BOEING

INVENTION DISCLOSURE

This form is to be used for disclosure to The Boeing Company of inventions, discoveries, improvements or innovations, whether or not considered patentable. See above for instructions.

A — TITLE OF INVENTION (Descriptive and Concise)

SWAN: Small-world Wide Area Networking ......

B — INVENTOR INFORMATION (Use Additional Sheet If Necessary)

	1	2	3	4
INVENTOR NAME (FIRST, M.I., LAST)	Virgil E. Bourassa	Fred B. Holt		
SOCIAL SECURITY NO.	[redacted]	[redacted]		
ORG. NO.	[redacted]	[redacted]		
MAIL STOP	[redacted]	[redacted]		
PHONE	[redacted]	[redacted]		
BOEING EMPLOYEE (ADD SUBSIDIARY)	☑ Boeing SSG ☐ MDC ☐ BNA	☑ Boeing SSG ☐ MDC ☐ BNA	☐ Boeing ☐ MDC ☐ BNA	☐ Boeing ☐ MDC ☐ BNA
CONTRACT EMPLOYEE	☐	☐	☐	☐
OTHER (SPECIFY)	☐	☐	☐	☐
MANAGER'S NAME	Robert Abarbanel	[redacted]		
PHONE	[redacted]	[redacted]		

C — STATE OF DEVELOPMENT (See Remarks On Back)

DATE CONCEIVED		DATE BUILT	DATE SATISFACTORILY TESTED	
11/96	☐ CONCEPT ONLY ☐ PROVEN ANALYTICALLY ☑ DESIGN COMPLETE	8/30/99	9/16/99	☐ PROTOTYPE ☑ IN PRODUCTION 11/11/99 DATE

D — APPLICATION OF THE INVENTION

PRODUCT/PROGRAM	PRODUCTION RELEASE E.G. PRR NO.	DATE
[redacted]		9/99

POTENTIAL CUSTOMER(S) IN ADDITION TO BOEING: [redacted]

E — DISCLOSURE OF INVENTION OUTSIDE BOEING

DISCLOSED TO:	NAME(S)	DATE(S)
☑ VENDOR ☐ CUSTOMER ☐ OTHER	[redacted]	4/8/99, 12/8/99, 12/9/99

PUBLISHED	PUBLICATION NAME	DATE	VOLUME NO.	PAGE
☐ YES ☑ NO				

F — DEVELOPMENT HISTORY

1. WHAT BOEING ACCOUNT OR WORK ORDER WERE YOU CHARGING TO WHEN YOU MADE THIS INVENTION?
   ACCOUNT OR WORK ORDER NO. FOR EACH INVENTOR (16-DIGIT CHARGELINE) 1) [redacted]
   2) [redacted] 3) ________ 4) ________

2. CHECK AS APPLICABLE:
   - ☐ THIS INVENTION WAS CONCEIVED OR FIRST BUILT AND TESTED IN THE COURSE OF WORK UNDER A U.S. GOVERNMENT CONTRACT.
     CONTRACT NO. OR OTHER IDENTIFICATION ________
   - ☑ THIS INVENTION WAS NEITHER CONCEIVED NOR FIRST BUILT AND TESTED IN THE COURSE OF WORK UNDER A U.S. GOVERNMENT CONTRACT.
   - ☐ THE FOLLOWING ADDITIONAL PARTIES MAY HAVE RIGHTS TO THIS INVENTION: N/A

3. RELATED INVENTION DISCLOSURE NOS: ________

DO NOT WRITE BELOW THIS LINE

DISCLOSURE NO.	DATE RECEIVED	DISCLOSURE ASSIGNED TO:	PE	IP

[redacted]

SWAN: SMALL–WORLD WIDE AREA NETWORKING

Introduction

The need has been increasing in large software projects, in collaborative workflow, and to facilitate enterprise–wide–engineering, for an effective means to allow scalable and reliable sharing of information across multiple processes. For example, [redacted] has proven valuable in allowing collaborative design reviews to take place at geographically distant sites, such as between Everett, WA, and St. Louis, MO. To enhance the value of [redacted] and enable additional world–wide electronic collaboration applications, programmers need a software mechanism allowing dozens, hundreds, or perhaps thousands of participating computer processes to simultaneously share information easily, quickly, and reliably across the world.

Problem Solved By This Invention

SWAN provides general world–wide ("wide–area") peer–to–peer communications among computer processes. It achieves this with high reliability and low latency, scaling from a single process to thousands of participating processes. The system is completely distributed among the participants, which may join, depart, or even fail, at any time and in any order.

The implementation doesn't require special hardware, or the intervention of system administrators. All computers can participate, without requiring root access, daemons, kernel modifications, or the addition of "well–known" port numbers.

Though openly accessible, SWAN does have rudimentary security. Joining a session is restricted to those processes sharing the SWAN code base and aware of the correct channel designation.

Background

There are four categories of computer network communication systems that might be applied to the problem of wide–area simultaneous sharing of information for the purpose of collaborative processing. These are: 1) point–to–point networking protocols, 2) client–server middleware, 3) multicast networking protocols, and 4) peer–to–peer middleware.

Point–To–Point Networking Protocols

A number of point–to–point networking protocols exist to allow direct one– or two–way communication between two computer processes. Examples include UNIX pipes, TCP/IP, UDP, IBM's SNA, and Xerox' XNS. Of these, only TCP/IP and UDP are universally available for communication between computers connected via the Internet or on the Boeing Intranet.

Using point–to–point connections directly does not scale easily as the number of participating processes grows. A process is limited in the number of such connections that can be made (roughly 60), and managing even a single connection is a complex task for programmers. Coordinating a communication session involving even a modest number of connections exacerbates the program complexity enormously. For all of these reasons, direct use of a point–

THE FOREGOING WAS EXPLAINED TO AND UNDERSTOOD BY

WITNESS SIGNATURE PRINT Robert M Abarbanel PHONE [redacted] DATE 12/23/99 ORGN # M/S [redacted]

WITNESS SIGNATURE PRINT [redacted] PHONE [redacted] DATE 12/23/99 ORGN # M/S [redacted]

INVENTOR SIGNATURE DATE 12/23/99 INVENTOR SIGNATURE DATE 23 Dec 99

DISCLOSURE NO. (ASSIGNED BY PATENT STAFF) DATE

to-point networking protocol is not a feasible mechanism for sharing information across a medium- to large-scale collaboration across a wide-area network.

Client-Server Middleware

To alleviate the complexity of programming directly at the network protocol level, client-server middleware is available to provide an easier programming abstraction. In client-server middleware, a number of "client" processes find or instantiate a single "server" process, forming a direct network connection between them. The client may then request services from the server, which often is given central authority over a resource, such as a database. Examples include database servers, remote procedure calls (RPC), and CORBA.

The client-server paradigm provided by this middleware, while providing a mechanism for sequenced resource sharing, is not feasible for collaborative information sharing. One client may be able to convey information directly to the server, but the other clients are unaware that the server has new information, forcing them to poll the server for possible new information. This creates a performance bottleneck as the number of participants increases, adds undue latency in disseminating the information, and wastes processing time as client processes continue to check for new information.

Some client-server middleware packages, such as CORBA, allow clients to register "callbacks," functions to be invoked when an event occurs. While this facility may make collaborative information sharing less onerous, for medium- to large-scale applications the single server is still a performance bottleneck.

Furthermore, the reliability of a collaborative application relying on a single server is poor, as loss of the server or difficulty in its instantiation completely destroys the integrity of the collaborative session.

Multicast Networking Protocols

Multicast networking protocols allow selective broadcast of messages to multiple recipients. It retains the complexity of direct network communication mentioned above, but is a natural choice for collaborative sharing. Currently, multicast is available for UDP messages, but virtually all UDP multicast traffic is limited to a single local-area network or, at most, a small set of connected local-area networks. UDP multicast, in its current implementation, could easily swamp the Internet otherwise, as it would have to saturate the Internet with each message to find all possible participants.

Several wide-area multicast networking protocols have been proposed, and some, such as IP Multicast, are in limited commercial and/or research deployment. These solutions require special router hardware and/or software to achieve data sharing without overwhelming the participating networks. Even if a standard solution were selected today, it would take years, or possibly decades, before the entire Internet infrastructure could be completely retrofitted with the new technology.

Additionally, the solutions proposed in this area, in an attempt to conserve bandwidth, are not constructed with reliability as a concern. By using minimum spanning trees among the routers

THE FOREGOING WAS EXPLAINED TO AND UNDERSTOOD BY

WITNESS SIGNATURE [signature] PRINT Robert M. Abarbanel PHONE [redacted] DATE 12/23/99 ORGN # M/S [redacted]

WITNESS SIGNATURE PRINT [redacted] PHONE [redacted] DATE 12/23/99 ORGN # M/S [redacted]

INVENTOR SIGNATURE [signature] DATE 12/23/99 INVENTOR SIGNATURE [signature] DATE 23 Dec 99

DISCLOSURE NO. (ASSIGNED BY PATENT STAFF) DATE

involved, any router failure can partition the collaborative session.

Peer–To–Peer Middleware

Peer–to–peer middleware provides the programmer with a software library that is intended to provide an easy–to–use abstraction, such as "publish–and–subscribe" or "shared objects," for immediately sharing information among a set of collaborating processes. Hidden from the programmer is how the actual communication takes place.

The underlying communication infrastructure may make use of a multicast network protocol, or a graph of point–to–point network protocols, or a combination of the two. The infrastructure in commercial use today, in products such as IBM's Sametime, Data Connection's DC–Share, and Microsoft's NetMeeting, is the T.120 Internet standard. That used in the current TeleFly infrastructure is called the RPC Herald. Both have the user (*not* the programmer), assemble a point–to–point graph of connections. For this reason, and others, neither is suitable for the needs of medium– to large–scale collaboration.

T.120 Internet Standard

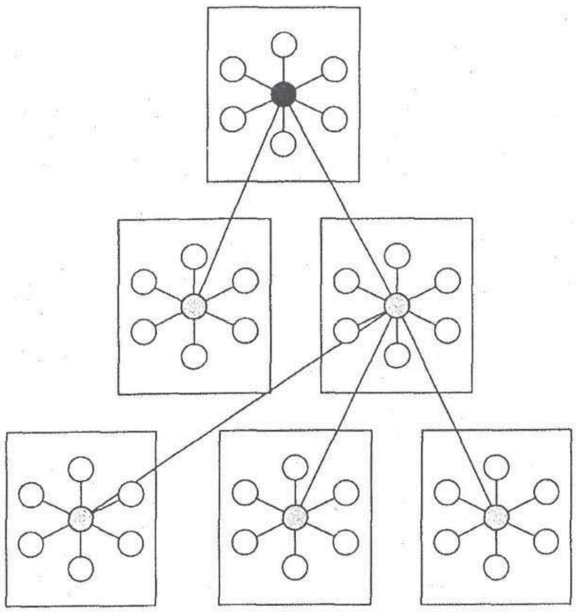

Figure 1. T.120 connection tree.

An example of a T.120 communication session is depicted in Figure 1. When first connecting to a session on a given host computer, a proxy process (depicted in gray and black in the figure), called an MCU, is instantiated by a daemon process (a resident process that listens for such requests, not depicted). This MCU forms a direct connection to the MCU of another host designated by the application user, or is designated as the *root* of the session (the black dot). The requesting process

THE FOREGOING WAS EXPLAINED TO AND UNDERSTOOD BY

WITNESS SIGNATURE PRINT PHONE DATE ORGN # M/S

Robert M. Aberbanel [redacted] 12/23/99 [redacted]

WITNESS SIGNATURE PRINT [redacted] PHONE [redacted] DATE 12/31/99 ORGN # M/S [redacted]

INVENTOR SIGNATURE DATE 12/23/99 INVENTOR SIGNATURE DATE 23 Dec 99

DISCLOSURE NO. (ASSIGNED BY PATENT STAFF) DATE

and all additional processes on the host wishing to join the session form a direct connection to the MCU process on that host. To share information, a process sends a message to its MCU, which is sent up the tree of MCUs to the root, then down the tree of MCUs and disseminated among their attached processes.

This scheme fails to solve the problem of medium– to large–scale collaboration for a number of reasons. First, the responsibility of determining the topology of the connection graph is foisted off on the application users, which, in addition to being a nuisance to the users, is not likely to result in an efficient structure for performance. The most common kind of connection scheme seen in practice is for all host MCUs to connect directly to the root MCU.

Second, the MCUs are performance, reliability, and scalability bottlenecks. All messages must be serialized through each MCU to a potentially large number of processes on the host. Loss of an MCU not only removes all of the processes on the host, but also prunes the subtree attached to it from the session. Furthermore, given operating system limitations, each MCU can accommodate, at most, about 60 client processes.

Third, the need to coordinate all messages through the root MCU not only makes that process a performance bottleneck, and a single point of failure for the session, but also causes the speed of communication to be limited by the slowest host and/or communication link in the tree. For example, NetMeeting's performance is reported to be intolerable with about 20 participants.

Finally, the T.120 daemon must be installed on each host participating in a session. This requires additional administration and maintenance, and limits the set of hosts that can join in a session. It also requires an additional "well–known" port number, which must be coordinated globally among all computers on the Internet.

[redacted]

THE FOREGOING WAS EXPLAINED TO AND UNDERSTOOD BY

WITNESS SIGNATURE PRINT Robert M. Abarbanel PHONE [redacted] DATE 12/23/99 ORGN # M/S [redacted]

WITNESS SIGNATURE PRINT [redacted] PHONE [redacted] DATE 12/23/99 ORGN # M/S [redacted]

INVENTOR SIGNATURE DATE 12/23/99 INVENTOR SIGNATURE DATE 23 Dec 99

DISCLOSURE NO. (ASSIGNED BY PATENT STAFF) DATE

DOCKET ALARM

Explore Litigation Insights

Docket Alarm provides insights to develop a more informed litigation strategy and the peace of mind of knowing you're on top of things.

Real-Time Litigation Alerts

Keep your litigation team up-to-date with **real-time alerts** and advanced team management tools built for the enterprise, all while greatly reducing PACER spend.

Our comprehensive service means we can handle Federal, State, and Administrative courts across the country.

Advanced Docket Research

With over 230 million records, Docket Alarm's cloud-native docket research platform finds what other services can't. Coverage includes Federal, State, plus PTAB, TTAB, ITC and NLRB decisions, all in one place.

Identify arguments that have been successful in the past with full text, pinpoint searching. Link to case law cited within any court document via Fastcase.

Analytics At Your Fingertips

Learn what happened the last time a particular judge, opposing counsel or company faced cases similar to yours.

Advanced out-of-the-box PTAB and TTAB analytics are always at your fingertips.

API

Docket Alarm offers a powerful API (application programming inter-face) to developers that want to integrate case filings into their apps.

LAW FIRMS

Build custom dashboards for your attorneys and clients with live data direct from the court.

Automate many repetitive legal tasks like conflict checks, document management, and marketing.

FINANCIAL INSTITUTIONS

Litigation and bankruptcy checks for companies and debtors.

E-DISCOVERY AND LEGAL VENDORS

Sync your system to PACER to automate legal marketing.

WHAT WILL YOU BUILD? | sales@docketalarm.com | 1-866-77-FASTCASE