序就不需要重画它们自己了。这样做的（附加）算法基于标识每个可见窗口的 z-排序，并确定它的哪一部分在任何给定时间可见。

以下事件在 06: 00: 00.00 和 06: 00: 00.03 之间（以及之后每 30 毫秒）发生：

1. 该系统环绕所有“输入模块”并接收下一循环的命令。

2. PhisicalToLogical()：将屏幕的物理位置内容读成逻辑表示。

3. 系统按照到达顺序来分配命令。主要的命令和所存取的相应动作为：

1) “Rotate Loop”（旋转环）（UI）：将分散的移动命令发送给 AnimatorManager 模块。

2) “Show Map”（显示映象）（UI）：将激活命令发送给 MapManager 模块。

3) “Hide Map”（隐藏映象）（UI）：将去激活命令发送给 MapManager 模块。

4) “End User Movement”（终端用户移动）（UI）：激活 AWPP。

5) “MoveToWindow”(移到窗口)（MAP）：向 AnimatorManager 发送将所需窗口置于中央的动画命令。

6) “RecorderLoop”（对环重新排序）（MAP/SYSTRAY）：重排窗口使不出现重叠。

7) “SaveLoop”（保存环）（MAP/SYSTRAY）：系统遍历虚拟环上的全部窗口应用程序，并将相应信息此处到文件中。

8) “LoadLoop”（载入环）（MAP/SYSTRAY）：系统从特定文件中读取信息，并根据所保存文件中的信息来执行应用程序并将其置于环上。

9) “New Window Create”（创建新窗口）（SYSHOOK）：计算新窗口打开的地方，并向 AnimatorManager 发送适当的动画命令。

10) “Close Window”（关闭窗口）（SYSHOOK）：向 AnimatorManager 发送“关闭”空出空间的动画命令。

11) “Switch Window”（切换窗口）（SYSHOOK）：该命令从 SystemHookManager 中接收，其时该模块标识切换窗口焦点的外部（用户或 OS）操作，像按 alt-tab 键或相应的 OS 调用。该系统向适当窗口发送“MoveToWindow”命令。

12) “Scroll Command”（滚动命令）（ANIM）：该命令通过 ScrollManager 模块处理，该模块执行每个窗口（和/或整个环）所需的单个累积滚动。

13) LogicalToPhysical()：从内部逻辑表示中更新物理显示。

核心技术实施例一可执行代码核心算法组（ECCAG）：

```
{
    CArray<WindowObject *, WindowObject *> &list=source->list;
    CArray<WindowObject *, WindowObject *> ylist;
    CArray<sortstruct, sortstruct &> XArray;
    CArray<sortstruct, sortstruct &> XUniqueArray;
    CArray<sortstruct, sortstruct &> YArray;
    CArray<sortstruct, sortstruct &> YUniqueArray;
    CArray<backstruct, backstruct &> ThresholdArray;
    CArray<backstruct, backstruct &> ThresholdUniqueArray;
```

```
CArray<CArray<BOOL, BOOL>, CArray<BOOL, BOOL> &> XMap;
CArray<CArray<BOOL, BOOL>, CArray<BOOL, BOOL> &> YMap;
CArray<CArray<CRect, CRect &>, CArray<CRect, CRect &> &> XRealRect;
CArray<CRect, CRect &> RectArray;
CArray<HWND, HWND &> ZOrderArray;
WindowList *rc = new WindowList;;
int i,j,k,l;
// stage 2.
for(i=0;i<list.GetSize();i++) {
  sortstruct temp;
  temp.hWnd=list[i]->hWnd;
  temp.x=list[i]->ws.left;
  XArray.Add(temp);
  temp.x=list[i]->ws.right;
  XArray.Add(temp);
} .
// Here we perform a bubble sort
for(i=0;i<XArray.GetSize();i++) {
  sortstruct temp;
  for(j=0;j<XArray.GetSize()-1;j++) {
        if(XArray[j].x>XArray[j+1].x) {
              temp=XArray[j+1];
              XArray[j+1]=XArray[j];
              XArray[j]=temp;
        }
  }
 }
// Remove duplicate entries
  // XArray ==> XUniqueArray
  for(i=0;i<XArray.GetSize();i++)
    XUniqueArray.SetAtGrow(i,XArray[i]);
  // Remove the duplicate entries from XUniqueArray.
```

```
        for(i=1;i<XUniqueArray.GetSize();) {
          int a,b;
          a=XUniqueArray[i].x;
          b=XUniqueArray[i-1].x;
          if(a==b)
                XUniqueArray.RemoveAt(i);
          else i++;
        }
          // Stage 4
        XMap.SetSize(XUniqueArray.GetSize());
        for(i=0;i<XMap.GetSize();i++)
          XMap[i].SetSize(list.GetSize());
        // stage 6
        // first column gets special treatment
        for(i=0;i<list.GetSize();i++)
          XMap[0][i]=RelateXWindow(XUniqueArray[0].x,list[i]->hWnd,XArray);
        for(j=1;j<XMap.GetSize();j++) {
          for(i=0;i<list.GetSize();i++) {
                XMap[j][i]=RelateXWindow(XUniqueArray[j].x,list[i]->hWnd,XArray);
                XMap[j][i]=(XMap[j-1][i] ^ XMap[j][i]);
          }
        }
#ifdef WINTESTDEBUG
        afxDump.SetDepth(1);
        afxDump << "Dumping XMap:\n";
        XMap.Dump(afxDump);
        afxDump << "\n";
        afxDump.Flush();

#endif
        // For every column do:
        for(j=0;j<XUniqueArray.GetSize()-1;j++) {
```

```
// Create ylist - the sublist of a column.
ylist.RemoveAll();
YArray.RemoveAll();
YUniqueArray.RemoveAll();
for(i=0;i<list.GetSize();i++) {
        if(XMap[j][i]) ylist.Add(list[i]);
}
// stage 2 again - this time for y
for(i=0;i<ylist.GetSize();i++) {
        sortstruct temp;
        temp.hWnd=ylist[i]->hWnd;
        temp.y=ylist[i]->ws.top;
        YArray.Add(temp);
        temp.y=ylist[i]->ws.bottom;
        YArray.Add(temp);
}
// Here we perform a bubble sort
        for(l=0;l<YArray.GetSize();l++) {
        sortstruct temp;
        for(i=0;i<YArray.GetSize()-1;i++) {
                if(YArray[i].y>YArray[i+1].y) {
                        temp=YArray[i+1];
                        YArray[i+1]=YArray[i];
                        YArray[i]=temp;
                }
        }
}
// Remove duplicate entries
for(i=0;i<YArray.GetSize();i++)
        YUniqueArray.Add(YArray[i]);
for(i=1;i<YUniqueArray.GetSize();) {
        if(YUniqueArray[i].y==YUniqueArray[i-1].y)
```

```
                YUniqueArray.RemoveAt(i);
        else i++;
}
// Normal size plus one to hold the totals
YMap.SetSize(YUniqueArray.GetSize());
ZOrderArray.RemoveAll();
for(i=0;i<YUniqueArray.GetSize();i++) {
        HWND tmphwnd=NULL;
        ZOrderArray.Add(tmphwnd);
}
for(i=0;i<YMap.GetSize();i++) {
        YMap[i].SetSize(ylist.GetSize());
}
// ZOrderWindowList is sorted from high to low, left to right
for(i=0;i<ylist.GetSize();i++) {
        if(YMap[0][i]=RelateYWindow(YUniqueArray[0].y,ylist[i]->hWnd,YArray))
{
                // See if we have to replace something
                if(ZOrderArray[0]==NULL)
                        ZOrderArray[0]=ylist[i]->hWnd;
                else {
if(ZOrderWindowList.IsLeftOf(ylist[i]->hWnd,ZOrderArray[0]))
                                ZOrderArray[0]=ylist[i]->hWnd;
                }
        }
}
for(k=1;k<YMap.GetSize();k++) {
        for(i=0;i<ylist.GetSize();i++) {
YMap[k][i]=RelateYWindow(YUniqueArray[k].y,ylist[i]->hWnd,YArray);
                if(YMap[k][i]=(YMap[k-1][i] ^ YMap[k][i])) {
                        if(ZOrderArray[k]==NULL)
                                ZOrderArray[k]=ylist[i]->hWnd;
```

```
                    else {
        if(ZOrderWindowList.IsLeftOf(ylist[i]->hWnd,ZOrderArray[k]))
                                ZOrderArray[k]=ylist[i]->hWnd;
                    }
                }
            }
        }
        // Stage 11 - calculate the list of rectangles
        CRect newRect;
        newRect.left=XUniqueArray[j].x;
        newRect.right=XUniqueArray[j+1].x;
        for(k=0;k<YUniqueArray.GetSize()-1;k++) {
            if(ZOrderArray[k]) {
                WindowObject *wo=new WindowObject;
                newRect.top=YUniqueArray[k].y;
                newRect.bottom=YUniqueArray[k+1].y;
                wo->ws=newRect;
                wo->hWnd=ZOrderArray[k];
                rc->list.Add(wo);
            }
        }
    }
    return rc;
}
```

因而，本发明一实施例的虚拟桌面一元-组织和控制系统（参见图 8）包括用于执行在计算机系统中的活动应用程序之间导航的方法中的各个步骤的软件。该方法包括以下步骤：在用户屏幕上，显示（800）具有对应于显示屏幕的多个区域的基本环形“虚拟表面”；使多个应用程序窗口边界（在用户的计算机处理环境中运行的进程）与该表面的各个部分相关联，从而至少一对应用程序窗口边界使至少一个边界关联于该表面的相邻部分；以及通过在所述至少一个边界上移动定点设备来从相应应用程序窗口进行导航（820）。

因而，本发明另一实施例的虚拟桌面一元-组织和控制系统（参见图 9）包括用于执行在计算机系统中的活动应用程序之间导航的方法中的各个步骤的软件。该方法包括以下步骤：提供（900）基本环形的虚拟表面；使窗口与该表面上的各个位置相关联（910）；以及通过在窗口边界上移动来在窗口之间导航（920），从而结

果窗口是与该表面边界上位置相关联的窗口。

此外，本发明又一实施例的虚拟桌面一元-组织和控制系统（参见图 10）包括用于虚拟桌面的软件。该虚拟桌面包括：分成对应于屏幕的各部分的基本环形虚拟表面（1010）；与表面的各个部分相关联的应用程序窗口（1020）；该表面的可视屏幕（1030）；在可视屏幕中显示的应用程序窗口（1040）；以及旋转该表面的用户工具（1050）一从而交换其中的可视屏幕部分。

类似地，本发明再一实施例的虚拟桌面一元-组织和控制系统（参见图 11）包括用于执行提供虚拟桌面的方法中的各个步骤的软件。该方法包括以下步骤：提供（1110）虚拟映象表面，该映象表面是环形的扁平环；首先使虚拟表面映象表面上的各个区域与多个显示区域相关联（1120），每一个显示区域都对应于物理显示屏幕；然后使得应用程序窗口与至少一个显示区域相关联（1130）；允许（1140）在表面上各显示区域之间导航，以在物理显示屏幕上显示相关联的应用程序；以及沿表面屏幕连续排列（1160）各个应用程序窗口而无重叠。

参看图 12，除了本发明的基本上在此描述和说明的全部实施例之外，有关于制造品和/或计算机程序产品的本发明各个实施例的一个并行集合，该制造品和/或计算机程序产品包括计算机可使用媒体（1210），它具有包含在虚拟桌面一元-组织和控制系统中的用于计算机处理环境的计算机可读程序代码，该计算机处理环境具有至少一个带有相应操作系统的处理单元，所述制造品中的计算机可读程序代码包括：

第一计算机可读程序代码（1220），用于使计算机在可实时访问的存储器媒体中形成并维护至少一个动态的基本循环的动作数据结构；

依赖于第一计算机可读软件，第二计算机可读程序代码（1230）用来使计算机运行进行中算法动作（与每个所述数据结构相关联），该进行中算法动作通常分别（I.）基于相应的操作系统数据访问，将在环境或其预定部分中运行的多个进程中的每个进程转变为相关联的图形表示；以及（II.）逻辑地将该表示分配到数据结构中的位置上；以及

依赖于第一计算机可读软件，第三计算机可读程序代码（1240）用来使计算机促使图形用户界面（与每个所述数据结构相关联）便于（I.）在显示设备上查看分配给至少一个数据结构或其一部分的表示，并（II.）组织该至少一个数据结构。

类似地（参看图 13），除了本发明的基本上在此描述和说明的全部实施例之外，有关于机器可读的程序存储设备（1310）的本发明各个实施例的一个并行集合，该

程序存储设备包括用于计算机处理环境的机器可执行指令的程序以执行虚拟桌面一元-组织和控制系统的方法步骤，该计算机处理环境具有至少一个带有相应操作系统的处理单元，所述虚拟桌面方法步骤包括：（A）（1320）在可实时访问的存储器媒体中，形成并维护至少一个动态的基本循环的动作数据结构；（B）（1330）与每个所述数据结构相关联地运行进行中算法动作，该进行中算法动作通常分别（I.）基于相应的操作系统数据访问，将在环境或其预定部分中运行的多个进程中的每个进程转变为相关联的图形表示；以及（II.）逻辑地将该表示分配到数据结构中的位置上；以及（C）（1340）与每个所述数据结构相关联，图形用户界面便于（I.）在显示设备上查看分配给至少一个数据结构或其一部分的表示，并（II.）组织该至少一个数据结构。

普通用户核心技术实施例的用户“遍历”

本发明的普通用户核心技术实施例的虚拟桌面一元-组织和控制系统称为“DeskLoops”一是由两个主要部分组成的软件：新的显示管理器和补充的“小型-映象”桌面导航工具。“DeskLoops”软件提供直观和可管理的虚拟桌面。本质上“DeskLoops”创建一新的虚拟概念“环”（基本循环的动作数据结构及其显像）同时应用程序窗口在显像中彼此相邻地放置（而不是彼此重叠）。

基本上，DeskLoops 引入桌面元-组织和（处理）控制的新的“环”概念。该概念的核心是通过在桌面上沿虚拟环展开各项（对应于各项的进程的视觉转换），使用户平稳连续地旋转桌面环的显像，从而获得与用户想要定义的一样大的有效桌面尺寸。

该环是“环形”的（动态的基本循环的电子数据结构）；因此，如果用户向右滚动，则显示将最终回到其原始起点。这样用户在环的环境中永远不会“迷路”。

当新的应用程序窗口打开时（例如 Microsoft Explorer），它自动地或通过在置于任务栏管理器之上的小图标上双击来结合到该“环”中。该窗口可在用户于屏幕上看到的当前应用程序的窗口之间结合到环中，或者结合到环的“右端”。在第一种情形中，桌面可延伸到侧面并在屏幕上显示的当前应用程序窗口之间打开一个“空白空间”，即将放置新窗口的空间。该放置操作可动态地或立即进行。当用户关闭一应用程序时，该环将闭合空缺空间，保持整齐有序的桌面。

显示管理器与用于应用程序操纵的操作系统内嵌方法完全兼容。

较佳特征

为了向用户提供有效直观的平台，自动窗口对准器的特征被结合到 DeskLoops

桌面平台中。自动窗口对准器负责将窗口的右侧或左侧与屏幕一侧对准。当用户将应用程序窗口打开成全屏显示时，该特征非常重要。在该情形中，自动窗口对准器使用户不用将桌面滚动到窗口的右侧或左侧与屏幕一侧对准的准确位置。

如果用户想要移动窗口的位置并再次将其置入“环”内，可通过双击置于任务栏管理器上的图标来非常简便地完成。

当用户按下 alt-tab 键或使用任务栏来切换应用程序时，该环将自动旋转以将选定应用程序带入视图中。此外，该平台使用户能以非常简便的方式在应用程序之间进行拖放（Microsoft Windows OS 特征）。用户将对象拖到屏幕末端直到目标应用程序显示，或者只是将对象放入该应用程序。‘粘滞窗口’特征使用户能定义当卷动桌面时在屏幕上保持其位置的应用程序窗口。该特征可用于诸如音乐播放器的应用程序；或者用户想要极快速进入同时使其窗口尺寸保持相对较小的任何应用程序。

DeskLoops 软件使用户能保存整个的环构造。用户后来可在几秒钟内将整个环载入桌面，从而创建预定义的工作环境。

界面

该界面是简单并且直观的，可基于鼠标、或任何其它定点或滚动设备、和/或键盘。当用户将鼠标指针指向显示器的右端时，桌面自动向右滚动，露出该环的“隐藏”部分。滚动继续，且滚动的速度可根据鼠标指针在显示屏上的相对高度来进行控制。

滚动桌面的另一种方法是通过按压鼠标的中间键使鼠标移向桌面应滚动的方向。例如，如果用户按压中间键并将鼠标移向右，则桌面将滚向右边。桌面的滚动速度取决于鼠标从用户按下中间键时的点移动开始与鼠标的当前位置的距离。当距离越大时，桌面移动就越快。

小型-映象导航工具

此外，DeskLoops 包括桌面的可调节尺寸的小型-映象显示。该小型-映象是整个虚拟桌面的缩略图，并且实际上是桌面的准确和完整的按比例缩小版本。用户能在按比例缩小版本中看到像它们显现在屏幕上一样的每个窗口或应用程序的内容。对当前未显示在屏幕上的环的“隐藏”部分中桌面的改变仍然在小型-映象显示中得到反映。当有比能显示在小型-映象上的应用程序窗口多的应用程序窗口时，用户也能滚动该小型-映象。

小型-映象可用来简便地导航该环－当用户点击映射上的窗口时，该环将自动

旋转到准确位置，以便于将该窗口置于显示屏中央。此外，小型-映象可用来特定地操纵桌面上任一给定窗口、应用程序或限定窗口组。该小型-映象能被配置为自动隐藏，且其显示屏可以是半透明的。

评论：本发明的工业可应用性－技术方面：本发明的虚拟桌面－元-组织和控制系统的具体例示可用标准的软件功能配置－虽然要由相当公开的“黑客”技术和工具作补充。

评论：本发明的工业可应用性－人机工程学方面：本发明的虚拟桌面－元-组织和控制系统的具体例示是直观的－既因为其概念组织的优势，又因为普通界面交互的优势（键盘命令、指向和点击动作、拖放动作等）。

评论：本发明的工业可应用性－经济方面：本发明的虚拟桌面－元-组织和控制系统的具体例示对普通和富有经验的用户而言都是效能成本合算的－而不像占用资源多的用户定制设计的虚拟桌面－元-组织和控制系统，它们是为行动协调者（例如项目管理、空中交通控制、C3、日间交易组的经纪人现金流管理、货币投机商、套汇投机商等）开发的。

特定重点特征

“DeskLoops 软件使用户能保存整个的环构造。用户后来可在几秒钟内将整个环载入桌面，从而创建预定义的工作环境。”预定义环甚至还提供“快照”工作环境的当前状态的能力。该技术沿着虚拟环展开桌面，使用户能平稳连续地旋转桌面环的显像，从而获得与用户想要定义的一样大的有效桌面尺寸。（摘自技术说明文档）用户喜欢自动地对用户需要作响应的功能极为强大的桌面。

对于手-眼协调连接以及多个应用程序之间的导航，本系统的自动窗口位置预测器（AWPP）特征被设计成改进用户的手-眼协作。在用户开始环的移动的任何时候，AWPP 投入动作。它计算当前移动的最可能结束位置，因此能补偿用户较小的不精确。例如，当用户将环向全尺寸窗口移动并结束移动时上述窗口未准确地与显示器边界对齐，则 AWPP 将假设用户实际上想要得到的是将窗口与显示器边界对齐，因此无缝地校正用户输入来获得该目的。

显示特征－图形用户界面“特定重点”程序功能

“窗口分组”：该特征根据其内容（及应用程序的类型）将窗口归入环的特定位置。例如，一种策略是全部 Word 窗口将组成一组，从而在打开新的 Word 窗口的任何时候，它毗邻于环中另一个打开的 Word 窗口。

“3D 支持”：近来，3D 显示器已引入市场。这些监视器支持 3D 幻象（类似于 iMax 影院）。在这种硬件上，该系统将便于超出标准硬件上可能的屏幕尺寸地来显示该环的较大部分。这将通过在显示屏的 3D 空间中呈现毗邻于前景窗口的环的一部分来实现，从而该环将显现为沿该屏幕弯曲。该用户将获得他位于该环中央的感觉。

“粘滞窗口”特征使用户能定义当卷动桌面时在屏幕上保持其位置的应用程序窗口。该特征可用于诸如音乐播放器的应用程序；或者用户想要极快速进入同时使其窗口尺寸保持相对较小的任何应用程序。

“多监视器支持”：该软件支持多监视器显示。屏幕尺寸将计算为全部监视器的总屏幕尺寸。

“环的紧致化”：通常，网页被设计成以“独立”模式由用户查看。即，单个全尺寸窗口。因此，许多网页包括在共享显示环境中被视为浪费的较大边界。在面向环显示中，这些边界不再是必要的，因此在启用时该特征将自动地调整尺寸或以其它方式改变 Web 浏览器来消除浪费边界，从而使用户能以紧凑和有效的方式来查看更多的信息。

“多环支持”：该软件的基本操作是使用一个虚拟环的操作。此外，该软件还支持多个环的同时存在。用户可单独添加、移除、命名并配置每个环的属性，并从一个环到另一个环地重新放置窗口。

该软件提供使用户能在各环之间导航的界面。

“所增大的最大窗口尺寸”：在操作系统的正常操作中，很难理解允许窗口大于屏幕尺寸。然而，面向环的显示使该选项变成可行。因此该软件支持尺寸大于屏幕的窗口的存在。例如，集成化开发环境（诸如 Microsoft .NET IDE）可受益于该选项。

注意：在描述本发明时，解释是根据当前所接受的技术理论（软件）或商用模型（管理、控制、组织等）进行的。这种理论和模型易遭绝对和根本的改变。常常这种改变会因为基本组成元件的表现得到改进，因为设想出这些元件之间的新变换，或者因为产生对这些元件或这些变换的新解释而出现。因此，注意本发明涉及各实施例中的特定技术实现是重要的。因此，在此涉及这些实施例的理论或模型相关解释是为了向本领域技术人员示教实际上如何实现这些实施例而提供的。对这些实施例的另外或等效的解释不会否定或改变其实现。

在此示出的数字、字母字符和罗马字仅为了便于说明，且决不应当视为在任何方法步骤上添加了特定顺序。类似地，本发明的各个实施例在此用一定程度的特殊性来进行描述。具体地，尽管本发明的各个实施例已参照包括实现本发明的较佳模式的特定示例进行描述，但是本领域技术人员将理解有上述系统和技术的许多变体和排列落于所附权利要求中陈述的本发明精神和范围内。

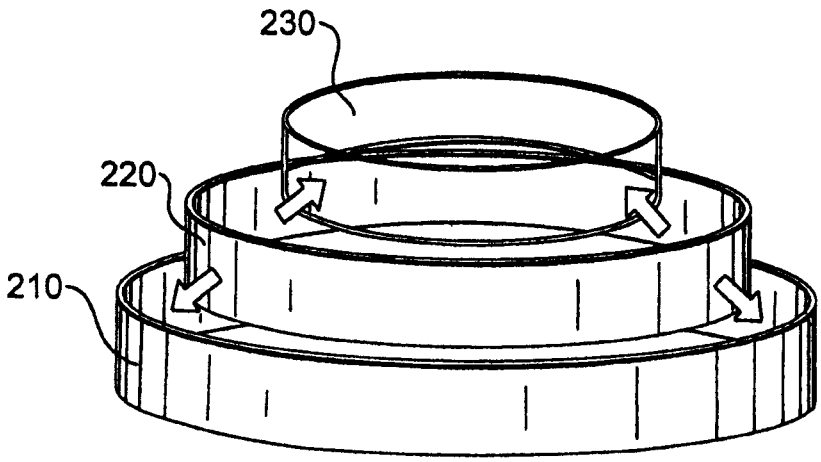


图 1

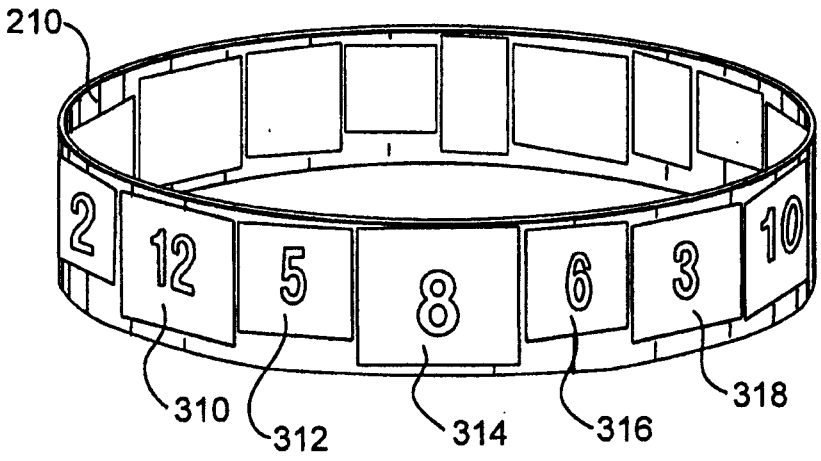


图 2

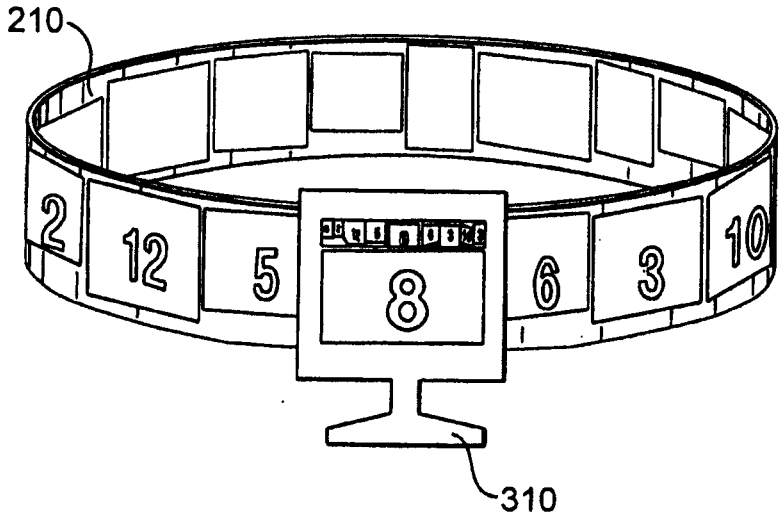


图 3

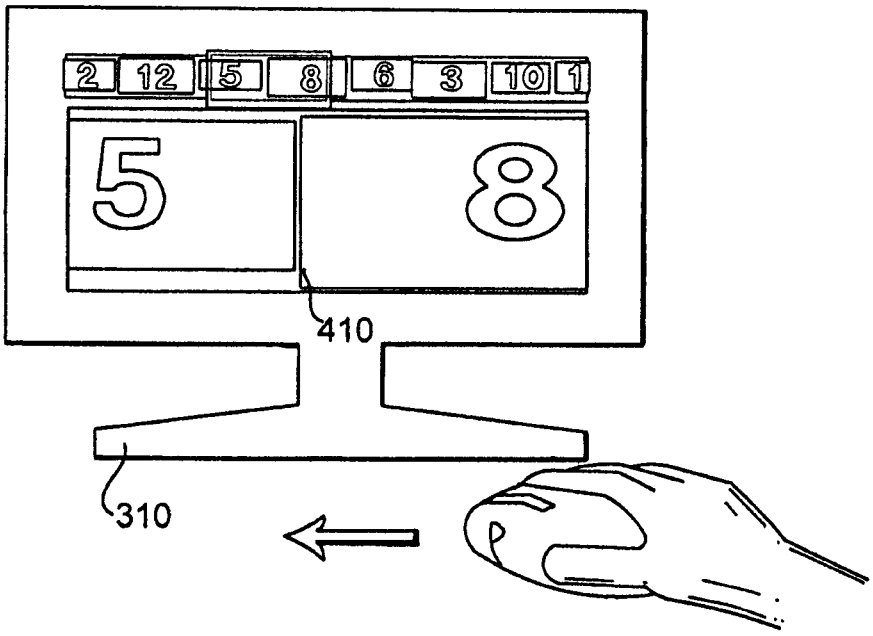


图 4

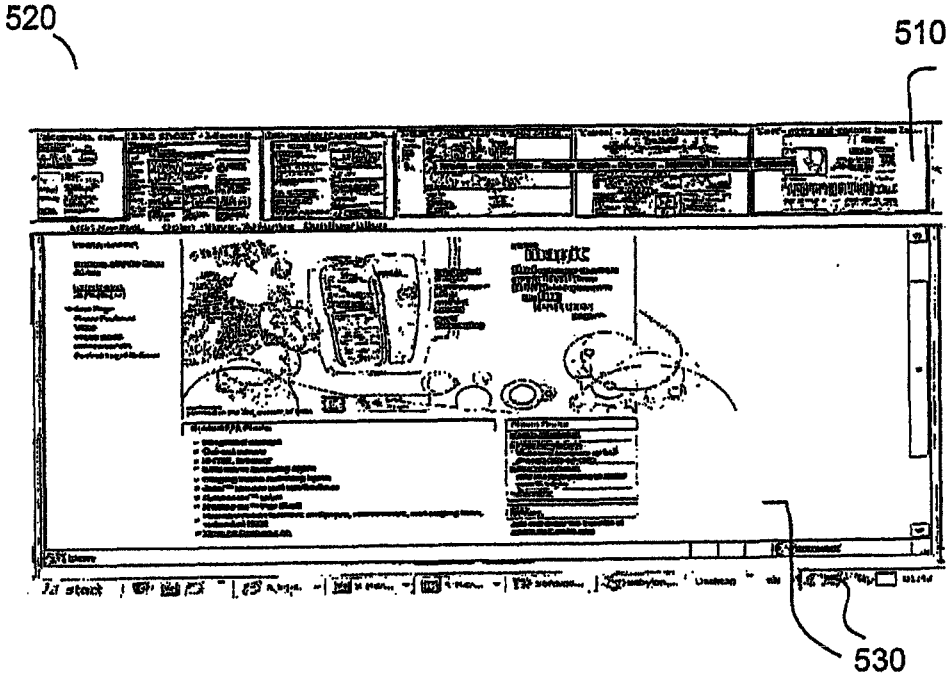


图 5

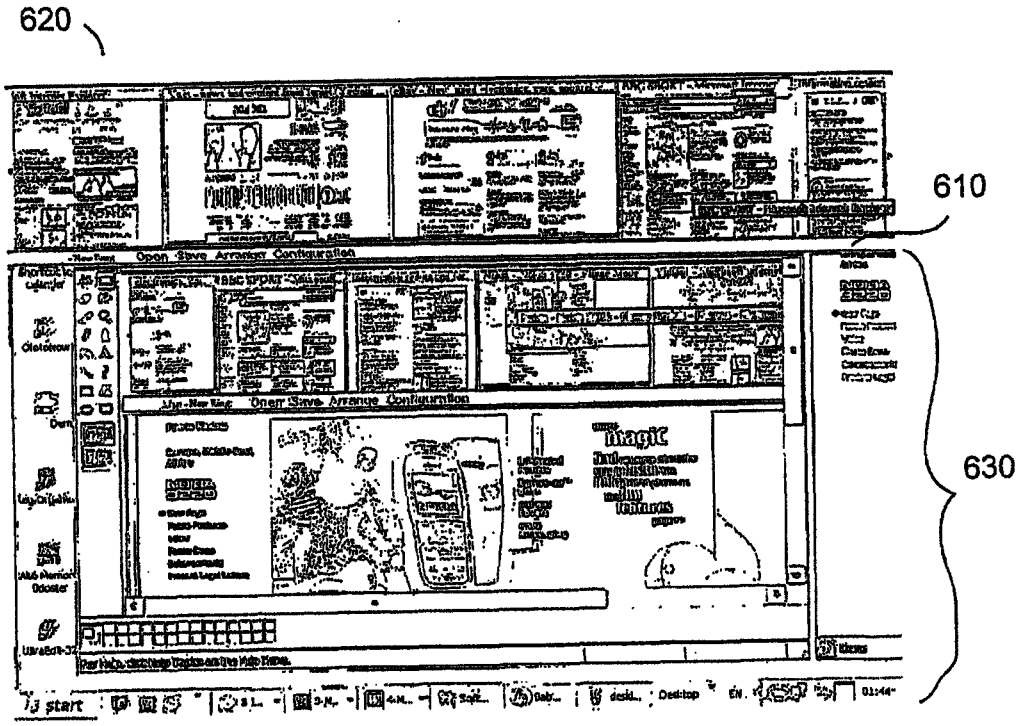


图 6

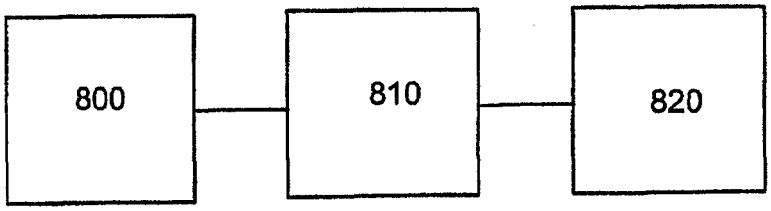

800
810
820

图 8

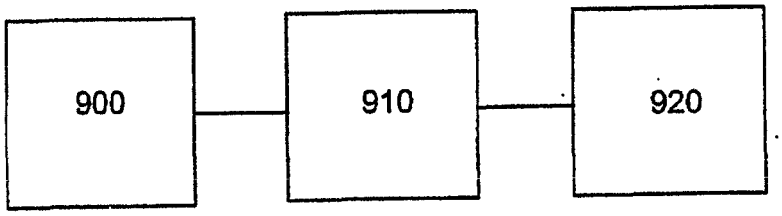

900
910
920

图 9

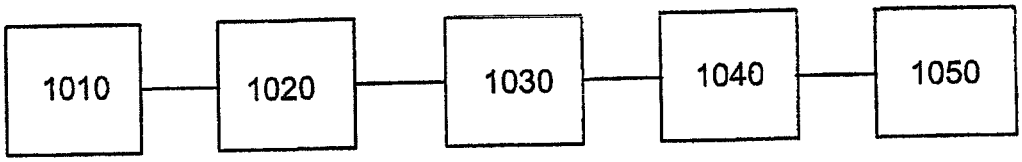

1010
1020
1030
1040
1050

图 10

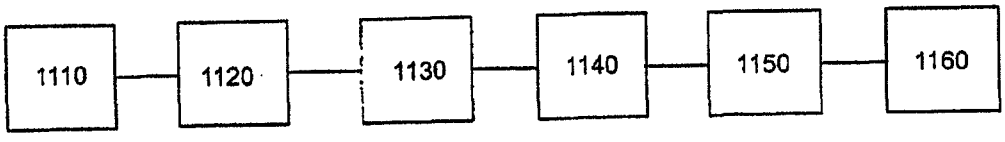

1110
1120
1130
1140
1150
1160

图 11

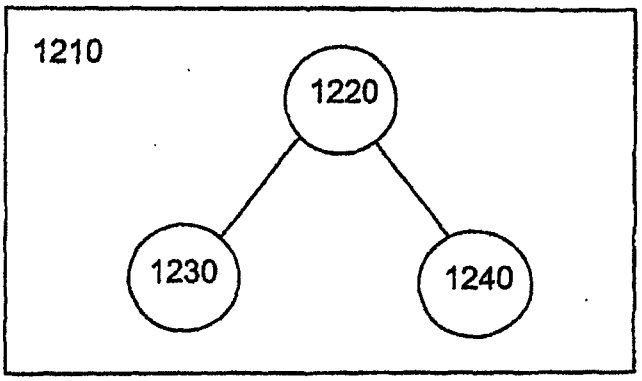


图 12

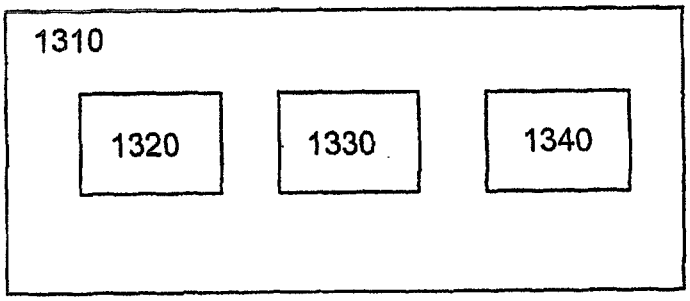


图 13

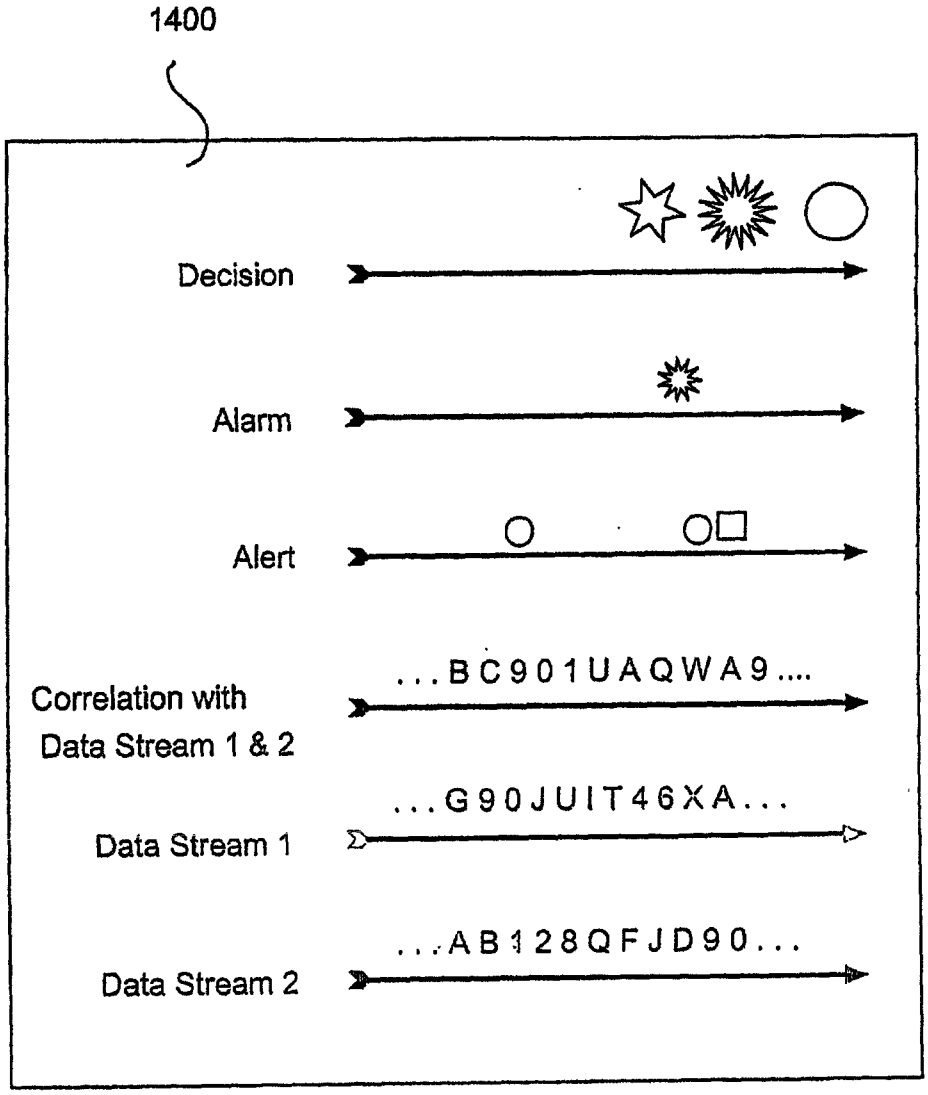

1400
Decision
Alarm
Alert
Correlation with
Data Stream 1 & 2
...BC901UAQWA9....
...G90JUIT46XA...
Data Stream 1
...AB128QFJD90...
Data Stream 2

图 14