

- 53 -

Segment Local Area Network" filed on January 14, 1991 (Attorney Docket No. 13283-NE.APP), incorporated herein by reference.

5 In normal operation, each station in the network is monitored by a single Monitor that is located on its local segment. The initial determination of the Monitor responsible for a station is based on the results of the autotopology mechanism. The user may override this initial default if required.

10 The user is informed of new stations appearing on any segment in the network via the alarm mechanism. As for other alarms, the user may select whether stations appearing on and disappearing from the network segment generate alarms and may modify the times used in the  
15 aging algorithms. When a new node alarm occurs, the user must add the new alarm to the map using the SNM tools. In this manner, the SNM system becomes aware of the nodes.

The sequence of events following the detection of  
20 a new node is:

1. the location of the node is determined automatically for the user.
2. the Monitor generates an alarm for the user indicating the new node and providing  
25 some or all of the following information:
  - mac address of node
  - ip address of node
  - segment that the node is believed to be  
30 located on
  - Monitor to be responsible for the node
3. the user must select the segment and add the node manually using the SNM editor

- 54 -

4. The update to the SNM database will be detected and the file reread. The Workstation database is reconstructed and the parse control records for the Monitors updated if required.
5. The Monitor responsible for the new node has its parse control record updated via SNMP set request(s).

An internal record of new nodes is required for the autotopology. When a new node is reported by a Network Monitor, the Management Workstation needs to have the previous location information in order to know which Network Monitors to involve in autotopology. For example, two nodes with the same IP address may exist in separate segments of the network. The history makes possible the correlation of the addresses and it makes possible duplicate address detection.

Before a new Monitor can communicate with the Management Workstation via SNMP it needs to be added to the SNM system files. As the SNM files are cached in the database, the file must be updated and the SNM system forced to reread it.

Thus, on the detection of a new Monitor the following events need to occur in order to add the Monitor to the Workstation:

1. The Monitor issues a trap to the Management Workstation software and requests code to be loaded from the Sun Microsystems boot/load server.
2. The code load fails as the Monitor is not known to the unix networking software at this time.
3. The Workstation confirms that the new Monitor does not exceed the configured system limits (e.g. 5 Monitors per

- 55 -

- Workstation) and terminates the initialization sequence if limits are exceeded. An alarm is issued to the user indicating the presence of the new Monitor and whether it can be supported.
- 5           4. The user adds the Monitor to the SNMP.HOSTS file of the SNM system, to the etc/hosts file of the Unix networking system and to the SNM map.
- 10          5. When the files have been updated the user resets the Monitor using the set tool (described later).
6. The Monitor again issues a trap to the Management Workstation software and
- 15           requests code to be loaded from the Sun boot/load server.
7. The code load takes place and the Monitor issues a trap requesting data from the Management Workstation.
- 20          8. The Monitor data is issued using SNMP set requests.

Note that on receiving the set request, the SNMP proxy rereads in the (updated) SNMP.HOSTS file which now includes the new Monitor. Also note that the SNMP hosts

25 file need only contain the Monitors, not the entire list of nodes in the system.

9. On completion of the set request(s) the Monitor run command is issued by the Workstation to bring the Monitor on line.
- 30           The user is responsible for entering data into the SNM database manually. During operation, the Workstation monitors the file write date for the SNM database. When this is different from the last date read, the SNM database is reread and the Workstation database
- 35 reconstructed. In this manner, user updates to the SNM

- 56 -

database are incorporated into the Workstation database as quickly as possible without need for the user to take any action.

When the Workstation is loaded, the database is  
5 created from the data in the SNM file system (which the user has possibly updated). This data is checked for consistency and for conformance to the limits imposed by the Workstation at this time and a warning is generated to the user if any problems are seen. If the data errors  
10 are minor the system continues operation; if they are fatal the user is asked to correct them and Workstation operation terminates.

The monitoring functions of the Management Workstation are provided as an extension to the SNM  
15 system. They consist of additional display tools (i.e., summary tool, values tool, and set tool) which the user invokes to access the Monitor options and a Workstation event log in which all alarms are recorded.

As a result of the monitoring process, the Monitor  
20 makes a large number of statistics available to the operator. These are available for examination via the Workstation tools that are provided. In addition, the Monitor statistics (or a selected subset thereof) can be made visible to any SNMP manager by providing it with  
25 knowledge of the extended MIB. A description of the statistics maintained are described elsewhere.

Network event statistics are maintained on a per network, per segment and per node basis. Within a node, statistics are maintained on a per address (as  
30 appropriate to the protocol layer - IP address, port number, ...) and per connection basis. Per network statistics are always derived by the Workstation from the per segment variables maintained by the Monitors. Subsets of the basic statistics are maintained on a node  
35 to node and segment to segment basis.

- 57 -

If the user requests displays of segment to segment traffic, the Workstation calculates this data as follows. The inter segment traffic is derived from the node to node statistics for the intersecting set of  
5 nodes. Thus, if segment A has nodes 1, 2, and 3 and segment B has nodes 20, 21, and 22, then summing the node to node traffic for

1 -> 20,21,22  
2 -> 20,21,22  
10 3 -> 20,21,22

produces the required result. On-LAN/off-LAN traffic for segments is calculated by a simply summing node to node traffic for all stations on the LAN and then subtracting this from total segment counts.

15 Alarms are reported to the user in the following ways:

1. Alarms received are logged in a Workstation log.
2. The node which the alarm relates to is highlighted on the map.
- 20 3. The node status change is propagated up through the (map) hierarchy to support the case where the node is not visible on the screen. This is as provided by SNM.

#### Summary Tool

25 After the user has selected an object from the map and invokes the display tools, the summary tool generates the user's initial screen at the Management Workstation. It presents a set of statistical data selected to give an overview of the operational status of the object (e.g., a  
30 selected node or segment). The Workstation polls the Monitor for the data required by the Summary Tool display screens.

The Summary Tool displays a basic summary tool screen such as is shown in Fig. 18. The summary tool  
35 screen has three panels, namely, a control panel 602, a

- 58 -

values panel 604, and a dialogs panel 606. The control panel includes the indicated mouse activated buttons. The functions of each of the buttons is as follows. The file button invokes a traditional file menu. The view  
5 button invokes a view menu which allows the user to modify or tailor the visual properties of the tool. The properties button invokes a properties menu containing choices for viewing and sometimes modifying the properties of objects. The tools button invokes a tools  
10 menu which provides access to the other Workstation tools, e.g. Values Tool.

The Update Interval field allows the user to specify the frequency at which the displayed statistics are updated by polling the Monitor. The Update Once  
15 button enables the user to retrieve a single screen update. When the Update Once button is invoked not only is the screen updated but the update interval is automatically set to "none".

The type field enables the user to specify the  
20 type of network objects on which to operate, i.e., segment or node.

The name button invokes a pop up menu containing an alphabetical list of all network objects of the type selected and apply and reset buttons. The required name  
25 can then be selected from the (scrolling) list and it will be entered in the name field of the summary tool when the apply button is invoked. Alternatively, the user may enter the name directly in the summary tool name field.

30 The protocol button invokes a pop up menu which provides an exclusive set of protocol layers which the user may select. Selection of a layer copies the layer name into the displayed field of the summary tool when the apply operation is invoked. An example of a protocol  
35 selection menu is shown in Fig. 19. It displays the

- 59 -

available protocols in the form of a protocol tree with multiple protocol families. The protocol selection is two dimensional. That is, the user first selects the protocol family and then the particular layer within that family.

As indicated by the protocol trees shown in Fig. 19, the capabilities of the Monitor can be readily extended to handle other protocol families. The particular ones which are implemented depend upon the needs of the particular network environment in which the Monitor will operate.

The user invokes the apply button to indicate that the selection process is complete and the type, name, protocol, etc. should be applied. This then updates the screen using the new parameter set that the user selected. The reset button is used to undo the selections and restore them to their values at the last apply operation.

The set of statistics for the selected parameter set is displayed in values panel 604. The members of the sets differ depending upon, for example, what protocol was selected. Figs. 20a-g present examples of the types of statistical variables which are displayed for the DLL, IP, UDP, TCP, ICMP, NFS, and ARP/RARP protocols, respectively. The meaning of the values display fields are described in Appendix I, attached hereto.

Dialogs panel 606 contains a display of the connection statistics for all protocols for a selected node. Within the Management Workstation, connection lists are maintained per node, per supported protocol. When connections are displayed, they are sorted on "Last Seen" with the most current displayed first. A single list returned from the Monitor contains all current connection. For TCP, however, each connection also contains a state and TCP connections are displayed as

- 60 -

Past and Present based upon the returned state of the connection. For certain dialogs, such as TCP and NFS over UDP, there is an associated direction to the dialog, i.e., from the initiator (source) to the receiver (sink).  
5 For these dialogs, the direction is identified in a DIR. field. A sample of information that is displayed in dialogs panel 606 is presented in Fig. 21 for current connections.

#### Values Tool

10 The values tool provides the user with the ability to look at the statistical database for a network object in detail. When the user invokes this tool, he may select a basic data screen containing a rate values panel 620, a count values panel 622 and a protocols seen panel  
15 626, as shown in Fig. 22, or he may select a traffic matrix screen 628, as illustrated in Fig. 23.

In rate values and count values panels 620 and 622, value tools presents the monitored rate and count statistics, respectively, for a selected protocol. The  
20 parameters which are displayed for the different protocols (i.e., different groups) are listed in Appendix II. In general, a data element that is being displayed for a node shows up in three rows, namely, a total for the data element, the number into the data element, and  
25 the number out of the data element. Any exceptions to this are identified in Appendix II. Data elements that are displayed for segments, are presented as totals only, with no distinction between Rx and Tx.

When invoked the Values Tool displays a primary  
30 screen to the user. The primary screen contains what is considered to be the most significant information for the selected object. The user can view other information for the object (i.e., the statistics for the other parameters) by scrolling down.



- 61 -

The displayed information for the count values and rate values panels 620 and 622 includes the following. An alarm field reports whether an alarm is currently active for this item. It displays as "\*" if active alarm is present. A Current Value/Rate field reports the current rate or the value of the counter used to generate threshold alarms for this item. This is reset following each threshold trigger and thus gives an idea of how close to an alarm threshold the variable is. A Typical Value field reports what this item could be expected to read in a "normal" operating situation. This field is filled in for those items where this is predictable and useful. It is maintained in the Workstation database and is modifiable by the user using the set tool. An Accumulated Count field reports the current accumulated value of the item or the current rate. A Max Value field reports the highest value recently seen for the item. This value is reset at intervals defined by a user adjustable parameter (default 30 minutes). This is not a rolling cycle but rather represents the highest value since it was reset which may be from 1 to 30 minutes ago (for a rest period of 30 minutes). It is used only for rates. A Min Value field reports the lowest value recently seen for the item. This operates in the same manner as Max Value field and is used only for rates.

A Percent (%) field reports only for the following variables:

off seg counts:  
     100(in count / total off seg count)  
     100(out count / total off seg count)  
     100(transit count / total off seg count)  
     100(local count / total off seg count)  
 off seg rates  
     100(transit rate / total off seg rate), etc.  
 protocols

- 62 -

100(frame rate this protocol / total frame  
rate)

On the right half of the basic display, there the  
following additional fields: a High Threshold field and a  
5 Sample period for rates field.

#### Set Tool

The set tool provides the user with the ability to  
modify the parameters controlling the operation of the  
Monitors and the Management Workstation. These  
10 parameters affect both user interface displays and the  
actual operation of the Monitors. The parameters which  
can be operated on by the set tool can be divided into  
the following categories: alarm thresholds, monitoring  
control, segment Monitor administration, and typical  
15 values.

The monitoring control variables specify the  
actions of the segment Monitors and each Monitor can have  
a distinct set of control variables (e.g., the parse  
control records that are described elsewhere). The user  
20 is able to define those nodes, segments, dialogs and  
protocols in which he is interested so as to make the  
best use of memory space available for data storage.  
This mechanism allows for load sharing, where multiple  
Monitors on the same segment can divide up the total  
25 number of network objects which are to be monitored so  
that no duplication of effort between them takes place.

The monitor administration variables allow the  
user to modify the operation of the segment Monitor in a  
more direct manner than the monitoring control variables.  
30 Using the set tool, the user can perform those operations  
such as reset, time changes etc. which are normally the  
prerogative of a system administrator.

Note that the above descriptions of the tools  
available through the Management Workstation are not  
35 meant to imply that other choices may not be made

- 63 -

regarding the particular information which is displayed and the manner in which it is displayed.

Adaptively Setting Network Monitor Thresholds:

The Workstation sets the thresholds in the Network  
5 Monitor based upon the performance of the system as  
observed over an extended period of time. That is, the  
Workstation periodically samples the output of the  
Network Monitors and assembles a model of a normally  
functioning network. Then, the Workstation sets the  
10 thresholds in the Network Monitors based upon that model.  
If the observation period is chosen to be long enough and  
since the model represents the "average" of the network  
performance over the observation period, temporary  
undesired deviations from normal behavior are smoothed  
15 out over time and model tends to accurately reflect  
normal network behavior.

Referring the Fig. 24, the details of the training  
procedure for adaptively setting the Network Monitor  
thresholds are as follows. To begin training, the  
20 Workstation sends a start learning command to the Network  
Monitors from which performance data is desired (step  
302). The start learning command disables the thresholds  
within the Network Monitor and causes the Network Monitor  
to periodically send data for a predefined set of network  
25 parameters to the Management Workstation. (Disabling the  
thresholds, however, is not necessary. One could have  
the learning mode operational in parallel with monitoring  
using existing thresholds.) The set of parameters may be  
any or all of the previously mentioned parameters for  
30 which thresholds are or may be defined.

Throughout the learning period, the Network  
Monitor sends "snapshots" of the network's performance to  
the Workstation which, in turn, stores the data in a  
performance history database 306 (step 304). The network  
35 manager sets the length of the learning period.

- 64 -

Typically, it should be long enough to include the full range of load conditions that the network experiences so that a representative performance history is generated. It should also be long enough so that short periods of  
5 overload or faulty behavior do not distort the resulting averages.

After the learning period has expired, the network manager, through the Management Workstation, sends a stop learning command to the Monitor (step 308). The Monitor  
10 ceases automatically sending further performance data updates to the Workstation and the Workstation processes the data in its performance history database (step 310). The processing may involve simply computing averages for the parameters of interest or it may involve more  
15 sophisticated statistical analysis of the data, such as computing means, standard deviations, maximum and minimum values, or using curve fitting to compute rates and other pertinent parameter values.

After the Workstation has statistically analyzed  
20 the performance data, it computes a new set of thresholds for the relevant performance parameters (step 312). To do this, it uses formulas which are appropriate to the particular parameter for which a threshold is being computed. That is, if the parameter is one for which one  
25 would expect to see wide variations in its value during network monitoring, then the threshold should be set high enough so that the normal expected variations do not trigger alarms. On the other hand, if the parameter is of a type for which only small variations are expected  
30 and larger variations indicate a problem, then the threshold should be set to a value that is close to the average observed value. Examples of formulae which may be used to compute thresholds are:

\* Highest value seen during learning period;

- 65 -

- \* Highest value seen during learning period + 10%;
- \* Highest value seen during learning period + 50%;
- 5 \* Highest value seen during learning period + user-defined percent;
- \* Any value of the parameter other than zero;
- \* Average value seen during learning period + 50%; and
- 10 \* Average value seen during learning period + user-defined percent.

As should be evident from these examples, there is a broad range of possibilities regarding how to compute a particular threshold. The choice, however, should  
15 reflect the parameter's importance in signaling serious network problems and its normal expected behavior (as may be evidenced from the performance history acquired for the parameter during the learning mode).

After the thresholds are computed, the Workstation  
20 loads them into the Monitor and instructs the Monitor to revert to normal monitoring using the new thresholds (step 314).

This procedure provides a mechanism enabling the network manager to adaptively reset thresholds in  
25 response to changing conditions on the network, shifting usage patterns and evolving network topology. As the network changes over time, the network manager merely invokes the adaptive threshold setting feature and updates the thresholds to reflect those changes.

### 30 The Diagnostic Analyzer Module:

The Management Workstation includes a diagnostic analyzer module which automatically detects and diagnoses the existence and cause of certain types of network problems. The functions of the diagnostic module may  
35 actually be distributed among the Workstation and the

- 66 -

Network Monitors which are active on the network. In principle, the diagnostic analyzer module includes the following elements for performing its fault detection and analysis functions.

5           The Management Workstation contains a reference model of a normally operating network. The reference model is generated by observing the performance of the network over an extended period of time and computing averages of the performance statistics that were observed  
10 during the observation period. The reference model provides a reference against which future network performance can be compared so as to diagnose and analyze potential problems. The Network Monitor (in particular, the STATS module) includes alarm thresholds on a selected  
15 set of the parameters which it monitors. Some of those thresholds are set on parameters which tend to be indicative of the onset or the presence of particular network problems.

          During monitoring, when a Monitor threshold is  
20 exceeded, thereby indicating a potential problem (e.g. in a TCP connection), the Network Monitor alerts the Workstation by sending an alarm. The Workstation notifies the user and presents the user with the option of either ignoring the alarm or invoking a diagnostic  
25 algorithm to analyze the problem. If the user invokes the diagnostic algorithm, the Workstation compares the current performance statistics to its reference model to analyze the problem and report its results. (Of course, this may also be handled automatically so as to not  
30 require user intervention.) The Workstation obtains the data on current performance of the network by retrieving the relevant performance statistics from all of the segment Network Monitors that may have information useful to diagnosing the problem.

- 67 -

The details of a specific example involving poor TCP connection performance will now be described. This example refers to a typical network on which the diagnostic analyzer resides, such as the network 5 illustrated in Fig. 25. It includes three segments labelled S1, S2, and S3, a router R1 connecting S1 to S2, a router R2 connecting S2 to S3, and at least two nodes, node A on S1 which communicates with node B on S3. On each segment there is also a Network Monitor 324 to 10 observe the performance of its segment in the manner described earlier. A Management Workstation 320 is also located on S1 and it includes a diagnostic analyzer module 322. For this example, the symptom of the network problem is degraded performance of a TCP connection 15 between Nodes A and B.

A TCP connection problem may manifest itself in a number of ways, including, for example, excessively high numbers for any of the following:

errors

20        packets with bad sequence numbers  
         packets retransmitted  
         bytes retransmitted  
         out of order packets  
         out of order bytes

25        packets after window closed  
         bytes after window closed  
         average and maximum round trip times

or by an unusually low value for the current window size. By setting the appropriate thresholds, the Monitor is 30 programmed to recognize any one or more of these symptoms. If any one of of the thresholds is exceeded, the Monitor sends an alarm to the Workstation. The Workstation is programmed to recognize the particular alarm as related to an event which can be further 35 analyzed by its diagnostic analyzer module 322. Thus,

- 68 -

the Workstation presents the user with the option of invoking its diagnostic capabilities (or automatically invokes the diagnostic capabilities).

In general terms, when the diagnostic analyzer is  
5 invoked, it looks at the performance data that the segment Monitors produce for the two nodes, for the dialogs between them and for the links that interconnect them and compares that data to the reference model for the network. If a significant divergence from the  
10 reference model is identified, the diagnostic analyzer informs the Workstation (and the user) about the nature of the divergence and the likely cause of the problem. In conducting the comparison to "normal" network performance, the network circuit involved in  
15 communications between nodes A and B is decomposed into its individual components and diagnostic analysis is performed on each link individually in the effort to isolate the problem further.

The overall structure of the diagnostic algorithm  
20 400 is shown in Fig. 26. When invoked for analyzing a possible TCP problem between nodes A and B, diagnostic analyzer 322 checks for a TCP problem at node A when it is acting as a source node (step 402). To perform this check, diagnostic algorithm 400 invokes a source node  
25 analyzer algorithm 450 shown in Fig. 27. If a problem is identified, the Workstation reports that there is a high probability that node A is causing a TCP problem when operating as a source node and it reports the results of the investigation performed by algorithm 450 (step 404).

30 If node A does not appear to be experiencing a TCP problem when acting as a source node, diagnostic analyzer 322 checks for evidence of a TCP problem at node B when it is acting as a sink node (step 406). To perform this check, diagnostic algorithm 400 invokes a sink node  
35 analyzer algorithm 470 shown in Fig. 28. If a problem is



- 69 -

identified, the Workstation reports that there is a high probability that node B is causing a TCP problem when operating as a sink node and it reports the results of the investigation performed by algorithm 470 (step 408).

5           Note that source and sink nodes are concepts which apply to those dialogs for which a direction of the communication can be defined. For example, the source node may be the one which initiated the dialog for the purpose of sending data to the other node, i.e., the sink  
10 node.

          If node B does not appear to be experiencing a TCP problem when acting as a sink node, diagnostic analyzer 322 checks for evidence of a TCP problem on the link between Node A and Node B (step 410). To perform this  
15 check, diagnostic algorithm 400 invokes a link analysis algorithm 550 shown in Fig. 29. If a problem is identified, the Workstation reports that there is a high probability that a TCP problem exists on the link and it reports the results of the investigation performed by  
20 link analysis algorithm 550 (step 412).

          If the link does not appear to be experiencing a TCP problem, diagnostic analyzer 322 checks for evidence of a TCP problem at node B when it is acting as a source node (step 414). To perform this check, diagnostic  
25 algorithm 400 invokes the previously mentioned source algorithm 450 for Node B. If a problem is identified, the Workstation reports that there is a medium probability that node B is causing a TCP problem when operating as a source node and it reports the results of  
30 the investigation performed by algorithm 450 (step 416).

          If node B does not appear to be experiencing a TCP problem when acting as a source node, diagnostic analyzer 322 checks for a TCP problem at node A when it is acting as a sink node (step 418). To perform this check,  
35 diagnostic algorithm 400 invokes sink node analyzer

- 70 -

algorithm 470 for Node A. If a problem is identified, the Network Monitor reports that there is a medium probability that node A is causing a TCP problem when operating as a sink node and it reports the results of  
5 the investigation performed by algorithm 470 (step 420).

Finally, if node A does not appear to be experiencing a TCP problem when acting as a sink node, diagnostic analyzer 322 reports that it was not able to isolate the cause of a TCP problem (step 422).

10 The algorithms which are called from within the above-described diagnostic algorithm will now be described. Referring to Fig. 27, source node analyzer algorithm 450 checks whether a particular node is causing a TCP problem when operating as a source node. The  
15 strategy is as follows. To determine whether a TCP problem exists at this node which is the source node for the TCP connection, look at other connections for which this node is a source. If other TCP connections are okay, then there is probably not a problem with this  
20 node. This is an easy check with a high probability of being correct. If no other good connections exist, then look at the lower layers for possible reasons. Start at DLL and work up as problems at lower layers are more fundamental, i.e., they cause problems at higher layers  
25 whereas the reverse is not true.

In accordance with this approach, algorithm 450 first determines whether the node is acting as a source node in any other TCP connection and, if so, whether the other connection is okay (step 452). If the node is  
30 performing satisfactorily as a source node in another TCP connection, algorithm 450 reports that there is no problem at the source node and returns to diagnostic algorithm 400 (step 454). If algorithm 450 cannot identify any other TCP connections involving this node  
35 that are okay, it moves up through the protocol stack

- 71 -

checking each level for a problem. In this case, it then checks for DLL problems at the node when it is acting as a source node by calling an DLL problem checking routine 510 (see Fig. 30) (step 456). If a DLL problem is found, 5 that fact is reported (step 458). If no DLL problems are found, algorithm 450 checks for an IP problem at the node when it is acting as a source by calling an IP problem checking routine 490 (see Fig. 31) (step 460). If an IP problem is found, that fact is reported (step 462). If 10 no IP problems are found, algorithm 450 checks whether any other TCP connection in which the node participates as a source is not okay (step 464). If another TCP connection involving the node exists and it is not okay, algorithm 450 reports a TCP problem at the node (step 15 466). If no other TCP connections where the node is acting as a source node can be found, algorithm 450 exits.

Referring to Fig. 28, sink node analyzer algorithm 470 checks whether a particular node is causing a TCP 20 problem when operating as a sink node. It first determines whether the node is acting as a sink node in any other TCP connection and, if so, whether the other connection is okay (step 472). If the node is performing satisfactorily as a sink node in another TCP connection, 25 algorithm 470 reports that there is no problem at the source node and returns to diagnostic algorithm 400 (step 474). If algorithm 470 cannot identify any other TCP connections involving this node that are okay, it then checks for DLL problems at the node when it is acting as 30 a sink node by calling DLL problem checking routine 510 (step 476). If a DLL problem is found, that fact is reported (step 478). If no DLL problems are found, algorithm 470 checks for an IP problem at the node when it is acting as a sink by calling IP problem checking 35 routine 490 (step 480). If an IP problem is found, that

- 72 -

fact is reported (step 482). If no IP problems are found, algorithm 470 checks whether any other TCP connection in which the node participates as a sink is not okay (step 484). If another TCP connection involving  
5 the node as a sink exists and it is not okay, algorithm 470 reports a TCP problem at the node (step 486). If no other TCP connections where the node is acting as a sink node can be found, algorithm 470 exits.

Referring to Fig. 31, IP problem checking routine  
10 490 checks for IP problems at a node. It does this by comparing the IP performance statistics for the node to the reference model (steps 492 and 494). If it detects any significant deviations from the reference model, it reports that there is an IP problem at the node (step  
15 496). If no significant deviations are noted, it reports that there is no IP problem at the node (step 498).

As revealed by examining Fig. 30, DLL problem checking routine 510 operates in a similar manner to IP problem checking routine 490, with the exception that it  
20 examines a different set of parameters (i.e., DLL parameters) for significant deviations.

Referring the Fig. 29, link analysis logic 550 first determines whether any other TCP connection for the link is operating properly (step 552). If a properly  
25 operating TCP connection exists on the link, indicating that there is no link problem, link analysis logic 550 reports that the link is okay (step 554). If a properly operating TCP connection cannot be found, the link is decomposed into its constituent components and an IP link  
30 component problem checking routine 570 (see Fig. 32) is invoked for each of the link components (step 556). IP link component problem routine 570 evaluates the link component by checking the IP layer statistics for the relevant link component.

- 73 -

The decomposition of the link into its components arranges them in order of their distance from the source node and the analysis of the components proceeds in that order. Thus, for example, the link components which make up the link between nodes A and B include in order: segment S1, router R1, segment S2, router R2, and segment S3. The IP data for these various components are analyzed in the following order:

IP data for segment S1  
 10 IP data for address R1  
 IP data for source node to R1  
 IP data for S1 to S2  
 IP data for S2  
 IP data for address R2  
 15 IP data for S3  
 IP data for S2 to S3  
 IP data for S1 to S3

As shown in Fig. 32, IP link component problem checking routine 570 compares IP statistics for the link component to the reference model (step 572) to determine whether network performance deviates significantly from that specified by the model (step 574). If significant deviations are detected, routine 570 reports that there is an IP problem at the link component (step 576).  
 25 Otherwise, it reports that it found no IP problem (step 578).

Referring back to Fig. 29, after completing the IP problem analysis for all of the link components, logic 550 then invokes a DLL link component problem checking routine 580 (see Fig. 33) for each link component to check its DLL statistics (step 558).  
 30

DLL link problem routine 580 is similar to IP link problem routine 570. As shown in Fig. 33, DLL link problem checking routine 580 compares DLL statistics for the link to the reference model (step 582) to determine  
 35

- 74 -

whether network performance at the DLL deviates significantly from that specified by the model (step 584). If significant deviations are detected, routine 580 reports that there is a DLL problem at the link component (step 586). Otherwise, it reports that no DLL problems were found (step 588).

Referring back to Fig. 29, after completing the DLL problem analysis for all of the link components, logic 550 checks whether there is any other TCP on the link (step 560). If another TCP exists on the link (which implies that the other TCP is also not operating properly), logic 550 reports that there is a TCP problem on the link (step 562). Otherwise, logic 550 reports that there was not enough information from the existing packet traffic to determine whether there was a link problem (step 564)

If the analysis of the link components does not isolate the source of the problem and if there were components for which sufficient information was not available (due possibly to lack of traffic over through that component), the user may send test messages to those components to generate the information needed to evaluate its performance.

The reference model against which comparisons are made to detect and isolate malfunctions may be generated by examining the behavior of the network over an extended period of operation or over multiple periods of operation. During those periods of operation, average values and maximum excursions (or standard deviations) for observed statistics are computed. These values provide an initial estimate of a model of a properly functioning system. As more experience with the network is obtained and as more historical data on the various statistics is accumulated the thresholds for detecting actual malfunctions or imminent malfunctions and the

- 75 -

reference model can be revised to reflect the new experience.

What constitutes a significant deviation from the reference model depends upon the particular parameter  
5 involved. Some parameters will not deviate from the expected norm and thus any deviation would be considered to be significant, for example, consider ICMP messages of type "destination unreachable," IP errors, TCP errors. Other parameters will normally vary within a wide range  
10 of acceptable values, and only if they move outside of that range should the deviation be considered significant. The acceptable ranges of variation can be determined by watching network performance over a sustained period of operation.

15 The parameters which tend to provide useful information for identifying and isolating problems at the node level for the different protocols and layers include the following.

TCP

20 error rate  
header byte rate  
packets retransmitted  
bytes retransmitted  
packets after window closed  
25 bytes after window closed

UDP

error rate  
header byte rate

IP

30 error rate  
header byte rate  
fragmentation rate  
all ICMP messages of type destination

- 76 -

unreachable, parameter problem,  
redirection

DLL

error rate

5 runts

For diagnosing network segment problems, the above-identified parameters are also useful with the addition of the alignment rate and the collision rate at the DLL. All or some subset of these parameters may be included  
10 among the set of parameters which are examined during the diagnostic procedure to detect and isolate network problems.

The above-described technique can be applied to a wide range of problems on the network, including among  
15 others, the following:

TCP Connection fails to establish  
UDP Connection performs poorly  
UDP not working at all  
IP poor performance/high error rate  
20 IP not working at all  
DLL poor performance/high error rate  
DLL not working at all

For each of these problems, the diagnostic approach would be similar to that described above, using, of course,  
25 different parameters to identify the potential problem and isolate its cause.

The Event Timing Module

Referring again to Fig. 5, the RTP is programmed to detect the occurrence of certain transactions for  
30 which timing information is desired. The transactions typically occur within a dialog at a particular layer of the protocol stack and they involve a first event (i.e., an initiating event) and a subsequent partner event or response. The events are protocol messages that arrive



- 77 -

at the Network Monitor, are parsed by the RTP and then passed to Event Timing Module (ETM) for processing. A transaction of interest might be, for example, a read of a file on a server. In that case, the initiating event  
5 is the read request and the partner event is the read response. The time of interest is the time required to receive a response to the read request (i.e., the transaction time). The transaction time provides a useful measure of network performance and if measured at  
10 various times throughout the day under different load conditions gives a measure of how different loads affect network response times. The layer of the communication protocol at which the relevant dialog takes place will of course depend upon the nature of the event.

15 In general, when the RTP detects an event, it transfers control to the ETM which records an arrival time for the event. If the event is an initiating event, the ETM stores the arrival time in an event timing database 300 (see Fig. 34) for future use. If the event  
20 is a partner event, the ETM computes a difference between that arrival time and an earlier stored time for the initiating event to determine the complete transaction time.

Event timing database 300 is an array of records  
25 302. Each record 302 includes a dialog field 304 for identifying the dialog over which the transactions of interest are occurring and it includes an entry type field 306 for identifying the event type of interest. Each record 302 also includes a start time field 308 for  
30 storing the arrival time of the initiating event and an average delay time field 310 for storing the computed average delay for the transactions. A more detailed description of the operation of the ETM follows.

Referring to Fig. 35, when the RTP detects the  
35 arrival of a packet of the type for which timing

- 78 -

information is being kept, it passes control to the ETM along with relevant information from the packet, such as the dialog identifier and the event type (step 320). The ETM then determines whether it is to keep timing information for that particular event by checking the event timing database (step 322). Since each event type can have multiple occurrences (i.e., there can be multiple dialogs at a given layer), the dialog identifier is used to distinguish between events of the same type for different dialogs and to identify those for which information has been requested. All of the dialog/events of interest are identified in the event timing database. If the current dialog and event appear in the event timing database, indicating that the event should be timed, the ETM determines whether the event is a starting event or an ending event so that it may be processed properly (step 324). For certain events, the absence of a start time in the entry field of the appropriate record 302 in event timing database 300 is one indicator that the event represents a start time; otherwise, it is an end time event. For other events, the ETM determines if the start time is to be set by the event type as specified in the packet being parsed. For example, if the event is a file read a start time is stored. If the event is the read completion it represents an end time. In general, each protocol event will have its own intrinsic meaning for how to determine start and end times.

Note that the arrival time is only an estimate of the actual arrival time due to possible queuing and other processing delays. Nevertheless, the delays are generally so small in comparison to the transaction times being measured that they are of little consequence.

In step 324, if the event represents a start time, the ETM gets the current time from the kernal and stores

- 79 -

it in start time field 308 of the appropriate record in event timing database 300 (step 326). If the event represents an end time event, the ETM obtains the current time from the kernel and computes a difference between  
5 that time and the corresponding start time found in event timing database 300 (step 328). This represents the total time for the transaction of interest. It is combined with the stored average transaction time to compute a new running average transaction time for that  
10 event (step 330).

Any one of many different methods can be used to compute the running average transaction time. For example, the following formula can be used:

$$\text{New Avg.} = [(5 * \text{Stored Avg.}) + \text{Transaction}$$
  
15  $\text{Time}] / 6.$

After six transactions have been timed, the computed new average becomes a running average for the transaction times. The ETM stores this computed average in the appropriate record of event timing database 300,  
20 replacing the previous average transaction time stored in that record, and it clears start time entry field 308 for that record in preparation for timing the next transaction.

After processing the event in steps 322, 326, and  
25 330, the ETM checks the age of all of the start time entries in the event timing database 300 to determine if any of them are too "old" (step 332). If the difference between the current time and any of the start times exceeds a preselected threshold, indicating that a  
30 partner event has not occurred within a reasonable period of time, the ETM deletes the old start time entry for that dialog/event (step 334). This insures that a missed packet for a partner event does not result in an erroneously large transaction time which throws off the  
35 running average for that event.

- 80 -

If the average transaction time increases beyond a preselected threshold set for timing events, an alarm is sent to the Workstation.

Two examples will now be described to illustrate the operation of the ETM for specific event types. In the first example, Node A of Fig. 25 is communicating with Node B using the NFS protocol. Node A is the client while Node B is the server. The Network Monitor resides on the same segment as node A, but this is not a requirement. When Node A issues a read request to Node B, the Network Monitor sees the request and the RTP within the Network Monitor transfers control to the ETM. Since it is a read, the ETM stores a start time in the Event Timing Database. Thus, the start time is the time at which the read was initiated.

After some delay, caused by the transmission delays of getting the read message to node B, node B performs the read and sends a response back to node A. After some further transmission delays in returning the read response, the Network Monitor receives the second packet for the event. At the time, the ETM recognizes that the event is an end time event and updates the average transaction time entry in the appropriate record with a new computed running average. The ETM then compares the average transaction time with the threshold for this event and if it has been exceeded, issues an alarm to the Workstation.

In the second example, node A is communicating with Node B using the Telnet protocol. Telnet is a virtual terminal protocol. The events of interest take place long after the initial connection has been established. Node A is typing at a standard ASCII (VT100 class) terminal which is logically (through the network) connected to Node B. Node B has an application which is receiving the characters being typed on Node A and, at

- 81 -

appropriate times, indicated by the logic of the applications, sends characters back to the terminal located on Node A. Thus, every time node A sends characters to B, the Network Monitor sees the  
5 transmission.

In this case, there are several transaction times which could provide useful network performance information. They include, for example, the amount of time it takes to echo characters typed at the keyboard  
10 through the network and back to the display screen, the delay between typing an end of line command and seeing the completion of the application event come back or the network delays incurred in sending a packet and receiving acknowledgment for when it was received.

15 In this example, the particular time being measured is the time it takes for the network to send a packet and receive an acknowledgement that the packet has arrived. Since Telnet runs on top of TCP, which in turn runs on top of IP, the Network Monitor monitors the TCP  
20 acknowledge end-to-end time delays.

Note that this is a design choice of the implementation and that all events visible to the Network Monitor by virtue of the fact that information is in the packet could be measured.

25 When Node A transmits a data packet to Node B, the Network Monitor receives the packet. The RTP recognizes the packet as being part of a timed transaction and passes control to the ETM. The ETM recognizes it as a start time event, stores the start time in the event  
30 timing database and returns control to the RTP after checking for aging.

When Node B receives the data packet from Node A, it sends back an acknowledgment packet. When the Network Monitor sees that packet, it delivers the event to the  
35 ETM, which recognizes it as an end time event. The ETM

- 82 -

calculates the delay time for the complete transaction and uses that to update the average transaction time. The ETM then compares the new average transaction time with the threshold for this event. If it has been  
5 exceeded, the ETM issues an alarm to the Workstation.

Note that this example is measuring something very different than the previous example. The first example measures the time it takes to traverse the network, perform an action and return that result to the  
10 requesting node. It measures performance as seen by the user and it includes delay times from the network as well as delay times from the File Server.

The second example is measuring network delays without looking at the service delays. That is, the ETM  
15 is measuring the amount of time it takes to send a packet to a node and receive the acknowledgement of the receipt of the message. In this example, the ETM is measuring transmissions delays as well as processing delays associated with network traffic, but not anything having  
20 to do with non-network processing.

As can be seen from the above examples, the ETM can measure a broad range of events. Each of these events can be measured passively and without the cooperation of the nodes that are actually participating  
25 in the transmission.

#### The Address Tracker Module (ATM)

Address tracker module (ATM) 43, one of the software modules in the Network Monitor (see Fig. 5), operates on networks on which the node addresses for  
30 particular node to node connections are assigned dynamically. An Appletalk® Network, developed by Apple Computer Company, is an example of a network which uses dynamic node addressing. In such networks, the dynamic change in the address of a particular service causes  
35 difficulty troubleshooting the network because the

- 83 -

network manager may not know where the various nodes are and what they are called. In addition, foreign network addresses (e.g., the IP addresses used by that node for communication over an IP network to which it is  
5 connected) can not be relied upon to point to a particular node. ATM 43 solves this problem by passively monitoring the network traffic and collecting a table showing the node address to node name mappings.

In the following description, the network on which  
10 the Monitor is located is assumed to be an Appletalk® Network. Thus, as background for the following discussion, the manner in which the dynamic node addressing mechanism operates on that network will first be described.

15 When a node is activated on the Appletalk® Network, it establishes its own node address in accordance with protocol referred to as the Local Link Access Protocol (LLAP). That is, the node guesses its own node address and then verifies that no other node on  
20 the network is using that address. The node verifies the uniqueness of its guess by sending an LLAP Enquiry control packet informing all other nodes on the network that it is going to assign itself a particular address unless another node responds that the address has already  
25 been assigned. If no other node claims that address as its own by sending an LLAP acknowledgment control packet, the first node uses the address which it has selected. If another node claims the address as its own, the first node tries another address. This continues until, the  
30 node finds an unused address.

When the first node wants to communicate with a second node, it must determine the dynamically assigned node address of the second node. It does this in accordance with another protocol referred to as the Name  
35 Binding Protocol (NBP). The Name Binding Protocol is

- 84 -

used to map or bind human understandable node names with machine understandable node addresses. The NBP allows nodes to dynamically translate a string of characters (i.e., a node name) into a node address. The node  
5 needing to communicate with another node broadcasts an NBP Lookup packet containing the name for which a node address is being requested. The node having the name being requested responds with its address and returns a  
10 original requesting node. The first node then uses that address its current communications with the second node.

Referring to Fig. 36, the network includes an Appletalk® Network segment 702 and a TCP/IP segment 704, each of which are connected to a larger network 706  
15 through their respective gateways 708. A Monitor 710, including a Real Time Parser (RTP) 712 and an Address Tracking Module (ATM) 714, is located on Appletalk network segment 702 along with other nodes 711. A  
20 Management Workstation 716 is located on segment 704. It is assumed that Monitor 710 has the features and capabilities previously described; therefore, those features not specifically related to the dynamic node  
25 Suffice it to say that Monitor 710 is, of course, adapted to operate on Appletalk Network segment 702, to parse and analyze the packets which are transmitted over that segment according to the Appletalk® family of protocols  
30 the network to Management Workstation 716 located on segment 704.

Within Monitor 710, ATM 714 maintains a name table data structure 730 such as is shown in Fig. 37. Name  
35 Table 720 includes records 722, each of which has a node name field 724, a node address field 726, an IP address



- 85 -

field 728, and a time field 729. ATM 714 uses Name Table 720 to keep track of the mappings of node names to node address and to IP address. The relevance of each of the fields of records 722 in Name Table 720 are explained in 5 the following description of how ATM 714 operates.

In general, Monitor 710 operates as previously described. That is, it passively monitors all packet traffic over segment 702 and sends all packets to RTP 712 for parsing. When RTP 712 recognizes an Appletalk 10 packet, it transfers control to ATM 714 which analyzes the packet for the presence of address mapping information.

The operation of ATM 714 is shown in greater detail in the flow diagram of Fig. 38. When ATM 714 15 receives control from RTP 712, it takes the packet (step 730 and strips off the lower layers of the protocol until it determines whether there is a Name Binding Protocol message inside the packet (step 732). If it is a NBP message, ATM 714 then determines whether it is new name 20 Lookup message (step 734). If it is a new name Lookup message, ATM 714 extracts the name from the message (i.e., the name for which a node address is being requested) and adds the name to the node name field 724 of a record 722 in Name Table 720 (step 736).

25 If the message is an NBP message but it is not a Lookup message, ATM 714 determines whether it is a Lookup Reply (step 738). If it is a Lookup Reply, signifying that it contains a node name/node address binding, ATM 714 extracts the name and the assigned node address from 30 the message and adds this information to Name Table 720. ATM 714 does this by searching the name fields of records 722 in Name Table 720 until it locates the name. Then, it updates the node address field of the identified record to contain the node address which was extracted 35 from the received NBP packet. ATM 714 also updates time

- 86 -

field 729 to record the time at which the message was processed.

After ATM 714 has updated the address field of the appropriate record, it determines whether any records 722  
5 in Name Table 720 should be aged out (step 742). ATM 714 compares the current time to the times recorded in the time fields. If the elapsed time is greater than a preselected time period (e.g. 48 hours), ATM 714 clears the record of all information (step 744). After that, it  
10 awaits the next packet from RTP 712.

As ATM 714 is processing each a packet and it determines either that it does not contain an NBP message (step 732) or it does not contain a Lookup Reply message (step 738), ATM 714 branches to step 742 to perform the  
15 age out check before going on to the next packet from RTP 712.

The Appletalk to IP gateways provide services that allow an Appletalk Node to dynamically connect to an IP address for communicating with IP nodes. This service  
20 extends the dynamic node address mechanism to the IP world for all Appletalk nodes. While the flexibility provided is helpful to the users, the network manager is faced with the problem of not knowing which Appletalk Nodes are currently using a particular IP address and  
25 thus, they can not easily track down problems created by the particular node.

ATM 714 can use passive monitoring of the IP address assignment mechanisms to provide the network manager a Name-to-IP address mapping.

30 If ATM 714 is also keeping IP address information, it implements the additional steps shown in Fig. 39 after completing the node name to node address mapping steps. ATM 714 again checks whether it is an NBP message (step 748). If it is an NBP message, ATM 714 checks whether it  
35 is a response to an IP address request (step 750). IP

- 87 -

address requests are typically implied by an NBP Lookup request for an IP gateway. The gateway responds by supplying the gateway address as well as an IP address that is assigned to the requesting node. If the NBP  
5 message is an IP address response, ATM 714 looks up the requesting node in Name Table 720 (step 752) and stores the IP address assignment in the IP address field of the appropriate record 722 (step 754).

After storing the IP address assignment  
10 information, ATM 714 locates all other records 722 in Name Table 720 which contain that IP address. Since the IP address has been assigned to a new node name, those old entries are no longer valid and must be eliminated. Therefore, ATM 714 purges the IP address fields of those  
15 records (step 756). After doing this cleanup step, ATM 714 returns control to RTP 712.

Other embodiments are within the following claims. For example, the Network Monitor can be adapted to identify node types by analyzing the type of packet  
20 traffic to or from the node. If the node being monitored is receiving mount requests, the Monitor would report that the node is behaving like node a file server. If the node is issuing routing requests, the Monitor would report that the node is behaving like a router. In  
25 either case, the network manager can check a table of what nodes are permitted to provide what functions to determine whether the node is authorized to function as either a file server or a router, and if not, can take appropriate action to correct the problem.

- 88 -

## APPENDIX I

## SNMP MIB Subset Supported

This is the subset of the standard MIB which can be obtained by monitoring.

Refer to RFC 1066 Management Information Base for an explanation on the items which follow.

System group:  
none

Interfaces group  
ifType  
ifPhysAddress  
ifOperStatus  
ifInOctets  
ifInUcastPkts  
ifInNUcastPkts  
ifOutOctets  
ifOutUcastPkts  
ifOutNUcastPkts

Address Translation group  
none

IP group  
ipForwarding  
ipDefaultTTL  
ipInReceives  
ipInHdrErrors  
ipInAddrErrors  
ipForwDatagrams  
ipReasmReqds  
ipFragCreates

IP Address Table  
ipAddress  
ipAdEntBcastAddr

IP Routing Table  
none

ICMP group  
icmpInMsgs  
icmplnErrors  
icmpInDestUnreachs  
icmpInTimeExcds  
icmpInParmProbs  
icmpInSrcQuenchs  
icmpInRedirects  
icmpInEchoes

App. I - 1

- 89 -

icmpInEchoReps  
icmpInTimestamps  
icmpInTimestampReps  
icmpInAddrMasks  
icmpInAddrmaskReps  
icmpOutMsgs  
icmpOutDestrUnreachs  
icmpOutTimeExcds  
icmpOutParmProbs  
icmpOutSrcQuenchs  
icmpOutRedirects  
icmpOutEchoes  
icmpOutEchoReps  
icmpOutTimestamps  
icmpOutTimestampReps  
icmpOutAddrMasks  
icmpOutAddrmaskReps

## TCP group

tcpActiveOpens  
tcpPassiveOpens  
tcpAttempFails  
tcpEstabResets  
tcpCurrEstab  
tcpInSegs  
tcpOutSegs  
tcpRetransSegs  
tcpConnTable

## UDP group

udpInDatagrams  
udpInErrors  
udpOutDatagrams  
udpOutErrors

## EGP group

egpInMsgs  
egpInErrors  
egpOutMsgs  
egpOutErrors

App. I - 2

- 90 -

## APPENDIX II

## MIB Definitions for Network Monitor

## 1. Common MIB Definitions

## Definitions

MIB_BUCKETS_PER_RATE	12
MIB_PROTOCOLS_PER_DIALOG	10
MibBucketsPerRate	12
MibProtocolsPerDialog	10
MIB_MAX_PROTOCOL	10
MIB_MAX_MOST_ACTIVE	5
MIB_MAX_DIALOG	3

## Structures Used

```

typedef struct {
    Byte    year
    Byte    month
    Byte    date
    Byte    day
    Byte    hour
    Byte    minute
    Byte    second
    Byte    unused
} MibTimeOfDay

typedef struct mib_count32_type {
    Uint32    accum        ( Long term accum. count)
    Uint32    current      ( Present running count)
    Uint32    highThld
} MibCount32

typedef struct mib_count64_type {
    Uint64    accum        ( Long term accum. count)
    Uint64    current      ( Present running count)
    Uint64    highThld
} MibCount64

typedef struct mib_meter_type {
    Uint32    current
    Uint32    high
    Uint32    low
    Uint32    highThld
} MibMeter
typedef struct mib_average_meter_type {
    Uint32    current

```

App. II - 1

- 91 -

```

    Uint32          high
    Uint32          low
    Uint32          highThld
} MibAverageMeter

```

```

typedef struct mib_percent_type {
    Uint32          current
    Uint32          high
    Uint32          low
    Uint32          highThld
} MibPercent

```

```

typedef struct mib_rolling_rate_type {
    Uint32          current
    Uint32          high
    Uint32          low
    Uint32          highThld
} MibRollingRate

```

```

typedef MibRollingRate MibRatePers
typedef MibRollingRate MibRatePerH

```

```

typedef Uint32 MibShortRatePersS
typedef Uint32 MibShortRatePerM

```

```

typedef struct mib_short_count32_type {
    Uint32          current      ( Present running count)
    Uint32          accum       ( Long term accum. count)
} MibShortCount32

```

```

typedef struct mib_bucket_rate_type {
    Uint32          current      ( Present rate)
    Uint32          rates[MIB_BUCKETS_PER_RATE]( 12 5 minute
count buckets )
    Uint32          maxRates[MIB_BUCKETS_PER_RATE]( 12 5-min.
max
rate buckets )
} MibBucketRate

```

#### Most Active Table Definitions

```

typedef struct mib_most_active_entry_type {
    MibAddress      address

```

App. II - 2

- 92 -

```

        MibCount32          packetCount
        MibRatePers        packetRate
    } MibMostActiveEntry

```

```

typedef struct mib_most_active_table_type {
    Uint32          numEntries
    Uint32          nextEntry
    MibMostActiveEntry mostActiveEntry[MIB_MAX_MOST_ACTIVE]
} MibMostActiveTable

```

**Protocol Table Definitions**

```

typedef struct mib_protocol_entry_type {
    Uint32          protocol
    MibCount32      packetCount
    MibRatePers     packetRate
} MibProtocolEntry

```

```

typedef struct mib_protocol_table_type {
    Uint32          numEntries
    Uint32          nextEntry
    MibProtocolEntry protocolEntry[MIB_MAX_PROTOCOL]
} MibProtocolTable

```

**Dialog Table Definitions**

```

typedef struct mib_transport_type {
    Uint32          transportProtocol
    Uint32          applicationProtocol
    Uint32          initiator
    Uint32          connectionRetries
    Uint32          addr1_window
    Uint32          addr2_window
    Uint32          state
    Uint32          closeReason
} MibTransportType

```

```

typedef struct mib_dialog_entry_type {
    MibAddress      addresses
    Uint32          protocolEntries
    Uint32          protocols[MIB_PROTOCOLS_PER_DIALOG]
    MibTimeOfDay    gmt
    Uint32          startTime
    Uint32          lastTime
    Uint32          alarmsSent
    MibCount32      packets
    MibRatePers     packetRate
}

```

App. II - 3



- 93 -

```

MibCount32          bytes
MibRatePerS        byteRate
MibCount32          errors
MibRatePerS        errorRate
MibCount32          fragments
MibRatePerS        fragmentRate
MibCount32          rexmits
MibRatePerS        rexmitRate
MibCount32          flowCtrls
MibRatePerS        flowCtrlRate
MibTransportType   transport
} MibDialogEntry

```

**Values for the initiator field**

```

ConnectionInitiatorUnknown  0
ConnectionInitiatorAddr1    1
ConnectionInitiatorAddr2    2

```

**Values for the connectionCloseReason field**

```

ConnectionCloseUnknown      0
ConnectionCloseFin          1
ConnectionCloseRst          2

```

**Values for the connectionState field**

```

ConnectionStateUnknown      0
ConnectionStateConnecting   1
ConnectionStateData         2
ConnectionStateClosing      3
ConnectionStateClosed       4

```

```

typedef struct mib_dialog_table_type {
    Uint32          numEntries
    Uint32          nextEntry
    MibDialogEntry dialogEntry[MIB_MAX_DIALOG]
} MibDialogTable

```

**2. Data link layer mib definitions for Network Monitor mib.****2.1 dll Segment -Summary Tool**

```

typedef struct {
    MibShortCount32  frames
    MibBucketRate   frameRate
}

```

App. II - 4

- 94 -

```

MibShortCount32      bytes
MibBucketRate        byteRate
MibShortCount32      errors
MibBucketRate        errorRate
Uint32                protocolCount
Uint32                mostActiveCount
Uint32                pairCount
MibShortCount32      rcvOffSegs
MibBucketRate        rcvOffSegRate
MibShortCount32      xmtOffSegs
MibBucketRate        xmtOffSegRate
MibShortCount32      transits
MibBucketRate        transitRate
MibShortCount32      bcasts
MibBucketRate        bcastRate
MibShortCount32      mcasts
MibBucketRate        mcastRate
MibShortCount32      collisions
MibShortRatePerS     collisionRate
MibShortCount32      alignmtErrors
MibShortRatePerS     alignmtErrorRate
} MibDllSegSumStats
    
```

**2.2 dll Segment -Values Tool**

```

typedef struct {
MibCount32      frames
MibRatePerS     frameRate
MibCount32      bytes
MibRatePerS     byteRate
MibCount32      errors
MibRatePerS     errorRate
MibCount32      rcvOffSegs
MibRatePerS     rcvOffSegRate
MibCount32      xmtOffSegs
MibRatePerS     xmtOffSegRate
MibCount32      transits
MibRatePerS     transitRate
MibCount32      bcasts
MibRatePerS     bcastRate
MibCount32      mcasts
MibRatePerS     mcastRate
MibCount32      collisions
MibRatePerS     collisionRate
MibCount32      alignmtErrors
MibRatePerS     alignmtErrorRate
MibCount32      enetFrames
MibRatePerS     enetFrameRate
MibCount32      llcFrames
MibRatePerS     llcFrameRate
MibCount32      runtFrames
MibRatePerS     runtFrameRate
    
```

App. II - 5

```
} MibDllSegValStats
```

**2.3 dll Address - Summary Tool**

```
typedef struct {
    MibShortCount32    frames
    MibBucketRate     frameRate
    MibShortCount32    bytes
    MibBucketRate     byteRate
    MibShortCount32    errors
    MibBucketRate     errorRate
    Uint32             protocolCount
    Uint32             mostActiveCount
    Uint32             pairCount
    MibShortCount32    rcvOffSegs
    MibBucketRate     rcvOffSegRate
    MibShortCount32    xmtOffSegs
    MibBucketRate     xmtOffSegRate
    MibShortCount32    xmtBcasts
    MibBucketRate     xmtBcastRate
    MibShortCount32    xmtMcasts
    MibBucketRate     xmtMcastRate
} MibDllAddrSumStats
```

**2.4 dll Address- Values Tool**

```
typedef struct {
    MibCount32         rcvFrames
    MibRatePers        rcvFrameRate
    MibCount32         rcvBytes
    MibRatePers        rcvByteRate
    MibCount32         rcvErrors
    MibRatePers        rcvErrorRate
    MibCount32         xmtFrames
    MibRatePers        xmtFrameRate
    MibCount32         xmtBytes
    MibRatePers        xmtByteRate
    MibCount32         xmtErrors
    MibRatePers        xmtErrorRate
    MibCount32         xmtBcasts
    MibRatePers        xmtBcastRate
    MibCount32         xmtMcasts
    MibRatePers        xmtMcastRate
    MibCount32         rcvOffSegs
    MibRatePers        rcvOffSegRate
    MibCount32         xmtOffSegs
    MibRatePers        xmtOffSegRate
    MibCount32         enetFrames
    MibRatePers        enetFrameRate
    MibCount32         llcFrames
    MibRatePers        llcFrameRate
}
```

- 96 -

```

        MibCount32          runtFrames
        MibRatePers        runtFrameRate
    } MibDllAddrValStats
    
```

**3. IP layer mib definitions for Network Monitor mib.**

**3.1 ip Segment - Summary Tool**

```

typedef struct {
    MibShortCount32          pkts
    MibBucketRate           pktRate
    MibShortCount32          bytes
    MibBucketRate           byteRate
    MibShortCount32          errors
    MibBucketRate           errorRate
    Uint32                   protocolCount
    Uint32                   mostActiveCount
    Uint32                   pairCount
    MibShortCount32          rcvOffSegs
    MibBucketRate           rcvOffSegRate
    MibShortCount32          xmtOffSegs
    MibBucketRate           xmtOffSegRate

    MibShortCount32          transits
    MibBucketRate           transitRate
    MibShortCount32          flowCtrls
    MibBucketRate           flowCtrlRate
    MibShortCount32          bcasts
    MibBucketRate           bcastRate
    MibShortCount32          mcasts
    MibBucketRate           mcastRate
    MibShortCount32          frgmts
    MibBucketRate           frgmtRate
} MibIpSegSumStats
    
```

**3.2 ip Segment - Values Tool**

```

typedef struct {
    MibCount32              pkts
    MibRatePers             pktRate
    MibCount32              bytes
    MibRatePers             byteRate
    MibCount32              errors
    MibRatePers             errorRate
    MibCount32              rcvOffSegs
    MibRatePers             rcvOffSegRate
    MibCount32              xmtOffSegs
    MibRatePers             xmtOffSegRate
    MibCount32              transits
    MibRatePers             transitRate
    
```

App. II - 7

- 97 -

```

MibCount32          bcasts
MibRatePerS         bcastRate
MibCount32          mcasts
MibRatePerS         mcastRate
MibCount32          hdrBytes
MibRatePerS         hdrByteRate
MibCount32          frgmts
MibRatePerS         frgmtRate
} MibIpSegValStats
    
```

**3.3 ip Address - Summary Tool**

```

typedef struct {
MibShortCount32     pkts
MibBucketRate       pktRate
MibShortCount32     bytes
MibBucketRate       byteRate
MibShortCount32     errors
MibBucketRate       errorRate
Uint32              protocolCount
Uint32              mostActiveCount
Uint32              pairCount
MibShortCount32     rcvOffSegs
MibBucketRate       rcvOffSegRate
MibShortCount32     xmtOffSegs
MibBucketRate       xmtOffSegRate
MibShortCount32     flowCtrls
MibBucketRate       flowCtrlRate
MibShortCount32     frgmts
MibBucketRate       frgmtRate
MibShortCount32     xmtBcasts
MibBucketRate       xmtBcastRate
MibShortCount32     xmtMcasts
MibBucketRate       xmtMcastRate
} MibIpAddrSumStats
    
```

**3.4 ip Address - Values Tool**

```

typedef struct {
MibCount32          rcvPkts
MibRatePerS         rcvPktRate
MibCount32          rcvBytes
MibRatePerS         rcvByteRate
MibCount32          rcvErrors
MibRatePerS         rcvErrorRate
MibCount32          xmtPkts
MibRatePerS         xmtPktRate
MibCount32          xmtBytes
MibRatePerS         xmtByteRate
MibCount32          xmtErrors
MibRatePerS         xmtErrorRate
MibCount32          rcvHdrBytes
MibRatePerS         rcvHdrByteRate
    
```

- 98 -

```

MibCount32          xmtHdrBytes
MibRatePerS         xmtHdrByteRate
MibCount32          rcvFrgmts
MibRatePerS         rcvFrgmtRate
MibCount32          xmtFrgmts
MibRatePerS         xmtFrgmtRate
MibCount32          xmtBcasts
MibRatePerS         xmtBcastRate
MibCount32          xmtMcasts
MibRatePerS         xmtMcastRate
MibCount32          rcvOffSegs
MibRatePerS         rcvOffSegRate
MibCount32          xmtOffSegs
MibRatePerS         xmtOffSegRate
} MibIpAddrValStats
    
```

**4. ICMP layer mib definitions for Network Monitor mib.**

**4.1 icmp Segment - Summary Tool**

```

typedef struct {
    MibShortCount32          pkts
    MibBucketRate           pktRate

    MibShortCount32          bytes
    MibBucketRate           byteRate

    MibShortCount32          errors
    MibBucketRate           errorRate

    Uint32                   mostActiveCount
    Uint32                   pairCount

    MibShortCount32          rcvOffSegs
    MibBucketRate           rcvOffSegRate
    MibShortCount32          xmtOffSegs
    MibBucketRate           xmtOffSegRate
    MibShortCount32          transits
    MibBucketRate           transitRate

    MibShortCount32          echoReq
    MibShortCount32          echoReply
    MibShortCount32          destUnr
    MibShortCount32          srcQuench
    MibShortCount32          redir
    MibShortCount32          timeExceeded
    MibShortCount32          paramProblem
    MibShortCount32          timestampReq
    MibShortCount32          timestampReply
    MibShortCount32          addrMaskReq
    MibShortCount32          addrMaskReply
} MibIcmpSegSumStats
    
```

App. II - 9

4.2 icmp Segment - Values Tool

```

typedef struct {
    MibCount32          pkts
    MibRatePers        pktRate

    MibCount32          bytes
    MibRatePers        byteRate

    MibCount32          errors
    MibRatePers        errorRate

    MibCount32          rcvOffSegs
    MibRatePers        rcvOffSegRate
    MibCount32          xmtOffSegs
    MibRatePers        xmtOffSegRate
    MibCount32          transits
    MibRatePers        transitRate

    MibCount32          echoReq
    MibRatePers        echoReqRate
    MibCount32          echoReply
    MibRatePers        echoReplyRate

    MibCount32          destUnrNet
    MibRatePers        destUnrNetRate
    MibCount32          destUnrHost
    MibRatePers        destUnrHostRate
    MibCount32          destUnrProtocol
    MibRatePers        destUnrProtocolRate
    MibCount32          destUnrPort
    MibRatePers        destUnrPortRate
    MibCount32          destUnrFrgmt
    MibRatePers        destUnrFrgmtRate
    MibCount32          destUnrSrcRoute
    MibRatePers        destUnrSrcRouteRate
    MibCount32          destUnrNetUnknown
    MibRatePers        destUnrNetUnknownRate
    MibCount32          destUnrHostUnknown
    MibRatePers        destUnrHostUnknownRate
    MibCount32          destUnrSrcHostIsolated
    MibRatePers        destUnrSrcHostIsolatedRate
    MibCount32          destUnrNetProhibited
    MibRatePers        destUnrNetProhibitedRate
    MibCount32          destUnrHostProhibited
    MibRatePers        destUnrHostProhibitedRate
    MibCount32          destUnrNetTos
    MibRatePers        destUnrNetTosRate
    MibCount32          destUnrHostTos

```

- 100 -

```

MibRatePers          destUnrHostTosRate

MibCount32           srcQuench
MibRatePers          srcQuenchRate

MibCount32           redirNet
MibRatePers          redirNetRate
MibCount32           redirHost
MibRatePers          redirHostRate
MibCount32           redirNetTos
MibRatePers          redirNetTosRate
MibCount32           redirHostTos
MibRatePers          redirHostTosRate

MibCount32           timeExceededInTransit
MibRatePers          timeExceededInTransitRate
MibCount32           timeExceededInReass
MibRatePers          timeExceededInReassRate

MibCount32           paramProblem
MibRatePers          paramProblemRate
MibCount32           paramProblemOption
MibRatePers          paramProblemOptionRate

MibCount32           timestampReq
MibRatePers          timestampReqRate
MibCount32           timestampReply
MibRatePers          timestampReplyRate

MibCount32           addrMaskReq
MibRatePers          addrMaskReqRate
MibCount32           addrMaskReply
MibRatePers          addrMaskReplyRate

} MibIcmpSegValStats
    
```

**4.3 icmp Address - Summary Tool**

```

typedef struct {
    MibShortCount32      pkts
    MibBucketRate        pktRate

    MibShortCount32      bytes
    MibBucketRate        byteRate

    MibShortCount32      errors
    MibBucketRate        errorRate
    Uint32               mostActiveCount
    Uint32               pairCount

    MibShortCount32      rcvOffSegs
    MibBucketRate        rcvOffSegRate
}
    
```

App. II - 11



- 101 -

```

MibShortCount32      xmtOffSegs
MibBucketRate        xmtOffSegRate

MibShortCount32      echoReq
MibShortCount32      echoReply
MibShortCount32      destUnr
MibShortCount32      srcQuench
MibShortCount32      redir
MibShortCount32      paramProblem
MibShortCount32      timeExceeded
MibShortCount32      timestampReq
MibShortCount32      timestampReply
MibShortCount32      addrMaskReq
MibShortCount32      addrMaskReply
} MibIcmpAddrSumStats
    
```

4.4 icmp Address- Values Tool

typedef struct {

```

MibCount32           rcvPkts
MibRatePerS          rcvPktRate
MibCount32           rcvBytes
MibRatePerS          rcvByteRate
MibCount32           rcvErrors
MibRatePerS          rcvErrorRate

MibCount32           xmtPkts
MibRatePerS          xmtPktRate
MibCount32           xmtBytes
MibRatePerS          xmtByteRate
MibCount32           xmtErrors
MibRatePerS          xmtErrorRate

MibCount32           rcvOffSegs
MibRatePerS          rcvOffSegRate
MibCount32           xmtOffSegs
MibRatePerS          xmtOffSegRate

MibCount32           rcvDestUnrNet
MibRatePerS          rcvDestUnrNetRate
MibCount32           rcvDestUnrHost
MibRatePerS          rcvDestUnrHostRate
MibCount32           rcvDestUnrProtocol
MibRatePerS          rcvDestUnrProtocolRate
MibCount32           rcvDestUnrPort
MibRatePerS          rcvDestUnrPortRate
MibCount32           rcvDestUnrFrgmt
MibRatePerS          rcvDestUnrFrgmtRate
MibCount32           rcvDestUnrSrcRoute
MibRatePerS          rcvDestUnrSrcRouteRate
MibCount32           rcvDestUnrNetUnknown
    
```

- 102 -

MibRatePers	rcvDestUnrNetUnknownRate
MibCount32	rcvDestUnrHostUnknown
MibRatePers	rcvDestUnrHostUnknownRate
MibCount32	rcvDestUnrSrcHostIsolated
MibRatePers	rcvDestUnrSrcHostIsolatedRate
MibCount32	rcvDestUnrNetProhibited
MibRatePers	rcvDestUnrNetProhibitedRate
MibCount32	rcvDestUnrHostProhibited
MibRatePers	rcvDestUnrHostProhibitedRate
MibCount32	rcvDestUnrNetTos
MibRatePers	rcvDestUnrNetTosRate
MibCount32	rcvDestUnrHostTos
MibRatePers	rcvDestUnrHostTosRate
MibCount32	rcvTimeExceededInTransit
MibRatePers	rcvTimeExceededInTransitRate
MibCount32	rcvTimeExceededInReass
MibRatePers	rcvTimeExceededInReassRate
MibCount32	rcvParamProblem
MibRatePers	rcvParamProblemRate
MibCount32	rcvParamProblemOption
MibRatePers	rcvParamProblemOptionRate
MibCount32	rcvSrcQuench
MibRatePers	rcvSrcQuenchRate
MibCount32	rcvRedirNet
MibRatePers	rcvRedirNetRate
MibCount32	rcvRedirHost
MibRatePers	rcvRedirHostRate
MibCount32	rcvRedirNetTos
MibRatePers	rcvRedirNetTosRate
MibCount32	rcvRedirHostTos
MibRatePers	rcvRedirHostTosRate
MibCount32	rcvEchoReq
MibRatePers	rcvEchoReqRate
MibCount32	rcvEchoReply
MibRatePers	rcvEchoReplyRate
MibCount32	rcvTimestampReq
MibRatePers	rcvTimestampReqRate
MibCount32	rcvTimestampReply
MibRatePers	rcvTimestampReplyRate
MibCount32	rcvAddrMaskReq
MibRatePers	rcvAddrMaskReqRate
MibCount32	rcvAddrMaskReply
MibRatePers	rcvAddrMaskReplyRate

- 103 -

MibCount32	xmtDestUnrNet
MibRatePers	xmtDestUnrNetRate
MibCount32	xmtDestUnrHost
MibRatePers	xmtDestUnrHostRate
MibCount32	xmtDestUnrProtocol
MibRatePers	xmtDestUnrProtocolRate
MibCount32	xmtDestUnrPort
MibRatePers	xmtDestUnrPortRate
MibCount32	xmtDestUnrFrgmt
MibRatePers	xmtDestUnrFrgmtRate
MibCount32	xmtDestUnrSrcRoute
MibRatePers	xmtDestUnrSrcRouteRate
MibCount32	xmtDestUnrNetUnknown
MibRatePers	xmtDestUnrNetUnknownRate
MibCount32	xmtDestUnrHostUnknown
MibRatePers	xmtDestUnrHostUnknownRate
MibCount32	xmtDestUnrSrcHostIsolated
MibRatePers	xmtDestUnrSrcHostIsolatedRate
MibCount32	xmtDestUnrNetProhibited
MibRatePers	xmtDestUnrNetProhibitedRate
MibCount32	xmtDestUnrHostProhibited
MibRatePers	xmtDestUnrHostProhibitedRate
MibCount32	xmtDestUnrNetTos
MibRatePers	xmtDestUnrNetTosRate
MibCount32	xmtDestUnrHostTos
MibRatePers	xmtDestUnrHostTosRate
MibCount32	xmtTimeExceededInTransit
MibRatePers	xmtTimeExceededInTransitRate
MibCount32	xmtTimeExceededInReass
MibRatePers	xmtTimeExceededInReassRate
MibCount32	xmtParamProblem
MibRatePers	xmtParamProblemRate
MibCount32	xmtParamProblemOption
MibRatePers	xmtParamProblemOptionRate
MibCount32	xmtSrcQuench
MibRatePers	xmtSrcQuenchRate
MibCount32	xmtRedirNet
MibRatePers	xmtRedirNetRate
MibCount32	xmtRedirHost
MibRatePers	xmtRedirHostRate
MibCount32	xmtRedirNetTos
MibRatePers	xmtRedirNetTosRate
MibCount32	xmtRedirHostTos
MibRatePers	xmtRedirHostTosRate
MibCount32	xmtEchoReq
MibRatePers	xmtEchoReqRate
MibCount32	xmtEchoReply

App. II - 14

- 104 -

```

MibRatePerS          xmtEchoReplyRate

MibCount32           xmtTimestampReq
MibRatePerS          xmtTimestampReqRate
MibCount32           xmtTimestampReply
MibRatePerS          xmtTimestampReplyRate

MibCount32           xmtAddrMaskReq
MibRatePerS          xmtAddrMaskReqRate
MibCount32           xmtAddrMaskReply
MibRatePerS          xmtAddrMaskReplyRate
    }
    
```

**5. TCP layer mib definitions for Network Monitor mib.**

**5.1 tcp Segment - Summary Tool**

```

typedef struct {

    MibShortCount32    pkts
    MibBucketRate      pktRate
    MibShortCount32    bytes
    MibBucketRate      byteRate

    MibShortCount32    errors
    MibBucketRate      errorRate

    Uint32             protocolCount
    Uint32             mostActiveCount
    Uint32             pairCount

    MibShortCount32    rcvOffSegs
    MibBucketRate      rcvOffSegRate
    MibShortCount32    xmtOffSegs
    MibBucketRate      xmtOffSegRate
    MibShortCount32    transits
    MibBucketRate      transitRate

    MibShortCount32    flowCtrls
    MibBucketRate      flowCtrlRate

    MibShortCount32    frgmts
    MibBucketRate      frgmtRate

    MibShortCount32    rexmts
    MibBucketRate      rexmtRate

} MibTcpSegSumStats
    
```

**5.2 tcp Segment - Values Tool**

- 105 -

```

typedef struct {
    MibCount32      pkts
    MibRatePers     pktRate

    MibCount32      bytes
    MibRatePers     byteRate

    MibCount32      errors
    MibRatePers     errorRate

    MibCount32      rcvOffSegs
    MibRatePers     rcvOffSegRate
    MibCount32      xmtOffSegs
    MibRatePers     xmtOffSegRate
    MibCount32      transits
    MibRatePers     transitRate

    MibCount32      hdrBytes
    MibRatePers     hdrByteRate
    MibCount32      frgmts
    MibRatePers     frgmtRate

    MibCount32      flowCtrls
    MibRatePers     flowCtrlRate

    MibCount32      rexmts
    MibRatePers     rexmtRate

    MibCount32      rexmtBytes
    MibRatePers     rexmtByteRate

    MibCount32      keepAlives
    MibRatePers     keepAliveRate

    MibCount32      windowProbes
    MibRatePers     windowProbeRate

    MibCount32      outOfOrder
    MibRatePers     outOfOrderRate

    MibCount32      afterWindow
    MibRatePers     afterWindowRate

    MibCount32      afterClose
    MibRatePers     afterCloseRate

    MibCount32      urgs
    MibRatePers     urgRate

    MibCount32      rststs
    MibRatePers     rstRate

```

App. II - 16

- 106 -

```

MibCount32      successfulConnections
MibRatePerH     successfulConnectionRate
MibCount32      connectionRetries
MibRatePerH     connectionRetryRate
MibCount32      failedConnections
MibRatePerH     failedConnectionRate
MibCount32      activeConnections
} MibTcpSegValStats
    
```

**5.3 tcp Address - Summary Tool**

```

typedef struct {
    MibShortCount32      pkts
    MibBucketRate        pktRate

    MibShortCount32      bytes
    MibBucketRate        byteRate

    MibShortCount32      errors
    MibBucketRate        errorRate

    Uint32               protocolCount
    Uint32               mostActiveCount
    Uint32               pairCount

    MibShortCount32      rcvOffSegs
    MibBucketRate        rcvOffSegRate
    MibShortCount32      xmtOffSegs
    MibBucketRate        xmtOffSegRate

    MibShortCount32      flowCtrls
    MibBucketRate        flowCtrlRate

    MibShortCount32      frgmts
    MibBucketRate        frgmtRate

    MibShortCount32      rexmts
    MibBucketRate        rexmtRate
} MibTcpAddrSumStats
    
```

**5.4 tcp Address- Values Tool**

```

typedef struct {
    MibCount32           rcvPkts
    MibRatePerS          rcvPktRate
    MibCount32           xmtPkts
    MibRatePerS          xmtPktRate
    
```

- 107 -

MibCount32	rcvBytes
MibRatePers	rcvByteRate
MibCount32	xmtBytes
MibRatePers	xmtByteRate
MibCount32	rcvErrors
MibRatePers	rcvErrorRate
MibCount32	xmtErrors
MibRatePers	xmtErrorRate
MibCount32	rcvOffSegs
MibRatePers	rcvOffSegRate
MibCount32	xmtOffSegs
MibRatePers	xmtOffSegRate
MibCount32	rcvHdrBytes
MibRatePers	rcvHdrByteRate
MibCount32	xmtHdrBytes
MibRatePers	xmtHdrByteRate
MibCount32	rcvFrgmts
MibRatePers	rcvFrgmtRate
MibCount32	xmtFrgmts
MibRatePers	xmtFrgmtRate
MibCount32	rcvRexmts
MibRatePers	rcvRexmtRate
MibCount32	xmtRexmts
MibRatePers	xmtRexmtRate
MibCount32	rcvRexmtBytes
MibRatePers	rcvRexmtByteRate
MibCount32	xmtRexmtBytes
MibRatePers	xmtRexmtByteRate
MibCount32	rcvKeepAlives
MibRatePers	rcvKeepAliveRate
MibCount32	xmtKeepAlives
MibRatePers	xmtKeepAliveRate
MibCount32	rcvWindowProbes
MibRatePers	rcvWindowProbeRate
MibCount32	xmtWindowProbes
MibRatePers	xmtWindowProbeRate
MibCount32	rcvOutOfOrder
MibRatePers	rcvOutOfOrderRate
MibCount32	xmtOutOfOrder
MibRatePers	xmtOutOfOrderRate
MibCount32	rcvAfterWindow
MibRatePers	rcvAfterWindowRate

App. II - 18

- 108 -

MibCount32	xmtAfterWindow
MibRatePers	xmtAfterWindowRate
MibCount32	rcvAfterClose
MibRatePers	rcvAfterCloseRate
MibCount32	xmtAfterClose
MibRatePers	xmtAfterCloseRate
MibCount32	rcvUrqs
MibRatePers	rcvUrgRate
MibCount32	xmtUrqs
MibRatePers	xmtUrgRate
MibCount32	rcvRsts
MibRatePers	rcvRstRate
MibCount32	xmtRsts
MibRatePers	xmtRstRate
MibCount32	successfulConnections
MibRatePerH	successfulConnectionRate
MibCount32	connectionRetries
MibRatePerH	connectionRetryRate
MibCount32	failedConnections
MibRatePerH	failedConnectionRate
MibCount32	activeConnections

6. UDP layer mib definitions for Network Monitor mib.

6.1 udp Segment -Summary Tool

```
typedef struct {
    MibShortCount32      pkts
    MibBucketRate        pktRate
    MibShortCount32      bytes
    MibBucketRate        byteRate
    MibShortCount32      errors
    MibBucketRate        errorRate
    MibShortCount32      protocolCount
    MibShortCount32      mostActiveCount
    MibShortCount32      pairCount
    MibShortCount32      rcvOffSegs
    MibBucketRate        rcvOffSegRate
    MibShortCount32      xmtOffSegs
    MibBucketRate        xmtOffSegRate
    MibShortCount32      transits
    MibBucketRate        transitRate
    MibShortCount32      flowCtrls
    MibBucketRate        flowCtrlRate
} MibUdpSegSumStats
```



- 109 -

**6.2 udp Segment - Values Tool**

```

typedef struct {
    MibCount32          pkts
    MibRatePerS        pktRate
    MibCount32          bytes
    MibRatePerS        byteRate
    MibCount32          errors
    MibRatePerS        errorRate
    MibShortCount32    protocolCount
    MibShortCount32    mostActiveCount
    MibShortCount32    pairCount
    MibCount32          rcvOffSegs
    MibRatePerS        rcvOffSegRate
    MibCount32          xmtOffSegs
    MibRatePerS        xmtOffSegRate
    MibCount32          transits
    MibRatePerS        transitRate
    MibCount32          flowCtrls
    MibRatePerS        flowCtrlRate
    MibCount32          hdrBytes
    MibRatePerS        hdrByteRate
} MibUdpSegValStats

```

**6.3 udp Address - Summary Tool**

```

typedef struct {
    MibShortCount32    pkts
    MibBucketRate      pktRate
    MibShortCount32    bytes
    MibBucketRate      byteRate
    MibShortCount32    errors
    MibBucketRate      errorRate
    MibShortCount32    protocolCount
    MibShortCount32    mostActiveCount
    MibShortCount32    pairCount
    MibShortCount32    rcvOffSegs
    MibBucketRate      rcvOffSegRate
    MibShortCount32    xmtOffSegs
    MibBucketRate      xmtOffSegRate
    MibShortCount32    flowCtrls
    MibBucketRate      flowCtrlRate
} MibUdpAddrSumStats

```

**6.4 udp Address- Values Tool**

```

typedef struct {
    MibCount32          rcvPkts
    MibRatePerS        rcvPktRate
    MibCount32          rcvBytes

```

App. II - 20

- 110 -

MibRatePerS	rcvByteRate
MibCount32	rcvErrors
MibRatePerS	rcvErrorRate
MibCount32	xmtPkts
MibRatePerS	xmtPktRate
MibCount32	xmtBytes
MibRatePerS	xmtByteRate
MibCount32	xmtErrors
MibRatePerS	xmtErrorRate
MibCount32	rcvHdrBytes
MibRatePerS	rcvHdrByteRate
MibCount32	xmtHdrBytes

**7. Monitor mib definitions for Network Monitor mib.**

```

typedef struct {
    int                length
    char              no[80]
} MibPhoneNumber

typedef struct {
    MacAddress         lanMacAddr
    IpAddress          lanIpAddr
    Uint32             lanTftpTimeout
    Uint32             lanTftpRetryLimit
    Uint32             lanSnmpTimeout
    Uint32             lanSnmpRetryLimit
    MibPhoneNumber    serialPhoneNo
    IpAddress          serialIpAddr
    Uint32             serialTftpTimeout
    Uint32             serialTftpRetryLimit
    Uint32             serialSnmpTimeout
    Uint32             serialSnmpRetryLimit
} MibWsParameters

typedef struct {
    MibAddress         address
    Uint32             flags
    MibDeviceType     type
    Uint32             parseControl
} MibParseControl

typedef struct {
    Uint32             numEntries
    Uint32             nextEntry
    MibParseControl   mibParseControl[MIB_MAX_PCR]
} MibParseControlOpaque

typedef struct {
    MacAddress         macAddr
    Byte               data[256]
}
    
```

- 111 -

```

        Uint32          length
derived
} MibAutoTopology

```

### 7.1 Monitor Control Group

```

typedef struct {
    Uint32          monReset
    MibTimeOfDay   monTOD
    Uint32          trapPermit
    Uint32          dupAddrTrapPermit
    Uint32          newNodeTrapPermit
    Uint32          shakeTime
    Uint32          wsMonLink
    Uint32          minTrapInterval
    Uint32          runMonitor
    MibWsParameters primaryWsParams
    MibWsParameters secondaryWsParams
    Uint32          debugLevel
    Uint32          parseCtrl
    Uint32          monitorSegment
    MibAutoTopology autoTopology
} MibMonitorControl

```

### 7.2 Monitor Statistics Group

```

typedef struct {
    MibCount32      dllDropped
    MibRatePerS     dllDroppedRate
    MibCount32      ipDropped
    MibRatePerS     ipDroppedRate
    MibCount32      icmpDropped
    MibRatePerS     icmpDroppedRate
    MibCount32      tcpDropped
    MibRatePerS     tcpDroppedRate
    MibCount32      udpDropped
    MibRatePerS     udpDroppedRate
    MibCount32      arpDropped
    MibRatePerS     arpDroppedRate
    MibCount32      nfsDropped
    MibRatePerS     nfsDroppedRate
    MibCount32      dbProblem
    MibShortCount32 cpuUtilization
    MibShortCount32 memoryUtilization

```

### 8. Alarm Mib Definitions

App. II - 22

- 112 -

**8.1 Counter alarm structure**

```

typedef struct {
    Uint32          alarm_class
    MibTimeOfDay   gmtime
    Uint32          time_ticks
    MibAddress     mon_address
    MibAddress     address
    Uint32         type
    Uint32         number
    MibCount32     value
    Uint32         user_data_length

    OPTIONAL
    Byte           user_data[MAX_ALARM_DATA]

OPTIONAL
} MibAlarmCounter

```

**8.2 Rate alarm structure**

```

typedef struct {
    Uint32          alarm_class
    MibTimeOfDay   gmtime
    Uint32          time_ticks
    MibAddress     mon_address
    MibAddress     address
    Uint32         type
    Uint32         number
    MibRollingRate value
    Uint32         rate_type
    Uint32         user_data_length

    OPTIONAL
    Byte           user_data[MAX_ALARM_DATA]

OPTIONAL
} MibAlarmRate

```

**8.3 Power-up alarm structure**

```

typedef struct {
    Uint32          alarm_class
    MibTimeOfDay   gmtime
    Uint32          time_ticks
    MibAddress     mon_address
    Uint32         alarm_reason
    Uint32         load_type
    Uint32         cpu_hw_rev
    Uint32         mon_link_hw_rev

```

App. II - 23

- 113 -

```

        Uint32          mgmt_link_hw_rev
        MibPhoneNumber mon_phone_no
        Uint32          error_type
        Uint32          error_code
        Uint32          error_param_1
        Uint32          error_param_2
        Uint32          error_param_3
    } MibAlarmPowerUp

```

#### 8.4 Link-up alarm structure

```

typedef struct {
    Uint32          alarm_class
    MibTimeOfDay   gmt
    Uint32          time_ticks
    MibAddress      mon_address
    Uint32          alarm_reason
    Uint32          load_type
    Uint32          cpu_hw_rev
    Uint32          mon_link_hw_rev
    Uint32          mgmt_link_hw_rev
    MibPhoneNumber mon_phone_no
    Uint32          error_type
    Uint32          error_code
    Uint32          error_param_1
    Uint32          error_param_2
    Uint32          error_param_3
} MibAlarmLinkUp

```

#### 8.5 New node alarm structure

```

typedef struct {
    Uint32          alarm_class
    MibTimeOfDay   gmt
    Uint32          time_ticks
    MibAddress      mon_address
    MibAddress      node_address
} MibAlarmNewNode

```

- 114 -

## APPENDIX III

PROTOCOL VARIABLES

The following is a list of some of the network variables for which data is gathered by the Monitor and a brief explanation of the variable, where appropriate.

DLL VariablesFrames

A frame is a series of bytes with predefined bit sequences that mark the frame's beginning and ending points. A DLL (data link layer) entity sends a message by putting it in a frame and transmitting it on the physical network. It's called a frame because the beginning and ending bit sequences "frame" the message.

Enclosed within the frame are the messages built by higher level protocols, such as IP and UDP. For example, an IP datagram must be placed in a frame before it can be transmitted.

Ethernet frames range from 64 to 1518 bytes in length.

Bytes

Monitor maintains a count and rate for bytes transmitted and received by all monitored objects. For example, for any node, you can monitor the number of bytes in or out to measure the traffic load with respect to that node. For a segment, you can monitor the number of bytes in and out of all nodes on the segment.

Error Frames

- A DLL Error Frame is logged in the following cases:
- \* If the frame is Ethernet, none are logged.
  - \* If the frame is IEEE 802.3:
    - Value of length parameter in header less than 3.

Alignment Errors

The number of frames observed for the selected segment with alignment errors. An alignment error is a frame with a length that is not an exact multiple of 8 bits. The following variables are available only for segments.

App. III - 1

- 115 -

### Collisions

The number of collisions observed on the selected segment. A collision occurs when two stations attempt to transmit simultaneously. A certain number of collisions are normal. The following variables are available only for segments.

A higher than typical value can mean that the physical interface for a single station has malfunctioned and is not following the protocol.

### Broadcast frame

A broadcast frame is a special frame that is received by all stations on the network. Common uses for broadcast frames include ARP (Address Resolution Protocol) and network testing.

### Multicast Frame

A multicast frame is a special frame that is received by a predetermined set of stations. Multicasting is used to send a message to a set of stations using a single frame, thus reducing network loading.

### Off-segment

Off-segment frames are frames that the Monitor observes on the local segment, but are destined for or originated by nodes not on the local segment. All off-segment frames then are either routed to, from, or across the local segment.

### Off-segment variables

Off-segment variables are a measure of the amount of routing or bridging that is occurring. Excessive off-segment traffic may mean that certain nodes on one segment are communicating primarily with nodes on other segments. If you identify these nodes and move them to the segments where their primary communications partners are, you may lessen the overall loading on your network.

### Off-segment Transit Frames

The number of frames observed on the selected segment not into or out of a node on the selected segment. For these frames, the selected segment is an intermediate hop in a route between the originating and destination

App. III - 2

- 116 -

segments. (This variable applies only to segments, not to nodes.)

### IP Variables

#### IP Packets

An IP packet or datagram is a string of bytes that is transferred as a unit across the IP network. It has two parts: the IP header, which contains control information such as the source and destination IP addresses; and the data to be transferred to the destination user.

#### Bytes

The Monitor maintains a count and rate for bytes into and out of all monitored objects. For example, you can monitor the number of bytes into or out of a chosen node to measure the traffic load with respect to that node. You can monitor the number of bytes into and out of all nodes on the segment.

#### IP Error Packets

An IP error packet is logged when the monitor observes a packet with an error in its IP header. Possible errors are as follows:

- \* IP header length is less than 20 bytes
- \* IP header length is greater than the length of the IP packet
- \* Packet length is less than the IP header length.
- \* If offset is set for fragmentation, but the frame should not be fragmented.

#### IP Fragments

If an IP datagram is too large to pass through a subnetwork or router, the IP router that is transmitting the original datagram divides it into fragment datagrams. The destination station reassembles the original datagram once it has received all the fragments.

Fragmentation usually occurs because packets are being routed through a network segment that has physical technology or configuration that restricts the IP datagram size to one smaller than the IP datagram size used on the originating segment.

App. III - 3



- 117 -

For example, the maximum frame size in an IEEE 802.5 physical network is 16000 octets, whereas the maximum frame size on an Ethernet physical network is about 1500 octets. In this case, a large frame originating on the IEEE 802.5 network would have to be divided into many fragments before it could be transmitted onto the Ethernet network.

Note that a fragment is a complete and correct IP datagram. Do not confuse IP fragments with the Ethernet fragment errors.

Higher than typical values for these parameters may mean that one or more commonly-used communications routes are forcing fragmentation to occur.

Example: new nodes have been added that access a server across a fragmenting route. The number of additional packets causes delays on the server's segment. The solution is to reconnect the new nodes to a different segment that has a non-fragmenting route to the server.

#### IP Header Bytes

The header is the portion of the IP packet that contains control information used by the protocol, such as source and destination IP addresses.

#### Broadcast and Multicast packets

A broadcast packet is special packet that is received by all stations on the network.

A multicast packet is a packet that is received by a predefined set of stations. Multicasting is used to send a message to a set of stations using a single packet.

#### IP Off-segment Packets

Off-segment packets are packets that the Monitor observes on the local segment, but are destined for, or originated by, stations not on the local segment. All off-segment packets, then, are either routed to, from, or across the local segment.

Off-segment values are a measure of the amount of routing or bridging that is occurring. Excessive off-segment traffic may mean that certain stations on one segment are communicating primarily with stations on other segments. If you identify these stations and

App. III - 4

- 118 -

move then to the segments where their primary communications partners are, you may lessen the overall loading on your network.

#### Off-segment Transit Packets

This parameter applies only to segment, not to nodes. The number of IP packets observed on the selected segment not destined for or originated by an object on the selected segment. For these packets, the selected segment is an intermediate hop in a route between the originating and destination segments.

#### Off-segment Transit Packets Rate

This parameter applies only to segments, not to nodes. The number of off-segment IP packets observed per second on the selected segment, not into or out of an object on the selected segment. For these packets, the selected segment is an intermediate hop in a route between the originating and destination segments.

### ICMP Variables

#### ICMP Packets

ICMP (Internet Control Message Protocol) packets are used to control, test, and report problems with, the network. Reading through the ICMP variable descriptions should give you a good idea of how ICMP is used. A high number of ICMP packets from any source wastes traffic capacity that could otherwise be used for data packets.

#### Bytes

The Monitor maintains a count and rate for the number of ICMP bytes in and out of all monitored objects. A high number of ICMP bytes from any source wastes traffic capacity that could otherwise be used for data.

#### ICMP Errors

An ICMP error is logged when the Monitor observes an ICMP packet with an error in its ICMP header. For example, a packet may have a length field with an illegal value in it. A node that generates ICMP errors may be having software problems.

App. III - 5

- 119 -

**Off-segment**

Off-segment packets are packets that the Monitor observes on the local segment that are destined for or sent by nodes not on the local segment. All off-segment packets are either routed to, from, or across the local segment.

A high number of ICMP packets from any source wastes traffic capacity that could otherwise be used for data packets. If there are a high number of in or transit off-segment ICMP packets, the source is on a different segment.

**Destination Unreachable Packets**

If for some reason a gateway cannot deliver an IP packet, it sends an ICMP Destination Unreachable packet to the sender. This packet informs the sender that the packet could not be delivered, and gives a reason. The Monitor keeps count of ICMP Destination Unreachable packets into and out of all objects, by reason. These are listed below.

**Net unreachable**

The network is having routing problems. Possible routing problems include: a non-operational link a node or router has an incorrect routing table

**Host unreachable**

See net unreachable.

**Protocol unreachable****Port unreachable****Frag needed / DF set**

This means fragmentation is needed but Don't Fragment flag was set. This message is sent when a router cannot forward a packet because it is too large for the next subnetwork in the route. Find out why fragmentation is being disallowed by the sending node - it may not be necessary. If it is necessary, then you must find or create an alternate route.

**Source route failed**

App. III - 6

- 120 -

**Destination net unknown**

The destination network is not in the router's current routing table. This may be because the source node entered the address incorrectly (a software problem) or because the router's routing table is corrupt or incomplete.

**Destination host unknown**

See destination net unknown

**Source host isolated**

**Destination net prohibited (communication with destination network administratively prohibited)**

**Net unreachable / TOS**

This means network is unreachable for this Type of Service. This message is sent when a router cannot forward a packet because the specified Type of Service is not available for this route. Find out why this Type of Service is being specified. It may be unnecessary.

**Host unreachable / TOS**

This means host is unreachable for this Type of Service.

**Time to Live Exceeded Packets**

An IP packet is allowed to remain in transit for a fixed time. This time is called "time to live" and is specified in the IP packet by the sender. If this time expires before the packet is delivered, the packet is discarded. This mechanism prevents packets that get "stuck" in circular routes from congesting the network forever.

This mechanism is enforced by the gateways that route the packet through the network. Each gateway reduces the packet's timer value by an appropriate amount, and then checks to make sure that it has not reached zero. If the timer has reached zero, the gateway discards the packet and transmits an ICMP Time to Live Count Exceeded packet back to the sender.

App. III - 7

- 121 -

Packets may get stuck in loops (circular routes) because a gateway or router has incorrect information in its routing table (example).

#### Reassembly Time Exceeded Packets

In routing an IP packet across a network, it is sometimes necessary to fragment it into smaller packets. This must be done to get it across a segment that cannot handle the packet at its original size.

Once a packet has been fragmented, it is not reassembled until the fragments reach the final destination. Since it is possible that one or more fragments will be lost before reaching the destination, the destination node waits only a fixed period of time to receive all the fragments. This is the reassembly time.

If the destination node has not received all of the fragments when the reassembly time expires, it sends an ICMP Fragment Reassembly Time Exceeded packet to the sender.

This problem typically occurs because one or more of the fragments has been lost.

#### Parameter Problem Packets

Part of each IP packet (the header) contains control information. A parameter is a unit of control information. For example, one parameter specifies the length of the packet, and another specifies whether or not fragmentation of this packet is allowed.

If a gateway detects a serious problem with a parameter, and it is not reportable through one of the other ICMP messages (such as Destination Unreachable), it sends an ICMP Parameter Problem packet back to the sender.

There is currently one specific reason tracked for the ICMP Parameter Problem packet:

Param option missing (missing option parameter)

#### Source Quench Packets

Gateways use the source quench mechanism to slow the rate of incoming packets. If a gateway is receiving packets too fast for it to keep up with, it will send

App. III - 8

- 122 -

an ICMP Source Quench Packet to one or more nodes to tell them to slow down.

#### Redirect Packets

The redirect mechanism allows gateways to send information about routes to hosts. This works as follows:

Each node maintains a table that contains, for each of the nodes with which it communicates, the physical address of a gateway. This gateway is the first step in the route to the destination node. When a node sends a datagram to a node that is not on its segment, it send it to the gateway indicating in its routing table for the destination node.

Gateways maintain more or less complete routing information. They check all datagrams to be routed off a segment to make sure that the optimum route is being used. For example, if there are two gateways available to Node a, and Node A attempts to send a datagram to Node B across Gateway 1 when Gateway 2 would be better, Gateway 1 will detect the problem.

When this occurs, the detecting gateway issues an ICMP Redirect packet to the sending node. This packet tells the node how it should change its routing table.

Nodes use this mechanism to learn routes from gateways. All a node really needs on startup is to know the address of a gateway. It attempts to route all of its off-segment messages through this gateway, and builds its routing table from the ICMP Redirect packets it receives back.

An ICMP Redirect packet contains a diagnostic code that specifies additional information. The Monitor counts the occurrences of each of these:

#### Redirect for net

This packet means that datagrams to nodes on this network should be routed differently.

#### Redirect for host

This packet means that a datagram to this host should be routed differently.

App. III - 9

- 123 -

**Redirect to TOS net**

This is a redirect for the network and type of service. This packet means that datagrams to hosts on this network should be routed differently in order to obtain this type of service.

**Redirect TOS host**

This is a redirect for the host and type of service. This packet means that a datagram to this host should be routed differently in order to obtain this type of service.

**Echo Packets**

The echo mechanism is used to verify that a destination is currently reachable, or to test the delay time between nodes. Echo is often referred to as "ping." The echo mechanism involves two ICMP packets: Echo Request and Echo Reply. The Monitor maintains counts for both of these.

Note that some diagnostic tools issue a series of ICMP Echo Request packets and then monitor and analyze the ICMP Echo Response packets.

A high number of these packets wastes traffic capacity.

**Echo Request**

This is a request that the addressed node send back an Echo Response packet.

**Echo Response**

This is a response packet sent by a node when it has received an Echo Request packet.

**Timestamp Packets**

The timestamp mechanism is used by nodes to synchronize their clocks. Node A sends an ICMP Timestamp Request packet to Node B, requesting that Node B return the current time of its system clock. Node B sends an ICMP Timestamp Response packet with the requested time to Node A. Node A can roughly synchronize its clock with Node B based on the response timestamp.

App. III - 10

- 124 -

**Timestamp Request**

This is a request that the addressed node send back a Timestamp Response packet.

**Timestamp Response**

This is a response packet sent by a node when it has received a Timestamp Request packet.

**Address Mask Packets**

The IP protocol's addressing scheme allows sites to group multiple physical networks (segments) into a single addressable subnet. The subnet addressing scheme allows a site to determine, to an extent, which IP address bits identify the network (including subnet) and which identify nodes in the local subnet. For example, a site may determine that the first three octets in the IP address specify the network, and the last octet specifies the node in the network.

The division of address bits between network and node is represented by an address mask. The address mask is a string of 32 bits, where each bit used to specify network is set to 1, and bits that identify node are set to 0.

A node learns the address mask for its local subnet by requesting the information from a gateway. To do so it sends an ICMP Address Mask Request message to the gateway. If it does not know the address of the gateway, it may broadcast the request. The gateway replies with an ICMP Address Mask Response.

**Address Mask Request**

This is a request that the addressed node send back an Address Mask Response packet.

**Address Mask Response**

This is a response packet sent by a node when it has received an Address Mask Request packet.

**TCP Variables****TCP Packets**

A TCP packet (sometimes referred to as a segment) is a string of bytes that is transferred as a unit across

App. III - 11



- 125 -

the IP network. It has two parts: the TCP header, which contains control information such as the source and destination TCP ports; and the data to be transferred to the destination user.

### Bytes

The Monitor maintains a count and rate for bytes into and out of all monitored objects. For example, you can monitor the number of bytes into or out of a chosen node to measure the traffic load with respect to that node. You can monitor the number of bytes into and out of all nodes on the segment. The byte count includes header and data bytes.

### Header Bytes

The header is the portion of the TCP packet that contains control information used by the protocol, such as source and destination TCP ports. Comparing the number of TCP header bytes to the total number of TCP bytes gives an idea of the amount of TCP overhead on a connection.

### Error Packets

A TCP error is logged for each packet observed with one of the following problems:

- \* length of TCP packet is less than 20 bytes
- \* TCP Header length is less than 20 bytes
- \* TCP header length is greater than the length of the TCP packet
- \* TCP header length is greater than 20 bytes but the length of the TCP packet is less than the TCP header length.

### Retransmissions

A Retransmission is a TCP packet that contains some data that has already been sent at least once. A Retransmission may or may not be an exact duplicate of the packet already transmitted.

Note that if the underlying packet delivery system (DLL) creates a duplicate, it is counted as a retransmission.

When a TCP entity sends a data packet to its remote partner, it waits a predetermined period of time (tracked by a retransmission timer) for an acknowledgement (ACK) from the remote partner. If this

App. III - 12

- 126 -

time expires without the ACK being received, it retransmits the data contained in the presumably lost packet. It may retransmit a packet identical to the one lost, or it may combine data from multiple lost packets into a new packet, or it may combine lost data with new data into a new packet.

Excessive retransmissions can mean that a gateway is overloaded or down, that a system is overloaded, or that network parameters are misconfigured. In general, small dedicated networks should see few retransmissions. Larger, more diverse networks with routers, bridges and gateways with different capabilities and capacities are likely to have more retransmissions.

#### Bytes Retransmitted

Byte Retransmitted are TCP data bytes that have already been sent at least once.

See Retransmissions.

#### Out of Order Packets

Out of Order Packets are packets containing bytes with lower sequence numbers than bytes in previously seen packets.

Packets do not necessarily arrive in the order they were sent in. The receiving node puts the data in the correct order once it has received all packets. A high value may mean that some packets are being sent by way of a slower route, or that there is an overloaded or down bridge or router.

#### Out of Order Bytes

Out of Order Bytes are bytes with lower sequence numbers than bytes seen in previous packets.

#### Data out of Window Packets

Data out of Window Packets are packets that contains data that is not within the boundaries of the receiving partner's currently advertised window. The data is either acknowledged data or data that the partner is not ready to receive.

App. III - 13

- 127 -

### Bytes out of Window

Bytes out of Window are bytes that are not within the boundaries of the receiving partner's currently advertised window. The data is either acknowledged data or data that the partner is not ready to receive.

### Packets after Close

Packets after Close are packets observed after a connection has been closed. These may be packets that had been "lost" on the network, or it may indicate a malfunction in the sending station.

### RST Packets

A packet in which the RST (reset) bit is set.

### SYN Control Packets

A packet in which the SYN bit is set.

### FIN Control Packets

A packet in which the FIN bit is set.

### URG Control Packets

An URG Control Packet is a packet in which the Urgent pointer is set.

The packet contains data that the receiving application should process as soon as possible. For example, the control-key sequences used by some applications are often sent as Urgent data.

### Keepalives

A Keepalive is a TCP packet that a user sends to check to see if a connection is still active. The Keepalive packet contains either not data or one garbage byte of data that is outside the remote partner's last advertised window. The remote partner responds with either an ACK, confirming that the connection is alive, or a RST, indicating that the connection had been dropped.

Although widely implemented, the keepalive mechanism is not part of the TCP protocol, so you will not necessarily see keepalive activity.

App. III - 14

- 128 -

Keepalives mean that a connection has been up for a long time without and activity. Resources may be unnecessarily tied up.

#### Window Probes

A Window Probe is a TCP packet that is sent to check the size of the remote partner's window when the last advertised window size was zero. The Window Probe packet contains one byte of data. The remote partner responds with an ACK packet, which contains the size of the remote partner's current window size.

Non-data packets, which may include window update information, may be lost and are not be retransmitted. It may therefore become necessary to check the remote partner's window size if that information has not been received for some period of time. This can mean that a node is runnind a faulty TCP implementation, that timers are misconfigured, or packets are being lost.

#### Window Update Only Packets

A Window Update Only packet is a packet that contains no data, but in which the advertised window size has been updated.

App. III - 15

## APPENDIX IV

## Summary Tool - Values Display Fields

---

Packet Rate	total packets per second at this protocol layer received and transmitted at segment or node
Byte Rate	total bytes per second at this protocol layer received and transmitted at segment or node
Errors	total errors at this protocol layer received and transmitted at segment or node
Broadcast Pkt Rate	total number packets per second at this protocol layer addressed to broadcast address
Multicast Pkt Rate	total number packets per second at this protocol layer addressed to multicast address
Source Quenches	total number of ICMP source quench packets received and transmitted from this segment or node.
Fragments	total number of IP fragmented packets received and transmitted from this segment or node.
Flow Controls	
UDP	total number of ICMP source quench packets received and transmitted on this UDP port.
TCP	total number of ICMP source quench packets received and transmitted on this TCP port.
NFS	total number of ICMP source quench packets received and transmitted on this NFS port.
Retransmissions	total number of TCP packets retransmitted on this TCP port.
Off Segment Packets	
in	% traffic at this protocol layer received by nodes on this segment originating from other segments  in = 100(packet rate / packet rate rcv from off seg)
out	% traffic at this protocol layer transmitted by nodes on this segment to nodes on other segments  out = 100(packet rate / packet rate xmt to off seg)
Transit	% traffic at this protocol layer originating from other segments which are addressed to nodes not on this segment  transit = 100(packet rate / packet rate transit)
Local	% Traffic at this protocol layer which originates and terminates on this segment  local = 100 -(in + out + transit)
Most Active Protocols	The five most active protocols running above this layer (ie the users of this layer). The protocols are displayed as % and ranked in decreasing order.  protocol % = 100(protocol packet rate/packet rate)

- 130 -

- Most Active Nodes**      The five most active nodes at this protocol layer. The nodes are displayed as % and ranked in decreasing order.  
node % =  $100(\text{node packet rate}/\text{packet rate})$
- ICMP Types Seen**      The total number of these specific ICMP packet types transmitted and received on this segment or node.
- Total Segment Bandwidth**      The % of the available bandwidth used by this protocol. If the screen is a segment display it is % used by all nodes on the segment, if it is a node display it is the % used by that node.  
% =  $100(8 * \text{frame rate} / 10000000)$
- Total Active Dialogs**      The number of dialogs detected for the node or segment at this protocol layer.
- 

APP.1W - 2

## 5. Actual Screens for Values Tool

## APPENDIX V

**5.1 Data Link Group**5.1.1 Definition

This screen summarizes the data link parameters.

5.1.2 Defaults

- 1 This is a "complete values" screen. It shows all of the values for the DLL protocol layer.
- 2 The user comes from a context of a specific segment or node and this screen must preserve that context.

5.1.3 Primary Screen Layout

---

Standard Column Headings

---

Frames	Rcv
	Xmt
	Total
Frm rate	Rcv
	Xmt
	Total
Bytes	Rcv
	Xmt
	Total
Byte rate	Rcv
	Xmt
	Total
Errors	Rcv
	Xmt
	Total
Error rate	Rcv
	Xmt
	Total
802.3 frames	Rcv
	Xmt
	Total
ethernet frames	Rcv
	Xmt
	Total
802.3 frame rate	Rcv
	Xmt
	Total
ethernet frame rate	Rcv
	Xmt
	Total
Bcast Xmt	
Bcast rate	
Mcast Xmt	
Mcast rate	
Off seg	Rcv
	Xmt
	[Transit]



[local]  
 Total  
 Off seg rate  
 Rcv  
 Xmt  
 [Transit]  
 [local]

Total  
 Runts Xmt  
 [Alignment]  
 [Collisions]

Protocol	Pkt Count	Pkt Rate	%
Protocol 1			
Protocol 2			
...			
Protocol n			

5.1.4 Secondary Screen Layout

Extended Column Headings

rows as for primary screen

**5.2 IP Group**

5.2.1 Definition

This screen provides information for the IP network layer running on the segment or node.

5.2.2 Defaults

- 1 This is a "complete values" screen. It shows all of the values for the IP protocol type
- 2 The user comes from a context of a specific segment or node and this screen must preserve that context

5.2.3 Primary Screen Layout

---

Standard Column Headings

---

Pkts  
 Pkt rate  
 Bytes  
 Byte rate  
 Errors  
 Error rate  
 Frags  
 Frag rate  
 Header bytes  
 Header rate  
 Bcast Xmt  
 Bcast rate  
 Mcast Xmt  
 Mcast rate  
 Off seg  
 Off seg rate

---

Protocol	Pkt Count	Pkt Rate	%
Protocol 1			
Protocol 2			
.			
.			
Protocol n			

5.2.4 Secondary Screen Layout

---

Extended Column Headings

---

rows as for primary screen

**5.3 ICMP Group**

5.3.1 Definition

This screen provides information for the ICMP protocol s/w running on the segment or node.

5.3.2 Defaults

- 1 This is a "complete values" screen. It shows all of the values for the ICMP protocol type
- 2 The user comes from a context of a specific segment or node and this screen must preserve that context.

5.3.3 Primary Screen Layout

---

**Standard Column Headings**

---

Pkts  
Pkt rate  
Bytes  
Byte rate  
Errors  
Error rate  
Off seg  
Off seg rate  
D.U. net  
D.U. host  
D.U. Prot  
D.U. port  
D.U. frag  
D.U. Src route  
D.U. Net Unk.  
D.U. Host Unk.  
D.U. Src Host Isol.  
D.U. Dnet Ad Prob  
D.U. Dhost Ad Prob  
D.U. Net Unr.  
D.U. Time Xd Trans  
D.U. Time Xd Reass  
Param prob  
Param opt miss.  
src quench  
redir net  
redir host  
redir tos net  
redir tos host  
Echo req  
Echo Resp  
Ts req  
Ts resp  
Addr mask req  
Addr mask resp

5.3.4 Secondary Screen Layout

---

Extended Column Headings

---

rows as for primary screen

**5.4 UDP Group**

5.4.1 Definition

This screen provides information for the UDP protocol s/w running on the segment or node.

5.4.2 Defaults

- 1 This is a "complete values" screen. It shows all of the values for the UDP protocol type
- 2 The user comes from a context of a specific segment or node and this screen must preserve that context.

5.4.3 Primary Screen Layout

---

Standard Column Headings

---

Pkts  
 Pkt rate  
 Bytes  
 Byte rate  
 Errors  
 Error rate  
 Header bytes  
 Header rate  
 off seg  
 off seg rate

---

Protocol	Pkt Count	Pkt Rate	%
----------	-----------	----------	---

---

Protocol 1  
 Protocol 2

Protocol n

5.4.4 Secondary Screen Layout

---

Extended Column Headings

---

rows as for primary screen

## **5.5 TCP Group**

### 5.5.1 Definition

This screen provides information for the TCP protocol s/w running on the segment or node.

### 5.5.2 Defaults

- 1 This is a "complete values" screen. It shows all of the values for the TCP protocol type
- 2 The user comes from a context of a specific segment or node and this screen must preserve that context

5.5.3 Primary Screen Layout

---

**Standard Column Headings**

---

number connections  
Pkts  
Pkt rate  
bytes  
Byte rate  
header bytes  
Hdr byt rt  
errors  
Error rate  
persists  
keep alives  
rexmits  
bytes rexit  
ack only pkt  
window probes  
pkts urg only  
window update only  
control pkts  
dup only pkts  
part dup pkts  
dup bytes  
out order pkts  
out order bytes  
data pkts after window  
bytes after window  
pkts after close  
dup acks  
ack pkts  
off seg  
off seg rate

---

Protocol	Pkt Count	Pkt Rate	%
Protocol 1			
Protocol 2			
...			
Protocol n			

5.5.4 Secondary Screen Layout

---

**Extended Column Headings**

---

rows as for primary screens

## 5.6 NFS Group

### 5.6.1 Definition

These screens provide information for the NFS protocol s/w running on the segment or node. The screens show the breakdown of activity by servers and clients for filesystems, directories and files.

### 5.6.2 Defaults -client/server

- 1 This is a "complete values" screen. It shows all of the values for the NFS protocol type
- 2 The user comes from a context of either a segment or a node and this screen must preserve that context.

- 140 -

5.6.3 Primary Screen Layout -client/server

---

**Standard Column Headings**

---

total nfs ops  
nfs ops rate  
read opss  
read rate  
write ops  
bytes read  
bte read rate  
bytes written  
bytes written rate  
write rate  
write cache  
create file  
remove file  
rename file  
create dir  
remove dir  
null ops  
get file attr  
set file attr  
look ups  
read link  
create link  
create sym lnk  
get fsys attr  
mount  
unmount  
readmount  
unmountall  
readexport

---

**File Systems on Server**

---

file system 1  
file system 2

file system n

5.6.4 Secondary Screen Layout

---

**Extended Column Headings**

---

rows as for primary screens

**5.6.5 Navigation**

APPENDIX V - 10



- 141 -

Double clicking on a file system invokes the file system screen for the selected file system.

#### 5.6.6 Defaults -file system

- 1 This is a "complete values" screen. It shows all of the values for the NFS protocol type for this file system.
- 2 The user comes from a context of either an nfs client or server and this screen must preserve that context.

APPENDIX V - 11

5.6.7 Primary Screen Layout -file system

---

**Standard Column Headings**

---

total nfs ops  
nfs ops rate  
read ops  
read op rate  
write ops  
write op rate  
bytes read  
bte read rate  
bytes written  
bytes written rate  
write cache  
create file  
remove file  
rename file  
create dir  
remove dir  
null ops  
get file attr  
set file attr  
look ups  
read link  
create link  
create sym lnk  
get fsys attr  
mount  
unmount

---

**Directories in File System**

---

directory 1  
directory 2  
.  
.  
directory n

5.6.8 Secondary Screen Layout

---

**Extended Column Headings**

---

rows as for primary screens

**5.6.9 Navigation**

Double clicking on a directory invokes the directory screen for the selected directory.

5.6.10 Defaults -directory

- 143 -

- 1 This is a "complete values" screen. It shows all of the values for the NFS protocol type for this directory.
- 2 The user comes from a context of an nfs file system and this screen must preserve that context.

5.6.11 Primary Screen Layout -directory

Standard Column Headings
total nfs ops
nfs ops rate
read ops
read ops rate
write ops
write ops rate
bytes read
bte read rate
bytes written
bytes written rate
write cache
create file
remove file
rename file
null ops
get file attr
set file attr
look ups
read link
create link
create sym lnk
create sym lnk
Attributes
type
mode
nlinks
uid
gid
size
blocksize
rdev
blocks
fileid
atime
mtime
ctime
Files in Directory
file 1
file 2
file n

5.6.12 Secondary Screen Layout

---

**Extended Column Headings**

---

rows as for primary screens

---

**5.6.13 Navigation**

Double clicking on a file invokes the file screen for the selected file.

**5.6.14 Defaults -file**

- 1 This is a "complete values" screen. It shows all of the values for the NFS protocol type for this file.
- 2 The user comes from a context of an nfs file directory and this screen must preserve that context.

- 146 -

5.6.15 Primary Screen Layout -file

---

**Standard Column Headings**

---

total nfs ops  
nfs ops rate  
read ops  
read ops rate  
write ops  
write ops rate  
bytes read  
bte read rate  
bytes written  
bytes written rate  
write cache  
null ops  
get file attr  
set file attr  
look ups  
read link  
create link  
create sym lnk

---

**Attributes**

---

type  
mode  
nlinks  
uid  
gid  
size  
blocksize  
rdev  
blocks  
fileid  
atime  
mtime  
ctime

5.6.16 Secondary Screen Layout

---

**Extended Column Headings**

---

rows as for primary screens

**5.7 ARP Group**

APPENDIX V - 16

- 147 -

5.7.1 Definition

This screen provides information for the ARP protocol s/w running on the segment or node.

5.7.2 Defaults

- 1 This is a "complete values" screen. It shows all of the values for the ARP protocol type
- 2 The user comes from a context of either a segment or a node and this screen must preserve that context.

APPENDIX V - 17

5.7.3 Primary Screen Layout

---

**Standard Column Headings**

---

TBD

5.7.4 Secondary Screen Layout

---

**Extended Column Headings**

---

rows as for primary screens

**5.8 RARP Group**5.8.1 Definition

This screen provides information for the RARP protocol s/w running on the segment or node.

5.8.2 Defaults

- 1 This is a "complete values" screen. It shows all of the values for the RARP protocol type
- 2 The user comes from a context of either a segment or a node and this screen must preserve that context.



5.8.3 Primary Screen Layout


---

 Standard Column Headings
 

---

TBD

5.8.4 Secondary Screen Layout


---

 Extended Column Headings
 

---

rows as for primary screens

**5.9 Telnet Group**5.9.1 Definition

This screen provides information for the Telnet protocol s/w running on the segment or node.

5.9.2 Defaults

- 1 This is a "complete values" screen. It shows all of the values for the Telnet protocol type
- 2 The user comes from a context of either a segment or a node and this screen must preserve that context.

5.9.3 Primary Screen Layout


---

 Standard Column Headings
 

---

TBD

5.9.4 Secondary Screen Layout


---

 Extended Column Headings
 

---

rows as for primary screens

**5.10 FTP Group**5.10.1 Definition

This screen provides information for the FTP protocol s/w running on the segment or node.

- 150 -

5.10.2 Defaults

- 1 This is a "complete values" screen. It shows all of the values for the FTP protocol type
- 2 The user comes from a context of either a segment or a node and this screen must preserve that context.

5.10.3 Primary Screen Layout


---

Standard Column Headings

---

TBD

---

5.10.4 Secondary Screen Layout


---

Extended Column Headings

---

rows as for primary screens

**5.11 Dialogue Data Group**5.11.1 Definition

This screen displays all of the Data available for a particular dialogue. This screen is shown when the user clicks on an entry in the Summary Tool dialogue information.

Each dialog screen represents a single dialog. Thus at the UDP or TCP level two nodes may have multiple dialogs (each with a unique port pair) and each of these will be represented as a separate entity.

Because the user cannot uniquely identify the dialog he requires from the menus (he does not know the port numbers involved) the only mechanism to invoke these screens is by selection of a dialog from the appropriate summary screen. This problem also prevents the user from 'clicking' through all the dialogs on ports between a node pair (may be addressed in later phase).

5.11.2 Defaults

- 1 This is a "complete values" screen. It shows all of the values available for the selected connection.
- 2 There are several different contexts for this screen. The user may select this option from the summary tools for all protocols. This screen must reflect the node, layer and specific connection context from which the user entered

APPENDIX V - 20

- 151 -

The content of this screen is essentially the same as the corresponding row entry from the Traffic matrix screen for the DLL and IP layers. Their inclusion is to provide the user with a consistent navigation paradigm across the layers (and to provide this functionality in release 1 which does not include the Traffic matrix support).

The data set displayed in this screen will be appropriate to the protocols used between the nodes. The variables shown are those selected for TCP/IP protocols. Where nodes converse using multiple protocols this will be expanded to select data from each protocol set.

APPENDIX V - 21

5.11.3 Primary Screen -DLL

node name	node name
mac address	mac address
ip address	ip address
Network Protocols:	
start time	last seen time

Standard Column Headings

---

frames  
bytes  
errors  
flow ctl  
ip frags  
tcp retransmissions

5.11.4 Secondary Screen Layout -DLL

Extended Column Headings

rows as for primary screens

5.11.5 Primary Screen -IP

node name	node name
mac address	mac address
ip address	ip address
Transport Protocols:	
start time	last seen time

Standard Column Headings

---

Pkts  
bytes  
header bytes  
errors  
fragments  
TCP retransmissions  
ICMP

5.11.6 Secondary Screen Layout -IP

Extended Column Headings

rows as for primary screens

5.11.7 Primary Screen -ICMP

This is invoked by selection of the ICMP entry from the IP screen.

node name	node name
mac address	mac address
ip address	ip address

Standard Column Headings

- Pkts
- Bytes
- Errors
- Off seg
- D.U. net
- D.U. host
- D.U. Prot
- D.U. port
- D.U. frag
- D.U. Src route
- D.U. Net Unk.
- D.U. Host Unk.
- D.U. Src Host isol.
- D.U. Dnet Ad Prob
- D.U. Dhost Ad Prob
- D.U. Net Unr.
- D.U. Time Xd Trans
- D.U. Time Xd Reass
- Param prob
- Param opt miss.
- src quench
- redir net
- redir host
- redir tos net
- redir tos host
- Echo req
- Echo Resp
- Ts req
- Ts resp
- Addr mask req
- Addr mask resp

5.11.8 Secondary Screen Layout

Extended Column Headings

rows as for primary screens

5.11.9 Primary Screen -UDP

node name	node name
mac address	mac address
ip address	ip address
port number	port number
Application Protocol:	
start time	last seen time

Standard Column Headings

---

Pkts  
bytes  
errors  
ip frags  
flow ctl

5.11.10 Secondary Screen Layout -UDP

Extended Column Headings

rows as for primary screens

5.11.11 Primary Screen -TCP

node name	node name
mac address	mac address
ip address	ip address
port number	port number
Application Protocol:	
Connection Status: [active, closed-ok, closed reset, unknown]	
start time	last seen time

---

Standard Column Headings

---

Pkts  
 bytes  
 header bytes  
 errors  
 pkts bad seq #  
 bytes not acked  
 persists  
 keep alives  
 pkts retransmit  
 bytes retransmit  
 ack only pkt  
 window probes  
 pkts urg only  
 window update only  
 control pkts  
 dup only pkts  
 part dup pkts  
 dup bytes  
 out order pkts  
 out order bytes  
 data pkts after window  
 bytes after window  
 pkts after close  
 dup acks  
 acks unseq data  
 ack pkts  
 bytes acked by acks  
 current window

5.11.12 Secondary Screen Layout -TCP

---

Extended Column Headings

---

rows as for primary screens

5.11.13 Primary Screen -NFS

node name	node name
mac address	mac address
ip address	ip address
port number	port number
start time	last seen time

---

Standard Column Headings

---

variables as for NFS Group

5.11.14 Secondary Screen Layout -NFS

---

Extended Column Headings

---

rows as for primary screens

5.11.15 Navigation

As for NFS group a hierarchy of screens is available:

- 1 client to server
- 2 client to file system
- 3 client to directory
- 4 client to file

**5.12 Traffic Matrix Group (Not in release 1)**

5.12.1 Definition

This screen shows traffic distribution between a selected node (or segment) and other nodes (or segments) in the network.

For the DLL and IP layers it is essentially a repeat of the dialogue screens. For the UDP and TCP layers however it represents a summation over multiple connections between the two nodes.

5.12.2 Defaults



- 157 -

- 1 The user comes from a context of a specific segment or node plus a protocol level and this screen must preserve this context.
- 2 If the selection propagated from the Summary Tool is a segment then the distribution is segment to segment. If the selection is a node then the distribution is node to node.
- 3 Values are shown in order of heaviest traffic to lightest.
- 4 The initial screen has the heaviest pairs of nodes or segments. Scrolled screens contain progressively lighter traffic loads.
- 5 The user can select the column by which the nodes are to be ordered and request reordering. This allows the user to use this screen look at flow control for example.
- 6 Double clicking on a node or segment in the display area allows the user to move to this object as the focus of the traffic matrix ie if the user is looking at a matrix for node A and selects node B (which is one of the nodes in the matrix) they will get the traffic matrix for B.
- 7 Double clicking on the node which is the focus of the matrix (eg A in the above example) selects the next segment or node, consistent with the current view. Node views click to other nodes on the segment. Segment views click to other segments. The segment (or) node selection will be ordered alphabetically.
- 8 The data maintained between two nodes (or segments) will be aged out if no communication between them occurs for a defined period (settable by the user -eventually).

APPENDIX V - 27

5.12.3 Primary Screen DLL

Node(Segment) Name

	frm	frm	byte	byte	err	err	flow	flow	tffc
	rate	rate	rate	rate	rate	rate	ctl	ct rt	%
node(segment)1									
node(segment)2									
.									
.									
node(segment)n									

This scrolls down to accomodate all nodes (or segments) required.

5.12.4 Secondary Screen

	frag	frag	tcp	tcp
	rate	rate	retrans	retrans
rows as primary screen				

5.12.5 Primary Screen IP

Node(Segment) Name

	pkt	pkt	err	err	frag	frag	icmp	flw	flw	tffc
	rate	rate	rate	rate	rate	rate	rate	ctl	ct rt	%
node(segment)1										
node(segment)2										
.										
.										
node(segment)n										

This scrolls down to accomodate all nodes (or segments) required.

5.12.6 Primary Screen ICMP

This is invoked by selection of the ICMP entry for a node (segment) pair. The user is vectored to the IP traffic matrix screen in this case.

5.12.7 Primary Screen TCP

Node(Segment) Name

	pkt	pkt	err	err	act	rxmt	rxmt	flw	flw	tlfc	#
		rate		rate	conn		rate	ctl	ct rt	%	conns
node(segment)1											
node(segment)2											
.											
.											
node(segment)n											

This scrolls down to accomodate all nodes (or segments) required.

5.12.8 Primary Screen UDP

Node(Segment) Name

	pkt	pkt	err	err	actv	flow	flow		ttc
		rate		rate	conn	ctl	ctl rt	%	
node(segment)1									
node(segment)2									
.									
.									
node(segment)n									

This scrolls down to accomodate all nodes (or segments) required.

5.12.9 Primary Screens NFS

5.12.9.1 Client to Server

Node(Segment) Name

	pkt	pkt	err	err	actv	flow	flow	ttc
		rate		rate	conn	ctl	ctl rt	%
node(segment)1								
node(segment)2								
.								
.								
node(segment)n								

File systems on this node

file system 1  
 file system 2  
 .  
 file system n

This scrolls down as required.

5.12.9.1.1 Navigation

Double clicking on a file system invokes the file system screen for the selected file system.

5.12.9.2 Client to File System

Node(Segment) Name  
File System name

---

	pkt	pkt	err	err	actv	flow	flow	ttc
		rate		rate	conn	ctl	ctl	rt %
node(segment)1								
node(segment)2								
.								
.								
node(segment)n								

---

Directories on this file system

---

directory 1  
directory 2  
.  
.  
directory n

This scrolls down as required.

5.12.9.2.1 Navigation

Double clicking on a directory invokes the directory screen for the selected directory.

5.12.9.3 Client to Directory

Node(Segment) Name  
File System name  
directory name

---

	pkt	pkt	err	err	actv	flow	flow	ttc
		rate		rate	conn	ctl	ctl	rt %
node(segment)1								
node(segment)2								
.								
.								
node(segment)n								

---

files in this directory

---

file 1  
file 2  
.  
.  
file n

This scrolls down as required.

5.12.9.3.1 Navigation

Double clicking on a file invokes the file screen for the selected file.

4.12.9.7 Client to File  
 Node(Segment) Name  
 File System name  
 directory name  
 file name

	pkt	pkt	err	err	actv	flow	flow	tffc
		rate		rate	conn	ctl	ctl	rt %
node(segment)1								
node(segment)2								
.								
.								
node(segment)n								

This scrolls down as required.

5.13 Summary Screen for Traffic Matrix

	Seg1	Seg2	Seg3	.....	Segn
Seg1		frame byte error	frame byte error		frame byte error
Seg2	frame byte error		frame byte error		frame byte error
Seg3	frame byte error	frame byte error			frame byte error
.					
.					
Segn	frame byte error	frame byte error	frame byte error	..... ..... .....	



- 165 -

Claims

1           1. A method for monitoring communications which  
2 occur in a network of nodes, each communication being  
3 effected by a transmission of one or more packets among  
4 two or more communicating nodes, each communication  
5 complying with a predefined communication protocol  
6 selected from among protocols available in said network,  
7 said method comprising  
8           detecting passively and in real time the contents  
9 of packets, and  
10           deriving, from said detected contents of said  
11 packets, communication information associated with  
12 multiple said protocols.

1           2. The method of claim 1 wherein said step of  
2 deriving communication information includes deriving  
3 communication information from associated with multiple  
4 layers of at least one of said protocols.

1           3. A method for monitoring communication dialogs  
2 which occur in a network of nodes, each dialog being  
3 effected by a transmission of one or more packets among  
4 two or more communicating nodes, each dialog complying  
5 with a predefined communication protocol selected from  
6 among protocols available in said network, said method  
7 comprising  
8           detecting the contents of packets, and  
9           deriving from said detected contents of said  
10 packets, information about the states of dialogs  
11 occurring in said network and which comply with different  
12 selected protocols available in said network.

1           4. The method of claim 3 wherein said step of  
2 deriving information about the states of dialogs  
3 comprises

- 166 -

4           maintaining a current state for each dialog, and  
5           updating the current state in response to the  
6           detected contents of transmitted packets.

1           5. The method of claim 3 wherein said step of  
2           deriving information about the states of dialogs  
3           comprises

4           maintaining, for each dialog, a history of events  
5           based on information derived from the contents of  
6           packets, and

7           analyzing the history of events to derive  
8           information about the dialog.

1           6. The method of claim 5 wherein said step of  
2           analyzing the history includes counting events.

1           7. The method of claim 5 wherein said step of  
2           analyzing the history includes gathering statistics about  
3           events.

1           8. The method of claim 5 further comprising  
2           monitoring the history of events for dialogs which  
3           are inactive, and  
4           purging from the history of events dialogs which  
5           have been inactive for a predetermined period of time.

1           9. The method of claim 4 wherein said step of  
2           deriving information about the states of dialogs  
3           comprises  
4           updating said current state in response to  
5           observing the transmission of at least two data related  
6           packets between nodes.

1           10. The method of claim 5 wherein said step of  
2           analyzing the history of events comprises

- 167 -

3 analyzing sequence numbers of data related packets  
4 stored in said history of events, and  
5 detecting retransmissions based on said sequence  
6 numbers.

1 11. The method of claim 4 further comprising  
2 updating the current state based on each new  
3 packet associated with said dialog, and  
4 if an updated current state cannot be determined,  
5 consulting information about prior packets associated  
6 with said dialog as an aid in updating said state.

1 12. The method of claim 5 further comprising  
2 searching said history of events to identify the  
3 initiator of a dialog.

1 13. The method of claim 5 further comprising  
2 searching the history of events for packets which  
3 have been retransmitted.

1 14. The method of claim 4 wherein  
2 the full set of packets associated with a dialog  
3 up to a point in time completely define a true state of  
4 the dialog at that point in time,  
5 said step of updating the current state in  
6 response to the detected contents of transmitted packets  
7 comprises generating a current state which may not  
8 conform to the true state.

1 15. The method of claim 5 wherein the step of  
2 updating the current state comprises updating the current  
3 state to "unknown".

1 16. The method of claim 14 further comprising  
2 updating the current state to the true state based on

- 168 -

3 information about prior packets transmitted in the  
4 dialog.

1 17. The method of claim 15 further comprising  
2 updating the current state to the true state based on  
3 information about prior packets transmitted in the  
4 dialog.

1 18. The method of claim 3 wherein said step of  
2 deriving information about the states of dialogs  
3 occurring in said network comprises parsing said packets  
4 in accordance with more than one but fewer than all  
5 layers of a protocol.

1 19. The method of claim 3 wherein each said  
2 communication protocol includes multiple layers, and each  
3 dialog complies with one of said layers.

1 20. The method of claim 3 wherein said protocols  
2 include a connectionless-type protocol in which the state  
3 of a dialog is implicit in transmitted packets, and said  
4 step of deriving information about the states of dialogs  
5 includes inferring the states of said dialogs from said  
6 packets.

1 21. The method of claim 4 further comprising  
2 parsing said packets in accordance a protocol and  
3 temporarily suspending parsing of some layers of  
4 said protocol when parsing is not rapid enough to match  
5 the rate of packets to be parsed.

1  
2 22. A method of analyzing the performance of a  
3 network of nodes which communicate via dialogs, each  
4 dialog being effected by a transmission of one or more  
5 packets among two or more communicating nodes, each

- 169 -

6 dialog complying with a predefined communication protocol  
7 selected from among protocols available in said network,  
8 said method comprising  
9 monitoring the operation of the network with  
10 respect to specific items of performance during normal  
11 operation,  
12 generating a model of said network based on said  
13 monitoring, and  
14 setting acceptable threshold levels for said  
15 specific items of performance based on said model.

1 23. The method of claim 22 further comprising  
2 monitoring the operation of the network with  
3 respect to the specific items of performance during  
4 periods which may include abnormal operation.

1 24. Apparatus for monitoring communication  
2 dialogs which occur in a network of nodes, each dialog  
3 being effected by a transmission of one or more packets  
4 among two or more communicating nodes, each dialog  
5 complying with a predefined communication protocol  
6 selected from among protocols available in said network,  
7 said apparatus comprising  
8 a monitor connected to the network medium for  
9 passively, and in real time, monitoring transmitted  
10 packets and storing information about dialogs associated  
11 with said packets, and  
12 a workstation for receiving said information about  
13 dialogs from said monitor and providing an interface to a  
14 user.

1 25. The apparatus of claim 24 wherein said  
2 workstation further comprises  
3 means for enabling a user to observe events of  
4 active dialogs.

- 170 -

1           26. Apparatus for monitoring packet  
2       communications in a network of nodes in which  
3       communications may be in accordance with multiple  
4       protocols, said apparatus comprising  
5           a monitor connected to a communication medium of  
6       the network for passively, and in real time, monitoring  
7       transmitted packets of different protocols and storing  
8       information about communications associated with said  
9       packtes, said communications being in accordance with  
10       different protocols, and  
11           a workstation for receiving said information about  
12       said communciations from said monitor and providing an  
13       interface to a user,  
14           said monitor and said workstation including means  
15       for relaying said information about multiple protocols  
16       with respect to communication in said different protocols  
17       from said monitor to said workstation in accordance with  
18       a single common network management protocol.

1           27. A method of diagnosing communication problems  
2       between two nodes in a network of nodes interconnected by  
3       links, comprising  
4           monitoring the operation of the network with  
5       respect to specific items of performance during normal  
6       operation,  
7           generating a model of normal operation of said  
8       network based on said monitoring, and  
9           setting acceptable threshold levels for said  
10       specific items of performance based on said model.

1           28. The method of claim 27 further comprising the  
2       steps of  
3           monitoring the operation of the network with  
4       respect to the specific items of performance during  
5       periods which may include abnormal operation, and

- 171 -

6           when abnormal operation of the network with  
7   respect to communication between the two nodes is  
8   detected, diagnosing the problem by separately analyzing  
9   the performance of each of the nodes and each of the  
10  links connecting the two nodes to isolate the abnormal  
11  operation.

1           29. A method of timing the duration of a  
2   transaction of interest occurring in the course of  
3   communication between nodes of a network, the beginning  
4   of said transaction being defined by the sending of a  
5   first packet of a particular kind from one node to the  
6   other, and the end of said transaction being defined by  
7   the sending of another packet of a particular kind  
8   between the nodes, comprising  
9        passively and in real time monitoring packets  
10  transmitted in the network,  
11       beginning to time said transaction upon the  
12  appearance of said first packet,  
13       determining when the other packet has been  
14  transmitted, and  
15       ending the timing of the duration of the  
16  transaction upon the appearance of the other packet.

1           30. A method for tracking node address to node  
2   name mappings in a network of nodes of the kind in which  
3   each node has a possibly nonunique node name and a unique  
4   node address within the network and in which node  
5   addresses can be assigned and reassigned to node names  
6   dynamically using a name binding protocol message  
7   incorporated within a packet, said method comprising  
8        monitoring packets transmitted in said network,  
9   and

- 172 -

- 10            updating a table linking node names to node
- 11 addresses based on information contained in said name
- 12 binding protocol messages in said packets.



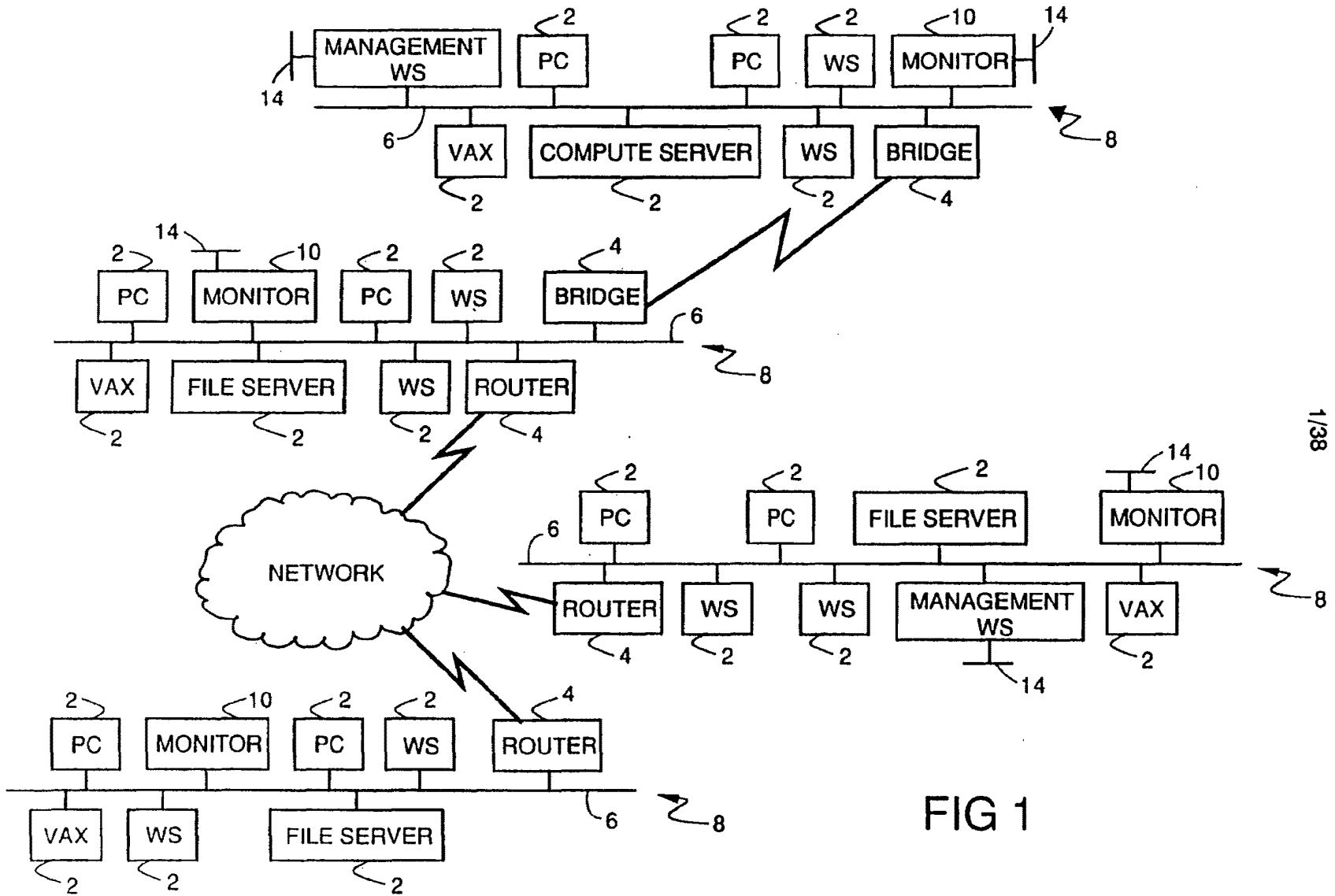


FIG 1

SUBSTITUTE SHEET

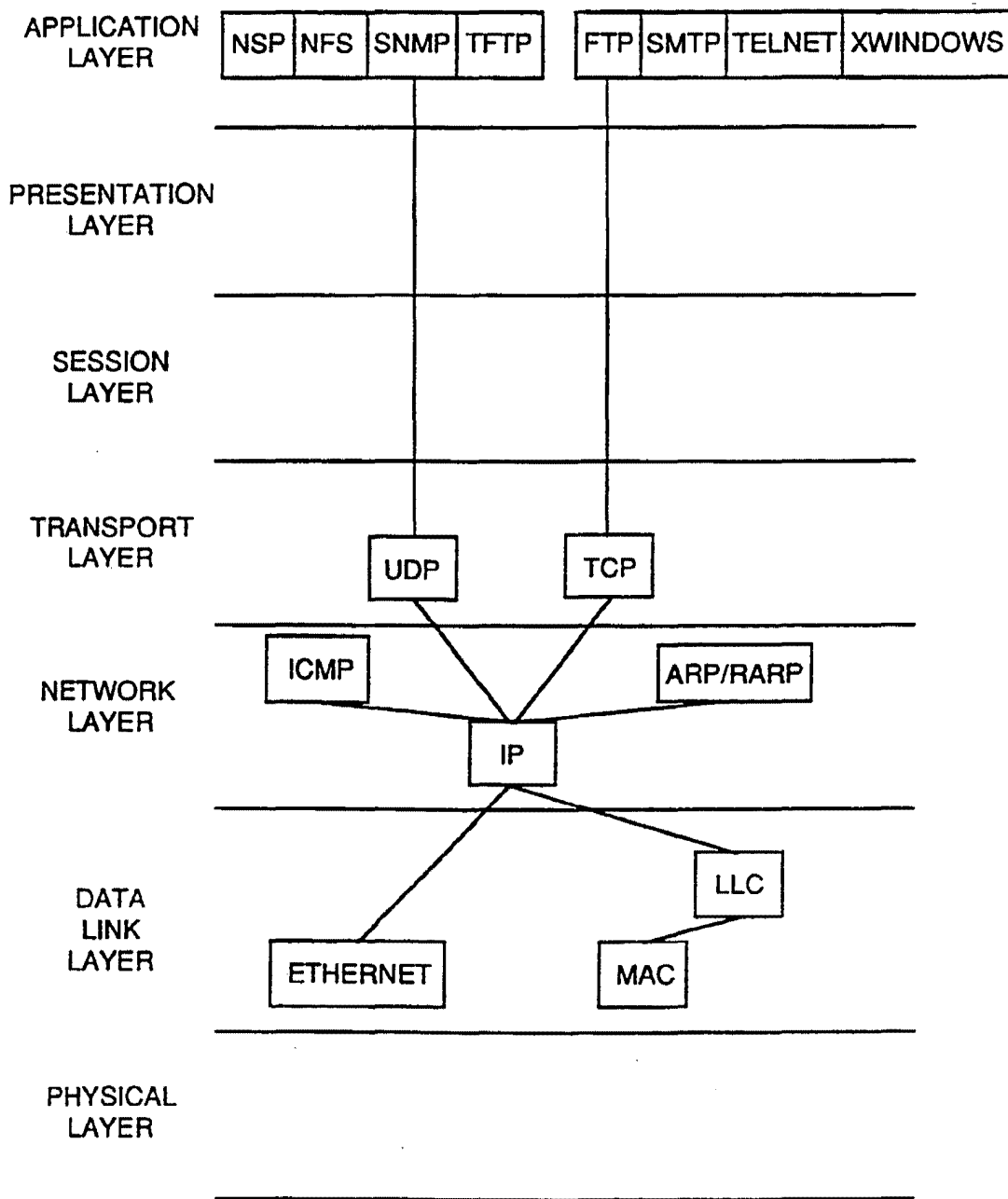


FIG 2

SUBSTITUTE SHEET

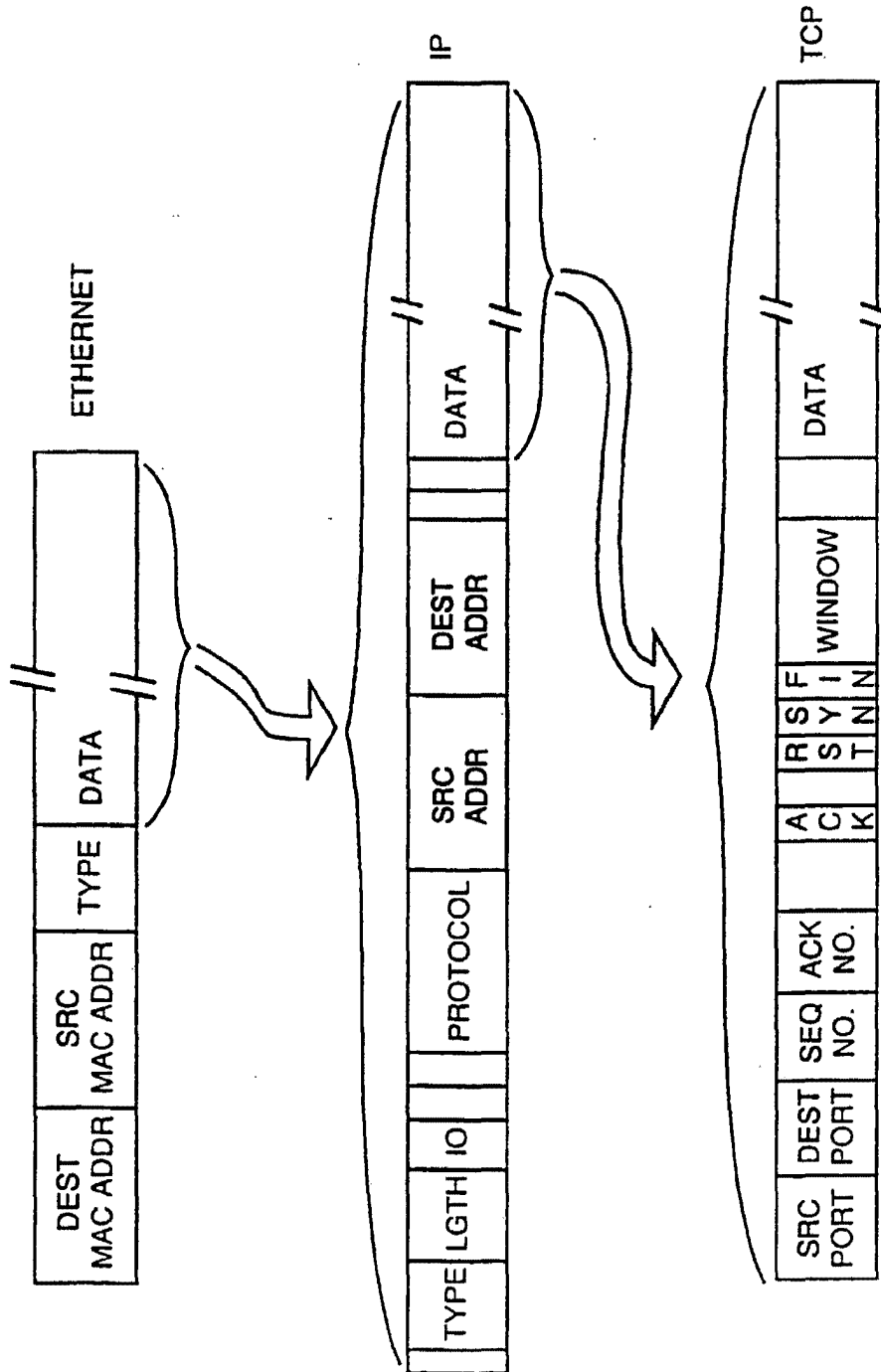


FIG 3

SUBSTITUTE SHEET

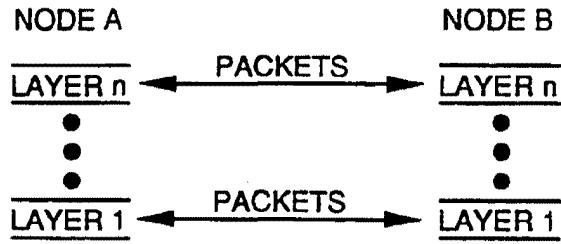


FIG 4

SUBSTITUTE SHEET

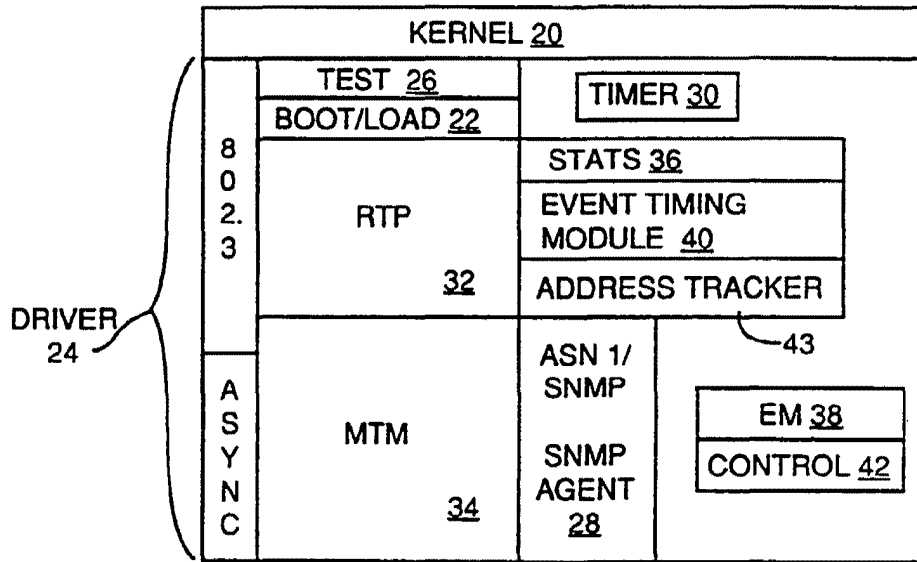


FIG 5

SUBSTITUTE SHEET

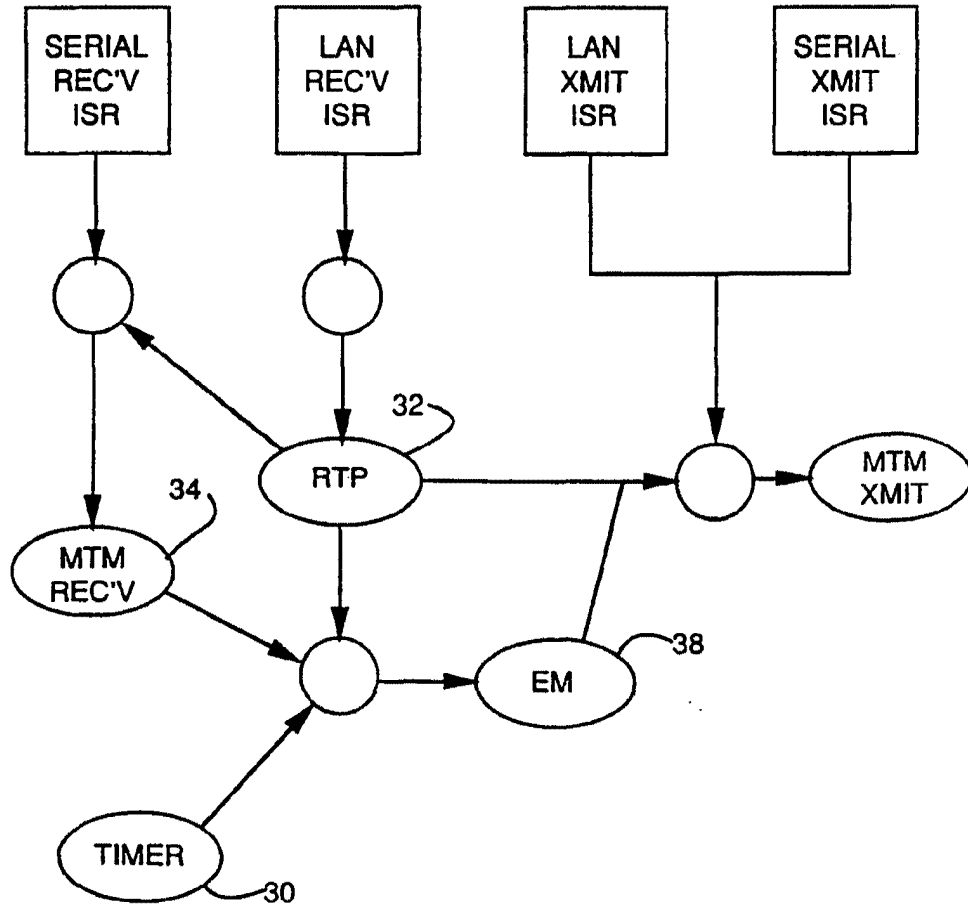


FIG 6

SUBSTITUTE SHEET

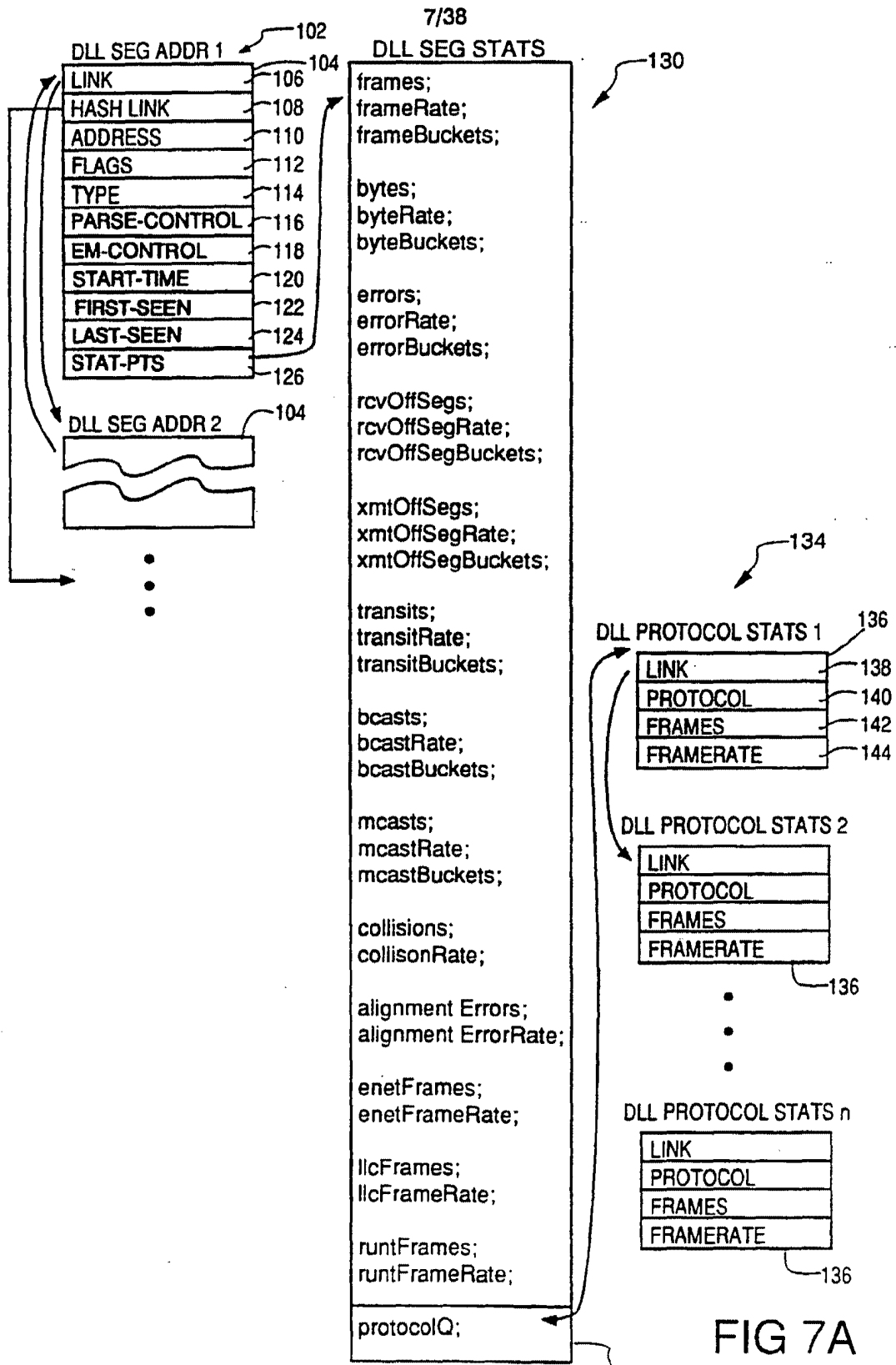


FIG 7A

SUBSTITUTE SHEET

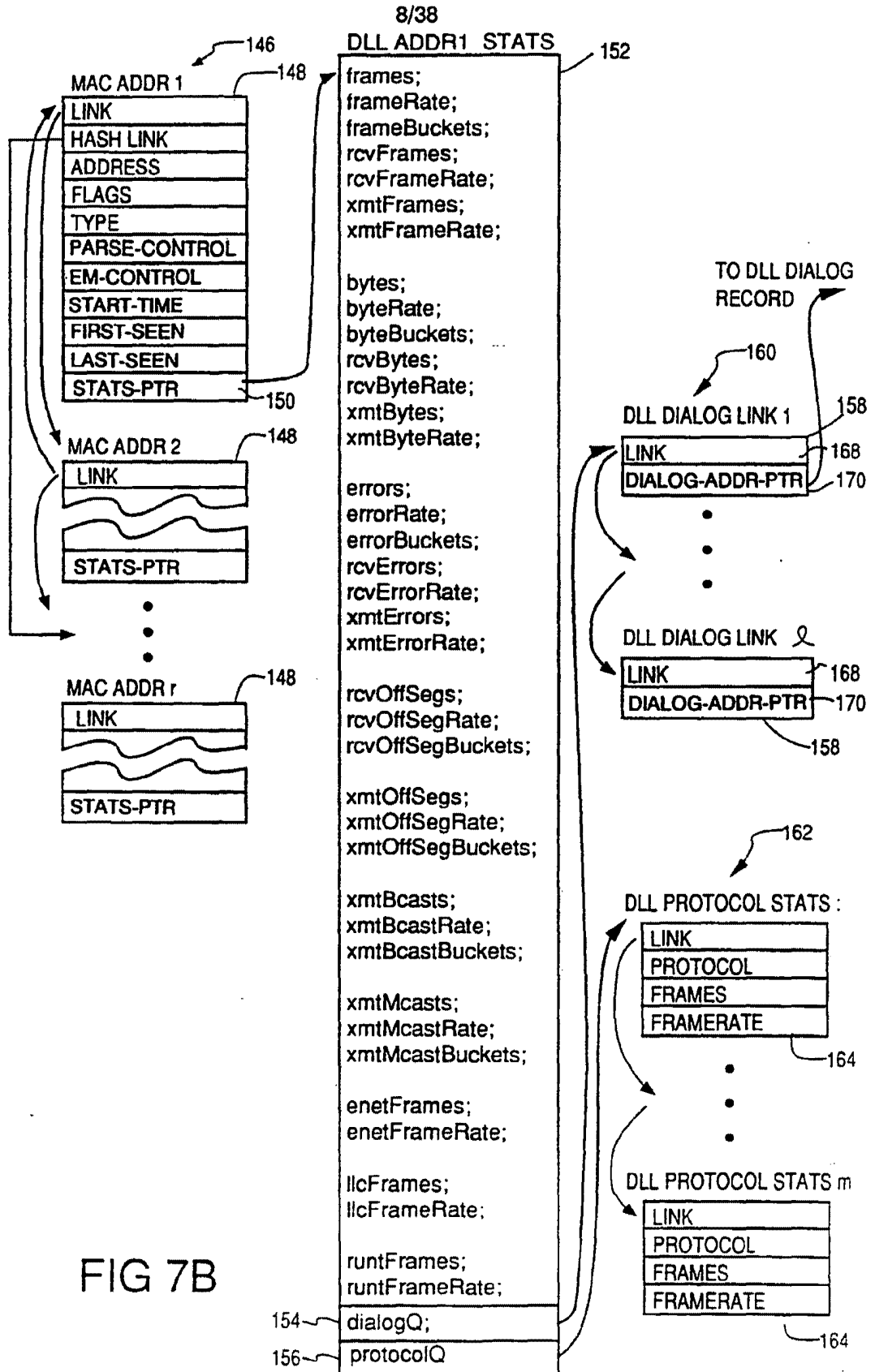


FIG 7B

SUBSTITUTE SHEET



9/38

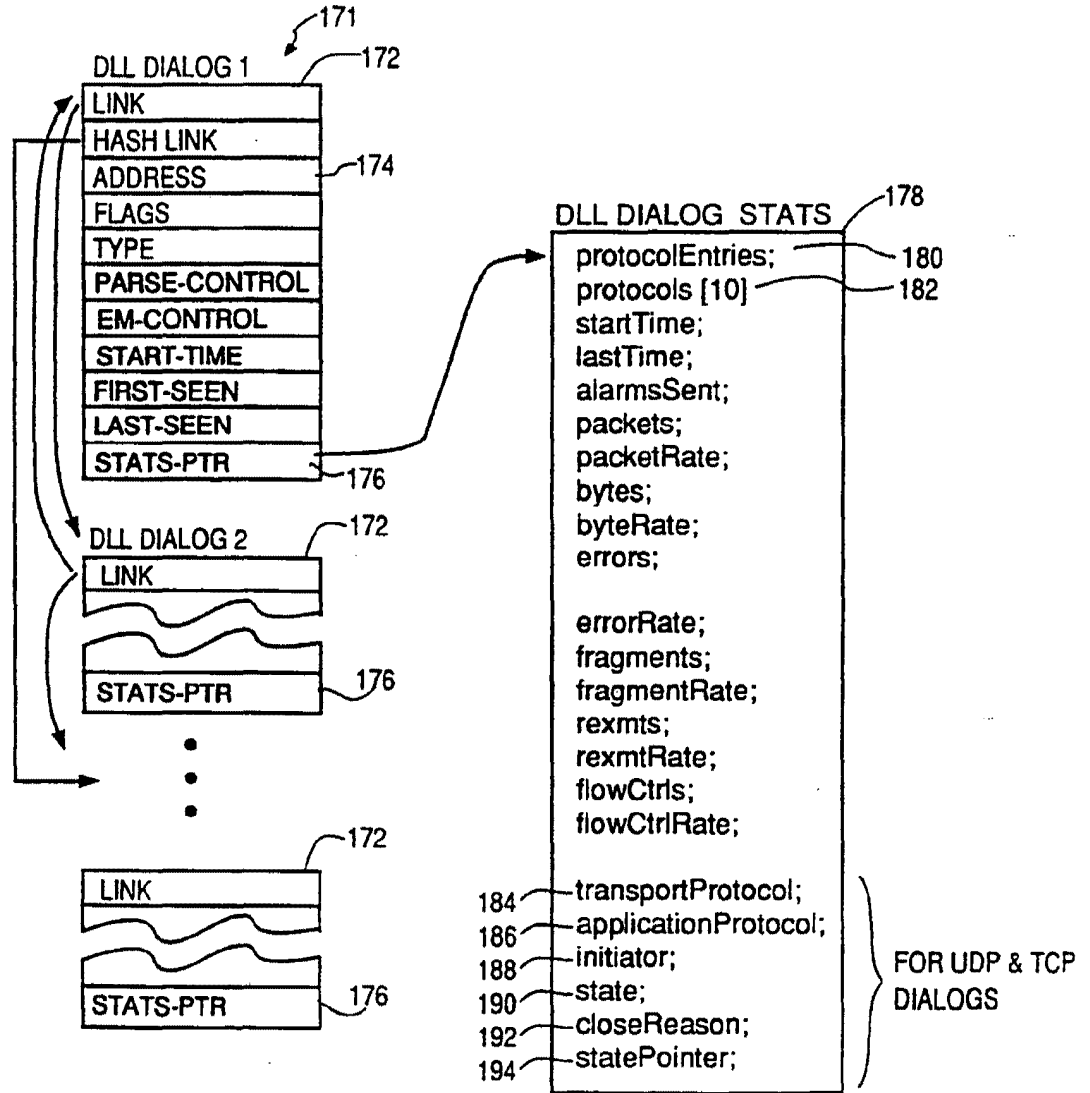


FIG 7C

SUBSTITUTE SHEET

SUBSTITUTE SHEET

STATE EVENT	UNKNOWN	CONNECTING	DATA	CLOSING	CLOSED	INACTIVE
UNKNOWN					S= UNKNOWN AFTER CLOSE ++	S= UNKNOWN
CONNECT REQ OR CONNECT CNF (E.G. TCP SYN)	S= CONNECTING	CONNECTION RETRY ++	S= UNKNOWN OUT OF ORDER++ ACTIVE CONN++	S= UNKNOWN OUT OF ORDER++	S=CONNECTING AFTER CLOSE ++	S=CONNECTING
ABORT (E.G. TCP RST)	S= CLOSED START CLOSE TIMER	S= CLOSED FAILED CONN ++ START CLOSE TIMER	S= CLOSED ACTIVE CONN -- START CLOSE TIMER	S= CLOSED ACTIVE CONN -- START CLOSE TIMER	AFTER CLOSE ++	S= CLOSED START CLOSE TIMER
DATA ACK (E.G. TCP ACK)	LOOK FOR DATA STATE	LOOK FOR DATA STATE	LOOK AT HISTORY	LOOK AT HISTORY	S= UNKNOWN AFTER CLOSE ++	S= UNKNOWN LOOK FOR DATA STATE
RELEASE REQ OR RELEASE CNF (E.G. TCP FIN)	S= CLOSING START CLOSE TIMER	S= CLOSING START CLOSE TIMER	S= CLOSING ACTIVE CONN -- START CLOSE TIMER		S= UNKNOWN AFTER CLOSE ++	S= CLOSING
DATA	LOOK FOR DATA STATE	LOOK FOR DATA STATE	LOOK AT HISTORY	OUT OF ORDER++	S= UNKNOWN AFTER CLOSE ++	S= UNKNOWN LOOK FOR DATA STATE
CLOSE TIMER EXPIRES				S= CLOSED		
INACTIVE TIMER EXPIRES	RECYCLE RESOURCES	RECYCLE RESOURCES FAILED CONN++	RECYCLE RESOURCES ACTIVE CONN --	RECYCLE RESOURCES	RECYCLE RESOURCES	RECYCLE RESOURCES

10/38

FIG 8

11/38

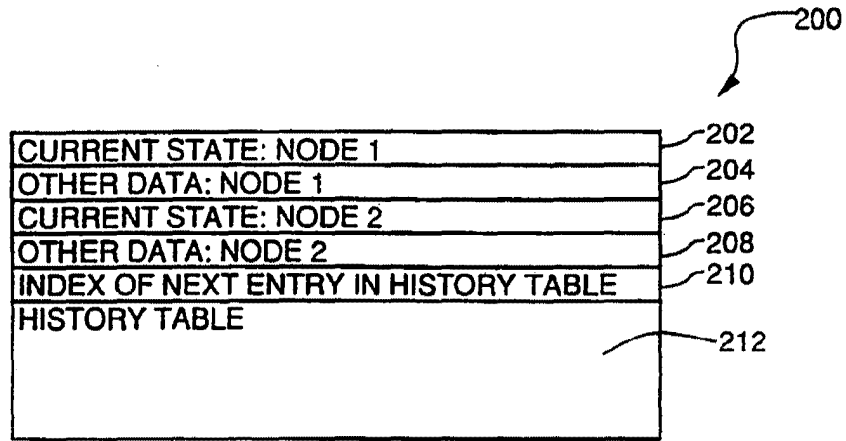


FIG 9A

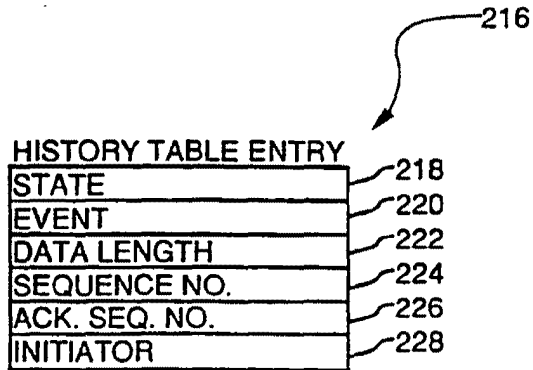
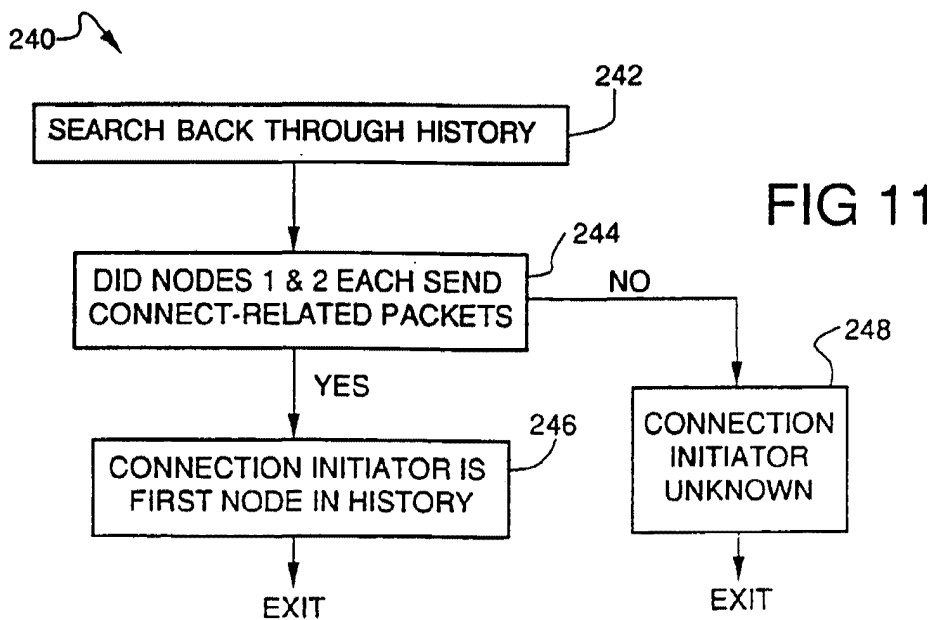
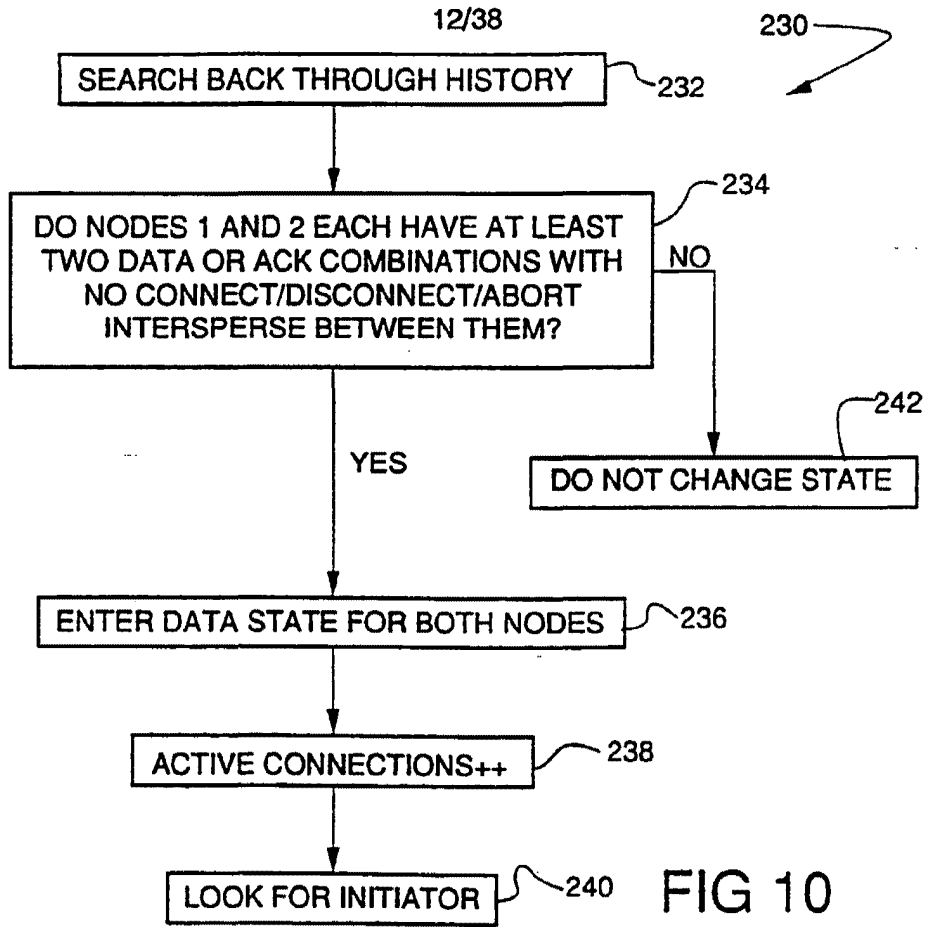


FIG 9B

SUBSTITUTE SHEET



SUBSTITUTE SHEET

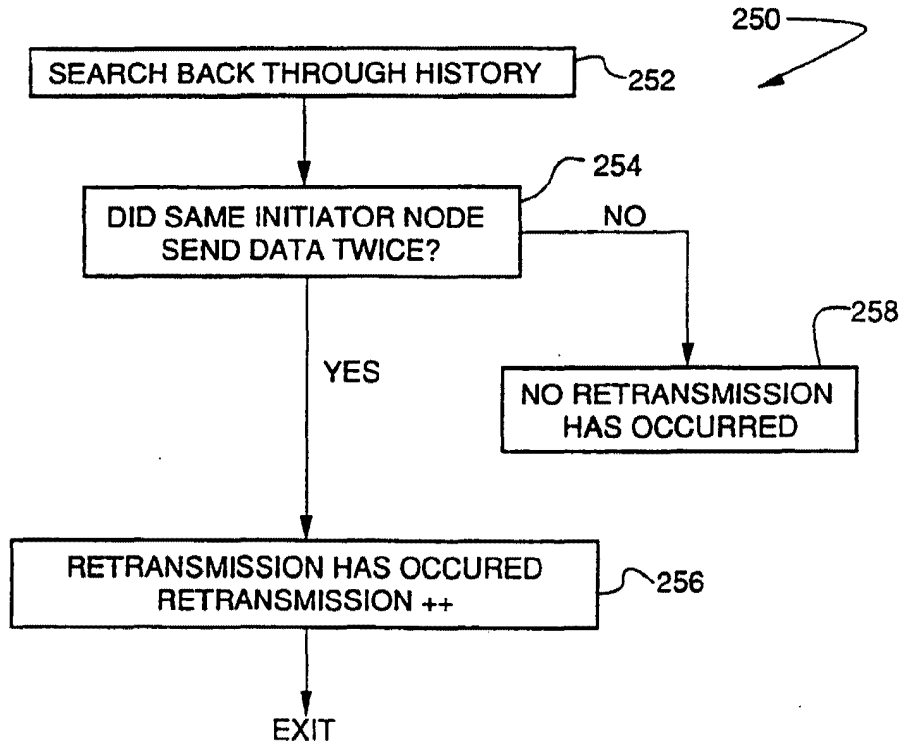


FIG 12

SUBSTITUTE SHEET

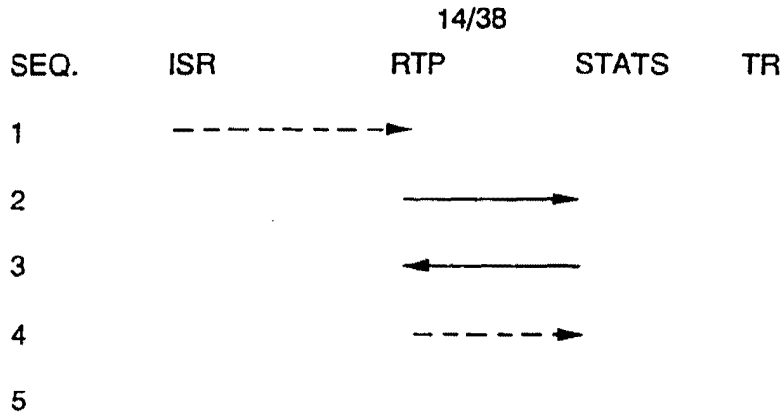


FIG 13

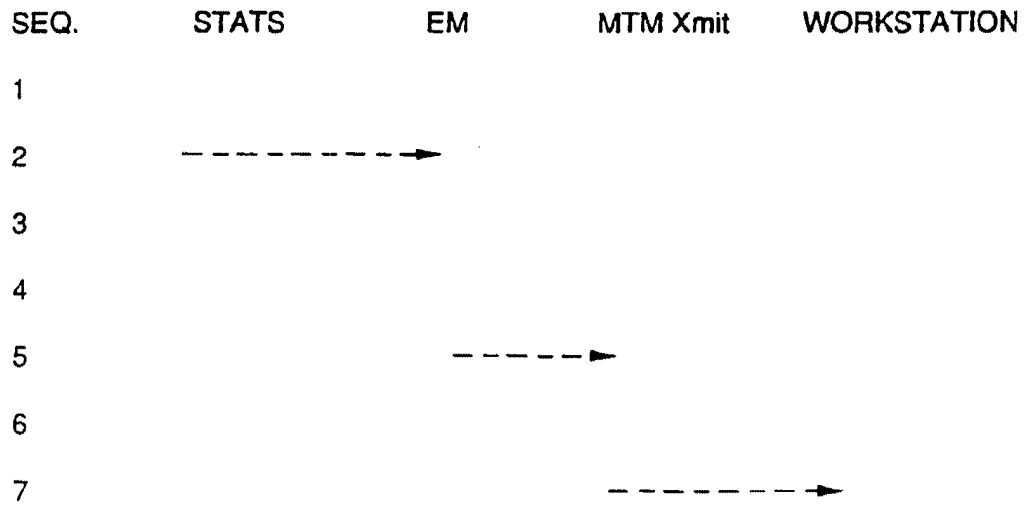


FIG 14

SUBSTITUTE SHEET

15/38

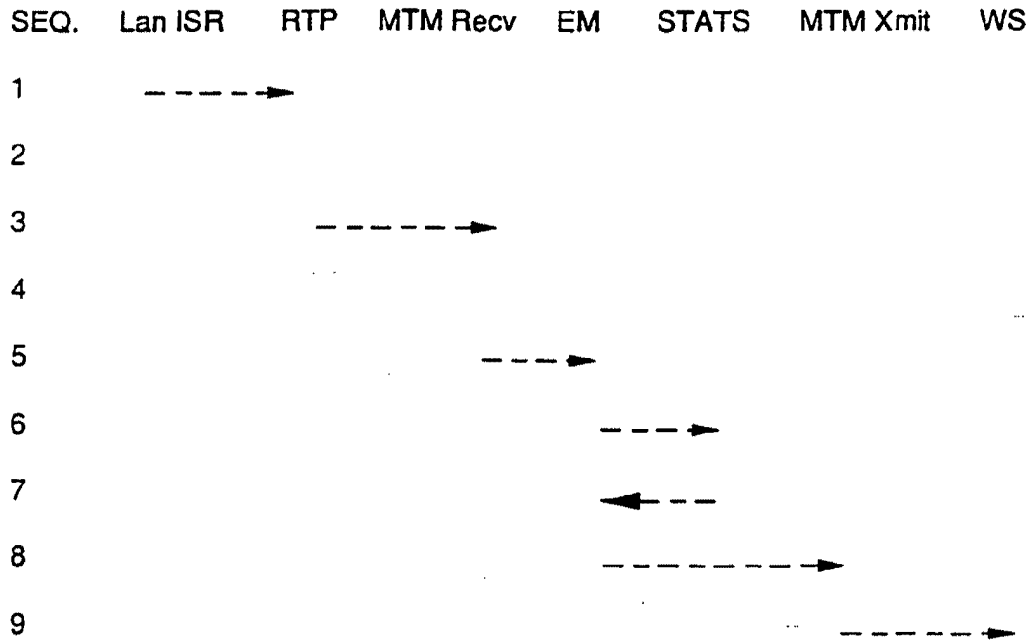


FIG 15

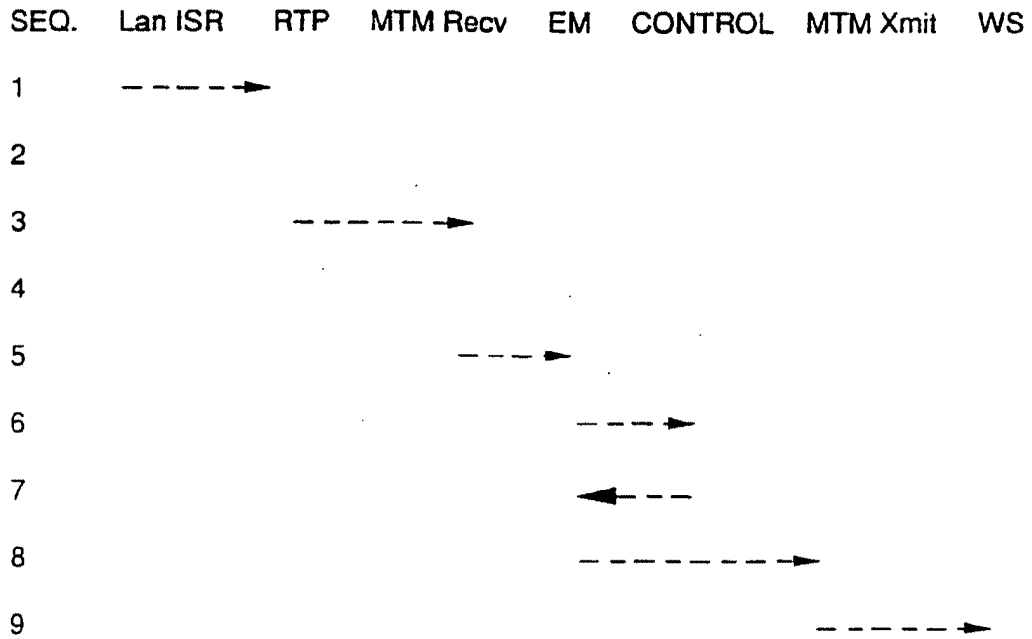


FIG 16

SUBSTITUTE SHEET

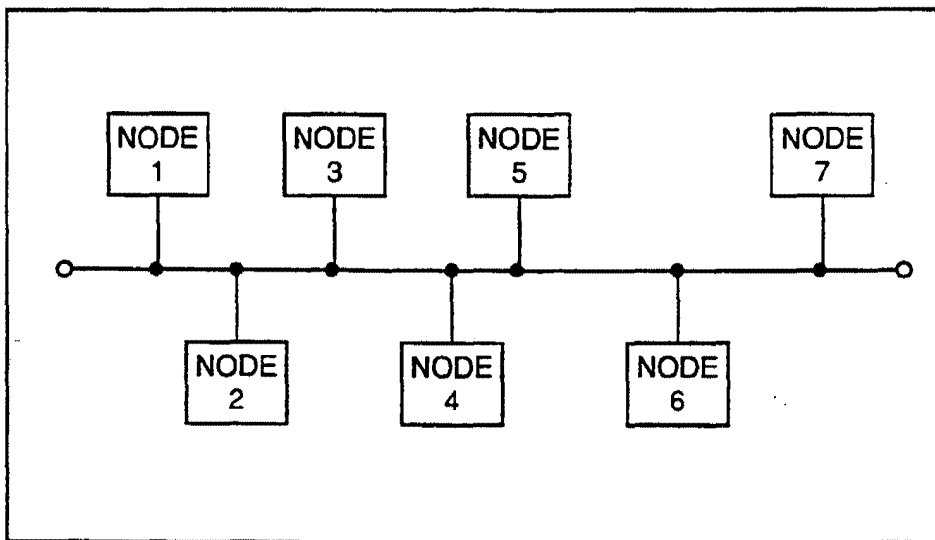


FIG 17

SUBSTITUTE SHEET



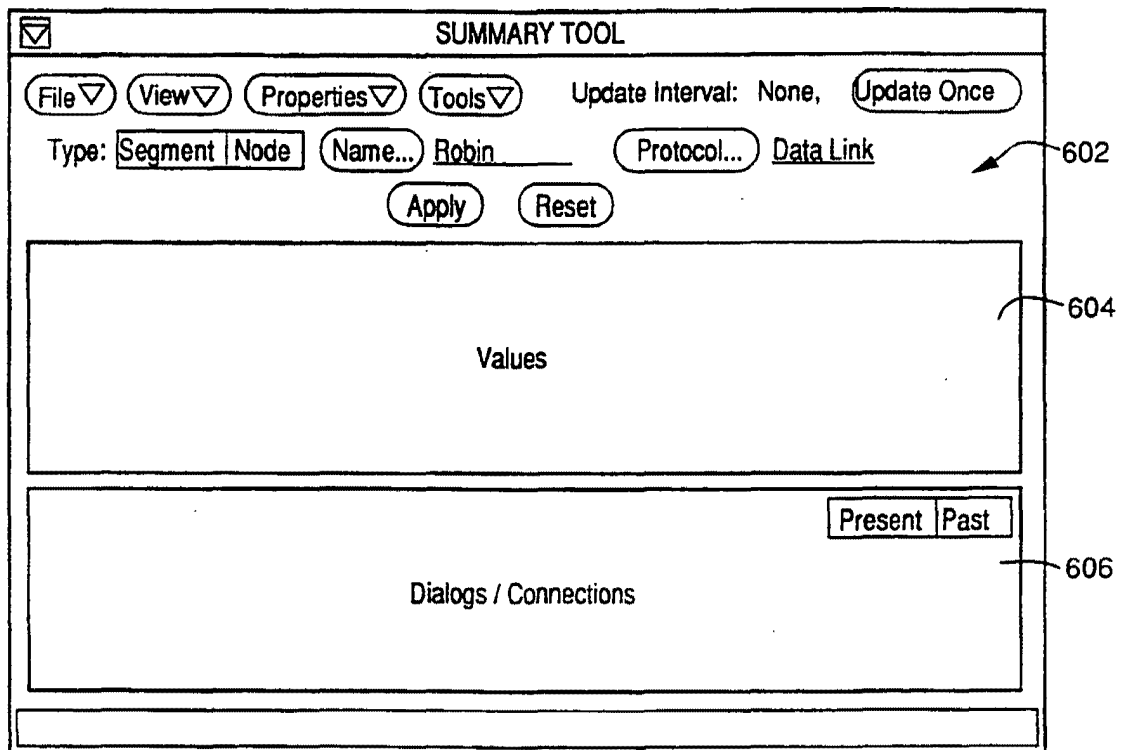


FIG 18

SUBSTITUTE SHEET

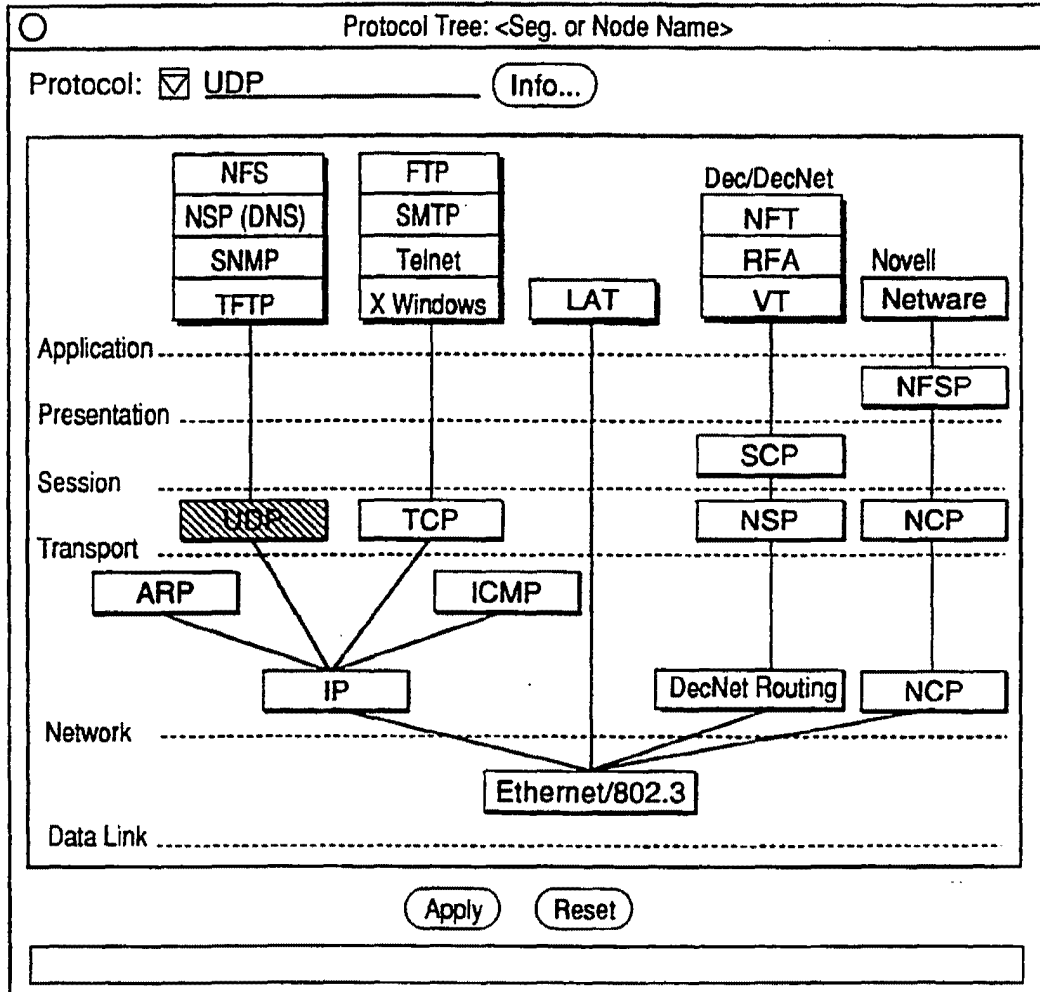


FIG 19

SUBSTITUTE SHEET

Data Link

19/38

	Current	5 Min.	15 Min.	10 Min. Max	60 Min. Max	Accum.Val.
Frame Rate:	nn,nnn /s	nn,nnn /s	nn,nnn /s	nn,nnn /s	nn,nnn /s	n,nnn,nnn
Byte Rate:	nn,nnn /s	nn,nnn /s	nn,nnn /s	nn,nnn /s	nn,nnn /s	n,nnn,nnn
Errors:	nnn,nnn	nnn,nnn	nnn,nnn	-	-	n,nnn,nnn
Broadcast Frm. Rate:	nn,nnn /s	nn,nnn /s	nn,nnn /s	nn,nnn /s	nn,nnn /s	n,nnn,nnn
Multicast Frm. Rate:	nn,nnn /s	nn,nnn /s	nn,nnn /s	nn,nnn /s	nn,nnn /s	n,nnn,nnn

Off Segment Frames						
In:	nnn %	nnn %	nnn %	nnn %	nnn %	n,nnn,nnn
Out:	nnn %	nnn %	nnn %	nnn %	nnn %	n,nnn,nnn
**Transit:	nnn %	nnn %	nnn %	nnn %	nnn %	n,nnn,nnn

Most Active Protocols (Frm. Rate)

Most Active Nodes (Frm. Rate)

1234567890123456	nnn %
<protocol>	nnn %
<protocol>	nnn %
<protocol>	nnn %
<protocol>	nnn %

1234567890123456	nnn %
<node>	nnn %
<node>	nnn %
<node>	nnn %
<node>	nnn %

Total Segment Bandwidth: nnn %

Total Active Dialogs: nn, nn

FIG 20A

IP

	Current	5 Min.	15 Min.	10 Min. Max	60 Min. Max	Accum.Val.
Packet Rate:	nn,nnn /s	nn,nnn /s	nn,nnn /s	nn,nnn /s	nn,nnn /s	n,nnn,nnn
Byte Rate:	nn,nnn /s	nn,nnn /s	nn,nnn /s	nn,nnn /s	nn,nnn /s	n,nnn,nnn
Errors:	nnn,nnn	nnn,nnn	nnn,nnn	-	-	n,nnn,nnn
Broadcast Pkt. Rate:	nn,nnn /s	nn,nnn /s	nn,nnn /s	nn,nnn /s	nn,nnn /s	n,nnn,nnn
Multicast Pkt. Rate:	nn,nnn /s	nn,nnn /s	nn,nnn /s	nn,nnn /s	nn,nnn /s	n,nnn,nnn
Flow Controls:	nnn,nnn	nnn,nnn	nnn,nnn	-	-	n,nnn,nnn
Fragments:	nnn,nnn	nnn,nnn	nnn,nnn	-	-	n,nnn,nnn

Off Segment Packets						
In:	nnn %	nnn %	nnn %	nnn %	nnn %	n,nnn,nnn
Out:	nnn %	nnn %	nnn %	nnn %	nnn %	n,nnn,nnn
**Transit:	nnn %	nnn %	nnn %	nnn %	nnn %	n,nnn,nnn

Most Active Protocols (Pkt. Rate)

Most Active Nodes (Pkt. Rate)

1234567890123456	nnn %
<protocol>	nnn %
<protocol>	nnn %
<protocol>	nnn %
<protocol>	nnn %

1234567890123456	nnn %
<node>	nnn %
<node>	nnn %
<node>	nnn %
<node>	nnn %

Total Segment Bandwidth: nnn %

Total Active Dialogs: nn, nn

FIG 20B

SUBSTITUTE SHEET

UDP		20/38					
	Current	5 Min.	15 Min.	10 Min. Max	60 Min. Max	Accum.Val.	
Packet Rate:	nn,nnn /s	nn,nnn /s	nn,nnn /s	nn,nnn /s	nn,nnn /s	n,nnn,nnn	
Byte Rate:	nn,nnn /s	nn,nnn /s	nn,nnn /s	nn,nnn /s	nn,nnn /s	n,nnn,nnn	
Errors:	nnn,nnn	nnn,nnn	nnn,nnn	-	-	n,nnn,nnn	
Flow Controls:	nnn,nnn	nnn,nnn	nnn,nnn	-	-	n,nnn,nnn	
Off Segment Packets							
In:	nnn %	nnn %	nnn %	nnn %	nnn %	n,nnn,nnn	
Out:	nnn %	nnn %	nnn %	nnn %	nnn %	n,nnn,nnn	
**Transit:	nnn %	nnn %	nnn %	nnn %	nnn %	n,nnn,nnn	
Most Active Protocols (Pkt. Rate)			Most Active Nodes (Pkt. Rate)				
1234567890123456	nnn %		1234567890123456	nnn %			
<protocol>	nnn %		<node>	nnn %			
<protocol>	nnn %		<node>	nnn %			
<protocol>	nnn %		<node>	nnn %			
<protocol>	nnn %		<node>	nnn %			
Total Segment Bandwidth:	nnn %		Total Active Dialogs:	nn, nnn			

FIG 20C

TCP							
	Current	5 Min.	15 Min.	10 Min. Max	60 Min. Max	Accum.Val.	
Packet Rate:	nn,nnn /s	nn,nnn /s	nn,nnn /s	nn,nnn /s	nn,nnn /s	n,nnn,nnn	
Byte Rate:	nn,nnn /s	nn,nnn /s	nn,nnn /s	nn,nnn /s	nn,nnn /s	n,nnn,nnn	
Errors:	nnn,nnn	nnn,nnn	nnn,nnn	-	-	n,nnn,nnn	
Flow Controls:	nnn,nnn	nnn,nnn	nnn,nnn	-	-	n,nnn,nnn	
Retransmissions:	nnn,nnn	nnn,nnn	nnn,nnn	-	-	n,nnn,nnn	
Off Segment Packets							
In:	nnn %	nnn %	nnn %	nnn %	nnn %	n,nnn,nnn	
Out:	nnn %	nnn %	nnn %	nnn %	nnn %	n,nnn,nnn	
**Transit:	nnn %	nnn %	nnn %	nnn %	nnn %	n,nnn,nnn	
Most Active Protocols (Pkt. Rate)			Most Active Nodes (Pkt. Rate)				
1234567890123456	nnn %		1234567890123456	nnn %			
<protocol>	nnn %		<node>	nnn %			
<protocol>	nnn %		<node>	nnn %			
<protocol>	nnn %		<node>	nnn %			
<protocol>	nnn %		<node>	nnn %			
Total Segment Bandwidth:	nnn %		Total Active Connections:	nn, nnn			

FIG 20D

SUBSTITUTE SHEET

21/38

ICMP

	Current	5 Min.	15 Min.	10 Min. Max	60 Min. Max	Accum.Val.
Packet Rate:	nn,nnn /s	nn,nnn /s	nn,nnn /s	nn,nnn /s	nn,nnn /s	n,nnn,nnn
Byte Rate:	nn,nnn /s	nn,nnn /s	nn,nnn /s	nn,nnn /s	nn,nnn /s	n,nnn,nnn
Errors:	nnn,nnn	nnn,nnn	nnn,nnn	-	-	n,nnn,nnn

Off Segment Packets

In:	nnn %	nnn %	nnn %	nnn %	nnn %	n,nnn,nnn
Out:	nnn %	nnn %	nnn %	nnn %	nnn %	n,nnn,nnn
**Transit:	nnn %	nnn %	nnn %	nnn %	nnn %	n,nnn,nnn

ICMP Types Seen (Count)

Address Mask:	nnn,nnn	Redirect:	nnn,nnn
Dst. Unreachable:	nnn,nnn	Source Quench:	nnn,nnn
Echo:	nnn,nnn	Time Exceeded:	nnn,nnn
Param. Problem:	nnn,nnn	Time Stamp:	nnn,nnn

Most Active Nodes (Pkt. Rate)

1234567890123456	nnn %
<node>	nnn %
<node>	nnn %
<node>	nnn %
<node>	nnn %

Total Segment Bandwidth: nnn %

FIG 20E

NFS

	Current	5 Min.	15 Min.	10 Min. Max	60 Min. Max	Accum.Val.
Packet Rate:	nn,nnn /s	nn,nnn /s	nn,nnn /s	nn,nnn /s	nn,nnn /s	n,nnn,nnn
Byte Rate:	nn,nnn /s	nn,nnn /s	nn,nnn /s	nn,nnn /s	nn,nnn /s	n,nnn,nnn
Errors:	nnn,nnn	nnn,nnn	nnn,nnn	-	-	n,nnn,nnn
Flow Controls:	nnn,nnn	nnn,nnn	nnn,nnn	-	-	n,nnn,nnn

Off Segment Packets

In:	nnn %	nnn %	nnn %	nnn %	nnn %	n,nnn,nnn
Out:	nnn %	nnn %	nnn %	nnn %	nnn %	n,nnn,nnn
**Transit:	nnn %	nnn %	nnn %	nnn %	nnn %	n,nnn,nnn

Most Active Nodes (Pkt. Rate)

1234567890123456	nnn %
<node>	nnn %
<node>	nnn %
<node>	nnn %
<node>	nnn %

Total Segment Bandwidth: nnn %

Total Active Dialogs: nn, nnn

FIG 20F

SUBSTITUTE SHEET

22/38

Arp/Rarp

	Current	5 Min.	15 Min.	10 Min. Max	60 Min. Max	Accum. Val.
Packet Rate:	nn,nnn /s	nn,nnn /s	nn,nnn /s	nn,nnn /s	nn,nnn /s	n,nnn,nnn
Byte Rate:	nn,nnn /s	nn,nnn /s	nn,nnn /s	nn,nnn /s	nn,nnn /s	n,nnn,nnn
Errors:	nnn,nnn	nnn,nnn	nnn,nnn	-	-	n,nnn,nnn

Off Segment Packets

	nnn %	nnn %	nnn %	nnn %	nnn %	n,nnn,nnn
In:	nnn %	nnn %	nnn %	nnn %	nnn %	n,nnn,nnn
Out:	nnn %	nnn %	nnn %	nnn %	nnn %	n,nnn,nnn
**Transit:	nnn %	nnn %	nnn %	nnn %	nnn %	n,nnn,nnn

Most Active Nodes (Pkt. Rate)

1234567890123456	nnn %
<node>	nnn %
<node>	nnn %
<node>	nnn %
<node>	nnn %

Total Segment Bandwidth: nnn %

FIG 20G

Start Time	Last Seen	Dir.	Partner	Node	Protocols	Packets Summary			Errors
						Rate	%	Count	
hh:mm:ss	hh:mm:ss	1234	1234567890123456	1234567890123456	nn,nnn /s	nnn %	n,nnn,nnn	nnn,nnn	
10:23:04	15:31:47	To	robin		XNS,XEROX-PUP	325 /s	6%	2,641	0
07:21:38	13:25:51	From	hawk		DOD-IP, X25	87 /s	3%	127	1
					BBN-SIMNET				
10/31/90	08:22:30	?	hawk		APPLETALK	13 /s	1%	24,192	4

FIG 21

SUBSTITUTE SHEET

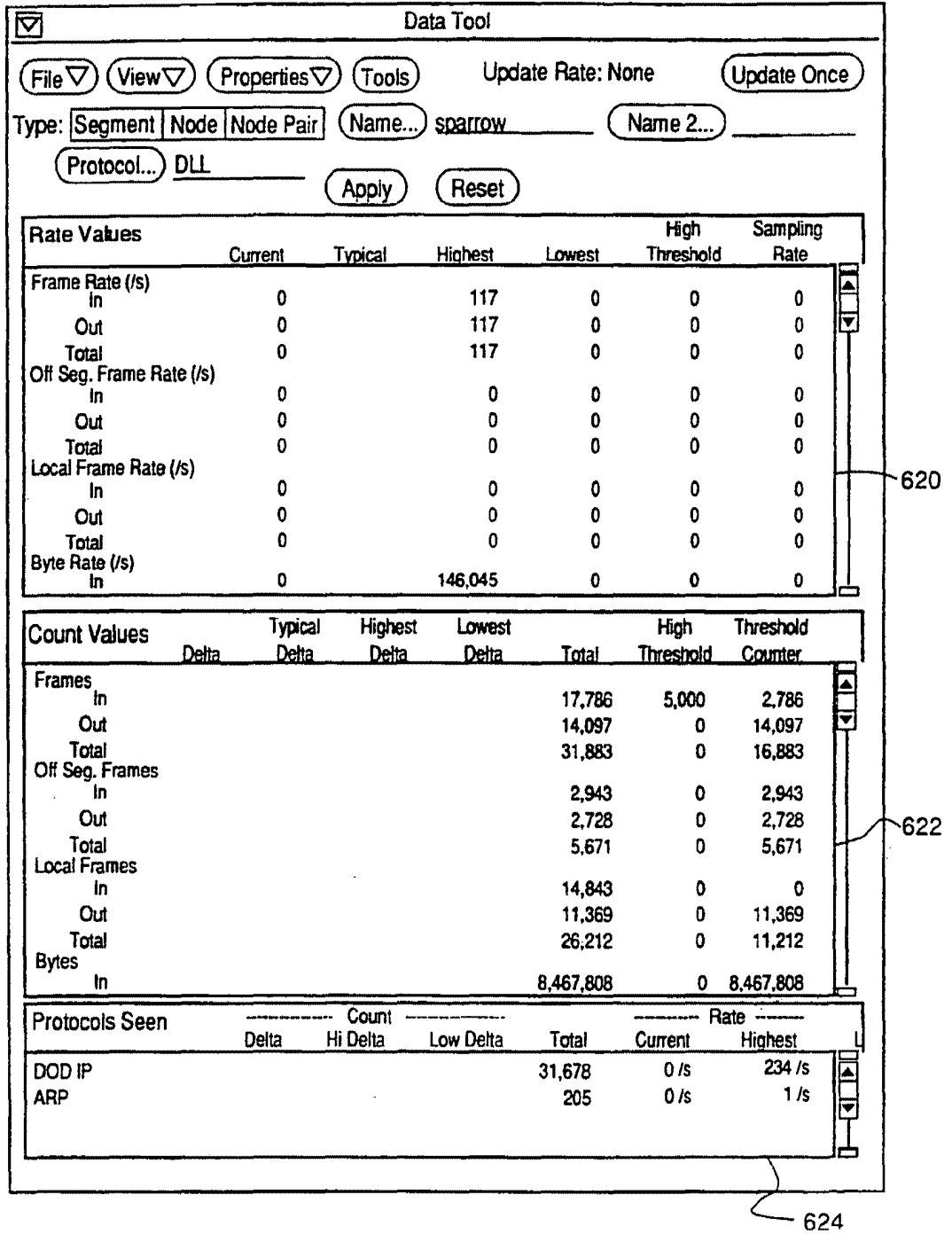


FIG 22

SUBSTITUTE SHEET

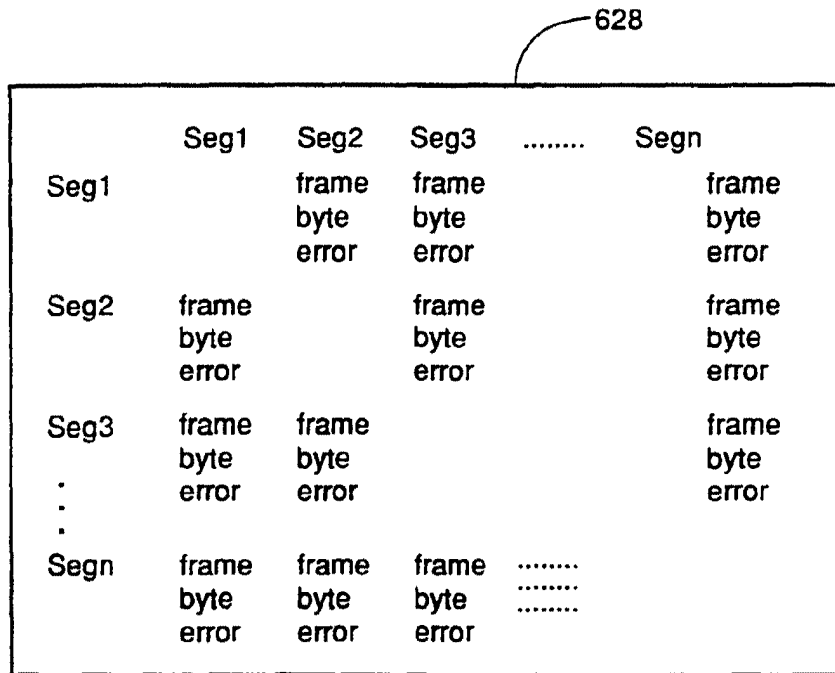


FIG 23

SUBSTITUTE SHEET



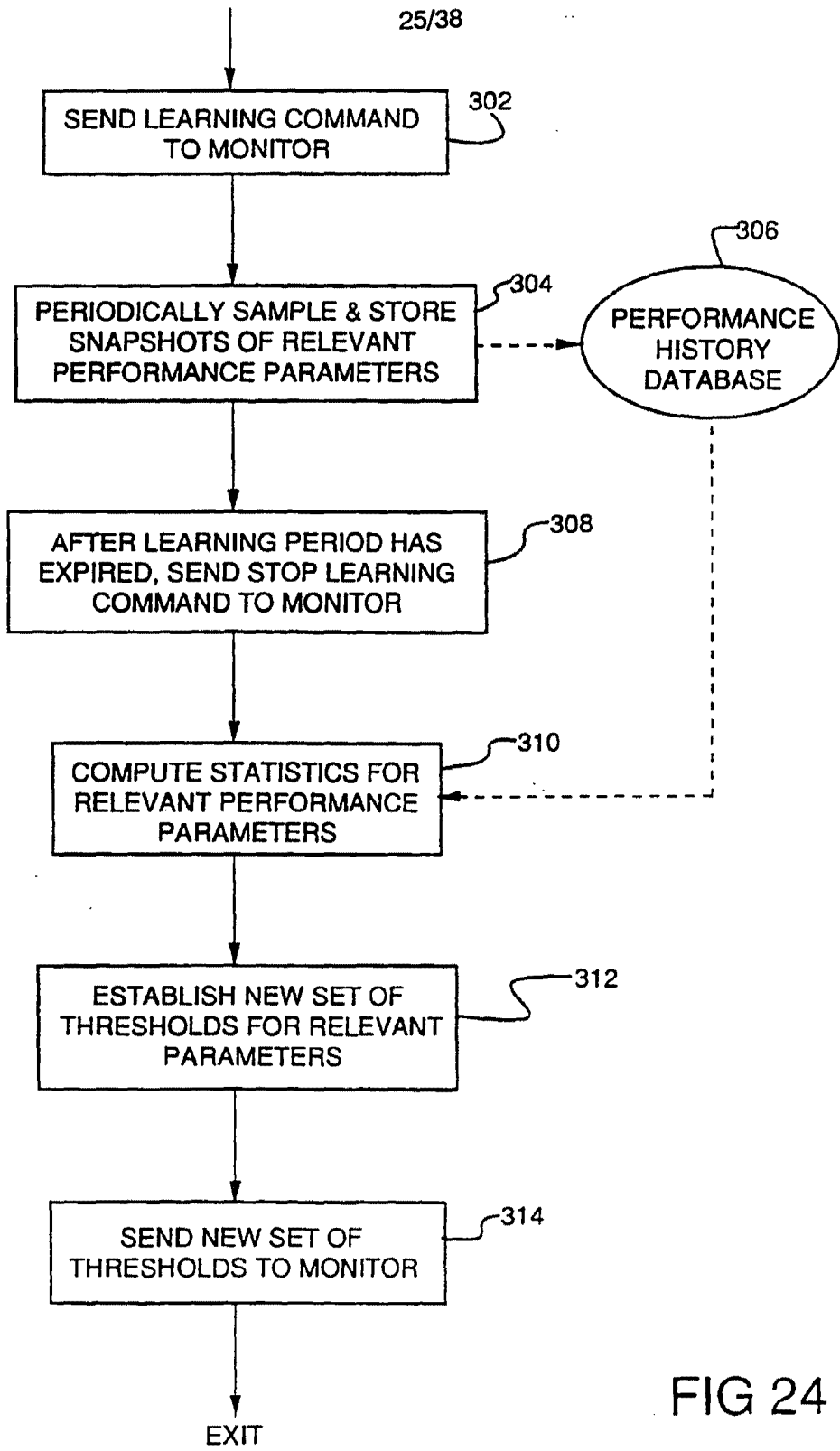


FIG 24

SUBSTITUTE SHEET

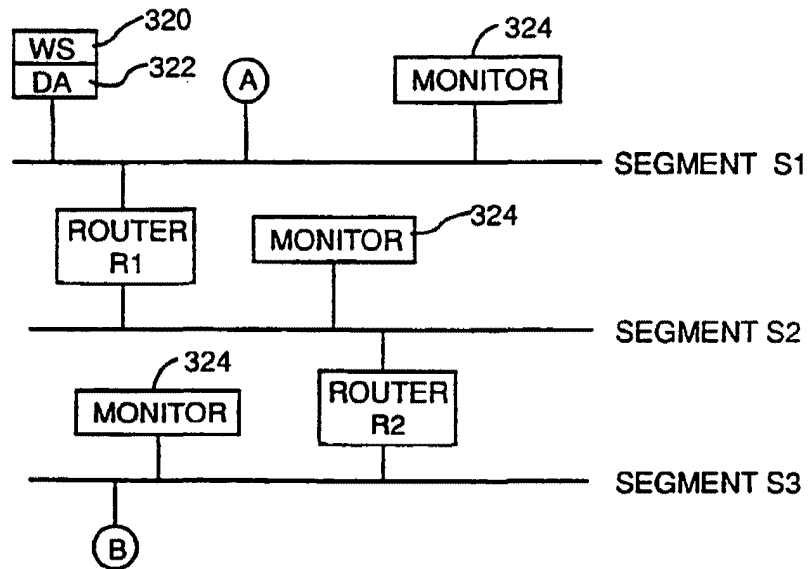


FIG 25

SUBSTITUTE SHEET

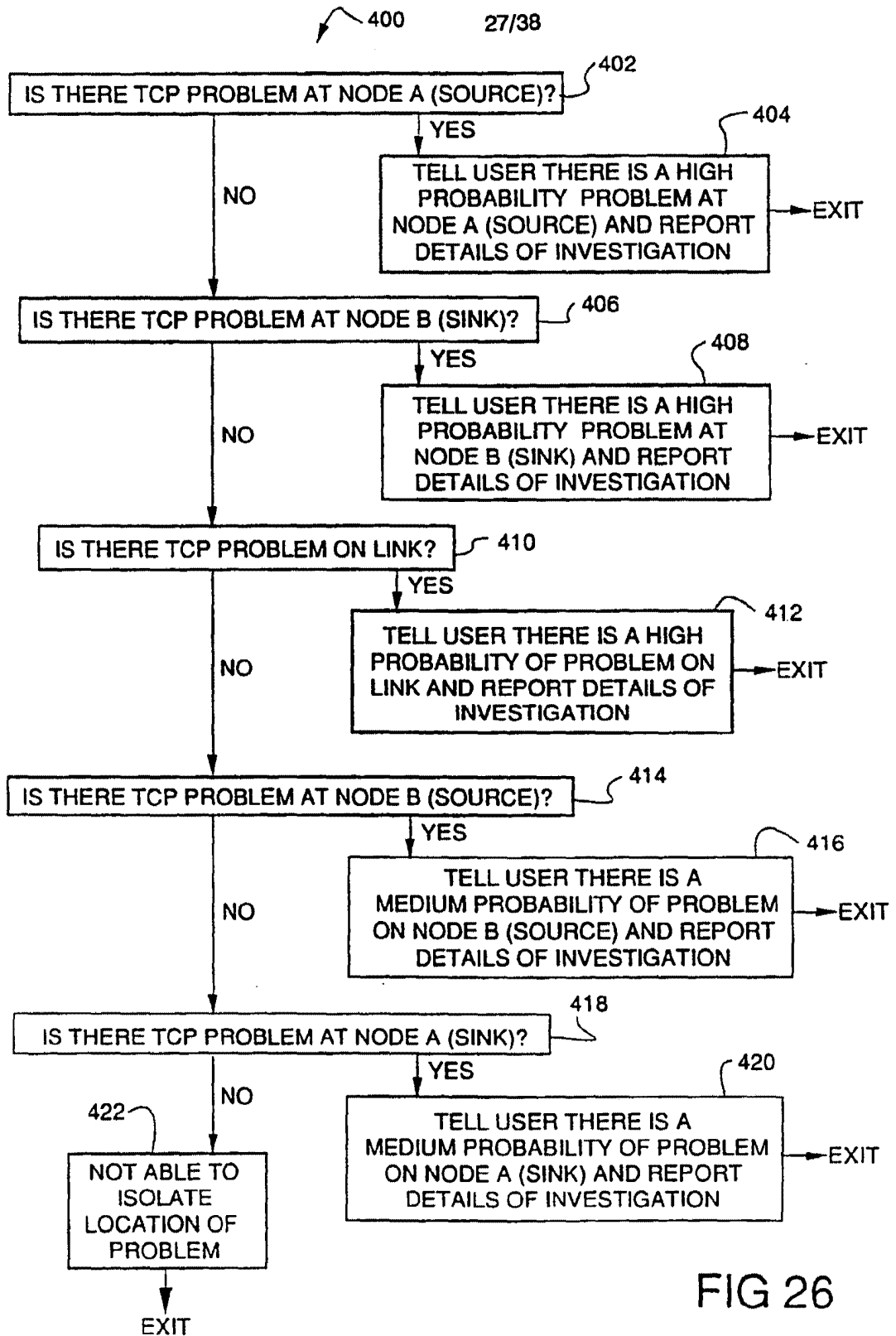


FIG 26

SUBSTITUTE SHEET

28/38

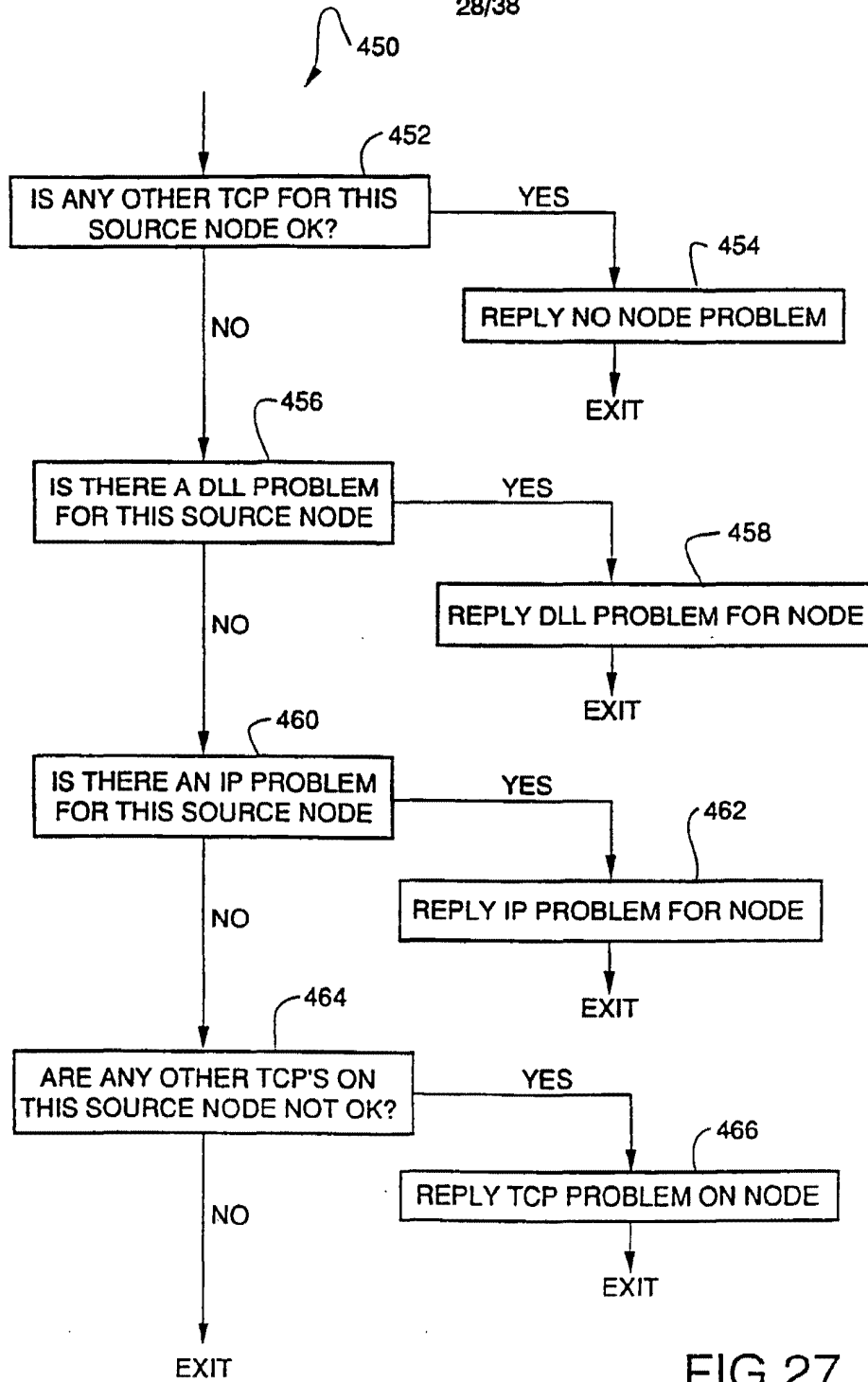


FIG 27

SUBSTITUTE SHEET

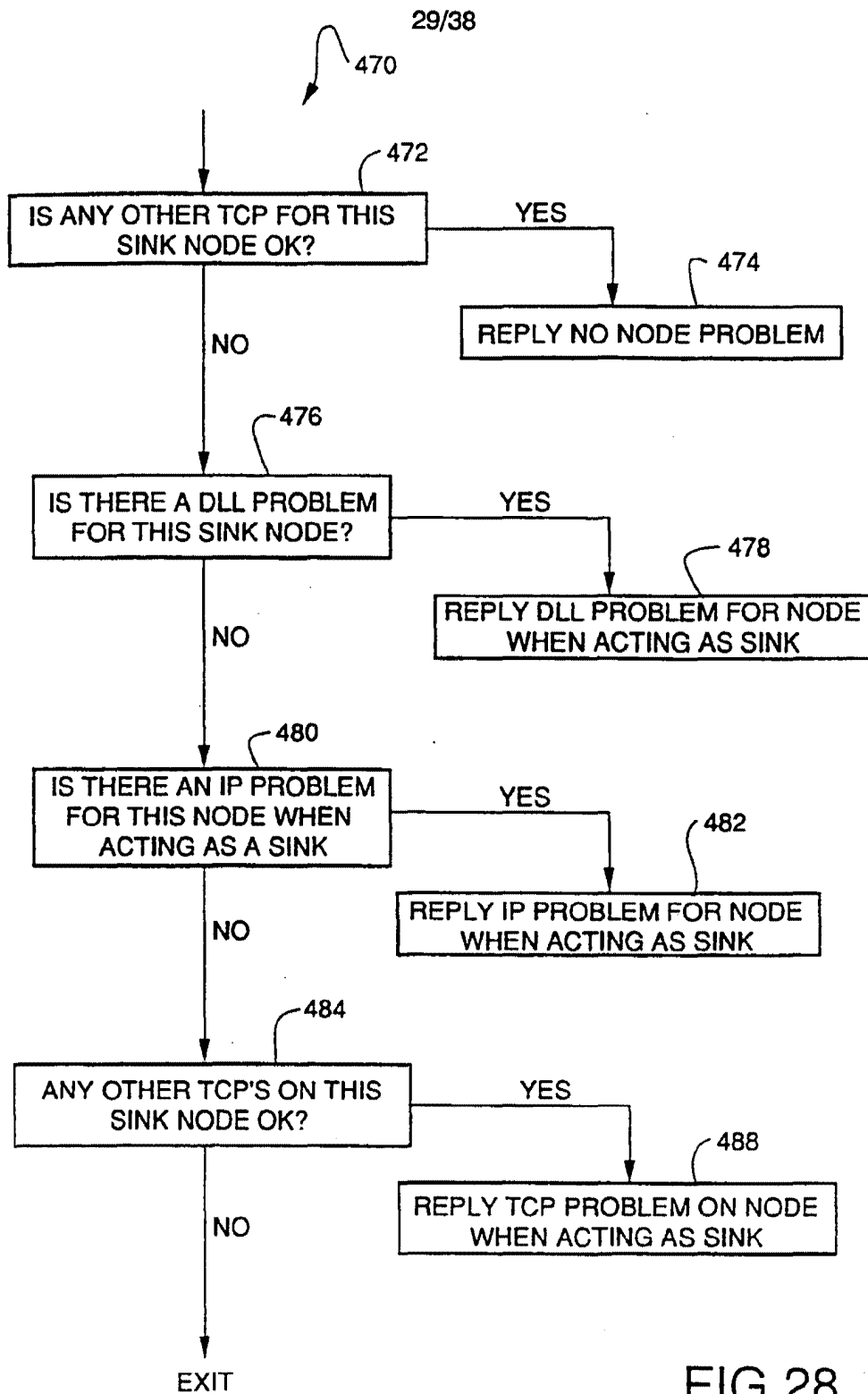


FIG 28

SUBSTITUTE SHEET

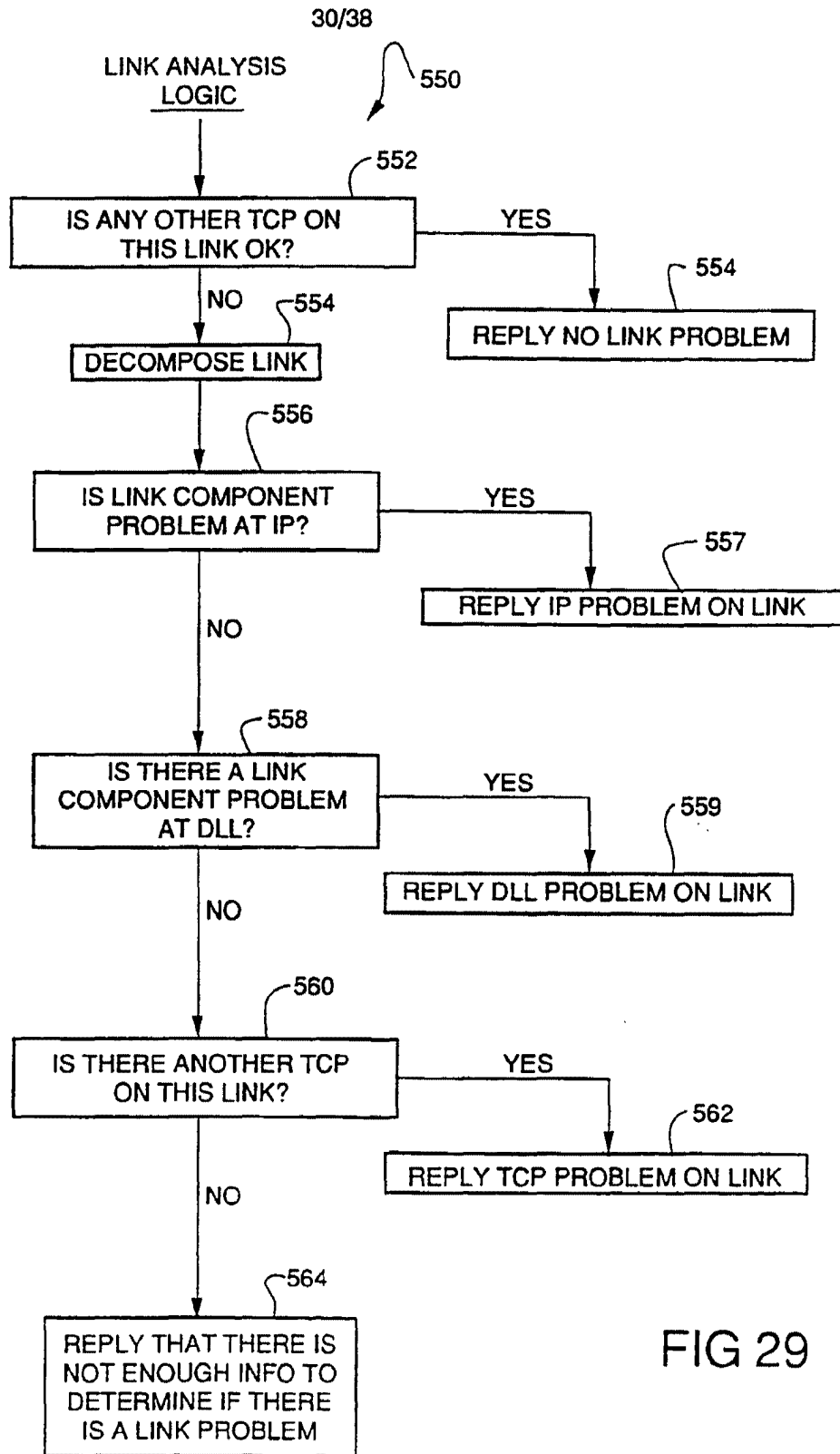


FIG 29

SUBSTITUTE SHEET

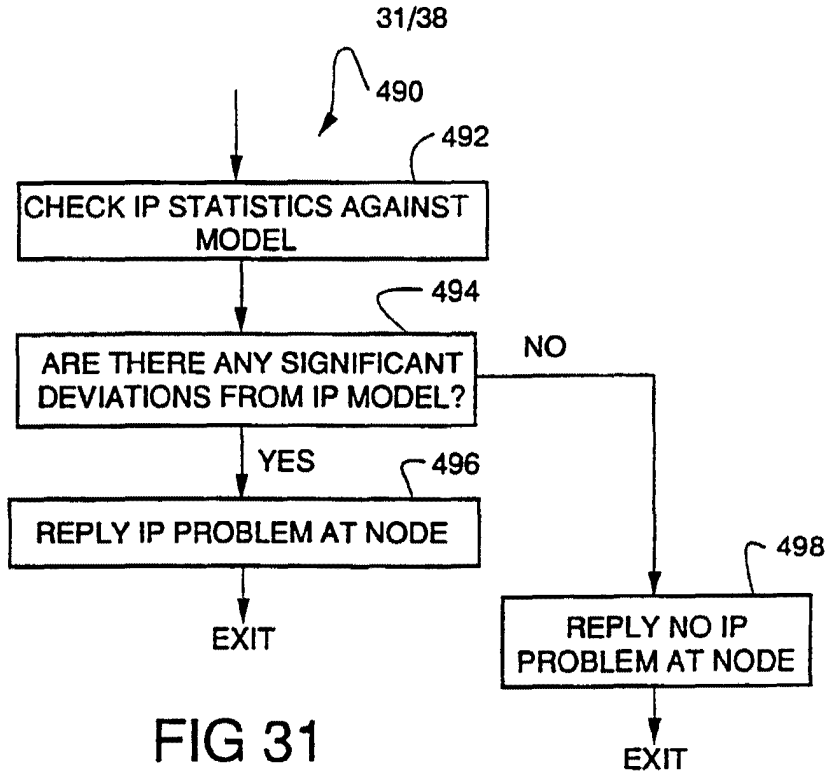


FIG 31

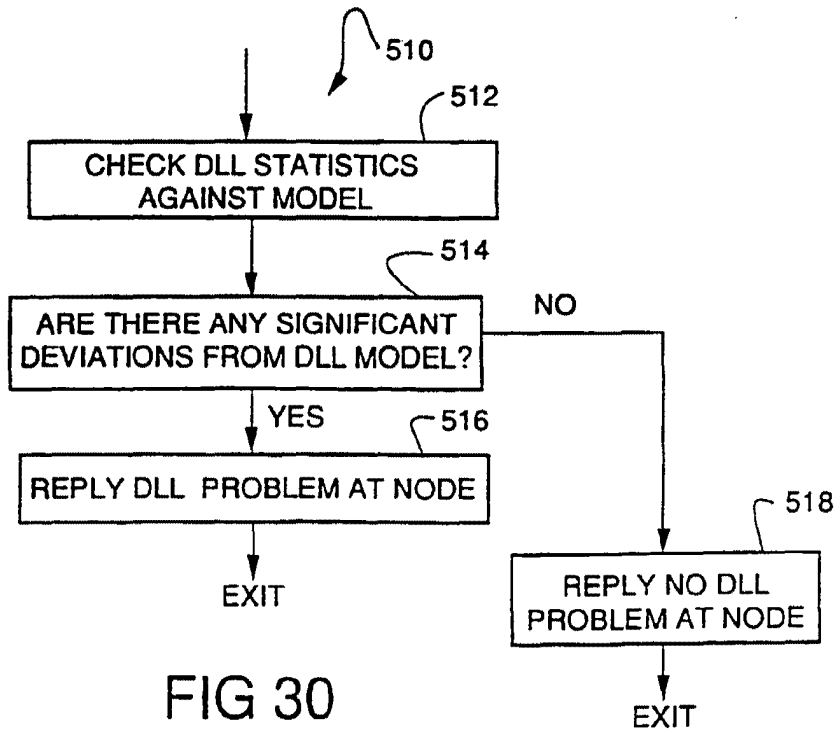


FIG 30

SUBSTITUTE SHEET

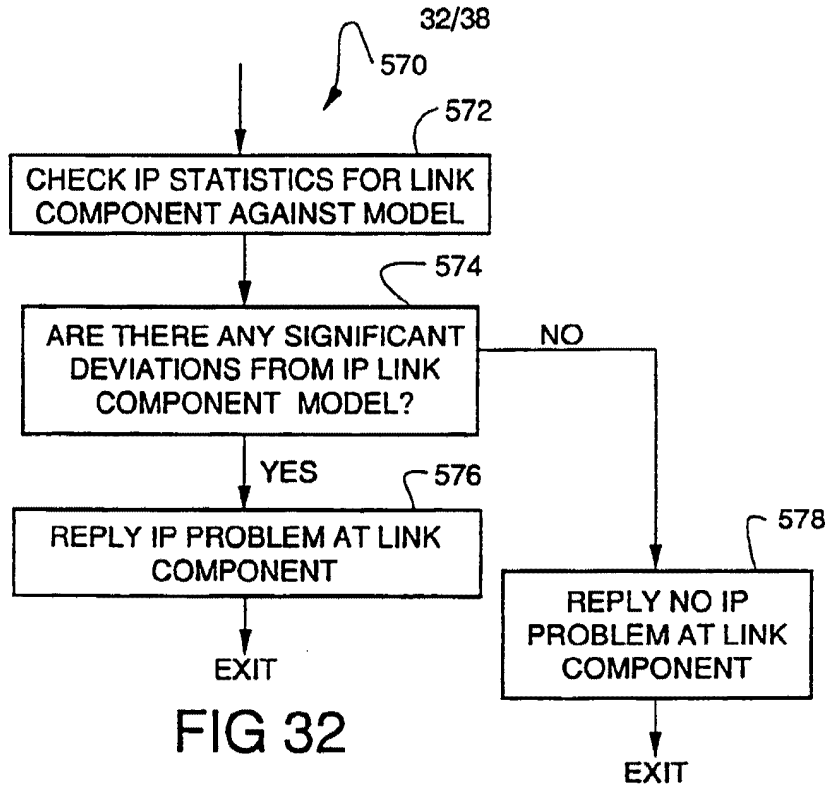


FIG 32

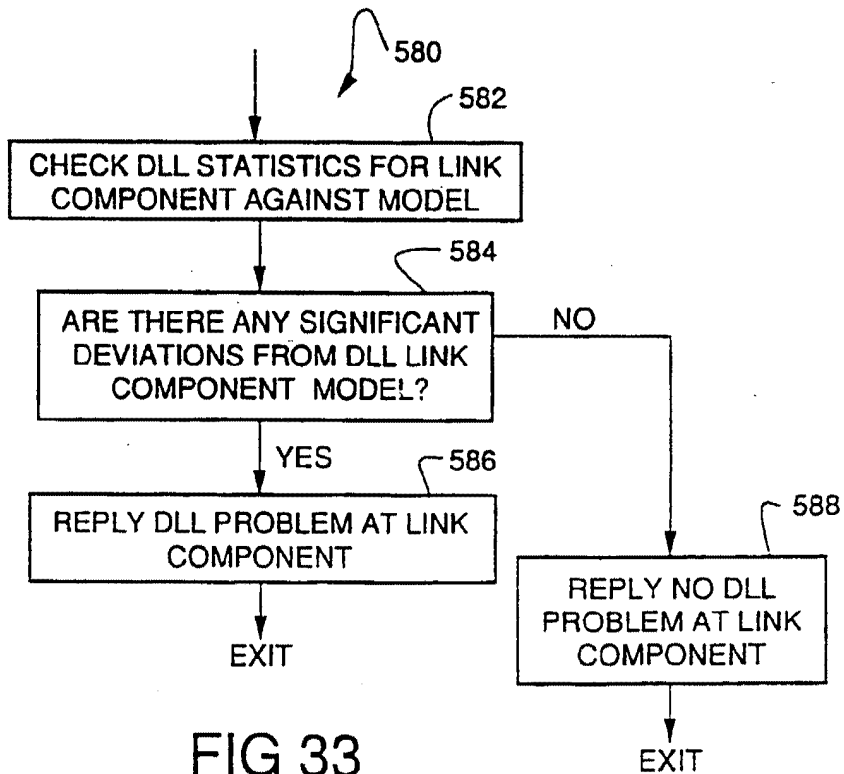


FIG 33

SUBSTITUTE SHEET.



DIALOG	ENTRY TYPE	START TIME	AVERAGE TRANSACTION TIME
//	//	//	//

FIG 34

SUBSTITUTE SHEET

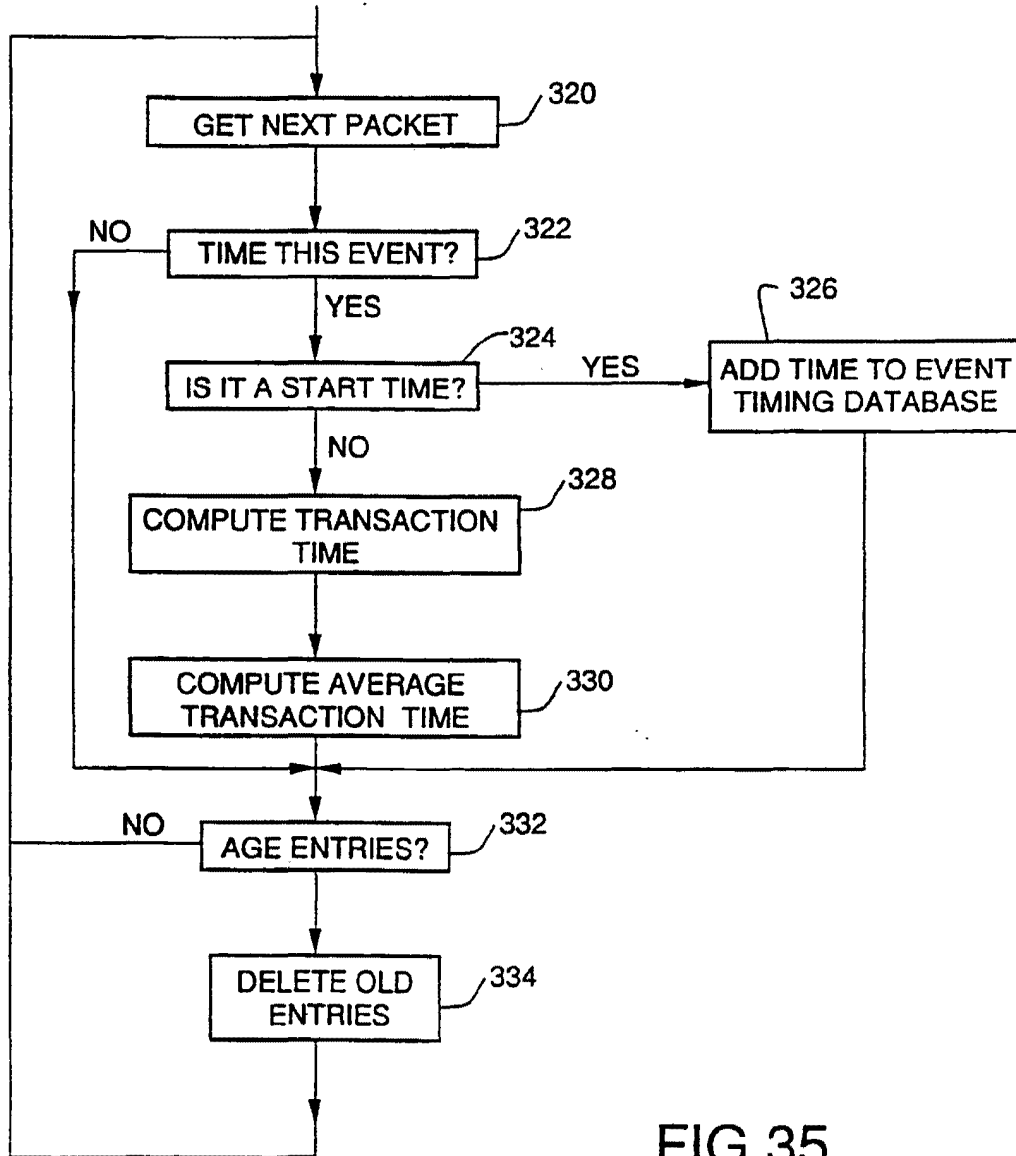


FIG 35

SUBSTITUTE SHEET

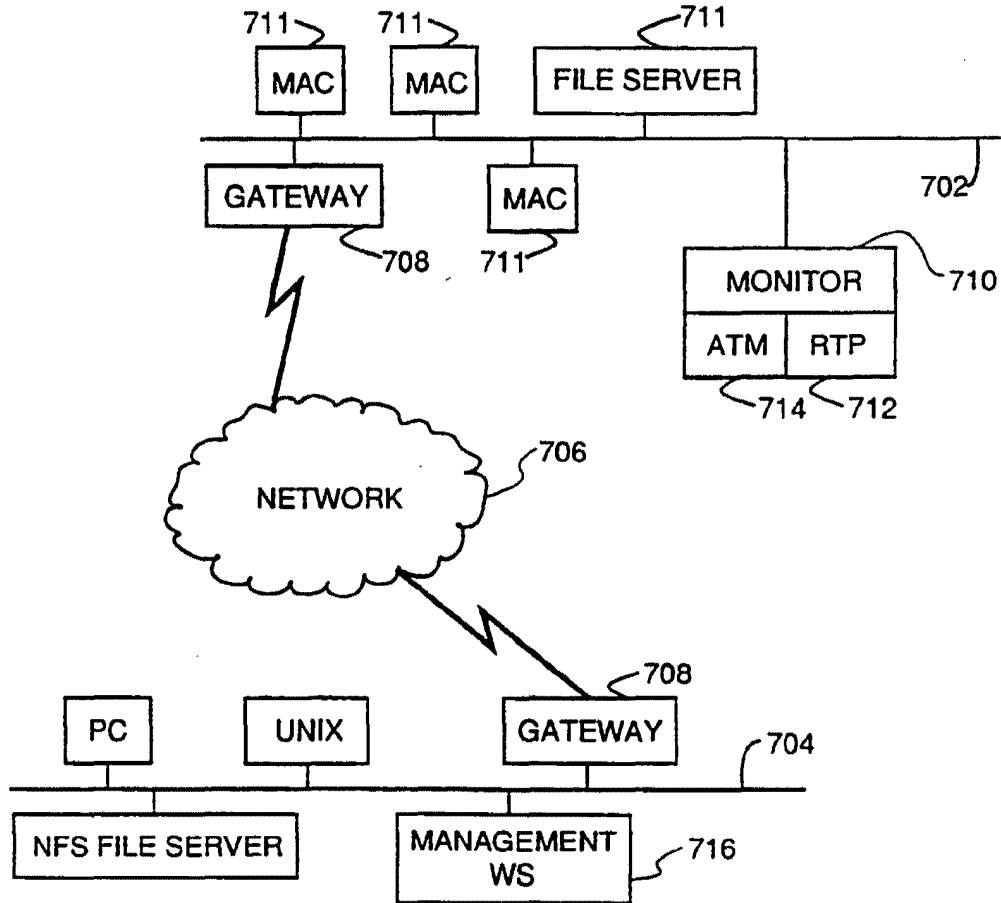


FIG 36

SUBSTITUTE SHEET

A table with four columns and five rows. The columns are labeled 'NODE NAME', 'NODE ADDRESS', 'IP ADDRESS', and 'TIME'. Callouts 724, 726, 728, and 729 point to the first, second, third, and fourth columns respectively. Callout 720 points to the entire table. Callouts 722 point to the first, second, third, and fourth rows of the table.

724 NODE NAME	726 NODE ADDRESS	728 IP ADDRESS	729 TIME

FIG 37

SUBSTITUTE SHEET

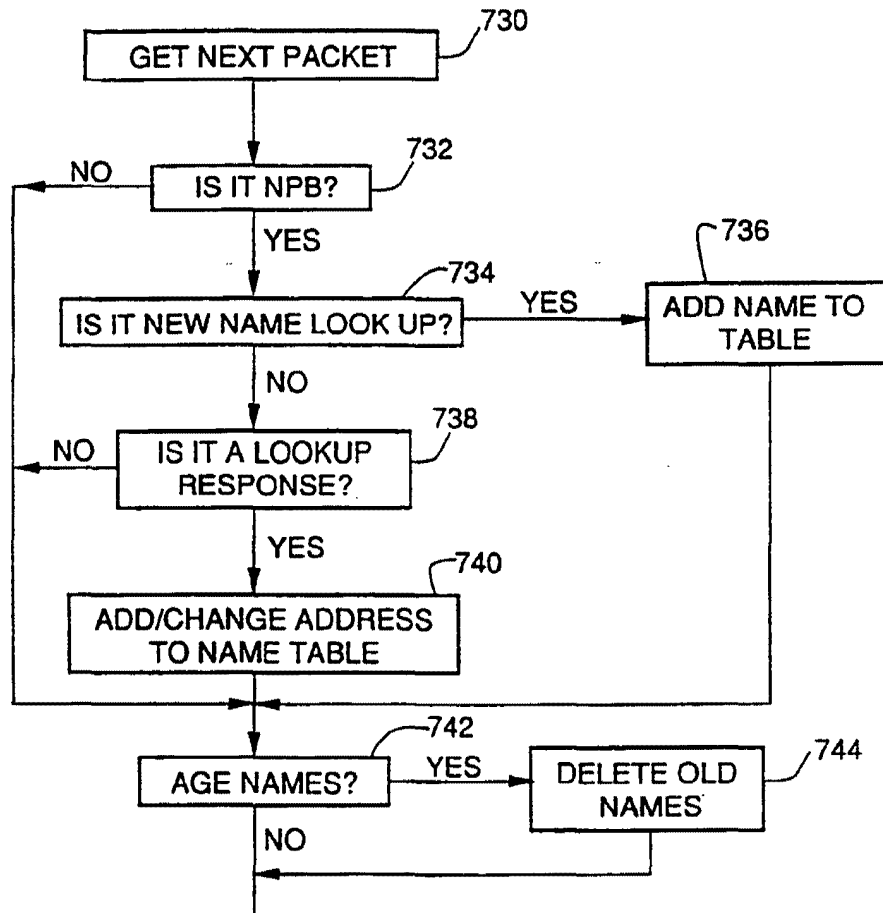


FIG 38

SUBSTITUTE SHEET

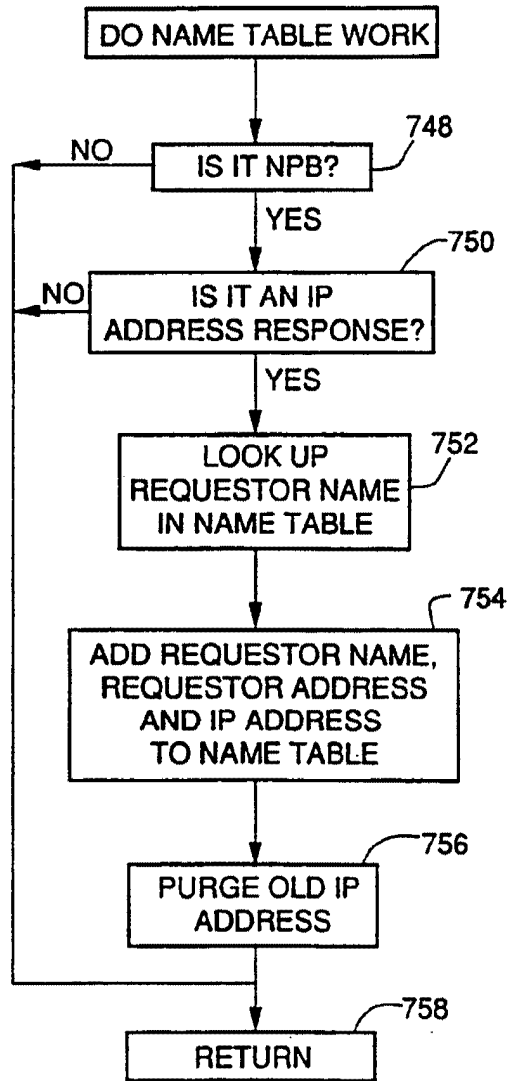


FIG 39

SUBSTITUTE SHEET

INTERNATIONAL SEARCH REPORT

International application No.  
PCT/US92/02995

<b>A. CLASSIFICATION OF SUBJECT MATTER</b> IPC(5) :H04J 3/14; H04J 3/24; H04L 12/56 US CL :370/13, 17, 94.1; 340/825.52 According to International Patent Classification (IPC) or to both national classification and IPC		
<b>B. FIELDS SEARCHED</b> Minimum documentation searched (classification system followed by classification symbols) U.S. : 370/60; 371/20.1; 340/825.06, 825.07, 825.53; 364/514, 550 Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched Electronic data base consulted during the international search (name of data base and, where practicable, search terms used) USPTO APS (Network Monitor); (Protocol analyzer)		
<b>C. DOCUMENTS CONSIDERED TO BE RELEVANT</b>		
Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X,P Y	US, A, 5,101,402 (Chiu et al) 31 March 1992 (31.03.92). Column 6, line 32 to column 8, line 10. Figs. 15 and 16.	1-7 24-26
X Y	US, A, 4,887,260 (Carden et al) 12 December 1989 (12.12.89). Column 3, lines 21-51; Column 5, lines 50-68; Column 6, line 48 to column 7, line 38; Fig. 6.	1,3-7,9 24-26,29
A,P	US, A, 5,025,491 (Tsuchiya et al) 18 June 1991 (18.06.91)	30
X	US, A, 4,817,080 (Soha) 28 March 1989 (28.03.89) column 4, lines 23.31; column 5, lines 19-37; claim 1; Fig.1 and 3.	1,24-26
<input type="checkbox"/> Further documents are listed in the continuation of Box C. <input type="checkbox"/> See patent family annex.		
* Special categories of cited documents: *A* document defining the general state of the art which is not considered to be part of particular relevance *E* earlier document published on or after the international filing date *L* document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified) *O* document referring to an oral disclosure, use, exhibition or other means *P* document published prior to the international filing date but later than the priority date claimed		
Date of the actual completion of the international search 08 JULY 1992		Date of mailing of the international search report 31 JUL 1992
Name and mailing address of the ISA/ Commissioner of Patents and Trademarks Box PCT		Authorized officer H. KIZOU Nguyen Hoc Ho NGUYEN HOC-HO INTERNATIONAL DIVISION



Europäisches Patentamt  
 European Patent Office  
 Office européen des brevets



Publication number: **0 497 022 A1**

**EUROPEAN PATENT APPLICATION**

Application number: **91300772.0**

Int. Cl.<sup>5</sup>: **G06F 15/40, G06F 9/44, H04M 3/56, H04N 7/14**

Date of filing: **31.01.91**

Date of publication of application: **05.08.92 Bulletin 92/32**

Inventor: **Jennings, Richard**  
 c/o Hewlett-Packard Limited, Nine Mile Ride  
 Wokingham, Berkshire, RG11 3LL(GB)

Designated Contracting States: **DE FR GB**

Inventor: **Baker, Colin**  
 c/o Hewlett-Packard Limited, HP Labs, Filton  
 Road  
 Stoke Gifford, Bristol, BS12 6QZ(GB)

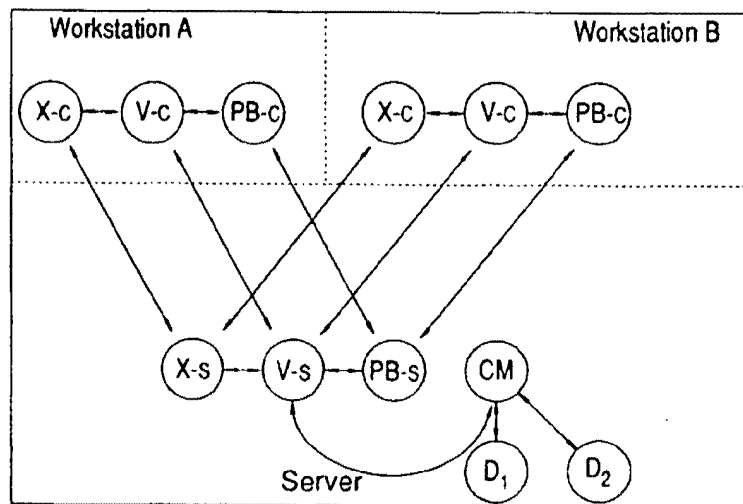
Applicant: **Hewlett-Packard Company**  
 Mail Stop 20 B-O, 3000 Hanover Street  
 Palo Alto, California 94304(US)

Inventor: **Wink, Martin**  
 c/o Hewlett-Packard Limited, Nine Mile Ride  
 Wokingham, Berkshire, RG11 3LL(GB)

Representative: **Smith, Denise Mary et al**  
 Hewlett-Packard Limited Cain Road  
 Bracknell, Berkshire RG12 1HN(GB)

**Conference system.**

The present invention relates to a distributed object-based computer system in which sharable objects are split into client and server components (see Figure 7). Each client object contains a reference to the associated server object component. By copying client object components to other users, these other users obtain access to the relevant server-object component. This feature is described in the context of a distributed conferencing system.



**FIG 7**

**EP 0 497 022 A1**



The present invention relates to a distributed computer system and relates particularly, but not exclusively, to a multimedia distributed object-based conference system.

The object-based approach to system development is becoming well-established. The basic idea is to program the system in terms of software objects, each having its own data and methods for operating on the data. Objects intercommunicate by means of messages. An advantage in encapsulating data and methods in this way is that the resulting system is relatively easy to maintain and develop. An example is NewWave Mail (produced and sold by Hewlett-Packard) which is an object-based electronic mail applications program in which messages and message components, such as text, distribution lists, etc. are treated as objects.

An object can be regarded as a discrete entity which can individually be moved, copied, destroyed, etc. An object is initially some data stored on disc or other medium. If object management software wishes to pass a message to it, one or more processes will be initiated which read the data as part of initialization. If an object is fully defined by its data and has no processes associated with it, it is said to be "inactive". If an object has one or more processes associated with it and is defined by the state of that process or processes and data then it is said to be "active".

A distributed object based system is one in which several workstations are interconnected over a network and messages between objects of the system can be sent over the network. Objects themselves may also be transmissible over the network. A network may comprise several interconnected intelligent workstations or a central computer connected to several terminals (workstations) or several interconnected server machines with intelligent workstations connected to each server, or a mixture of these possibilities. The term "workstation" is intended to be applicable to all of these possibilities.

In a distributed object based system there are benefits in splitting sharable semantic and presentation parts so as to enable more than one user to access the semantic part of a shared object. For example, in the context of a distributed conferencing system a whiteboard object would have a semantic part defining the state of the object and a presentation part for defining the appearance of the object to be displayed to a user and for enabling the user to make input. Several users may have access to a presentation part for viewing the whiteboard object so that they can each make contributions in a manner similar to a group of people clustered around a real whiteboard.

The workstations may be arranged in a client-server arrangement with semantic object parts stored on server machines and presentation object parts stored on client machines. Alternatively, semantic object parts may be distributed around user machines on a network of intelligent workstations.

According to the present invention we provide an object based distributed computer system comprising a network of workstations and means for transmitting objects between workstations characterized by objects including a first object type for storing data and a second object type for presenting data to a user, wherein objects of the second type reference an associated object of the first type to enable a plurality of users of workstations to access data of the object of the first type, comprising means for transmitting an object of the second type between workstations thereby to create a reference to the associated object of the first type for each workstation receiving an object of the second type.

The present invention provides an effective way of enabling further users to have access to a semantic object part, either for the purpose of autonomous working or for the purpose of participating in a joint activity.

In the embodiment to be described, the system comprises means for copying an object of the second type between workstations. In that embodiment transmitted objects of the second type include an identifier for the associated object of the first type.

The system according to the present invention may be in the form of a conferencing system comprising means enabling users of the workstations to participate in a meeting over the network wherein objects of the first type store meeting data and objects of the second type are for presenting meeting data. The invention also provides a method of convening a meeting using such a system comprising transmitting an object of the second type between workstations thereby to create a reference to the associated object of the first type for each workstation receiving an object of the second type.

It is believed that poor communications are a major cause of the poor performance of distributed teams of people working on a given project. The present invention advantageously provides an improved conference system for facilitating distributed meetings.

A particular embodiment of the present invention will now be described, by way of example, with reference to the accompanying drawings in which:

- Figure 1 is a diagram of a distributed system according to the present invention;
- Figure 2 shows the major components of a server and workstation of the system;
- Figure 3 shows a voice and data network structure;

Figure 4 shows video facilities for a client workstation;  
 Figure 5 shows a video network structure;  
 Figure 6 illustrates the main objects in the system;  
 Figure 7 illustrates the functionally split nature of the objects in the system;  
 Figure 8 shows the major components of the system infrastructure;  
 Figure 9 shows a typical Venue;  
 Figure 10 shows a CoMedian directory;  
 Figures 11 - 14 illustrate message sequences for system operations;  
 Figures 15 - 27 show screens during a typical user session.

10 The main components of a multi-media distributed object-based conferencing system according to the invention will first be described.

Referring to Figure 1, a multimedia distributed object-based conference system according to the present invention is indicated at 10. The system 10 comprises servers S connected over a network 12. The network 12 may be a wide area network (WAN) or a local area network (LAN) or a metropolitan area network (MAN). Client workstations C are connected to each of the servers S. Each site requires a server S.

15 Servers S communicate with each other by opening virtual circuits between pairs of servers. Although in principle, client workstations C could communicate directly with each other, this creates practical problems and therefore each client workstation C has only one virtual channel open to its local server S to enable client workstations to communicate with each other via servers S.

20 Referring to Figure 2, each server S comprises:

- hardware 14, such as an HP9000 300 HP-UX computer (HP is a trade mark of Hewlett Packard Company);
- operating system software 16, such as HP-UX software;
- Remote Object Access Manager (ROAM) software 18 for managing communications with client workstations C connected to the server S and other servers on the network;
- 25 COM software 20 providing object management facilities;
- server objects 21 which are objects to be shared between users and which correspond to the semantic object parts mentioned in the introduction.

Each client workstation C comprises:

- 30 hardware 22, such as an IBM-AT compatible PC;
- operating system software 24, such as DOS software;
- windowing software 26, such as MS Windows applications software;
- an object management facility (OMF) 28, such as a Standard NewWave OMF. (Newwave is a trade mark of Hewlett-Packard Company used for a family of applications software);
- 35 objects software 30, such as NewWave objects and specialized client objects 32 and a ROAM object 34 for handling communication with objects on other computers. The client objects 32 correspond to the presentation object parts mentioned in the introduction.

40 The user of a client workstation C therefore has a windowed user interface within which to manipulate objects of the system and can cause objects to be transmitted over the network 12 via the associated server S.

The system 10 provides multimedia facilities to users. For example, each client workstation C may have voice and/or video communication facilities as well as data communication facilities.

45 A possible voice and data network structure 40 is shown in Figure 3. In each of two sites designated A and B, a networked PC server 42 is connected to the local PABX. The PC server 42 contains one or more multi-port telephone interface cards (such as the VBX-300 card made by Natural Microsystems Inc.). The PABX is controlled by the PC server 42 and users can use their existing standard desk telephones 44 which are connected to the local PABX and conveniently located near their client workstations C.

Each of the sites A and B comprises a LAN and a LAN/WAN bridge interconnecting the LAN with a WAN.

50 In use, the PC server 42 receives commands from servers S to set up, maintain and close down telephone conference calls. To the PABX, the PC server 42 appears as a normal telephone user and can therefore dial other users adding them in to conference calls using DTMF.

55 In order to conduct conferences over a wider area, PC servers 42a and 42b on respective sites A and B connect to each other over the public switched telephone network (PSTN) and add in their own local users to the conference.

Referring to Figure 4, each client workstation C with video facilities has a video camera 46, two or more VHF TV receivers 48, a microphone 50, a preamplifier 51 and a VHF modulator 52.

Furthermore, the client workstations C may be fitted with video cards to enable a user to view video in

windows.

A possible video network is shown in Figure 5. The video network is based on a central video switch 54 connected using a star topology to client workstations C. Video signals are modulated on to VHF carriers and transmitted over standard analogue cabling 56. The video switch 54 is a conventional cable television switch. Several such switches can be cascaded in a bar arrangement for large systems.

For long distance video communications, a device 58 for compressing and decompressing video signals (a "codec") may be used and the signals are transmitted using ISDN telephone lines.

The architecture of the object-based system 10 will now be described.

With reference to Figure 6, the structure of one user's portion of the system is represented. The functions of the objects are as follows:

a Venue object (V) is an electronic meeting place allowing control over media channels and providing a location for storing shared objects. A user may have several Venue objects;

a Phone Booth object (PB) controls the creation of Venue objects and oversees the setting up, maintenance and closing down of conferences. The PB comprises a processor for handling incoming and outgoing calls;

a Connection Manager object (CM) controls driver components (D: ... D<sub>n</sub>) which handle media connections for the system 10;

a Directory object (D) which provides a list of potential meeting participants.

Object X represents another system object for performing a specific meeting-related function, eg. a whiteboard function.

Figure 6 is a conceptual representation of the system 10 and the arrows represent inter-object communication. In the embodiment being described, the system comprises client workstations C and servers S and most of the objects referred to in Figure 6 are functionally-split into a server component and one or more client components as indicated in Figure 7.

The server objects handle the centralized and distribution - oriented aspects whereas the client objects handle the presentation aspects. Hence shared applications can be written with one server object connected to a plurality of client objects on different client workstations.

In Figure 7, PB-s means a Phone Booth server object and PB-c means a Phone Booth client object, and so on.

In this embodiment, the client objects are implemented as NewWave objects ie. several new classes of NewWave objects have been added: Venue objects, ROAM objects, Whiteboard objects, Phone Booth objects. Thus the semantic part of these functionally split objects runs on an HP-UX server and the user interface runs on MS-DOS NewWave client workstations.

The client workstations are each running an object-based system of the type described in European Patent Application No.339220A, the description of which is incorporated herein as Appendix A. Appendices A-D mentioned in attached Appendix A are not attached as part of this application but are incorporated herein by reference. Appendix A describes how objects are linked together by parent-child links and how objects can be copied. During a copy operation, the container of the object to be copied sends a message to the OMF28 asking the OMF28 to copy the relevant object and identifying the container object which is to receive the copy object.

The OMF28 performs the copy function and then sends a message to the target container object instructing it to insert the copy object as one of its child objects.

Mailing an object involves serialising the object, transmitting it to its destination and deserialising it. Serialising an object involves converting it to files, say DOS files, containing the data of the object and information about its properties and its child objects.

Server objects are not linked by parent-child links in the manner in which client objects are so linked. All client objects contain a reference to their associated server object. Figure 8 shows the form of data item 60 used to name objects. The data item 60 is an eight-byte array following the convention used for Internet Protocol (IP) addresses. The first 64 bits is a machine identifier M I/D comprising a 32 bit server IP address and a 32 bit machine IP address. For a server object the server IP address and the machine IP address will be the same whereas for a client object these will be different. If there is only one domain per machine, the domain identifier D I/D is zero. The object identifier O I/D comprises a 32 bit generation count and a 16 bit tag. The 16 bit tag uniquely identifies the object within the relevant storage domain. Since tags are reusable when an object is deleted a generation count is used to ensure that each object is uniquely-named in time. The generation count is simply the time in seconds.

When a client object is closed (inactive) it appears as an icon on a user's screen. The user opens the object by clicking on the icon. Opening a client object causes it to send a message to its associated server object informing the server object that the client object is now active i.e. a Here Am I message. Until then,

the server object is unaware of the existence of the client object. In other words, links between client and server objects are non-persistent and 'weak' i.e. the existence of a server object does not guarantee the existence of a corresponding client object and vice-versa. Server objects only store the identities of corresponding client objects which are currently active. Opening a client object means that a user can view the state of the object and can make input to it. The client object regularly updates, and is updated by, the server object.

Figure 9 depicts the components involved in a typical active server object which is associated with client objects on two different client workstations C<sub>1</sub> and C<sub>2</sub>. Each object is given a unique object identifier comprising components identifying the relevant client/server machine, the relevant storage domain and a number for the particular object. On the client side, the system has an object management facility (OMF) 60 for keeping a record of what objects are presently on the particular client workstation and which is involved in object creation and deletion, object naming, object activation and deactivation and inter-object message routing. This is a standard NewWave OMF. There is a client object manager library (COMLIB-C) 61 statically linked to each client object CO providing access to the functionality of a ROAM client object 62. In other words, the COMLIB-C 61 has been added to standard NewWave objects to form the client objects for functionally split objects. Communication through the COMLIB-C 61 is network transparent, i.e. objects only need to know the object identifiers of other objects, not their locations.

On the server side there is a primitive object management facility (COM-S) 63 providing file management and object naming and message sending facilities in conjunction with the operating system software 64. A server object manager library (COMLIB-S) 65 is statically linked to each server object SO enabling access to the functionality of the object management facility 63 and a ROAM server object 66.

When client object CO<sub>1</sub> wishes to send a message to the corresponding server object SO, the ROAM client object 62 passes the message to the ROAM server object 66 which passes the message on to the server object SO. Messages from the server object SO to client objects are sent in the reverse manner. If a message is to be sent between objects on the same server the COMLIB-S 65 sends it directly without involving the ROAM server object 66. Messages are also sent between servers via the ROAM server object 66 and, in this way, communication between client workstations connected to different server machines is possible.

The functionality of certain objects in the system will now be described. The term "click" will be used in this specification to denote a selection made by the user of a workstation using an input device, such as a mouse. The term "drag" will denote moving the input device whilst such a selection is made so as to "drag" an item across the screen.

The Venue provides an electronic meeting room, inside of which person-to-person calls, group meetings and presentations to large groups can be held.

Venues provide a binding between the people involved in a meeting, the data which they are sharing, and the media channels connecting them. They are scalable from just two people up to many people, the exact number is subject to technical constraints. This allows a meeting to start off as a simple phone call between two people, build up as experts are brought in, to become a full group discussion without having to decide to move to a different object because the nature of the meeting has changed.

The Venue is a shared object and therefore exists on a server machine. The client workstations have Venue client objects which provide an interface to the Venue server objects running on the corresponding server. There may be many Venue client objects on different client workstations for a particular Venue server object.

Figure 10 shows the appearance of a Venue to a user. The Venue is being viewed in a window 70 having a title bar 72 and a menu bar 74. At the top is a participants' area 76 where the people in the Venue can be seen and where their media channels can be controlled. Beneath that is a shared area 78 where objects for use in the meeting are stored.

The participants in a Venue are displayed side by side, with each participant being represented by a still bitmap 80, a name 82 accompanied by an indication of whether that user is present in the meeting or absent and status banner 84 indicating that an absent user has been invited to the meeting, and a row of media control buttons 86. The bitmap 80 may be replaced by a motion video window when video in windows is available and the video channel is in use.

Beneath the participants' area are three media buttons 86 for telephone, video and data and each one can be in one of four states. The states are:

55

Button Appearance	Meaning
No button	This person does not have this media channel available.
White, unhighlighted	The media channel is available, but not chosen for use.
Black	The media channel has been selected, but is inactive because the person is not present in the Venue or the connection has not been completed yet.
Red	The media channel is being used.

10 The lower portion of the Venue is taken up by the shared object area 78. This acts as a shared folder, storing objects on the server and making them accessible to all users of the Venue. Inactive objects are represented by an icon such as icon 88 in Figure 10. Objects in the shared object area 78 may be client objects e.g. Whiteboard client objects, or may be standard NewWave objects. It is possible to move objects into and out of the shared object area 78 of the Venue-client object. Moving a functionally-split object such as a Whiteboard object into the shared object area 78 does not entail moving the Whiteboard-server object but just the Whiteboard-client object. The OMF28 instructs the Venue client object to insert the Whiteboard-client object as one of its children. The Whiteboard-client object is then serialised by the Venue-client object and sent to the Venue-server object. The Venue-server object updates its other active Venue-client object with the news that a new Whiteboard object is available in the Venue and these Venue-client objects display the Whiteboard-client object icon in their shared object areas 78 accordingly. The Whiteboard-server object remains on whatever server it was initially stored. Subsequent opening of the Whiteboard object by any of the users of the Venue cause a copy of the Whiteboard-client object to be serialised by the Venue-server and sent to the relevant client-workstation where it is deserialised providing access to the contents of the Whiteboard object for that user. When that user subsequently closes (deactivates) the Whiteboard object, the copy of the Whiteboard-client object remains on that machine for subsequent use.

25 In contrast, if a NewWave object icon is moved into the shared object area 78 of a Venue-client object, this causes the NewWave object to be serialised and sent from the client workstation to the server machine which stores the relevant Venue-server object. The Venue-server object then instructs its other active Venue-client objects to display the relevant NewWave object icon. Subsequent opening of the NewWave object by a user of such an active Venue-client object causes a copy of the NewWave object to be made and sent to the relevant client workstation. Each such user thus obtains a separate copy of the NewWave object and changes which a user makes are not reflected in the copies held on the other users' machines. This is a consequence of the non-functionally split nature of NewWave objects and is an implementational feature rather than one which is important to the present invention.

35 There is one Phone Booth server object on every server machine and one Phone Booth client object on every client workstation. The Phone Booth client object arranges for the creation and activation of Venue client objects on client workstations and the Phone Booth server object manages the creation of Venue server objects and the convening of Venues. On opening a Phone Booth client object the user is presented with a directory 90 of possible meeting participants as shown in Figure 11. The directory 90 comprises a list 40 of potential participants, an area 94 for displaying a picture of a participant, a media selection area 96 and an options area 98 displaying three options: Convene, Select and Cancel. Unavailable media options are greyed out in the area 96.

45 When a name is selected by choosing the Select option and then selecting a name from the directory 90, a picture of that participant appears in the area 94 as shown. The media connections are selectable by checking the relevant boxes in the media selection area 96. Checking the box beside the name of the person in the area 94 adds that person to the list of Venue participants. In addition, the initials of the media options chosen (Phone, Video, Data) appear against the participant's name in the list 92. A previously selected participant can be de-selected by de-checking the box beside the name of that person in the area 94. Taking the Cancel option means that none of the changes made since the window for the directory 90 was brought up will be implemented. The **Convene** option will be described later.

50 There is also a Connection Manager object on each server machine providing the facility to interconnect users using different media. The Connection Manager object handles the generic operations involved in establishing non-data interconnections. Drivers for each medium available, eg, video, telephone, handle the specific operations involved in carrying out switching requests during use. The Connection Manager object performs the following services:

- 55 maintains a list of media resources available in the system:
  - detects when resources fail
  - monitors resource/channel availability (ie, monitors, microphones, speakers, cameras);
  - sets up connections between people using different media:

- point-to-point
- multi-point: all that are available

maintains list of established connections and ensures synchronization with other networks, ie, maintains a model of the state of other networks;

- 5 optimizes switching to prevent unnecessary disconnect -connect transactions;
- provides an interface for monitoring and auditing;
- provides interface to media drivers.

Another functionally split object which is provided in this system is the Whiteboard. A Whiteboard object provides users with a shared computer whiteboard facility so that a user can draw, write and type on his/her  
 10 Whiteboard or acquire an image from another source and the input will be visible to other users viewing the same Whiteboard on different client workstations. Thus the Whiteboard object is an information sharing medium which allows users to look at a picture of what they are discussing.

Figure 12 shows an example of the appearance of a Whiteboard client object. The Whiteboard is being viewed in a window 100 having a title bar 102 and a menu bar 104. A drawing area 106 of the window 100  
 15 is devoted to displaying the contents of the Whiteboard, in this case a map showing the location of a Hewlett-Packard office. At the bottom of the window 100 is an area 108 indicating the range of tools which are available to the user of the Whiteboard. These tools comprise:

20	a scroller	110
	a pointer	112
	a selection of different coloured pens	114
	an eraser	116
	a text selector	118

25 Apart from the pointer 112, the tools are personal to a user ie each of the users viewing the same Whiteboard could be using the same tool eg. a red pen, without having to wait until another of the users had finished using that tool.

The scroller 110 can be used to scroll the entire window 100 around the Whiteboard. Selecting this tool  
 30 turns the cursor into a compass enabling the view of the Whiteboard to be click-dragged around by the user.

Only one user can move the pointer 112 at a time. A user takes control of the pointer by clicking on the pointer logo, - this turns the cursor into a pointer. At this time, the other users viewing the Whiteboard cannot see the pointer 112. To show the pointer 112, the user needs to click it down at a chosen point in  
 35 the drawing area 106. The pointer 112 then becomes visible to all of the Whiteboard users at that chosen position. The cursor of the user who has just moved the pointer 112 reverts to the default arrow.

Likewise the seven coloured pens are selectable and deselectable by clicking on the appropriate pen logo, enabling different users to make input in different colours.

The eraser 116 is selectable to remove marks on the Whiteboard. Also, direct typing of text onto the  
 40 Whiteboard can be done by selecting the text selector 118.

In the area 108 there is also room for a status message 120. As users open or close the Whiteboard other users are notified by a status message.

Modes of operation of a system according to the present invention will now be described, concentrating first on utilization of the Venue.

45 Once a user selects participants and media as described with reference to Figure 11 and selects the Convene option a process of events is initiated to create a new Venue object. Figure 13 shows the objects and the numbered sequence of messages. Figure 13 depicts a server machine S and two client workstations A and B connected to the server machine S. On each client workstation there is initially a Phone Booth client object PB-c. On the server machine S there is initially a Phone Booth server object PB-s and a Connection Manager object CM.  
 50

On selecting the **Convene** option using client workstation A, which causes an input (dotted line referenced 1) to the Phone Booth client object PB-c, a message (referenced 2) is sent from the Phone Booth client object PB-c to the Phone Booth server object PB-s on the server machine S causing the Phone Booth server object to create a new Venue server object V-s using a **Venue Start** message (referenced 3).  
 55 The Phone Booth server object PB-s then sends a **Ring** message (referenced 4) to the Phone Booth client object PB-c on client workstation B causing a dialogue box to appear on the screen of client workstation B inviting the user to take part in the proposed meeting. That user accepts or declines the invitation causing a corresponding message (referenced 5) to be sent from the Phone Booth client object PB-c on client

workstation B to the Phone Booth server object PB-s. If the invitation is accepted a **Create Venue** message (referenced 6) is sent from the Phone Booth server object PB-s to the Phone Booth client object PC-c<sup>1</sup> which causes it to create a new Venue client object V-c<sup>1</sup> on client workstation B involving sending a **Here Is Parent** message (referenced 7) to the new Venue - client object V-c<sup>1</sup> to notify it of the identity of the Venue server object V-s. The new Venue client object V-c<sup>1</sup> then sends a message (referenced 8) to the Venue server object V-s requesting information about the contents of the Venue. The reply from the Venue server object V-s is referenced 9 in Figure 13.

Messages corresponding to those referenced 6-9 are sent between the server S and client workstation A so as also to create a new Venue-client object V-c on that workstation and these messages are referenced 10-13 in Figure 13.

Finally, the Venue server object V-s sends a request (referenced 14) to the Connection Manager object CM to set up the chosen media connections and the Connection Manager object instructs the relevant media drivers accordingly (dotted line referenced 15).

The users of client workstations A and B can then communicate using the newly created Venue.

It is also possible to convene an existing Venue by selecting the **Convene** option within the Venue. This initiates a sequence of events which will be described with reference to Figure 14. Again, a server machine S and two client workstations A and B are represented.

The user selection of the **Convene** option is referenced 1 in Figure 14. This causes the Venue client object V-c to send a **Convene Request** message (referenced 2) to the Venue server object V-s which notifies the Phone Booth server object PB-s of the convene request in a message referenced 3 which identifies the intended meeting participants. The Phone Booth server object PB-s sends a **Ring** message (referenced 4) to the Phone Booth client objects PB-c on the workstations of the intended meeting participants causing a dialogue box to be displayed on these workstations inviting the users to partake in a meeting. When these users accept or decline the invitation this causes a reply message (referenced 5) to be sent from each Phone booth client object PB-c<sup>1</sup> to the Phone Booth server object PB-s.

The next step is for the Phone Booth server object PB-s to instruct (message referenced 6) the Phone Booth client objects PB-c<sup>1</sup> to create new Venue client objects V-c<sup>1</sup> on machines where a Venue client object linked to the Venue server object V-s is not already stored. Such new Venue client objects V-c<sup>1</sup> then send a message (referenced 8) to the Venue server object V-s requesting information about the contents of the Venue so that the appropriate icons can be displayed in the shared area 78 of Figure 10 on the respective client workstations. The reply message containing information about the contents of the Venue from the Venue server object V-s is referenced 9 in Figure 13.

The Venue server object V-s then sends a request (referenced 10) to the Connection Manager object CM to set up the chosen media connections and the Connection Manager object instructs the relevant media drivers (not shown) accordingly (dotted line referenced 11). The distributed meeting can then proceed.

A user can also set up a new Venue by selecting a **Create a New** menu option in NewWave Office (Figures 14-17 of Appendix A). On opening the new Venue-client object a Venue-server object also needs to be created. Figure 15 depicts the process. A server machine is indicated by S and a client workstation by C.

The act of opening the new Venue-client object V-c causes it to send a message (referenced 1) to the Phone Booth client object PB-c which triggers a message (referenced 2) to be sent from the Phone Booth client object PB-c to the Phone Booth server object PB-s requesting creation of a new Venue server object V-s. The Phone Booth server object PB-s creates a new Venue server object V-s using a **Venue Start** message (referenced 3). Next the new Venue-server object V-s sends a **Here Is Parent** message (referenced 4) to the Venue-client object V-c containing the ID of the Venue-server object. The new Venue client object V-c then sends a message (referenced 5) to the Venue server object V-s requesting information about the contents of the Venue and there is a corresponding reply (referenced 6) from the Venue server object.

It is possible to add new meeting participants to an active Venue by selecting an **Add New Member** menu option. This causes a directory of potential participants to be displayed as shown in Figure 11 to enable the selection of one or more further participants and associated media connections. Information on these choices is conveyed from the Venue client object to the Venue server object which updates the control panels of the relevant Venue client objects. Chosen new meeting participants are not aware of any change until someone convenes a meeting.

When a user elects to close a Venue by selecting a **CLOSE** option this causes a message to be sent from the relevant Venue-client object to its Venue-server object informing the Venue-server object that the Venue-client object is deactivating. The Venue-server object then messages the Connection Manager object

to disconnect the media connections for the Venue-client object which is deactivating. The Venue-server object sends messages to all of its other Venue-client objects informing them of the deactivation of the particular Venue-client object so that these other Venue-client objects alter their appearance to indicate that the relevant meeting member is now absent.

5 Another way of setting up a distributed meeting is for a user to copy an existing Venue-client object to the desired meeting participants. A Venue-client object is a reference to a Venue-server object. Copying a Venue-client object to other workstations creates a reference to the relevant Venue-server object on those other workstations because in the copying process the Venue-client object's reference to its Venue-server object is preserved.

10 There are different ways in which a Venue-client object can be copied to other workstations. One way is to include the Venue-client object in an electronic mail message. For this option, an electronic mail message is created in the normal manner e.g. using Hewlett-Packard's NewWave Mail and a Venue-client object is included in the message using a standard copy operation. When the or each addressee receives the message, they place the Venue-client in their collection of objects in preparation for the forthcoming meeting. At the relevant time, the meeting participants open their Venue-client objects to commence the meeting. On opening the Venue-client objects, their 32 bit machine IP address is automatically updated and the Venue-client objects send a Here Am I message to the associated Venue-server object.

15 Another option is for the user wishing to set up a distributed meeting to copy the relevant Venue-client object and to serialise the copy of the Venue-client object to a file on floppy disc (or other shared medium such as a network drive). This file may then be transported to the workstations of the intended meeting participants and deserialised thereby providing each of these participants with a copy of the Venue-client object and thereby means for accessing the associated Venue-server objects in order to take part in the distributed meeting.

20 A new Whiteboard-client object can also be created using the "Create A New" option in NewWave Office. On opening the Whiteboard-client object a new Whiteboard server object needs to be created. The process is analogous to that described with reference to Figure 15 replacing references to Venue objects with references to Whiteboard objects.

25 A new Whiteboard object can also be created inside a Venue by selecting the "Create a New" option inside the Venue. In this case, the Venue-client object automatically activates the new Whiteboard-client object in order to initiate creation of a new Whiteboard server object (again using a process analogous to that shown in Figure 15).

30 In the same manner as a Venue-client object can be copied and transmitted in an electronic mail message or via floppy disc, a Whiteboard-client object can be so utilised. Again the advantage of creating a reference to the relevant Whiteboard server object for the recipients of the copied Whiteboard-client objects is obtained since each copy of the Whiteboard-client object contains a reference to the Whiteboard server object (as described with reference to Figure 8).

35 Also as previously described, a Whiteboard-client object can be moved into the shared items area of a Venue object by a user causing copies of the Whiteboard-client object to be made available to the other users of the Venue object thereby giving access to the associated Whiteboard server object to these users.

40 An exemplary user session will now be described with reference to Figures 16 to 33 involving hypothetical users Martin, Richi and Ed.

Figure 16 shows a screen of a client workstation (Martin's) running Hewlett Packard NewWave Software. A window 126 has:

a title bar 128 carrying the title "NewWave Office";

45 a menu bar 130 offering the following options:

Action, Edit, Objects, View, Settings, Task and Help;

a system menu box 132;

size boxes 134 and 136;

a vertical scroll bar 138 with scroll arrows 140 and 142 and a scroll box 144;

50 a horizontal scroll bar 146 with scroll arrows 148 and 150 and a scroll box 152;

The window 126 displays icons for some standard tools at the top: Waste Basket 154, Agent 156, Printer 160, In Tray 162, Out Tray 164, File Drawer 166. The icons 168, 170 and 172 respectively on the left hand side represent work-related items:

55 "Project Meeting" a Venue-client object representing a reference to a Venue server object on the local server machine;

"Design Notes" a Whiteboard-client object representing a reference to a Whiteboard server object on the local server machine;

"Design Principles" a NewWave document object fully contained on the client workstation.



To "open" an object, the user double clicks on the relevant icon. Referring to Figure 17, Martin has opened the Project Meeting Venue which is shown in a window 174. The window 174 has a menu bar 176 which has similar options to the menu bar 130 of the window 126 except a Meeting option instead of the Setting option. The window 174 displays a participants area 178, showing only Martin, and a shared items area 180 which is empty. Underneath a bit map 182 of Martin is a name bar 184 which includes a notification of presence and three media control buttons 185-7 for Phone, Video and Data respectively. Only the Data button 187 is highlighted in this example, ie, blacked out in Figure 17.

On selecting the Meeting option from the menu bar 176 of the window 174, a CoMedian directory window 190 appears, Figure 18. The reference numerals for the CoMedian directory which were used in Figure 11 will be retained here. Martin selects the name Richard Jennings from the list 92 of potential participants causing a picture of Richard to appear in the area 94 together with crosses in the video and data boxes in the area 96 to indicate Richard's media selections. This means that Richard will be contacted through the system for data sharing with both video and audio travelling over video connections. Martin then clicks on the Convene button in the options area 98 to add Richard to the Venue which causes Richard's image to join Martin's image in the Venue as shown at 192 in Figure 19. Richard is marked as absent at 194 and a banner 196 is displayed indicating that he has been invited. Martin has selected both video and data connections for himself in order to match what was selected for Richard. This causes the video and data buttons 186 and 187 to be highlighted in a first colour to show that they are currently in use albeit only locally to Martin's own workstation. Richard's video and data media buttons 186a and 187a are highlighted in a second colour to indicate that they have been requested but are not yet in use.

While waiting for Richard to join the Venue, Martin is moving the Design Notes and Design Principles objects 170 and 172 into the shared items area 180 of the Venue by clicking on each object and dragging it to the area 180.

Moving now to Richard's workstation, shown in Figure 20, the invitation to join the Venue has reached his machine and has caused a bell 200 to appear at the bottom of his screen. The bell 200 is flashing and making a ringing sound to attract his attention. Richard clicks on the bell 200 and the result is shown in Figure 21. An invitation message box 202 is brought up telling Richard that he has been invited to a meeting and giving the name of the meeting and the name of the person who convened the meeting. The invitation message box 202 comprises two options: Accept and Decline. Richard clicks on the Accept option to accept the invitation to join the meeting.

Referring to Figure 22 accepting the invitation causes a Venue client object automatically to be created and a window 204 to be opened for Richard. The chosen media connections have been set up so that Richard can now see and hear Martin and the objects that Martin has placed into the shared items area 180 are available to him. Figure 23 shows that Martin can see the same Venue having the same contents on his workstation. Referring to Figure 24, during the meeting, Martin has opened a window 206 on the Design Notes whiteboard object. Martin informs Richard of this so that Richard can also view the whiteboard object and then both Martin and Richard can scribble on the whiteboard and view each others input. When their meeting is finished both Martin and Richard close and save the Venue.

Figure 25 shows the Venue object 168 saved in Richard's NewWave office. In Figure 26, Richard has just opened his NewWave office and is viewing the Venue 168 in a window 208. Martin is not present (although he would be if, coincidentally, he had his Venue open at the same time as Richard. In that situation, the relevant media connections would automatically be set up). Referring to Figure 27, Richard has selected the Meeting menu item using the cursor 210 so as to bring up the CoMedian directory 212 and he has selected Ed Davies in the manner previously described. Ed Davies does not have video capability, instead he is selected by telephone. Clicking on the Select button will cause Ed to be added to the Venue without beginning a Convene operation.

Referring to Figure 28, Richard is about to initiate a Convene operation by selecting the Action item from the menu bar 214 of the window 208, and selecting the Convene option from the corresponding menu 216. Since Ed does not have video capabilities, the audio from his telephone would be mixed into the video feed into Martin and Richard and their audio signals would be sent to Ed's telephone during their distributed meeting.

Turning now to Figure 29, a new session is beginning on Richard Jennings's workstation. A window 220 contains Richard's NewWave Office. Richard has created an outgoing message represented by the icon 222 called "Meeting Request" (using the "Create a New" option from the Action Menu - see Figures 14 to 17 of Appendix A). In Figure 30, on opening the outgoing message 222 it is displayed in a window 224. Richard has completed the distribution list 226 and written a cover note 228.

Referring to Figure 31, a new Venue-client object represented by the icon 230 is created (again using the "Create a New" option). The Venue-client object 230 is copied and dragged into the window 224

displaying the message. This is achieved by clicking on the icon 230 and pressing the control key whilst dragging the icon into the message. (This is an alternative method from the user perspective to the copy procedure described with reference to Figures 18-20 of Appendix A.) The bar 232 labelled "Part 3" in Figure 32 shows that the message now contains a copy of the Venue-client object. The message window 5 224 is then closed (Figure 33). To send the message 222 it can be dragged onto the Out Tray icon 234. This causes a copy of the message, including the Venue-client object which it contains, to be sent to the people on the distribution list. The Out Tray object 234 initiates the serialisation of the message components to enable these to be transmitted over the network. On receipt at the respective destinations, the In Tray object represented by icon 236 deserialises the message components so that these can be 10 viewed and manipulated by the recipients. The recipients can drag the Venue-client object out of the message and into their main NewWave Office window (220). At the appointed time, the three participants open their Venue-client objects to begin a distributed meeting. During the meeting, the users can open shared objects e.g. a Whiteboard object, and modify these interactively as well as interacting through their telephone and video interconnections. For example, input made by each user to a Whiteboard-client object 15 is relayed to the Whiteboard server-object which updates all of the other corresponding active Whiteboard-client objects of the changes.

Although only Venue shared objects and Whiteboard shared objects are available to a user in this embodiment, it is envisaged that further possibilities for shared objects are a fax object, a discourse 20 structurer object and tools to control the external media such as a virtual monitor manager and a video cassette recorder controller.

It is envisaged that a system according to the present invention may not entail the use of dedicated server machines but that server objects could run on user workstations given a suitable inter-object messaging infrastructure.

25

30

35

40

45

50

55

APPENDIX A

5

10

15

20

Brief Description of the Drawings

25

Figure 1 is a block diagram of a computer in accordance with the preferred embodiment of the present invention.

30

Figures 2 and 2A show block diagrams which illustrate the relationship between objects, applications and data files in accordance with the preferred embodiment of the present invention.

35

Figure 3 shows a plurality of objects linked in accordance with a preferred embodiment of the present invention.

40

45

50

55

5 Figure 4 shows a series of objects serving as folders, as parents of objects containing data, in accordance with a preferred embodiment of the present invention.

10 Figure 5 illustrates the screen display which results from linking of various objects in accordance with a preferred embodiment of the present invention.

15 Figure 6 shows the linking of objects in order to create the screen display shown in Figure 5.

20 Figure 7 shows how three objects may be linked together in accordance with a preferred embodiment of the present invention.

25 Figure 8 and Figure 9 illustrate how an object may be copied in accordance with a preferred embodiment of the present invention.

30 Figure 10 and Figure 11 illustrate the copying of a public object in accordance to a preferred embodiment of the present invention.

35 Figures 12 through Figure 71 show the appearance on a screen of a session in which a user manipulates objects in accordance with a preferred embodiment of the present  
40 invention. Also shown are block diagrams, of how objects appearing to the user are linked in accordance to the  
45 preferred embodiment of the present invention.

50 Figure 72 is a block diagram of an Object Management Facility (OMF) in accordance with the preferred embodiment of the present invention.

55

5 Figure 73 shows a block diagram of the organization of HPOMF.CAT, a system file included in the OMF shown in Figure 72.

10 Figure 74 shows the relation between a global parent and global objects in accordance with the preferred embodiment of the present invention.

15 Figure 75 is a block diagram which shows how system files within the OMF shown in Figure 72 accesses data files and applications from a memory shown in Figure 1.

20 Figure 76 is a block diagram of the organization of the memory shown in Figure 75.

25 Figure 77 and Figure 78 show objects and links in accordance with the preferred embodiment of the present invention.

30 Figure 79 is a block diagram of the organization of HPOMF.XRF, a system file included in the OMF shown in Figure 72.

35 Figure 80 shows a view specification record in accordance with the preferred embodiment of the present invention.

40 Figure 81 shows the use of a snapshot in accordance with a preferred embodiment of the present invention.

45 Figure 82 shows the data path of a view when there is no snapshot, in accordance with a preferred embodiment of the present invention.

50

55

5           Figure 83 shows the data path of a view when there is a  
snapshot, in accordance with a preferred embodiment of the  
present invention.

#### 10                           Description of the Preferred Embodiment

15           Figure 1 shows a computer 18 having a monitor 14, a  
keyboard 19 and a mouse 20. A portion of computer main  
memory 17 is shown by an arrow 9 to be within computer 18.  
20           Within computer memory main 17 is shown an object management  
facility (OMF) 100, an application 101, an application 102,  
an application 103, an application 104, an application 105  
25           and an application 106.

30           Each of applications 101 to 106 store data using  
objects. For instance, in Figure 2, application 101 is  
shown to have stored data using an object 202, an object  
203, an object 204 and an object 205. Similarly,  
35           application 106 is shown to have stored data in an object  
207, an object 208, an object 209 and an object 210. OMF  
100 stores information indicating which objects go with  
40           which application. Objects which are associated with a  
single application are considered to be objects of the same  
45           type, or the same class. For instance, object 202, 203, 204  
and 205 are of the same class because each is associated  
50           with application 101. Similarly objects 207, 208, 209 and  
210 are of the same class because each is associated with  
55           application 106. All objects of the same class use the same  
application. When an application is being run by computer

18, OMF 100 informs the application which object the  
5 application should access for data. That object is then  
considered to be active. An object is inactive when the  
10 application the object is associated with is not being run  
by computer 18, or when the application the object is  
associated with is being run, but is not being run with the  
15 data of that object.

Active objects can communicate with each other using  
20 messages. For example if two instances of application 101  
are being run by computer 18, one with the data of object  
202 and the other with the data of object 203, object 202  
25 and object 203 are both active. Therefore object 202 may  
send a message 211 to object 203. Similarly, if computer 18  
is running application 101 with the data of object 202, and  
30 is running application 106 with the data of object 207,  
object 202 and object 207 are both active. Therefore,  
35 object 202 may send a message 212 to object 207.

Messages, such as message 211 and 212 may be formatted  
40 to be sent and received by all types of objects. This  
allows for free communication between all active objects.  
This also allows new object types to be defined and added to  
45 the system without requiring that the existing object types  
be updated to use the new type.

Each object has associated with <sup>it</sup> a set of data files.  
50 For instance, object 210 is shown to have associated with it  
a data file 221, a data file 222 and a data file 223. Data  
55

5 in data files 221, 222 and 223 are in a format which can be  
interpreted by application 106.

10 Each object has associated with it a list of  
properties. Each property has a name and a value which may  
be accessed by specifying the name. In addition, each class  
15 of objects has associated with it a list of properties that  
are common to all objects of that class. For instance, in  
Figure 2A, object 205 and application 101 are shown. Object  
20 205 has associated with it a property 231, a property 232,  
and a property 233. Application 101 has associated with it  
a property 131, a property 132 and a property 133.

25 Property lists can contain any number of properties.  
Each property value can be from zero to 3,2762 bytes in  
length. Properties are used to store descriptive  
30 information about objects and classes, such as names,  
comments and so on.

35 Objects may have references to other objects. These  
references are called links. Links are directional: one  
object is called the parent, the other the child. Each link  
40 has a reference name which is a number that is assigned by  
the parent object to identify each of its children. All of  
45 an object's children, its children's children, and so on are  
collectively called that object's descendents. Similarly,  
an object's parents, its parents' parents, and so on, are  
50 collectively called that object's ancestors. In the  
preferred embodiment of the present invention, an object  
55 which may be manipulated by a user, can have zero or more



5 children and one or more parents. An object is not  
allowed to become its own descendent.

10 In Figure 3 is shown an object 301, an object 302, an  
object 303, an object 304, an object 305, an object 306, an  
object 307, an object 308 and an object 309. Objects 301-  
15 309 have links with reference names which are numbers shown  
in parenthesis by each link. Object 301 has a link 310,  
with reference name "1", to object 302. Object 301 has a  
20 link 311, with reference name "2", to object 303. Object  
302 has a link 312, with reference name "7", to object 304.  
Object 302 has a link 313, with reference name "8", to  
25 object 305. Object 303 has a link 314, with reference name  
"1", to object 306. Object 303 has a link 315, with  
30 reference name "4", to object 307. Object 304 has a link  
316, with reference name "1", to object 308. Object 305 has  
a link 317, with reference name "7", to object 308. Object  
35 306 has a link 318, with reference name "8", to object 309.  
Object 307 has a link 319, with reference name "9", to  
40 object 306. Object 307 has a link 320, with reference name  
"13", to object 309. Object 308 has a link 321, with  
reference name "1", to object 309. Object 308 has a link  
45 322, with reference name "3", to object 303.

50 Object 301 is a parent of 302 and 303. Object 303 is a  
child of object 301 and of object 308. Each of objects 302-  
309 are descendents of object 301. Descendents of object  
303 are objects 306, 307 and 309. Object 309 has for  
55

5 ancestors all of objects 301-308. Object 303 has for  
ancestors objects 301, 302, 304, 305 and 308. And so on.

10 Active objects can dynamically make and delete links to  
other objects. When a link to an object is deleted, OMF 100  
checks if the object has any other parents. If not, OMF 100  
15 destroys the object by deleting the data files of the object  
and reclaiming other storage space associated with the  
object.

20 Object links may be used for various purposes. For  
example, folders may be in the form of objects. The  
children of objects used as folders may be objects  
25 containing data for use with various applications, or the  
objects may be other folders. Figure 4 shows an example  
of the use of objects as folders. An object 401 (also  
30 called folder 401), an object 402 (also called folder 402),  
an object 403 (also called folder 403) and an object 404  
(also called folder 404) are used as folders. Folder 401  
35 contains an object 405, used to contain data, an object 406,  
used to contain data, an object 407, used to contain data,  
40 and folder 402. Folder 402 contains an object 408, used to  
contain data, folder 403 and folder 404. Folder 403  
45 contains an object 409, used to contain data, and an object  
410, used to contain data. Folder 404 contains an object  
50 411, used to contain data, an object 412, used to contain  
data and an object 413, used to contain data.

55 A more sophisticated use of links is to construct  
compound objects. For instance in Figure 5, a document 510

5 contains lines of text 511, lines of text 512, a graphics  
figure 513, a graphics figure 514 and spreadsheet data 515.  
As shown in Figure 6, text and formatting data is stored in  
10 an object 611, graphics data for graphics figure 513 is  
stored in an object 612, graphics data for graphics figure  
514 is stored in an object 613 and spreadsheet data 515 is  
15 stored in object 614. Links that are used to build compound  
objects always have some kind of data transfer associated  
20 with the link and hence are called data links. In Figure 6  
is shown a data link 615, a data link 616 and a data link  
617. In document 510, data from object 612, object 613 and  
25 object 614 are merely displayed, therefore data link 614,  
data link 615 and data link 616 are visual data links. In a  
30 visual data link, the parent will send requests to its child  
to display data within the parent's window.

35 In Figure 7, an object 701, which contains data for a  
first spreadsheet, is linked through data link 704 to an  
object 702, which contains data for a second spreadsheet,  
40 and is linked through data link 705 to an object 703, which  
contains data for a third spreadsheet. The first  
spreadsheet uses data from the second spreadsheet and from  
45 the third spreadsheet. Since the first spreadsheet does  
more than merely display data from the second and the third  
50 spreadsheets, data link 704 and data link 705 are called  
data-passing data links.

55 OMF 100 does the "bookkeeping" when objects are copied  
or mailed. When an object is copied, OMF 100 makes copies

of data files associated with the object. If the object  
5 being copied has children, OMF 100 also makes copies of the  
object's descendents, and builds links between the new  
10 objects to give the new compound object the same structure  
as the original.

For instance, Figure 8 shows object 308, from Figure 3,  
15 and the descendents of object 308. When OMF makes a copy of  
object 308, OMF copies each of object 308's descendents and  
the links shown in Figure 8. Figure 9 shows a copy of  
20 object 308. Object 308a is a copy of object 308. Object  
303a is a copy of object 303. Object 306a is a copy of  
25 object 306. Object 307a is a copy of object 307. Object  
309a is a copy of object 309. Link 321a is a copy of link  
321. Link 322a is a copy of link 322. Link 314a is a copy  
30 of link 314. Link 315a is a copy of link 315. Link 318a is  
a copy of link 318. Link 319a is a copy of link 319. Link  
35 320a is a copy of link 320.

In the preferred embodiment, the default behavior  
40 results in the copy of a parent's children when the parent  
is copied. However, when a child is designated as "public"  
it is not copied. Rather, a copy of the parent includes a  
45 link to the child. For instance, in Figure 10, a parent  
object 161 is to be copied. Parent object 161 is linked to  
a child object 162 through a link 163. Child object 162 is  
50 a public object. As shown in Figure 11, copying of parent  
object 161 results in new object 161a being linked to object

55

5 162 through a new link 163a. Object 161a is a copy of  
object 161. Link 163a is a copy of link 163.

10 In Figure 12 through Figure 71, it is shown how objects  
are displayed to a user on monitor 14. In Figure 12 a  
"NewWave Office" desktop is shown to include icons labelled  
15 as "File Drawer", "Waste Basket", "Diagnostic", "Printers",  
"Star" and "My Folder". A user (not shown) has manipulated  
a cursor 781, using keyboard 19 or mouse 20, to select "My  
20 Folder".

Figure 13 shows how the objects displayed on monitor 14  
are linked. NewWave Office (shown as an object 700) is the  
25 parent of "File Drawer" (shown as an object 701) through a  
link 711, of "Waste Basket" (shown as an object 702) through  
a link 712, of "Diagnostic" (shown as an object 703) through  
30 a link 713, of "Printers" (shown as an object 704) through a  
link 714, of "My Folder" (shown as an object 705) through a  
link 715 and of "Star" (shown as an object 706) through a  
35 link 716.

40 In Figure 14, the user, using cursor 781, has selected  
"Create a New..." in a pull down menu 782. As a result of  
this selection a dialog box 779 appears as shown in Figure  
45 15. Using cursor 781, the user has highlighted the icon  
"Layout" and using keyboard 19 has typed in the name "Paste  
Up" as a name for a new object to be created. Cursor 781  
50 now points to a region labelled "OK". Once this region is  
selected, a new object titled "Paste Up" is created, as is  
55 shown in Figure 16.

In Figure 17, "Paste Up" is shown as an object 707  
5 linked as a child of NewWave Office through a link 717.

The basic clipboard operations are Cut, Copy, and  
Paste. The user must select the data that is to be moved or  
10 copied, and then give either the Cut command or the Copy  
command. Cut moves the selected data to the clipboard  
15 (deleting it from its original location). Copy makes a copy  
of the selected data on the clipboard. The user must then  
select the location where he wants the data to be moved or  
20 copied to, and give the Paste command. This command copies  
the contents of the clipboard to the selected location.

In Figure 18 a user is shown to have selected "Paste  
25 Up". The selection is represented by the icon for "Paste  
Up" being displayed using inverse video. With cursor 781,  
30 the user selects "Copy" from a pull down menu 783. In  
Figure 18A a Clipboard object 720 is shown to be a parent of  
an object 708 through a link 721. Object 708, is a copy of  
35 object 707 ("Paste Up").

As shown in Figure 19, next the user selects "Paste"  
40 from pull down menu <sup>u</sup>783. The result, shown in Figure 20, is  
the addition of an object 708, pointed to by cursor 781,  
45 which is a copy of the original "Paste Up" object 707.

In Figure 21, the new object is shown as object 708  
linked as a child of NewWave Office through a link 718.

In Figure 22, "My Folder", has been opened by double  
50 clicking the icon for "My Folder" using cursor 781. The  
result is a new window 785 representing "My Folder".  
55

5           In Figure 23, using cursor 781, "Paste Up" (object 708)  
is shown being dragged to window 785. In Figure 24, the  
process is complete and "Paste Up" (object 708) is now in  
10 window "My Folder". In Figure 25, "Paste Up", shown as  
object 708, is now a child of "My Folder" through link 728.

15           The user sets up multiple links by using the Share  
command. This command is an extension of the clipboard  
metaphor common in software packages today for moving and  
20 copying data around the system. The clipboard is a special  
buffer that the system uses to hold data that is in transit.

25           In one way, the Share command operates similarly to the  
Cut or Copy command described above. That is, using Share,  
Cut, or Copy, the user selects some data first and then  
30 gives the Share command, which results in something being  
put on the clipboard. In the case of the Share command,  
however, what is put on the clipboard is neither the actual  
35 data nor a copy of the actual data. Instead, it is a link  
to the selected data. When this link is pasted, a permanent  
connection is made between the original data and the  
40 location of the Paste. Through use of OMF 100, this link is  
used by the involved applications to provide easy access to  
45 the original data (in its full application) and automatic  
updating when the original data is modified.

50           In Figure 26, the NewWave Office window has been  
activated. "Paste Up" (object 707) has been selected, as  
evidenced by "Paste Up" (object 707) being in inverse video.  
55 Using cursor 781, "Share" from menu 783 is selected. In

Figure <sup>26A</sup>~~720~~, Clipboard object 720 is shown to be a parent of  
 5 "Paste Up" object 707 through a link 722.

In Figure 27, window 785 has been activated. From a  
 10 menu 787, "Paste" is selected. The result, shown in Figure  
 28, is an icon 707a appearing in window 785, which indicates  
 that "Paste Up" (object 707) is shared by window 785 and the  
 15 NewWave Office window. In Figure 28A, as a result of the  
 paste, "Paste Up" is now shown to be both a child of  
 20 Clipboard 720 through link 722 and a child of "My Folder"  
 705 through a link 727. In Figure 29, showing just the  
 interconnection of objects visible to the user, "Paste Up"  
 25 (object 707) is shown to be a child of "My Folder" 705  
 through link 727. Since "Paste Up" (object 707) is shared,  
 not copied, "Paste Up" (object 707) remains a child of  
 30 NewWave Office through link 717.

One key feature of data links is automated data  
 35 transfer. When a child object is open and the user changes  
 a part of it which is "shared out", then it makes a call to  
 OMF 100. OMF 100 checks if any of the object's parents  
 40 "care" about this particular change. If they care and if  
 they are also open, OMF 100 sends to the parents a message  
 45 informing them that new data is available. The parent can  
 then send messages to the child to produce or display the  
 data. This feature allows the user to establish compound  
 50 objects with complex data dependencies, and then have  
 changes made to any sub-part be automatically reflected in  
 55 other parts. For example, changing a number in a



5 spreadsheet could cause a graph to be re-drawn, and updated  
as a figure in a document. And since an object can have  
many parents, a single object can be used as "boiler plate"  
10 for any number of other objects. A change in the boiler  
plate will be reflected in all the objects which have links  
to it. Automated data transfer is illustrated in the  
15 following discussion.

In Figure 30, window 785 for "My Folder" has been  
20 closed. In Figure 31, cursor 781 is used to select "Create  
a New..." from pull down menu 782. As a result of this  
selection dialog box 779 appears as shown in Figure 32.  
25 Using cursor 781, the icon HPText has been highlighted and  
using keyboard 19 the name "Sample Text" has been typed in  
as the name for a new object to be created. Cursor 781 now  
30 points to a region labelled "OK". Once this region is  
selected, a new object titled "Sample Text" is created, as  
35 is shown in Figure 33.

In Figure 34, "Sample Text" (object 709) is shown to be  
40 a child of NewWave Office through a link 719. In Figure 34,  
since "My Folder" has been closed, "Paste Up" (object 708),  
link 728 and link 727 are not shown. However, these still  
45 exist, but are not currently visible to a user.

In Figure 35, placing cursor 781 on the icon "Sample  
50 Text" and double clicking a button on mouse 20 results in  
"Sample Text" being opened. In Figure 36, an open window  
789 for "Sample Text" is shown.

55

In Figure 37 a window 791 for "Paste Up" (object 707)  
5 has been opened by double clicking on the icon for "Paste  
Up". In Figure 38, using Cursor 781, controlled by mouse  
20, a portion 790 of the text of "Sample Text" has been  
10 selected. The portion in inverse video stating "New Wave  
Office environment" is portion 790.

15 In Figure 39, cursor 781 is used to select the  
selection "Share" in a pull down menu 792. In Figure 40,  
an area 793 in window 791 is selected using cursor 781. In  
20 Figure 41, a selection "Paste" is selected from a pull down  
menu 794 using cursor 781. In Figure 42, "Sample Text" is  
25 linked to "Paste Up" (object 707) and displayed text 790 is  
displayed in "Paste Up" window 791. In Figure 43 "Sample  
Text" (object 709) is shown to be a child of "Paste Up"  
30 (object 707) through a link 729. In Figure 42, displayed  
text 790 is shown in gray because "Star" window 789 is open.  
35 In Figure 44, "Star" window 789 is closed so displayed text  
790 is clearly displayed.

40 In Figure 45, a region 795 of window 791 is selected  
using cursor 781. Figure 46 shows cursor 781 dragging the  
icon "Star" into region 795 of window 791.

45 In Figure 47, data from "Star" (object 706) is now  
displayed in region 795 of window 791. As may be seen in  
Figure 48, "Star" (object 706) is now a child of "Paste Up"  
50 (object 707) through a link 726.

In Figure 49, a user has placed cursor 781 over region  
55 795 of window 791 and double clicked a button on mouse 20.

5 The result is the opening and display of "Star" (object 706)  
in a window 796. Figure <sup>50</sup>~~40~~ shows the use of cursor 781 to  
select selection "Ellipse" in a menu window 797 which  
10 results in the data within "Star" (object 706) being changed  
from a star to an ellipse. As shown in Figure 51, the  
result is a change both in data displayed in window 796 and  
15 data displayed in region 795 of window 791.

In Figure 52, cursor 781 is used to define a region 797  
20 in window 791. In Figure 53, cursor 781 is used to select a  
selection "Create a New..." in pull down menu 798. As a  
result of this selection dialog box 799 appears in Figure  
25 54. Dialog box 799 contains icons for the two classes of  
objects available which are able to display data in region  
797 of window 791. Using cursor 781, the icon "HP Shape"  
30 has been highlighted. Using keyboard 19 the name "New  
Shape" has been typed in as the name for a new object to be  
35 created. Cursor 781 now points to a regions labelled "OK".  
Once this region is selected, a new object titled "New  
Shape" is created. Data for "New Shape" is displayed in  
40 region 797 of window 791 as is shown in Figure 55. In  
Figure 56, "New Shape", (object 750) is shown to be a child  
45 of "Paste Up" (object 707) through a link 760.

In Figure 57 a window 800 for "New Shape" was opened by  
50 placing cursor 781 over region 797 of window 791 and  
clicking twice on a button on mouse 20. In Figure 58,  
cursor 781 is used to select the selection "Triangle" from a  
55 pull down menu 801. The result, as shown in Figure 59, is

5 that a triangle is now displayed both in window 800 and in  
region 797 of window 791.

10 In Figure 60, window 800 has been closed. In Figure  
61, "New Shape" is selected by placing cursor 781 over  
region 797 of window 796, and clicking a button on mouse 20.  
15 In Figure 62, cursor 781 is used to select selection "Share"  
from pull down menu 794. In Figure 63, cursor 781 is used  
to select a region 802 of window 791. In Figure 64, cursor  
20 781 is used to select selection "Paste" from pull down menu  
794. The result, as shown in Figure 65, is the sharing of  
"New Shape" with data from "New Shape" being displayed in  
25 region 797 and in region 802 of window 791. In Figure 66,  
"New Shape" (object 750) is shown to have an additional link  
770, from its parent "Paste Up" (object 707).

30 In Figure 67, region 797 has been selected using cursor  
781. Cursor 781 is then used to select selection "Cut" from  
35 pull down menu 794. The result, as seen in Figure 68, is  
that region 781 has been removed from window 791. In Figure  
69, cursor 781 is used to select selection "Paste" from pull  
40 down menu 783. The result, shown in Figure 70, is an icon  
for "New Shape", pointed to by cursor 781. In Figure 71,  
45 "New Shape (object 750) is shown to now be a child of  
NewWave Office (object 100), through a link 780.

50 In Figure 72, OMF 100 is shown to contain seven system  
files: system file 601, system file 602, system file 603,  
system file 604, system file 605, system file 606 and system  
55 file 607. OMF interface 599 serves as interface of OMF to

other programs running on computer 18. System files 601-607 serve as a data base that provides various information.

5 They provide information about object properties such as what class each object is what is the name of each object.  
10 System files 601-607 provide information about classes of objects such as what application is associated with each class of objects, what icon represents objects of a  
15 particular class and lists of what messages (such as those shown in Figure 2) can be processed by objects of a particular class. System files 601-607 also contain  
20 information about links between parent and child objects including a list of parents and reference names of each link from a parent for each object; a list of children and  
25 reference names of each link to a child for each object; and additional information to manage data exchange across data  
30 links. Additionally, system files 601-607 contain general information such as what files are installed in the  
35 operating system for each class that is installed, and what objects have requested automatic restart when the OMF 100 is  
40 restarted.

In the preferred embodiment of the present invention system file 601 is referred to as HPOMF.CAT, system file 602  
45 is referred to as HPOMF.CLS, system file 603 is referred to as HPOMF.XRF, system file 604 is referred to as HPOMF.PRP, system file 605 is referred to as HPOMF.INS, system file 606  
50 is referred to as HPOMF.SDF and system file 607 is referred

55

to as HPOMFICO.NWE. A description of each system file is now given.

5           System file 601, HPOMF.CAT, is also referred to as  
SYSCAT. HPOMF.CAT is a catalog of all the existing objects  
10           in the system. In Figure 73, HPOMF.CAT is shown to be  
record oriented. HPOMF.CAT has a plurality of file records.  
15           In Figure 73, file record 0 through file record 8 are shown,  
although HPOMF.CAT may contain many more file records than  
are shown in Figure 73. File record 0 is a header which  
20           contains various signatures and is used to manage a list of  
free file records. A signature is some known value which if  
present indicates that the file is not corrupted. File  
25           record 1 through file record 8 and additional file records  
(not shown) either define an existing object, or are free.  
In the preferred embodiment HPOMF.CAT can grow dynamically,  
30           as more file records are needed, but cannot shrink.

          File record 1 defines a special object called the  
35           global parent. The global parent has a form different than  
every other object, and may be regarded as a "pseudo"  
object. Figure 74 shows the global parent to be the parent  
40           of global object 250 through link 260, global object 251  
through link 261, global object 252 through link 262, global  
45           object 253 through link 263, global object 254 through link  
264 and global object 255 through link 265, as shown.  
Global objects 250-255 are also within HPOMF.CAT. Each  
50           global object 250-255 may be a parent of one or more objects  
in HPOMF.CAT. Each object in HPOMF.CAT which is not a

55

global object, is a descendent of global object. Although  
 5 Figure 74 shows only six global objects, the number of  
 global objects operating on a system is a matter of system  
 configuration. Any object in the system can refer to a  
 10 global object by ~~by~~ using the reference name of the link to  
 that global object from the global parent.

15 As may be seen from Figure 73, file records in  
 HPOMF.CAT are numbered consecutively. These numbers serve  
 as tags, which identify each object.

20 In the preferred embodiment of the present invention,  
 each record is 128 bytes in length. The fields for file  
 25 record 0 are listed in Table 1 below:

Table 1

30	lFirstFreeEntry	Contains the record number of the first free record in HPOMF.CAT, or "0" if there are no free records.
35	FileId	Contains the null terminated string "HPOMF.CAT". This serves as a signature.
40	Version	Contains the file format version number, which also serves as a signature.
45	lMaxRecordNumber	Contains the number of the highest record ever allocated from within HPOMF.CAT (this highest record may or may not be free).

50 Table 2, below, contains the fields for file records in  
 HPOMF.CAT for file records other than file record 0:

55

Table 2

5	lFirstFreeEntry	Is "-1" if this record defines an object, otherwise this record is free and this field is the record number of the next free record, or "0" if there are no more free records. If the record is free, none of the other fields in the record is meaningful.
10		
15	TypeInClass	Specifies the class of this object. This is the number of the record in HPOMF.CLS that indicates to which class the object belongs (see discussion of class above).
20		
25	SysCatFlags	Specifies if the object is global if the bit masked by the number 20 (hexadecimal) is set in this byte. In the preferred embodiment all other bit positions must contain "0" and are not used.
30		
35	properties	Specifies the number of properties, the length of the property names and the location in HPOMF.PRP of the object's properties. See the description of HPOMF.PRP below for further definition of the structure of this field.
40		
45	fastprops	Certain object properties, such as name, are so heavily accessed that they are stored directly in this field, rather than indirectly in the properties file. Properties stored in this field are called "fast properties."

50           System file 602, HPOMF.CLS is also referred to as  
 SYSCLASS. This system file is a list of all installed  
 classes in the system. It is record oriented. The first  
 55 record, numbered 0, is a header which contains various



signatures (see above) and is used to manage a list of free  
 5 records. All other records either define an installed class  
 or are free. In the preferred embodiment HPOMF.CLS can grow  
 10 dynamically, but cannot shrink.

Each file record in HPOMF.CLS is thirty-two bytes in  
 length. HPOMF.CLS file record 0 (the header) contains the  
 15 following fields listed in Table 3:

Table 3

20	lFirstFreeEntry	Contains the record number of the first free record in HPOMF.CLS, or "0" if there are no free records.
25	FileId	Contains the null terminated string "HPOMF.CLS"
30	Version	Contains the file format version number.
35	lMaxRecordNumber	Contains the number of the highest record ever allocated from within HPOMF.CLS (this highest record may or may not be free).

Table 4, below, contains the fields for file records in  
 40 HPOMF.CLS for file records other than file record 0:

45

50

55

Table 4

5	lFirstFreeEntry	Is "-1" if this record defines an installed class, otherwise this record is free and this field is the record number of the next free record, or "0" if there are no more free records. If the record is free, none of the other fields in the record is meaningful.
10		
15	ModuleFileName	Specifies the name of the application associated with objects of this class as a null-terminated string.
20	properties	Specifies the number of properties, the length of the property names and the location in HPOMF.PRP of the object's properties. See the description of HPOMF.PRP below for further definition of the structure of this field.
25		

30 In Figure 75, the relationship of HPOMF.CAT and HPOMF.CLS is shown. Within each object entry within HPOMF.CAT, the record number, which is an object's tag, serves as an identifier 650 of data files in a mass storage memory 170 associated with the object. The field "TypeInClass" serves as an identifier 651 of the class entry in HPOMF.CLS, which identifies the class of each object. Within each class entry in HPOMF.CLS, the field "ModuleFileName" serves as an identifier 652 of the application file in mass storage memory 170 which is associated with the class.

50 In Figure 76, the organization of a portion of mass storage memory 170 is shown. A root directory 660 contains pointers to an HPNWDATA directory 661 and HPNWPROG directory

5 668. HPNWPROG directory 668 is the location of storage for  
applications files, represented by arrows 669. HPNWDATA  
contains a plurality of HPOMFddd directories, represented by  
10 directories 662, 663, 664, 665 and 666. In the HPOMFddd  
directories are stored data files associated with objects.  
The "ddd" in HPOMFddd stands for a three digit, leading  
15 zeros, hexadecimal number. Each HPOMFddd directory has a  
different "ddd" hexadecimal number. The "ddd" number  
20 indicates which HPOMFddd directory stores data files for a  
particular object. Data files for a particular object are  
stored in the HPOMFddd directory which has a "ddd" number  
25 equal to the tag for the object divided by an integer  
number, e.g., fifty four. Within each HPOMFddd directory,  
30 files are stored by tag numbers, e.g. data file names have  
the format xxxxxxxx.lll, where "xxxxxxx" is an eight digit  
leading zeros hexadecimal tag, and "lll" are a reference  
35 chosen by the application.

System file 603, HPOMF.XRF is also referred to as  
40 SYSXREF. This file is a list of all the links existing in  
the system. It is record oriented, but does not have a  
header record. Each record file is either free, or defines  
45 an existing link, or is used as an overflow record from the  
previous record to specify additional view specification  
information. Records that contain view specifications are  
50 called view specification file records. View specification  
file records can be identified only by a previous record  
55 which defines an existing data link; view specification file

5 records cannot be identified by the content within a view  
specification file record. HPOMF.XRF is increased in size  
16K bytes at a time. A newly allocated portion of HPOMF.XRF  
10 is filled with zeros. File records within HPOMF.XRF which  
are free or which define a link have the following fields  
listed in Table 5:

15

Table 5

20	ParentTag	Contains the tag (HPOMF.CAT record number) of the parent object of this link. If this field is 0, then this record does not define a link and is free.
25	ChildTag	Contains the tag of the child object of this link. If ParentTag in this record is 0, and this field is also 0, then no record beyond this record in HPOMF.XRF defines a link.
30	RefName	Contains the reference name that the parent has assigned to the link. This field is meaningless if ParentTag or ChildTag is zero. Otherwise, if the top three bits of this value are 110, the next record in the file is a view specification.

35

40

File records within HPOMF.XRF which are view  
specification file records have the following fields listed  
45 in Table 5A:

50

55

Table 5A

5	DataId	Contains the value that the child has assigned to identify the part of itself that is being viewed through the link.
10	Snapshot	Contains the tag (HPOMF.CAT record number) of the object which is the view's snapshot, or if zero, the view has no snapshot. For further discussion of snapshots, see below.
15	Misc	Composed of several bit fields described below:
20	VS_NEWDATASET	Set if child has told OMF that new data is available, but has not been announced to the parent. The hexadecimal number 8000 0000 is a mask which indicates which bits are used for this bit field.
25	VS_NEWDATAANNOUNCED	Set if child has told OMF to announce new data to parent, but parent was inactive and was not notified. The hexadecimal number 4000 0000 is a mask which indicates which bits are used for this bit field.
30	VS_SNAPSHOTOLD	Set if child has told OMF that the view's snapshot is out-of-date. The hexadecimal number 2000 0000 is a mask which indicates which bits are used for this bit field.
35	VS_WANTMESSAGES	Set if child has told OMF that it wants to process view messages when snapshot is out-of-date. The hexadecimal number 1000 0000 is a mask which indicates which bits are used for this bit field.
40		
45		
50		
55		

5	VS_TEXTDISKLOC	File position in HPOMF.PRP where a view's 32 character textual data ID is located. This contains zero if no textual data ID has been defined by the child. The low order five bits of the file position are always zero and are thus not stored in the Misc field. The hexadecimal number OFFF FFE0 is a mask which indicates which bits are used for this bit field.
10		
15	VS_INITIALIZED	Set if the view specification has been initialized. If clear, all information in the view specification is zero. The hexadecimal number 0000 0010 is a mask which indicates which bits are used for this bit field.
20		
25	VS_RESERVED	Reserved for future expansion. The hexadecimal number 0000 0008 is a mask which indicates which bits are used for this bit field.
30		
35	VS_VIEWCLASS	Specifies the view class the child assigned to the view. The view class defines what view methods are available to the parent. The hexadecimal number 0000 0007 is a mask which indicates which bits are used for this bit field.
40		
45		

For example, in Figure 77, Object 671 is a folder and has a tag of "6". Object 671 is a parent of an object 672 through a link 674 and a parent of an object 673 through a link 675. Object 672 has a tag of "12". Link 674 as a reference name "1". Object 673 has a tag of "19". Link

5 675 has a reference name "7". Reference names are picked by  
the parent object and need to be unique for the particular  
parent object; however, other parents may have a link with  
10 the same reference name provided each reference name is  
unique for each parent.

Figure 79 shows a block diagram of HPOMF.XRF 603.  
15 HPOMF.XRF contains an entry for each link between parents  
and children. In HPOMF.XRF 603 column 731 contains the tag  
of the parent for each link. Column 732 contains the tag of  
20 the child for each link. Column 733 contains the reference  
name for each link. The first three bit positions of column  
25 733, shown in Figure 79 as sub-column 734, indicate whether  
a view specification file record is present ("110") whether  
no view specification file record follows ("000") or whether  
30 the link is between is a link from the global parent to a  
global object ("100").

35 As may be seen, entry 735 is an entry which describes  
link 674 shown in Figure 77. That is, in column 731 of  
entry 735 there is the parent tag "6". In column 732 there  
40 is the child tag "12" and in column 733 there is the  
reference name "1". Since object 671 is a folder, there is  
45 no view, therefore the three bits within subcolumn 734 would  
be "000".

50 Similarly, entry 736 is an entry which describes link  
675 shown in Figure 77. That is, in column 731 of entry 736  
there is the parent tag "6". In column 732 there is the  
55 child tag "19" and in column 733 there is the reference name

5 "7". Since object 671 is a folder, there is no view,  
therefore the three bits within subcolumn 73<sup>4</sup> would be  
"000".

10 In Figure 78, Object 676 is a document and has a tag of  
"17". Object 676 is a parent of an object 677 through a  
link 679 and a parent of an object 678 through a link 680.  
15 Object 677 has a tag of "8". Link 679 as a reference name  
"1". Object 678 has a tag of "21". Link 680 has a  
20 reference name "3".

In Figure 79, an entry 737 describes link 679 shown in  
Figure 78. That is, in column 731 of entry 737 there is the  
25 parent tag "17". In column 732 there is the child tag "8"  
and in column 733 there is the reference name "1". Object  
676 is a document, and assuming there is a view associated  
30 with link 679, the three bits within subcolumn 73<sup>4</sup> contain  
the three bits "110" and entry 738 is a view specification  
35 record.

Similarly, an entry 739 describes link 680 shown in  
Figure 78. That is, in column 731 of entry 739 there is the  
40 parent tag "17". In column 732 there is the child tag "21"  
and in column 733 there is the reference name "3". Assuming  
45 there is a view associated with link 680, the three bits  
within subcolumn 73<sup>4</sup> contain the three bits "110" and entry  
740 is a view specification record.

50 In Figure 80, view specification record 740 is shown to  
include a field 741 which contains a data identification for  
55 the view, a field 742 which indicates whether there is a



5 snapshot used in the view, and a field 743 which contains  
miscellaneous information about the view. The data  
identification number is used by the child object of the  
link, to determine what data is sent through the link.

10 Figures 37 - 43 show the establishment of a link with  
a view. As has been discussed before, in Figure 37 window  
15 791 for "Paste Up" (object 707) has been opened by double  
clicking on the icon for "Paste Up". In Figure 38, using  
Cursor 781, controlled by mouse 20, portion 790 of the text  
20 of "Sample Text" has been selected. The portion in inverse  
video stating "New Wave Office environment" is portion 790.

25 In Figure 39, cursor 781 is used to select the  
selection "Share" in a pull down menu 792. Once "Share" is  
selected, child object 709 ("Sample Text") creates a data  
30 identification number which identifies portion 790 of the  
text to child object 709. Child object 709 also causes OMF  
100 to put a link to child object 709 on clipboard 720--  
35 Child object 709 communicates to OMF 100 through command set  
forth in Appendix B, attached hereto--. Child object 709  
40 also informs OMF 100 what data identification number is  
associated with the new link between the child 709 and  
45 clipboard 720. If there is a snapshot associated with the  
link, child 709 will also inform OMF 100 if there is a  
snapshot associated with the link. Snapshots are discussed  
50 more fully below. As a result OMF 100 will make an entry in  
HPOMF.XRF 603 for a link between clipboard 720 and child  
object 709. The view specification record for the link will  
55

5 include the data identification number given to OMF 100 by  
child 709.

10 In Figure 40, area 793 in window 791 is selected using  
cursor 781. In Figure 41, a selection "Paste" is selected  
from a pull down menu 794 using cursor 781. At this point  
parent object 707 ("Paste Up") requests OMF 100 for a link  
15 making him the parent of what is on clipboard 720. The view  
specification record for the <sup>link</sup> between clipboard 720 and child  
20 709 is copied for link 729 between parent 707 and child 709.  
In Figure 43 "Sample Text" (object 709) is shown to be a  
child of "Paste Up" (object 707) through link 729.

25 In Figure 42, "displayed text 790 is displayed in  
"Paste Up" window 791. In accomplishing this, parent object  
30 707 makes a call to OMF 100 asking that a message be sent to  
the object identified by the reference name for link 729.  
This message requests the child object 709 to display data  
35 from this link into a location specified by parent object  
707. OMF 100 takes the message from parent 707, adds the  
40 data identification number from the view specification  
record for link 729, and delivers the message to child 709.  
Child 709 displays the data in the specified location, in  
45 this case area 793. The name of the message sent from  
parent 707 to OMF 100 to child 709 is "DISPLAY\_VIEW",  
50 further described in Appendix B, attached hereto.

Another message "PRINT\_SLAVE", also described in  
Appendix B, may be used when it is desired to print data on  
55 a printer rather than display data on a terminal screen.

5 In addition, Parent 707 may send a "GET\_SIZE" message  
to child object 709. In a "GET\_SIZE" message, parent object  
707 identifies a reference name for link 729 and indicates  
10 coordinates for a display. OMF 100 takes the GET\_SIZE  
message from parent 707, adds the data identification number  
from the view specification record for link 729, and  
15 delivers the message to child 709. Child 709 returns to  
parent 707 the size of the portion of the specified area  
that child 709 would use to display the data. This allows  
20 parent 707 to modify the region reserved for displaying data  
from child 709 when child 709 is not able to scale the data  
25 to fit in the region specified by parent 707.

30 When a data from a child object is being displayed by a  
parent object, and the child object changes the displayed  
data, the child objects notifies OMF 100 that there has been  
a change in the data object. For example, as described  
35 above, in Figure 47, data from "Star" (object 706) now  
displayed in region 795 of window 791. And, as may be seen  
in Figure 48, "Star" (object 706) is a child of "Paste Up"  
40 (object 707) through a link 726. Since data is being passed  
from child object 706 to parent object 707, link 726 is a  
45 data link which includes a view specification.

In Figure 49, the method for changing data in child  
50 object 706 is shown. A user places cursor 781 over region  
795 of window 791 and double clicks a button on mouse 20.  
The result is the opening and display of "Star" (object 706)  
55 in a window 796. Using cursor 781 to select selection

"Ellipse" in a menu window 797 results in the data within  
5 "Star" (object 706) being changed from a star to an ellipse.  
As shown in Figure 51, the result is a change both in data  
displayed in window 796 and data displayed in region 795 of  
10 window 791.

Child object 706 accomplishes this change by making a  
15 call to OMF 100 stating that data associated with the data  
identification number associated with link 726 is changed.  
OMF 100 looks up all of the links that use the data  
20 identification number. If the parent object of any of the  
links is not active, OMF 100 sets the bit  
25 VS\_NEWDATAANNOUNCED for that link in HPOMF.XRF. When the  
parent object is activated, the parent object can then  
request the new data.

30 If the parent object is active, OMF 100 will send a  
message to the parent object saying that new data is  
available. OMF 100 will identify to the parent object the  
35 reference name of the link for which there is additional  
data. The parent object sends a message to the child object  
40 if it wants the new data displayed. In the present case  
parent object 707 is active , and has requested the new data  
45 to be displayed in region 795 of window 791. A further  
description of the View Specifications are found in  
Appendixes B, C and D.

50 The advantage of the present invention is that parent  
object 707 is able to communicate with child object 706  
through OMF 100, without parent object 707 or child object  
55

706 knowing the identity or any other details about each  
 other. The parent object identifies the link using only the  
 5 reference name of the link. The child object identifies the  
 link using just the data identification number of the link.  
 10 OMF 100 does all the translation and identification of which  
 links and which objects are involved.

System file 604, HPOMF.PRP, is also referred to as  
 15 SYSPROP. HPOMF.PRP contains all the object and class  
 properties except for the fast object properties which are  
 contained in HPOMF.CAT. Each record in system file 601  
 20 (HPOMF.CAT) and system file 602 (HPOMF.CLS) has a properties  
 field, as described above. Each properties field contains  
 25 the fields described in Table 6 below:

Table 6

30	DirDiskLoc	Contains the position (byte offset) within HPOMF.PRP of the property list directory.
35	nProps	Contains the number of properties in the property list. This is the number of entries in the directory entry array described below.
40	PoolSize	Contains the combined length of all the names of the properties in the property list, including a null-terminating byte for each name. This is the size of the directory name pool described below.
45		

50 For each object and for each class, at the DirDiskLoc  
 position in the HPOMF.PRP file is the property directory for  
 55 that object or that class. The directory has two major

portions: the entry array, followed by the name pool. The  
 5 entry array has one entry for each property in the property  
 list. Each entry has fields set out in Table 7 below:

Table 7

10	ValueLen	Specifies the length in bytes of the associated property. This can be zero.
15	ValueDiskLoc	Contains the position within HPOMF.PRP of the value of the associated property. If ValueLen is zero, this is also zero, and there is no value stored anywhere.
20	CacheOffset	This field is only used at run time and is not meaningful in the file.
25		

Immediately following the entry array is the name pool.  
 This portion of HPOMF.PRP contains the null-terminated names  
 30 of properties in the property list, in the same order as the  
 entry array. Properties may include such things as titles,  
 35 user comments, date and time of creation, the user who  
 created the object, etc. For more information on  
 properties, see Appendix D.

40 HPOMF.PRP grows dynamically as need. At the beginning  
 of HPOMF.PRP there is a 128 byte bitmap which controls the  
 45 allocation of the first 1024 pages of HPOMF.PRP. Each page  
 is 32 bytes in length. These pages immediately follow the  
 bit map. The bitmap is an array of words with the most  
 50 significant bit of each word used first. Thus, bits 15  
 through 0 of the first word of the bitmap control the  
 55 allocation of pages 0 through 15 of the file, respectively.

5           When storage in the first 1024 pages is insufficient, a  
second bitmap is added to the file following page 1023.

10           This bitmap controls the allocation of pages 1024 through  
2047, which immediately follow the second bitmap.

          Additional bitmaps and pages are added in the same way, as  
needed.

15           Each directory and property value is stored as a single  
block in the file, i.e., as a contiguous run of pages that  
20           are all allocated in the same bitmap. This causes the  
restriction that no directory or value can exceed 32K bytes  
(1024 times 32) in length.

25           System file 605, HPOMF.INS, is also referred to as  
SYSINSTL. HPOMF.INS contains a list of the files that were  
30           copied to the system when each class was installed. This  
information is used so that these files can be deleted when  
the class is de-installed.

35           The very beginning of HPOMF.INS is a double word value  
which serves as a validity/version identifier. In the  
40           preferred embodiment the value of this double word must be  
0101ABCD hex to be valid. In Table 8, this number is stored  
as shown because of the protocols for storage in the  
45           particular processor used by the preferred embodiment, i.e.  
an 80286 microprocessor made by Intel Corporation.

50           Following the double word comes a series of variable  
length records. There is one record for each installed  
class. The first word of each record is the length of the  
55           rest of the record, in bytes. This is followed by the null-

terminated class name of the installed class. Then follows  
 5 the file names of the files copied to the OMF directories,  
 each terminated by a null byte, and preceded by a byte which  
 gives the length of the file name, including the length byte  
 10 and the null terminator. If the file name begins with the  
 special character "\*", the file is assumed to be located in  
 15 the HPNWPROG directory. If the file name begins with the  
 special character "+" the file is assumed to be located in  
 the HPNWDATA directory.

20 For example, assume two classes are installed: class  
 "AB" and class "CDE". Class "AB" caused two files to be  
 25 installed: "Z" to HPNWPROG directory 668 and "YY" to the  
 HPNWDATA directory. Class "CDE" caused 1 file to be  
 installed: "XXX" to HPNWPROG directory 668. Given this  
 30 case Table 8 below shows the contents of HPOMF.INS for this  
 example:

35 Table 8

offset	content	comments
0	CD AB 01 01	File header/version check
4	0C 00	Length of AB record (12 decimal)
6	41 42 00	"AB" + Null
9	04	Length of length byte "*"Z" + Null
A	2A 5A 00	"*Z" + Null
45 D	05	Length of length byte + "+YY" + Null
E	2B 59 59 00	"YY" + Null
12	0A 00	Length of CDE record (10 decimal)
50 14	43 44 45 00	"CDE" + Null
18	06	Length of length byte + "*XXX" + Null
19	2A 58 58 58 00	"*XXX" + Null

55



5           System File 606, HPOMF.SDF is also referred to as the  
"shutdown file". HPOMF.SDF exists only when the system has  
been cleanly shut down. It is deleted as the system starts,  
10 and created as it shuts down. On startup, if this file is  
missing, OMF assumes that the last session ended abnormally,  
and so it goes through its crash recovery procedures to  
15 validate and repair the system files as best it can. The  
system files can be in an invalid but predictable state on a  
crash. These errors are corrected without user  
20 intervention. Certain other kinds of file consistency  
errors are detected, but are not really possible from an  
25 "ordinary" system crash. These errors are in general not  
correctable and the OMF will not allow the system to come up  
in this case.  
30

          If HPOMF.SDF is present, it contains a list of objects.  
When the system is being shut down normally, each object  
35 which is active at the time can request that the OMF restart  
them when the system is restarted. The list of objects,  
then is the list of tags of objects which have requested  
40 that they be restarted when the system is restarted.

          The first word in HPOMF.SDF is a flag word. If this  
45 word is non-zero, OMF will execute its crash recovery code  
even though HPOMF.SDF exists. Normal shutdown will set this  
flag when producing the file if some serious error occurred  
50 in the session being ended.

          After the first word, the rest of the file is a  
55 sequence of three byte records. The first two bytes of each

5 record contain the tag of the object to be restored. The  
 least significant byte is first. The third byte is not used  
 in the preferred embodiment, and is zero.

10 For example, if the system is shut down cleanly in the  
 last session and two objects, having tags of 2 and 7,  
 respectively, have requested restart, the contents of  
 15 HPOMF.SDF will be as set out in Table 9 below.

Table 9

offset	content	comments
0	00 00	Indicates no crash recovery needed
2	02 00	Tag of first object to restart
4	00	Unused and reserved
5	07 00	Tag of second object to restart
7	00	Unused and reserved

30 System file 7, HPOMFICO.NWE, is a Microsoft Windows  
 dynamic library executable file which contains a dummy entry  
 point and no data. Microsoft Windows is a program sold by  
 35 Microsoft Corporation, having a business address at 16011 NE  
 36th Way, Redmond, WA 98073-9717. HPOMFICO.NWE also  
 contains as "resources" the icons of each installed class.  
 40 OMF modifies HPOMFICO.NWE directly during run time, and  
 loads and unloads it to get the icon resources from it. The  
 format of HPOMFICO.NWE is defined in Microsoft Windows  
 45 documentation distributed by Microsoft Corporation.

50 Normally working with a view (see discussion on views  
 above) causes a child's application to be invoked. Where  
 large applications are involved, this can cause a lot of

55

unnecessary overhead. The use of snapshots allow this overhead to be eliminated.

5 A snapshot is an object that uses executable code from  
a separate library referred to as a dynamic access library  
(or DAL) rather than using the full application executable  
10 code. The only data file associated with a snapshot  
contains data which is to be sent from a child object to a  
15 parent object. The code which encapsulates the data file  
although referred to as a dynamic library, is still stored  
in directory HPOMFPROG (directory 668).

20 For example, Figure 81 shows a parent object 501 linked  
to a child object 502 through a link 504. Associated with  
25 link 504 is a snapshot 503. Once child object has designated  
snapshot 503 in a view specification record for link 504,  
snapshot 503 is able to provide data from child object 502  
30 to parent 501 without the necessity of invoking an  
application associated with child object 502.

35 As shown in Figure 82, when there is no snapshot, child  
object 502 must be active in order to send view data 522 to  
parent object 501, in order for parent object 501 to display  
40 view data 522 in a window display 521. In Figure 83,  
however, snapshot 503 is shown to provide view data 522 to  
parent object 501 without the necessity of child 502 being  
45 active. Further implementation details of snapshots are  
given in Appendix B, Appendix C and Appendix D.

50 Appendix A is a list of major data structures within  
OMF 100.

55

Appendix B is a description of functions which OMF  
interface 599 recognizes in the preferred embodiment of the  
5 present invention.

Appendix C (HP NewWave Environment: Program Design  
Examples) Gives examples of how the preferred embodiment of  
10 the present invention may be implemented, including detail as  
to how OMF 100 allows data to be viewed between windows  
15 displayed on monitor 14.

Appendix D (Chapter 2 of Programmer's Guide) gives a  
further overview of the preferred embodiment of the present  
20 invention. further detail as to the operation of the  
preferred embodiment of the present invention.

25

30

35

40

45

50

55

5  
10  
15  
20  
25  
30  
35  
40  
45  
50  
55

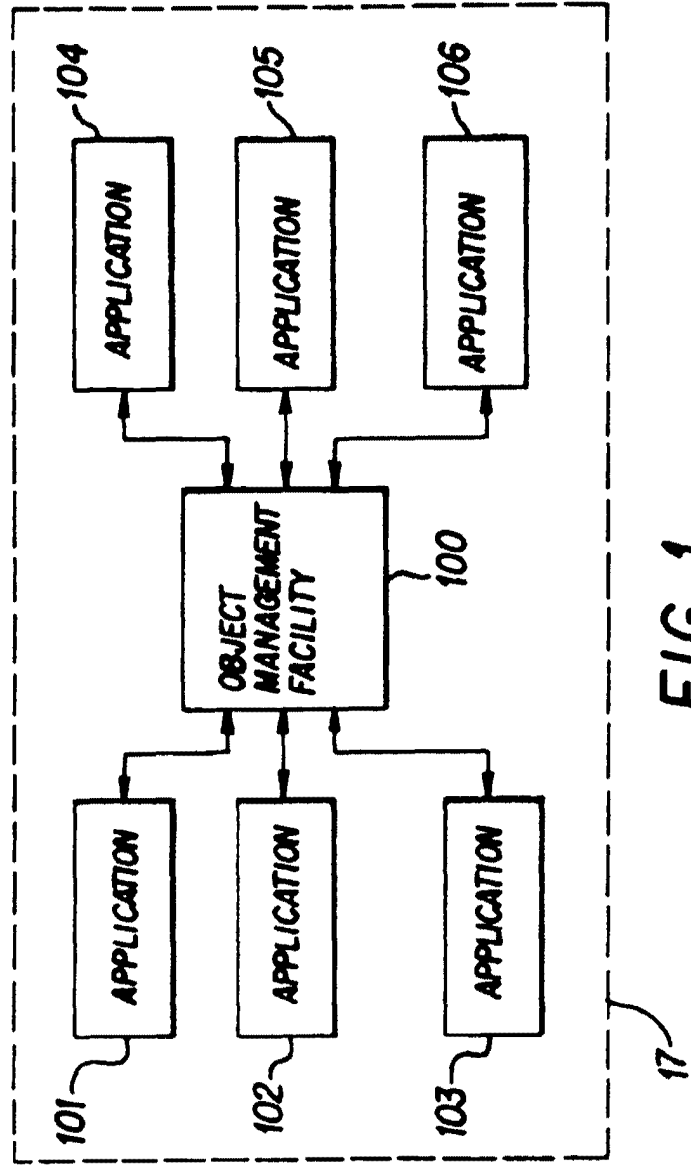
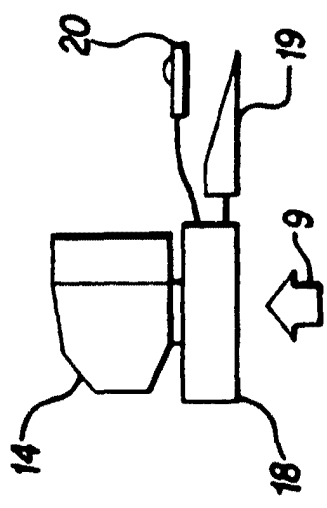


FIG. 1

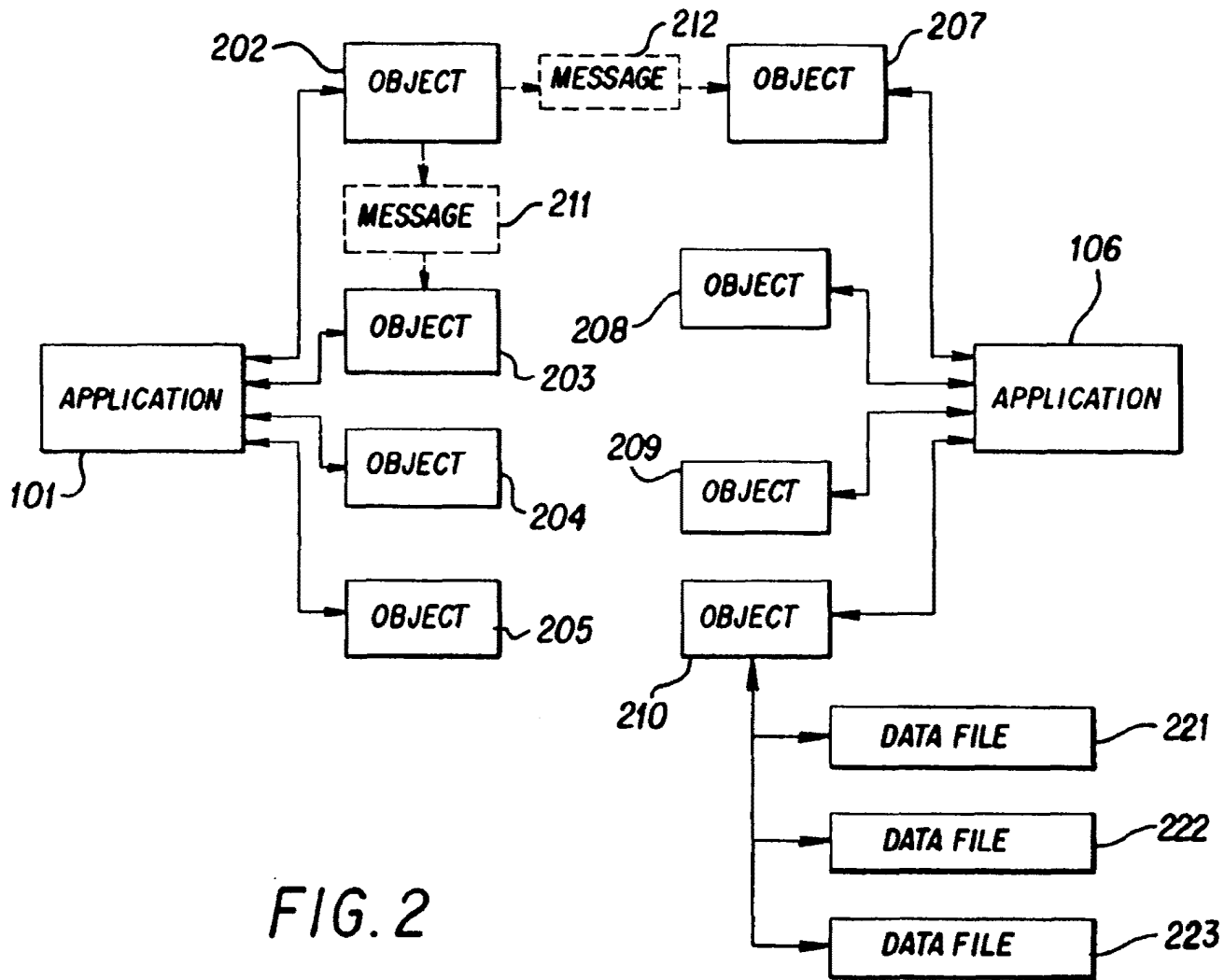


FIG. 2

5  
10  
15  
20  
25  
30  
35  
40  
45  
50  
55

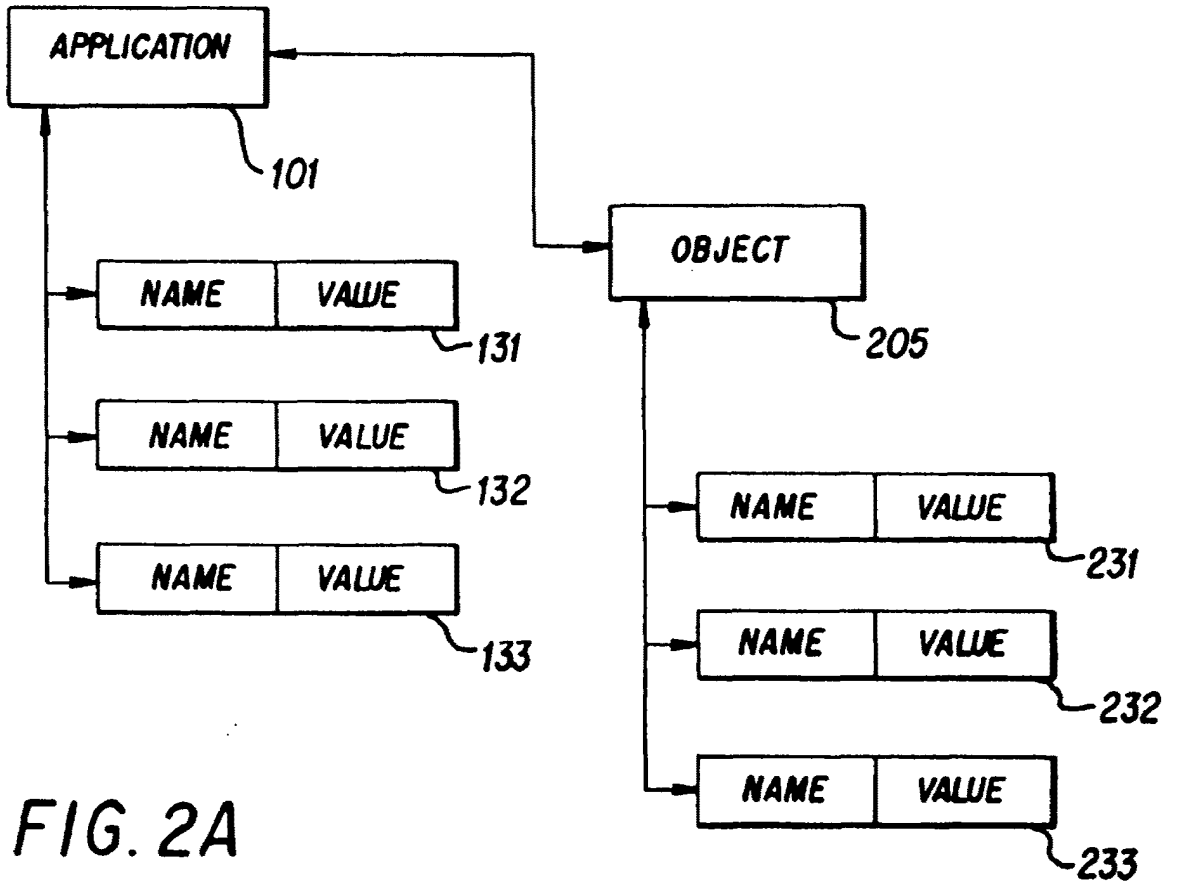


FIG. 2A

56

EP 0 497 022 A1

55 50 45 40 35 30 25 20 15 10 5

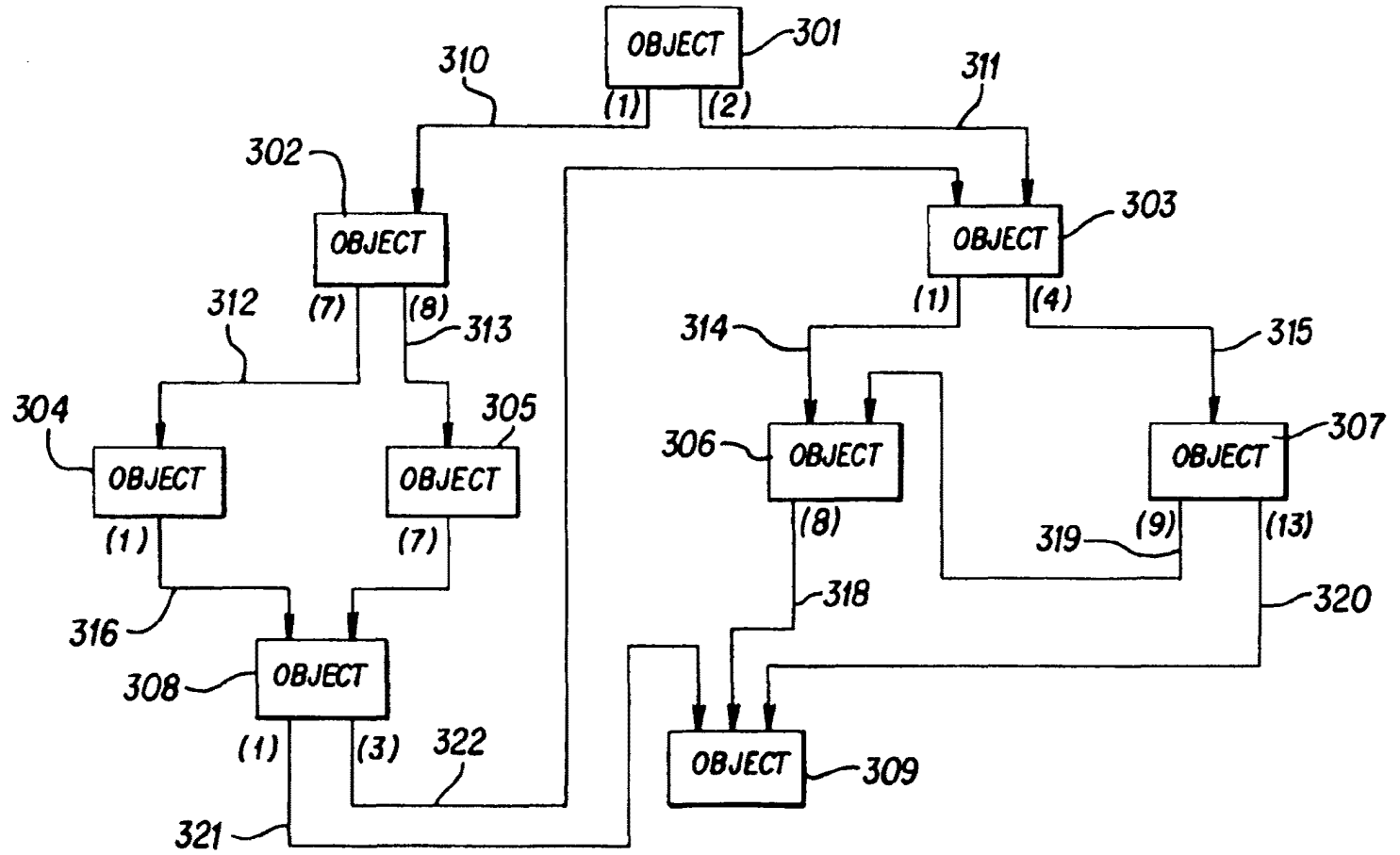


FIG. 3

57

EP 0 497 022 A1



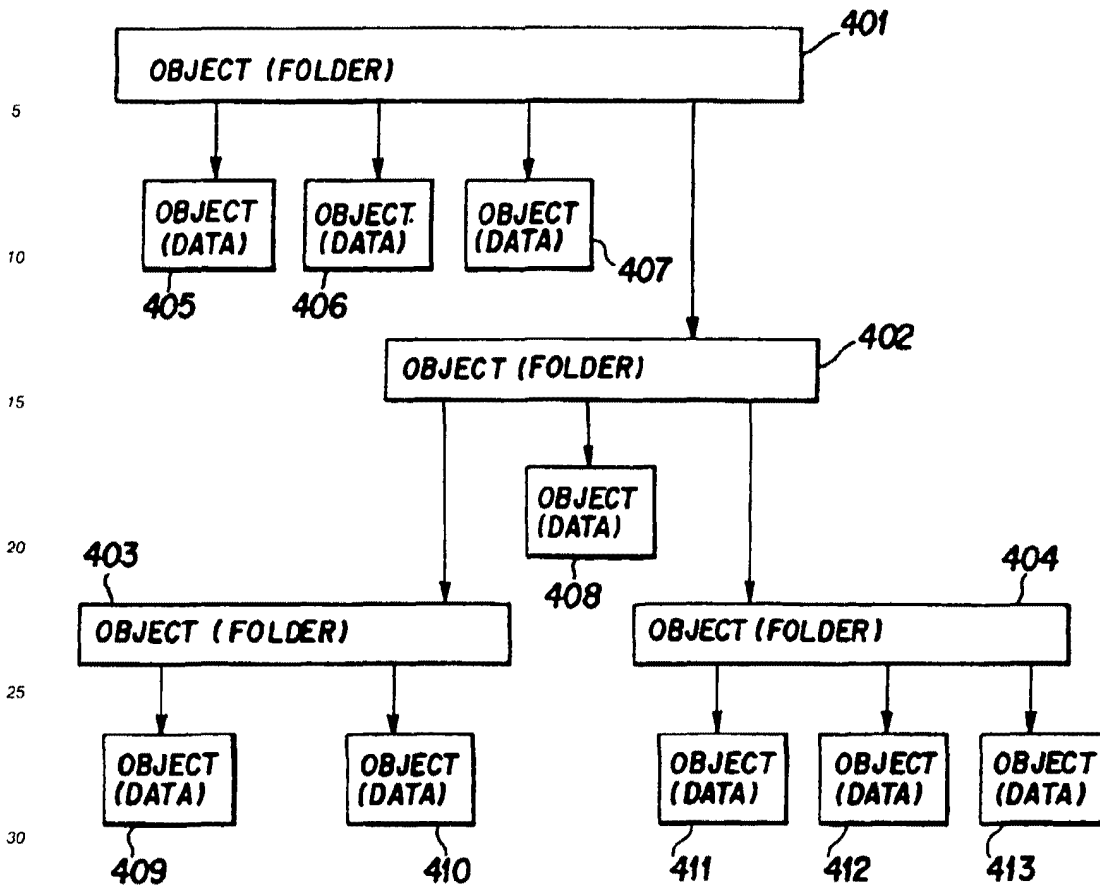
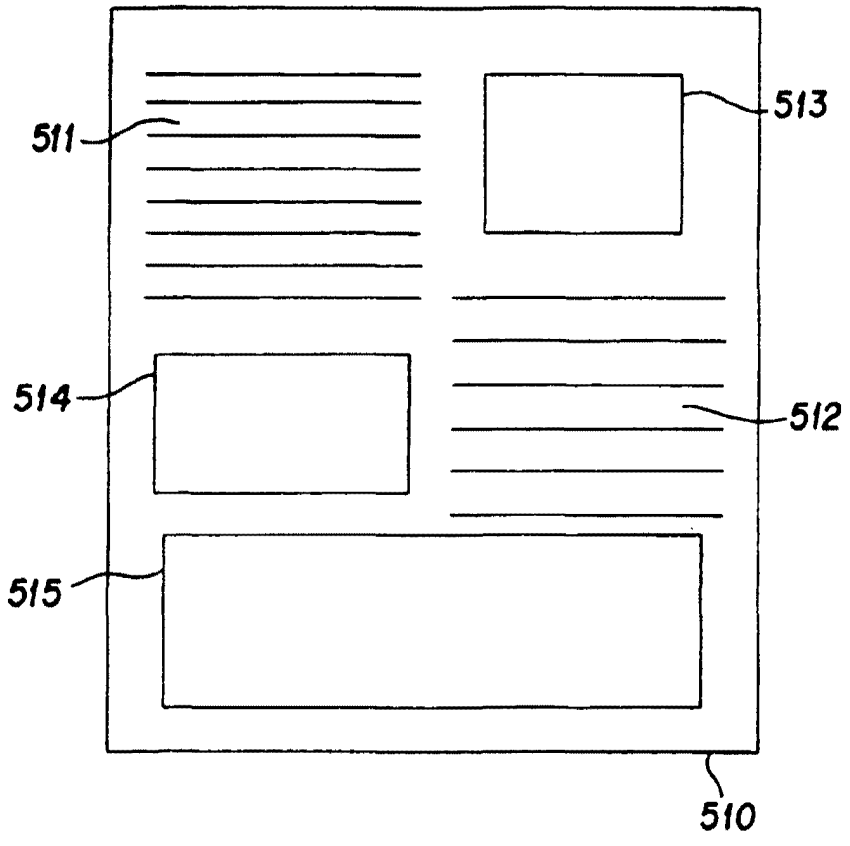
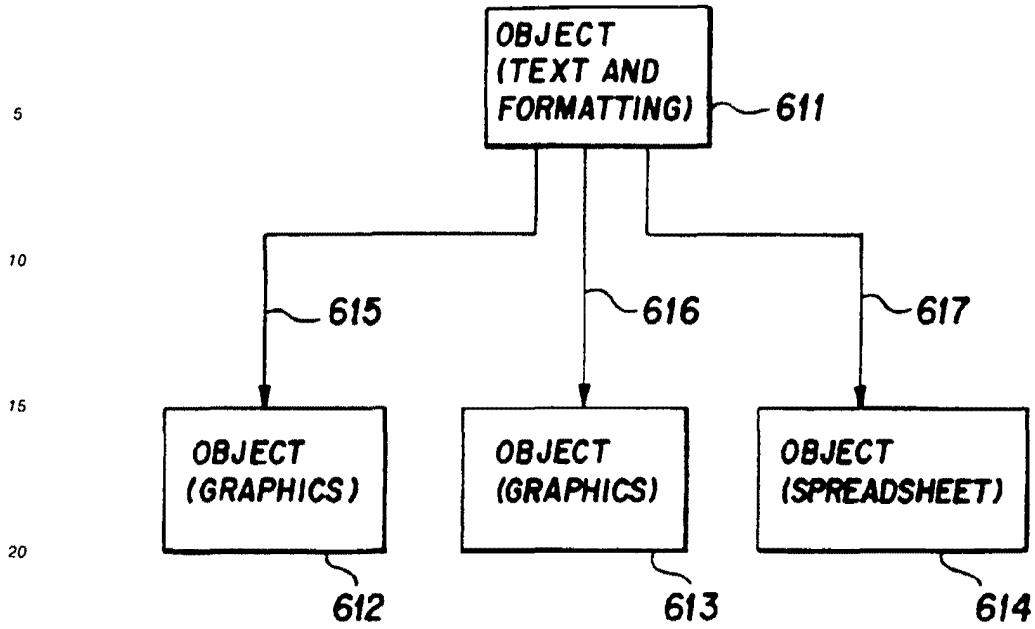


FIG. 4

FIG. 5

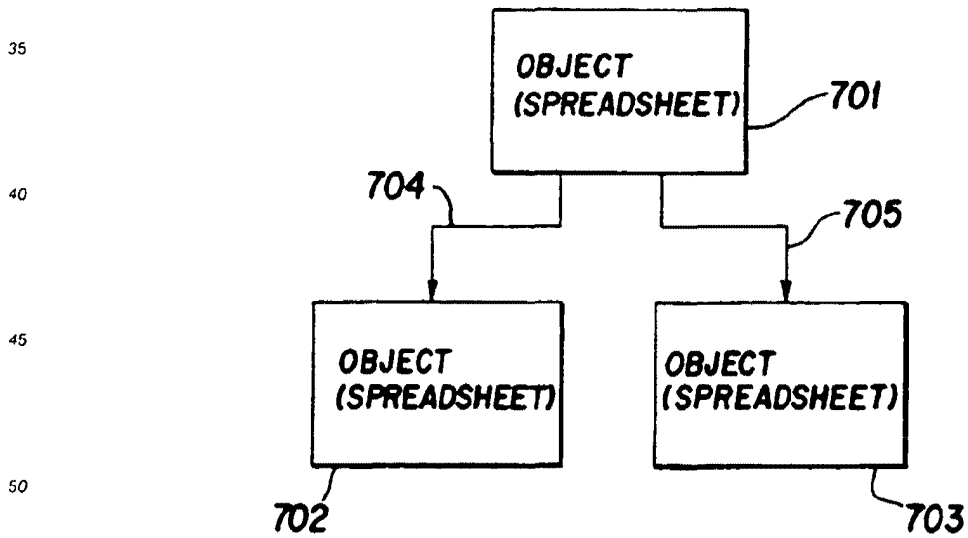
5  
10  
15  
20  
25  
30  
35  
40  
45  
50  
55





30

**FIG. 6**



**FIG. 7**

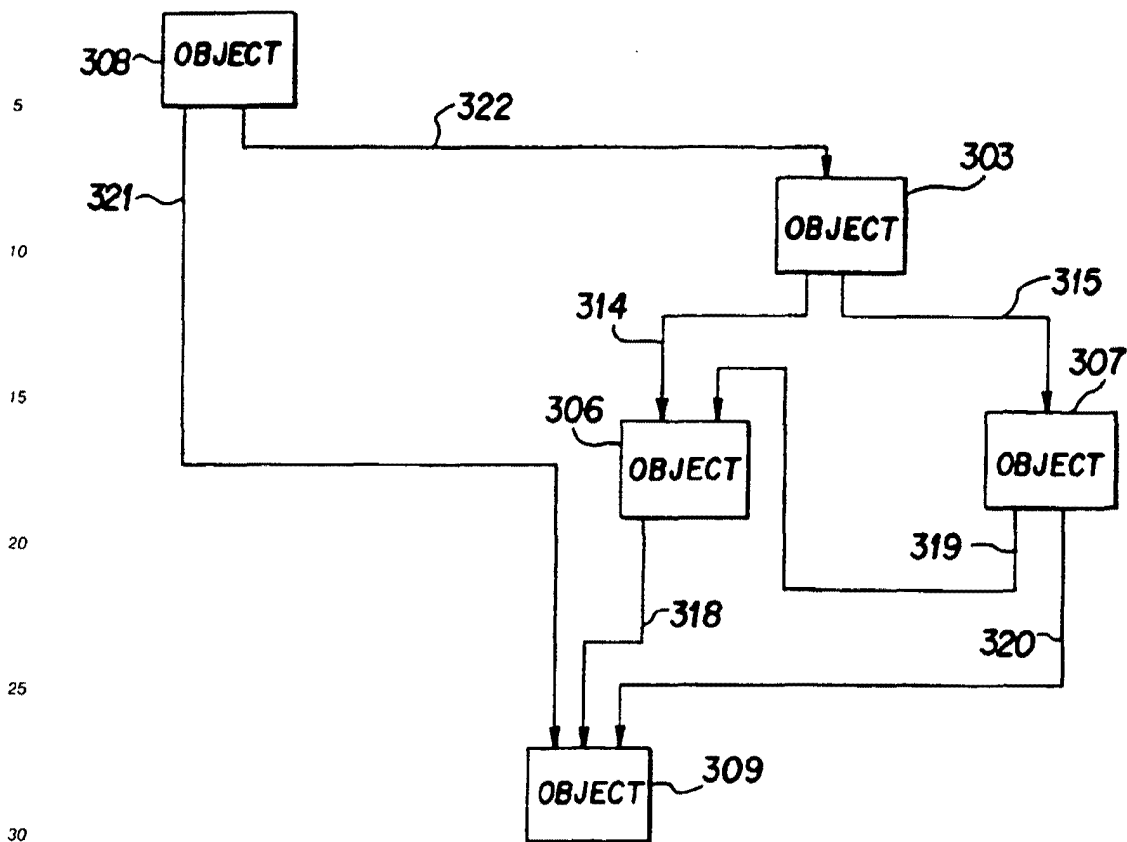


FIG. 8

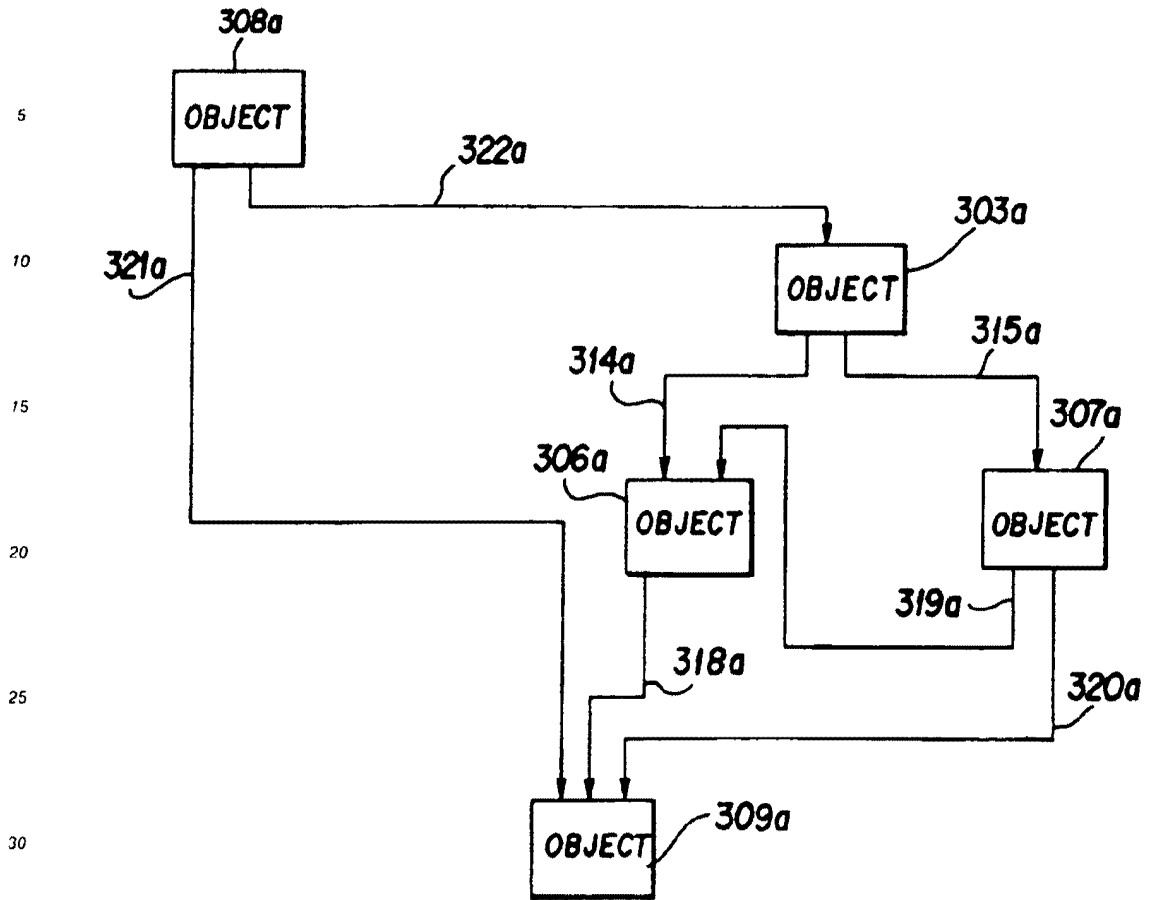
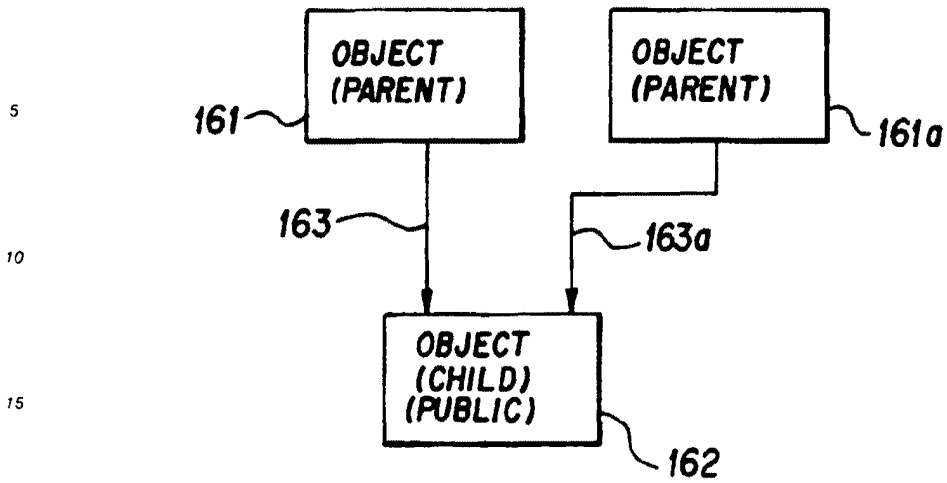


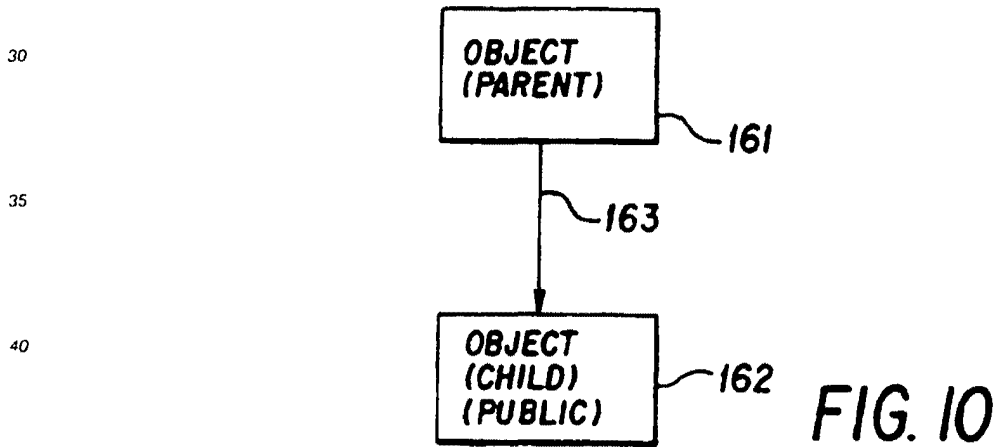
FIG. 9



20

25

**FIG. 11**



45

50

55

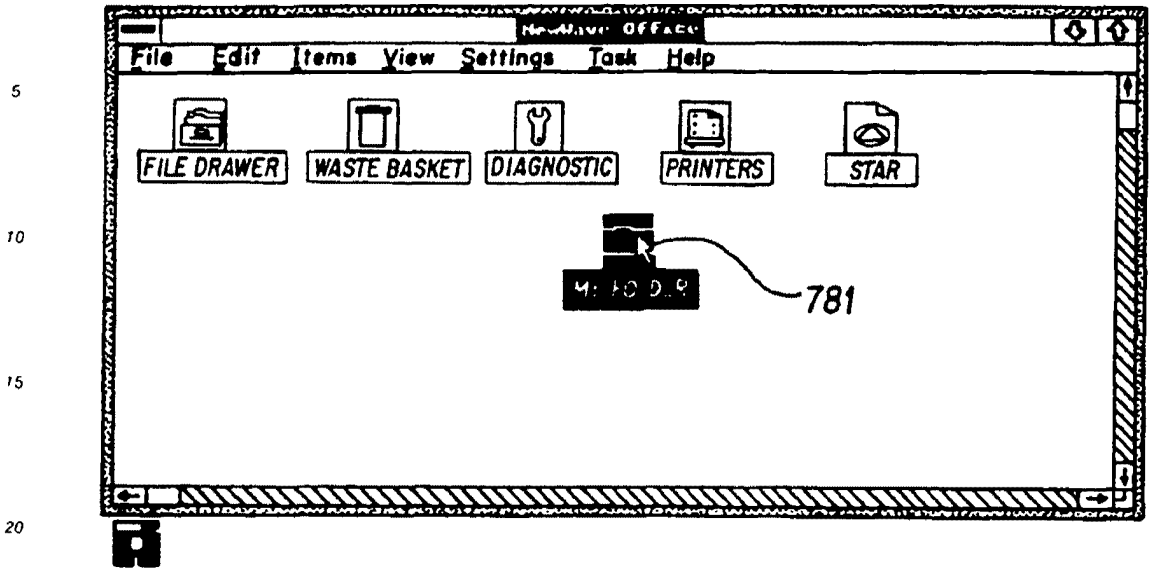


FIG. 12

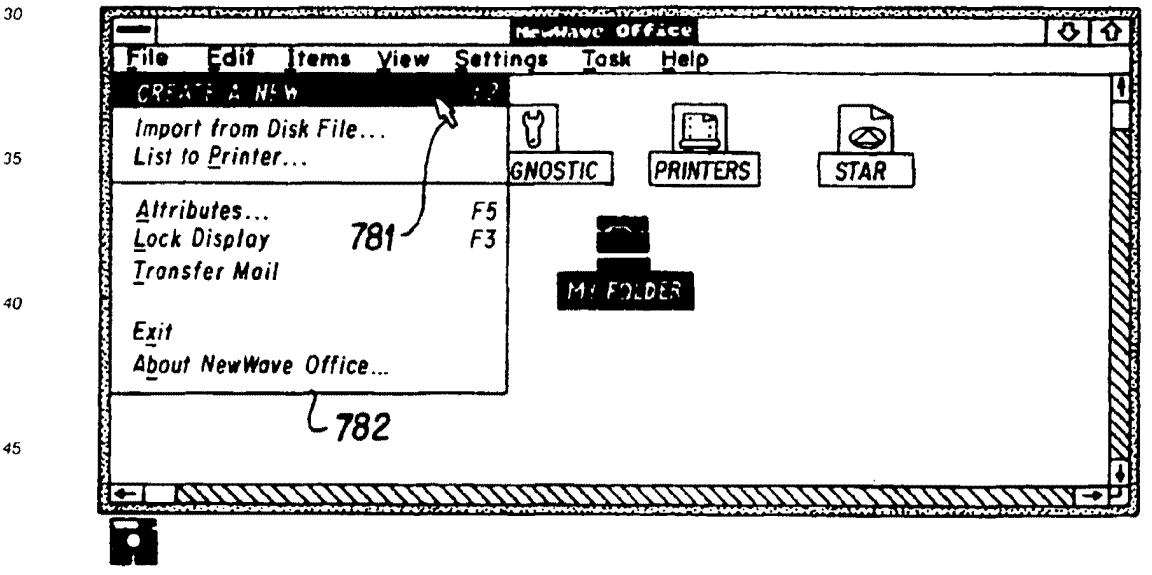


FIG. 14

50  
55

55 50 45 40 35 30 25 20 15 10 5

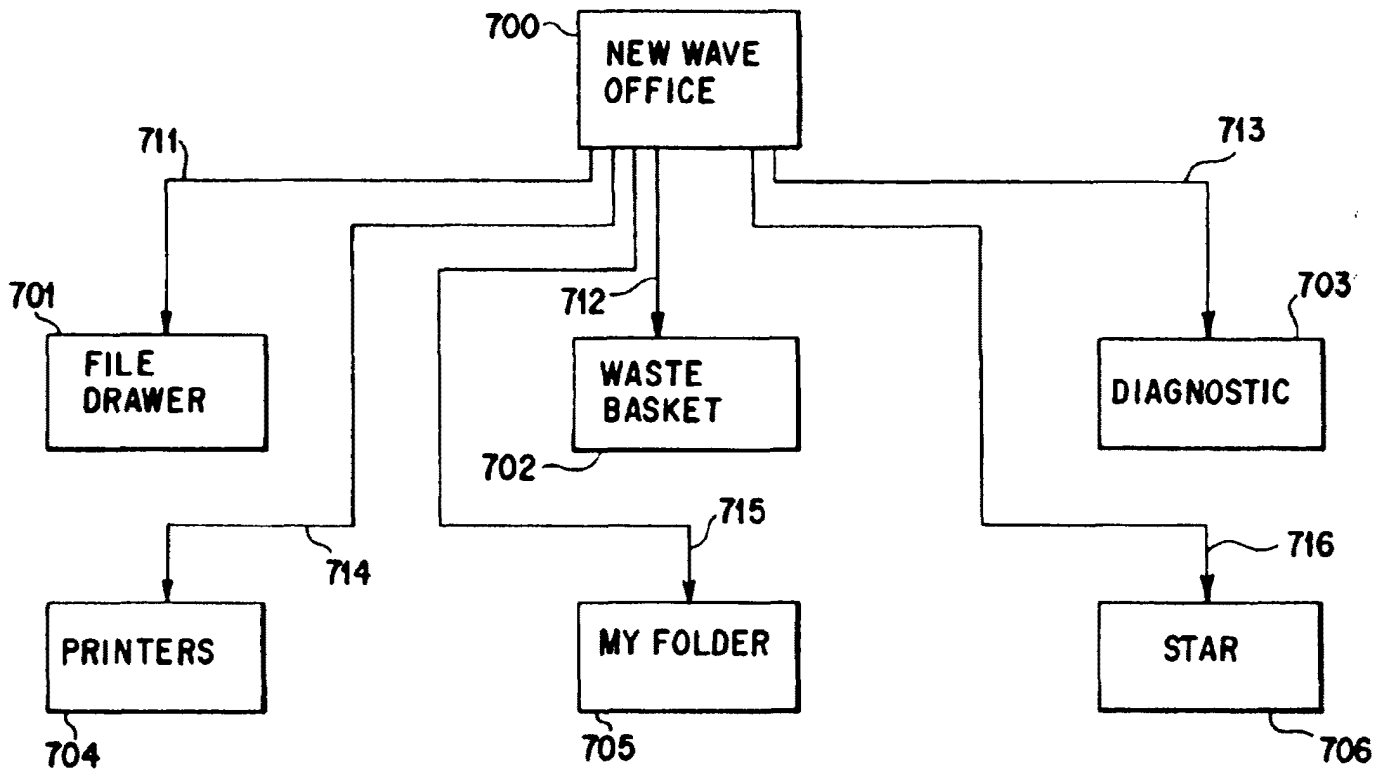


FIG. 13

65

EP 0 497 022 A1

SKYPE-N2P00283587



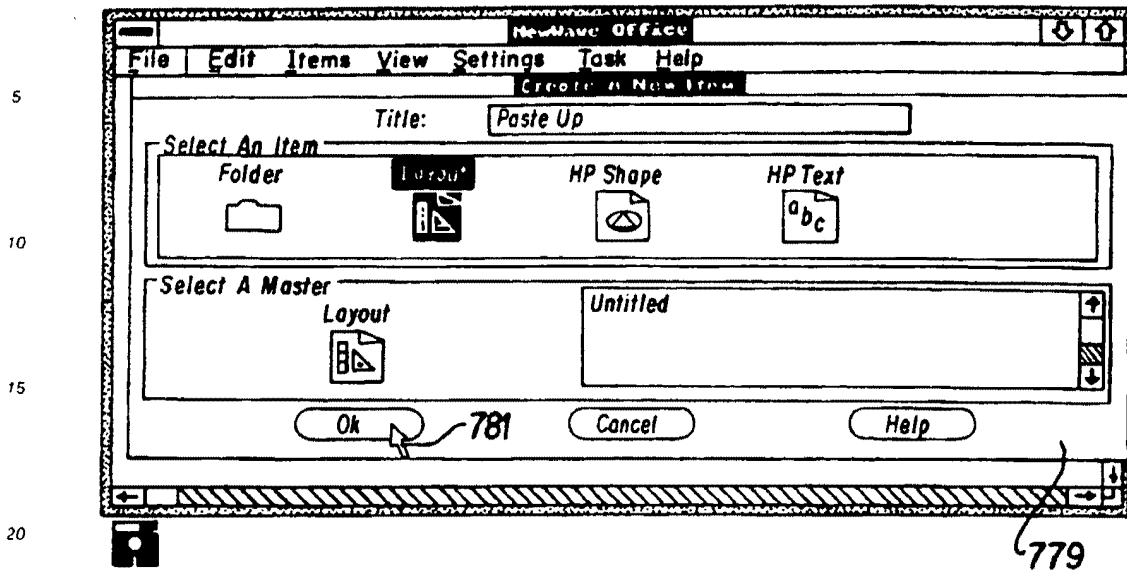


FIG. 15

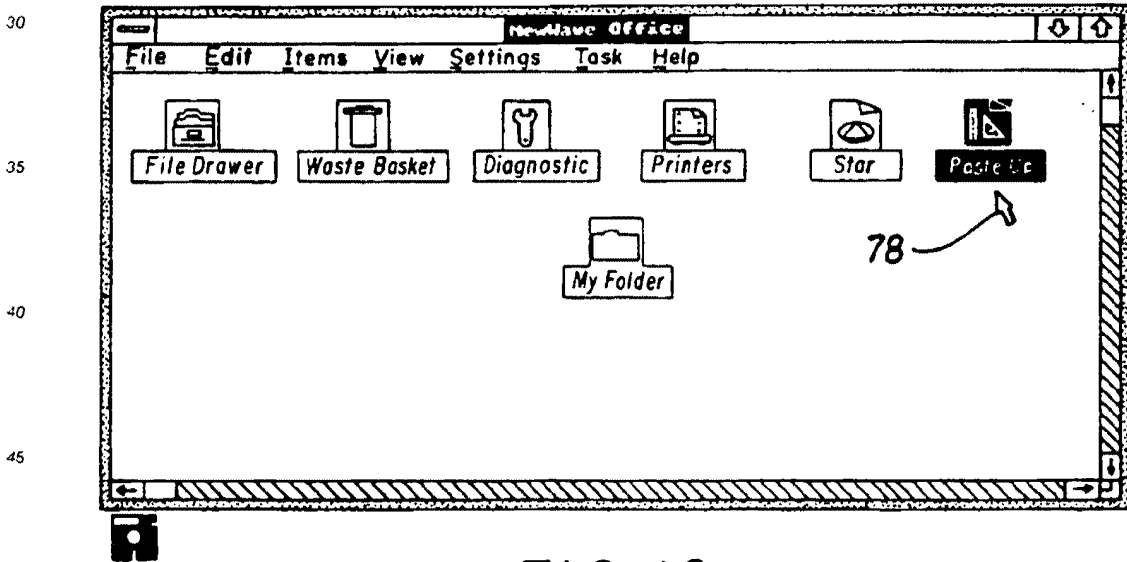


FIG. 16

55  
50  
45  
40  
35  
30  
25  
20  
15  
10  
5

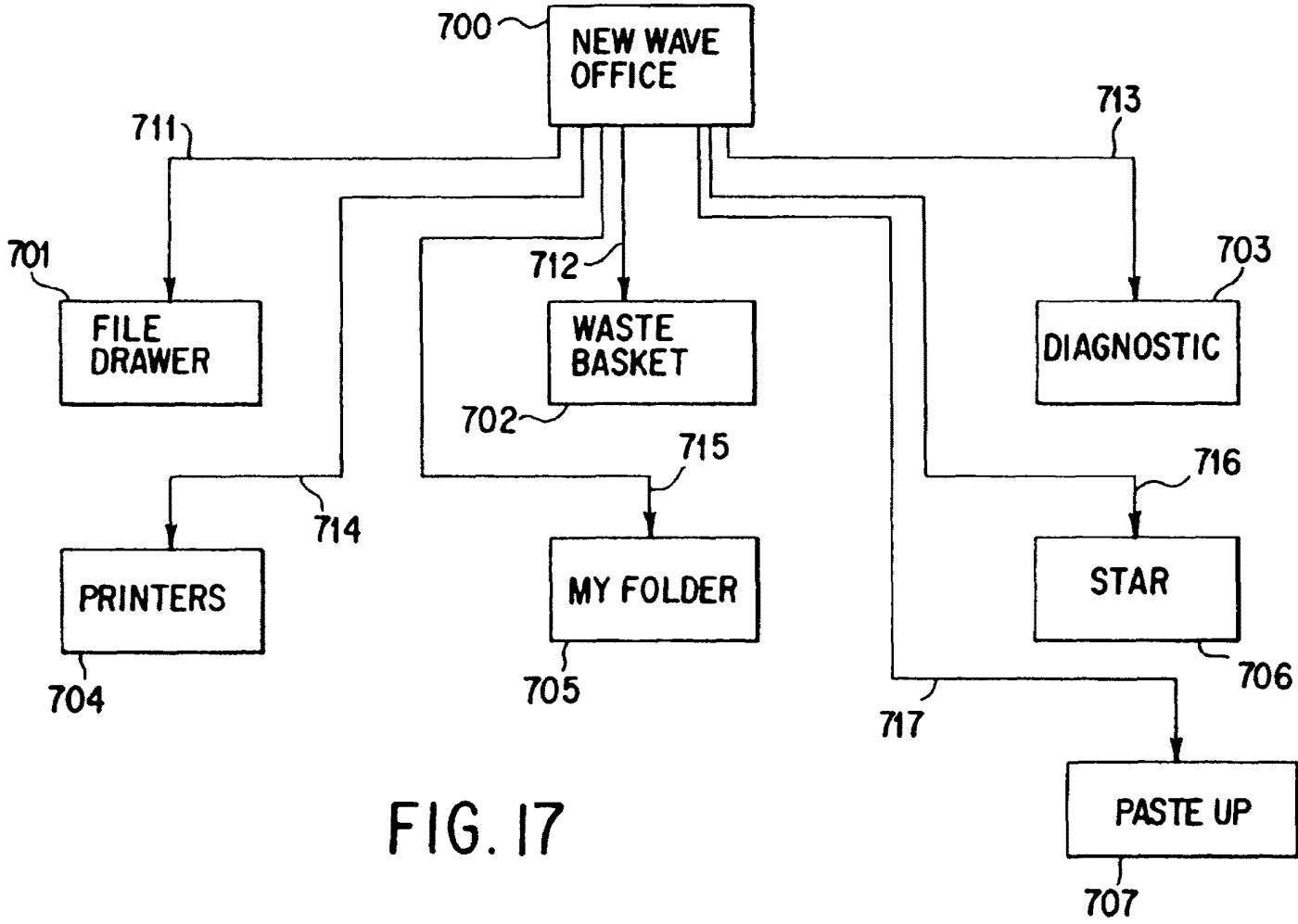


FIG. 17

EP 0 497 022 A1

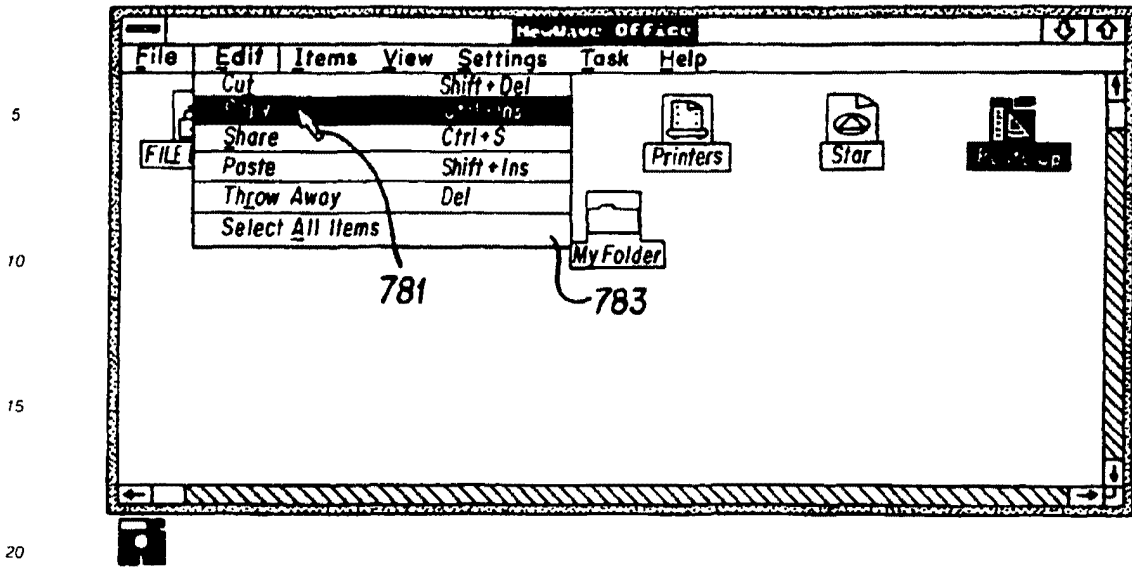


FIG. 18

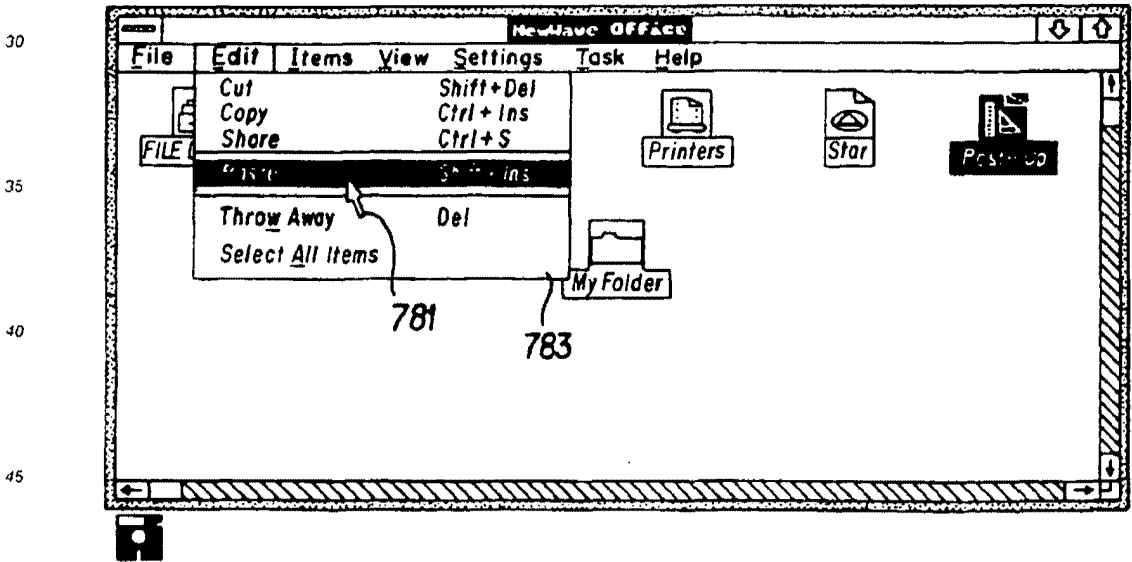


FIG. 19

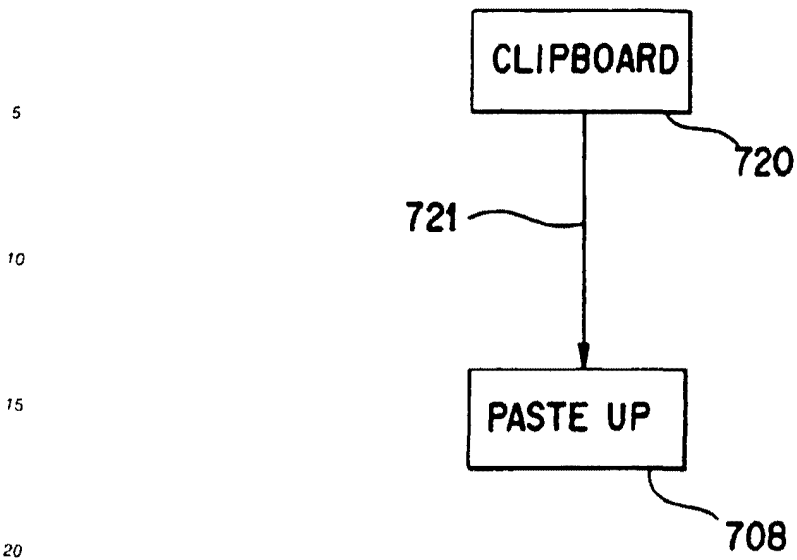


FIG. 18A

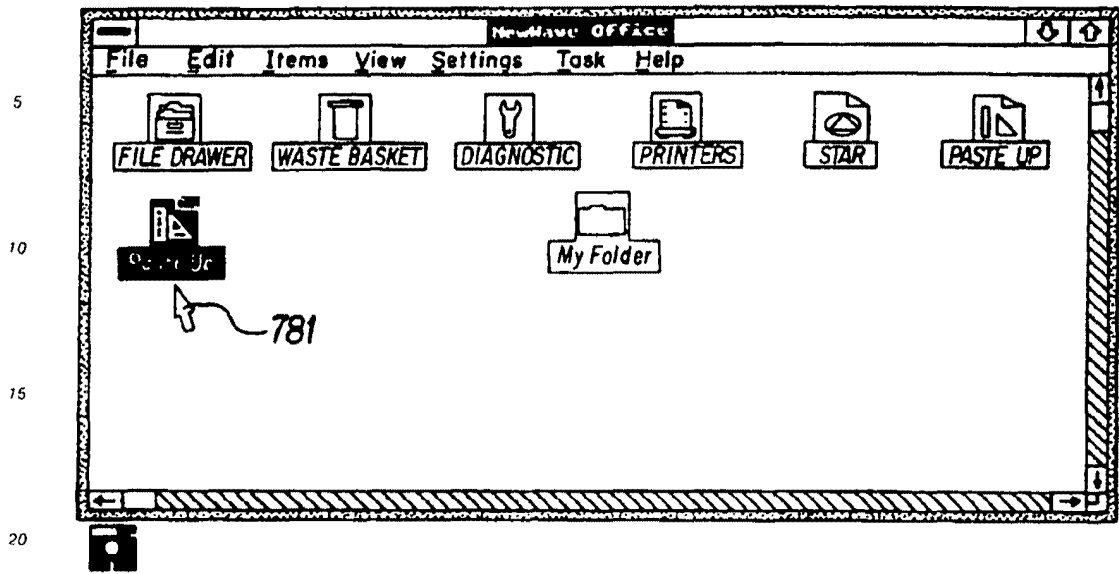


FIG. 20

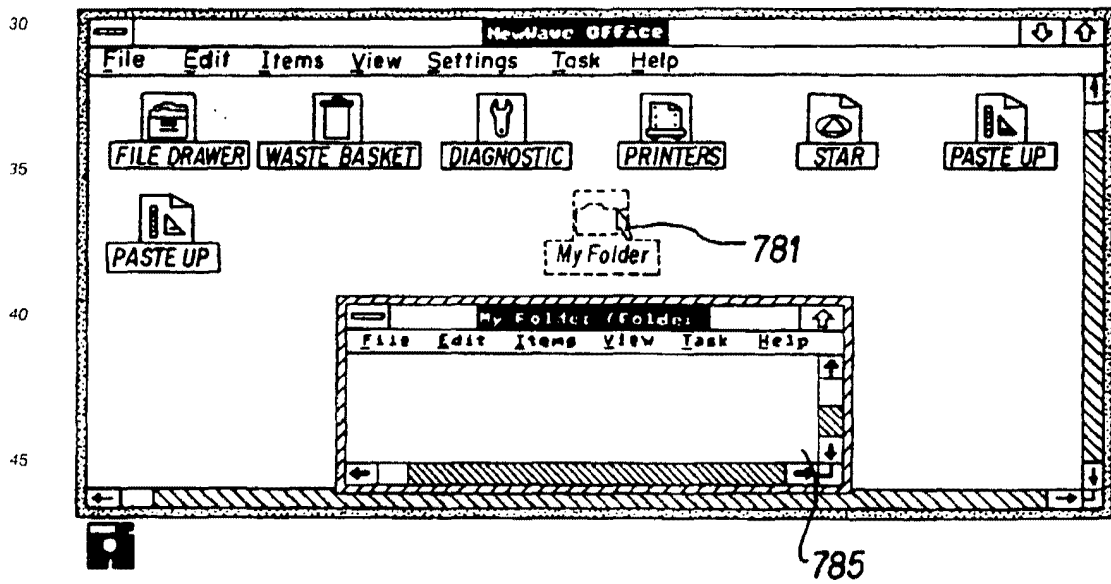


FIG. 22

55 50 45 40 35 30 25 20 15 10 5

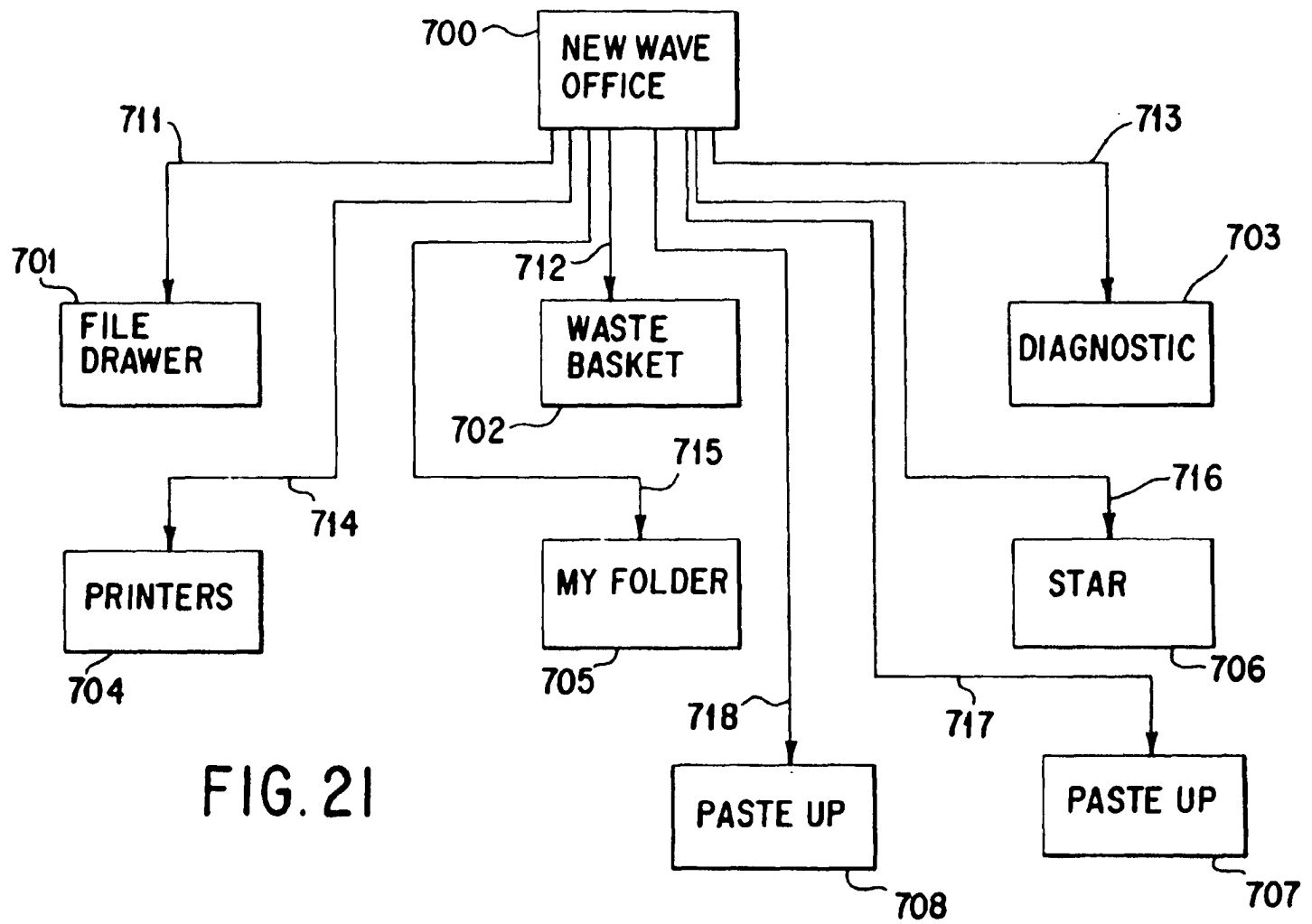


FIG. 21

71

EP 0 497 022 A1

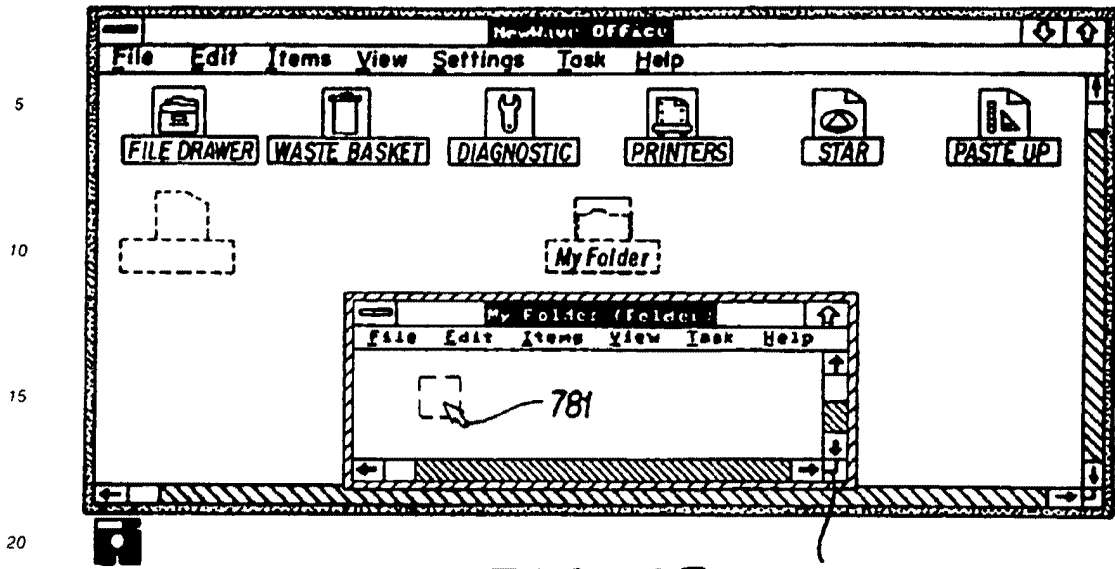


FIG. 23 785

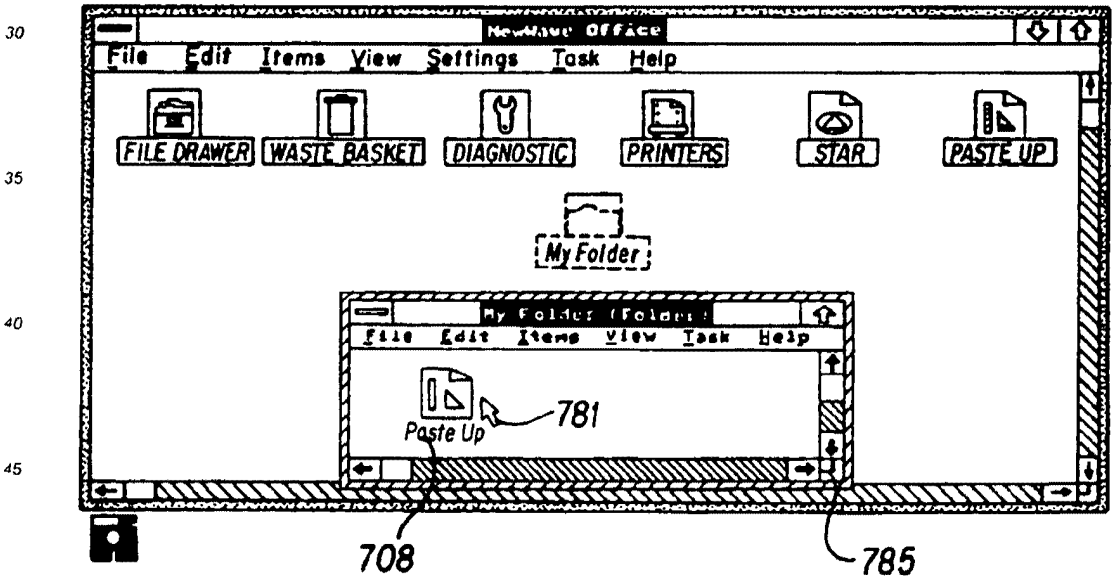


FIG. 24 708 785

55

50

45

40

35

30

25

20

15

10

5

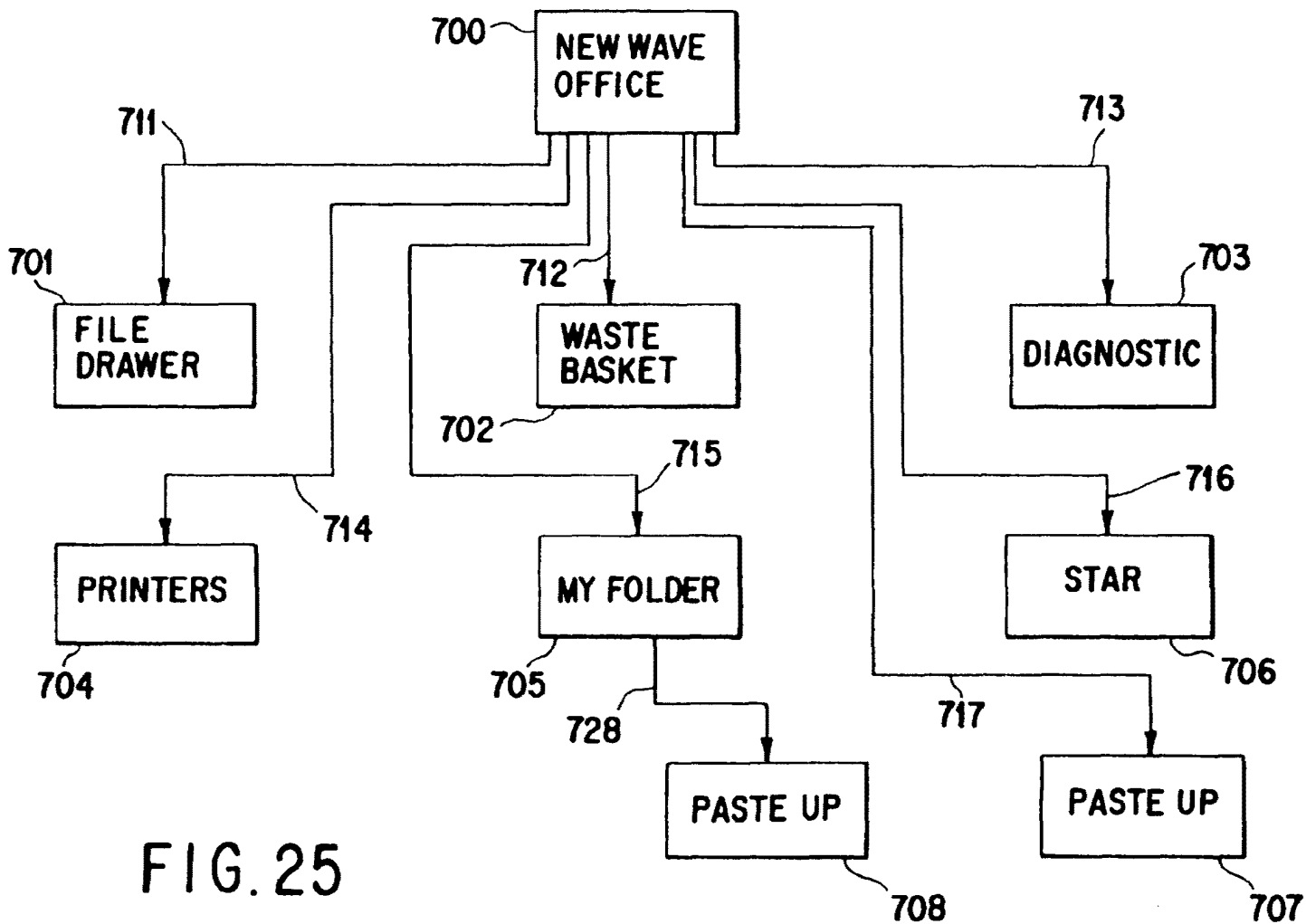


FIG. 25

73

EP 0 497 022 A1

SKYPE-N2P00283595



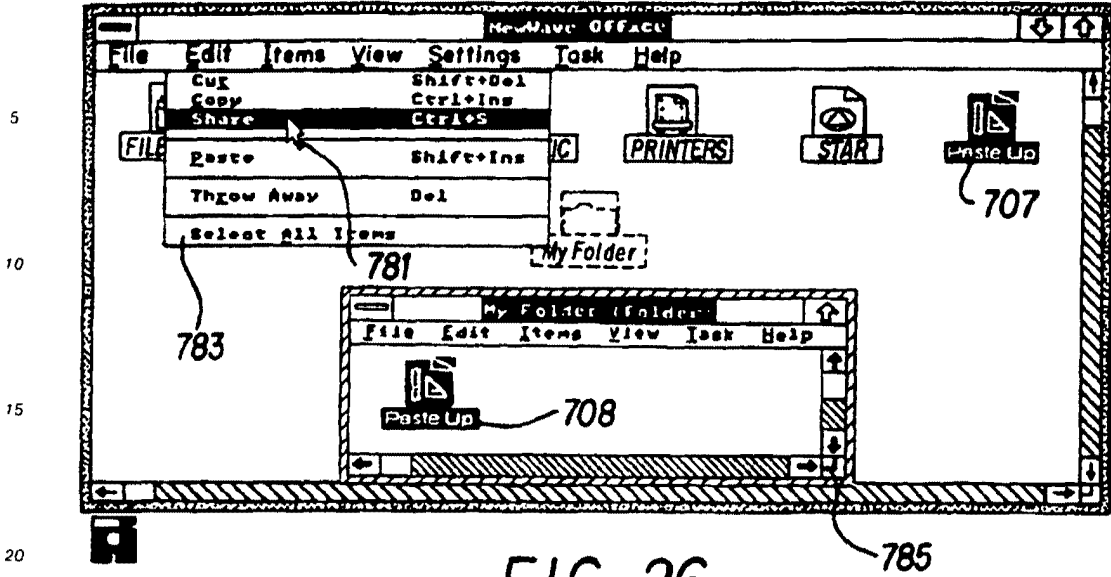


FIG. 26

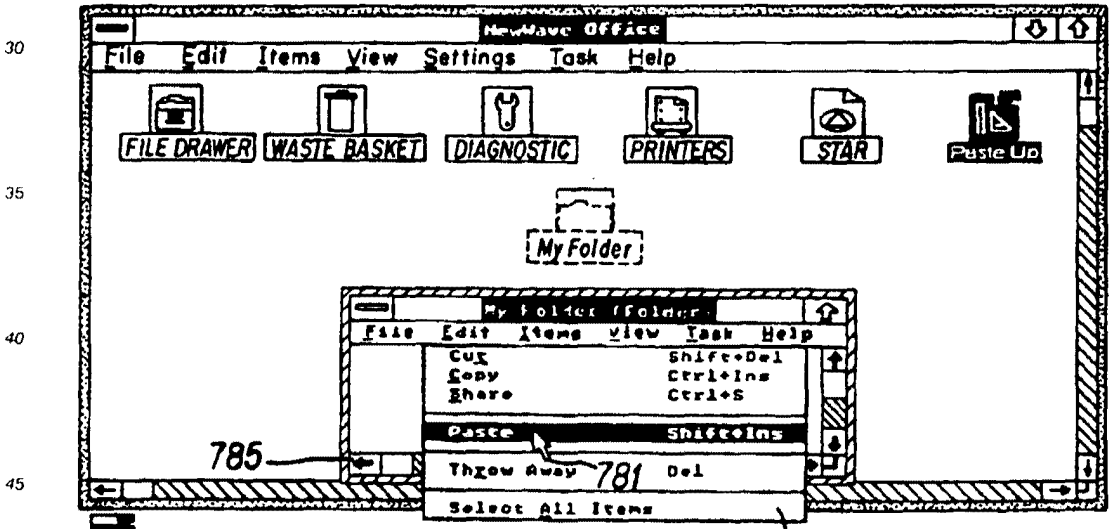
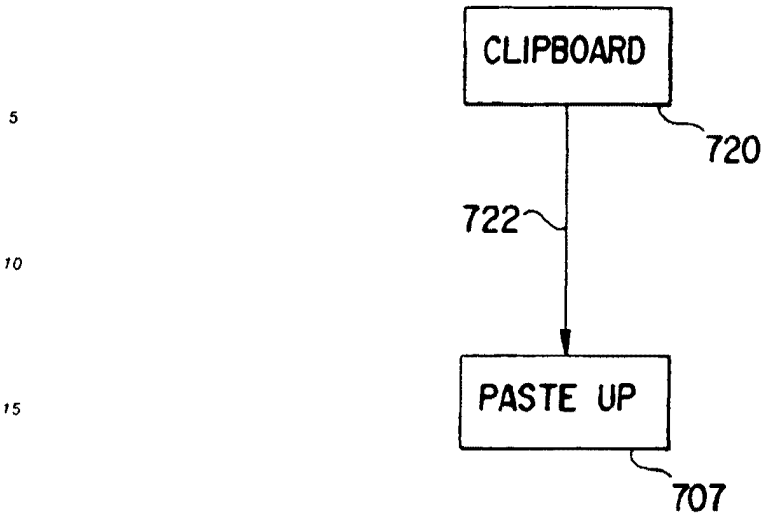


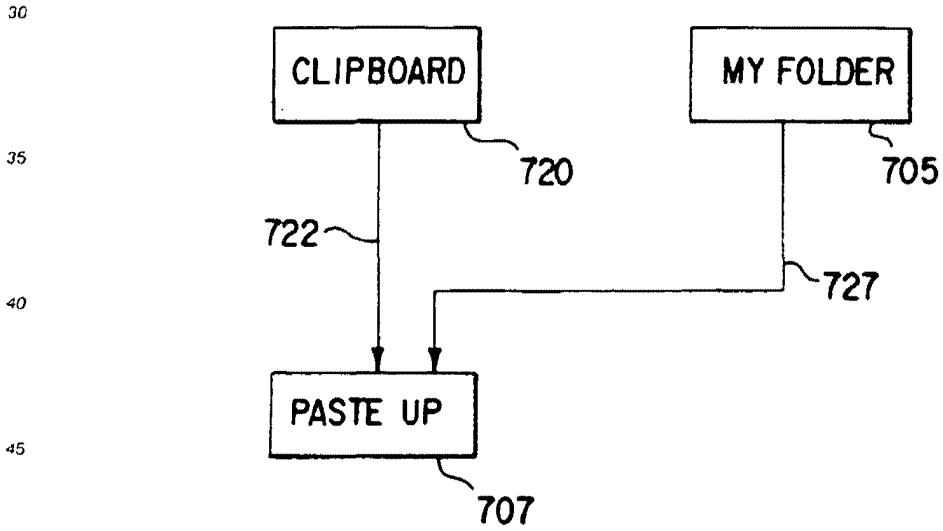
FIG. 27



20

25

FIG. 26A



50

55

FIG. 28A

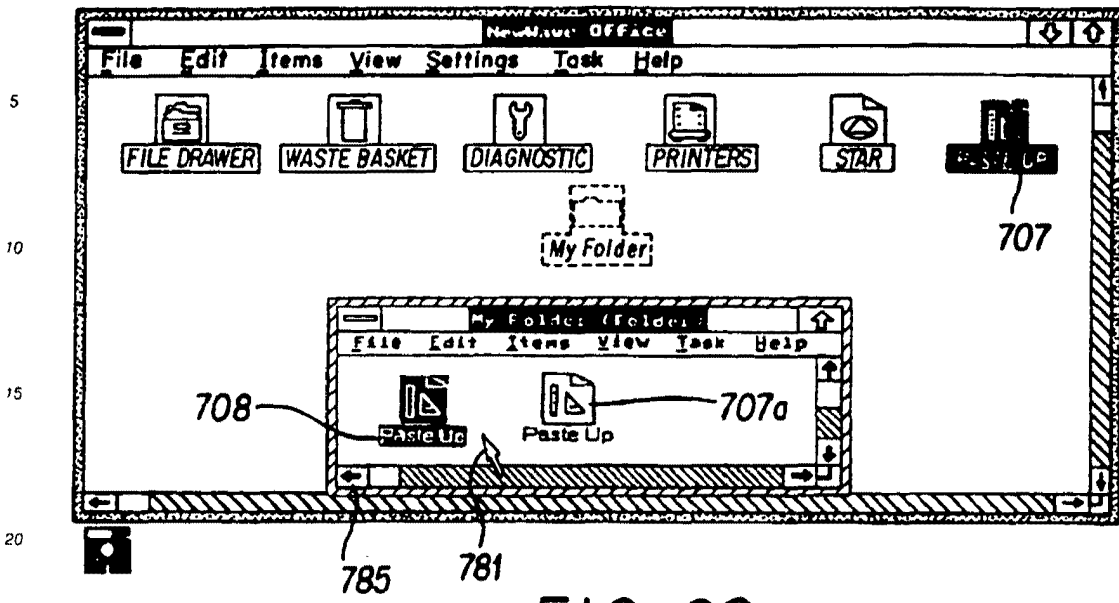


FIG. 28

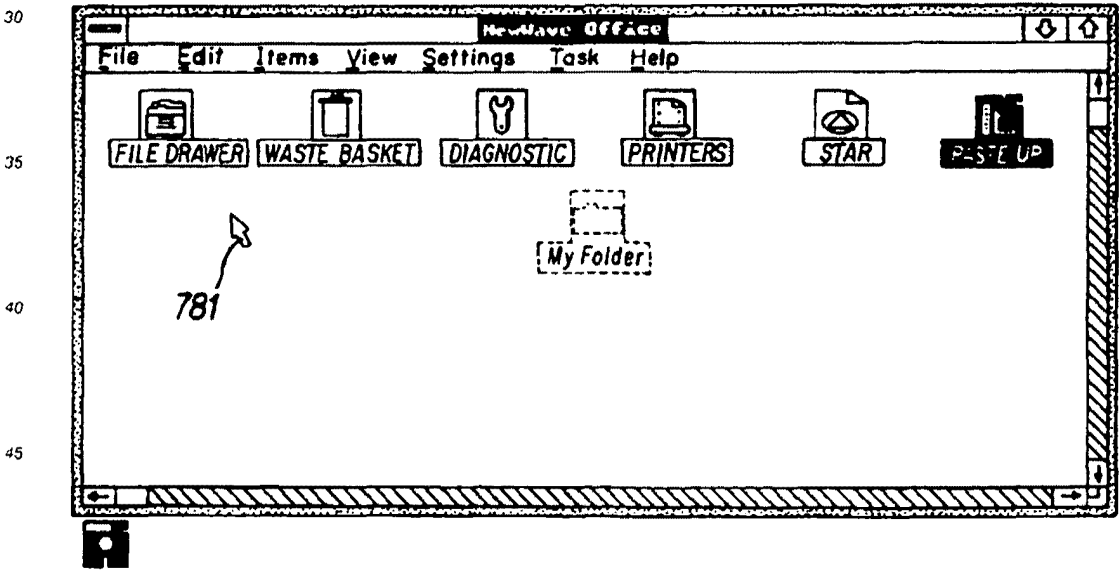


FIG. 30

55 50 45 40 35 30 25 20 15 10 5

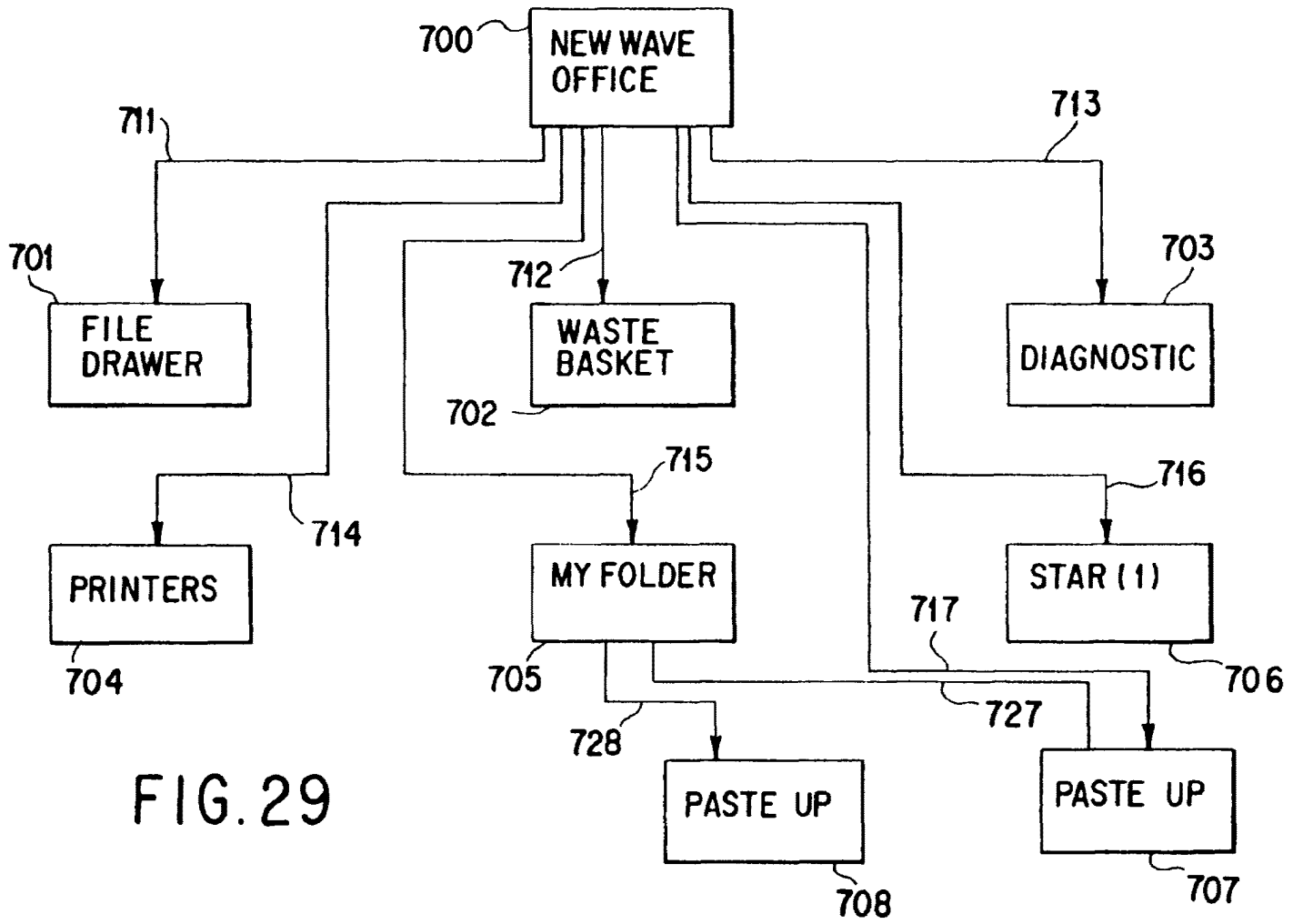


FIG. 29

77

EP 0 497 022 A1

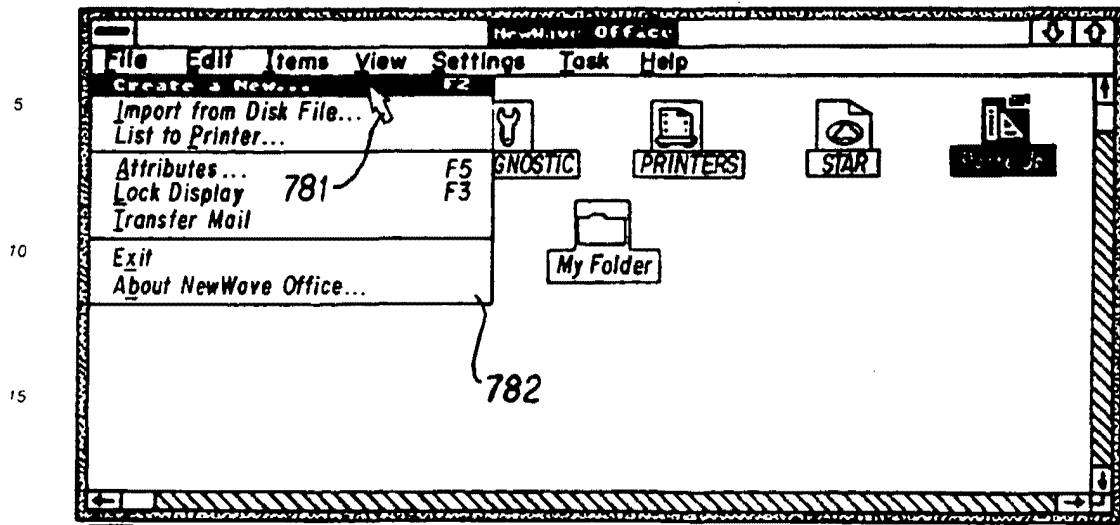


FIG. 31

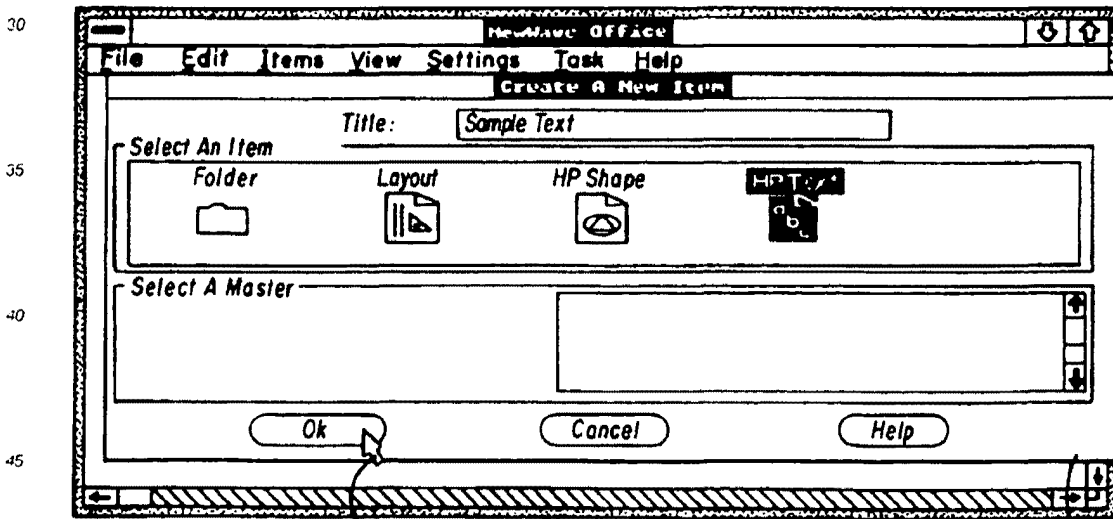


FIG. 32

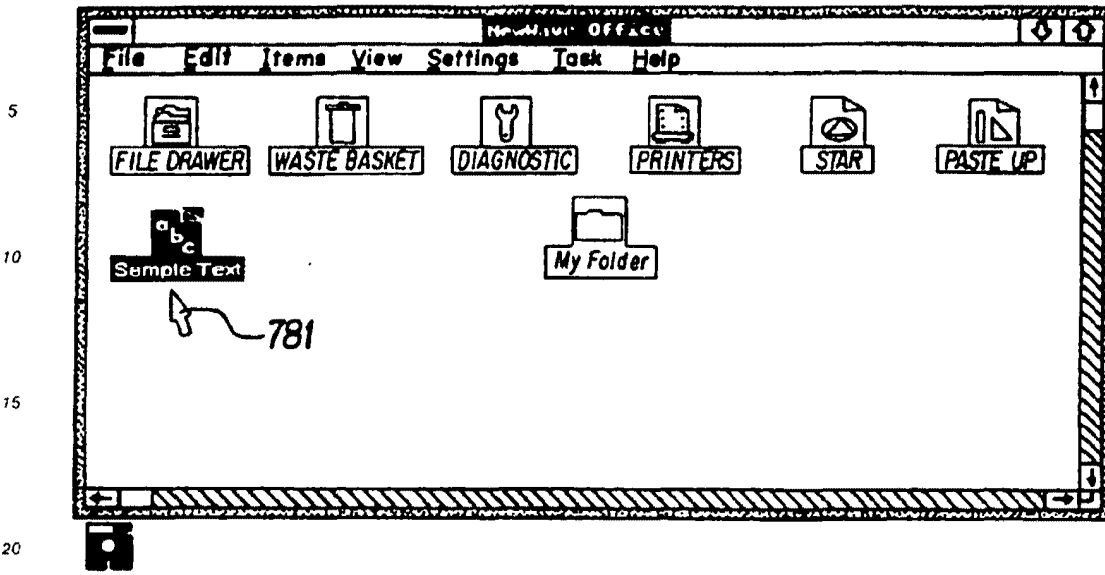


FIG. 33

25

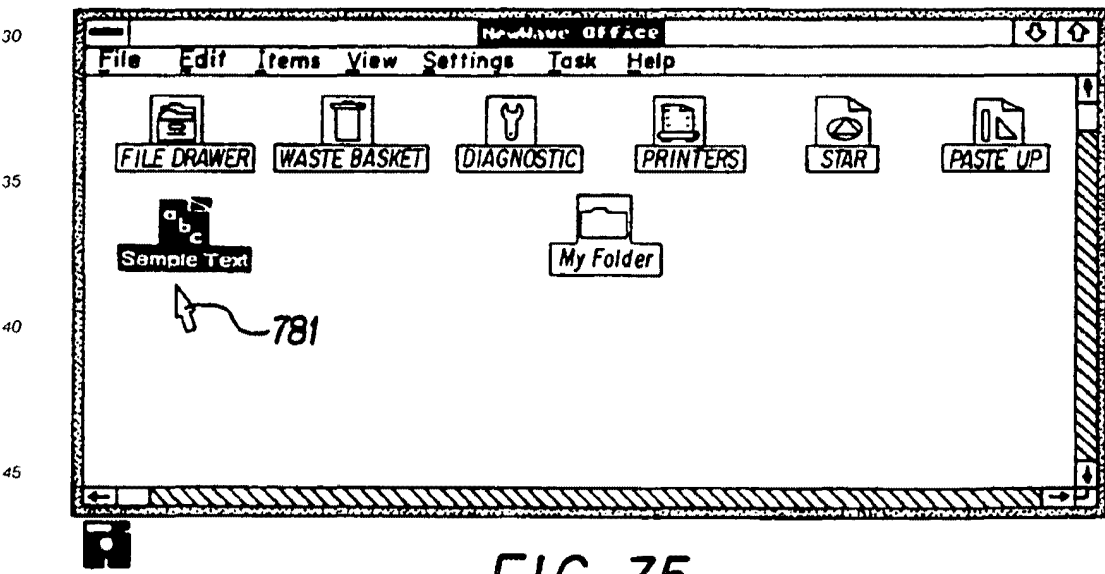


FIG. 35

50

55

55 50 45 40 35 30 25 20 15 10 5

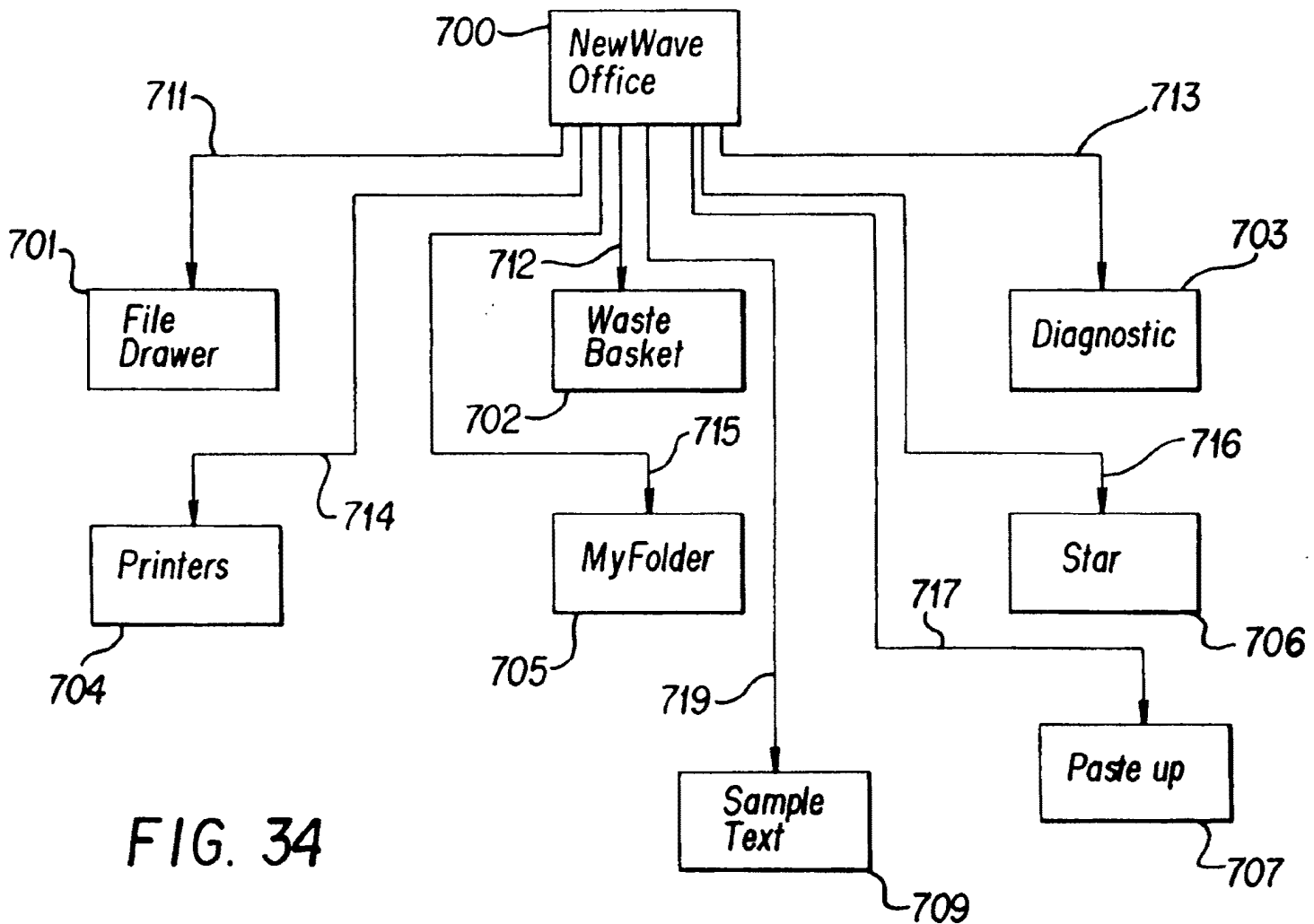


FIG. 34

80

EP 0 497 022 A 1

SKYPE-N2P00283602

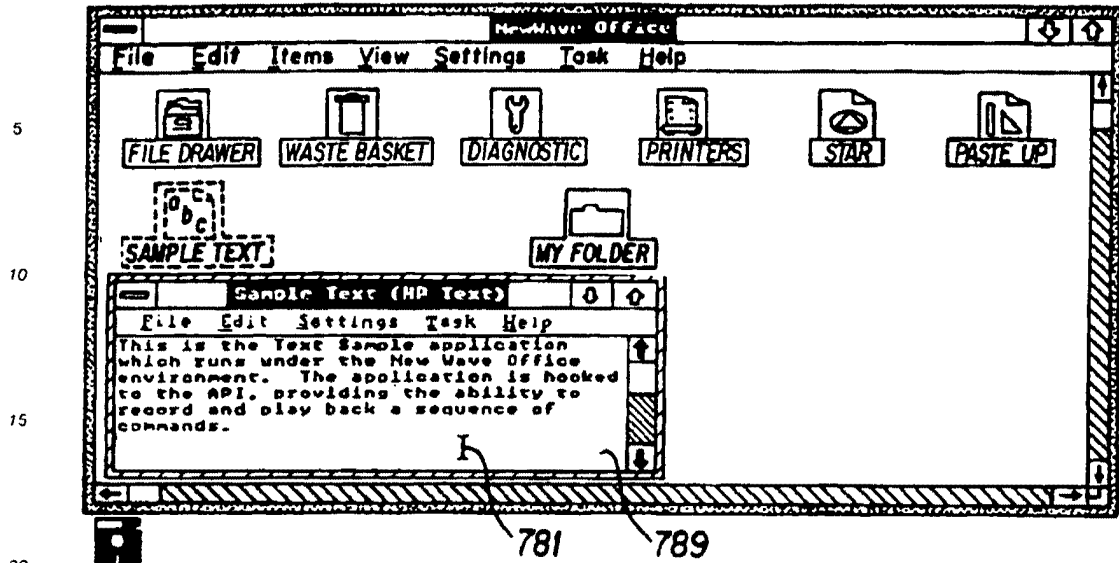


FIG. 36

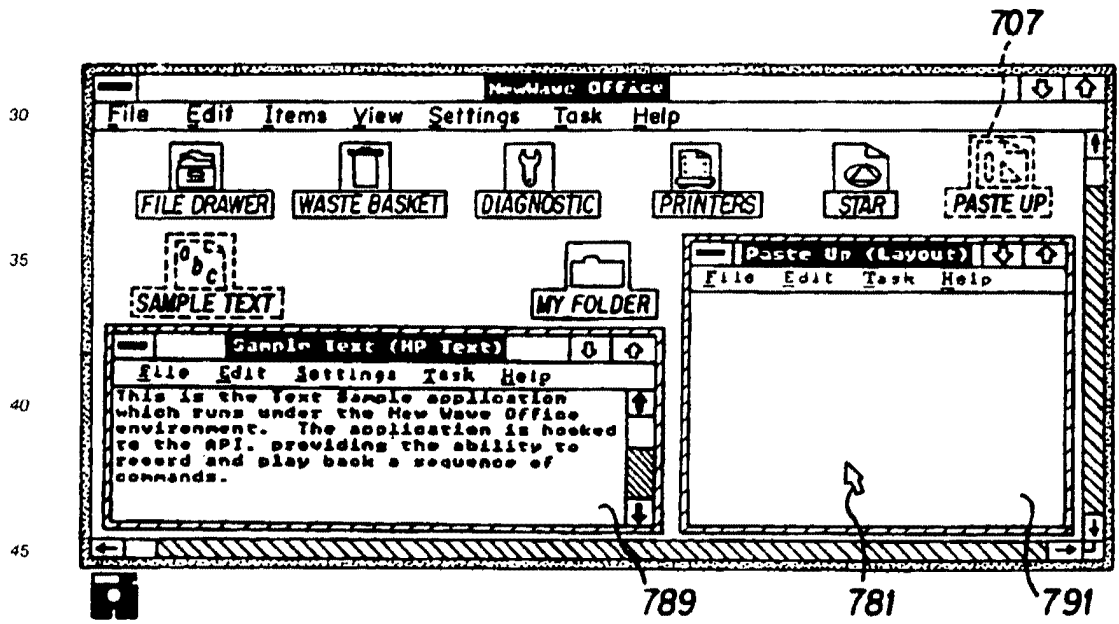


FIG. 37



707

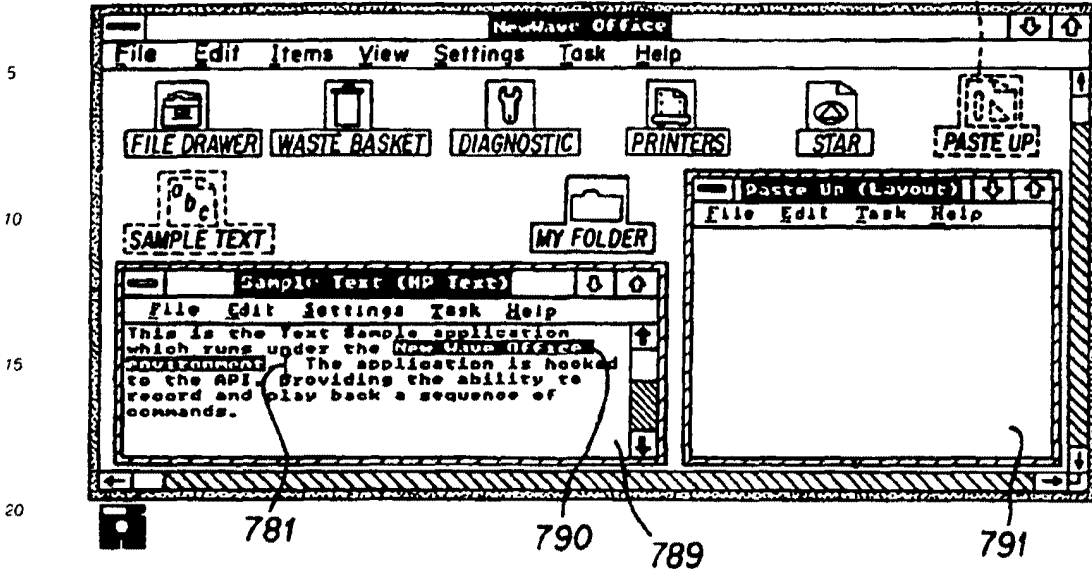


FIG. 38

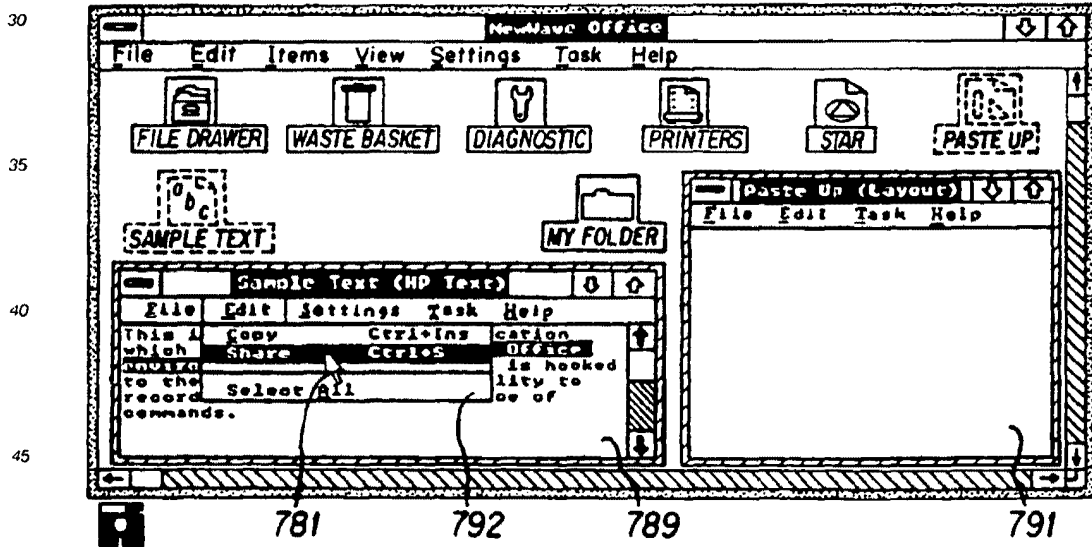


FIG. 39

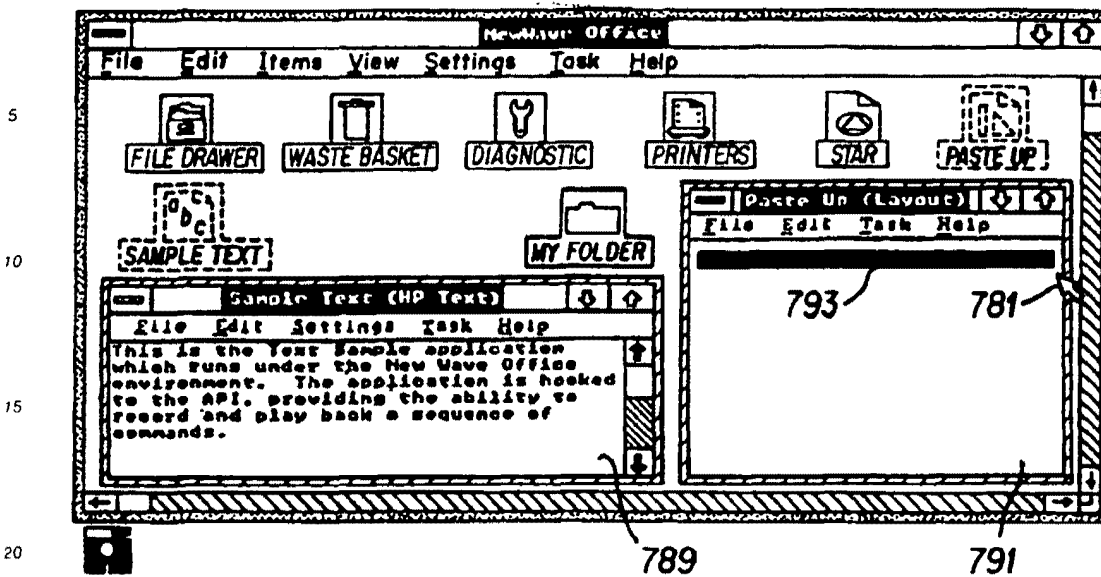


FIG. 40

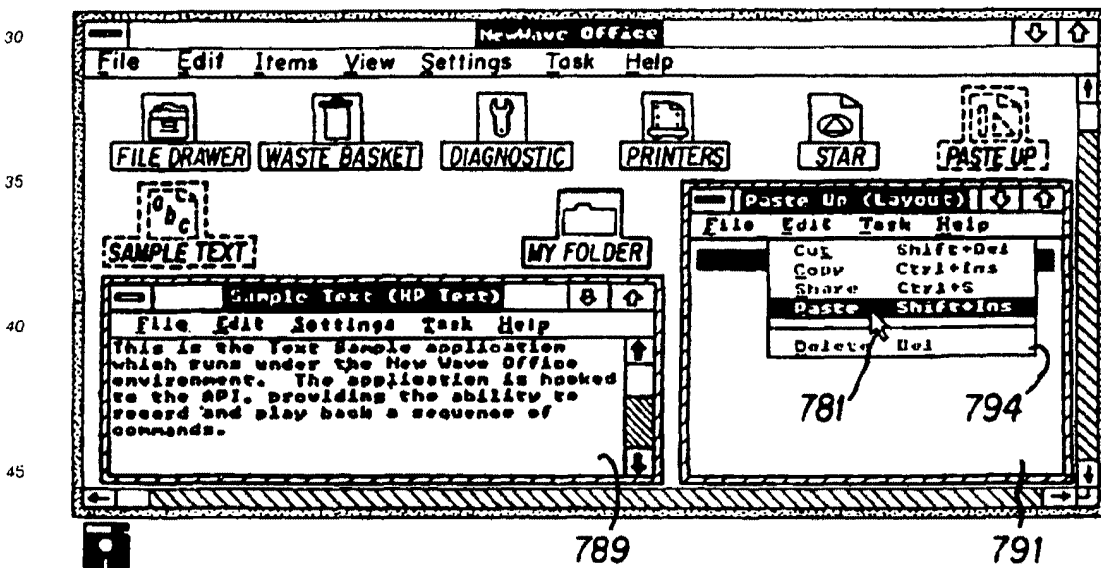


FIG. 41

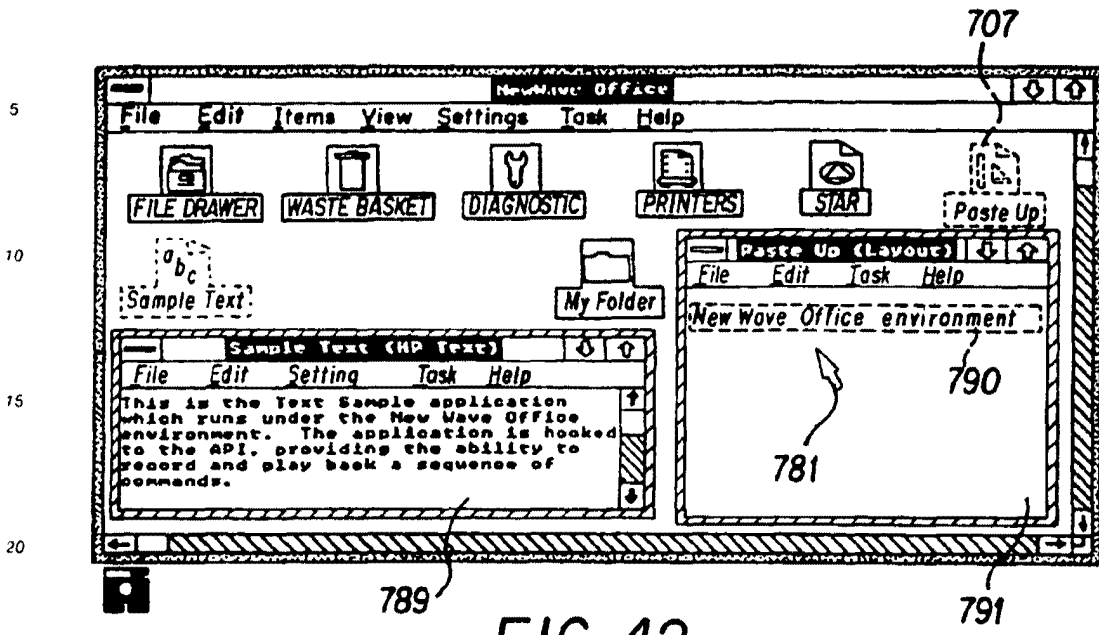


FIG. 42

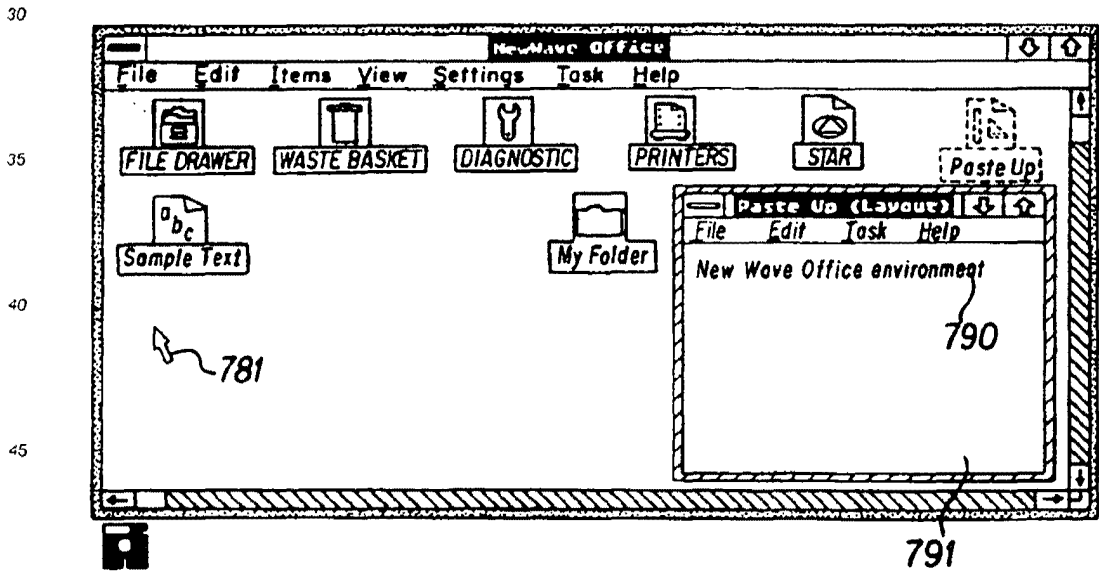


FIG. 44

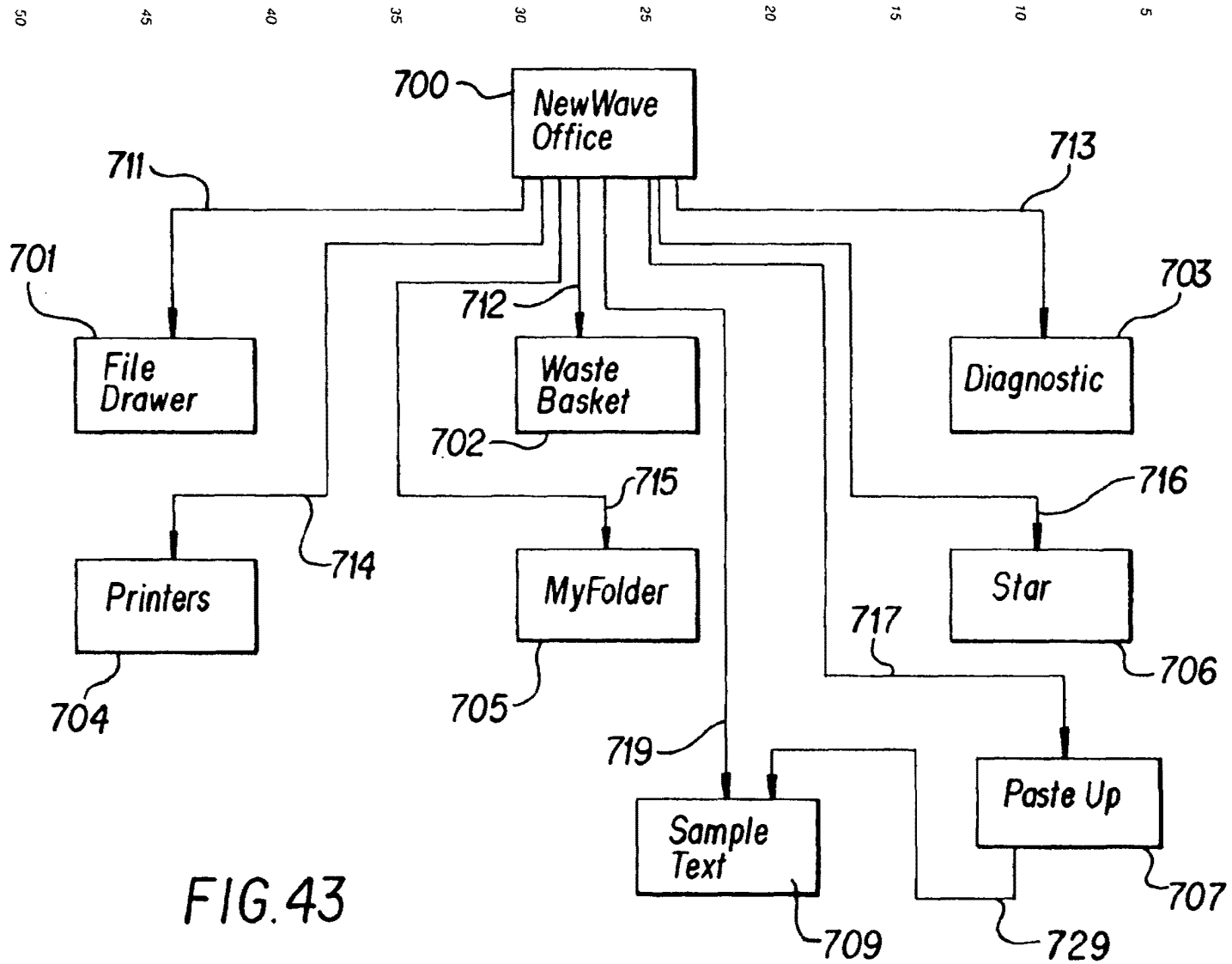


FIG. 43

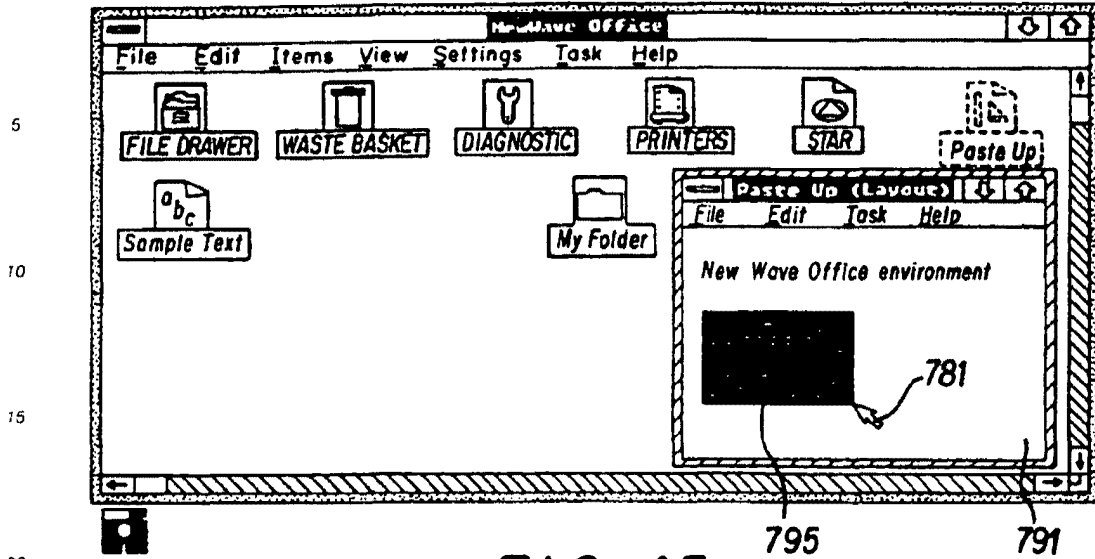


FIG. 45

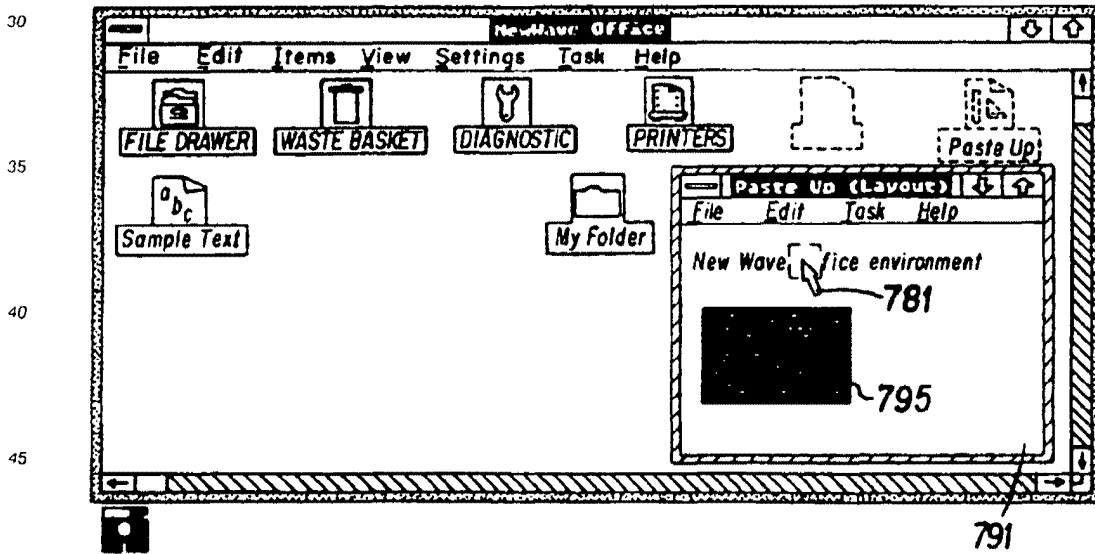


FIG. 46

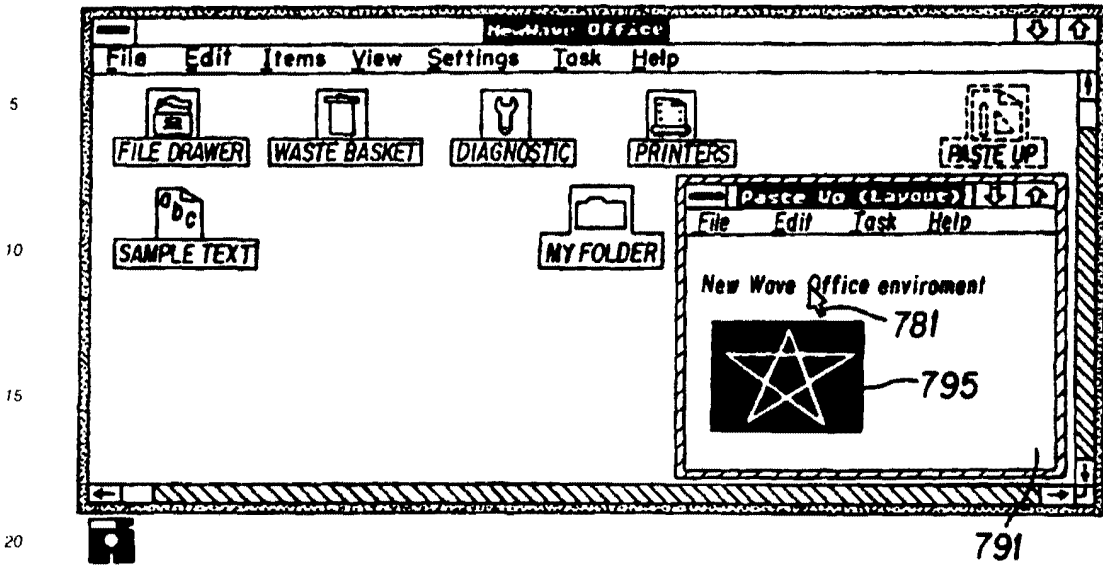


FIG. 47

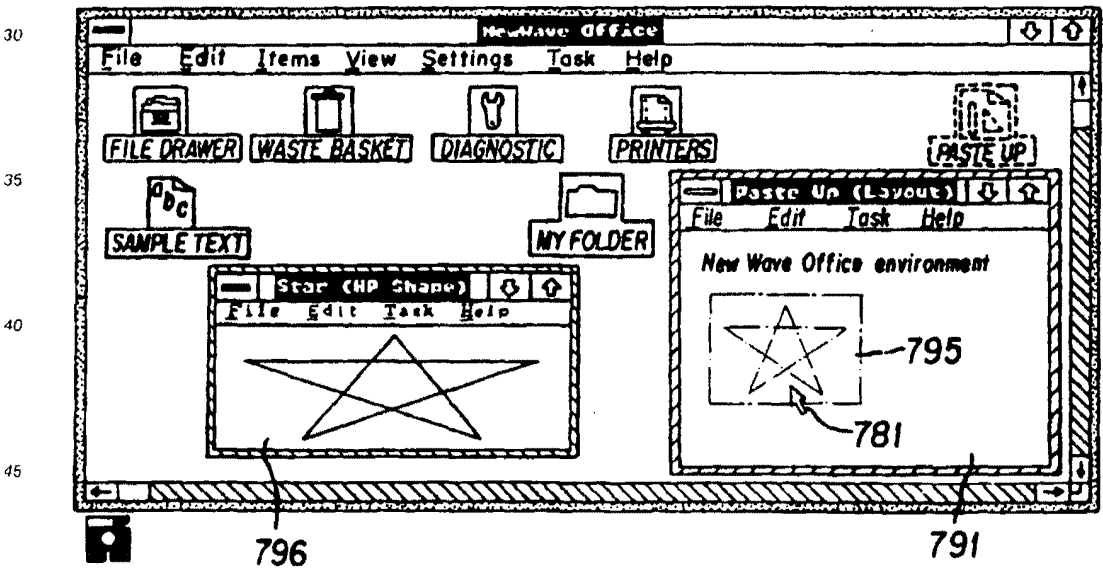


FIG. 49

55 50 45 40 35 30 25 20 15 10 5

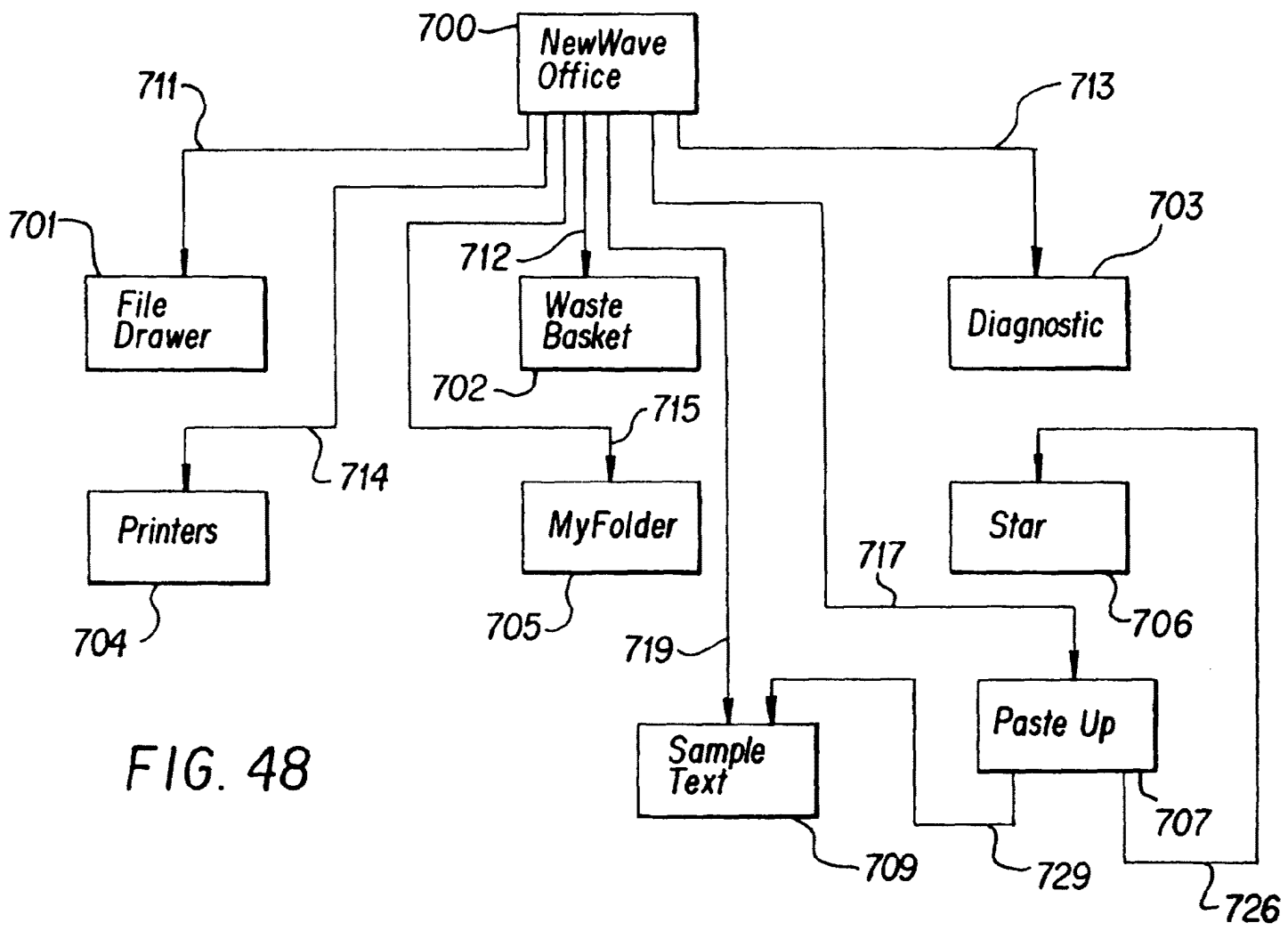


FIG. 48

88

EP 0 497 022 A1

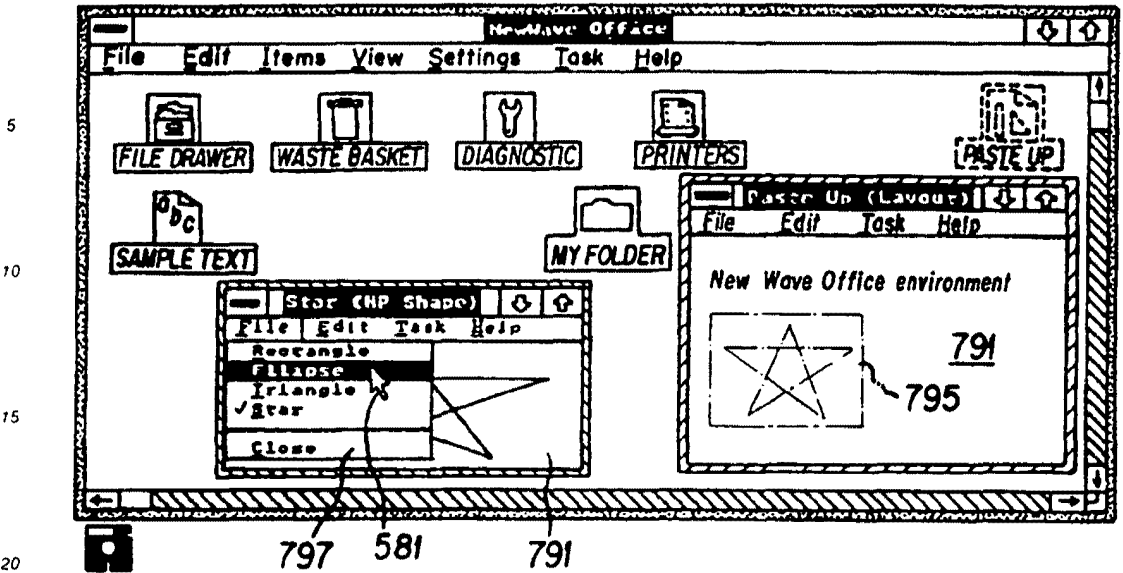


FIG. 50

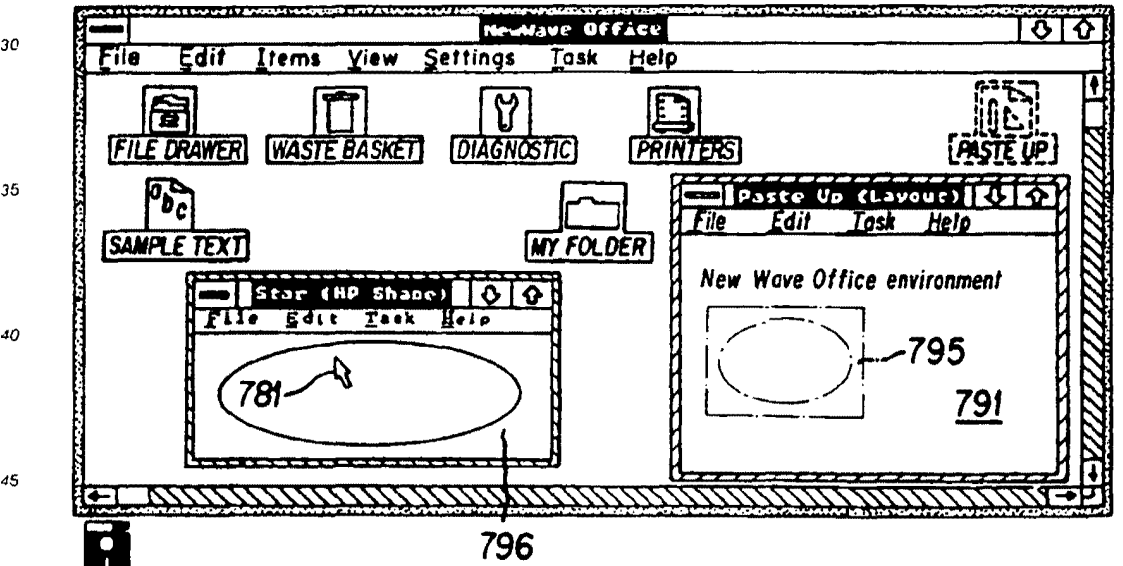


FIG. 51



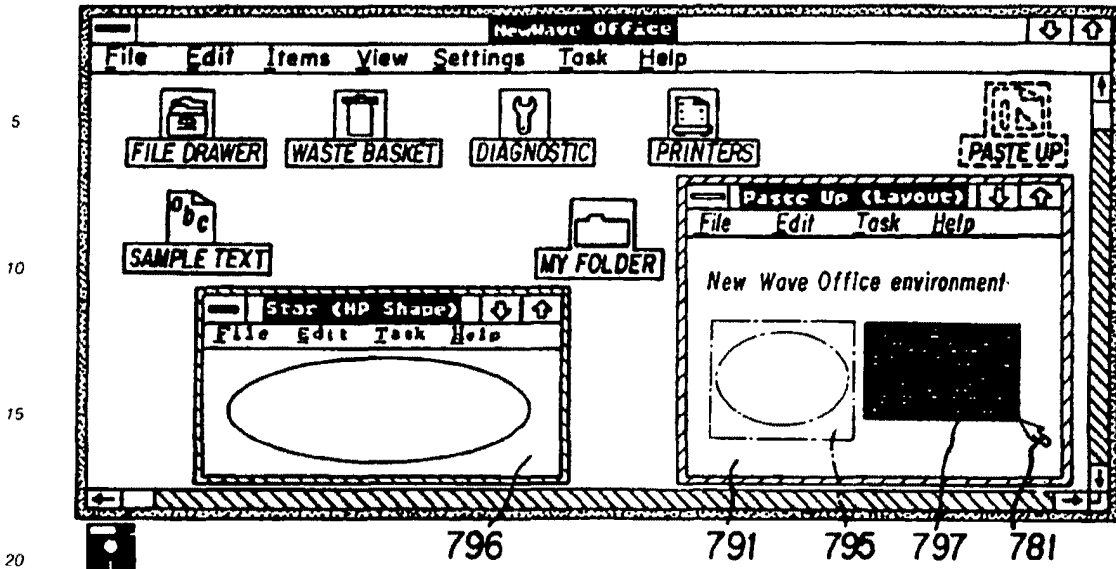


FIG. 52

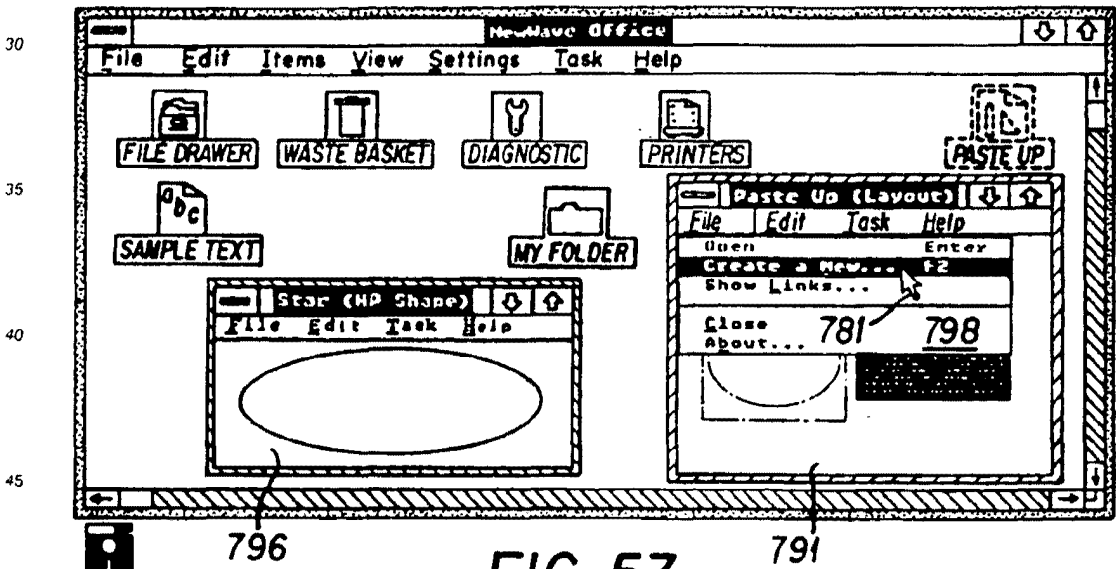


FIG. 53

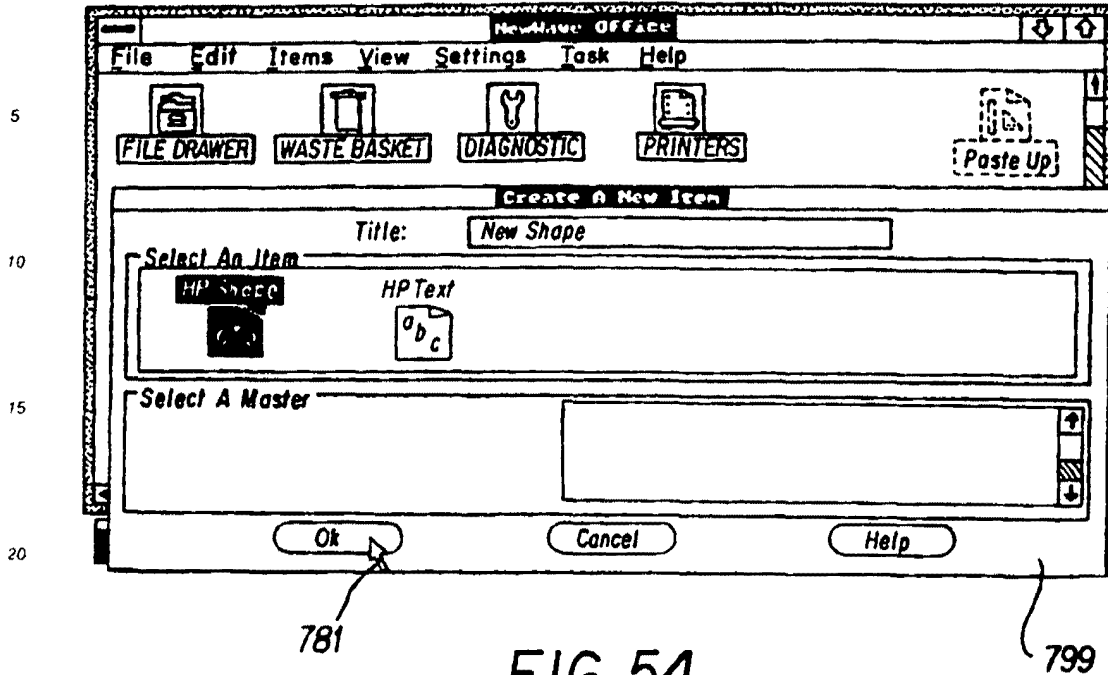


FIG. 54

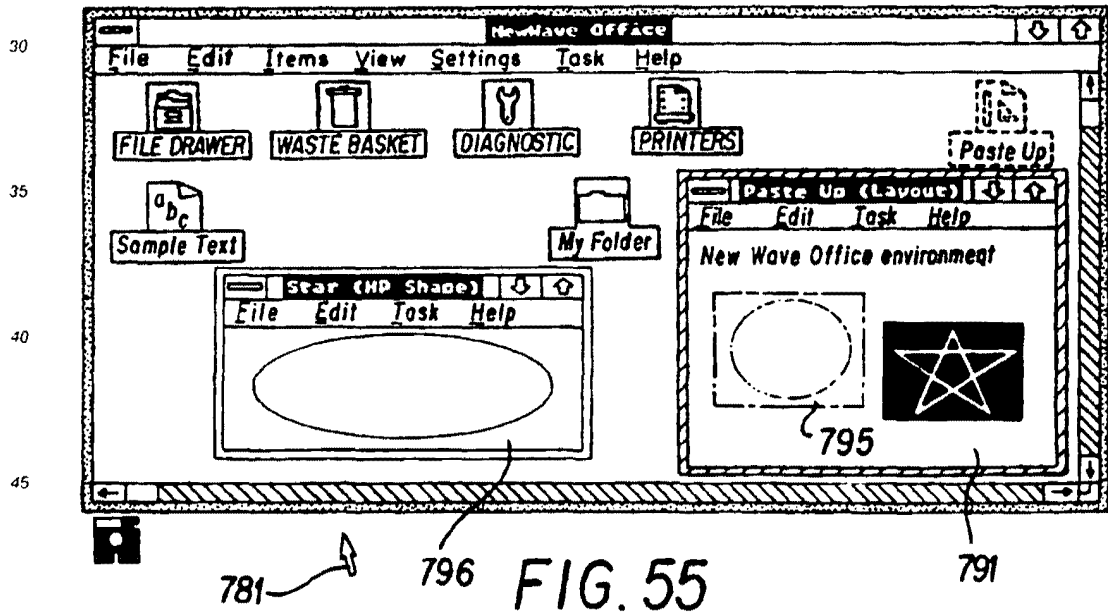


FIG. 55

55  
50  
45  
40  
35  
30  
25  
20  
15  
10  
5

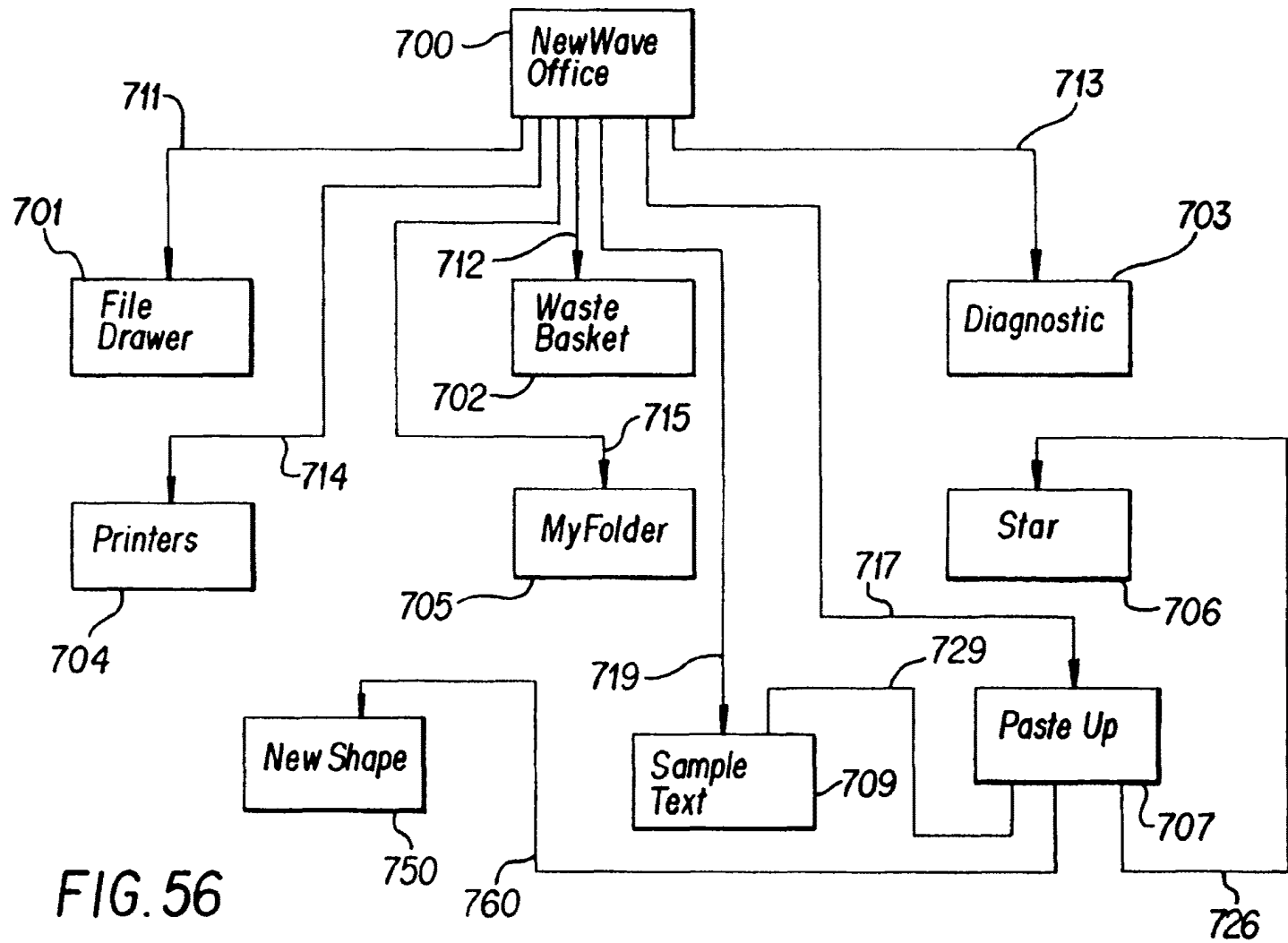
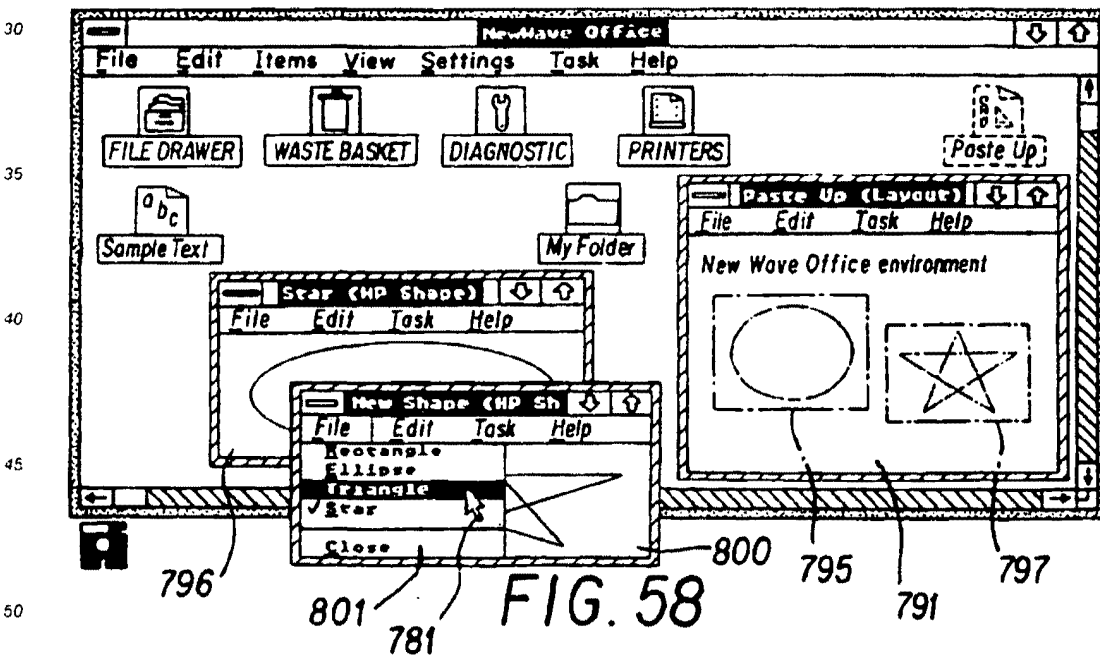
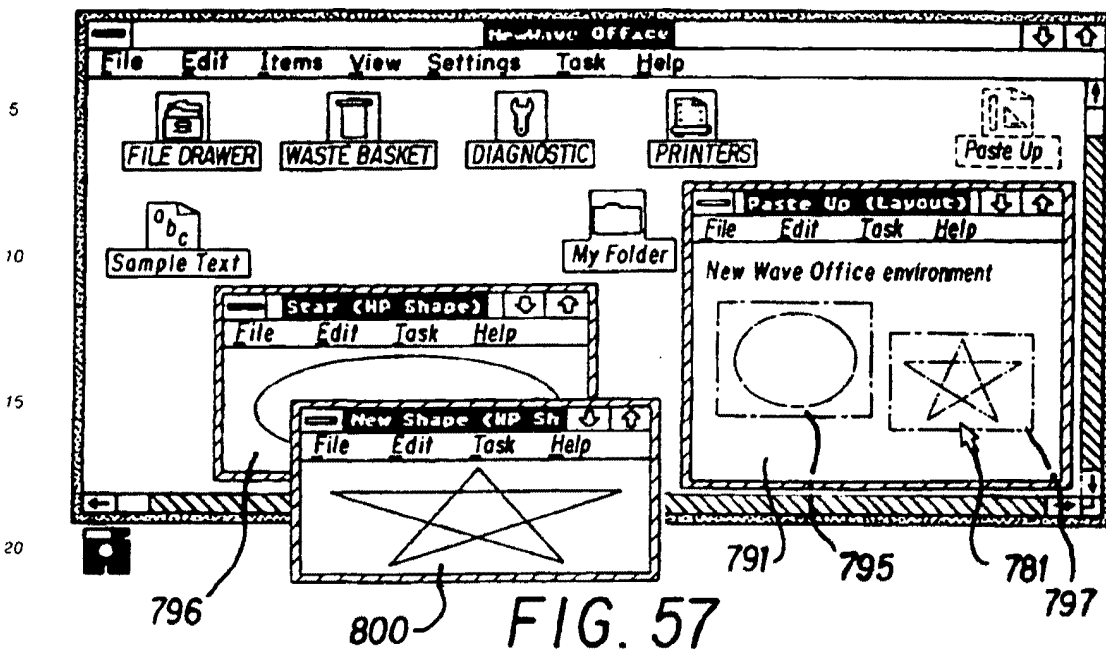


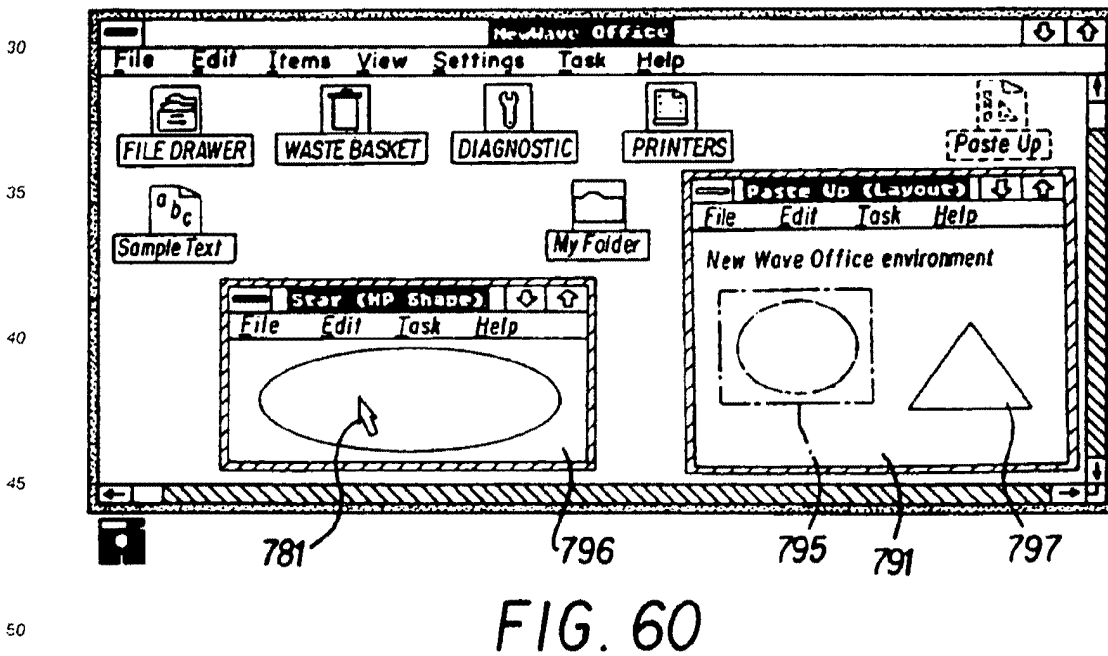
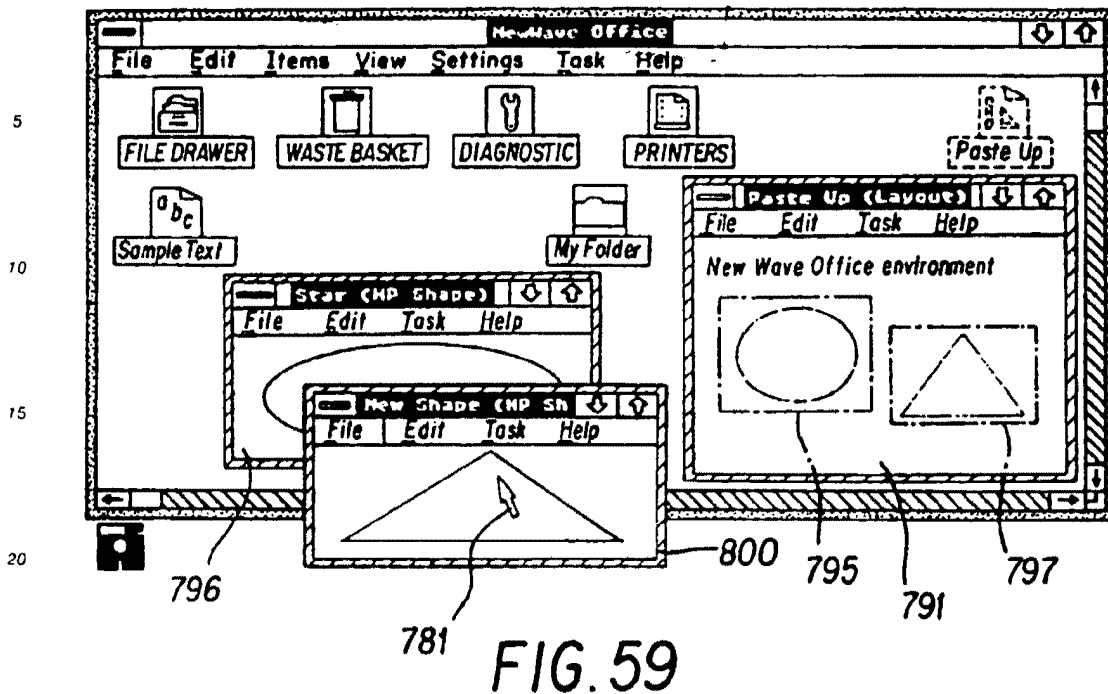
FIG. 56

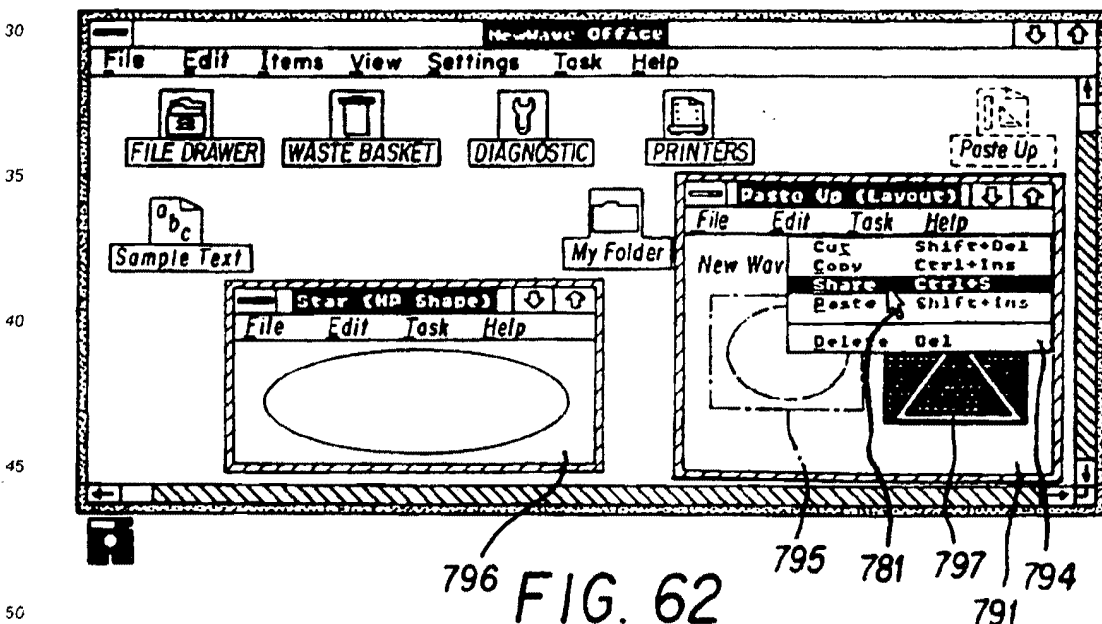
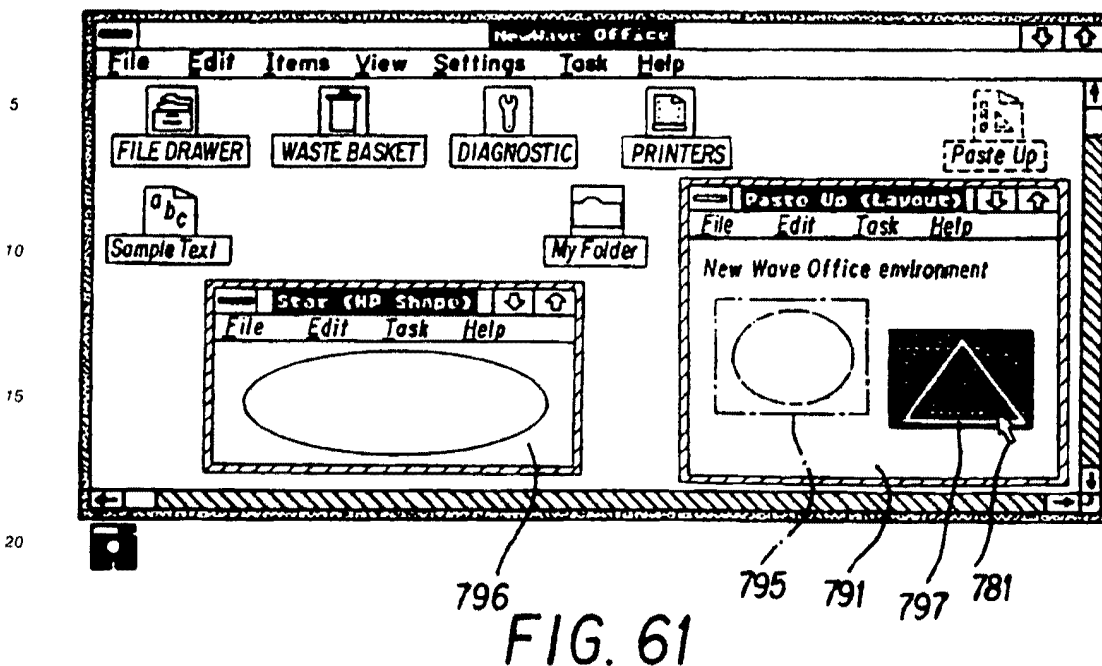
92

EP 0 497 022 A1

SKYPE-N2P00283614







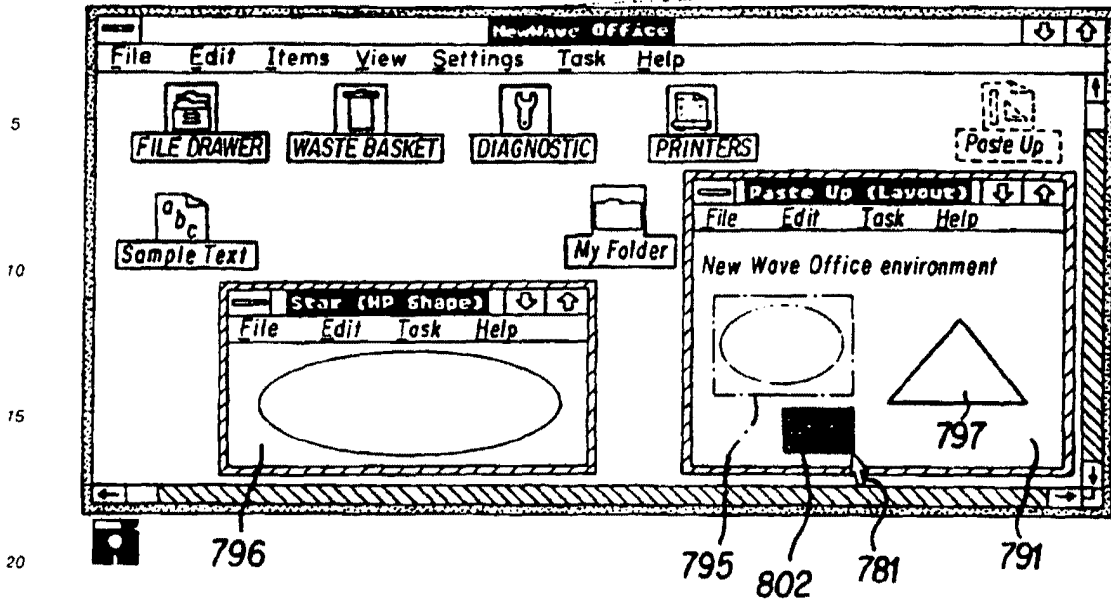


FIG. 63

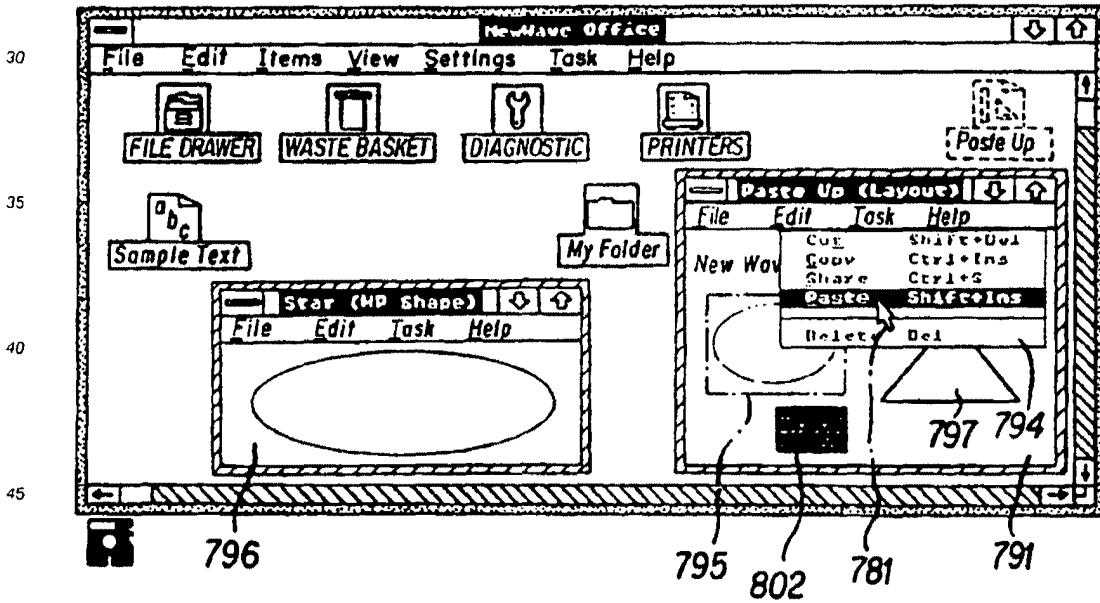


FIG. 64

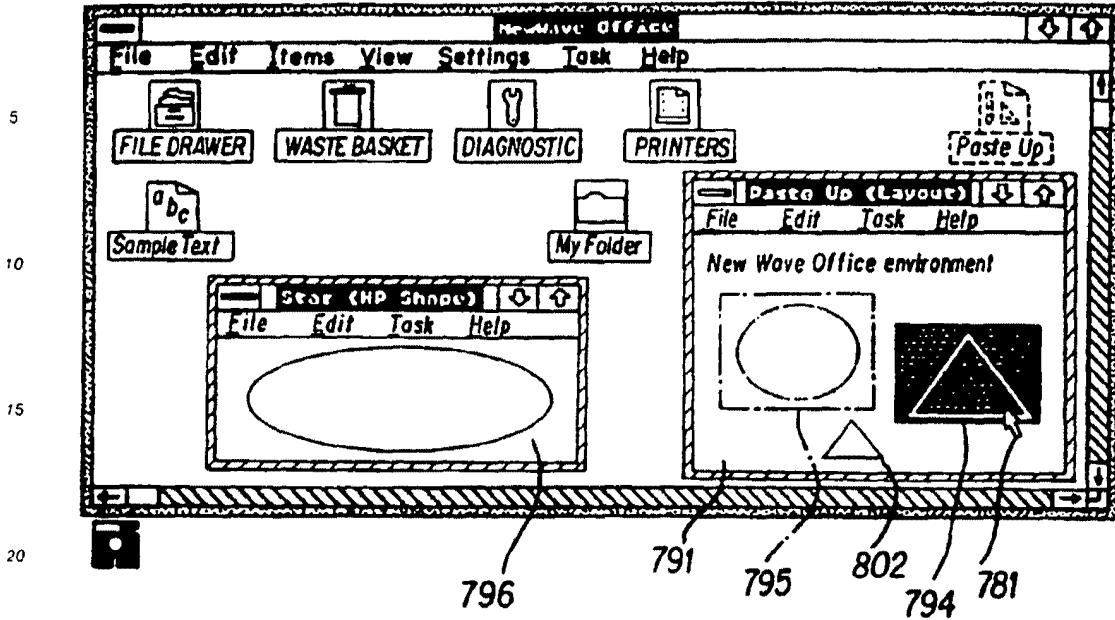


FIG. 65

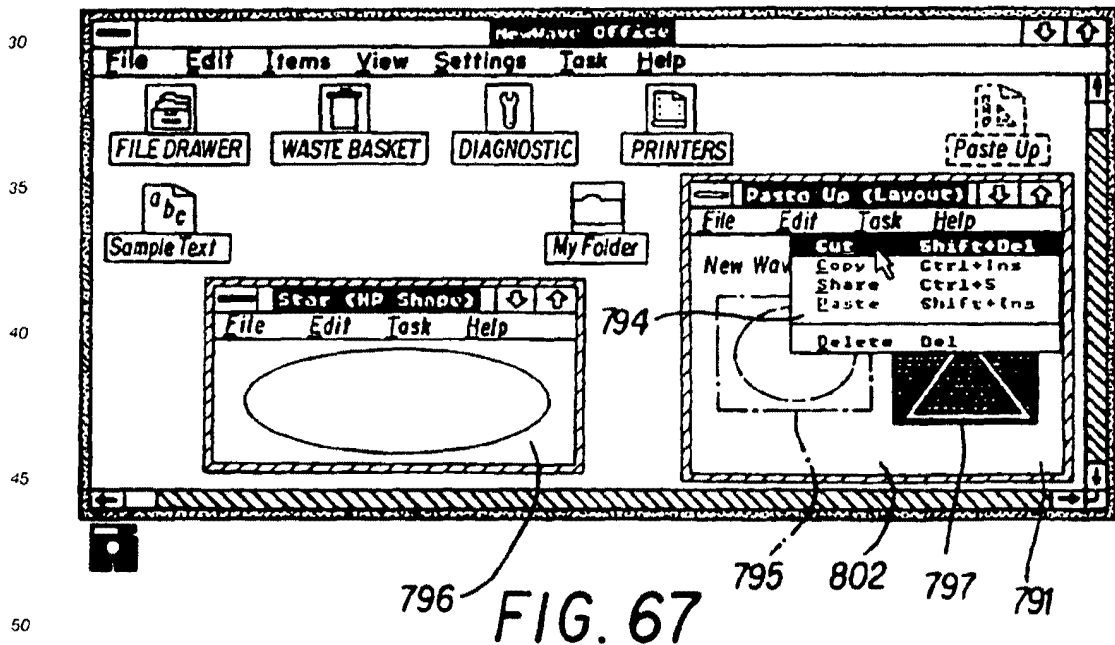


FIG. 67



55 50 45 40 35 30 25 20 15 10 5

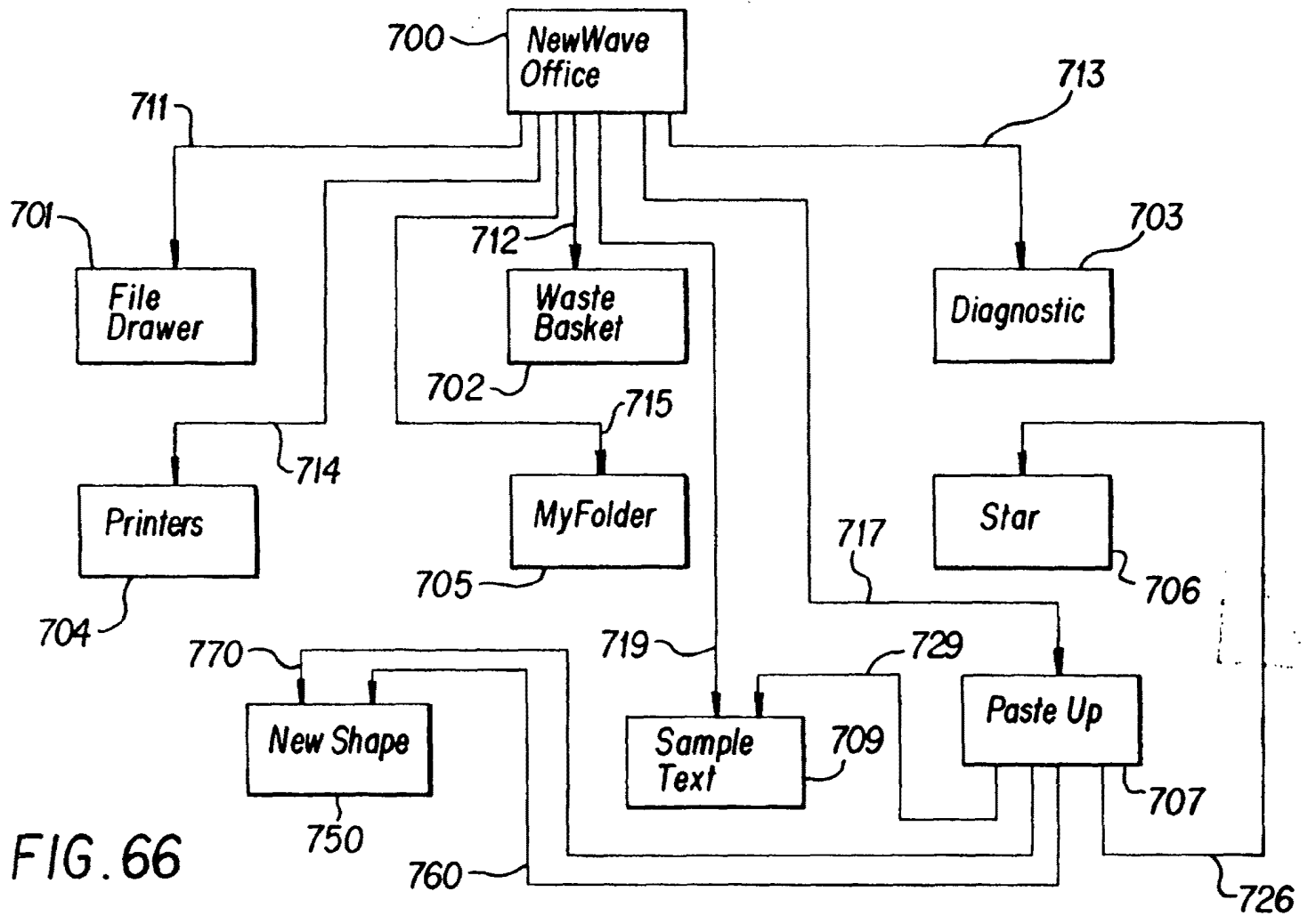


FIG. 66

98

EP 0 497 022 A1

SKYPE-N2P00283620

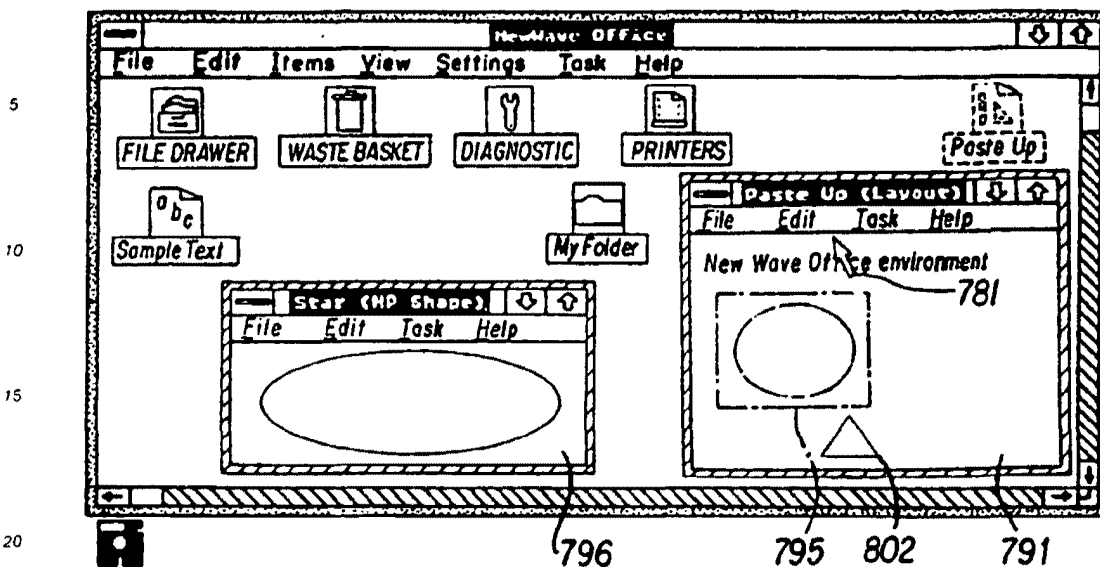


FIG. 68

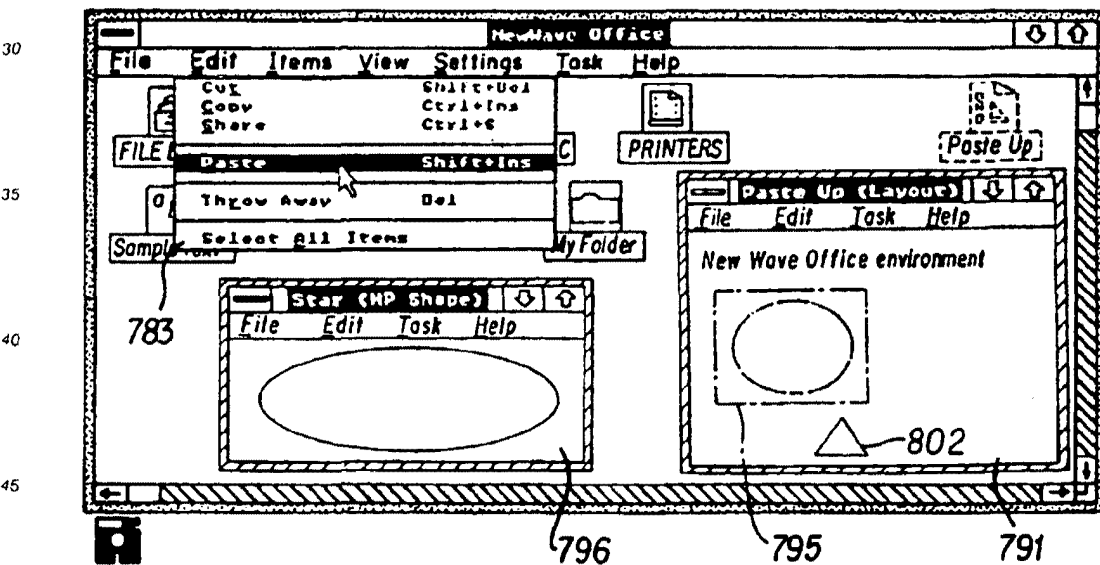


FIG. 69

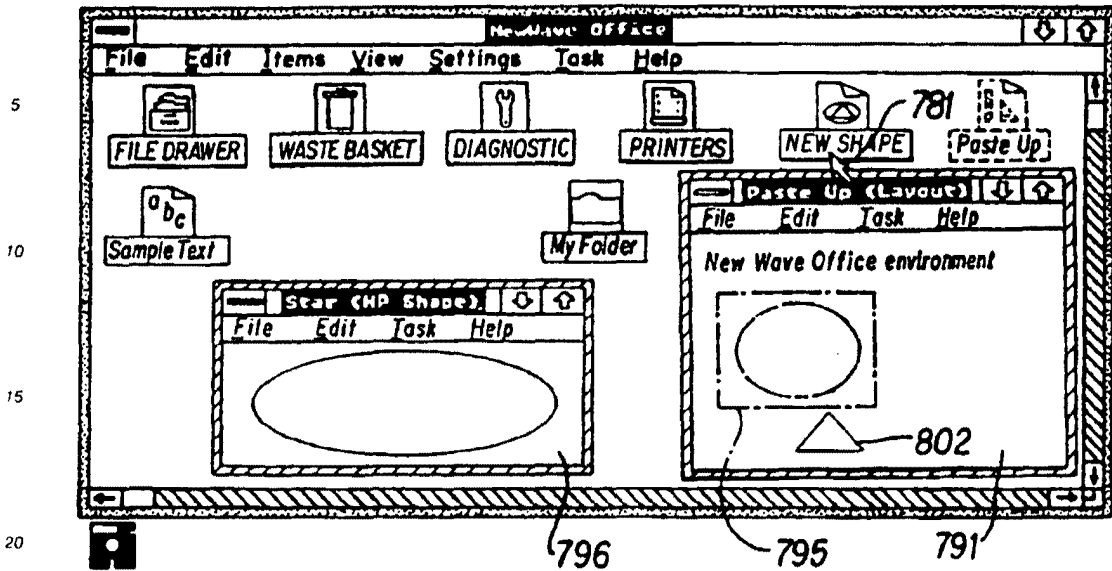


FIG. 70

25

30

35

40

45

50

55

55 50 45 40 35 30 25 20 15 10 5

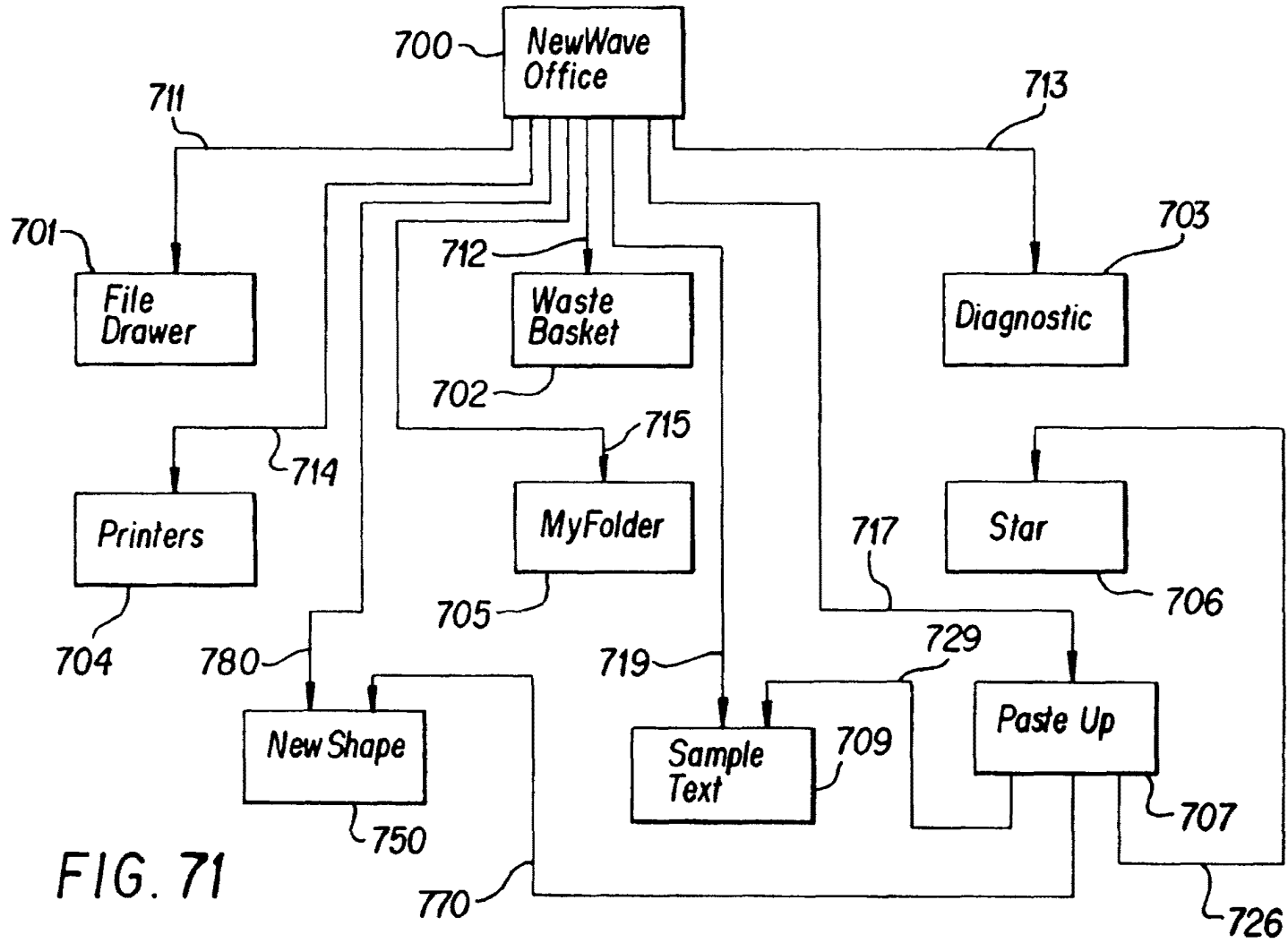


FIG. 71

101

EP 0 497 022 A1

SKYPE-N2P00283623

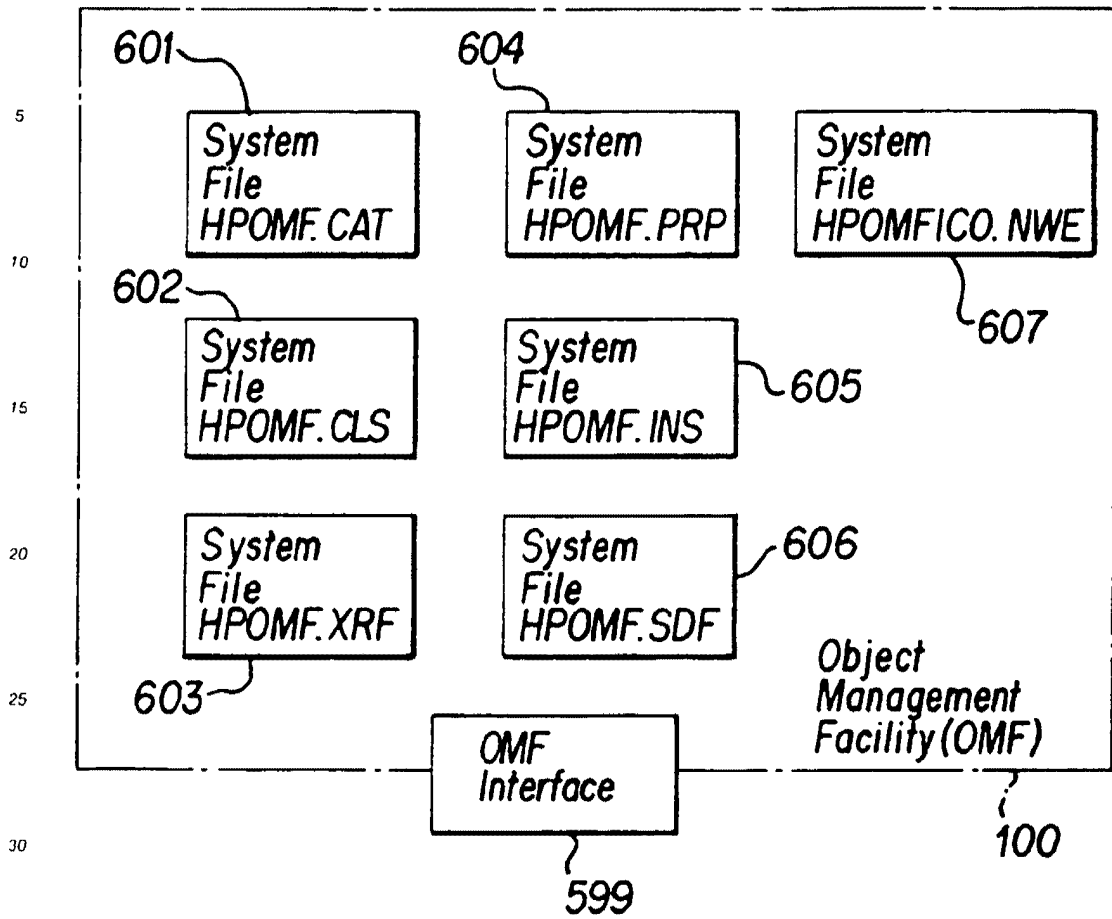


FIG. 72

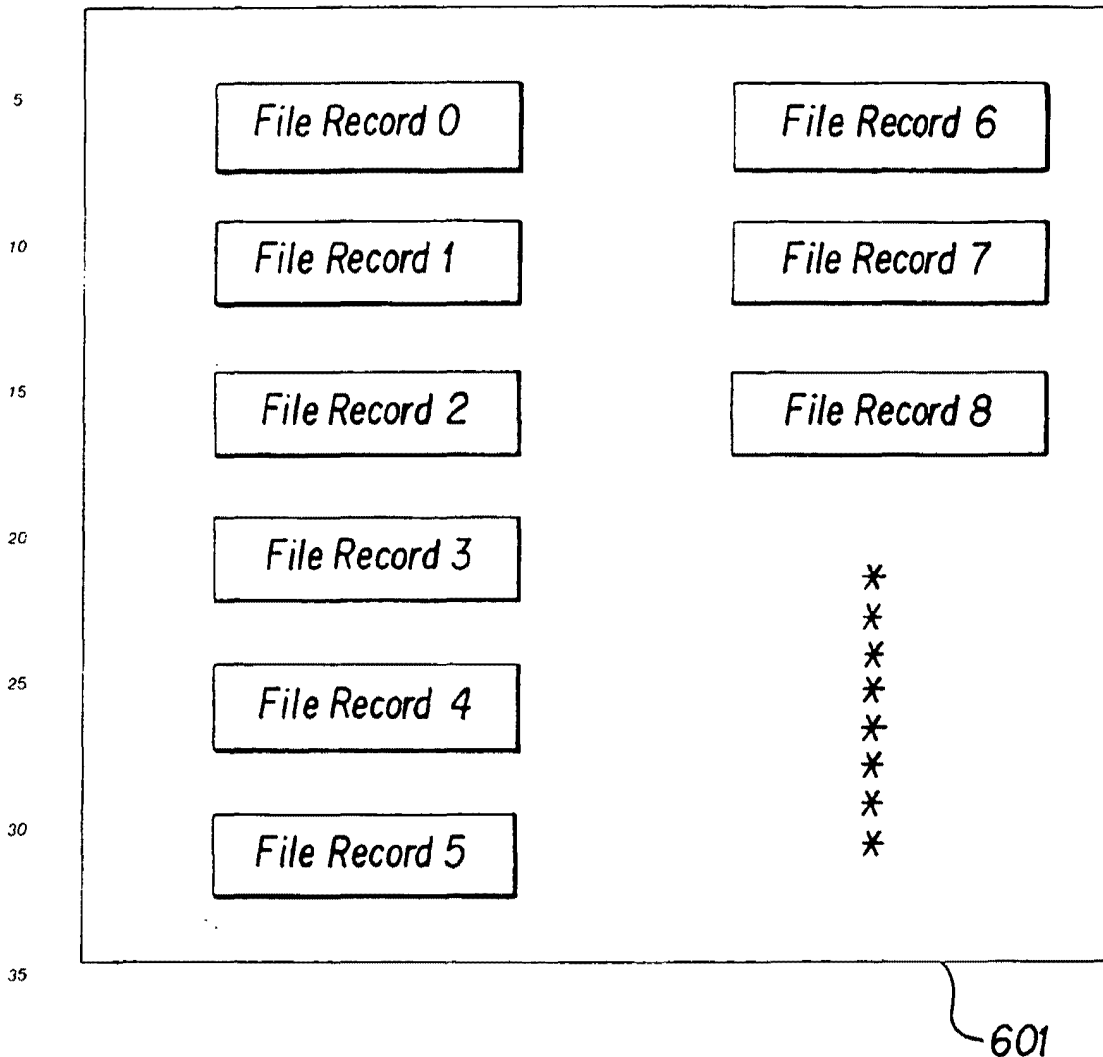
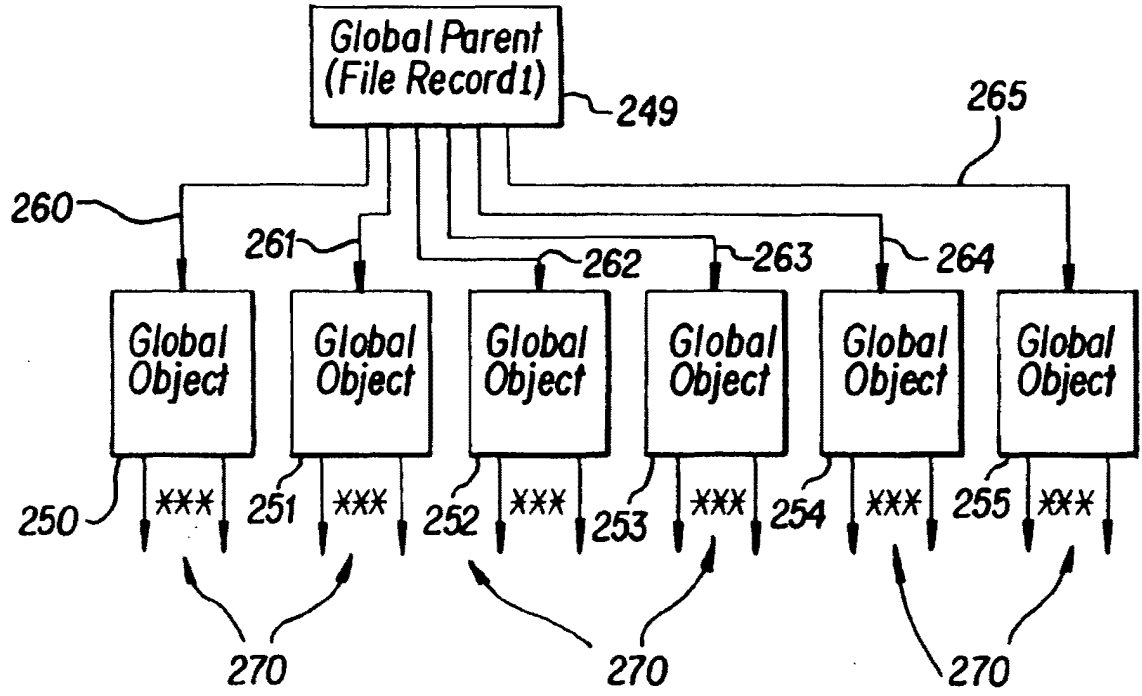


FIG. 73

5  
10  
15  
20  
25  
30  
35  
40  
45  
50  
55



104

EP 0 497 022 A 1

FIG. 74

55 50 45 40 35 30 25 20 15 10 5

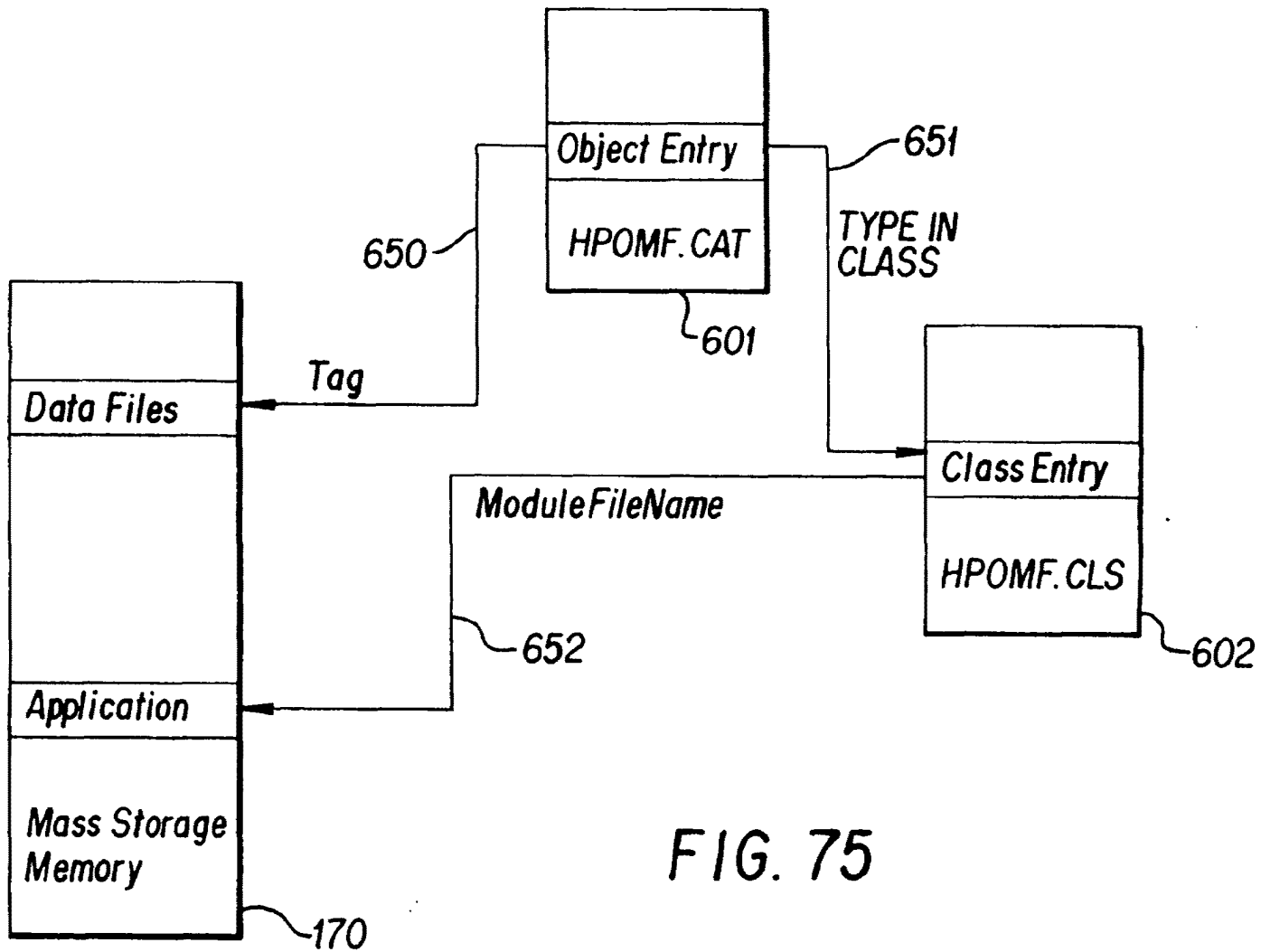


FIG. 75

EP 0 497 022 A1

SKYPE-N2P00283627



5  
10  
15  
20  
25  
30  
35  
40  
45  
50

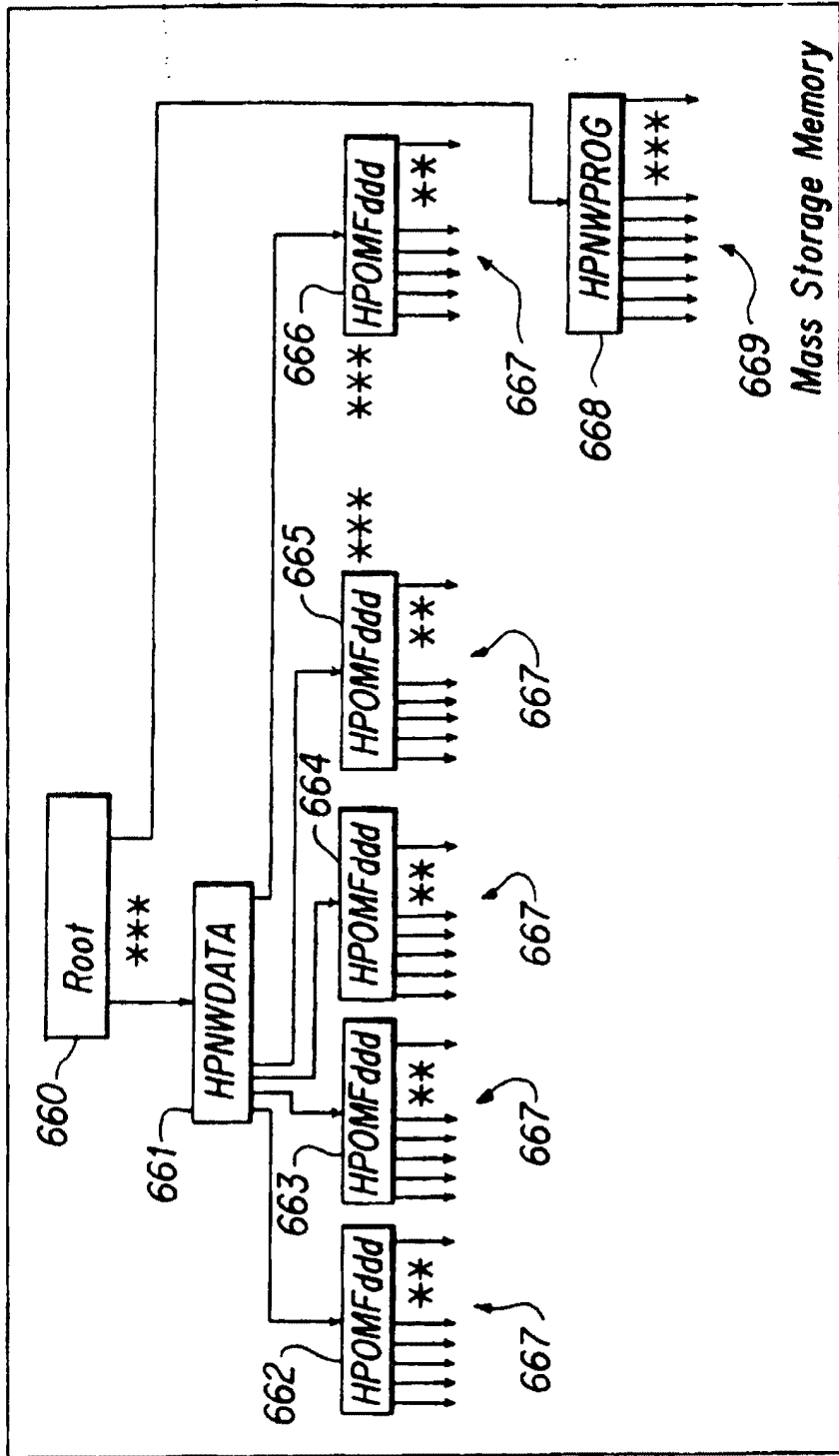


FIG. 76

170

55

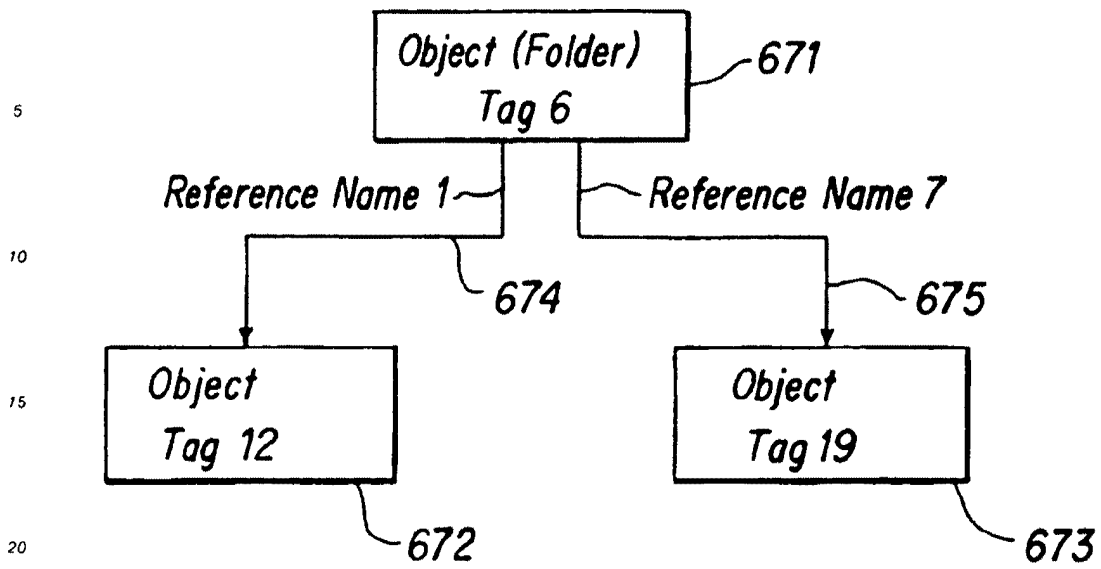


FIG. 77

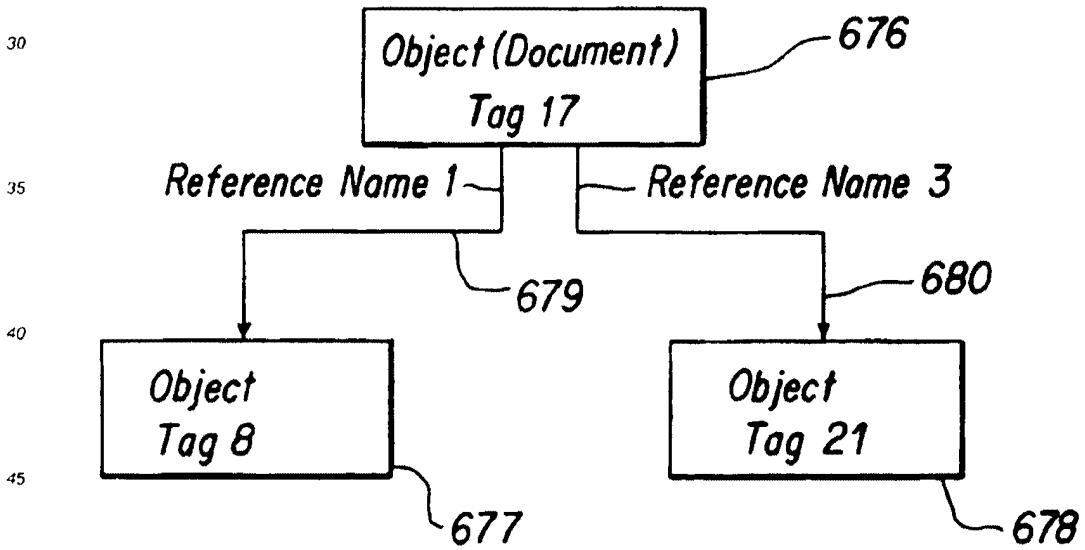


FIG. 78

50

55

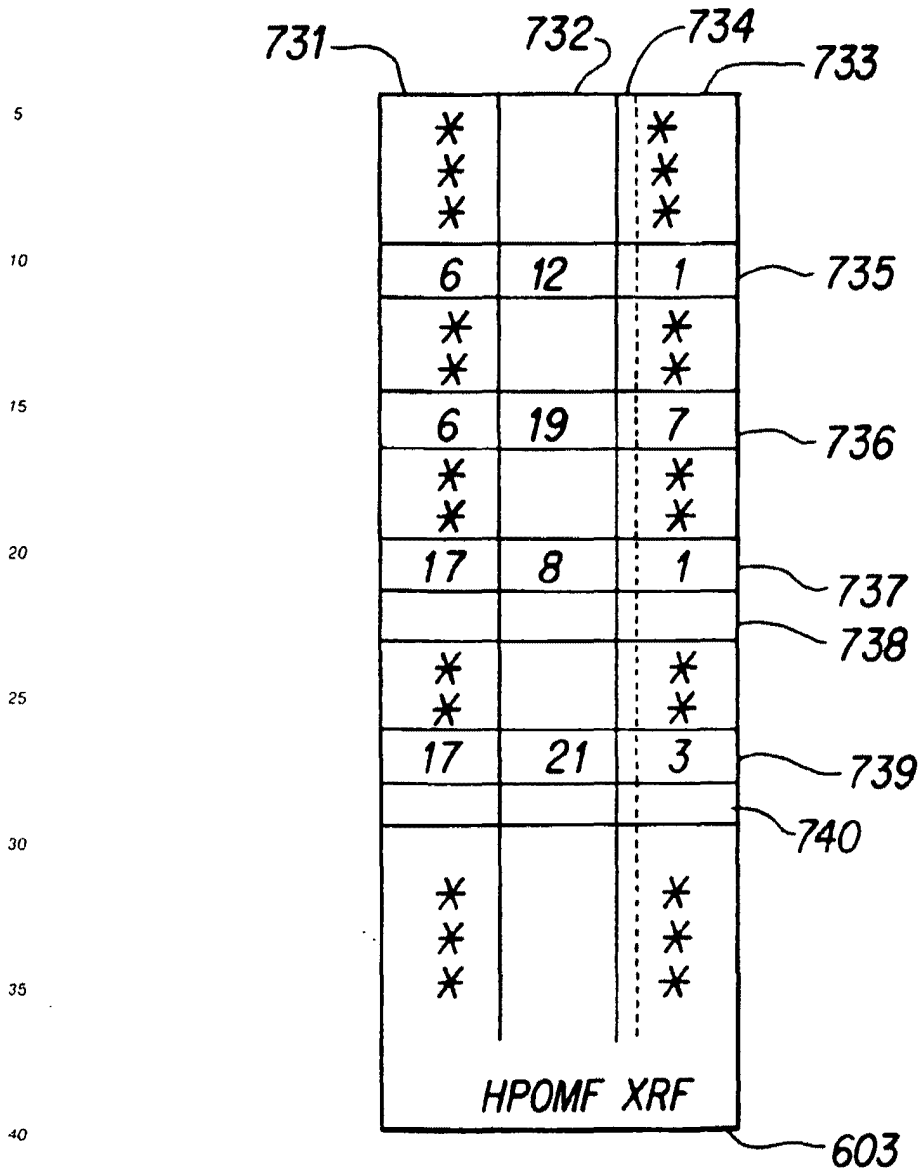
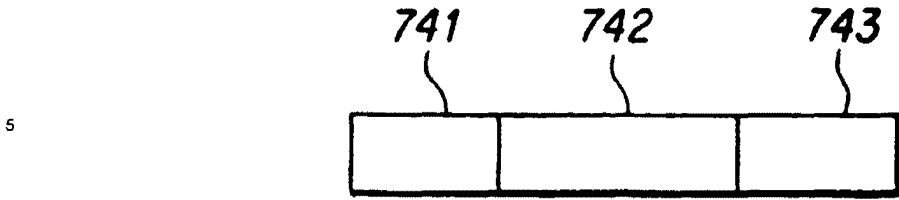


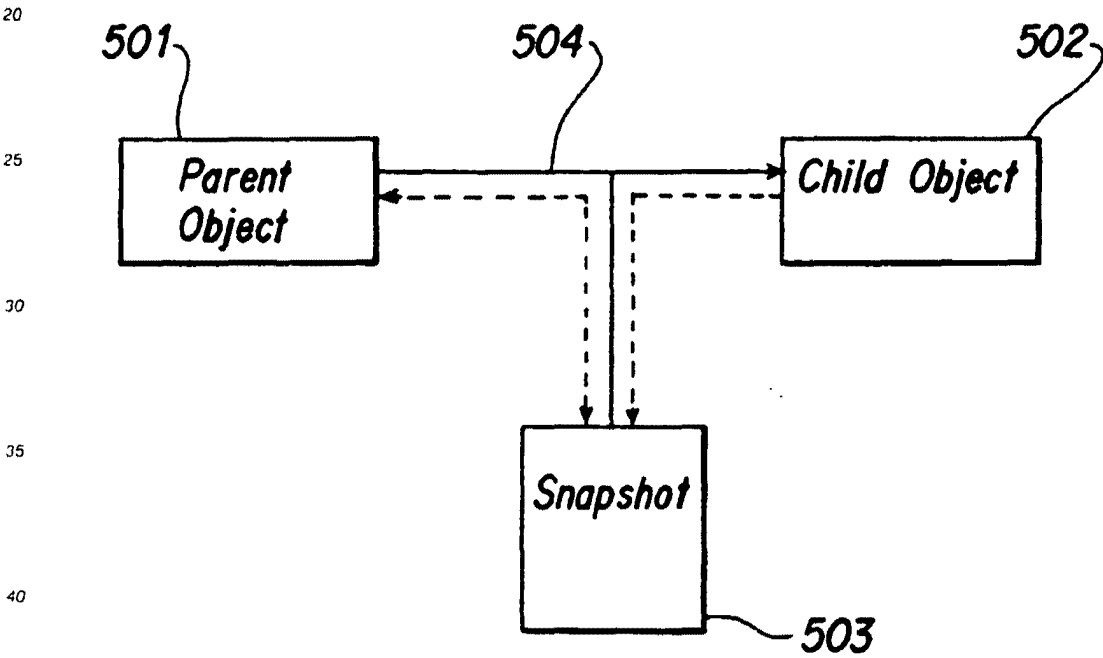
FIG. 79



10 *View Specification Record 740*

**FIG. 80**

15



45 **FIG. 81**

50

55

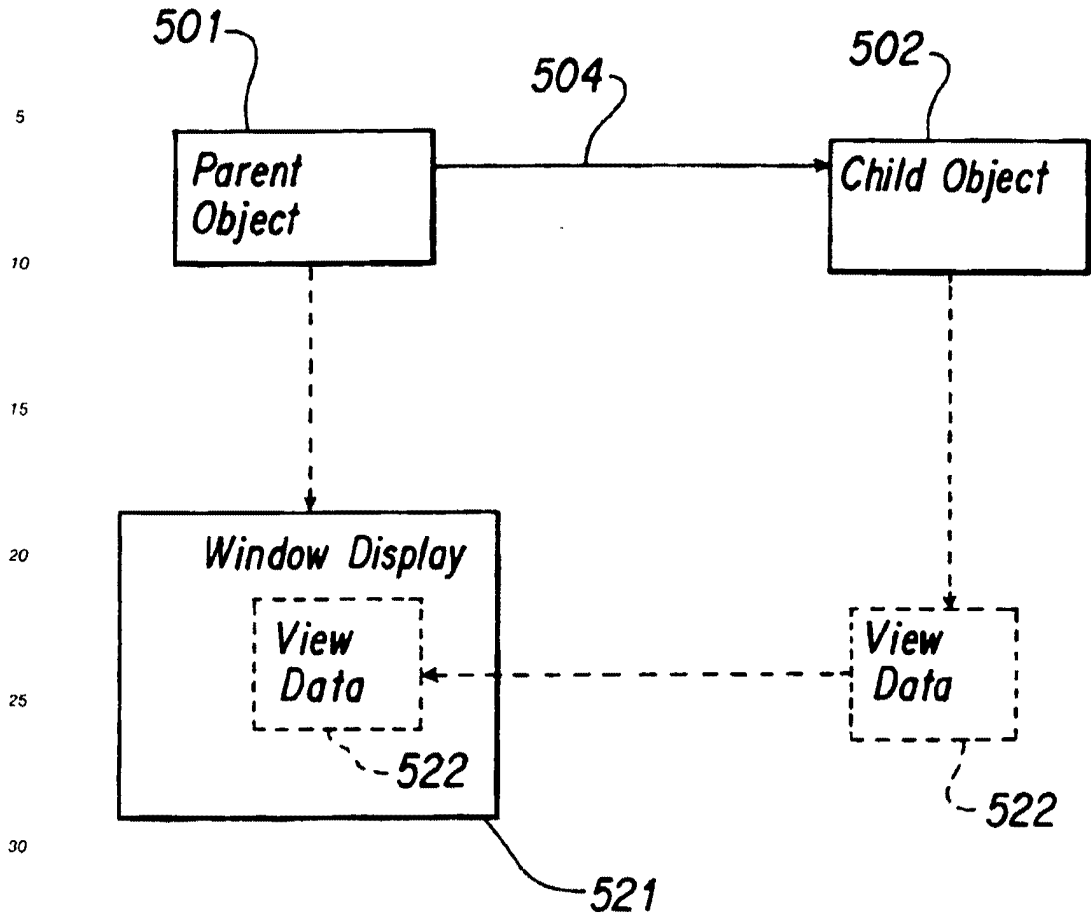


FIG. 82

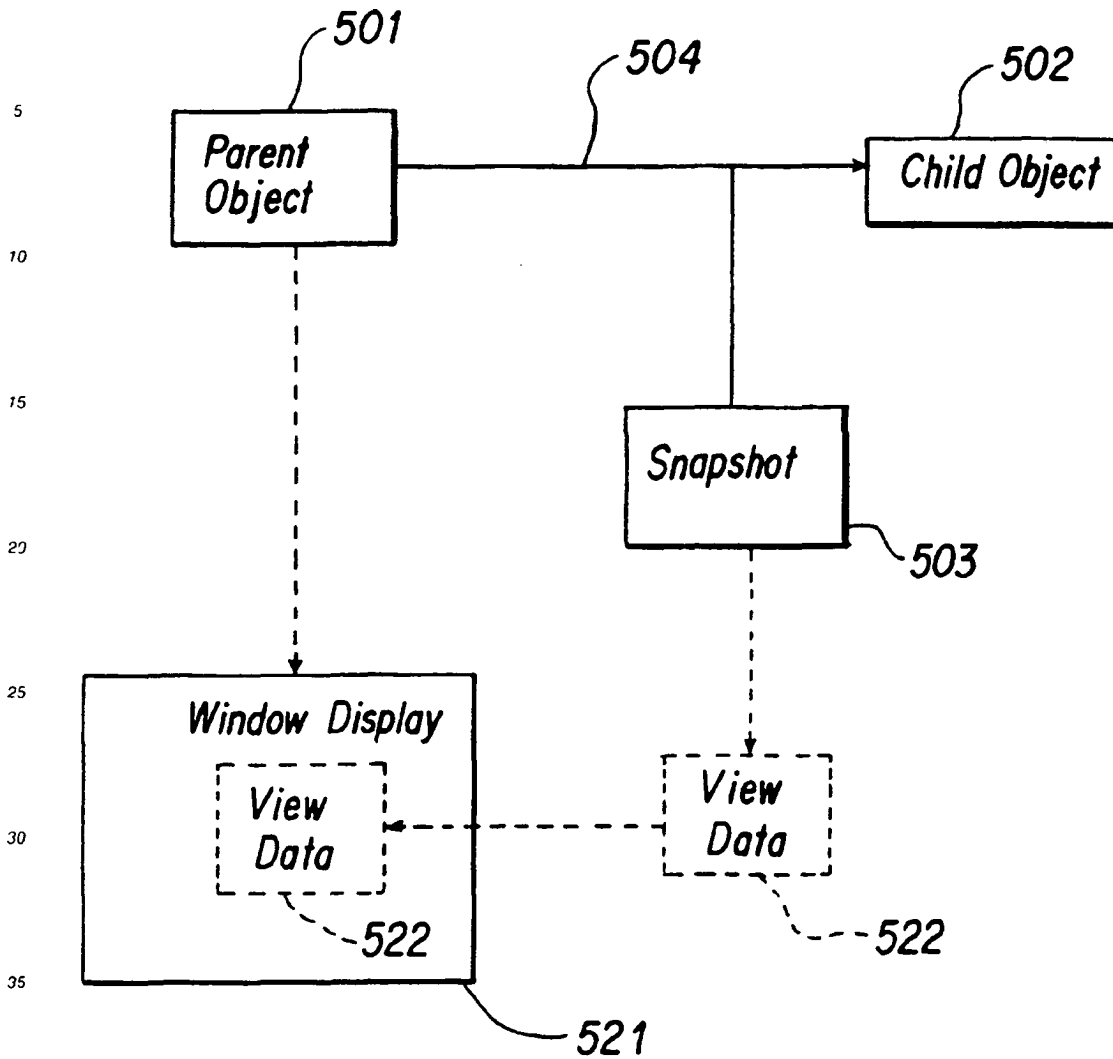


FIG. 83

Claims

1. An object based distributed computer system comprising a network of workstations and means for transmitting objects between workstations characterised by objects including a first object type for storing data and a second object type for presenting data to a user, wherein objects of the second type (V-c) reference an associated object of the first type (V-s) to enable a plurality of users of workstations to access data of the object of the first type, comprising means for transmitting an object of the second type (V-c) between workstations thereby to create a reference to the associated object of the first type (V-s) for each workstation receiving an object of the second type.
2. A system according to claim 1 comprising means for copying an object of the second type (V-c) between workstations.

3. A system according to claim 1 or claim 2 wherein transmitted objects of the second type (V-c) include an identifier (60) for the associated object of the first type (V-s).
- 5 4. A system according to any preceding claim in the form of a conferencing system comprising means enabling users of the workstations to participate in a meeting over the network wherein objects of the first type (V-s) store meeting data and objects of the second type (V-c) are for presenting meeting data.
- 10 5. A method of convening a meeting using a system as claimed in claim 4 comprising transmitting an object of the second type (V-c) between workstations thereby to create a reference to the associated object of the first type (V-s) for each workstation receiving an object of the second type.

15

20

25

30

35

40

45

50

55

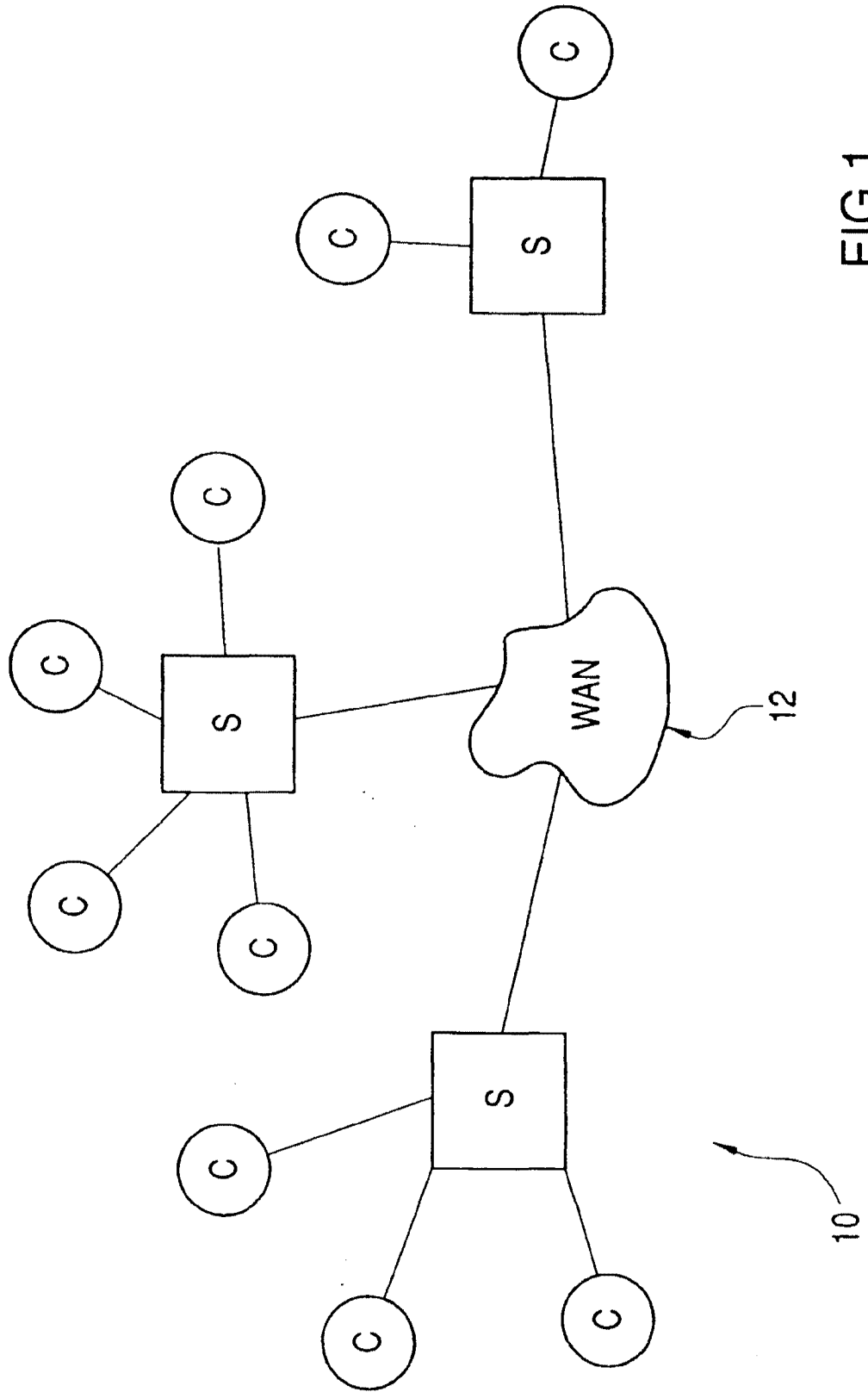


FIG 1



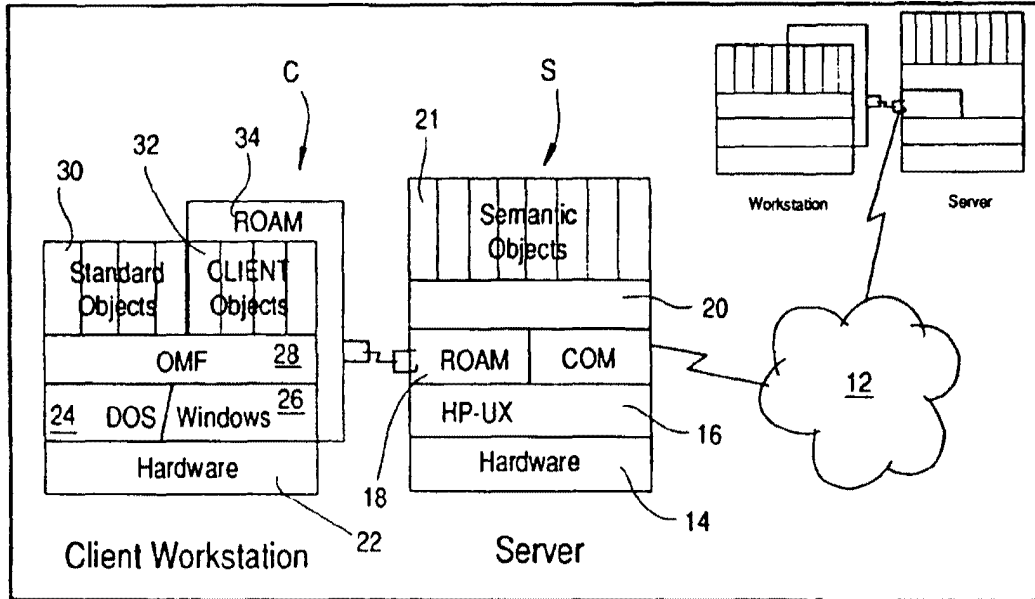


FIG 2

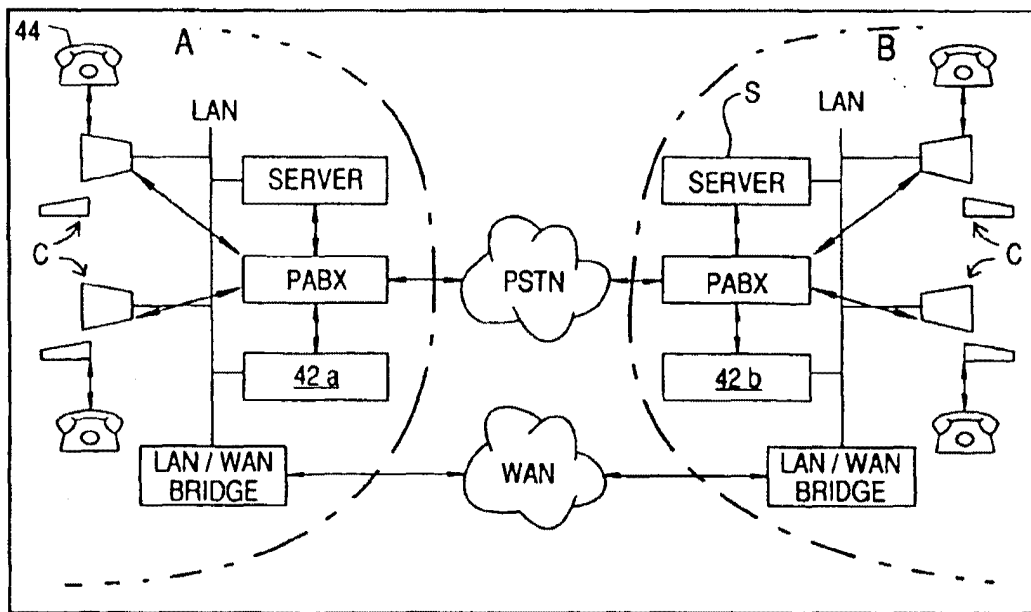


FIG 3

40

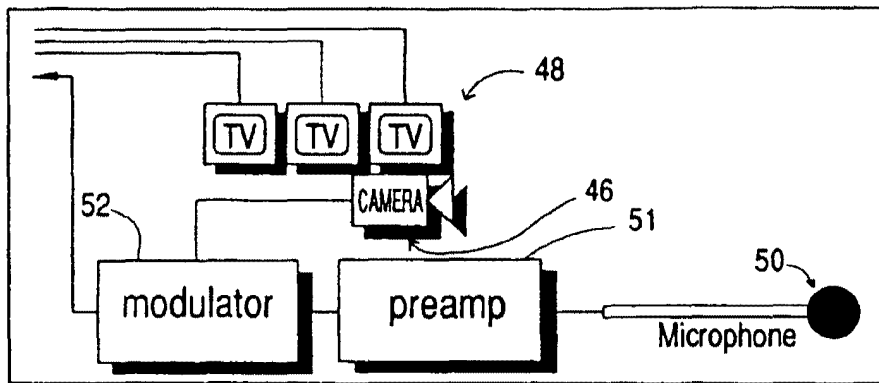


FIG 4

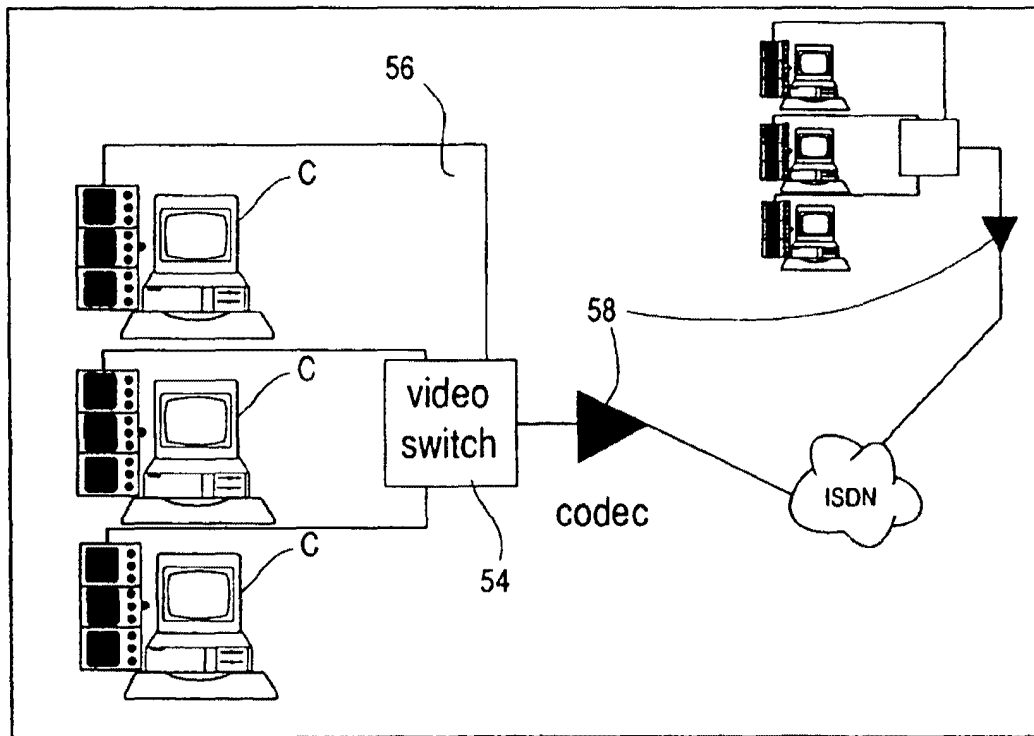


FIG 5

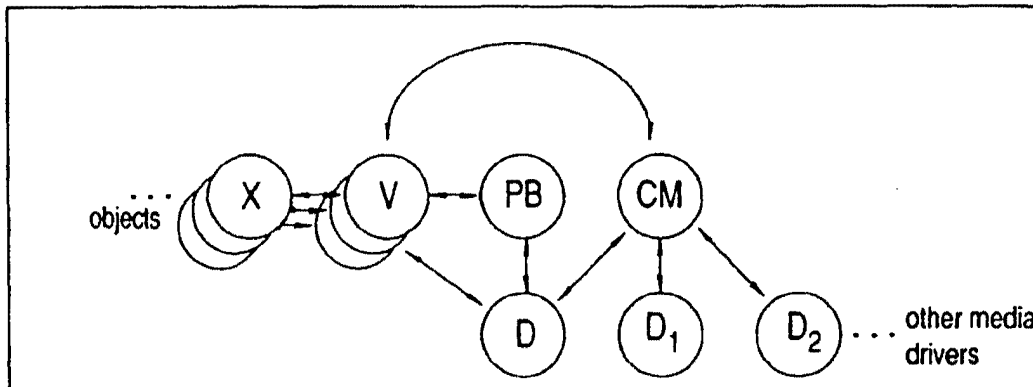


FIG 6

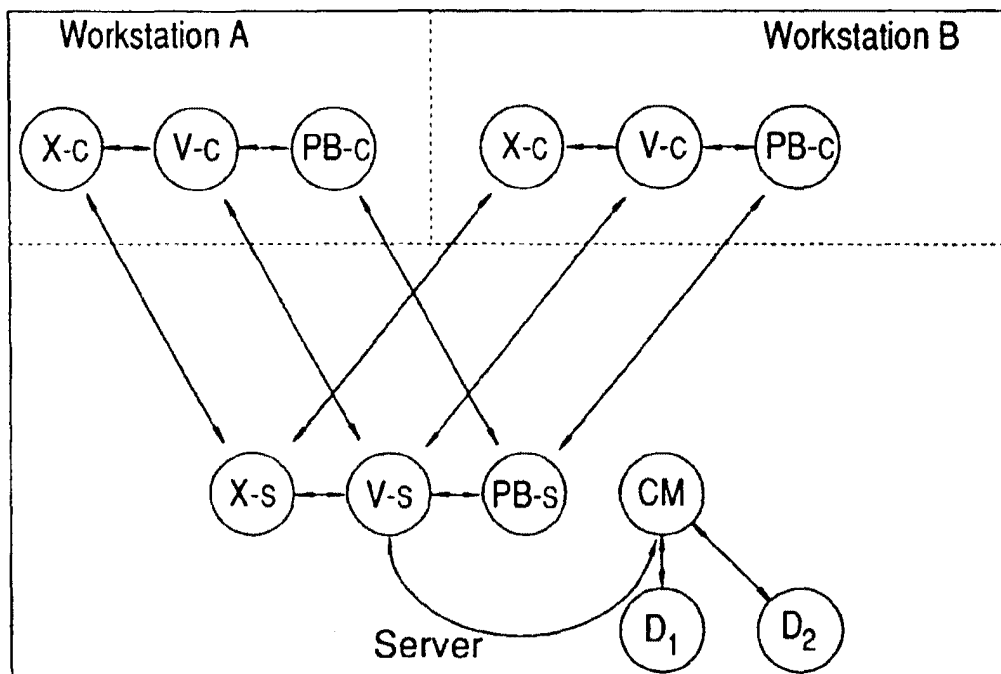


FIG 7

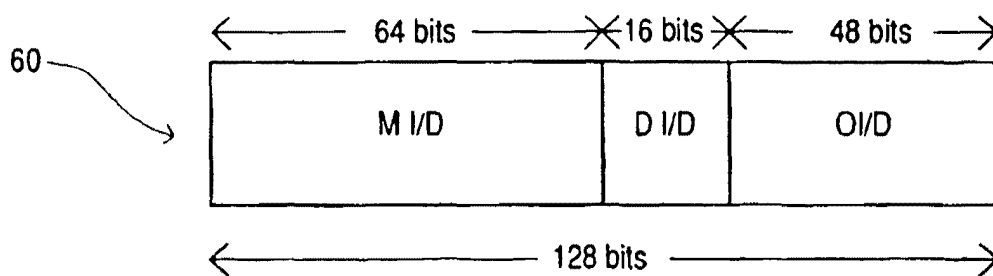


FIG 8

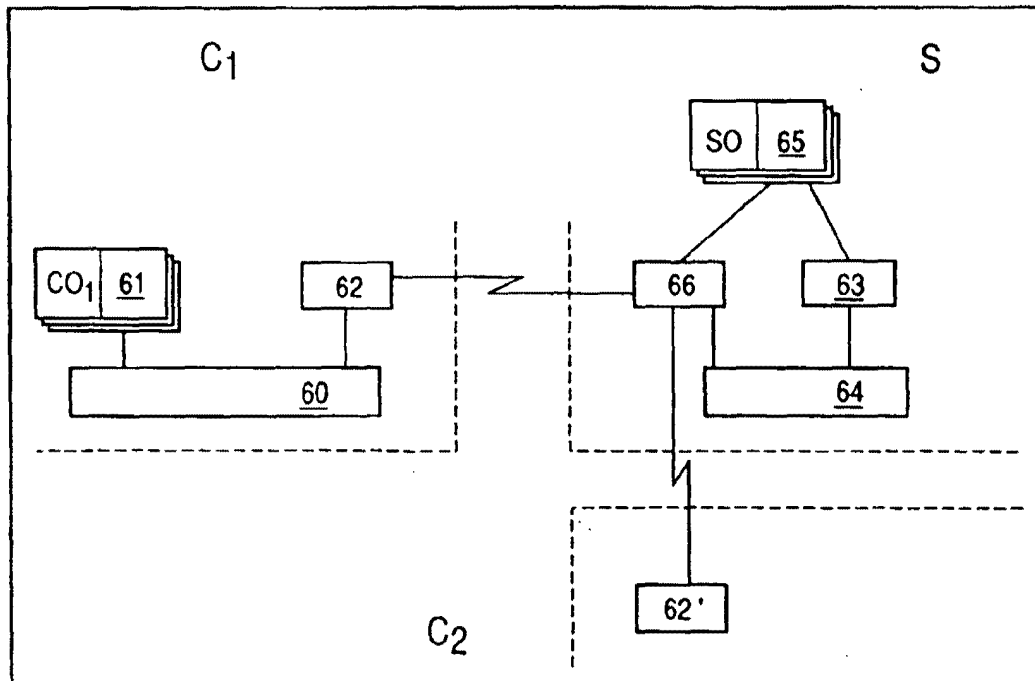


FIG 9

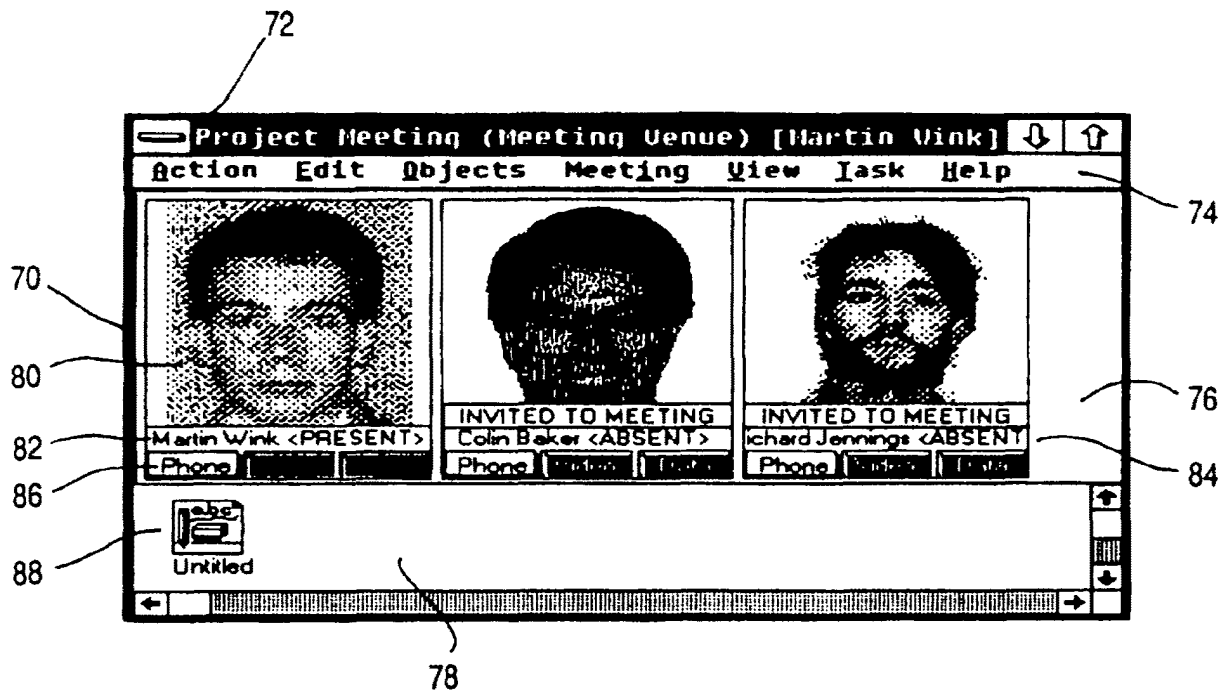


FIG 10

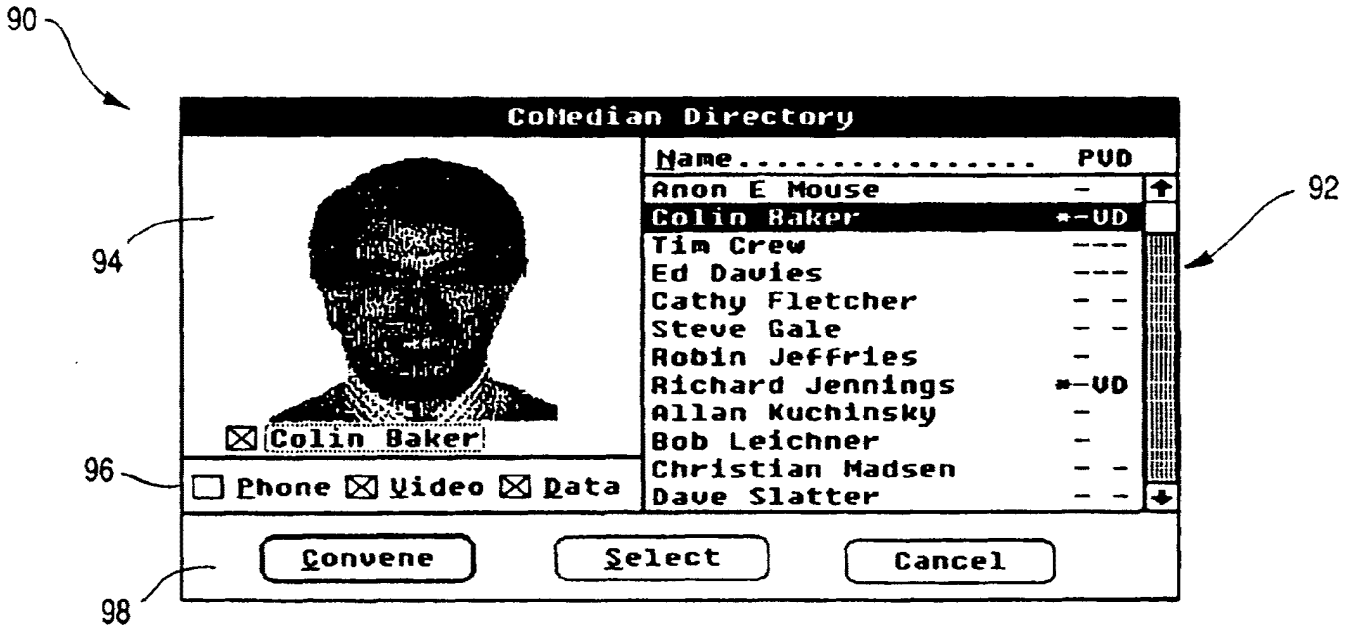


FIG 11

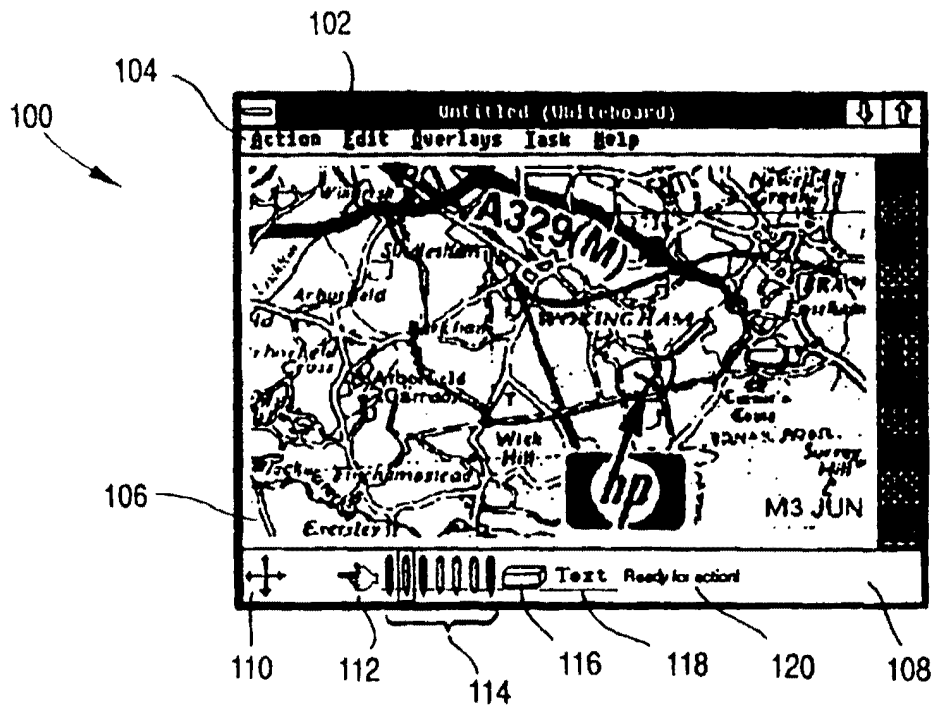


FIG 12



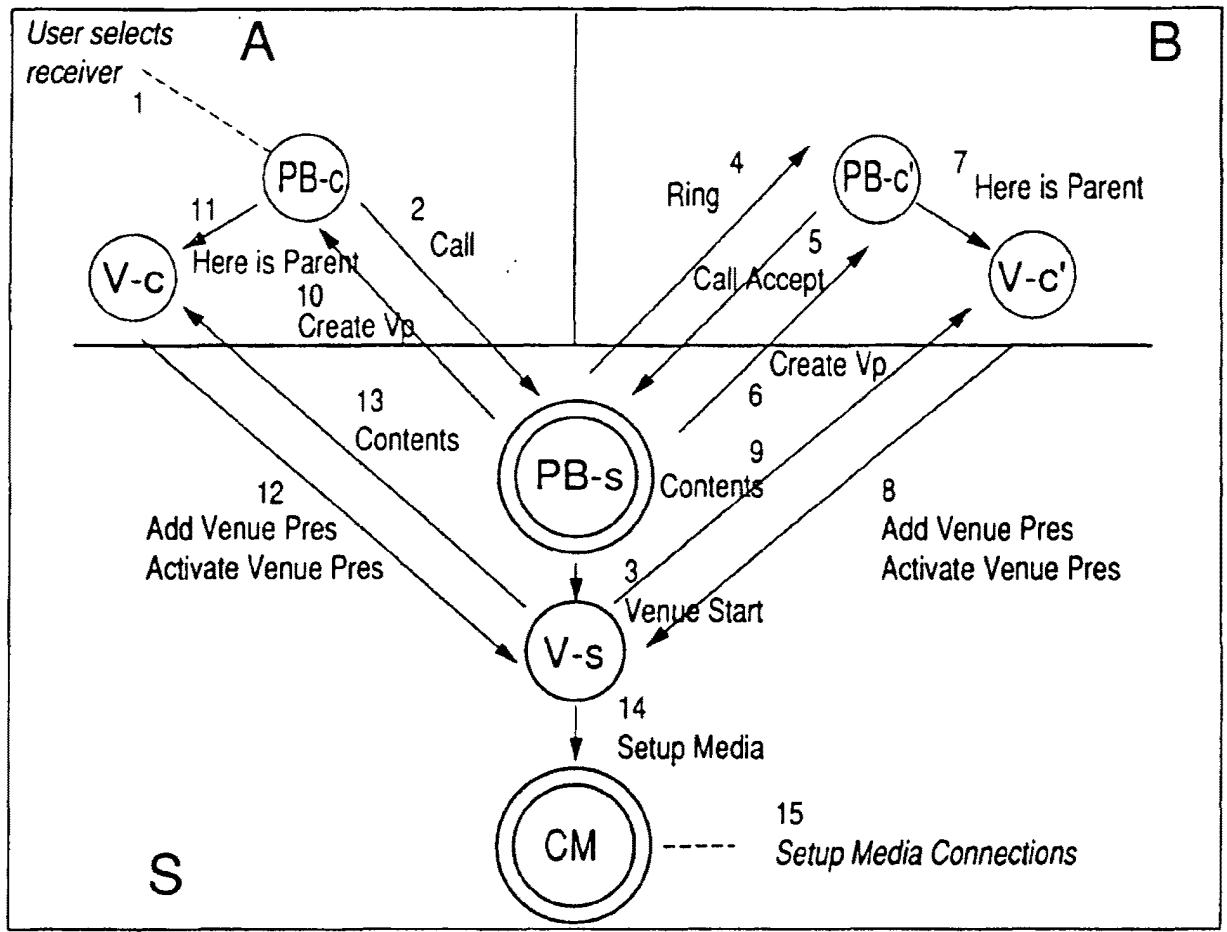


FIG 13

EP 0 497 022 A1

122

SKYPE-N2P00283644

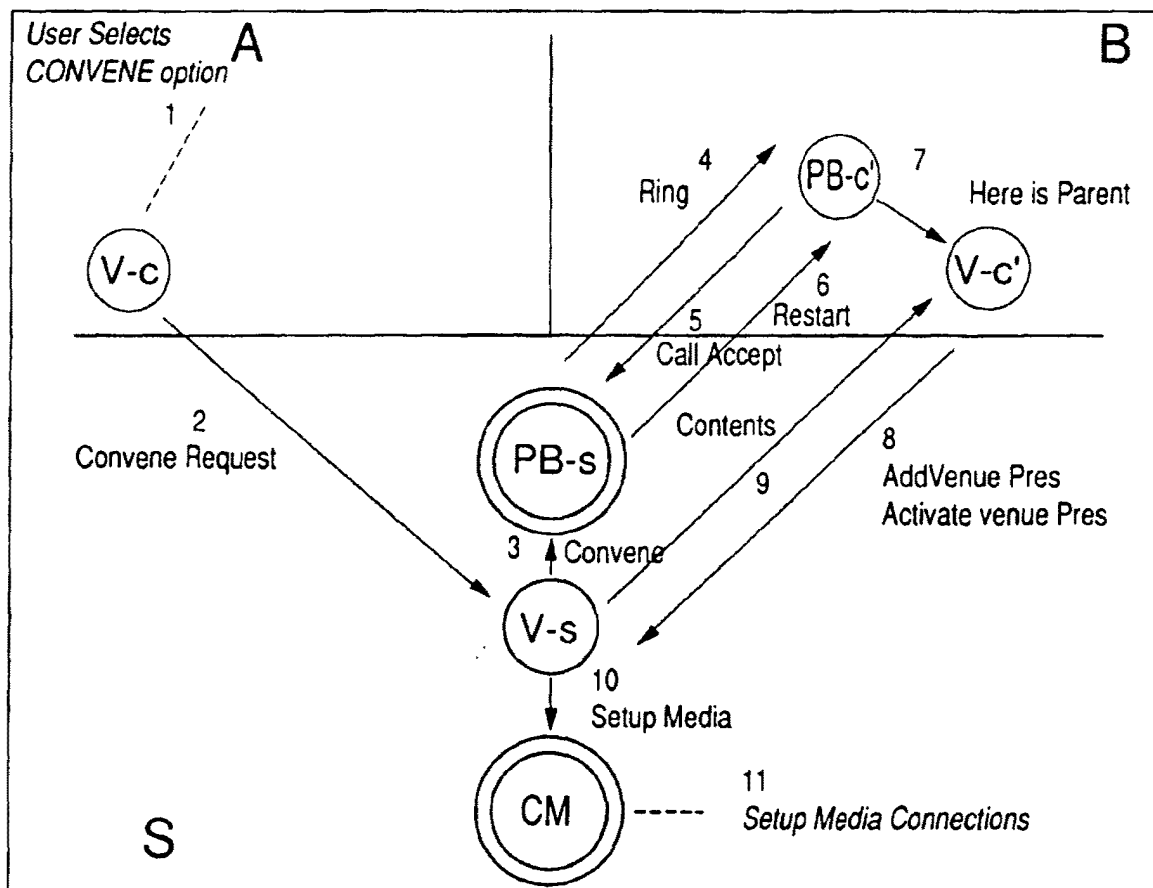


FIG 14

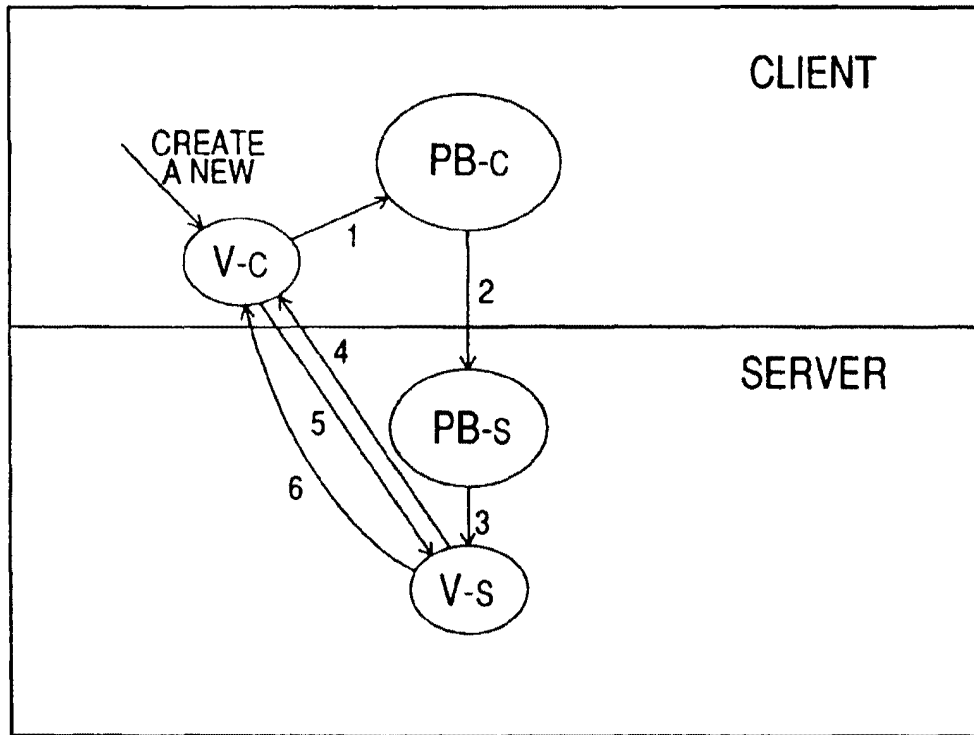


FIG 15

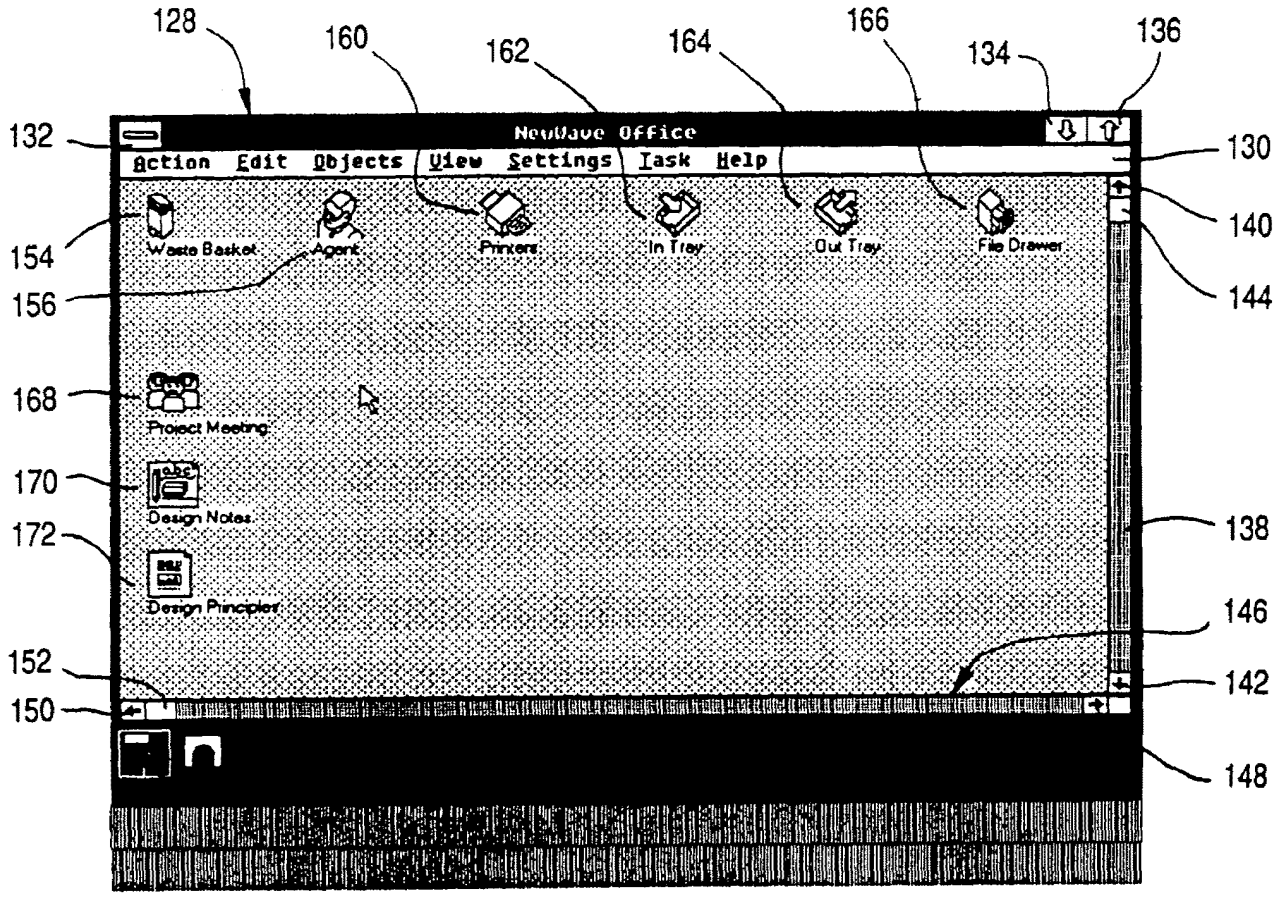


FIG 16

126

125

SK YPE-N2P00283647

EP 0 497 022 A 1

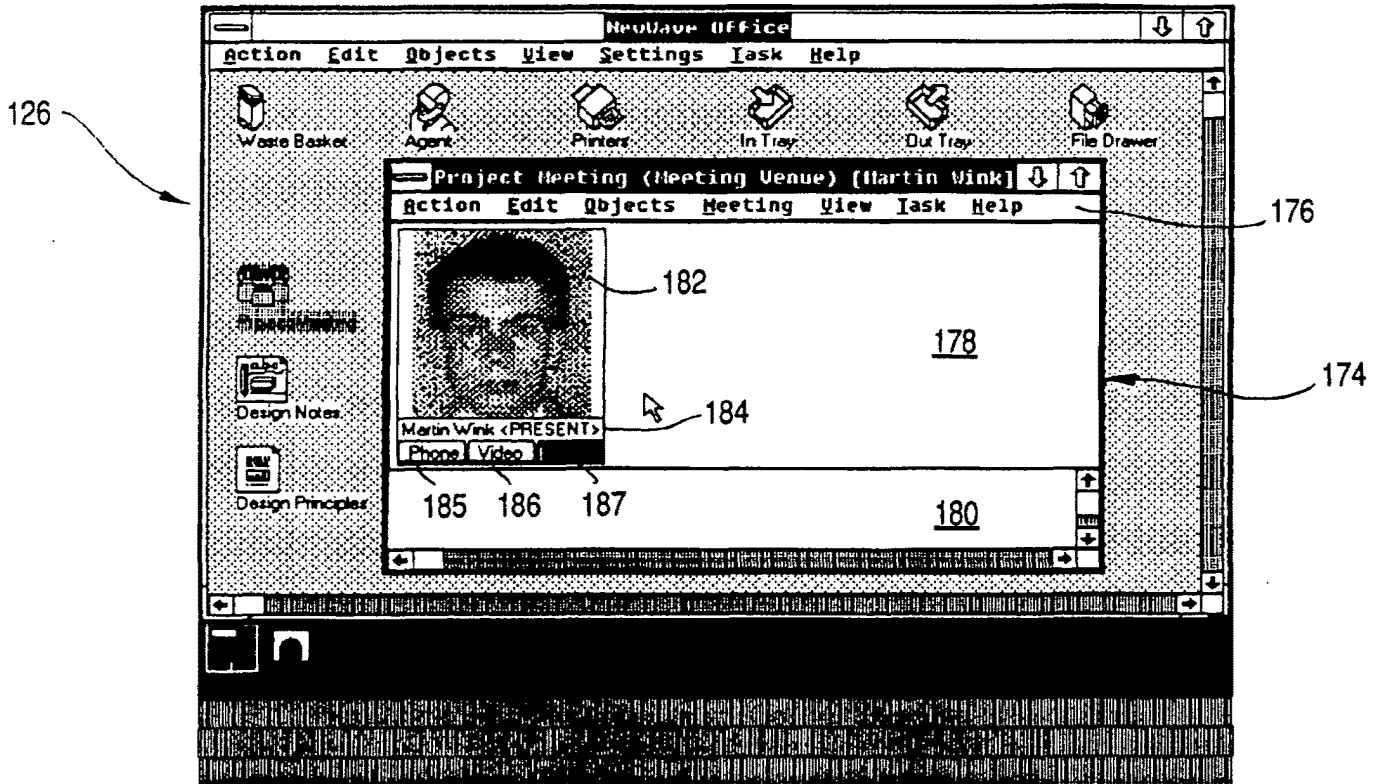


FIG 17

EP 0 497 022 A1

126

SKYPE-N2P00283648



FIG 18

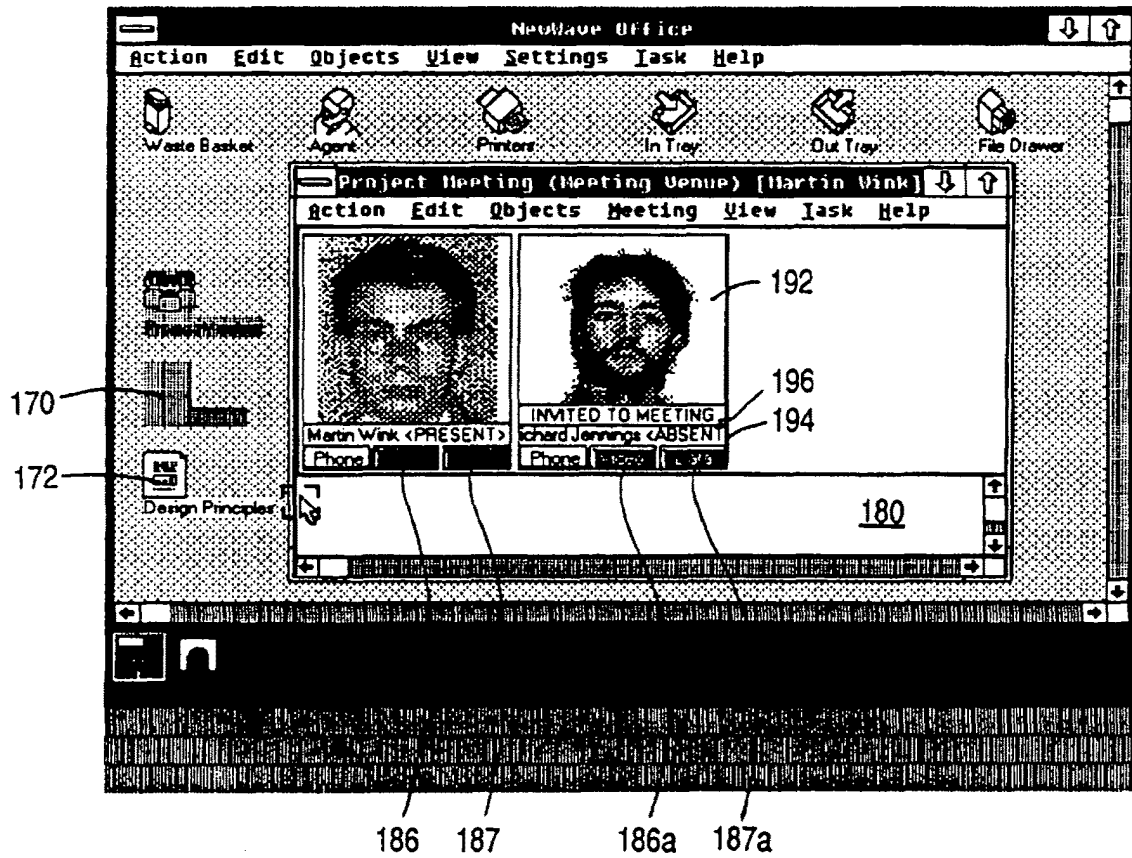


FIG19

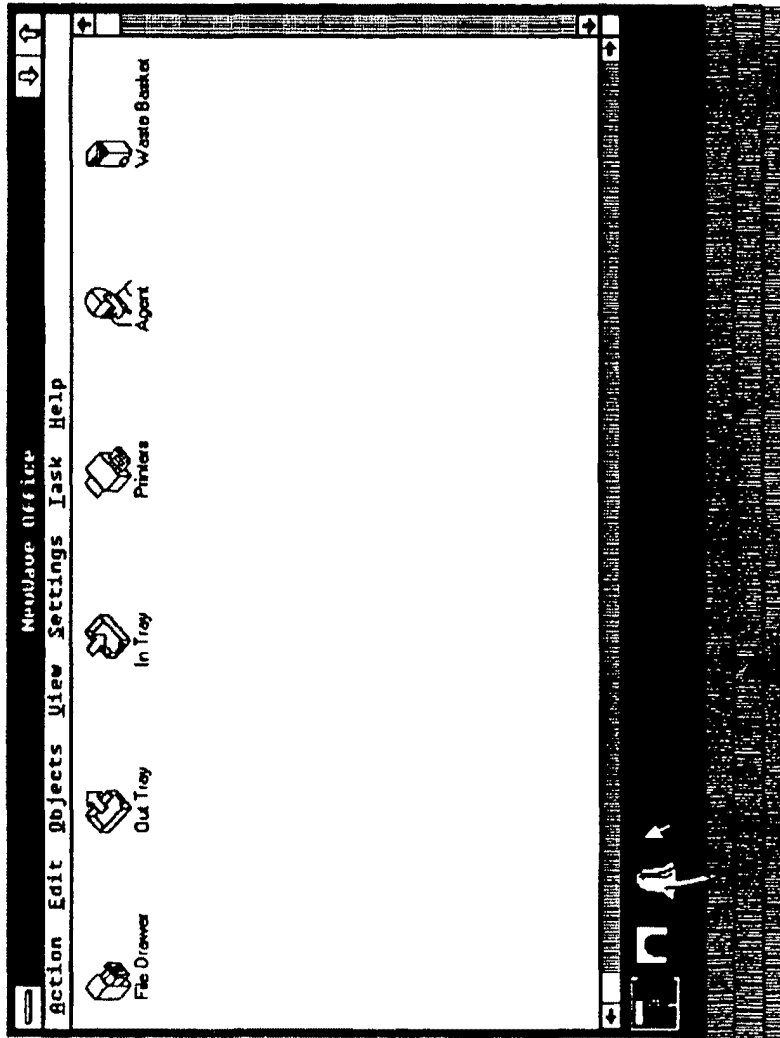


FIG 20



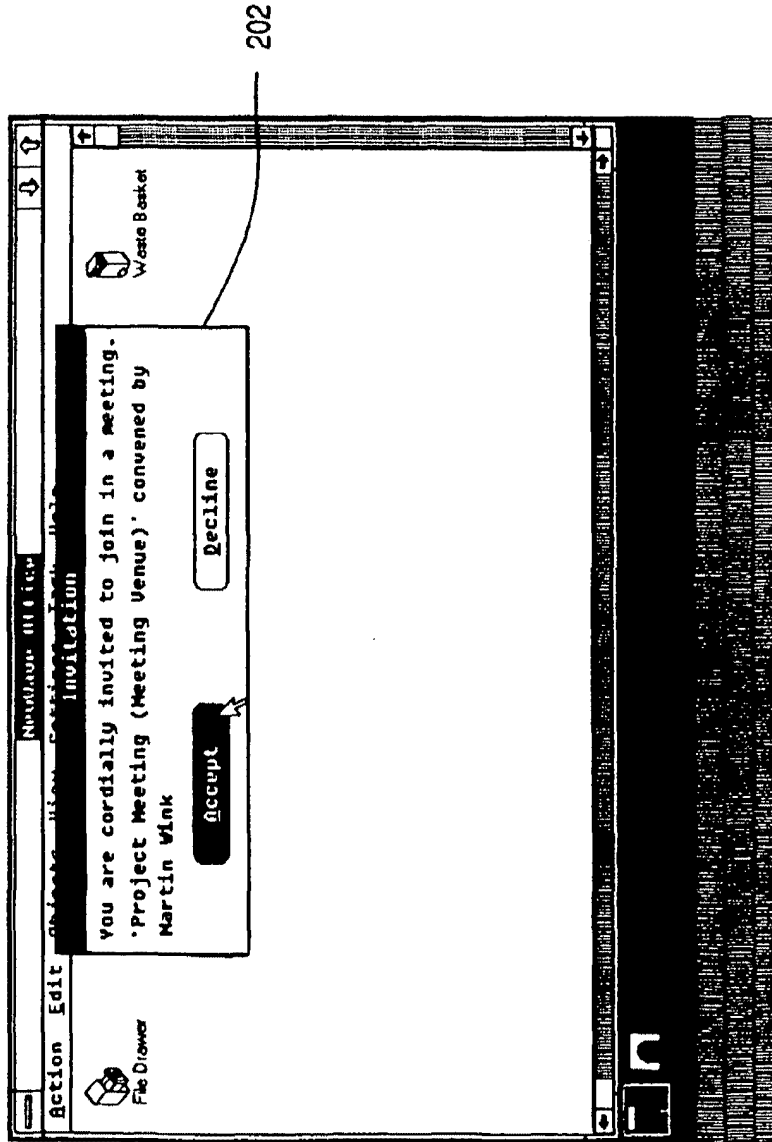


FIG 21

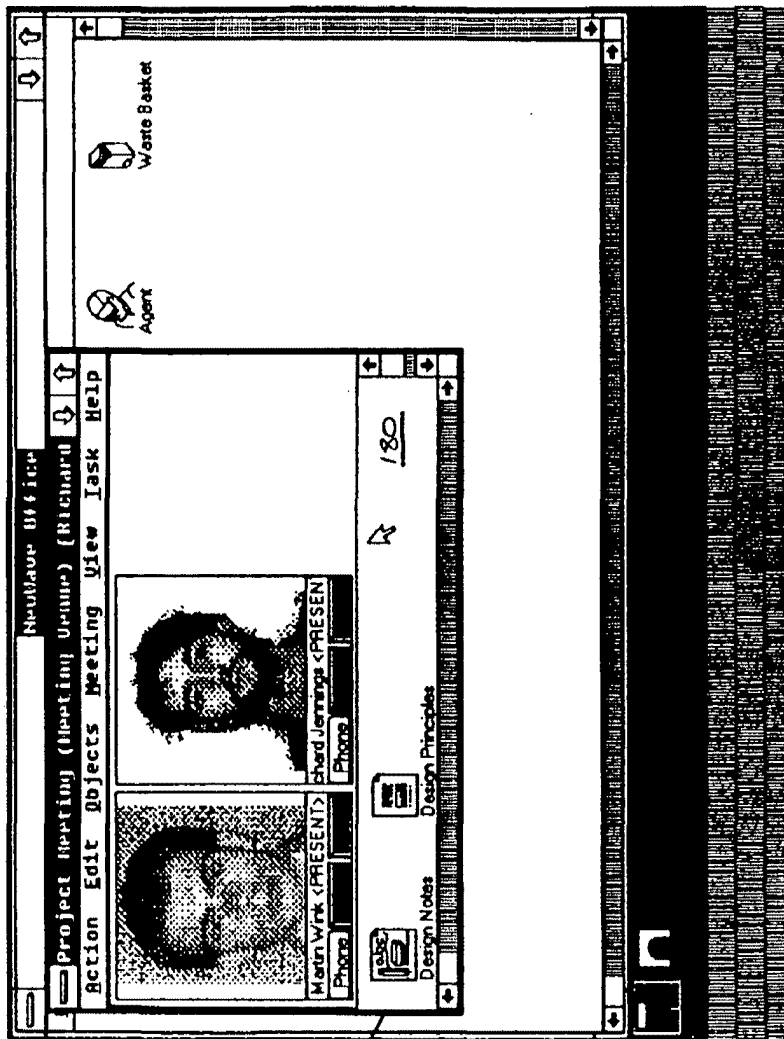


FIG 22

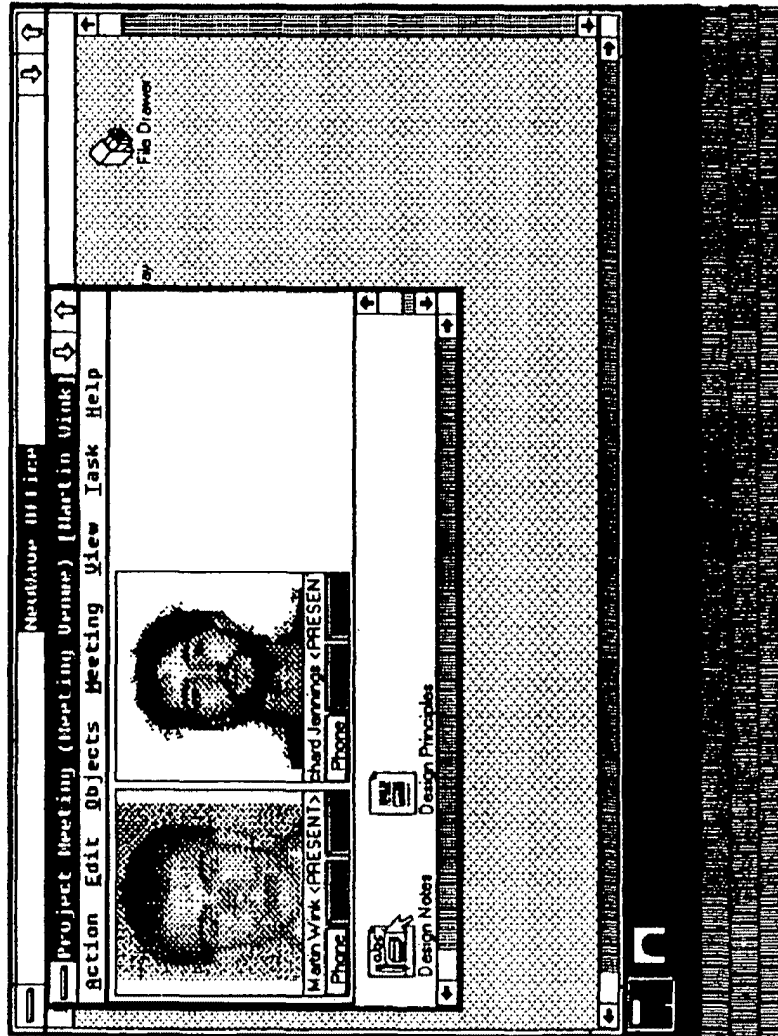
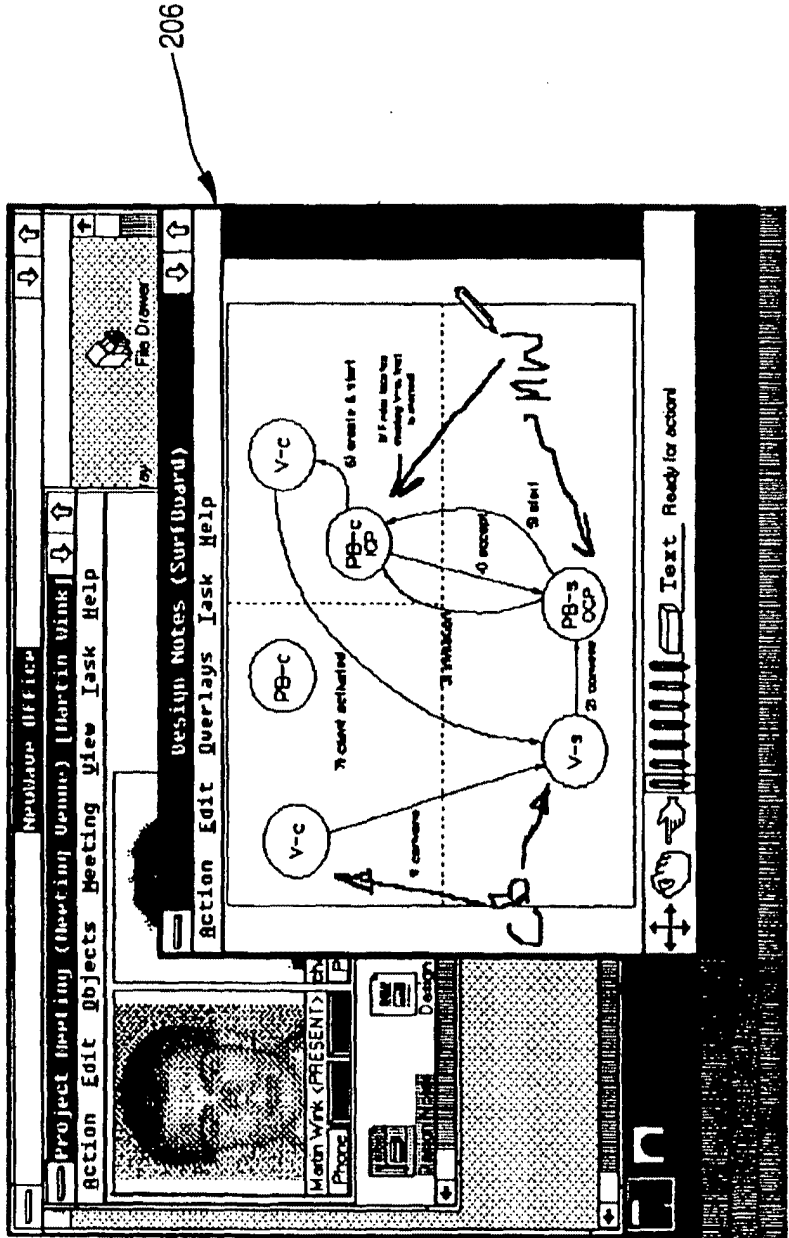
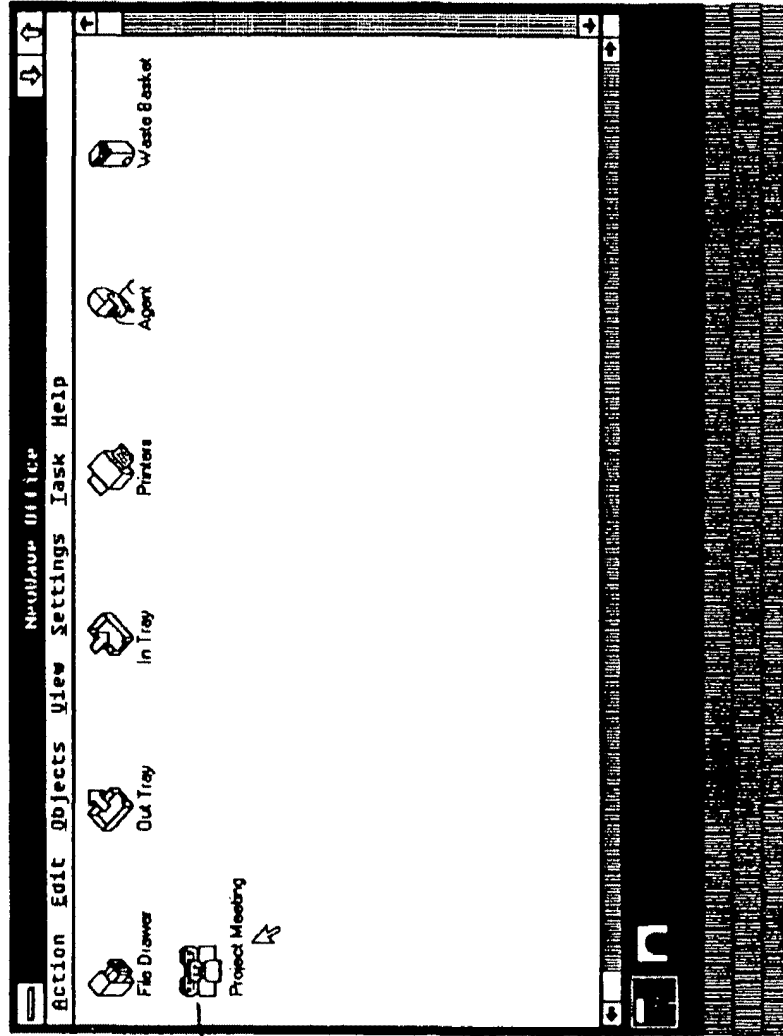


FIG 23



206

FIG 24



168

FIG 25

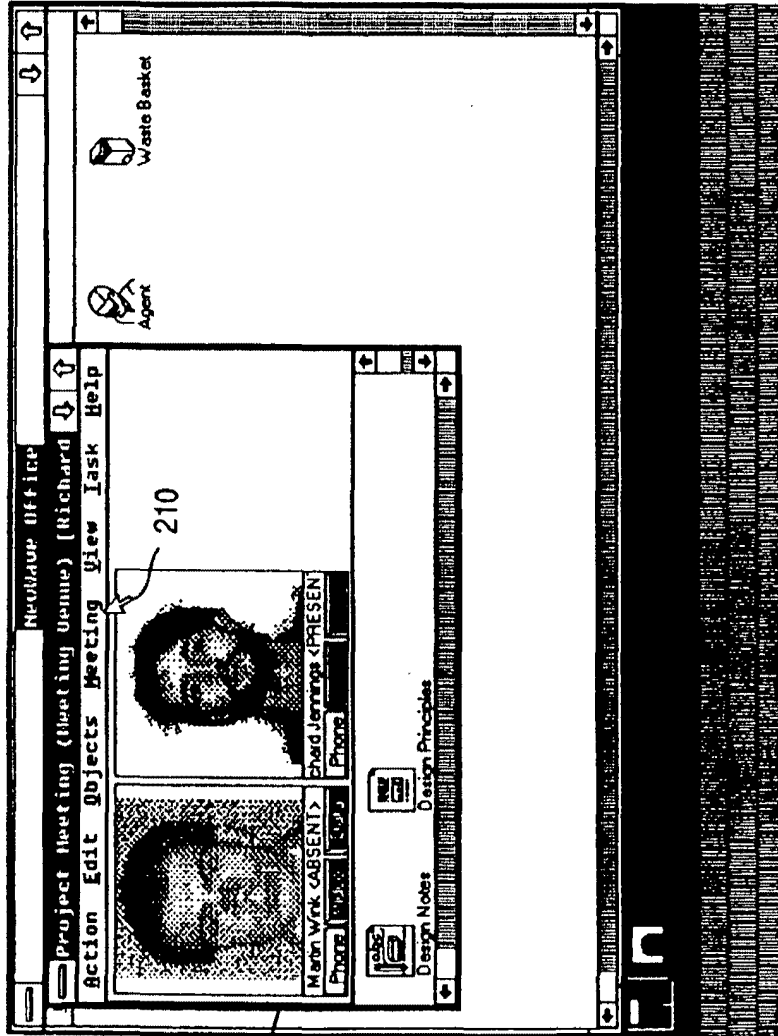
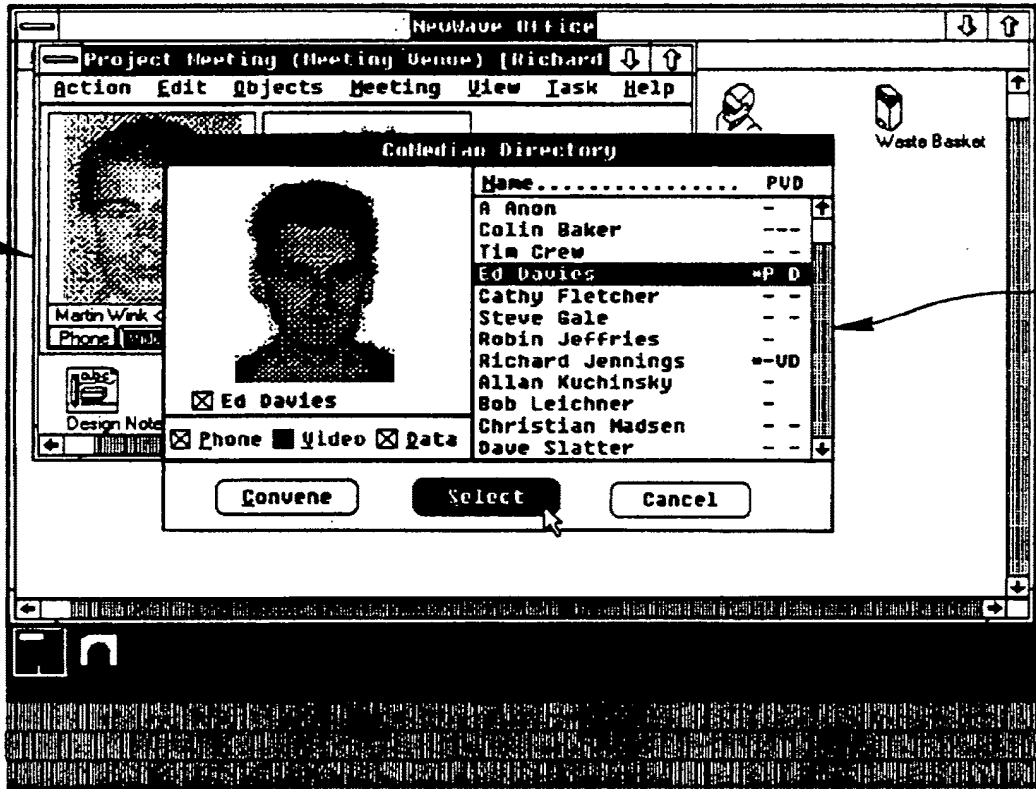


FIG 26



208

212

FIG 27

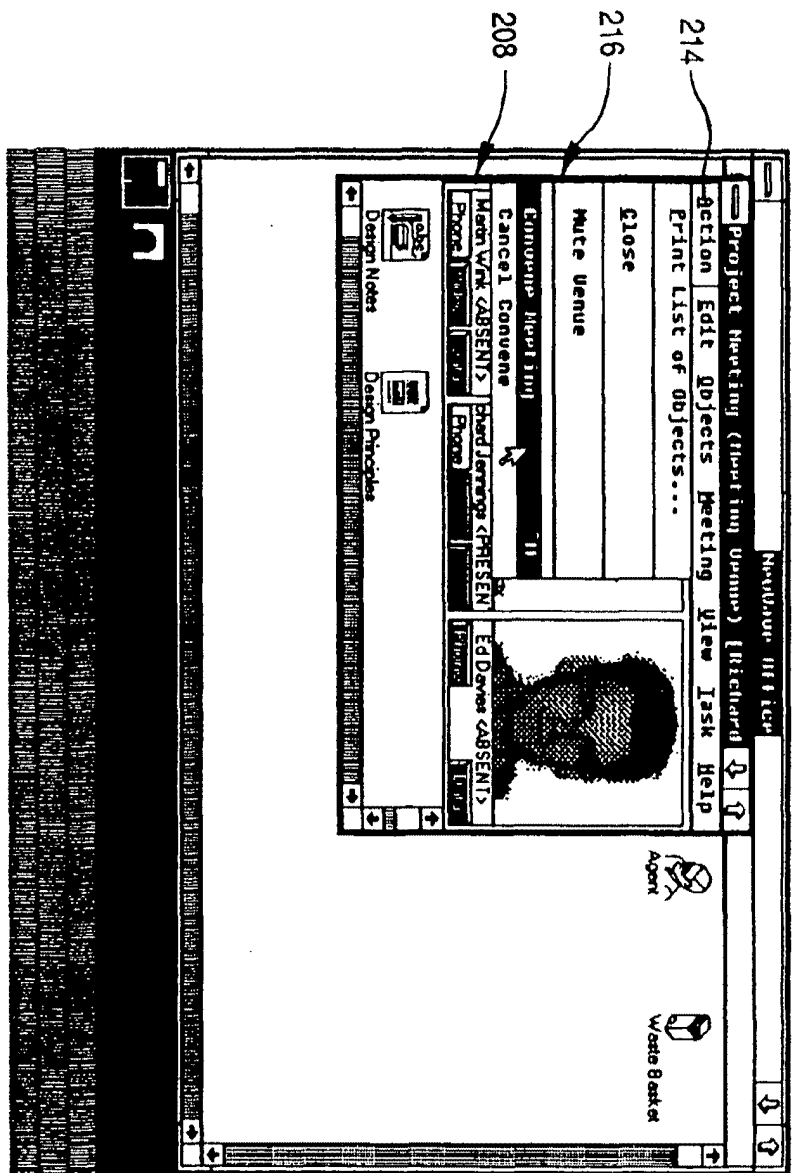


FIG 28



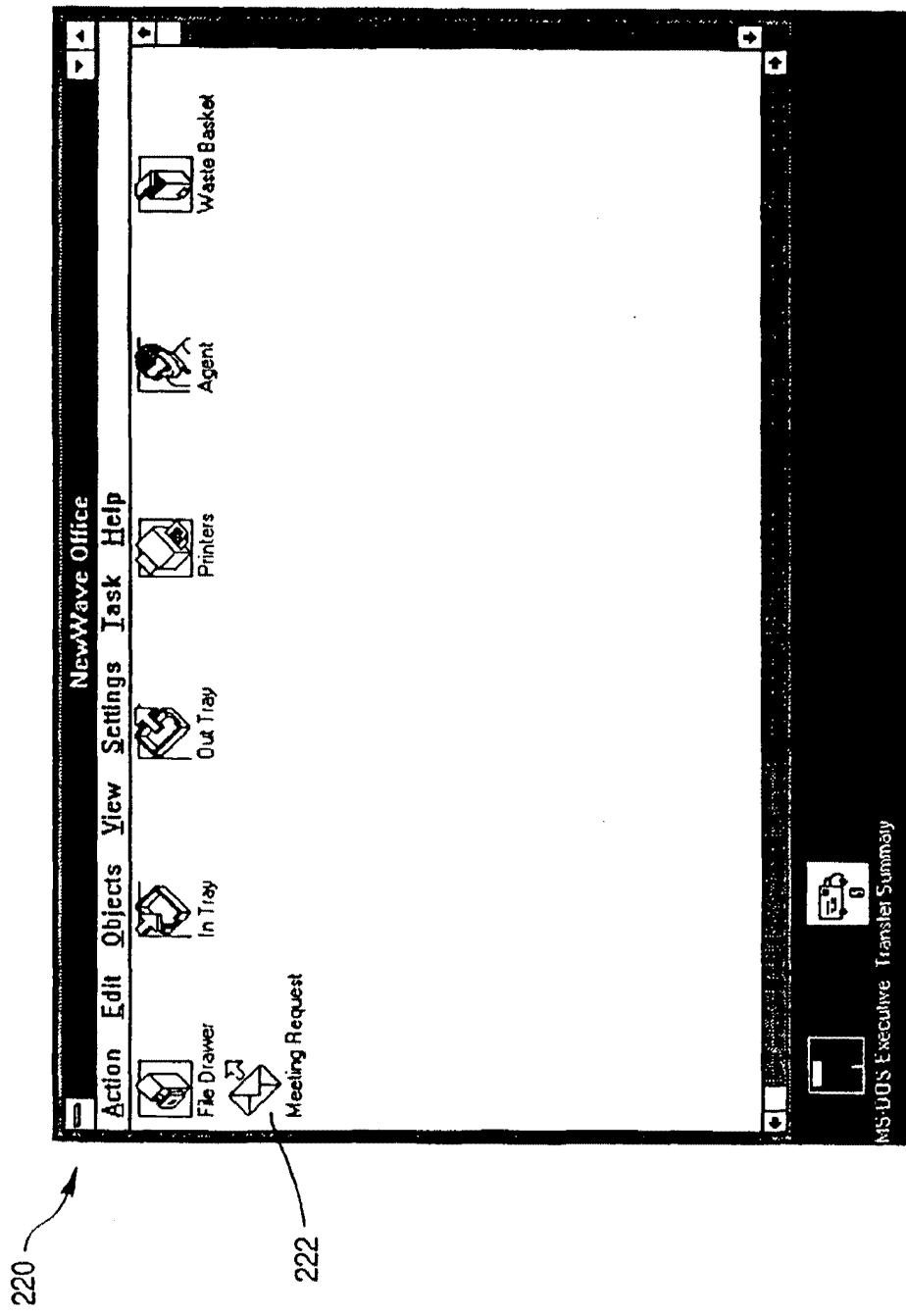
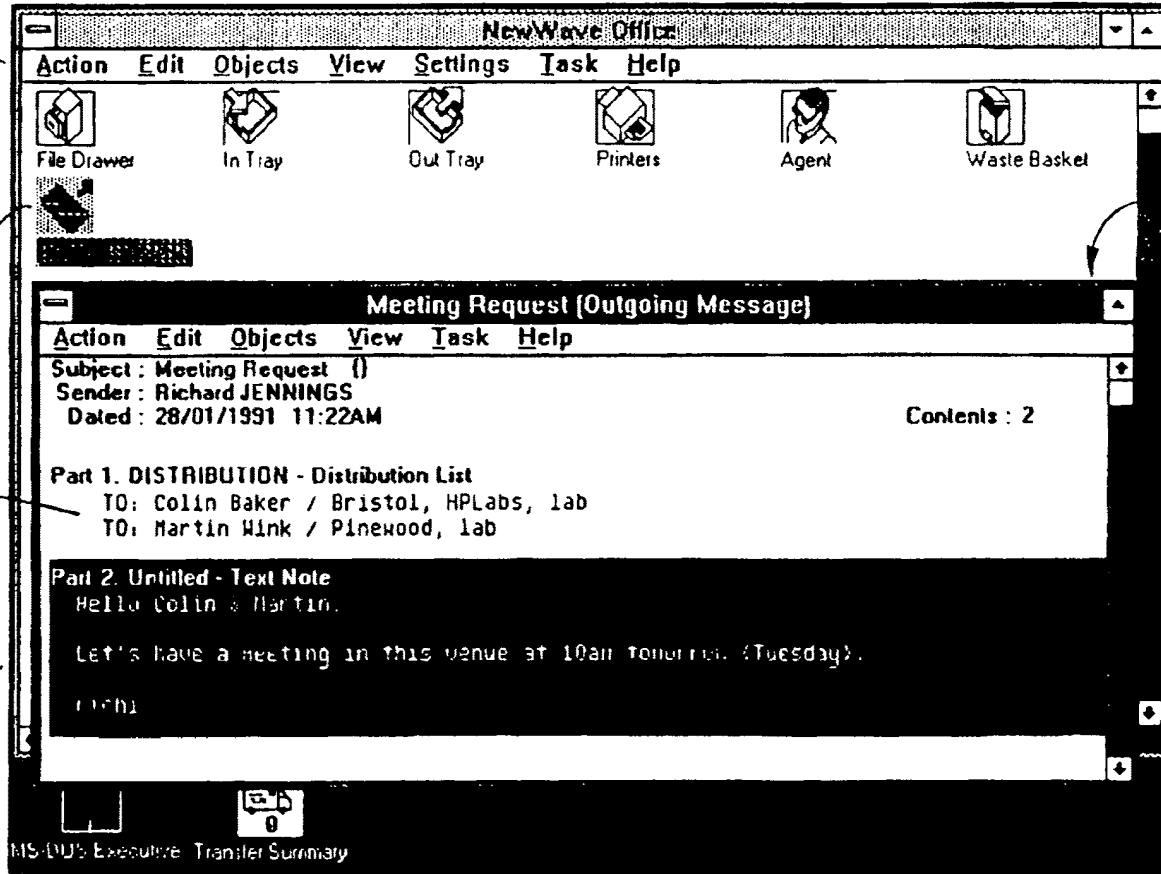


FIG 29

220



222

224

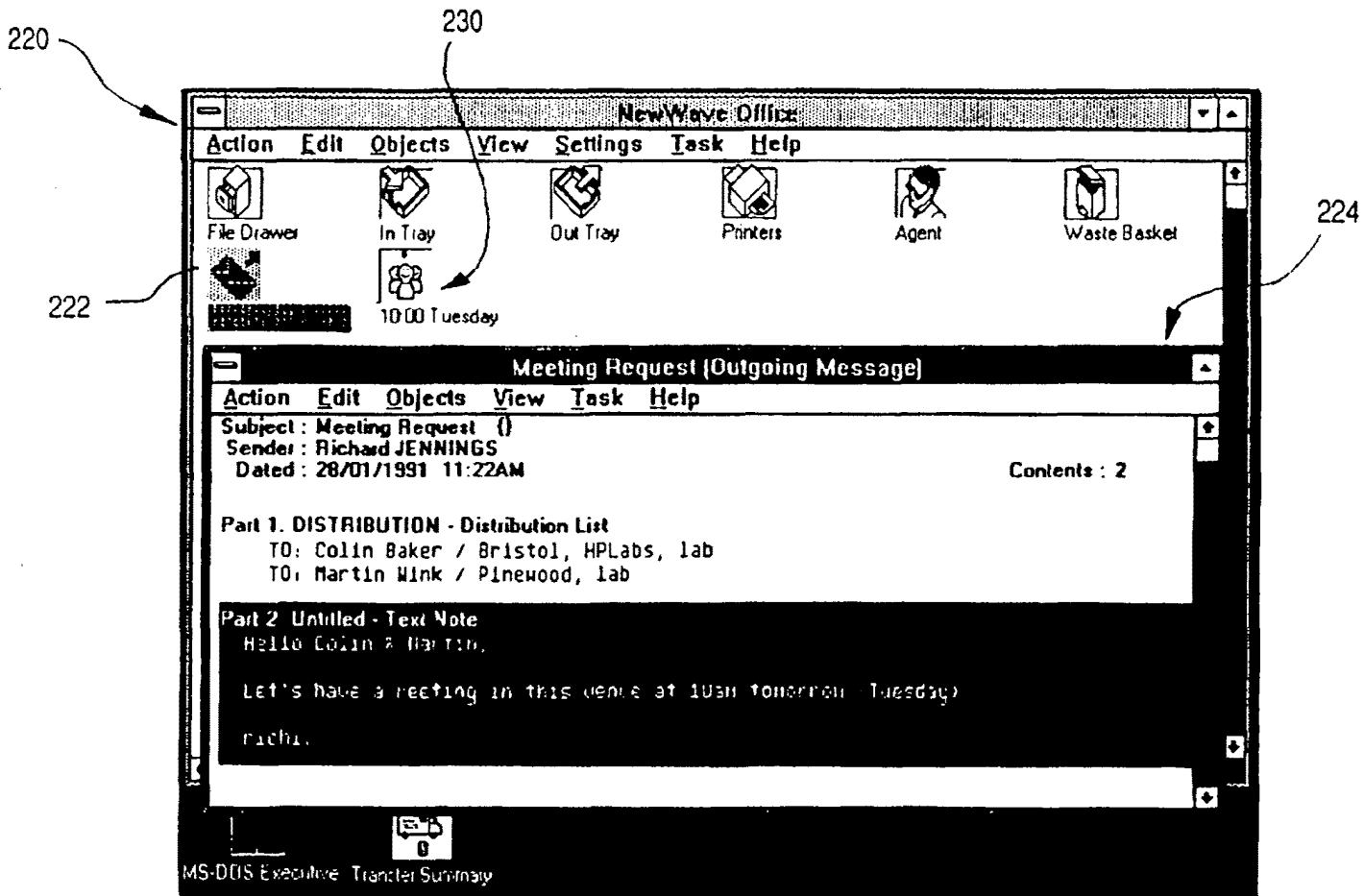
226

228

139

EP 0 497 022 A1

FIG 30



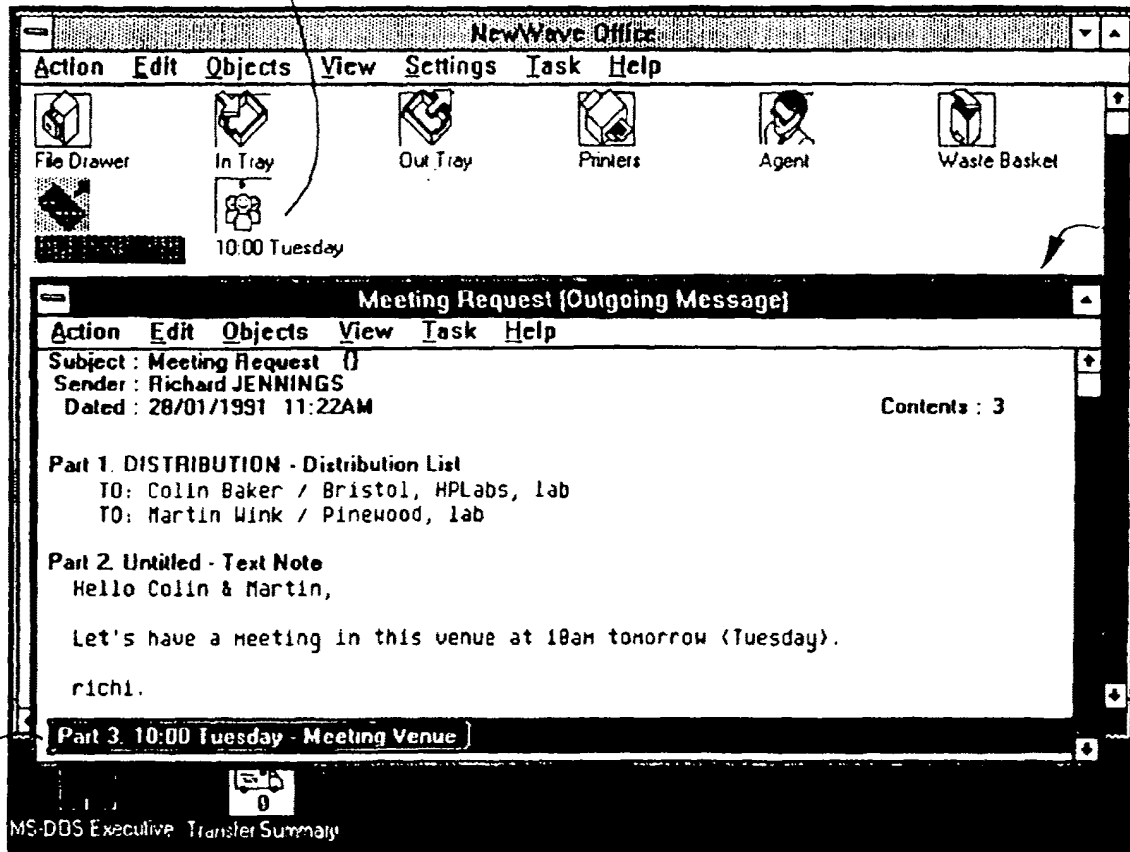
140

EP 0 497 022 A1

FIG 31

220

230



224

232

141

EP 0 497 022 A1

FIG 32

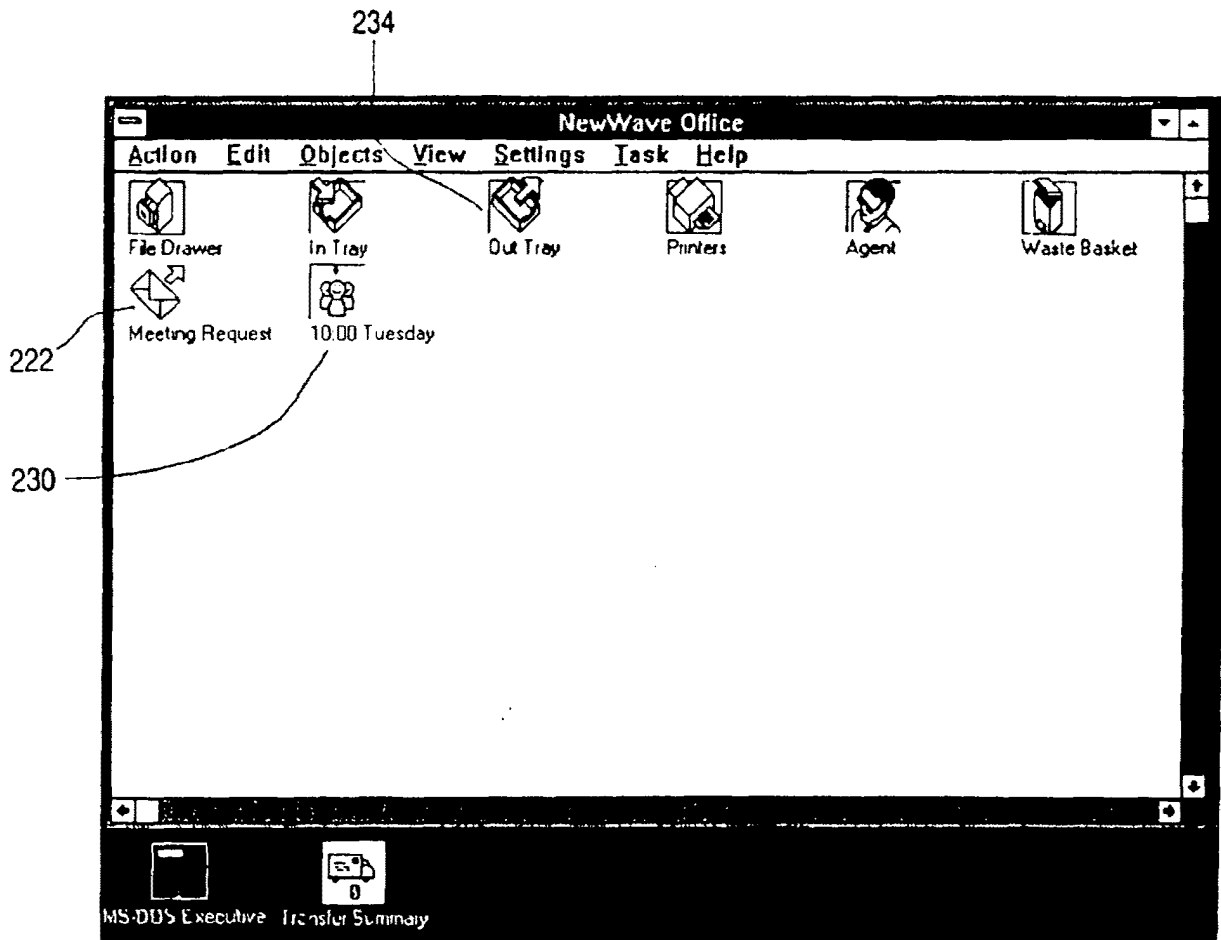


FIG 33

142

EP 0 497 022 A1

SKYPE-N2P00283664



DOCUMENTS CONSIDERED TO BE RELEVANT			
Category	Citation of document with indication, where appropriate, of relevant passages	Relevant to claim	CLASSIFICATION OF THE APPLICATION (Int. Cl.5)
X	EP-A-408 812 (HEWLETT PACKARD) * abstract *	1-3	G06F15/40 G06F9/44 H04M3/56 H04N7/14
Y	* page 3, line 29 - line 51 *	4,5	
Y	COMPUTER. vol. 18, no. 10, October 1985, SILVER SPRING, MARYLAND, US pages 33 - 45; SUNIL SARIN ET AL.: 'Computer-Based Real-Time Conferencing Systems' * page 33, column 1, line 1 - page 38, column 1, line 44 * * page 39, column 3, line 31 - page 42, column 1, line 25 * * page 43, column 1, line 33 - page 44, column 1, line 25 *	4,5	
			TECHNICAL FIELDS SEARCHED (Int. Cl.5)
			G06F H04N H04M
The present search report has been drawn up for all claims			
Place of search THE HAGUE		Date of completion of the search 18 SEPTEMBER 1991	Examiner KINOMA Y.
CATEGORY OF CITED DOCUMENTS		T : theory or principle underlying the invention F : earlier patent document, but published on, or after the filing date D : document cited in the application I : document cited for other reasons A : technological background O : non-written disclosure P : intermediate document & : member of the same patent family, corresponding document	
X : particularly relevant if taken alone Y : particularly relevant if combined with another document of the same category			

EPO FORM 1403 03/82 (P/0401)



12 **EUROPEAN PATENT APPLICATION**

21 Application number: **93630052.4**

51 Int. Cl.<sup>5</sup>: **H04L 12/58**

22 Date of filing: **19.07.93**

30 Priority: **30.07.92 US 922314**  
 43 Date of publication of application:  
**02.02.94 Bulletin 94/05**  
 64 Designated Contracting States:  
**CH DE ES FR GB IT LI NL SE**  
 71 Applicant: **YEDA RESEARCH AND  
 DEVELOPMENT COMPANY, LTD.  
 The Weizmann Institute of Science  
 76 100 Rehovot (IL)**

72 Inventor: **Shapiro, Ehud  
 Meonot Wolfson, Weizmann Institute of  
 Science  
 Rehovot 76100 (IL)**  
 74 Representative: **Waxweiler, Jean et al  
 OFFICE DENNEMEYER & ASSOCIATES Sàrl,  
 P.O. Box 1502  
 L-1015 Luxembourg (LU)**

54 **A method for establishing an interactive communication between users at different workstations in a network.**

57 A method for establishing an interactive communication between at least first and second workstations in a computer network system having a communication protocol for despatching messages between different workstations and being further adapted to exchange batch messages by means of an electronic mail program stored in each of the workstations. The batch messages are categorized such that a batch message of a predetermined category informs a receiving workstation that a sending workstation wishes to establish an interactive communication between a specified first logical port in the sending workstation and a specified second logical port in the receiving station. A batch message of the predetermined category having therein a reference to the first logical port is sent from the first workstation to the second workstation so as to be received thereby and is stored in a storage device containing a list of batch messages. Upon noting the presence in the storage device of a batch message of the predetermined category, the communication protocol is utilized to send an initiation signal from the second logical port in the second workstation to the first logical port in the first workstation. Upon receipt of the initiation signal, an interactive two-way communication is established between the first logical port of the first workstation and the second logical port of the second workstation.

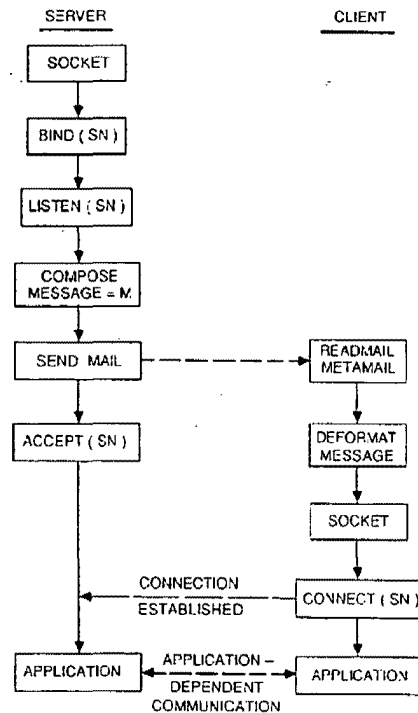


Fig. 5

EP 0 581 722 A1

**FIELD OF THE INVENTION**

The invention relates to the use of "electronic mail" for affording message exchange between computer users. In particular, the invention permits communication to be established between users at different terminals connected to a mini-computer or a main frame operating under a multi-tasking operating system, or between users at workstations or network stations (X-terminals) connected by a communication network. It should be noted that throughout the specification and claims, the term "workstation" is used generally to denote any device for communicating interactively with a computer and including a display monitor and an input device such as a keyboard, mouse and so on.

**BACKGROUND OF THE INVENTION**

In recent years, there has been a constant growth of computer network installations. This, coupled with the spread of minicomputers and main frames connected to a plurality of terminals, have encouraged the use of network orientated applications. A typical such application is "electronic mail", which enables at least two users to exchange messages.

Several standards, such as "X.400" for OSI or "Multipurpose Internet Mail Extension" (MIME) for "Internet" have been defined to permit message exchange of multiple data types. The messages may consist of different authorized types, such as computer-executable files, computer program-processable files (such as spreadsheets), audio and video sequences, or a combination thereof, as in the case of multi-media.

The electronic mail methods used nowadays are of off-line or batched character, whereby a message, e.g. a text file, is sent from User A to User B and stored in User B's data storage device, usually referred to as a "mailbox". User B, at his earliest convenience which may, of course, be some considerable time later, accesses his mailbox to read any pending messages, and so finds User A's text file. He may then respond by sending an acknowledgement message to User A. This simplified protocol demonstrates the off-line character of the connection.

It is possible to re-configure the computer prompt appearing on the screen of User B's computer at the moment the message arrives, whereby User B is provided with immediate feedback that a message is waiting for him. This, in turn, permits User B to generate an immediate response to User A. Nevertheless, the batched character of the communication is retained, since User B's response resides in User A's mailbox, and in order to retrieve it, User A must invoke a series of instructions, including entering the mailbox and selecting therefrom the message whose contents are to be displayed. If he wishes to answer it immediately, he must prepare a message of a type supported by the electronic mail program, and send the message through the network to User B.

It would clearly be preferable to invoke an interactive session in which a message sent by User A is immediately displayed on User B's screen for direct response by User B. Consider, for example, a firm in which each employee uses a PC all of which are interconnected by a standard network. Suppose an employee (User A) sends a draft of an important letter to his boss (User B). The boss, after reviewing the received draft, wishes to establish an interactive communication with User A, and optionally to involve in the discussion the department manager (User C). Obviously, a copy of User A's draft is sent to User C, so as to permit a discussion to be conducted between the three participants.

Alternatively, User A may be replaced by a "groupware application" running on a server. Such a groupware application is shared by several users as in the case of a document edited simultaneously by two or more authors. Conventional groupware applications permit each of the authors to effect simultaneous editing of the document whilst being logged into different workstations. However, suppose that during the course of editing, it is required to involve an additional author who is a specialist in a certain topic discussed in the groupware document. In such case, it is required to establish an interactive session between the groupware application (running on one computer) and the specialist user who is generally logged into a different workstation.

It is clear that the above requirements cannot be realized in currently available electronic mailing systems which, as explained above, are not interactive.

In contrast to batch-type electronic mailing systems, it is also known to provide an interactive on-line communication between users across a computer network. Thus, for computers operating under the UNIX operating system, there is provided a facility "TALK" whereby such interactive communication may be achieved between several users. However, "TALK" and other similar interactive communication methods are intrusive since only the user who establishes the communication has control as to when the communication is to be established, whilst all of the remaining users are likely to be disturbed during the performance of other tasks. Thus, typically, if User A invokes the "TALK" facility in order to initiate an interactive communication with User B, a message flashes on the screen of User B's workstation in order to inform him that User A wishes to establish



a communication. If User B is otherwise preoccupied and ignores the message, it will reappear at regular intervals until finally User B also invokes the "TALK" facility in order to communicate with User A. Until such communication is established, the constant reappearance of warning-type messages on the screen of User B's workstation is inevitably intrusive and can be most irritating to User B.

5 Furthermore, if User B repeatedly ignores the messages which appear on his screen advising him of User A's desire to establish an interactive dialogue and User A responds in kind, by exiting from the "TALK" facility, no trace of the previous attempts to establish such dialogue is left in User A's workstation. Thus, if and when User B is eventually free to respond to the message originally dispatched by User A, there exists no trace on User A's workstation of his original attempts to establish such communication and it is thus now User B, and  
10 no longer User A, who must initiate the communication.

Additionally, facilities such as "TALK" are intended only for invoking interactive communications between users working at respective workstations and have no provision for establishing such communication between a user and an application or between two applications such as, for example, a groupware document being edited simultaneously by different users at respective workstations.

15

### BRIEF SUMMARY OF THE INVENTION

It is an object of the invention to provide a method for establishing an interactive dialogue between two or more workstations in such a manner as to preserve the non-intrusive character of batch-type electronic mail systems, whilst nevertheless permitting an interactive multi-way communication to be conducted between the participants.

According to the invention there is provided for use in a computer network system comprising at least first and second workstations adapted to send and receive messages by utilizing a suitable communication protocol and further adapted to exchange batch messages by means of an electronic mail program stored in each of  
20 said at least first and second workstations;

25 a method for establishing an interactive communication between said at least first and second workstations, said method characterized by the steps of:

(i) categorizing said batch messages such that a batch message of a predetermined category informs a receiving workstation that a sending workstation wishes to establish an interactive communication between a specified first logical port in the sending workstation and a specified second logical port in the receiving workstation;

(ii) sending a batch message of the predetermined category having therein a reference to said first logical port from the first workstation to the second workstation so as to be received thereby and stored in storage means containing a list of batch messages;

35 (iii) monitoring at the second workstation all batch messages in said storage means at specified periods of time;

(iv) noting the presence in said storage means of a batch message of said predetermined category;

(v) utilizing the communication protocol to send an initiation signal from the second logical port in the second workstation to the first logical port in the first workstation; and

40 (vi) responsive to receipt of the initiation signal, establishing an interactive two-way communication between the first logical port of the first workstation and the second logical port of the second workstation.

In accordance with such a method, the message may be sent directly from an application running in the first workstation for subsequent storage in the mailbox in the second workstation. Upon scanning the mailbox in the second workstation, the user finds a message of the predetermined category, indicating that another  
45 user or application on the network wishes to establish a two-way interactive communication with him.

In normal batch-orientated electronic mail systems, each message in a user's mailbox has a corresponding title, by means of which the awaiting message can be identified. It is preferable to embed within the title of the awaiting message some indication that the message is adapted for establishing a two-way interactive communication with a sending workstation. Alternatively, this fact may not be apparent from the title of the message  
50 itself, in which case the receiving workstation will not afford the awaiting message any special priority, although the very act of reading the message will invoke the required interactive communication.

According to a preferred embodiment, the method is used in order to establish an interactive communication between a groupware application running on the first workstation and at least one user working at a second workstation who accesses the groupware application via a suitable interface window. In such a system, the  
55 interface window at the second workstation is associated with the second logical port thereof so that when the desired two-way interactive communication is established between the first and second logical ports of the first and second workstations, respectively, the groupware application running on the first workstation will interact directly with the user via the interface window of the second workstation.

**BRIEF DESCRIPTION OF THE DRAWINGS**

For a clearer understanding of the invention and to see how the same may be carried out in practice, some preferred embodiments will now be described, by way of non-limiting example only, with reference to the accompanying drawings, in which:

**Figs. 1a and 1b** are block diagrams showing functionally a computer network system and a detail of a workstation thereof for implementing an electronic mail method according to the invention;

**Fig. 2** is a simplified flow diagram showing the principal operating steps associated with a sending workstation in the system shown in Fig. 1;

**Fig. 3** is a simplified flow diagram showing the principal operating steps associated with a receiving workstation in the system shown in Fig. 1;

**Fig. 4** is a flow diagram showing in somewhat greater detail the operating steps shown in Fig. 3; and

**Fig. 5** is a composite flow diagram showing the principal operating steps of the system depicted in Fig. 1 operating under UNIX and utilizing the Internet communication protocol.

**DETAILED DESCRIPTION OF PREFERRED EMBODIMENTS**

Figs. 1a and 1b show a computer network system 10 comprising a server 11 coupled, via a communication network depicted generally as 12 to a plurality of client computers 13 to 17, respectively. Each of the computers 13 to 17 constitutes a workstation associated with which is a storage device 19 and a display device 20. Typically, each of the computers 13 to 17 is adapted to run several tasks simultaneously, data associated with each task being displayed on the display device 20 in a corresponding window 21 thereof.

In accordance with a preferred embodiment, the communication network 12 utilizes the Internet standard as a communication protocol and further utilizes the so-called "Multipurpose Internet Mail Extension" (MIME) for the Internet standard. MIME provides means for exchanging messages between users in an Internet-oriented communication network. The messages may be one of several different categories such as, for example, computer executable files, computer program processable files (such as spreadsheets), audio and video sequences or a combination thereof. The MIME standard permits the definition of an application's specific category. This feature is exploited by the invention for defining a unique category to represent an electronic mail message which is configured to establish a two-way interactive communication between the server 11 and one or more of the client computers 13 to 17.

Typical mail exchange under the MIME standard between computers inter-connected in a network as shown in Fig. 1a is implemented as follows. The server 11 as well as each of the client computers 13 to 17 has access to a stored mail reader program. Upon receipt of a mail message sent from a user of one of the client computers, the received message is stored in a so-called "mailbox" in the storage device 19 associated with the receiving computer. The user of the receiving computer reads the stored message on the display device 20, and by doing so he activates a mail reader program which, in turn, calls a metamail program which assumes that the messages stored in the mailbox are in a format which conforms to the MIME standard.

Once the user has selected the desired mail message to be read, the metamail program accesses the desired message and retrieves therefrom the message category. It will be recalled that the message category typically defines the type of data associated with the transmitted message.

A list of authorized categories is stored in a database file designated as "mailcap" which instructs the metamail program what action to perform with respect to each of the authorized categories. Thus, if the message is a text file category, then a suitable entry in the mailcap file may cause a standard text editor to be invoked on the user's screen. If, on the other hand, the authorized category denotes that the message is an audio file, then a suitable entry within the mailcap file will specify an audio program which can play back the audio message.

In accordance with the invention, a new authorized category is stored in the mailcap denoting that a batched message associated therewith serves the purpose of establishing an interactive communication between a specified first logical port in the sending workstation and a window 21 in the receiving workstation bound to a specified second logical port. In the description which follows the new, authorized category will be denoted by the term "Active Mail". Associated with the Active Mail category is a procedure which performs a series of steps for establishing an interactive communication between the first and second logical ports in the sending and receiving workstations, respectively.

Referring now to Figs. 2 and 3 of the drawings, there are shown simplified flow diagrams relating to the principal operating steps associated with the establishment of an interactive communication by means of electronic mail in accordance with the modified MIME standard. Throughout the following explanation, it will be assumed that a groupware application running on the server 11 wishes to establish an interactive communi-

cation with the client computer 13. In a manner that will be described in detail below, the network address of the logical port in the server 11 is encoded and embedded in a suitably constructed mail message which further includes the message category.

5 Thus in accordance with the terminology introduced above, the message is categorized as "Active Mail" and specifies the name of the sending socket. The thus encoded message is then sent to the client computer 13 where it is stored in the mailbox thereof. Thereafter, the user working at the client computer 13 scans his mailbox and reads the awaiting message using the electronic mail program in the normal way. The awaiting message may, or may not, be flagged as being of type Active Mail in the title by means of which it is identified in the mailbox, thereby prompting the user to read the message contents with some urgency.

10 In any event, on reading the awaiting message, the message category is decoded and the mailcap is accessed in order to determine the operating instruction which must now be invoked responsive to a message category of the decoded type.

Referring to Fig. 4, this step will now be elaborated on assuming that the relevant instructions stored in the mailcap is a program called "am connect".

15 Initially, the "am connect" program decodes the network address of the logical port in the server 11 to which a connection is to be established and which is embedded in the received message. Thereafter, a suitable logical port is defined in the receiving workstation for communicating with the decoded network address of the logical port in the server 11. This is followed by a series of operating system primitives which open connection between the two network addresses of the respective logical ports.

20 Upon completion of the open communication, there exists a bi-directional communication channel between the respective network addresses of logical port in the server 11 and that in the client workstation 13. The "am connect" program may now open a window on the screen of the client workstation 13 for displaying and mediating an interactive communication between the user of client workstation 13 and the groupware application running on the server 11. Obviously, the application which mediates between the user of client workstation 13 and the groupware application running on the server 11 may independently conduct the interactive communication once the communication channel has been established. In such case, the "am connect" program may terminate upon establishment of a successful connection.

25 Referring now to Fig. 5, the situation described above generally with respect to Figs. 2 to 4 of the drawings will be described with particular reference to the UNIX operating system and with regard to a communication protocol and an electronic mail system which conform with the "Internet" and "MIME" standard, respectively. Suitable program modules written in the computer programming language "C" for carrying out the routines shown functionally in Fig. 5 are included in an Appendix hereto.

30 Before describing how an interactive communication is established using a suitably modified electronic mail program operating under UNIX, a further consideration should be understood. In UNIX, the logical ports are implemented using the so-called "socket" mechanism. The socket mechanism enables a workstation having one global Internet address to support a plurality of tasks. In order to distinguish between the various tasks running on the same workstation, a unique global identifier is associated with each of the running tasks so as to render the task identifiable by other tasks running on the network. Thus, a message which is sent to a specific task in a workstation has to encode both the physical destination address of the receiving workstation, as well as the logical port ("socket") which identifies the application running thereon.

35 In other words, since the UNIX operating system is a multi-tasking environment, it is not enough to define only the destination address of the receiving computer: a logical port associated with a specific task or application must also be specified. The combination of the physical address and the logical port is referred to as a "network address of the logical port". In the terminology of UNIX and Internet, a network address of the logical port is referred to as a "socket name".

40 The following description assumes that communication between two workstations employs the Internet "stream" communication protocol. However, it will be apparent that other protocols may equally well be employed such as, for example, the Internet "datagram" protocol and so on.

45 It should further be understood that, in accordance with the UNIX operating system, once a connection is established between respective sockets in different workstations, there exists a logical bidirectional connection between the sockets, whereupon the application which interacts through the respective socket may refer thereto as if it were a standard output or input stream. Thus, if a connection is established between a first socket in a first workstation and a second socket in a second workstation, the application associated with the first socket may interact with the application associated with the second socket simply by invoking "WRITE" or "READ" instructions. The underlying communication layers in the operating system structure will take care to ensure that the message is properly routed to the required destination.

50 It should also be added, for the sake of completeness, that the socket mechanism is well known to those versed in the UNIX operating system and is therefore not discussed in greater detail.

In the description which follows, whenever a service supported by the UNIX operating system is invoked, the service name will be symbolically indicated, omitting, for the sake of simplicity, reference to any parameters transferred to the service or received therefrom. The precise syntax for calling the UNIX services is familiar to those skilled in the art and likewise, since the services themselves are not a specific feature of the present invention, a detailed description thereof is unnecessary.

At the outset, the groupware application running on the server 11 invokes the "socket" primitive supplied by the operating system in order to obtain a socket number which will be associated therewith. This having been achieved, it is required to bind the socket number to a socket name (SN) and this is achieved by the "bind" service which is fed with the socket number obtained as a parameter in the previous stage. The "bind" service performs a series of steps, some of which are responsible for the binding of the logical socket to the global Internet address. To this end, the standing groupware application has a defined logical port or socket bound to which is a global Internet address through which communication with another workstation in the network may be established.

Thereafter, the "listen" service is called, this being responsible for controlling the number of simultaneous communication acknowledgements that the server 11 can handle. Upon completion of the initialization phase, the groupware application running on the server 11 prepares an Active Mail message which conforms to the MIME standard. The message is categorized as "Active Mail" and has embedded therein the encoded socket name obtained by the previous step.

The message is now sent across the network to its destination, i.e. client workstation 13. The server 11 now performs the "accept" service which, when invoked, permits the server 11 to receive communication requests addressed to a specific socket name so as to establish communication with a calling workstation.

Meanwhile, at the client workstation 13, the user activates the Read Mail program for accessing his mailbox. Upon selecting the message sent from the server 11, the "Metamail" service is activated which assumes that the message conforms to the MIME standard and retrieves therefrom the message category, i.e. "Active Mail".

The thus decoded message category is cross-referenced in the mailcap file in order to determine which service is to be invoked responsive to a message of category Active Mail.

In this case, it is assumed that the mailcap file includes an entry which specifies that the Active Mail category corresponds to the service "am connect". As a result, the "am connect" program is invoked which performs several steps.

First, the encoded socket number embedded in the message is decoded. Thereafter, the "socket" service is invoked in order to obtain a socket number in client workstation 13 which will be connected to the pre-defined socket of the server 11. The "connect" service is now called whereby a bi-directional communication channel between the two respective sockets is established.

From the perspective of the "connect" service, the originating computer is the client workstation 13 and the destination computer is the server 11. Since the server 11 is in the "Accept" status awaiting a communication request for coupling a remote workstation to the same socket of the server 11 as is now requested by the "connect" service, the required bi-directional connection is now established. At this stage, a window is opened on the user's screen of the client workstation 13, whereby the application in the client workstation 13 mediates between the underlying socket connection and the thus-defined window. This gives rise to application-dependent communication between the groupware application running on the server 11 and the mediating application in the client workstation 13, whereby the user of the client workstation 13 may interactively communicate with the groupware application running on the server 11 through the corresponding sockets in the server 11 and the client workstation 13.

In the case, as described above, where a groupware application initiates the communication so as to permit multiple, simultaneous processing thereof by a plurality of independent users, there is an implicit assumption that the server, on which the groupware application is loaded, is logged on or active when the user at the second workstation reads the appropriate mailbox message. Such an assumption is likely to be valid, particularly in cases in which the groupware application initiates the communication from a server.

It should further be noted that, whilst in the preferred embodiment, only two workstations are interconnected for two-directional interactive communication, in general a sending workstation can be connected to any number of receiving workstations in an analogous manner to that described. Thus, for example, the first workstation may be associated through a third logical port, different to the first logical port, to a fourth logical port associated with a third workstation. In similar manner, each of the second and third workstations may likewise be linked to yet further workstations.

Thus, the invention as described, permits not only simultaneous, real-time editing, for example, of a groupware document but allows another user not presently involved to be invited to participate. The invitation to participate, being effected through electronic mail, is non-intrusive, although the very act of reading the dispatch-

ed message causes the desired two-way interactive communication to be established.

In the preferred embodiment described above, only a single groupware application is connected to a single window in a receiving workstation. However, it will be readily appreciated that any number of applications can be connected to corresponding windows via appropriate logical ports in either a single workstation or, indeed, in a plurality of workstations.

It will further be noted that the invention produces a bi-directional communication which at its most general is between a server application and a client application, as shown in Fig. 5. In such case, if the client application operates within a window on the client's workstation, then the client application must perform additional steps in order to route the communication to the appropriate window.

However, if the window system of the receiver of an Active Mail message run a network-based window system, such as X under UNIX, then a simpler variant of the above protocol is available. In accordance with such a protocol, upon establishing the two-way interactive communication, the receiver notifies the sender of the global Internet address of its workstation (or X terminal) and executes a command which allows the server to interact directly with the receiver's window system. In such a case, the interactive communication is not between an application running on the server and a process on the client's workstation which then talks to the window, but rather a direct connection between the application running on the server and the window on the receiver's screen.

Furthermore, whilst in the preferred embodiment the two-way interactive communication or dialogue is effected through the computer network, this is not a requirement of the invention. Thus, consider a receiving user whose workstation is connected to the receiving user's telephone line either directly or via a PBX. The act of reading his mailbox and finding a message of the Active Mail category, may, for example, automatically dial the sender and permit the receiving user to establish a dialogue with the sender via the telephone. This approach can, likewise, be extended to any number of participants using shared telephone or conferencing techniques.

Yet a further use of the invention is to effect an interactive communication between two applications running on respective workstations, whilst obviating the need for human interference. Thus, for example, consider a program which prompts a user to enter information and then continues operation along different branches, in accordance with the data entered by the operator.

Instead of a human operator providing the desired information, it is clearly possible to incorporate the responses in a data file for remote reading by the application. In such case, the invention may be employed to initiate an interactive communication between the workstation on which the application is loaded to the remote workstation on which the data file is loaded. The data file itself is, of course, incorporated within an application which, upon sensing the presence of an Active Mail-type message in its mailbox, automatically reads the message so as to establish the required interactive communication with the sending workstation.

The present invention also permits a logical port to be forwarded from a linked user to a non-linked user, so as to connect the non-linked user to the application. Thus, suppose that User B receives from User A a message of the Active Mail category having embedded therein the logical port associated with User A's workstation. He may perform the steps according to the invention in order to establish an interactive communication with User A. Additionally, or alternatively, providing that the "listen" service will manage a sufficient number of connection requests, he may forward the Active Mail message to a third user, User C who will then see the message in his mailbox as if it were directed from User A, there being no reference at all to the fact that this message was, in fact, dispatched by User B.

User C can then establish an interactive communication with User A in the normal manner. This facility is rendered possible because the logical port associated with the sending workstation is embedded in the message dispatched thereby: the embedding being effected when the message is configured and not when it is dispatched.

Such an approach might be used, for example, when User A dispatches a message to User B who reads the Active Mail-type message in order to establish the required interactive communication, and then decides that it would be beneficial to involve a third participant, User C. In such case, he need only forward the Active Mail message to User C, there being no requirement for User B to enter the electronic mail program, construct a suitable Active Mail-type message and dispatch it to User C.

In the detailed description of a preferred embodiment, no mention has been made of the type of data associated with the message other than that the message itself must, of course, be of the Active Mail category. However, in addition to the Active Mail message which simply establishes the required interactive communication, there may be attached thereto any other message of a supported category such as, for example, text, audio, graphics and so on. Such an approach obviates the need for two separate messages to be sent: one for establishing the required interactive communication and the other for dispatching electronic mail in the normal manner.

It will be understood that such an approach is possible only under electronic mail standards which support multiple message categories and attachments. Whilst this true of the MIME standard for "Internet" it is not, of course, universally true. Nevertheless, with slight modification, the invention is applicable even to electronic mail standards which are less versatile than MIME.

5 Thus, suppose that the electronic mail standard supports only text messages which may include typed attachments. Then the mail reading program can be upgraded to handle Active Mail attachments by calling AM Connect to process them. To invoke Active Mail, a suitably coded text message is written and despatched by electronic mail from the sending to the receiving workstations. Such a message might include an attachment of type "Active Mail" and the sending socket number (SN) might be included in the attachment. On reading  
10 such a message, the mail reading program reads the attachment and knows to connect the receiving workstation to socket SN of the sending workstation.

Suppose, however, that only text is supported by the electronic mail standard. In this case, to invoke Active Mail, a suitably coded text message is written and despatched by electronic mail from the sending to the receiving workstation. Such a message might include a banner reading: "Active Mail" and the sending socket name (SN) might be included as part of the message. The mail reading program can then be modified so that  
15 on reading such a message, the mail reading program sees the banner "Active Mail" and, upon decoding the socket name (SN) from the remainder of the message, knows to connect the receiving workstation to socket SN of the sending workstation.

Although the invention described with particular reference to Fig. 5 of the drawings relates to a network operating under UNIX and employing the Internet communications protocol, the more general description relating to Figs. 2 to 4 is applicable both to the Internet protocol and to other network protocols. Thus, it is contemplated that the arrangement described with reference to Figs. 2 to 4 may be easily adapted to any system having mail capabilities and a Point-to-Point communication, such as NOVELL and Net-Bios based networks, including LAN-Manager.

25 No mention has been made so far with regard to disconnecting a client from the server after having established an interactive communication in accordance with the invention. It should be noted that standard means may be employed by the client and/or the server in order to effect such disconnection.

Finally, whilst the invention has been described with reference to establishing an interactive communication between two or more workstations in the same network, it will be clear that it is equally feasible for the workstations to be in different interconnected networks. All that is required is for a unique network name to be reserved for the logical port in each workstation.

35

40

45

50

55

## APPENDIX

```

5
                                     Example of Server Code
/*
 *          Server Demo.
 * Send mail messages containing the local addr to several mail clients
10 * and create a hand-shake.
 *
 */

#define      MarkError(Str,Code) {printf("Error : %s\n",Str); exit(Code); }

15 #define      MAX_SIZE      1024
#define      QUEUE_LEN      5
#define      TMP_FILE_NAME  "/tmp/am.tmp"
#define      SEND_MAIL_CMD  "/usr/lib/sendmail"
#define      SUBJECT        "Subject: ActiveMail connection\n"
20 #define      CONTENT_TYPE  "Content-Type: ActiveMail\n\n"

#include      <errno.h>
#include      <stdio.h>
#include      <string.h>
25 #include      <fcntl.h>
#include      <signal.h>
#include      <sys/time.h>
#include      <sys/ioctl.h>
#include      <sys/types.h>
30 #include      <sys/stat.h>
#include      <sys/socket.h>
#include      <netinet/in.h>
#include      <nctdb.h>
#include      <sys/param.h>

35

char  Buffer[MAX_SIZE]; /* General purpose buffer */

40 main()
{

    struct hostent      *hostP;
    struct sockaddr_in  ServerSa;
    char                HostName[MAXHOSTNAMELEN];
45     int                ServerLen, ServerFd;

    /*
    * Get host name and host network parameters. Fill the socket name
    * with the network addr.
50     */

    if ( gethostname(HostName,MAXHOSTNAMELEN) )
        MarkError("Cannot get host name", 1);

55

```

```

5      if ( (hostP = gethostbyname(HostName)) == NULL)
          MarkError("Cannot get host network params", 1);
      if (hostP->h_addrtype != AF_INET)
          MarkError("Invalid address type", 1);

/*
 * Set the socket name parameters, and let the UNIX choose a port for you.
 */

10     bzero((char *)&ServerSa, sizeof(ServerSa));
        ServerSa.sin_family = AF_INET;
        bcopy(hostP->h_addr, &(ServerSa.sin_addr.s_addr), hostP->h_length);
        ServerSa.sin_port = 0;

15     if ((ServerFd = socket(AF_INET, SOCK_STREAM, 0)) < 0)
          MarkError("Cannot open socket", 1);
        if ( bind(ServerFd, &ServerSa, sizeof ServerSa) < 0)
          MarkError("Cannot bind socket", 1);

20     /*
        * Check that we got a valid port.
        */

        ServerLen = sizeof(ServerSa);
25     if ( getsockname(ServerFd, (struct sockaddr *) &ServerSa, &ServerLen) < 0)
          MarkError("Cannot get socket name", 1);
        if ( ntohs(ServerSa.sin_port) == 0)
          MarkError("Uncorrect port number", 1);

30     /*
        * Listen on the socket and call the requests handling routine.
        */

        if ( listen(ServerFd, QUEUE_LEN) < -1 )
35         MarkError("Cannot listen", 1);

        HandleRequests(ServerFd, &ServerSa);

    } /* main */

40

/******
 *
 * For input Socket and its name SocketName, send a formatted form of the
 * name to mail clients, and wait for connection requests from the client
45 * on Socket. Upon such a request establish a hand-shake.
 *
*****/

50     HandleRequests(Socket, SocketName)
        int Socket;
        struct sockaddr_i *SocketName;
        {

55         struct sockaddr_in ClientSa;

```



```

char        MailMsg[30];
char        *p;
5 char      SendMailCmd[MAX_SIZE];
int         ClientLen;
int         ClientFd, TmpFd;
int         NumOfClients=0;
short      StrLen, BytesRead;

10 /*
   * Format the Socket-Name, and write it to a temporary file.
   * The file will be sent by mail to the client.
   */

15 SocketNameToStr(SocketName, MailMsg);

if ( (TmpFd = open( TMP_FILE_NAME, O_WRONLY | O_CREAT)) < 0)
  MarkError("Cannot open tmp file", TmpFd);
if ( write(TmpFd, SUBJECT, strlen(SUBJECT)) != strlen(SUBJECT) )
  MarkError("Cannot write to tmp file", 1);
20 if ( write(TmpFd, CONTENT_TYPE, strlen(CONTENT_TYPE)) != strlen(CONTENT_TYPE) )
  MarkError("Cannot write to tmp file", 1);
if ( write( TmpFd, MailMsg, strlen(MailMsg)) != strlen(MailMsg) )
  MarkError("Cannot write to tmp file", 1);
close(TmpFd);
25 chmod( TMP_FILE_NAME, S_IRUSR | S_IWUSR | S_IROTH);

/*
 * Loop :
 * Get mail target, send it the local socket name and wait for a
30 * connection request.
 * Once a connection is established, create a hand-shake.
 */

while(1) {

35   printf("\nEnter mail address [name@addr] > ");
   scanf("%s", Buffer);
   printf("Sending mail to %s ... \n", Buffer);
   sprintf( SendMailCmd, "%s %s < %s", SEND_MAIL_CMD, Buffer, TMP_FILE_NAME);
   system(SendMailCmd);

40   ClientLen = sizeof(ClientSa);
   if ( (ClientFd = accept(Socket, &ClientSa, &ClientLen)) < 0 )
     MarkError("Cannot accept", 1);

45   /*
    * Write a message to the client.
    */

   sprintf(Buffer, "You are client number %d", NumOfClients++);
   StrLen = strlen(Buffer);
50   write(ClientFd, (char *)&StrLen, sizeof(StrLen));
   write(ClientFd, Buffer, strlen(Buffer));

55

```

```

/*
 * Read the acknowledgment from the client.
 */
5 BytesRead = recv(ClientFd, (char *)&StrLen, sizeof(StrLen), 0);
  if ( BytesRead != sizeof(StrLen) )
    MarkError("Cannot read message size", 1);

  BytesRead = recv(ClientFd, Buffer, StrLen, 0);
10  if ( BytesRead != StrLen )
    MarkError("Cannot read message size", 1);
    Buffer[BytesRead] = '\0';
    printf("Recieved message : %s\n", Buffer);

15  } /* while */

} /* HandleRequests */

20
/*.....
 *
 * Format SocketName to a string containing the family, port and address.
 * Output the formatted form in Str.
 *
25 .....*/

SocketNameToStr(SocketName, Str)
struct sockaddr_in *SocketName;
char *Str;
30 {

    sprintf( Str, "%4hx %4hx %8lx\n", SocketName->sin_family,
              SocketName->sin_port,
              SocketName->sin_addr.s_addr );
35 } /* SocketNameToStr */

40

45

50

55

```

## Example of Client Code

```

/*
 *          Client Demo.
5  * On input fn in the command line, connect to a server whose address is
 * formatted in the file. Establish a hand-shake and terminate.
 */

10 #define      MarkError(Str,Code) {printf("Error : %s\n",Str); exit(Code); }

#define      MAX_SIZE  1024
#define      HDR_LEN   2

15 #include     <errno.h>
#include     <stdio.h>
#include     <string.h>
#include     <fcntl.h>
#include     <signal.h>
#include     <sys/time.h>
20 #include     <sys/ioctl.h>
#include     <sys/types.h>
#include     <sys/socket.h>
#include     <netinet/in.h>
#include     <nctdb.h>
25 #include     <sys/param.h>

char  Buffer[MAX_SIZE]; /* General purpose buffer */

main(argc,argv)
30 int  argc;
char  *argv[];
{

    struct sockaddr_in  ServerSa;
35 char          *p;
    int           ClientFd;
    short        BytesRead, StrLen;
    FILE         *Fin;

40 if (argc != 2) {
    printf("Usage : client FileName\n");
    exit(0);
    }

45 /*
 * Open the file and read the formatted internet addr from it.
 */

    if ( (Fin = fopen( argv[1], "r")) == NULL )
50     MarkError("Cannot open file", 1);

    fscanf(Fin, "%4hx %4hx %8lx", &(ServerSa.sin_family),
           &(ServerSa.sin_port),
           &(ServerSa.sin_addr.s_addr) );

55 fclose(Fin);

```

```

/*
 * Connect with the server.
 */
5      if ((ClientFd = socket(AF_INET, SOCK_STREAM, 0)) < 0)
        MarkError("Cannot open socket", 1);
      if ( connect(ClientFd, &ServerSa, sizeof(ServerSa)) < 0)
        MarkError("Cannot connect to server", 1);

10     /*
      * Read a message from the server.
      */

      BytesRead = recv(ClientFd, (char *)&StrLen, sizeof(StrLen), 0);
15     if ( BytesRead != sizeof(StrLen) )
        MarkError("Cannot read message size", 1);

      BytesRead = recv(ClientFd, Buffer, StrLen, 0);
      if ( BytesRead != StrLen )
20     MarkError("Cannot read message size", 1);
      Buffer[BytesRead] = '\0';
      printf("Received message : %s\n", Buffer);

      /*
25     * Write an acknowledgment to the server.
      */

      strcpy(Buffer, "Got Your message");
      StrLen = strlen(Buffer);
30     write(ClientFd, (char *)&StrLen, sizeof(StrLen));
      write(ClientFd, Buffer, StrLen);

      close(ClientFd);

35     } /* main */

```

#### Claims

- 40
1. For use in a computer network system (10) comprising at least first and second workstations (11, 13, 14, 15, 16, 17) adapted to send and receive messages by utilizing a suitable communication protocol and further adapted to exchange batch messages by means of an electronic mail program stored in each of said at least first and second workstations;
    - 45 a method for establishing an interactive communication between said at least first and second workstations, said method characterized by the steps of:
      - (i) categorizing said batch messages such that a batch message of a predetermined category informs a receiving workstation that a sending workstation wishes to establish an interactive communication between a specified first logical port in the sending workstation and a specified second logical port in the receiving workstation;
      - 50 (ii) sending a batch message of the predetermined category having therein a reference to said first logical port from the first workstation to the second workstation so as to be received thereby and stored in a storage means (19) containing a list of batch messages;
      - (iii) monitoring at the second workstation all batch messages in said storage means (19) at specified periods of time;
      - 55 (iv) noting the presence in said storage means (19) of a batch message of said predetermined category;
      - (v) utilizing the communication protocol to send an initiation signal from the second logical port in the second workstation to the first logical port in the first workstation; and

- (vi) responsive to receipt of the initiation signal, establishing an interactive two-way communication between the first logical port of the first workstation and the second logical port of the second workstation.
2. The method according to Claim 1, wherein the first logical port of the first workstation is associated with a groupware application.
  3. The method according to Claim 1, wherein the first logical port of the first workstation and the second logical port of the second workstation are each associated with respective applications which interactively communicate with each other.
  4. The method according to Claim 1, wherein the computer network system comprises at least two interconnected networks and said first and second workstations are located in different ones of said interconnected networks.
  5. The method according to Claim 1, wherein said batch message includes an attachment having therein data of a category supported by the electronic mail program, whereby upon reading the message, said data is automatically output to the second workstation.
  6. The method according to Claim 1, wherein the first logical port in the first workstation serves only to establish a communication whereupon the communication is subsequently routed via a third logical port to the first workstation, thereby permitting multiple connections to be established to the first workstation via said first logical port.
  7. The method according to Claim 1, wherein the first workstation establishes said interactive two-way communication with said second workstation and with at least one third workstation whereby all three workstations communicate simultaneously.
  8. The method according to Claim 7, wherein the second workstation and said at least one third workstation are each coupled to different logical ports in the first workstation.
  9. The method according to Claim 1, wherein the computer network system operates under UNIX.
  10. The method according to Claim 1, wherein the computer network system operates under NOVELL.
  11. The method according to Claim 1, wherein the computer network system operates under LAN Manager.
  12. The method according to Claim 1, wherein:
    - multiple message categories are supported by the electronic mail program, and
    - a unique message category is defined indicating that the sending workstation wishes to establish said communication with the receiving workstation.
  13. The method according to Claim 1, wherein:
    - multiple message categories are not supported by the electronic mail program,
    - the electronic mail program supports attachments, and
    - said reference to the first logical port is included within an attachment which serves as an argument to the electronic mail program on reading the batch message at the receiving workstation.
  14. The method according to Claim 1, wherein:
    - multiple message categories are not supported by the electronic mail program,
    - the electronic mail program does not support attachments, a banner is included within the batch message to inform the electronic mail program that the sending workstation wishes to establish said communication with the receiving workstation, and
    - the first logical port of the sending workstation is encoded within the batch message and serves as an argument to the electronic mail program on reading the batch message at the receiving workstation.
  15. The method according to Claim 1, wherein:
    - the second workstation runs a network based window system associated with a global network address of the second workstation and having means for determining whether a process running on a remote workstation is authorized to open a window (21) on the second workstation and communicate with said window (21), and

prior to step (v) there is included the further step of:

(ivb) authorizing the first logical port of the first workstation to open a window (21) on the second workstation;

whereby:

5 in step (v) the communication protocol is utilized in order to supply the global network address of the second workstation to the first logical port in the first workstation, and

upon performing step (vi), a signal is sent from the first workstation to the second workstation in order to open a window (21) on the second workstation with which the first workstation may interact with a user on the second workstation.

10

16. The method according to Claim 15, wherein:

the network based window system is X said means for determining whether a process running on a remote workstation is authorized to open a window (21) on the second workstation and communicate with said window (21) comprising a list of addresses each in respect of a remote workstation which is au-

15

thorized to open said window (21) and communicate therewith, and  
in step (ivb) the first workstation is added to the list of addresses in the second workstation.

20

25

30

35

40

45

50

55

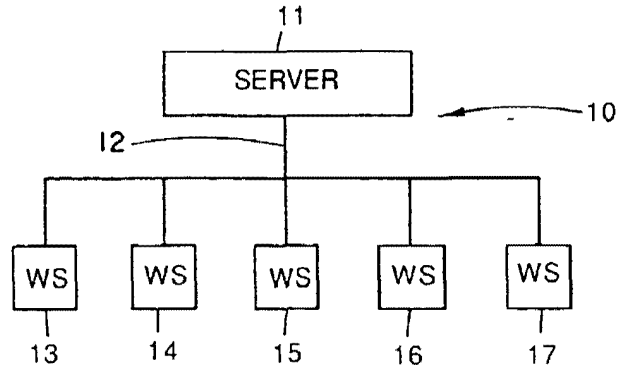


Fig.1 a

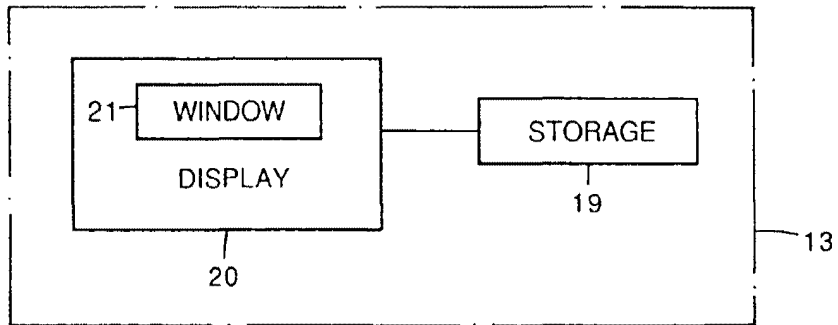


Fig.1 b

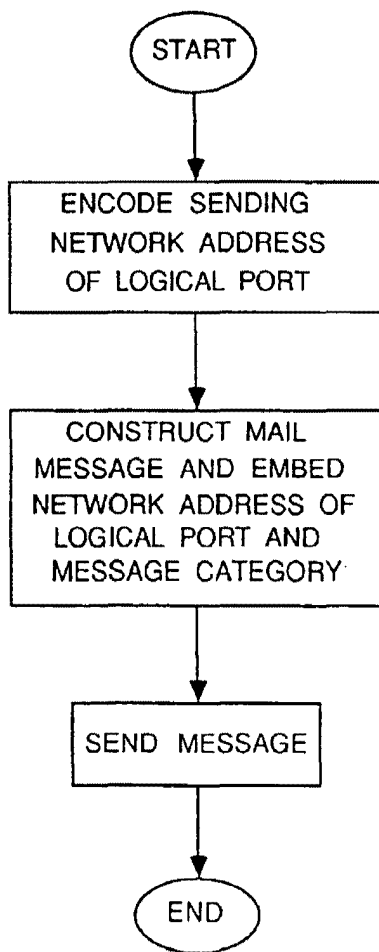


Fig.2



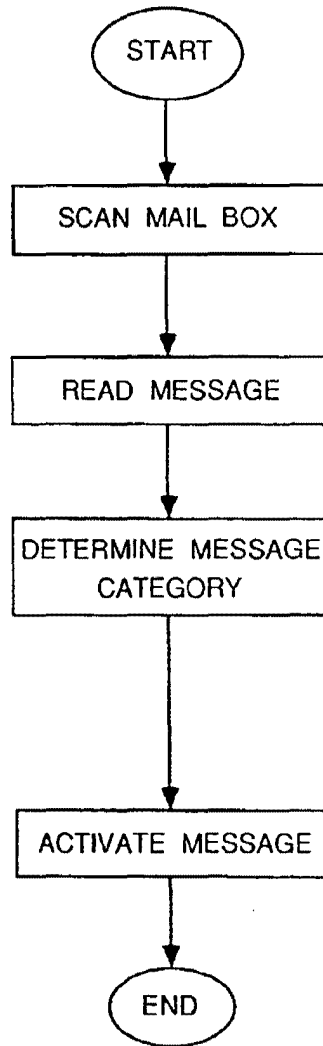


Fig.3

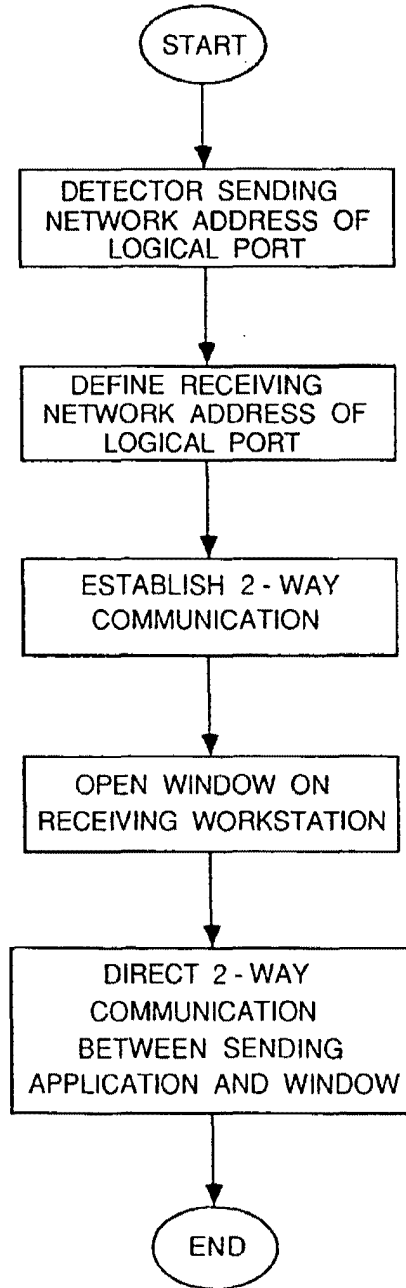


Fig.4

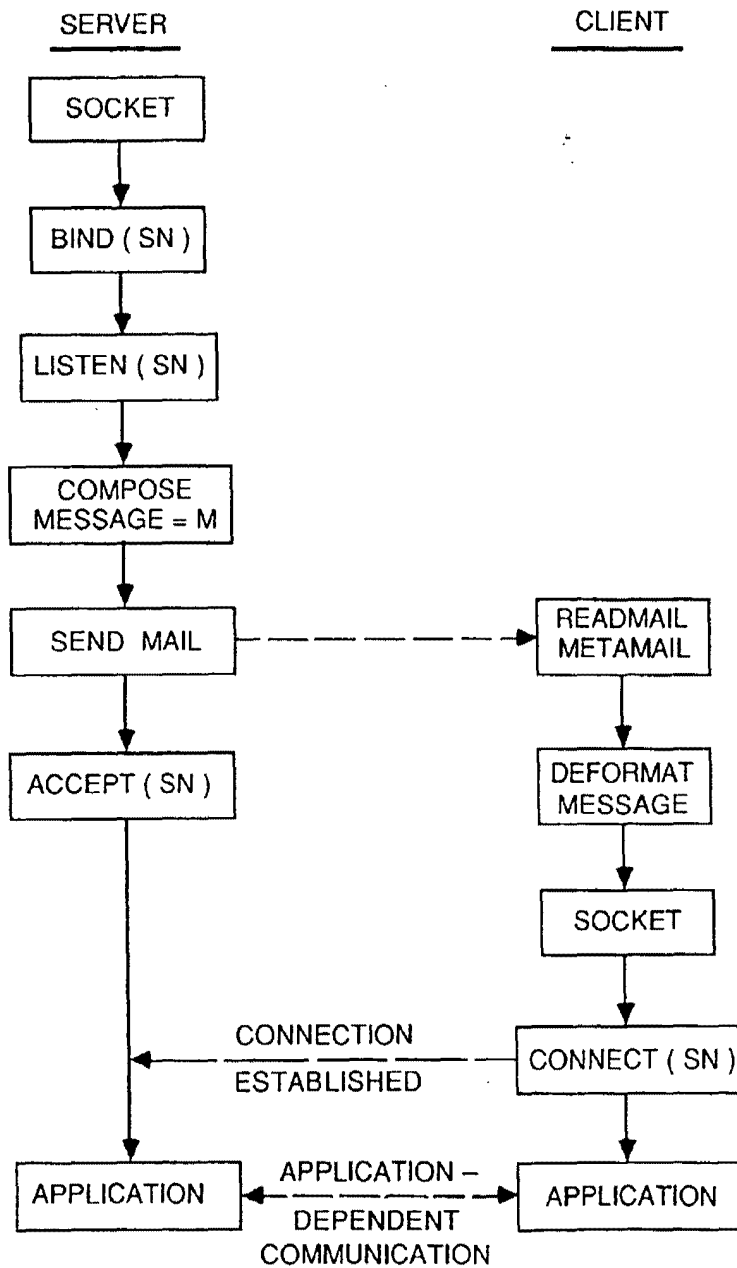


Fig.5



European Patent  
Office

EUROPEAN SEARCH REPORT

Application Number

EP 93 63 0052

DOCUMENTS CONSIDERED TO BE RELEVANT			
Category	Citation of document with indication, where appropriate, of relevant passages	Relevant to claims	CLASSIFICATION OF THE APPLICATION (Int. Cl.5)
A	WO-A-9 003 074 (CAPRICOM, S.A.) * page 2, line 14 - page 3, line 25 * * page 5, line 20 - page 7, line 3 * ----	1-16	H04L12/58
A	US-A-5 040 141 (K.YAZIMA ET AL) * column 3, line 12 - line 51 * ----	1-16	
A	US-A-5 127 003 (W.J.DOLL ET AL) * column 7, line 64 - column 8, line 49 * * column 10, line 5 - line 21 * -----	1-16	
			TECHNICAL FIELDS SEARCHED (Int. Cl.5)
			G06F H04L
The present search report has been drawn up for all claims			
Place of search <b>THE HAGUE</b>		Date of completion of the search <b>21 OCTOBER 1993</b>	Examiner <b>CANOSA ARESTE C.</b>
<b>CATEGORY OF CITED DOCUMENTS</b> X : particularly relevant if taken alone Y : particularly relevant if combined with another document of the same category A : technological background U : non-written disclosure P : intermediate document		T : theory or principle underlying the invention E : earlier patent document, but published on, or after the filing date D : document cited in the application L : document cited for other reasons & : member of the same patent family, corresponding document	

EPO FORM 1503 (04.82) (P0401)



Re-exam

**IN THE UNITED STATES PATENT AND TRADEMARK OFFICE**

In re PATENT APPLICATION OF:

Net2Phone, Inc.

Control No.: 90/010,416

Issue Date: August 22, 2000

Title: **POINT-TO-POINT INTERNET  
PROTOCOL**

Attorney Docket: 2655-0188

Group Art Unit: 3992

Examiner: KOSOWSKI, Alexander  
J.

Date: June 11, 2009

Confirmation No.: 1061

**INFORMATION DISCLOSURE STATEMENT**

Hon. Commissioner of Patents  
P.O. Box 1450  
Alexandria, VA 22313-1450

Sir:

Pursuant to 37 C.F.R. § 1.56, the attention of the Patent and Trademark Office is hereby directed to the reference(s) listed on the attached PTO-1449. One copy of each non-U.S. Patent reference is attached. It is respectfully requested that the information be expressly considered during the prosecution of this application, and that the reference(s) be made of record therein and appear among the "References Cited" on any patent to issue therefrom.

The submission of any document herewith, which is not a statutory bar, is not intended that any such document constitutes prior art against any of the claims of the present application or is considered to be material to patentability as defined in 37 C.F.R. § 1.56(b). Applicants do not waive any rights to take any action which would be appropriate to antedate or otherwise remove as a competent reference against the claims of the present application.

**N2P-IDS00376**

This Information Disclosure Statement is being filed within three (3) months of the U.S. filing date OR before the mailing date of a first Office Action on the merits. No certification or fee is required.

This Information Disclosure Statement is being filed more than three (3) months after the U.S. filing date AND after the mailing date of the first Office Action on the merits, but before the mailing date of a Final Rejection or Notice of Allowance.

I hereby certify that each item of information contained in this Information Disclosure Statement was cited in a communication from a foreign patent office in a counterpart foreign application not more than three (3) months prior to the filing of this Information Disclosure Statement. 37 C.F.R. § 1.97(e)(1).

I hereby certify that no item of information in this Information Disclosure Statement was cited in a communication from a foreign patent office in a counterpart foreign application or, to my knowledge after making reasonable inquiry, was known to any individual designated in 37 C.F.R. § 1.56(c) more than three (3) months prior to the filing of this Information Disclosure Statement. 37 C.F.R. § 1.97(e)(2).

Attached is our check no. \_\_\_\_\_ in the amount required under 37 C.F.R. § 1.17(p). Please credit or debit Deposit Account No. 501860 as needed to ensure consideration of the disclosed information. A duplicate copy of this paper is attached.

This Information Disclosure Statement is being filed more than three (3) months after the U.S. filing date and after the mailing date of a Final Rejection or Notice of Allowance, but before payment of the Issue Fee. Applicant(s) hereby requests that the Information Disclosure Statement be considered. Attached is our check in the amount required under 37 C.F.R. § 1.17(p). Please credit or debit

Deposit Account No. 501860 as needed to ensure consideration of the disclosed information. A duplicate copy of this paper is attached.

- I hereby certify that each item of information contained in this Information Disclosure Statement was cited in a communication from a foreign patent office in a counterpart foreign application not more than three (3) months prior to the filing of this Information Disclosure Statement. 37 C.F.R. § 1.97(e)(1).
- I hereby certify that no item of information in this Information Disclosure Statement was cited in a communication from a foreign patent office in a counterpart foreign application and, to my knowledge after making reasonable inquiry, was known to any individual designated in 37 C.F.R. § 1.56(c) more than three (3) months prior to the filing of this Information Disclosure Statement. 37 C.F.R. § 1.97(e)(2).
- Relevance of the non-English language reference(s) is/are discussed in the present specification.
- The reference(s) was/were cited in a counterpart foreign application. An English language version of the foreign search report is attached for the Examiner's information.
- A concise explanation of the relevance of the non-English language reference(s) appear(s) in the Appendix hereto.
- The Examiner's attention is directed to co-pending U.S. Patent Application No. \_\_\_\_\_, filed \_\_\_\_\_, which is directed to related technical subject matter. The identification of this U.S. Patent Application is not to be construed as a waiver of secrecy as to that application now or upon issuance of the present application as a patent. The Examiner is respectfully requested to consider the cited application and the art cited therein during examination.

Copies of the references were cited by or submitted to the Office in parent Application No. \_\_\_\_\_, filed \_\_\_\_\_, which is relied upon for an earlier filing date under 35 U.S.C. 120. Thus, Form PTO 1449 is attached without copies of these references. 37 C.F.R. § 1.98(d).

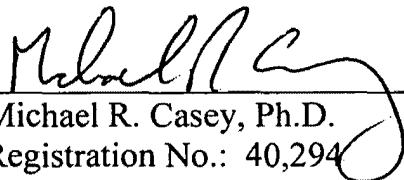
**CHARGE STATEMENT:** Deposit Account No. 501860, order no. 2655-0188.  
The Commissioner is hereby authorized to charge any fee specifically authorized hereafter, or any missing or insufficient fee(s) filed, or asserted to be filed, or which should have been filed herewith or concerning any paper filed hereafter, and which may be required under Rules 16-18 (missing or insufficiencies only) now or hereafter relative to this application and the resulting Official Document under Rule 20, or credit any overpayment, to our Accounting/Order Nos. shown above, for which purpose a duplicate copy of this sheet is attached

**This CHARGE STATEMENT does not authorize charge of the issue fee until/unless an issue fee transmittal sheet is filed.**

CUSTOMER NUMBER  
**42624**

Davidson Berquist Jackson & Gowdey LLP  
4300 Wilson Blvd., 7th Floor,  
Arlington Virginia 22203  
Main: (703) 894-6400 • FAX: (703) 894-6430

Respectfully submitted,

By:   
Michael R. Casey, Ph.D.  
Registration No.: 40,294



**CERTIFICATE OF SERVICE**

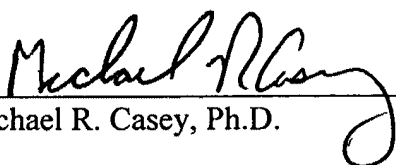
The undersigned hereby certifies that, on June 11, 2009, the Information Disclosure Statements (with references in electronic format, as agreed by requestor) filed in Reexam Control

Numbers:

- 1) 90/010,422;
- 2) 90/010,424;
- 3) 90/010,421;
- 4) 90/010,416; and
- 5) 90/010,423

were served by FedEx (tracking no. 796686808721), on Requestor:

Blakely, Sokoloff, Taylor & Zafman LLP  
1279 Oakmead Parkway  
Sunnyvale, CA 94085-4040

  
\_\_\_\_\_  
Michael R. Casey, Ph.D.

<b>INFORMATION DISCLOSURE STATEMENT BY APPLICANT</b> FORM PTO-1449 (modified)  Sheet 1 of 67	Reexam number	90/010,416
	First Named Inventor	Hutton
	Patent Under Re-Exam	6108704
	Issue Date	2000/08/22
	Group Art Unit	3992
	Examiner Name	KOSOWSKI, ALEXANDER J
	Attorney Docket No.	2655-0188
	Confirmation No.	1061

U.S. PATENT DOCUMENTS				
Examiner Initials*	Cite No.	Document No.	Publication/ Issue Date	Name of Patentee or Applicant of Cited Document
	1-1	US-4313035	1982/01/26	Jordan et al.
	1-2	US-4423414	1983/12/27	Bryant et al.
	1-3	US-4491693	1985/01/01	Sano et al.
	1-4	US-4602132	1986/07/22	Nagatomi et al.
	1-5	US-4653090	1987/03/24	Hayden, C.
	1-6	US-4658093	1987/14/04	Hellman
	1-7	US-4706274	1987/11/10	Baker et al.
	1-8	US-4754479	1988/06/28	Bicknell et al.
	1-9	US-4755985	1988/07/05	Jayapalan et al.
	1-10	US-4756020	1988/07/05	Fodale
	1-11	US-4759056	1988/07/19	Akiyama
	1-12	US-4800488	1989/24/01	Agrawal et al.
	1-13	US-4823374	1989/04/18	Verlohr
	1-14	US-4827411	1989/05/02	Arrowood
	1-15	US-4899333	1990/06/02	Roediger
	1-16	US-4899373	1990/02/06	Lee et al.
	1-17	US-4914571	1990/04/03	Baratz et al.
	1-18	US-4928306	1990/05/22	Biswas et al.
	1-19	US-4953159	1990/08/28	Hayden, C.
	1-20	US-4962449	1990/10	Schlesinger
	1-21	US-5109403	1992/04/28	Sutphin
	1-22	US-5113499	1992/05	Ankney et al.
	1-23	US-5127001	1992/30/06	Steagall, et al.
	1-24	US-5134648	1992/07/28	Hochfield et al.

Examiner Signature		Date Considered	
--------------------	--	-----------------	--

\*Examiner: Initial if reference was considered, whether or not citation is in conformance with MPEP 609. Draw a line through citation if not in conformance and not considered. Include a copy of this form with next communication to applicant.

N2P-IDS00227

**INFORMATION DISCLOSURE  
STATEMENT BY APPLICANT**  
FORM PTO-1449 (modified)

Sheet 2 of 67

Reexam number	90/010,416
First Named Inventor	Hutton
Patent Under Re-Exam	6108704
Issue Date	2000/08/22
Group Art Unit	3992
Examiner Name	KOSOWSKI, ALEXANDER J
Attorney Docket No.	2655-0188
Confirmation No.	1061

**U.S. PATENT DOCUMENTS**

Examiner Initials*	Cite No.	Document No.	Publication/ Issue Date	Name of Patentee or Applicant of Cited Document
	2-1	US-5136716	1992/08/04	Harvey et al.
	2-2	US-5153908	1992/10/06	Kakizawa et al.
	2-3	US-5159592	1992/10	Perkins
	2-4	US-5164988	1992/11/17	Matyas et al.
	2-5	US-5185860	1993/02/09	Wu
	2-6	US-5195086	1993/03/16	Baumgartner et al.
	<del>2-7</del>	<del>US-5301324</del>	<del>1994/04</del>	<del>Dewey et al.</del>
	2-8	US-5315705	1994/24/05	Iwami Naoko et al
	2-9	US-5319705	1994/07/06	Halter et al.
	2-10	US-5325524	1994/06/28	Black et al.
	2-11	US-5329619	1994/07/12	Page et al.
	2-12	US-5388213	1995/02/07	Oppenheimer et al.
	2-13	US-5402477	1995/03/28	McMahan et al.
	2-14	US-5402528	1995/03/28	Christopher et al.
	2-15	US-5408526	1995/04/18	McFarland et al.
	2-16	US-5408619	1995/04/18	Oran
	2-17	US-5425028	1995/13/06	Britton et al.
	2-18	US-5434913	1995/07	Tung et al.
	2-19	US-5440632	1995/08/08	Bacon et al.
	2-20	US-5452289	1995/09/19	Sharma et al.
	2-21	US-5461668	1995/10/24	Zdenek et al.
	2-22	US-5469500	1995/21/11	Satter et al.
	2-23	US-5475819	1995/12	Miller et al.
	2-24	US-5481720	1996/02/01	Loucks et al.

Examiner Signature		Date Considered	
--------------------	--	-----------------	--

\*Examiner: Initial if reference was considered, whether or not citation is in conformance with MPEP 609. Draw a line through citation if not in conformance and not considered. Include a copy of this form with next communication to applicant.

N2P-IDS00228

**INFORMATION DISCLOSURE  
STATEMENT BY APPLICANT**  
FORM PTO-1449 (modified)

Sheet 3 of 67

Reexam number	90/010,416
First Named Inventor	Hutton
Patent Under Re-Exam	6108704
Issue Date	2000/08/22
Group Art Unit	3992
Examiner Name	KOSOWSKI, ALEXANDER J
Attorney Docket No.	2655-0188
Confirmation No.	1061

**U.S. PATENT DOCUMENTS**

Examiner Initials*	Cite No.	Document No.	Publication/ Issue Date	Name of Patentee or Applicant of Cited Document
	3-1	US-5499295	1996/12/03	Cooper
	3-2	US-5502727	1996/03/26	Catanzaro et al.
	3-3	US-5515508	1996/05/07	Pettus et al.
	3-4	US-5533110	1996/02/07	Pinard et al.
	3-5	US-5555290	1996/09/10	McLeod et al.
	3-6	US-5608786	1997/04/03	Gordon
	3-7	US-5615257	1997/03/25	Pezzullo et al.
	3-8	US-5621789	1997/04/15	McCalmont et al.
	3-9	US-5627978	1997/05/06	Altom et al.
	3-10	US-5649194	1997/07	Miller et al.
	3-11	US-5671412	1997/09/23	Christiano
	3-12	US-5684951	1997/04/11	Goldman et al.
	3-13	US-5689641	1997/11/18	Ludwig et al.
	3-14	US-5692180	1997/11	Lee
	3-15	US-5724648	1998/03/03	Shaughnessy et al.
	3-16	US-5734828	1998/31/03	Pendse et al.
	3-17	US-5774656	1998/06/30	Hattori et al.
	3-18	US-5790803	1998/04/08	Kinoshita et al.
	3-19	US-5815665	1998/29/09	Teper et al.
	3-20	US-5819084	1998/10/08	Shapiro, E.
	3-21	US-5825865	1998/20/10	Oberlander et al.
	3-22	US-5844978	1998/12/01	Reuss et al.
	3-23	US-5883956	1999/03/16	Le et al.
	3-24	US-5953350	1999/09	Higgins

Examiner Signature		Date Considered	
--------------------	--	-----------------	--

\*Examiner: Initial if reference was considered, whether or not citation is in conformance with MPEP 609. Draw a line through citation if not in conformance and not considered. Include a copy of this form with next communication to applicant.

N2P-IDS00229

**INFORMATION DISCLOSURE  
STATEMENT BY APPLICANT**  
FORM PTO-1449 (modified)

Sheet 4 of 67

Reexam number	90/010,416
First Named Inventor	Hutton
Patent Under Re-Exam	6108704
Issue Date	2000/08/22
Group Art Unit	3992
Examiner Name	KOSOWSKI, ALEXANDER J
Attorney Docket No.	2655-0188
Confirmation No.	1061

**U.S. PATENT DOCUMENTS**

Examiner Initials*	Cite No.	Document No.	Publication/ Issue Date	Name of Patentee or Applicant of Cited Document
	4-1	US-5956485	1999/09/21	Periman
	4-2	US-6009469	1999/12	Mattaway et al.
	4-3	US-6031836	2000/02	Haserodt
	4-4	US-6047054	2000/04	Bayless et al.
	4-5	US-6067350	2000/05/23	Gordon, A.
	4-6	US-6108704	2000/08	Hutton et al.
	4-7	US-6131121	2000/10	Mattaway et al.
	4-8	US-6360266	2002/03/19	Pettus
	4-9	US-6513066	2003/01	Hutton et al.
	4-10	US-6701365	2004/02/03	Glenn W. Hutton
	4-11			
	4-12			
	4-13			
	4-14			
	4-15			
	4-16			
	4-17			
	4-18			
	4-19			
	4-20			
	4-21			
	4-22			
	4-23			
	4-24			

Examiner Signature		Date Considered	
--------------------	--	-----------------	--

\*Examiner: Initial if reference was considered, whether or not citation is in conformance with MPEP 609. Draw a line through citation if not in conformance and not considered. Include a copy of this form with next communication to applicant.

**INFORMATION DISCLOSURE  
STATEMENT BY APPLICANT**  
FORM PTO-1449 (modified)

Sheet 5 of 67

Reexam number	90/010,416
First Named Inventor	Hutton
Patent Under Re-Exam	6108704
Issue Date	2000/08/22
Group Art Unit	3992
Examiner Name	KOSOWSKI, ALEXANDER J
Attorney Docket No.	2655-0188
Confirmation No.	1061

**FOREIGN PATENT DOCUMENTS**

Examiner Initials*	Cite No.	Document No.	Publication Date	Name of Patentee or Applicant of Cited Document	Notes
	5-1	EP-0455402-A2	11-06-1991	Hewlett-Packard Company	
	5-2	EP-0556012-A2	08-18-1993	Wada et al.	
	5-3	EP-0581722	02/02/1994	Yeda R&D Co., Ltd.	
	5-4	EPO 0497022A1	19920508	Jennings et al.	
	5-5	WO-9219054	10-29-1992	Ferdinand et al.	
	5-6				
	5-7				
	5-8				
	5-9				
	5-10				
	5-11				
	5-12				
	5-13				
	5-14				
	5-15				
	5-16				
	5-17				
	5-18				
	5-19				
	5-20				
	5-21				
	5-22				
	5-23				
	5-24				

Examiner Signature		Date Considered	
--------------------	--	-----------------	--

\*Examiner: Initial if reference was considered, whether or not citation is in conformance with MPEP 609. Draw a line through citation if not in conformance and not considered. Include a copy of this form with next communication to applicant. Notes: If identified, the following is provided: EA = English Abstract, T = Translation, PF = Patent Family.

N2P-IDS00231

**INFORMATION DISCLOSURE  
STATEMENT BY APPLICANT**

FORM PTO-1449 (modified)

Sheet 6 of 67

Reexam number	90/010,416
First Named Inventor	Hutton
Patent Under Re-Exam	6108704
Issue Date	2000/08/22
Group Art Unit	3992
Examiner Name	KOSOWSKI, ALEXANDER J
Attorney Docket No.	2655-0188
Confirmation No.	1061

**NON-PATENT REFERENCES**

Examiner Initials*	Cite No.	Non-patent Reference bibliographic information, where available	Notes
	6-1	"A Low Cost Solution for: Using your WAN as a Voice Communication Tool" VocalTec White Paper (dated 06/03/94)	
	6-2	"CyberPhone Annoucement" Internet Posting in Newsgroups comp.speech, June 8, 1995.	
	6-3	"CyberPhone!" Internet Posting in Newsgroups comp.speech, April 14, 1995.	
	6-4	"Electric Magic Company Provides Internet Alternative to Long-Distance Calls", Electric Magic Company Press Release (March 13, 1995)	
	6-5	"Electric Magic Company Releases NetPhone 1.2 and Netpub Server", Electric Magic Company Press Release (June 1995)	
	6-6	"Frequently-Asked Questions about Tribal Voices PowWow" Version 0.34, March 4, 1996.	
	6-7	"Frequently-Asked Questions about Tribal Voices PowWow" Version 0.43, May 1, 1996.	

Examiner Signature		Date Considered	
--------------------	--	-----------------	--

\*Examiner: Initial if reference was considered, whether or not citation is in conformance with MPEP 609. Draw a line through citation if not in conformance and not considered. Include a copy of this form with next communication to applicant. Notes: If identified, the following is provided: EA = English Abstract, T = Translation, PF = Patent Family.

N2P-IDS00232

<b>INFORMATION DISCLOSURE STATEMENT BY APPLICANT</b> FORM PTO-1449 (modified)  Sheet 7 of 67	Reexam number	90/010,416
	First Named Inventor	Hutton
	Patent Under Re-Exam	6108704
	Issue Date	2000/08/22
	Group Art Unit	3992
	Examiner Name	KOSOWSKI, ALEXANDER J
	Attorney Docket No.	2655-0188
Confirmation No.	1061	

NON-PATENT REFERENCES			
Examiner Initials*	Cite No.	Non-patent Reference bibliographic information, where available	Notes
	7-1	"Frequently-Asked Questions about Tribal Voices PowWow" Version 0.45, May 31, 1996.	
	7-2	"Frequently-Asked Questions about Tribal Voices PowWow" Version 0.47, June 12, 1996.	
	7-3	"Frequently-Asked Questions about Tribal Voices PowWow" Version 0.48, June 25, 1996.	
	7-4	"Frequently-Asked Questions about Tribal Voices PowWow" Version 0.59, October 30, 1996.	
	7-5	"NetPhone Gets Internet Users Talking at Local Rates" MacUser UK, March 3, 1995, pg. 27.	
	7-6	"NetPhone Gives Your Mac Voice Over the Internet" Inside the Internet Rocket Science for the Rest of Us. Vol. 2 Num. 3, June 1995.	
	7-7	"NetPhone" MacWorld, July 1995.	

Examiner Signature		Date Considered	
--------------------	--	-----------------	--

\*Examiner: Initial if reference was considered, whether or not citation is in conformance with MPEP 609. Draw a line through citation if not in conformance and not considered. Include a copy of this form with next communication to applicant. Notes: If identified, the following is provided: EA = English Abstract, T = Translation, PF = Patent Family.



<b>INFORMATION DISCLOSURE STATEMENT BY APPLICANT</b> FORM PTO-1449 (modified)	Reexam number	90/010,416
	First Named Inventor	Hutton
	Patent Under Re-Exam	6108704
	Issue Date	2000/08/22
	Group Art Unit	3992
	Examiner Name	KOSOWSKI, ALEXANDER J
	Attorney Docket No.	2655-0188
	Confirmation No.	1061
Sheet 8 of 67		

NON-PATENT REFERENCES			
Examiner Initials*	Cite No.	Non-patent Reference bibliographic information, where available	Notes
	8-1	"NetPhone" West Coast Online, Ver. 3.02 (#26), April 1995.	
	8-2	"PowWow 1.3b Now Available!" Google Newsgroup comp.os.ms-windows.misc Discussion Posting (dated April 22, 1995)	
	8-3	1996-1997 Buyer's Guide, CTI for Management	
	8-4	Abbe Cohen, Inessential Zephyr (Aug. 23, 1993).	
	8-5	Adam Gaffin, VocalTec Ware Lets Users Make Voice Calls over 'Net, NETWORK WORLD (Feb. 13, 1995).	
	8-6	Alexander Schill, ed., DCE—The OSF Distributed Computer Environment: Client/Server Model and Beyond, Lecture Notes in Computer Science 731, Karlsruhe University (1993).	
	8-7	Analysis of DCE Security Draft (Sept. 18, 1996).	

Examiner Signature		Date Considered	
--------------------	--	-----------------	--

\*Examiner: Initial if reference was considered, whether or not citation is in conformance with MPEP 609. Draw a line through citation if not in conformance and not considered. Include a copy of this form with next communication to applicant. Notes: If identified, the following is provided: EA = English Abstract, T = Translation, PF = Patent Family.

N2P-IDS00234

**INFORMATION DISCLOSURE  
STATEMENT BY APPLICANT**  
FORM PTO-1449 (modified)

Sheet 9 of 67

Reexam number	90/010,416
First Named Inventor	Hutton
Patent Under Re-Exam	6108704
Issue Date	2000/08/22
Group Art Unit	3992
Examiner Name	KOSOWSKI, ALEXANDER J
Attorney Docket No.	2655-0188
Confirmation No.	1061

**NON-PATENT REFERENCES**

Examiner Initials*	Cite No.	Non-patent Reference bibliographic information, where available	Notes
	9-1	Andrew D. Birrell, et al., Grapevine: An Exercise in Distributed Computing, Communications of the ACM (April 1982).	
	9-2	Andrew D. Birrell, et al., Grapevine: An Exercise in Distributed Computing, COMMUNICATIONS OF THE ACM, vol. 25, No. 4, Apr. 1982.	
	9-3	Andrew D. Birrell, et al., Implementing Remote Procedure Calls, ACM Transactions on Computer Systems (Feb. 1984).	
	9-4	Andrew S. Tanenbaum, COMPUTER NETWORKS, 2d ed. (Prentice-Hall, 1988).	
	9-5	Andy Patrizio, Telecom, Digital Limits Begin to Blur with 'Phone Calls' Across Internet, PC WEEK, vol. 12, no. 6 (Feb. 13, 1995).	
	9-6	Antonio Ruiz, Voice and Telephony Applications for the Office Workstation, IEEE 1st International Conference on Computer Workstations, San Jose, California (Nov. 11-14, 1985).	
	9-7	AVC-650: Technical Issues Concerning Real-Time Protocol in H.32Z Systems in ATM and Other Packet-Switched Computer Networks, July 9, 1994.	

Examiner Signature		Date Considered	
--------------------	--	-----------------	--

\*Examiner: Initial if reference was considered, whether or not citation is in conformance with MPEP 609. Draw a line through citation if not in conformance and not considered. Include a copy of this form with next communication to applicant. Notes: If identified, the following is provided: EA = English Abstract, T = Translation, PF = Patent Family.

N2P-IDS00235

<b>INFORMATION DISCLOSURE STATEMENT BY APPLICANT</b> FORM PTO-1449 (modified)	Reexam number	90/010,416
	First Named Inventor	Hutton
	Patent Under Re-Exam	6108704
	Issue Date	2000/08/22
	Group Art Unit	3992
	Examiner Name	KOSOWSKI, ALEXANDER J
	Attorney Docket No.	2655-0188
	Confirmation No.	1061

Sheet 10 of 67

NON-PATENT REFERENCES			
Examiner Initials*	Cite No.	Non-patent Reference bibliographic information, where available	Notes
	10-1	AVC-655: Communication Procedure for H.222.1, July 1, 1994.	
	10-2	AVC-666: H.32X Communication Modes, Terminal Types and Interworking Scenarios, July 1994.	
	10-3	AVC-683: Update Draft H.32Z Following Grimstad Meeting, Nov. 1994.	
	10-4	AVC-696: An Example of Call Setup Procedure in a H.32Z Terminal, Nov. 1994.	
	10-5	AVC-702: Terminal to Terminal Signaling in H.32X, Oct. 24, 1994.	
	10-6	AVC-707R: Report of the Seventeenth Experts Group Meeting in Singapore (1-11 July 1994) – Part 1 and Part II, Nov. 11, 1994.	
	10-7	AVC-716: Draft Recommendation H.32X, Jan. 1995.	

Examiner Signature		Date Considered	
--------------------	--	-----------------	--

\*Examiner: Initial if reference was considered, whether or not citation is in conformance with MPEP 609. Draw a line through citation if not in conformance and not considered. Include a copy of this form with next communication to applicant. Notes: If identified, the following is provided: EA = English Abstract, T = Translation, PF = Patent Family.

N2P-IDS00236

<b>INFORMATION DISCLOSURE STATEMENT BY APPLICANT</b> FORM PTO-1449 (modified)  Sheet 11 of 67	Reexam number	90/010,416
	First Named Inventor	Hutton
	Patent Under Re-Exam	6108704
	Issue Date	2000/08/22
	Group Art Unit	3992
	Examiner Name	KOSOWSKI, ALEXANDER J
	Attorney Docket No.	2655-0188
	Confirmation No.	1061

NON-PATENT REFERENCES			
Examiner Initials*	Cite No.	Non-patent Reference bibliographic information, where available	Notes
	11-1	AVC-718: Draft H.32X, Jan. 1995.	
	11-2	AVC-743R: Report of the Eighteenth Experts Group Meeting in Kamifukuoka (24-27 January 1995), Jan. 27, 1995.	
	11-3	AVC-748: Update of Draft Recommendation H.322, May 1995.	
	11-4	AVC-750: Report of the Study Group 15 Meeting Held During 6-17 February 1995, Feb. 24, 1995.	
	11-5	AVC-752: Open Issues Towards the Stockholm Meeting, Mar. 17, 1995.	
	11-6	AVC-758: Draft Recommendation H.323 Visual Telephone Systems and Terminal Equipment for Local Area Networks Which Provide A Non-Guaranteed Quality of Service, Rev. May 12, 1995.	
	11-7	AVC-767: Logical Channel Set-up Procedure, Apr. 28, 1995.	

Examiner Signature		Date Considered	
--------------------	--	-----------------	--

\*Examiner: Initial if reference was considered, whether or not citation is in conformance with MPEP 609. Draw a line through citation if not in conformance and not considered. Include a copy of this form with next communication to applicant. Notes: If identified, the following is provided: EA = English Abstract, T = Translation, PF = Patent Family.

N2P-IDS00237

<b>INFORMATION DISCLOSURE STATEMENT BY APPLICANT</b> FORM PTO-1449 (modified)  Sheet 12 of 67	Reexam number	90/010,416
	First Named Inventor	Hutton
	Patent Under Re-Exam	6108704
	Issue Date	2000/08/22
	Group Art Unit	3992
	Examiner Name	KOSOWSKI, ALEXANDER J
	Attorney Docket No.	2655-0188
	Confirmation No.	1061

NON-PATENT REFERENCES			
Examiner Initials*	Cite No.	Non-patent Reference bibliographic information, where available	Notes
	12-1	AVC-799: Comments on Draft H.323 and H.22Z, May 11, 1995.	
	12-2	AVC-800R: Report of the Nineteenth Experts Group Meeting in Haninge (15-18 May 1995), May 18, 1995	
	12-3	AVC-813: Signaling Recommendation Within the Scope of H.323, Sept. 10, 1995.	
	12-4	AVC-819: LAN Addressing Plan in H.323, Sept. 10, 1995	
	12-5	AVC-830: Connection Management Procedures for H.323, Oct. 24-27, 1995.	
	12-6	AVC-842: Gateway, Gatekeeper and Terminal Procedures in H.323, Oct. 17, 1995.	
	12-7	Avnish Aggarwal, et al., RFC 1002: Protocol Standard for a NetBIOS Service on a TCP/UDP Transport: Detailed Specifications (Mar. 1987).	

Examiner Signature		Date Considered	
--------------------	--	-----------------	--

\*Examiner: Initial if reference was considered, whether or not citation is in conformance with MPEP 609. Draw a line through citation if not in conformance and not considered. Include a copy of this form with next communication to applicant. Notes: If identified, the following is provided: EA = English Abstract, T = Translation, PF = Patent Family.

**INFORMATION DISCLOSURE  
STATEMENT BY APPLICANT**  
FORM PTO-1449 (modified)

Sheet 13 of 67

Reexam number	90/010,416
First Named Inventor	Hutton
Patent Under Re-Exam	6108704
Issue Date	2000/08/22
Group Art Unit	3992
Examiner Name	KOSOWSKI, ALEXANDER J
Attorney Docket No.	2655-0188
Confirmation No.	1061

**NON-PATENT REFERENCES**

Examiner Initials*	Cite No.	Non-patent Reference bibliographic information, where available	Notes
	13-1	Barbara Darrow, Internet Phone Chat Software Prompts Spat; IRC Operators Rebuffed Use of Their Systems, COMPUTER RESELLER NEWS (Mar. 20, 1995).	
	13-2	Barry Michael Arons, The Audio-Graphical Interface to a Personal Integrated Telecommunications System, Masters Thesis, Massachusetts Institute of Technology (June 1984).	
	13-3	Barry Phillips, Casting the Net for New Media, OEM MAGAZINE, no. 320 (1995).	
	13-4	Belville, Sharon, "Zephyr on Athena", Athena Documentation, September 10, 1991, Version 3.	
	13-5	Ben Mesander, et al., The Client-To-Client Protocol (Aug. 12, 1994).	
	13-6	Bill Welsh, H.245 Implementors' Guide (undated but references April 1996)	
	13-7	Bob Blakley's Email to sig-dce-security, DCE Delegation Proposal Review, July 7, 1992.	

Examiner Signature		Date Considered	
--------------------	--	-----------------	--

\*Examiner: Initial if reference was considered, whether or not citation is in conformance with MPEP 609. Draw a line through citation if not in conformance and not considered. Include a copy of this form with next communication to applicant. Notes: If identified, the following is provided: EA = English Abstract, T = Translation, PF = Patent Family.

N2P-IDS00239

<b>INFORMATION DISCLOSURE STATEMENT BY APPLICANT</b> FORM PTO-1449 (modified)	Reexam number	90/010,416
	First Named Inventor	Hutton
	Patent Under Re-Exam	6108704
	Issue Date	2000/08/22
	Group Art Unit	3992
	Examiner Name	KOSOWSKI, ALEXANDER J
	Attorney Docket No.	2655-0188
	Confirmation No.	1061
Sheet 14 of 67		

NON-PATENT REFERENCES			
Examiner Initials*	Cite No.	Non-patent Reference bibliographic information, where available	Notes
	14-1	Brad Curtis Johnson, A Distributed Computing Environment Framework: An OSF Perspective (1991).	
	14-2	Brent Nordin, et al., Remote Operation Across a Network of Small Computers (Association of Computing Machinery, 1986).	
	14-3	Brian Fox, et al., GNU Finger program documentation, Free Software Foundation (1992).	
	14-4	Bruce Brown, BugNet Bug/Fix List, NEWSBYTES (Dec. 13, 1995).	
	14-5	Bruce Brown, BugNet Bug/Fix List, NEWSBYTES (Dec. 13, 1996).	
	14-6	Butler W. Lampson, et al., A Distributed Systems Architecture for the 1990's (Dec, 17, 1989),	
	14-7	Buy Memory Configured Expressly for Your Computer, SAN JOSE MERCURY NEWS (July 16, 1995).	

Examiner Signature		Date Considered	
--------------------	--	-----------------	--

\*Examiner: Initial if reference was considered, whether or not citation is in conformance with MPEP 609. Draw a line through citation if not in conformance and not considered. Include a copy of this form with next communication to applicant. Notes: If identified, the following is provided: EA = English Abstract, T = Translation, PF = Patent Family.

N2P-IDS00240

<b>INFORMATION DISCLOSURE STATEMENT BY APPLICANT</b> FORM PTO-1449 (modified)	Reexam number	90/010,416
	First Named Inventor	Hutton
	Patent Under Re-Exam	6108704
	Issue Date	2000/08/22
	Group Art Unit	3992
	Examiner Name	KOSOWSKI, ALEXANDER J
	Attorney Docket No.	2655-0188
Sheet 15 of 67	Confirmation No.	1061

NON-PATENT REFERENCES			
Examiner Initials*	Cite No.	Non-patent Reference bibliographic information, where available	Notes
	15-1	C. Anthony DellaFera, et al., Section E.4.1: Zephyr Notification Service, ATHENA TECHNICAL PLAN (July 29, 1988).	
	15-2	C. Anthony DellaFera, et al., Section E.4.1: Zephyr Notification Service, Project Athena Technical Plan (June 5, 1989).	
	15-3	C. Anthony DellaFera, et al., The Athena Notification Service: Zephyr (1987).	
	15-4	C. Anthony DellaFera, et al., The Athena Notification Service: Zephyr (Dec. 31, 1987).	
	15-5	C. Anthony DellaFera, et al., The Zephyr Notification Service (undated).	
	15-6	C. Anthony DellaFera, et al., The Zephyr Notification Service, USENIX Winter Conference, Feb. 9-12, 1988	
	15-7	C. Anthony DellaFera, The Zephyr Notification Service, MIT Project Athena, Winter Usenix Conference (Feb. 12, 1988).	

Examiner Signature		Date Considered	
--------------------	--	-----------------	--

\*Examiner: Initial if reference was considered, whether or not citation is in conformance with MPEP 609. Draw a line through citation if not in conformance and not considered. Include a copy of this form with next communication to applicant. Notes: If identified, the following is provided: EA = English Abstract, T = Translation, PF = Patent Family.

N2P-IDS00241



**INFORMATION DISCLOSURE  
STATEMENT BY APPLICANT**  
FORM PTO-1449 (modified)

Sheet 16 of 67

Reexam number	90/010,416
First Named Inventor	Hutton
Patent Under Re-Exam	6108704
Issue Date	2000/08/22
Group Art Unit	3992
Examiner Name	KOSOWSKI, ALEXANDER J
Attorney Docket No.	2655-0188
Confirmation No.	1061

**NON-PATENT REFERENCES**

Examiner Initials*	Cite No.	Non-patent Reference bibliographic information, where available	Notes
	16-1	C. Malamud, et al., RFC 1528: Principles of Operation for the TPC.INT Subdomain: Technical Procedures (Oct. 1993).	
	16-2	C. Malamud, et al., RFC 1530: Principles of Operation for the TPC.INT Subdomain: General Principles and Policy (Oct. 1993).	
	16-3	C. Sunshine, et al., IEN 135: Addressing Mobile Hosts in the ARPA Internet Environment (Oct. 1985).	
	16-4	C. Yang, RFC 1789: INETPhone: Telephone Services and Servers on Internet (Apr. 1995).	
	16-5	Calls Waiting on the Internet Although Telephone Software Makes 'Free' Long Distance Possible, it's a Long Way from Practical, KANSAS CITY STAR (July 14, 1996).	
	16-6	Carl Sunshine, IEN 178: Addressing Problems in Multi-Network Systems (Apr. 1981).	
	16-7	Charles E. Perkins, et al., A Mobile Networking System Based on Internet Protocol, IEEE PERSONAL COMMUNICATIONS (First Quarter 1994).	

Examiner Signature		Date Considered	
--------------------	--	-----------------	--

\*Examiner: Initial if reference was considered, whether or not citation is in conformance with MPEP 609. Draw a line through citation if not in conformance and not considered. Include a copy of this form with next communication to applicant. Notes: If identified, the following is provided: EA = English Abstract, T = Translation, PF = Patent Family.

N2P-IDS00242

<b>INFORMATION DISCLOSURE STATEMENT BY APPLICANT</b> FORM PTO-1449 (modified)	Reexam number	90/010,416
	First Named Inventor	Hutton
	Patent Under Re-Exam	6108704
	Issue Date	2000/08/22
	Group Art Unit	3992
	Examiner Name	KOSOWSKI, ALEXANDER J
	Attorney Docket No.	2655-0188
Sheet 17 of 67	Confirmation No.	1061

NON-PATENT REFERENCES			
Examiner Initials*	Cite No.	Non-patent Reference bibliographic information, where available	Notes
	17-1	Charlie Kaufman's Email to dmackey re DCE 1.1 Delegation Proposal for Review, June 22, 1992.	
	17-2	Chii-Ren Tsai, et al., Distributed Audit with Secure Remote Procedure Calls (1991).	
	17-3	Christopher Schmandt, et al., An Audio and Telephone Server for Multi-Media Workstations, IEEE (1988).	
	17-4	Christopher Schmandt, et al., Phone Slave: A Graphical Telecommunications Interface, Society for Information Display, 1984 International Symposium Digest of Technical Papers (June 1984).	
	17-5	Chuck Kane, List of IRC servers as of Feb. 1, 1995, available at <a href="http://ftp.funet.fi/pub/unix/irc/does/servers.950201">http://ftp.funet.fi/pub/unix/irc/does/servers.950201</a> .	
	17-6	Clinton Wilder, Pulling in the Net – InfoSeek, VocalTec Offer Search and Voice Options to Internet Users Online, INFORMATIONWEEK, no. 516 (1995).	
	17-7	Common Desktop Environment 1.0—Advanced User's and System Administrator's Guide, Addison-Wesley Publishing Co. (1995).	

Examiner Signature		Date Considered	
--------------------	--	-----------------	--

\*Examiner: Initial if reference was considered, whether or not citation is in conformance with MPEP 609. Draw a line through citation if not in conformance and not considered. Include a copy of this form with next communication to applicant. Notes: If identified, the following is provided: EA = English Abstract, T = Translation, PF = Patent Family.

N2P-IDS00243

<b>INFORMATION DISCLOSURE STATEMENT BY APPLICANT</b> FORM PTO-1449 (modified)	Reexam number	90/010,416
	First Named Inventor	Hutton
	Patent Under Re-Exam	6108704
	Issue Date	2000/08/22
	Group Art Unit	3992
	Examiner Name	KOSOWSKI, ALEXANDER J
	Attorney Docket No.	2655-0188
Sheet 18 of 67	Confirmation No.	1061

NON-PATENT REFERENCES			
Examiner Initials*	Cite No.	Non-patent Reference bibliographic information, where available	Notes
	18-1	Common Desktop Environment 1.0—User's Guide, Addison-Wesley Publishing Co. (1995).	
	18-2	Communications Connectivity Networking, MICROSOFT SYSTEMS JOURNAL, vol. 10, no. 1 (Jan. 1995).	
	18-3	Comp.Speech FAQ Archive; Comp.Speech FAQ Web Page, COMP.SPEECH NEWSGROUP. July 17, 1995)	
	18-4	Comp.Speech FAQ Weekly Reminder, COMP.SPEECH NEWSGROUP (June 21, 1995)	
	18-5	Contents, Preface, and Index to Open Software Foundation, X/Open Preliminary Specification—X/Open DCE: Authentication and Security Services (March 1996).	
	18-6	Conversation Excerpt from ftp://svr-ftp.eng.cam.ac.uk/pub/pub/comp.speech/archive/subject5xxx.txt accessed on 11/28/2007	
	18-7	Craig Crossman, Free Calls on Internet are CB-Style No Longer, MIAMI HERALD (June 26, 1995).	

Examiner Signature		Date Considered	
--------------------	--	-----------------	--

\*Examiner: Initial if reference was considered, whether or not citation is in conformance with MPEP 609. Draw a line through citation if not in conformance and not considered. Include a copy of this form with next communication to applicant. Notes: If identified, the following is provided: EA = English Abstract, T = Translation, PF = Patent Family.

<b>INFORMATION DISCLOSURE STATEMENT BY APPLICANT</b> FORM PTO-1449 (modified)	Reexam number	90/010,416
	First Named Inventor	Hutton
	Patent Under Re-Exam	6108704
	Issue Date	2000/08/22
	Group Art Unit	3992
	Examiner Name	KOSOWSKI, ALEXANDER J
	Attorney Docket No.	2655-0188
Sheet 19 of 67	Confirmation No.	1061

NON-PATENT REFERENCES			
Examiner Initials*	Cite No.	Non-patent Reference bibliographic information, where available	Notes
	19-1	Craig Crossman, Make Long Distance Calls Via the Internet, RECORD (July 3, 1995).	
	19-2	D. O'Mahoney, 1st Generation Internet Phones (1998).	
	19-3	D. Reed, RFC 1324: A Discussion on Computer Network Conferencing (May 1992).	
	19-4	D. Zimmerman, RFC 1288: The Finger User Information Protocol (Dec. 1991).	
	19-5	Dale Skran, Draft ITU-T Recommendation H.225.0— Line Transmission of Non-Telephone Signals, Media Stream Packetization and Synchronization on Non-Guaranteed Quality of Service LANs (May 28, 1996).	
	19-6	Dale Skran, ed. ASN.1 for H.225.0 (June 18, 1996).	
	19-7	Dan Cohen, IEN 31: On Name, Addresses and Routings (II) (Apr. 28, 1978).	

Examiner Signature		Date Considered	
--------------------	--	-----------------	--

\*Examiner: Initial if reference was considered, whether or not citation is in conformance with MPEP 609. Draw a line through citation if not in conformance and not considered. Include a copy of this form with next communication to applicant. Notes: If identified, the following is provided: EA = English Abstract, T = Translation, PF = Patent Family.

N2P-IDS00245

<b>INFORMATION DISCLOSURE STATEMENT BY APPLICANT</b> FORM PTO-1449 (modified)  Sheet 20 of 67	Reexam number	90/010,416
	First Named Inventor	Hutton
	Patent Under Re-Exam	6108704
	Issue Date	2000/08/22
	Group Art Unit	3992
	Examiner Name	KOSOWSKI, ALEXANDER J
	Attorney Docket No.	2655-0188
Confirmation No.	1061	

**NON-PATENT REFERENCES**

Examiner Initials*	Cite No.	Non-patent Reference bibliographic information, where available	Notes
	20-1	Dan Keating, Ring! It's Computer Calling Phone By Internet Has Gotten Better, MIAMI HERALD (May 22, 1996).	
	20-2	Daniel C. Swinehart, Telephone Management in the Etherphone System, IEEE (1987).	
	20-3	Daniel H. Craft, Resource Management in a Decentralized System, OPERATING SYSTEMS REVIEW, vol. 17, no. 5 (Association for Computing Machinery, Oct. 1983).	
	20-4	Danny Cohen, IEN 23: On Name, Addresses and Routings (Jan. 23, 1978).	
	20-5	Dave Lindbergh, H.323 Encryption, Document: CNC-96-22 (April 15, 1996).	
	20-6	David D. Clark, RFC 814: Name, Addresses, Ports, and Routes (July 1982).	
	20-7	David Gertler, Hardware and Software Tidbits from CEBIT, SEYBOLD REPORT ON DESKTOP PUBLISHING, vol. 9, no. 8 (Apr. 3, 1995).	

Examiner Signature		Date Considered	
--------------------	--	-----------------	--

\*Examiner: Initial if reference was considered, whether or not citation is in conformance with MPEP 609. Draw a line through citation if not in conformance and not considered. Include a copy of this form with next communication to applicant. Notes: If identified, the following is provided: EA = English Abstract, T = Translation, PF = Patent Family.

<b>INFORMATION DISCLOSURE STATEMENT BY APPLICANT</b> FORM PTO-1449 (modified)  Sheet 21 of 67	Reexam number	90/010,416
	First Named Inventor	Hutton
	Patent Under Re-Exam	6108704
	Issue Date	2000/08/22
	Group Art Unit	3992
	Examiner Name	KOSOWSKI, ALEXANDER J
	Attorney Docket No.	2655-0188
	Confirmation No.	1061

**NON-PATENT REFERENCES**

Examiner Initials*	Cite No.	Non-patent Reference bibliographic information, where available	Notes
	21-1	David Hafke, New on the Net -- Talk It Up, WINDOWS MAGAZINE, no. 711 (1996).	
	21-2	David Harvey, All the News That's Fit to Speak, NETGUIDE, no. 301 (1996).	
	21-3	David R. Cheriton, et al., A Decentralized Naming Facility (Stanford University, Feb. 1, 1986).	
	21-4	David R. Cheriton, The V Distributed System, COMMUNICATIONS OF THE ACM, vol. 31, no. 3 (Apr. 1988).	
	21-5	David Rapp, I've Got to Get a Message to You, Instant Messaging Started as an MIT Computer-Science Department Project, TECHNOLOGY REVIEW (2002).	
	21-6	DCE 1.0 Security Technology, architectural overview documents, Walter Tuvell, Feb 1997	
	21-7	DCE 1.1 Security Technology, architectural overview documents May 1994	

Examiner Signature		Date Considered	
--------------------	--	-----------------	--

\*Examiner: Initial if reference was considered, whether or not citation is in conformance with MPEP 609. Draw a line through citation if not in conformance and not considered. Include a copy of this form with next communication to applicant. Notes: If identified, the following is provided: EA = English Abstract, T = Translation, PF = Patent Family.

<b>INFORMATION DISCLOSURE STATEMENT BY APPLICANT</b> FORM PTO-1449 (modified)	Reexam number	90/010,416
	First Named Inventor	Hutton
	Patent Under Re-Exam	6108704
	Issue Date	2000/08/22
	Group Art Unit	3992
	Examiner Name	KOSOWSKI, ALEXANDER J
	Attorney Docket No.	2655-0188
Confirmation No.	1061	
Sheet 22 of 67		

NON-PATENT REFERENCES			
Examiner Initials*	Cite No.	Non-patent Reference bibliographic information, where available	Notes
	22-1	DCE RPC Internals and Data Structures (Aug. 1993).	
	22-2	Dean Adams, ed., Security Survival: An indispensable guide to securing your business, X/Open Co. (1996).	
	22-3	Decided H.225.0 (June 19, 1996).	
	22-4	Derek C. Oppen, et al., The Clearinghouse: A Decentralized Agent for Locating Named Objects in a Distributed Environment (Association for Computing Machinery, 1983).	
	22-5	Description of New Zephyr Protocol (undated).	
	22-6	Digiphone Specifications, from Q1.11 of Section 1 of the comp.speech FAQ Home Page (dated Jan. 6, 1997)	
	22-7	Douglas B. Terry, et al., The Berkeley Internet Name Domain Server, USENIX Association Software Tools Users Group, Summer Conference, Salt Lake City, Utah (June 12-15, 1984).	

Examiner Signature		Date Considered	
--------------------	--	-----------------	--

\*Examiner: Initial if reference was considered, whether or not citation is in conformance with MPEP 609. Draw a line through citation if not in conformance and not considered. Include a copy of this form with next communication to applicant. Notes: If identified, the following is provided: EA = English Abstract, T = Translation, PF = Patent Family.

N2P-IDS00248

**INFORMATION DISCLOSURE  
STATEMENT BY APPLICANT**  
FORM PTO-1449 (modified)

Sheet 23 of 67

Reexam number	90/010,416
First Named Inventor	Hutton
Patent Under Re-Exam	6108704
Issue Date	2000/08/22
Group Art Unit	3992
Examiner Name	KOSOWSKI, ALEXANDER J
Attorney Docket No.	2655-0188
Confirmation No.	1061

**NON-PATENT REFERENCES**

Examiner Initials*	Cite No.	Non-patent Reference bibliographic information, where available	Notes
	23-1	Douglas B. Terry, Structure freeName Management for Evolving Distributed Environments, IEEE 6th International Conference on Distributed Computing Systems, Cambridge, Massachusetts (May 19-23, 1986).	
	23-2	Douglas Brian Terry, Distributed Name Servers: Naming and Caching in Large Distributed Computing Environments, Ph.D. Thesis, University of California, Berkeley (Feb. 21, 1985).	
	23-3	Douglas E. Comer, INTERNETWORKING WITH TCP/IP: VOL. 1: PRINCIPLES, PROTOCOLS, AND ARCHITECTURE, 1st ed. (Prentice-Hall, 1988).	
	23-4	Douglas E. Comer, INTERNETWORKING WITH TCP/IP: VOL. 1: PRINCIPLES, PROTOCOLS, AND ARCHITECTURE, 3d ed. (Prentice-Hall, 1995).	
	23-5	Douglas E. Comer, INTERNETWORKING WITH TCP/IP: VOLUME 1: PRINCIPLES, PROTOCOLS, AND ARCHITECTURES, 2d ed. (Prentice-Hall, 1991).	
	23-6	Douglas W. Johnson, Internet-Connected Phone Calls Dial in to Lower Prices, COMPUTERWORLD (Feb. 19, 1996).	
	23-7	Draft ITU-T Recommendation G.723—Dual Rate Speech Coder for Multimedia Communications Transmitting at 5.3 & 6.3 KBIT/S (Oct. 17, 1995).	

Examiner Signature		Date Considered	
--------------------	--	-----------------	--

\*Examiner: Initial if reference was considered, whether or not citation is in conformance with MPEP 609. Draw a line through citation if not in conformance and not considered. Include a copy of this form with next communication to applicant. Notes: If identified, the following is provided: EA = English Abstract, T = Translation, PF = Patent Family.

N2P-IDS00249



<b>INFORMATION DISCLOSURE STATEMENT BY APPLICANT</b> FORM PTO-1449 (modified)  Sheet 24 of 67	Reexam number	90/010,416
	First Named Inventor	Hutton
	Patent Under Re-Exam	6108704
	Issue Date	2000/08/22
	Group Art Unit	3992
	Examiner Name	KOSOWSKI, ALEXANDER J
	Attorney Docket No.	2655-0188
	Confirmation No.	1061

NON-PATENT REFERENCES			
Examiner Initials*	Cite No.	Non-patent Reference bibliographic information, where available	Notes
	24-1	Draft ITU-T Recommendation H.323 Line Transmission of Non-Telephone Signals: Visual Telephone Systems and Equipment for Local Area Networks Which Provide a Non-Guaranteed Quality of Service, (May 28, 1996).	
	24-2	Draft Recommendation H.323 Visual Telephone Systems and Terminal Equipment for Local Area Networks Which Provide A Non-Guaranteed Quality of Service, Sept. 8, 1995.	
	24-3	Draft Recommendation H.323—Visual Telephone Systems and Equipment for Local Area Networks Which Provide A Non-Guaranteed Quality of Service (May 28, 1996).	
	24-4	E.D. Sykas, et al., Overview of the CCITT X500 Recommendations Series (Butterworth-Heinemann, 1991).	
	24-5	Electric Magic Company Sales Invoices, February 23, 1995 thru December 3, 1995.	
	24-6	Electric Magic Company, Beta Test License Agreement (dated May 30, 1995)	
	24-7	Elizabeth Feinler, et al., RFC 810: DoD Internet Host Table Specification (Mar. 1, 1982).	

Examiner Signature		Date Considered	
--------------------	--	-----------------	--

\*Examiner: Initial if reference was considered, whether or not citation is in conformance with MPEP 609. Draw a line through citation if not in conformance and not considered. Include a copy of this form with next communication to applicant. Notes: If identified, the following is provided: EA = English Abstract, T = Translation, PF = Patent Family.

<b>INFORMATION DISCLOSURE STATEMENT BY APPLICANT</b> FORM PTO-1449 (modified)  Sheet 25 of 67	Reexam number	90/010,416
	First Named Inventor	Hutton
	Patent Under Re-Exam	6108704
	Issue Date	2000/08/22
	Group Art Unit	3992
	Examiner Name	KOSOWSKI, ALEXANDER J
	Attorney Docket No.	2655-0188
	Confirmation No.	1061

NON-PATENT REFERENCES			
Examiner Initials*	Cite No.	Non-patent Reference bibliographic information, where available	Notes
	25-1	Ellen Massmer, PictureTel Brings Video to the Lan Network World (Sept. 4, 1995).	
	25-2	E-mail from Dale Skran to jtoga@ibeam.jf.intel.com, phone numbers for email list (Jan. 6, 1997).	
	25-3	E-mail from Dale Skran to jtoga@ibeam.jf.intel.com, mailing list to enter (Jan. 6, 1997).	
	25-4	E-mail from Ofer Shapiro to Bob Bell, et al., RE: Destination side gateway problem (July 29, 1996).	
	25-5	E-mail from Sakae Okubo to Experts of ITU-T SG16 Q.12/16, Q.13/16 and Q.14/16, Notice of the Q.12-14/16 Sunriver meeting (July 17, 1997).	
	25-6	Email from Sakae Okubo to yves.robin-champigneu10issy.cnet.fr, et al., Working tools of SG16 experts groups (May 8, 1997).	
	25-7	E-mail from Vineet Kumar to h323implementors@mailbag.jf.intel.com Receiver associating a logical channel with a RTP stream (Aug. 5, 1996).	

Examiner Signature		Date Considered	
--------------------	--	-----------------	--

\*Examiner: Initial if reference was considered, whether or not citation is in conformance with MPEP 609. Draw a line through citation if not in conformance and not considered. Include a copy of this form with next communication to applicant. Notes: If identified, the following is provided: EA = English Abstract, T = Translation, PF = Patent Family.

N2P-IDS00251

**INFORMATION DISCLOSURE  
STATEMENT BY APPLICANT**  
FORM PTO-1449 (modified)

Sheet 26 of 67

Reexam number	90/010,416
First Named Inventor	Hutton
Patent Under Re-Exam	6108704
Issue Date	2000/08/22
Group Art Unit	3992
Examiner Name	KOSOWSKI, ALEXANDER J
Attorney Docket No.	2655-0188
Confirmation No.	1061

**NON-PATENT REFERENCES**

Examiner Initials*	Cite No.	Non-patent Reference bibliographic information, where available	Notes
	26-1	Erdos, Marlena and Pato, Joseph, "Extending the OSF DCE Authorization System to Support Practical Delegation," February 11, 1993	
	26-2	Eric C. Rosen, IEN 183: Logical Addressing (May 1981).	
	26-3	Eric C. Rosen, IEN 188: Issues in Interneting Part 3: Addressing (June 1981).	
	26-4	Etherphone: Collected Papers 1987-1988, Xerox PARC, CSL-89-2 (May 1989).	
	26-5	Eve M. Schooler, Case Study: Multimedia Conference Control in a Packet-Switched Teleconferencing System, JOURNAL OF INTERNETWORKING: RESEARCH AND EXPERIENCE, vol. 4, no. 2 (June 1993).	
	26-6	Eve M. Schooler, et al., An Architecture for Multimedia Connection Management, Proceedings IEEE 4th Comsoc International Workshop on Multimedia Communications, MM '92, Monterey, California (Apr. 1992).	
	26-7	Eve M. Schooler, The Connection Control Protocol: Architecture Overview (Jan. 28, 1992).	

Examiner Signature		Date Considered	
--------------------	--	-----------------	--

\*Examiner: Initial if reference was considered, whether or not citation is in conformance with MPEP 609. Draw a line through citation if not in conformance and not considered. Include a copy of this form with next communication to applicant. Notes: If identified, the following is provided: EA = English Abstract, T = Translation, PF = Patent Family.

N2P-IDS00252

**INFORMATION DISCLOSURE  
STATEMENT BY APPLICANT**  
FORM PTO-1449 (modified)

Sheet 27 of 67

Reexam number	90/010,416
First Named Inventor	Hutton
Patent Under Re-Exam	6108704
Issue Date	2000/08/22
Group Art Unit	3992
Examiner Name	KOSOWSKI, ALEXANDER J
Attorney Docket No.	2655-0188
Confirmation No.	1061

**NON-PATENT REFERENCES**

Examiner Initials*	Cite No.	Non-patent Reference bibliographic information, where available	Notes
	27-1	Eve M. Schooler, The Connection Control Protocol: Specification, Version 1.1 (Jan. 29, 1992).	
	27-2	Eve M. Schooler, The Impact of Scaling on Multimedia Connection Architecture, MULTIMEDIA SYSTEMS, vol. 1 (Association for Computing Machinery, 1993).	
	27-3	Exportability of DCE Multi-Crypto Feature by Walter Tuvell, March 5, 1996.	
	27-4	F. Anklesaria, et al., RFC 1436: The Internet Gopher Protocol (A Distributed Document Search and Retrieval Protocol) (Mar. 1993).	
	27-5	FAQ: How Can I Use the Internet as a Telephone, Ver. 0.2 (Apr. 27, 1995)	
	27-6	FAQ: How Can I Use the Internet as a Telephone, Ver. 0.4 (Feb. 23, 1996)	
	27-7	Fax from Ryan Holmquist to Dale Skran (May 30, 1996).	

Examiner Signature		Date Considered	
--------------------	--	-----------------	--

\*Examiner: Initial if reference was considered, whether or not citation is in conformance with MPEP 609. Draw a line through citation if not in conformance and not considered. Include a copy of this form with next communication to applicant. Notes: If identified, the following is provided: EA = English Abstract, T = Translation, PF = Patent Family.

N2P-IDS00253

<b>INFORMATION DISCLOSURE STATEMENT BY APPLICANT</b> FORM PTO-1449 (modified)	Reexam number	90/010,416
	First Named Inventor	Hutton
	Patent Under Re-Exam	6108704
	Issue Date	2000/08/22
	Group Art Unit	3992
	Examiner Name	KOSOWSKI, ALEXANDER J
	Attorney Docket No.	2655-0188
Sheet 28 of 67	Confirmation No.	1061

NON-PATENT REFERENCES			
Examiner Initials*	Cite No.	Non-patent Reference bibliographic information, where available	Notes
	28-1	FLEXIm v3.0 Programmer's Guide, Globetrotter Software, Inc. (Aug. 1994).	
	28-2	Full Duplex Internet Voice Comms Available, NEWSBYTES (Feb. 14, 1995).	
	28-3	Gary A. Thom, H.323: The Multimedia Communications Standard for Local Area Networks, IEEE Communications Magazine (December 2006).	
	28-4	Gilbert Held, The ABCs of IP Addressing, CRC PRESS LLC (2002).	
	28-5	Gligor, et al. "On Inter-realm Authentication in Large Distributed Systems" May 2, 1992	
	28-6	Google Groups "CyberPhone" Search Results, search conducted on November 28, 2007.	
	28-7	Goretsky, Aryeh "PowWow Quick Installation Guide", 1996	

Examiner Signature		Date Considered	
--------------------	--	-----------------	--

\*Examiner: Initial if reference was considered, whether or not citation is in conformance with MPEP 609. Draw a line through citation if not in conformance and not considered. Include a copy of this form with next communication to applicant. Notes: If identified, the following is provided: EA = English Abstract, T = Translation, PF = Patent Family.

**INFORMATION DISCLOSURE  
STATEMENT BY APPLICANT**  
FORM PTO-1449 (modified)

Sheet 29 of 67

Reexam number	90/010,416
First Named Inventor	Hutton
Patent Under Re-Exam	6108704
Issue Date	2000/08/22
Group Art Unit	3992
Examiner Name	KOSOWSKI, ALEXANDER J
Attorney Docket No.	2655-0188
Confirmation No.	1061

**NON-PATENT REFERENCES**

Examiner Initials*	Cite No.	Non-patent Reference bibliographic information, where available	Notes
	29-1	Green, Andrew, NetPhone Tasks and Plans, Email, 2 pages (printed 02/02/95)	
	29-2	Greg Wood. Computer VAR Takes His First Computer Telephony Plunge, COMPUTER TELEPHONY (Sept. 1996).	
	29-3	Gursharan S. Sidu, et al., Inside AppleTalk, 2d ed. (Addison-Wesley Publishing Co., 1990).	
	29-4	H. Schulzrinne, et al., RFC 1889: RTP: A Transport Protocol for Real-Time Applications (Jan. 1996).	
	29-5	Handwritten Notes, Electric Magic Company (dated July 22, 1994 thru August 30, 1995)	
	29-6	How Can I use the Internet as a telephone? from Q1.11 of Section 1 of the comp.speech FAQ Home Page (dated March 19, 1996)	
	29-7	Hussein M. Abdel-Wahab, XTV: A Framework for Sharing X Window Clients in Remote Synchronous Collaboration, IEEE Conference on Communications Software: Communications for Distributed Applications & Systems (Apr. 1991).	

Examiner Signature		Date Considered	
--------------------	--	-----------------	--

\*Examiner: Initial if reference was considered, whether or not citation is in conformance with MPEP 609. Draw a line through citation if not in conformance and not considered. Include a copy of this form with next communication to applicant. Notes: If identified, the following is provided: EA = English Abstract, T = Translation, PF = Patent Family.

<b>INFORMATION DISCLOSURE STATEMENT BY APPLICANT</b> FORM PTO-1449 (modified)  Sheet 30 of 67	Reexam number	90/010,416
	First Named Inventor	Hutton
	Patent Under Re-Exam	6108704
	Issue Date	2000/08/22
	Group Art Unit	3992
	Examiner Name	KOSOWSKI, ALEXANDER J
	Attorney Docket No.	2655-0188
	Confirmation No.	1061

**NON-PATENT REFERENCES**

Examiner Initials*	Cite No.	Non-patent Reference bibliographic information, where available	Notes
	30-1	I C. Weider, et al., RFC 1727: A Vision of an Integrated Information Service (Dec. 1994).	
	30-2	Inder Gopal, et al., Directories for Networks with Casually Connected Users (IEEE, 1988).	
	30-3	Information Technology – Database Language SQL (Proposed revised text of DIS 9075), DIGITAL EQUIPMENT CORP. (July 1992).	
	30-4	InterFACE from Hijinx Specifications, from Q1.11 of Section 1 of the comp.speech FAQ Home Page (dated March 19, 1996)	
	30-5	Internet Phone from VocalTec Specifications, from Q1.11 of Section 1 of the comp.speech FAQ Home Page (dated March 19, 1996)	
	30-6	Internet PHONE Release 4, Users Manual, VocalTech 1996.	
	30-7	Internet Telephone Companies Racing to Market, VOICE TECHNOLOGY & SERVICES NEWS, vol. 14 no. 20 (Oct. 3, 1995).	

Examiner Signature		Date Considered	
--------------------	--	-----------------	--

\*Examiner: Initial if reference was considered, whether or not citation is in conformance with MPEP 609. Draw a line through citation if not in conformance and not considered. Include a copy of this form with next communication to applicant. Notes: If identified, the following is provided: EA = English Abstract, T = Translation, PF = Patent Family.

**INFORMATION DISCLOSURE  
STATEMENT BY APPLICANT**  
FORM PTO-1449 (modified)

Sheet 31 of 67

Reexam number	90/010,416
First Named Inventor	Hutton
Patent Under Re-Exam	6108704
Issue Date	2000/08/22
Group Art Unit	3992
Examiner Name	KOSOWSKI, ALEXANDER J
Attorney Docket No.	2655-0188
Confirmation No.	1061

**NON-PATENT REFERENCES**

Examiner Initials*	Cite No.	Non-patent Reference bibliographic information, where available	Notes
	31-1	INTRODUCTION TO OSF DCE (Prentice-Hall, Inc., 1992).	
	31-2	ITU-T Recommendation X.500--Information technology -- Open Systems Interconnection -- The Directory: Overview of concepts, models and services (Aug. 1997).	
	31-3	ITU-T Recommendation X.501--Information technology -- Open Systems Interconnection -- The Directory: Models (Aug. 1997).	
	31-4	J. Oikarinen, et al., RFC 1459: Internet Relay Chat Protocol (May 1993).	
	31-5	J. Pato, Hierarchical Trust Relationships for Inter-Cell Authentication, Slides, (July 7 1992).	
	31-6	J. Pato, RFC 7.0: Hierarchical Trust Relationships for Inter-Cell Authentication (July 1992).	
	31-7	J. Postel, et al., RFC 959: File Transfer Protocol (FTP) (Oct. 1985).	

Examiner Signature		Date Considered	
--------------------	--	-----------------	--

\*Examiner: Initial if reference was considered, whether or not citation is in conformance with MPEP 609. Draw a line through citation if not in conformance and not considered. Include a copy of this form with next communication to applicant. Notes: If identified, the following is provided: EA = English Abstract, T = Translation, PF = Patent Family.



<b>INFORMATION DISCLOSURE STATEMENT BY APPLICANT</b> FORM PTO-1449 (modified)  Sheet 32 of 67	Reexam number	90/010,416
	First Named Inventor	Hutton
	Patent Under Re-Exam	6108704
	Issue Date	2000/08/22
	Group Art Unit	3992
	Examiner Name	KOSOWSKI, ALEXANDER J
	Attorney Docket No.	2655-0188
	Confirmation No.	1061

NON-PATENT REFERENCES			
Examiner Initials*	Cite No.	Non-patent Reference bibliographic information, where available	Notes
	32-1	J. Postel, RFC 765: File Transfer Protocol (June 1980).	
	32-2	J. Postel, RFC 925: Multi-LAN Address Resolution (Oct. 1984).	
	32-3	J. Saltzer, RFC 1498: On the Naming and Binding of Network Destinations (Aug. 1993).	
	32-4	Jack Rickard, Voice Over Internet – the Internet Phone, BOARDWATCH MAGAZINE, vol. 9, no. 4 (Apr. 1995).	
	32-5	James M. Bloom, et al., Experiences Implementing BIND, A Distributed Name Server for the DARPA Internet (June 9-13, 1986).	
	32-6	James Martin, et al., TCP/IP NETWORKING: ARCHITECTURE, ADMINISTRATION, AND PROGRAMMING (Prentice Hall, 1994).	
	32-7	James Staten, NetPhone 1.2 Calls the Web, MACWEEK, vol. 9 no. 27 (July 10, 1995).	

Examiner Signature		Date Considered	
--------------------	--	-----------------	--

\*Examiner: Initial if reference was considered, whether or not citation is in conformance with MPEP 609. Draw a line through citation if not in conformance and not considered. Include a copy of this form with next communication to applicant. Notes: If identified, the following is provided: EA = English Abstract, T = Translation, PF = Patent Family.

N2P-IDS00258

**INFORMATION DISCLOSURE  
STATEMENT BY APPLICANT**  
FORM PTO-1449 (modified)

Sheet 33 of 67

Reexam number	90/010,416
First Named Inventor	Hutton
Patent Under Re-Exam	6108704
Issue Date	2000/08/22
Group Art Unit	3992
Examiner Name	KOSOWSKI, ALEXANDER J
Attorney Docket No.	2655-0188
Confirmation No.	1061

**NON-PATENT REFERENCES**

Examiner Initials*	Cite No.	Non-patent Reference bibliographic information, where available	Notes
	33-1	Jennifer G. Steiner, et al., Kerberos: An Authentication Service for Open Network Systems, USENIX Winter Conference, Dallas, Texas (Feb. 9-12, 1988).	
	33-2	Joe Maloney, DCE: Focus on Security, the Internet and the Future (printed 4/25/2002, date unknown)	
	33-3	Joe Pato, et al., Distributed Computing Environment (DCE) Design of the Security Services and Facilities (Aug. 10, 1992).	
	33-4	Joe Pato, Extending the DCE Authorization Model to Support Practical Delegation—Extended Summary (July 7 1992).	
	33-5	Joe Pato, RFC 3.0: Extending the DCE Authorization Model to Support Practical Delegation—Extended Summary (June 1992).	
	33-6	Joe Pato, RFC 6.0: A Generic Interface for Extended Registry Attributes (June 1992).	
	33-7	John A. Pershing, Jr., et al., IEN 162: Transport, Addressing, and Routing in the Wideband Net (Oct. 1980).	

Examiner Signature		Date Considered	
--------------------	--	-----------------	--

\*Examiner: Initial if reference was considered, whether or not citation is in conformance with MPEP 609. Draw a line through citation if not in conformance and not considered. Include a copy of this form with next communication to applicant. Notes: If identified, the following is provided: EA = English Abstract, T = Translation, PF = Patent Family.

**INFORMATION DISCLOSURE  
STATEMENT BY APPLICANT**  
FORM PTO-1449 (modified)

Sheet 34 of 67

Reexam number	90/010,416
First Named Inventor	Hutton
Patent Under Re-Exam	6108704
Issue Date	2000/08/22
Group Art Unit	3992
Examiner Name	KOSOWSKI, ALEXANDER J
Attorney Docket No.	2655-0188
Confirmation No.	1061

**NON-PATENT REFERENCES**

Examiner Initials*	Cite No.	Non-patent Reference bibliographic information, where available	Notes
	34-1	John F. Shoch, IEN 19: Inter-Network Naming, Addressing, and Routing (Jan. 1978).	
	34-2	John Ioannidis, et. al., IP-based Protocols for Mobile Internetworking, COLUMBIA UNIV., DEPT. OF COMPUTER SCIENCE (1991).	
	34-3	John R. Pickens, et al., RFC 756: The NIC Name Server – A Datagram Based Information Utility (July 1979).	
	34-4	John T. Kohl, The Zephyr Notification Service, First International Athena Technical Conference (Apr. 11, 1991).	
	34-5	John Veizades, et al., Service Location Protocol, INTERNET DRAFT (May 2, 1995).	
	34-6	Jon Hill, et al., Pow Wow, PC MAGAZINE, vol. 15 no. 17 (Oct. 8, 1996).	
	34-7	Jon Hill, TeleVox, PC MAGAZINE, vol. 15 no. 17 (Oct. 8, 1996).	

Examiner Signature		Date Considered	
--------------------	--	-----------------	--

\*Examiner: Initial if reference was considered, whether or not citation is in conformance with MPEP 609. Draw a line through citation if not in conformance and not considered. Include a copy of this form with next communication to applicant. Notes: If identified, the following is provided: EA = English Abstract, T = Translation, PF = Patent Family.

N2P-IDS00260

<b>INFORMATION DISCLOSURE STATEMENT BY APPLICANT</b> FORM PTO-1449 (modified)  Sheet 35 of 67	Reexam number	90/010,416
	First Named Inventor	Hutton
	Patent Under Re-Exam	6108704
	Issue Date	2000/08/22
	Group Art Unit	3992
	Examiner Name	KOSOWSKI, ALEXANDER J
	Attorney Docket No.	2655-0188
Confirmation No.	1061	

NON-PATENT REFERENCES			
Examiner Initials*	Cite No.	Non-patent Reference bibliographic information, where available	Notes
	35-1	Jon Livesey, Inter-process Communication and Naming in the Mininet System, Eighteenth Annual IEEE Computer Society International Conference, San Francisco, California (1979).	
	35-2	Jon Postel, RFC 921: Domain Name System Implementation Schedule — Revised (Oct. 1984).	
	35-3	José M. Bernabeu-Auban, et al., Optimizing a Generalized Polling Protocol for Resource Finding over a Multiple Access Channel, COMPUTER NETWORKS AND ISDN SYSTEMS 27 (1995).	
	35-4	Josina M. Arfman, et al., Project Athena: Supporting Distributed Computing at MIT, IBM SYSTEMS JOURNAL (1992).	
	35-5	K. Harrenstien, et al., RFC 811: Hostname Server (Oct. 1985).	
	35-6	K. Harrenstien, et al., RFC 952: DoD Internet Host Table Specification (Oct. 1985).	
	35-7	K. Harrenstien, RFC 742: Name/Finger (Dec. 30, 1977).	

Examiner Signature		Date Considered	
--------------------	--	-----------------	--

\*Examiner: Initial if reference was considered, whether or not citation is in conformance with MPEP 609. Draw a line through citation if not in conformance and not considered. Include a copy of this form with next communication to applicant. Notes: If identified, the following is provided: EA = English Abstract, T = Translation, PF = Patent Family.

<b>INFORMATION DISCLOSURE STATEMENT BY APPLICANT</b> FORM PTO-1449 (modified)  Sheet 36 of 67	Reexam number	90/010,416
	First Named Inventor	Hutton
	Patent Under Re-Exam	6108704
	Issue Date	2000/08/22
	Group Art Unit	3992
	Examiner Name	KOSOWSKI, ALEXANDER J
	Attorney Docket No.	2655-0188
	Confirmation No.	1061

NON-PATENT REFERENCES			
Examiner Initials*	Cite No.	Non-patent Reference bibliographic information, where available	Notes
	36-1	Kahane, Opher et al., "Call Management Agent System Specification" VoIP Forum Technical Committee Contribution (dated Aug. 15, 1996)	
	36-2	Karl Auerbach, et al., RFC 1001: Protocol Standard for a NetBIOS Service on a TCP/UDP Transport: Concepts and Methods (Mar. 1987).	
	36-3	Keith A. Lantz, et al., Towards a Universal Directory Service, 4th PODC Conference Proceedings (Association for Computing Machinery, 1985).	
	36-4	Ken Harrenstien, et al., RFC 811: Hostnames Server (Mar. 1, 1982).	
	36-5	Ken Harrenstien, RFC 811: Hostnames Server (Mar. 1, 1982).	
	36-6	Ken Harrenstien, RFC 812: Nicname/Whois (Mar. 1, 1982).	
	36-7	Kenneth Hart, Startups, industry mainstays add to Internet phone menu, COMMUNICATIONSWEEK INT'L (Nov. 27, 1995).	

Examiner Signature		Date Considered	
--------------------	--	-----------------	--

\*Examiner: Initial if reference was considered, whether or not citation is in conformance with MPEP 609. Draw a line through citation if not in conformance and not considered. Include a copy of this form with next communication to applicant. Notes: If identified, the following is provided: EA = English Abstract, T = Translation, PF = Patent Family.

<b>INFORMATION DISCLOSURE STATEMENT BY APPLICANT</b> FORM PTO-1449 (modified)	Reexam number	90/010,416
	First Named Inventor	Hutton
	Patent Under Re-Exam	6108704
	Issue Date	2000/08/22
	Group Art Unit	3992
	Examiner Name	KOSOWSKI, ALEXANDER J
	Attorney Docket No.	2655-0188
Sheet 37 of 67	Confirmation No.	1061

NON-PATENT REFERENCES			
Examiner Initials*	Cite No.	Non-patent Reference bibliographic information, where available	Notes
	37-1	Klaus Zeuge, et al., The Client-to-Client Protocol (CTCP) (published no later than Aug. 12, 1994).	
	37-2	Kohl, John T., "Zephyr Installation and Operation Guide", DRAFT - November 20, 1989	
	37-3	Koster, Steven "The Phone Companies Worst Nightmare" Hotwired, April 1995.	
	37-4	L. Landweber, et al., Architecture of the CSNET Name Server (Association for Computing Machinery, 1983).	
	37-5	L. Peter Deutsch, RFC 606: Host Names On-Line (Dec. 1973).	
	37-6	Larry L. Peterson, A Yellow-Pages Service for a Local-Area Network (Association for Computing Machinery, 1988).	
	37-7	Larry L. Peterson, The Profile Naming Service, ACM TRANSACTIONS ON COMPUTER SYSTEMS, vol. 6, No. 4, (Nov. 1988).	

Examiner Signature		Date Considered	
--------------------	--	-----------------	--

\*Examiner: Initial if reference was considered, whether or not citation is in conformance with MPEP 609. Draw a line through citation if not in conformance and not considered. Include a copy of this form with next communication to applicant. Notes: If identified, the following is provided: EA = English Abstract, T = Translation, PF = Patent Family.

<b>INFORMATION DISCLOSURE STATEMENT BY APPLICANT</b> FORM PTO-1449 (modified)  Sheet 38 of 67	Reexam number	90/010,416
	First Named Inventor	Hutton
	Patent Under Re-Exam	6108704
	Issue Date	2000/08/22
	Group Art Unit	3992
	Examiner Name	KOSOWSKI, ALEXANDER J
	Attorney Docket No.	2655-0188
	Confirmation No.	1061

NON-PATENT REFERENCES			
Examiner Initials*	Cite No.	Non-patent Reference bibliographic information, where available	Notes
	38-1	Lisa Zahn, et al., Network Computing Architecture, Prentice Hall (1990).	
	38-2	List of Names from a DCE Meeting; attendees from DISA, OSF, DEC, Mitre, HP, Open Market and others (undated)	
	38-3	Listserve postings by Jon Postel, Dynamic Updated Proposal, dated Sept. 1 and 9, 1993.	
	38-4	Listserve postings by Susan Thomson, DNS Dynamic Updates, dated July 14, 1994.	
	38-5	Lon Wagner, New Software Lets Users Talk for Cheap, VIRGINIAN-PILOT (Mar. 26, 1995).	
	38-6	M. Bever, et al., Distributed Systems, OSF DCE, and Beyond (1993).	
	38-7	M.D. Kudlick, RFC 608: Host Names On-Line (Jan. 10, 1974).	

Examiner Signature		Date Considered	
--------------------	--	-----------------	--

\*Examiner: Initial if reference was considered, whether or not citation is in conformance with MPEP 609. Draw a line through citation if not in conformance and not considered. Include a copy of this form with next communication to applicant. Notes: If identified, the following is provided: EA = English Abstract, T = Translation, PF = Patent Family.

<b>INFORMATION DISCLOSURE STATEMENT BY APPLICANT</b> FORM PTO-1449 (modified)  Sheet 39 of 67	Reexam number	90/010,416
	First Named Inventor	Hutton
	Patent Under Re-Exam	6108704
	Issue Date	2000/08/22
	Group Art Unit	3992
	Examiner Name	KOSOWSKI, ALEXANDER J
	Attorney Docket No.	2655-0188
	Confirmation No.	1061

NON-PATENT REFERENCES			
Examiner Initials*	Cite No.	Non-patent Reference bibliographic information, where available	Notes
	39-1	Making the Most of IP Telephony, VocalTec Annual Report 1997	
	39-2	Mark Crispin, RFC 752: A Universal Host Table (Jan. 2, 1979).	
	39-3	Mark Reid, Ptel Call Control Procedure in H.323 (June 16, 1995).	
	39-4	Markus Sohlenkamp & Greg Chwelos, Integrating Communication, Cooperation, and Awareness: The DIVA Virtual Office Environment (1994).	
	39-5	Mic Bowman, et al., Unifers: An Attribute-based Name Server, SOFTWARE PRACTICE AND EXPERIENCE, vol. 20(4) (Apr. 1990).	
	39-6	Michael D. Schroeder, et al., Experience with Grapevine: The Growth of a Distributed System, ACM Transactions on Computer Systems (Feb. 1984).	
	39-7	Michael D. Schroeder, et al., Experience with Grapevine: The Growth of a Distributed System, ACM TRANSACTIONS ON COMPUTER SYSTEMS, vol. 2, No. 1 (Feb. 1984).	

Examiner Signature		Date Considered	
--------------------	--	-----------------	--

\*Examiner: Initial if reference was considered, whether or not citation is in conformance with MPEP 609. Draw a line through citation if not in conformance and not considered. Include a copy of this form with next communication to applicant. Notes: If identified, the following is provided: EA = English Abstract, T = Translation, PF = Patent Family.



<b>INFORMATION DISCLOSURE STATEMENT BY APPLICANT</b> FORM PTO-1449 (modified)	Reexam number	90/010,416
	First Named Inventor	Hutton
	Patent Under Re-Exam	6108704
	Issue Date	2000/08/22
	Group Art Unit	3992
	Examiner Name	KOSOWSKI, ALEXANDER J
	Attorney Docket No.	2655-0188
Sheet 40 of 67	Confirmation No.	1061

NON-PATENT REFERENCES			
Examiner Initials*	Cite No.	Non-patent Reference bibliographic information, where available	Notes
	40-1	Michael F. Schwartz, et al., A Comparison of Internet Resource Discovery Approaches, COMPUTING SYSTEMS (Aug. 1992).	
	40-2	Michael F. Schwartz, et al., A Name Service for Evolving, Heterogeneous Systems, ACM (1987).	
	40-3	Michael J. Bibeau, A Formative Evaluation of CU-SeeMe, Masters Thesis, Virginia Polytechnic Institute and State University (Feb. 20, 1995) (including CU-SeeMe Users Manual by same author published Jan. 1995).	
	40-4	Michelle Slatalla, Hold the Phone! You Can Call Long Distance on a Computer For Pennies, But it has its Drawbacks, NEWSDAY (Mar. 14, 1995).	
	40-5	Mike Kong, et al., Network Computing System Reference Manual, Prentice Hall (1990).	
	40-6	Mike Kudlick, et al., RFC 627: ASCII Text File of Hostnames (Mar. 25, 1974).	
	40-7	Mitch Wagner, Phone Home Cheaply Over the I-Way, OPEN SYSTEMS TODAY (Feb. 20, 1995).	

Examiner Signature		Date Considered	
--------------------	--	-----------------	--

\*Examiner: Initial if reference was considered, whether or not citation is in conformance with MPEP 609. Draw a line through citation if not in conformance and not considered. Include a copy of this form with next communication to applicant. Notes: If identified, the following is provided: EA = English Abstract, T = Translation, PF = Patent Family.

N2P-IDS00266

<b>INFORMATION DISCLOSURE STATEMENT BY APPLICANT</b> FORM PTO-1449 (modified)	Reexam number	90/010,416
	First Named Inventor	Hutton
	Patent Under Re-Exam	6108704
	Issue Date	2000/08/22
	Group Art Unit	3992
	Examiner Name	KOSOWSKI, ALEXANDER J
	Attorney Docket No.	2655-0188
Sheet 41 of 67	Confirmation No.	1061

NON-PATENT REFERENCES			
Examiner Initials*	Cite No.	Non-patent Reference bibliographic information, where available	Notes
	41-1	MITRE Fort Meade Site DCE Meeting Sign In Sheet 1/16/1995	
	41-2	Mostafa H. Ammar, et al., Using Hint Tables to Locate Resources in Distributed Systems (IEEE, 1988).	
	41-3	Motorola Micro TAC International 5000 Series Manual (undated)	
	41-4	Motorola Micro TAC International 7000 Series (dated 5/94)	
	41-5	Motorola Micro TAC International 7500 Series (undated)	
	41-6	Motorola Micro TAC International 8000 Series (undated)	
	41-7	Nate Zelnick, Chat on the Web: An Overview, INTERACTIVE CONTENT, vol. 2, no. 17 (Sept. 1995).	

Examiner Signature		Date Considered	
--------------------	--	-----------------	--

\*Examiner: Initial if reference was considered, whether or not citation is in conformance with MPEP 609. Draw a line through citation if not in conformance and not considered. Include a copy of this form with next communication to applicant. Notes: If identified, the following is provided: EA = English Abstract, T = Translation, PF = Patent Family.

N2P-IDS00267

<b>INFORMATION DISCLOSURE STATEMENT BY APPLICANT</b> FORM PTO-1449 (modified)  Sheet 42 of 67	Reexam number	90/010,416
	First Named Inventor	Hutton
	Patent Under Re-Exam	6108704
	Issue Date	2000/08/22
	Group Art Unit	3992
	Examiner Name	KOSOWSKI, ALEXANDER J
	Attorney Docket No.	2655-0188
	Confirmation No.	1061

NON-PATENT REFERENCES			
Examiner Initials*	Cite No.	Non-patent Reference bibliographic information, where available	Notes
	42-1	Nautilus: Secure Computer Telephony, from Q1.11 of Section 1 of the comp.speech FAQ Home Page (dated Aug. 7, 1996)	
	42-2	NetPhone 1.0 User Manual, Electric Magic Company (document includes date 94-12-31)	
	42-3	NetPhone 1.1 User Manual, Electric Magic Company (document includes date 1995-02-16)	
	42-4	NetPhone Demo Instructions, Electric Magic Company, 1994.	
	42-5	NetPhone Development Plan (undated)	
	42-6	NetPhone Development Plan v0.1 (undated)	
	42-7	NetPhone Digital User Manual, Electric Magic Company, 95-02-26.	

Examiner Signature		Date Considered	
--------------------	--	-----------------	--

\*Examiner: Initial if reference was considered, whether or not citation is in conformance with MPEP 609. Draw a line through citation if not in conformance and not considered. Include a copy of this form with next communication to applicant. Notes: If identified, the following is provided: EA = English Abstract, T = Translation, PF = Patent Family.

<b>INFORMATION DISCLOSURE STATEMENT BY APPLICANT</b> FORM PTO-1449 (modified)  Sheet 43 of 67	Reexam number	90/010,416
	First Named Inventor	Hutton
	Patent Under Re-Exam	6108704
	Issue Date	2000/08/22
	Group Art Unit	3992
	Examiner Name	KOSOWSKI, ALEXANDER J
	Attorney Docket No.	2655-0188
	Confirmation No.	1061

NON-PATENT REFERENCES			
Examiner Initials*	Cite No.	Non-patent Reference bibliographic information, where available	Notes
	43-1	NetPhone Digital User Manual, Electric Magic Company, 95-03-12.	
	43-2	NetPhone Information Manual, Electric Magic Company, May 30, 1995.	
	43-3	NetPhone Testing Notes, September 28, 1994.	
	43-4	Netphone, Change Notes, December 6.	
	43-5	Nigel Hinds, et al., Name Space Models for Locating Services, IBM Canada Laboratory Technical Report 74.074 (1991).	
	43-6	Norbert Leser, Towards a Worldwide Distributed File System: The OSF DCE File System as an example (Sept. 27, 1990).	
	43-7	Open Group, Cambridge Information (June 23, 1997).	

Examiner Signature		Date Considered	
--------------------	--	-----------------	--

\*Examiner: Initial if reference was considered, whether or not citation is in conformance with MPEP 609. Draw a line through citation if not in conformance and not considered. Include a copy of this form with next communication to applicant. Notes: If identified, the following is provided: EA = English Abstract, T = Translation, PF = Patent Family.

N2P-IDS00269

<b>INFORMATION DISCLOSURE STATEMENT BY APPLICANT</b> FORM PTO-1449 (modified)	Reexam number	90/010,416
	First Named Inventor	Hutton
	Patent Under Re-Exam	6108704
	Issue Date	2000/08/22
	Group Art Unit	3992
	Examiner Name	KOSOWSKI, ALEXANDER J
	Attorney Docket No.	2655-0188
Sheet 44 of 67	Confirmation No.	1061

NON-PATENT REFERENCES			
Examiner Initials*	Cite No.	Non-patent Reference bibliographic information, where available	Notes
	44-1	Open Software Foundation Security Sig (March 19, 1996).	
	44-2	Open Software Foundation, AES/Distributed Computing RPC Volume, PTR Prentice Hall (1994).	
	44-3	Open Software Foundation, DCE Internals Course, Instructor Guide Vol. 1 (1992).	
	44-4	Open Software Foundation, DCE Internals Course, Instructor Guide Vol. 2 (1992).	
	44-5	Open Software Foundation, Industry Analysis of DCE (May 15, 1990).	
	44-6	Open Software Foundation, Introduction to OSF DCE , Prentice Hall (1992).	
	44-7	Open Software Foundation, Open Line Magazine (May/June 1990).	

Examiner Signature		Date Considered	
--------------------	--	-----------------	--

\*Examiner: Initial if reference was considered, whether or not citation is in conformance with MPEP 609. Draw a line through citation if not in conformance and not considered. Include a copy of this form with next communication to applicant. Notes: If identified, the following is provided: EA = English Abstract, T = Translation, PF = Patent Family.

**INFORMATION DISCLOSURE  
STATEMENT BY APPLICANT**  
FORM PTO-1449 (modified)

Sheet 45 of 67

Reexam number	90/010,416
First Named Inventor	Hutton
Patent Under Re-Exam	6108704
Issue Date	2000/08/22
Group Art Unit	3992
Examiner Name	KOSOWSKI, ALEXANDER J
Attorney Docket No.	2655-0188
Confirmation No.	1061

**NON-PATENT REFERENCES**

Examiner Initials*	Cite No.	Non-patent Reference bibliographic information, where available	Notes
	45-1	Open Software Foundation, OSF DCE Administration Guide Core Components, PTR Prentice Hall (1993).	
	45-2	Open Software Foundation, OSF DCE Administration Guide—Extended Services, PTR Prentice Hall (1993).	
	45-3	Open Software Foundation, OSF DCE Administration Guide—Introduction, PTR Prentice Hall (1993).	
	45-4	Open Software Foundation, OSF DCE Administration Reference, PTR Prentice Hall (1993).	
	45-5	Open Software Foundation, OSF DCE Application Development Guide, PTR Prentice Hall (1993).	
	45-6	Open Software Foundation, OSF DCE Application Development Reference, PTR Prentice Hall (1993).	
	45-7	Open Software Foundation, OSF DCE User's Guide and Reference, PTR Prentice Hall (1993).	

Examiner Signature		Date Considered	
--------------------	--	-----------------	--

\*Examiner: Initial if reference was considered, whether or not citation is in conformance with MPEP 609. Draw a line through citation if not in conformance and not considered. Include a copy of this form with next communication to applicant. Notes: If identified, the following is provided: EA = English Abstract, T = Translation, PF = Patent Family.

N2P-IDS00271

<b>INFORMATION DISCLOSURE STATEMENT BY APPLICANT</b> FORM PTO-1449 (modified)  Sheet 46 of 67	Reexam number	90/010,416
	First Named Inventor	Hutton
	Patent Under Re-Exam	6108704
	Issue Date	2000/08/22
	Group Art Unit	3992
	Examiner Name	KOSOŃSKI, ALEXANDER J
	Attorney Docket No.	2655-0188
Confirmation No.	1061	

**NON-PATENT REFERENCES**

Examiner Initials*	Cite No.	Non-patent Reference bibliographic information, where available	Notes
	46-1	OSF DCE USER'S GUIDE AND REFERENCE (Prentice-Hall, Inc., 1993).	
	46-2	P. Deutsch, et al., RFC 1835: Architecture of the Whois++ Service (Aug. 1995).	
	46-3	P. Faltstrom, et al., RFC 1914: How to Interact with a Whois++ Mesh (Feb. 1996).	
	46-4	P. Mockapetris, RFC 882: Domain Names -- Concepts and Facilities (Nov. 1983).	
	46-5	P. Mockapetris, RFC 883: Domain Names -- Implementation and Specification (Nov. 1983).	
	46-6	P. Venkat Rangan, et al., Software Architecture for Integration of Video Services in the Etherphone System, IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS, vol. 9, No. 9 (Dec. 1991).	
	46-7	P.M. Gopal, et al., Consistent Resource Registration, IBM TECHNICAL DISCLOSURE BULLETIN, vol. 37, no. 9 (Sept. 1994).	

Examiner Signature		Date Considered	
--------------------	--	-----------------	--

\*Examiner: Initial if reference was considered, whether or not citation is in conformance with MPEP 609. Draw a line through citation if not in conformance and not considered. Include a copy of this form with next communication to applicant. Notes: If identified, the following is provided: EA = English Abstract, T = Translation, PF = Patent Family.

<b>INFORMATION DISCLOSURE STATEMENT BY APPLICANT</b> FORM PTO-1449 (modified)	Reexam number	90/010,416
	First Named Inventor	Hutton
	Patent Under Re-Exam	6108704
	Issue Date	2000/08/22
	Group Art Unit	3992
	Examiner Name	KOSOWSKI, ALEXANDER J
	Attorney Docket No.	2655-0188
Sheet 47 of 67	Confirmation No.	1061

NON-PATENT REFERENCES			
Examiner Initials*	Cite No.	Non-patent Reference bibliographic information, where available	Notes
	47-1	Part 1 of Open Software Foundation, X/Open Preliminary Specification—X/Open DCE: Authentication and Security Services (March 1996).	
	47-2	Part 2 chapter 2 thru 5 of Open Software Foundation, X/Open Preliminary Specification—X/Open DCE: Authentication and Security Services (March 1996).	
	47-3	Part 2 chapter 6 thru 13 of Open Software Foundation, X/Open Preliminary Specification—X/Open DCE: Authentication and Security Services (March 1996).	
	47-4	Part 3 and Part 4 of Open Software Foundation, X/Open Preliminary Specification—X/Open DCE: Authentication and Security Services (March 1996).	
	47-5	Pato, Joseph N., A Generic Interface for Extended Registry Attributes, July 7, 1992.	
	47-6	Paul Albitz, et al., DNS AND BIND IN A NUTSHELL (O'Reilly & Associates, 1992).	
	47-7	Paul Mockapetris, RFC 1034: Domain Name – Concepts and Facilities (Nov. 1987).	

Examiner Signature		Date Considered	
--------------------	--	-----------------	--

\*Examiner: Initial if reference was considered, whether or not citation is in conformance with MPEP 609. Draw a line through citation if not in conformance and not considered. Include a copy of this form with next communication to applicant. Notes: If identified, the following is provided: EA = English Abstract, T = Translation, PF = Patent Family.

N2P-IDS00273



**INFORMATION DISCLOSURE  
STATEMENT BY APPLICANT**  
FORM PTO-1449 (modified)

Sheet 48 of 67

Reexam number	90/010,416
First Named Inventor	Hutton
Patent Under Re-Exam	6108704
Issue Date	2000/08/22
Group Art Unit	3992
Examiner Name	KOSOWSKI, ALEXANDER J
Attorney Docket No.	2655-0188
Confirmation No.	1061

**NON-PATENT REFERENCES**

Examiner Initials*	Cite No.	Non-patent Reference bibliographic information, where available	Notes
	48-1	Paul Mockapetris, RFC 1035: Domain Name – Implementation and Specification (Nov. 1987).	
	48-2	Paul V. Mockapetris, et al., Development of the Domain Name Server, COMPUTER COMMUNICATION REVIEW, vol. 18, no. 4 (Aug. 1988).	
	48-3	Paul V. Mockapetris, et al., Development of the Domain Name System, COMPUTER COMMUNICATION REVIEW (Aug. 1988).	
	48-4	Phoning By Web, SAN FRANCISCO CHRONICLE (Mar. 12, 1996).	
	48-5	PictureTel Corp., 10-K405/A (filed Jan. 13, 1998).	
	48-6	PictureTel LiveLan (printed 12/3/2007)	
	48-7	Ping Lin's Email to mackey, Comments on DCE 1.1 Delegation RFC, July 2, 1992.	

Examiner Signature		Date Considered	
--------------------	--	-----------------	--

\*Examiner: Initial if reference was considered, whether or not citation is in conformance with MPEP 609. Draw a line through citation if not in conformance and not considered. Include a copy of this form with next communication to applicant. Notes: If identified, the following is provided: EA = English Abstract, T = Translation, PF = Patent Family.

N2P-IDS00274

<b>INFORMATION DISCLOSURE STATEMENT BY APPLICANT</b> FORM PTO-1449 (modified)  Sheet 49 of 67	Reexam number	90/010,416
	First Named Inventor	Hutton
	Patent Under Re-Exam	6108704
	Issue Date	2000/08/22
	Group Art Unit	3992
	Examiner Name	KOSOWSKI, ALEXANDER J
	Attorney Docket No.	2655-0188
	Confirmation No.	1061

NON-PATENT REFERENCES			
Examiner Initials*	Cite No.	Non-patent Reference bibliographic information, where available	Notes
	49-1	Polle T. Zellweger, et al., An Overview of the Etherphone System and its Applications (IEEE, 1988).	
	49-2	Postel, RFC 791: Internet Protocol: Darpa Internet Program Protocol Specification (Sept. 1981).	
	49-3	Postel, RFC 793: Transmission Control Protocol: Darpa Internet Program Protocol Specification (Sept. 1981).	
	49-4	PowWow For Microsoft Windows User's Guide, Version 1.4B, Documentation by Token White Man (dated 1995)	
	49-5	PowWow For Microsoft Windows User's Guide, Version 1.5, Documentation by Aryeh Goretsky (dated 1995)	
	49-6	PowWow For Microsoft Windows User's Guide, Version 1.6 beta 2, Documentation by Aryeh Goretsky (dated 1995)	
	49-7	PowWow For Microsoft Windows User's Guide, Version 1.6 beta, Documentation by Aryeh Goretsky (dated 1995)	

Examiner Signature		Date Considered	
--------------------	--	-----------------	--

\*Examiner: Initial if reference was considered, whether or not citation is in conformance with MPEP 609. Draw a line through citation if not in conformance and not considered. Include a copy of this form with next communication to applicant. Notes: If identified, the following is provided: EA = English Abstract, T = Translation, PF = Patent Family.

<b>INFORMATION DISCLOSURE STATEMENT BY APPLICANT</b> FORM PTO-1449 (modified)	Reexam number	90/010,416
	First Named Inventor	Hutton
	Patent Under Re-Exam	6108704
	Issue Date	2000/08/22
	Group Art Unit	3992
	Examiner Name	KOSOWSKI, ALEXANDER J
	Attorney Docket No.	2655-0188
Sheet 50 of 67	Confirmation No.	1061

NON-PATENT REFERENCES			
Examiner Initials*	Cite No.	Non-patent Reference bibliographic information, where available	Notes
	50-1	PowWow For Microsoft Windows User's Guide, Version 1.7 beta 1, Documentation by Aryeh Goretsky (dated 1995)	
	50-2	PowWow For Microsoft Windows User's Guide, Version 1.7 beta 2, Documentation by Aryeh Goretsky (dated 1995)	
	50-3	PowWow For Microsoft Windows User's Guide, Version 1.7 beta 3, Documentation by Aryeh Goretsky (dated 1995)	
	50-4	PowWow For Microsoft Windows User's Guide, Version 1.7 beta 4, Documentation by Aryeh Goretsky (dated 1995)	
	50-5	PowWow For Microsoft Windows User's Guide, Version 2.0 beta 1, Documentation by Aryeh Goretsky (dated 1995, 1996)	
	50-6	PowWow For Microsoft Windows User's Guide, Version 2.1, Documentation by Aryeh Goretsky (dated 1995, 1996)	
	50-7	PowWow For Microsoft Windows User's Guide, Version 2.2 beta 1, Documentation by Aryeh Goretsky (dated 1995, 1996)	

Examiner Signature		Date Considered	
--------------------	--	-----------------	--

\*Examiner: Initial if reference was considered, whether or not citation is in conformance with MPEP 609. Draw a line through citation if not in conformance and not considered. Include a copy of this form with next communication to applicant. Notes: If identified, the following is provided: EA = English Abstract, T = Translation, PF = Patent Family.

N2P-IDS00276

<b>INFORMATION DISCLOSURE STATEMENT BY APPLICANT</b> FORM PTO-1449 (modified)  Sheet 51 of 67	Reexam number	90/010,416
	First Named Inventor	Hutton
	Patent Under Re-Exam	6108704
	Issue Date	2000/08/22
	Group Art Unit	3992
	Examiner Name	KOSOWSKI, ALEXANDER J
	Attorney Docket No.	2655-0188
	Confirmation No.	1061

NON-PATENT REFERENCES			
Examiner Initials*	Cite No.	Non-patent Reference bibliographic information, where available	Notes
	51-1	PowWow For Microsoft Windows User's Guide, Version 2.2 beta 2, Documentation by Aryeh Goretsky (dated 1995, 1996)	
	51-2	PowWow For Microsoft Windows User's Guide, Version 2.3, Documentation by Aryeh Goretsky (dated 1995, 1996)	
	51-3	PowWow For Microsoft Windows User's Guide, Version 2.31, Documentation by Aryeh Goretsky (dated 1995, 1996)	
	51-4	PowWow For Microsoft Windows User's Guide, Version 2.32, Documentation by Aryeh Goretsky (dated 1995, 1996)	
	51-5	PowWow For Microsoft Windows User's Guide, Version 3.0 beta 3, Documentation by Aryeh Goretsky (dated 1995, 1996)	
	51-6	PowWow User Local Server Version 1.0 beta 2 Release Notes (Dated June 18, 1996)	
	51-7	PowWow User Location Server for Microsoft Windows NT and 95 Version 1.0 beta 2 Installation Guide, by Goretsky, Aryeh (dated 1996)	

Examiner Signature		Date Considered	
--------------------	--	-----------------	--

\*Examiner: Initial if reference was considered, whether or not citation is in conformance with MPEP 609. Draw a line through citation if not in conformance and not considered. Include a copy of this form with next communication to applicant. Notes: If identified, the following is provided: EA = English Abstract, T = Translation, PF = Patent Family.

N2P-IDS00277

**INFORMATION DISCLOSURE  
STATEMENT BY APPLICANT**  
FORM PTO-1449 (modified)

Sheet 52 of 67

Reexam number	90/010,416
First Named Inventor	Hutton
Patent Under Re-Exam	6108704
Issue Date	2000/08/22
Group Art Unit	3992
Examiner Name	KOSOWSKI, ALEXANDER J
Attorney Docket No.	2655-0188
Confirmation No.	1061

**NON-PATENT REFERENCES**

Examiner Initials*	Cite No.	Non-patent Reference bibliographic information, where available	Notes
	52-1	PowWow Version Release Notes (covering versions 1.4b to 2.32) (dated June 26, 1996)	
	52-2	PowWow32 Release Notes (PowWow Versions 3.0 beta 3 and 3.0 beta 2) (dated November 21, 1996)	
	52-3	Prospectus for VocalTech Ordinary Shares, February 6, 1996	
	52-4	Questions and Comments: DCE RFC 6.0 "A Generic Interface for Extended Registry Attributes" Commentary by Bob Blakley, July 6, 1992	
	52-5	R. Braden, RFC 1644 T/TCP – TCP Extensions for Transactions Functional Specifications (July 1994).	
	52-6	R. Droms, RFC 1531: Dynamic Host Configuration Protocol (Oct. 1993).	
	52-7	R. Droms, RFC 1541: Dynamic Host Configuration Protocol (Oct. 1993).	

Examiner Signature		Date Considered	
--------------------	--	-----------------	--

\*Examiner: Initial if reference was considered, whether or not citation is in conformance with MPEP 609. Draw a line through citation if not in conformance and not considered. Include a copy of this form with next communication to applicant. Notes: If identified, the following is provided: EA = English Abstract, T = Translation, PF = Patent Family.

N2P-IDS00278

<b>INFORMATION DISCLOSURE STATEMENT BY APPLICANT</b> FORM PTO-1449 (modified)  Sheet 53 of 67	Reexam number	90/010,416
	First Named Inventor	Hutton
	Patent Under Re-Exam	6108704
	Issue Date	2000/08/22
	Group Art Unit	3992
	Examiner Name	KOSOWSKI, ALEXANDER J
	Attorney Docket No.	2655-0188
Confirmation No.	1061	

NON-PATENT REFERENCES			
Examiner Initials*	Cite No.	Non-patent Reference bibliographic information, where available	Notes
	53-1	R.C. Summers, Local-Area Distributed Systems, IBM SYSTEMS JOURNAL, vol. 28, No. 2 (1989).	
	53-2	Raj Pandya, Emerging Mobile and Personal Communication Systems, IEEE COMMUNICATIONS MAGAZINE (June 1995).	
	53-3	RFC 1001: Protocol Standard for a NetBIOS Service on a TCP/UDP Transport: Concepts and Methods, Mar. 1987.	
	53-4	RFC 1057: RPC Remote Procedure Call Protocol Specification Version 2, June 1988	
	53-5	Richard Karpinski, Internet Phones Battle for the Market, INTERACTIVE AGE, no. 212 (1995).	
	53-6	Richard Karpinski, Upgrading Internet Phone – VocalTec Offers Full-Duplex Version, Eliminating Voice Delays, INTERACTIVE AGE, no. 216 (1995).	
	53-7	Richard T. Snodgrass, Developing Time-Oriented Database Applications in SQL, MORGAN KAUFMANN PUBLISHERS (2000).	

Examiner Signature		Date Considered	
--------------------	--	-----------------	--

\*Examiner: Initial if reference was considered, whether or not citation is in conformance with MPEP 609. Draw a line through citation if not in conformance and not considered. Include a copy of this form with next communication to applicant. Notes: If identified, the following is provided: EA = English Abstract, T = Translation, PF = Patent Family.

N2P-IDS00279

<b>INFORMATION DISCLOSURE STATEMENT BY APPLICANT</b> FORM PTO-1449 (modified)	Reexam number	90/010,416
	First Named Inventor	Hutton
	Patent Under Re-Exam	6108704
	Issue Date	2000/08/22
	Group Art Unit	3992
	Examiner Name	KOSOWSKI, ALEXANDER J
	Attorney Docket No.	2655-0188
Sheet 54 of 67	Confirmation No.	1061

NON-PATENT REFERENCES			
Examiner Initials*	Cite No.	Non-patent Reference bibliographic information, where available	Notes
	54-1	Rivka Tadjer, Internet Communications Solutions: How Well Do They Work?, COMPUTER SHOPPER, vol. 15, no. 6 (June 1995).	
	54-2	Rivka Tadjer, Internet Phones to Upstage Videoconferencing Products? Talk is Cheaper with Local Worldwide Dialing, COMPUTER SHOPPER, vol. 15, no. 5 (May 1995).	
	54-3	Rob Walters, COMPUTER TELEPHONE INTEGRATION (Artech House, 1993).	
	54-4	Robert E. Kahn, et al., Advances in Packet Radio Technology, Proceedings Of The IEEE (Nov. 1978).	
	54-5	Robert Gurwitz, et al., IEN 212: IP — Local Area Network Addressing Issues (Sept. 1982).	
	54-6	Robert J. Williams, User Location Service (Feb. 1996).	
	54-7	Robert Joseph Fowler, Decentralized Object Finding Using Forwarding Addresses, Ph.D. Thesis, University of Washington (Dec. 1985).	

Examiner Signature		Date Considered	
--------------------	--	-----------------	--

\*Examiner: Initial if reference was considered, whether or not citation is in conformance with MPEP 609. Draw a line through citation if not in conformance and not considered. Include a copy of this form with next communication to applicant. Notes: If identified, the following is provided: EA = English Abstract, T = Translation, PF = Patent Family.

N2P-IDS00280

<b>INFORMATION DISCLOSURE STATEMENT BY APPLICANT</b> FORM PTO-1449 (modified)  Sheet 55 of 67	Reexam number	90/010,416
	First Named Inventor	Hutton
	Patent Under Re-Exam	6108704
	Issue Date	2000/08/22
	Group Art Unit	3992
	Examiner Name	KOSOWSKI, ALEXANDER J
	Attorney Docket No.	2655-0188
	Confirmation No.	1061

NON-PATENT REFERENCES			
Examiner Initials*	Cite No.	Non-patent Reference bibliographic information, where available	Notes
	55-1	Robert Richardson, Internet Phone, LAN MAGAZINE, vol. 10, no. 7 (July 1995).	
	55-2	Robert Richardson, Pow Wow, Anyone? A Web Chat That Works, LAN MAGAZINE, vol. 10 no. 9 (Sept. 1995).	
	55-3	Robert S. French, et al., The Zephyr Programmer's Manual, Rev. 2.1 (May 5, 1989).	
	55-4	Rosen, Nick "Internet Opens Line on Cheap Global Phone Calls" The Guardian, February 10, 1995, A1.	
	55-5	S. Waldbusser, et al., RFC 1742: AppleTalk Management Information Base II (Jan. 1995).	
	55-6	S.R. Ahuja, et al., The Rapport Multimedia Conferencing System, ACM (1988).	
	55-7	Sakae Okubo, et al., Draft ITU-T Recommendation H.245—Line Transmission of Non-Telephone Signals: Control Protocol for Multimedia Communication (Nov. 14, 1995).	

Examiner Signature		Date Considered	
--------------------	--	-----------------	--

\*Examiner: Initial if reference was considered, whether or not citation is in conformance with MPEP 609. Draw a line through citation if not in conformance and not considered. Include a copy of this form with next communication to applicant. Notes: If identified, the following is provided: EA = English Abstract, T = Translation, PF = Patent Family.

N2P-IDS00281



**INFORMATION DISCLOSURE  
STATEMENT BY APPLICANT**  
FORM PTO-1449 (modified)

Sheet 56 of 67

Reexam number	90/010,416
First Named Inventor	Hutton
Patent Under Re-Exam	6108704
Issue Date	2000/08/22
Group Art Unit	3992
Examiner Name	KOSOWSKI, ALEXANDER J
Attorney Docket No.	2655-0188
Confirmation No.	1061

**NON-PATENT REFERENCES**

Examiner Initials*	Cite No.	Non-patent Reference bibliographic information, where available	Notes
	56-1	Sakae Okubo, et al., ITU-T Recommendation H.245—Line Transmission of Non-Telephone Signals: Control Protocol for Multimedia Communication (May 20, 1996).	
	56-2	Sakae Okubo, et al., Line Transmission of Non-Telephone Signals: Control Protocol for Multimedia Communication, Recommendation H245 (May 20, 1996).	
	56-3	Sakae Okubo, et al., ITU-T Standardization of Audiovisual Communication Systems in ATM and LAN Environments (April 17, 1996).	
	56-4	Sape J. Mullender, et al., Distributed Match-Making for Processes in Computer Networks (Association for Computing Machinery, 1985).	
	56-5	Sape Mullender, ed., Distributed Systems, ACM Press (1992).	
	56-6	Sapwater, E. "Webbed", 2 pages (undated)	
	56-7	Saruchi Mohan, Internet Phone Accepting Calls, COMPUTERWORLD (Feb. 27, 1995).	

Examiner Signature		Date Considered	
--------------------	--	-----------------	--

\*Examiner: Initial if reference was considered, whether or not citation is in conformance with MPEP 609. Draw a line through citation if not in conformance and not considered. Include a copy of this form with next communication to applicant. Notes: If identified, the following is provided: EA = English Abstract, T = Translation, PF = Patent Family.

N2P-IDS00282

**INFORMATION DISCLOSURE  
STATEMENT BY APPLICANT**  
FORM PTO-1449 (modified)

Sheet 57 of 67

Reexam number	90/010,416
First Named Inventor	Hutton
Patent Under Re-Exam	6108704
Issue Date	2000/08/22
Group Art Unit	3992
Examiner Name	KOSOWSKI, ALEXANDER J
Attorney Docket No.	2655-0188
Confirmation No.	1061

**NON-PATENT REFERENCES**

Examiner Initials*	Cite No.	Non-patent Reference bibliographic information, where available	Notes
	57-1	Savetz, Kevin "Net as Phone" Internet World, July 1995.	
	57-2	Schill, et al., ed., IFIP/IEEE International Conference on Distributed Platforms—Client/Server and Beyond: DCE, CORBA, ODP & Advanced Distribution Applications, Technical University Bergakademie Freiburg (1996).	
	57-3	Schulzrinne, Service Conference Invitation Protocol, INTERNET DRAFT (Feb. 22, 1996).	
	57-4	Scott Kahn, Leave Your Message on My PC After the Beep, PC WEEK (Oct. 3, 1994).	
	57-5	Sharon Fisher, Fruits of Athena - Academic Projects Like Athen Have Given the World Its First Inkling of What Computer Interoperability is All About, COMMUNICATIONS WEEK (1992).	
	57-6	Snell, Jason "Foiling Ma Bell" MacUser, July, 1995.	
	57-7	Speak Freely, from Q1.11 of Section 1 of the comp.speech FAQ Home Page (dated March 19, 1996)	

Examiner Signature		Date Considered	
--------------------	--	-----------------	--

\*Examiner: Initial if reference was considered, whether or not citation is in conformance with MPEP 609. Draw a line through citation if not in conformance and not considered. Include a copy of this form with next communication to applicant. Notes: If identified, the following is provided: EA = English Abstract, T = Translation, PF = Patent Family.

N2P-IDS00283

<b>INFORMATION DISCLOSURE STATEMENT BY APPLICANT</b> FORM PTO-1449 (modified)  Sheet 58 of 67	Reexam number	90/010,416
	First Named Inventor	Hutton
	Patent Under Re-Exam	6108704
	Issue Date	2000/08/22
	Group Art Unit	3992
	Examiner Name	KOSOWSKI, ALEXANDER J
	Attorney Docket No.	2655-0188
Confirmation No.	1061	

NON-PATENT REFERENCES			
Examiner Initials*	Cite No.	Non-patent Reference bibliographic information, where available	Notes
	58-1	Staff Phone List (July 1, 1997).	
	58-2	Steinberg, Bob "Will Politics Interfere With The Global Internet?" Mashpee Enterprise, April 28, 1995.	
	58-3	Stephen A. Uhler, PhoneStation, Moving the Telephone onto the Virtual Desktop, 1993 Winter Usenix, San Diego, California (Jan. 25-29, 1993).	
	58-4	Steve Hamm, The Merry Pranksters, PC WEEK, vol. 12 no. 34 (Aug. 28, 1995).	
	58-5	Stuart Harris, THE IRC SURVIVAL GUIDE: TALK TO THE WORLD WITH INTERNET RELAY CHAT (Addison-Wesley, Feb. 1995).	
	58-6	Sun Microsystems, Inc., RFC 1050: RPC: Remote Procedure Call Protocol Specification Version 2 (June 1988).	
	58-7	Surfers Can Drop Phones, ELECTRONICS TIMES (Feb. 16, 1995).	

Examiner Signature		Date Considered	
--------------------	--	-----------------	--

\*Examiner: Initial if reference was considered, whether or not citation is in conformance with MPEP 609. Draw a line through citation if not in conformance and not considered. Include a copy of this form with next communication to applicant. Notes: If identified, the following is provided: EA = English Abstract, T = Translation, PF = Patent Family.

<b>INFORMATION DISCLOSURE STATEMENT BY APPLICANT</b> FORM PTO-1449 (modified)	Reexam number	90/010,416
	First Named Inventor	Hutton
	Patent Under Re-Exam	6108704
	Issue Date	2000/08/22
	Group Art Unit	3992
	Examiner Name	KOSOWSKI, ALEXANDER J
	Attorney Docket No.	2655-0188
Sheet 59 of 67	Confirmation No.	1061

NON-PATENT REFERENCES			
Examiner Initials*	Cite No.	Non-patent Reference bibliographic information, where available	Notes
	59-1	Susan Thomson, et al., DNS Dynamic Updates, IETF DNSIND WORKING GROUP (July 1994).	
	59-2	T. Berners-Lee, et al., RFC 1738: Uniform Resource Locators (URL) (Dec. 1994).	
	59-3	Tamila Baron, Hearing Voices on the Net, COMMUNICATIONS WEEK (Feb. 20, 1995).	
	59-4	Tamila Baron, VocalTec, Motorola Team Up for Internet Phone and Modem Bundle, COMMUNICATIONS WEEK, no. 549 (1995).	
	59-5	Ted Anderson's Email to dmackey re DCE 1.1 Delegation Proposal for Review, June 23, 1992.	
	59-6	Ted Anderson's Email to pato, Re: RFC 7.0 (really glp92), July 21, 1992.	
	59-7	The 4.BSD-Lite distribution announcement, COMPUTER SYSTEMS RESEARCH GROUP (Mar. 1, 1994), and related newsgroup postings, dated Apr. 21-22, 1994.	

Examiner Signature		Date Considered	
--------------------	--	-----------------	--

\*Examiner: Initial if reference was considered, whether or not citation is in conformance with MPEP 609. Draw a line through citation if not in conformance and not considered. Include a copy of this form with next communication to applicant. Notes: If identified, the following is provided: EA = English Abstract, T = Translation, PF = Patent Family.

<b>INFORMATION DISCLOSURE STATEMENT BY APPLICANT</b> FORM PTO-1449 (modified)	Reexam number	90/010,416
	First Named Inventor	Hutton
	Patent Under Re-Exam	6108704
	Issue Date	2000/08/22
	Group Art Unit	3992
	Examiner Name	KOSOWSKI, ALEXANDER J
	Attorney Docket No.	2655-0188
	Confirmation No.	1061
Sheet 60 of 67		

NON-PATENT REFERENCES			
Examiner Initials*	Cite No.	Non-patent Reference bibliographic information, where available	Notes
	60-1	The Electric Magic Company, Business Plan, Version 0.1 draft, April 17, 1995.	
	60-2	The Open Group Organization Chart (Oct. 1997).	
	60-3	The Open Group Organization Chart October 1996 (Confidential)	
	60-4	The OSF Distributed Computing Environment: Building on International Standards, OSF White Paper (Apr. 1992).	
	60-5	The VocalChat User's Guide, September 28, 1993.	
	60-6	Thomas Maresca, The Internet Phone Company?, CONSUMER INFORMATION APPLIANCE, no. 55 (Feb. 1995).	
	60-7	TIMOP: DCE Time Operations Sample Application. (undated)	

Examiner Signature		Date Considered	
--------------------	--	-----------------	--

\*Examiner: Initial if reference was considered, whether or not citation is in conformance with MPEP 609. Draw a line through citation if not in conformance and not considered. Include a copy of this form with next communication to applicant. Notes: If identified, the following is provided: EA = English Abstract, T = Translation, PF = Patent Family.

<b>INFORMATION DISCLOSURE STATEMENT BY APPLICANT</b> FORM PTO-1449 (modified)	Reexam number	90/010,416
	First Named Inventor	Hutton
	Patent Under Re-Exam	6108704
	Issue Date	2000/08/22
	Group Art Unit	3992
	Examiner Name	KOSOWSKI, ALEXANDER J
	Attorney Docket No.	2655-0188
Sheet 61 of 67	Confirmation No.	1061

**NON-PATENT REFERENCES**

Examiner Initials*	Cite No.	Non-patent Reference bibliographic information, where available	Notes
	61-1	Timothy J. O'Malley, Analysis of the Zephyr Communication Paradigm, Bachelor of Science in Computer Science and Engineering, Thesis, Massachusetts Institute of Technology (May 1993).	
	61-2	Todd Copilevitz, Heard on the Internet, THE STAR-LEDGER (March 7, 1995).	
	61-3	Tom Lyons, Network Computing System Tutorial, Prentice Hall (1991).	
	61-4	Tony Pompili, VocalTec: The Internet Phone Number?, PC MAGAZINE (May 16, 1995).	
	61-5	Translation of Japanese Patent Application No. Sho 63[1988]-131637 (Original dated June 3, 1988)	
	61-6	Transparencies: Walter Tuvell, DCE 1.0 Security Technology--Detailed Architectural Overview (Feb. 1997).	
	61-7	V. Jacobson, et al., RFC 1185: TCP Extension for High-Speed Paths (Oct. 1990).	

Examiner Signature		Date Considered	
--------------------	--	-----------------	--

\*Examiner: Initial if reference was considered, whether or not citation is in conformance with MPEP 609. Draw a line through citation if not in conformance and not considered. Include a copy of this form with next communication to applicant. Notes: If identified, the following is provided: EA = English Abstract, T = Translation, PF = Patent Family.

<b>INFORMATION DISCLOSURE STATEMENT BY APPLICANT</b> FORM PTO-1449 (modified)	Reexam number	90/010,416
	First Named Inventor	Hutton
	Patent Under Re-Exam	6108704
	Issue Date	2000/08/22
	Group Art Unit	3992
	Examiner Name	KOSOWSKI, ALEXANDER J
	Attorney Docket No.	2655-0188
Sheet 62 of 67	Confirmation No.	1061

NON-PATENT REFERENCES			
Examiner Initials*	Cite No.	Non-patent Reference bibliographic information, where available	Notes
	62-1	VocalChat 1.01 Network Information (undated)	
	62-2	VocalChat Early Beta Release 1.02B Information (undated)	
	62-3	VocalChat GTI 2.12 Beta Retrival Instructions and Information (undated)	
	62-4	VocalChat Version 1.0, README.TXT, November, 1993.	
	62-5	VocalChat Version 1.01, README.TXT, March, 1994.	
	62-6	VocalChat Version 2.01 and Wan 2.01, README.TXT. May 1994.	
	62-7	VocalTec Annual Report, 1996	

Examiner Signature		Date Considered	
--------------------	--	-----------------	--

\*Examiner: Initial if reference was considered, whether or not citation is in conformance with MPEP 609. Draw a line through citation if not in conformance and not considered. Include a copy of this form with next communication to applicant. Notes: If identified, the following is provided: EA = English Abstract, T = Translation, PF = Patent Family.

<b>INFORMATION DISCLOSURE STATEMENT BY APPLICANT</b> FORM PTO-1449 (modified)	Reexam number	90/010,416
	First Named Inventor	Hutton
	Patent Under Re-Exam	6108704
	Issue Date	2000/08/22
	Group Art Unit	3992
	Examiner Name	KOSOWSKI, ALEXANDER J
	Attorney Docket No.	2655-0188
Sheet 63 of 67	Confirmation No.	1061

NON-PATENT REFERENCES			
Examiner Initials*	Cite No.	Non-patent Reference bibliographic information, where available	Notes
	63-1	VocalTec Cross-Reference Sheet, Pursuant to Item 501 of Reg. S-K (dated January, 1996)	
	63-2	VocalTec Internet Phone Information Sheet, 2 pages. (dated June 1995)	
	63-3	VocalTec Internet Phone Version 3.0 Build 17, README.TXT, August 11, 1995.	
	63-4	VocalTec Internet Phone Version 3.2 Build 21, README.TXT, March 25, 1996.	
	63-5	VocalTec SEC 20-F Filing, 1996	
	63-6	VocalTec SEC F-1 Filing, December 22, 1995	
	63-7	VocalTec SEC F-1 Filing, January 5, 1996	

Examiner Signature		Date Considered	
--------------------	--	-----------------	--

\*Examiner: Initial if reference was considered, whether or not citation is in conformance with MPEP 609. Draw a line through citation if not in conformance and not considered. Include a copy of this form with next communication to applicant. Notes: If identified, the following is provided: EA = English Abstract, T = Translation, PF = Patent Family.



<b>INFORMATION DISCLOSURE STATEMENT BY APPLICANT</b> FORM PTO-1449 (modified)  Sheet 64 of 67	Reexam number	90/010,416
	First Named Inventor	Hutton
	Patent Under Re-Exam	6108704
	Issue Date	2000/08/22
	Group Art Unit	3992
	Examiner Name	KOSOWSKI, ALEXANDER J
	Attorney Docket No.	2655-0188
Confirmation No.	1061	

NON-PATENT REFERENCES			
Examiner Initials*	Cite No.	Non-patent Reference bibliographic information, where available	Notes
	64-1	Voice Over the Internet, BOARDWATCH MAGAZINE, vol. IX, no. 1 (Jan. 1995).	
	64-2	W. David Albrecht, CPA Firms on the World Wide Web, OHIO CPA JOURNAL (June 1996).	
	64-3	W. Simpson, RFC 1661: The Point-to-Point Protocol (PPP) (July 1994).	
	64-4	W. Yeong, et al., RFC 1777. Lightweight Directory Access Protocol (Mar. 1995).	
	64-5	Walt and mactcp's ip addresses and code (undated)	
	64-6	Walter Tuvell, DCE 1.0 Security Technology- Detailed Architectural Overview (Feb. 1997).	
	64-7	Walter Tuvell, DCE 1.0 Security Technology Detailed Architectural Overview, Draft (Feb. 1997).	

Examiner Signature		Date Considered	
--------------------	--	-----------------	--

\*Examiner: Initial if reference was considered, whether or not citation is in conformance with MPEP 609. Draw a line through citation if not in conformance and not considered. Include a copy of this form with next communication to applicant. Notes: If identified, the following is provided: EA = English Abstract, T = Translation, PF = Patent Family.

<b>INFORMATION DISCLOSURE STATEMENT BY APPLICANT</b> FORM PTO-1449 (modified)  Sheet 65 of 67	Reexam number	90/010,416
	First Named Inventor	Hutton
	Patent Under Re-Exam	6108704
	Issue Date	2000/08/22
	Group Art Unit	3992
	Examiner Name	KOSOWSKI, ALEXANDER J
	Attorney Docket No.	2655-0188
	Confirmation No.	1061

NON-PATENT REFERENCES			
Examiner Initials*	Cite No.	Non-patent Reference bibliographic information, where available	Notes
	65-1	Walter Tuvell, DCE 1.0 Security Technology: Detailed Architectural Overview (Feb. 1997).	
	65-2	Walter Tuvell, DCE 1.0 Security Technology: Detailed Architectural Overview (May 1994).	
	65-3	Walter Tuvell, DCE Multi-Crypto Support—Proposal to NSA for Funding and Exportability of Multiple Cryptographic Mechanisms in OSF's Distributed Computing Environment (Sept. 12, 1995).	
	65-4	Walter Tuvell, Distribution & The Infobahn (1996).	
	65-5	Walter Tuvell, Exportability of DCE Multi-Crypto Feature (March 5, 1996).	
	65-6	Walter Tuvell, RFC 98.0: Challenges Concerning Public-Key in DCE (Dec. 1996).	
	65-7	Walter Tuvell, System V/ONC Comparison to AIX/NCS (Oct. 3, 1988).	

Examiner Signature		Date Considered	
--------------------	--	-----------------	--

\*Examiner: Initial if reference was considered, whether or not citation is in conformance with MPEP 609. Draw a line through citation if not in conformance and not considered. Include a copy of this form with next communication to applicant. Notes: If identified, the following is provided: EA = English Abstract, T = Translation, PF = Patent Family.

<b>INFORMATION DISCLOSURE STATEMENT BY APPLICANT</b> FORM PTO-1449 (modified)  Sheet 66 of 67	Reexam number	90/010,416
	First Named Inventor	Hutton
	Patent Under Re-Exam	6108704
	Issue Date	2000/08/22
	Group Art Unit	3992
	Examiner Name	KOSOWSKI, ALEXANDER J
	Attorney Docket No.	2655-0188
	Confirmation No.	1061

NON-PATENT REFERENCES			
Examiner Initials*	Cite No.	Non-patent Reference bibliographic information, where available	Notes
	66-1	Walter Tuvell, The DCE Dance: Application Development in 29 Easy Steps (Sept. 1991).	
	66-2	Walter Tuvell, The OSF Distributed Computing Environment (DE). (undated)	
	66-3	Web Phone, from Q1.11 of Section 1 of the comp.speech FAQ Home Page (dated March 19, 1996)	
	66-4	WebSTAR Technical Reference (formerly MacHTTP), StarNine Technologies, 1995.	
	66-5	Wei Hu, DCE Security Programming, O'Reilly & Associates (July 1995).	
	66-6	Welch, Nathalie "Vendors Ring in New Telephony Options" MacWeek, April 10, 1995, pg 18.	
	66-7	Wendy Woods, Newsbytes Daily Summary, NEWSBYTES (June 10, 1994).	

Examiner Signature		Date Considered	
--------------------	--	-----------------	--

\*Examiner: Initial if reference was considered, whether or not citation is in conformance with MPEP 609. Draw a line through citation if not in conformance and not considered. Include a copy of this form with next communication to applicant. Notes: If identified, the following is provided: EA = English Abstract, T = Translation, PF = Patent Family.

<b>INFORMATION DISCLOSURE STATEMENT BY APPLICANT</b> FORM PTO-1449 (modified)	Reexam number	90/010,416
	First Named Inventor	Hutton
	Patent Under Re-Exam	6108704
	Issue Date	2000/08/22
	Group Art Unit	3992
	Examiner Name	KOSOWSKI, ALEXANDER J
	Attorney Docket No.	2655-0188
Confirmation No.	1061	
Sheet 67 of 67		

NON-PATENT REFERENCES			
Examiner Initials*	Cite No.	Non-patent Reference bibliographic information, where available	Notes
	67-1	William M. Bulkeley, On-line: Hello, world. Audible Chats on the Internet, WALL STREET JOURNAL (Feb. 10, 1995).	
	67-2	Winther, Mark. "The World Wide Web Phones Home: Internet Telephony Market Assessment, 1996-1999", International Data Corporation White Paper (dated 1996)	
	67-3	Xerox System Integration Standard Clearinghouse Protocol (April 1984).	
	67-4	Yakov Rekhter, et al., Dynamic Updates in the Domain Name System (DNS):Architecture and Mechanism, Internet-Draft, DNSIND Working Group (July 15, 1994).	
	67-5		
	67-6		
	67-7		

Examiner Signature		Date Considered	
--------------------	--	-----------------	--

\*Examiner: Initial if reference was considered, whether or not citation is in conformance with MPEP 609. Draw a line through citation if not in conformance and not considered. Include a copy of this form with next communication to applicant. Notes: If identified, the following is provided: EA = English Abstract, T = Translation, PF = Patent Family.



⑪ Publication number : **0 556 012 A2**

⑫ **EUROPEAN PATENT APPLICATION**

⑰ Application number : **93300919.3**

⑸ Int. Cl.<sup>5</sup> : **H04Q 7/00, H04L 12/56**

⑱ Date of filing : **09.02.93**

⑳ Priority : **10.02.92 JP 23506/92**  
**16.09.92 JP 246855/92**  
**10.11.92 JP 299531/92**

④③ Date of publication of application :  
**18.08.93 Bulletin 93/33**

⑧④ Designated Contracting States :  
**DE FR GB**

⑦① Applicant : **MATSUSHITA ELECTRIC**  
**INDUSTRIAL CO., LTD.**  
**1006, Kadoma**  
**Kadoma-shi, Osaka-fu 571 (JP)**

⑦② Inventor : **Wada, Hiromi**  
**15-10, Higashigaoka, Uzumasa**  
**Neyagawa-shi, Osaka 572 (JP)**  
Inventor : **Yozawa, Takashi**  
**5-16-19, Shinke, Aoo**  
**Mino-shi, Osaka 562 (JP)**  
Inventor : **Ohnishi, Tatsuya**  
**281-5, Kawahara, Aza, Sasabe**  
**Kawanishi-shi, Hyogo 666-01 (JP)**

⑦④ Representative : **Cummings, Sean Patrick et al**  
**David Kettle Associates Audrey House Ely**  
**Place**  
**London EC1N 6SN (GB)**

⑤④ **Migration communication control device.**

⑤⑦ Disclosed is a migration communication control device constructed to control a continuous communication between a mobile node and a node unaffected the mobile node's migration. The migration communication control device comprises a first migration control unit, a second migration control unit on the mobile node, and a third migration control unit on the partner node. The first migration control unit comprises a packet transfer unit and an address post unit. The packet transfer unit receives a packet which was destined for an outdated address of the mobile node, generates a conversion packet which holds an updated address instead of the outdated address, and then transmits the conversion packet, while an address post unit transmits an address post message which indicates the updated address to the third migration control unit. The second migration control unit comprises a migration post unit and a packet resumption unit. The migration post unit transmits to the first migration control unit a migration post message which indicates the updated address when the mobile node migrates to another network while a packet resumption unit receives the conversion packet from both the first migration control unit and the third migration control unit and resumes an original packet from the conversion packet. The third migration control unit comprises a packet conversion unit which converts a destination address of a packet into the updated address, then transmits it to the mobile node.

**EP 0 556 012 A2**

## BACKGROUND OF THE INVENTION

### (1) Field of the Invention

The present invention relates to a migration communication control device that controls a communication between a mobile node and a corresponding node to enable them to communicate continuously when the former migrates by managing addresses assigned to the former each time it migrates across networks.

### (2) Description of the Related Art

Recent progress in the field of electronic technology makes it possible to assemble smaller and lighter portable computers. These portable computers referred to as mobile nodes are designed so that they can migrate across networks: they are unplugged from a network and plugged in another and communicate with a stationary node. Thus, each of them is assigned a specific address to prove its identity. The address, in general, includes location information as to which network the mobile nodes are currently plugged in, and for this reason, a new address is assigned each time they migrate.

For example, the address composed of a network address unit for specifying a network in which the mobile node is currently plugged in and a node address unit for proving the mobile node's identity in the network, or the address used in a conventional network architecture such as Internet Protocol (details of which are in Internet Protocol, RFC791, Jon Postel, Sep., 1981), they must be changed every time the mobile nodes migrate.

However, once the mobile node migrates to another network, a communication with the stationary node will be terminated. This is because a packet is transmitted to its old address only to be wasted.

Thus, to enable the mobile node and stationary node to communicate continuously when the former migrates, it is necessary to control the communication by managing the steadily changing address.

To date, two address managing methods have been proposed: one by Sony Computer Science Laboratory Inc. and one by the Department of Computer Science at Columbia University.

Sony Computer Science Laboratory Inc. proposed a method using VIP (Virtual Internet Protocol), details of which are on "VIP : Lower Layer Internet Protocol", Fumio Teraoka, Yasuhiko Yokote, Mario Tokoro, Proceed of Data Processing Convention : Multimedia Communication and Distributed Processing.

In this method, each mobile node is assigned a VIP (Virtual Internet Protocol) address and a PIP (Physical Internet Protocol) address. The former is an unchanged address used in a communication application for packet transmission and reception;

and the latter is an address changed for every migration to specify an update physical location of the mobile node. Data related to both addresses are held in a cache of a gateway. Under these conditions, the stationary node transmits a packet to the mobile node to the VIP address thereof, and the packet is converted into another packet addressed to the PIP address when it passes the gateway, thence transmitted to the mobile node via the gateways placed in a route onwards. These gateways collect data related to a correlation between the VIP and PIP addresses from the header of the packet upon the receipt thereof, thus updates data in the cache, and hence are able to convert other packets addressed to the VIP addresses into the packets addressed to the PIP addresses based on the correlation entered in the cache.

In this method, in short, the use of the address constituting with the VIP and PIP addresses enables the mobile node and the stationary node to communicate continuously when the former migrates.

The Department of Computer Science at Columbia University proposed a method using an Internet Protocol address of which network address unit does not specify the network which the mobile node is currently plugged in but declares itself to be the mobile node, hence a certain value is given as the network address unit to all the mobile nodes. As well, the method uses an MSS (Mobile Support Station) installed at each network to manage the IP addresses and control a packet route to the mobile node. The MSS is designed so that it collects data related to the update physical location of the mobile nodes by referring other MSSs.

Given these conditions, when the stationary node transmits a packet to the mobile node when it migrates, it first transmits the packet to a first MSS installed in its network; thence the first MSS transfers the packet to a second MSS installed in a network which the mobile node is currently plugged in; and finally the second MSS transfers the packet to the mobile node.

In this method, in short, the use of the MSS enables the mobile node and the stationary node to continue the communication when the former migrates.

In the first method, however, all the nodes must be constructed so that they understand both the VIP and PIP addresses, causing them to extend a scale functionally, otherwise making it impossible to apply this method to apparatuses employed in existing networks. In addition, the communication via the gateways reduces communication efficiency compared with direct packet transmission, because the gateways check whether they have received the packet addressed to the VIP address or PIP address each time they receive it, as well as whether or not to collect the data therefrom to update those in the cache.

In the second method, each network must have

the MSS, and the communication via the MSSs makes it impossible to transmit the packet directly, thereby reducing the communication efficiency.

#### SUMMARY OF THE INVENTION

The present invention therefore has an object to provide a migration communication control device that is available to any apparatus employed in existing networks. Also the present invention has another object to provide a migration communication control device that enables the mobile node and stationary node to communicate continuously when the former migrates by transmitting and receiving the packet directly besides transferring the packet as has been done when the mobile node migrates across the networks.

The above objects are fulfilled by a migration communication control device constructed to control a communication between a mobile node and a partner node, the mobile node migrating across networks and obtaining an address assigned on each network while the partner node being a communication partner of the mobile node, comprising a first migration control unit, a second migration control unit, a third migration control unit, the second migration control unit being placed on the mobile node and the third migration control unit being placed on the partner node, wherein the first migration control unit comprises a packet transfer unit for receiving a packet which was destined for an outdated address of the mobile node, the outdated address assigned when the mobile node migrated to a network to which the first migration control unit is attached, generating a conversion packet which holds an updated address instead of the outdated address, and transmitting the conversion packet; and an address post unit for transmitting an address post message which indicates the updated address of the mobile node to the third migration control unit, the third migration control unit transmitting the packet received by the packet transfer unit, and the second migration control unit comprises a migration post unit for transmitting to the first migration control unit a migration post message which indicates the updated address of the mobile node when the mobile node migrates to another network; and a packet resumption unit for receiving the conversion packet from both the first migration control unit and the third migration control unit and resuming an original packet from the conversion packet, and the third migration control unit comprises a packet conversion unit for converting a destination address of a packet, the packet to be transmitted to the mobile node, into the updated address indicated by the address post message, the address post message sent by the first migration control unit, and transmitting it to the mobile node.

The migration post unit in the second migration

control unit may transmit an identification key included in the migration post message, the identification key being employed to identify the mobile node.

The identification key may be an address of the mobile node assigned at one network before the network to which the mobile node is currently attached.

The identification key may be an address of the mobile node assigned before its initial migration.

The second migration control unit may be constructed to transmit to the third migration control unit the packet which has the same format as the resumed packet.

The first migration control unit may further comprise an address hold unit for holding the outdated address and the updated address by corresponding them with each other; and an address comparison unit for comparing the destination address of the received packet with the outdated address, wherein the packet transfer unit generates the conversion packet and transmits it when the address comparison unit detects that the destination address of the received packet coincides with the outdated address.

The first migration control unit may further comprise an address hold unit for holding the outdated address and the updated address by corresponding them with each other; and an address comparison unit for comparing the destination address of the packet received by the packet transfer unit with the outdated address, wherein the address post unit transmits the address post message which indicates the updated address of the mobile node to the third migration control unit, the third migration control unit transmitting the packet received by the packet transfer unit, when the address comparison unit detects that the destination address of the packet coincides with the outdated address.

The second migration control unit may further comprise an address hold unit for holding the outdated address and the updated address by corresponding them with each other; and an address comparison unit for comparing the updated address with the destination address of the packet received from one of the first migration control unit and the third migration control unit, wherein the packet resumption unit resumes the original packet from the conversion packet when the address comparison unit detects that the updated address coincides with the destination address of the packet received from one of the first migration control unit and the third migration control unit.

The third migration control unit may further comprise an address hold unit for holding the outdated address and the updated address of the mobile node by corresponding them with each other; and an address comparison unit for comparing the outdated address in the address hold unit with the destination address of the packet to be transmitted to the mobile node, wherein the packet conversion unit converts the des-

destination address of the packet to be transmitted to the mobile node into the updated address which corresponds to the outdated address in the address hold unit when the address comparison unit detects the outdated address in the address hold unit coincides with the destination address of the packet.

There may be a plurality of the first migration control units, and the second migration control unit transmits the migration post message to at least one of the first migration control units.

The migration post unit in the second migration control unit may transmit the migration post message to the first migration control unit which is attached to the network to which the mobile node was attached before its migration, each of the first migration control units has a migration post unit for transmitting to one of the other first migration control units a migration post message to post the same address as the updated address indicated by the migration post message received from the second migration control unit, and each of the first migration control units has a migration post unit for transmitting a migration post message from one of the other first migration control units to another first migration control unit to post the same address as the updated address indicated by the received migration post message.

Each of the first migration control units and the second migration control unit may further comprise a pointer hold unit for holding pointers related to the first migration control unit to which the migration post message is transmitted, and wherein the migration post unit in each of the first migration control units and the migration post unit in the second migration control unit transmit the migration post message to each of the addresses related to each of the pointers.

Each of the pointers may be a broadcast address of the network to which one of the first migration control units is attached.

Each of the pointers may be an address which is assigned to one of the first migration control units uniquely.

Each of the pointers may be the address of the mobile node which is assigned when the mobile node is attached to the same network as is the first migration control unit, and the migration post unit in the first migration control unit and the migration post unit in the second migration control unit obtain the broadcast address of the network to which each of the first migration control units is attached with referring to the address of the mobile node, and transmits the migration post message to the obtained broadcast address.

The pointer hold unit in the second migration control unit may hold a pointer related to a first migration control unit for the latest migration, which is the first migration control unit being attached to one network before the network to which the mobile node is currently attached, and the pointer hold unit in the first migration control unit holds a pointer related to an-

other first migration control unit attached to the same network as was the mobile node attached before migrating to the network to which the first migration control unit is attached.

The second migration control unit may further transmit to the first migration control unit the pointer by sending thereto the migration post message, the pointer to be held by the first migration control unit.

The first migration control unit may store into the pointer hold unit the pointer when it receives from the second migration control unit the migration post message by corresponding the pointer with the updated address indicated by the received migration post message.

Each of the first migration control units may further comprise an address hold unit for holding the outdated address and the updated address by corresponding them with each other, wherein a migration post message unit stores into the address hold unit the outdated address and the updated address by corresponding them with each other when it receives from the second migration control unit the migration post message, while converts the updated address in the address hold unit into the updated address indicated by the migration post message when it receives from the first migration control unit the migration post message and the outdated address indicated by the migration post message coincides with one of the updated addresses in the address hold unit.

The first migration control unit may be placed on a gateway, which connects networks.

The first migration control unit may be placed on the network as an individual node.

The migration post unit in the second migration control unit may transmit the migration post message to a home migration control unit, the home migration control unit being the first migration control unit which is attached to a network where the mobile node left for its initial migration, and the home migration control unit may further comprise a home migration post unit for transmitting a migration post message to a first migration control unit for the latest migration, the first migration control unit for the latest migration being the first migration control unit which is attached to the network where the mobile node left for the latest migration, to post the same updated address as is indicated by the migration post message received from the second migration control unit.

The first migration control unit may further comprise a migration post unit for transmitting the migration post message indicating the updated address of the mobile node to one of the other first migration control units when the conversion packet destined for the outdated address of the mobile node was sent therefrom to the first migration control unit.

The migration post unit in the second migration control unit may transmit to the home migration control unit the migration post message where a home



address and the updated address are corresponded with each other, the home address assigned when the mobile node is attached to the same network as is the home migration control unit, and each of the packet transfer unit and the address post unit in the home migration control unit may transmit the conversion packet and the address post message respectively with referring to the above home address and the updated address.

The second migration control unit may further comprise an outdated address post unit for transmitting to the first migration control unit for the latest migration an outdated address post message where the outdated address and the home address are corresponded with each other, the outdated address being assigned to the mobile node before the latest migration, the home migration post unit in the home migration control unit may transmit to the said first migration control unit for the latest migration the migration post message where the above home address and the updated address are corresponded with each other, and the packet transfer unit and the address post unit in the first migration control unit for the latest migration may transmit the conversion packet and the address post message respectively in accordance with the outdated address and the updated address, the outdated address and the updated address being corresponded with each other via the home address.

The outdated address post unit in the second migration control unit may transmit the above outdated address post message at a migration of the mobile node preceding the latest migration, and each of the migration post units in the second migration control unit and the home migration post unit in the home migration control unit may transmit the above migration post message at the latest migration of the mobile node.

The second migration control unit may further comprise a home migration control unit pointer hold unit for holding a pointer related to the home migration control unit, the migration post unit in the second migration control unit transmits the migration post message to the address related to the pointer, the home migration control unit may further comprise a pointer hold unit for the latest migration for holding a pointer related to the first migration control unit for the latest migration, and the home migration post unit in the home migration control unit may transmit the migration post message to the address related to the pointer.

Each of the above pointers may be the broadcast address of the network to which each of the first migration control units is attached.

Each of the above pointers may be the address assigned to each of the first migration control units uniquely.

The second migration control unit may further comprise a pointer obtainment unit for requesting to

the first migration control unit for the latest migration the pointer related to the first migration control unit for the latest migration, and the migration post unit in the second migration control unit may post the obtained pointer to the home migration control unit together with the updated address by sending thereto the migration post message.

The migration post unit in the second migration control unit may post to the home migration control unit the pointer at the migration of the mobile node preceding the latest migration, while the migration post unit may post the above updated address at the latest migration of the mobile node.

The first migration control unit may further comprise an address post suppressing unit for suppressing transmission of the address post message from the address post unit to the third migration control unit, and the address post suppressing unit may suppress transmission of the address post message when none of the first migration control units is attached to the same network as is the mobile node.

The second migration control unit may further comprise a detect unit for detecting whether or not the first migration control unit is attached to the network to which the mobile node migrates, the migration post unit in the second migration control unit may transmit to the home migration control unit the migration post message which includes the detecting result of the above detect unit together with the updated address, the home migration post unit in the home migration control unit may transmit to the first migration control unit for the latest migration the migration post message which includes the detecting result of the above detect unit together with the updated address, and the address post suppressing unit in each of the home migration control unit and the first migration control unit for the latest migration may suppress the transmission of the address post message in accordance with the detecting result of the above detect unit.

The first migration control unit may further comprise a packet transfer suppressing unit for suppressing transfer of the packet conducted by the packet transfer unit.

The first migration control unit may further comprise an address post suppressing unit for suppressing transmission of the address post message from the address post unit to the third migration control unit, and the address post suppressing unit in the first migration control unit being attached to a network to which the mobile node is not attached, may suppress the transmission of the address post message when the packet transfer suppressing unit in the first migration control unit for the latest migration suppresses transfer of the packet.

The second migration control unit may further comprise a detect unit for detecting whether or not the packet transfer suppressing unit in the first migration

control unit suppresses the transfer of the packet, the first migration control unit being attached to the network to which the mobile node migrates, and the migration post unit in the second migration control unit transmits to the home migration control unit the migration post message which includes the detecting result of the above detect unit together with the updated address, the home migration post unit in the home migration control unit may transmit to the first migration control unit for the latest migration the migration post message which includes the detecting result of the detect unit together with the updated address, and the address post suppressing unit in each of the home migration control unit and the first migration control unit for the latest migration may suppress the transmission of the address post message in accordance with the detecting result of the above detect unit.

The packet transfer suppressing unit in the first migration control unit for the latest migration may suppress the transfer of the packet conducted by the packet transfer unit, when the packet transfer suppressing unit in the first migration control unit being attached to the network to which the mobile node migrates suppresses the transfer of the packet.

The above objects may also be fulfilled by a packet transfer migration control unit in a migration communication control device, the migration communication control device being constructed to control a communication between a mobile node and a partner node, the mobile node migrating across networks and obtaining an address assigned on each network while the partner node being a communication partner of the mobile node, comprising a packet transfer unit for receiving a packet which was transmitted by the partner node to an outdated address of the mobile node, the outdated address being assigned when the mobile node migrated to a network to which the packet transfer migration control unit is attached, generating a conversion packet which holds an updated address instead of the outdated address, and transmitting the conversion packet; and an address post unit for transmitting an address post message which indicates the updated address of the mobile node to the partner node, the partner node transmitting the packet received by the packet transfer unit.

The above objects may further be fulfilled by a mobile node migration control unit in a migration communication control device, the migration communication control device being constructed to control a communication between a mobile node which migrates across networks and obtains an address assigned on each network and a partner node which is a communication partner of the mobile node, being placed on the mobile node and comprising a migration post unit for transmitting to a packet transfer migration control unit a migration post message which indicates an updated address of the mobile node when the mobile

node migrates to another network, the packet transfer migration control unit for receiving a packet which was transmitted by the partner node to an outdated address of the mobile node, the outdated address assigned when the mobile node migrated to a network to which the migration control unit for packet transfer is attached, generating a conversion packet which holds the updated address instead of the outdated address, and transmitting the conversion packet; and a packet resumption unit for receiving the conversion packet from both the packet transfer migration control unit and the mobile node, and resuming an original packet from the conversion packet.

The above objects are finally fulfilled by a partner node migration control unit in a migration communication control device, the migration communication control device being constructed to control a communication between a mobile node which migrates across networks and obtains an address assigned on each network and a partner node which is a communication partner of the mobile node, being placed on the mobile node and comprising an address post message receiving unit for receiving an address post message which indicates an updated address of the mobile node from a packet transfer migration control unit, the packet transfer migration control unit transmitting an address post message which indicates the updated address of the mobile node to the partner node; and a packet conversion unit for converting a destination address of a packet, the packet to be transmitted to the mobile node, into the updated address indicated by the address post message, and transmitting it to the mobile node.

According to the above construction, the migration communication control device of the present invention transfers and converts the packet using the address assigned to the mobile node each time it migrates across networks, obviating particular addresses or devices such as the VIP address used conventionally. For this reason, the migration communication control device of the present invention can be applied to the existing partner node and mobile node so that they can communicate continuously by transferring the packet. Moreover, it is advantageous that the migration communication control device of the present invention is not necessarily applied to all the nodes to enhance communication efficiency; the present invention can be applied only to where necessary on the existing networks. More precisely, when any existing partner node communicates with the mobile node when it migrates, the packet can be transmitted directly from the mobile nodes to the existing partner node; and it can be transferred via the first migration control unit from the existing partner node to the mobile node, thereby enhancing communication efficiency.

Furthermore, when the partner node employs the migration communication control device of the

present invention, communication efficiency is further enhanced thanks to the direct packet transmission and reception made possible by posting the update address of the mobile node from the first migration control unit to the third migration control unit.

Also, the devices such as MSS or a gateway employing the VIP are not necessarily installed at every network to which the mobile node migrates. To be precise, according to the present invention, the continuous communication is implemented even when the mobile node migrates to a network at which no special devices including above ones are installed.

#### BRIEF DESCRIPTION OF THE DRAWINGS

These and the other objects, advantages and features of the invention will become apparent from the following description thereof taken in conjunction with the accompanying drawings which illustrate a specific embodiment of the invention. In the drawings:

FIG. 1 is a block diagram depicting a construction of a migration communication control device in a first embodiment of the present invention;

FIG. 2 is a block diagram depicting a detailed construction of the migration communication control device employed as a mobile node in the first embodiment of the present invention;

FIG. 3 is a block diagram depicting a detailed construction of the migration communication control device employed as a gateway in the first embodiment of the present invention;

FIG. 4 is a block diagram showing a detailed construction of the migration communication control device employed as a stationary node in the first embodiment of the present invention;

FIG. 5 is a block diagram showing a detailed construction of the migration communication control device employed as an individual node in the first embodiment of the present invention;

FIG. 6 is an illustration showing a first example of a network to which the migration communication control devices in FIG. 2, 3, 4 are attached;

FIG. 7 is an illustration showing a second example of the network to which the migration communication control devices in FIG. 2, 3, 4 are attached;

FIG. 8 is an illustration showing a third example of the network to which the migration communication control devices in FIG. 2, 3, 4 are attached;

FIG. 9 is an illustration showing a fourth example of the network to which the migration communication control devices in FIG. 2, 3, 4 are attached;

FIG. 10 is an illustration showing (a) data in a data hold unit 1 in the mobile node (b) data in a data hold unit 1 in the migration communication control devices each employed as the gateway, the stationary node, and the individual node.

FIG. 11 is an illustration showing a format of a packet in the first embodiment of the present invention;

FIG. 12 is an illustration showing a format of a packet in the first embodiment of the present invention;

FIG. 13 is an illustration showing a content of the data hold unit 1 in the migration communication control device employed as the gateway;

FIG. 14 is an illustration showing a content of the data hold unit 1 in the migration communication control device employed as the individual node;

FIG. 15 is an illustration showing an example of a network to which the migration communication control device is attached in a second embodiment of the present invention;

FIG. 16 is a detailed block diagram depicting a home migration communication control device in the second embodiment of the present invention;

FIG. 17 is an illustration showing a content of a home mobile host list hold unit in the second embodiment of the present invention;

FIG. 18 is a detailed block diagram depicting the visitor migration communication control device in the second embodiment of the present invention;

FIG. 19 is an illustration showing a content of a visitor mobile host list hold unit in the second embodiment of the present invention;

FIG. 20 is a detailed block diagram depicting a migration address unit in the second embodiment of the present invention;

FIG. 21 is an illustration showing a content of an address hold unit in the migration address unit in the second embodiment of the present invention;

FIG. 22 is a detailed block diagram depicting a migration address unit in the second embodiment of the present invention;

FIG. 23 is an illustration showing a content of the address hold unit in the migration address unit in the second embodiment of the present invention;

FIG. 24 is an illustration showing a format of a data packet in the second embodiment of the present invention;

FIG. 25 is an illustration showing a format of a packet transfer message in the second embodiment of the present invention;

FIG. 26 is an illustration showing a flow of a data packet transmitted between devices in the second embodiment of the present invention;

FIG. 27 is an illustration showing a communication sequence in FIG. 26;

FIG. 28 is an illustration showing a construction of each data packet in FIG. 26;

FIG. 29 is an illustration showing a change in the content of each hold unit in FIG. 26;

FIG. 30 is an illustration showing a flow of each data packet transmitted between devices at an operation example in the second embodiment of

the present invention;  
 FIG. 31 is an illustration showing a communication sequence in FIG. 30;  
 FIG. 32 is an illustration showing a construction of each data packet in FIG. 30;  
 FIG. 33 is an illustration showing a change in the address hold unit in each device in FIG. 33;  
 FIG. 34 is an illustration showing a flow of a data packet transmitted between devices at an operation example in the second embodiment of the present invention;  
 FIG. 35 is an illustration showing the communication sequence in FIG. 34;  
 FIG. 36 is an illustration showing a construction of each data packet in FIG. 34;  
 FIG. 37 is an illustration showing a change in the address hold unit in each device in FIG. 34;  
 FIG. 38 is an illustration showing a flow of each data packet transmitted between devices at an operation example in the second embodiment of the present invention;  
 FIG. 39 is an illustration showing a communication sequence in FIG. 38;  
 FIG. 40 is an illustration showing a construction of each data packet in FIG. 38;  
 FIG. 41 is an illustration showing a change in the address hold unit in each device in FIG. 38;  
 FIG. 42 is an illustration showing a flow of each data packet transmitted between devices in the second embodiment of the present invention;  
 FIG. 43 is an illustration showing a flow of each data packet transmitted between devices in the second embodiment of the present invention;  
 FIG. 44 is an illustration showing a flow of each data packet transmitted between devices in the second embodiment of the present invention; and  
 FIG. 45 is an illustration showing a flow of each data packet transmitted between devices in the second embodiment of the present invention.

#### DESCRIPTION OF THE PREFERRED EMBODIMENT

##### [Embodiment 1]

A construction of a migration communication control device in a first embodiment of the present invention is described hereunder with referring to FIGs. Hereinafter, the mobile node and partner node in the related art as well as in the summary of the invention are referred to as a mobile host and a stationary host, respectively.

FIG. 1 is an illustration showing the construction of the migration communication control device comprising a data hold unit 1, an application unit 2, a migration address unit 3, and a communication control unit 4.

The data hold unit 1 holds a couple of addresses

of a mobile host by corresponding them. Each of the addresses in the data hold unit 1 is assigned before and after a migration of the mobile host.

The application unit 2 checks a connection as well as monitors a timer. The unit 2 is relevant for a higher layer in OSI model, which includes an application layer. For example, the unit 2 operates as TCP at TCP/IP (Transmission Control Protocol/Internet Protocol) or a layer which is higher than TCP.

The migration address unit 3 processes a migration address with referring to data in the data hold unit 1. The concrete operation of the migration address unit 3 varies depending on a type of the migration communication control device comprising the unit 3, and this will be described in detail later.

The communication control unit 4 controls the communication. The unit 4 is relevant for a lower layer in the OSI model. For example, the unit 4 operates as a layer which is lower than IP at TCP.

The application unit 2 and the communication control unit 4 are the same units as ones implemented on a general host. Besides the unit 2 and 4, the migration communication device in the first embodiment of the present invention includes the data hold unit 1 and the migration address unit 3; thereby implements an operation unique to this case. That is, the data hold unit 1 and the migration address processing unit 3 are attached to the mobile host which migrates across networks, or a stationary host which is attached to a network fixedly (for example, a gateway or a server); otherwise, they operate alone. Each device comprising the unit 1 and 3 supports a continuous communication unaffected by migration of the mobile host besides providing its own function.

The data hold unit 1 and the mobile address unit 3, which are included in the devices attached to the network, are described in FIGs. 2, 3, 4, 5. FIG. 2 shows a migration communication control device where the unit 1 and the unit 3 are attached to the mobile host which migrates across networks; FIG. 3 shows a migration communication control device where the unit 1 and the unit 3 are attached to a gateway which connects the networks; FIG. 4 shows a migration communication control device where the unit 1 and the unit 3 are attached to the stationary host, which is the communication partner of the mobile host; and FIG. 5 shows a migration communication control device attached to the network itself.

The migration communication control device in FIG. 2 (hereinafter referred to as a mobile host) further includes the application unit 2, the communication control unit 4, and an address obtainment unit 25, besides the data hold unit 1 and the migration address unit 3.

Each of the application unit 2 and the communication control unit 4 operates as the above; while the unit 2 together with the unit 4 operate as a conventional stationary host.

The address obtainment unit 25 obtains an address of the mobile host assigned when it has migrated to another network. Although other options can be considered, such as employing a manual setting by an operator or communicating with a server computer which administrates addresses of the network, it is supposed here that the address is obtained in accordance with an instruction of a system administrator or the operator. The address obtainment unit 25 is also possessed by a general host and will not be described in detail.

The addresses held in the data hold unit 1 are obtained by the address obtainment unit 25.

The migration address unit 3 (enclosed with a broken line) consists of a response message transmission unit 20, a marked packet conversion unit 21, a migration address setting unit 26, a migration post transmission unit 27, a reception packet unit 28, and a marked packet resumption unit 29.

The response message transmission unit 20 transmits the packet which responds to the received packet if the response is needed.

The marked packet conversion unit 21 converts a packet received from the response message transmission unit 20 as well as the application unit 2 into a marked packet by converting the address of the received packet and marking the packet.

The migration address setting unit 26 stores the address obtained by the address obtainment unit 25 into the data hold unit 1. The address obtained by the unit 25 is the address of the mobile host assigned after the migration, and the unit 26 stores it into unit 1 by corresponding it to the address of the mobile host assigned before the migration.

The migration post transmission unit 27 posts via the communication control unit 4 that the address obtained by the unit 25 is held in the data hold unit 1 together with the correspondence between a couple addresses each of which assigned before and after the migration.

The reception packet unit 28 detects whether or not the received packet is marked, and sends the unmarked packet to the application unit 2 while sending the marked packet to the marketed packet resumption unit 29.

The marked packet resumption unit 29 resumes the marked packet.

The migration communication control device in FIG. 3 (hereinafter referred to as a gateway) further includes the application unit 2 and the communication control unit 4 besides the data hold unit 1 and the migration address unit 3 (enclosed with a broken line).

Each of the application unit 2 and the communication control unit 4 operates described the above, and the unit 2 together with the unit 4 operate as a conventional gateway.

The data hold unit 1 holds the correspondence between a couple of the addresses of the mobile host

each of which assigned before and after migration.

The migration address unit 3 consists of a reception packet unit 35, a migration post information unit 36, an address comparison unit 37, an address conversion post transmission unit 38, and a marked packet conversion unit 39.

The reception packet unit 35 detects whether or not the received packet is the packet comprising a migration post message, which is transmitted by the mobile host. The unit 35 then sends the migration post message to the migration post information unit 36 while sending the other packets to the address comparison unit 37.

In accordance with the migration post message received from the reception packet unit 35, the migration post information unit 36 stores in the data hold unit 1 the correspondence between a couple of the addresses of the mobile host each of which assigned before and after the migration. The unit 36 also sends the migration post message to the address conversion post transmission unit 38.

The address comparison unit 37 detects whether or not the destination address of the packet received from the reception packet unit 35 coincides with the address of the mobile host assigned before migration, which is held in the data hold unit 1. When they coincide with each other, the unit 37 further sends to the marked packet conversion unit 39 the address assigned after the migration, which corresponds to the address which coincides with the destination address, as well as the packet received from the reception packet unit 35. On the other hand, when they do not coincided with each other, the unit 37 implements a function of a gateway by sending the packet to the application unit 2.

The address conversion post transmission unit 38 transmits to the destination address of the above packet received from the reception packet unit 35 an address conversion post message to inform that the address of the mobile host changes when the address comparison unit 37 detects a coincidence. Also the unit 38 transmits the address conversion post message to the network which satisfies the following two conditions: (1) the network where the address assigned before the migration, which is held in the data hold unit 1, is other than 0 (2) the migration communication control device employs as the gateway is not attached to the network. When the address conversion post message is transmitted to the network, which satisfies the above conditions, its destination address is a broadcast address of the network. The broadcast address consists of a network part and a host part, and every bit of the host part is 1.

The marked packet conversion unit 39 generates a marked packet when the address comparison unit 37 detects a coincidence. The unit 39 generates it by marking a general packet after converting the destination address of the packet. Then, the unit 39 trans-

mits it.

The migration communication control device in FIG. 4 (hereinafter referred to as a stationary host) further includes the application unit 2 and the communication control unit 4 besides the data hold unit 1 and the migration address unit 3 (enclosed with a broken line).

Each of the application unit 2 and the communication control unit 4 operates as described the above, and the unit 2 together with the unit 4 operate as a conventional stationary host (not migrate).

The data hold unit 1 holds the correspondence between a couple of the addresses of the mobile host each of which assigned before and after the migration.

The migration address unit 3 consists of a reception packet unit 45, a marked packet resumption unit 46, an address conversion post information unit 47, an address comparison unit 48, and a marked packet conversion unit 49.

The reception packet unit 45 detects whether the received packet is the packet comprising the address conversion post message, the marked packet, or the other packets. The address conversion post message is transmitted by the gateway. Then the unit 45 sends the address conversion post message to the address conversion post information unit 47, the marked packet to the marked packet resumption unit 46, and the other packets to the application unit 2.

The marked packet resumption unit 46 resumes the unmarked packet from the marked packet, which is received from the reception packet unit 45.

The address conversion post information unit 47 obtains from the packet comprising the address conversions post message, which is received from the reception packet unit 45, the correspondence between the address of the mobile host assigned before the migration and the one assigned after the migration, and stores it into the data hold unit 1.

The address comparison unit 48 detects whether or not destination address of the packet received from the application unit 2 coincides with the address of the mobile host assigned before migration, which is held in the data hold unit 1. When they coincide with each other, the unit 48 further sends to the marked packet conversion unit 49 the address assigned after the migration, which corresponds to the address which coincides with the destination address, as well as the packet received from the application unit 2. On the other hand, when they do not coincided with each other, the unit 48 sends the packet to the communication control unit 4.

The marked packet conversion unit 49 generates a marked packet when the address comparison unit 37 detects a coincidence. The unit 49 generates it by marking a general packet after converting the destination address of the packet. Then, the unit 49 transmits it.

The migration communication control device in FIG. 5, which is attached to the network by itself, consists of the data hold unit 1, the migration address unit 3 (enclosed with a broken line), and the communication control unit 4.

The data hold unit 1 holds the correspondence between a couple of the addresses of the mobile host each of which assigned before and after the migration.

The migration address unit 3 consists of the reception packet unit 35, the migration post information unit 36, the address comparison unit 37, the address conversion post transmission unit 38, and the marked packet conversion unit 39. The units integrating the migration address unit 3 operate substantially same as equivalent units integrating the gateway in FIG. 3 except the following.

In FIG. 3 the address conversion post transmission unit 38 transmits the address conversion post message to the network satisfying both of the two conditions, which are described in the above; whereas, the address conversion post transmission unit 38 in FIG. 5 transmits the address conversion post message to the broadcast address of the network as long as the network satisfies the first condition, that is it transmits the address conversion post message to the network when the address assigned before the migration, which is held in the data hold unit 1, is other than 0.

FIG. 6 shows a first example of a network to which the migration communication control device as the mobile host in FIG. 2, the migration communication control device as the gateway in FIG. 3, and migration communication control device as the stationary host in FIG. 4 are attached. In the figure numeral 11 denotes a mobile host in FIG. 2, which migrates from a network A to a network B and obtains an address  $\alpha$  assigned on the network A as well as an address  $\beta$  assigned on the network B.

Numeral 12 denotes a stationary host in FIG. 3, which is attached to the network B and obtains an address  $\gamma$  assigned thereon.

Numeral 12' denotes a stationary host in FIG. 3, which is attached to the network A and obtains an address  $\gamma'$  assigned thereon.

Numeral 13 denotes a gateway in FIG. 3, which has an address g. The gateway 13 is attached to both the network A and the network B.

The address on each network is assigned by a system administrator.

FIG. 7 shows a second example of a network to which the mobile host in FIG. 2, the gateway in FIG. 3, and the stationary host in FIG. 4 are attached. The stationary host is not illustrated in FIG. 7 since its location does not affect the communication with the mobile host.

In the figure the mobile host 11 migrates across network 1-4, and obtains an address m, m', m'', m'''

assigned on each network respectively.

The network 5 as well as each of the network 1-4 (hereinafter referred to as the net 5, and the net 1-4 respectively) are connected with each other by a gateway 1-4, as shown in the figure.

A gateway 1-4 (hereinafter referred to as gw 1-gw 4) is the migration communication control device employed as the gateway in FIG. 3.

FIG. 8 shows a third example of the network to which the mobile host in FIG. 2, the gateway in FIG. 3, and the stationary host in FIG. 4 are attached. Construction of this network is substantially same as the second example of the network in FIG. 6 although operation thereof is different from the second example, which will be described later.

FIG. 9 shows a fourth example of the network to which the mobile host in FIG. 2, the migration communication control device in FIG. 5, the stationary host in FIG. 4 are attached. The migration communication control device as the stationary host will not be described here.

In the figure, numeral 11 denotes the mobile host which migrates across the network 1-4 and obtains the address  $m$ ,  $m'$ ,  $m''$ ,  $m'''$  assigned on each network respectively.

The network 5 as well as each of the network 1-4 (hereinafter referred to as the net 5, and the net 1-4 respectively) are connected with each other by a gw 1-4, as shown in the figure.

Each of the migration communication control unit 1-4 (hereinafter referred to as S1-S4) is relevant for the one in the FIG. 5.

An address used in the first embodiment of the present invention is described hereunder. Each address consists of a network part, which is assigned on each network and shared by every host attached to that network, as well as a host part, which is assigned to each host uniquely.

A broadcast address is a special kind of address, which can be divided into two types. The first one is the broadcast address used as the destination address in transmitting a packet from a network to another network, such as the broadcast address where every bit of the host part is 1. When the first type of the broadcast address is used as the destination address of the packet, the packet is transferred by the gateway to the network directed by the network part of the broadcast address. The other one is used in transmitting a packet within a network, such as the broadcast address where every bit of both the host part and the network part is 1. When the second type of the broadcast address is used as the destination address of the packet, the packet is transmitted to all the devices attached to the network, which includes the broadcast address. However, the gateway does not transfer the packet to any other network.

Operations of the migration communication control device in the first embodiment of the present in-

vention are described hereunder with referring to drawings.

(operation example in FIG. 6)

In FIG. 6, when the mobile host migrates from the network A to the network B, the migration communication control device is operated as follows.

In a first operation, the mobile host and the gateway operate when the mobile host migrates across networks.

In a second operation, the stationary host transmits a packet to an address of the mobile host which was assigned before the migration.

In a third operation, the stationary host transmits the packet to an address of the mobile host which has been assigned after the migration.

In a fourth operation, the mobile host receives the packet which is transmitted by the stationary host.

In a fifth operation, the mobile host sends a response message to the stationary host.

(first operation in FIG. 6)

In FIG. 6 the mobile host 11 attached to the network A (enclosed with a broken line) migrates to the network B to complete ongoing communication with the stationary host 12, which is attached to the network B. When migrating to the network B, the address obtainment unit 25 in the mobile host 11 (FIG. 2) obtains the address  $\beta$  assigned on the network B.

Immediately after obtaining the address  $\beta$ , the address obtainment unit 25 gives the address  $\beta$  to the migration address setting unit 26 and the migration post transmission unit 27. The migration address setting unit 26 stores the address  $\beta$  into the data hold unit 1 by corresponding it to the address  $\alpha$ , which is the address assigned before the migration. FIG. 10 (a) shows the content of the data hold unit 1. The migration post transmission unit 27 gives to the gateway 13 via the communication control unit 4 a packet comprising migration post message and the correspondence between the address  $\alpha$  and the address  $\beta$ , so that the gateway 13 will know that the mobile host 11 has migrated to the network B. The mobile host 11 can transmit the packet both before and after the migration. In FIG. 6 a packet 51 is transmitted before the migration, and its format is shown in FIG. 11 (a). As shown in FIG. 11 (a), the packet 51 consists of a destination address 91, a source address 92, and data 93. The data 93 further comprise a message type 98, an address before migration 94, and an address after migration 95.

Receiving from the communication control unit 4 the packet 51, the gateway 13 sends it to the reception packet unit 35, the unit 4 and the unit 35 being in FIG. 3. From the message type 98 in FIG. 11 (a), the gateway 13 identifies the packet 51 with the migra-

tion post message, and gives the packet 51 to the migration post information unit 36. The migration post information unit 36 obtains from the data 93 in the data packet 51 the address before migration  $\alpha$  and the address after the migration  $\beta$ ; then stores them into the data hold unit 1 by corresponding them with each other. The content of the data in the data hold unit 1 is shown in FIG. 10 (b).

Additionally, the destination address 91 of the packet in FIG. 11 (a), can be the broadcast address of the network A, where the network part names the network A and every bit of the host part is 1. When the broadcast address is employed, every stationary host attached to the network A, including the gateway 13, receives the correspondence of the addresses each of which assigned before and after the migration. In this case communication control unit 4 in the stationary host 12' receives the data packet 51, and gives it to the reception packet unit 45, the unit 4 and the unit 45 in FIG. 4. From the message type 98 in FIG. 11 (a), the reception packet unit 45 identifies the packet 51 with the migration post message, and gives the packet 51 to the address conversion post information unit 47. The unit 47 obtains from the data 93 in the data packet 51 the address before migration  $\alpha$  and the address after the migration  $\beta$  and stores them into the data hold unit 1 by corresponding them with each other. Once those addresses are stored in the data hold unit 1, the stationary host 12' can transmit a packet to the address assigned after the migration instead of transmitting it to the address before the migration, the same to other stationary hosts attached to the network A.

(second operation in FIG. 6)

In the second operation, the stationary host 12 transmits a packet to the address assigned before the migration after the mobile host 11 migrates to the network B and obtains the address  $\beta$  assigned on the network B. It is supposed that the mobile host 11 transmits the packet 51, which comprises the migration post message, to the gateway 13 rather than to the broadcast address of the network A.

The stationary host 12, which is not notified that the mobile host 11 has migrated to the network B, transmits the packet to the address  $\alpha$  of the mobile host, which was assigned before the migration. A packet 52 in FIG. 6 is transmitted by the stationary host 12 to the address  $\alpha$  of the mobile host 11, and its format is shown in FIG. 11 (f). The packet 52 is received by the gateway 13. Because the gateway 13 is located between the source address of the packet 52 and the address of the mobile host  $\alpha$  assigned before the migration, and also it is attached to the network A, to which the mobile host 11 was attached before the migration.

The gateway 13 employs its devices in FIG. 3 to

implement its functions including reception of the packet. That is, the communication control unit 4 in the gateway 13 receives the packet 52, and sends it to the reception packet unit 35 in the migration address unit 3. The reception packet unit 35 identifies the packet 52 with a general packet and gives it to the address comparison unit 37. The unit 37 compares the destination address  $\alpha$  of the packet 52 with the address before the migration, which is held in the data hold unit 1; then detects whether or not they coincide with each other. When the destination address of the received packet does not coincide with the address assigned before the migration, the address comparison unit 37 sends the packet to the application unit 2. On the other hand, when they coincide with each other, the address comparison unit 37 obtains from the data hold unit 1 the address  $\beta$  of the mobile host assigned after the migration, which corresponds to the address  $\alpha$ ; then sends it both to the address conversion post transmission unit 38 and the marked packet conversion unit 39.

As is described the above, the packet 52 is transmitted to the address  $\alpha$  of the mobile host 11 by the stationary host 12. Therefore, the address conversion post transmission unit 38 notifies the stationary host 12 that the address of the mobile host 11 has changed by transmitting thereto the packet 53. FIG. 11 (b) shows the packet 53. Simultaneously, the marked packet conversion unit 39 converts the packet 52 into the packet 53 by rewriting the destination address of the packet 52 to the address  $\beta$  assigned after the migration, returning thereto the previous destination address of the packet 52 as additional information, and marking to show that its destination address has changed; then sends the packet to the communication control unit 4. Thereby, the packet 52, which is converted into the marked packet 52', is transferred from the address  $\alpha$  of the mobile host 11 assigned before the migration to the address  $\beta$  assigned after the migration. FIG. 12 (e) shows the packet 52'.

Receiving the packet 53 from the communication control unit 4 in the stationary host 12, it sends its packet 53 to the reception packet unit 45, the unit 4 and the unit 45 being in FIG. 4. From the message type 98 in FIG. 11 (b), the reception packet unit 45 identifies the packet 53 with the address conversion post message, and gives the packet 53 to the address conversion post information unit 47. The address conversion post information unit 47 obtains from the data 93 in the data packet 53 the address before migration  $\alpha$  and the address after the migration  $\beta$ ; then stores them into the data hold unit 1 by corresponding them with each other. Thereby, the stationary host 12 obtains the address of the mobile host 11 assigned after the migration, so that a direct communication between the stationary host 12 and the mobile host 11 is implemented.



In the second operation the migration communication control device comprising the units in FIG. 4 is employed as the stationary host 12. However, a conventional stationary host, which is not constructed as the migration communication control device can also be communication partner of the mobile host if it is attached to a network. Therefore, hereunder a communication between the mobile host 11 and the conventional stationary host is described.

When the conventional stationary host transmits a packet to the address of the mobile host 11 assigned before the migration after the mobile host 11 has migrated to another network, the gateway 13 transfers the packet to the address of the mobile host 11 assigned after the migration as well as sends to the stationary host the packet 53 comprising the address conversion post message in FIG. 11 (c). This operation is same as the above.

However, when receiving the packet 53, the stationary host disposes it since it does not support the address conversion post message and judges the packet 53 is not a required packet. Thus, the conventional stationary host cannot utilize the packet 53 to detect the address of the mobile host assigned after the migration nor hold the correspondence of the addresses each assigned before and after the migration.

Therefore, the stationary host gives the packet only to the address of the mobile host 11 assigned before the migration. Then, the gateway transfers the packet to the address of the mobile host 11 assigned after the migration, and the mobile host 11 receives the packet. The message from the mobile host 11, such as the response message, is transmitted to the stationary host directly, so that it is received by the stationary host without fail.

Thus, the conventional stationary host transmits a packet to the mobile host indirectly and receives a packet from the mobile host directly. Continuous communication unaffected by the mobile host's migration can be implemented, even when the conventional stationary host is employed.

(third operation in FIG. 6)

In the third operation, the stationary host 12 transmits the packet to the address  $\beta$  of the mobile host 11 assigned after the migration with referring to the correspondence of the addresses each assigned before and after the migration, which is held in the data hold unit 1. The third operation is described hereunder with referring to FIG. 4.

The stationary host 12 employs its devices in FIG. 4 to implement conversion of the destination address and the transmission of the packet, both of which integrate the third operation. That is, application unit 2 sends to the address comparison unit 48 the packet 54, whose destination address is the ad-

dress  $\alpha$  of the mobile host 11 assigned before the migration. FIG. 11 (f) shows a format of the packet 54. Then, the comparison unit 48 obtains the destination address of the packet 54 and detects whether or not it coincides with the address before the migration, which is held in the data hold unit 1.

The comparison unit 48 sends the packet 54 to the communication control unit 4 when the above addresses do not coincide with each other while it sends the packet 54 to the marked packet conversion unit 49 when the above addresses coincide with each other. In the third operation the coincidence is detected since the corresponded between the address  $\alpha$  and the address  $\beta$  is stored in the data hold unit 1. Therefore, the packet 54 is sent to the marked packet conversion unit 49. Then the marked packet conversion unit 49 obtains from the data hold unit 1 the address  $\beta$  of the mobile host assigned after the migration, which corresponds to the address  $\alpha$  as well as converts the packet 54 into the packet 54' by converting the destination address  $\alpha$  into the address  $\beta$ , returning thereto the original destination address  $\alpha$  as additional information, and marking the packet 54 to show that its destination address has changed; then sends the packet 54' to the communication control unit 4. FIG. 11 (c) shows a format of the packet 54'. Since the destination address of the packet 54' is an updated address of the mobile host 11, the packet 54' is given to the mobile host 11 without fail.

(fourth operation in FIG. 6)

In the fourth operation, the mobile host 11 receives the marked packet 54' and obtains the original unmarked packet 54 by resuming the packet 54'. This operation is described hereunder with referring to FIG. 2.

The mobile host 11 employs its devices in FIG. 2 to implement its operation. That is, the communication control unit 4 receives the packet 54' and sends it to the reception packet unit 28. The reception packet unit 28 detects that the received packet 54' is marked, and sends it to the marked packet resumption unit 29. The unit 29 obtains the original destination address  $\alpha$ , which is held in the additional information 97, and replaces the current destination address  $\beta$  of the packet 54' with the address  $\alpha$ . Then it sends the packet 54' to the application unit 2. Thus, the mobile host 11 can receive the packet destined for its outdated address.

(fifth operation in FIG. 6)

In the fifth operation, the mobile host 11 sends to the stationary host 12 a packet comprising a response message (hereinafter referred to as a response packet) or a packet excluding the response message (hereinafter referred to as a non-response

packet). A type of the received packet determines whether or not it is responded with the response packet.

When the packet 54' is responded with a response packet, the mobile host 11 employs its devices in FIG. 2 to send the response packet. That is, the response message transmission unit 20 builds the response packet, and sends it to the marked packet conversion unit 21 together with the destination address  $\alpha$  of the packet 54'.

The mobile host 12 also employs its devices to send the non-response packet 55. That is, the application unit 2 gives the address  $\alpha$  assigned before the migration and the non-response packet to the marked packet conversion unit 21. The unit 21 sends the received packet to the stationary host 12 via the communication control unit 4 without marking it. FIG. 11 (e) shows the packet sent by the unit 21 to the stationary host 12.

The communication control unit 4 in the stationary host 4 receives the packet 55, and gives it to the reception packet unit 45. The unit 45 detects that the packet 55 is the non-response packet, so that it gives the packet 55 to the application unit 2. Thus, the stationary host and the mobile host implement a continuous communication unaffected by mobile host's migration. Although the migration communication control device is employed as the stationary host 12 in this embodiment, the conventional host can also be employed to transmit the non-response packet.

In the above, the unmarked response packet and the unmarked non-response packet are sent to the mobile stationary host 12. On the other hand, hereunder the operation of the mobile host 11 at conversion of the response packet and the non-response packet into the marked ones is described. This will be employed effectively in a communication between mobile hosts.

Receiving the unmarked packet from the application unit 2, the marked packet conversion unit 21 generates a packet 55' where the destination address and the source address are the address  $\gamma$  of the stationary host 12 and the address  $\beta$  assigned after the migration respectively. Also in generating the packet 55', the application unit 2 gives to the received packet the address  $\alpha$  assigned before the migration as additional information as well as marks the received packet to indicate that the destination address has converted. FIG. 11 (d) shows a format of the packet 55'. Then the application unit 2 sends the packet 55' to the stationary host 12 via the communication control unit 4.

The communication control unit 4 in the stationary host 12 receives the packet 55', and sends it to the reception packet unit 45. Detecting the packet 55' is the marked packet, the reception packet unit 45 sends it to the marked packet resumption unit 46. The unit 46 resumes the packet 55' into the packet 55 by

unmarking it and replacing the source address thereof with the address  $\alpha$  assigned before the migration, which is held as the additional information. A format of the packet 55 is shown in FIG. 11 (e). Thus, the stationary host and the mobile host implement a continuous communication unaffected by mobile host's migration.

(operation example in FIG. 7)

In FIG. 7, when the mobile host migrates across the network 1, 2, 3, and 4, and obtains a temporary address assigned on each network, the newest address of the mobile host is transmitted to the stationary host, which operates as communication partner.

(migration from network 1 to network 2)

The address of the mobile host is  $m$  when it is attached to the network 1. When migrating from the network 1 to the network 2, the mobile host 11 replaces its address with  $m'$  assigned on the network 2. Then the mobile host 11 notifies the migration communication control device attached to the network 1 that it has migrated to the network 2 by sending thereto a packet comprising a migration post message. In FIG. 7 the migration communication control device  $gw$  1,  $gw$  2 attached to the network 1 receive the migration post packet 61, and store it into its own data hold unit 1. The operation in FIG. 7 is substantially same as the operation in FIG. 6 except that in FIG. 7 the packet 61 holds the address of the mobile host assigned before the last migration besides the correspondence of the addresses each assigned before and after the current migration. The address assigned before the last migration makes the  $gws$  prepare for further migration of the mobile host, which will be described later. A format of the packet 61 is shown in FIG. 12 (a). Since the migration from the network 1 to the network 2 is the first migration in FIG. 7, the packet 61 holds 0 at the address assigned before the last migration.

The  $gw$  1 and the  $gw$  2 store in the data hold unit 1 the correspondence of the addresses each assigned before and after the migration, as well as the address assigned before the last migration. As shown in FIG. 13 (a),  $m-m'$  and 0 are stored in the data hold unit 1 of each of the  $gw$  1 and the  $gw$  2.

Then, the  $gw$  1 and the  $gw$  2 detects from 0 at the address assigned before the last migration that no migration had been conducted before the current migration.

The broadcast address of the network 1 can be employed as the destination address of the migration post packet 61. If the packet is destined for the broadcast address, every host attached to the network 1, which includes the  $gw$  1 and the  $gw$  2, will hold the correspondence of the addresses each of which as-

signed before and after the migration as well as the address assigned before the last migration. Thereby, the hosts attached to the network 1 can communicate with the mobile host directly.

(migration from network 2 to network 3)

When migrating from the network 2 to the network 3, the mobile host 11 obtains  $m''$  at the address assigned after the migration. Then the mobile host 11 notifies the gw 2 and a gw 3, both of which are attached to the network 2, that the mobile host 11 has migrated to the network 3 by transmitting thereto a packet comprising the migration post message, referred to as a packet 62 in FIG. 7. FIG. 12 (b) shows a format of the packet 62, which is transmitted to the gw 2. The broadcast address of the network 2 can be employed as the destination address of the packet 62. When the packet 62 is transmitted to the broadcast address of the network 2, every host attached to the network 2, which includes the gw 2 and the gw 3, holds the correspondence of the addresses each assigned before and after the migration.

The gw 2 employs its devices in FIG. 3 to process the packet 62. That is, receiving the packet 62, the gw 2 sends it to the migration post information unit 36 via the communication control unit 4 and the reception packet unit 35, then refers to the data hold unit 1 where  $m \rightarrow m'$  and 0 are still held at the address correspondence and at the address assigned before the last migration respectively. The migration post information post unit 36 obtains from the packet 62  $m'-m''$  as the newly assigned correspondence between the addresses each of which assigned before and after the current migration, the migration from the network 2 to the network 3. Then, it detects whether or not the address  $m'$  coincides with the address held in the data hold unit 1 as the address assigned after the last migration. Since the unit 36 detects the coincidence, it replaces the address  $m'$  in the unit 1 with the address  $m''$  as well as replaces the correspondence  $m-m'$  with the correspondence  $m-m''$ .

Also the migration post information unit 36 sends to the data hold unit 1 the address  $m$  assigned before the last migration together with the address correspondence  $m'-m''$  obtained from the current migration. Now the data hold unit 1 in the gw 2 holds the address  $m$  at the address assigned before the last migration and the address correspondence  $m'-m''$  at the correspondence of the addresses each of which assigned before and after the migration as well as the address 0 at the address assigned before the last migration as well as the address correspondence  $m-m'$  at the correspondence of the addresses each of which assigned before and after the migration. After updating as well as adding the addresses in the data hold unit 1, the migration post information unit 36 sends to the address conversions post transmission

unit 38  $m'-m''$  as the newly obtained correspondence of the addresses before and after the current migration.

The address conversion post transmission unit 38 detects the network satisfying the following conditions with referring to the data hold unit 1 and then transmits the address conversion post message to the broadcast address of the detected network. That is, the address conversion post message is transmitted to the network where the address assigned before the migration, which is held in the data hold unit 1, is other than 0 as well as the migration communication control device employed as the gateway is not attached. Although in the migration from the network 2 to the network 3, the data hold unit 1 holds  $m$  at the address assigned before the last migration, the gw 2 is attached to the network 1; therefore, the unit 38 does not transmit the address conversion post to the network 1.

The packet 62 is also received by gw 3. When receiving the packet 62, the gw 3 employs its own devices in FIG. 3 to process the packet 62, which is substantially same as does the gw 2 except the following. That is, the address conversion post transmission unit 38 of the gw 3 detects that the gw 3 is not attached to the network 1. Also it is detected that the mobile host 11, attached to the network 1, has the address  $m$  as the address assigned before the last migration. Therefore, the unit 38 of the gw 3 transmits to the broadcast address of the network 1 a packet comprising the address conversion post message, which is referred to as a packet 63. FIG. 12 (c) shows the packet 63.

The packet 63 is received by the gw 2, the gw 1, both of which are attached to the network 1. Although it is also received by the stationary host 11, this will not be described here. Obtaining the current address correspondence  $m'-m''$  from the packet 63, where  $m'$  coincides with the address which has been held in the hold unit 1 at the address obtained after the migration, the gw 1 changes the  $m-m'$  in the data hold unit 1 into the  $m-m''$  by replacing  $m'$  with  $m''$  as the address assigned after migration.

On the other hand, the data hold unit 1 of the gw 2 had gained from the packet 62 the above information before receiving the packet 63. Therefore the content of the unit 1 of the gw 2 does not change across reception of the packet 63. This is because the gws of the present invention locate on a gateway, which connects a couple of networks. Due to its location, each gw receives packets from two networks. However, actually the packet 62 is destined for the network 2 and the packet 63 is destined for the network 1. Therefore, even though the gw 2, which are attached to both the network 1 and the network 2, receives both the packet 62 and 63 by the gw 2, this will not cause any problem in the communication between the stationary host 12 and the mobile host 11.

FIG. 13 (b) shows the content of the data hold unit 1 in each of the gws.

(migration from network 3 to network 4)

When migrating from the network 3 to the network 4, the mobile host 11 obtains  $m''$  as the address assigned after the migration. Then the mobile host 11 sends to the gw 3 and a gw 4, both of which are attached to the network 3, a packet comprising the migration post message. The packet received by the gw 3 is referred to as a packet 64. The broadcast address of the network 3 can be employed as the destination address of the packet 64. When the packet 64 is destined for the broadcast address of the network 3, every host attached to the network 2, which includes the gw 3 and the gw 4, obtains from the packet the correspondence of the addresses each of which assigned before and after the migration from the network 3 to the network 4.

The gw 3 employs its devices in FIG. 3 to process the packet 64. That is, receiving the packet 64, the gw 3 converts the content of the data hold unit 1 by replacing the address correspondence  $m-m''$  with  $m-m'''$ , newly holding  $m''-m'''$  obtained from the packet 64 as well as the address  $m'$  assigned before the last migration. Then, the address conversion post transmission unit 38 of the gw 3 transmits the address conversion post message to the network satisfying the following condition. That is, the address conversion post message is transmitted to the network where the address assigned before the migration, which is held in the data hold unit 1, is other than 0 as well as the gw 3 it self is not attached. The packet including the address conversion post message is referred to a packet 65, and the packet is transmitted to the broadcast address of the network 1. FIG. 7 (c) shows the packet 65.

The packet 64 is also received by gw 4. When receiving the packet 64, the gw 4 renews the content of the data hold unit 1 by replacing  $m'-m''$  with  $m'-m'''$  as well as newly holding the address  $m'$  as the address assigned before the last migration. Further, the address conversion post transmission unit 38 of the gw 4 detects that the gw 4 is not attached to the network 2 which has the address other than 0 at the address assigned before the last migration; therefore, the unit 38 of the gw 4 transmits a packet comprising the address conversion post message, which is referred to as a packet 66, to the broadcast address of the network 2. FIG. 7 (c) shows the packet 66.

Receiving the packet 65, 66, the gw 2 and the gw 1 renew the content of its data hold unit 1, which is substantially the same as the above.

The gw 3 and the gw 2 receives the same information twice since the former receives the packet 64 and 65 while the latter receives the packet 65 and 66. This is because gws of the present invention locate on

a gateway and receives packets from a couple of networks, which is described the above.

FIG. 13 (c) shows the content of the data hold unit 1 in each of the gws. Thus, according to the gws of the present invention, the packet transmitted to any of the addresses  $m, m', m''$  is transferred by the gws to the updated address of the mobile host, the gws also notify the stationary host of the updated address.

For example, when the stationary host is not notified of the updated address of the mobile host and transmits a packet to the address  $m'$ , the packet is received by the gw 2 and the gw 3, both of which are attached to the network 2. Then, the gw 2 and the gw 3 transfers the packet to the updated address of the mobile host as well as notifies the stationary host of the updated address. Thereby, the stationary host obtains the updated address of the mobile host, so that it will be able to communicate with the mobile host directly. The packet destined for the address  $m'$  is received by both the gw 2 and the gw 3, since they are attached to the network 2. Thus, the mobile host receives the same packet twice, once from the gw 2 and the other time from the gw 3, and the stationary host receives the same message twice; however, the repeated packet or the message can be simply ignored, so that this will not cause any problem in the communication between the stationary host and the mobile host. The repeated packet or the message is observed when the two gws are attached to each network in FIG. 7; whereas it is not observed when only one migration communication control device is attached to each network, which will be described later at the operation in FIG. 9.

(operation example in FIG. 8)

In FIG. 6, FIG. 7, the stationary host transmits the data packet to the outdated address after mobile host notifies the gws that it has migrated to another network. Then the gws transmit the address conversion post message to the stationary host. However, in FIG. 8 the gws convert the destination address of data the packet from the outdated address into the updated address assigned after the migration instead of transmitting the address conversion post message.

A packet 71, 72 in FIG. 8 are substantially same as the packet 51, 52 in FIG. 6. The operation conducted before the packet 72 is transmitted by the stationary host 12 and is received by the gateway 13 is substantially same as the first operation in FIG. 6. The operation which follows reception of the packet 72 is described hereunder with referring to FIG. 3.

The gate way 13 employs its units in FIG. 3 to process the packet 72. The communication control unit 4 receives the packet 72 and gives it to the reception packet unit 35 in the migration address unit 3. Detecting that the packet 72 is a general packet, the re-

ception packet unit 35 sends it to the address comparison unit 37. The address comparison unit 37 detects whether or not the destination address of the packet 72 coincides with the address in the data hold unit 1 at the address assigned before the migration.

When no coincidence is found, the address comparison unit 37 gives the packet 72 to the application unit 2. On the other hand, a coincidence is found, the address assigned after the migration, which corresponds with the address identical to the destination address of the packet 72, is obtained from the data hold unit 1, and is sent to the marked packet conversion unit 39 together with the packet 72. The marked packet conversion unit 39 generates a packet 72' where the destination address of the packet 72 is replaced with the address assigned after the migration, which is sent by the address comparison unit 37, the destination address of the packet 72 is added as additional address, and a mark is set to indicate that the destination address has converted. Then the packet 72' is sent to the communication control unit 4. FIG. 12 (e) shows a format of the packet 72', where identical numerals denotes the same units in FIG. 11. The packet 72' is sent to the mobile host 11 without fail since its destination address is the updated address thereof.

(operation example in FIG. 9)

In FIG. 9, the mobile host migrates across network 1, 2, 3, and 4. In FIG. 7 the gw 1-gw 4 are employed as the migration communication control devices; whereas in FIG. 9 the gw 1-gw 4 are employed simply as gateways to connect networks, and also another migration communication control device is attached to each network. The operation of the migration communication control device, which is connected to the network alone, at processing the migration post message or the address conversion post message is substantially same as one of the gw 1-gw 4 in FIG. 7. The flow of the migration post message and the address migration post message are mainly described hereunder.

(migration from network 1 to network 2)

When migrating from the network 1 to the network 2, the mobile host 11 sends a packet comprising the migration post message to the migration communication control device, which is attached to the network 1. In FIG. 9 (a) a migration post packet 81 is transmitted to a migration communication control device S1, which is attached to the network 1. The destination address of the packet 81 can be the broadcast address of the network 1.

The device S1 processes the packet 81 by employing its devices in FIG. 3. Receiving the packet 81, the device S1 stores into the data hold unit 1 the cor-

respondence of the addresses each assigned before and after the migration as well as the address assigned before the last migration. The migration post information unit 36 transmits the packet 81 to the address conversion post transmission unit 38; however, since the unit 38 detects that the address assigned before the last migration is 0, it does not transmit the address conversion post message to any network. The content of the data hold unit 1 in the S1-S4 are shown in FIG. 14 (a).

(migration from network 2 to network 3)

When migrating from the network 2 to the network 3, the mobile host 11 notifies the S2, which is attached to the network 2, that it has migrated to the network 3 by transmitting thereto the packet comprising the migration post message, which is referred to as a packet 82 in FIG. 9 (b).

The S2 employs its devices in FIG. 3 to process the packet 82. That is, it converts the content of the data hold unit 1 by renewing and adding new information, and finally holds in the unit 1 the address m'-m" at the correspondence of the addresses each of which assigned before and after the migration as well as the address m assigned before the last migration. Then, the migration post information unit 36 gives the newly obtained correspondence m'-m" to the address conversion post transmission unit 38.

The address conversion post transmission unit 38 detects whether or not the address assigned before the last migration, which is held in the data hold unit 1, is 0. If the address is not 0, the unit 38 transmits the address conversion post message to the broadcast address of the network which includes the detected address. In FIG. 9 (b) the address m is held at the address assigned before the last migration, so that the unit 38 transmits the packet 83 to the broadcast address of the network 1.

When receiving the packet 83, the migration communication control device S1, which is attached to the network 1, renews the content of the data hold unit 1 by newly holding the address correspondence m-m" as well as the address 0 at the address assigned before the last migration. Detecting 0 at the address assigned before the last migration, the address conversion post transmission unit 38 does not transmit the address conversion post to any network. The content of the data hold unit 1 in the S1-S4 are shown in FIG. 14 (b).

(migration from network 3 to network 4)

When migrating from the network 3 to the network 4, the mobile host 11 notifies the communication migration control device S3, which is attached to the network 3, that it has migrated to the network 4 by transmitting thereto a packet comprising the mi-

gration post message, referred to as a packet 84 in FIG. 9 (c).

The migration communication control device S3 employs its devices in FIG. 3 to process the packet 84. That is, it newly holds into the data hold unit 1 the address correspondence  $m''-m'''$  as well as the address  $m'$  assigned before the last migration. Then, the address conversion post transmission unit 38 in the S3 transmits a packet comprising the address conversion post message, referred to a packet 85 in FIG. 9 (c), to the broadcast address of the network 2 since the address  $m'$  is held at the address assigned before the last migration in the data host unit 1.

When receiving the packet 85, the migration communication control device S2 employs its devices in FIG. 3 to process it. That is, it newly holds into the data hold unit 1 the address correspondence  $m'-m''$  as well as the address  $m$  assigned before the last migration. Then, the address conversion post transmission unit 38 in the S2 transmits a packet comprising the address conversion post message, referred to a packet 86 in FIG. 9 (c), to the broadcast address of the network 2 since the address  $m$  is held at the address assigned before the last migration in the data hold unit 1.

When receiving the packet 86, the migration communication control device S1 employs its devices in FIG. 3 to process it. That is, it newly holds into the data hold unit 1 the address correspondence  $m-m''$  as well as the address 0 at the address assigned before the last migration. The address conversion post transmission unit 38 in the S1 does transmit the address conversion post since 0 is detected at the address assigned before the last migration. The content of the data hold unit 1 in each of the S1-S4 are shown in FIG. 14 (c). Thus, according to the migration communication control device S1-S4 of the present invention, the S1-S4 are notified of the updated address of the mobile host at every migration, so that the packet transmitted to any of the addresses  $m, m', m''$  is transferred thereby to the updated address of the mobile host. The S1-S4 also notify the stationary host of the updated address of the mobile host.

The operation in FIG. 9 differs from the operation in FIG. 7 in that each network has just one communication migration control device (one of the S1-S4), so that the migration post and the address conversion transmitted to S1-S4 are not duplicated.

In the format shown in FIG. 11 and 12, the mark 96 or the message type 93 indicates kind of packet. That is, mark 96 indicates whether or not the packet is marked while the message type 93 indicates whether it is the packet comprising the migration post message, the packet comprising the address conversion post message, and the general packet. Further, a protocol type can also be employed to indicate which migration communication control device is employed. For example, when TCP/IP is employed, the

protocol number at the IP header thereof distinguishes the packet employed in the embodiment from other packets. That is, when the protocol number in the packet is identical with the one, which has been assigned to the protocol number field, the packet is the one employed in the embodiment.

In the first embodiment of the present invention, a nonvolatile storage can be employed as the data hold unit 1 of the mobile host. If so, the communication can be resumed even after the host or the gateway is turned off as well as after the system is reset.

Also even when the stationary host employs the nonvolatile storage as the data hold unit 1, it can resume the communication, which has interrupted by the switch off or the system reset, rather fast since it obtains from another host the updated address of the mobile host instead of receiving from the gateway the address conversion post message which shows the updated address.

For example, it is supposed in FIG. 7 that the mobile host 11 migrates from the network 1 to the network 4. The data hold unit 1 of the migration communication device holds the address correspondence  $m-m''$  since it has communicated with the mobile host, which is attached to the network 4, at least once. According to the migration communication control device in the embodiment described the above, the packet is transferred from the outdated address to the updated address of the mobile host and the stationary host is notified of the updated address; therefore, even when the address information in the data hold unit is lost by switch off thereof, the stationary host will obtain the updated address. Restart of the communication can also be implemented by employing a specific host such as a server. That is, the server may be constructed to obtain the updated address of the mobile host at every migration, and give it to the stationary host whenever requested. In this case a packet comprising the address inquiry should be generated beforehand.

Also in the fifth operation in FIG. 6, the mobile host 11 employs the application unit 2 and sends to the marked packet conversion unit 21 the address assigned before the migration when transmitting the non-response address to the stationary host after it has migrated to another network. Instead of sending the non-response address, the application unit 2 can transmit a connection identifier to the marked packet conversion unit 21. In this case the data hold unit of the migration communication control device, employed as the mobile host, holds a correspondence between the connection identifier and the address that had been assigned when the connection was established instead of holding the correspondence between the correspondence of the addresses each assigned before and after the migration. Then, the unit 21 obtains the source address of the packet by detecting the address which corresponds to the identi-

fier, which is held in the data hold unit 1.

As is described the above, the mobile host can employ the broadcast address of the network when transmitting the migration post to the migration communication control devices. When the broadcast address is employed, every host attached to the network, to which the migration communication control device is also attached, obtains the updated address of the mobile host. This implements a direct communication between the mobile host and the stationary host, which improves efficiency of the communication.

The address assigned before the last migration, which is held in the hold unit 1, can be replaced with the broadcast address assigned to the network to which the mobile host is attached before the last migration. If the broadcast address is employed, the gateway employed as the migration communication control device (gws) or the migration communication control device (Ss) needs to include the broadcast address in the address conversion post message. In this case both devices can obtain the broadcast address from the data hold unit; therefore, the operation thereof at requesting the broadcast address will be eliminated.

When storage capacity of the data hold unit 1 is limited, the data hold unit 1 holds only the useful data by disposing the useless data, which is least recently retrieved therefrom by the address comparison unit.

#### [Embodiment 2]

In FIG. 15 network A, B, and C are connected in a line via gateways 143 and 143', the gateway 143 placing between the network A and B while the gateway 143' placing between the network B and C.

A home migration communication control device 101 including a migration address unit 144 is attached to the network A; a visitor migration communication control device 109 including a migration address unit 145 is attached to the network B; and a visitor migration communication control device 109' including a migration address unit 145' is attached to the network C. A mobile host 146 including a migration address unit 115 is attached to the network A as its home network, and a stationary host 151 including a migration address unit 125 is also attached to the network A.

The mobile host 146 migrates across the network A, B, and C. It has a home address  $\alpha$  assigned when it is attached to the network A, as well as other addresses assigned depending on where it migrates, such as a temporary address  $\beta$  on the network B and a temporary address  $\gamma$  on the network C.

Also each of the home migration communication control device 101, the visitor migration communication control device 109, 109' which are identical in its construction and the stationary host 151 has an address Ha, Va, Va', and Sa respectively assigned on

the network.

Detailed function of the above devices 101, 109, 109', 146, and 151 is described hereunder, in which like components are labeled with like reference numerals.

[home migration communication control device 101]

When the mobile host 146 migrates from the home network to another network, it is assigned the temporary address. However if the stationary host 151 is not notified of that migration, it transmits an original data packet (hereinafter referred to as a non-capsulated data packet) to the home address  $\alpha$  of the mobile host 146. When the noncapsulated data packet is destined for the outdated address of the home mobile host 146, the home migration communication control device 101 transfers that noncapsulated data packet from there to the updated address, that is the temporary address  $\beta$  or  $\gamma$  of the mobile host. Then, the device 101 posts to the stationary host 151 the temporary address  $\beta$  or  $\gamma$  here, so that the stationary host 151 will be able to communicate directly with the mobile host. The device 101 also posts the same information to the visitor migration communication control device 109, 109', so that the devices 109, 109' will implement the same function with the home migration communication control device 101.

As shown in FIG. 16 the home migration communication control device 101 consists of the migration address unit 144 and a communication control unit 108. The migration address unit 144 further comprises a home mobile host (MH) list hold unit 102, a packet transfer unit 103, a mobile host (MH) transfer unit 104, an address inquiry unit 105, a packet monitoring unit 106, an address post unit 107.

Next the function of each component integrating the device 101 will be described. The communication control unit 108 mainly controls the communication of protocols located in lower layers including a physical layer, such as the protocol lower than IP.

The address post unit 107 receives from the mobile host 146 an data packet including an address post message. The address post message is generated when the mobile host 146 migrates to the network B or C, and posts the temporary address  $\beta$  or  $\gamma$  of the mobile host to the device 101. The unit 107 sends the address post message to the mobile host transfer unit 104 as well as sends a response message to the mobile host 146. FIG. 28 (3) is an example of the address post message, which includes the home address  $\alpha$  as well as the temporary address  $\beta$  or  $\gamma$  of the mobile host 146, a value of an autonomous flag F, and a broadcast address Bba, Cba on the network B, C. The autonomous flag F will be described later. FIG. 28 (4) is an example of the response message.

A mobile host transfer unit 104 stores the address post message into the home mobile host list hold unit

102, notifies the visitor migration communication control device 109 or 109' of the migration of the mobile host 146 by sending thereto a mobile host transfer message, and receives the data packet including the response. Further, according to a direction given by the packet transfer unit 103, the unit 104 transmits the mobile host transfer message both to the stationary host 151 and the device 109 or 109'. The unit 103 gives the direction when the value of the autonomous flag F is 1.

FIG. 32 (3) and FIG. 36 (5) are examples of the mobile host transfer message including the home address  $\alpha$ , the temporary address  $\beta$  or  $\gamma$ , and the autonomous flag F. Since the mobile host transfer message is sent to the stationary host 151 is sent only when the autonomous flag F is 1; therefore, it does not necessarily include the value of the flag F. However, the identical message is sent both to the stationary host 151 and the visitor migration communication control device 109, 109' in this embodiment to simplify the construction of the mobile host transfer unit 104. FIG 32 (4) is an example of the response message.

As shown in FIG. 17, the home mobile host list hold unit 102 holds the home address  $\alpha$ , the temporary address  $\beta$ ,  $\gamma$ , the value of the autonomous flag F, and the broadcast address Bba, Cba on the network B, C, all of which are obtained from the mobile host transfer unit 104.

The packet monitoring unit 106 receives the packet destined for the home address  $\alpha$  of the mobile host 146, then sends it to the packet transfer unit 103 when the stationary host 151 transmits the packet to the home address  $\alpha$  of the mobile host 146 after the mobile host 146 has migrated to another network.

The packet transfer unit 103 has a payload including the noncapsulated data packet and the packet transfer message informing the transfer of the noncapsulated data packet, generates another data packet, and sends it to the temporary address  $\beta$ ,  $\gamma$  of the mobile host 146. FIG. 32 (2) is an example of the packet transfer message. As is described the above, the packet transfer unit 103 directs the mobile host transfer unit 104 to transmit the mobile host transfer message to the stationary host 151 only when the autonomous flag in the home mobile host list hold unit 102 shows the value of 1. The operation conducted when the flag F is 1 will be described later.

When the stationary host 151 has problems in communicating with the mobile host 146 such as receiving the unusual mobile host transfer message, the address inquiry unit 105 is employed to solve the problems. That is, receiving from the stationary host 151 an address inquiry message, the address inquiry unit 105 transmits to the stationary host 151 a data packet which responds to the address inquiry by showing the address to be used in the communication. The address inquiry message includes a type field 132, a flag field 133, a sequence field 134, and

a home address field 138, each of which having value 5, 1, a certain number, and  $\alpha$  respectively; while the response message includes a temporary address field 139 filled with the temporary address  $\beta$ ,  $\gamma$  as well as the flag field with 2, besides the type field 132, the sequence field 134, and the home address field 138 filled with the same values in the address inquiry message.

[visitor migration communication control device 109]

The visitor migration communication control device 109 implements the same function with the home migration communication control device 101. That is, when the stationary host 151 transmits an encapsulated data packet to the temporary address  $\beta$  of the mobile host 146, which is the updated address thereof since the mobile host has migrated to the network C, the visitor migration communication control device 109 transfers that encapsulated data packet from the temporary address  $\beta$  to temporary address  $\gamma$ . Then, the device 109 posts to the stationary host 151 the temporary address  $\gamma$ , so that the stationary host 151 will be able to communicate directly with the mobile host 146. However, whether or not the device 109 provides the above packet transfer service will be determined in accordance with a processing load put on the device 109 or with a initial setting given by a system operator; thus, the packet transfer service of the device 109 is not necessarily an obligation.

As shown in FIG. 18, the visitor migration communication control device 109 consists of the migration address unit 145 and the communication control unit 108. The migration address unit 145 further comprises the packet monitoring unit 106, a visitor mobile host list hold unit 110, a packet transfer unit 111, a mobile host transfer unit 112, a mobile host visit unit 113, and an autonomous support unit 114. The unit 106 and the unit 108 function the same as those in the home migration communication control device 101.

Receiving an autonomous packet transfer support check message inquiring if the visitor migration communication control device 109 provides the packet transfer service, the autonomous support unit 114 responds to it with the response message where the autonomous flag F shows 1 when the device 109 provides that service or 0 when it does not provide that service. FIG. 28 (1) is an example of the autonomous packet transfer support check message, while FIG. 28 (2) is an example of the response message including the autonomous flag F and the broadcast address Bba.

Receiving from the mobile host 146 the mobile host visit message which informs that the mobile host 146 has migrated to the network B, the mobile host unit 113 responds it with the response message after storing the mobile host visit message into the visitor mobile host list hold unit 110. The mobile host visit



message includes the home address  $\alpha$  and the temporary address  $\beta$  of the mobile host 146. FIG. 28 (5) is the format of the mobile host visit message, while the FIG. 28 (6) is the format of the response message.

Receiving from the mobile host transfer unit 104 in the device 101 the mobile transfer message informing that the mobile host 146 has migrated to the network C, the mobile host transfer unit 112 stores in the visitor mobile host list hold unit 110 the updated temporary address  $\gamma$  of the mobile host 146 and the value of the autonomous flag F by corresponding them to the home address  $\alpha$ . The unit 112 also transmits to the stationary host 151 the mobile host transfer message in accordance with the direction from the packet transfer unit 111, as does the mobile host transfer unit 104 in the device 101.

As shown in FIG. 19, the visitor mobile host list hold unit 110 holds the home address  $\alpha$  and the temporary address  $\beta$  on the network B, which are obtained from the mobile host 146 via the mobile host visit unit 113, as well as the temporary address  $\gamma$  and value on the autonomous flag F, which are obtained from the home migration communication control device 101 via the mobile host transfer unit 112.

The packet transfer unit 111, as does the packet transfer unit 103 in the home migration communication control device 101, transmits to the temporary address  $\gamma$  the data packet including the transfer message as well as orders the mobile host transfer unit 112 to transmit the mobile host transfer message.

[mobile host 146]

As shown in FIG. 20, the mobile host 146 includes the migration address unit 115, an address obtainment unit 116, the communication control unit 108, and an application processing unit 124 which mainly controls the communication of protocols located in higher layers including an application layer, such as TCP or layers located higher than it.

The migration address unit 115 comprises the a packet transmission unit 117, a transfer packet reception unit 118, an address hold unit 119, a migration unit 120, an autonomous support unit 121, an address post unit 122, a mobile host visit unit 123.

The migration address unit 115 comprising the above units is employed in transfer of data to the temporary address  $\beta$  or  $\gamma$  when the mobile host 146 migrates to the network B or C. Also receiving the data packet destined for the temporary address  $\beta$  or  $\gamma$  including the packet transfer message and the noncapsulated data packet, the device 115 transmits the noncapsulated data to the application processing unit 124.

In accordance with the order given by the application processing unit 124 when the mobile host migrates to the network B, C, the migration unit 120 con-

trols the address obtainment unit 116, the autonomous support unit 121, the address post unit 122, the mobile host visit unit 123, and the address hold unit 119.

5 Directed by the migration processing unit 120, the address obtainment unit 116 obtains the temporary address  $\beta, \gamma$  of the mobile host 146 assigned when it migrates to the network B, C respectively. BOOTP in "Bill Croft and John Gilmore, BOOTSTRAP PRO-  
10 Tocol RFC951, Sep., 1985" is an example of obtaining the temporary address; besides employing the BOOTP, the operator may input the temporary address  $\beta, \gamma$  assigned by a system administrator of the network B, C.

15 Directed by the migration unit 120, the autonomous support unit 121 sends the autonomous packet transfer support check message to inquire if the visitor migration communication control device 109, 109' attached to the network B, C provides the packet transfer service and receives the response message to the inquiry. The autonomous packet transfer support check message is also sent to obtain the broadcast address Bba and Cba on the network B and C respectively.

25 Directed by the migration unit 120, the address post unit 122 sends the address post message to notify the home migration communication control device 101 of the temporary address  $\beta, \gamma$ . The address post message also informs whether or not the device 109, 109' provides the packet transfer service as well as the broadcast address Bba, Cba on the network B, C. If the response message from the visitor migration communication control device 109, 109' has the value 1 of the autonomous flag F, the mobile host visit unit 123 transmits to the visitor migration communication control device 109, 109' the mobile host visit message including the home address  $\alpha$  as well as the temporary address  $\beta, \gamma$  respectively.

40 As shown in FIG. 21, the address hold unit 119 previously holds the home address  $\alpha$  of the mobile host 146 and the broadcast address Aba on the network A. Now, the unit 119 newly holds the temporary address  $\beta$  or  $\gamma$  obtained from the address obtainment unit 116 via the migration unit 120 and the broadcast address Bba or Cba obtained from the autonomous support unit 121 via the migration unit 120.

45 When the mobile host 146 is attached to the network A and receiving a data packet destined for the home address  $\alpha$ , the transfer packet reception unit 118 sends data etc. in the noncapsulated data packet to the application processing unit 124. On the other hand, when the mobile host 146 is attached to the network B and receiving a data packet destined for the temporary address  $\beta$ , the data packet including the packet transfer message and the noncapsulated data packet destined for  $\alpha$ , the unit 118 sends to the application processing unit 124 data etc. in the noncapsulated data. Thus, the application processing

unit 124 receives the data without being affected by the migration of the mobile across the networks.

Receiving the data to be transmitted and the instruction from the application processing unit 124, the packet transmission unit 117 generates a noncapsulated data packet whose destination address is the home address  $\alpha$  and transmits it.

[stationary host 151]

As shown in FIG. 22, the stationary host 151 comprises the migration address unit 125 and the application processing unit 161 which mainly controls the communication of a protocol located in higher layers including application layer, such as TCP or layers located higher than the TCP and the communication control unit 108.

The migration address unit 125 comprises a transfer packet transmission unit 126, a packet reception unit 127, an address hold unit 128, an address inquiry unit 129, and the mobile host transfer unit 130.

The migration address unit 125 comprising the above units generates a noncapsulated data packet and sends it to the home address  $\alpha$  when it is not notified that the mobile host 146 migrate to the network B or C and obtained the temporary address  $\beta$  or  $\gamma$  respectively. The unit 125 also generates an encapsulated data packet including as a payload the noncapsulated data packet and a data transfer message, which informs transfer of the noncapsulated data packet and sends it to the temporary address  $\beta$ ,  $\gamma$ , when it is notified of the migration.

Receiving from the home migration communication control device 101 and the visitor migration communication control device 109, 109' the data packet including the mobile host transfer message which informs the migration of the mobile host 146, the mobile host transfer unit 130 stores into the address hold unit 128 the home address  $\alpha$  and the temporary address  $\beta$  or  $\gamma$  of the mobile host 146 assigned on the network B or C respectively.

As shown in FIG. 23, the address hold unit 128 holds the home address  $\alpha$ , the temporary address  $\beta$  or  $\gamma$  by corresponding them.

Directed by the application unit 161, the transfer packet transmission unit 126 generates a data packet destined for the home address  $\alpha$ , and transmits it. However, if the address hold unit 128 holds the temporary address  $\beta$  or  $\gamma$  besides the home address  $\alpha$ , the unit 126 generates an encapsulated data packet destined for the temporary address  $\beta$  or  $\gamma$ , which includes as a payload a noncapsulated data packet and a packet transfer message, which informs transfer of the noncapsulated data packet, and transmits it.

As is described the above, both the home migration communication control device 101 and the visitor migration communication control device 109, 109' generate the encapsulated data packet includ-

ing the packet transfer message and the noncapsulated data and transmits it to the current temporary address of the mobile host 146. Owing to the device 101 or 109, 109', the stationary host 151 is able to transmit to the mobile host 146 both the noncapsulated data packet destined for the home address  $\alpha$  and the encapsulated data packet destined for the temporary address  $\beta$  or  $\gamma$  without failure even when the address hold unit 128 fails to hold the current temporary address  $\beta$  or  $\gamma$  and the stationary host 151 transmits the data packet to the outdated address of the mobile host 146.

The packet reception unit 127 receives a data packet which is sent from the mobile host 146 and has Sa as its destination address, and sends the data etc. in it to the application unit 161.

When the address inquiry unit 129 has problems such as that it received an illegal mobile host transfer message or that it cannot communicate with the mobile host 146 successfully, it transmits a data packet including an address inquiry message in order to inquire of the host migration communication control device 101 the address which is currently used to communicate with the mobile host 146.

[construction of data packet]

As shown in FIG. 24 (a), (b), (c), there are three kinds of data packets, each data packet 210, 220, 230, includes each of header 211, 221, 231 and payload 212, 222, 232 respectively.

The header 211 of the data packet 210 includes a destination address 201, and a source address 202. Also the payload 212 consists of a transmission data 203.

The header 221 of the data packet 220 includes the destination address 201 and the source address 202. Also the payload 222 consists of a message 204.

The header 231 of the data packet 230 includes the destination address 201 and the source address 202. Also the payload 232 consists of the message 204, which is employed as the packet transfer message, and a noncapsulated data packet 210. Also each header 211, 221, 231 includes information showing presence or absence of the message 204 as a protocol number etc.

The message 204 includes some of the fields in FIG. 25 in accordance with its type.

The type of the message 204 is indicated in the message type field 132. Besides the above types, the message 204 is also employed as an echo message for examining whether or not a host employs an appropriate operation in accordance with the message.

A flag field 133 indicates whether or not the message 204 is a response. When the message 204 is not the response, the field 133 further indicates whether or not the message 204 requests a response.

A sequence field 134 gives a single number both to the request message and its response message, thereby the request message and the response message are corresponded.

An autonomous flag field 135 contains a value of the autonomous flag F indicating whether or not the visitor migration communication control device 109,109' provide the packet transfer service.

A counter field 136 contains a counter indicating the number of the visitor migration communication control devices employed to transfer the encapsulated data packet consisting of the packet transfer message and the noncapsulated data packet. The visitor migration communication control device increments the counter in the received message packet by 1, and gives it to the message to be transmitted. When the incremented number is greater than the predetermined number, the received message packet is disposed.

A status field 137 of the response message indicates presence or absence of an error in a transmission/reception of the data packet. For example, it indicates an error in authentication information, which will be described later, or the address inquiry message which cannot or should not be responded.

A home address field 138, a temporary address field 139, and a broadcast address field 140 indicates the home address as well as the temporary address of the mobile host 146 or the broadcast address on its home network or on the network it migrates. However, what the broadcast address field 140 indicates depends on type of the message 204. Whether the message 204 is the request or the response also decides the content of the broadcast address field 140.

The authentication information field 141 indicates if a source address coincides with the sender's address.

[outline of communication operation]

The home migration communication control device 101 and the visitor migration communication control device 109,109' is basically employed to transfer the data packet transmitted by the stationary host 151 as well as post to the stationary host 151 the updated temporary address of the mobile host 146. Understanding of such operations will be helped by the following two points.

1. Transfer of the data packet and posting of the updated temporary address are conducted only when the mobile host 146 migrates from its home network to another network. The home network refers to the one to which the home migration communication control device is attached.
2. Posting of the updated temporary address is conducted only when the autonomous flag F is 1, which indicates the visitor migration communication control device 109, attached to the same net-

work as is the mobile host 146, provides the packet transfer service. Otherwise, the data packet transmitted by the stationary host 151 to the posted temporary address will not be received by the mobile host 146 when the mobile host 146 migrates to another network.

[communication operation 1]

An example of the communication operation is described hereunder. In the communication operation 1 the visitor migration communication control device 109,109' provides the packet transfer service when the mobile host 146 migrates from the network A to the network B, further from the network B to the network C.

[migration from network A to network B]

The operation at the migration of the mobile host 146 from the network A to the network B is described with referring to FIGs. 26-29. FIG. 26 shows a flow of the data packet transmitted between the devices; FIG. 27 shows a communication sequence of the data packet; FIG. 28 shows construction of each data packet; and FIG. 29 shows the content of the address hold unit 119 etc.

When the mobile host 146 is attached to the network A, the home mobile host list hold unit 102 in the home migration communication control device 101 holds the home address  $\alpha$  both as the home address and the temporary address of the mobile host 146. Thereby the home migration communication control device 101 detects that the mobile host 146 is attached to the network A.

The address hold unit 119 in the mobile host 146 holds the home address  $\alpha$  and the broadcast address Aba on the network A.

When the mobile host 146 migrates to the network B, the application unit 124 orders the operation of the migration unit 120 in accordance with the instruction given by the operator. The temporary address  $\beta$  is assigned to the mobile host 146 on the network B, and the address obtainment unit 116 obtains it. The migration unit 120 stores into the address hold unit 119 the temporary address  $\beta$  together with the home address  $\alpha$  and the broadcast address Aba.

(1) The autonomous support unit 121 transmits to the visitor migration communication control device 109, which is attached to the network B, the data packet including the autonomous packet transfer support check message 147 which holds the home address  $\alpha$  and the temporary address  $\beta$ . The destination address of the data packet is the broadcast address shared by every network, such as an address where every bit is 1. The message 147 does not necessarily hold the home address  $\alpha$  and the temporary address  $\beta$  although

they can be used in checking the security of the network if it does. Also the message 147 holding the home address  $\alpha$  and the temporary address  $\gamma$  can take the place of a mobile host visit message 146, which will be described later.

(2) The autonomous support unit 114 in the visitor migration communication control device 109 responds to the autonomous support unit 121 with the response message 147R where broadcast address Bba is set and the autonomous flag F in the autonomous flag field 135 indicates 1 to inform that the device 109 provides the packet transfer service.

The mobile host 146 transmits the data packet to the visitor migration communication control device 109. The broadcast address Bba is employed as the destination address of the data packet and it is set in the response message 147R; however, this is not an obligation.

That is, when the response message 147R does not hold the broadcast address Bba, the following means can be employed. First, the broadcast address shared by every network can be employed, which is described in the above. Second, the source address, which is set in the header of the data packet comprising the response message 147R, can be employed. Third, a so called name service can be employed, where a server device on the network system informs the broadcast address Bba. Finally, when the address assigned to each of the devices, which are attached to the network, consists of the network address being unique for the network and a device address being unique for the devices, and the broadcast address on each network consists of such network address and the device address where the value of every bit is 1, the network address Bba can be generated by employing the network address included in the temporary address  $\beta$  of the mobile host 146.

(3) The address post unit 122 transmits to the home migration communication control device 101 the address post message 148. The message 148 includes the value 1 of the autonomous flag F, which is obtained from the response message, home address  $\alpha$ , the temporary address  $\beta$  on the network B, and the broadcast address Bba, and the broadcast address Aba is the destination address of the address post message 148.

When the address post unit 107 in the home migration communication control device 101 receives the address post message 148, the mobile host transfer unit 104 stores in the home mobile host list hold unit 102 the temporary address  $\beta$ , the value 1 of the autonomous flag 1, and the broadcast address Bba by corresponding them to the home address  $\alpha$ . Since the home address  $\alpha$

had been stored as the temporary address before the temporary address  $\beta$  was stored, the mobile host transfer unit 104 knows that the mobile host 146 has migrated from the network A to the network B; therefore, it does not transmit the mobile host transfer message to the visitor migration communication control device 109,109'. That is, the data packet transmitted by the stationary host 151 to the home address  $\alpha$  of the mobile host 146 is received by the home migration communication control device 101 and transferred thereby to the temporary address  $\beta$ ; therefore, the visitor migration communication control device 109,109' is not employed here.

(4) The address post unit 107 notifies the address post unit 122 that it has received the address post message 148 by sending the response message 148R.

(5) Since the visitor migration communication control device 109 provides the packet transfer service, the mobile host visit unit 123 transmits to the visitor migration communication control device 109 the mobile host visit message 149 including the home address  $\alpha$  and the temporary address  $\beta$ , so that the device 109 is notified that the mobile host 146 has migrated to the network B. The mobile host visit message 149 is destined for the broadcast address Bba.

The mobile host visit unit 113 in the visitor migration communication control device 109 receives the mobile host visit message 149 and stores into the visitor mobile host list hold unit 110 the home address  $\alpha$  as well as the temporary address  $\beta$ . The temporary address  $\beta$  is stored also as the updated temporary address of the mobile host 146, which will be assigned when the mobile host 146 migrates from the network B to another network; thereby, the visitor migration communication control device 109 detects that the mobile host is currently attached to the network B.

(6) The mobile host visit unit 113 notifies the mobile host visit unit 123 by sending the response message 149R that it has received the mobile host visit message 149.

[communication between the stationary host 151 and the mobile host 146 on the network B]

The operation at the communication between the stationary host 151 and the mobile host 146 when the mobile host is attached to the network B is described hereunder with referring to FIGs. 30-33, which are relevant for FIGs.26-29.

(1) The application unit 161 in the stationary host 151 directs the transmission of the noncapsulated data packet, whose destination is the home address  $\alpha$ , despite the migration of the mobile host 146. Immediately after the mobile host 146

migrates to the network B, that is, when the address hold unit 128 does not hold the home address  $\alpha$  and the temporary address  $\beta$ , the transfer packet transmission unit 126 is not notified of the migration; therefore, it generates the noncapsulated data packet 152 and transmits it to the home address  $\alpha$  in accordance with the direction from the application unit 151.

The noncapsulated data packet 152 is not received by the mobile host 146, which is not attached to the network A, but by the packet monitoring unit 106 in the home migration communication control device 101 since the home mobile host list hold unit 102 in the device 101 holds the home address  $\alpha$  as well as the temporary address  $\beta$ , which coincides with the destination address of the noncapsulated data packet 152.

(2) The packet transfer unit 103 in the home migration communication control device 101 generates an encapsulated data packet including the noncapsulated data packet 152, which is received by the packet monitoring unit 106, and the packet transfer message 153, which informs the transfer of the noncapsulated data packet 152; and transmits it to the temporary address  $\beta$ . The packet transfer message 153 includes the value 0 in the field 133, which indicates that no response is requested, as well as the value 0 on the counter in the field 136, which indicates that the packet transfer message is the first message added to the noncapsulated data packet 152. As is described, no response is requested by the packet transfer message 153. That is, the application unit 161 of the stationary host 151 and the application unit of the mobile host 146, rather than the home migration communication control device 101 and the migration address unit 115, confirm that the mobile host 146 receives the noncapsulated data packet 152.

The transfer packet reception unit 118 in the mobile host 146 receives the encapsulated data packet including the packet transfer message 153 and the noncapsulated data packet 152, since it is destined for the temporary address  $\beta$ , which is held in the address hold unit 119. The unit 118 then detects that the destination address of the noncapsulated data packet 152 is the home address  $\alpha$ , and sends the data etc. in the noncapsulated data packet 152 to the application unit 124.

Thus, the communication between the application unit 124 and the application unit 161 is not affected by the migration of the mobile host 146. (3) The packet transfer unit 103 transmits the encapsulated data packet including the data packet transfer message. It also directs, after detecting that the autonomous flag F indicates 1, the mobile host transfer unit 104 to transmit to the sta-

tionary host 151 the data packet including the mobile host transfer message 154 where the home address  $\alpha$  and the temporary address  $\beta$  are set. Finally, the unit 104 transmits the data packet to the stationary host 151.

The mobile host transfer unit 130 in the stationary host 151 receives the mobile host transfer message and stores into the address hold unit 128 the home address  $\alpha$  and the temporary address  $\beta$ .

(4) The mobile host transfer unit 130 responds to the mobile host transfer unit 104 with the response message 154R.

(5) When the application unit 161 directs the transmission of the noncapsulated data packet to the home address  $\alpha$  after the address hold unit 128 holds the home address  $\alpha$  and the temporary address  $\beta$ , the transfer packet transmission unit 126 first generates a noncapsulated data packet destined for the home address  $\alpha$ , then generates an encapsulated data packet including it and a packet transfer message 155. The encapsulated data packet is then transmitted to the temporary address  $\beta$ . Thus, once the home migration communication control device 101 notifies the stationary host 151 of the home address  $\alpha$  and the temporary address  $\beta$ , the stationary host 151 is able to transmit the data packet to the temporary address  $\beta$  of the mobile host 146, and the home migration communication control device 101 is not employed.

On the other hand, when data is transmitted from the mobile host 146 to the stationary host 151, the  $S_a$  is employed as the destination address  $\alpha$  and the home address is employed as the source address; and the noncapsulated data packet is transmitted from the address  $\alpha$  to the address  $S_a$ .

Thus, even when all the noncapsulated data transmitted by the stationary host 151 is destined for the home address  $\alpha$ , the home migration communication device 101 transfers the data to the updated temporary address of the mobile host; thereby, the communication between the mobile host 146 and the stationary host 151 is implemented, and the conventional device can be employed as the stationary host 151, which broadens a practicability of the network system.

Whereas, when the network system checks the original source address of the data packet or a transfer path of the data packet, the transmission unit may be built in the mobile host 146 like the transfer packet transmission unit 126 in the stationary host 151, and also the reception unit may be built in the stationary host 151 like the transfer packet reception unit 118 in the mobile host 146; and the encapsulated data packet including the packet transfer message and the noncapsulated data packet may be transmitted therebetween.

[migration from network B to network C]

The operation at the migration of the mobile host 146 from the network B to the network C is described hereunder with referring to FIGs. 34-37, relevant for FIGs. 26-29.

(1)-(4) The operation related to transmission of an autonomous packet transfer support check message 147', a response message 147R', an address post message 148', and a response message 148' between the mobile host 146 and the visitor migration communication control device 109' is substantially same as the operation related to transmission of messages between the mobile host 146 and the visitor migration communication control device 109, which is conducted when the mobile host 146 migrates to the network B. However, the operation at the migration from the network A to the network B and the operation at the migration from the network B and the network C are different from each other in part of the operation of the home migration communication control device 101 conducted after it responds to the received address post message 148' with the response message 148R.

(5) When the address post unit 107 receives the address post message 148', the mobile host transfer unit 104 in the home migration communication control device 101 detects that the mobile host been attached to the network B before migrating to the network C since the temporary address  $\beta$  has been stored as the temporary address. Then, the mobile host transfer unit 104 sends to the visitor migration communication control device 109 the data packet including both the home address  $\alpha$  and the temporary address  $\gamma$ , so that the device 109 transfers the data packet transmitted by the stationary host 151 from the temporary address  $\beta$  to the temporary address  $\gamma$ . The data packet received by the visitor migration communication control device is destined for the broadcast address Bba.

In accordance with the address post message 148', the mobile host transfer unit 104 stores into the home move host list hold unit 102 the temporary address  $\gamma$ , the value 1 of the autonomous flag F, and the broadcast address Cba by corresponding them to the home address  $\alpha$ .

Receiving the data packet including the mobile host transfer message 150, the mobile host transfer unit 112 in the visitor migration communication control device 109 stores into the visitor mobile host list hold unit 110 the temporary address  $\gamma$  newly assigned to the mobile host 146 and the value 1 of the autonomous flag F by corresponding them to the home address  $\alpha$ .

(6) The mobile host transfer unit 112 notifies the mobile host transfer unit 104 that it has received

the mobile host transfer message 150 by sending thereto the response message 150R.

(7), (8) The transmission of a mobile host visit message 149' and a response message 149R' between the mobile host 146 and the visitor migration communication control device 109', which is conducted when the device 109' provides the packet transfer service, is substantially same as the transmission of messages between the mobile host 146 and the visitor migration communication control device 109, which is conducted when the mobile host 146 migrates to the network B.

[communication between mobile host 146 attached to network C and stationary host 151]

Transmission of the data packet from the stationary host 151 to the mobile host 146 when the mobile host is attached to the network C is described with referring to FIG. 38-41, which are relevant for FIG. 26-29.

The transmission is substantially same as the transmission between the stationary host 151 and the mobile host 146 when the mobile host 146 is attached to the network B, except that the visitor migration communication control device 109 instead of the home migration communication control device 101 is employed.

(1) When the stationary host 151 is not notified that the mobile host 146 has migrated from the network B to the network C, the stationary host 151 generates the encapsulated data packet including the noncapsulated data packet, which is destined for the home address  $\alpha$ , and the packet transfer message 156; then transmits it to the temporary address  $\beta$ . This is substantially the same as (5) in the communication between the stationary host 151 and the mobile host 146 attached the network B.

The data packet transmitted by the stationary host is not received by the mobile host 146 since the mobile host is not attached to the network B. The data packet is received by the packet monitoring unit 106 in the visitor migration communication control device 109 since the visitor mobile host list hold list unit thereof holds the temporary address  $\beta$  besides the temporary address  $\gamma$ .

(2) The visitor migration communication control device 109 transmits to the temporary address  $\gamma$  of the mobile host 146 the data packet including the packet transfer message 157, which is substantially same as (2) in the communication between the stationary host 151 and the mobile host 146 on the network B except a difference described hereunder.

The home mobile host migration communi-

cation control device 101 receives the noncapsulated data packet 152 and generates an encapsulated data packet comprising the received noncapsulated data packet 152 and the packet transfer message 153. On the other hand, the visitor migration communication control device 109 receives the encapsulated data packet comprising the packet transfer message 156 and the packet transfer unit 111 converts the data packet by changing the destination address from the temporary address  $\beta$  into the temporary address  $\gamma$  as well as converting the packet transfer message 156 into the packet transfer message 157, whose value on the counter is incremented by 1.

(3)-(5) The visitor migration communication control device 109, the stationary host 151, and the mobile host 146 on the network C operate substantially same as the home migration communication control device 101, the stationary host 151, and the mobile host 146 on the network B, which is described the above in (3)-(5); thereby the mobile host transfer message 158 and the response message 158R are transmitted, and the data packet including the packet transfer message 160 is transmitted by the stationary host 151 to the mobile host 146 attached to the network C.

If the stationary host 151 does not transmit any data packet to the mobile host 146, which is attached to the network B, the stationary host is not notified of either the temporary address  $\beta$  or the temporary address  $\gamma$ ; therefore, the stationary host 151 transmits the data packet to the home address  $\alpha$  even when the mobile host 146 has migrated from the network B to the network C. When this occurs, the home migration communication control device 101, as does the visitor migration communication device 109, transfers the data packet from the home address  $\alpha$  to the temporary address  $\gamma$ ; then notifies the stationary host 151 of the updated temporary address  $\gamma$  of the mobile host 146 so that the stationary host 151 will be able to directly transmit the data packet, which comprises the packet transfer message, to the mobile host 146 attached to the network C.

Further, when the mobile host 146 migrates to the network, to which the visitor migration communication control device is attached to provide the packet transfer service, the stationary host 151 may transmit the data packet destined for any of the addresses  $\alpha$ ,  $\beta$ , or  $\gamma$ . When the data packet is transmitted to the home address  $\alpha$  or the temporary address  $\gamma$ , the home migration communication control device 101 or the visitor migration communication control device 109, which is notified of the updated temporary address of the mobile host 146, transfers the data packet to the updated temporary address; then it notifies the stationary host 151 of the updated temporary address of the mobile host.

When the data packet is transmitted to the temporary address  $\beta$  of the mobile host 146, the visitor migration communication control device 109 receives it. Since the device 109 is notified of only the temporary address  $\gamma$ , it transmits the data packet comprising the packet transfer message to the temporary address  $\gamma$  as well as transmits the mobile host transfer message to notify the stationary host 151 of the temporary address  $\gamma$ . The visitor migration communication control device 109' receives the data packet comprising the packet transfer message, which is destined for the temporary address  $\gamma$ , and transmits it to the updated temporary address of the mobile host 146; then transmits the mobile host transfer message to notify the stationary host 151 of the updated temporary address. Also the visitor migration communication control device 109' obtains the address of the visitor migration communication control device 109 from the source address of data packet transmitted thereby, and transmits the mobile host transfer message to the device 109. Thus, the visitor migration communication control device 109' obtains the updated temporary address of the mobile host 146, and transfers the data packet to the mobile host 146 as well as notifies stationary host 151 of the obtained updated temporary address.

[communication operation 2]

Another example of the communication operation is described hereunder. In the communication operation 2 the visitor migration communication control device 109 does not provide the packet transfer service when the mobile host 146 migrates from the network A to the network B, further from the network B to the network C.

As shown in FIG. 42, when the device 109 does not provide the packet transfer service, the autonomous packet transfer support check message 181, transmitted by the mobile host 146 which has migrated from the network A to the network B, is responded with the response message 181R where the autonomous flag F in the autonomous flag field 135 indicates 0. Thereby, the autonomous flag field 135 in the address post message 182, which is transmitted by the mobile host 146 to the home migration communication control device 101, obtains the value 0, and the value 0 is held in the home mobile host list hold unit 102 in the device 101. The mobile host 146 does not transmit the mobile host visit message to the visitor migration communication control device 109.

As shown in FIG. 43, receiving from the stationary host 151 the noncapsulated data packet 183, which is destined for the home address  $\alpha$ , the home migration communication control device generates the encapsulated data packet comprising the received noncapsulated data packet 183 and the packet transfer message 184, and transmits it to the tem-

porary address  $\beta$ , as is in the communication operation 1.

However, recognizing the value 0 on the autonomous flag F, which is held in the home mobile host list hold unit 102, the device 101 does not transmit to the stationary host 151 the mobile host transfer message including the temporary address  $\beta$ . Therefore, every data packet transmitted by the stationary host 151 is destined for the home address  $\alpha$ , and it is transferred to the mobile host 146 by the home migration communication control device 101. Thus, the stationary host 151 is not notified of the temporary address  $\beta$  since the data packet transmitted to the address other than the home address  $\alpha$  is not transferred by the device 109; therefore it is not received by the mobile host 146 when it departs the network B to migrate to the network C.

When the visitor migration communication control device 109', which is attached to the network, provides the packet transfer service, the home migration communication control device 101 notifies the stationary host 151 of the temporary address  $\gamma$  when it transmits the noncapsulated data to the home address  $\alpha$ , so that the stationary host 151 is able to directly transmit the data packet comprising the noncapsulated data packet and the packet transfer message to the mobile host 146 on the network C.

When the visitor migration communication control device 109 does not provide the packet transfer service, the home migration communication control device 101 does not necessarily notify the device 109 of the temporary address  $\gamma$  of the mobile host 146 assigned when it has migrated from the network B to the network C. However, the construction of the device 101 will be simplified if it conducts the same operation either or not the packet transfer service is provided since the visitor migration communication control device 109 ignores the mobile host transfer message.

Also the device 109 may respond to the autonomous packet transfer support check message 181 only when it provides the data packet transfer service; therefore, the presence or absence of the response message 181R indicates to the mobile host 146 whether or not the data packet transfer service is provided. In the above operation the value 0 of the autonomous F also indicates that the packet transfer service is not provided, whereas absence of the response message to the message 181 can indicate the absence of the packet transfer service, which will simplify construction of mobile host 146.

[communication operation 3]

The final example of the communication operation is described hereunder. In the communication operation 3 the visitor migration communication control device 109' does not provide the packet transfer service while the visitor migration communication control

device 109 does.

As shown in FIG. 44, when the packet transfer service is not provided by the visitor migration communication control device 109', the mobile host 146 transmits to the home migration communication control device 101 the address post message 182' where the value 0 is set at the autonomous flag F. Then, the home migration communication control device 101 transmits to the device 109 the mobile host transfer message 185 by setting the value 0 at the autonomous flag F.

When detecting the value 0 at the autonomous flag F, the visitor migration communication control device 109 ceases to provide the packet transfer service.

As shown in FIG. 45, even after cease of the data packet transfer service, the stationary host 151 may transmit to the temporary address the data packet comprising the noncapsulated data packet and the packet transfer message 186.

When this happens, the visitor migration communication control device 109 obtains the noncapsulated data packet 187 from the received encapsulated data packet and transmits it to its destination address, the home address  $\alpha$ . The noncapsulated data packet 187 is then received by the home migration communication control device 101, which is attached to the network A. Finally, the home migration communication control device 101 transfers the noncapsulated data packet 187 together with the packet transfer message 188 to the temporary address  $\gamma$  of mobile host 146, which is attached to the network C.

The visitor migration communication control device 109 notifies the stationary host 151 that the mobile host 146 is attached to the network A instead of the network C by sending the mobile host transfer message 189 where the home address  $\alpha$  is set in the temporary address field 139. Then, the stationary host 151 transmits the noncapsulated data packet 187 to the home address  $\alpha$ , and it is transferred by the home migration communication control device 101, which is employed to take the place of the visitor migration communication control device 109. As another option, the device 109 may send the mobile host transfer message 189 where the invalid address is set, such as the address where every bit is 1. Then, the home migration communication control device 101 may notify the stationary host 151 of the home address  $\alpha$  in accordance with the address inquiry obtained from the stationary host 151.

The operation described the above will be employed when the visitor migration communication control device 109 ceases to provide the packet transfer service operation regardless whether or not the device 109' provides the packet transfer service.

On the other hand, the visitor migration communication device 109 may restart the packet transfer service even when the device 109' ceases to provide



the service.

In this case, the home migration communication control device 101 needs to provide the visitor migration communication control device 109 with the updated temporary address at every migration of the mobile host 146 unless the mobile host migrates to the network to which another visitor migration communication control device is attached and provides the packet transfer service. To realized it, for example, when the value of the autonomous flag F in the address post message is 0 to indicate that the device 109' does not provide the packet transfer service, the broadcast address Bba as the destination address of the mobile host transfer message, which is transmitted to the device 109, will not be renewed.

Additionally, the broadcast address as the destination address of the data packet, which is transmitted by the mobile host 146, can be replaced with the address Ha, Va, Va', each of which is unique to each device. The address unique to each device will be obtained by detecting the source address of the data packet received from each device, or by employing a so called name service.

Also in the second embodiment, the home migration communication control device 101 detects whether or not the mobile host 146 is attached to the same network from what is held as the temporary address in the address hold unit; to be precise, whether or not the home address  $\alpha$  is held as the temporary address. However, this can also be detected by knowing in which table the temporary address is held. For example, when the device 101 and the mobile host 146 are attached to the same network, the first table holds the addresses, such as the home address  $\alpha$ ; whereas, the second table holds the addresses when the device 101 and the mobile host 146 are attached to the different network from each other. Value of the autonomous flag F, 0 or 1, can also be utilized in the same way.

Further, the home migration communication control device 101 and the visitor migration communication control device 109, 109' may be employed as a host such as the mobile host 146 or the stationary host 151.

Finally, the home migration communication control device 101, the visitor migration communication control device 109, the mobile host 146, and the stationary host 156 may be constructed identically and can be replaced with each other.

Although in the embodiment the application unit 124 starts its operation before being notified of updated temporary address  $\beta$ ; therefore it always transmits the data packet to the home address  $\alpha$  of the mobile host 146, it can transmit the data to the temporary address  $\beta$  if it starts its operation after obtaining the temporary address  $\beta$ .

Although the present invention has been fully described by way of examples with reference to the ac-

companying drawings, it is to be noted that various changes and modifications will be apparent to those skilled in the art. Therefore, unless otherwise such changes and modifications depart from the scope of the present invention, they should be constructed as being included therein.

## Claims

1. A migration communication control device constructed to control a communication between a mobile node and a partner node, the mobile node migrating across networks and obtaining an address assigned on each network while the partner node being a communication partner of the mobile node, comprising a first migration control unit, a second migration control unit, a third migration control unit, the second migration control unit being placed on the mobile node and the third migration control unit being placed on the partner node,

wherein the first migration control unit comprises:

packet transfer means for receiving a packet which was destined for an outdated address of the mobile node, the outdated address assigned when the mobile node migrated to a network to which the first migration control unit is attached, generating a conversion packet which holds an updated address instead of the outdated address, and transmitting the conversion packet; and

address post means for transmitting an address post message which indicates the updated address of the mobile node to the third migration control unit, the third migration control unit transmitting the packet received by the packet transfer means, and

the second migration control unit comprises:

migration post means for transmitting to the first migration control unit a migration post message which indicates the updated address of the mobile node when the mobile node migrates to another network; and

packet resumption means for receiving the conversion packet from both the first migration control unit and the third migration control unit and resuming an original packet from the conversion packet, and

the third migration control unit comprises:

packet conversion means for converting a destination address of a packet, the packet to be transmitted to the mobile node, into the updated address indicated by the address post message, the address post message sent by the first migration control unit, and transmitting it to the mobile

node.

2. The migration communication control device of Claim 1, wherein the migration post means in the second migration control unit transmits an identification key included in the migration post message, the identification key being employed to identify the mobile node. 5
3. The migration communication control device of Claim 2, wherein the identification key is an address of the mobile node assigned at one network before the network to which the mobile node is currently attached. 10
4. The migration communication control device of Claim 2, wherein the identification key is an address of the mobile node assigned before its initial migration. 15
5. The migration communication control device of Claim 1; wherein the second migration control unit is constructed to transmit to the third migration control unit the packet which has the same format as the resumed packet. 20
6. The migration communication control device of Claim 1, wherein the first migration control unit further comprises: 25
  - address hold means for holding the outdated address and the updated address by corresponding them with each other; and
  - address comparison means for comparing the destination address of the received packet with the outdated address, wherein
  - the packet transfer means generates the conversion packet and transmits it when the address comparison means detects that the destination address of the received packet coincides with the outdated address. 30
7. The migration communication control device of Claim 1, wherein the first migration control unit further comprises: 35
  - address hold means for holding the outdated address and the updated address by corresponding them with each other; and
  - address comparison means for comparing the destination address of the packet received by the packet transfer means with the outdated address, wherein
  - the address post means transmits the address post message which indicates the updated address of the mobile node to the third migration control unit, the third migration control unit transmitting the packet received by the packet transfer means, when the address comparison means detects that the destination address of the packet 40

coincides with the outdated address.

8. The migration communication control device of Claim 1, wherein the second migration control unit further comprises: 5
  - address hold means for holding the outdated address and the updated address by corresponding them with each other; and
  - address comparison means for comparing the updated address with the destination address of the packet received from one of the first migration control unit and the third migration control unit, wherein
  - the packet resumption means resumes the original packet from the conversion packet when the address comparison means detects that the updated address coincides with the destination address of the packet received from one of the first migration control unit and the third migration control unit. 10
9. The migration communication control device of Claim 1, wherein the third migration control unit further comprises: 15
  - address hold means for holding the outdated address and the updated address of the mobile node by corresponding them with each other; and
  - address comparison means for comparing the outdated address in the address hold means with the destination address of the packet to be transmitted to the mobile node, wherein
  - the packet conversion means converts the destination address of the packet to be transmitted to the mobile node into the updated address which corresponds to the outdated address in the address hold means when the address comparison means detects the outdated address in the address hold means coincides with the destination address of the packet. 20
10. The migration communication control device of Claim 1, wherein there are a plurality of the first migration control units, and the second migration control unit transmits the migration post message to at least one of the first migration control units. 25
11. The migration communication control device of Claim 10, wherein the migration post means in the second migration control unit transmits the migration post message to the first migration control unit which is attached to the network to which the mobile node was attached before its migration, 30
  - each of the first migration control units has migration post means for transmitting to one of the other first migration control units a migration post message to post the same address as the 35

30

updated address indicated by the migration post message received from the second migration control unit, and

each of the first migration control units has migration post means for transmitting a migration post message from one of the other first migration control units to another first migration control unit to post the same address as the updated address indicated by the received migration post message.

12. The migration communication control device of Claim 11, wherein each of the first migration control units and the second migration control unit further comprise pointer hold means for holding pointers related to the first migration control unit to which the migration post message is transmitted, and wherein

the migration post means in each of the first migration control units and the migration post means in the second migration control unit transmit the migration post message to each of the addresses related to each of the pointers.

13. The migration communication control device of Claim 12, wherein each of the pointers is a broadcast address of the network to which one of the first migration control units is attached.

14. The migration communication control device of Claim 12, wherein each of the pointers is an address which is assigned to one of the first migration control units uniquely.

15. The migration communication control device of claim 12, wherein each of the pointers is the address of the mobile node which is assigned when the mobile node is attached to the same network as is the first migration control unit, and

the migration post means in the first migration control unit and the migration post means in the second migration control unit obtain the broadcast address of the network to which each of the first migration control units is attached with referring to the address of the mobile node, and transmits the migration post message to the obtained broadcast address.

16. The migration communication control device of Claim 12, wherein the pointer hold means in the second migration control unit holds a pointer related to a first migration control unit for the latest migration, which is the first migration control unit being attached to one network before the network to which the mobile node is currently attached, and

the pointer hold means in the first migration control unit holds a pointer related to another

first migration control unit attached to the same network as was the mobile node attached before migrating to the network to which the first migration control unit is attached.

17. The migration communication control device of Claim 12, wherein the second migration control unit further transmits to the first migration control unit the pointer by sending thereto the migration post message, the pointer to be held by the first migration control unit.

18. The migration communication control device of Claim 17, wherein the first migration control unit stores into the pointer hold means the pointer when it receives from the second migration control unit the migration post message by corresponding the pointer with the updated address indicated by the received migration post message.

19. The migration communication control device of Claim 11, wherein each of the first migration control units further comprises:

address hold means for holding the outdated address and the updated address by corresponding them with each other, wherein

migration post message means stores into the address hold means the outdated address and the updated address by corresponding them with each other when it receives from the second migration control unit the migration post message, while converts the updated address in the address hold means into the updated address indicated by the migration post message when it receives from the first migration control unit the migration post message and the outdated address indicated by the migration post message coincides with one of the updated addresses in the address hold means.

20. The migration communication control device of Claim 1, wherein the first migration control unit is placed on a gateway, which connects networks.

21. The migration communication control device of Claim 1, wherein the first migration control unit is placed on the network as an individual node.

22. The migration communication control device of Claim 10, wherein the migration post means in the second migration control unit transmits the migration post message to a home migration control unit, the home migration control unit being the first migration control unit which is attached to a network where the mobile node left for its initial migration, and

the home migration control unit further

5

10

15

20

25

30

35

40

45

50

55

31

comprises home migration post means for transmitting a migration post message to a first migration control unit for the latest migration, the first migration control unit for the latest migration being the first migration control unit which is attached to the network where the mobile node left for the latest migration, to post the same updated address as is indicated by the migration post message received from the second migration control unit.

23. The migration communication control device of Claim 22, wherein the first migration control unit further comprises migration post means for transmitting the migration post message indicating the updated address of the mobile node to one of the other first migration control units when the conversion packet destined for the outdated address of the mobile node was sent therefrom to the first migration control unit.

24. The migration communication control device of Claim 22, wherein the migration post means in the second migration control unit transmits to the home migration control unit the migration post message where a home address and the updated address are corresponded with each other, the home address assigned when the mobile node is attached to the same network as is the home migration control unit,

and each of the packet transfer means and the address post means in the home migration control unit transmits the conversion packet and the address post message respectively with referring to the above home address and the updated address.

25. The migration communication control device of Claim 24, wherein the second migration control unit further comprises an outdated address post means for transmitting to the first migration control unit for the latest migration an outdated address post message where the outdated address and the home address are corresponded with each other, the outdated address being assigned to the mobile node before the latest migration,

the home migration post means in the home migration control unit transmits to the said first migration control unit for the latest migration the migration post message where the above home address and the updated address are corresponded with each other, and

the packet transfer means and the address post means in the first migration control unit for the latest migration transmit the conversion packet and the address post message respectively in accordance with the outdated address and the updated address, the outdated ad-

dress and the updated address being corresponded with each other via the home address.

26. The migration communication control device of the Claim 25, wherein the outdated address post means in the second migration control unit transmits the above outdated address post message at a migration of the mobile node preceding the latest migration, and

each of the migration post means in the second migration control unit and the home migration post means in the home migration control unit transmits the above migration post message at the latest migration of the mobile node.

27. The migration communication control device of Claim 22, wherein the second migration control unit further comprises home migration control unit pointer hold means for holding a pointer related to the home migration control unit,

the migration post means in the second migration control unit transmits the migration post message to the address related to the pointer,

the home migration control unit further comprises pointer hold means for the latest migration for holding a pointer related to the first migration control unit for the latest migration, and

the home migration post means in the home migration control unit transmits the migration post message to the address related to the pointer.

28. The migration communication control device of Claim 27, wherein each of the above pointers is the broadcast address of the network to which each of the first migration control units is attached.

29. The migration communication control device of Claim 27, wherein each of the above pointers is the address assigned to each of the first migration control units uniquely.

30. The migration communication control device of Claim 27, wherein the second migration control unit further comprises pointer obtainment means for requesting to the first migration control unit for the latest migration the pointer related to the first migration control unit for the latest migration, and

the migration post means in the second migration control unit posts the obtained pointer to the home migration control unit together with the updated address by sending thereto the migration post message.

31. The migration communication control device of Claim 30, wherein the migration post means in the second migration control unit posts to the

home migration control unit the pointer at the migration of the mobile node preceding the latest migration, while the migration post means posts the above updated address at the latest migration of the mobile node.

32. The migration communication control device of Claim 22, wherein the first migration control unit further comprises address post suppressing means for suppressing transmission of the address post message from the address post means to the third migration control unit, and

the address post suppressing means suppresses transmission of the address post message when none of the first migration control units is attached to the same network as is the mobile node.

33. The migration communication control device of Claim 32, wherein the second migration control unit further comprises detect means for detecting whether or not the first migration control unit is attached to the network to which the mobile node migrates,

the migration post means in the second migration control unit transmits to the home migration control unit the migration post message which includes the detecting result of the above detect means together with the updated address,

the home migration post means in the home migration control unit transmits to the first migration control unit for the latest migration the migration post message which includes the detecting result of the above detect means together with the updated address, and

the address post suppressing means in each of the home migration control unit and the first migration control unit for the latest migration suppress the transmission of the address post message in accordance with the detecting result of the above detect means.

34. The migration communication control device of Claim 22, wherein the first migration control unit further comprises packet transfer suppressing means for suppressing transfer of the packet conducted by the packet transfer means.

35. The migration communication control device of Claim 34, wherein the first migration control unit further comprises address post suppressing means for suppressing transmission of the address post message from the address post means to the third migration control unit, and the address post suppressing means in the first migration control unit being attached to a network to which the mobile node is not attached, suppresses the transmission of the address post message

when the packet transfer suppressing means in the first migration control unit for the latest migration suppresses transfer of the packet.

- 5 36. The migration communication control device of Claim 35, wherein the second migration control unit further comprises detect means for detecting whether or not the packet transfer suppressing means in the first migration control means suppresses the transfer of the packet, the first migration control means being attached to the network to which the mobile node migrates, and

the migration post means in the second migration control unit transmits to the home migration control unit the migration post message which includes the detecting result of the above detect means together with the updated address,

the home migration post means in the home migration control unit transmits to the first migration control unit for the latest migration the migration post message which includes the detecting result of the detect means together with the updated address, and

the address post suppressing means in each of the home migration control unit and the first migration control unit for the latest migration suppresses the transmission of the address post message in accordance with the detecting result of the above detect means.

37. The communication control device of Claim 36, wherein the packet transfer suppressing means in the first migration control unit for the latest migration suppresses the transfer of the packet conducted by the packet transfer means, when the packet transfer suppressing means in the first migration control unit being attached to the network to which the mobile node migrates suppresses the transfer of the packet.

38. A packet transfer migration control unit in a migration communication control device, the migration communication control device being constructed to control a communication between a mobile node and a partner node, the mobile node migrating across networks and obtaining an address assigned on each network while the partner node being a communication partner of the mobile node, comprising:

packet transfer means for receiving a packet which was transmitted by the partner node to an outdated address of the mobile node, the outdated address being assigned when the mobile node migrated to a network to which the packet transfer migration control unit is attached, generating a conversion packet which holds an updated address instead of the outdated address, and transmitting the conversion packet;

and

address post means for transmitting an address post message which indicates the updated address of the mobile node to the partner node, the partner node transmitting the packet received by the packet transfer means. 5

39. A mobile node migration control unit in a migration communication control device, the migration communication control device being constructed to control a communication between a mobile node which migrates across networks and obtains an address assigned on each network and a partner node which is a communication partner of the mobile node, being placed on the mobile node and comprising: 10

migration post means for transmitting to a packet transfer migration control unit a migration post message which indicates an updated address of the mobile node when the mobile node migrates to another network, the packet transfer migration control unit for receiving a packet which was transmitted by the partner node to an outdated address of the mobile node, the outdated address assigned when the mobile node migrated to a network to which the migration control unit for packet transfer is attached, generating a conversion packet which holds the updated address instead of the outdated address, and transmitting the conversion packet; and 20

packet resumption means for receiving the conversion packet from both the packet transfer migration control unit and the mobile node, and resuming an original packet from the conversion packet. 25

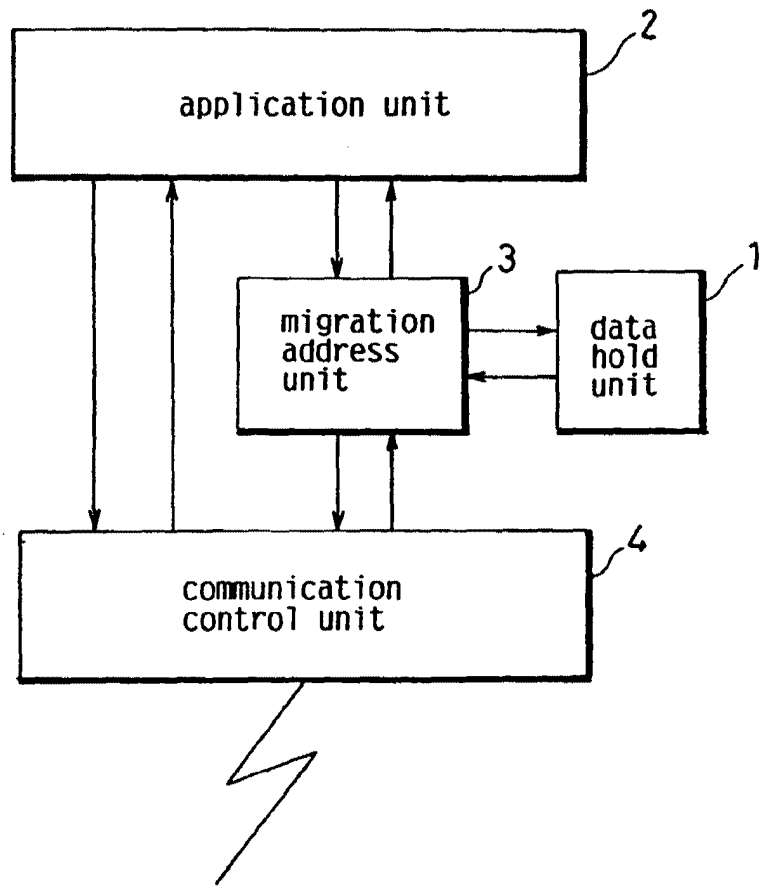
40. A partner node migration control unit in a migration communication control device, the migration communication control device being constructed to control a communication between a mobile node which migrates across networks and obtains an address assigned on each network and a partner node which is a communication partner of the mobile node, being placed on the mobile node and comprising: 40

address post message receiving means for receiving an address post message which indicates an updated address of the mobile node from a packet transfer migration control unit, the packet transfer migration control unit transmitting an address post message which indicates the updated address of the mobile node to the partner node; and 45

packet conversion means for converting a destination address of a packet, the packet to be transmitted to the mobile node, into the updated address indicated by the address post message, and transmitting it to the mobile node. 50

34

FIG. 1



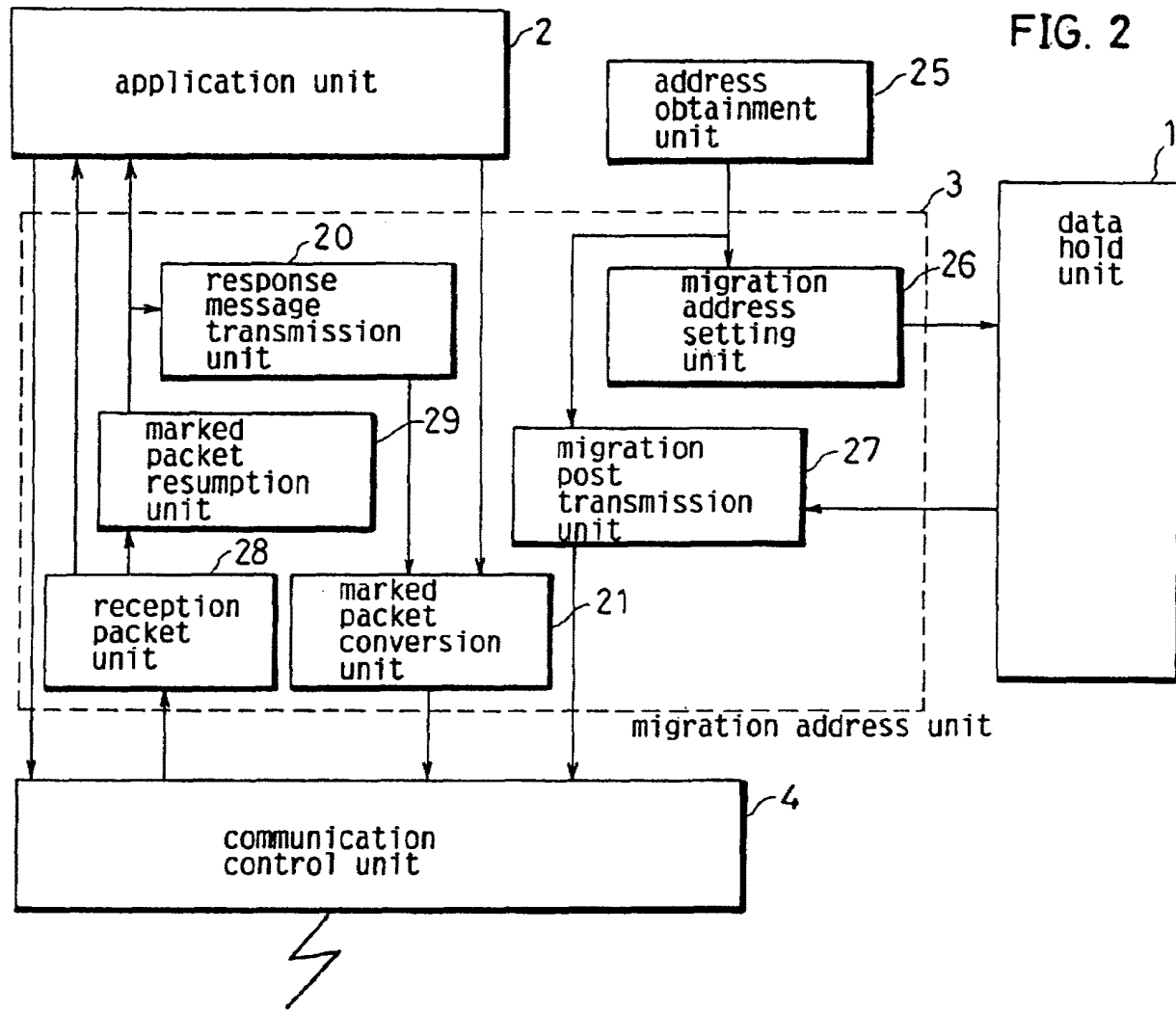


FIG. 2



FIG. 3

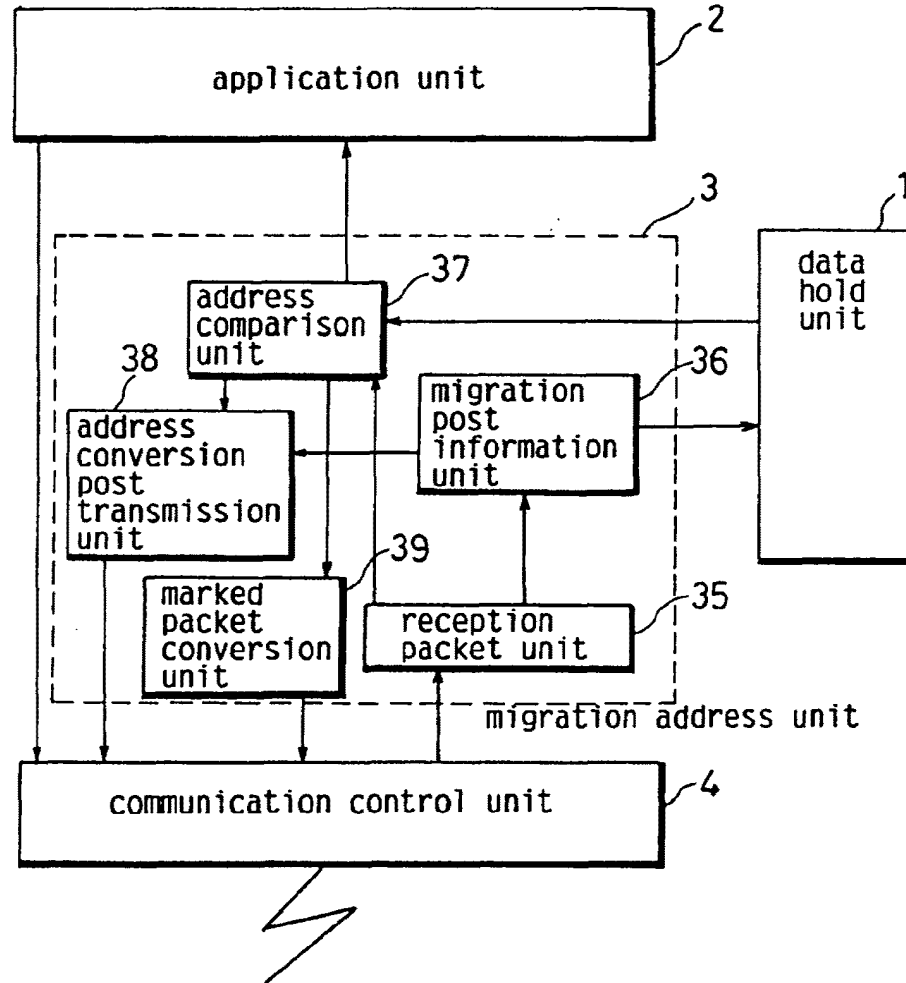


FIG. 4

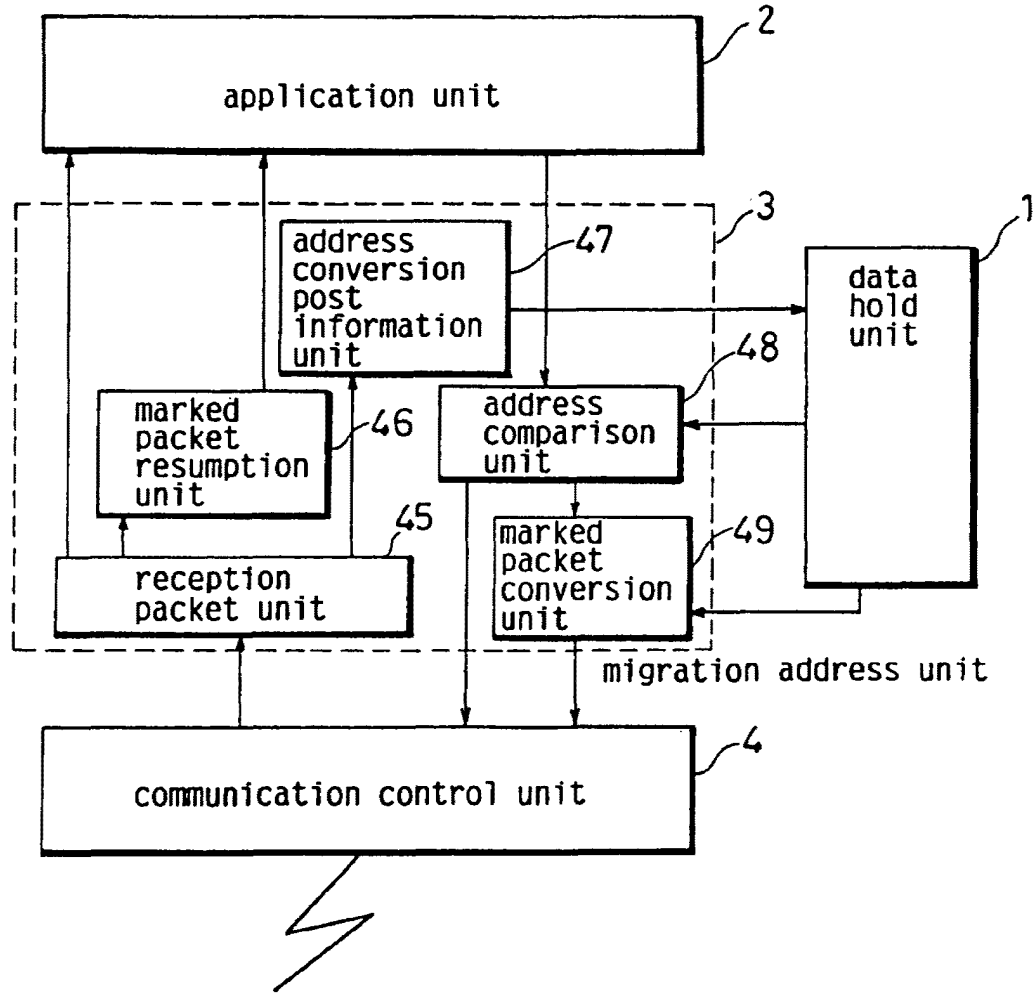
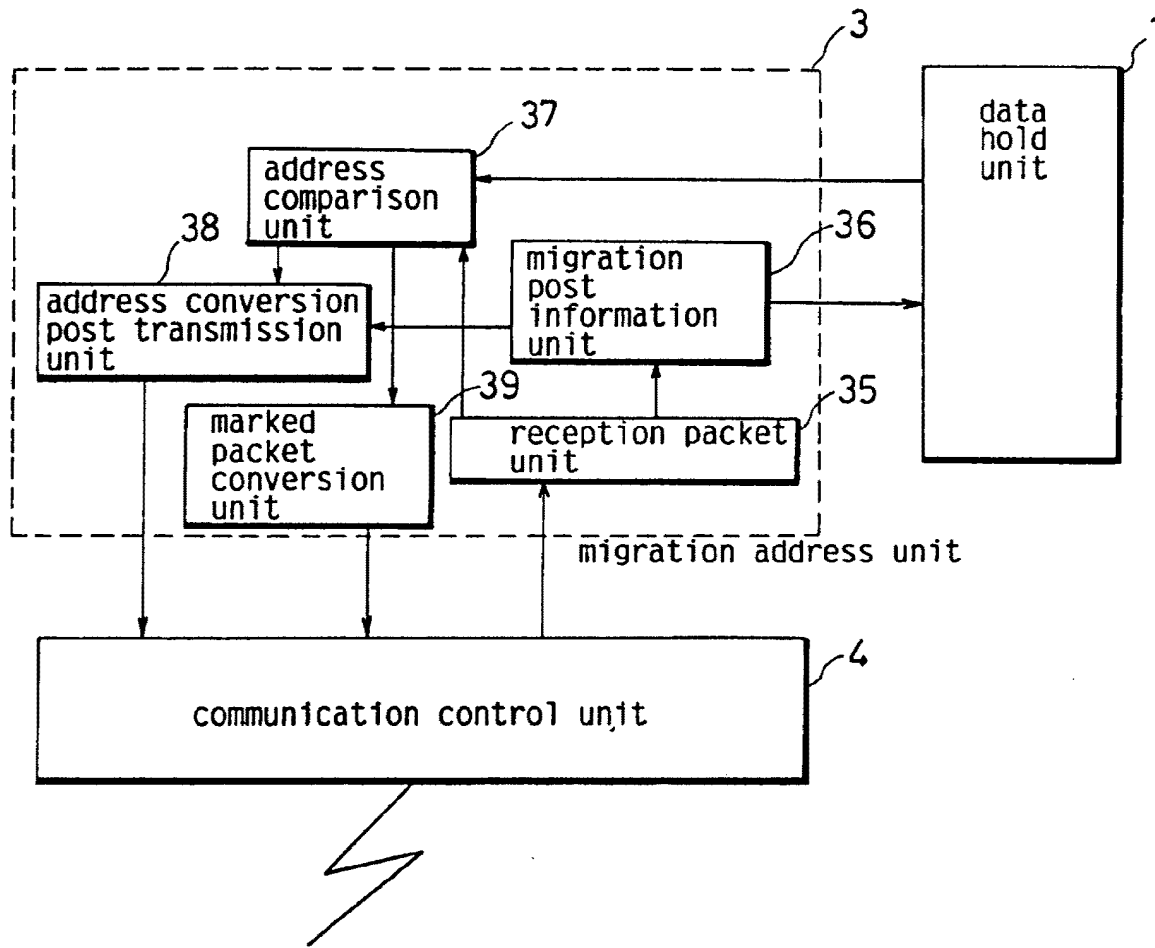


FIG. 5



39

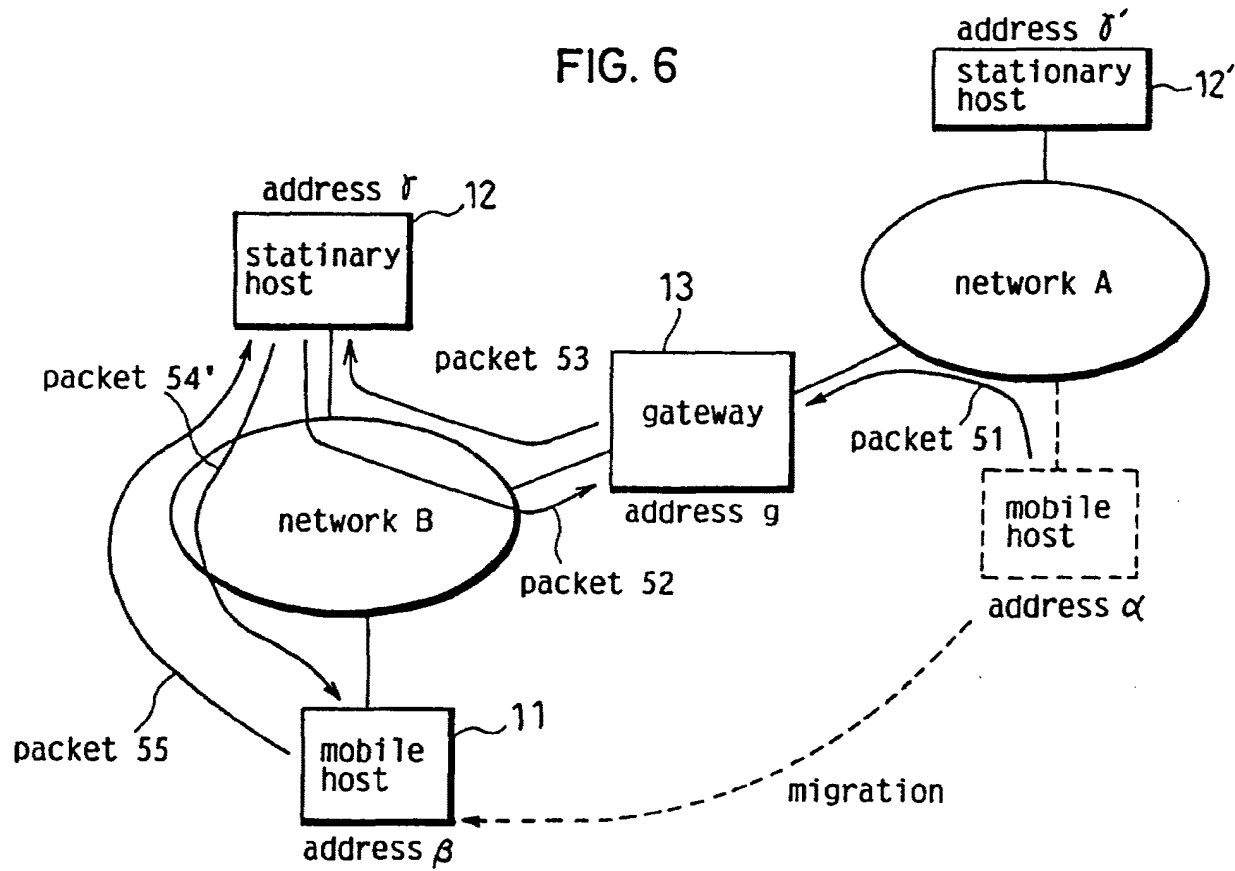


FIG. 7

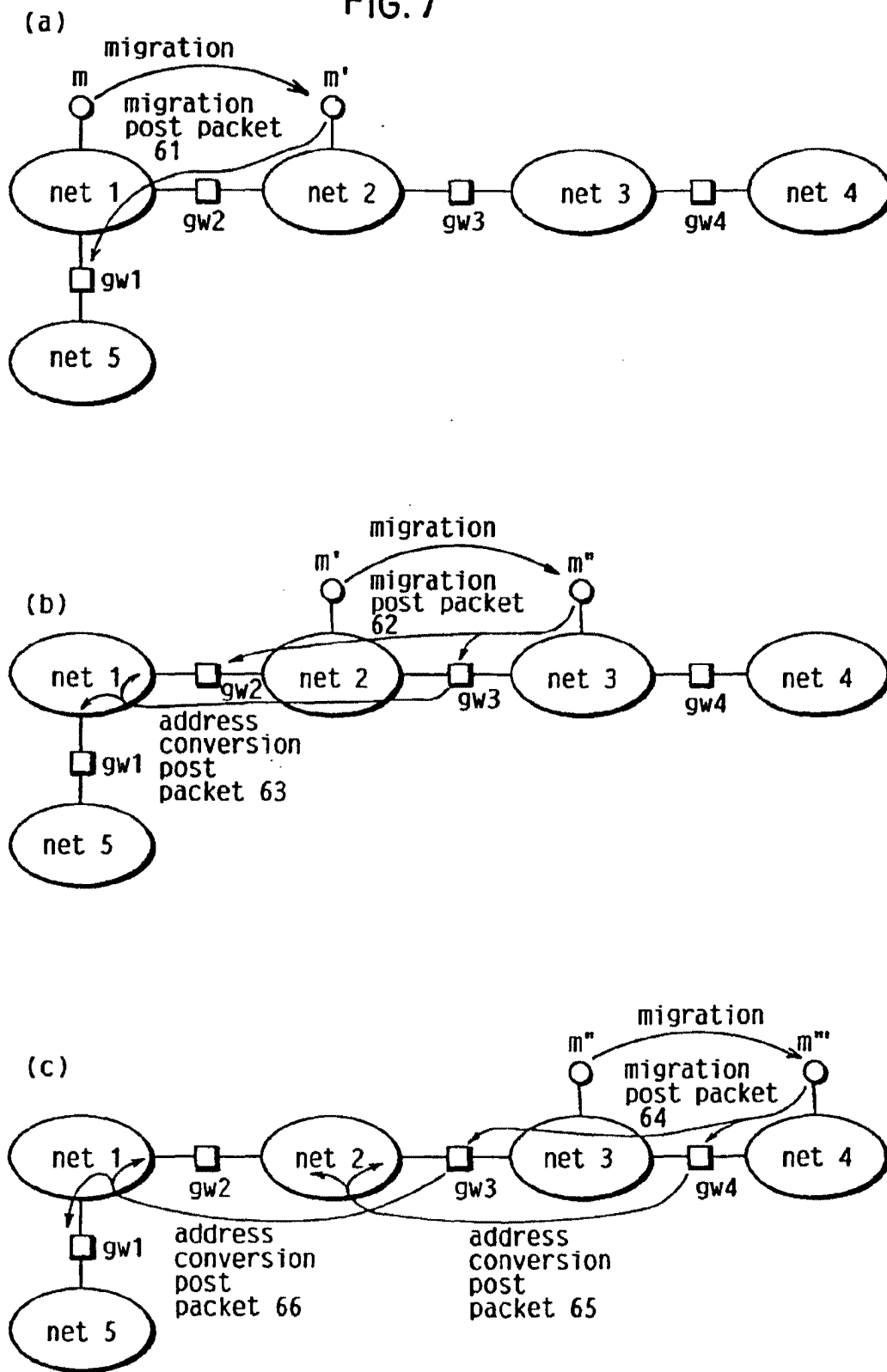


FIG. 8

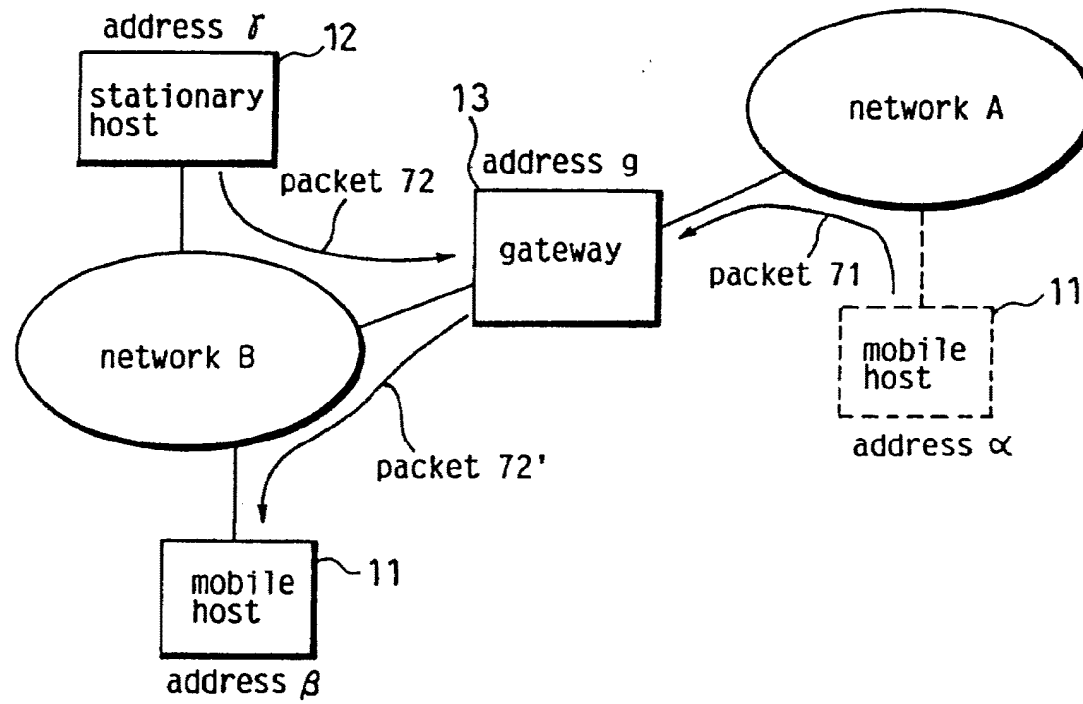


FIG. 9

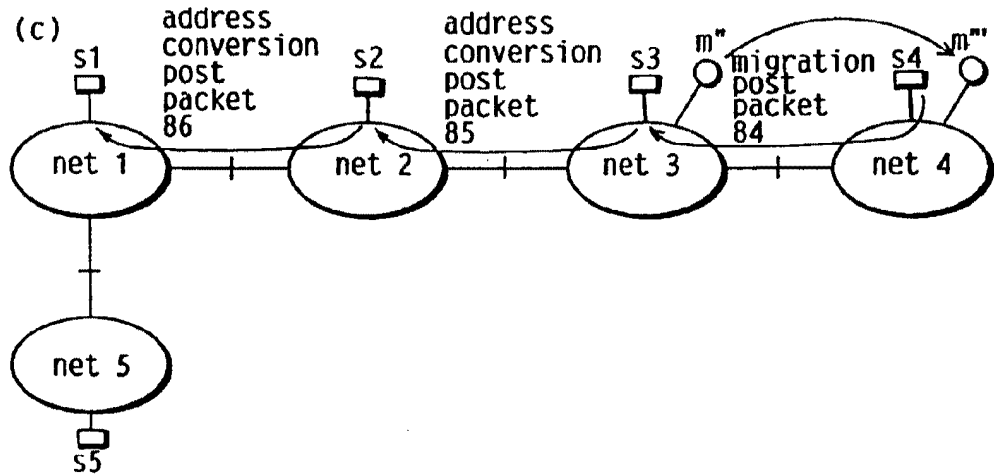
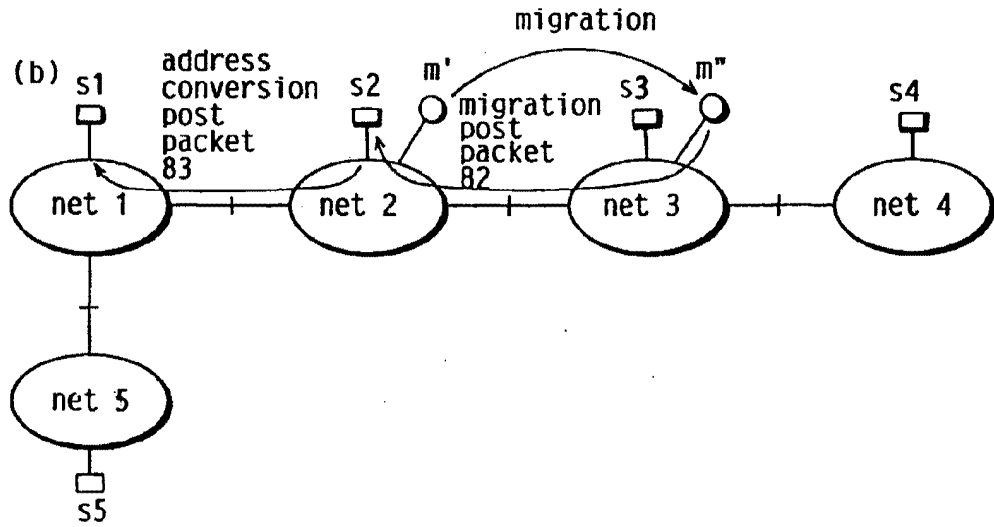
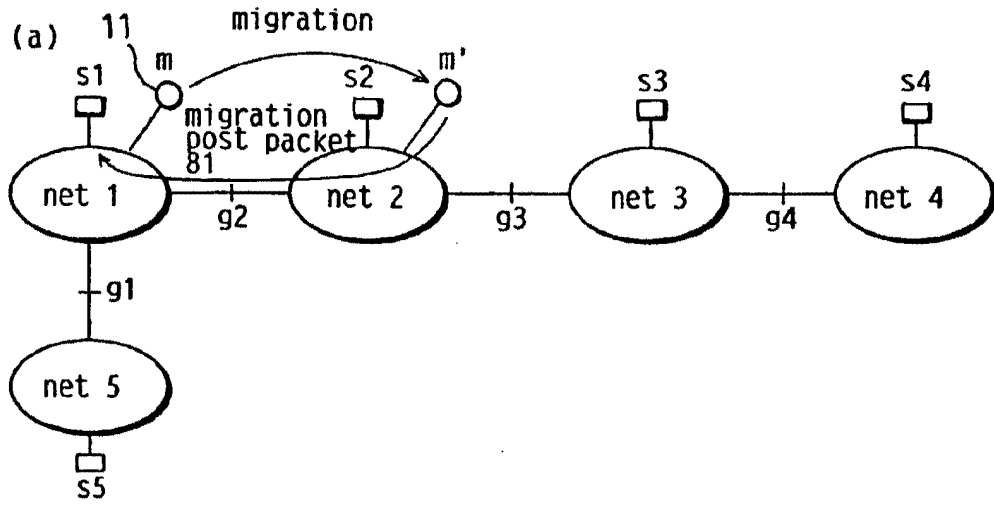


FIG. 10

(a)

address before migration	address after migration
address $\alpha$	address $\beta$

(b)

address before migration	address after migration
address $\alpha$	address $\beta$
address $X$	address $Y$



FIG. 11

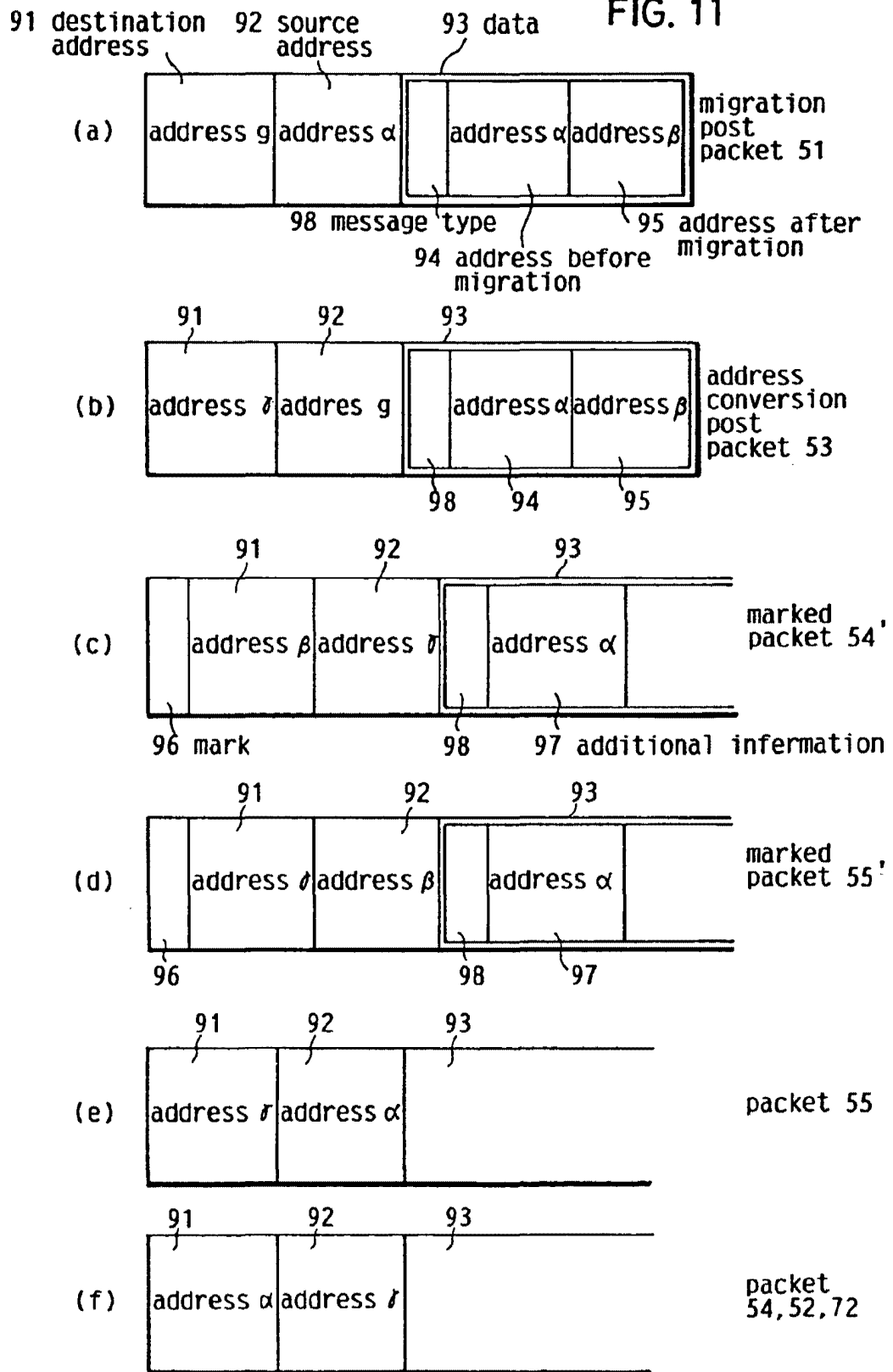
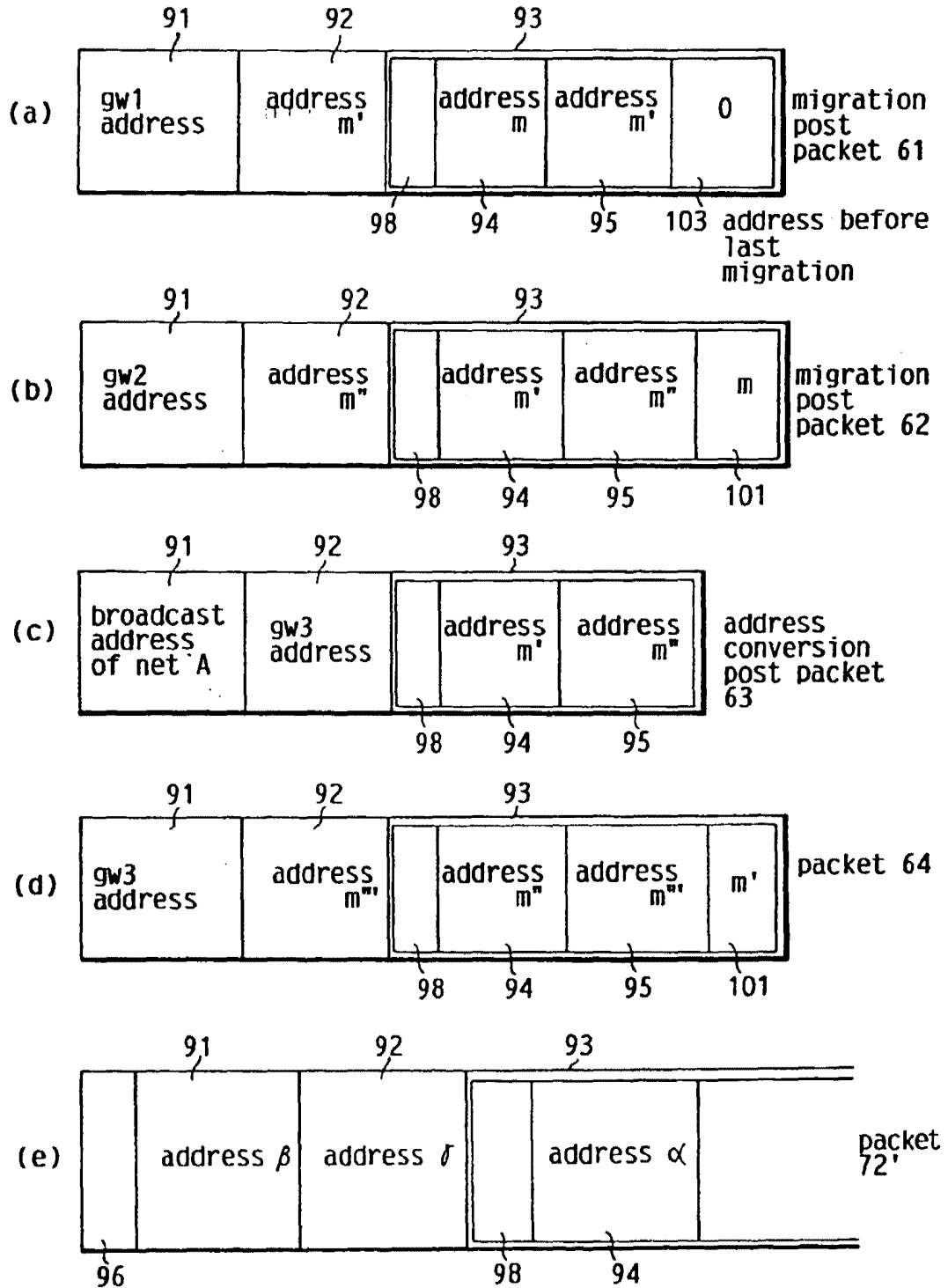


FIG. 12



(a) migration from network A to network B

FIG. 13

gateway	address correspondence	address before last migration
gw1	$m \rightarrow m'$	0
gw2	$m \rightarrow m'$	0
gw3	—	—
gw4	—	—

(b) migration from network B to network C

gateway	address correspondence	address before last migration
gw1	$m \rightarrow m''$	0
gw2	$\frac{m \rightarrow m''}{m' \rightarrow m''}$	$\frac{0}{m}$
gw3	$m' \rightarrow m''$	m
gw4	—	—

(c) migration from network C to network D

gateway	address correspondence	address before last migration
gw1	$m \rightarrow m'''$	0
gw2	$\frac{m \rightarrow m'''}{m' \rightarrow m'''}$	$\frac{0}{m}$
gw3	$\frac{m' \rightarrow m'''}{m'' \rightarrow m'''}$	$\frac{m}{m'}$
gw4	$m'' \rightarrow m'''$	m'

(a) migration from network A to network B

FIG. 14

migration communication control device	content of hold unit	
	address correspondence	address before last migration
S1	$m \rightarrow m'$	0
S2	—	—
S3	—	—
S4	—	—

(b) migration from network B to network C

migration communication control device	content of hold unit	
	address correspondence	address before last migration
S1	$m \rightarrow m''$	0
S2	$m' \rightarrow m''$	m
S3	—	—
S4	—	—

(c) migration from network C to network D

migration communication control device	content of hold unit	
	address correspondence	address before last migration
S1	$m \rightarrow m'''$	0
S2	$m' \rightarrow m'''$	m
S3	$m'' \rightarrow m'''$	m'
S4	—	—

FIG. 15

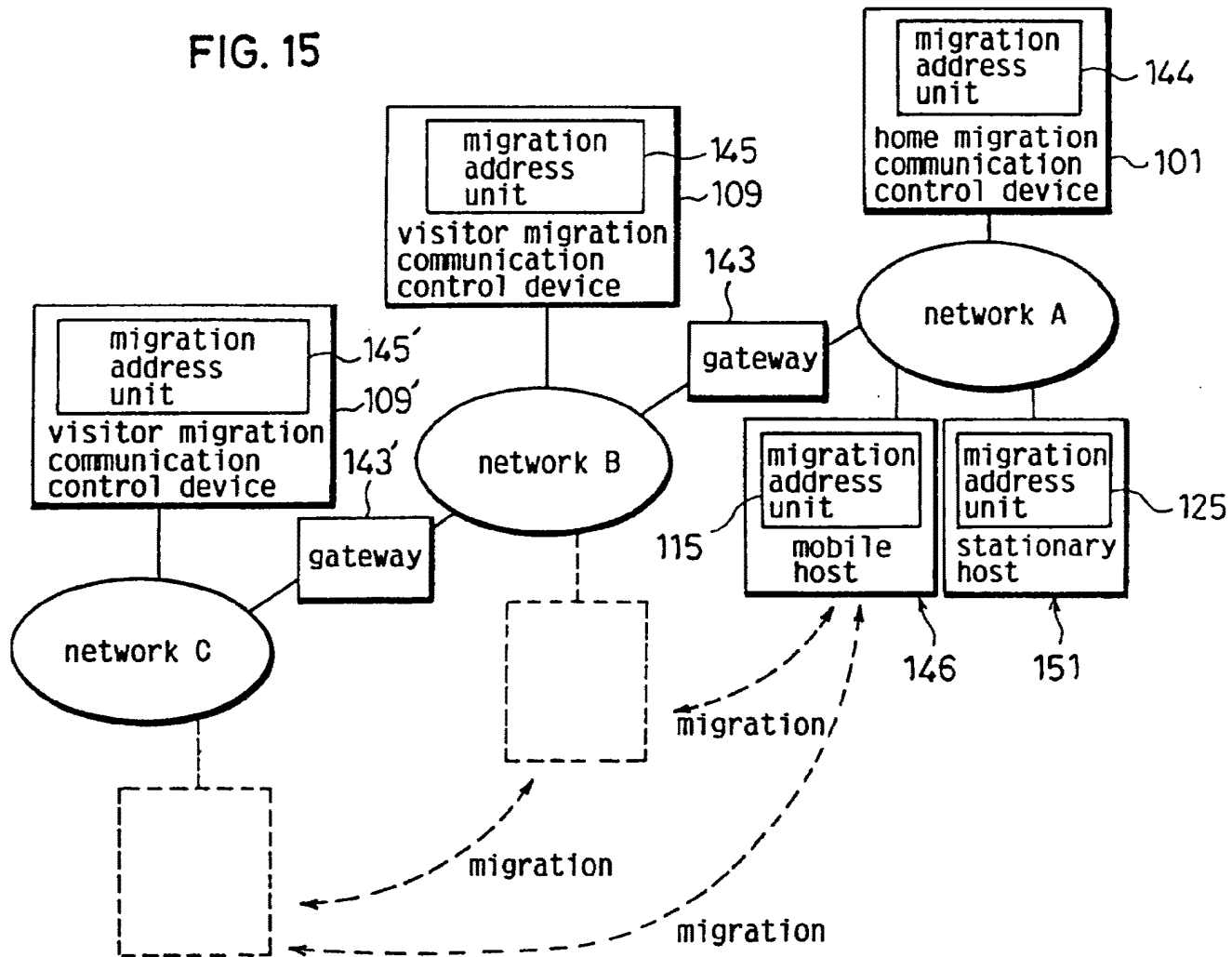


FIG. 16

home migration communication control device 101

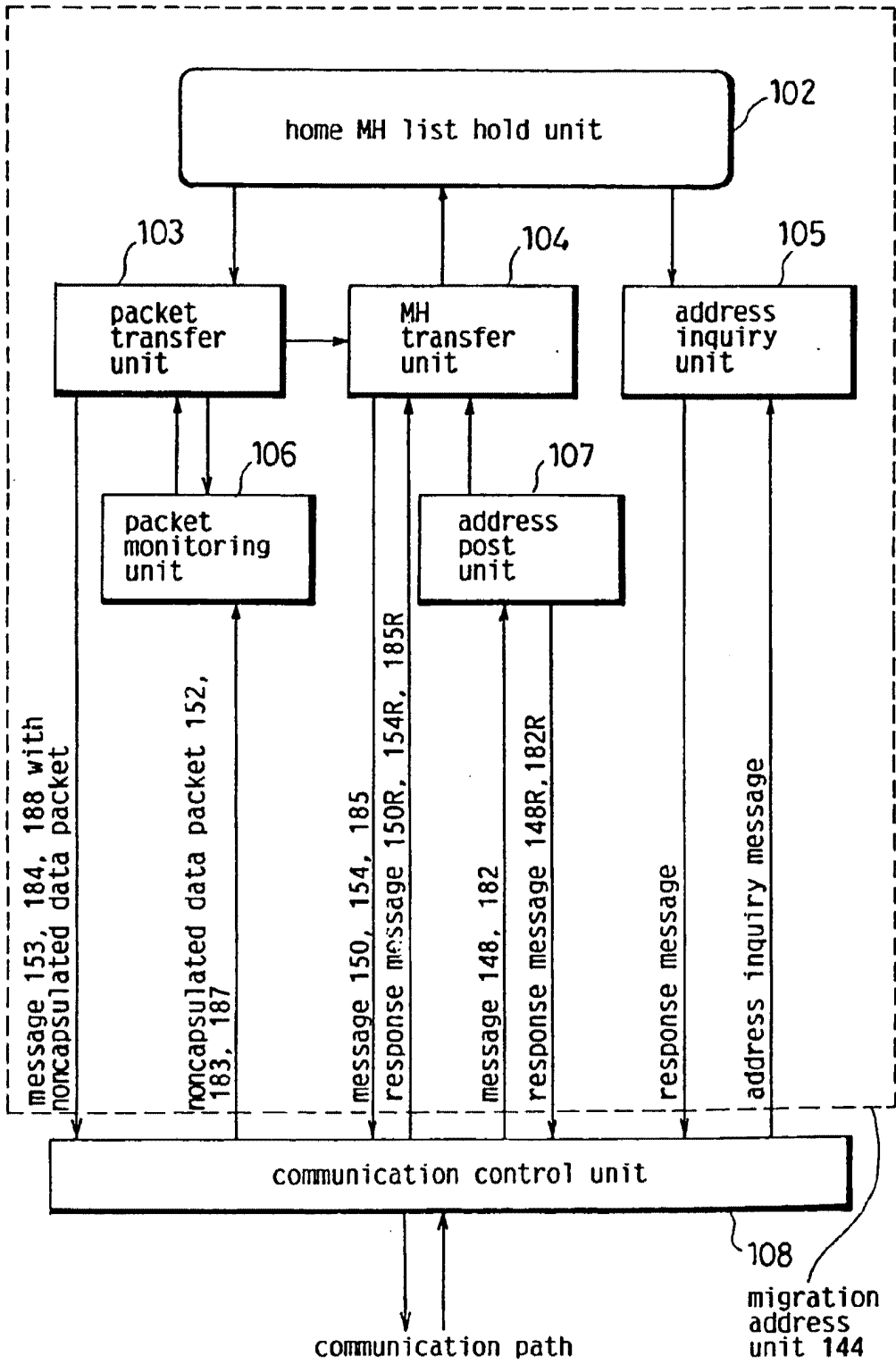


FIG. 17

MH's home address	MH's current temporary address	autonomous flag F	current broadcast address
$\alpha$	$\beta$ or $\delta$	1	Bba or Cba

FIG. 18

visitor migration communication control device 109(109')

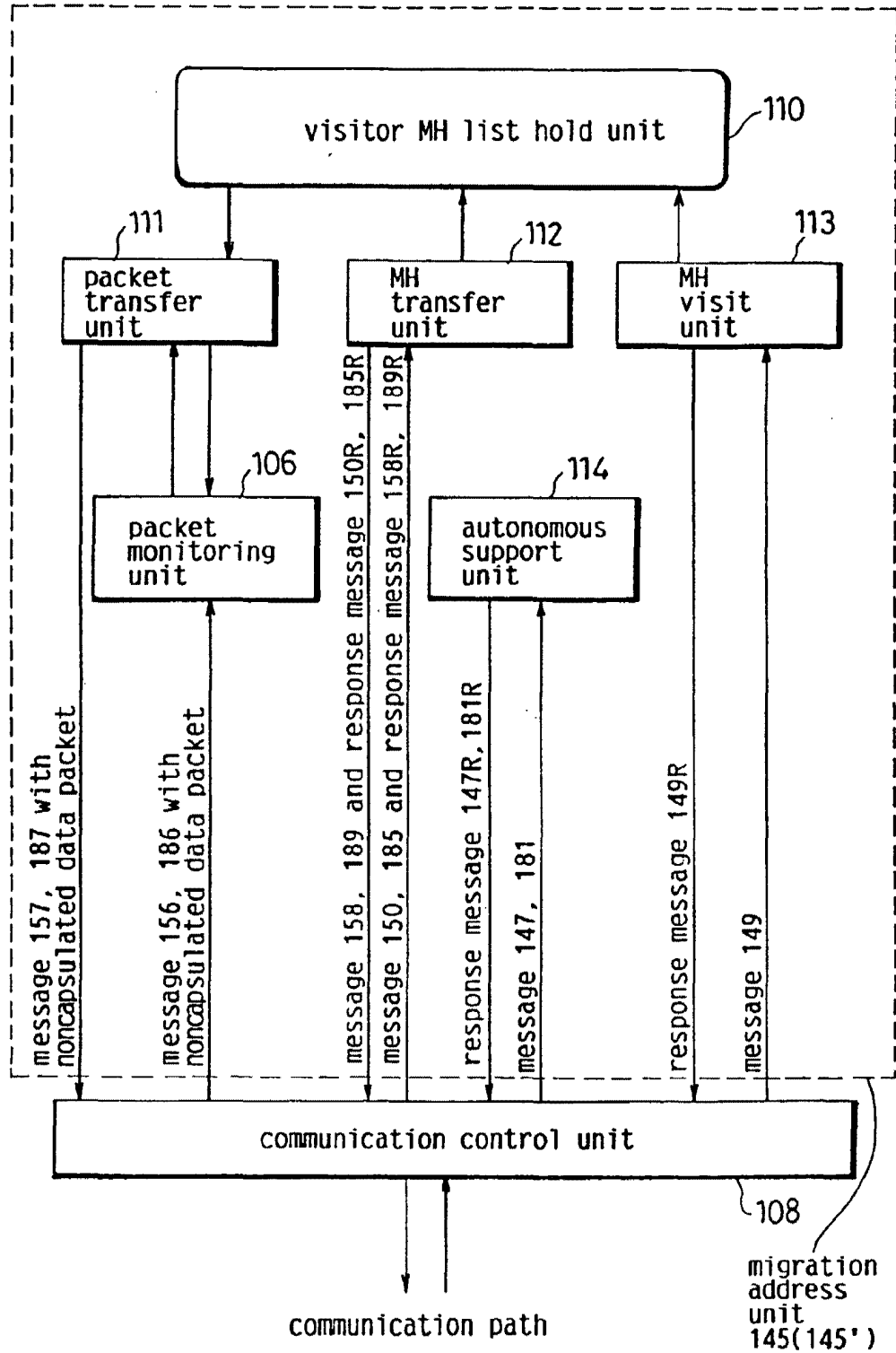




FIG. 19

MH's home address	temporary address	temporary address after migration	autonomous flag F
$\alpha$	$\beta$	$\delta$	1

FIG. 20

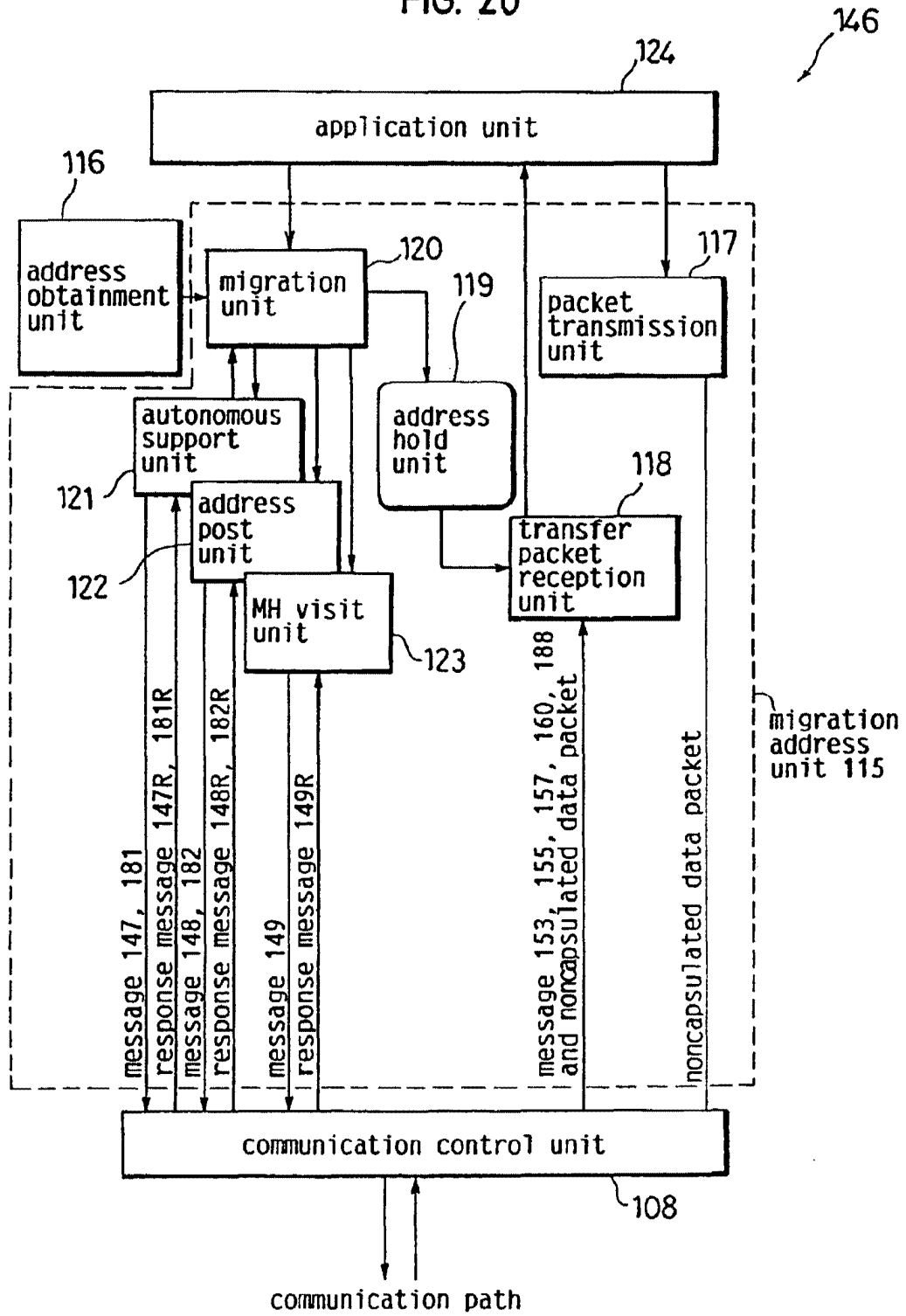


FIG. 21

home address	broadcast address of home network	current temporary address	broadcast address
$\alpha$	Aba	$\beta$ or $\delta$	Bba or Cba

FIG. 22

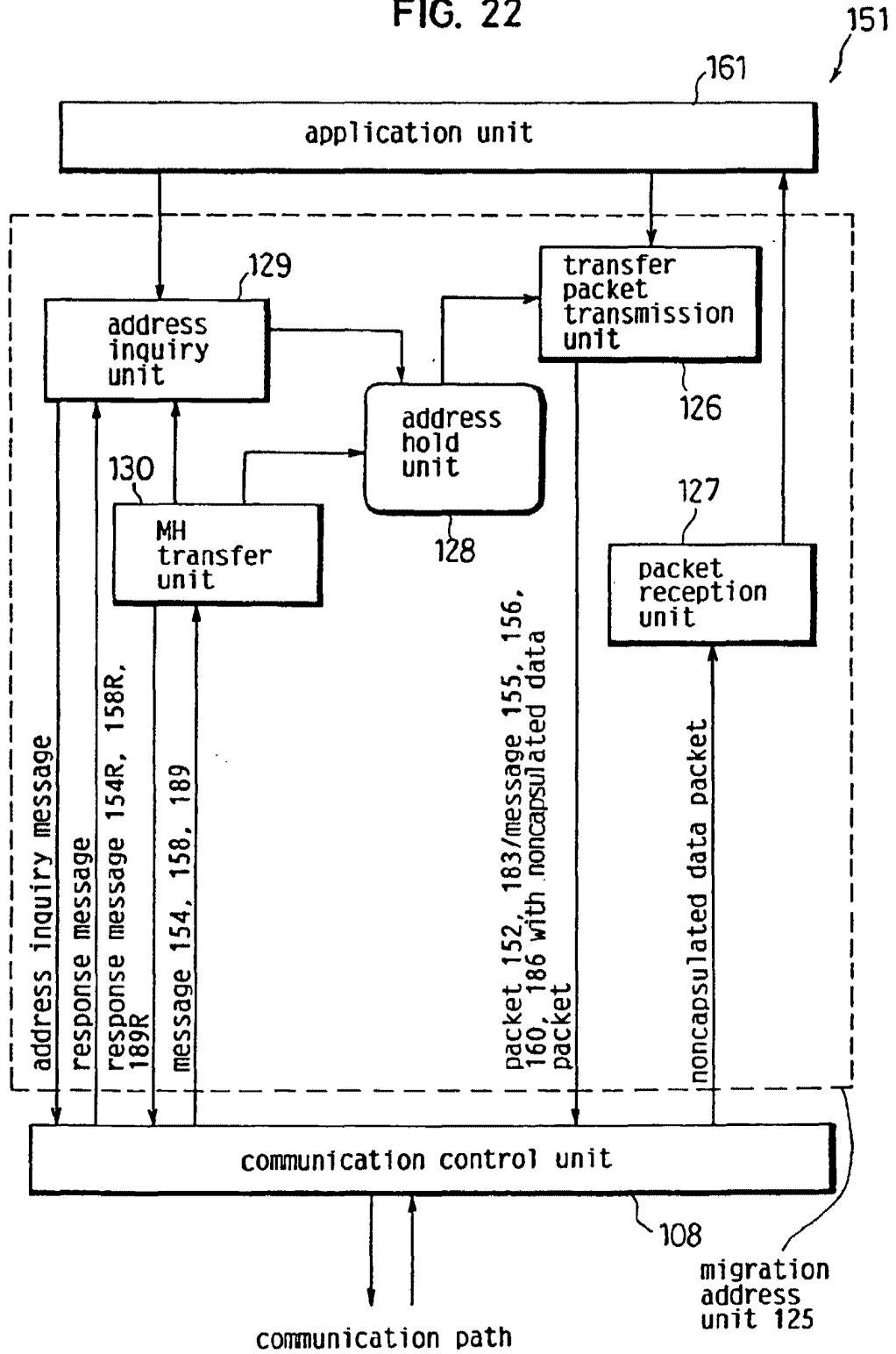


FIG. 23

MH's home address	MH's temporary address
$\alpha$	$\beta$ or $\delta$

FIG. 24

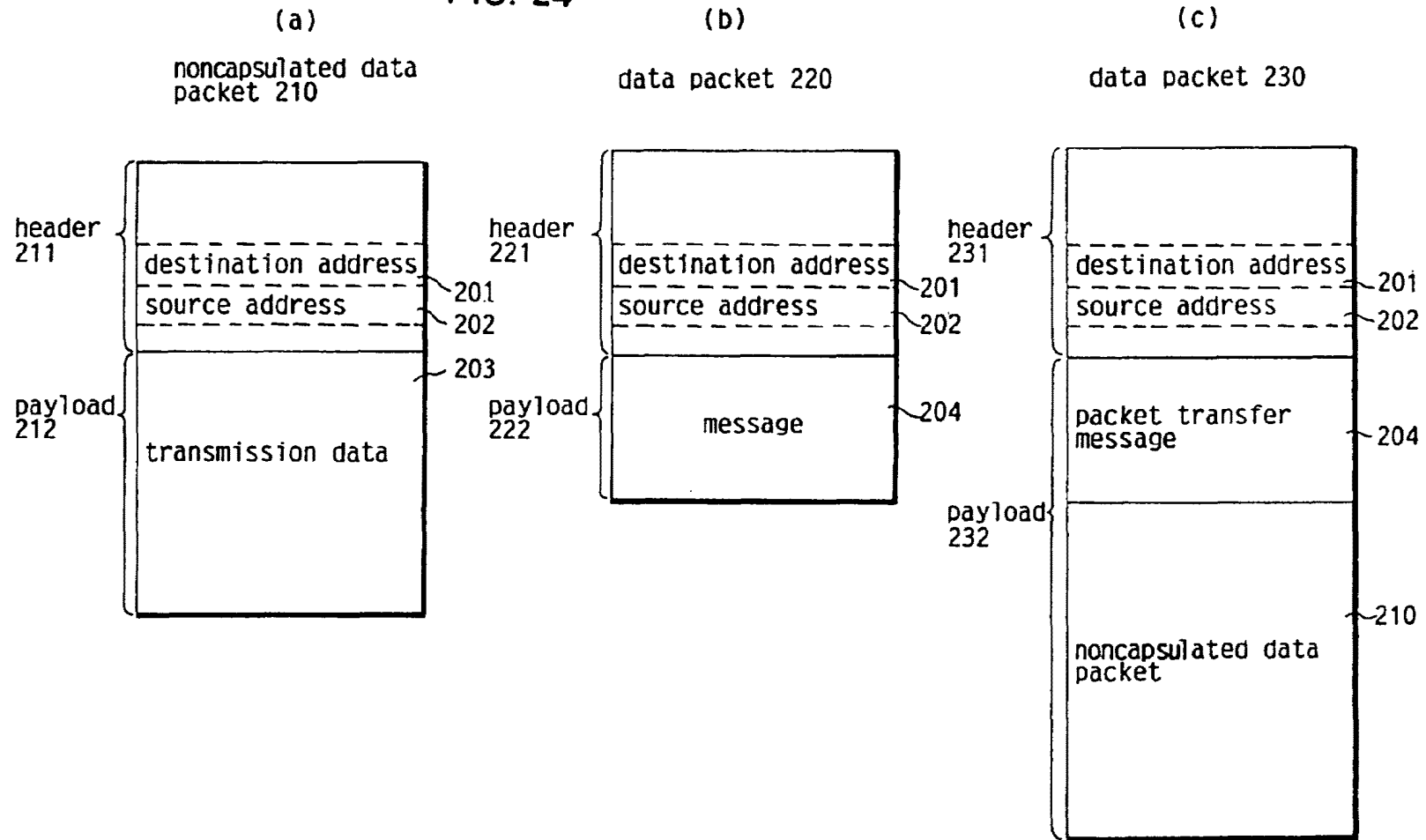
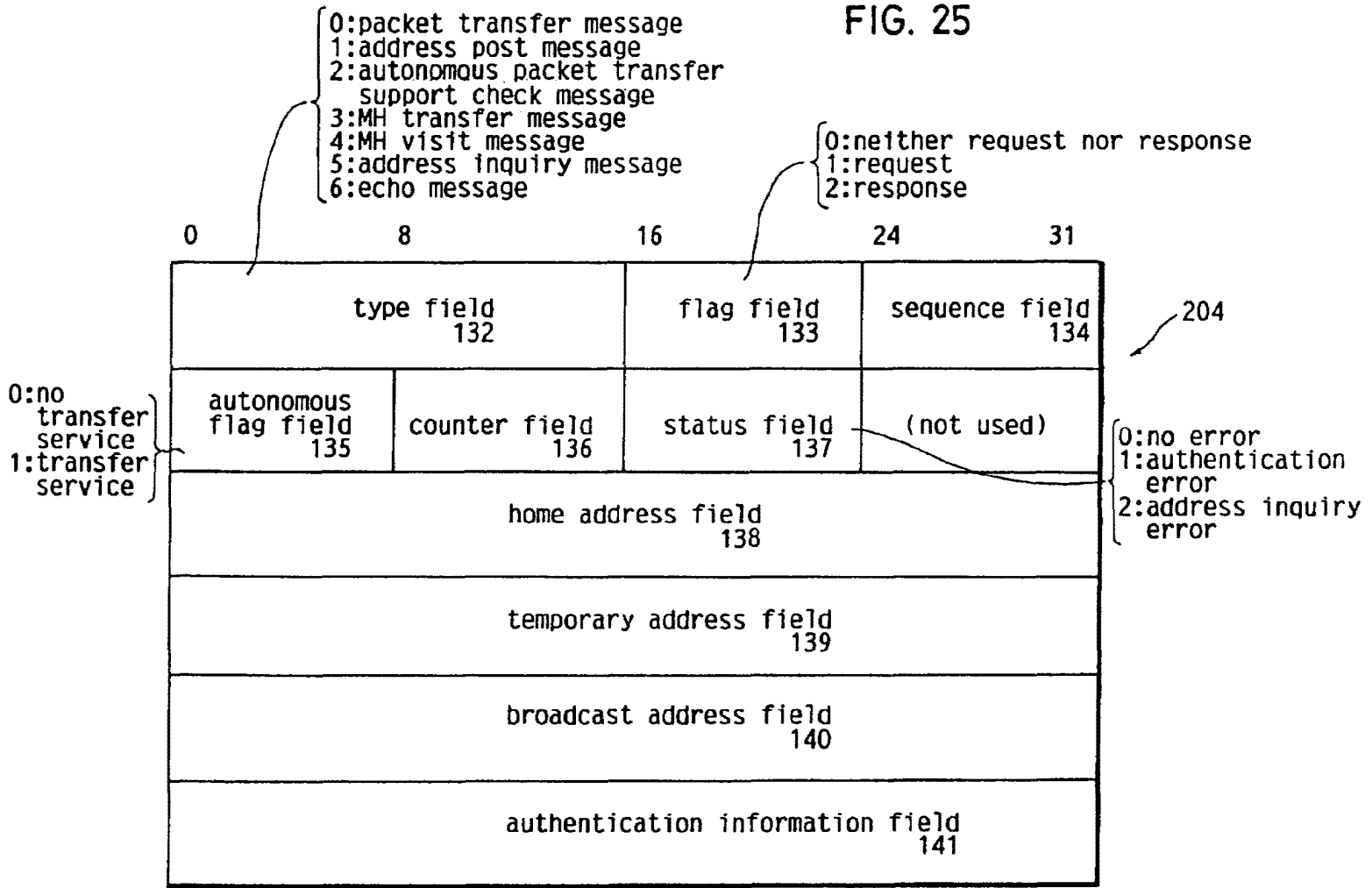


FIG. 25



EP 0 556 012 A2

FIG. 26

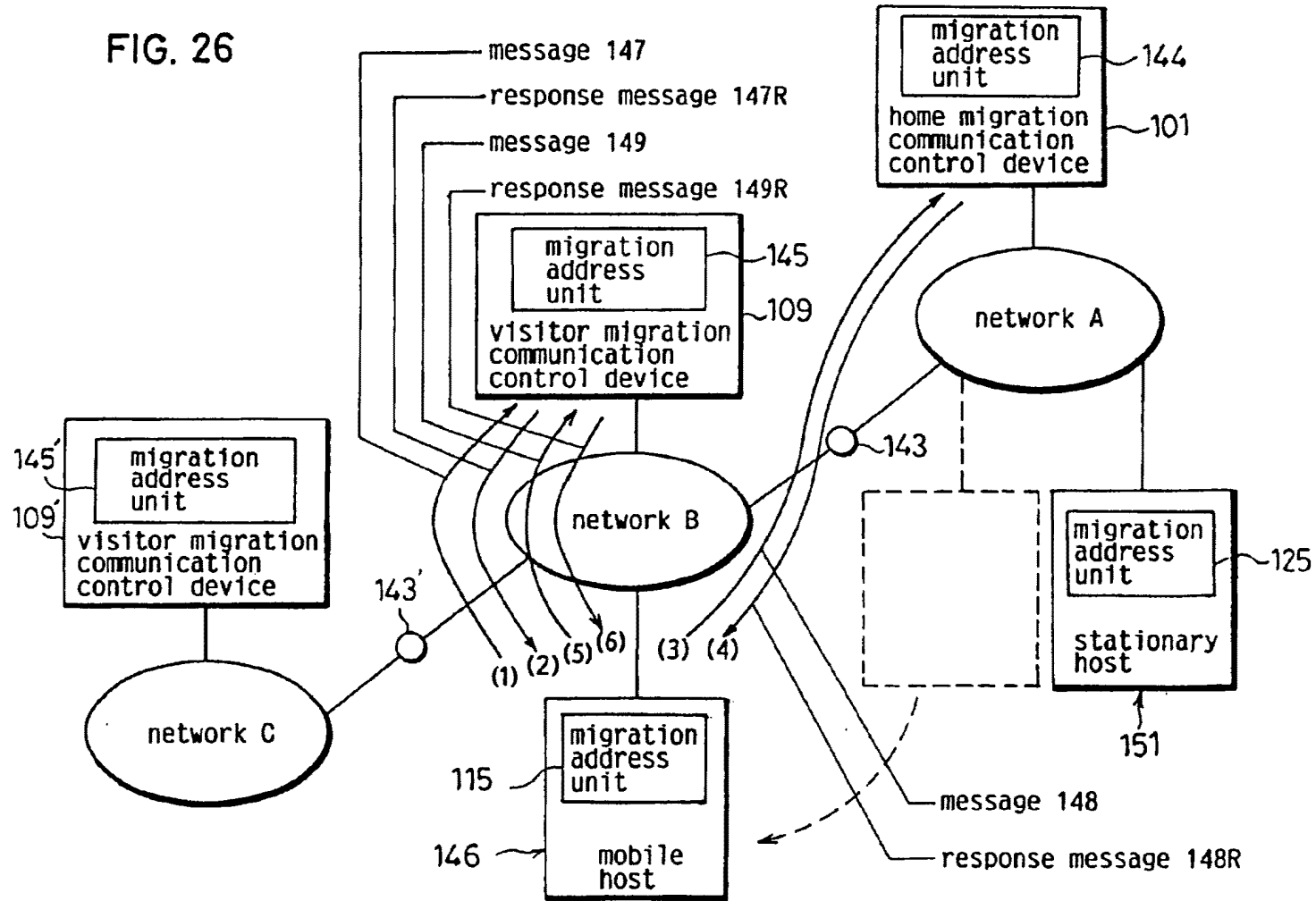
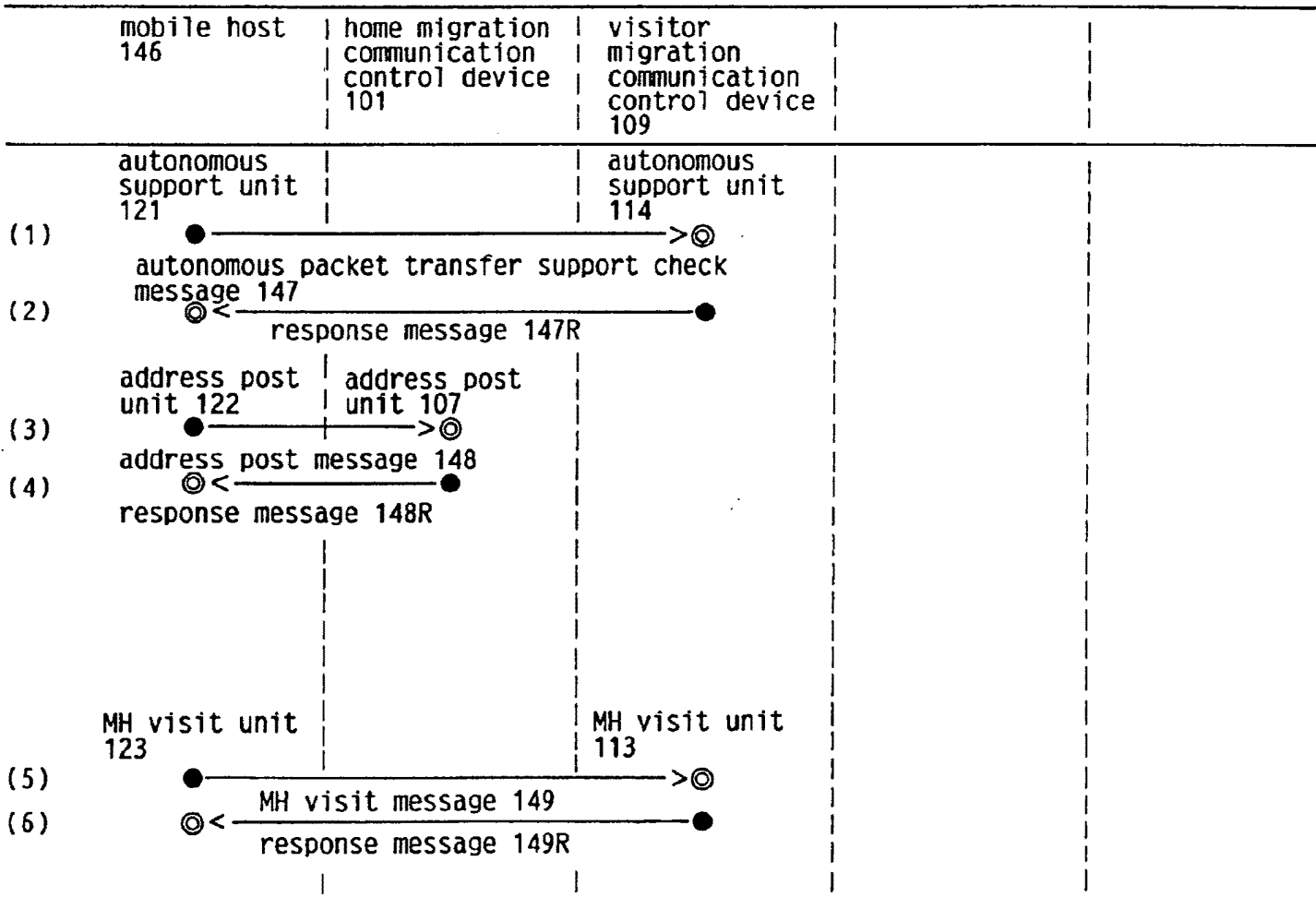
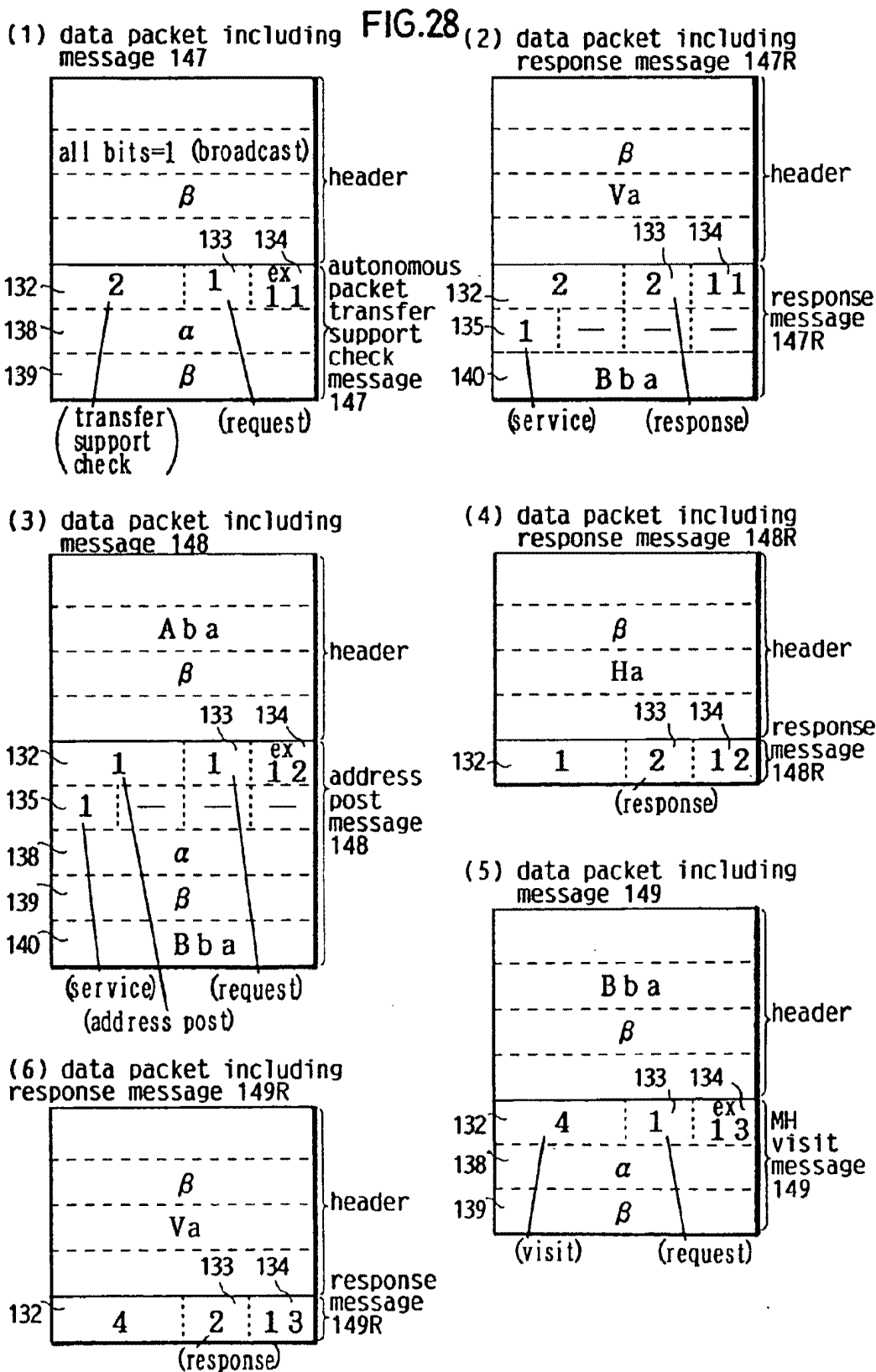




FIG. 27 migration from network A to network B



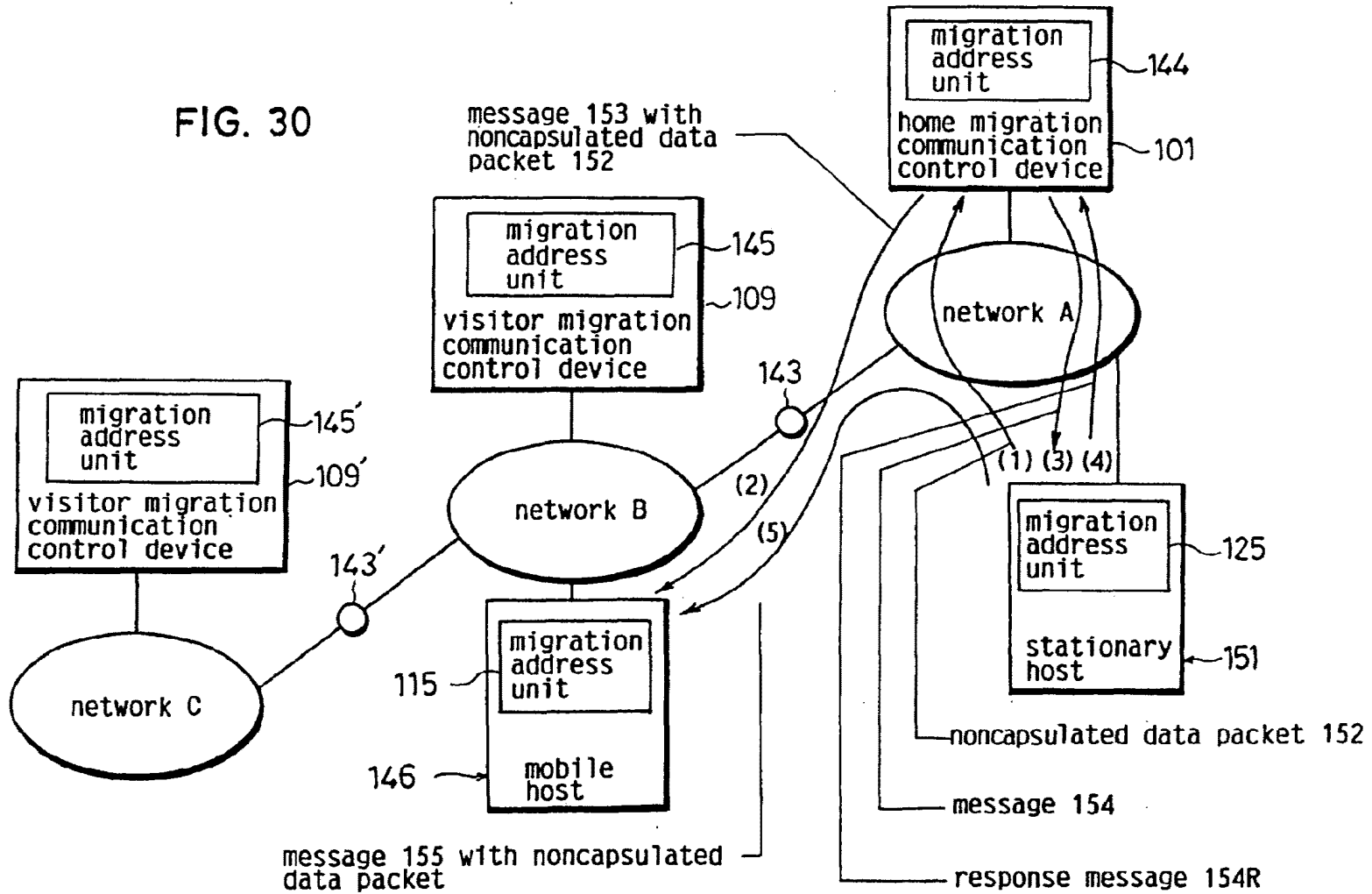
EP 0 556 012 A2



address after obtaintment of $\beta$	address hold unit 119(146)				home MH host list hold unit 102(101)				visitor MH list hold unit 110(109)				address hold unit 128(151)						
	$\alpha$	Aba	$\beta$	-	$\alpha$	$\alpha$	-	-	-	-	$\alpha$	$\beta$	$\beta$	-	-				
(1)																			
(2)	$\alpha$	Aba	$\beta$	-															
(3)					$\alpha$	$\beta$	1	Bba											
(4)																			
(5)										$\alpha$	$\beta$	$\beta$	-						
(6)																			
		home address	broadcast address of home network	current temporary address	current broadcast address	MH's home address	MH's current temporary address	autonomous flag F	current broadcast address	MH's home address	temporary address	temporary address after migration	autonomous flag F	MH's home address	temporary address	temporary address after migration	autonomous flag F	MH's home address	MH's temporary address

FIG. 29

FIG. 30

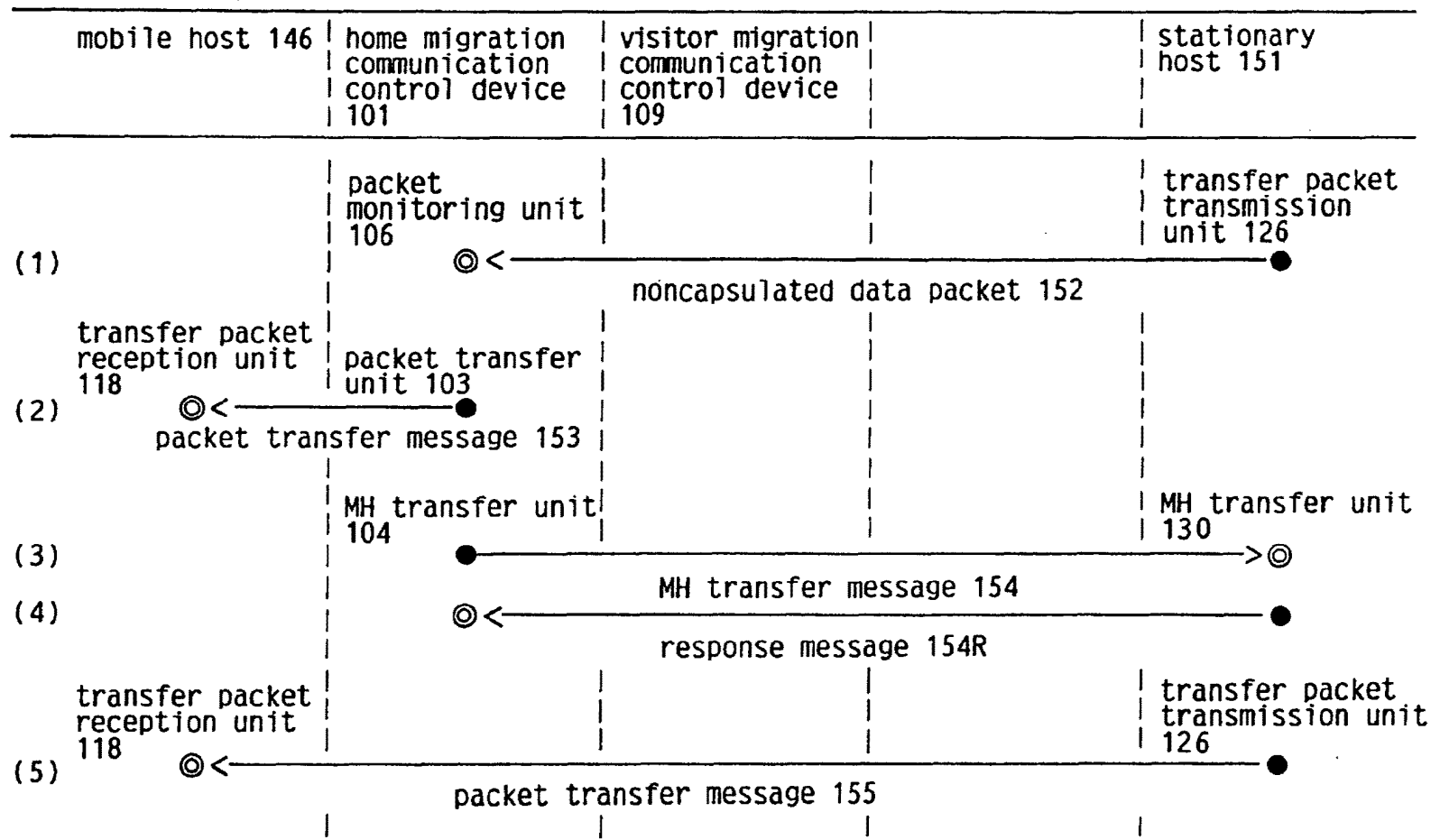


EP 0 556 012 A2

64

FIG. 31

data packet from SH 151 to MH 146 on network B

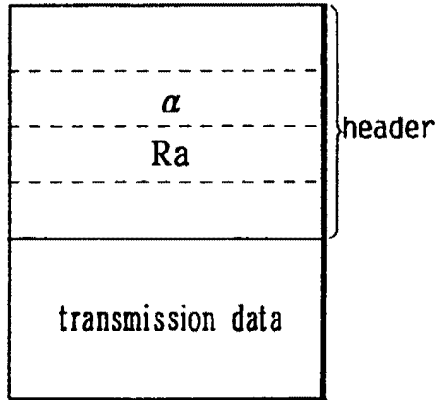


EP 0 556 012 A2

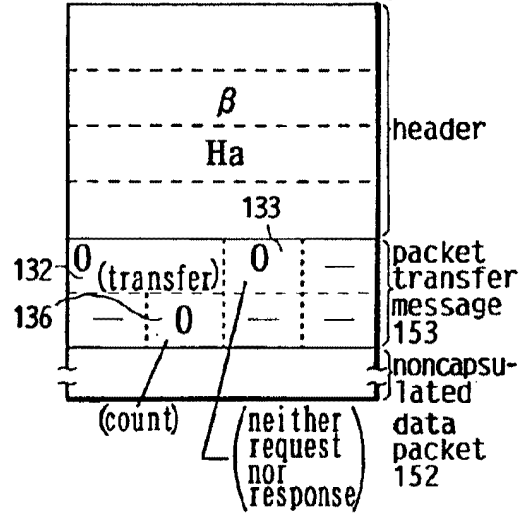
65

FIG. 32

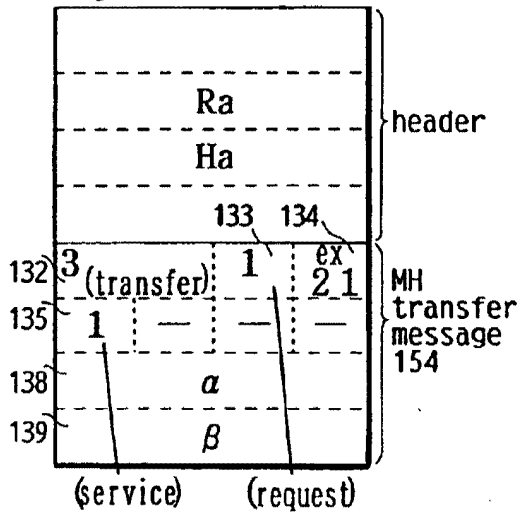
(1) noncapsulated data packet 152



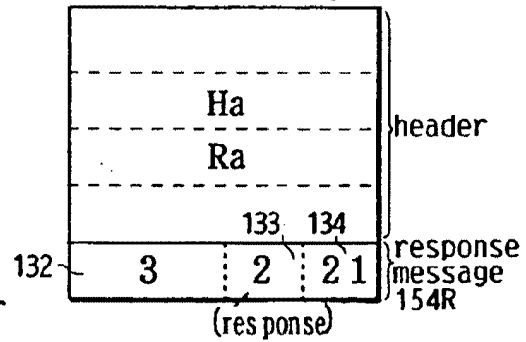
(2) data packet including message 153 and noncapsulated data packet



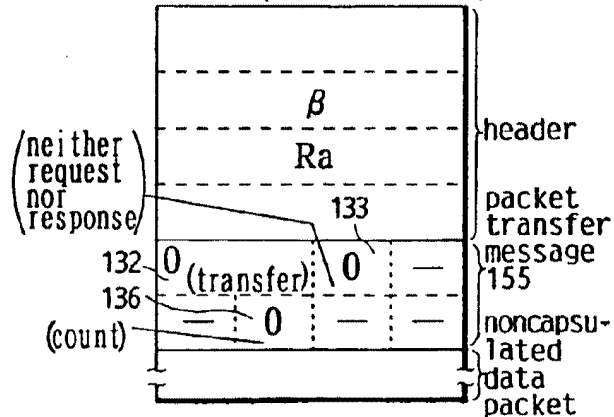
(3) data packet including message 154



(4) data packet including response message 154R



(5) data packet including message 155 and noncapsulated data packet



address before communication	address hold unit 119(146)				home MH 1st hold unit 102(101)				visitor MH 1st hold unit 110(109)				visitor MH 1st hold unit 110'(109')				address hold unit 128(151)	
	$\alpha$	Aba	$\beta$	Bba	$\alpha$	$\beta$	1	Bba	$\alpha$	$\beta$	$\beta$	1	1	1	1	1	$\alpha$	$\beta$
(1)																		
(2)																		
(3)																	$\alpha$	
(4)																		$\beta$
(5)																		

FIG. 33

FIG. 34

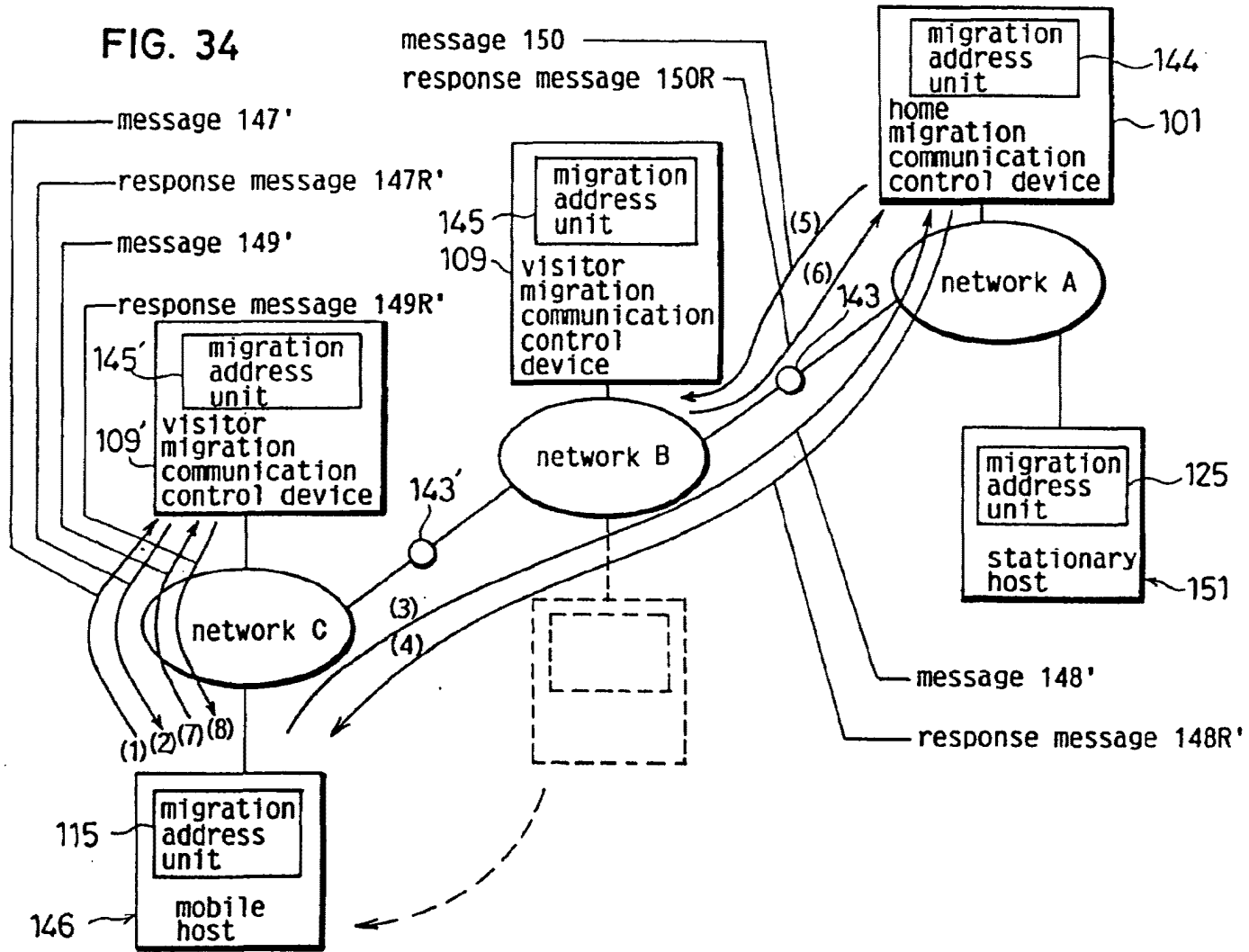
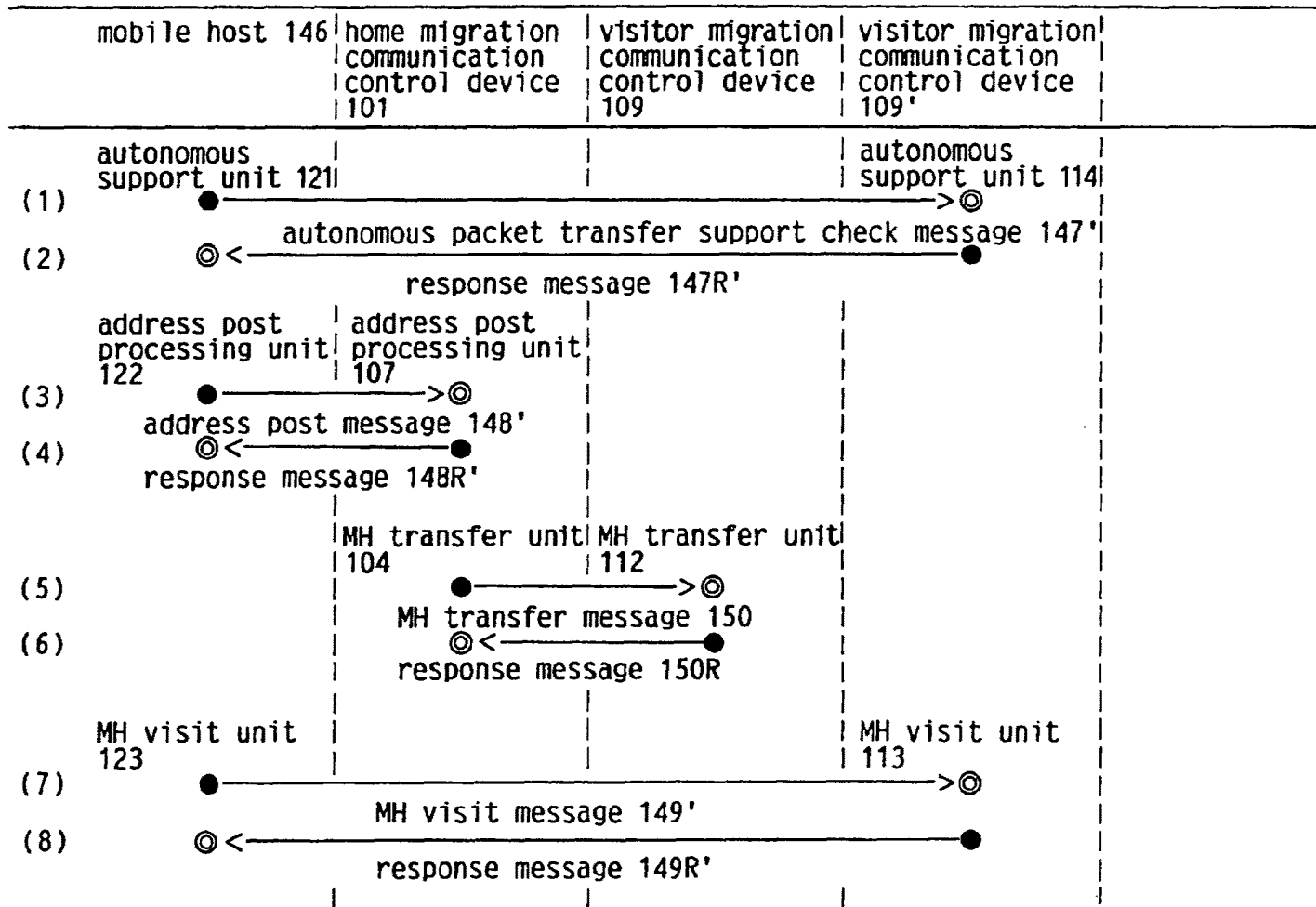




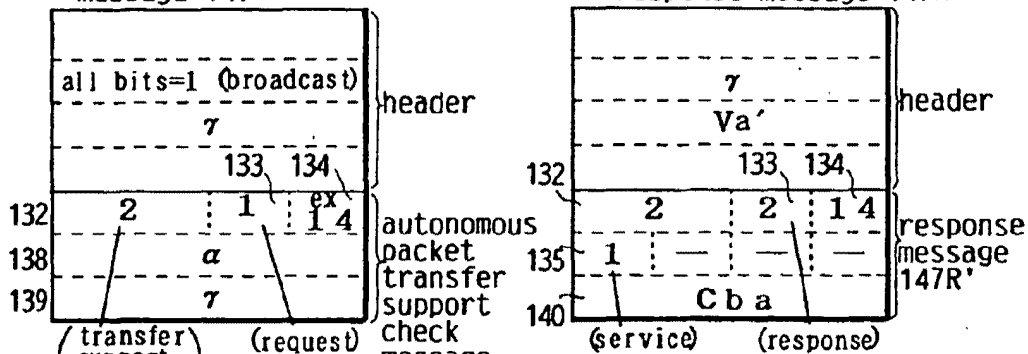
FIG. 35 migration from network B to network C



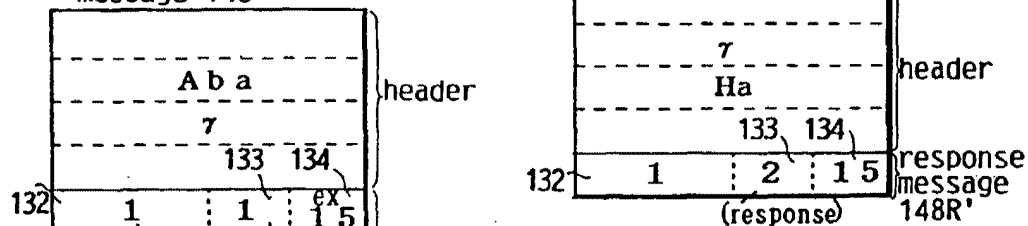
EP 0 556 012 A2

69

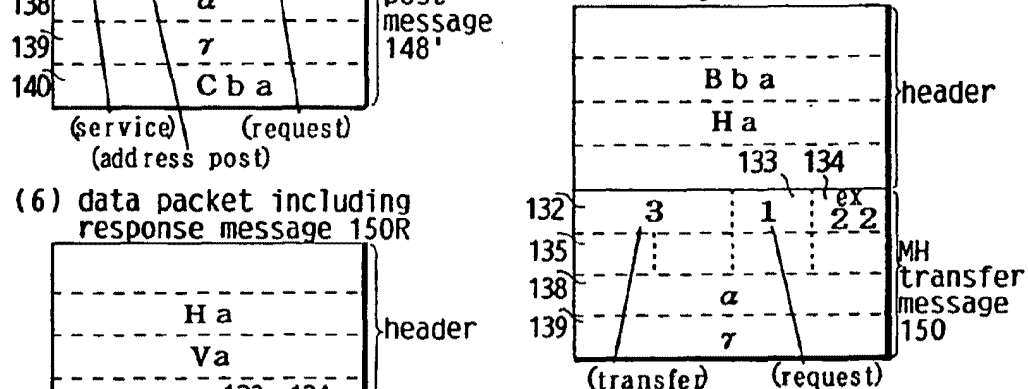
(1) data packet including message 147' (2) data packet including response message 147R'



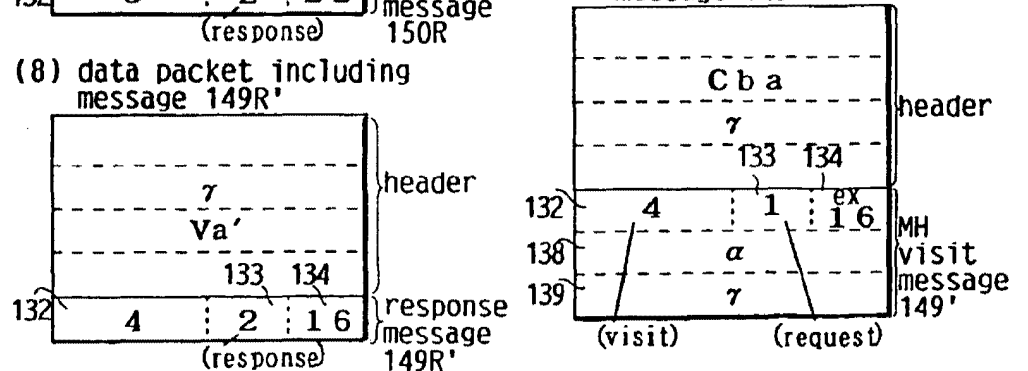
(3) data packet including message 148' (4) data packet including response message 148R'



(5) data packet including message 150 (6) data packet including response message 150R



(7) data packet including message 149' (8) data packet including response message 149R'



address after obtaintment of $j$	address hold unit 119(146)				home MH 1st hold unit 102(101)				visitor MH 1st hold unit 110(109)				visitor MH 1st hold unit 110'(109')				address hold unit 128(151)	
	$\alpha$	Aba	$\beta$	Bba	$\alpha$	$\beta$	1	Bba	$\alpha$	$\beta$	$\beta$	—	—	—	—	$\alpha$	$\beta$	
(1)																		
(2)	$\alpha$	Aba	$\beta$	Bba														
(3)					$\alpha$	$\beta$	1	Cba										
(4)																		
(5)					$\alpha$	$\beta$	$\beta$											
(6)																		
(7)															$\alpha$	$\beta$		
(8)																		

FIG. 37

FIG. 38

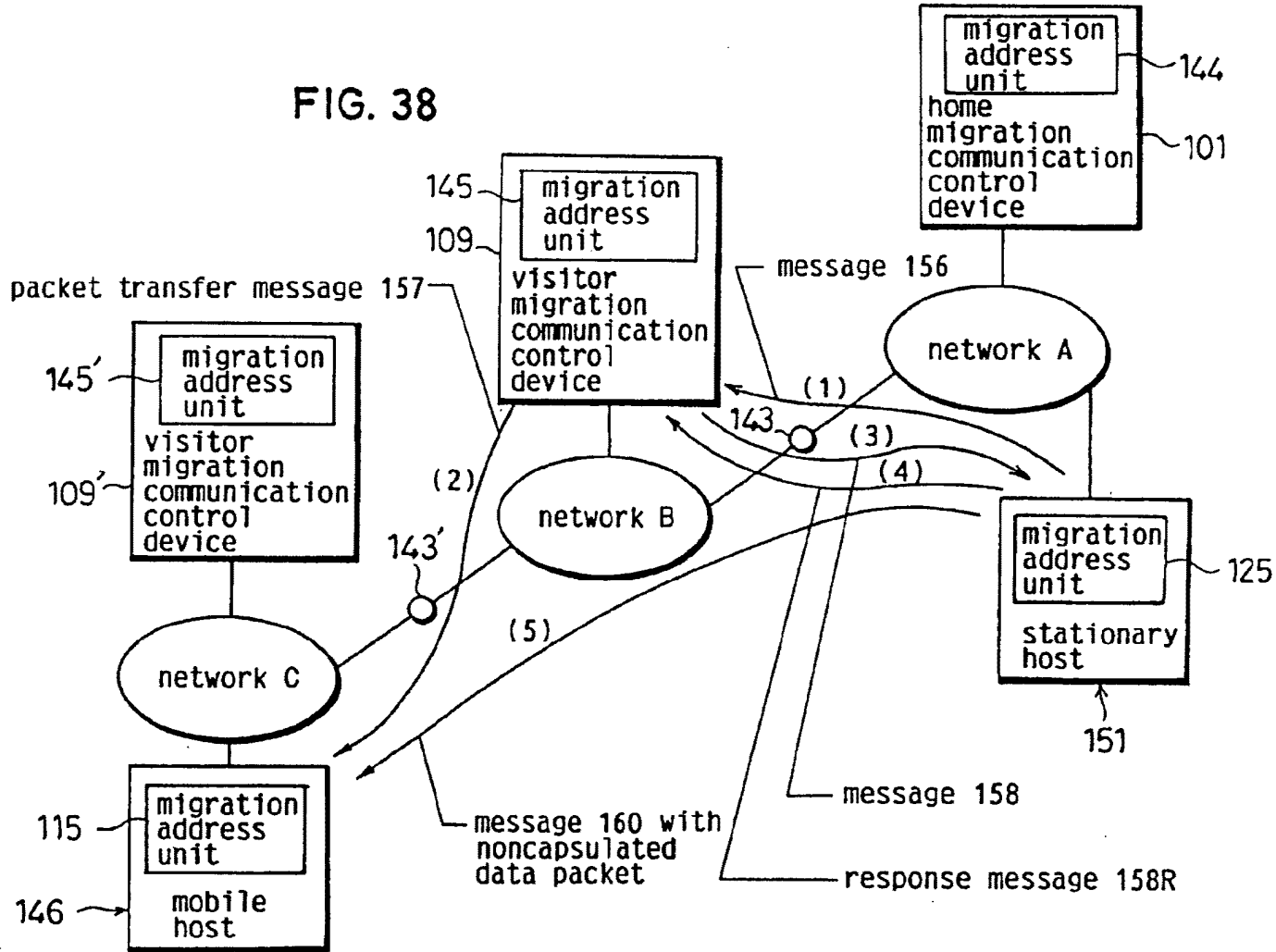
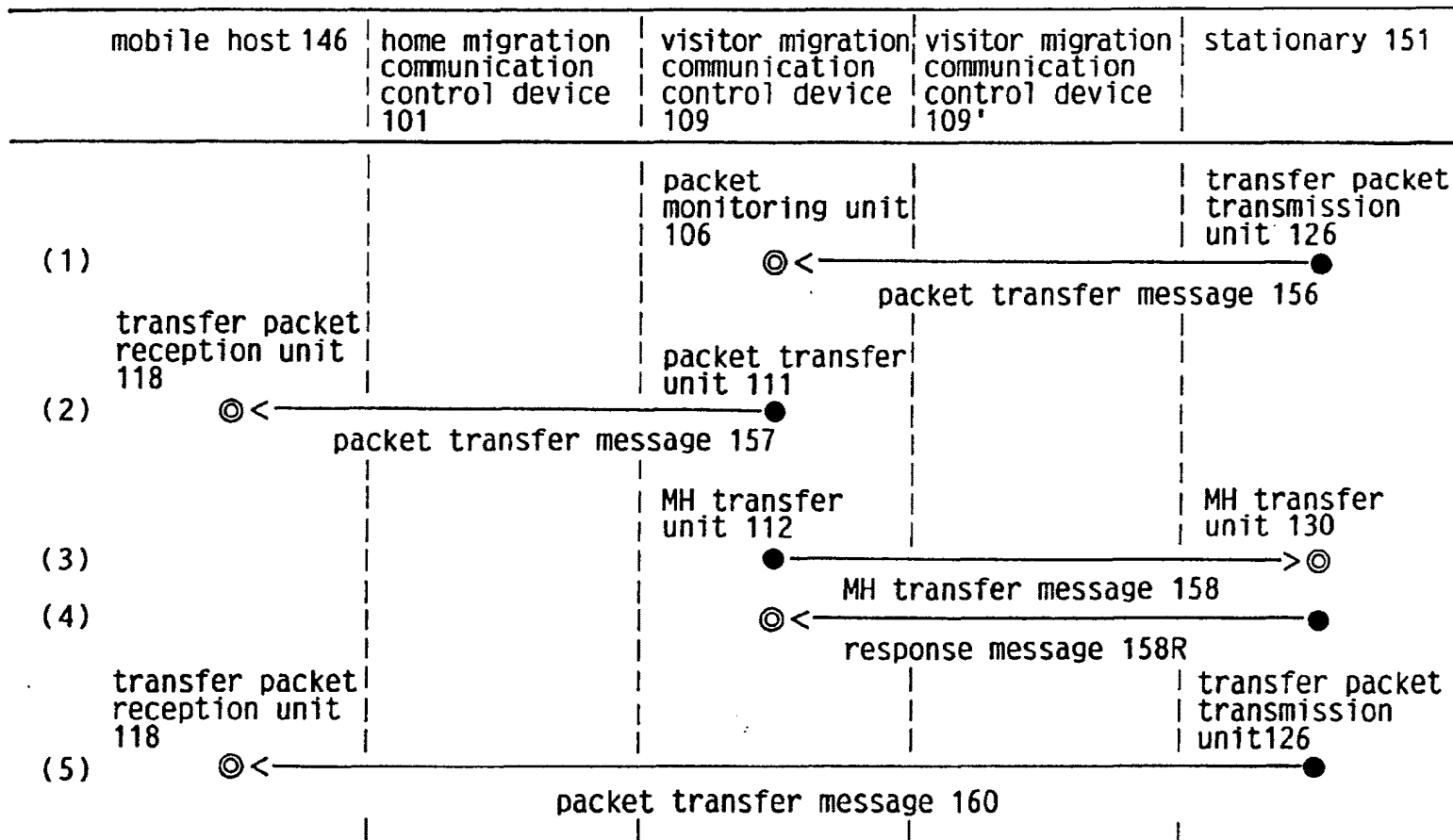


FIG. 39

data packet from SH 151 to MH on network C

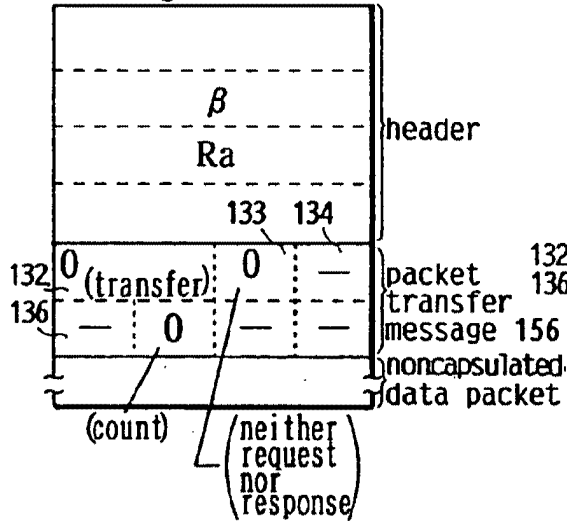


EP 0 556 012 A2

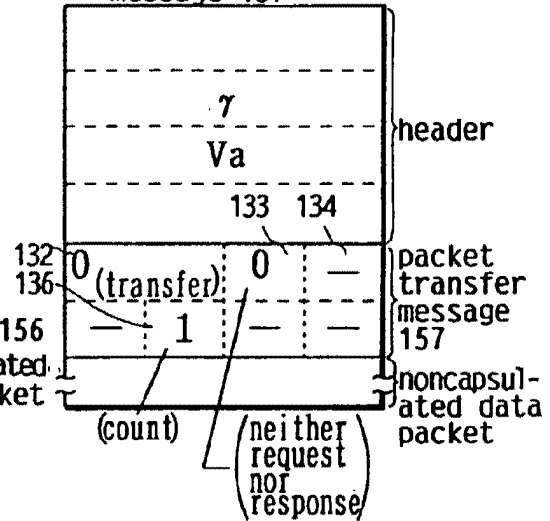
73

FIG. 40

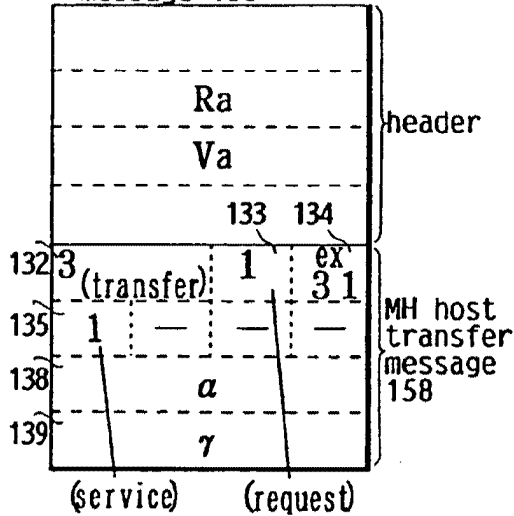
(1) data packet including message 156



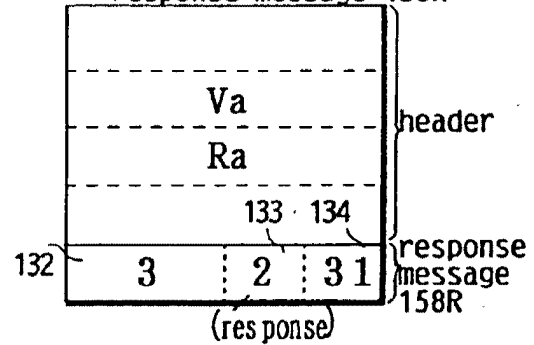
(2) data packet including message 157



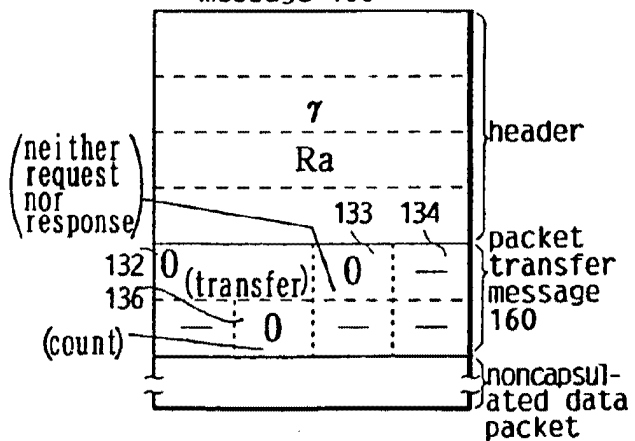
(3) data packet including message 158



(4) data packet including response message 158R



(5) data packet including message 160



address before communica- tion	$\alpha$	home address	address hold unit 119	
		Aba		broadcast address of home network
		$\gamma$		current temporary address
		Cba		current broadcast address
	(1)	$\alpha$	MH's home address	home MH 1st hold unit 102
(2)	$\gamma$	MH's current temporary address		
(3)	1	autonomous flag F		
(4)	Cba	broadcast address		
(5)	$\alpha$	MH's home address	visitor MH 1st hold unit 110	
	$\beta$	temporary address		
	$\gamma$	temporary address after migration		
	1	autonomous flag F		
	$\alpha$	MH's home address	visitor MH 1st hold unit 109	
	$\gamma$	temporary address		
	$\gamma$	temporary address after migration		
	-	autonomous flag F		
	$\alpha$	MH's home address	address hold unit 128	
	$\beta$	MH's temporary address		

FIG. 41

FIG. 42

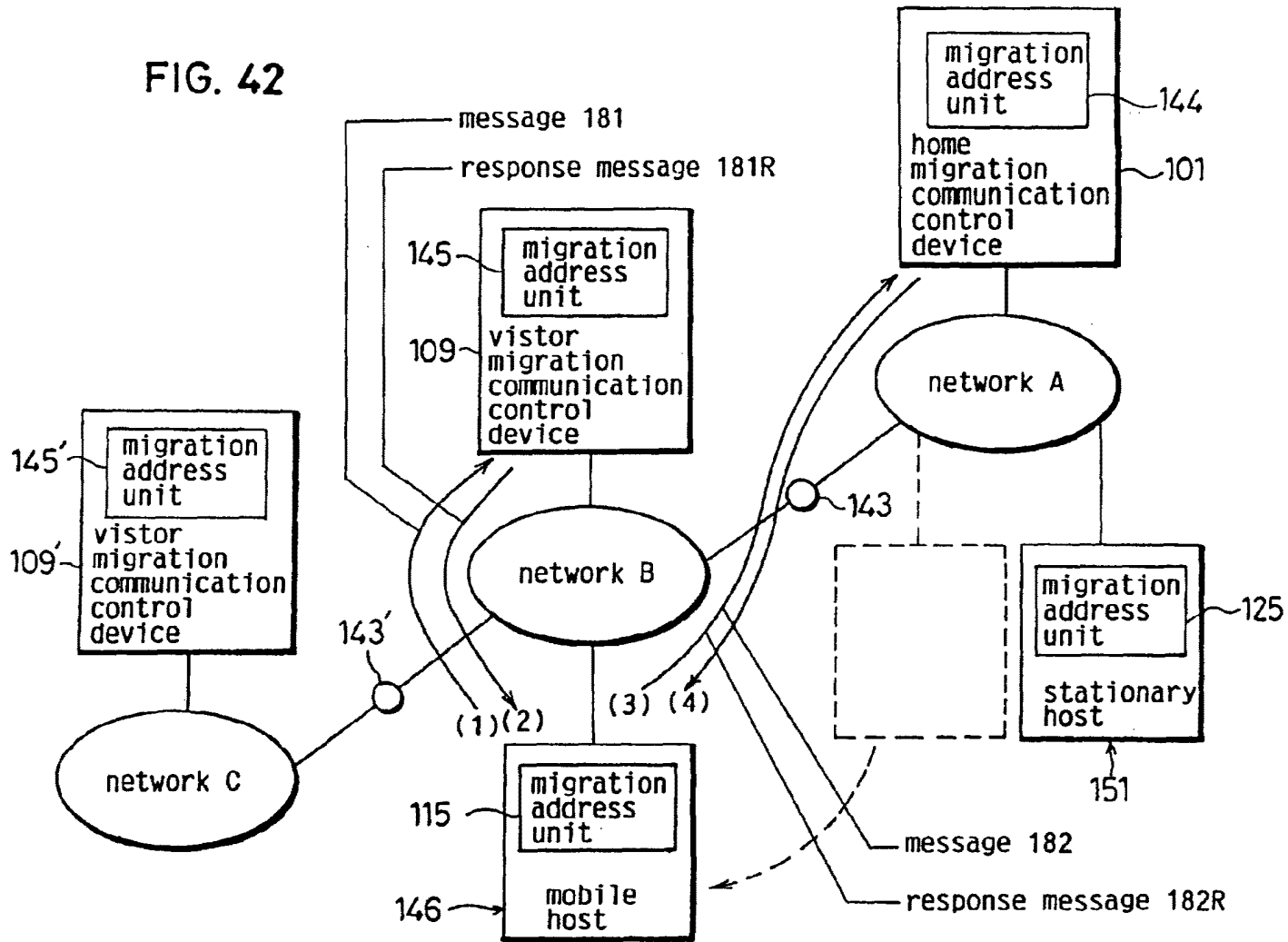
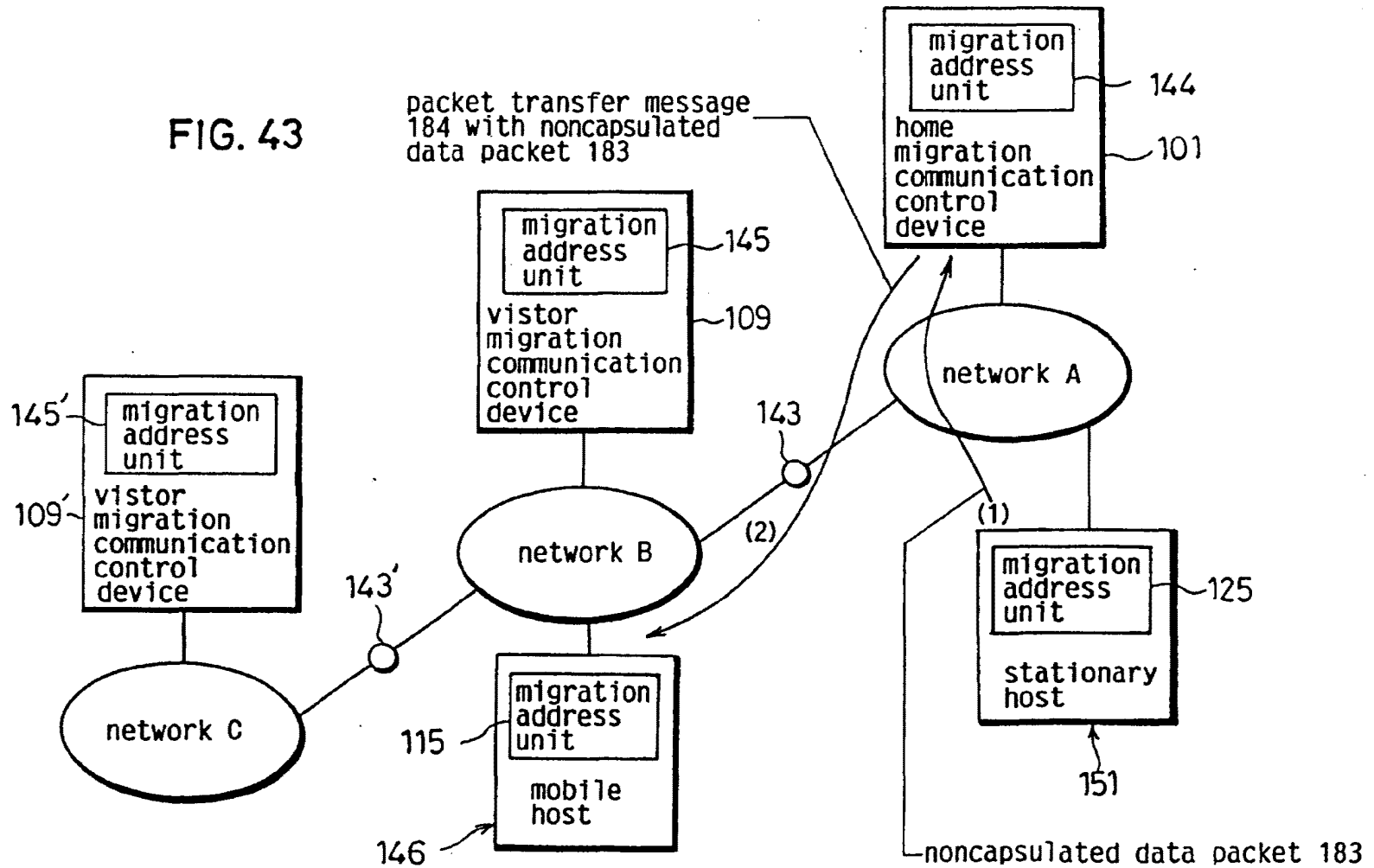




FIG. 43



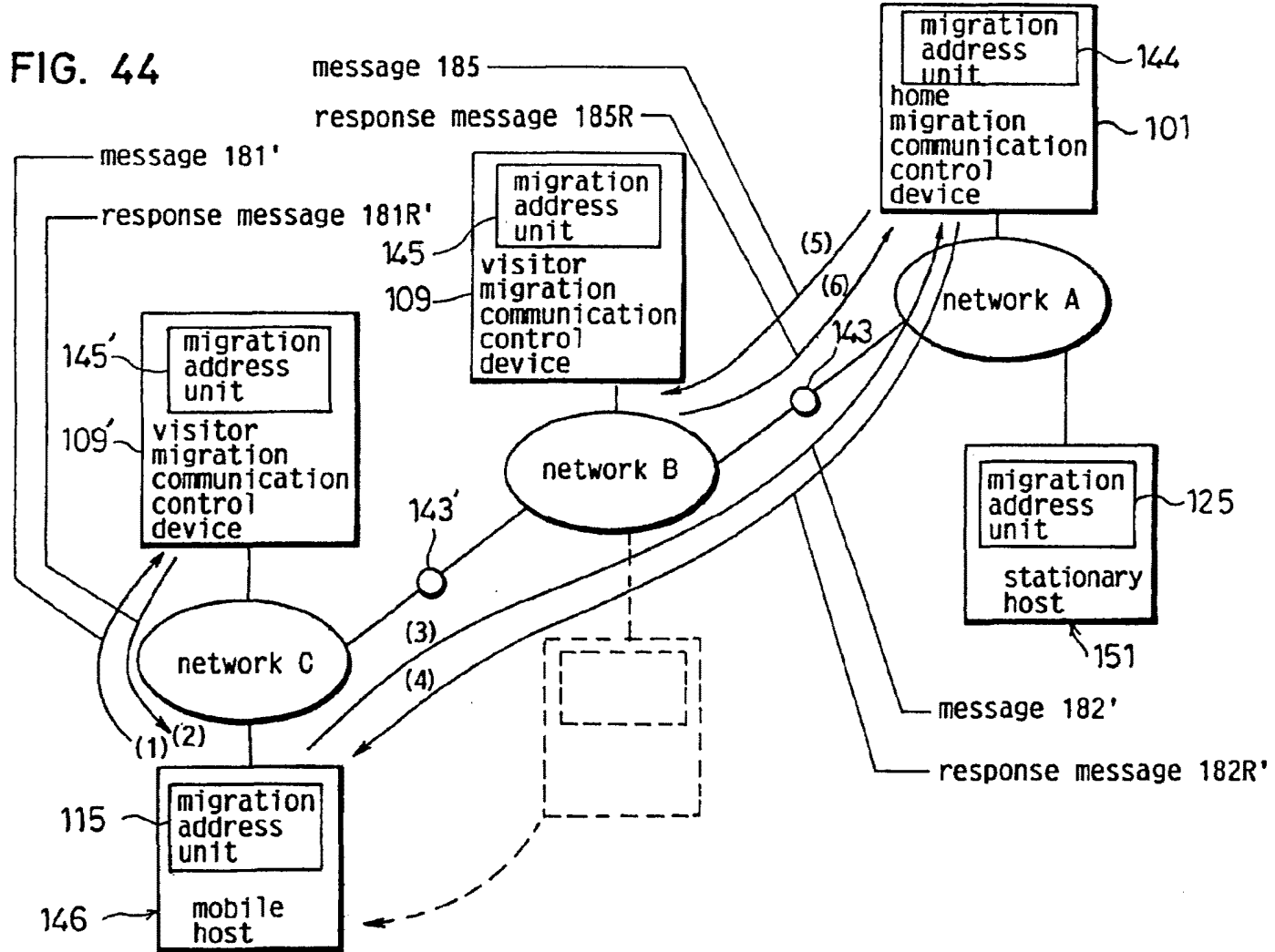
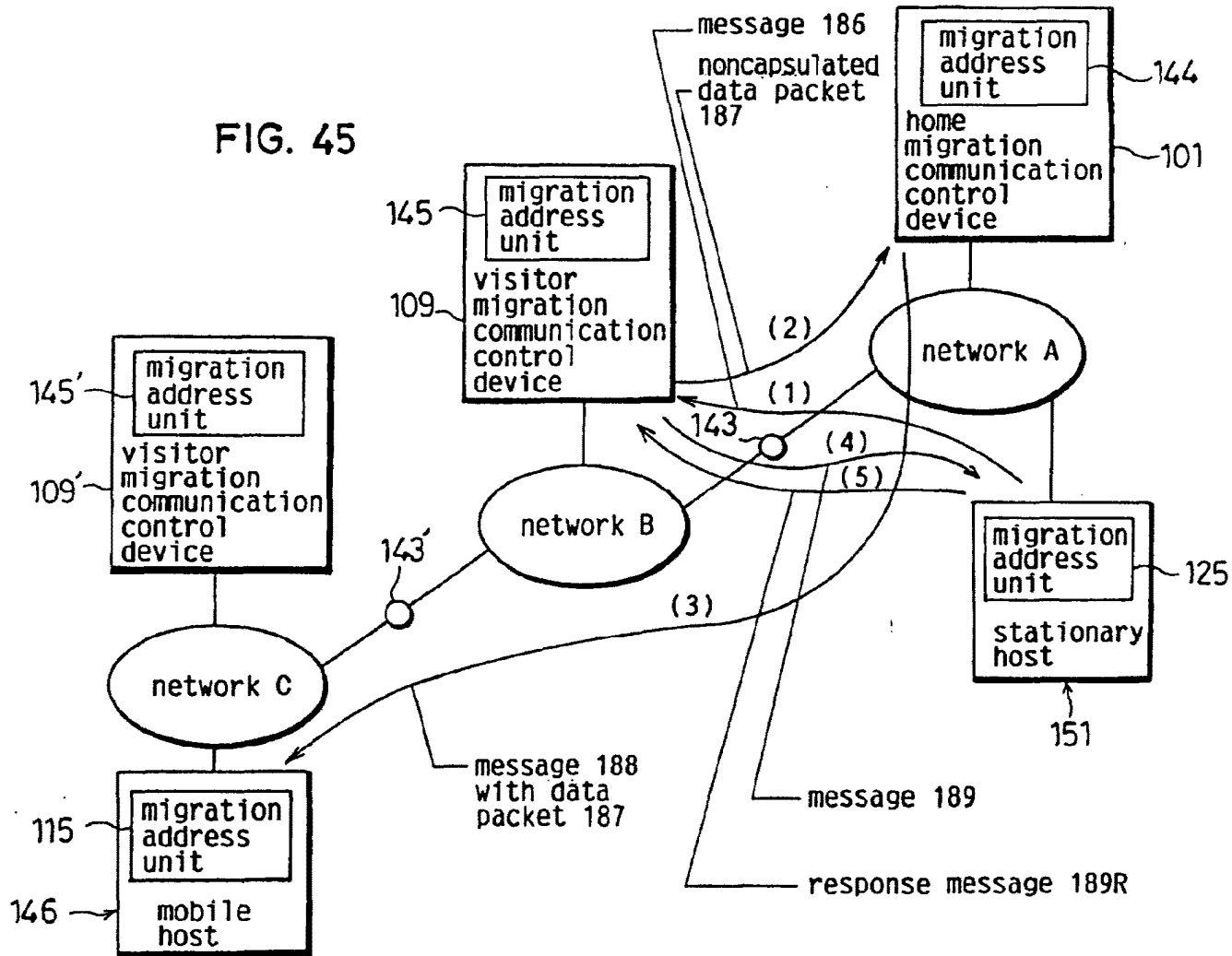


FIG. 45





**(12) EUROPEAN PATENT APPLICATION**

**(21) Application number : 91303643.0**

**(51) Int. Cl.<sup>5</sup> : H04L 12/24**

**(22) Date of filing : 23.04.91**

**(30) Priority : 03.05.90 US 519187**

**(72) Inventor : Wu, Jeff C.  
2630 Wapiti Road  
Fort Collins, Colorado 80525 (US)**

**(43) Date of publication of application :  
06.11.91 Bulletin 91/45**

**(74) Representative : Colgan, Stephen James et al  
CARPMAELS & RANSFORD 43 Bloomsbury  
Square  
London WC1A 2RA (GB)**

**(84) Designated Contracting States :  
DE FR GB**

**(71) Applicant : Hewlett-Packard Company  
Mail Stop 20 B-O, 3000 Hanover Street  
Palo Alto, California 94304 (US)**

**(54) Automatic discovery of network elements.**

**(57)** Disclosed is a computer network node discovery system that provides a general way of discovering network elements, or nodes, connected to a computer network, and a specific algorithm for discovering nodes connected to a TCP/IP network, using the SNMP protocol available within the TCP/IP network software. Some nodes on a network, called discovery agents, can convey knowledge of the existence of other nodes on the network. The network discovery system queries these agents and obtains the information they have about other nodes on the network. It then queries each of the nodes obtained to determine if that node is also a discovery agent. In this manner, most of the nodes on a network can be discovered. The process of querying discovery agents to obtain a list of nodes known to the discovery agents is repeated at timed intervals to obtain information about nodes that are not always active. In a TCP/IP network, discovery agents are nodes that respond to queries for an address translation table which translates internet protocol (IP) addresses to physical addresses. The data from each node's address translation table is used to obtain both the IP and the physical address of other nodes on the network. These nodes are then queried to obtain additional information. After all the nodes on a network are discovered, the list of nodes is written to a database where it can be displayed by the network manager or other users of the network.

**EP 0 455 402 A2**

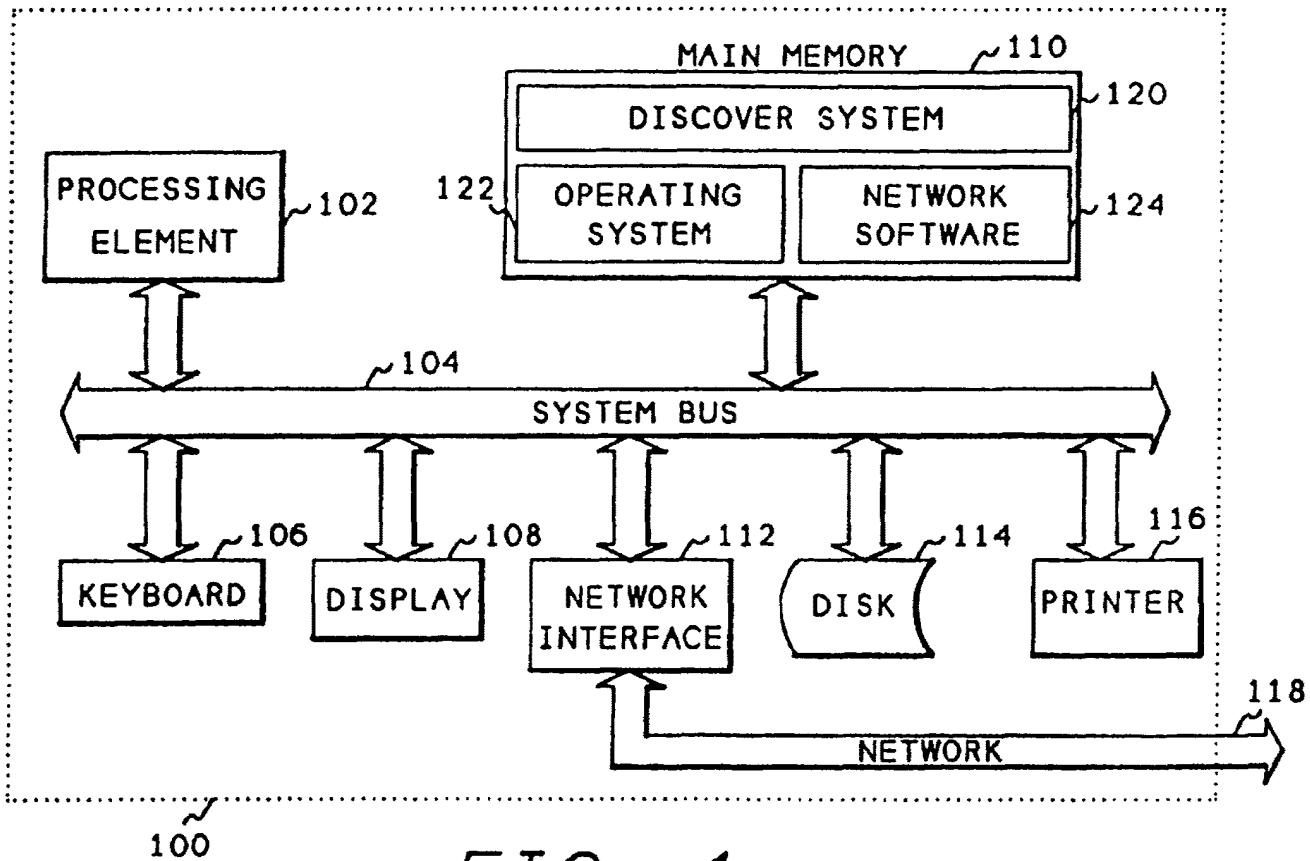


FIG. 1

### FIELD OF THE INVENTION

This invention relates to computer systems and more particularly to computer networks that interconnect computers. Even more particularly, the invention relates to determining the nodes connected to a network.

### BACKGROUND OF THE INVENTION

Computer networks are collections of hardware and software that connect computers and allow them to send information from one computer to another electronically. A computer network is comprised of the physical hardware connections between the various computers, for example telephone lines or a coax cable, and the software used to send and receive data and to route the data to the selected computer on the network.

A local area network (LAN) is a network connection between computers in close proximity, typically less than one mile, and usually connected by a single cable such as coax cable. A wide area network (WAN) is a network of computers located at longer distances, often connected by telephone lines or satellite links. Network software may sometimes be used with both types of networks. For example, a popular network is the Department of Defense Internetworking protocol suite, known as Transmission Control Protocol/Internet Protocol (TCP/IP). This system was originally developed by the Defense Advanced Research Projects Agency (DARPA) and has now been widely distributed to Universities and industry.

When a network is fast growing, that is, network elements or nodes are being added frequently, a network administrator may not know all of the nodes connected to the network. Also, a network administrator new to his or her job may not be familiar with the nodes on the network. Determining the nodes manually is a difficult problem. The administrator may contact all the users of the network known to the administrator, however, infrequent users may be forgotten and not contacted. Also, if a node is connected to the network, but not active because the computer is not powered up or is inoperative, that node may not be included in the list. In a very short local area network, a network administrator may physically trace the cable of the network to determine which nodes are located on the network. However, since longer local area networks can extend as far as a mile, through many floors and offices within a building, physical tracing may be impossible. In a wide area network, physical tracing is almost always impossible.

For some commonly used networks, special equipment can be purchased that will determine the nodes located on the network and the distance between them. This equipment, called a probe, is often limited by the other components of the network, how-

ever. For example, in a local area network, a repeater unit may be used to extend the effective distance of the local area network to a distance greater than is capable with a single cable. A repeater unit amplifies signals, and therefore will not allow a probe to determine the location of nodes beyond the repeater.

Other units connected to the network may obscure nodes. For example a bridge unit connects two similar networks but only passes messages that are being sent from a node on one side of the bridge to a node on the other side of the bridge. It will not pass messages between nodes on the same side, in order to reduce the traffic on the other side of the bridge. A bridge will prevent a probe from determining the nodes on the other side of the bridge. A gateway is a unit that connects dissimilar networks to pass messages. Because a gateway may have to reformat a message to accommodate a different network protocol, it will prevent a probe from finding nodes beyond the gateway.

There is need in the art then for a method of determining the nodes on a local area network. There is further need in the art for determining such nodes without the use of special equipment. A still further need is for a method that will determine which nodes are located beyond the repeater units, bridges, and gateways on a network.

### SUMMARY OF THE INVENTION

It is an object of the present invention to provide a method of determining the elements or nodes connected to a network.

It is another object of the invention to provide a method of discovering network nodes on a TCP/IP network.

Another object of the invention is to determine which discovered nodes are discovery agents and can convey knowledge of the existence of other nodes on the network.

Another object is to query all discovery agents and ask for other nodes on the network

A further object is to query all TCP/IP nodes to retrieve the address translation table from the TCP/IP node.

The above and other objects of the invention are accomplished in a system which provides a general way of discovering network elements, or nodes, and a specific algorithm for discovering nodes within a TCP/IP network, using a standard Simple Network Management Protocol (SNMP), which is available within the TCP/IP network.

Some nodes on a network can convey knowledge of the existence of other nodes on a network, and are called discovery agents. When a network contains discovery agents, these agents can be queried to obtain the information they have about other nodes on the network. By obtaining a list of nodes from a single

discovery agent, and querying each of the nodes obtained to determine if it is also a discovery agent, most of the nodes on a network can be discovered.

The process of querying discovery agents to obtain a list of nodes known to be discovery agents, must be repeated at timed intervals. At any given time on a network, one or more nodes may not be responding to the network, either because it is inoperative, or because it is not powered up. Therefore, if the discovery process is attempted during this time, these unavailable nodes will not be discovered. By repeating the discovery process over time at regular intervals, additional nodes on a network can be discovered.

In a TCP/IP network, discovery agents are nodes that respond to queries for an address translation table. Within TCP/IP network, every node will have an internet protocol (IP) address. This address is a 32 bit number and is unique to all nodes within the TCP/IP network. Although the IP address is probably unique to all nodes everywhere that use the TCP/IP protocol, the physical address of a node on a particular network will be different from the IP address. For example, some types of LANs use an 8 bit address, and can therefore use the low order 8 bits of the IP address, however, some other types of LANs use a 48 bit address and cannot use the internet address. Therefore, every node within a TCP/IP network must have an address translation table which translates the IP address to the physical address. The data from each node's address translation table can be used to obtain both the IP and the physical address of other nodes on the network. Again, as described in the above general algorithm, the queries should be repeated at timed intervals to insure that recently activated nodes are discovered. Another reason for repeating the discovery process over timed intervals in a TCP/IP network is that some of the information within a node's address translation table may be purged if the node does not use the information after a period of time. This purge is used to reduce the table size requirements within a node. By repeating the queries at timed intervals, the greatest amount of translation table information may be obtained.

#### BRIEF DESCRIPTION OF THE DRAWINGS

The above and other objects, features, and advantages of the invention will be better understood by reading the following more particular description of the invention, presented in conjunction with the following drawings, wherein:

Fig. 1 shows a block diagram of the hardware of the node that runs the process of the present invention;

Fig. 2 shows a diagram of a typical computer interconnection network;

Figs. 3 through 5 show a hierarchy diagram of the

modules of the discovery system of the present invention;

Fig. 6 shows a flowchart of the main module of the invention;

Fig. 7 shows a flowchart of the self-seed module of the invention;

Fig. 8 shows a flowchart of the process-node module of the invention;

Fig. 9 shows a flowchart of the process-ping module of the invention;

Fig. 10 shows a flowchart of the process-IFIP module of the invention;

Fig. 11 shows a flowchart of the store-IP module of the invention;

Fig. 12 shows a flowchart of the store-IF module of the invention;

Fig. 13 shows a flowchart of the invalidnode module of the invention;

Fig. 14 shows a flowchart of the findnode module of the invention;

Fig. 15 shows a flowchart of the addnode module of the invention;

Fig. 16 shows a flowchart of the process-AT module of the invention; and

Fig. 17 shows a flowchart of the store-AT module of the invention.

#### DESCRIPTION OF THE PREFERRED EMBODIMENT

The following description is of the best presently contemplated mode of carrying out the present invention. This description is not to be taken in a limiting sense but is made merely for the purpose of describing the general principles of the invention. The scope of the invention should be determined by referencing the appended claims.

Fig. 1 shows a block diagram of the computer hardware that contains the discovery system of the present invention. Referring now to Fig. 1, a computer system 100 contains a processing element 102. The processing element 102 communicates to other elements within the computer system 100 over a system bus 104. A keyboard 106 is used to input information from a user of the system, and a display 108 is used to output information to the user. A network interface 112 is used to interface the system 100 to a network 118 to allow the computer system 100 to act as a node on a network. A disk 114 is used to store the software of the discovery system of the present invention, as well as to store the data base collected by the discovery system. A printer 116 can be used to provide a hard copy output of the nodes of the network discovered by the discovery system. A main memory 110 within the system 100 contains the discovery system 120 of the present invention. The discovery system 120 communicates with an operating system 122 and network software 124 to discover the nodes on the

network 118.

Fig. 2 shows a diagram of a network. Referring now to Fig. 2, a network 202 contains a node 206. Node 206 contains the processor 100 (Fig. 1) which contains the discovery system software of the present invention. Node 206 is attached to a first network segment 118. The network segment 118 is connected to a repeater 212 which is connected to a second network sequent 214. This second network system 214 has nodes 216 and 218 attached to it. A repeater, such as repeater 212, allows network sequents to be connected to allow a network to be extended over a longer distance. An important characteristic of a repeater is that there is no translation of data passing through it. That is, every message that is transmitted on one network segment, will pass unchanged through a repeater to the other network segment. Therefore, any messages broadcast, for example, by node 206 will be received by node 216 and node 218 after these messages pass through repeater 212.

Network segment 118 is also attached to a bridge 208 which connects it to a third network sequent 210. A bridge will only pass messages that are being transmitted from a node on one side of the bridge to a node on the other side of the bridge. It will block messages that are transmitted from a node on one side of the bridge to a node on that same side of the bridge. This characteristic reduces network traffic on various sequents of a network.

Segment 118 is also attached to a router/gateway 220 which connects is to a fourth network segment 222. Routers are devices that connect network segments which have similar characteristics. Gateways are devices which connect networks having different types of characteristics. For example, a gateway might connect a local area network to a wide area network.

Because bridges, routers, and gateways, must process the messages sent over the network, they also must contain information about which nodes are on the network. Therefore, bridges, routers, and gateways are authoritative sources of information for determining the nodes on the network. A protocol defines the format of messages that are sent across a network. One popular protocol is the Department of Defense Internetworking Protocol Suite, popularly known as TCP/IP. Because it was developed by the Department of Defense, this protocol is widely available and used extensively, particularly in a university environment. Also, this suite of protocols is very popular on the UNIX operating system and has seen wide distribution there. The internet protocol (IP) uses a single thirty-two bit address for all nodes that can be connected to the internet at any location. Physical addresses within a particular type of network, are normally different from an IP address. If a network address is very small, perhaps eight bits, it may be the same as the low order eight bits of the IP address. If

a network address is large, for example, some LANs use forty-eight bit addresses, it is impossible for these addresses to correspond directly to IP addresses. Therefore, both an IP address and a physical address exist for each node on a network. Devices such as routers, gateways, and bridges, which can send messages from one network to another must be able to translate between IP addresses and physical addresses. Therefore, these devices have translation tables which allow them to translate between these two types of addresses. By accessing these translation tables, one of the nodes on a network can obtain information about the other nodes on the network. The existence of these translation tables allow the method of the present invention to perform its function.

A network probe 224 is also attached to the network 118. A network probe 224 is a device that assists in locating defective nodes and assists in repairing those nodes. Since it is a testing device, it may or may not be attached to a network at any given time. When a probe is attached to a network, the discovery system of the present invention can query the probe and use information obtained from the probe to assist in discovering other nodes on the network.

Figs. 3 through 5 show a hierarchy diagram of the modules of the software of the present invention. Referring now to Figs. 3 through 5, discovery module 302 is the main module of the system. Discovery calls self-seed block 304 to start the process of building a database about the network, and it calls process-node block 306 to process information about each node that it obtained from self-seed. Process-node block 306 calls process-ping block 308 to query a node on the network to determine if that node is active. Process-node block 306 also calls process-IFIP block 310 for each IP address that it obtains. Process-IFIP block 310 calls store-IP block 402 for each IP address, and store-IP block 402 calls invalidnode block 406, findnode block 408, and addnode block 410, for each IP address. For each IF entry (physical address) received, process-IFIP block 310 calls store-IF block 404. For each address translation table entry, process-node block 306 calls process-AT block 312 which in turn calls store-AT block 502. Store-AT block 502 calls invalidnode at block 504, findnode block 506, and addnode block 508.

Fig. 6 shows a flowchart of the discovery module block 302 (Fig. 3). Referring now to Fig. 6, after entry block 602 gets any options that the user wishes to enter. Block 604 then initializes the database used to permanently store the nodes, and loads node list from existing entries in the database. If a database for the network does not exist, the discovery system has the ability to create that database. If a database of the network already exists, the discovery system will use the node information which is already available in that database to query other nodes within the system.

Block 606 then initializes domains. A domain



defines the limit beyond which the user of the discovery system does not wish to find nodes. That is, the domain limits the range of the discovery process. This limitation is necessary on large networks, to keep the amount of processing to reasonable level. Furthermore, a user usually is only interested in the nodes on a particular network segment, or the network segment connected by repeaters and possibly bridges.

Block 608 then calls Fig. 7 to self-seed the system. If no entries were available in the database, the discovery system can self-seed by sending a broadcast message and determine who responds to that message. After returning from self-seed, block 610 points to the first node list entry. As discussed earlier, the node list will contain a list of the nodes already known to the system. This list can be input from the database, or the list can be started from self-seed module. After pointing to the first entry, block 612 determines if there are more entries to process. If there are no more entries to process, block 612 transfers to block 614 which will wait a predetermined period of time before reprocessing the entire node list. Typically, block 614 will wait for approximately thirty seconds. By reprocessing the node list periodically, additional nodes can be discovered. This is because a node may be inactive on the system at any given time and might not be discovered by a single pass through the network. By waiting and reprocessing the node list, nodes that were inactive may now be active and additional information can be obtained.

If more entries in the node list exist, block 612 transfers to block 616 to process one of the nodes. After processing that node, block 616 transfers to block 618 which points to the next node list entry and returns to block 612 to process the next node.

Fig. 7 shows a flowchart for the self-seed block 304 (Fig. 3) which obtains initial information about nodes on the network. Referring now to Fig. 7, after entry, block 702 sends an SP broadcast request to all nodes on the network. SNMP stands for Simple Network Management Protocol, and is a part of the TCP/IP network software. After sending the broadcast request, block 702 transfers to block 704 which receives SNMP messages from the nodes. If more SNMP messages are available, block 704 transfers to block 706 which adds a node to the node list for each message received. In this manner, all nodes that are currently active on the network can be queried to obtain initial information about the node. After all SNMP messages have been received, block 704 returns to the caller.

Another way of self-seeding is to query the address translation table for the node that is executing the discovery system. This table will contain the addresses of other nodes on the network, and these addresses are then used to start the discovery process.

Fig. 8 is a flowchart of the process-node block 306

(Fig. 3). The process-node module of Fig. 8 is called from the discovery module of Fig. 6 once for each entry in the node list. Therefore, when Fig. 8 is called, the address of a single node is passed to it. Referring now to Fig. 8, after entry, block 802 determines whether the node is within a domain. As discussed earlier, the domain defines the limits beyond which the discovery program does not wish to discover new nodes. If the node is within the domain, block 802 transfers to block 804 which calls the process-ping module of Fig. 9 to determine whether the node is active. After returning from Fig. 9, block 804 transfers to block 806 to determine whether the state of the node has changed since the last information was obtained. That is, when the process-ping module queries the node, it determines the state of the node at the present time. This state is compared, in block 806, with the state of the node as it was known previously in the database. If that state has changed, block 806 transfers to block 808 to store the new state in the database. Control then returns to block 810 which calls process-IFIP to retrieve the IF and IP tables from the node. After returning from Fig. 10, block 810 transfers to block 812 which determines whether the node responded to an SNMP request. If the node did respond to the SNMP request, block 812 transfers to block 814 which determines whether the node is currently in the database. If the node is not in the database, block 814 transfers to block 816 to add the node to the database. Control then continues at block 818 which calls Fig. 16 to retrieve the address translation table from the node. Control then returns to the caller.

Fig. 9 shows a flowchart of the process-ping module block 308 (Fig. 3). This module is called to determine whether a node is active on the network. Referring now to Fig. 9, after entry block 902 determines whether the ping interval has elapsed. The ping interval is used to prevent a node from being queried too often. If the ping interval has not elapsed, block 902 returns to the caller. If the ping interval has elapsed, block 902 transfers to block 904 which sends an ICMP-echo message to the node. The ICMP-echo protocol is defined as a part of TCP/IP and is used to cause the node to return an acknowledgement to a message. Block 904 then transfers to block 906 which determines whether a response has been received from the other node. If a response has not been received within a predetermined amount of time, typically block 906 transfers to block 910 which sets a flag to indicate that the node failed to respond. If the node does respond, block 906 transfers to block 908 which sets a flag to indicate that the node did respond and then block 912 sets a new ping interval which will prevent the node from being pinged for the period of the interval. The ping interval is typically five minutes. Block 912 then returns to the caller.

Fig. 10 shows a flowchart of the process-IFIP module block 310 (Fig. 3). The IF and IP tables are

available in a node to define the translation of physical addresses to IP addresses. The information is available as two different tables, with an index contained in the IF table to cross-reference to the IP table within the node. By obtaining these two tables, the discovery system can determine what the other interfaces to which a node is connected, and therefore determine other networks to which the node is connected. Referring now to Fig. 10, after entry, block 1002 determines whether the IFIP interval has elapsed. The IFIP interval is similar to the ping interval described with respect to Fig. 9, and is used to keep a node from being queried too often. If the IFIP interval has not elapsed, block 1002 returns to the caller. If the IFIP has elapsed, block 1002 transfers to block 1004 which sends an SNMP message to request the node to send its next IP table entry to the discovery node. When an entry is received, block 1006 calls store-IP module of Fig. 11 to store the node within the node list. Block 1007 then transfers back to block 1004 if more IP entries are available. After all the entries are all stored in the node list, block 1007 transfers to block 1008 which sets a new IFIP interval of typically greater than 10 hours. Block 1010 then sends an SNMP message to request that the node send its next IF table entry to the discovery node. When an IF table entry is received, block 1012 calls the store-IF module of Fig. 12. Block 1014 then transfers back to block 101 if more entries are available. After receiving and storing all the IF table entries, block 1014 returns to the caller. Each IF table entry contains an index into the IP table. By using this index, physical addresses in the IF table can be matched with the IP address.

Fig. 11 shows a flowchart of the store-IP process block 402 (Fig. 4). Referring now to Fig. 11, after entry block 1102 calls Fig. 14 to find the node in the node list. The node will be found if the discovery system has already encountered this node in its process. Block 1304 then determines whether the node exists, and if the node does not exist, block 1104 transfers to block 1106 which calls Fig. 13 to determine whether the node is valid. Block 1108 then determines if the node is valid and if it is valid, block 1108 transfers to block 1110 to add the node to the node list. After adding the node, or if the node already existed, control goes to block 1112 which updates the state information about the node. After updating the node state information or if the node was not valid, Fig. 11 returns to the caller.

Fig. 12 is a flowchart of the store-IF process of block 404 (Fig. 4). This module is called for each table entry in the IF table received from a node. Referring now Fig. 12, after entry, block 1202 finds the IP index within the IF record. As described earlier, each IF table entry will have a corresponding IP table entry, and the IP entry is referenced by an index value contained in the IF entry. Block 1204 then determines whether a matching IP record exists. If a matching IP record does exist, block 1204 transfers to block 1206

which moves the physical address from the IP record to the node record in the node list. Block 1208 then updates any state information in the node record. After updating the state information, or if there were no matching IP record, Fig. 12 returns to its caller.

Fig. 13 shows a flowchart of the invalidnode module block 406 (Fig. 4). Referring now Fig. 13, after entry, block 1302 determines whether the address of the node is simply the loopback address of another node. Each node has a loopback address associated with it for use in testing the node. Because the loopback address refers to the same node, no additional information can be obtained from that node and the loopback address is never stored as a node address. If the IP address is not equal to the loopback address, block 1302 transfers to block 1304 to determine whether the node is within the domain. As described earlier, the domain is used to determine the limits beyond which the discovery system will not attempt to discover new nodes. If the node is within the domain, block 1304 transfers to block 1306 which returns an indication that the node is valid. If the node is not within the domain or if the IP address equals the loopback address, control transfers to block 1308 which returns an error indication indicating that node is not valid. Control then returns to the caller.

Fig. 14 is a flowchart of the findnode module block 408 (Fig. 4). The module is used to find a node within the node list. Referring now Fig. 14, after entry, block 1402 gets the node list entry. Block 1404 then determines whether the IP address matches the entry in the list. If a match does occur, block 1404 transfers to block 1408 which returns an indication that the node is in the node list. If the IP address does not match, block 1404 transfers to block 1406 which gets the next node list entry and block 1410 then determines whether the end of table has been reached. If the end of the list has not been reached, block 1410 transfers back to block 1404 to check the entry just found. If the end of the list has occurred, block 1410 transfers to block 1412 which returns an error indication indicating that the node is not in the node list.

Fig. 15 shows a flowchart of the process of adding a node to the node list. Referring now to Fig. 15, after entry, block 1502 performs a hash operation on the IP address to create a pointer into the node list. Block 1504 then allocates memory for a node record, and block 1506 stores the data available for the node into the node record at the location pointed to by the hashed IP address. Block 1506 then returns to the caller.

Fig. 16 shows a flowchart of the process-AT module of block 312 (Fig. 3). This module is called by the process-node module for each entry in the node list. Referring now to Fig. 16, after entry, block 1602 determines whether the AT interval has expired. The AT interval is used to prevent a node from being polled too frequently. If the AT interval has not expired, block

1602 simply returns to the caller. If the AT interval has expired, block 1602 transfers to block 1604 which sends an SNMP message to request that the node send its next address translation table entry to the discovery node. When an entry is received, block 1606 is called to store the table entry. Block 1607 then transfers back to block 1604 if more table entries are available. After storing all the table entries, block 1607 transfers to block 1608 which updates the node's state information in the node list. Block 1610 then sets a new AT interval, typically fifteen seconds, and returns to the caller.

Fig. 17 shows a flowchart of the store-AT module of block 502 (Fig. 5). Referring now to Fig. 17, after entry, block 1702 calls the findnode module Fig. 14 to determine whether the node is already in the node list. If the node is in the node list, block 1704 transfers to block 1712. If the node is not in the node list, block 1704 transfers to block 1706 which calls Fig. 13 to determine whether the node is a valid node. If the node is not valid, block 1708 returns to the caller. If the node is valid, block 1708 transfers to block 1710 which calls Fig. 15 to add the node to the node list. After adding the node to the node list, or if the node already existed, control transfers block 1712 which updates the state information about the node in the node list before returning to the caller.

In addition to querying nodes on the network, the discovery system can also query any network probes that may be attached to the network. Information about other nodes on the network can be obtained from these probes, and the discovery system can use this information to assist in discovering other nodes on the network.

Having thus described a presently preferred embodiment of the present invention, it will now be appreciated that the objects of the invention have been fully achieved, and it will be understood by those skilled in the art that many changes in construction and circuitry and widely differing embodiments and applications of the invention will suggest themselves without departing from the spirit and scope of the present invention. The disclosures and the description herein are intended to be illustrative and are not in any sense limiting of the invention, more preferably defined in scope by the following claims.

#### Claims

1. A computer network node discovery process (120) for determining nodes (206, 216, 218) connected to a computer network (118), said process (120) comprising the steps of:
  - (a) obtaining (306), from one node of a set of known nodes on said computer network (118), a list of addresses of one or more other nodes with which said one node communicates;
  - (b) repeating step (a) for each of said other nodes obtained; and
  - (c) storing said list of node addresses in a file (808); whereby said list of node addresses may be displayed to a user of said computer network.
2. The process of claim 1 further comprising the step of:
  - (d) repeating steps (a) through (c) at regular time intervals.
3. The process of claim 2 further comprising the step of:
  - (a1) obtaining from each bridge unit (208) connected to said network (118) a list of addresses of all nodes accessible by said bridge unit (208).
4. The process of claim 3 further comprising the step of:
  - (a2) obtaining from each router unit (220) connected to said network (118) a list of addresses of all nodes accessible by said router unit (220).
5. The process of claim 4 further comprising the step of:
  - (a3) obtaining from each gateway unit (220) connected to said network (118) a list of addresses of all nodes accessible by said gateway unit (220).
6. The process of claim 5 further comprising the step of:
  - (a4) obtaining from any network probe device (224) connected to said network (118) a list of addresses of all nodes known to said network probe device (224).
7. A computer network node discovery process (120) for determining nodes connected to a TCP/IP computer network (118), said process comprising the steps of:
  - (a) obtaining (306), from one node of a set of known nodes on said computer network, an address translation table containing a list of addresses of other nodes with which said one node communicates;
  - (b) repeating step (a) for each of said other nodes in said address translation table;
  - (c) storing said list of nodes in a file (808); and
  - (d) repeating steps (a) through (c) at regular time intervals.
8. The process of claim 7 further comprising the steps of:
  - (a1) obtaining from each bridge unit (208) con-

- connected to said network (118) an address translation table containing a list of addresses of nodes accessible from said bridge unit (208);
- (a2) obtaining from each router unit (220) connected to said network (118) an address translation table containing a list of addresses of nodes accessible from said router unit (220);
- (a3) obtaining from each gateway unit (220) connected to said network (118) an address translation table containing a list of addresses of nodes accessible from said gateway unit (220);
- (a4) obtaining from any network probe devices (224) attached to said network (118) a list of addresses of all nodes known to said network probe (224); and
- (a5) obtaining from each node in said network (118) an interface table and an internet protocol table which defines other networks and nodes to which said node is connected.
9. A computer network node discovery process (120) for determining nodes connected to a computer network (118), said process comprising the steps of:
- (a) sending a general response message (307) to all nodes on said network;
- (b) creating a node list (410) containing the address of each node responding to said general response message;
- (c) obtaining (306), from each node in said node list, a second list of addresses of other nodes with which said node communicates;
- (d) adding each node (410) in said second list to said node list;
- (e) repeating steps (c) through (d) for each of said nodes in said second list;
- (f) storing said node list in a file (808); and
- (g) repeating steps (a) through (f) at regular time intervals.
10. The process of claim 9 further comprising the steps of:
- (c1) obtaining from each bridge unit (208) connected to said network (118) a list of addresses of all nodes accessible by said bridge unit (208);
- (c2) obtaining from each router unit (220) connected to said network (118) a list of addresses of all nodes accessible by said router unit (220);
- (c3) obtaining from each gateway unit (220) connected to said network (118) a list of addresses of all nodes accessible by said gateway unit (220); and
- (c4) obtaining from any network probe devices (224) attached to said network (118) a list of addresses of all nodes known to the network probe (224).

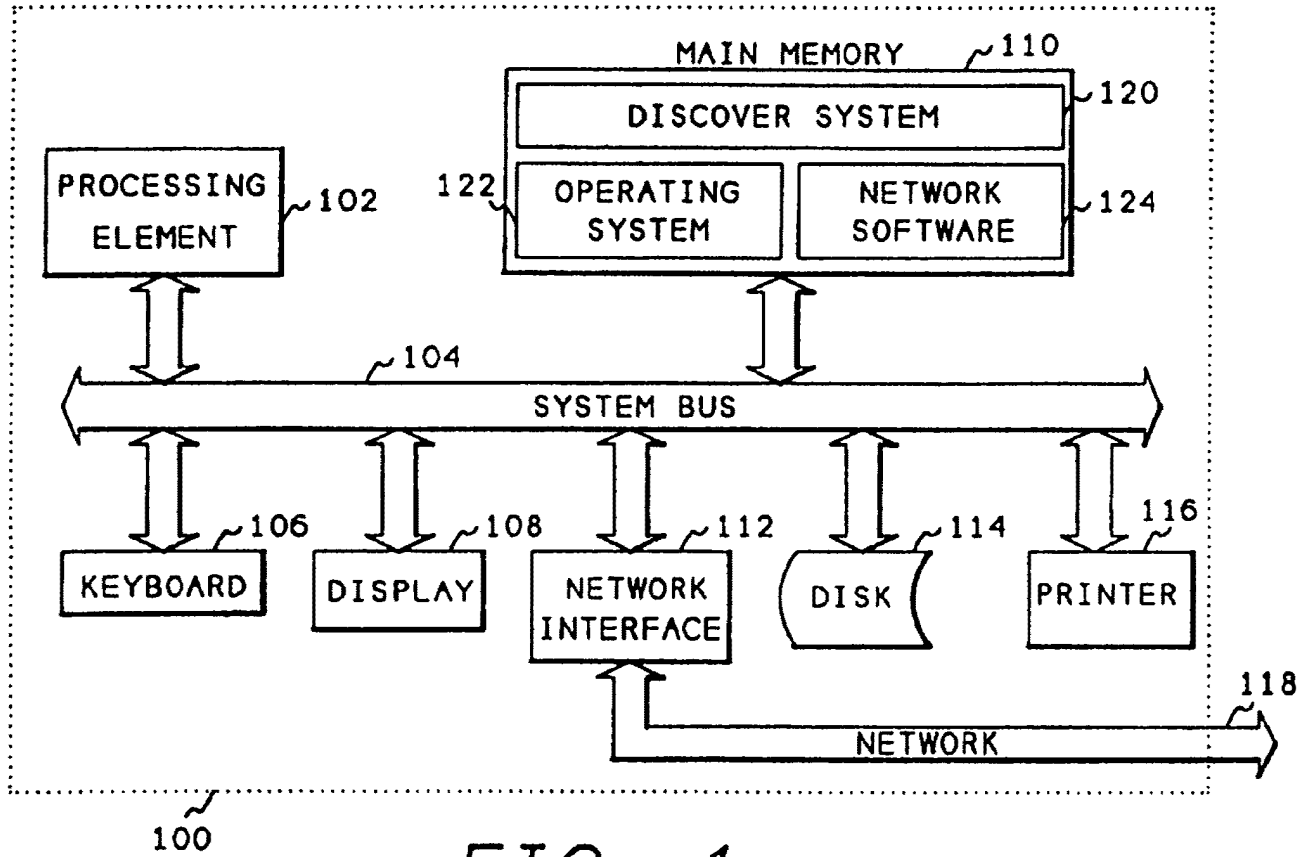


FIG. 1

10

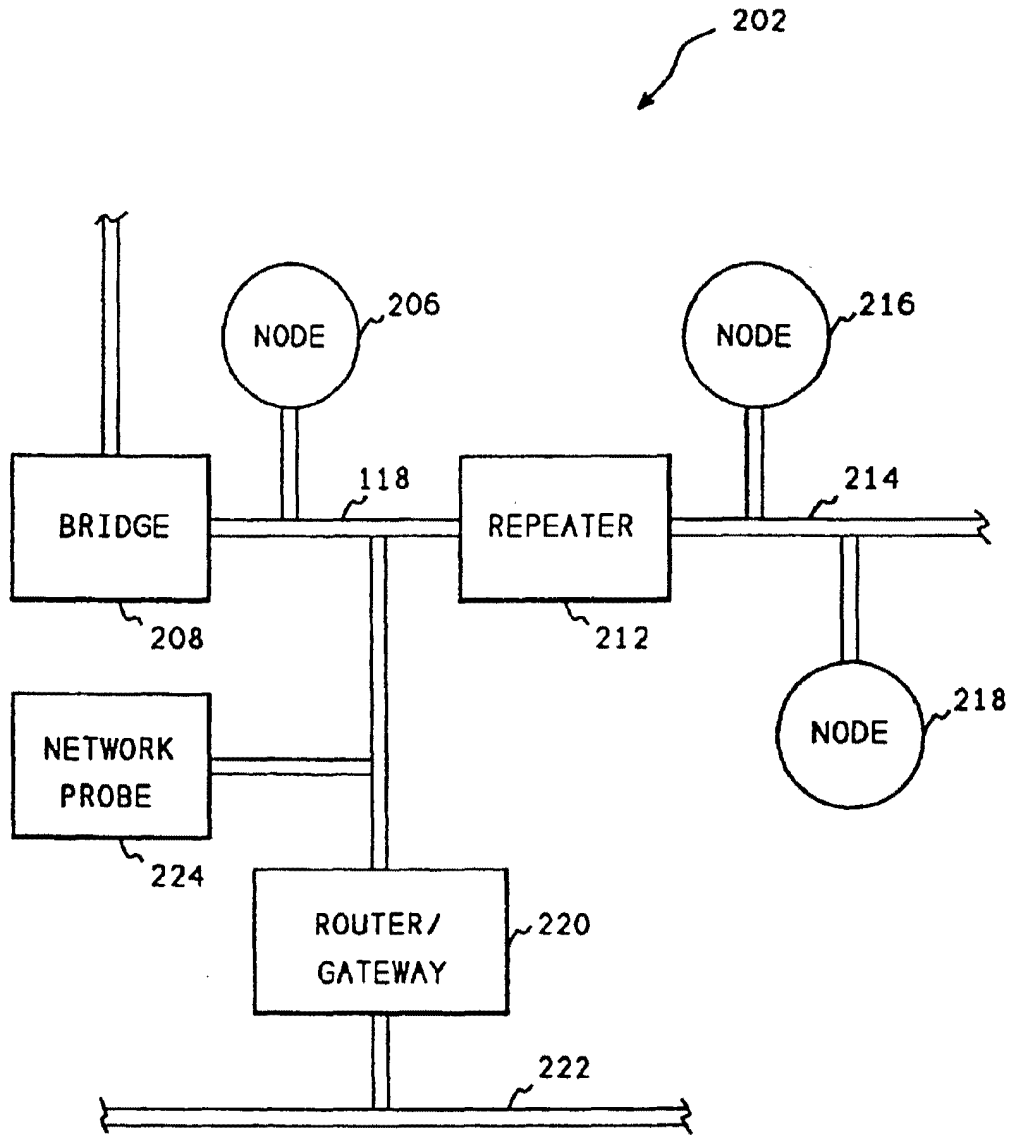


FIG. 2

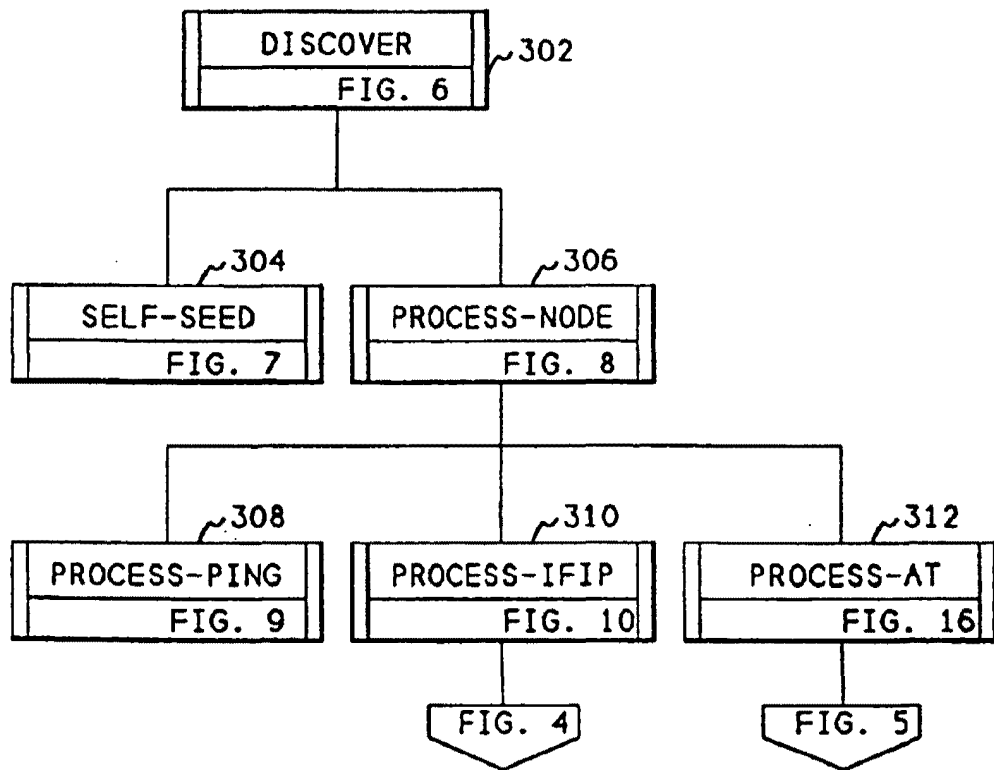


FIG. 3

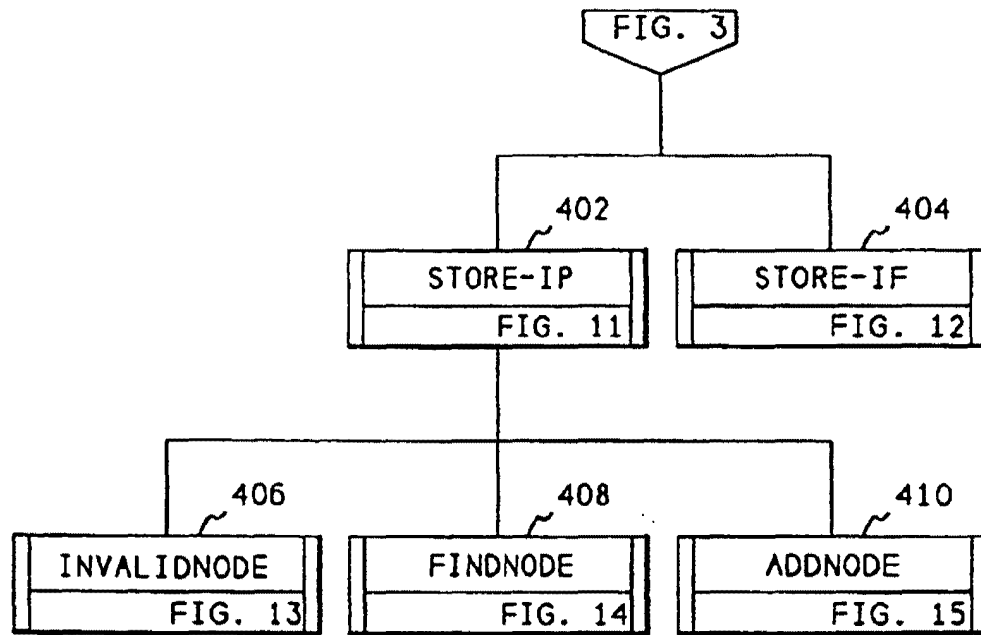


FIG. 4



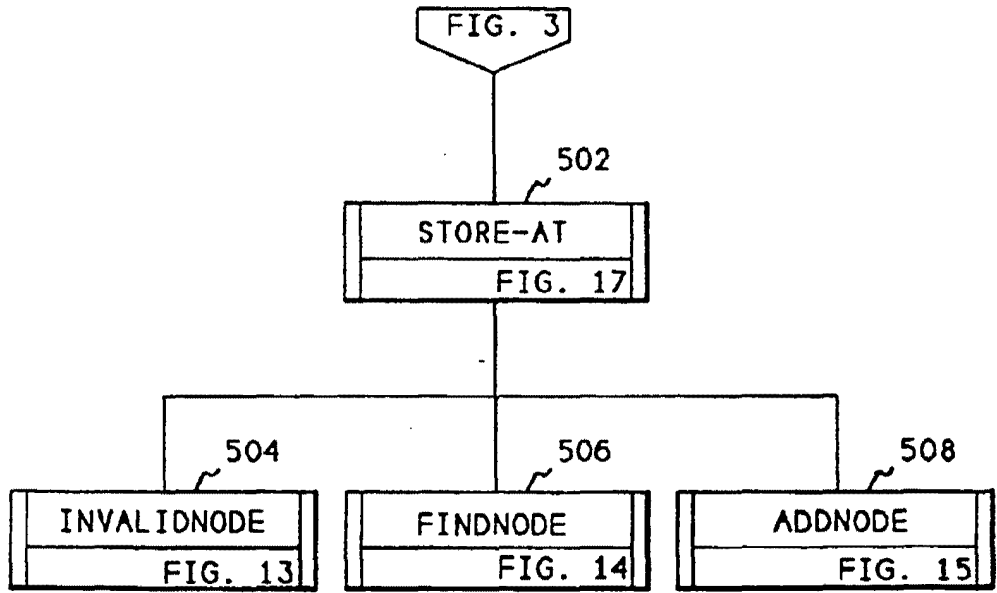


FIG. 5

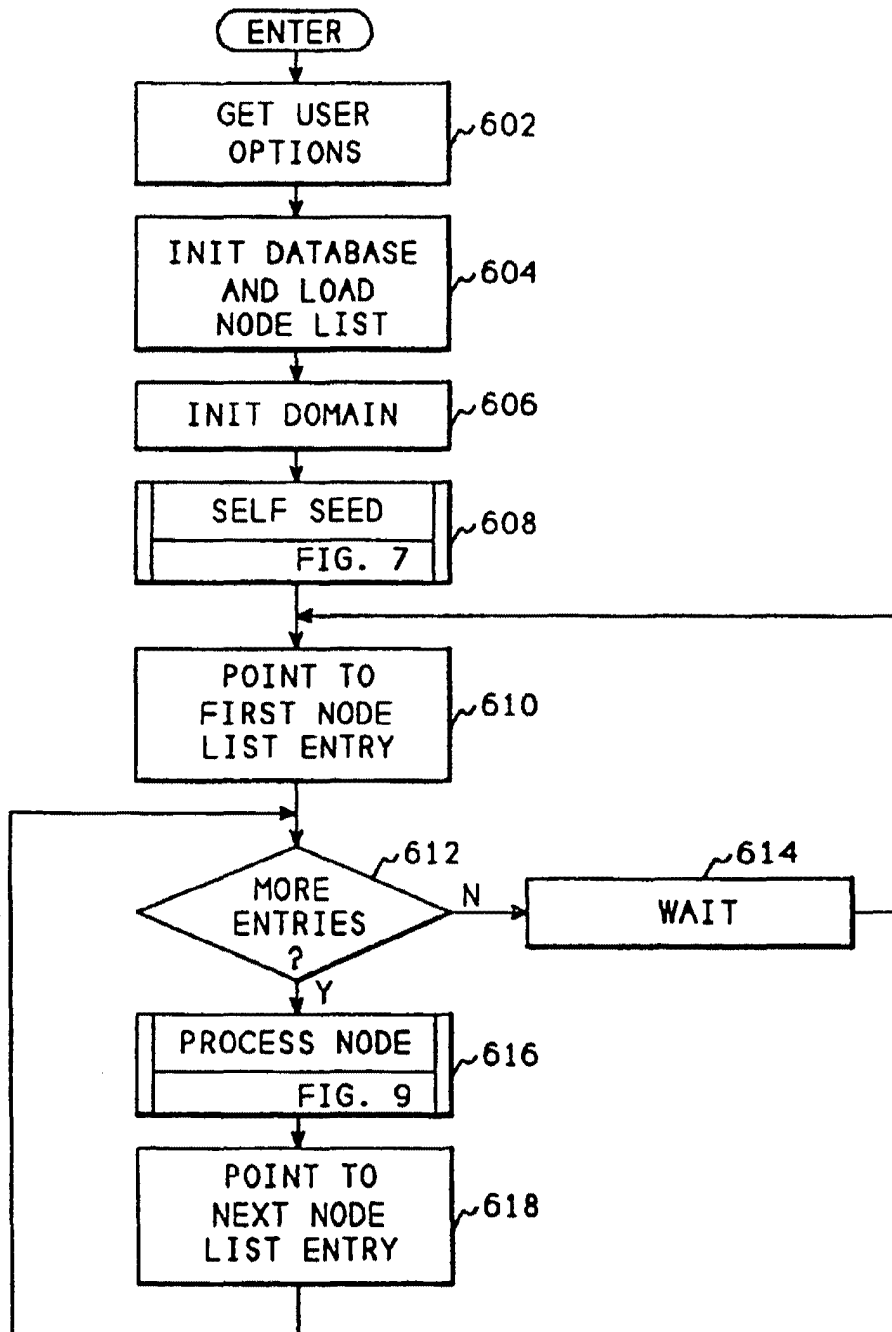


FIG. 6

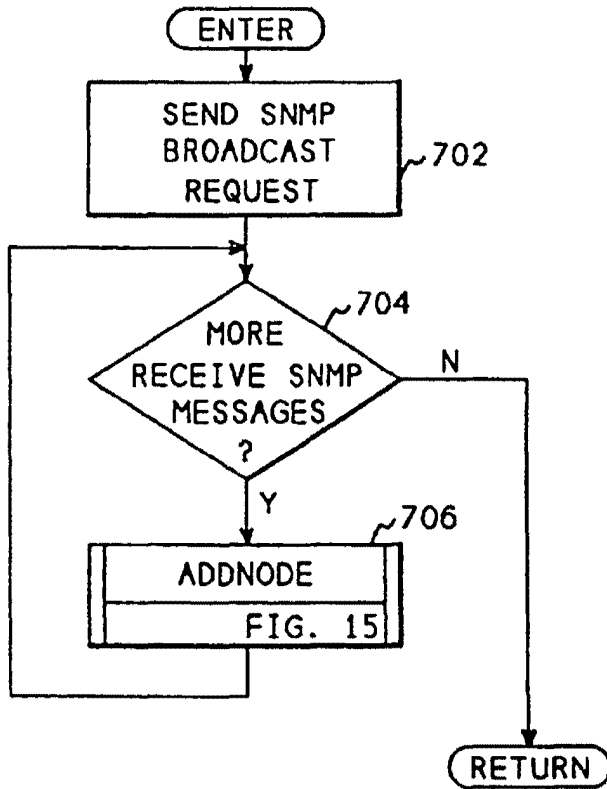


FIG. 7

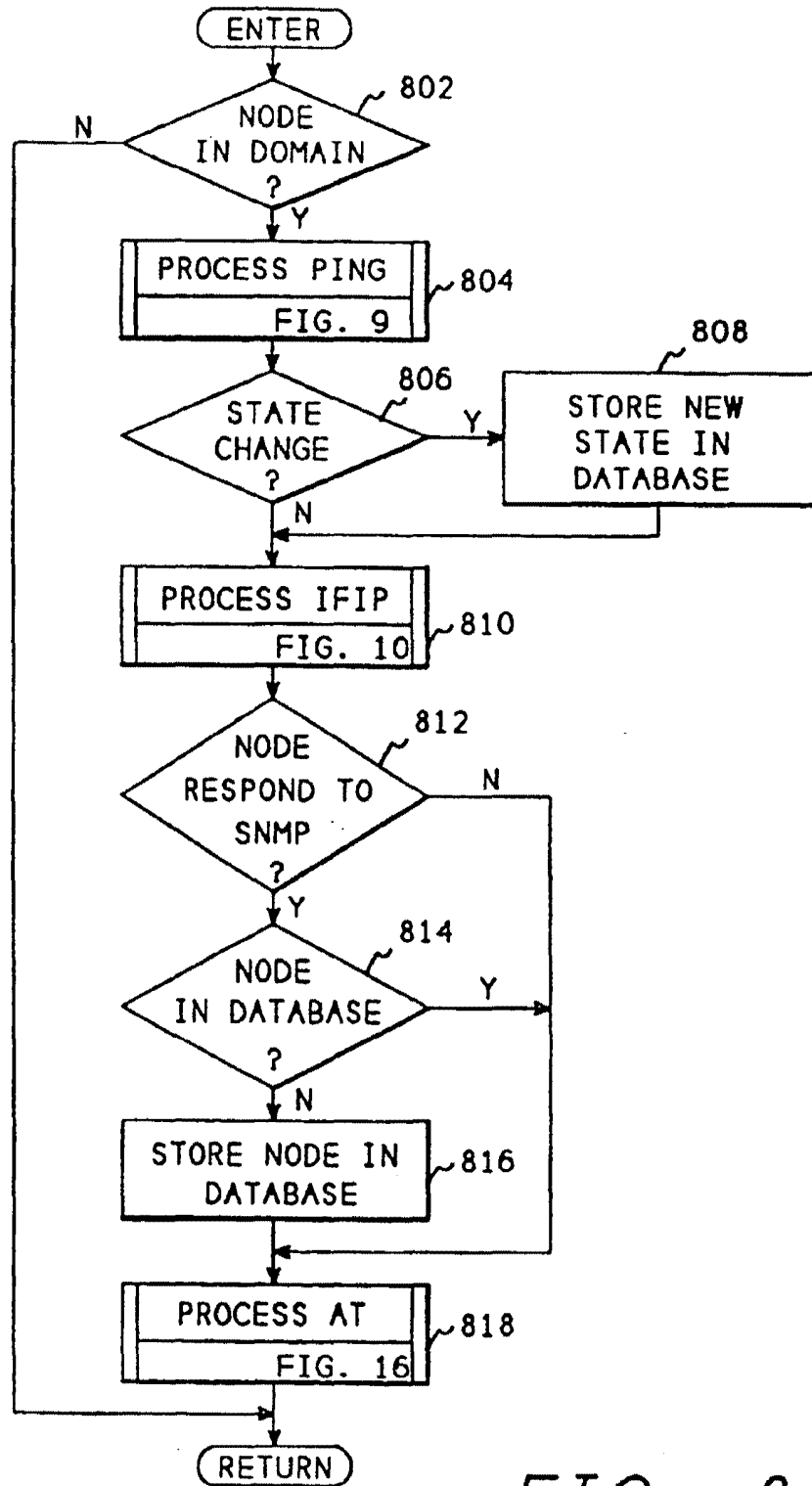


FIG. 8

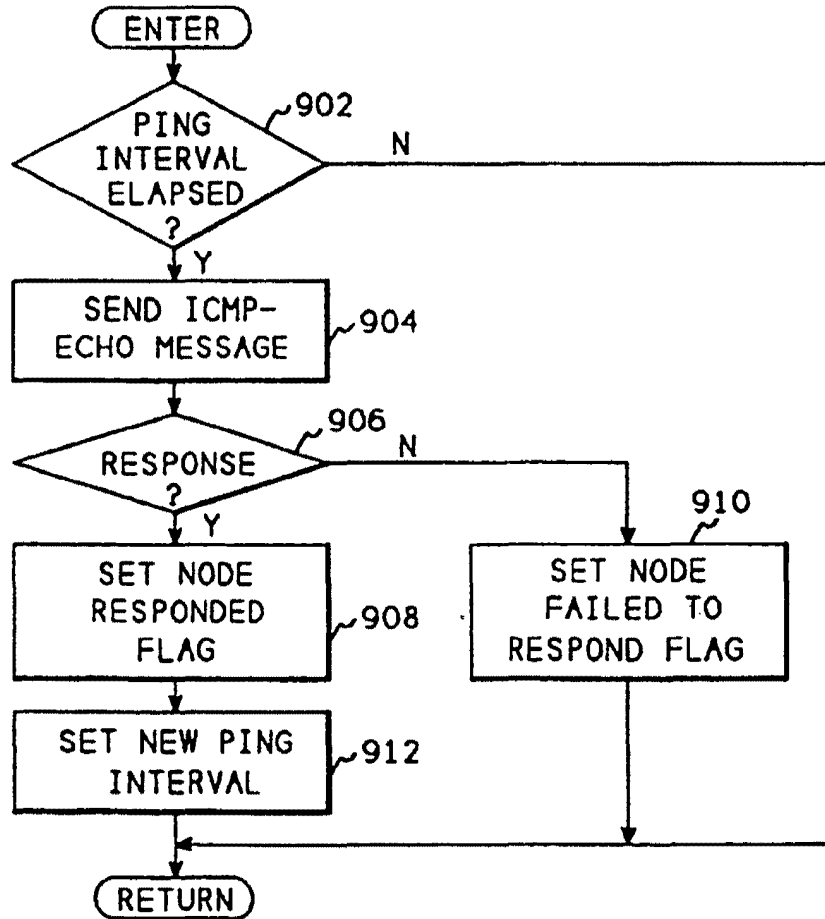


FIG. 9

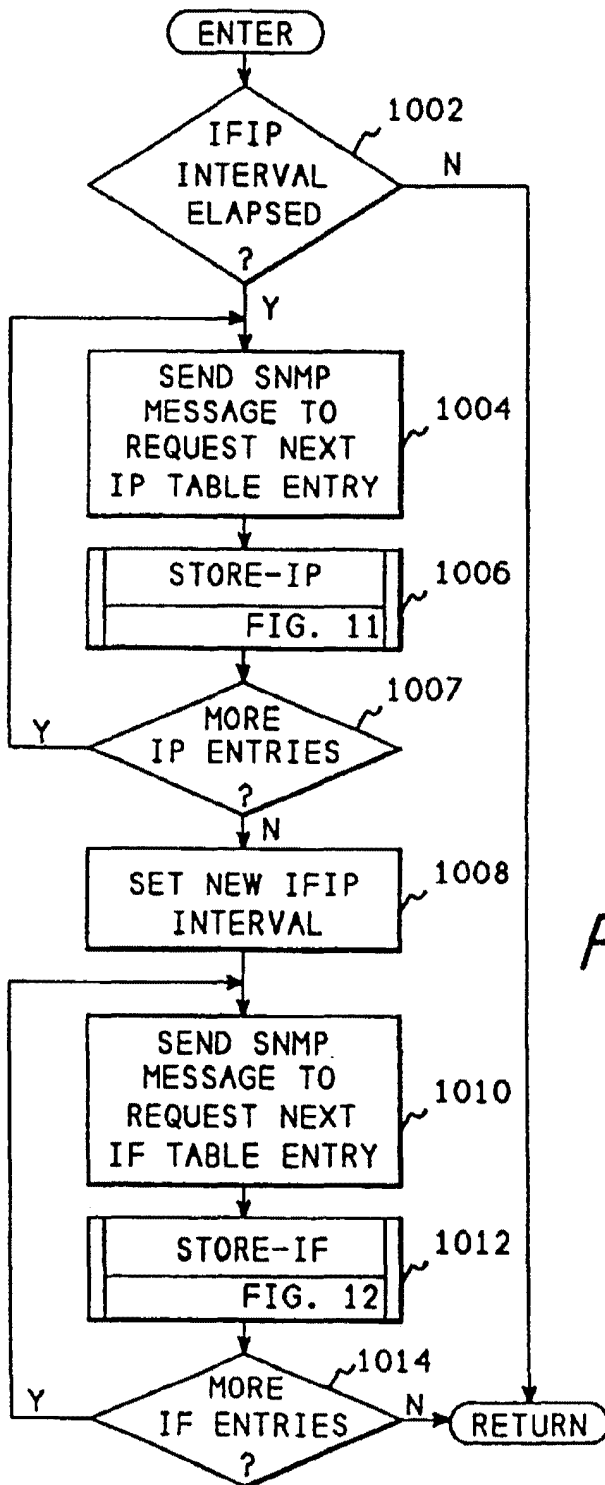


FIG. 10

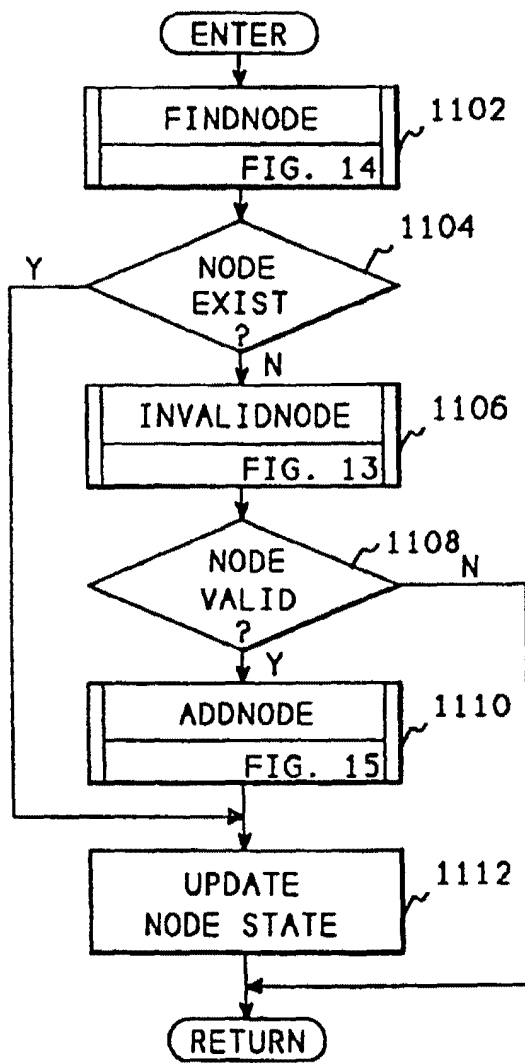


FIG. 11

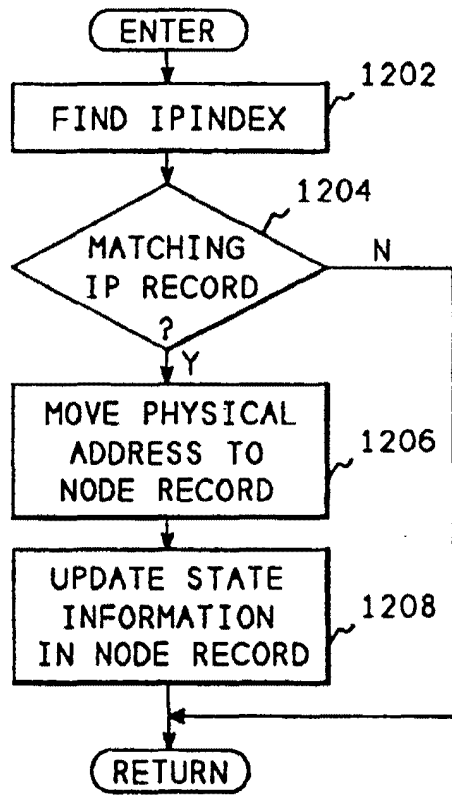


FIG. 12



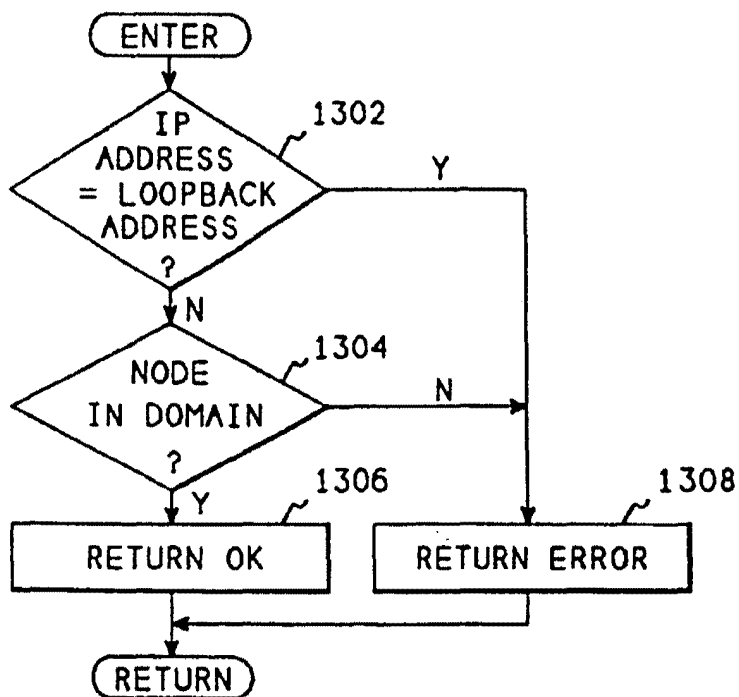


FIG. 13

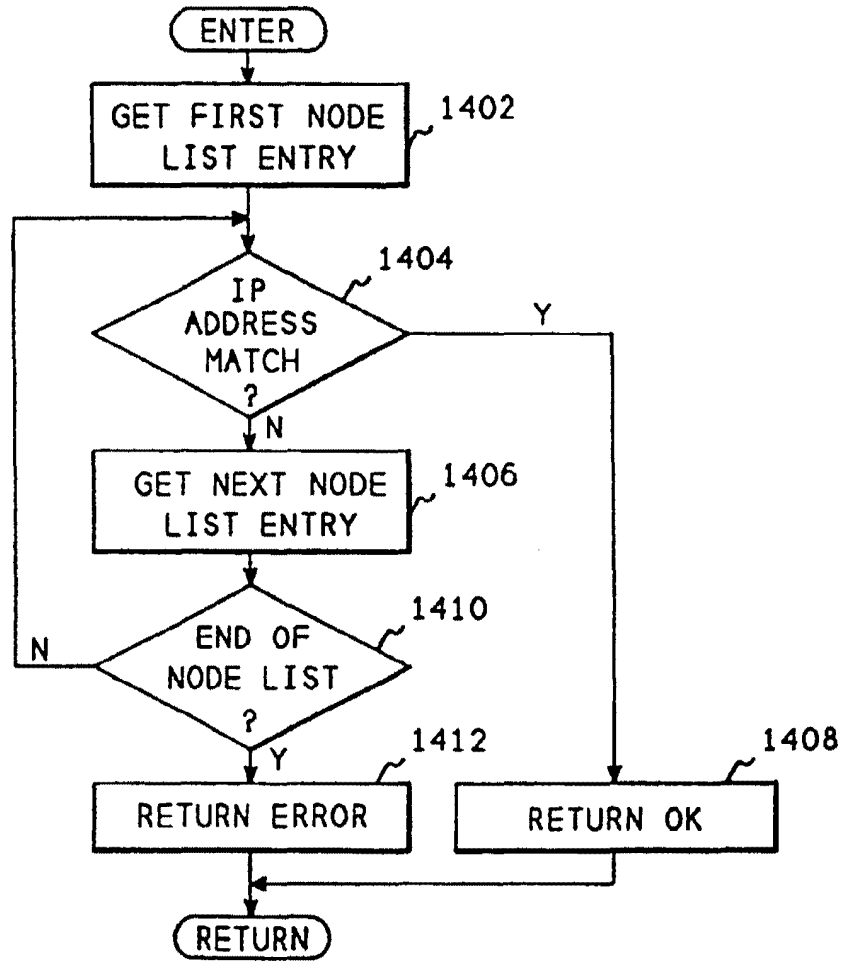


FIG. 14

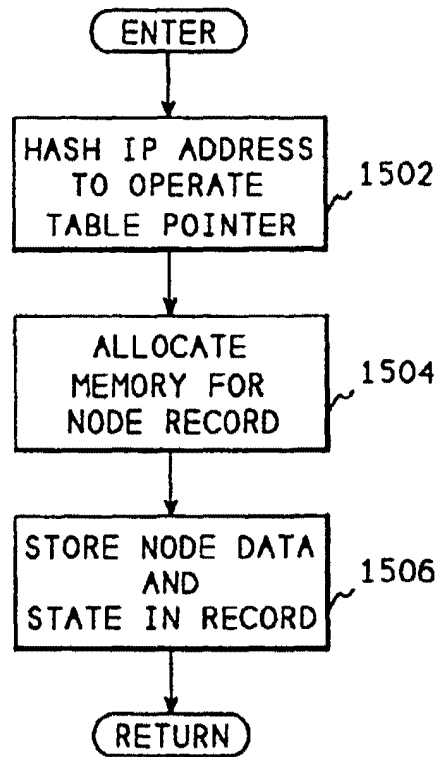


FIG. 15

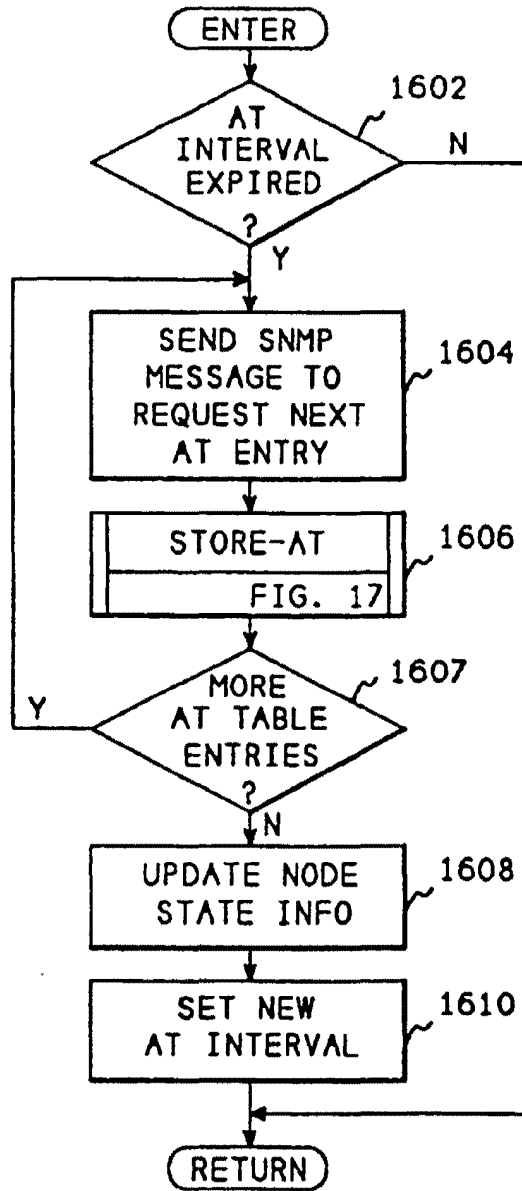


FIG. 16