DATA TRANSFER TO NODES OF A COMMUNICATION NETWORK USING A DATA CHANNEL AND A CONTROL CHANNEL

Inventor: Michael J. Sheehan, Naperville, IL (US)

Correspondence Address:
Duff Boren & Fishman, LLP
1526 Spruce Street
Suite 320
Boulder, CO 80302 (US)

Assignee: Lucent Technologies Inc.

Appl. No.: 11/095,147
Filed: Mar. 31, 2005

Related U.S. Application Data
Continuation-in-part of application No. 10/786,911, filed on Feb. 25, 2004.

Publication Classification

Int. Cl. 7 G06F 15/16
U.S. Cl. 709/231

ABSTRACT

Methods and software products are described for efficiently transferring a data stream to nodes of a communication network. According to the invention, one of the nodes executes code to establish a control channel with one or more remote nodes. The code then replicates itself on the node, and transmits the replication of the code over the control channel to the remote node(s). The remote nodes then execute the code as described any number of times until the control channel is established with each of the nodes in the communication network, and each of the nodes includes the code. The nodes then execute the code to establish a data channel for the nodes. With the control channel and the data channel established, an originating process may then transmit a data stream over the data channel to one or more of the nodes.
FIG. 2

200

START

EXECUTE CODE THAT ESTABLISHES A CONTROL CHANNEL WITH A REMOTE NODE, GENERATES A REPLICATION OF ITSELF, AND TRANSMITS THE REPLICATION OF THE CODE TO THE REMOTE NODE.

202

CONTROL CHANNEL AND CODE FOR EACH NODE?

NO

YES

EXECUTE THE CODE TO ESTABLISH A DATA CHANNEL WITH EACH NODE

204

TRANSMIT A DATA STREAM OVER THE DATA CHANNEL

206

TRANSMIT CONTROL INFORMATION OVER THE CONTROL CHANNEL

208

TRANSMIT STATUS INFORMATION OVER THE CONTROL CHANNEL

210

END
FIG. 9

START

EXECUTE THE CODE TO ESTABLISH A CONTROL CHANNEL WITH EACH NODE

EXECUTE THE CODE TO ESTABLISH A DATA CHANNEL WITH EACH NODE

TRANSMIT A DATA STREAM OVER THE DATA CHANNEL

TRANSMIT CONTROL INFORMATION OVER THE CONTROL CHANNEL

TRANSMIT STATUS INFORMATION OVER THE CONTROL CHANNEL

END
DATA TRANSFER TO NODES OF A
COMMUNICATION NETWORK USING A DATA
CHANNEL AND A CONTROL CHANNEL

RELATED APPLICATIONS

[0001] This patent application is a continuation-in-part of the co-pending patent application having the Ser. No. 10/786,911, filed on Feb. 25, 2004, and entitled “Data Transfer to Nodes of a Communication Network Using Self-Replicating Code”, which is incorporated by reference herein.

BACKGROUND OF THE INVENTION

[0002] 1. Field of the Invention

[0003] The invention is related to the field of communications, and in particular, to transferring data to nodes of a communication network using a data channel and a control channel.

[0004] 2. Statement of the Problem

[0005] Businesses, companies, universities, etc., commonly have a network that interconnects hundreds or thousands of devices. The network may be a Local Area Network (LAN), a Metropolitan Area Network (MAN), a Wide Area Network (WAN) or another network. Network personnel may want to transfer data, such as information, files, software, instructions, etc., to multiple devices on the network. On large networks, it may be inefficient for network personnel to transfer the data individually to each device. If the network personnel tried to transfer the data simultaneously to multiple devices, the network bandwidth may be quickly used up, particularly if the data size is large. Thus, network personnel are continually searching for ways to transfer data to multiple devices on a network to increase efficiency without sacrificing the bandwidth of the network.

[0006] One way to transfer data to multiple devices on the network is by multicasting the data over the network. The term “multicast” may be used to describe broadcasting messages or packets to a selected group of devices situated on the network. Typical applications that may benefit from such a system may include video and music broadcasting. Network personnel may use multicasting to transfer data to multiple devices on a LAN. This may solve the problem of handling each device individually. For instance, if network personnel want to install new software on multiple devices connected to the LAN, the network personnel may multicast the software over the LAN to the devices.

[0007] One problem with multicasting is that the receiving devices need the appropriate software to receive the multicast. If the receiving devices do not have the appropriate software, then the appropriate software needs to be installed on the receiving devices prior to receiving the data. Also, it is difficult to transmit data securely with multicasting.

[0008] Another problem with multicasting is that the sender may not know whether or not the receiving devices received the software. User Datagram Protocol (UDP) is one protocol used for multicasting, and UDP does not guarantee delivery of a message or require an acknowledgment from the receiving device. Thus, the sender is unaware of the transfer status of data. Network personnel may want some type of acknowledgment to ensure that the transfer of the software was successful.

[0009] Another concern of network personnel may be viruses. One of the more famous viruses is the Internet worm. Because of the Internet worm, the term “worm” is commonly received in a negative sense as being software that replicates itself repeatedly on a network to “infest” the network (i.e., a virus). However, a worm, as understood by many skilled in the art, is simply code that is self-replicating. Worms do not have to be programmed to propagate uncontrollably or to do harm to a network.

[0010] A traditional worm may replicate itself on a series of devices. A worm (parent worm) on a first device may replicate itself (child worm) on another remote device, then on another remote device, etc. After the parent worm replicates itself on another device to generate the child worm, the parent worm generally does not maintain a communication channel with the child worm. Thus, the parent worm and the child worm are not able to communicate with one another to transfer data. It may be desirable to use worms, or other self-replicating code or software, in a positive manner to transfer data through a network.

SUMMARY OF THE SOLUTION

[0011] The invention solves the above and other problems by using self-replicating code in a communication network to transfer data to nodes of the communication network. The nodes of the communication network do not initially have the code required for the data transfer. The self-replicating code of the invention replicates itself on nodes of the communication network while establishing a control channel and a data channel for the nodes. An originating process of the code can then transfer control information over the control channel to the nodes, receive status information over the control channel from the nodes, and transfer a data stream over the data channel to the nodes.

[0012] Advantageously, network personnel may use the self-replicating code to establish the data channel and to transfer data efficiently over the data channel to multiple nodes on a communication network without sacrificing the bandwidth of the network. Network personnel do not need to pre-install code on the nodes of the communication network to facilitate the data transfer. Network personnel may also monitor the status of a data transfer in real time as the nodes transmit status information over the control channel to the originating process.

[0013] One embodiment of the invention includes a method of transferring a data stream to nodes of a communication network. For the method, one of the nodes executes code to establish a control channel with one or more remote nodes. The code then replicates itself on the node, and transmits the replication of the code over the control channel to the remote node(s). The remote nodes then execute the code as described any number of times until the control channel is established with each of the nodes, and each of the nodes includes the code.

[0014] The nodes then execute the code to establish a data channel with each of the nodes. The data channel is independent of the control channel. The data channel may have a topology that is different than the topology of the control channel. For instance, the data channel may have a linear chain topology, while the control channel has a binary tree topology.
With the control channel and the data channel established, an originating process may then transmit a data stream over the data channel to one or more of the nodes. The originating process may also transmit control information over the control channel to one or more of the nodes. The originating process may also receive status information over the control channel from one or more of the nodes.

In other embodiments, the code is already installed on the nodes of the communication network. As described above, the nodes execute the code to establish a control channel with each of the nodes. The nodes then execute the code to establish a data channel with each of the nodes. The data channel is independent of the control channel.

The invention may include other exemplary embodiments described below.

DESCRIPTION OF THE DRAWINGS

The same reference number represents the same element on all drawings.

FIG. 1 illustrates a communication network in an exemplary embodiment of the invention.

FIG. 2 is a flow chart illustrating a method of transferring data to nodes of a communication network in an exemplary embodiment of the invention.

FIGS. 3A-3D illustrate how the method of FIG. 2 may work in operation in an exemplary embodiment of the invention.

FIG. 4 illustrates another communication network in an exemplary embodiment of the invention.

FIG. 5 illustrates the processes of a swormlet in an exemplary embodiment of the invention.

FIG. 6 illustrates how swormlets spawn on nodes of the communication network of FIG. 4 in an exemplary embodiment of the invention.

FIG. 7 illustrates a topology for a control channel and a topology for a data channel in an exemplary embodiment of the invention.

FIG. 8 illustrates an example of a physical topology for a Local Area Network (LAN) in an exemplary embodiment of the invention.

FIG. 9 is a flow chart illustrating another method of transferring data to nodes of a communication network in an exemplary embodiment of the invention.

DETAILED DESCRIPTION OF THE INVENTION

FIGS. 1, 2, 3A-3D, 4-9 and the following description depict specific exemplary embodiments of the invention to teach those skilled in the art how to make and use the best mode of the invention. For the purpose of teaching inventive principles, some conventional aspects of the invention have been simplified or omitted. Those skilled in the art will appreciate variations from these embodiments that fall within the scope of the invention. Those skilled in the art will appreciate that the features described below can be combined in various ways to form multiple variations of the invention. As a result, the invention is not limited to the specific embodiments described below, but only by the claims and their equivalents.

FIG. 1 illustrates a communication network 100 in an exemplary embodiment of the invention. Communication network 100 includes a plurality of nodes 101-104 capable of communicating with one another over a network facility 110. Network facility 110 may comprise a LAN, a MAN, a WAN, or another type of network. A node may comprise any system, device, component, card, etc., for communication network 100, such as a computer. Communication network 100 may include other nodes, components, devices, or systems not shown in FIG. 1.

To efficiently transfer data, such as a large data stream, to the nodes 101-104 of communication network 100, self-replicating code is used to establish a control channel for the nodes 101-104, to spawn itself on the nodes 101-104, and to establish a data channel for the nodes 101-104. The control channel and the data channel are independent channels. The control channel is used for transferring control information, status information, etc. The data channel is used for transferring data streams.

Assume that node 101 includes an originating process that spawns the self-replicating code. Also assume that the other nodes 102-104 do not have the code. According to the invention, the code replicates itself onto successive nodes 102-104 in communication network 100 to establish both the control channel and the data channel for the nodes 101-104, as is described in the following method.

FIG. 2 is a flow chart illustrating a method 200 of transferring data to nodes 101-104 of communication network 100 in an exemplary embodiment of the invention. Once again, assume node 101 is the only node having the code. For method 200, node 101 executes the code to establish a control channel with one or more remote nodes (i.e., one or more of nodes 102-104). Code comprises any instructions executable by a processor, such as machine language instructions, programming language instructions (compiled or non-compiled), interpretive language instructions, etc. The code is stored on storage media. The code then replicates itself on node 101, and transmits the replication of the code over the control channel to the remote node(s) (i.e., one or more of nodes 102-104). When the code "replicates" itself, the replication of the code represents a copy or replica of the code, a substantially similar copy or replica of the code, or a subset or derivative of the code. The remote nodes then execute the code as described in step 202 any number of times until the control channel is established with each of the nodes 101-104, and each of the nodes 101-104 includes the code.

FIGS. 3A-3C illustrate how step 202 of method 200 may work in operation in one embodiment of the invention. FIG. 3A illustrates nodes 101-103 of communication network 100. Node 101 includes code 301 and nodes 102-103 do not initially include code resembling or corresponding to code 301. According to step 202 of method 200, node 101 executes code 301 to establish a control channel 311 with remote nodes 102 and 103 (see FIG. 3B). Node 101 also executes code 301 to replicate itself, and transmit the replication of the code 301 to nodes 102 and 103 over the control channel 311. Node 101 may perform this operation for multiple nodes not shown in FIG. 3B.
After receiving the code, nodes 102 and 103 may then execute the code 301 as described in step 202 to establish a control channel with remote nodes. Node 103 does not have a remote node (or child node) in this embodiment, so node 103 does not perform step 202. Node 102 does have a remote node in node 104. Therefore, according to step 202, node 102 executes code 301 to establish a control channel 311 with remote node 104 (see FIG. 3C). Node 102 also executes code 301 to replicate itself, and transmit the replication of the code 301 to remote node 104 over the control channel 311. Node 102 may perform this operation for multiple nodes not shown in FIG. 3C.

Referring back to FIG. 2, step 202 is performed on the nodes 101-104 until each of the nodes 101-104 includes the code and each of the nodes 101-104 is connected to the control channel. Method 200 then proceeds to step 204. In step 204, the nodes 101-104 execute the code to establish a data channel with each of the nodes 101-104. The data channel is independent of the control channel. Nodes 101-104 may receive control information over the control channel to assist in establishing the data channel.

FIG. 3D illustrates how step 204 of method 200 may work in operation in one embodiment of the invention. Nodes 101-104 execute the code to establish the data channel 312. The data channel is independent of the control channel, and may take on a different topology than the control channel. As shown in FIG. 3D, node 101 establishes a data channel 312 with node 102. Node 102 establishes the data channel 312 with node 103. Node 103 establishes the data channel 312 with node 104. The topology of the data channel 312 is a linear chain while the topology of the control channel 311 is a binary tree. The topology of the control channel 311 and the data channel 312 is controlled by the originating process to maximize the efficiency of both channels.

Referring back to FIG. 2, method 200 proceeds to step 206 after the control channel and the data channel are established. In step 206, the originating process executing in node 101 transmits a data stream over the data channel to one or more of the nodes 102-104. A data stream may comprise any information, software, instructions, etc., targeted for transfer in the communication network 100. Responsive to receiving the data stream, the nodes 102-104 may execute the code to handle the data stream. For instance, the nodes 102-104 may execute the code to perform one or more local processes on the data stream, such as storing the data stream.

In step 208, the originating process may also transmit control information over the control channel to one or more of the nodes 102-104. Control information comprises any operations, administration, or management information used by the nodes in the communication network 100. In step 210, one or more of the nodes may transmit status information over the control channel to the originating process in node 101.

Advantageously, network personnel may use the self-replicating code as described in this embodiment to efficiently transfer a data stream to multiple nodes 101-104 on communication network 100, without sacrificing the bandwidth of the network 100. Also, network personnel do not need to pre-install code on nodes 102-104 of communication network 100 to facilitate the data transfer, as the self-replicating code can spawn itself on the nodes 102-104 of the network 100 to facilitate the data transfer.

Once the code is spawned on each node 101-104, the code can be used to maintain the same data channel and control channel. Alternatively, the original data channel and control channel may be torn down after the data stream is transmitted to the appropriate nodes. To subsequently transfer another data stream, the code, which is already resident on each node, may be executed on the nodes to establish a new control channel and a new data channel, as is illustrated in FIG. 9.

FIG. 9 is a flow chart illustrating another method 900 of transferring data to nodes 101-104 of communication network 100 in an exemplary embodiment of the invention. Assume each of the nodes 101-104 includes the code needed to establish a control channel and a data channel. The code may be the result of self-replicating code that previously spawned itself on nodes 101-104 (as according to FIG. 2). The code may also have been installed in another manner. For method 900, each node 101-104 executes the code to establish a control channel with each of the nodes 101-104 in step 902. In step 904, the nodes 101-104 execute the code to establish a data channel with each of the nodes 101-104. Nodes 101-104 may receive control information over the control channel to assist in establishing the data channel. Once again, the new control channel and the new data channel are independent channels. The new control channel and new data channel may be the same as before, or may be different depending on which data stream needs to be transmitted to which nodes. The topology of both the control channel and the data channel once again depends on the originating process.

Method 200 proceeds to step 906 after the control channel and the data channel are established. In step 906, the originating process executing in node 101 transmits a data stream over the data channel to one or more of the nodes 102-104. Responsive to receiving the data stream, the nodes 102-104 execute the code to handle the data stream. For instance, the nodes 102-104 may execute the code to perform one or more local processes on the data stream, such as storing the data stream. In step 908, the originating process may also transmit control information over the control channel to one or more of the nodes 102-104. In step 910, one or more of the nodes 102-104 may transmit status information over the control channel to the originating process in node 101.

FIG. 4 illustrates another communication network 400 in an exemplary embodiment of the invention. Communication network 400 includes a plurality of nodes 401-432 capable of communicating with one another over a LAN 440. LAN 440 comprises any kind of trusted network where access to the nodes 401-432 from the LAN 440 is not blocked by firewalls or other protection mechanisms. In other embodiments, LAN 440 may comprise an intranet, a MAN, a WAN, or another type of enterprise network. A node may comprise any system, device, component, card, etc. for communication network 400, such as a computer. Nodes 401-432 are remote from each other, meaning that they can communicate with one another over LAN 440. Communication network 400 may include other nodes, components, devices, or systems not shown in FIG. 4.

Assume that node 401 is a controller node operated by network personnel. Further assume that network person-
nel want to transfer a large data stream to all nodes 402-432 in communication network 400. For instance, the network personnel may need to install new software on nodes 402-432, where the software comprises a large data stream. It would be inefficient for the network personnel to install the software individually on each machine. If the network personnel tried to simultaneously transfer the software as a large data stream to multiple nodes 402-432, then the bandwidth of LAN 440 may be compromised. The network personnel may use self-replicating code according to the invention to efficiently transfer the software to nodes 402-432 without compromising the bandwidth of LAN 440.

[0045] In this embodiment, self-replicating code may be referred to as a streaming worm. A streaming worm (swarmlet) comprises any software, code, or instructions that can replicate itself, as well as a chunk of arbitrary payload code, on a set of nodes in a communication network, while streaming data, control information, and/or status information between the nodes. A hybrid swarm is described in this embodiment that streams data on a data channel while transmitting the control and status information over a independent control channel.

[0046] Each hybrid swarm is composed of one or more swarmlets. Each swarmlet spans at least two nodes and includes a plurality of processes. FIG. 5 illustrates a swarmlet 500 in an exemplary embodiment of the invention. A single swarmlet is shown in FIG. 5 to illustrate the processes in a swarmlet. In actual operation, swarmlet 500 has one or more parent swarmlets (not shown) and possibly one or more child swarmlets (not shown).

[0047] The swarmlet 500 spans one or more parent nodes 502 and 504 and a child node 506 in this embodiment. One of the parent nodes is a control parent node 502 that establishes a control channel (illustrated with solid arrows) with the child node 506. The other parent node is a data parent node 504 that establishes a data channel (illustrated with dotted arrows) with the child node 506. The control parent node 502 and the data parent node 504 are shown as separate nodes in this embodiment, but the two could be the same node.

[0048] The swarmlet 500 includes a transceiver process 510, a control process 511, a status process 512, a payload process 513, a data routing process 514, a data transmitter process 515, and a data receiver process 516. The transceiver process 510 for the swarmlet 500 is executed on the control parent node 502. The data transmitter process 515 is executed on the data parent node 504. The remaining processes are executed on the child node 506.

[0049] The transceiver process 510 executes on the control parent node 502 and establishes a bi-directional control channel with the control process 511 and/or the status process 512 on the child node 506. The transceiver process 510 acts as a conduit for payload code and other program code, control information, status information, and return data from the child node 506. The transceiver process 510 may transmit status information to a status process of a parent swarmlet (not shown). The transceiver process 510 may also receive control information from a control process of the parent swarmlet.

[0050] The control process 511 executes on the child node 506 and is responsible for distributing the swarmlet code and payload code over the control channel. The control process 511 then initiates the status process 512, the payload process 513, the data routing process 514, and optionally one or more transceiver processes and the data transmitter processes for additional child swarmlets (not shown), if necessary. The control process 511 then locally processes and/or routes any control operations that it receives to its child swarmlets, and reports whenever any of its child processes dies. Although it normally sends status information to the local status process 512, it maintains a backup status connection to the transceiver process 510 in case the status process 512 dies. Control operations may be addressed to one or more individual swarmlets, or broadcast to all swarmlets.

[0051] The status process 512 executes on the child node 506 and is responsible for packetizing return data received from the local payload process 513, and multiplexing it with status packets from other processes in the local swarmlet, as well as return data and status packets from its child swarmlets. The status process 512 then transmits this multiplexed status and return data stream to the transceiver process 510 of the control parent node 502.

[0052] The payload process 513 executes on the child node 506 and comprises the code to be locally performed on a node. The payload process 513 can do anything allowed by applicable security protocols. The payload process 513 reads data sent by the originating process through the data routing process 514, and sends data and status information back to the originating process through the status process 512.

[0053] The data transmitter process 515 executes on the data parent node 504 (which may be the same or different than the control parent node 502) and is responsible for transmitting the data it receives from a data routing process of a parent swarmlet to data receiver process 516 of the child node 506.

[0054] The data receiver process 516 executes on the child node 506 and is responsible for receiving data from the data transmitter process 515 of the data parent node 504 and passing it through a First In First Out (FIFO) buffer 518 to the data routing process 514. The FIFO 518 is included because the data receiver process 516 and data routing process 514 belong to different process families.

[0055] The data routing process 514 executes on the child node 506 and is responsible for reading data from the FIFO 518, and routing the data to the payload process 513 and/or a transceiver process of any of child swarmlets as appropriate. The maximum rate at which data is sent downstream may be optionally throttled as directed by the main originator process.

[0056] Referring back to FIG. 4, assume that node 401 includes an originating process (not shown) that originates the hybrid swarm. Because network personnel want to transfer a large data stream (i.e., software) to many nodes of the network 400, a linear topology may be the most efficient topology for the data channel. A linear topology may work the best because the bandwidth of LAN 440 does not have to be split between the swarmlets of the hybrid swarm. Only one copy of the large data stream needs to be transferred over the LAN 440 at a time with the linear topology, as is illustrated below. Even though a linear topology may work best for the data channel, the linear topology may not work
For the control channel. Therefore, a different topology is used for the control channel.

FIG. 6 illustrates a hybrid swarm in an exemplary embodiment of the invention. The hybrid swarm originates in an originating program being executed in a node, such as node 401 in this embodiment. Node 401 executes the originating program, being stored on a storage media, to initiate an originating process 602. The originating process 602 in node 401 spawns swarmlet 610 (referred to in FIG. 6 as S-LET). More particularly, the originating process 602 initiates a transceiver process 611 on node 401. The transceiver process 611 executes on node 401 to establish a control channel with node 402. The transceiver process 611 then transfers a copy of the swarmlet code for swarmlet 610 over the control channel (illustrated by solid arrows) to node 402. The transceiver process 611 may transfer the copy of the swarmlet code to node 402 using a remote shell command, such as remsh, rsh, or ssh. The transceiver process 611 may use the remote shell command to transfer and remotely execute an in-line Perl bootstrap script on node 402 that pulls the swarmlet code over the control channel from node 401.

Node 402 executes the swarmlet code to perform a control process 612 for swarmlet 610. The control process 612 initiates the other processes for swarmlet 610, such as the status process 613, the payload process 614, and the data routing process 615. The control process 612 receives the payload code from the originating process 602 over the control channel. The control process 612 replicates the payload code and forwards the copy of the payload code to the payload process 614. The control process 612 forwards the payload code to another swarmlet (if necessary) as is described below.

Swarmlet 610 then spawns two child swarmlets 620 and 630. Control process 612 initiates a transceiver process 621 on node 402 for swarmlet 620. The transceiver process 621 executes on node 402 to establish a control channel with node 403. The transceiver process 621 then transfers a copy of the swarmlet code for swarmlet 620 over the control channel to node 403. Control process 612 also initiates a transceiver process 631 on node 402 for swarmlet 630. The transceiver process 631 executes on node 402 to establish a control channel with node 418. The transceiver process 631 then transfers a copy of the swarmlet code for swarmlet 630 over the control channel to node 418.

Node 403 executes the swarmlet code to perform a control process 622 for swarmlet 620. The control process 622 initiates the other processes for swarmlet 620, such as the status process 623, the payload process 624, and the data routing process 625. The control process 622 receives the payload code from the control process 612 of swarmlet 610 over the control channel through the transceiver process 621. The control process 622 replicates the payload code and forwards the copy of the payload code to nodes 404 and 411.

Node 418 also receives a copy of the swarmlet code for the data receiver process 637 over the data channel from node 417. Node 418 executes the swarmlet code to initiate the data receiver process 637 for node 418. The data receiver process 637 communicates with a data transmitter process 626 over the data channel to receive a data stream from the data transmitter process 626. The data receiver process 627 transmits the data stream to the FIFO 628. The data routing process 625 then reads the data stream from the FIFO 628 and transmits the data stream to the data transmitter process of the swarmlet of node 404.

Swarmlets are spawned in a similar manner through nodes 404-432 of the communication network 400 (see FIG. 4) as designated by the originating process 602 to establish the control channel between the nodes. FIG. 7 illustrates the topology of the control channel as created by the swarm in communication network 400. The solid arrows in FIG. 7 illustrate the control channel. The control channel has a binary tree topology in this embodiment. The control channel may take on other topologies in other embodiments.

Once the swarmlets are spawned and control channel is established between the nodes 401-432, the data channel can then be established between the nodes 401-432. The data channel may be established simultaneous with the control channel in other embodiments.
(see FIG. 4) as designated by the originating process 602. FIG. 7 also illustrates the topology of the data channel in the communication network 400. The dotted arrows in FIG. 7 illustrate the data channel. The data channel has a linear topology (as compared to the binary tree topology of the control channel) to most efficiently transfer large data files.

[0068] When in operation, data originator process 604 transfers a data stream to swormlet 610. The data stream in this embodiment comprises software that is a large data stream. The data transmitter process 616 transmits the data stream over the data channel to the data receiver process 617. The data receiver process 617 transmits the data stream to the FIFO 618. The data routing process 615 then reads the data stream from the FIFO 618. The data routing process 615 replicates the data stream, and forwards a copy of the data stream to the payload process 614. The data routing process 615 forwards the data stream to another swormlet 620 as addressed or specified by the originating process 602. The payload process 614 receives the copy of the data stream from the data routing process 615. The payload process 614 executes the payload code to locally handle the data stream. For instance, the payload code may direct the payload process 614 to store the data stream in a known directory on node 402.

[0069] The control process 612 may receive control instructions from the originating process 602. The control process 612 may execute the control instructions on the payload process 614. The control process 612 may also forward the control instructions to another swormlet (if necessary).

[0070] In executing the payload code, the payload process 614 may generate output data. The output data may comprise periodic progress information about storing the large data file. The payload process 614 forwards the output data to the status process 613. The status process 613 receives the output data from the payload process 614 and/or status information from any child swormlets. The status process 613 then transmits the output data and/or status information from the payload process 614 and child swormlets over the control channel. The status information may comprise feedback on the success of storing the large data stream on node 402.

[0071] The same process occurs in each node 402-432 to transmit the data stream to each node 402-432. Alternatively, the originating process 602 may designate which nodes are to receive the data stream, and in which order, based on a map, a set of instructions, an addressing scheme, etc. The originating process 602 may address the data stream to one or more swormlets 610, 620 or 630, or broadcast to all swormlets 610, 620, and 630.

[0072] The topology of the data channel may depend on the physical topology of LAN 440. For instance, assume that LAN 440 includes a plurality of switching hubs connected to different nodes. FIG. 8 illustrates a physical topology of LAN 440 in an exemplary embodiment of the invention. In FIG. 8, node 401 is connected to switching hub 801, nodes 402-405 are connected to switching hub 802, and nodes 406-409 are connected to switching hub 804. In defining the topology of the data channel, the originating process 602 ensures that the data stream is not transmitted over each segment of LAN 440 more than once. For instance, the data channel is defined as a linear chain (see dotted arrows) through nodes 402-405 of switching hub 802 and then through nodes 406-409 of switching hub 804, etc.

[0073] Once the code is spawned on each node 401-432, the code can be used to maintain the same data channel and control channel. Alternatively, the original data channel and control channel may be torn down after the data stream is transmitted to the appropriate nodes. To subsequently transfer another data stream, the code, which is already resident on each node, may be executed on the nodes to establish a new control channel and a new data channel. Once again, the new control channel and the new data channel are independent channels. The new control channel and new data channel may be the same as before, or may be different depending on which data stream needs to be transmitted to which nodes. When the new control channel and new data channel are established, the channels can advantageously be used as described above to transmit the data stream.

[0074] Transferring the data stream to the nodes 402-432 of the communication network 400 in FIG. 4 according to the process described above provides many advantages. First, the nodes 402-432 do not need to initially have any special code to receive the data stream. The swormlets will replicate themselves on the nodes 402-432 to provide the needed code. The nodes may then execute the code in parallel to more efficiently transfer the data stream over the data channel. Second, network personnel can get feedback on the status of the transfer. Each swormlet maintains the control channel and provides status information back to the originating process 602. Thus, network personnel can view the status information to determine the status of the data stream transfer. Third, the bandwidth of the LAN 440 will most likely not be compromised by the data stream transfer. Only one copy of the data stream should be present on the LAN 440 at any one time in a linear topology, as the nodes pass the one copy of the data stream, block by block, from node to node until the final node is reached.

I claim:

1. A method of transferring a data stream to a plurality of nodes of a communication network, the method comprising:
   - executing code from an originating process that establishes a control channel with a remote node, generates a replication of itself, and transmits the replication over the control channel to the remote node, wherein the code is executed on one or more of the plurality of nodes until the control channel is established with each of the plurality of nodes and each of the plurality of nodes includes the code;
   - executing the code on the plurality of nodes to establish a data channel with each of the plurality of nodes; and
   - transmitting a data stream over the data channel from the originating process to at least one of the plurality of nodes.

2. The method of claim 1 further comprising:
   - transmitting control information over the control channel from the originating process to at least one of the plurality of nodes.

3. The method of claim 2 further comprising:
   - transmitting status information over the control channel from at least one of the plurality of nodes to the originating process.
4. The method of claim 1 wherein the data channel is independent from the control channel.

5. The method of claim 1 wherein a topology of the control channel is different than a topology of the data channel.

6. The method of claim 5 wherein the topology of the control channel comprises a binary tree, and the topology of the data channel comprises a linear chain.

7. The method of claim 1 wherein the code comprises a streaming worm.

8. The method of claim 1 further comprising:
   - tearing-down the control channel and the data channel;
   - executing the code to establish a new control channel; and
   - executing the code to establish a new data channel.

9. A software product for transferring a data stream to a plurality of nodes of a communication network, the software product comprising:
   - storage media that stores an originating program; and
   - the originating program executable by processing systems to spawn code that:
     - establishes a control channel with a remote node, generates a replication of itself, and transmits the replication over the control channel to the remote node, until the control channel is established with each of the plurality of nodes and each of the plurality of nodes includes the code; and
     - establishes a data channel with each of the plurality of nodes to facilitate the transmission of a data stream over the data channel from the originating process to at least one of the plurality of nodes.

10. The software product of claim 9 wherein the originating program executes an originating process, and the code transmits control information over the control channel from the originating process to at least one of the plurality of nodes.

11. The software product of claim 10 wherein the code transmits status information over the control channel from at least one of the plurality of nodes to the originating process.

12. The software product of claim 9 wherein the data channel is independent from the control channel.

13. The software product of claim 9 wherein a topology of the control channel is different than a topology of the data channel.

14. The software product of claim 13 wherein the topology of the control channel comprises a binary tree, and the topology of the data channel comprises a linear chain.

15. The software product of claim 9 wherein the code comprises a streaming worm.

16. A method of transferring a data stream to a plurality of nodes of a communication network, the method comprising:
   - executing code on the plurality of nodes to establish a control channel with each of the plurality of nodes;
   - executing the code on the plurality of nodes to establish a data channel with each of the plurality of nodes; and
   - transmitting a data stream over the data channel from an originating process operating in one of the nodes to at least one of the other nodes;

   wherein the data channel is independent from the control channel.

17. The method of claim 16 further comprising:
   - transmitting control information over the control channel from the originating process to at least one of the other nodes.

18. The method of claim 17 further comprising:
   - transmitting status information over the control channel from at least one of the other nodes to the originating process.

19. The method of claim 16 wherein a topology of the control channel is different than a topology of the data channel.

20. The method of claim 19 wherein the topology of the control channel comprises a binary tree, and the topology of the data channel comprises a linear chain.

* * * * *