US 20100046399A1

(19) **United States**

(12) **Patent Application Publication**

(10) **Pub. No.: US 2010/0046399 A1**

(43) **Pub. Date: Feb. 25, 2010**

Kakivaya et al.

(54) **RENDEZVOUSING RESOURCE REQUESTS WITH CORRESPONDING RESOURCES**

(75) Inventors: **Gopala Krishna R. Kakivaya**, Sammamish, WA (US); **Richard L. Hasha**, Seattle, WA (US); **Thomas Lee Rodeheffer**, Mountain View, CA (US)

Correspondence Address:
WORKMAN NYDEGGER/MICROSOFT
1000 EAGLE GATE TOWER, 60 EAST SOUTH TEMPLE
SALT LAKE CITY, UT 84111 (US)

(73) Assignee: **Microsoft Corporation**, Redmond, WA (US)

(21) Appl. No.: **12/611,825**

(22) Filed: **Nov. 3, 2009**

Related U.S. Application Data

(63) Continuation of application No. 10/971,451, filed on Oct. 22, 2004.

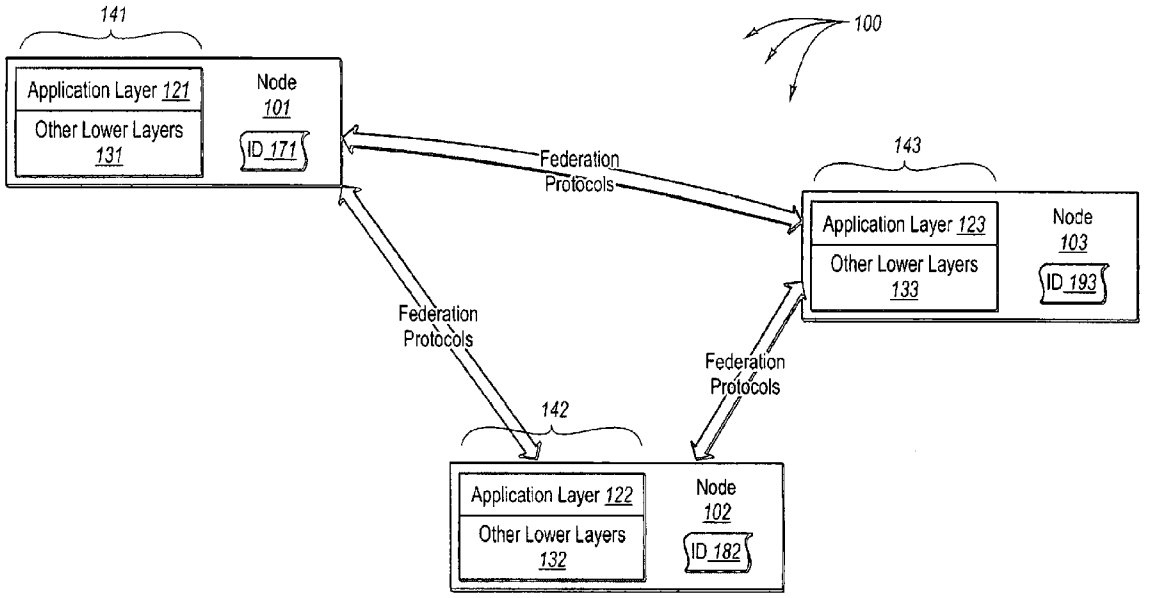
Publication Classification

(51) **Int. Cl.**
H04L 12/28 (2006.01)

(52) **U.S. Cl.** **370/254**

(57) **ABSTRACT**

The present invention extends to methods, systems, and computer program products for rendezvousing resource requests with corresponding resources. Doubly linked sorted lists are traversed using modulo arithmetic in both directions. Sorted lists can be partitioned based on a multiple proximity metrics. Node routing tables provide a logarithmic index to nodes within the ID space of the federation infrastructure to facilitate more efficient routing. Messages can be routed to nodes within a ring and proximally routed to nodes in other partitioned rings.



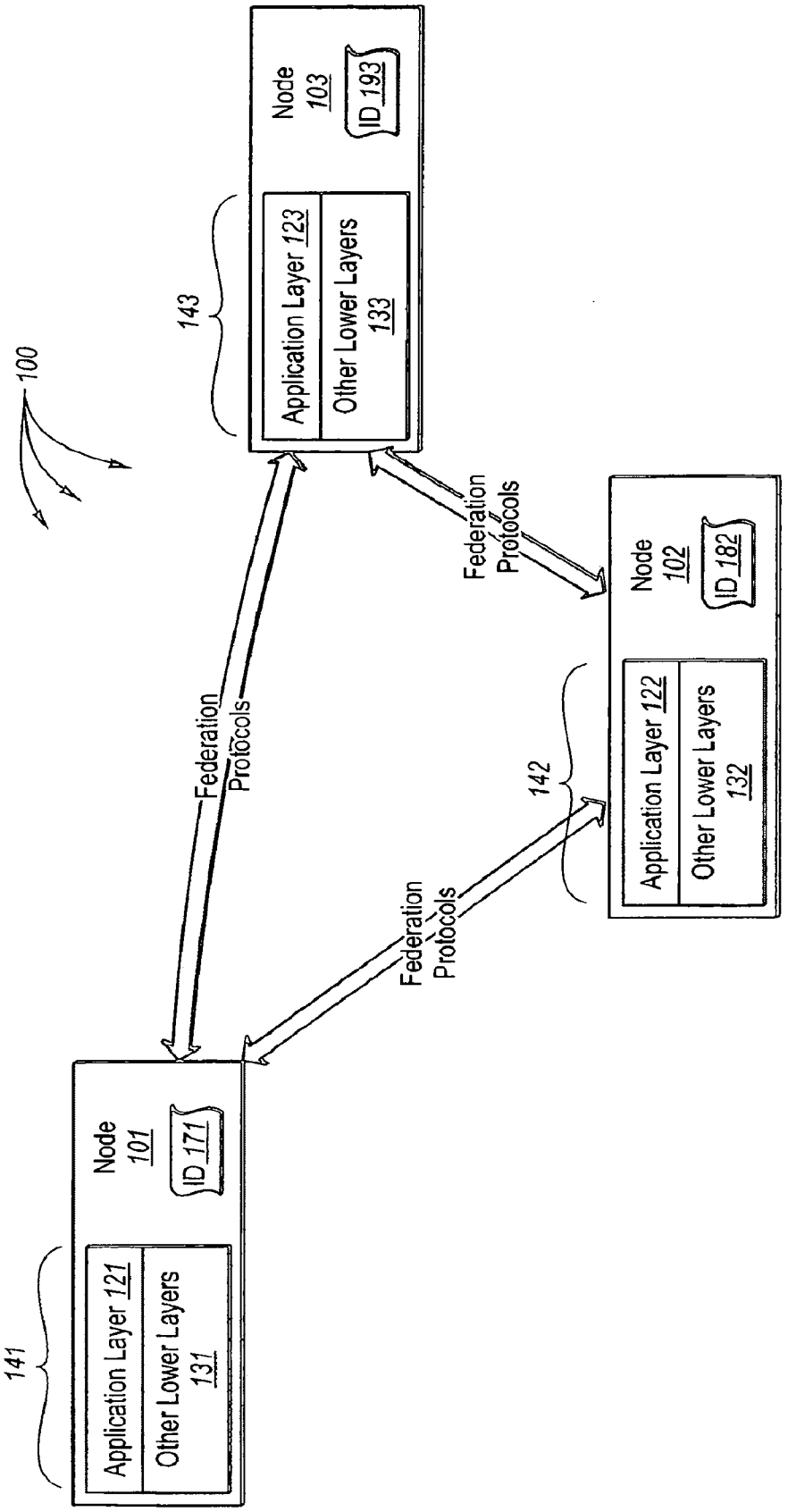

100
143
Node 103
ID 193
Application Layer 123
Other Lower Layers 133
Federation Protocols
Federation Protocols
Node 102
ID 182
142
Application Layer 122
Other Lower Layers 132
Federation Protocols
141
Node 101
ID 171
Application Layer 121
Other Lower Layers 131

Fig. 1

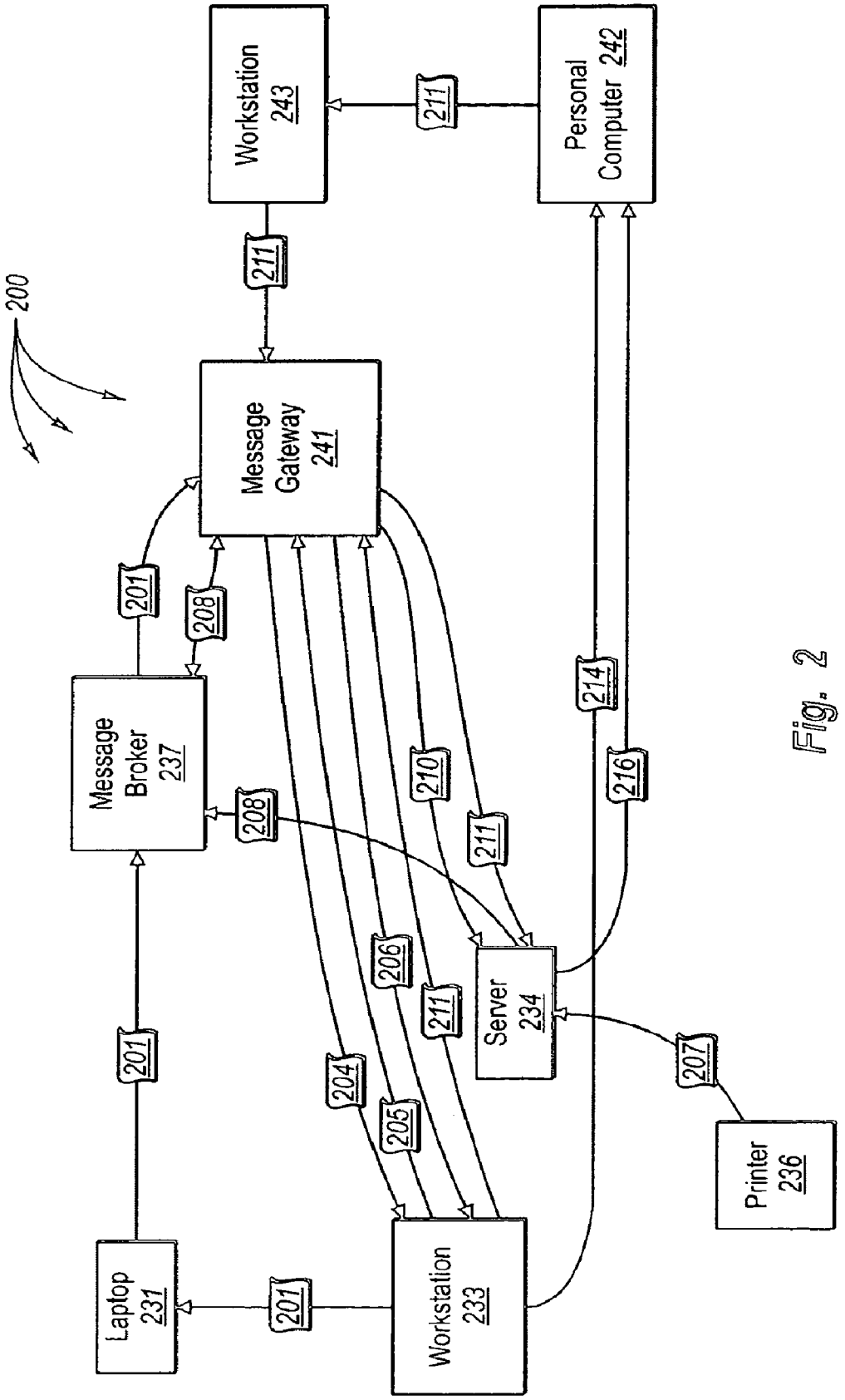

200
Workstation 243
Personal Computer 242
211
Message Gateway 241
201
208
Message Broker 237
210
214
216
Server 234
206
204
205
207
Printer 236
Laptop 231
Workstation 233

Fig. 2

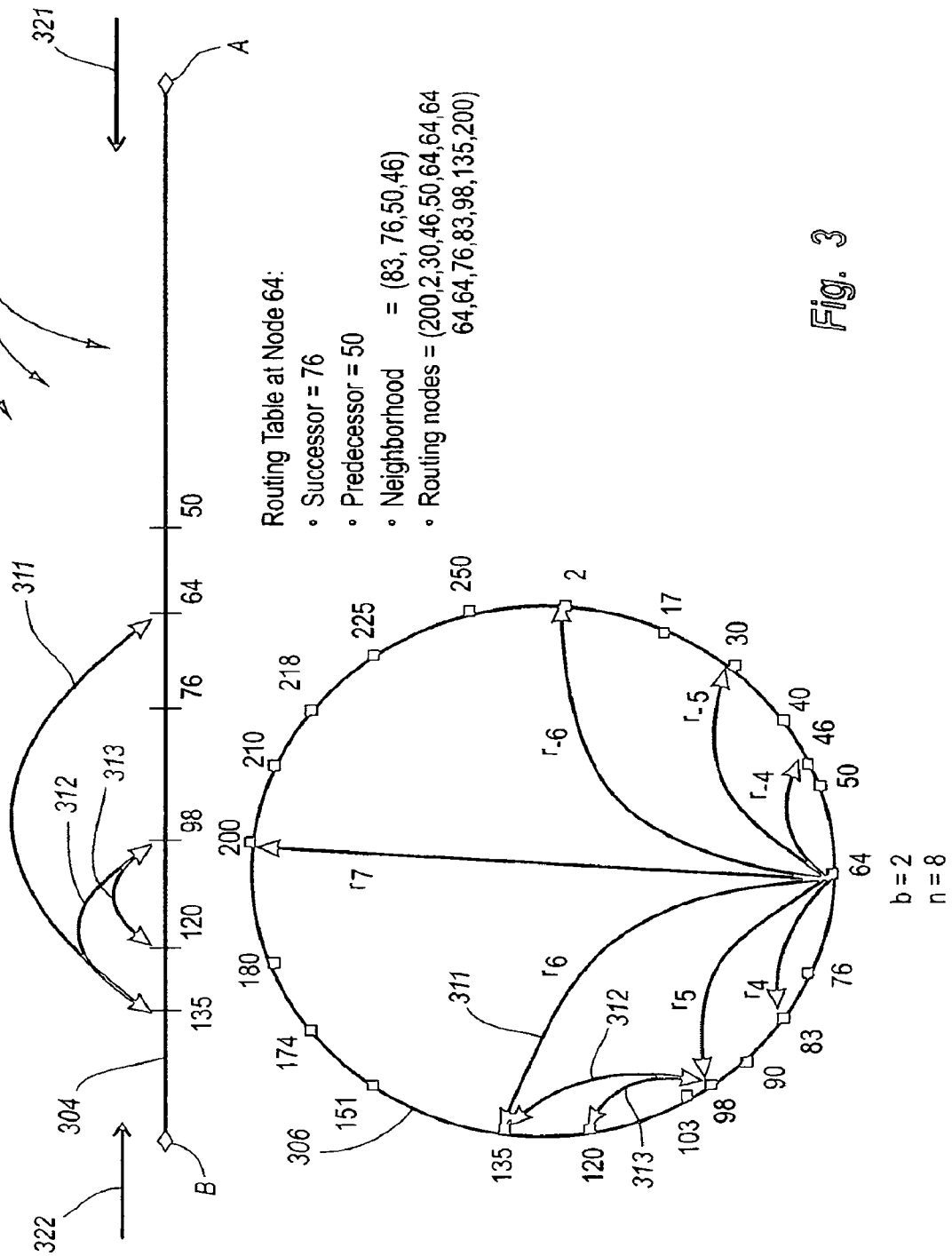

Routing Table at Node 64:
• Successor = 76
• Predecessor = 50
• Neighborhood = (83, 76,50,46)
• Routing nodes = (200,2,30,46,50,64,64,64
64,64,76,83,98,135,200)
b = 2
n = 8
A
B
300
304
306
311
312
313
321
322
r7
r6
r5
r4
r-6
r-5
r-4

Fig. 3

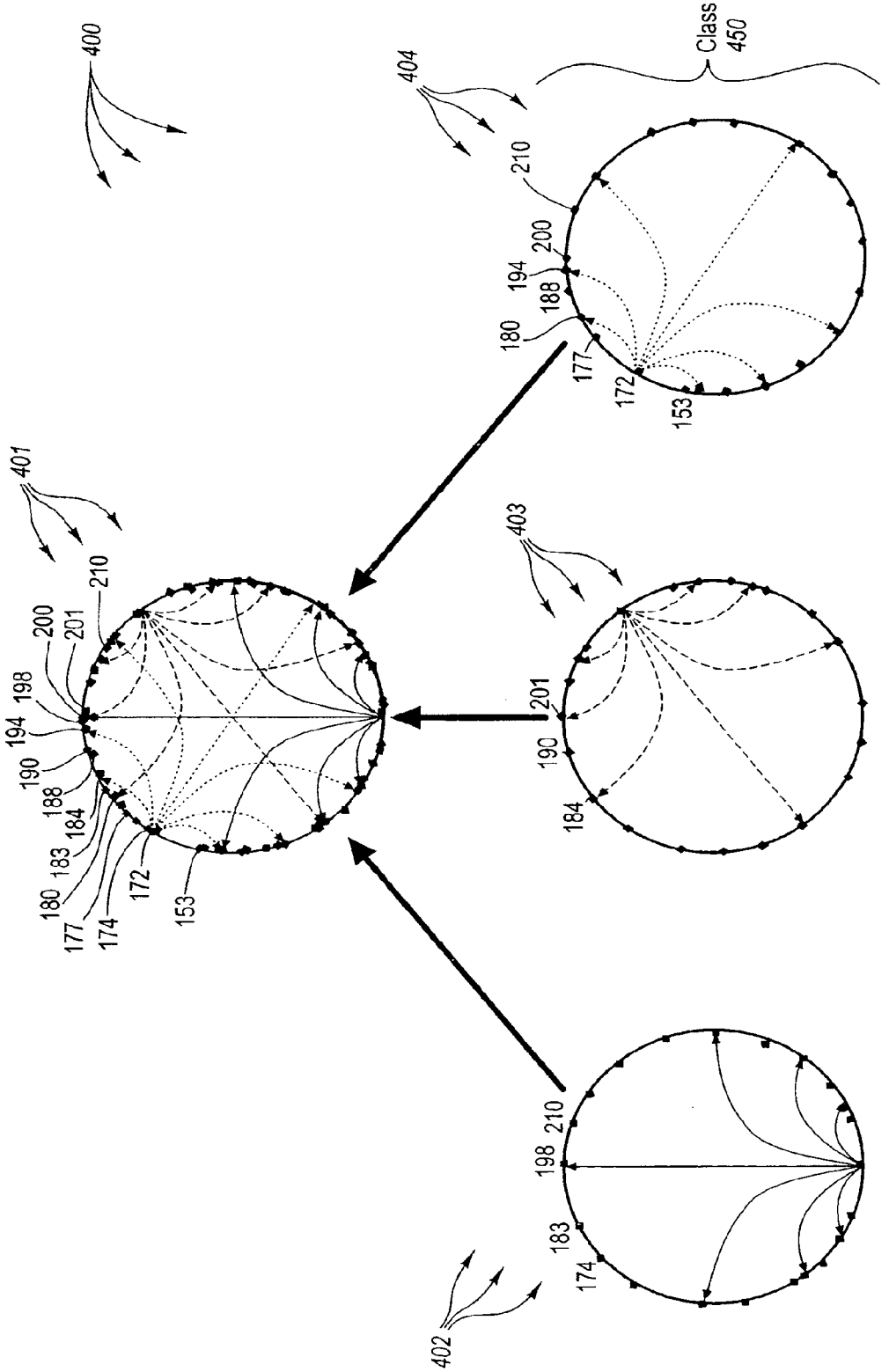

400
401
402
403
404
Class
450
210
200
201
198
194
190
188
184
183
180
177
174
172
153

Fig. 4

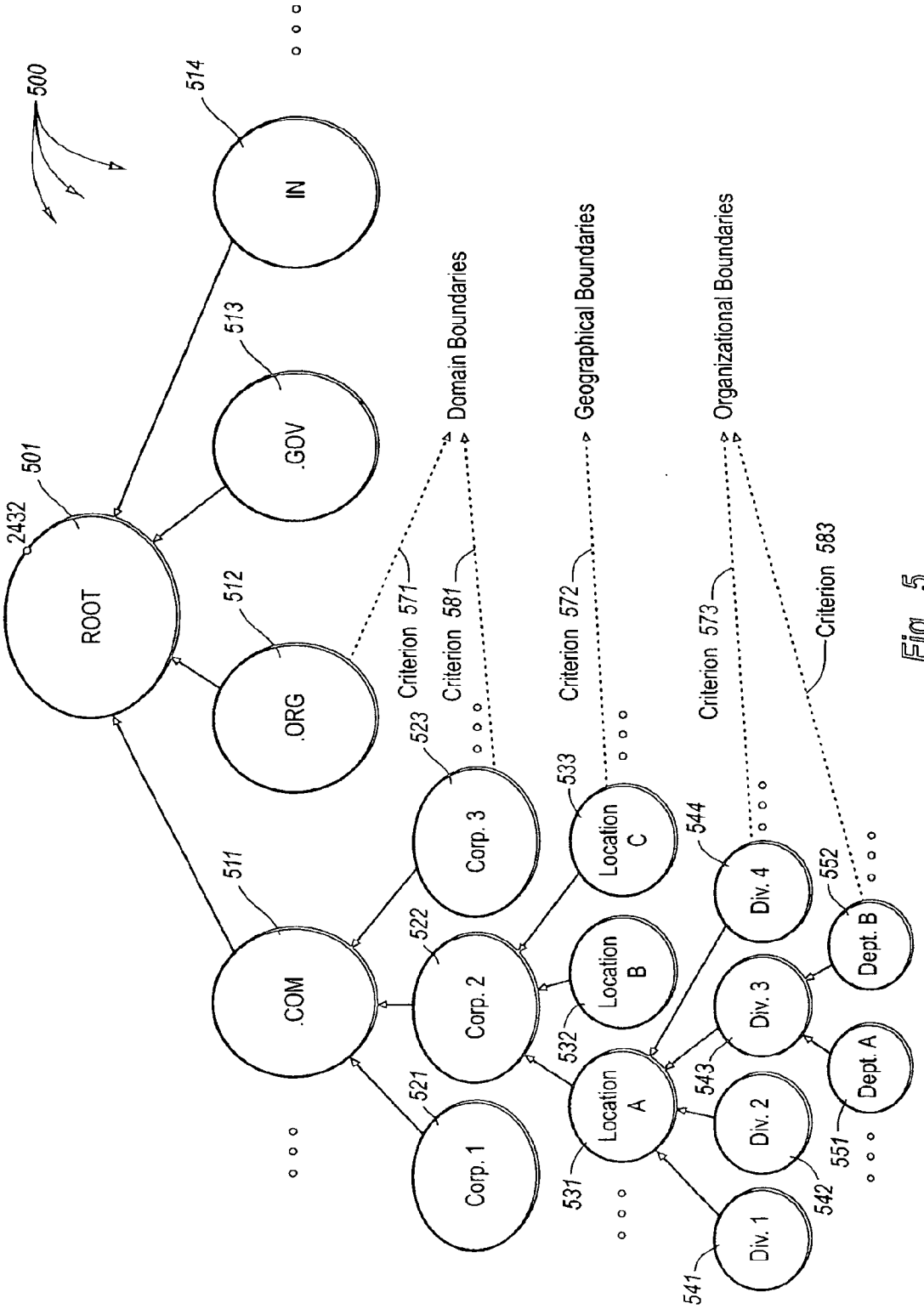
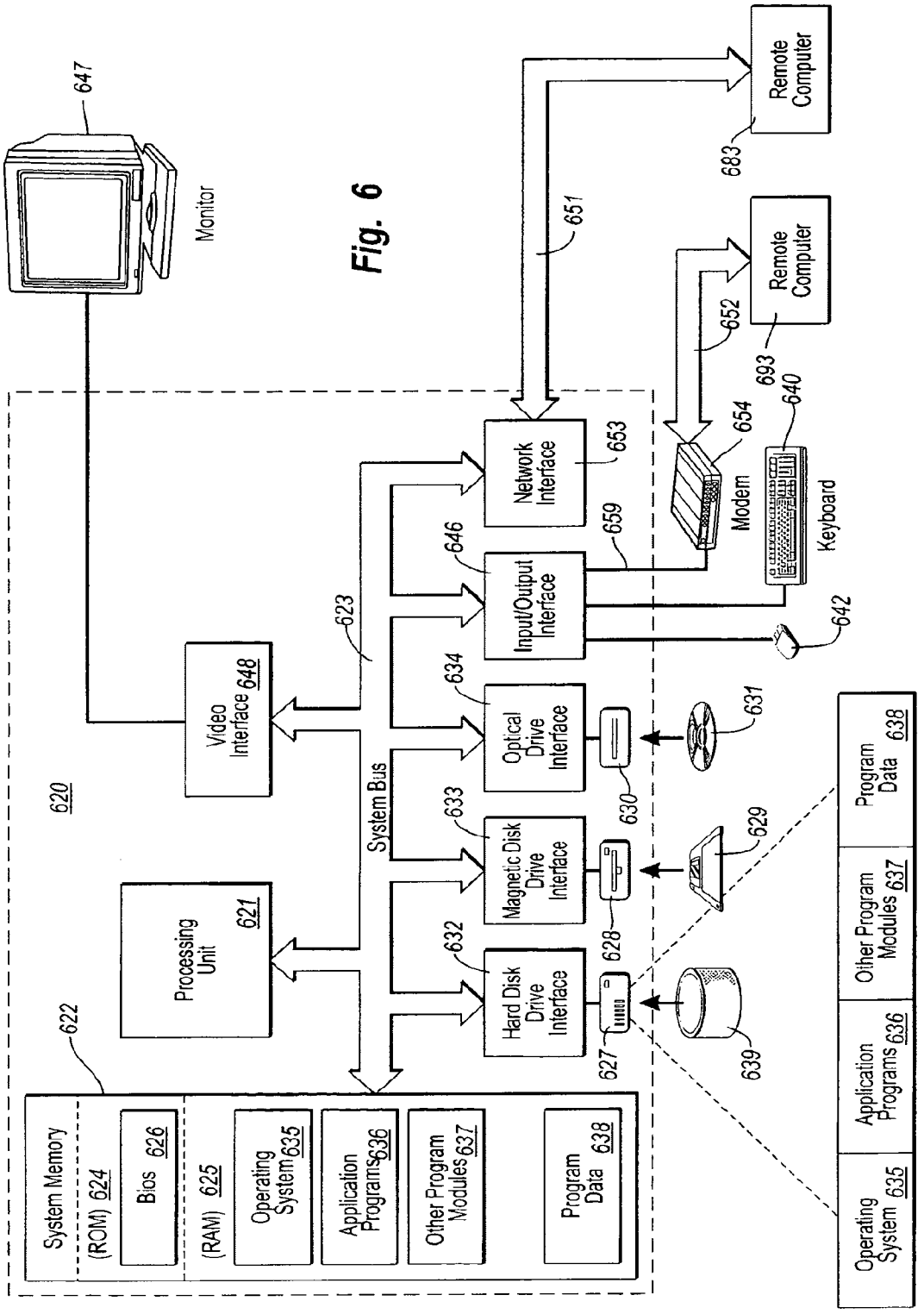

500
514
IN
513
.GOV
2432
501
ROOT
512
.ORG
511
.COM
Domain Boundaries
Criterion 571
Criterion 581
Geographical Boundaries
Criterion 572
Organizational Boundaries
Criterion 573
Criterion 583
523
Corp. 3
522
Corp. 2
521
Corp. 1
533
Location C
532
Location B
531
Location A
544
Div. 4
543
Div. 3
542
Div. 2
541
Div. 1
552
Dept. B
551
Dept. A

Fig. 5


647
Monitor
Fig. 6
Remote Computer
683
651
Remote Computer
652
693
640
654
Modem
Keyboard
642
Network Interface
653
646
Input/Output Interface
659
623
634
Optical Drive Interface
631
630
620
Video Interface 648
System Bus
633
Magnetic Disk Drive Interface
629
628
Processing Unit
621
632
Hard Disk Drive Interface
627
639
622
System Memory
(ROM) 624
Bios 626
(RAM) 625
Operating System 635
Application Programs 636
Other Program Modules 637
Program Data 638
Operating System 635
Application Programs 636
Other Program Modules 637
Program Data 638

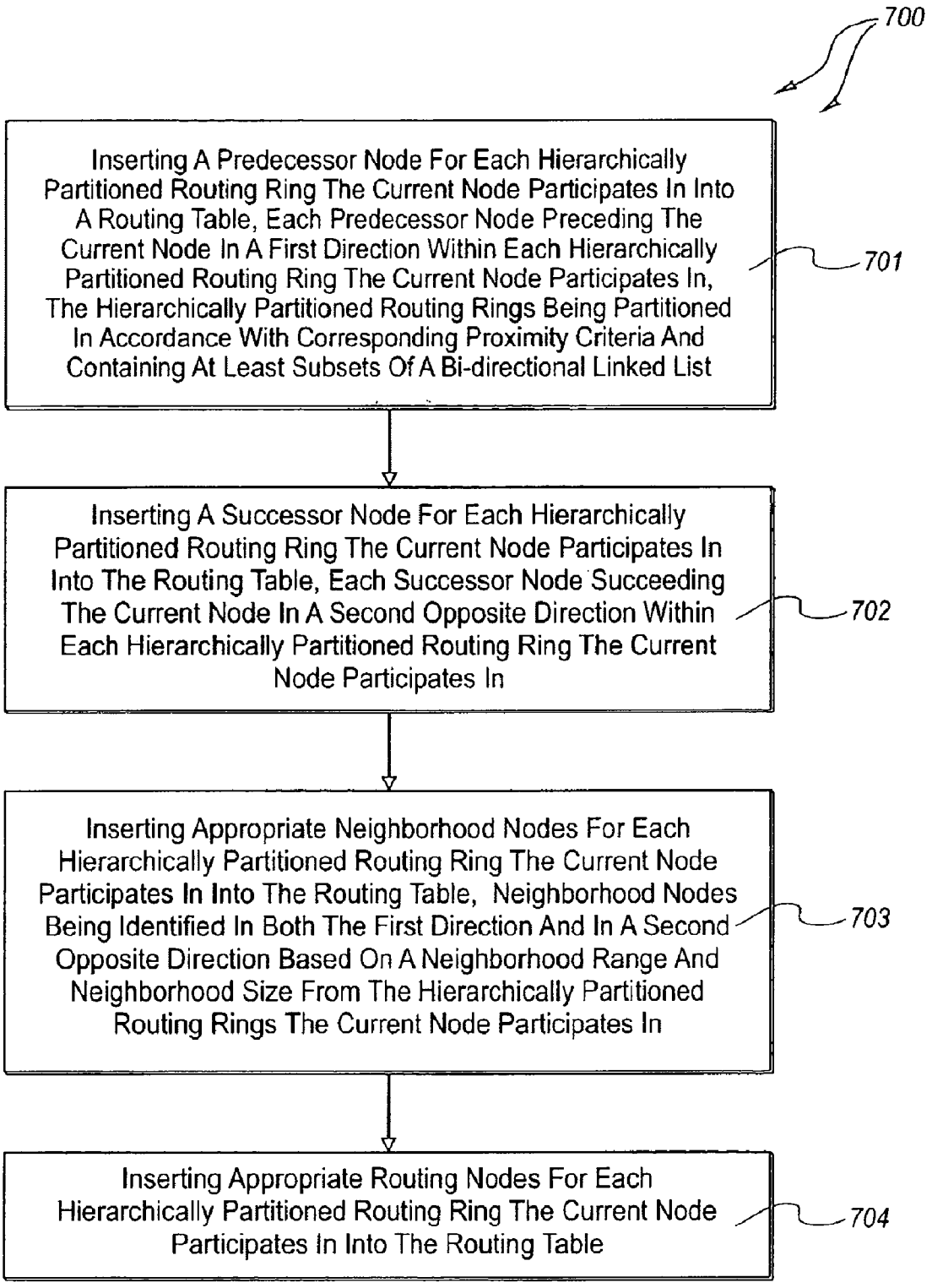

700
Inserting A Predecessor Node For Each Hierarchically Partitioned Routing Ring The Current Node Participates In Into A Routing Table, Each Predecessor Node Preceding The Current Node In A First Direction Within Each Hierarchically Partitioned Routing Ring The Current Node Participates In, The Hierarchically Partitioned Routing Rings Being Partitioned In Accordance With Corresponding Proximity Criteria And Containing At Least Subsets Of A Bi-directional Linked List
701
Inserting A Successor Node For Each Hierarchically Partitioned Routing Ring The Current Node Participates In Into The Routing Table, Each Successor Node Succeeding The Current Node In A Second Opposite Direction Within Each Hierarchically Partitioned Routing Ring The Current Node Participates In
702
Inserting Appropriate Neighborhood Nodes For Each Hierarchically Partitioned Routing Ring The Current Node Participates In Into The Routing Table, Neighborhood Nodes Being Identified In Both The First Direction And In A Second Opposite Direction Based On A Neighborhood Range And Neighborhood Size From The Hierarchically Partitioned Routing Rings The Current Node Participates In
703
Inserting Appropriate Routing Nodes For Each Hierarchically Partitioned Routing Ring The Current Node Participates In Into The Routing Table
704

Fig. 7

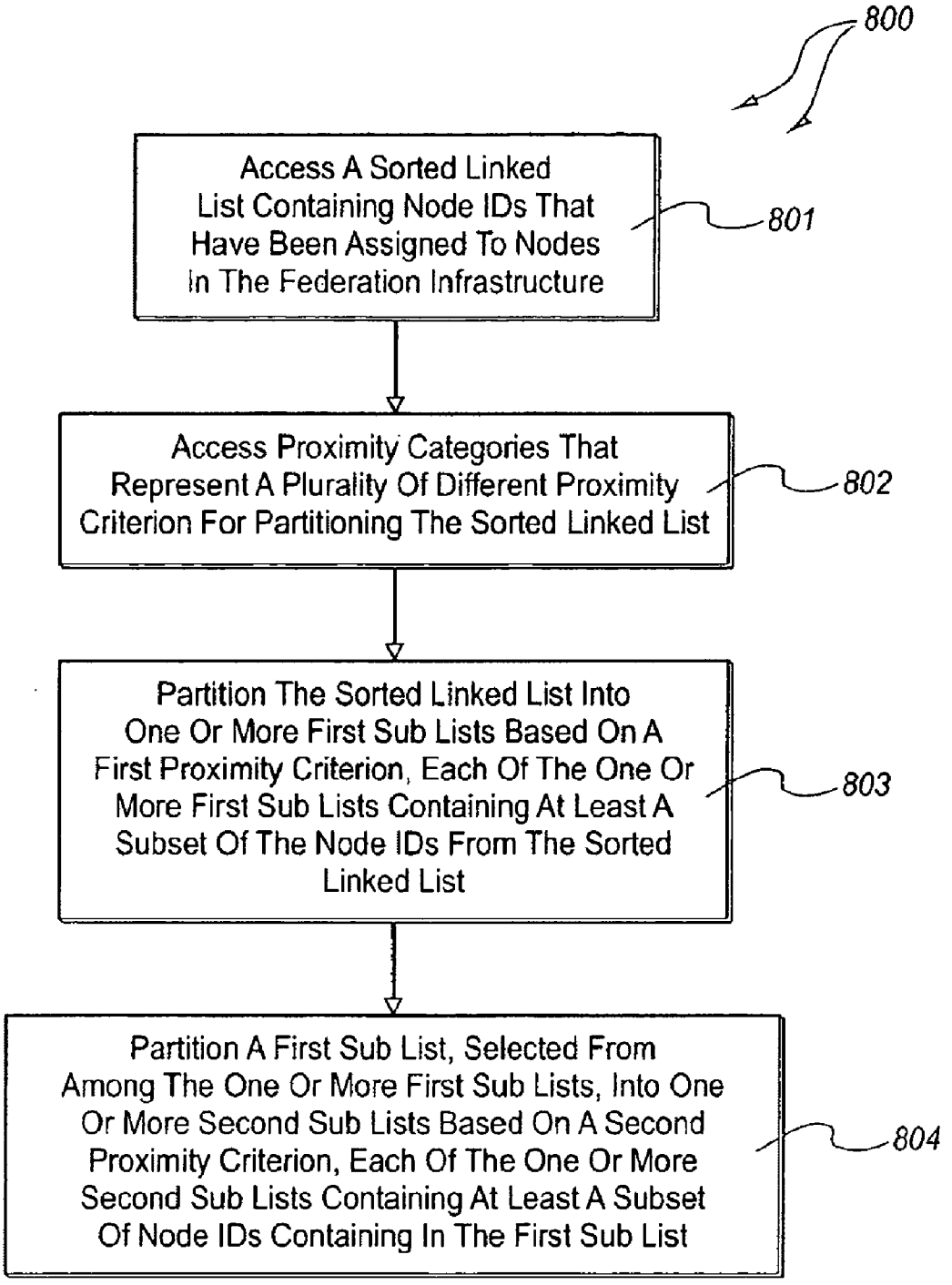

800
Access A Sorted Linked List Containing Node IDs That Have Been Assigned To Nodes In The Federation Infrastructure
801
Access Proximity Categories That Represent A Plurality Of Different Proximity Criterion For Partitioning The Sorted Linked List
802
Partition The Sorted Linked List Into One Or More First Sub Lists Based On A First Proximity Criterion, Each Of The One Or More First Sub Lists Containing At Least A Subset Of The Node IDs From The Sorted Linked List
803
Partition A First Sub List, Selected From Among The One Or More First Sub Lists, Into One Or More Second Sub Lists Based On A Second Proximity Criterion, Each Of The One Or More Second Sub Lists Containing At Least A Subset Of Node IDs Containing In The First Sub List
804

Fig. 8

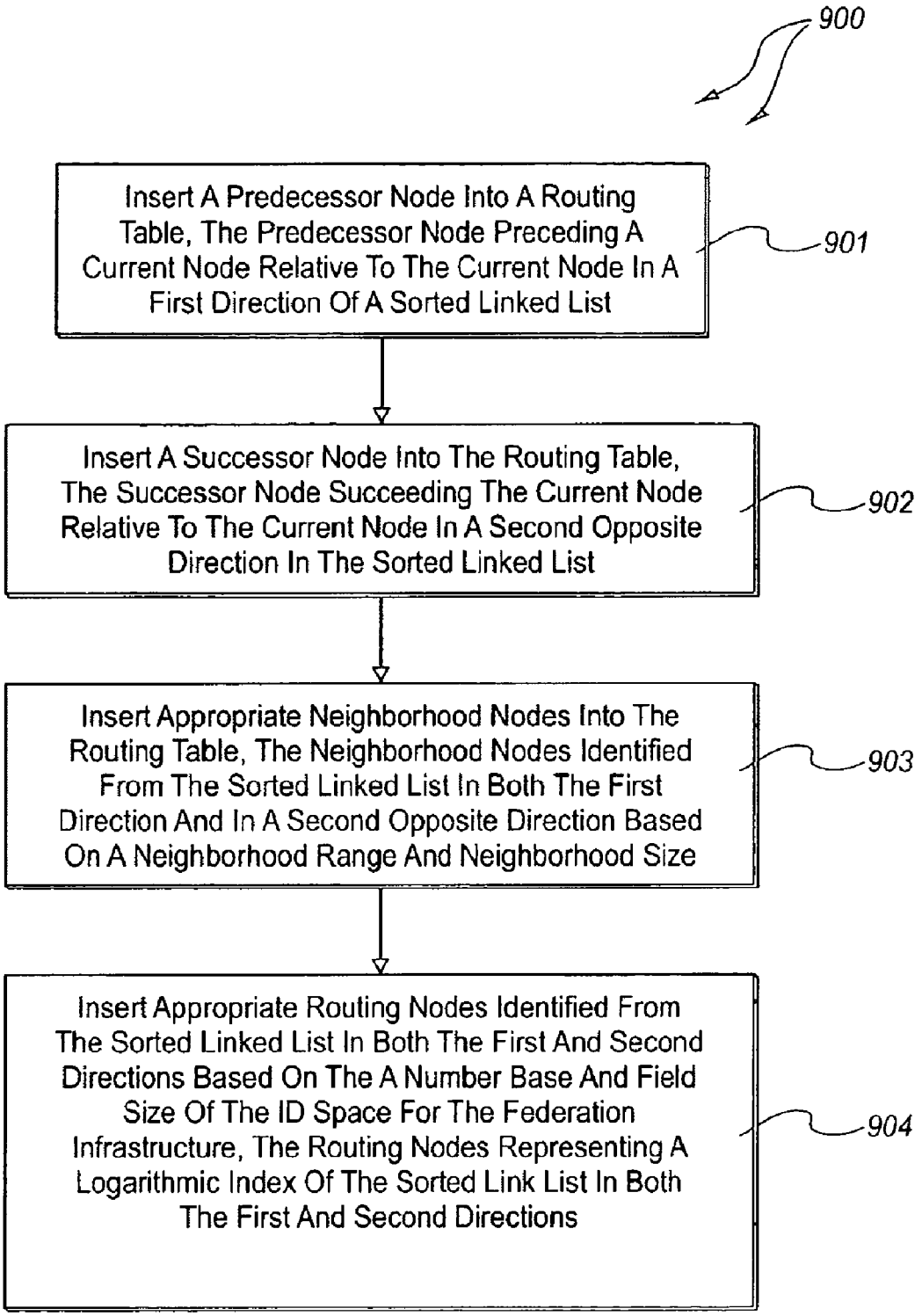

900
Insert A Predecessor Node Into A Routing Table, The Predecessor Node Preceding A Current Node Relative To The Current Node In A First Direction Of A Sorted Linked List
901
Insert A Successor Node Into The Routing Table, The Successor Node Succeeding The Current Node Relative To The Current Node In A Second Opposite Direction In The Sorted Linked List
902
Insert Appropriate Neighborhood Nodes Into The Routing Table, The Neighborhood Nodes Identified From The Sorted Linked List In Both The First Direction And In A Second Opposite Direction Based On A Neighborhood Range And Neighborhood Size
903
Insert Appropriate Routing Nodes Identified From The Sorted Linked List In Both The First And Second Directions Based On The A Number Base And Field Size Of The ID Space For The Federation Infrastructure, The Routing Nodes Representing A Logarithmic Index Of The Sorted Link List In Both The First And Second Directions
904

Fig. 9

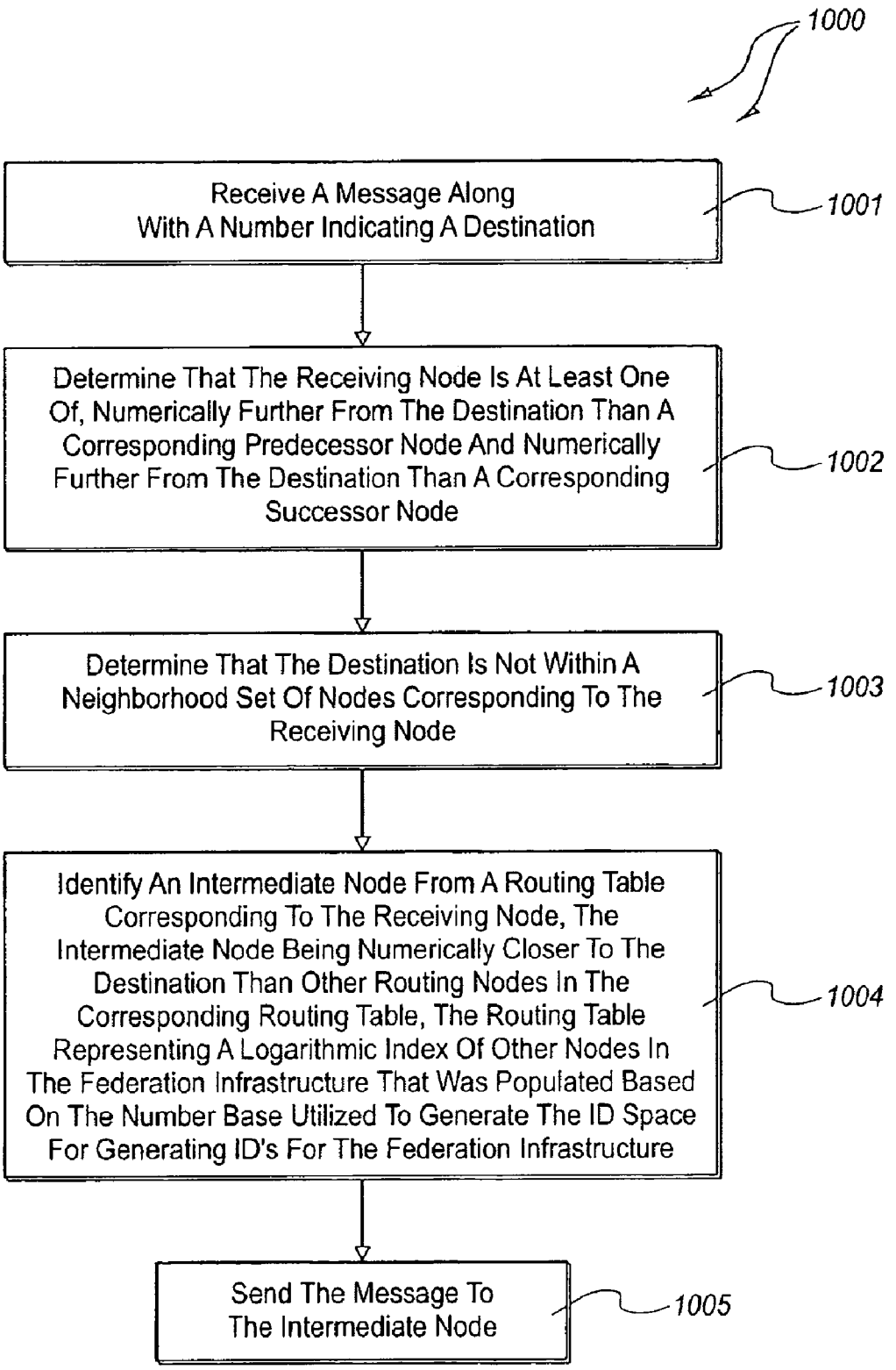

1000
Receive A Message Along With A Number Indicating A Destination
1001
Determine That The Receiving Node Is At Least One Of, Numerically Further From The Destination Than A Corresponding Predecessor Node And Numerically Further From The Destination Than A Corresponding Successor Node
1002
Determine That The Destination Is Not Within A Neighborhood Set Of Nodes Corresponding To The Receiving Node
1003
Identify An Intermediate Node From A Routing Table Corresponding To The Receiving Node, The Intermediate Node Being Numerically Closer To The Destination Than Other Routing Nodes In The Corresponding Routing Table, The Routing Table Representing A Logarithmic Index Of Other Nodes In The Federation Infrastructure That Was Populated Based On The Number Base Utilized To Generate The ID Space For Generating ID's For The Federation Infrastructure
1004
Send The Message To The Intermediate Node
1005

Fig. 10

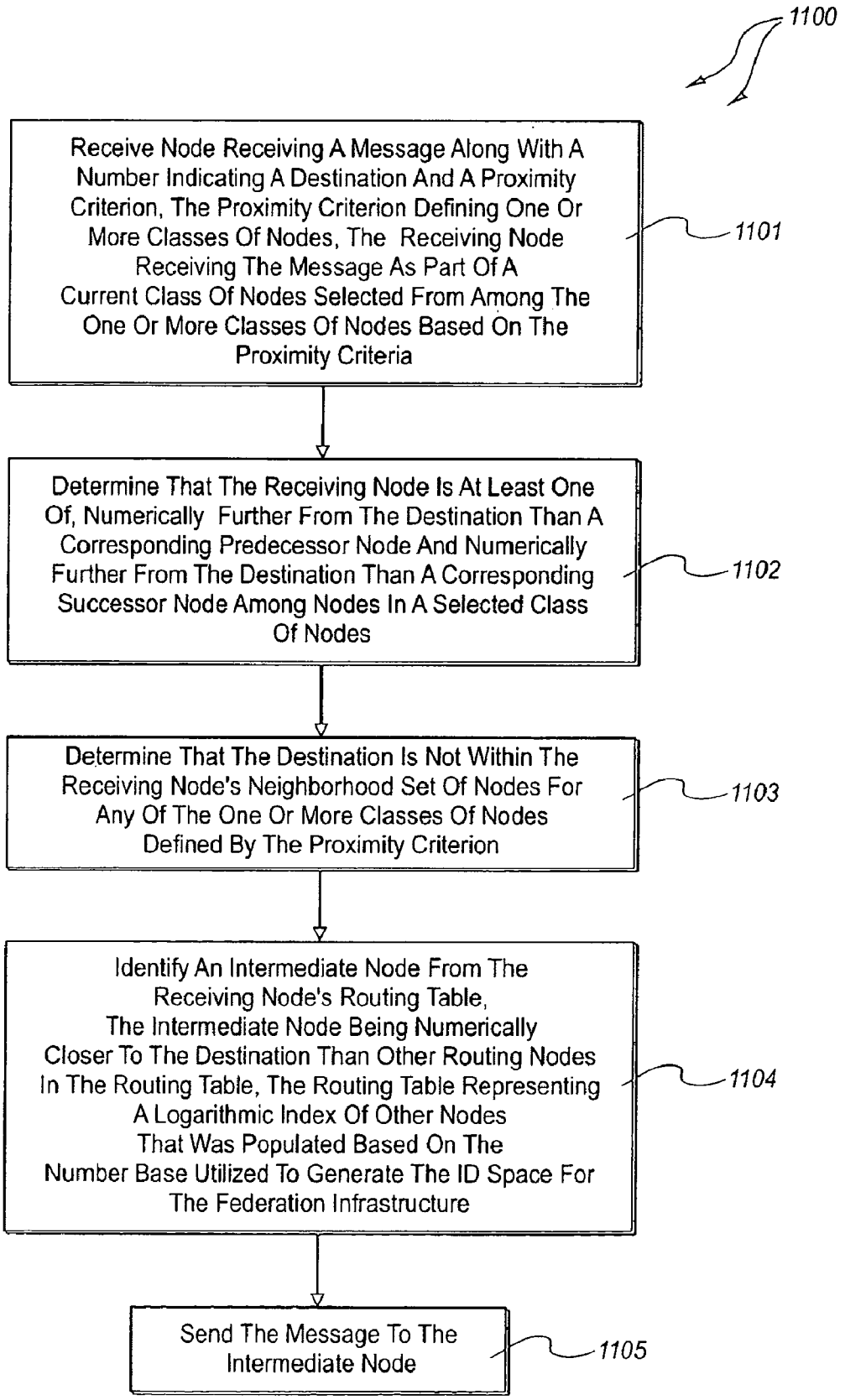

1100
Receive Node Receiving A Message Along With A Number Indicating A Destination And A Proximity Criterion, The Proximity Criterion Defining One Or More Classes Of Nodes, The Receiving Node Receiving The Message As Part Of A Current Class Of Nodes Selected From Among The One Or More Classes Of Nodes Based On The Proximity Criteria
1101
Determine That The Receiving Node Is At Least One Of, Numerically Further From The Destination Than A Corresponding Predecessor Node And Numerically Further From The Destination Than A Corresponding Successor Node Among Nodes In A Selected Class Of Nodes
1102
Determine That The Destination Is Not Within The Receiving Node's Neighborhood Set Of Nodes For Any Of The One Or More Classes Of Nodes Defined By The Proximity Criterion
1103
Identify An Intermediate Node From The Receiving Node's Routing Table, The Intermediate Node Being Numerically Closer To The Destination Than Other Routing Nodes In The Routing Table, The Routing Table Representing A Logarithmic Index Of Other Nodes That Was Populated Based On The Number Base Utilized To Generate The ID Space For The Federation Infrastructure
1104
Send The Message To The Intermediate Node
1105

Fig. 11

RENDEZVOUSING RESOURCE REQUESTS WITH CORRESPONDING RESOURCES

CROSS-REFERENCE TO RELATED APPLICATIONS

[0001] This application is a continuation of U.S. patent application Ser. No. 10/971,451, filed Oct. 22, 2004, and entitled "Rendezvousing Resource Requests with Corresponding Resources," which is herein incorporated by reference in its entirety.

BACKGROUND OF THE INVENTION

[0002] 1. The Field of the Invention

[0003] The present invention relates to accessing resources and, more particularly, to rendezvousing resource requests with corresponding resources.

[0004] 2. Background and Relevant Art

[0005] Computer systems and related technology affect many aspects of society. Indeed, the computer system's ability to process information has transformed the way we live and work. Computer systems now commonly perform a host of tasks (e.g., word processing, scheduling, and database management) that prior to the advent of the computer system were performed manually. More recently, computer systems have been coupled to one another and to other electronic devices to form both wired and wireless computer networks over which the computer systems and other electronic devices can transfer electronic data. As a result, many tasks performed at a computer system (e.g., voice communication, accessing electronic mail, controlling home electronics, Web browsing, and printing documents) include electronic communication between a number of computer systems and/or other electronic devices via wired and/or wireless computer networks.

[0006] However, to utilize a network resource to perform a computerized task, a computer system must have some way to identify and access the network resource. Accordingly, resources are typically assigned unique identifiers, for example, network addresses, that uniquely identify resources and can be used to distinguish one resource from other resources. Thus, a computer system that desires to utilize a resource can connect to the resource using the network address that corresponds to the resource. However, accessing a network resource can be difficult if a computer system has no prior knowledge of a network address for a network resource. For example, a computer system can not print a document at a network printer unless the computer system (or another networked computer system) knows the network address of the network printer.

[0007] Accordingly, various mechanisms (e.g., Domain Name System ("DNS"), Active Directory ("AD"), Distributed File Systems ("DFS")) have been developed for computer systems to identify (and access) previous unknown resources. However, due to the quantity and diversity of resources (e.g., devices and services) that are accessible via different computer networks, developers are often required to develop applications that implement a variety of different resource identification and access mechanisms. Each different mechanism may have different coding requirements and may not provide a developer with all the functionality that is needed in an application.

[0008] For example, although DNS has a distributed administration architecture (i.e., centralized management is not required), DNS is not sufficiently dynamic, not self-organizing, supports a weak data and query model, and has a fixed set of roots. On the other hand, AD is sufficiently dynamic but requires centralized administration. Further, aspects of different mechanisms may not be compatible with one another. For example, a resource identified using DNS may not be compatible with DFS routing protocols. Thus, a developer may be forced to choose the most suitable mechanism and forgo the advantages of other mechanisms.

[0009] Mechanisms for identifying resources can be particularly problematic in peer-to-peer networks. DNS provides a lookup service, with host names as keys and IP addresses as values, that relies on a set of special root servers to implement lookup requests. Further, DNS requires management of information (NS records) for allowing clients to navigate the name server hierarchy. Thus, a resource must be entered into DNS before the resource can be identified on a network. On larger scale networks where nodes frequently connect and disconnect form the network relying on entry of information is not always practical. Additionally, DNS is specialized to the task of find hosts or services and is not generally applicable to other types of resources.

[0010] Accordingly, other mechanisms for resource identification and access have been developed to attempt to address these shortcomings. A number of mechanisms include distributed lookup protocols that are more scalable than DNS. These mechanisms use various node arrangements and routing algorithms to route requests to corresponding resources and to store information for lookup.

[0011] At least one of these mechanisms utilizes local multi-level neighbor maps at each node in a network to route messages to a destination node. This essentially results in an architecture where each node is a "root node" of a corresponding tree of nodes (the nodes in its neighbor map). Messages are incrementally routed to a destination ID digit by digit (e.g., ***6=>**46=>, *346=>2346, where *s represent wildcards). The routing efficiency of these types of mechanisms is O (log N) routing hops and require nodes to maintain a routing table of O (log N) size.

[0012] At least one other of these mechanisms assigns nodes a unique ID that is taken from a linear ring of numbers. Nodes maintain routing tables that contain pointers to their immediate successor node (according to ID value) and to those nodes whose ID values are the closest successor of the value ID+2^L. The routing efficiency of these types of mechanisms is also O (log N) routing hops and require nodes to maintain a routing table of O (log N) size.

[0013] At least one further mechanisms requires O (log $N^{1/d}$) routing hops and requires nodes to maintain a routing table of O (D) size. Thus, the routing efficiency of all of these mechanisms depends, at least in part, on the number of nodes in the system.

[0014] Further, since IDs (for at least some of the mechanisms) can be uniformly distributed around a ring, there is always some possibility that routing between nodes on the ring will result in some inefficiency. For example, routing hops can cross vast geographic distances, cross more expensive links, or pass through insecure domains, etc. Additionally, when message routing involves multiple hops, there is some chance that such events will occur multiple times. Unfortunately, these mechanisms do not take into account the proximity of nodes (physical or otherwise) with respect one another. For example, depending on node distribution on a ring, routing a message from New York to Boston could

involve routing the message from New York, to London, to Atlanta, to Tokyo, and then to Boston.

[0015] Accordingly, at least one other more recent mechanism takes proximity into account by defining proximity as a single scalar proximity metric (e.g., IP routing hops or geographic distance). These mechanisms use the notion of proximity-based choice of routing table entries. Since there are many "correct" node candidates for each routing table entry, these mechanisms attempt to select a proximally close node from among the candidate nodes. For these mechanisms can provide a function that allows each node to determine the "distance" of a node with a given IP address to itself. Messages are routed between nodes in closer proximity to make progress towards a destination before routing to a node that is further away. Thus, some resources can be conserved and routing is more efficient.

[0016] Unfortunately, these existing mechanisms typically do not provide for, among other things, symmetric relationships between nodes (i.e., if a first node considers a second node to be its partner, the second node considers the first node as a partner as well), routing messages in both directions (clockwise and counterclockwise) on a ring, partitioning linked lists of nodes based on a plurality of proximity metrics, and routing messages based on a plurality of proximity metrics proximity. Therefore systems, methods, computer program products that utilize these mechanisms to rendezvous resource requests with a corresponding resource would be advantageous.

BRIEF SUMMARY OF THE INVENTION

[0017] The foregoing problems with the prior state of the art are overcome by the principles of the present invention, which are directed towards methods, systems, and computer program products for rendezvousing resource requests with corresponding resources. In some embodiments, the nodes of a federation infrastructure are partitioned. A sorted linked list containing node IDs that have been assigned to nodes in the federation infrastructure is accessed. Proximity categories that represent a plurality of different proximity criterion for partitioning the sorted link list are accessed. The sorted linked list is partitioned into one or more first sub lists based on a first proximity criterion, each of the one or more first sub lists containing at least a subset of the node IDs from the sorted linked list. A first sub list, selected from among the one or more first sub lists, is partitioned in to one or more second sub lists based on a second proximity criterion, each of the one or more second sub lists containing at least a subset of node IDs contained in the first sub list.

[0018] In other embodiments, for example as depicted in FIG. **3**, a node routing table is populated. An immediate predecessor node is inserted into the routing table. An immediate successor node is inserted into the routing table. Appropriate neighborhood node identifiers are inserted into the routing table, the neighborhood nodes identified from the sorted linked list in both the first direction and in a second opposite direction based on a predetermined or estimated neighborhood range and neighborhood size. Appropriate routing nodes identifiers are inserted into the routing table, the routing nodes identified from the sorted linked list in both the first and second directions based on the number base and field size of the ID space for the federation infrastructure, the routing nodes representing a logarithmic index of the sorted link list in both the first and second directions.

[0019] In yet other embodiments, a node routing table can be populated taking proximity criteria in to account. A predecessor node for each hierarchically partitioned routing ring the current node participates in is inserted into a routing table, each hierarchically partitioned routing ring being partitioned in accordance with corresponding proximity criteria and containing at least subsets of the bi-directional linked list of a parent ring. A successor node for each hierarchically partitioned routing ring the current node participates is inserted into the routing table. Appropriate neighborhood nodes for each hierarchically partitioned routing ring the current node participates in are inserted into the routing table. Appropriate routing nodes for each hierarchically partitioned routing ring the current node participates in are inserted into the routing table.

[0020] In further other embodiments, a message is routed, potentially based on one or more proximity criteria defining a corresponding one or more classes of nodes, towards a destination node. A receiving node receives a message along with a destination number indicating a destination node and optionally one or more proximity criteria. The receiving node, potentially among nodes in a current class of nodes, determines it is at least one of numerically further from the destination number than a corresponding predecessor node and numerically further from the destination number than a corresponding successor node. It is determined that the destination is not in a neighborhood set of nodes, potentially among nodes in the current class of nodes, corresponding to the receiving node.

[0021] An intermediate node from a routing table corresponding to the receiving node is identified, the intermediate node being numerically closer to the destination number than other routing nodes in the corresponding routing table. The message is sent to the intermediate node. The intermediate node can continue routing the message. The message eventually reaches the destination node when a node that receives the message is numerically closer to the destination number than either its successor or predecessor nodes. In embodiments that route based on one or more proximity criteria, this numerical closeness may be with respect to nodes in a selected class of nodes.

[0022] Thus, routing a message based on proximity criteria includes routing to a destination node (ID) by progressively moving closer to the destination node within a given proximal ring (class of nodes) until no further progress can be made by routing within that ring. Determining that no further progress can be made occurs when the destination number lies between the current node's ID and its successor or predecessor nodes' IDs. At this point, the current node starts routing via its partner nodes in the next larger proximal ring in which it participates. This process of progressively moving towards the destination node by climbing along the partitioning path towards the root ring terminates when the destination node is reached.

[0023] These and other objects and features of the present invention will become more fully apparent from the following description and appended claims, or may be learned by the practice of the invention as set forth hereinafter.

BRIEF DESCRIPTION OF THE DRAWINGS

[0024] To further clarify the above and other advantages and features of the present invention, a more particular description of the invention will be rendered by reference to specific embodiments thereof which are illustrated in the

appended drawings. It is appreciated that these drawings depict only typical embodiments of the invention and are therefore not to be considered limiting of its scope. The invention will be described and explained with additional specificity and detail through the use of the accompanying drawings in which:

[0025] FIG. **1** illustrates an example of a federation infrastructure.

[0026] FIG. **2** illustrates an example of a computer architecture that facilitates routing request indirectly to partners.

[0027] FIG. **3** illustrates an example binary relationship between nodes in a federation infrastructure in the form of a sorted list and corresponding ring.

[0028] FIG. **4** illustrates an example ring of rings that facilitates proximal routing.

[0029] FIG. **5** illustrates an example proximity induced partition tree of rings that facilitates proximal routing.

[0030] FIG. **6** illustrates a suitable operating environment for the principles of the present invention.

[0031] FIG. **7** illustrates an example flow chart of a method for populating a node routing table that takes proximity criteria into account

[0032] FIG. **8** illustrates an example flow chart of a method for partitioning the nodes of a federation infrastructure.

[0033] FIG. **9** illustrates an example flow chart of a method for populating a node routing table.

[0034] FIG. **10** illustrates an example flow chart of a method for numerically routing a message towards a destination node.

[0035] FIG. **11** illustrates an example flow chart of a method for proximally routing a message towards a destination node.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

[0036] The foregoing problems with the prior state of the art are overcome by the principles of the present invention, which are directed towards methods, systems, and computer program products for rendezvousing resource requests with corresponding resources. In some embodiments, the nodes of a federation infrastructure are partitioned. A sorted linked list containing node IDs that have been assigned to nodes in the federation infrastructure is accessed. Proximity categories that represent a plurality of different proximity criterion for partitioning the sorted link list are accessed. The sorted linked list is partitioned into one or more first sub lists based on a first proximity criterion, each of the one or more first sub lists containing at least a subset of the node IDs from the sorted linked list. A first sub list, selected from among the one or more first sub lists, is partitioned in to one or more second sub lists based on a second proximity criterion, each of the one or more second sub lists containing at least a subset of node IDs contained in the first sub list.

[0037] In other embodiments, for example as depicted in FIG. **3**, a node routing table is populated. An immediate predecessor node is inserted into the routing table. An immediate successor node is inserted into the routing table. Appropriate neighborhood node identifiers are inserted into the routing table, the neighborhood nodes identified from the sorted linked list in both the first direction and in a second opposite direction based on a predetermined or estimated neighborhood range and neighborhood size. Appropriate routing nodes identifiers are inserted into the routing table, the routing nodes identified from the sorted linked list in both the first and second directions based on the number base and field size of the ID space for the federation infrastructure, the routing nodes representing a logarithmic index of the sorted link list in both the first and second directions.

[0038] In yet other embodiments, a node routing table can be populated taking proximity criteria in to account. A predecessor node for each hierarchically partitioned routing ring the current node participates in is inserted into a routing table, each hierarchically partitioned routing ring being partitioned in accordance with corresponding proximity criteria and containing at least subsets of the bi-directional linked list of a parent ring. A successor node for each hierarchically partitioned routing ring the current node participates is inserted into the routing table. Appropriate neighborhood nodes for each hierarchically partitioned routing ring the current node participates in are inserted into the routing table. Appropriate routing nodes for each hierarchically partitioned routing ring the current node participates in are inserted into the routing table.

[0039] In further other embodiments, a message is routed, potentially based on one or more proximity criteria defining a corresponding one or more classes of nodes, towards a destination node. A receiving node receives a message along with a destination number indicating a destination node and optionally one or more proximity criteria. The receiving node, potentially among nodes in a current class of nodes, determines it is numerically further from the destination number than a corresponding predecessor node and numerically further from the destination number than a corresponding successor node. It is determined that the destination is not in a neighborhood set of nodes, potentially among nodes in the current class of nodes, corresponding to the receiving node.

[0040] An intermediate node from a routing table corresponding to the receiving node is identified, the intermediate node being numerically closer to the destination number than other routing nodes in the corresponding routing table. The message is sent to the intermediate node. The intermediate node can continue routing the message. The message eventually reaches the destination node when a node that receives the message is numerically closer to the destination number than either its successor or predecessor nodes. In embodiments that route based on one or more proximity criteria this numerical closeness may be with respect to nodes in a selected class of nodes.

[0041] Thus, routing a message based on proximity criteria includes routing to a destination node (ID) by progressively moving closer to the destination node within a given proximal ring (class of nodes) until no further progress can be made by routing within that ring. Determining that no further progress can be made occurs when the destination number lies between the current node's ID and its successor or predecessor nodes' IDs. At this point, the current node starts routing via its partner nodes in the next larger proximal ring in which it participates. This process of progressively moving towards the destination node by climbing along the partitioning path towards the root ring terminates when the destination node is reached.

[0042] Embodiments within the scope of the present invention include computer-readable media for carrying or having computer-executable instructions or data structures stored thereon. Such computer-readable media may be any available media, which is accessible by a general-purpose or special-purpose computer system. By way of example, and not limitation, such computer-readable media can comprise physical

storage media such as RAM, ROM, EPROM, CD-ROM or other optical disk storage, magnetic disk storage or other magnetic storage devices, or any other media which can be used to carry or store desired program code means in the form of computer-executable instructions, computer-readable instructions, or data structures and which may be accessed by a general-purpose or special-purpose computer system.

[0043] In this description and in the following claims, a "network" is defined as one or more data links (of possibly different speeds) that enable the transport of electronic data between computer systems and/or modules (e.g., hardware and/or software modules). When information is transferred or provided over a network or another communications connection (either hardwired, wireless, or a combination of hardwired or wireless) to a computer system, the connection is properly viewed as a computer-readable medium. Thus, any such connection is properly termed a computer-readable medium. Combinations of the above should also be included within the scope of computer-readable media. Computer-executable instructions comprise, for example, instructions and data which cause a general-purpose computer system or special-purpose computer system to perform a certain function or group of functions. The computer executable instructions may be, for example, binaries, intermediate format instructions such as assembly language, or even source code. In some embodiments, hardware modules, such as, for example, special purpose integrated circuits or Gate-arrays are optimized to implement the principles of the present invention.

[0044] In this description and in the following claims, a "node" is defined as one or more software modules, one or more hardware modules, or combinations thereof, that work together to perform operations on electronic data. For example, the definition of a node includes the hardware components of a personal computer, as well as software modules, such as the operating system of the personal computer. The physical layout of the modules is not important. A node can include one or more computers coupled via a network. Likewise, a node can include a single physical device (such as a mobile phone or Personal Digital Assistant "PDA") where internal modules (such as a memory and processor) work together to perform operations on electronic data. Further, a node can include special purpose hardware, such as, for example, a router that includes special purpose integrated circuits.

[0045] Those skilled in the art will appreciate that the invention may be practiced in network computing environments with many types of node configurations, including, personal computers, laptop computers, hand-held devices, multi-processor systems, microprocessor-based or programmable consumer electronics, network PCs, minicomputers, mainframe computers, mobile telephones, PDAs, pagers, routers, gateways, brokers, proxies, firewalls, redirectors, network address translators, and the like. The invention may also be practiced in distributed system environments where local and remote nodes, which are linked (either by hardwired data links, wireless data links, or by a combination of hardwired and wireless data links) through a network, both perform tasks. In a distributed system environment, program modules may be located in both local and remote memory storage devices.

[0046] Federation Architecture

[0047] FIG. **1** illustrates an example of a federation infrastructure **100**. The federation infrastructure **100** includes nodes **101**, **102**, **103**, that can form different types of federating partnerships. For example, nodes **101**, **102**, **103** can be federated among one another as peers without a root node. Each of nodes **101**, **102**, and **103** has a corresponding ID **171**, **182**, and **193** respectively.

[0048] Generally, the nodes **101**, **102**, **103**, can utilize federation protocols to form partnerships and exchange information (e.g., state information related to interactions with other nodes). The formation of partnerships and exchange of information facilitates more efficient and reliable access to resources. Other intermediary nodes (not shown) can exist between nodes **101**, **102**, and **103** (e.g., nodes having IDs between **171** and **193**). Thus, a message routed, for example, between node **101** and node **103**, can be pass through one or more of the other intermediary nodes.

[0049] Nodes in federation infrastructure **100** (including other intermediary nodes) can include corresponding rendezvous protocol stacks. For example, nodes **101**, **102**, and **103** include corresponding rendezvous protocol stacks **141**, **142**, and **143** respectively. Each of the protocols stacks **141**, **142**, and **143** includes an application layer (e.g., application layers **121**, **122**, and **123**) and other lower layers (e.g., corresponding other lower layers **131**, **132**, and **133**). Each layer in a rendezvous protocol stack is responsible for different functionality related to rendezvousing a resource request with a corresponding resource.

[0050] For example, other lower layers can include a channel layer, a routing layer, and a function layer. Generally, a channel layer is responsible for reliably transporting a message (e.g., using WS-ReliableMessaging and Simple Object Access Protocol ("SOAP")) from one endpoint to another (e.g., from node **101** to node **103**). The channel layer is also responsible for processing incoming and outgoing reliable messaging headers and maintaining state related to reliable messaging sessions.

[0051] Generally, a routing layer is responsible for computing the next hop towards a destination. The routing layer is also responsible for processing incoming and outgoing addressing and routing message headers and maintaining routing state. Generally, a function layer is responsible for issuing and processing rendezvous protocol messages such as join and depart requests, pings, updates, and other messages, as well as generation of responses to these messages. The function layer processes request messages from the routing layer and sends back corresponding response messages, if any, to the originating node using the routing layer. The function layer also initiates request messages and utilizes the routing layer to have the requests messages delivered.

[0052] Generally, an application layer processes non-rendezvous protocol specific data delivered from the function layer (i.e., application messages). The function layer can access application data from the application layer and get and put application data in rendezvous protocol messages (e.g., pings and updates). That is, the function layer can cause application data to be piggybacked on rendezvous protocol messages and can cause the application data to be passed back to the application layer in receiving rendezvous protocol nodes. In some embodiments, application data is used to identify resources and resource interests. Thus, an application layer can include application specific logic and state that processes data received from and sent to the other lower layers for purposes of identifying resources and resource interests.

[0053] Federating Mechanisms

[0054] Nodes can federate using a variety of different mechanisms. A first federating mechanism includes peer nodes forwarding information to all other peer nodes. When a node is to join a federation infrastructure, the node utilizes a broadcast/multicast discovery protocol, such as, for example, WS-Discovery to announce its presence and issues a broadcast/multicast find to detect other nodes. The node then establishes a simple forwarding partnership with other nodes already present on the network and accepts new partnerships with newly joining nodes. Thereafter, the node simply forwards all application specific messages to all of its partner nodes.

[0055] A second federating mechanism includes peer nodes that most efficiently transmit application specific messages to their destination(s). When a new node is to join a federation infrastructure, the new node utilizes a broadcast/multicast discovery protocol, such as, for example, WS-Discovery to announce its presence and issues a broadcast/multicast find to detect other nodes that are part of the federation infrastructure. Upon detecting another node, the new node establishes a partnership with the other node. From the established partnership, the new node learns about the presence of other nodes already participating in federation infrastructure. It then establishes partnerships with these newly-learned nodes and accepts any new incoming partnership requests.

[0056] Both node arrivals/departures and registrations of interest in certain application specific messages are flooded through the federation infrastructure resulting in every node having global knowledge of other partner nodes and registrations of interest in application specific messages. With such global knowledge, any node can send application specific messages directly to the nodes that have expressed interest in the application specific message.

[0057] A third federating mechanism includes peer nodes indirectly forwarding all application specific messages to their destination/s. In this third mechanism, nodes are assigned identifiers (ID's), such as, for example, a 128-bit or 160-bit ID. The node responsible for a maintaining registration of interest in a given application specific message can be determined to be the one whose ID is closest to the one obtained by mapping (e.g., hashing) the destination identity (e.g. URI) of the application specific message to this 128-bit or 160-bit ID-space.

[0058] In this third mechanism, node arrivals and departures are flooded over the entire fabric. On the other hand, registrations of interest in certain application specific messages are forwarded to the nodes determined to be responsible for maintaining such registration information. For scalability, load balancing, and fault-tolerance, the node receiving registration of interest in certain application specific messages can reliably flood that registration information within its neighborhood set. The neighborhood set for a specified node can be determined to be the set of nodes having IDs within a predefined range on either side of the ID of specified node.

[0059] Similar to the second mechanism, a newly-joining node utilizes a broadcast/multicast discovery protocol, such as, for example, WS-Discovery to announce its presence and issues a local broadcast/multi-cast find to detect a node that is already part of the federation infrastructure. The new node establishes a partnership with the discovered node and uses that partnership to learn about the presence of other new nodes participating in the federation infrastructure. The new node then establishes further partnerships with the newly discovered nodes and accepts any new incoming partnership requests. The new node accepts incoming registrations of interest in certain application layer specific resources from its partners for which it is responsible and may flood them over its neighborhood set. Thus, messages can generally be forwarded to their final destination via intermediary routing nodes (e.g., that a newly joining node has partnered with or that a partner node is aware of).

[0060] In response to receiving an incoming application specific message, the new node forwards the message to the partner node that may be responsible for maintaining the registration information for the destination specified in the message. Thus, when using this third mechanism, every node in the federation infrastructure has global knowledge of all other nodes but the registration information is efficiently partitioned among the nodes. Application specific messages are transmitted to their final destination via only the partner's nodes that may have the responsibility for maintaining registration information of interest in those application specific messages. Thus, indirection is accomplished by forwarding only to the partner node that has global knowledge of the registration information of interest for the message being processed. This is in contrast to the first mechanism where the indirection is accomplished by forwarding to all the partner nodes.

[0061] A fourth federating mechanism includes peer nodes that route messages to other peer nodes. This fourth mechanism differs from the third mechanism at least in that both node arrivals/departures and registrations of interest in certain application specific messages are all routed instead being flooded. Routing protocols are designed to guarantee rendezvous between application specific messages and the registration messages that express interest in those application specific messages.

[0062] FIG. **2** illustrates an example of a computer architecture **200** that facilitates routing requests indirectly to partners. Computer architecture **200** depicts different types of computer systems and devices potentially spread across multiple local discovery scopes participating in a federation infrastructure.

[0063] Workstation **233** can include a registered PnP provider instance. To inform its partners of the presence of this PnP provider instance, workstation **233** routes registration request **201** over the federation infrastructure. Registration request **201** is initially forwarded to laptop **231**, which in turn forwards registration request **201** to message broker **237**, which in turn forwards registration request **201** to message gateway **241**. Message gateway **241** saves the registration information registration request **201** in its database and returns success message **204** to workstation **233**.

[0064] Subsequently, another registered provider instance, this time that of running services, comes alive within the workstation **233**. This time the node is aware that message gateway **241** is responsible for registrations and forwards registration request **205** to message gateway **241** directly. Message gateway **241** saves the registration information registration request **205** in its database and returns success message **206** to workstation **233**.

[0065] Subsequently, the printer **236** (e.g., a UPnP printer) is powered on and sends announcement **207**. Server **234** detects announcement **207** and routes registration request **208** to message broker **237**. Message broker **237** forwards registration request **208** to message gateway **241**. Message gate-

way **241** saves the registration information registration request **208** in its database and returns success message **210** to server **234**.

[0066] Subsequently, personal computer **242** issues lookup request **211** to discover all devices. Since personal computer **242** doesn't know where to forward lookup request **211**, it routes lookup request **211** through workstation **243**. As registration and lookup requests are routed to the same destination, the routing protocol essentially guarantees rendezvous between the two requests resulting in workstation **243** forwards find request **211** to message gateway **241**. Message gateway **241** looks up the registration information maintained by it and forwards find request **211** to both the workstation **233** and server **234**. Workstation **233** and server **234** send response messages **214** and **216** respectively to personal computer **242**.

[0067] This fourth mechanism works by routing (instead of flooding) a request to the node (message gateway **241**) that has global knowledge of the registrations specified in a request. This fourth mechanism, as will be described in further detail below, essentially guarantees that routing can be accomplished in O (log N) hops, where N is the number of nodes participating in the federation infrastructure. Since this fourth mechanism efficiently partitions both node partnership and registration information, it scales to very large networks, even the Internet.

[0068] Although a number of federating mechanisms have been described, it would be apparent to one skilled in the art, after having reviewed this description, that other federation mechanisms are possible.

[0069] Relationship Between Nodes in a Federation

[0070] Accordingly, a federation consists of a set of nodes that cooperate among themselves to form a dynamic and scalable network in which information can be systematically and efficiently disseminated and located. Nodes are organized to participate in a federation as a sorted list using a binary relation that is reflexive, anti-symmetric, transitive, total, and defined over the domain of node identities. Both ends of the sorted list are joined, thereby forming a ring. Thus, each node in the list can view itself as being at the middle of the sorted list (as a result of using modulo arithmetic). Further, the list is doubly linked so that any node can traverse the list in either direction.

[0071] Each federating node can be assigned an ID (e.g., by a random number generator with duplicate detection) from a fixed set of IDs between 0 and some fixed upper bound. Thus, adding 1 to an ID of the fixed upper bound results in an ID of zero (i.e., moving from the end of the linked list back to the beginning of the linked listed . In addition, a 1:1 mapping function from the value domain of the node identities to the nodes themselves is defined.

[0072] FIG. **3** depicts an example linked list **304** and corresponding ring **306**. Given such a ring, the following functions can be defined:

[0073] RouteNumerically (V, Msg): Given a value V from the value domain of node identities and a message "Msg," deliver the message to node X whose identity can be mapped to V using the mapping function.

[0074] Neighborhood (X, S): Neighborhood is the set of nodes on the either side of node X with cardinality equal to S.

[0075] When every node in the federation has global knowledge of the ring, RouteNumerically (V, Msg) is implemented by directly sending Msg to the node X, whose identity is obtained by applying the mapping function to V. Alternately, when nodes have limited knowledge of other nodes (e.g., only of immediately adjacent nodes), RouteNumerically (V, Msg) is implemented by forwarding the message to consecutive nodes along the ring until it reaches the destination node X.

[0076] Alternately (and advantageously), nodes can store enough knowledge about the ring to perform a distributed binary search (without having to have global knowledge or implement routing between immediately adjacent nodes). The amount of ring knowledge is configurable such that maintaining the ring knowledge has a sufficiently small impact on each node but allows increased routing performance from the reduction in the number of routing hops.

[0077] As previously described, IDs can be assigned using the "<" (less than) relation defined over a sufficiently large, bounded set of natural numbers, meaning its range is over a finite set of numbers between 0 and some fixed value, inclusive. Thus, every node participating in the federation is assigned a natural number that lies between 0 and some appropriately-chosen upper bound, inclusive. The range does not have to be tight and there can be gaps between numbers assigned to nodes. The number assigned to a node serves as its identity in the ring. The mapping function accounts for gaps in the number space by mapping a number falling in between two node identities to the node whose identity is numerically closest to the number.

[0078] This approach has a number of advantages. By assigning each node a uniformly-distributed number, there is an increased likelihood that all segments of the ring are uniformly populated. Further, successor, predecessor, and neighborhood computations can be done efficiently using modulo arithmetic.

[0079] In some embodiments, federating nodes are assigned an ID from within an ID space so large that the chances of two nodes being assigned the same ID are highly unlikely (e.g., when random number generation is used). For example, a node can be assigned an ID in the range of 0 to b^n-1, where b equals, for example, 8 or 16 and n equals, for example, 128-bit or 160-bit equivalent digits. Accordingly, a node can be assigned an ID, for example, from a range of 0 to $16^{40}-1$ (or approximately 1.461502E48). The range of 0 to $16^{40}-1$ would provide, for example, a sufficient number of IDs to assign every node on the Internet a unique ID.

[0080] Thus, each node in a federation can have:

[0081] An ID which is a numerical value uniformly distributed in the range of 0 to b^n-1; and

[0082] A routing table consisting of (all arithmetic is done modulo b^n):

[0083] Successor node (s);

[0084] Predecessor node (p);

[0085] Neighborhood nodes ($p_k, \ldots, p_1, p, s, s_1, \ldots, s_j$) such that s_j.s.id>(id+u/2), j≧v/2−1, and p_k.p.id<(id−u/2), and k≧v/2−1; and

[0086] Routing nodes ($r_{-(n-1)}, \ldots, r_{-1}, r_1, \ldots, r_{n-1}$) such that $r_{\pm i}$=RouteNumerically ($id \pm b^i$, Msg).

where b is the number base, n is the field size in number of digits, u is the neighborhood range, v is the neighborhood size, and the arithmetic is performed modulo b^n. For good routing efficiency and fault tolerance, values for u and v can be u=b and v≧max($\log_2(N)$, 4), where N is the total number of nodes physically participating in the federation. N can be estimated from the number of nodes present on a ring segment whose length is greater than or equal to b, for example, when

there is a uniform distribution of IDs. Typical values for b and n are b=8 or 16 and n=128-bit or 160-bit equivalent digits.

[0087] Accordingly, routing nodes can form a logarithmic index spanning a ring. Depending on the locations of nodes on a ring, a precise logarithmic index is possible, for example, when there is an existing node at each number in the set of $id \pm b^i$ where i=(1, 2, . . . (n−1)). However, it may be that there are not existing nodes at each number in the set. IN those cases, a node closest to $id \pm b^i$ can be selected as a routing node. The resulting logarithmic index is not precise and may even lack unique routing nodes for some numbers in the set.

[0088] Referring again to FIG. **3**, FIG. **3** illustrates an example of a binary relation between nodes in a federation infrastructure in the form of sorted list **304** and corresponding ring **306**. The ID space of sorted list **304** is in the range 0 to 2^8-1 (or 255). That is, b=2 and n=8. Thus, nodes depicted in FIG. **3** are assigned IDs in a range from 0 to 255. Sorted list **304** utilizes a binary relation that is reflexive, anti-symmetric, transitive, total, and defined over the domain of node identities. Both ends of sorted list **304** are joined, thereby forming ring **306**. This makes it possible for each node in FIG. **3** to view itself as being at the middle of sorted list **304**. The sorted list **304** is doubly linked so that any node can traverse the sorted list **304** in either direction. Arithmetic for traversing sorted list **304** (or ring **306**) is performed modulo 2^8. Thus, 255 (or the end of sorted list **304**)+1=0 (or the beginning of sorted list **304**).

[0089] The routing table indicates that the successor to ID **64** is ID **76** (the ID immediately clockwise from ID **64**). The successor can change, for example, when a new node (e.g., with an ID of **71**) joins or an existing node (e.g., ID **76**) leaves the federation infrastructure. Likewise, the routing table indicates that the predecessor to ID **64** is ID **50** (the ID immediately counters clockwise from ID **64**). The predecessor can change, for example, when a new node (e.g., with an ID of **59**) joins or an existing node (e.g., ID **50**) leaves the federation infrastructure.

[0090] The routing table further indicates that a set of neighborhood nodes to ID **64** have IDs **83**, **76**, **50** and **46**. A set of neighbor nodes can be a specified number of nodes (i.e., neighborhood size v) that are within a specified range (i.e., neighbor range u) of ID **64**. A variety of different neighborhood sizes and neighbor ranges, such as, for example, V=4 and U=10, can potentially be used to identify the set of neighborhood nodes. A neighborhood set can change, for example, when nodes join or leave the federation infrastructure or when the specified number of nodes or specified range is changed.

[0091] The routing table further indicates that ID **64** can route to nodes having IDs **200**, **2**, **30**, **46**, **50**, **64**, **64**, **64**, **64**, **76**, **83**, **98**, **135**, and **200**. This list is generated by identifying the node closest to each number in the set of $id \pm 2^i$ where i=(1, 2, 3, 4, 5, 6, 7). That is, b=2 and n=8. For example, the node having ID **76** can be identified from calculating the closest node to $64+2^3$, or 72.

[0092] A node can route messages (e.g., requests for access to resources) directly to a predecessor node, a successor node, any node in a set of neighborhood nodes, or any routing node. In some embodiments, nodes implement a numeric routing function to route messages. Thus, RouteNumerically (V, Msg) can be implemented at node X to deliver Msg to the node Y in the federation whose ID is numerically closest to V, and return node Y's ID to node X. For example, the node having ID **64** can implement RouteNumerically (243, Msg) to cause a message to be routed to the node having ID **250**. However, since ID **250** is not a routing node for ID **64**, ID **64** can route the message to ID **2** (the closest routing node to 243). The node having ID **2** can in turn implement RouteNumerically (243, Msg) to cause the message to be routed (directly or through further intermediary nodes) to the node having ID **250**. Thus, it may be that a RouteNumerically function is recursively invoked with each invocation routing a message closer to the destination.

[0093] Advantageously, other embodiments of the present invention facilitate partitioning a ring into a ring of rings or tree of rings based on a plurality of proximity criteria of one or more proximity categories (e.g., geographical boundaries, routing characteristics (e.g., IP routing hops), administrative domains, organizational boundaries, etc.). It should be understood a ring can be partitioned more than once using the same type of proximity criteria. For example, a ring can be partition based on a continent proximity criteria and a country proximity criteria (both of a geographical boundaries proximity category).

[0094] Since IDs can be uniformly distributed across an ID space (a result of random number generation) there is a high probability that any given segment of a circular ID space contains nodes that belong to different proximity classes provided those classes have approximately the same cardinality. The probability increases further when there are a sufficient number of nodes to obtain meaningful statistical behavior.

[0095] Thus, neighborhood nodes of any given node are typically well dispersed from the proximality point of view. Since published application state can be replicated among neighborhood nodes, the published information can be well dispersed as well from the proximality point of view.

[0096] FIG. **4** illustrates a ring of rings **400** that facilitates proximal routing. Ring **401** can be viewed as a master or root ring, and contains all the nodes in each of the rings **402**, **403**, and **404**. Each of the rings **402**, **403**, and **404** contain a subset of nodes from ring **401** that are partitioned based on a specified proximity criterion. For example, ring **401** may be partitioned based on geographic location, where ring **402** contains nodes in North America, ring **403** contains nodes in Europe, and ring **404** contains nodes in Asia.

[0097] In a numerical space containing 65,536 (2^{16}) IDs, routing a message from a North American node having an ID **5,345** to an Asian node having an ID **23,345** can include routing the message within ring **402** until a neighbor node of the Asian node is identified. The neighbor node can then route the message to the Asian node. Thus, a single hop (as opposed to multiple hops) is made between a North American node and an Asian node. Accordingly, routing is performed in a resource efficient manner.

[0098] FIG. **5** illustrates an example proximity induced partition tree of rings **500** that facilitates proximal routing. As depicted, partition tree of rings **500** includes a number of rings. Each of the rings represents a partition of a sorted linked list. Each ring including a plurality a nodes having IDs in the sorted linked list. However for clarity due to the number of potential nodes, the nodes are not expressly depicted on the rings (e.g., the ID space of partition tree **500** may be b=16 and n=40).

[0099] Within partition tree **500**, root ring **501** is partitioned into a plurality of sub-rings, including sub-rings **511**, **512**, **513**, and **514**, based on criterion **571** (a first administrative domain boundary criterion). For example, each component of a DNS name can be considered a proximity criterion with the partial order among them induced per their order of appear-

ance in the DNS name read right to left. Accordingly, sub-ring **511** can be further partitioned into a plurality of sub-rings, including sub-rings **521**, **522**, and **523**, based on criterion **581** (a second administrative domain boundary criterion).

[0100] Sub-ring **522** can be further partitioned into a plurality of sub-rings, including sub-rings **531**, **532**, and **533**, based on criterion **572** (a geographic boundary criterion). Location based proximity criterion can be partially ordered along the lines of continents, countries, postal zip codes, and so on. Postal zip codes are themselves hierarchically organized meaning that they can be seen as further inducing a partially ordered sub-list of proximity criteria.

[0101] Sub-ring **531** can be further partitioned into a plurality of sub-rings, including sub-rings **541**, **542**, **543**, and **544**, based on criterion **573** (a first organizational boundary criterion). A partially ordered list of proximity criterion can be induced along the lines of how a given company is organizationally structured such as divisions, departments, and product groups. Accordingly, sub-ring **543** can be further partitioned into a plurality of sub-rings, including sub-rings **551** and **552**, based on criterion **583** (a second organizational boundary criterion).

[0102] Within partition tree **500**, each node has a single ID and participates in rings along a corresponding partition path starting from the root to a leaf. For example, each node participating in sub-ring **552** would also participate in sub-rings **543**, **531**, **522**, **511** and in root **501**. Routing to a destination node (ID) can be accomplished by implementing a RouteProximally function, as follows:

[0103] RouteProximally (V, Msg, P): Given a value V from the domain of node identities and a message "Msg," deliver the message to the node Y whose identity can be mapped to V among the nodes considered equivalent by the proximity criteria P.

[0104] Thus, routing can be accomplished by progressively moving closer to the destination node within a given ring until no further progress can be made by routing within that ring as determined from the condition that the destination node lies between the current node and its successor or predecessor node. At this point, the current node starts routing via its partner nodes in the next larger ring in which it participates. This process of progressively moving towards the destination node by climbing along the partitioning path towards the root ring terminates when the closest node to the destination node is reached within the requested proximal context, as originally specified in the RouteProximally invocation.

[0105] Routing hops can remain in the proximal neighborhood of the node that originated the request until no further progress can be made within that neighborhood because the destination node exists outside it. At this point, the proximity criterion is relaxed to increase the size of the proximal neighborhood to make further progress. This process is repeated until the proximal neighborhood is sufficiently expanded to include the destination node (ID). The routing hop made after each successive relaxation of proximal neighborhood criterion can be a potentially larger jump in proximal space while making a correspondingly smaller jump in the numerical space compared to the previous hop. Thus, only the absolutely required number of such (inter-ring) hops is made before the destination is reached.

[0106] It may be the case that some hops are avoided for lookup messages since published application data gets replicated down the partition tree when it is replicated among the neighborhood nodes of the destination node.

[0107] To accomplish proximal routing, each federation node maintains references to its successor and predecessor nodes in all the rings it participates as a member (similar to successor and predecessor for a single ring)—the proximal predecessor, proximal successor, and proximal neighborhood. In order to make the routing efficient, the nodes can also maintain reference to other nodes closest to an exponentially increasing distance on its either half of the ring as routing partners (similar to routing nodes for a single ring). In some embodiments, routing partner nodes that lie between a pair of consecutive successor or predecessor nodes participate in the same lowest ring shared by the current node and the node numerically closest to it among the successor or predecessor node pairs respectively. Thus, routing hops towards a destination node transition into using a relaxed proximity criterion (i.e., transitioning to a higher ring) only when absolutely needed to make further progress. Accordingly, messages can be efficiently rendezvoused with a corresponding federation node.

[0108] In some embodiments, nodes implement a proximal routing function to route messages based on equivalence criteria relations. Thus, given a number V and a message "Msg", a node can implement RouteProximally (V, Msg, P) to deliver the message to the node Y whose identify can be mapped to V among the nodes considered equivalent by proximity criterion P. The proximity criterion P identifies the lowest ring in the partition tree that is the common ancestor to all the nodes considered proximally equivalent by it. It can be represented as a string obtained by concatenating the proximity criterion found along the path from the root ring to the ring identified by it separated by the path separator character '/'. For example, the proximity criterion identifying sub-ring **542** can be represented as "Proximity:/.COM/Corp2/LocationA/Div2". Each ring in the partition tree **500** can be assigned a unique number, for example, by hashing its representational string with a SHA based algorithm. If the number 0 is reserved for the root ring, it can be inferred that RouteNumerically (V, Msg)≡RouteProximally (V, Msg, 0).

[0109] For example, a node in sub-ring **544** can implement RouteProximally to identify a closer node in sub-ring **531** (e.g., to a node in sub-ring **513**). In turn, sub-ring **531** can implement RouteProximally to identify a closer node in sub-ring **522**. Likewise, sub-ring **522** can implement RouteProximally to identify a closer node in sub-ring **511**. Similarly, sub-ring **511** can implement RouteProximally to identify a closer node in ring **501**. Thus, it may be that a RouteProximally function is recursively invoked with each invocation routing a message closer to the destination.

[0110] Thus, when proximity criterion is taken into account, routing hops on a path to a final destination can remain within the proximity of a node that originates a request, while making significant progress between the originating node and the destination node in a numerical space, until either the destination node is reached or no further progress can be made under the chosen proximity criterion at which point it is relaxed just enough to make further progress towards the destination. For example, proximity criterion can be relaxed enough for a message to be routed from ring **531** up to ring **522**, etc.

[0111] Utilizing the above approach to proximity, it is possible to confine published information to a given ring. For example, organizations may like to ensure that organization specific information is not available to entities outside of their trust domains either (1) implicitly in the form of neighbor-

hood replication to nodes outside of their domains or (2) explicitly in the form of servicing lookup requests for such information. The first aspect is satisfied by replicating published information only among the nodes neighboring the target ID within the specified ring. Because all messages originated by a node are routed by successively climbing the rings to which it belongs towards the root ring, there is a high likelihood that all lookup requests originated within an organization will be able to locate the published information confined to it thereby implicitly satisfying the second aspect.

[0112] Also, organizations dislike nodes automatically federating with nodes outside of their trust domain. This can happen, for example, when a visiting sales person connects his/her laptop computer to the network in the customer premises. Ideally, the laptop computer belonging to the sales person wishes to locate information published in its home domain and/or federate with the nodes in its home domain starting at its lowest preferred proximity ring. It will typically not be permitted to federate with the nodes in the customer's domain. Supporting this scenario requires ability to locate seed nodes in the home domain. Such seed nodes can be used for locating information published in the home domain, to join the home federation, and selectively import and export published information across domains. Seed nodes are also sometimes referred as message gateways.

[0113] In other embodiments, an entity publishes references to seed nodes in the root ring. Seed nodes can be published at the unique number (such as the one obtained by hashing its representational string) associated with the ring (as a target ID). Seed node information can further be on-demand cached by the nodes in various rings that are on the path to the corresponding target IDs in the root ring. Such on-demand caching provides for improved performance and reduction in hotspots that might occur when semi-static information is looked up quite frequently. Seed node information can also be obtained via other means such as DNS

[0114] To provide fault tolerance for confined published information, each node can maintain a set of neighborhood nodes in all of the rings it participates in. Given the above, the state maintained by a node can be summarized as follows:

[0115] An ID which is a numerical value uniformly distributed in the range of 0 to b^n-1.

[0116] A routing table consisting of (all arithmetic is done modulo b^n):

[0117] For each ring, say ring d, in which the node participates

[0118] Successor node (s_d)

[0119] Predecessor node (p_d)

[0120] Neighborhood nodes (p_{kd}, . . . , p_{1d}, p_d, s_d, s_{1d}, . . . , s_{jd}) such that $s_{jd}.s_d.id>(id+u/2)$, $j\geqq v/2-1$, $p_{kd}.p_d.id<(id-u/2)$, and $k\geqq v/2-1$.

[0121] Routing nodes ($r_{-(n-1)}$, . . . , r_{-1}, r_1, . . . , r_{n-1}) such that $r_{\pm i}$=RouteProximally ($id\pm b^i$, updateMsg, d) such that $s_d\leqq id+b^i\leqq s_{d+1}$ or $p_{d+1}\leqq id-b^i\leqq p_d$ as appropriate.

where b is the number base, n is the field size in number of digits, u is the neighborhood range, and v is the neighborhood size.

[0122] Note that a subset of the neighborhood nodes maintained by a given node in ring "d" can appear again as neighborhood nodes in the child ring "d+1" in which the given node participates as well. As such one can derive the upper bound on the total number of neighborhood nodes maintained by a given node across all the D rings it participates as D*max(u, v)/2. This considers that only one reference to a given node is kept and the worst case upper bound is for a balanced tree.

[0123] It should be noted that when a ring is partitioned into a plurality of corresponding sibling sub-rings, it is permitted for a specified node to simultaneously participate in more than one of the plurality of corresponding sibling sub-rings, for example, through aliasing. Aliasing can be implemented to associate different state, for example, from different sub-rings, with the specified node. Thus, although aliases for a given node have the same ID, each alias can have distinct state associated with them. Aliasing allows the specified node to participate in multiple rings having distinct proximity criteria that are not necessarily common ancestors of more specific proximity criteria. That is, the specified node can participate in multiple branches of the proximity tree.

[0124] For example, a dual NIC (wired and wireless) laptop can be considered to be proximally equivalent to both other wireless and wired nodes sharing the same LAN segments as the laptop. But, these two distinct proximity criteria can be modeled as sub-criteria that are applicable only after application of a different higher priority proximity criterion, such as, for example, one based on organizational membership. As the laptop belongs to the same organization, the aliased nodes in the two sub-rings representing 1) membership in the wired and 2) membership in the wireless LAN segments merge into a single node in the ring representing the organization to which the laptop belongs. It should be understand that the RouteProximally works as expected without any modifications in the presence of aliasing.

[0125] Each proximal ring can be configured in accordance with (potentially different) ring parameters. Ring parameters can be used to define a neighborhood (e.g., ring parameters can represent a neighborhood range, a neighborhood size, ping message and depart message timing and distribution patterns for ping and depart messages), indicate a particular federating mechanisms (e.g., from among the above-described first through fourth federating mechanisms previously described or from among other federating mechanisms), or define communication specifics between routing partners in the same proximal ring. Some ring parameters may be more general, applying to a plurality of different federating mechanisms, while other ring parameters are more specific and apply to specific type of federating mechanism.

[0126] Ring parameters used to configure a higher level proximal ring can be inherited in some embodiments by lower level proximal rings. For example, it may be that ring **543** inherits some of the ring parameters of ring **531** (which in turn inherited from ring **522**, etc.). Thus, a neighborhood size and neighborhood range associated with ring **531** is also associated with ring **541**.

[0127] However, inherited ring parameters can be altered and/or proximal rings can be individually configured in accordance with different ring parameters. For example, it may be that ring **511** is for an administrative domain that contains a large number of nodes and thus the above-described fourth federating mechanism is more appropriate for ring **511**. On the other hand, it may be that ring **521** is for a small business with a relatively smaller number of nodes and thus the above-described second federating mechanism is more appropriate for ring **521**. Thus, the ring parameters associated with ring **521** can be set to (or inherited parameters changed to) different values than the ring parameters associated with ring **511**. For example, a ring parameter indicating a particular type of federating mechanisms can be different

between rings **511** and **521**. Similarly parameters defining a neighborhood can be different between rings **511** and **521**. Further, ring **521** can be configured in accordance with specific parameters that are specific to the above-described second federating mechanism, while ring **511** is configured in accordance additional with specific parameters that are specific to the above-described fourth federating mechanism.

[0128] Accordingly, proximal rings can be flexibly configured based on the characteristics (e.g., number, included resources, etc.) of nodes in the proximal rings. For example, an administrator can select ring parameters for proximal rings using a configuration procedure (e.g., through a user-interface). A configuration procedure can facilitate the configuration of inheritance relationships between proximal rings as well as the configuration of individual proximal rings, such as, for example, to override otherwise inherited ring parameters.

[0129] FIG. **8** illustrates an example flow chart of a method **800** for partitioning the nodes of a federation infrastructure. The method **800** will be described with respect to the rings of partition a tree **500** in FIG. **5**. Method **800** includes an act of accessing a sorted linked list containing node IDs that have been assigned to nodes in a federation infrastructure (act **801**). For example, the sorted linked list represented by ring **501** can be accessed. The node IDs of the sorted linked list (the nodes depicted on ring **501**) can represent nodes in a federation infrastructure (e.g., federation infrastructure **100**).

[0130] Method **800** includes an act of accessing proximity categories that represent a plurality of different proximity criteria for partitioning the sorted linked list (act **802**). For example, proximity criterion representing domain boundaries **561**, geographical boundaries **562**, and organizational boundaries **563** can be accessed. However, other proximity criteria, such as, trust domain boundaries, can also be represented in accessed proximity criterion. Proximity categories can include previously created partially ordered lists of proximity criteria. A ring can be partitioned based on partially ordered lists of proximity criteria.

[0131] Method **800** includes an act of partitioning the sorted link list into one or more first sub lists based on a first proximity criterion, each of the one or more first sub lists containing at least a subset of the node IDs from the sorted linked list (act **803**). For example, ring **501** can be partitioned into sub-rings **511**, **512**, **513**, and **514** based on criterion **571**. Each of sub-rings **511**, **512**, **513**, and **514** can contain a different sub-set of node IDs from ring **501**.

[0132] Method **800** includes an act of partitioning a first sub list, selected from among the one or more first sub lists, into one or more second sub lists based on a second proximity criterion, each of the one or more second sub lists containing at least a subset of node IDs contained in the first sub list (act **804**). For example, sub-ring **511** can be partitioned into sub-rings **521**, **522**, and **523** based on criterion **581**. Each of the sub-rings **521**, **522**, and **523** can contain a different sub-set of node IDs from sub-ring **511**.

[0133] FIG. **9** illustrates an example flow chart of a method **900** for populating a node's routing table. The method **900** will be described with respect to the sorted linked list **304** and ring **306** in FIG. **3**. Method **900** includes an act of inserting a predecessor node into a routing table, the predecessor node preceding a current node relative to the current node in a first direction of a sorted linked list (act **901**). For example, the node having ID **50** can be inserted into the routing table as a predecessor for the node having ID **64** (the current node). Moving in a clockwise direction **321** (from end A of sorted linked list **304** towards end B of sorted linked list **304**), the node having ID **50** precedes the node having ID **64**. Inserting a predecessor node can establish a symmetric partnership between the current node and the predecessor node such that current node is a partner of predecessor node and the predecessor node is a partner of the current node

[0134] Method **900** includes an act of inserting a successor node into the routing table, the successor node succeeding the current node relative to the current node in the first direction in the sorted linked list (act **902**). For example, the node having ID **76** can be inserted into the routing table as a successor for the node having ID **64** (the current node). Moving in a counter-clockwise direction **322**, the node having ID **76** succeeds the node having ID **64**. Inserting a successor node can establish a symmetric partnership between the current node and the successor node such that current node is a partner of the successor node and the successor node is a partner of the current node.

[0135] Method **900** includes an act of inserting appropriate neighborhood nodes into the routing table, the neighborhood nodes identified from the sorted linked list in both the first direction and in a second opposite direction based on a neighborhood range and neighborhood size (act **903**). For example, the nodes having IDs **83**, **76**, **50**, and **46** can be inserted into the routing table as neighborhood nodes for the node having ID **64** (the current node). Based on a neighborhood range of 20 and a neighborhood size 4, the nodes having IDs **83** and **76** can be identified in clockwise direction **321** and the nodes having IDs **50** and **46** can be identified in counter-clockwise direction **322** (moving from end B of sorted linked list **304** towards end A of sorted linked list **304**). It may be that in some environments no appropriate neighborhood nodes are identified. Inserting a neighborhood node can establish a symmetric partnership between the current node and the neighborhood node such that current node is a partner of the neighborhood node and the neighborhood node is a partner of the current node.

[0136] Method **900** includes an act of inserting appropriate routing nodes into the routing table, the routing nodes identified from the sorted linked list in both the first and second directions based on the a number base and field size of the ID space for the federation infrastructure, the routing nodes representing a logarithmic index of the sorted link list in both the first and second directions (act **904**). For example, the nodes having IDs **200**, **2**, **30**, **46**, **50**, **64**, **64**, **64**, **64**, **64**, **76**, **83**, **98**, **135** and **200** can be inserted into the routing table as routing nodes for the node having ID **64**. Based on the number base 2 and field size of 8 the nodes having IDs **64**, **64**, **76**, **83**, **98**, **135** and **200** can be identified in direction **321** and the nodes having IDs **64**, **64**, **50**, **46**, **30**, **2**, and **200** can be identified in direction **322**. As depicted inside ring **306**, the routing nodes represent a logarithmic index of the sorted link list **304** in both clockwise direction **321** and counter-clockwise direction **322**. Inserting a routing node can establish a symmetric partnership between the current node and the routing node such that current node is a partner of the routing node and the routing node is a partner of the current node.

[0137] FIG. **7** illustrates an example flow chart of a method **700** for populating a node routing table that takes proximity criteria into account. The method **700** will be described with respect to the rings in FIG. **5**. Method **700** includes an act of inserting a predecessor node for each hierarchically partitioned routing ring the current node participates in into a

routing table (act **701**). Each predecessor node precedes the current node in a first direction (e.g., clockwise) within each hierarchically partitioned routing ring the current node participates in. The hierarchically partitioned routing rings are partitioned in accordance with corresponding proximity criteria and contain at least subsets of a bi-directionally linked list (and possibly the whole bi-directionally linked list). For example, it may be that a specified node participates in root ring **501** and sub-rings **511**, **522**, **523**, **531**, and **542**. Thus, a predecessor node is selected for the specified node from within each of the rings **501** and sub-rings **511**, **522**, **523**, **531**, and **542**.

[0138] Method **700** includes an act of inserting a successor node for each hierarchically partitioned routing ring the current node participates in into the routing table (act **702**). Each successor node succeeding the current node in the first direction within each hierarchically partitioned routing ring the current node participates in. For example, a successor node is selected for the specified node from within each of the rings **501** and sub-rings **511**, **522**, **523**, **531**, and **542**.

[0139] Method **700** includes an act of inserting appropriate neighborhood nodes for each hierarchically partitioned routing ring the current node participates in into the routing table (act **703**). The neighborhood nodes can be identified in both the first direction (e.g., clockwise) and in a second opposite direction (e.g., counter clockwise) based on a neighborhood range and neighborhood size from the hierarchically partitioned routing rings the current node participates in. For example, neighborhood nodes can be identified for the specified node from within each of the rings **501** and sub-rings **511**, **522**, **523**, **531**, and **542**.

[0140] Method **700** includes an act of inserting appropriate routing nodes for each hierarchically partitioned routing ring the current node participates in into the routing table (act **704**). For example, routing nodes can be identified for the specified node from within each of the rings **501** and sub-rings **511**, **522**, **523**, **531**, and **542**.

[0141] In some embodiments, appropriate routing nodes are inserted for each proximity ring d except the leaf ring (or leaf rings in embodiments that utilize aliasing), in which the node Y participates. Appropriate routing nodes can be inserted based on the following expression(s):

if $Y.s_d.id < Y.id + b^i < Y.s_{d+1}.id$ is true, then use ring d; or

if $Y.p_d.id < Y.id - b^i < Y.p_{d+1}.id$ is true, then use ring d.

[0142] If a ring has not been identified in the previous step, use the lead (e.g., ring **501**) ring as ring d. Now, ring d is the proximity ring in which node Y should look for the routing partner closest to z.

[0143] FIG. **10** illustrates an example flow chart of a **1000** method for routing a message towards a destination node. The method **1000** will be described with respect to the sorted linked list **304** and ring **306** in FIG. **3**. Method **1000** includes an act of a receiving node receiving a message along with a number indicating a destination (act **1001**). For example, the node having ID **64** can receive a message indicating a destination of **212**.

[0144] Method **1000** includes an act of determining that the receiving node is at least one of numerically further from the destination than a corresponding predecessor node and numerically further from the destination than a corresponding successor node (act **1002**). For example, in direction **322**, ID **64** is further from destination **212** than ID **50** and, in direction **321**, ID **64** is further from destination **212** than ID **76**. Method **1000** includes an act of determining that the destination is not within a neighborhood set of nodes corresponding to the receiving node (act **1003**). For example, the node with ID **64** can determine that destination **212** is not within the neighborhood set of **83**, **76**, **50**, and **46**.

[0145] The method **1000** includes an act of identifying an intermediate node from a routing table corresponding to the receiving node, the intermediate node being numerically closer to the destination than other routing nodes in the corresponding routing table (act **1004**). For example, the node having ID **64** can identify the routing node having ID **200** as being numerically closer to destination **212** that other routing nodes. The method **1000** includes an act of sending the message to the intermediate node (act **1005**). For example, the node having ID **64** can send the message to the node having ID **200**.

[0146] FIG. **11** illustrates an example flow chart of a method **1100** for routing a message towards a destination node based on proximity criteria. The method **1100** will be described with respect to the rings in FIG. **4** and FIG. **5**. Method **1100** includes an act of a receiving node receiving a message along with a number indicating a destination and a proximity criterion (act **1101**). The proximity criterion defines one or more classes of nodes. The receiving node receives the message as part of a current class of nodes selected from among the one or more classes of nodes based on the proximity criterion. For example, the node having ID **172** can receive a message indicating a destination of **201** and proximity criterion indicating that the destination node be part of classes represented by ring **401**. The node having ID **172** can receive the message as part of ring **404**.

[0147] Method **1100** includes an act of determining that the receiving node is at least one of, numerically further from the destination than a corresponding predecessor node and numerically further from the destination than a corresponding successor node, among nodes in a selected class of nodes (act **1102**). For example, within ring **404**, the node with ID **172** is further from destination **201** than the node having ID **174** in the clockwise direction and is further from destination **201** than the node having ID **153** in the counterclockwise direction.

[0148] Method **1100** includes an act of determining that the destination is not within the receiving node's neighborhood set of nodes for any of the one or more classes of nodes defined by the proximity criterion (act **1103**). For example, the node having ID **172** can determine that destination **201** is not in a corresponding neighborhood set in ring **404** or in ring **401**.

[0149] Method **1100** includes an act of identifying an intermediate node from the receiving node's routing table, the intermediate node being numerically closer to the destination than other routing nodes in the routing table (act **1104**). For example, the node having ID **172** can identify the node having ID **194** as being numerically closer to destination **201** than other routing nodes in ring **404**. The method **1100** includes an act of sending the message to the intermediate node (act **1105**). For example, the node having ID **172** can send the received message to the node having ID **194**. The node having ID **172** can send the received message to the node having ID **194** to honor a previously defined partially ordered list of proximity criterion

[0150] Node **194** may be as close to destination **201** as is possible within ring **404**. Thus, proximity can be relaxed just enough to enable further routing towards the destination to be

made in ring **401** in the next leg. That is, routing is transitioned from ring **404** to ring **401** since no further progress towards the destination can be made on ring **404**. Alternately, it may be that the node having ID **201** is within the neighborhood of the node having ID **194** in ring **401** resulting in no further routing. Thus, in some embodiments, relaxing proximity criteria to get to the next higher ring is enough to cause further routing.

[0151] However, in other embodiments, incremental relaxation of proximity criteria causing transition to the next higher ring continues until further routing can occur (or until the root ring is encountered). That is, a plurality of transitions to higher rings occurs before further routing progress can be made. For example, referring now to FIG. **5**, when no further routing progress can be made on ring **531**, proximity criteria may be relaxed enough to transition to ring **511** or even to root ring **501**.

[0152] FIG. **6** and the following discussion are intended to provide a brief, general description of a suitable computing environment in which the invention may be implemented. Although not required, the invention will be described in the general context of computer-executable instructions, such as program modules, being executed by computer systems. Generally, program modules include routines, programs, objects, components, data structures, and the like, which perform particular tasks or implement particular abstract data types. Computer-executable instructions, associated data structures, and program modules represent examples of the program code means for executing acts of the methods disclosed herein.

[0153] With reference to FIG. **6**, an example system for implementing the invention includes a general-purpose computing device in the form of computer system **620**, including a processing unit **621**, a system memory **622**, and a system bus **623** that couples various system components including the system memory **622** to the processing unit **621**. Processing unit **621** can execute computer-executable instructions designed to implement features of computer system **620**, including features of the present invention. The system bus **623** may be any of several types of bus structures including a memory bus or memory controller, a peripheral bus, and a local bus using any of a variety of bus architectures. The system memory includes read only memory ("ROM") **624** and random access memory ("RAM") **625**. A basic input/output system ("BIOS") **626**, containing the basic routines that help transfer information between elements within computer system **620**, such as during start-up, may be stored in ROM **624**.

[0154] The computer system **620** may also include magnetic hard disk drive **627** for reading from and writing to magnetic hard disk **639**, magnetic disk drive **628** for reading from or writing to removable magnetic disk **629**, and optical disk drive **630** for reading from or writing to removable optical disk **631**, such as, or example, a CD-ROM or other optical media. The magnetic hard disk drive **627**, magnetic disk drive **628**, and optical disk drive **630** are connected to the system bus **623** by hard disk drive interface **632**, magnetic disk drive-interface **633**, and optical drive interface **634**, respectively. The drives and their associated computer-readable media provide nonvolatile storage of computer-executable instructions, data structures, program modules, and other data for the computer system **620**. Although the example environment described herein employs magnetic hard disk **639**, removable magnetic disk **629** and removable optical disk **631**, other types of computer readable media for storing data can be used, including magnetic cassettes, flash memory cards, digital versatile disks, Bernoulli cartridges, RAMs, ROMs, and the like.

[0155] Program code means comprising one or more program modules may be stored on hard disk **639**, magnetic disk **629**, optical disk **631**, ROM **624** or RAM **625**, including an operating system **635**, one or more application programs **636**, other program modules **637**, and program data **638**. A user may enter commands and information into computer system **620** through keyboard **640**, pointing device **642**, or other input devices (not shown), such as, for example, a microphone, joy stick, game pad, scanner, or the like. These and other input devices can be connected to the processing unit **621** through input/output interface **646** coupled to system bus **623**. Input/output interface **646** logically represents any of a wide variety of different interfaces, such as, for example, a serial port interface, a PS/2 interface, a parallel port interface, a Universal Serial Bus ("USB") interface, or an Institute of Electrical and Electronics Engineers ("IEEE") 1394 interface (i.e., a FireWire interface), or may even logically represent a combination of different interfaces.

[0156] A monitor **647** or other display device is also connected to system bus **623** via video interface **648**. Speakers **669** or other audio output device is also connected to system bus **623** via audio interface **649**. Other peripheral output devices (not shown), such as, for example, printers, can also be connected to computer system **620**.

[0157] Computer system **620** is connectable to networks, such as, for example, an office-wide or enterprise-wide computer network, a home network, an intranet, and/or the Internet. Computer system **620** can exchange data with external sources, such as, for example, remote computer systems, remote applications, and/or remote databases over such networks.

[0158] Computer system **620** includes network interface **653**, through which computer system **620** receives data from external sources and/or transmits data to external sources. As depicted in FIG. **6**, network interface **653** facilitates the exchange of data with remote computer system **683** via link **651**. Network interface **653** can logically represent one or more software and/or hardware modules, such as, for example, a network interface card and corresponding Network Driver Interface Specification ("NDIS") stack. Link **651** represents a portion of a network (e.g., an Ethernet segment), and remote computer system **683** represents a node of the network.

[0159] Likewise, computer system **620** includes input/output interface **646**, through which computer system **620** receives data from external sources and/or transmits data to external sources. Input/output interface **646** is coupled to modem **654** (e.g., a standard modem, a cable modem, or digital subscriber line ("DSL") modem) via link **659**, through which computer system **620** receives data from and/or transmits data to external sources. As depicted in FIG. **6**, input/output interface **646** and modem **654** facilitate the exchange of data with remote computer system **693** via link **652**. Link **652** represents a portion of a network and remote computer system **693** represents a node of the network.

[0160] While FIG. **6** represents a suitable operating environment for the present invention, the principles of the present invention may be employed in any system that is capable of, with suitable modification if necessary, implementing the principles of the present invention. The environment illustrated in FIG. **6** is illustrative only and by no means represents even a small portion of the wide variety of environments in which the principles of the present invention may be implemented.

[0161] In accordance with the present invention, nodes, application layers, and other lower layers, as well as associated data, including routing tables and node IDs may be stored and accessed from any of the computer-readable media associated with computer system **620**. For example, portions of such modules and portions of associated program data may be included in operating system **635**, application programs **636**, program modules **637** and/or program data **638**, for storage in system memory **622**.

[0162] When a mass storage device, such as, for example, magnetic hard disk **639**, is coupled to computer system **620**, such modules and associated program data may also be stored in the mass storage device. In a networked environment, program modules depicted relative to computer system **620**, or portions thereof, can be stored in remote memory storage devices, such as, system memory and/or mass storage devices associated with remote computer system **683** and/or remote computer system **693**. Execution of such modules may be performed in a distributed environment as previously described.

[0163] The present invention may be embodied in other specific forms without departing from its spirit or essential characteristics. The described embodiments are to be considered in all respects only as illustrative and not restrictive. The scope of the invention is, therefore, indicated by the appended claims rather than by the foregoing description. All changes, which come within the meaning and range of equivalency of the claims, are to be embraced within their scope.

What is claimed and desired secured by United States Letters Patent is:

1. In a federation infrastructure, a method for creating and maintaining a node distribution data structure, the method comprising:

an act of accessing from a current node at least one routing table in the federation infrastructure to determine identification information for each node in the federation infrastructure of which the routing table has knowledge, wherein one or more of the nodes in the routing table references at least one other node in the federation infrastructure;

an act of subsequently accessing the other nodes in the federation infrastructure by following the nodes pointed to by each node; and

based on the accessed nodes, an act of forming a node distribution data structure that points to every node in the federation infrastructure.

2. The method as recited in claim **1**, wherein the nodes and routing tables are members of the same proximal ring.

3. The method as recited in claim **2**, wherein at least the node identification information and any node references maintained in the routing table are used to address a message to at least one other node in the federation infrastructure.

4. The method as recited in claim **2**, wherein at least the node identification information and any node references maintained in the routing table are used to determine at least one node from a set of nodes to address a message to.

5. The method as recited in claim **3**, further comprising sending a plurality of messages instead of a single message.

6. The method as recited in claim **1**, wherein the nodes and routing tables are members of different proximal rings.

7. The method as recited in claim **6**, wherein at least the node identification information and any node references maintained in the routing table are used to address a message to at least one other node in the federation infrastructure.

8. The method as recited in claim **6**, wherein at least the node identification information and any node references maintained in the routing table are used to determine at least one node from a set of nodes to address a message to.

9. The method as recited in claim **1**, wherein the node distribution data structure comprises a connectivity graph.

10. In a federation infrastructure, a system configured to create and maintain a node distribution data structure, the system comprising:

an accessing module configured to access from a current node at least one routing table in the federation infrastructure to determine identification information for each node in the federation infrastructure of which the routing table has knowledge, wherein one or more of the nodes in the routing table points to at least one other node in the federation infrastructure;

a second accessing module configured to subsequently access the other nodes in the federation infrastructure by following the nodes pointed to by each node; and

a distributed forming module configured to form, based on the accessed nodes, a node distribution data structure that points to every node in the federation infrastructure.

11. The system as recited in claim **10**, wherein the nodes and routing tables are members of the same proximal ring.

12. The system as recited in claim **11**, wherein at least the node identification information and any node references maintained in the routing table are used to address a message to at least one other node in the federation infrastructure.

13. The system as recited in claim **11**, wherein at least the node identification information and any node references maintained in the routing table are used to determine at least one node from a set of nodes to address a message to.

14. The system as recited in claim **12**, further comprising sending a plurality of messages instead of a single message.

15. The system as recited in claim **10**, wherein the nodes and routing tables are members of different proximal rings.

16. The system as recited in claim **15**, wherein at least the node identification information and any node references maintained in the routing table are used to address a message to at least one other node in the federation infrastructure.

17. The system as recited in claim **15**, wherein at least the node identification information and any node references maintained in the routing table are used to determine at least one node from a set of nodes to address a message to.

18. The system as recited in claim **10**, wherein the node distribution data structure comprises a connectivity graph.

19. In a federation infrastructure, a system including a fully connected node distribution data structure, the system comprising:

a receiving module for receiving at a current node an indication that a message is to be sent to from the current node to each of the nodes of the federation infrastructure; and

a sending module for sending the message to at least one other node in the federation infrastructure, whereupon each receiving node accesses stored state information to send the message to at least one other node until all nodes have received the message at least once.

20. The system as recited in claim **19**, wherein the message is addressed to a particular identified node among the plurality of nodes in the federation infrastructure.

* * * * *