[19]中华人民共和国专利局

[11] 授权公告号 CN 1026445C

[12]发明专利说明书

[21] ZL 专利号 92114194.7

[51]Int.Cl[5]
G06F 3/033

[45]授权公告日 1994 年 11 月 2 日

[24]颁证日 94.8.31
[21]申请号 92114194.7
[22]申请日 92.12.9
[30]优先权
[32]91.12.31[33]US[31]07/816,424

[73]专利权人 国际商业机器公司 地址 美国纽约

[72]发明人 林·J·H·巴斯姆
斯特福·T·伊根 哈温·G·科勒 G06F 13/00
瑞姆德·F·若门 G06F 15/16
杰夫瑞·J·V·慧科勒

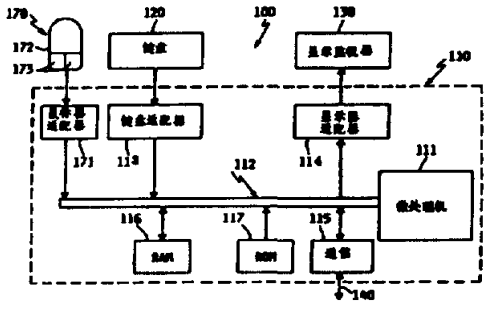
[74]专利代理机构 中国国际贸易促进委员会专利商标事务所
代理人 姜 华

说明书页数: 附图页数:

[54]发明名称 对数据处理系统中从属工作站的鼠标器或类似指示设备的支持

[57]摘要

一个从属的、不可编程的、基于字符的工作站终端（DWS）支持一个鼠标器。全部位于 DWS 中的终端控制器处理和显示鼠标器的移动情况，向一个管理许多 DWS 的工作站控制器（WSC）发送鼠标器按钮启动和鼠标器当前位置信息。WSC 根据一组规则处理按钮启动信息，并向服务于多个 WSC 的主处理机（MP）发送数据流。MP 部分地对代表鼠标器事件的数据流作出响应，执行用户应用程序。MP 也以数据流形式向 WSC 发送鼠标器规则。



权 利 要 求 书

1. 一个数据处理系统，其特征在于：

一个执行多个交互用户应用程序的主处理机；

至少一个工作站控制器，包括：

将所说的工作站控制器与所说的主处理机相连的装置，用于在所说的工作站控制器和所说的各个应用程序之间传送数据流，某些所说的数据流是鼠标器数据流，它们对由用户启动鼠标器，以及根据所说的鼠标器启动对由所说应用程序确定的操作作出响应；

存储 *WSC* 数据和程序代码的存储器装置；

用于发送和接收数据帧的一个通信适配器，所说的某些数据帧是鼠标器事件数据帧；

执行所说 *WSC* 程序代码的一个微处理机，用于将选择的所说数据流转换成所说数据帧，以及将选择的所说鼠标器按钮事件转换成数据流并且将它们传送到所说的主处理机；

多个与字符有关的从属工作站；

全都与所说的一个工作站控制器相连的终端，每个所说的终端包括：

具有特定的行和列的一个显示器，每个行列都能显示一组字

符中的一个字符；

接受用户各个击键信号的一个键盘；

一个鼠标器，包括产生代表移动事件数据的装置，以及当所说用户启动鼠标器时产生按钮事件的按钮装置；

用于发送和接收所说数据帧的一个通信适配器；

存储 *DWS* 数据和程序代码的存储器装置；

执行所说的 *DWS* 程序代码的一个 *DWS* 微处理机，用于处理所接收到的数据帧，以便在所说显示器的特定行和列显示所说的特定字符，用于传送代表所说击键动作的数据帧，用于根据所说鼠标器移动事件存储当前鼠标器位置，以及根据所说鼠标器按钮事件，将所说鼠标器事件数据帧传送给所说的工作站控制器。

2. 根据权利要求 1 的系统，其特征在于每个所说的鼠标器按钮数据帧包括一种类型的鼠标器按钮事件，以及所说当前鼠标器在所说显示器中所占的行列位置的信息。

3. 根据权利要求 2 的系统，其特征在于所说的鼠标器具有多个单独可启动的按钮装置，并且其中所说的鼠标器事件数据帧还包括哪一个所说的按钮装置被所说用户启动的信息。

4. 根据权利要求 1 的系统，其特征在于所说的 *WSC* 存储器包括规定一组与不同的所说鼠标器事件数据帧有关的操作的逻辑处理过程，并且其中所说的 *WSC* 微处理机根据来自所说从属终端的所说数据帧，执行所说的操作。

5. 根据权利要求 4 的系统，其特征在于所说的主处理机至少对一个所说的应用程序作出响应，将所说的逻辑处理过程包含在一个所说的鼠标器数据流中向所说的工作站控制器发送。

6. 根据权利要求 4 的系统，其特征在于所说的数据帧包括一种类型的鼠标器事件以及所说的当前鼠标器位置的信息，并且其中所说的逻辑处理过程既对所说的事件类型作出响应，又对所说显示器上多个预定区域中的一个区域内的所说当前位置作出响应。

7. 根据权利要求 4 的系统，其特征在于所说的逻辑处理过程存储在所说的应用程序无关的所说的 *WSC* 存储器中。

8. 根据权利要求 4 的系统，其特征在于在低于所说的主处理机级的所说的工作站控制器中执行所说的某些操作，使得所说的应用程序执行起来与所说的某些操作无关。

9. 一个数据处理系统，其特征在于：

一个执行多个交互应用程序的主机处理机；

至少一个通信媒介；

多个远离所说主机处理机的从属工作站终端，并且全部都通过所说的一个通信媒介与之相连，每个所说的从属工作站终端包括：

具有有限数目的独立的行和列的一个显示器，每个行列都能显示一组字符中的一个字符；

接受用户各个击键信号的一个键盘；

一个鼠标器，包括产生代表移动事件数据的装置，以及当所说用户启动鼠标器时产生按钮事件的按钮装置；

用于在所说的一个通信媒介上与所说的主机处理机交换所说的数据帧的一个通信适配器；

存储 *DWS* 数据和程序代码的存储器装置；

执行所说的 *DWS* 程序代码的一个 *DWS* 微处理机，用于接收所说的数据帧，以便在所说显示器的特定行和列显示所说的特定字符，用于传送代表所说击键动作的数据帧，用于根据所说鼠标器移动事件存储当前鼠标器位置，以及根据所说鼠标器按钮事件，将所说鼠标器事件数据帧传给所说的工作站控制器。

10. 根据权利要求 9 的系统，其特征在于每个所说的鼠标器按钮数据帧包括一种类型的鼠标器按钮事件，以及所说当前鼠标器在所说显示器中所占的行列位置的信息。

11. 根据权利要求 10 的系统，其特征在于所说的鼠标器具有多个单独可启动的按钮装置，并且其中所说的鼠标器事件数据帧还包括哪一个所说的按钮装置被所说用户启动的信息。

12. 按照权利要求 9 的系统，其特征在于所说字符组的子集确定可在所说显示器上的某些行和列位置显示的图形的各个不同的部分。

13. 数据处理系统的一个基于字符的从属工作站终端，该系统

有一个主机处理机，用于通过在一个通信媒介上进行数据帧交换而同时控制多个从属工作站终端，所说的从属工作站终端的特征在于：

在有限的多个位置的每一个位置上显示来自一组字符的多个单独的字符的一个显示器；

至少一个和所说主机处理机分开且远离的外壳；

与所说的单个通信媒介相连的一个通信适配器；

接受用户各个击键信号的一个键盘；

一个可由所说用户移动的鼠标器，用于产生代表鼠标器移动事件的数据，并且包括至少一个可由所说用户启动的按钮，用于产生代表鼠标器按钮事件的数据。

存储 *DWS* 数据和代码的存储器装置；

执行所说的 *DWS* 程序代码的一个 *DWS* 微处理机，用于接收来自所说的工作站控制器的所说的数据帧并将其中的字符显示于在所说数据帧中规定的位置，用于将所说的各个击键信号以分离的所说数据帧的形式向所说工作站控制器发送，用于根据代表所说鼠标器移动事件的所说数据，在所说存储器装置中存储所说鼠标器的一个当前位置，以及根据代表所说鼠标器按钮事件的所说数据，发送对所说鼠标器按钮事件解码的鼠标器数据帧。

14. 根据权利要求 13 的终端，其特征在于所说的有限的多个位置是指所说显示器上的行和列。

15. 根据权利要求 14 的终端，其特征在于所说字符组中的一个字符是一个鼠标，并且其中所说的 *DWS* 微处理机对所说 *DWS* 存储器中的所说当前位置作出响应，以便在所说显示器上由所说当前位置规定的行和列显示所说的鼠标。

16. 根据权利要求 14 的终端，其特征在于所说字符组中的一个字符是一个文本光标，并且其中所说的 *DWS* 控制器在所说显示器上由来自所说主机处理机的所说数据帧中的一个规定的行和列位置显示所说的文本光标。

17. 根据权利要求 13 的终端，其特征在于每个所说的鼠标器按钮数据帧包括一种类型的鼠标器按钮事件，以及所说当前鼠标器在所说显示器中所占的行列位置的信息。

18. 根据权利要求 17 的终端，其特征在于所说的鼠标器事件数据帧代表所说按钮装置的一次按下动作，一次释放动作，以及一声“卡搭”声。

19. 根据权利要求 17 的终端，其特征在于所说的鼠标器事件数据帧还代表所说按钮装置的两声“卡搭”声。

20. 根据权利要求 13 的终端，其特征在于所说的鼠标器具有多个所说的按钮，并且其中每个所说的鼠标器事件数据帧都包括由所说按钮中的一个按钮规定的部分。

21. 根据权利要求 13 的终端，其特征在于所说的 *DWS* 微处理机对来自所说的工作站控制器的所说数据帧中的一个预定的数

据帧作出响应，而后向所说的工作站控制器传送代表存储在所说终端存储器中的所说当前位置的唯一数据帧的一个序列。

22. 根据权利要求 21 的终端，其特征在于所说的一个终端在预定的时间间隔传送所说的唯一数据帧。

说　　明　　书

对数据处理系统中从属工作站的鼠标器或类似指示设备的支持

本发明涉及电子数据处理系统，特别是在由单个处理机或工作站控制器控制的各个以多字符为基础的从属工作站或哑终端中使用如鼠标器那样的指示设备的装置。

交互应用程序和系统已经经历了众多的阶段并发展到更自然的用户交互作用的形式。个人计算机和智能工作站盛行采用信息显示结构和非键盘式输入设备。被称为鼠标器的指示设备特别受到软件和系统设计人员的青睐。这种设备以及类似部件，比如转球式指示器、跟踪指示器和数字图形输入板，都有一个主体，用户可用它在平面上移动，在主体上还有一个或多个按钮或其它控制部分，用户可用它进行操作。移动鼠标器使显示屏上的鼠标或类似符号做相应的移动，使它对准显示屏上的一个信息。按动按钮便向应用程序或系统的某些部分发出信号，实现由按动按钮的类型和程序来确定一种功能。例如，当鼠标对准数据输入显示域时，按下和释放（“卡搭”一声）鼠标器左按钮便向个人计算机的图象管理（*Presentation Manager*）程序发出信号，将数据光标移向该区域。在短时间内按下释放

按钮两次(“卡搭”两声),则将输入进输入区域的数据传送至应用程序。其它按动方式,例如“拉和压”(在按下按钮的状态下移动鼠标器),在本领域中也是人所共知的。以后,“鼠标器”将用作描述包括所有类似指示设备的通用术语。

一大批用户在向指示设备的大规模进军中落在了后面。大多数中型和主机计算机系统与几十、几百甚至几千的廉价的称为从属的、不可编程的或哑终端的终端相连,这些终端统称*DWS*。这种终端确实包括可编程微处理机,但它们自己不为用户执行应用程序。多个*DWS*可以直接与系统的主处理机相连,或者与工作站控制器(*WSC*)相连。一个或多个*WSC*可以局部地或远距离地与主要执行应用程序的主处理机相连。*WSC*或主处理机,总起来说中央处理机或主机处理机,通过数据帧与每个*DWS*发生通信联系,数据帧规定了在*DWS*显示上的确切的数据格式并从*DWS*接收各个击键信号或用户的其它操作信号以及状态信息。此外,*DWS*一般还是与字符有关的,不具有图形接口。这就是说,显示屏总是以字符的行数和列数组织的。可以被显示的不同的字符模式由*DWS*内字符发生器中的模式确定;虽然多个或扩展的字符组有时可以静态地或在主机的间接控制下确定,但是只有某些字符模式或符号模式出现在屏幕上,且仅出现在屏幕上的预定位置。(虽然有些*DWS*终端支持包括直线和圆圈等在内的基本“图形”,但还不曾有支持图形用户接口的以字符为基础的*DWS*。所谓的*X*终端确实能支持图形接口,但不是

以字符为基础，它们的屏幕是内部所有点可寻址的，或称 *APA*。）

支持一个鼠标器或类似指示设备需要密集的数据流。支持处理机必须对鼠标器在一秒钟内取样许多次（通常在 20 次至 200 次的范围内），以监测鼠标器主体的移动情况和按钮的按动情况。图象管理（*Presentation Manager*）或应用程序要求鼠标在显示屏上必须移动足够快和足够频繁，以避免出现图象混乱和颤动现象，并且还必须对按键的操作迅速做出反应。

传统的解决办法是在采用应用程序的同时仅在智能工作站（*IWS*）中实现鼠标器功能；*IWS* 处理机和它的显示及输入设备之间的很宽的带宽使得用户能基本上实时地从鼠标器事件中觉察到必要的反馈。然而，*IWS* 终端价格昂贵。虽然二者的内部处理能力都将大幅度增加，但是在可预见的未来，*IWS* 终端的价格将维持在 *DWS* 的两部或三倍的水平。

在 *DWS* 上支持一个鼠标器的直截了当的方法是从 *DWS* 向主机（直接向主处理机或通过 *WSC*）发送数据帧，监测鼠标器的移动情况和按钮的按动情况，并且将数据帧返回 *DWS*，以便移动鼠标和/或在终端进行任何其它的操作。然而，为了管理许多 *DWS* 鼠标器，数据的传送速率使得联接终端和主机的电缆或其它媒介的带宽大大增加，同时也对主处理机提出了更高的要求。

本发明提供了在廉价的从属工作站（*DWS*）上对鼠标器或类似指示设备的支持，它没有显著地提高 *DWS* 的价格，也没有显著地增

加与具有许多这种终端的系统的主处理机通信的数据量。

这种对鼠标器的支持不需要改动主处理机的硬 件，应用程序的使用方式也简单。本发明适合于以一种特别简单的方式使用从系统中的主处理机分离的工作站控制器(*WSC*)；这种控制器能控制执行不同的主处理机应用程序的多个 *DWS*。

简言之，本发明使得在一个不可编程的或从属的终端使用鼠标器成为可能。因此，它向中型和主机数据处理系统提供了许多有益的现代用户接口。本发明通过以一种新颖的方式，在局部 *DWS* 终端和主 处理机之间分开管理鼠标器支持功能来实现上述和其它目的。

根据本发明的数据处理系统包括一台主处理机，用于执行多个应用程序和管理大量的不可编程或从属工作站。主机分成主处理机和工作站控制器(*WSC*)，主处理机实际上执行用户应用程序，而工作站控制器管理许多 *DWS* 终端并与主处理机通信。通信媒介在主机(或主处理机本身或 *WSC*)和大量的实际上分开的从属终端之间传送各个数据帧。终端包括一个显示器，一个由用户输入主要数据的键盘，以及一个鼠标器或类似指示设备。*DWS* 终端中的控制器存储鼠标器的当前位置，并在显示器上显示该位置，但不需要向主处理机传送鼠标器的移动情况。当操作者按下鼠标器上的按钮或 启动类似控制时 ，终端向系统主机发送当前鼠标器位置以及“鼠标器事件 ”的特性 。然后主机(主处理机和/或工作站控制器)决定采取什么操作 ，以及向终端返回什么数据 (如果有的话)。

图 1 是本发明的包括一个鼠标器的典型的从属工作站(*DWS*)的框图。

图 2 是典型的工作站控制器(*WSC*)的框图。

图 3 是包括图 1 和 2 所示部件的典型的数据处理系统的框图。

图 4 是表示图 3 系统中显示数据流的高标准框图。

图 5 表示包括文本和鼠标的显示板的例子。

图 6 是表示在图 3 的主处理机中执行典型的应用程序的过程的流程图。

图 7 是表示图 2 的 *WSC* 控制器运行过程的流程图。

图 8 是表示在对鼠标器的支持中图 1 的 *DWS* 的运行过程的流程图。

图 1 是根据本发明改进的一个从属工作站(*DWS*)100 的高标准框图。装在外壳中的控制器 110 包括具有总线 112 的微处理机 111，该总线 112 与几个适配器连接，使得与有限个数的外部设备的通信成为可能。例如，键盘适配器 113 能使微处理机 111 和键盘 120 进行通信，使用户能输入字符。显示适配器 114 使信息传送至显示监视器 130。经过电缆或其它常规的媒介 140，通信适配器 115 在 *DWS*100 和工作站控制器(*WSC*)之间传送信号。随机存取存储器(*RAM*)116 与总线 112 相连，用于存储终端中的可变更的数据，一般情况下它包括一个显示缓冲区。只读存储器(*ROM*)117 为大量的常规终端控制程序存储固定的程序码，这些程序监控 *DWS*100 的内部功能，这些

功能包括运行处理、诊断、通信程序，以及驱动在字符设置或多个可选择设置中为多个字符中的每一个产生显示模式的常规的字符发生器。

在本发明中，*DWS*100 还包括一个常规的鼠标器 170，它与终端控制器 110 中的适配器 171 相连。鼠标器具有一个可移动的手持主体 172，它产生常规的“位置事件”信号，指出操作者将它移动到多远处。鼠标器还有一些(通常为 1、2 或 3)按钮 173，操作者可以有选择地按下和释放这些按钮，每次按下和释放一个按钮就通过适配器 171 和总线 112 向微处理机发送一个独特的“按钮事件”信号。类似的指示设备如转球式指示器，可以以同样的方法予以使用；这种设备经常以与鼠标器相同的方式连接，并且有的甚至直接模拟一个鼠标器接口。因此，这种设备对实现本发明的目的来说等效于一个鼠标器，并且在以后的描述中可以代替鼠标器。

如 100 那样的 *DWS* 称作“从属的”，因为它不能独立于系统中的主处理机而实现任何有用的功能。它有时也被称作哑终端或不可编程终端。另一类终端有时被称作灵巧终端或智能工作站(*IWS*)，这类终端还包括个人计算机。两类终端的差别不能仅仅用内部处理能力的不同来表述。解释一下以往的定义，一个终端被称其为哑终端，不是由于它“无知”，而是由于它“不可教育”。这就是说，*DWS* 总是以同样的方式进行工作，而 *IWS* 或者本身直接执行用户的应用程序，或者至少根据系统的不同应用程序的性能显著地修改其操作。一般

情况下，*DWS* 除了必要的与键盘和显示监视器的连接以及与系统的通信之外，不能进行任何有意义的内部处理。因此，在监视器上显示的所有信息都必须通过通信适配器向 *RAM* 提供。同样，所有的用户击键信号都要暂时存储在 *RAM* 中，然后通过电缆向系统传送。能够降低 *DWS* 终端价格的一个主要因素是它们绝大多数都是严格地按字符字位的，即它们的显示 屏 是按从字符发生器产生的一组中取出的字符的行和列组织的。虽然不同的字符组有时可以存在内部或向下寄存，但在任何屏幕上可显示的字符和符号都是当时严格地从字符组中取出的。在最佳实 施例中 ，*DWS*100 是以 *IBM*5250 终端系列为基础的，并增加鼠标器硬件 170 和在 *ROM*117 中的用于支持鼠标器的少量编码。许多其它类型的终端可以经简单的修改，然后用在本发明中。

在最佳数据处理系统中，主机处理机包括一个实际上执行用户应用程序的主处理机和一个或多个 *WSC* 处理机。一个典型的大系统可能包括多达三十或四十个 *WSC* 。常规的 *WSC* 为多个 *DWS*100 提供信息传送和控制。典型的 *WSC* 可以控制八根电缆 140，每根电缆与七个 *DWS* 相连，每个 *DWS* 有一个地址，通过它该 *DWS* 就能被 *WSC* 识别。*WSC* 功能通常由一个硬件和软件包实现，它是被唯一识别的，且和主处理机硬件和软件包分开。*WSC* 硬件一般包括一块或多块插入插件槽的电路板，这些插件槽位于系统的主机处理机或中央电子设备的外壳之中。*WSC* 软件通过 *WSC* 硬件执

行，与主处理机中执行的软件无关。

如果 *WSC* 作为一块插件或其它可插部件装入主机处理机或主机，那么 *WSC* 可以通过与主机处理机相关的底板总线或通道直接与主处理机通信。*WSC* 还可以装在一个独立的外壳中，有自己的电源和其它设施，能够从主处理机进行摇控。在这种情况下，*WSC* 通过常规的通信线路，采用任何一种标准的通信协议，例如 *IBM Systems Network Architecture* (*SNA*)，与主处理机通信。以这种方式 *WSC* 可以控制一组或一组以上的电缆连接的 *DWS*，这些 *DWS* 可离开主处理机任意距离。（另一方面，轮询响应又限制了 *WSC* 和 *DWS* 之间的距离，通常该距离限制在 5000 英尺）。

图 2 是与多个 *DWS* 设备 100 通信的 *WSC*200 的框图。*WSC* 包括许多装在外壳 210 中的电路。在该实施例中微处理机 211 通常与一个系统总线适配器 212 相连，该适配器 212 插入与主处理机相连的系统或底板总线 220 中。*WSC* 还有一根与 *ROM*214 和 *RAM*215 相连的内部总线 213。除了包括常规的用于控制 *WSC*200 的代码之外，*ROM*214 还包括支撑鼠标器的代码，这在以后说明。*RAM*215 包括常规的数据缓冲区及其它可变更的信息，此外，*RAM*215 还包括根据本发明处理鼠标器事件的规则和数据。内部总线 213 还与通信适配器 216 相连，通过许多电缆 140 或其它常规媒介适配器 216 与许多 *DWS*100 相连。

包括来自所有相连的 *DWS* 的击键信号的数据帧被

*WSC*200 接收，并存储在 *RAM*215 中，用于以后与系统的主处理机的通信，或由 *WSC* 本身做内部处理。在最佳实施例中，*WSC* 是一个用于 *IBM AS*/400 数据处理系统的特征卡，然而如上所述，*WSC* 可以远离主处理机安装，例如它可以是 *IBM*5394 通信控制器。

图 3 是整个系统 300 的框图。主处理机 310 执行存储在主系统 *RAM*320 中的应用程序，并与底板 *I/O* 总线 330 相连。总线 330 连接多种设备 340，比如磁盘驱动器，磁带驱动器，以及通信信道。*WSC*200 经线 220 与该总线相连。*IBM AS*/400 再次可以作为该系统 300 的一个例子。

在本发明的情况下，主处理机 310 构造出一个“数据流”，用于将从应用程序得到的信息传送给适当的终端 100。每个数据流从处理机 320 传送至 *WSC*200，该数据流包括表示屏幕显示和输入区域定义界的信息。*WSC* 将该信息保留在图 2 中的用特定的 *DWS*100 标识的 *RAM*215 的一部分内。然后，*WSC* 将屏幕显示数据以一个数据流的形式传送至适当的 *DWS*。无论何时用户在 *DWS* 击键，*DWS* 就向 *WSC* 发出信号，表明在接收到从 *WSC* 发出的下一个轮询信号之后，击一次键是有效的。*WSC* 周期性地轮询所有的 *DWS* 终端，那些有击键信号的终端在那时将击键信号以称为轮询响应数据帧的数据帧的形式传送给 *WSC*。本发明将新型的数据帧加到传统的数据帧上，用于处理鼠标器事件。

图 4 是表示系统 300 中数据流的简化的框图。主处理机 310 执

行交互应用程序 410，作为其普通的操作功能的一部分。通过处理从终端发出的击键数据和产生向终端传送的显示信息，这些应用程序不时地与远处的连接在整个系统中的终端通信。当应用程序 410 需要和远处的终端 100 通信时，就为应用程序接口（*API*）调用一个程序；一种这样的程序是常规的显示数据管理程序（*DDM*）420。当终端有一个要显示的信息时，*DDM*420 根据一种常规的格式构造出一个数据流 430，并将它传送给 *WSC*200。*WSC* 有选择地与 *DWS* 设备 100 相互作用，启动适合的设备，并将数据帧 440 中要显示的信息传给被选择的设备，然后该设备进行显示。*IBM*5250 产品对数据流的特殊要求在公开发表的文件 *SA*21—9247—6“*IBM*5250 信息显示系统——功能参考手册”中有说明。

图 5 是在主机中执行的典型的应用程序的 *DWS* 显示屏的实例。字符信息在固定的行中显示，如虚构的线 510 表示的那样。在每一行内，字符在固定的列位置显示，如虚构的线 520 表示的那样。常规的屏幕如 *IBM*5250 系列显示 24 行，每行 80 列，其它屏幕可能显示更多的行和/或列。应用程序可以将某些行和列集中起来构成域 530，例如成为输入域 520。通常文本光标 540 指示下一个用户字符击键将出现的行/列位置。

包括鼠标器的设备允许纯粹的字符模式显示采用许多较新的图形用户接口（*GUI*）制图。例如，屏幕 500 可以显示 *GUI* 上卷条 550，它由一些占据了屏幕的一个字符行和列的特定字符构成，并给出常

规的图形上卷条的外形。具体地说，敞开的方块字符551的单个一列表示上卷条的底，而一个或多个填充了的方块字符552表示上卷条滑动器的当前位置。上下箭头553和554是确定的字符，它们出现在上卷条的端部。鼠标560是在屏幕500上的单独的一个具体字符，它占据一个字符的行和列，它还可以以已经在鼠标位置显示的字符图像相反的视频图像表示，或以其它醒目的方式表示。将该鼠标放在形成上卷条550的字符的不同的行/列位置并按动左鼠标器按钮，通过使在文本字符的行中显示不同字符，而使上卷条滑动器看起来移动。按动鼠标器按钮是指在一个识别的行/列位置按下按钮。一般来说，个人计算机和智能工作站采用几种类型的鼠标按钮事件表示不同的动作。除了单个的“卡搭”声，当在一段预定的时间内按下和释放两次按钮时，会发出两个“卡搭”声。在按钮按下的状态下移动鼠标器，然后在不同位置又释放按钮，这种情况被某些系统定义为“拉”。

本发明允许在一个纯粹的以字符为基础的*DWS*中出现*GUI*式制图的全部设置。除了上述的上卷条，图5还显示了其它的以字符为基础的图形，它们与*GUI*图形外观相同，如动作条或菜单条571，向下拉菜单572，选择区域573，控制按钮574，以及选择框575。

图6是表示典型的应用程序410和显示数据管理程序420的工作过程的高标准流程图600，这两个程序都在图4所示的系统主机的主处理机310中运行。图6表示同时在主处理机中运行的许多应用程序中的一个应用程序；它以常规的方式与一个连接到*WSC*上

的 *DWS* 终端相关联。当一个具体的应用程序在框 610 开始后，在框 611 进行常规的预置步骤。

其中，框 612 向 *WSC* 发送一个数据流 613，用于对与许多鼠标器按钮事件有关的含义下定义。这可以以一种判定树或一组规则的形式存储在 *WSC* 中。此外，*WSC* 可以有缺席规则，这些规则用于任何应用程序，表示在这一步骤不执行该程序。例如，可以通过实行已知的“屏幕擦去”（*screen—scraping*）或“反向工程”（*reverse—engineering*），在屏幕上找到各种符号表或区域，并且当按下鼠标器按钮时在这些区域的行/列位置中采取常规的操作。此外，每个用户可以在他们自己的 *DWS* 终端规定进行键盘操作的规则，从而为其终端的使用期定义他们自己的一组规则或提问档。

以下是用户可以通过具有一个鼠标器的 *DWS* 实现的各种类型的功能。

对显示屏上特定位置的输入区域来说，一个应用程序可以指示按动左鼠标器按钮选择该输入区域。当用户按动左按钮而鼠标正位于由输入域确定的行和列时，终端以一个轮询响应数据帧的形式将该鼠 标器事件发送给 *WSC*，并且该 *WSC* 将一个数据帧返回终端，使文本光标移动到该输入区域中鼠标的行/列位置，该 *WSC* 还向主处理机发送一个数据帧或“引起注意标识符”（*AID*），它们 包括先前由应用程序向 *WSC* 发送 、用于标识各种域的一个值。然后，应用程序向 *WSC* 发送文本字符，以便在终端上显示。这些字符能够为由该

值代表的特定区域提供鼠标器驱动上下文有关求助文本或超级文本求助文本。

应用程序可以在其向 *WSC* 输送的数据流中确定 *GUI* 式制图，如上卷条、菜单条等，并且可以接收输入的作为当鼠标位于这些图形中时发生的鼠标器按钮事件结果的数据流或 *AID*。当在滑动器轴字符 551 或轴端部的箭头 553、554 用户按动鼠标器按钮，或当在滑动器 552 本身用户按动按钮并在保持按钮按下的情况下拉滑动器，那么前面所提到的上卷条 550，例如可以使应用程序进行不同的预定操作。当 *WSC* 接收一个定义 *GUI* 式制图的输出的数据流时，*WSC* 便将其类型和图形的行/列位置存储在它的 *RAM* 中。

例如，当用户在动作条项目按下左边的鼠标器按钮时，*DWS* 便向 *WSC* 发送一个鼠标数据帧；然后 *WSC* 从它的 *RAM* 中检索图形（即“动作条”的类型，并向主处理机发送一个 *AID*。主处理机然后通过发送一个定义向下拉菜单的输出数据流作出响应。如果用户在一个如图 5 所示的选择框 575 按下左鼠标器按钮，那么 *DWS* 便向 *WSC* 发送一个数据帧，指出哪一个框被选中了。然而，此时 *WSC* 不向主处理机发送 *AID* 数据流；选择是这样做出的，即通过 *WSC* 向 *DWS* 写入一个填充的检查框字符，以代替空的检查框字符。之后，当用户在区域中的所有选择上都已经按动按钮时，便通过按下键盘上的“*Enter*（进入）”键或“*OK*”控制按钮，发出信号，表示结束。当 *WSC* 收到这个击键信号时，就将在区域中选择的所有选择框集中起来，并

同时向主处理机发送一个 *AID*。这种操作称之为“悬置操作”，以别于“立即操作”，立即操作是指一旦用户进行了一次单个的鼠标器操作，便立即发送 *AID*。

对某些应用程序来说，在定义的区域内按动鼠标器，通过将该区域内的字符向 *WSC* 移动，就可以实现一种“删除”操作；移动鼠标器并再次按动按钮，通过将一个适当的显示数据框送回终端，就可以在另一区域“填补”该字符。这种“删除与填补”操作完全可以用这种方式来处理，即对在主处理机中运行的应用程序来说是完全透明的。

即使是不用鼠标器的常规的键盘驱动应用程序也能够从本发明中受益。这是通过为 *WSC* 提供一个鼠标器操作缺席设置来实现的。例如，许多应用程序在屏幕的底部显示如“*F12*＝*CANCEL*”（“*F12*＝取消”）那样的符号表。然后，如果用户在该符号表的行和列中按动左边的鼠标器按钮（或无论哪一个规定为该功能的按钮），*WSC* 就会探测到这一事件，并向应用程序返回一个包括“*F12*”击键信息的数据流，就好像用户已经按下键盘上标有“*F12*”的功能键那样。可以从主处理机中的显示数据管理程序或一些其它的程序源中将缺席操作永久装入 *WSC* 的 *ROM*，或加载到 *WSC* 的 *RAM*。这叫“屏幕擦去模式”（*screen—scraping mode*）。

任何已知类型的逻辑判定树确定当进行鼠标器操作时，什么功能（如果有的话）将被实现。现说明如下：

>>如果按下左鼠标器按钮，

>> 如果按下“*shift*(移动)”键，则将文本光标移向鼠标器指针的行/列位置。

>> 如果删除操作悬置，那么进行填补操作，并将删除操作悬置标记复位(看下面)。

>> 如果鼠标在一个动作条、向下拉菜单、控制按钮或其它 *GUI* 式图形上，那么选择该图形，并向应用程序发送一个 *AID*，标识该图形。

>>如果鼠标在一个垂直的上卷条上，

>> 如果在上卷条字符的上箭头上，则向应用程序发送上卷一行的指令。

>> 如果在下箭头上，则向应用程序发送下卷一行的指令。

>> 如果在滑动器上的上卷条轴上，则单独发送上卷指令(即与应用程序有关的上卷量)。

>>如果在滑动器下的轴上，则单独发送下卷指令。

>>如果鼠标在一个水平的上卷条上，

>>如果在左箭头字符上，则向应用程序发送向左卷一列的指令。

>>如果在右箭头上，则向应用程序发送向右卷一列的指令。

>>如果在滑动器左边的轴上，则单独发送左卷指令(即

与应用程序有关的上卷量)。

>>如果在滑动器右边的轴上,则单独发送右卷指令。

>>如果鼠标在一个选择区域,

>>选择在鼠标位置上的项目。

>> 如果菜单是一个单选菜单,则向下选择所有的其它项目。

>> 如果该区域是定义成一个"左按钮按下可选择区域"的输入域,则将文本光标移向鼠标器指针所在的行和列,并向主处理机发送一个应用程序确定的 *AID*。

>> 检查是否存在反向工程(屏幕擦去)匹配,如果存在,则使屏幕和规则相反,鼠标位置和定义的功能相反,以及向应用程序传送一个或多个适当的击键信号。

>> 否则,通过将该鼠标位置上的字符图像反向并记录该字符的位置,开始删除操作。

>>如果释放左鼠标器按钮,

>> 如果已经开始删除操作,并且鼠标指针位于删除操作开始位置或超出该位置的屏幕位置,则完成删除操作,设置一个删除操作悬置标记。

>> 否则,将删除开始标记复位,如果需要则取消反向图像字符,并不再采取进一步操作。

在框 620 实行应用程序 600 中的常规功能。当框 631 探测到应

用程序需要向*WSC*发送一个常规的数据流时，在630便出现送到*WSC*去的输出。在框632对数据打包，并发送该数据，如633所示。在框640进行更常规的处理。在650出现来自*WSC*的输入。当框651确定，可以从*WSC*得到数据流653，则在框652接收该数据流。只要应用程序在运行过程中，框660就实现进一步的功能，并将控制返回框620。主要的功能框610—660实际上可以以任何次序执行。

图7是表示在执行图6所示的如600那样的应用程序中*WSC*的运行过程的流程图700。*WSC*完成多种操作700，除了某些常规的内务操作以外，每个*WSC*还与一个*DWS*终端相连，这些未在图中画出。

框710从图6的主机接收数据流613和633。框711探测是否能从主机得到数据流。如果所得到的数据流是一个如633那样的常规数据流，则在框712进行任何必要的操作。对一个鼠标器按钮事件确定的数据流613来说，框713将代表该事件的信息存在*WSC*，从而*WSC*控制程序600能够利用它们；如前所述，鼠标器按钮事件可以作为一个判定树即一组规则来存储，或者以其它常规的结构，比如具有可寻址的入口表格来存储。

在框720实现常规的功能。为了在730修改终端显示，在框731探测一个是否需要修改的信号，该信号可由框720或一些其它功能产生。然后在框732形成一个常规的数据帧733并向特定的*DWS*

终端发送，该终端受到该控制器程序700的控制。*WSC*周期性地向终端发送轮询信号740。这时，框741使框742产生一个数据帧743，并向该终端发送。

框750处理从终端得到的轮询响应。如果在给定的时间间隔内框751没有探测到一个响应信号，则框752进行常规处理。如果响应是一个包括字符或其它标准按键（或其它一些常规条件）的数据帧753，则在框754处理它。

但是如果框751检测出在数据帧753中有一个鼠标器按钮事件，则框755采用由框713存储的规定，选择合适的操作，如虚线701所示。在框756进行相应的处理，如前面结合图6中的表所描述的那样。这些操作中的某些可能需要向终端传送一个数据帧，如虚线702所示；有些可能需要向主机传送一个数据流，如虚线703所示；而有些操作仅需要在*WSC*中进行，如虚线704所示。

也许是由于从框756来了一个鼠标器按钮事件703，反正无论何时只要框761检测到必须发送一个数据流，框760都向主机发送一个数据流。在框762以常规方式形成和发送该数据流653。

框770在*WSC*中进行其它的处理，可能对鼠标器按钮事件作出响应，如虚线704所示。控制过程返回到框710。*WSC*控制程序700基本上是中断驱动；主要的框710—770可以在任何时间序列执行。

图8是表示在图1的一个*DWS*终端100中的运行情况流程图

800。

框 810 实现常规的功能。如果在 820 发现有一个操作员输入信号，则在框 821 确定它为何种操作。如果输入信号是一个需要向 *WSC* 或主机传送的按键信号，那么在框 823 产生一个向 *WSC* 发送的数据帧。如果该输入只在终端产生严格的局部作用，那么框 824 以常规方式处理该输入。然而，最佳实施例确实为框 824 增加了一个简便易行的程序。用户可以按动一个唯一的键序列为鼠标器设置双“卡搭”声时间间隔。当检测到该序列时，框 824 在显示屏的特定位置显示一个数，如“0.5”。用户可以按动光标上/下键（或其它键）使该数增加或减少一个预定的量，如 0.1 秒。当用户按下“*Enter*”键或其它键表示操作完成时，框 824 将最终的数作为按动按钮之间的最大秒数予以存储，这将用作双“卡搭”声。

如果操作者输入的是一个鼠标器事件，那么框 821 向框 825 发出信号，确定该事件的种类。如果该事件是一个位置事件，那么框 826 移动在屏幕上显示的鼠标，并记录在屏幕上的新的行/列位置。（此外，如下所述，如果 *WSC* 已经指出周期性鼠标移动事件将要被传送，并且该周期已经经过，那么当前的鼠标行/列位置以一个数据帧的形式向 *WSC* 传送）。如果该鼠标事件是一个按钮事件，那么框 827 确定它是哪一类型的按钮事件。本执行程序确定三种类型的按钮事件。当按下按钮时，发生的是一个“向下”事件。当释放按钮时，发生的是一个“向上”事件。当在一个预定时间间隔内按下并释放按

钮时，发生的是一个双“卡搭”声事件。当向上和向下事件发生时，便将它们发送，并且还与时间标记一道存在 *DWS* 存储器中。如果框 828 检测到序列向下/向上/向下，其中第二个向下的标记小于某一值并且鼠标器位置不变化，那么框 827 指出这是一个双“卡搭”声事件。这样，一个双“卡搭”声发送四个鼠标事件帧：向下、向上、双“卡搭”声和向上；当后来按钮从双“卡搭”声状态释放时，发送最后的向上事件。*WSC* 规则必须设计成能对这种序列正确解码；通常，任何屏幕位置可以由按钮向下事件或双“卡搭”声事件选择，但不能由二者同时选择。（然而，其它实施例能在终端处理这种序列，并仅仅发送一个事件数据帧。）然后框 828 产生一个数据帧，指出事件的类型是什么，是哪一个按钮，以及在按钮事件发生时鼠标的行/列位置。下表表示一个鼠标器按钮事件数据帧的格式：

区域：	指令	按钮	鼠标行列	
值：	向下	左	1	1
	向上	中	…	…
	双“卡搭”声	右	24	80
	移动			

此外，当被 *WSC* 请求时，一个确定的“移动”指令值使 *WSC* 不管按钮区域，而仅仅向 *WSC* 报告鼠标行和列的位置。

在 *IBM5250* 协议中，为发送从电缆上地址为 *N* 的一个 *DWS* 终端得到的鼠标器按钮事件所用的完全序列是：

>>*WSC* 向所有终端发送一个轮询数据帧。

>> 终端 *N* 返回一个包括该终端状态的轮询响应数据帧。这个数据帧的两个字节中有一位是“引起注意的状态”位，对这种类型的终端来说，这个位迄今未被使用。本发明设置这个位为指示鼠标器按钮事件可以获得。

>> *WSC* 发送一个在终端 *N* 中被分配了一个新地址的读指令数据帧，不管它有什么数据都请求发送。

>>终端 *N* 传送上述按钮事件数据帧。

因为在轮询响应帧中的“引起注意的状态”位的值已经告诉 *WSC*，鼠标按钮事件正悬置，所以按钮事件帧本身不需要包括任何标识用的专门编码。

虽然在图 8 中并未示出，但一些实施例还是希望即使在没有按钮事件的情况下，也能将鼠标器位置变化发送给 *WSC*。如上面指出的那样，这将导致在终端和 *WSC* 之间出现不能接受的高的数据传输速率，特别在 *WSC* 控制许多终端的情况下更是如此。然而，一些应用程序和/或 *WSC* 操作可能对某些操作请求短时间间隔的运动信息。例如，对拉或拉/压功能来说，*WSC* 检测到左按钮按下，以及一个确定目标如上卷条滑动器的行/列位置，并且向 *DWS* 发送一个数据帧，请求框 828 每 50 毫秒发送一个唯一的鼠标器位置数据帧，直到按钮释放。用户释放按钮之后，*WSC* 接收到表示释放的数据帧，这使得 *WSC* 向 *DWS* 发送另一个唯一的数据帧，请求不再发送运动

事件。

框 830 处理从 *WSC* 得到的数据帧。当 831 检测到接收了一个数据帧，则如果有一个数据帧如 733 要求，就在框 832 修改该显示。如果数据帧是一个轮询信号如 743，则框 833 发送一个返回数据帧 753。该数据帧可能已经是一个由框 824 产生的常规数据帧，如虚线 801 所示，或是从框 827 来的一个鼠标器按钮事件数据帧，如虚线 802 所示。

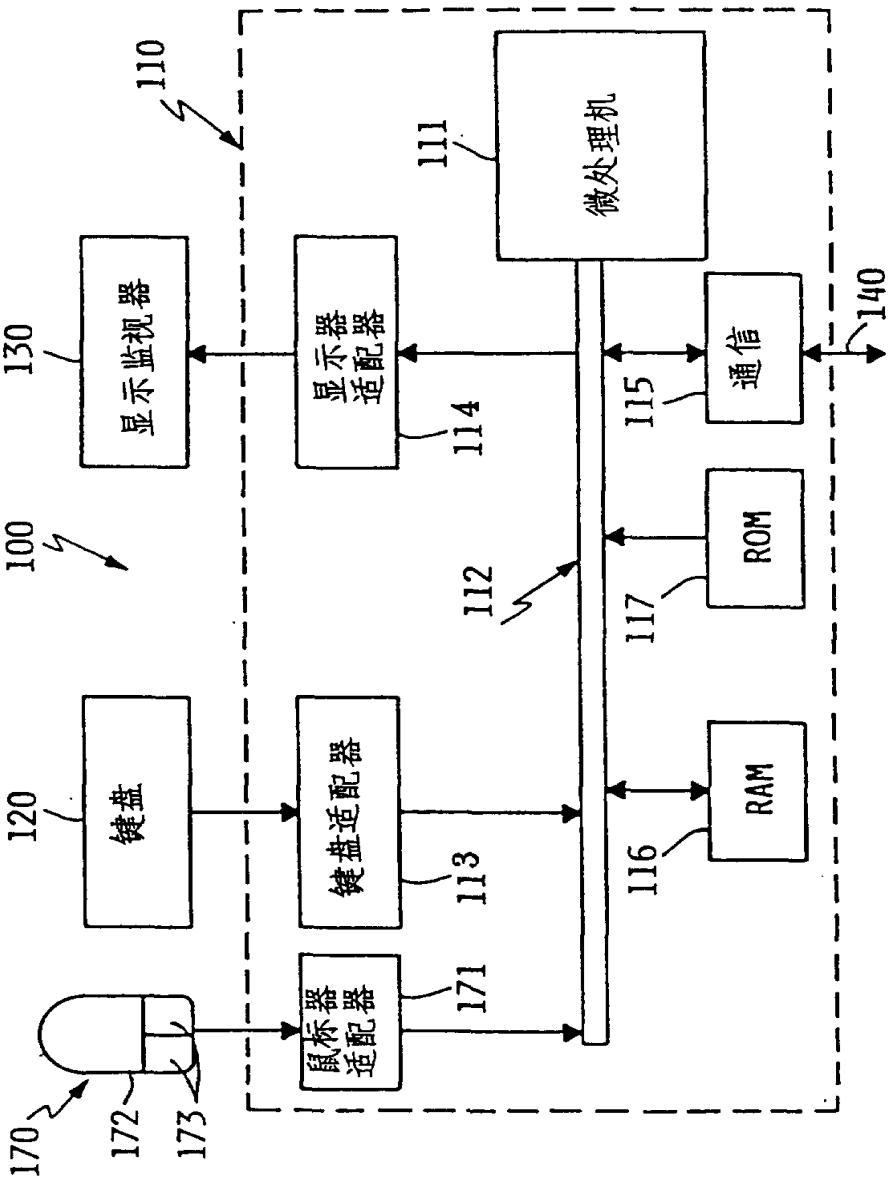

110
111
微处理机
130
显示监视器
显示器适配器
114
115
通信
140
100
112
ROM
117
120
键盘
键盘适配器
113
RAM
116
鼠标器适配器
171
170
172
173

图. 1

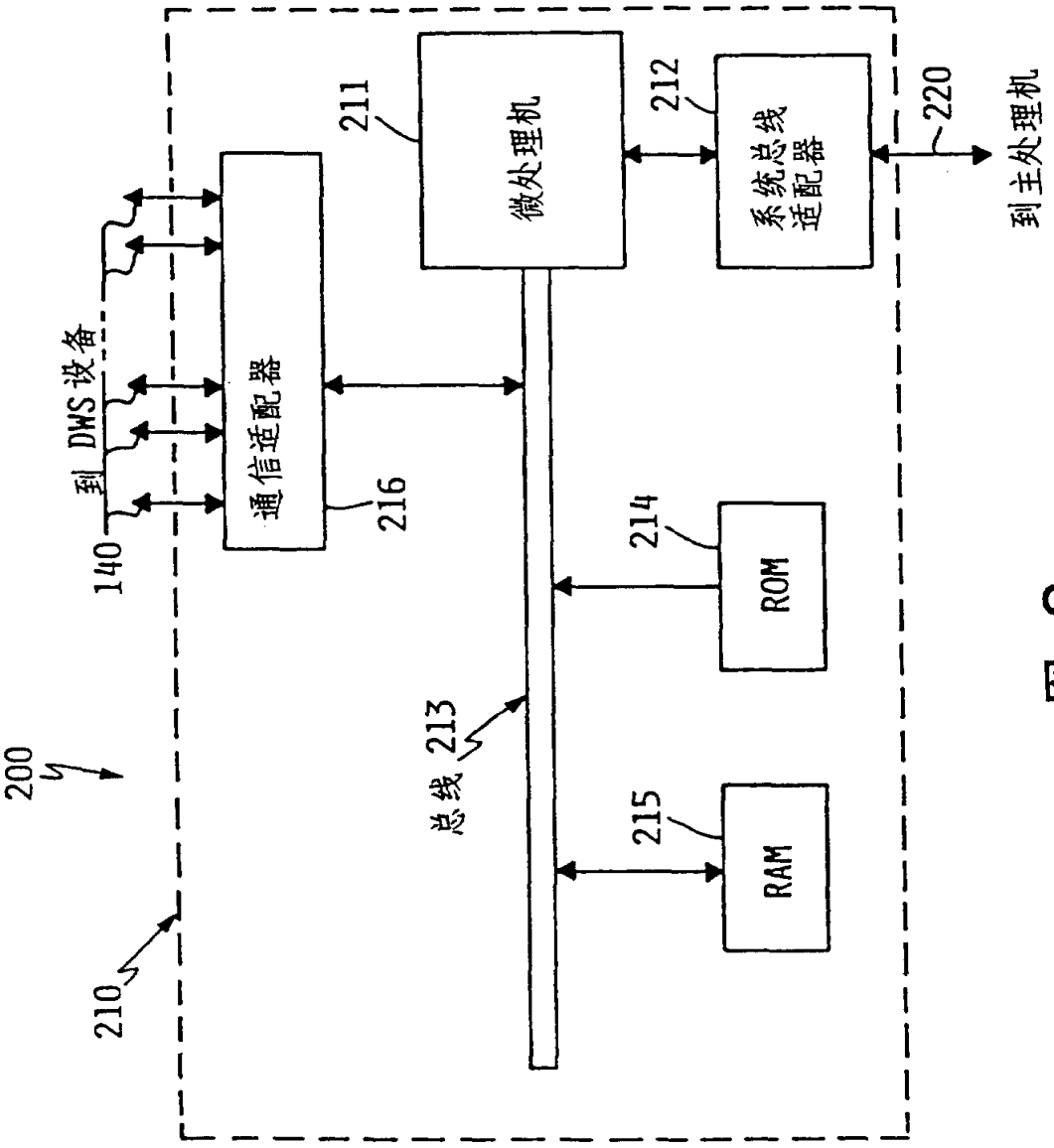

200
210
到 DWS 设备
140
通信适配器
216
211
微处理机
212
系统总线
适配器
220
到主处理机
总线 213
214
ROM
215
RAM

图. 2

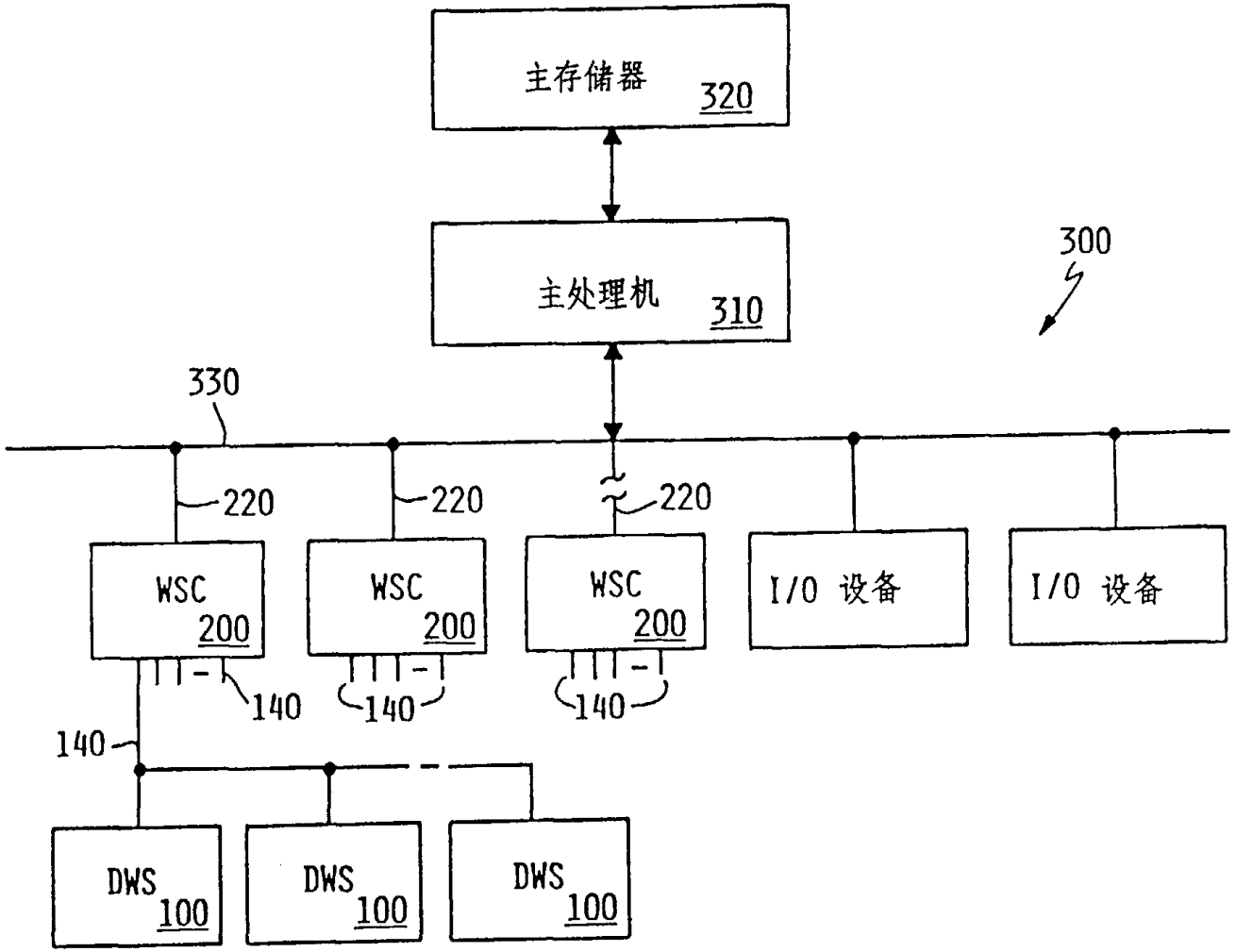

主存储器
320
主处理机
310
300
330
220
220
220
WSC
200
WSC
200
WSC
200
I/O 设备
I/O 设备
140
140
140
140
DWS
100
DWS
100
DWS
100

图. 3

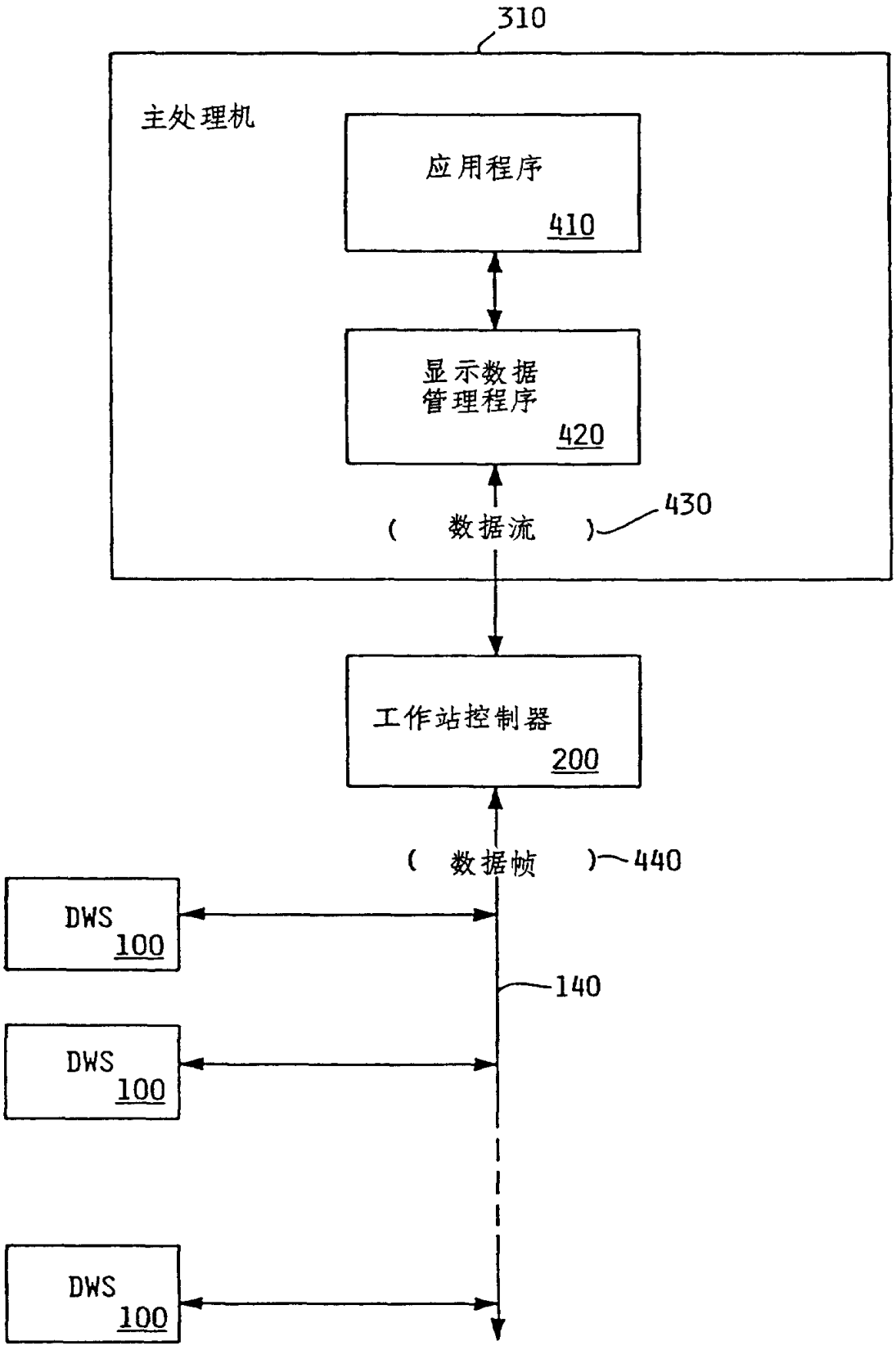

310
主处理机
应用程序
410
显示数据
管理程序
420
(数据流)
430
工作站控制器
200
(数据帧)
440
DWS
100
DWS
100
DWS
100
140

图. 4

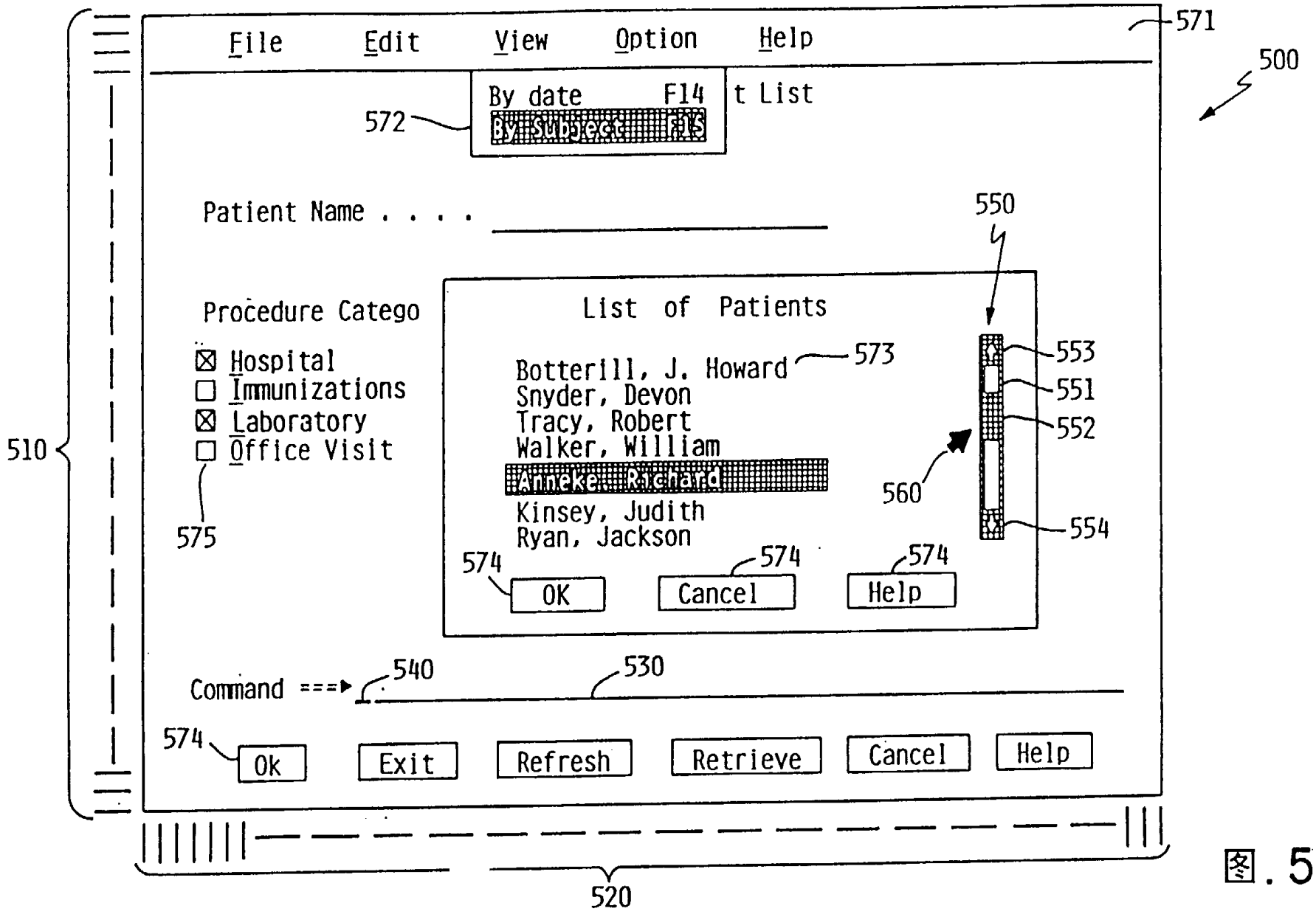

File
Edit
View
Option
Help
571
500
By date F14
572
By Subject F15
t List
Patient Name . . . .
550
Procedure Catego
List of Patients
Hospital
Immunizations
Laboratory
Office Visit
575
Botterill, J. Howard
573
Snyder, Devon
Tracy, Robert
Walker, William
Anneke, Richard
Kinsey, Judith
Ryan, Jackson
553
551
552
554
560
574
OK
Cancel
Help
510
Command ===>
540
530
Ok
Exit
Refresh
Retrieve
Cancel
Help
520

图. 5

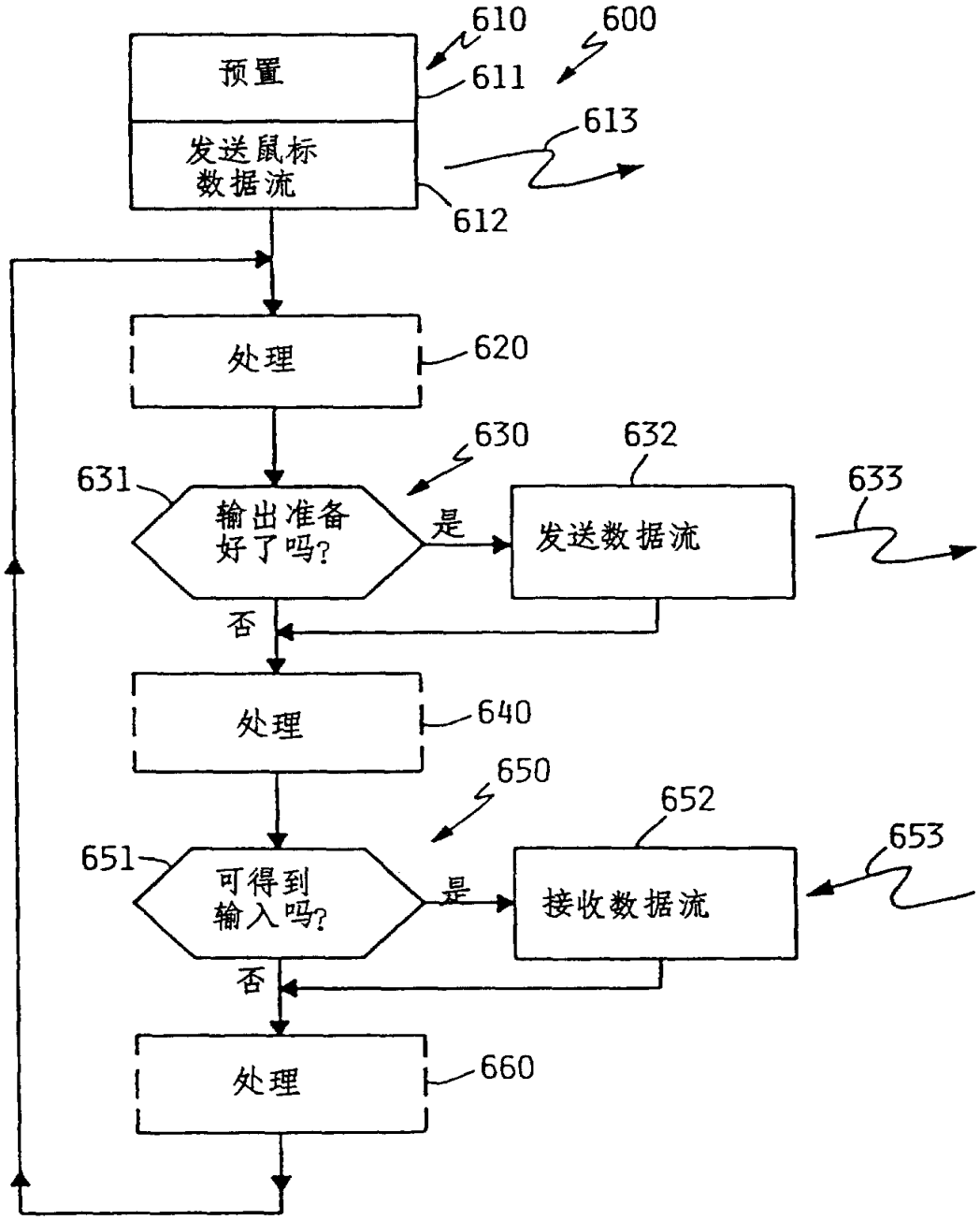

610
600
预置
611
613
发送鼠标
数据流
612
处理
620
630
632
631
输出准备
好了吗?
是
发送数据流
633
否
处理
640
650
652
651
可得到
输入吗?
是
接收数据流
653
否
处理
660

图.6

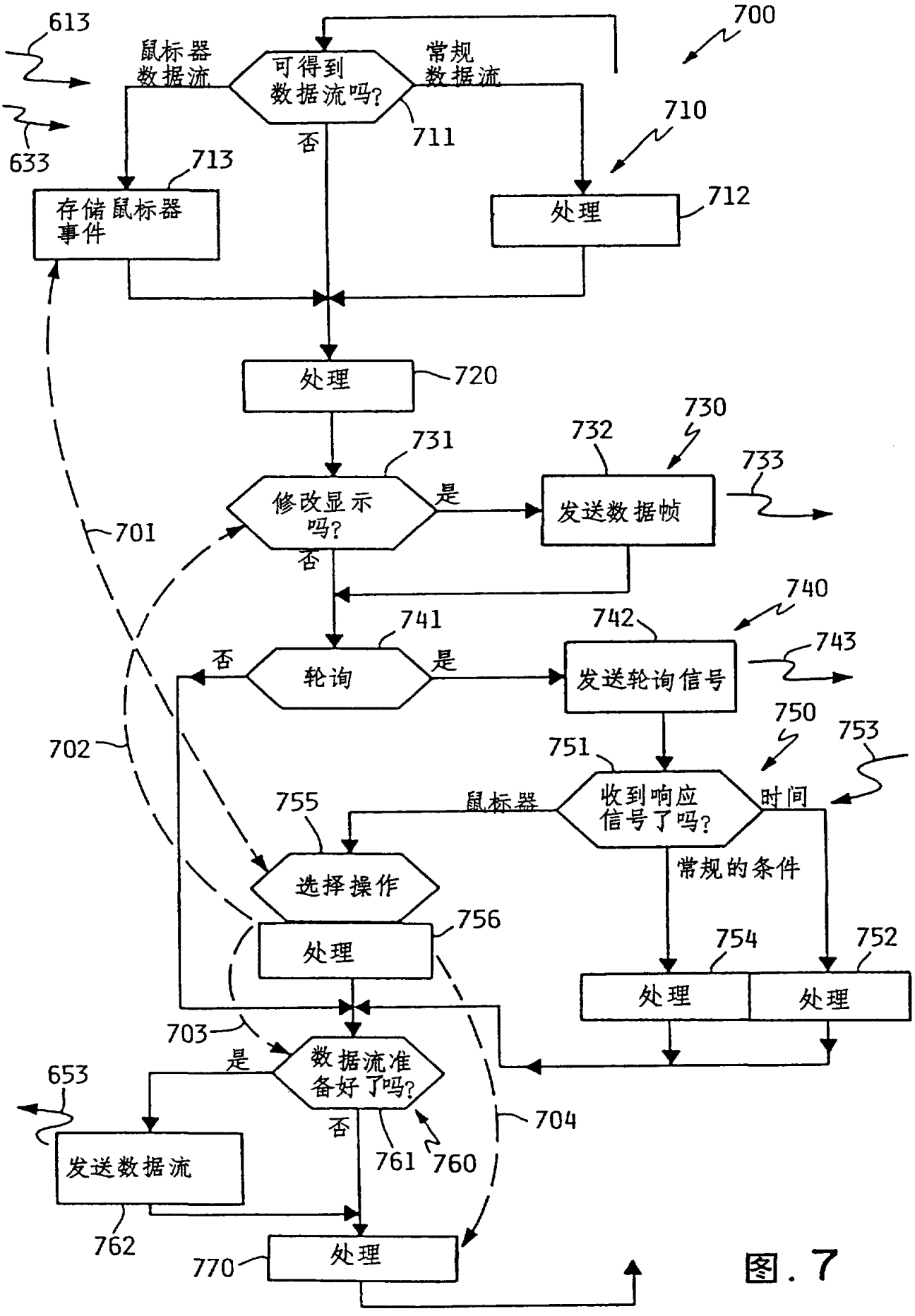

613
700
鼠标器
数据流
可得到
数据流吗?
常规
数据流
711
710
633
713
否
存储鼠标器
事件
处理
712
处理
720
730
731
732
733
修改显示
吗?
是
发送数据帧
701
否
740
741
742
743
否
轮询
是
发送轮询信号
750
753
751
702
收到响应
信号了吗?
时间
鼠标器
755
常规的条件
选择操作
756
处理
754
752
处理
处理
703
是
数据流准
备好了吗?
653
704
否
761
760
发送数据流
762
处理
770

图. 7

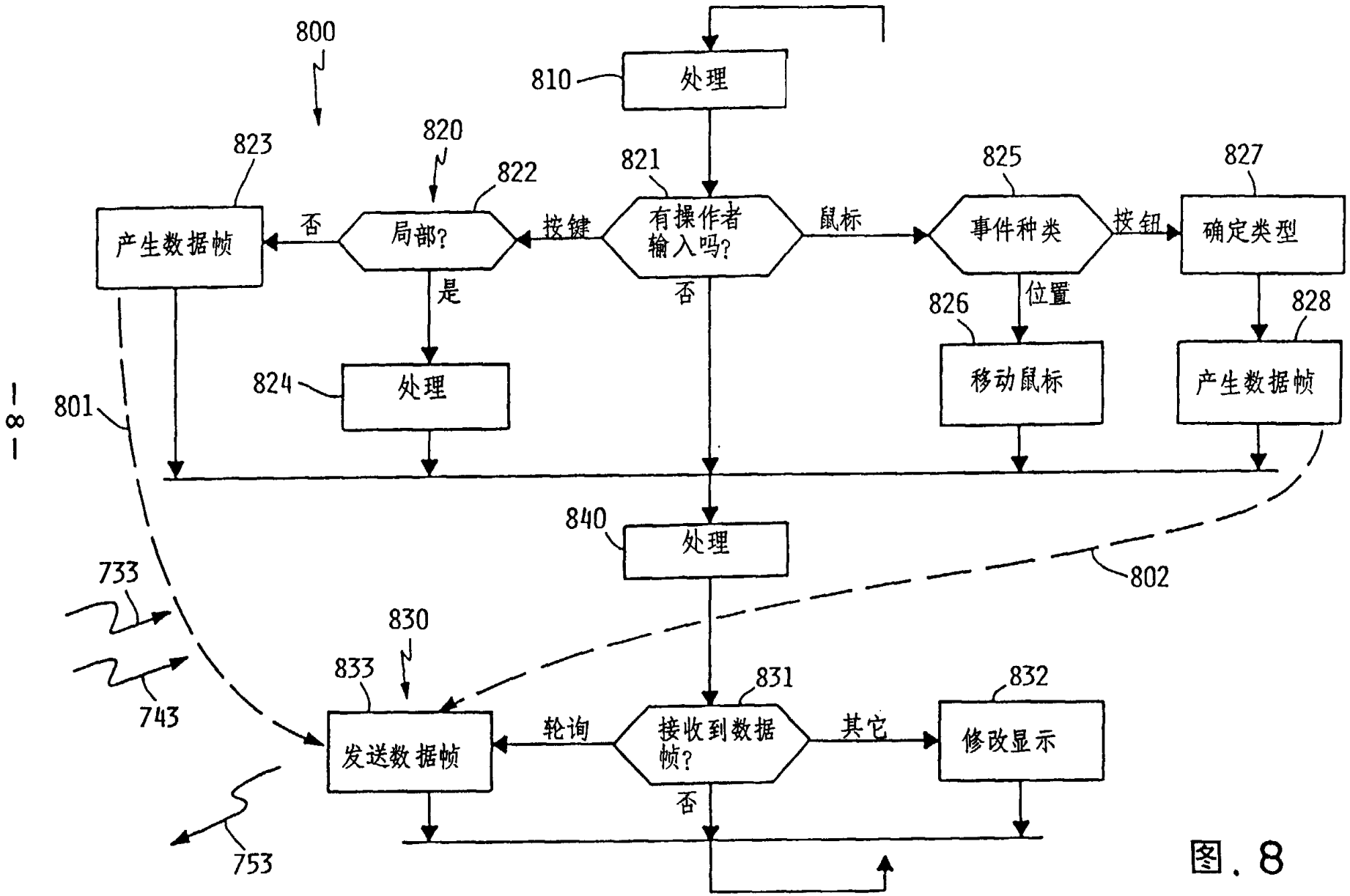

800
810
处理
820
823
822
821
825
827
产生数据帧
否
局部?
按键
有操作者输入吗?
鼠标
事件种类
按钮
确定类型
是
否
位置
826
828
824
处理
移动鼠标
产生数据帧
801
840
处理
802
733
830
833
831
832
743
轮询
接收到数据帧?
其它
发送数据帧
修改显示
否
753

图. 8