FILE TRANSFER MECHANISM

This invention relates to a file transfer mechanism for transferring a series of data files from one computer to another computer and also to a method of transferring a series of data files from one computer to another computer.

When transferring a large series of files from one computer to another computer, it is desirable to do so in a manner which does not cause problems for the computer receiving the files and which enables the files to be transferred in a reliable manner.

According to a first aspect of this invention there is provided a computer having a mechanism for transferring a series of data files between said computer and another computer, said file transfer mechanism being arranged to transfer a series of files between said computer and another computer in accordance with a protocol for transferring a series of files from a first computer (the transmitting computer) to a second computer (the receiving computer) which includes the following sequence of operations: (i) the receiving computer sends a poll message to the transmitting computer asking for a list of files which are available for transfer; (ii) the transmitting computer sends a reply message to the receiving computer containing a list of files which are available for transfer; and (iii) the receiving computer selects files from the list received from the transmitting computer and attempts to copy each selected file.

The protocol used in the file transfer mechanism of this invention provides the advantages that the files are transferred at a time selected by the receiving computer and that only those files which have been selected by the receiving computer are transferred. Also, because the files are copied under the control of the receiving computer, the protocol provides the further advantage that the files may be copied in a reliable manner.

This Sheet Has Been Replaced



This Sheet Has Been Replaced

According to a second aspect of this invention, there is provided a computer having a mechanism for transferring a series of data files between said computer and another computer, said file transfer mechanism being arranged to transfer a series of between said computer and another computer in accordance with a protocol for transferring a series of files from a first computer (the transmitting computer to a second computer (the receiving computer) which includes the following sequence of operations: (i) the transmitting computer sends a message to the receiving computer stating that it has files available for transfer; (ii) the receiving computer sends a poll message to the transmitting computer asking for a list of files which are available transfer; (iii) the transmitting computer sends a reply message to the receiving computer containing a list of files which are available for transfer; and (iv) the receiving computer selects files from the list received from the transmitting computer and attempts to copy each selected file.

According to a third aspect of this invention there is a provided a computer having a mechanism for transferring a series of data files between said computer and another computer, said file transfer mechanism being arranged to transfer a series of files between said computer and another computer in accordance with a protocol for transferring a series of files from a first computer (the transmitting computer) to a second computer (the receiving computer) which includes the following sequence of operations: (i) the receiving computer sends a poll message to the transmitting computer asking for a list of files which are available for transfer; (ii) the transmitting computer sends a poll message to the receiving computer stating that there will be a delay in providing a list of files available for transfer; (iii) after a delay the transmitting computer sends a further message to the receiving computer stating that it has files available for transfer; (iv) the receiving computer sends a poll message to the transmitting computer asking for a list

of files which are available for transfer; (v) the transmitting computer sends a reply message to the receiving computer containing a list of files which are available for transfer; and (vi) the receiving computer selects files from the list received from the transmitting computer and attempts to copy each selected file.

According to a fourth aspect of this invention there is provided a method of transferring a series of data files from a first computer (the transmitting computer) to a second computer (the receiving computer) including the steps of: (i) the receiving computer sending a poll message to the transmitting computer asking for a list of files which are available for transfer; (ii) the transmitting computer sending a reply message to the receiving computer containing a list of files which are available for transfer; and (iii) the receiving computer selecting files from the list received from the transmitting computer and attempting to copy each selected file.

According to a fifth aspect of this invention there is provided a method of transferring a series of data files from a first computer (the transmitting computer) to a second computer (the receiving computer) including the steps of: (i) the transmitting computer sending a message to the receiving computer stating that it has files available for transfer; (ii) the receiving computer sending a poll message to the transmitting computer asking for a list of files which are available for transfer; (iii) the transmitting computer sending a reply message to the receiving computer containing a list of files which are available for transfer; and (iv) the receiving computer selecting files from the list received from the transmitting computer and attempting to copy each selected file.

According to a sixth aspect of this invention, there is provided a method of transferring a series of data files from a first computer (the transmitting computer) to a second computer (the receiving computer) including the steps of:

This Sheet Has Been Replaced

(i) the receiving computer sending a poll message to the transmitting computer asking for a list of files which are available for transfer; (ii) the transmitting computer sending a reply message to the receiving computer stating that there will be a delay in providing a list of files available for transfer; (iii) after a delay the transmitting computer sending a further message to the receiving computer stating that it has files available for transfer; (iv) the receiving computer sending a poll message to the transmitting computer asking for a list of files which are available for transfer; (v) the transmitting computer sending a reply message to the receiving computer containing a list of files which are available for transfer; and (vi) the receiving computer selecting files from the list received from the transmitting computer and attempting to copy each selected file.

This invention will now be described in more detail, by way of example, with reference to the drawings in which:

Figure 1 is a block diagram of two computers between which a series of files may be transferred and each of which embodies this invention;

Figure 2 is a block diagram showing the software components of one of the computers of Figure 1;

Figure 3 shows the individual parts of the file transfer mechanism component of the software and the relationship between these processes and the application programs component of the software;

Figure 4 is a graph illustrating a basic service provided by a protocol used in the file transfer mechanism;

Figure 5 is a graph showing an alarm service provided by the protocol used in the file transfer mechanism;

Figure 6 is a graph illustrating a delayed service provided by the protocol used in the file transfer mechanism;

Figure 7 is a graph showing a message service used in the protocol provided by the file transfer mechanism;

This Sheet Has Been Replaced



This Sheet Has Been Replaced

Figure 8 is a table illustrating the relationship between the states of a finite state machine used in a computer receiving data files and associated events; and

Figure 9 is a table illustrating the relationship between the states of a finite state machine used in a computer transmitting data files and associated events.

Referring now to Figure 1, there are shown two computers, namely, computer A and computer B between which data files may be transmitted via a communications link 10. Each of computers A and B uses a UNIX operating system. Each of computers A and B may also transmit data files to, and receive data files from, other computers, not shown.

The present invention is particularly suitable for transferring a series of data files from a computer which produces such files in very large numbers. Examples of such computers may be found in a system for billing telephone calls. In such a system, the data files may be records of individual telephone calls.

Referring now to Figure 2, there are shown the main components of the software (in addition to the operating system) of each of the computers A and B. These components comprise an application programs component 20, a file transfer mechanism component 22, a file transfer protocol component 24 and a communication component 26. The application programs component 20 comprises one or more application programs and the nature of these programs will depend upon the operations performed by the computer. For example, where the computer forms part of a billing system, the application programs will perform the functions required of that computer within the billing system. The details of such application programs do not form part of the present invention. The file transfer mechanism 22 is responsible for the transfer of a series of files between the computer and another computer and thus provides overall control of each transfer of a series of files. The file transfer protocol component 24 is responsible for ensuring each file is copied in a reliable manner. The file transfer mechanism component

22 will be described below in grater detail. In the present example, the file transfer protocol component 24 uses the File Transfer Protocol (ftp) available on a UNIX operating system. By way of an alternative, the file transfer protocol component 24 may use the File Transfer and Access Management (FTAM) protocol of Open Systems Interconnection (OSI). The communication component 26 is responsible for controlling the transmission of individual bytes of data between the computer and another computer and, in the present example, the communication component 26 uses the two well known protocols known by the collective name TCP/IP and these two protocols are available on UNIX operating systems.

Referring now to Figure 3, there are shown the parts of the file transfer mechanism 22 and the relationship of these parts with the application programs 20. The parts comprise a receiver 30, a despatcher 32, a file store 34 and a calendar 36. The receiver 30 is responsible for receiving data files from another computer and it transfers received files to a database for use by one or more of the application programs 20. Data files generated by one or more of the application programs 20 are transferred to the file store 34 in preparation for transfer to one or more other computers. Where the computer transmits files to only one other computer, the data files are stored in a single queue. Where it transmits files to more than one computer, the data files are stored in an individual queue for each of the other computers. Thus, the queue or each queue of data files forms a series of files which are available for transfer to another computer.

The despatcher 32 is responsible for transmitting files from the file store 34 to one or more other computers. The file transfer mechanism 22 provides the opportunity for data files to be transmitted to other computers in accordance with a pre-defined schedule. Where data files are transmitted in accordance with the schedule, the calendar 36 provides the necessary time keeping function.

This Sheet Has Been Replaced



This Sheet Has Been Replaced



The transfer mechanism 22 is responsible for controlling each transfer of a series of files between the computer in which it is located and another computer. Each transfer is controlled in accordance with a protocol which is illustrated in Figures 4 to 7. This protocol provides a basic service illustrated in Figure 4, an alarm service illustrated in Figure 5, a delayed service illustrated in Figure 6 and a message service illustrated in Figure 7. In each of Figures 4 to 7, the computer which is receiving data files is referred to as an "initiator" and a computer which is transmitting files is referred to as a "responder".

Referring now to Figure 4, in the basic service, in order to commence each transfer of a series of files the initiator sends a poll message 40 to the responder. At the same time, the initiator starts a timer. The poll message 40 includes a unique identifier for this transfer and details of the data files which it wishes to receive. These details are expressed as parameters which define the type of files which it wishes to receive. The unique identifier may contain the name of the initiator and a serial number for this transfer. If the timer expires before a reply message is received from the responder, the initiator retransmits the poll message 40 to the responder. In the event that the initiator transmits the poll message a preset number of times without receiving a reply message from the responder it ceases to send poll messages and issues an alarm. If the responder is able to generate a list of files which are available for transfer after receiving a poll message 40, it sends a reply message 42. The reply message 42 contains the unique identifier for this transfer as specified in the poll message 40 and also the list of files. For each file, the list gives the name of the file, the size in bytes of the file and the date of creation of the file.

On receiving the reply message 42, the initiator selects those files from the list which it wishes to receive. It then requests the responder to copy each file to it in turn. In Figure 4, the requests for individual files are

This Sheet Has Been Replaced

shown by dotted lines 44 and the copying operation for each file is shown by a solid, arrow headed line 46. The files are copied using the File Transfer Protocol (ftp) provided on the UNIX operating system mentioned above. It is important to note that the responder does not push any file to the initiator without being specifically requested to do so. Consequently, there is no danger of the initiator receiving a file which it is not prepared to receive. If the initiator fails to copy a file, it proceeds by attempting to copy the next file which it has selected from the list. It continues in this manner until it has attempted to copy all of the files. It then makes a further attempt to copy those files which it has not received. It continues attempting to copy each file up to a preset number of times and, if it fails to copy any file, it raises an alarm.

After copying all the selected files, or attempting to do so up to the preset number of times, the initiator sends a message 48 to the responder. This message contains for each selected file the name of the file and a flag representing the outcome of the attempt to copy it. After receiving the message 48, the initiator erases the data files which have been successfully copied from its file store 34. In a modification, the responder transfers the files which have been copied successfully to a further file store, not shown, from which they are eventually erased.

Referring now to Figure 5, at the commencement of the alarm service, the responder sends an alarm message 60 to the initiator to inform it that it wishes to transfer a series of files to it. The responder may issue the message 60 because a point has been reached in the agreed schedule at which files are to be transmitted, or because an application program has reached a point in its execution at which it wishes to transfer files to the initiator or because the queue of files in the responder for transfer to the initiator has reached a preset threshold. At the time that the responder sends the alarm message 60 to the initiator, it starts a timer. The alarm message 60 contains a unique

identifier for the transfer together with an indication that files are available for transfer. If the timer expires before the responder receives a reply from the initiator, it retransmits the alarm message 60 and resets the timer. It retransmits the alarm message 60 upon each expiry of the timer up to a preset number of times and then raises an alarm.

If the initiator is able to receive files, in response to the alarm message 60, it sends a poll message 62 to the responder. The poll message 62 is identical to the poll message 40 described with reference to Figure 4 except that it contains the unique identifier contained in the alarm message 60 and an acknowledgement that it has received this alarm message. After the responder has received the poll message 62, the alarm service continues in the manner described with reference to the basic service after the responder has received the poll message 40.

Thus, the transfer of messages and data files between the initiator and responder in the alarm service is identical to that in the basic service except that the alarm service includes the alarm message 60 and the poll message 62 uses the unique identifier contained in the alarm message 60 and acknowledges receipt of the alarm message.

The responder uses the delayed service when it requires some time to make files requested by the initiator available for transfer. For example, if the responder contains an application program which is capable of retrieving the requested files from an archive and transferring these to the data store of the associated transfer mechanism, there will be a delay in making these files available for transfer.

The delayed service commences in the same way as the basic service with the initiator sending a poll message 70 to the responder. The poll message 70 is identical to the poll message 40 of the basic service and, if necessary, is retransmitted up to the preset number of times.

This Sheet Has Been Replaced

This Sheet Has Been Replaced

After receiving the poll message 70, the responder sends an acknowledgement message 72. The acknowledgement message 72 includes the unique identifier of the poll message 70 and a statement that there will be a delay before files are available for transfer.

After a delay, when either files are available for transfer or the responder has found that it is not able to provide any files of the type requested in the poll message 70, it transmits an alarm message 74 to the initiator. The alarm message 74 contains the unique identifier of the poll message 70 and indicates either that files are available for transfer or that files cannot be made available for transfer. If the responder does not receive a reply to the alarm message 74, it retransmits the alarm message 74 in the same manner as described for the alarm message 60 of the alarm service.

In response to the alarm message 74, the initiator transmits a further poll message 76. The poll message 76 includes the unique identifier of the poll message 70 and also the details of the data files which it wishes to receive.

In response to the poll message 76, the responder transmits a reply message 42 which identical in format to the reply message 42 used in the basic service. The files are then transferred and acknowledged in the same manner as described with reference to the basic service.

Thus, the delayed service is identical to the basic service but additionally includes the poll message 70, the acknowledgement 72 and the alarm message 74. Also, the poll message 76 re-uses the unique identifier from the poll message 70 and acknowledges the alarm message 74.

In each of the basic service, the alarm service and the delayed service, the transfer of data files is controlled by the initiator. More specifically, in each service, the initiator receives a list of files in response to a poll message and thus there is no danger that the initiator will receive a list of files until it is ready to process the

list. In each service, after receiving the list, the files are selected and copied under the control of the initiator and so there is no danger that the initiator will receive any file which is not ready to receive. The alarm service provides the further advantage that the responder can indicate that files are ready for transfer.

Although the basic service, alarm service and delayed service have each been described as forming part of the protocol used by the file transfer mechanism, it is to be appreciated that each of these services is also an individual protocol which could be used on its own or could be modified to include further messages.

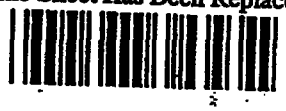
The message service is used to transfer a comparatively small amount of data from the responder to the initiator. At the commencement of the message service, the initiator sends a request message 80 to the responder and at the same time it starts a timer. The request message 80 contains a unique identifier and also the details of the data required from the responder. If the timer expires before receiving a reply message from the responder, the initiator retransmits the request message 80 and restarts the timer. If no reply is received, it retransmits the request message 80 and restarts the timer up to a preset number of times.

If the responder is able to provide the data, it sends a reply message 82 to the initiator which contains the unique identifier of the request message 80 and the requested data.

For each individual transfer using one of the four services described above, both the initiator and responder create a finite state machine. Each finite state machine changes state after a given event has occurred. Each event is associated with an action process comprising a set of steps.

The events are set out in Table 1. In this table, the suffix "req" denotes the sending of a message and the suffix "ind" denotes the receipt of a message. Each finite state machine can assume six states and these six states are set

This Sheet Has Been Replaced



This Sheet Has Been Replaced

out in Table 2. It should be noted that the meaning of each state differs between the initiator and the responder.

TABLE 1

1.	POLLreq	Application program on the Initiator sends a poll request to the Responder.
2.	POLLind	The Responder receivers a poll request message.
3.	POLLRESPreq	Application program on the Responder sends a poll response back to the Initiator
4.	POLLRESPind	The Initiator receives a poll response message.
5.	POLLACKreq	Application program on the Responder sends a poll acknowledgement back to the Initiator.
6.	POLLACKind	The Initiator receives a poll acknowledgement message.
7.	ACKreq	Application program on the Initiator sends an acknowledgement to the Responder.
8.	ACKind	The Responder receives an acknowledgement message.
9.	ALARMreq	Application program on the Responder sends an alarm to the Initiator.
10.	ALARMind	Initiator receives an alarm message.
11.	MESSAGEreq	Application program on the Initiator sends a message to the Responder.
12.	MESSAGEind	Responder receives a message.
13.	RESPONSEreq	Application program on the Responder sends a response message back to the Initiator.
14.	RESPONSEind	Initiator receives a response message.
15.	TIMEOUTind	a timer has expired.

Note that some of the events are handled by the Initiator and the others by the Responder.

This Sheet Has Been Replaced



TABLE 2

0.	IDLE	
	Initiator:	no activity.
	Responder:	no activity.
1.	POLL	
	Initiator:	a poll message has been sent to the Responder - awaiting a reply.
	Responder:	received and passed on a poll request to application program - awaiting a reply from application program.
2.	ACK	
	Initiator:	a poll response message received - awaiting acknowledgement from application program.
	Responder:	poll response message sent - awaiting acknowledgement.
3.	DELAY	
	Initiator:	a poll acknowledgement received - awaiting an alarm.
	Responder:	poll acknowledgement sent - awaiting alarm request from application program.
4.	ALARM	
	Initiator:	an alarm received - awaiting a poll request from application program.
	Responder:	alarm message sent - awaiting a poll request.
5.	MESSAGE	
	Initiator:	a message request sent to Responder - awaiting response.
	Responder:	message request received and passed to application program - awaiting response from application program.

The state tables for the initiator and responder are shown, respectively, in Figures 8 and 9. In each of these figures, the events are set out in the left hand column and the six states are set out across the top of the table. The

numerals in the boxes indicate the new state following a particular event and the letters and numerals contained in brackets provide a reference to the action process associated with a particular event. Thus, in the initiator states table shown in Figure 8, following an event POLLreq, the state changes from state 0 (IDLE) to state 1 (POLL). The event POLLreq is associated with an action process A0. The various action processes will now be described. For each action process, the series of steps differs between the initiator and the responder.

Action Process A0

In this action process, the initiator generates a new finite state machine and a unique identifier for the transfer, formats the poll message and sends it to the responder, starts a timer and then enters State 1. The responder generates a new finite state machine, passes the poll message to an application program in the responder and then enters State 1.

Action Process A4

In this action process, the initiator formats a poll message and sends it to the responder, starts a timer and then enters State 1. The responder passes a poll message to an application program and then enters State 1.

Action Process B1

In this action process, the initiator selects the data files which it wishes to receive from the list of data files supplied by the responder, determines the total size of the data files and checks that it has free space available and then copies each file in turn. If it fails to copy a file, it attempts to copy the next one and it repeats its attempt to copy each file up to a preset number of times. It then enters State 2 and prepares to send an acknowledgement message. For this action process, the responder formats a

This Sheet Has Been Replaced



poll response message, sends it to the initiator and then enters State 2.

Action Process C1

In this action process, the initiator changes the state of its finite state machine for the transfer to State 3. In this action process, the responder formats a poll acknowledgment message, sends it to the initiator and enters State 3.

Action Process D2

In this action process, the initiator formats an acknowledgement message, sends it to the responder, causes its finite state machine to enter State 0 and then deletes the finite state machine. The responder passes the acknowledgement message to the application program, causes its finite state machine to enter State 0 and then deletes the finite state machine.

Action Process E0

In this action process, the initiator creates a new finite state machine, stores the unique identifier received in the alarm message from the responder, passes the alarm message to the application program and then causes its finite state machine to enter State 4. The responder generates a new finite state machine and an unique identifier for this transfer, formats an alarm message and sends it to the initiator and then causes its finite state machine to enter State 4.

Action Process E3

In this action process, the initiator passes the alarm message to the application program and then causes the finite state machine for this transfer to enter State 4. The responder formats an alarm message, sends it to the initiator and causes its finite state machine for this transfer to enter State 4.

This Sheet Has Been Replaced



This Sheet Has Been Replaced



Action Process F0

In this action process, the initiator creates a new finite state machine and an unique identifier for this transfer, formats a message request and sends it to the responder and causes the finite state machine to enter State 5. The responder generates a new finite state machine, passes the request message to an application program and causes the finite state machine to enter State 5.

Action Process G5

In this action process, the initiator passes the response message to an application program, causes the finite state machine to enter State 0 and deletes the finite state machine. The responder formats a response message and sends it to the initiator, causes the finite state machine to enter State 0 and then deletes the finite state machine.

Action Process H1

This action process relates only to the initiator. The initiator increments a counter for counting the number of retries. If the counter has not yet reached the preset maximum value, it formats the poll message, sends it the responder and restarts the timer. If the reset counter has reached the preset maximum number, it provides an indication that there is an error, changes the state of the finite state machine to State 0 and then deletes the finite state machine.

Action Process H4

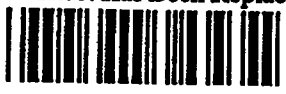
This action process relates only to the responder. In this action process, the responder increments a counter which counts the number of retries. If the counter has not yet reached a preset maximum value, it formats an alarm message and sends it to the initiator and then restarts the timer. If the counter has reached the preset maximum value, it provides an indication that an error has occurred, causes its finite state machine to enter State 0 and then deletes the finite state machine.

Action Process H5

This action process relates only to the initiator. In this action process, the initiator increments a counter for counting the number of retries. If the counter has not reached a preset maximum value, it formats a request message, sends it to the responder and then restarts the timer. If the counter has reached the preset maximum value, it provides an indication that an error has occurred, changes the state of the finite state machine to State 0 and then deletes the finite state machine.

Computers A and B together with other computers may form part of the system under the overall control of a central computer. Each computer in the system may send a log of each transfer to the central computer. If a queue of data files in a file transfer mechanism of one of the computers reached an excessive, the computer can send an indication of this to the central computer which can then take appropriate action to deal with this problem.

This Sheet Has Been Replaced



This Sheet Has Been Replaced

CLAIMS

1. A computer having a mechanism for transferring a series of data files between said computer and another computer, said file transfer mechanism being arranged to transfer a series of files between said computer and another computer in accordance with a protocol for transferring a series of files from a first computer (the transmitting computer) to a second computer (the receiving computer) which includes the following sequence of operations:

(i) the receiving computer sends a poll message to the transmitting computer asking for a list of files which are available for transfer;

(ii) the transmitting computer sends a reply message to the receiving computer containing a list of files which are available for transfer; and

(iii) the receiving computer selects files from the list received from the transmitting computer and attempts to copy each selected file.

2. A computer having a mechanism for transferring a series of data files between said computer and another computer, said file transfer mechanism being arranged to transfer a series of between said computer and another computer in accordance with a protocol for transferring a series of files from a first computer (the transmitting computer to a second computer (the receiving computer) which includes the following sequence of operations:

(i) the transmitting computer sends a message to the receiving computer stating that it has files available for transfer;

(ii) the receiving computer sends a poll message to the transmitting computer asking for a list of files which are available transfer;

(iii) the transmitting computer sends a reply message to the receiving computer containing a list of files which are available for transfer; and

* C L M S *

This Sheet Has Been Replaced

(iv) the receiving computer selects files from the list received from the transmitting computer and attempts to copy each selected file.

3. A computer having a mechanism for transferring a series of data files between said computer and another computer, said file transfer mechanism being arranged to transfer a series of files between said computer and another computer in accordance with a protocol for transferring a series of files from a first computer (the transmitting computer) to a second computer (the receiving computer) which includes the following sequence of operations:

(i) the receiving computer sends a poll message to the transmitting computer asking for a list of files which are available for transfer;

(ii) the transmitting computer sends a poll message to the receiving computer stating that there will be a delay in providing a list of files available for transfer;

(iii) after a delay the transmitting computer sends a further message to the receiving computer stating that it has files available for transfer;

(iv) the receiving computer sends a poll message to the transmitting computer asking for a list of files which are available for transfer;

(v) the transmitting computer sends a reply message to the receiving computer containing a list of files which are available for transfer; and

(vi) the receiving computer selects files from the list received from the transmitting computer and attempts to copy each selected file.

4. A computer as claimed in any one of claims 1 to 3, in which the protocol includes an additional operation in which the receiving computer sends a message to the transmitting computer acknowledging the files which have been copies successfully.

* C L M S *

This Sheet Has Been Replaced

5. A computer as claimed in any one of the preceding claims, in which the file transfer mechanism comprises a file store for storing files to be transmitted, a dispatcher for transmitting files and a receiver for receiving files.

6. A computer as claimed in any one of the preceding claims, in which, for each sequence of operations for transferring a series of files, the file transfer mechanism has a finite state machine which changes state at the completion of each operation.

7. - A method of transferring a series of data files from a first computer (the transmitting computer) to a second computer (the receiving computer) including the steps of:

(i) the receiving computer sending a poll message to the transmitting computer asking for a list of files which are available for transfer;

(ii) the transmitting computer sending a reply message to the receiving computer containing a list of files which are available for transfer; and

(iii) the receiving computer selecting files from the list received from the transmitting computer and attempting to copy each selected file.

8. A method of transferring a series of data files from a first computer (the transmitting computer) to a second computer (the receiving computer) including the steps of:

(i) the transmitting computer sending a message to the receiving computer stating that it has files available for transfer;

(ii) the receiving computer sending a poll message to the transmitting computer asking for a list of files which are available for transfer;

(iii) the transmitting computer sending a reply message to the receiving computer containing a list of files which are available for transfer; and

* C L M S *

This Sheet Has Been Replaced

(iv) the receiving computer selecting files from the list received from the transmitting computer and attempting to copy each selected file.

9. A method of transferring a series of data files from a first computer (the transmitting computer) to a second computer (the receiving computer) including the steps of:

(i) the receiving computer sending a poll message to the transmitting computer asking for a list of files which are available for transfer;

(ii) the transmitting computer sending a reply message to the receiving computer stating that there will be a delay in providing a list of files available for transfer;

(iii) after a delay the transmitting computer sending a further message to the receiving computer stating that it has files available for transfer;

(iv) the receiving computer sending a poll message to the transmitting computer asking for a list of files which are available for transfer;

(v) the transmitting computer sending a reply message to the receiving computer containing a list of files which are available for transfer; and

(vi) the receiving computer selecting files from the list received from the transmitting computer and attempting to copy each selected file.

10. A method as claimed in any one of claims 7 to 9, including the step of the receiving computer sending a message to the transmitting computer acknowledging the files which have been copied successfully.

* C L M S *