US 20060277262A1

(19) **United States**

(12) **Patent Application Publication** (10) **Pub. No.: US 2006/0277262 A1**

Boss et al. (43) **Pub. Date: Dec. 7, 2006**

(54) **SESSION MANAGEMENT ENHANCEMENTS FOR INSTANT MESSAGING APPLICATIONS**

(76) Inventors: **Gregory Jensen Boss**, American Fork, UT (US); **Rick Allen Hamilton II**, Charlottesville, VA (US); **Aroopratan D. Pandya**, Hopewell JCT, NY (US); **James Wesley Seaman**, Falls Church, VA (US)

Correspondence Address:
IBM CORPORATION (RUS)
c/o Rudolf O Siegesmund Gordon & Rees, LLp
2100 Ross Avenue
Suite 2600
DALLAS, TX 75201 (US)

(21) Appl. No.: **11/146,989**

(22) Filed: **Jun. 6, 2005**

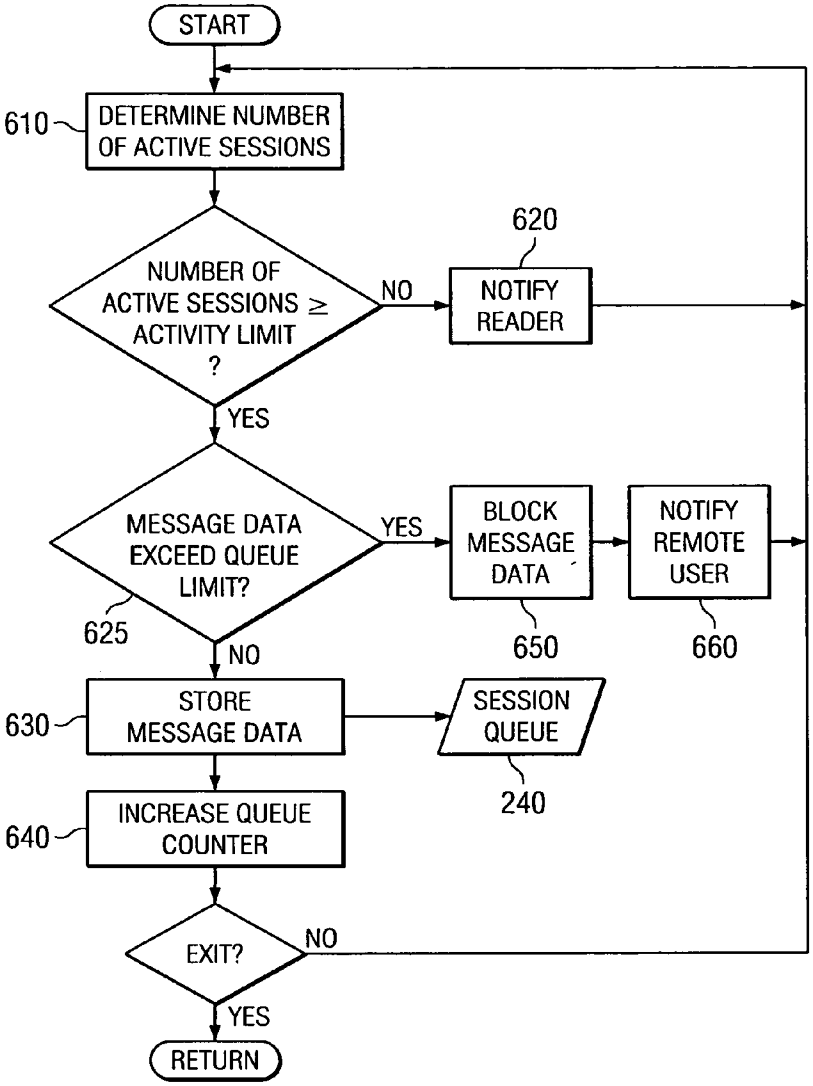
Publication Classification

(51) **Int. Cl.**
G06F 15/16 (2006.01)

(52) **U.S. Cl.** **709/206**

(57) **ABSTRACT**

The invention comprises an improved demand-based messaging system that enables a user to effectively manage multiple messaging sessions. The messaging system comprises a messaging program operable on a plurality of electrical computers or data processing machines connected by one or more communications media. The messaging program comprises a conventional message composer program, a conventional message transport program, a conventional message reader program, and an inventive, user-configurable, policy-driven session management program.



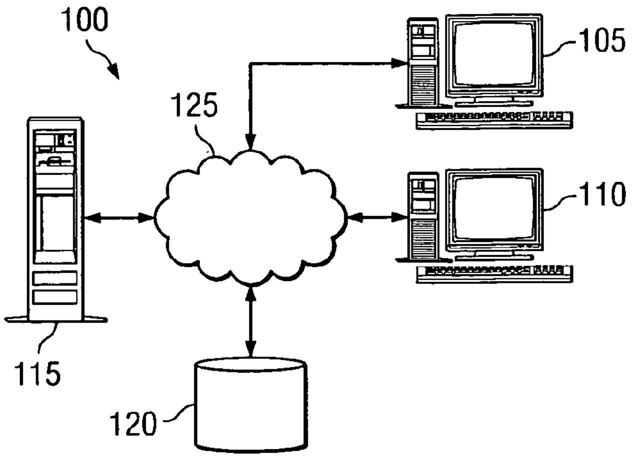


FIG. 1
(PRIOR ART)

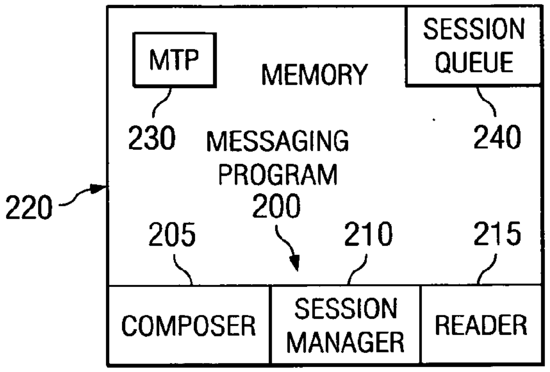


FIG. 2

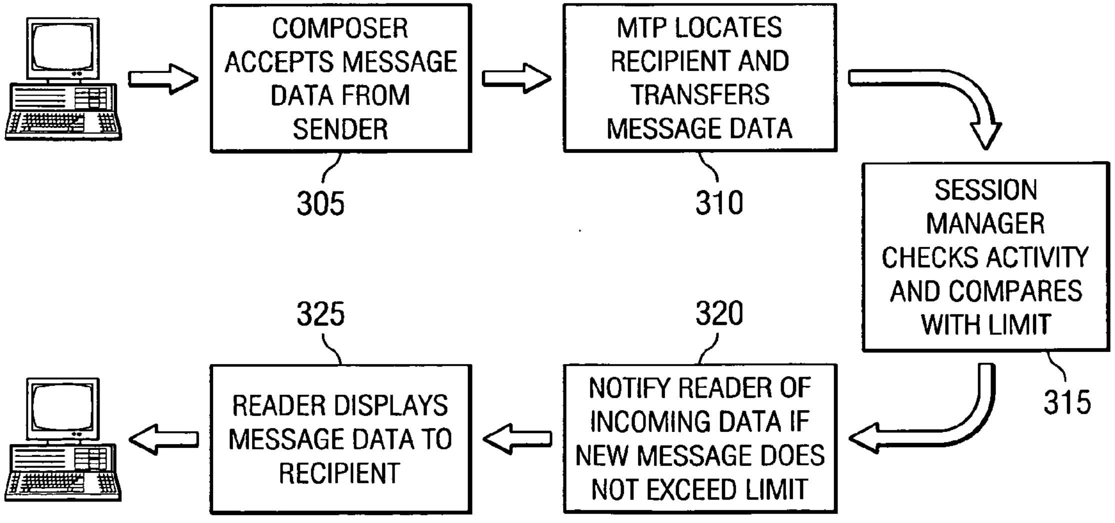
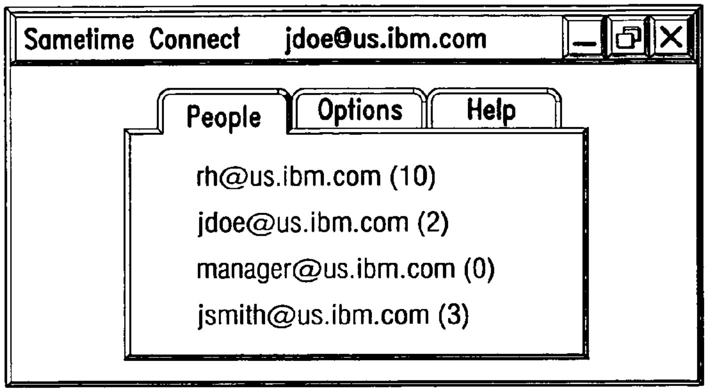

COMPOSER ACCEPTS MESSAGE DATA FROM SENDER
305
MTP LOCATES RECIPIENT AND TRANSFERS MESSAGE DATA
310
SESSION MANAGER CHECKS ACTIVITY AND COMPARES WITH LIMIT
315
320
NOTIFY READER OF INCOMING DATA IF NEW MESSAGE DOES NOT EXCEED LIMIT
325
READER DISPLAYS MESSAGE DATA TO RECIPIENT

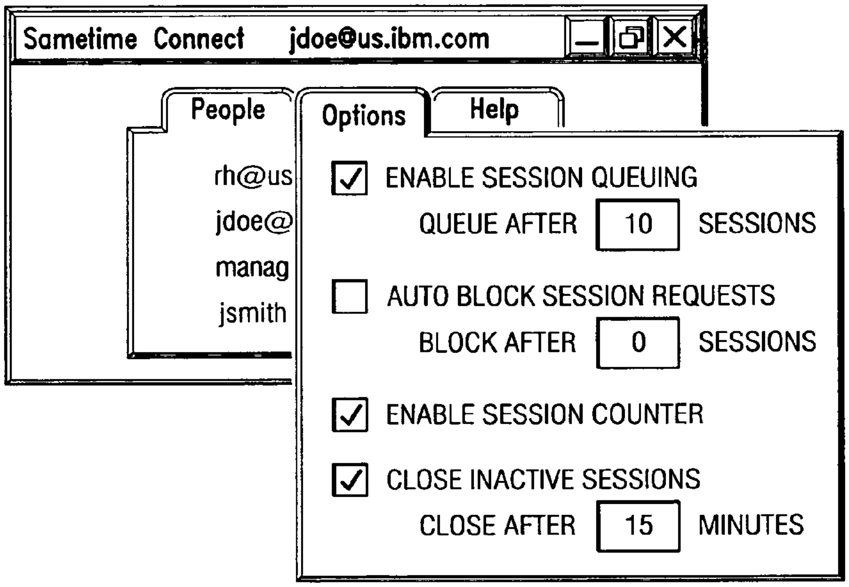
FIG. 3



rh@us.ibm.com (10)
jdoe@us.ibm.com (2)
manager@us.ibm.com (0)
jsmith@us.ibm.com (3)

FIG. 4

FIG. 5

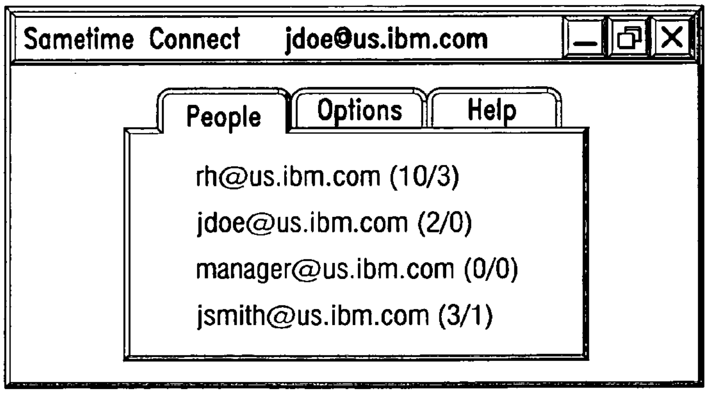


☑ ENABLE SESSION QUEUING
QUEUE AFTER 10 SESSIONS

□ AUTO BLOCK SESSION REQUESTS
BLOCK AFTER 0 SESSIONS

☑ ENABLE SESSION COUNTER

☑ CLOSE INACTIVE SESSIONS
CLOSE AFTER 15 MINUTES



rh@us.ibm.com (10/3)
jdoe@us.ibm.com (2/0)
manager@us.ibm.com (0/0)
jsmith@us.ibm.com (3/1)

FIG. 7

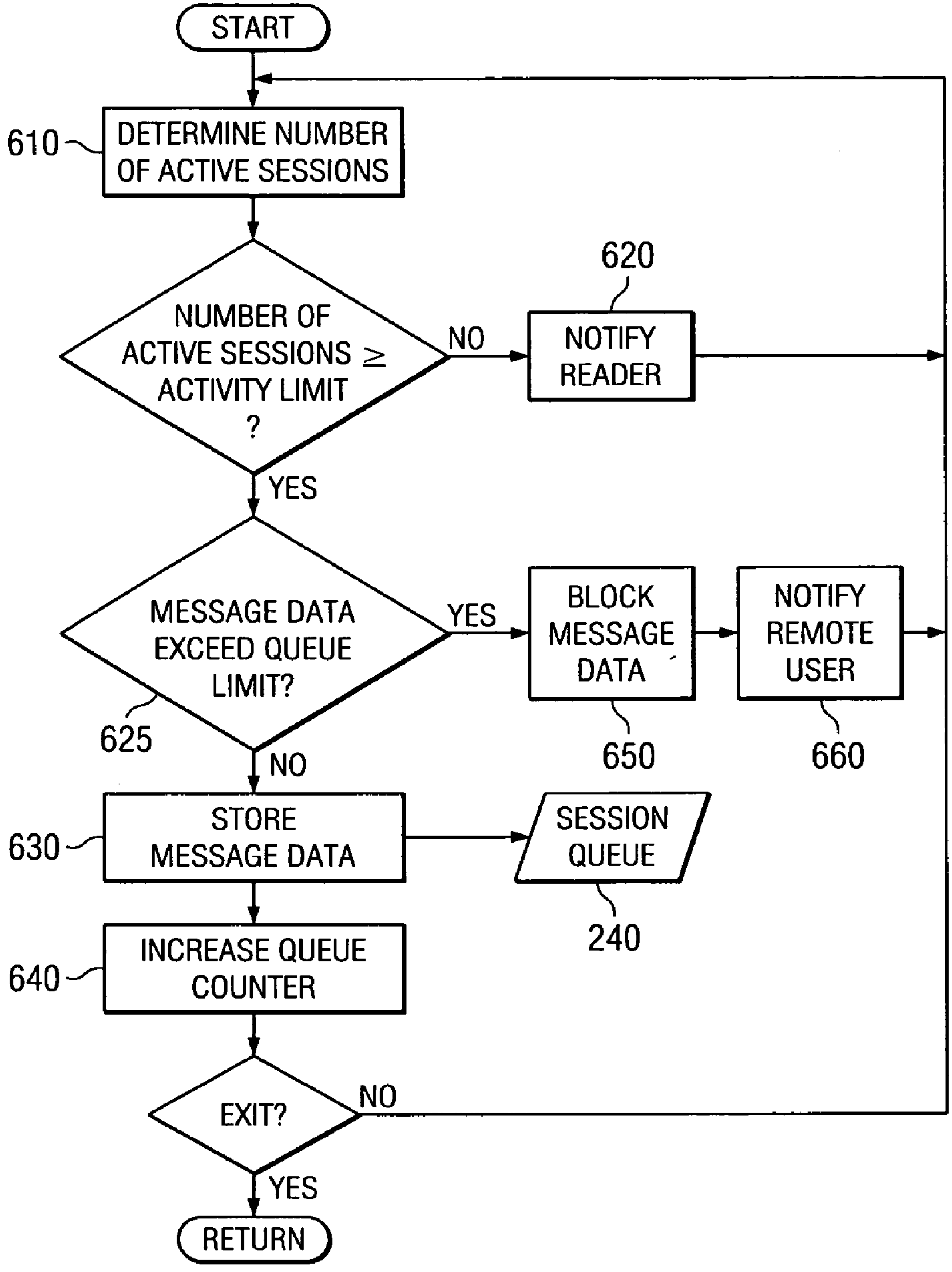

START
610
DETERMINE NUMBER OF ACTIVE SESSIONS
NUMBER OF ACTIVE SESSIONS ≥ ACTIVITY LIMIT ?
NO
620
NOTIFY READER
YES
MESSAGE DATA EXCEED QUEUE LIMIT?
625
YES
BLOCK MESSAGE DATA
650
NOTIFY REMOTE USER
660
NO
630
STORE MESSAGE DATA
SESSION QUEUE
240
640
INCREASE QUEUE COUNTER
EXIT?
NO
YES
RETURN

FIG. 6

SESSION MANAGEMENT ENHANCEMENTS FOR INSTANT MESSAGING APPLICATIONS

FIELD OF THE INVENTION

[0001] The present invention is a process for using electrical computers or digital processing systems to transfer data via one or more communications media. In particular, the present invention comprises an improved demand-based messaging system that enables a message recipient to manage messaging sessions.

BACKGROUND OF THE INVENTION

[0002] Demand-based messaging is a communication service that allows people to exchange message data, such as text, over a network or other communications media, in real time. Probably the most common medium for exchange is the Internet, but as wireless phone networks continue to expand, their popularity for text messaging is also expanding. U.S. Pat. No. 6,301,609 issued to Aravamudun et al., and U.S. Patent Publications Nos. 2002/0035605 and 2004/0254998, for example, illustrate the move toward an exchange medium that unifies traditional and wireless communications. Instant messaging (IM) is perhaps the most widely known and used embodiment of demand-based messaging. Today, most network and online service providers offer some form of IM service. According to some estimates, the top three instant messaging service providers serve over forty million users. Instant messaging services also are being rapidly deployed and integrated into enterprise infrastructure. International Business Machines, Inc. (IBM), for example, has deployed LOTUS SAMETIME instant messaging applications for employees world-wide. Other examples of IM applications that are popular today include MSN Messenger and Yahoo/AOL Instant Messenger. Web-based interfaces are also gaining popularity, as illustrated in U.S. Pat. No. 6,651,086 issued to Manber et al., which describes how a user can join conversations about topics that are presented as web content.

[0003] IM users typically use a networked computer and an IM client program to exchange messages with one another in conversational style. An IM client provides an interface for users to compose, send, receive, and read messages. In a graphical display, an IM client usually includes at least two windows: a window for composing and sending messages, and a window for displaying messages as users take turns sending and receiving them. IM sessions (colloquially referred to as "chats") are often lengthy, with multiple participants each taking many turns "speaking" in the chat window. It is common for one user to have multiple IM chats running simultaneously, usually in separate windows.

[0004] Demand-based messaging services, including instant messaging services, no doubt owe much of their success to the convenience and efficiency with which communications can be exchanged. Unfortunately, the popularity of such messaging services directly affects the convenience and efficiency with which users can exchange communications, even making such communications disruptive at times. For example, it is not uncommon for users to have important instant messaging sessions active with one or more users, while other users continue to interrupt the flow of communications with unrelated messages.

[0005] Current messaging applications provide minimal configuration and control to a user, often including only rudimentary means for blocking individual users from initiating a messaging session. With such limited means for managing the exchange of communications, any given messaging user is available to some sub-set of other users. Thus, existing messaging applications remain too cumbersome to manage communications effectively and there remains a need to advance the state of the art of demand-based messaging to overcome these shortcomings.

SUMMARY OF THE INVENTION

[0006] The invention comprises an improved demand-based messaging system that enables a user to effectively manage multiple messaging sessions.

[0007] The messaging system comprises a messaging program operable on a plurality of electrical computers or data processing machines connected by one or more communications media. The messaging program comprises a conventional message composer program, a conventional message transport program, a conventional message reader program, and an inventive, user-configurable, policy-driven session management program.

BRIEF DESCRIPTION OF DRAWINGS

[0008] The novel features believed characteristic of the invention are set forth in the appended claims. The invention itself, however, as well as a preferred mode of use, further objectives and advantages thereof, will be understood best by reference to the following detailed description of an illustrative embodiment when read in conjunction with the accompanying drawings, wherein:

[0009] **FIG. 1** represents an exemplary network of hardware devices;

[0010] **FIG. 2** is a schematic of a memory having the components of the present invention stored therein;

[0011] **FIG. 3** provides a functional overview of the messaging program of the present invention, as it interacts with a first user sending a message to a second user over a network of hardware devices;

[0012] **FIG. 4** illustrates a hypothetical interface to the messaging program of the present invention;

[0013] **FIG. 5** is an exemplary interface, in which controls for the present invention are integrated into a conventional interface for selecting options;

[0014] **FIG. 6** is a flowchart of the present invention as it implements a queuing policy; and

[0015] **FIG. 7** illustrates an alternative embodiment of the hypothetical interface to the messaging program of the present invention.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENT

[0016] The principles of the present invention are applicable to a variety of computer hardware and software configurations. The term "computer hardware" or "hardware," as used herein, refers to any machine or apparatus that is capable of accepting, performing logic operations on, storing, or displaying data, and includes without limitation processors and memory; the term "computer software" or "software," refers to any set of instructions operable to cause computer hardware to perform an operation. A "computer," as that term is used herein, includes without limitation any useful combination of hardware and software, and a "computer program" or "program" includes without limitation any software operable to cause computer hardware to accept, perform logic operations on, store, or display data. A computer program may, and often is, comprised of a plurality of smaller programming units, including without limitation subroutines, modules, functions, methods, and procedures. Thus, the functions of the present invention may be distributed among a plurality of computers and computer programs. The invention is described best, though, as a single computer program that configures and enables one or more general-purpose computers to implement the novel aspects of the invention. For illustrative purposes, the inventive computer program will be referred to as the "messaging program."

[0017] Additionally, the messaging program is described below with reference to an exemplary network of hardware devices, as depicted in **FIG. 1**, through which the messaging program can transfer data from one user to another. A "network" comprises any number of hardware devices coupled to and in communication with each other through a communications medium, such as the Internet. A "communications medium" includes without limitation any physical, optical, electromagnetic, or other medium through which hardware or software can transmit data. For descriptive purposes, exemplary network **100** has only a limited number of nodes, including workstation computer **105**, workstation computer **110**, server computer **115**, and persistent storage **120**. Network connection **125** comprises all hardware, software, and communications media necessary to enable communication between network nodes **105-120**. Unless otherwise indicated in context below, all network nodes use publicly available protocols or messaging services to communicate with each other through network connection **125**.

[0018] Messaging program **200** and its components, composer **205**, session manager **210**, and reader **215** typically are stored in a memory, represented schematically as memory **220** in **FIG. 2**. The term "memory," as used herein, includes without limitation any volatile or persistent medium, such as an electrical circuit, magnetic disk, or optical disk, in which a computer can store data or software for any duration. A single memory may encompass and be distributed across a plurality of media and network nodes. Thus, **FIG. 2** is included merely as a descriptive expedient and does not necessarily reflect any particular physical embodiment of memory **220**. As depicted in **FIG. 2**, though, memory **220** may include additional data and programs. Of particular import to messaging program **200**, memory **220** may include message transfer program (MTP) **230** and session queue **240**, with which messaging program **200** interacts.

[0019] **FIG. 3** provides a functional overview of messaging program **200**, as it interacts with a first user sending a message to a second user over a computer network. Inasmuch as the following discussion largely addresses the features of messaging program **200** from a recipient's perspective, a message recipient (the second user in the example of **FIG. 3**) is referred to as the "local user." A user that initiates a message is referred to as the "remote user." Messaging program **200** is loaded into the memory of both users' computers. Composer **205** accepts message data from the remote user (**305**), and then, upon the remote user's request, MTP **230** locates the local user's computer on the network and transfers the message data to the local user's computer (**310**). Session manager **210** on the local user's computer then examines the local user's current activity and the local user's activity policy (**315**). In general, the local user's activity comprises the number of messaging sessions in which the local user is presently "active," while the local user's activity policy indicates the threshold of activity at which session manager **210** should intervene to control the flow of messages or message notices to reader **215**. The term "active" in this context is inherently subjective, and session manager **210** can be configured to recognize active sessions based upon context or a user's preference. Thus, while there are many ways to define an "active" session, a specific definition is not material to the inventive features of messaging program **200**. Referring again to **FIG. 3** for illustration, if the local user's activity policy permits, session manager **210** notifies reader **215** of incoming message data (**320**), and reader **215** displays the incoming message data to the local user (**325**). Alternatively, MTP **230** transfers the message data to a messaging server (such as server computer **115**), and MTP **230** on the local user's computer retrieves the message data from the messaging server. Those skilled in the art will appreciate that many similar functional variations are possible and not all are described here.

[0020] As briefly described above, session manager **210** is driven by a user-configurable activity policy that reflects a local user's preferences for accepting new message data while actively engaged in other messaging sessions. Thus, an activity policy is a flexible concept in which many variations of the policies described in detail here are possible. In the preferred embodiment, though, session manager **210** implements high-level functions in the following configurable policies: a "courtesy" policy, a "blocking" policy, a "queuing" policy, and an "inactive session" policy. The functions of each policy are described in more detail in the following discussion.

[0021] If the courtesy policy is applied, session manager **210** broadcasts the number of sessions that the local user has active at any given time, so that other instances of messaging program **200** running on remote computers can display this number to remote users. Thus, the courtesy policy causes session manager **210** to provide information to remote users, but relies upon the remote users to exercise their own personal judgment regarding the propriety of requesting a session with someone that already has multiple active sessions. **FIG. 4** illustrates a hypothetical interface to session manager **210** in which the courtesy policy has been applied, where the parenthetical number next to each user's identify indicates the number of sessions that the user has active. The hypothetical interface of **FIG. 4** is representative of both a local user's and a remote user's interface, depending upon context.

[0022] The blocking policy configures session manager **210** to block new message data if the number of active sessions exceeds an "activity limit," or if the request is from a remote user identified in a "block list." The activity limit represents the maximum number of sessions that can be active at any given time, while the block list identifies specific remote users whose requests should be denied at any given time. Both the activity limit and the block list are user-configurable parameters. The local user may configure these parameters through a graphical interface, in which case session manager **210** stores the parameters in a configuration file, or the local user may edit the configuration file directly. **FIG. 5** is an exemplary interface, in which controls for the parameters described herein are integrated into a conventional interface for selecting options. Many other techniques for configuring and storing parameters are well-known in the art, and the particular technique applied is not material to the inventive features of messaging program **200**. Referring again to **FIG. 4** for illustration, if the user identified as "rh@us.ibm.com" is the local user and has set the activity limit to 10 active sessions, any new message data from the remote user (identified as "jdoe@us.ibm.com" in this example) would be blocked until at least one of the local user's current sessions ends or becomes inactive. Session manager **210** optionally notifies the remote user that the request was blocked based upon the local user's number of active sessions.

[0023] The queuing policy extends the blocking policy so that session manager **210** stores new message data in session queue **240**, rather than blocking the new message data, if the number of active sessions exceeds the activity limit. **FIG. 6** is a flowchart of session manager **210** as it implements a queuing policy. The maximum size of session queue **240**, and thus the number of new messages that session manager **210** can store therein, is a user-configurable parameter, referred to herein as the "queue limit." Thus, with the queuing policy applied, session manager **210** first determines the number of active sessions (**610**). If the number of active sessions does not exceed the activity limit, then messaging program **200** proceeds in a conventional manner and notifies reader **215** that new message data has been received (**620**). If the number of active sessions has reached the activity limit, then session manager **210** determines if the new message data would exceed the queue limit (**625**). If the new message data does not exceed the queue limit, session manager **210** stores the new message data (**630**) in session queue **240** and increments a queued message counter (**640**), but does not notify reader **215**. But if the new message data does exceed the queue limit, then session manager **210** blocks the message data (**650**) and, optionally, notifies the remote user (**660**) that the message data has been blocked because of activity limits. When an active session ends or becomes inactive, session manager **210** decreases the queued session counter and notifies reader **215** that the new message data is ready to be retrieved from session queue **240**. Reader **215** then removes the new message data from session queue **240** and processes the new message data in a conventional manner that is familiar to those skilled in the art.

[0024] Interfaces to messaging program **200**, such as the interface illustrated in **FIG. 4**, may be modified to display queue status data, such as the queued message counter and a given user's position within a queue. **FIG. 7** illustrates an exemplary modified interface for a user identified as jdoe@us.ibm.com. In **FIG. 7**, queue status data is displayed for some subset of users selected by jdoe@us.ibm.com, and the meaning of the parenthetical numbers depends upon context. For example, if jdoe@us.ibm.com is a remote user and sends message data to local user rh@us.ibm.com, then the parenthetical information on display of **FIG. 7** would indicate that local user rh@us.ibm.com has 10 active sessions and that jdoe@us.ibm.com's request is the third in the local user's queue. If jdoe@us.ibm.com has not sent any messages to rh@us.ibm.com, then the parenthetical information next to rh@us.ibm.com indicates that rh@us.ibm.com has 10 active sessions and a total of three messages in the queue. In yet another context, the parenthetical information next to jdoe@us.ibm.com indicates that jdoe@us.ibm.com has two active sessions and no messages waiting in the queue. Additionally, if either user places a mouse cursor over the number, or takes some other appropriate triggering action, messaging program **200** would display the identities of the active users, the queued users, or both.

[0025] The inactive session policy allows a user to configure an "inactive" session parameter that defines an inactive session in terms of a period of time in which no messages are exchanged with a given remote user. A user also can configure an inactive session policy so that messaging program **200** takes user-selected action when a session is identified as inactive, such as archiving the session messages and closing the session interface.

[0026] A preferred form of the invention has been shown in the drawings and described above, but variations in the preferred form will be apparent to those skilled in the art. The preceding description is for illustration purposes only, and the invention should not be construed as limited to the specific form shown and described. The scope of the invention should be limited only by the language of the following claims.

What is claimed is:

1. A demand-based messaging system for opening and managing messaging sessions, the system comprising:

a configuration means for configuring an activity limit parameter;

a receiving means for receiving a message from a remote user;

a tracking means for determining the number of messaging sessions that are active before processing the message;

an activity display means for displaying the number of active messaging sessions; and

a session management means comprising a blocking means for blocking the message if the number of active messaging sessions exceeds the activity limit parameter.

2. The demand-based messaging system of claim 1, wherein the session management means further comprises:

a queuing means for storing the blocked message in a session queue until the tracking means determines that the number of active messaging sessions is less than the activity limit parameter; and

a message processing means for displaying the message to a local user and removing the message from the session queue if the tracking means determines that the number of active messaging sessions is less than the activity limit parameter.

3. The demand-based messaging system of claim 2 wherein the session management means further comprises a queue display means for displaying the number of messages in the session queue.

4. The demand-based messaging system of claim 2 wherein the session management means further comprises a queue display means for displaying the number of messages in the session queue before the queuing means stores the message in the session queue.

5. The demand-based messaging system of claim 1 further comprising a session deactivation means for deactivating messaging sessions that are inactive.

6. The demand-based messaging system of claim 2 further comprising a session deactivation means for deactivating messaging sessions that are inactive.

7. The demand-based messaging system of claim 1 wherein the configuration means further comprises a means for configuring a block list parameter comprising one or more remote user identities, and the blocking means blocks the message if the remote user's identity matches a remote user identity in the block list parameter.

8. A computer-implemented method of managing messaging sessions, the method comprising:

configuring an activity limit parameter;

receiving a message from a remote user;

determining the number of messaging sessions that are active before processing the message;

displaying the number of active messaging sessions; and

blocking the message if the number of active messaging sessions exceeds the activity limit parameter.

9. The computer-implemented method of claim 8 further comprising:

storing the blocked message in a session queue until the number of active messaging sessions is less than the activity limit parameter; and

displaying the message to a local user and removing the blocked message from the session queue if the number of active messaging sessions is less than the activity limit parameter.

10. The computer-implemented method of claim 9 further comprising:

displaying the number of messages stored in the session queue.

11. The computer-implemented method of claim 9 further comprising:

displaying the number of messages stored in the session queue before storing the message in the session queue.

12. The computer-implemented method of claim 8 further comprising:

deactivating messaging sessions that are inactive.

13. The computer-implemented method of claim 9 further comprising: deactivating messaging sessions that are inactive.

14. The computer-implemented method of claim 8 further comprising:

configuring a block list parameter comprising one or more remote user identities; and

blocking the message if the remote user's identity matches a remote user identity in the block list parameter.

15. The computer-implemented method of claim 9 further comprising:

configuring a block list parameter comprising one or more remote user identities; and

blocking the message if the remote user's identity matches a remote user identity in the block list parameter.

16. A computer-readable medium having computer-executable instructions for performing a method of managing messaging sessions, wherein the method comprises:

configuring an activity limit parameter;

receiving a message from a remote user;

determining the number of messaging sessions that are active before processing the message;

displaying the number of active messaging sessions; and

blocking the message if the number of active messaging sessions exceeds the activity limit parameter.

17. The computer-readable medium of claim 16, wherein the method further comprises:

storing the blocked message in a session queue until the number of active messaging sessions is less than the activity limit parameter; and

displaying the message to a local user and removing the blocked message from the session queue if the number of active messaging sessions is less than the activity limit parameter.

18. The computer-readable medium of claim 17, wherein the method further comprises:

displaying the number of messages stored in the session queue.

19. The computer-readable medium of claim 16, wherein the method further comprises:

configuring a block list parameter comprising one or more remote user identities; and

blocking the message if the remote user's identity matches a remote user identity in the block list parameter.

20. The computer-readable medium of claim 17, wherein the method further comprises:

configuring a block list parameter comprising one or more remote user identities; and

blocking the message if the remote user's identity matches a remote user identity in the block list parameter.

* * * * *