(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) **World Intellectual Property Organization**
International Bureau

(43) **International Publication Date**
29 September 2011 (29.09.2011)

PCT

(10) **International Publication Number**
WO 2011/117921 A1

(51) **International Patent Classification**:
G06F 17/30 (2006.01)

(21) **International Application Number**:
PCT/JP2010/002103

(22) **International Filing Date**:
25 March 2010 (25.03.2010)

(25) **Filing Language**: English

(26) **Publication Language**: English

(71) **Applicant** *(for all designated States except US)*: **HITACHI, LTD.** [JP/JP]; 6-6, Marunouchi 1-chome, Chiyoda-ku, Tokyo, 1008280 (JP).

(72) **Inventors; and**
(75) **Inventors/Applicants** *(for US only)*: **KANO, Yoshiki** [JP/JP]; c/o Hitachi, Ltd., Systems Development Laboratory, 1099, Ohzenji, Asao-ku, Kawasaki-shi, Kanagawa, 2150013 (JP). **UTSUNOMIYA, Naoki** [JP/JP]; c/o Hitachi, Ltd., Systems Development Laboratory, 1099, Ohzenji, Asao-ku, Kawasaki-shi, Kanagawa, 2150013 (JP). **IWASAKI, Masaaki** [JP/JP]; c/o Hitachi, Ltd., Systems Development Laboratory, 1099, Ohzenji, Asao-ku, Kawasaki-shi, Kanagawa, 2150013 (JP). **TAKAOKA, Nobumitsu** [JP/JP]; c/o Hitachi, Ltd., Systems Development Laboratory, 1099, Ohzenji, Asao-ku, Kawasaki-shi, Kanagawa, 2150013 (JP). **NAKAMURA, Takaki** [JP/JP]; c/o Hitachi, Ltd., Systems Development Laboratory, 1099, Ohzenji, Asao-ku, Kawasaki-shi, Kanagawa, 2150013 (JP). **TAGUCHI, Yuichi** [JP/JP]; c/o Hitachi, Ltd., Systems Development Laboratory, 1099, Ohzenji, Asao-ku, Kawasaki-shi, Kanagawa, 2150013 (JP).

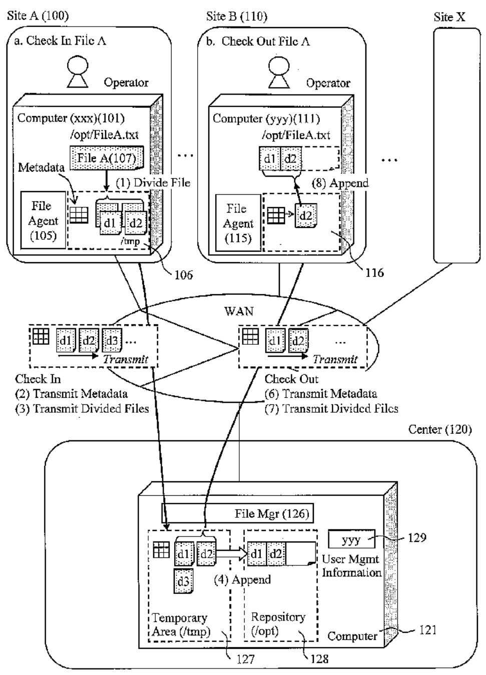
(74) **Agent**: **HIRAKI, Yusuke**; Kamiya-cho MT Bldg. 19F, 3-20, Toranomon 4-chome, Minato-ku, Tokyo, 1050001 (JP).

(81) **Designated States** *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PE, PG, PH, PL, PT, RO, RS, RU, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

[Continued on next page]

(54) **Title**: METHOD FOR CONCURRENCY CONTROL IN A FILE VERSIONING SYSTEM

[Fig. 4]



(57) **Abstract**: A file, for which writing has been reserved, in a file-sharing system can be efficiently read. A check-in file agent is provided in a computer that executes check-in of a file as the file-write reserving processing, a file manager is provided in a file-sharing computer, and a check-out file agent is provided in a computer that executes check-out of a file as the file reading processing. Upon request for check-in of a file, the check-in file agent divides the file into divided files and generates metadata that indicates the order of division of the file, and transmits them to the file-sharing computer for storage therein. Upon request for check-out of the file, the check-out file agent receives the metadata from the file-sharing computer, and based on the metadata, sequentially receives the divided files to generate a file to be checked out.

(84) **Designated States** *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:

— *with international search report (Art. 21(3))*

Description

Title of Invention: METHOD FOR CONCURRENCY CONTROL IN A FILE VERSIONING SYSTEM

Technical Field

[0001] The present invention relates to a file-sharing system and a method for managing files, and a program. For example, the invention relates to processing performed when, during check-in processing of a file, a check-out request for the same file is issued.

Background Art

[0002] Using a file-sharing system allows a plurality of users to edit and update files from document management software or file version management software. Such a file-sharing system places a write lock on a file in order to prevent a plurality of users from simultaneously updating the same file (see Fig. 1), by executing file-write reserving processing which is called "check-in" processing in which information to the effect that file writing processing is to be executed is provided. When a file is checked in, read access to the file is prohibited, that is, the file is read-locked until the completion of the writing in order to prevent other users from acquiring incomplete information. Therefore, while a given user is checking in a file, other users cannot check out (i.e., read) the file.

[0003] In Fig. 1, a computer 121 at a center 120 has a file manager (File Mgr) and is configured to process check-in/check-out requests. A computer 101 or 111 at each site has a file agent that transfers files through check-in or check-out processing by communicating with the computer at the center. In Fig. 1, the computer 101 at a site A performs check-in processing, and the computer 111 at a site B performs check-out processing. Though not shown, each computer is loaded with an operating system (OS) that is operative to perform storage or management of files, execution of programs, and the like in order to execute the file agent or the file manager.

[0004] Such a conventional file-sharing system is based on the assumption that it is executed over a LAN with a band as wide as 100 Mbps, for example. Thus, the time in which other users must wait for a read-locked file while the file is checked in has not been a big problem except under special circumstances such as when the access is concentrated or when the file size is too large.

Citation List

Patent Literature

[0005] PTL 1: JP Patent Publication (Kokai) No. 7-28679 A (1995)

Summary of Invention

Technical Problem

[0006] However, when files are shared over a narrow-bandwidth network typified by the Internet, the wait time for a user who requests for check-out of a file, in particular, has become longer. Such a problem will be described in more detail with reference to a sequence figure (Fig. 2) of the computer 101 at the site A and the computer 121 at the center. In a sequence 2001, the computer 101 at the site A transmits a check-in request to the computer 121 at the center. In a sequence 2002, the computer at the center, after generating a file to be stored therein and acquiring a read/write lock, acknowledges the receipt of the request by returning a response to the computer at the site A. In a sequence 2003, the computer at the site A transmits file data. In a sequence 2004 after the completion of the file transmission, the computer at the center releases the lock and informs the site A of the completion of the file transmission. Meanwhile, if the computer 111 at the site B requests for check-out (2005) while a lock is placed in sequence 2001 through sequence 2004, errors occur (2006) as the read/write lock has already been acquired, and thus the check-out processing cannot be performed. After sequence 2004, the read/write lock is released, upon which the relevant file becomes accessible through check-out processing (2007). As described above, a user who requests for check-out of a file must wait for a long time due to the other user's check-in processing of the same file.

[0007] Further, with regard to files that are shared with file-sharing systems, there have been an increasing number of files that are megabytes to gigabytes in size for use in CAD, CAM, large-scale source code, or virtual machines, in addition to conventional files that are several kilobytes in size for use in offices. With the increased size of the files, the wait time also tends to become longer. The present invention has been made in view of the foregoing circumstances, and provides a technique that allows files to be transferred in response to a check-out request for a file even when check-in processing of the same file, which would require a long transmission time, is being executed.

Solution to Problem

[0008] According to the present invention, in order to solve the aforementioned problem, a target file to be checked in is divided into a plurality of divided files, and a server that requests for check-in of the target file is configured to transfer the plurality of divided files, together with metadata thereon, to a center. Then, the divided files are transferred to a server that requests for check-out of the file, based on the metadata.

[0009] That is, in the file-sharing system in accordance with the present invention, a first computer (a computer 101 at a site A), when executing file-write reserving processing (check-in processing), divides a target file to be transferred to the center into a plurality of divided files, and transmits the plurality of divided files to the center. A third computer (a computer 121 at the center) receives the plurality of divided files for storage therein. A second computer (a computer 111 at a site B), when executing file reading processing (check-out processing) of the target file during the execution of the file-write reserving processing, acquires available divided files from among the plurality of divided files, and combines the acquired divided files to generate the target file.

[0010] The first computer generates a plurality of pieces of metadata for identifying each of the plurality of divided files and appropriately combining the plurality of divided files, and transmits the plurality of pieces of metadata to the third computer at the center before transmitting the plurality of divided files. The second computer, before acquiring the divided files, acquires the plurality of pieces of metadata, and, based on the metadata, acquires corresponding divided files.

[0011] Further, the second computer, in executing the file reading processing of the target file, determines from which server (an acquisition target server) including the first computer (the server (the computer 101) at the site A) and the third computer (the computer 121 at the center) the plurality of divided files are to be acquired, and, based on the plurality of pieces of metadata, requests the acquisition target server to transmit the plurality of divided files. In this case, the second computer checks the line conditions of the network and determines, based on the check result, a server with the best line conditions to be the acquisition target server. The line conditions of the network can be checked using any one of the network response comparison processing, network hop count comparison processing, and load comparison processing for the computer at each server.

[0012] The file-sharing system may further include, separately from the third computer (the computer 121 at the center), an auxiliary central server (a computer 131 at a center B) for storing files. In such a case, the third computer (121) transmits copies of the plurality of divided files to the auxiliary central server. The second computer acquires the plurality of pieces of metadata from the third computer, but acquires divided files corresponding to the acquired plurality of pieces of metadata from the auxiliary central server.

[0013] The second computer, upon request for access to part of the target file from an application, identifies a divided file that is needed to be the access based on the metadata, and acquires the identified divided file from the third computer. At this time, the third computer, if the identified divided file has not been received yet from the first server, acquires the relevant divided file from the first computer and then transmits the acquired divided file to the second computer.

[0014] Further features of the present invention will become apparent from the following best mode for carrying out the invention and the accompanying drawings.

Advantageous Effects of Invention

[0015] According to the present invention, files can be efficiently transferred in response to a check-out request for a file even when the file is being checked in.

Brief Description of Drawings

[0016] [fig.1]Fig. 1 is a diagram showing the logical configuration of a conventional file-sharing system.

[fig.2]Fig. 2 is a diagram for describing the processing sequence of each computer of a conventional file-sharing system.

[fig.3]Fig. 3 is a diagram showing the physical configuration of a file-sharing system in accordance with the present invention.

[fig.4]Fig. 4 is a diagram showing the logical configuration of a file-sharing system in accordance with the first embodiment of the present invention.

[fig.5]Fig. 5 is a diagram showing the structure of directories in accordance with the present invention.

[fig.6]Fig. 6 is a diagram showing the structure of metadata in accordance with the present invention.

[fig.7]Fig. 7 is a diagram for describing the processing sequence of check-in/check-out processing performed between computers in accordance with the first embodiment.

[fig.8]Fig. 8 is a flowchart for describing the processing of generating divided files based on a check-in instruction issued by an operator 102 of a file agent 105 in accordance with the first embodiment.

[fig.9]Fig. 9 is a flowchart for describing the file transfer processing of the file agent 105 in executing check-in.

[fig.10]Fig. 10 is a flowchart for describing the processing performed by a file manager 126 of a computer 121 at a center upon receipt of a check-in request in accordance with the first embodiment.

[fig.11]Fig. 11 is a flowchart for describing the processing of checking out a file that is being checked in in accordance with the first embodiment.

[fig.12]Fig. 12 is a flowchart for describing the processing performed by the file manager 126 at the center upon receipt of a check-out instruction from a file agent 115 in accordance with the first embodiment.

[fig.13]Fig. 13 is a diagram showing the logical configuration of a file-sharing system in accordance with the second embodiment of the present invention.

[fig.14]Fig. 14 is a diagram for describing the processing sequence in accordance with the second embodiment.

[fig.15]Fig. 15 is a diagram showing a network management table in accordance with the second embodiment.

[fig.16]Fig. 16 is a flowchart for describing the processing of checking out a file that is being checked in in accordance with the second embodiment.
[fig.17]Fig. 17 is diagram showing the logical configuration of a file-sharing system in accordance with the third embodiment of the present invention.
[fig.18]Fig. 18 is a flowchart for describing the processing performed by the file manager of the computer 121 in response to a check-in request in accordance with the third embodiment.
[fig.19]Fig. 19 is diagram showing the logical configuration of a file-sharing system in accordance with the fourth embodiment of the present invention.
[fig.20]Fig. 20 is a sequence management table for each file in accordance with the fourth embodiment.
[fig.21]Fig. 21 is a flowchart for describing the check-out processing in accordance with the fourth embodiment.
[fig.22]Fig. 22 is a flowchart for describing the processing performed by the file agent 115 in response to an I/O (read/write) request in accordance with the fourth embodiment.
[fig.23]Fig. 23 is a flowchart for describing the processing performed by the file manager 126 at the center upon receipt of a file acquisition request in accordance with the fourth embodiment.
[fig.24]Fig. 24 is a flowchart for describing the processing performed by the file agent 105 on the check-in side upon receipt of a file acquisition request in accordance with the fourth embodiment.
[fig.25]Fig. 25 is a flowchart for describing the check-in processing of the file agent 105 in accordance with the fourth embodiment.

Description of Embodiments

[0017] Hereinafter, embodiments of the present invention will be described with reference to the accompanying drawings. It should be noted that this embodiment only illustrates examples for implementing the present invention and thus is not to be construed as limiting the technical scope of the present invention. Structures that are common throughout the drawings will be assigned the same reference numerals.

Embodiment 1

[0018] (1) The First Embodiment

In order to archivecheck-out processing of a file that is being checked in, metadata is prepared for a target file to be checked in (divided files), and the divided files are transferred in accordance with the metadata during the check-out processing.

<Physical Configuration of File-Sharing System>

Fig. 3 is a diagram illustrating the schematic physical configuration of a file-sharing system in accordance with each embodiment of the present invention. The file-sharing system includes one or more centers 120 and one or more sites (a site A 100, a site B 110,..., and a site X). Each center and each site are connected via a wide area network (WAN) 130 typified by the Internet.

[0019] Each center includes a computer 121, a router 141, and a LAN (Local Area Network) 142. Examples of wide area networks include, but are not limited to, networks that can use protocols such as IP (Internet Protocol), ATM (Asynchronous Transfer Mode), Ethernet (Registered Trademark), and frame relay. The router connects the LAN to the wide area network to perform communication when communication with external computers is necessary. The router also performs, upon receiving a communication request from the wide area network, data transfer with the computer within the center. Examples of routers include, but are not limited to, those that can use protocols such as IP, ATM, and Ethernet (registered trademark).

[0020] The LAN has a function of establishing communication between the router and the computer. Examples of LANs include, but are not limited to, those that can use protocols such as IP, ATM, and Ethernet (registered trademark).

[0021] The computer 121 at the center 120 includes an NIC (Network Interface Card) 125 that communicates with the LAN, a CPU (Central Processing Unit) 123, memory (Mem) 124, and a disk 122. It is assumed that the sizes of a router, a LAN, and a computer used at each site are substantially equal to or smaller than those at the center. Although the devices such as the router 104, the LAN 103, or the computer 101 may differ in size from those at the center, there are no functional differences between such devices. The computer also includes a keyboard that can be operated by an operator and a display. Accordingly, an operator can operate files through the computer.

<Logical Configuration of File-Sharing System>

Fig. 4 is a diagram showing the logical configuration of a file-sharing system in accordance with the first embodiment of the present invention.

[0022] The computers 101 and 111 at the sites respectively have installed thereon file agents 105 and 115 that are configured to transmit and receive files in response to check-in and check-out requests. The computer 120 at the center has installed thereon a file manager (File Mgr) 126 that is configured to transmit and receive files in response to a check-in or check-out request and store the files in a repository 128 provided within the disk 122. Each computer has installed thereon an execution environment such as a typical operating system (OS) that operates to archive the file agent or the file manager. However, detailed description of such an operating system will be omitted as it is not directly related to the present invention.

[0023] Each computer with the file agent installed thereon prepares a local file system, which is provided by the OS installed on the computer, and also prepares a temporary area (/tmp) 106 or 116 on the local file system to temporarily store divided files obtained from a file 107 for transfer of the files. The computer with the file manager installed thereon prepares a temporary area (/tmp) 127 for temporary storage of files and a repository 128 for long-term storage of files that are stored in execution of check-in processing, and also prepares user management information 129, based on which the presence/absence of access from (an)other computer(s) is determined. The overview of the check-in/check-out processing in accordance with this embodiment will be described with reference to a sequence figure of Fig. 7 and Fig. 4.

<Directories of Files>

Fig. 5 is a diagram showing the structure of directories in accordance with the present invention. In the present invention, each computer has a directory hierarchy shown in Fig. 5. The way in which directories are used differs depending on the computer 101 or 111 at each site or the computer 121 at the center. Thus, the processing of each computer will be described hereinafter. In the computer 101, a file that has been stored by a user for check-in purposes is placed in a /opt directory 501. The file agent 105 divides the file and the resulting divided files are stored in a temporary area (/tmp) 502. It should be noted that although /opt is used as a directory prepared for check-in purposes, other directory names can also be used.

[0024] The file manager 126 in the computer 121 receives the divided files transmitted from the computer 101 and stores them into the temporary area (/tmp) 127. Then, the file manager 126 combines the received divided files and stores them into the repository area (/opt) 128.

[0025] The file agent 115 in the computer 111 receives the divided files transmitted from the computer 121 and stores them into the temporary area (/tmp)116. Then, the file agent 115 combines the divided files and stores them as a file into a check-out directory area (in this embodiment, /opt).

<Structure of Metadata>

Fig. 6 is a diagram showing the structure of metadata in accordance with the present invention. Metadata 600 is data generated for, when the computer at the site A divides a file into a plurality of divided files with the file agent, each divided file, and is used to identify each divided file that has been generated and reproduce the original file by combining the divided files. The metadata 600 includes a metadata file name 601, original file name 602, sequence number 603 used in connecting files, last sequence number 607, and divided-file name 604. It should be noted that a storage server name 605 for storing data, and file size 606 are not used in this embodiment. Thus, detailed description thereof will be omitted.

[0026] For the metadata file name 601 of the metadata 600, a file name "met" is used as the extension of the divided-file name. The original file name 602 indicates the name of a

file before divided. For the sequence number 603, the number of each divided file generated by the file agent 105 as described below is used. For the divided-file name 604, a file name composed of the file name and the sequence number is used. It should be noted that the metadata file name 601 and the divided-file name 604 shown herein are only examples of this embodiment. Thus, such names can be decided by different methods as long as they can help to identify the file transmission order and the files.
<Sequence of the Check-in/Check-out Processing>
Fig. 7 is a diagram showing a sequence of the check-in/check-out processing performed between the computers in accordance with this embodiment. First, the sequence processing performed between the computer 101 at the site A and the computer 121 at the center when an operator of the site A checks in a file will be described.

[0027] Sequence 701: The file agent 105 at the site A 100, based on a file stored in the check-in directory area (/opt), generates divided files and metadata thereon in the temporary area (/tmp) 106 (which corresponds to the processing (1) in Fig. 4). The details of the processing of generating the divided files and metadata thereon will be described below with reference to Fig. 8.

[0028] Sequence 702: The file agent 105 informs the file manager 126 at the center 120 of the check-in request together with the file name.

[0029] Sequence 703: The file manager 126 generates a target file to be checked in in the repository (/opt) 128. At this time, the file manager 126 acquires a write lock to prevent (an)other server(s) from simultaneously writing data to the file.

[0030] Sequence 704: The file manager 126 acknowledges (ACK) the receipt of the request by returning a response to the file agent 105 of the computer 101.
Sequence 705: The file agent 105 at the site A sequentially reads metadata on the divided files to be transmitted, and transmits all of the metadata to the file manager 126 at the center 120 (which corresponds to the processing (2) in Fig. 4).

[0031] Sequence 706: The file agent 105 reads the divided files to be transmitted in the order of the sequence number, and transmits them to the file manager 126 at the center 120 (which corresponds to the processing (3) in Fig. 4).

[0032] Sequence 707: The file manager 126 stores the divided files that have been transmitted into the temporary area (/tmp) 127. Then, the file manager 126, upon completion of the reception of the divide files, checks the sequence number described in the metadata on each divided file, and appends the divided files to the file stored in the repository (/opt) 128 in accordance with the sequence numbers (which corresponds to the processing of (4) in Fig. 4).

[0033] Sequence 708: After sequences 706 and 707 are repeated, the file manager 126 checks from the last sequence number 607 of the metadata that all of the divided files have been transmitted, and then releases the write lock.

[0034] Sequence 709: The file manager 126 informs the file agent 105 at the site A of the completion of the file transmission.

[0035] Next, the sequence processing performed between the computer at the site B 110 and the computer 121 at the center when an operator of the site B 110 checks out the file (which is the target file to be checked in) via the computer 111 will be described.

[0036] Sequence 711: The file agent 115 of the computer 111 at the site B 110 informs the file manager 126 of the computer 121 at the center of the file name of the target file to be checked out.

[0037] Sequence 712: The file manager 126 transmits all metadata (metadata on all divided files) to the file agent 115 at the site B 110 (which corresponds to the processing (6) in Fig. 4).

[0038] Sequence 713: The file agent 115, based on the received metadata, requests for the divided files to the file manager 126 at the center 120.

[0039] Sequence 714: The file manager 126 at the center 120 transmits the divided files to the file agent 115 at the site B 110 (which corresponds to the processing (7) in Fig. 4).

[0040] Sequence 715: The file agent 115 stores the divided files that have been transmitted into the temporary area 116, and appends, upon completion of the transmission of the divided files, the divided files to the target file to be checked out (which corresponds to the processing (8) in Fig. 4). It should be noted that the file agent 115 generates a file at the time of the initial appending.

[0041] Sequence 716: The file agent 115 at the site B 110 repeats sequence 713 through sequence 715, and, upon checking from the last sequence number 607 of the metadata that all of the divided files have been acquired, informs the file manager at the center of the completion of the acquisition of the divided files.

[0042] The check-in and check-out sequence processing in accordance with this embodiment is as described above. Dividing a target file into a plurality of divided files and processing them as described above makes it possible to immediately transmit the divided files, which have just been transferred through the check-in processing, to a site that is requesting for check-out of the same target file. Thus, the wait time for a site that requests for check-out can be shortened to the minimum.

<Processing of Generating Divided Files>

Fig. 8 is a flowchart for describing the processing of generating divided files based on a check-in instruction issued by an operator 102 of the file agent 105. This processing corresponds to the processing performed in sequence 701 of Fig. 7. Steps of the processing will be described below.

[0043] Step 801: The file agent 105 acquires the total size of the target file to be checked in (stored in /opt), and initializes the read block number (which corresponds to the current pointer) and the sequence number 603 that are used for generating divided files (sets the both values to #0, for example). In addition, the file agent 105 calculates the last sequence number 607 from the total file size and the size of each divided file (e.g., 64 MB) determined by the system.

[0044] Step 802: The file agent 105 generates divided files. At this time, the name determined with reference to Fig. 6 is used for the name of each divided file.

[0045] Step 803: The file agent 105 copies data on the target file to be checked in, in the range of from the read block to the divided file size, into each divided file.

[0046] Step 804: The file agent 105 generates metadata on each divided file. At this time, the sequence number 603 and the divided-file name 604, which have been previously determined, are used as they are. With regard to the last sequence number, a value obtained by dividing the file size 606 by the size of each divided file is used.

[0047] Step 805: The file agent 105 increments the read block number by the size of the divided file, and also increments the sequence number by one.

[0048] Step 806: If the file agent 105 determines that the read block number is less than or equal to the number of blocks corresponding to the total file size, the flow proceeds to step 802, and if the file agent 105 determines that the read block number is greater than the number of the blocks corresponding to the total file size, the flow ends.

The check-in processing of the file agent 105 is as described above.

<File Transmission Processing in Executing Check-in>

Fig. 9 is a flowchart for describing the details of the processing of transmitting files by the file agent 105 in executing check-in (which corresponds to the processing performed by the file agent in sequences 702 through 706 in Fig. 7). Steps of the processing will be described below.

[0049] Step 901: The file agent 105 transmits a check-in request for a target file together with the file name thereof to the file manager 126 of the computer 121 at the center 120.

[0050] Step 902: The file agent 105 checks if the file manager 126 has acknowledged (ACK) the receipt of the request. If the answer to step 902 is Yes, the flow proceeds to step 904, and if the answer to step 902 is No, the flow proceeds to step 903.

[0051] Step 903: The file agent 105 stops the processing and waits for a predetermined period of time (for example several seconds).

[0052] Step 904: The file agent 105 transmits metadata on all divided files to the file manager 126 at the center 120.

[0053] Step 905: The file agent 105 reads metadata with the initial sequence number (e.g., the sequence number #0) from the temporary area (/tmp) 106.

[0054] Step 906: The file agent 105 reads the divided-file name described in the read metadata, and transmits the corresponding divided file to the file manager 126 at the center 120.

[0055] Step 907: The file agent 105 checks if the acquired metadata is metadata with the last sequence number. If the answer to step 907 is Yes, the flow ends, and if the answer to step 907 is No, the flow proceeds to step 908.

[0056] Step 908: The file agent 105 selects metadata with the next sequence number, and reads it. Then, the file agent 105 transmits a divided file corresponding to the metadata to the center 120 in the same manner as that described above.

[0057] The processing of transmitting files by the file agent 105 in executing check-in is as described above.

<Processing Performed at the Center upon Receipt of a Check-in Request>

Fig. 10 is a flowchart for describing the processing performed by the file manager 126 of the computer 121 at the center 120 upon receipt of a check-in request in accordance with the resent invention (which corresponds to the processing performed by the file manager in sequences 702 through 706 in Fig. 7). Steps of the processing will be described below.

[0058] Step 1001: The file manager 126, upon receiving a check-in request, generates a file in the repository directory 128 based on the received file name. At this time, the file manager 126 also acquires a write lock on the same file.

[0059] Step 1002: The file manager 126 returns ACK, which indicates that the computer 121 is prepared to receive divided files, to the file agent 105 of the computer 101 at the site A 100.

[0060] Step 1003: The file manager 126 receives metadata on all divided files from the file agent 105 on the computer 101 at the site A 100, and stores it into the temporary area (/tmp) 127.

Step 1004: The file manager 126 reads metadata with the initial sequence number (e.g., #0) on the target file to be checked in.

[0061] Step 1005: The file manager 126 determines if a divided file corresponding to the divided-file name 604 described in the metadata resides in the temporary area 127. If the answer to step 1005 is Yes, the flow proceeds to step 1007, and if the answer to step 1005 is No, the flow proceeds to step 1006.

[0062] Step 1006: The file manager 126 stops the processing and waits for a predetermined period of time (several seconds).

[0063] Step 1007: The file manager 126 reads the divided file corresponding to the acquired metadata.

[0064] Step 1008: The file manager 126 appends the read divided file to the file generated in the repository 128.

[0065] Step 1009: The file manager 126 determines if the sequence number of the currently read metadata is the last sequence number 607 described in the metadata. If the answer to step 1009 is Yes, the flow proceeds to step 1011, and if the answer to step 1009 is No, the flow proceeds to step 1010.

Step 1010: The file manager 126 reads metadata with the next sequence number.

[0066] Step 1011: The file manager 126 refers to the user access information 129 to check for access to the divided file. If the answer to step 1011 is Yes, the flow proceeds to step 1012, and if the answer to step 1011 is No, the flow proceeds to step 1013.

[0067] Step 1012: The file manager 126 stops the processing and waits for a predetermined period of time (several seconds).

[0068] Step 1013: The file manager 126 deletes the non-accessed metadata (metadata that is not checked out) and the corresponding divided files from the temporary area (/tmp)127.

[0069] The processing performed by the file manager 126 upon receipt of a check-in request is as described above.

<Processing of Checking out a File that is Being Checked in>

Fig. 11 is a flowchart for describing the processing in which an operator uses the file agent 115 at the site B 110 to check out a file which is being checked in from another site (the site A 100). Steps of the processing will be described below.

[0070] Step 1101: The file agent 115 generates a target file to be checked out in a check-out directory (/opt).

[0071] Step 1102: The file agent 115 transmits a check-out request together with a file name to the file manager 126 of the computer 121 at the center 120.

[0072] Step 1103: The file agent 115 receives metadata on all divided files from the file manager 126.

[0073] Step 1104: The file agent 115 selects a metadata file with the initial sequence number (in this embodiment, zero).

[0074] Step 1105: The file agent 115 acquires a divided file described in the selected metadata from the center 120.

[0075] Step 1106: The file agent 115 appends the divided file to the target file to be checked out, and deletes the appended divided file and the corresponding metadata file from the temporary area (/tmp) 116.

[0076] Step 1107: The file agent 115 checks for the presence of the next metadata file. If the answer to step S1107 is Yes, the flow ends, and if the answer to step 1107 is No, the flow proceeds to step 1108.

Step 1108: The file agent 115 reads a metadata file with the next sequence number.

[0077] The processing of checking out a file, which is being checked in from a different site, by an operator via the file agent 115 is as described above.

<Processing Performed at the Center upon Receipt of a Check-out Request>

Fig. 12 is a flowchart for describing the processing performed by the file manager

126 at the center 120 upon receipt of a check-out request from the file agent 115. The processing will be described below.

[0078] Step 1201: The file manager 126 at the center 120 stores the server (computer) name as the user access information 129.

Step 1202: The file manager 126 transmits to the file agent 115 at the site B 110 all metadata files for the target file to be checked out.

[0079] Step 1203: The file manager 126 determines (checks) if it has received information from the file agent 115 at the site B 110 to the effect that the file agent 115 has received all divided files corresponding to the metadata files (if sequence 716 in Fig. 7 is completed). If the answer to step 1203 is Yes, the flow proceeds to step 1206, and if the answer to step 1203 is No, the flow proceeds to step 1204.

[0080] Step 1204: The file manager 126 determines if it has received a request for acquisition of the divided files. If the answer to step 1204 is Yes, the flow proceeds to step 1205, and if not, the flow proceeds to step 1203.

[0081] Step 1205: The file manager 126 transmits files (divided files) based on the file names described in the request for acquisition of the divided files.

[0082] Step 1206: As the site B 110 has received all divided files, the file manager 126 deletes the server (computer) name from the user management information.

[0083] The processing executed by the file manager 126 upon receipt of a check-out instruction from the file agent 115 is as described above.

<Conclusion of the First Embodiment>

According to this embodiment, it is possible to check out (transmit) a file that is being checked in, whereby check-out time can be reduced than that of the conventional check-out processing that is performed after the check-in processing.

[0084] Further, in execution of the check-out processing, received data can be sequentially read as the received data can be sequentially used. Further, if an application is used that permits received data to be sequentially written, editing becomes also possible.

[0085] It should be noted that migration of the divided files from the temporary area 127 to the repository 128 to generate a combined (appended) file can be executed at any of the following timings: (i) collectively executed after the reception of all divided files, (ii) sequentially executed after the reception of each divided file, with each divided file being held in the temporary area 127 until the completion of the reception of all divided files, and (iii) sequentially executed after the reception of each divided file, with the divided files that have been combined (appended) and the metadata being sequentially deleted from the temporary area 127. In the case of (i) and (ii), if a check-out request is received after the generation of the combined file, the combined file is transmitted as the target file to be checked out to the source of the request. If a check-out request is received before the generation of the combined file, it is acceptable as long as the aforementioned processing of transferring divided files is executed as the metadata and all divided files reside in the temporary area 127. Meanwhile, in the case of (iii), if a check-out request is received after some, but not all, of the divided files have been combined, the files that have been combined in the repository 128 and metadata that remains in the temporary area 127 are transmitted together to the source of the request. Then, the computer at a site that is the source of the request (in this embodiment, the computer 111 at the site B 110) acquires the divided files based on the metadata.

[0086] As described above, according to the first embodiment, in execution of the check-in processing, a target file to be transferred is divided into a plurality of divided files, and metadata (identification data that can identify each divided file and includes data that allows the divided files to be appropriately combined later at the time of combining the divided files) corresponding to each divided file is produced. Transfer of each divided file is executed based on such metadata. Accordingly, it is possible to reduce the check-in wait time for a user who requests for check-out of a file on a file-sharing system that uses a wide area network (WAN).

[0087] In addition, as the temporary area for storing divided files and metadata is provided separately from the repository for storing combined files, management of files that are not yet combined and files that are combined can be carried out easily. Thus, it is possible to respond to check-out requests that occur at various timings.

Embodiment 2

[0088] (2) Second Embodiment

In the second embodiment, in the processing of checking out a file that is being checked in in accordance with the first embodiment, a network that is in the shortest network distance is searched for to execute file transfer, with a view to increasing the data transfer speed. Hereinafter, difference from the first embodiment will mainly be discussed. It should be noted that the physical configuration of this embodiment is similar to that of the first embodiment (Fig. 1). Hereinafter, changes made to some parts will be described. As used herein, the "network distance" means the number of routers (a command "trace route" is used) that exist between the center and the other sites, or the response time when ping (a program configured to transmit an ICMP packet and to request for a response from the transmission destination) is executed between the center and the other sites.

<Logical Configuration of the File-Sharing System>

Fig. 13 is a diagram showing the logical configuration of a file-sharing system in accordance with the second embodiment of the present invention. The logical configuration of this file-sharing system differs from that of the first embodiment in that, when a check-out request from the operator of the site B 110 is processed, if it is determined that the response from the site A is faster than the response from the center 120, the file agent 115, instead of acquiring divided files from the center 120, selects the file agent 105 of the computer 101 with a faster response using a network management table (Net Mgr(Management) Table) 119, so that file transfer is executed.

[0089] Thus, the processing of the file agent 115 of the computer 111 and the file manager 126 at the center 120 of this embodiment differ from those of the previous embodiment. Further, the network management table 119 is added to the computer 111 at the site B 110.

<Sequence of the Check-in/Check-out Processing>

Fig. 14 is a diagram for describing the sequence of the check-in/check-out processing in accordance with the second embodiment. This sequence differs from that in Fig. 7 in that sequences 1412, 1420, 1413, and 1414 are needed instead of sequences 712, 713, and 714. Hereinafter, only the processing of each sequence that has been changed from Fig. 7 will be described. The processing of the other sequences is the same as that of the first embodiment.

[0090] Sequence 1412: The file manager 126 at the center 120 transmits all metadata to the file agent 115 at the site B 110. At this time, the file manager 126 stores, as the additional items of the metadata, the server name (e.g., xxx.org) of the computer 101 at the site A 100 and the server name (e.g., aaa.org) of the computer 121 at the center as the storage server name 605 in Fig. 6 (which corresponds to the processing of (6) in Fig. 13).

[0091] Sequence 1420: The file agent 115 at the site B 110, based on the item: storage server name 605 of the metadata, executes ping to each server to measure the network line conditions, and creates a network management table such as the one shown in Fig. 15 which includes a server (computer) name 1501 and the response time 1502 to select a server with the shortest response time. In this embodiment, the response time is measured using ping to measure the network line conditions. However, it is also possible to use a hop count 1503 of the network (select a server with a small hop count), load 1504 on the CPU of each computer (select a server with a small load), or the like. Alternatively, it is also possible to determine the priority for acquisition of divided files based on a value obtained by weighing the response time, hop count, and load (corresponding to the performance of a server to be selected) (placing the highest priority on the load).

[0092] Sequence 1413: The file agent 115 at the site B 110 creates an URL (e.g., http://xxx.org/tmp/File A-1.div) for enabling access to files in the server selected in sequence 1420, and requests for divided files to the file agent 105 on the computer 101 at the site A.

[0093] Sequence 1414: The file agent 105 transmits the divided files to the file agent 115 at the site B 110 via the WAN 130 (which corresponds to the processing (7) in Fig. 13).

[0094] It should be noted that if any site (server) that has already checked out the relevant file and holds it exists within a short network distance, other than the site A that is executing check-in or the center 120, the relevant file can be received from such site.

<Processing of Checking out a File that is being Checked in>

Fig. 16 is a flowchart for describing the processing of checking out a file that is being checked in in accordance with this embodiment. Fig. 16 differs from Fig. 11 in that step 1105 is replaced with steps 1601 and 1602. Only the steps that have been changed will be described below.

[0095] Step 1601: The file agent 115 at the site B 110 checks the network line conditions and selects a server (including the center 120) with the shortest response time. In this embodiment, an appropriate server is selected by checking which of the site 100 that is executing check-in and the center 120 is located within the shortest network distance (herein, the shortest response time). Such processing is the same as the processing performed in the aforementioned sequence 1420.

[0096] Step 1602: The file agent 115 at the site B 110 acquires from the server selected in step 1601 divided files described in the acquired metadata. Such processing is the same as the processing performed in the aforementioned sequence 1413.

[0097] The file agent 105 of the computer 101 at the site A 100 performs, in addition to the functions described in the first embodiment, transmission of divided files in response to a file request (http) from the file manager 126 of the computer 121 at the center 120. In this case, the file agent 115 at the site B 110 requests for access to the divided files stored in the temporary area 106 at the site A 100. In this embodiment, authentication is not executed upon request for files by the file agent 115. However, authentication with https(Hypertext Transfer Protocol over Secure Socket Layer), SSL,(Secure Socket Layer) and the like can be performed for security purposes.

[0098] It should be noted that the check-in processing performed by the file agent 105 at the site A 100 and the processing performed by the file manager 126 at the center 120 upon receipt of check-in and check-out requests are the same as the processing described in the first embodiment. However, in the second embodiment, step 1203 to step 1206, in which divided files are transmitted, of the processing in Fig. 12 are not executed, but instead, only metadata is transferred.

<Conclusion of the Second Embodiment>

According to this embodiment, it is possible to achieve, in addition to the advantageous effects of the first embodiment, data transfer from an appropriate computer in accordance with the line conditions and to achieve a reduction in the network bandwidth on the center side by transferring data between the sites.

[0099] Further, in the file-sharing system, divided files are received from a server having the original checked-in data. Thus, it is also possible to reduce the network bandwidth of the wide area network used by the file-sharing system because the transfers among servers is distributed, not consolidated.

[0100] It should be noted that when check-out requests are issued in parallel from a plurality of sites at different moments in time, for example, computers at all of the sites need not acquire desired divided files from a single server. That is, such desired divided files can be acquired from a computer with the best line conditions which is selected from among a computer that requested for check-in (e.g., the computer at the site A), a computer at the center, and a computer at another site that is executing check-out (e.g., the computer at the site X). Accordingly, load on each server can further be reduced.

Embodiment 3

[0101] (3) Third Embodiment

The third embodiment has, in addition to or separately from the feature of the second embodiment, the following feature: copies of the divided files held on the computer 121 at the center 120 are transferred to and stored into another center (a center B 130), whereby load distribution of the performance of the center 120 is achieved and the transfer time in check-out is reduced. Hereinafter, difference from the second embodiment will mainly be discussed based on the assumption that the aforementioned feature is provided in addition to the feature of the second embodiment. It should be noted that the physical configuration of the file-sharing system is similar to that shown in Fig. 1.

<Logical Configuration of the File-Sharing System>

Fig. 17 is a diagram showing the logical configuration of a file-sharing system in accordance with the third embodiment of the present invention. Fig. 17 firstly differs from Fig. 13 in that a computer 131 at another center (a center B 130 in the drawing) that performs load distribution of the center A 120 is prepared. The computer 131 has about the same physical configuration as the computer 121 at the center A 120, and is connected to the center A 100 via a router and a LAN. A brief summary of the operation of the computer 131 is as follows: at the same time as the check-in processing performed by the computer 101 at the center A 100, divided files 140 stored in the temporary area 127 are transferred to and stored into a temporary area (/tmp) 137 by executing file transmission processing via a http protocol to the computer 131 at the center B 130. In addition, bbb.org (a server name representing the computer 131 at the center B 130) is added as a new server for the served server name in the metadata. The computer 111 at the site B 110 that has received the metadata with the extension checks the network conditions (e.g., checks the load on the network using a ping command) including the added computer 131, and then divided files are transferred.

[0102] Such processing can also be performed while switching the source, from which divided files are acquired, between the center A 120 and the center B 130.

<Processing performed at the Center A upon Receipt of a Check-in Request>

Fig. 18 is a flowchart for describing the processing performed by the file manager 126 of the computer 121 at the center A 120 upon receipt of a check-in request from the computer 101 at the site A 100 in accordance with the third embodiment of the present invention. Fig. 18 is basically based on Fig. 10. Thus, only the changed points will be described below.

[0103] Step 1809: The file manager 126 stores the divided files into the temporary area (/tmp) 137 of the computer 130 through the put processing with http. At this time, the file manager 126 adds a server name (e.g., bbb.org) of the computer (the computer 131 at the center B 130) in which the divided files have been newly stored, as the entry 605 of the served server (the storage server) in the metadata.

[0104] Step 1813: The file manager 126 deletes the received metadata files and divided files from the temporary area (/tmp) 127 of the computer 121 in the same manner as in Fig. 10, and also deletes the divide files (copies) from the temporary area (/tmp) 137 of the computer 131 at the center B 130 (or the computer 131 deletes the copied divided files in response to a delete request from the file manager 126).

<Conclusion of the Third Embodiment>

According to this embodiment, copies of the divided files held by the computer 121 at the center A 120 are stored into another center (the center B 130), whereby it becomes possible to achieve, in addition to the advantageous effects of the second embodiment, a further reduction in the transfer time in check-out. In addition, load for server and network on the center A 120 can be reduced.

[0105] It should be noted that even when a configuration such as the one described in the second embodiment is not used, it is still possible to alleviate the load on the network only by storing copies of the divided files, which are stored on the computer 121 at the center A 120, into another center (the center B 130), whereby the advantageous effect that the transfer time in check-out can be reduced is expected to be achieved.

Embodiment 4

[0106] (4) Fourth Embodiment

According to the fourth embodiment, when an application in a host (e.g., the computer 111 at the site B 110) executing check-out processing has accessed a given address of a target file, a divided file that includes the accessed address is preferentially transferred. Hereinafter, difference from the first embodiment will mainly be discussed. It should be noted that the physical configuration of the file-sharing system of this embodiment is similar to that in Fig. 1.

<Logical Configuration of File-Sharing System>

Fig. 19 is a diagram showing the logical configuration of a file-sharing system in accordance with the fourth embodiment of the present invention. The configuration of Fig. 19 is basically the same as that in Fig. 4 but differs in that an application executes access to a desired file; an update history table 150 for managing an updated file area is further provided; and, though not shown in Fig. 19, a file sequence management table 5000 (see Fig. 20) for each file is further provided.

[0107] A brief summary of the operation of this configuration is as follows: the computer 111 at the site B 110 generates in advance a file 118 that is an empty file (which will be filled with data each time a divided file is acquired), and the file agent 115 monitors access from the application. When the file is accessed, if necessary data (divided files) have not been read in yet and are not stored in the temporary area (/tmp) 116 either, the file agent 115 acquires new files from the computer 121 at the center 120. For example, when a request to access divided files d1, d2,...is issued, if the temporary area 116 does not contain such files, the file agent 115 acquires such divided files from the center 120 using a GET command.

[0108] It should be noted that the entire complete file is not visible to an operator who uses the application as the application accesses only part of the file. If an application that requires part of a file to be read or written is used, such a request can be met.

<File Sequence Management Table>

Fig. 20 is a diagram showing a sequence management table for each file in accordance with the fourth embodiment of the present invention. This management table is stored in memory of the computer 111 and includes as an entry a file name (File) 5001, sequence number (Seq#) 5002, start address BLK(Start) 5003 and end address BLK(End) 5004 of each sequence number, and size 5006 per sequence. This table is generated when the initial metadata is read as described below.

<Processing of Checking out a File that is being Checked in>

Fig. 21 is a flowchart for describing the processing of checking out a file that is being checked in in accordance with the fourth embodiment of the present invention. The basic processing of the flowchart is the same as that in Fig. 11. However, the flowchart in Fig. 21 includes, in order to acquire a divided file with a sequence number that is included in an I/O (read/write) access request described below, an additional step of checking if, after the normal reception of a divided file, processing of receiving another divided file is being executed, whereby transmission of a file through I/O access described below is executed. Hereinafter, the additional steps will be described.

[0109] Step 2101: The file agent 115 generates the file 118 with stored therein "null" data based on the file size of the metadata. Although "null" is used herein as dummy data for the file until divided files acquired are stored therein, other values like zero, other sequential number based on pre-number and so on can also be used.

[0110] Step 2102: The file agent 115 copies data on the received divided file into an area determined by the start address BLK 5003 and the end address 5004 of the sequence number in the file-sequence table 5000, within an area of the generated file 118. Then, the file agent 115 deletes the copied divided file and the corresponding metadata from the temporary area 116.

[0111] Step 2103: The file agent 115 checks if the currently processed metadata on the target file to be checked out is the last metadata, that is, the file agent 115 checks for the presence of the next metadata file. If the answer to step 2103 is Yes, the flow ends, and if the answer to step 2103 is No, the flow proceeds to step 2104.

[0112] Step 2104: The file agent 115 checks if another divided file is being acquired through I/O processing (if "Get file" is executed for another divided file). If the answer to step 2104 is Yes, the flow proceeds to step 2105, and if the answer to step 2104 is No, the flow proceeds to step 1108. Determination of if another divided file is being transmitted can be conducted by checking if step 2205 or step 2206 is being executed.

[0113] Step 2105: The file agent 115 enters a standby mode until the completion of the current file transmission. Execution of the present processing takes precedence over the transmission of the next file.

<Processing of I/O Request from the Application>

Fig. 22 is a flowchart for describing the operation of the file agent 115 in processing an I/O (read/write) request from the application in accordance with this embodiment. The file agent 115 at the site B 110 monitors a file I/O request from the application, and performs, if there has been a request for access to an area of the sequence number, I/O access to the file after reading the file. Such processing will be described below.

[0114] Step 2201: The file agent 115 calculates the sequence number of the file from the access pointer of the file in the I/O request (I/O access pointer). Such a sequence number is obtained from the quotient of (the BLK number of the file access pointer)/(the size 2006 per sequence).

[0115] Step 2202: The file agent 115 refers to a metadata file group in the temporary area 116 to check for the presence of a metadata file with the relevant sequence number. This is because, if the metadata is present, it means that the corresponding divided file has not been received yet, and if the metadata is absent within last sequence number 200, it means that the corresponding divided file has already been received. If the answer to step 2202 is Yes, the flow proceeds to step 2203 based on the assumption that the corresponding divided file has not been received yet, and if the answer to step 2202 is No, the flow proceeds to step 2207 based on the assumption that the corresponding divided file has already been received.

[0116] Step 2203: The file agent 115 checks if another divided file is being transmitted through different processing. If the answer to step 2203 is Yes, the flow proceeds to step 2204, and if the answer to step 2203 is No, the flow proceeds to step 2205.

[0117] Step 2204: The file agent 115 waits until the completion of the transmission of the currently transmitted file. Execution of the present processing takes precedence over the transmission of the next file.

[0118] Step 2205: The file agent 115 reads metadata with the relevant sequence number, and acquires the corresponding divided file from the computer 121 at the center 120.

[0119] Step 2206: The file agent 115 copies data on the received divided file into an area determined by the start address BLK 5003 and the end address 5004 of the sequence number in the file-sequence table 5000, within an area of the file 118. Then, the file agent 115 deletes the copied divided file and the corresponding metadata from the temporary area 116.

[0120] Step 2207: The file agent 115 accesses the relevant area in response to an I/O request (for read/write processing).

Step 2208: The file agent 115 checks if the sum of the current I/O access pointer and the BLK size per sequence is less than the BLK size requested by the I/O. If the answer to step 2208 is Yes, the flow proceeds to step 2201, and if the answer to step 2208 is No, the flow ends. This is because the desired data of a file may reside across a plurality of divided files in some cases. In such cases, processing for acquiring further necessary divided files is executed.

<Processing Performed at the Center upon Receipt of a File Acquisition Request>

Fig. 23 is a flowchart for describing the processing performed by the file manager 126 at the center 120 upon receipt of a file acquisition request in accordance with this embodiment. Procedures of the processing will be described below.

[0121] Step 2301: The file manager 126 checks if a requested divided file is stored in the temporary area 127. If the answer to step 2301 is Yes, the flow proceeds to step 2303, and if the answer to step 2301 is No, the flow proceeds to step 2302.

[0122] Step 2302: The file manager 126 acquires the requested divided file from a file agent (e.g., the file agent 105) that is executing check-in (GET processing). Such processing corresponds to sequence (3) in Fig. 19.

[0123] Step 2303: The file manager 126 transmits the requested divided file to a file agent (e.g., the file agent 115) that is the source of the request.

<Processing Performed at the Site on the Check-in Side upon Receipt of a File Acquisition Request>

Fig. 24 is a flowchart for describing the processing performed by the file agent 105 on the check-in side upon receipt of a file acquisition request in accordance with the present invention. Procedures of the processing will be described below.

[0124] Step 2401: The file agent 105 checks if another divided file of a file, which is being checked in, is being transmitted. Determination of if another divided file is being transmitted can be conducted by checking if step 908 and step 906 are being executed. If the answer to step 2401 is Yes, the flow proceeds to step 2402, and if the answer to step 2401 is No, the flow proceeds to step 2403.

[0125] Step 2402: The file agent 105 enters a standby mode until the completion of the transmission of the currently transmitted another divided file. Execution of the present processing takes precedence over the transmission of the next file.

[0126] Step 2403: The file agent 105 transmits the requested divided file to the source of the request (herein, the center 120).

[0127] Step 2404: The file agent 105 deletes metadata on the transmitted divided file from the temporary area.

<Check-in Processing by File Agent>

Fig. 25 is a flowchart for describing a file transfer operation executed through the check-in processing of the file agent 105 in accordance with this embodiment. The basic operation is the same as that in Fig. 9. However, processing for interrupting file transmission is added. Hereinafter, difference from the processing in Fig. 9 will be described.

[0128] Step 2500: After the transmission of the divided file corresponding to the metadata (step 906), the file agent 105 checks if the divided file is the data with the last sequence number. If the answer to step 2500 is Yes, the flow ends, and if the answer to step 2500 is No, the flow proceeds to step 2501.

[0129] Step 2501: The file agent 105 checks if another divided file, which is subjected to GET processing, is being transmitted. Determination of if another divided file is being transmitted can be conducted by checking if step 2403 and step 2404 are being executed. If the answer to step 2501 is Yes, the flow proceeds to step 2502, and if the answer to step 2501 is No, the flow proceeds to step 908.

[0130] Step 2502: The file agent 105 enters a standby mode until the completion of the current file transmission. Execution of the present processing takes precedence over the transmission of the next file.

<Conclusion of the Fourth Embodiment>

In this embodiment, description has been made of a case in which the present access method is applied to the first embodiment. However, the present access method can also be applied to the second and third embodiments. In such cases, the processing of interrupting file transmission described in this embodiment needs to be performed in the check-in processing of the file agent 105 and the check-out processing of the file agent 115.

[0131] According to this embodiment, when an application in a computer that executes check-out requests for access to a given address of a file, a divided file that includes the accessed address can be preferentially transmitted, whereby check-out time seen by an operator can be reduced.

[0132] Further, in such a file-sharing system, it becomes possible to edit a file via an application even when the file is being transferred, by devising the transmission order of file data.

(5) Supplement

It should be noted that the present invention can also be realized by a program code of software that implements the function of the embodiments. In such a case, a storage medium having recorded thereon the program code is provided to a system or an apparatus, and a computer (or a CPU or a MPU) in the system or the apparatus reads the program code stored in the storage medium. In this case, the program code itself read from the storage medium implements the function of the aforementioned embodiments, and the program code itself and the storage medium having recorded thereon the program code constitute the present invention. As the storage medium for supplying such a program code, for example, a flexible disk, CD-ROM, DVD-ROM, a hard disk, an optical disc, a magneto-optical disc, a CD-R, a magnetic tape, a non-volatile memory card, ROM, or the like is used.

[0133] Further, based on an instruction of the program code, an OS (Operating System) running on the computer or the like may perform some or all of actual processes, and the function of the aforementioned embodiments may be implemented by those processes. Furthermore, after the program code read from the storage medium is written to the memory in the computer, the CPU or the like of the computer may, based on the instruction of the program code, perform some or all of the actual processes, and the function of the aforementioned embodiments may be implemented by those processes.

[0134] Moreover, the program code of the software that implements the function of the embodiments may be distributed via a network, and thereby stored in storage means such as the hard disk or the memory in the system or the apparatus, or the storage medium such as a CD-RW or the CD-R, and at the point of use, the computer (or the CPU or the MPU) in the system or the apparatus may read the program code stored in the storage means or the storage medium and execute the program code.

Reference Signs List

[0135] 100 Site A

101 Computer (Server) at Site A

102 Operator of Site A

103 LAN at Site A

104 Router at Site A

105 File agent

106 Temporary Area
107 File
110 Site B
111 Computer (Server) at Site B
115 File Agent
116 Temporary Area
120 Center
121 Computer at Center (Central Server)
122 Disk
123 CPU
124 Memory
125 NIC
141 Router
142 LAN
1501 Computer Name
1502 Response Time
1503 Hop Count
1504 Load
5001 File name
5002 Sequence Number
5003 Start Block
5004 End Block
5006 Size per Sequence

Claims

[Claim 1] A file-sharing system comprising:
a first server computer with a first computer (101);
a second server computer with a second computer (111); and
a central server computer with a third computer (121), wherein:
the first server computer, the second server computer, and the central server computer are connected via a network,
the first computer (101), when executing file-write reserving processing, divides a target file to be transferred to the central server computer into a plurality of divided files, and transmits the plurality of divided files to the central server computer,
the third computer (121) receives the plurality of divided files for storage therein, and
the second computer (111), when executing file reading processing of the target file during the execution of the file-write reserving processing, acquires available divided files from among the plurality of divided files, and combines the acquired divided files to generate the target file.

[Claim 2] The file-sharing system according to claim 1, wherein:
the first computer (101) generates the plurality of divided files and a plurality of pieces of metadata for identifying each of the plurality of divided files and appropriately combining the plurality of divided files, transmits a request to the third computer (121) to perform the file-write reserving processing, and, upon receiving an acknowledgment of receipt of the request from the third computer (121), collectively transmits the plurality of pieces of metadata and sequentially transmits the plurality of divided files to the third computer (121),
the third computer (121) places, upon receiving the request to perform the file-write reserving processing, a write lock on the target file, transmits, upon receiving a request to perform the file reading processing from the second computer (111) during the execution of the file-write reserving processing, the plurality of pieces of metadata to the second computer (111), transmits, in response to a request for acquisition of the divided files from the second computer (111) based on the metadata, the relevant divided files to the second computer (111), combines the plurality of divided files received from the first computer (101) at a predetermined timing to generate the target file, releases the

write lock, and informs the first computer (101) of the completion of the file writing, and
the second computer (111) combines the plurality of divided files acquired from the third computer (121) to generate the target file, and informs the third computer (121) of the completion of the file reading processing.

[Claim 3] The file-sharing system according to claim 1, wherein:
the first computer (101) generates a plurality of pieces of metadata for identifying each of the plurality of divided files and appropriately combining the plurality of divided files, and transmits the plurality of pieces of metadata to the third computer (121) before transmitting the plurality of divided files, and
the second computer (111) acquires the plurality of pieces of metadata before acquiring the divided files, and acquires, based on the metadata, corresponding divided files.

[Claim 4] The file-sharing system according to claim 3, wherein:
the third computer (121) includes a temporary area for temporarily storing data and a repository area for storing a complete file, and
the third computer (121) stores into the temporary area the received plurality of pieces of metadata and divided files, stores into the repository area a combined file that is obtained by combining the plurality of divided files based on the plurality of pieces of metadata, and finalizes the file-write reserving processing.

[Claim 5] The file-sharing system according to claim 3, wherein the second computer (111) combines the acquired divided files, and deletes the divided files that have been combined and the corresponding metadata.

[Claim 6] The file-sharing system according to claim 3, wherein the second computer (111), when executing the file reading processing of the target file, determines from which server computer (an acquisition target server computer) including the first server computer and the central server computer the plurality of divided files are to be acquired, and, based on the plurality of pieces of metadata, requests the acquisition target server computer to transmit the plurality of divided files.

[Claim 7] The file-sharing system according to claim 6, wherein the second computer (111) checks line conditions of the network, and determines, based on a check result, a server computer with the best line conditions to be the acquisition target server computer.

[Claim 8] The file-sharing system according to claim 7, wherein the second computer (111), in checking the line conditions of the network, uses one of network response comparison processing, network hop count comparison processing, and load comparison processing for the computer at each server computer.

[Claim 9] The file-sharing system according to claim 3, further comprising, separately from the central server computer, an auxiliary central server computer (131) for storing files, wherein:
the third computer (121) of the central server computer transmits copies of the plurality of divided files to the auxiliary central server computer (131), and
the second computer (111) acquires the plurality of pieces of metadata from the central server computer, and acquires, based on the acquired plurality of pieces of metadata, the plurality of divided files that constitute the target file from the auxiliary central server computer (131).

[Claim 10] The file-sharing system according to claim 3, wherein the second computer (111), upon request for access to part of the target file from an application, identifies a divided file that is needed to be the access based on the metadata, and acquires the identified divided file from the central server computer.

[Claim 11] The file-sharing system according to claim 10, wherein the third computer (121), if the identified divided file has not been received yet from the first server computer, requests the first computer (101) to transmit the divided file, and transmits, after having received the divided file, the divided file to the second computer (111).

[Claim 12] A method for managing files with a file-sharing system, the file-sharing system comprising a first server computer with a first computer (101), a second server computer with a second computer (111), and a central server computer with a third computer (121), all of the first server computer, the second server computer, and the central server computer being connected via a network, the method comprising:
causing the first computer (101) to, when executing file-write reserving processing, divide a target file to be transferred to the central server computer into a plurality of divided files, and to sequentially transmit the plurality of divided files to the central server computer,
causing the third computer (121) to receive the plurality of divided files for storage therein, and

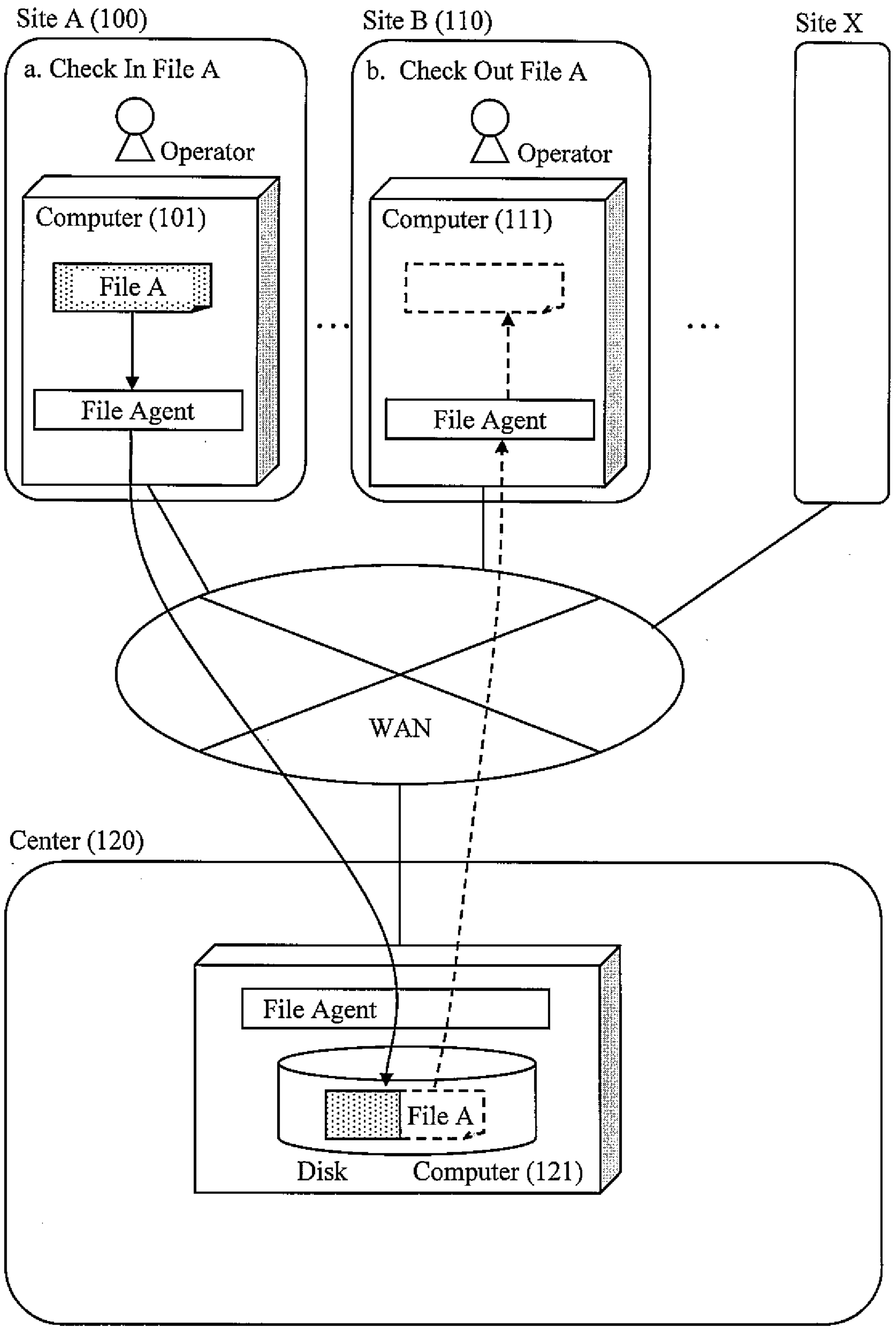
causing the second computer (111) to, when executing file reading processing of the target file during the execution of the file-write reserving processing, acquire available divided files from among the plurality of divided files and to combine the acquired divided files to generate the target file.

[Claim 13] The file managing method according to claim 12, wherein:
the first computer (101) generates the plurality of divided files and a plurality of pieces of metadata for identifying each of the plurality of divided files and appropriately combining the plurality of divided files, and then, transmits a request to the third computer (121) to perform the file-write reserving processing,
the third computer (121), in response to the request to perform the file-write reserving processing, returns an acknowledgment of receipt of the request to the first computer (101) and, in response to the request to perform the file-write reserving processing, places a write lock on the target file,
the first computer (101), upon receiving the acknowledgment of the receipt of the request, collectively transmits the plurality of pieces of metadata and sequentially transmits the plurality of divided files to the third computer (121),
the second computer (111), during the execution of the file-write reserving processing, transmits a request to the third computer (121) to perform the file reading processing,
the third computer (121), in response to the request to perform the file reading processing from the second computer (111), transmits the plurality of pieces of metadata to the second computer (111),
the second computer (111), based on the plurality of pieces of metadata, transmits a request for acquisition of divided files to the third computer (121),
the third computer (121), in response to the request for acquisition of the divided files, transmits the relevant divided files to the second computer (111),
the third computer (121) combines the plurality of divided files received from the first computer (101) at a predetermined timing to generate the target file, releases the write lock, and informs the first computer (101) of the completion of the file writing, and
the second computer (111) combines the plurality of divided files acquired from the third computer (121) to generate the target file, and

informs the third computer (121) of the completion of the file reading processing.

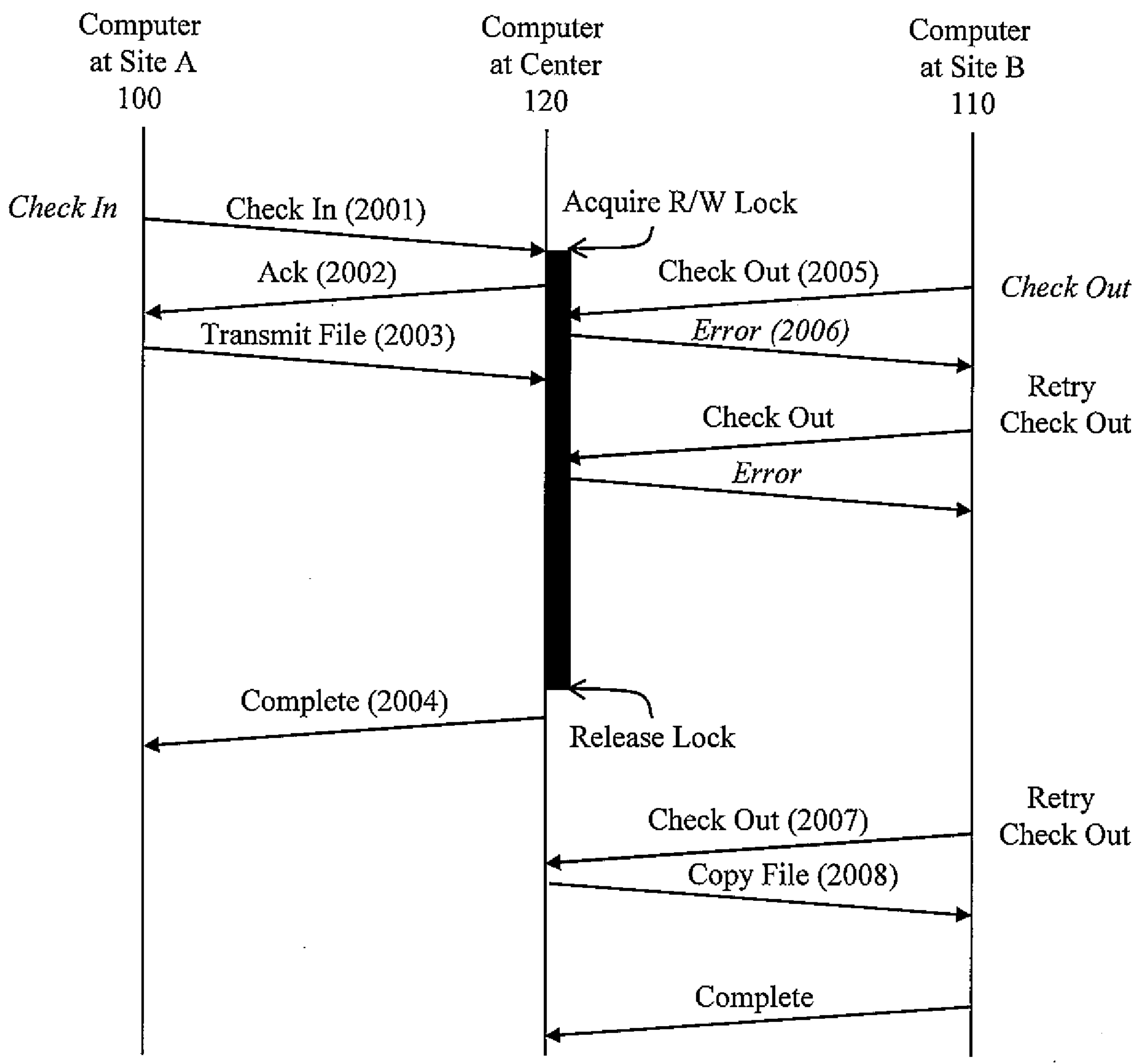
[Claim 14] A program for causing a plurality of computers to function as the first to third computers (101, 111, 121) of the file-sharing system according to claim 1.

[Fig. 1]



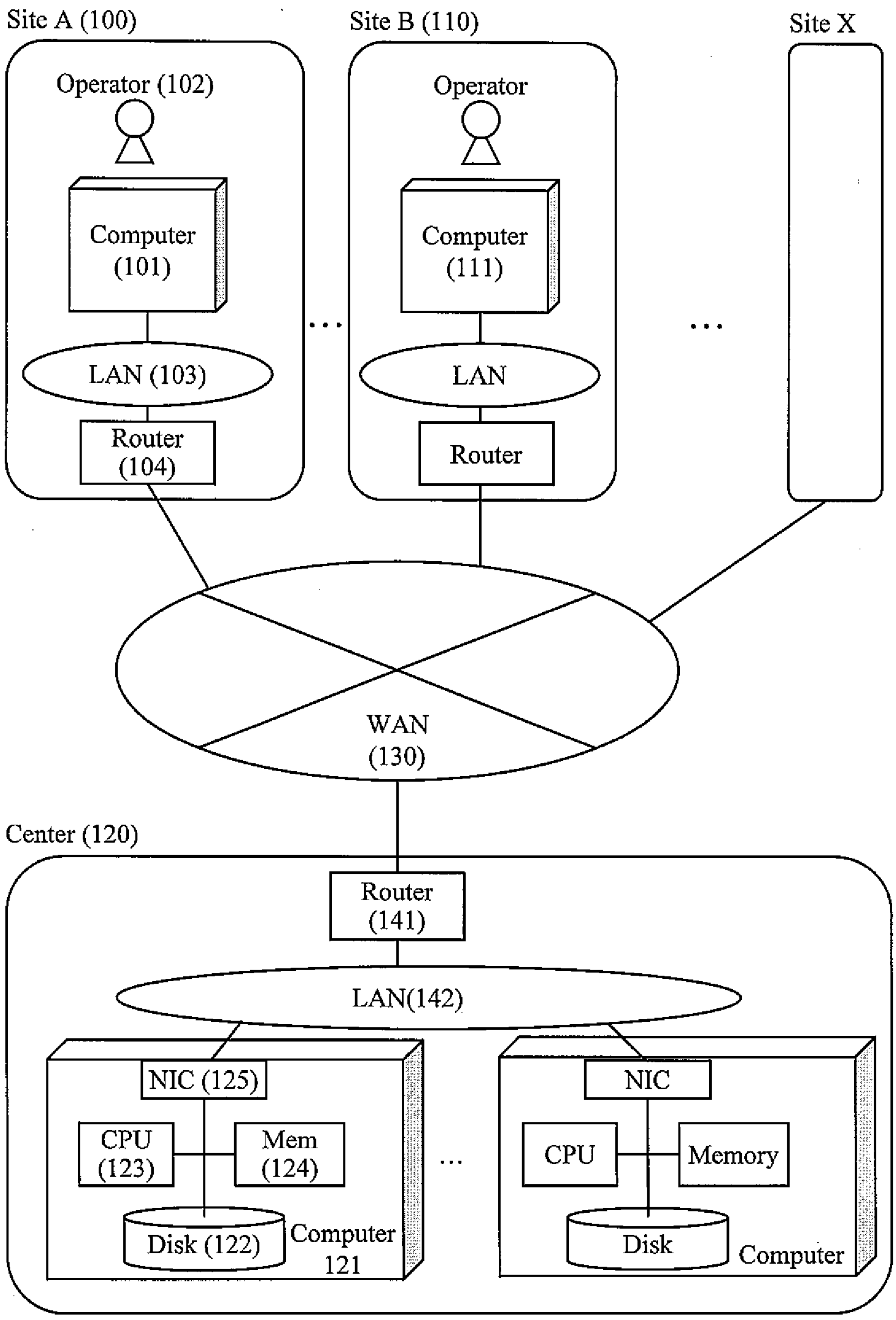
(Prior Art)

[Fig. 2]

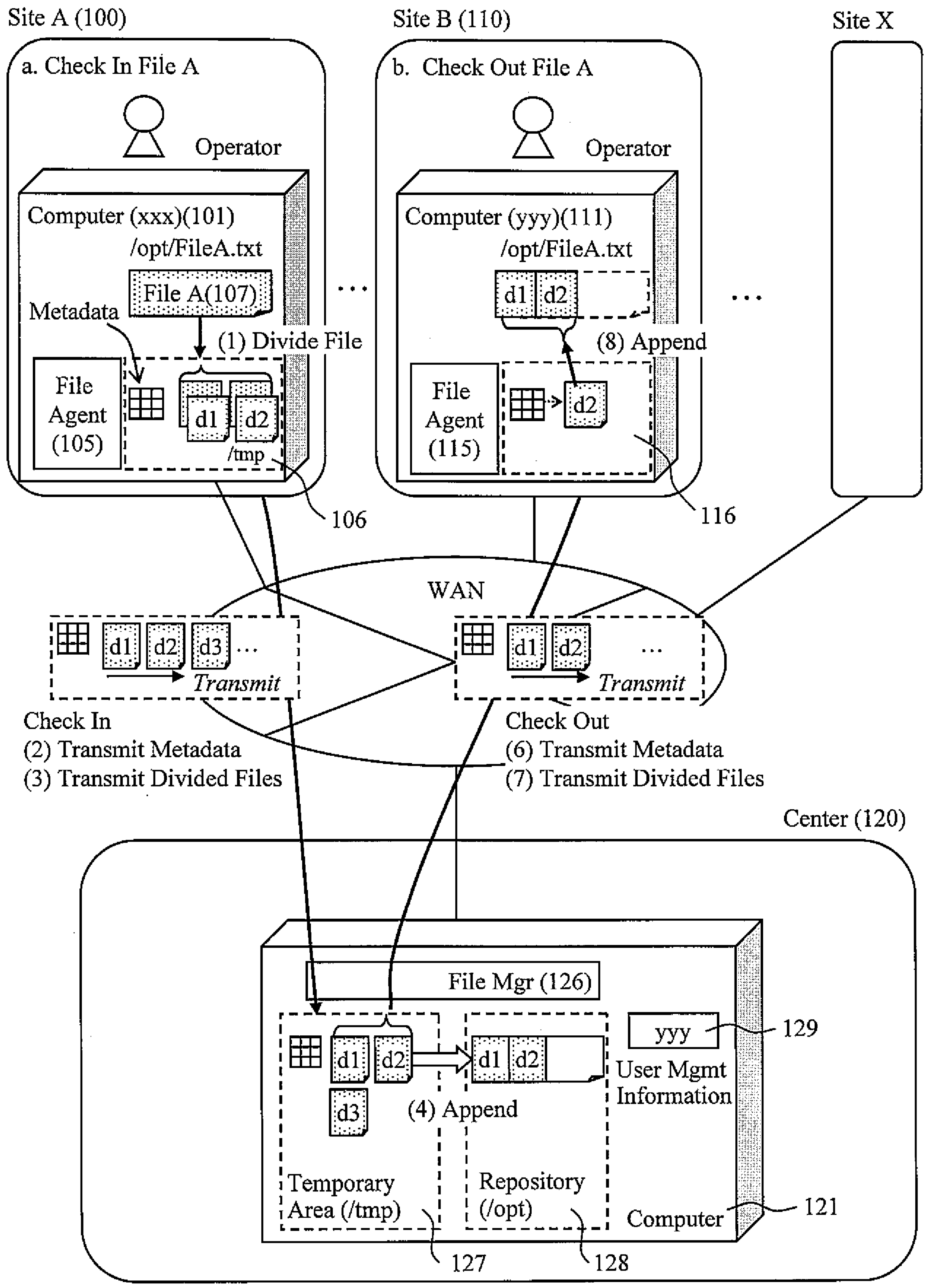


(Prior Art)

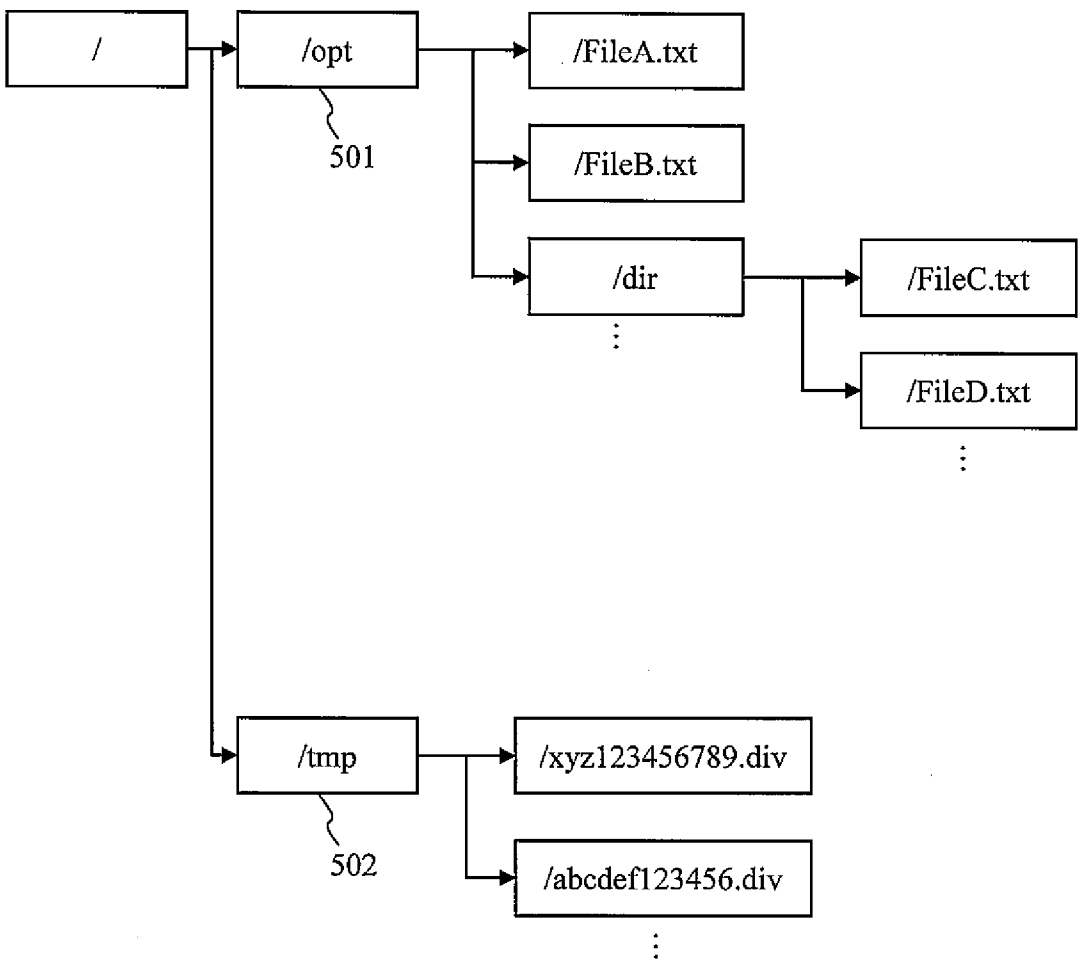
[Fig. 3]


Site A (100)
Operator (102)
Computer (101)
LAN (103)
Router (104)
Site B (110)
Operator
Computer (111)
LAN
Router
Site X
WAN (130)
Center (120)
Router (141)
LAN(142)
NIC (125)
CPU (123)
Mem (124)
Disk (122)
Computer 121
NIC
CPU
Memory
Disk
Computer

[Fig. 4]


Site A (100)
a. Check In File A
Operator
Computer (xxx)(101)
/opt/FileA.txt
File A(107)
Metadata
(1) Divide File
File Agent (105)
d1
d2
/tmp
106
Site B (110)
b. Check Out File A
Operator
Computer (yyy)(111)
/opt/FileA.txt
d1
d2
(8) Append
File Agent (115)
d2
116
Site X
WAN
d1 d2 d3 ...
Transmit
Check In
(2) Transmit Metadata
(3) Transmit Divided Files
d1 d2 ...
Transmit
Check Out
(6) Transmit Metadata
(7) Transmit Divided Files
Center (120)
File Mgr (126)
d1 d2 d3
(4) Append
d1 d2
Temporary Area (/tmp)
Repository (/opt)
yyy
User Mgmt Information
Computer
127
128
129
121

[Fig. 5]


/
/opt
501
/FileA.txt
/FileB.txt
/dir
/FileC.txt
/FileD.txt
/tmp
502
/xyz123456789.div
/abcdef123456.div

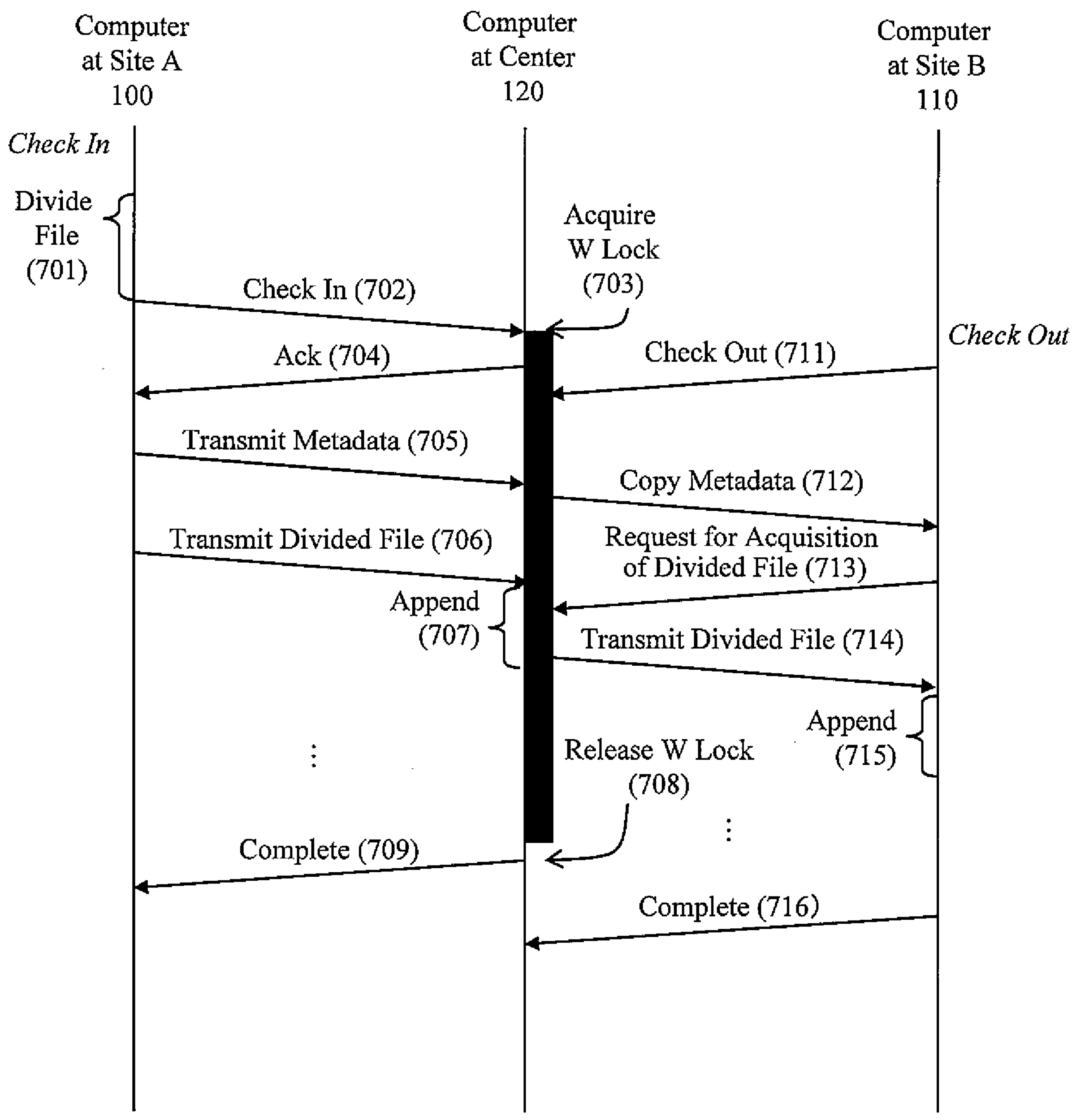
[Fig. 6]

600

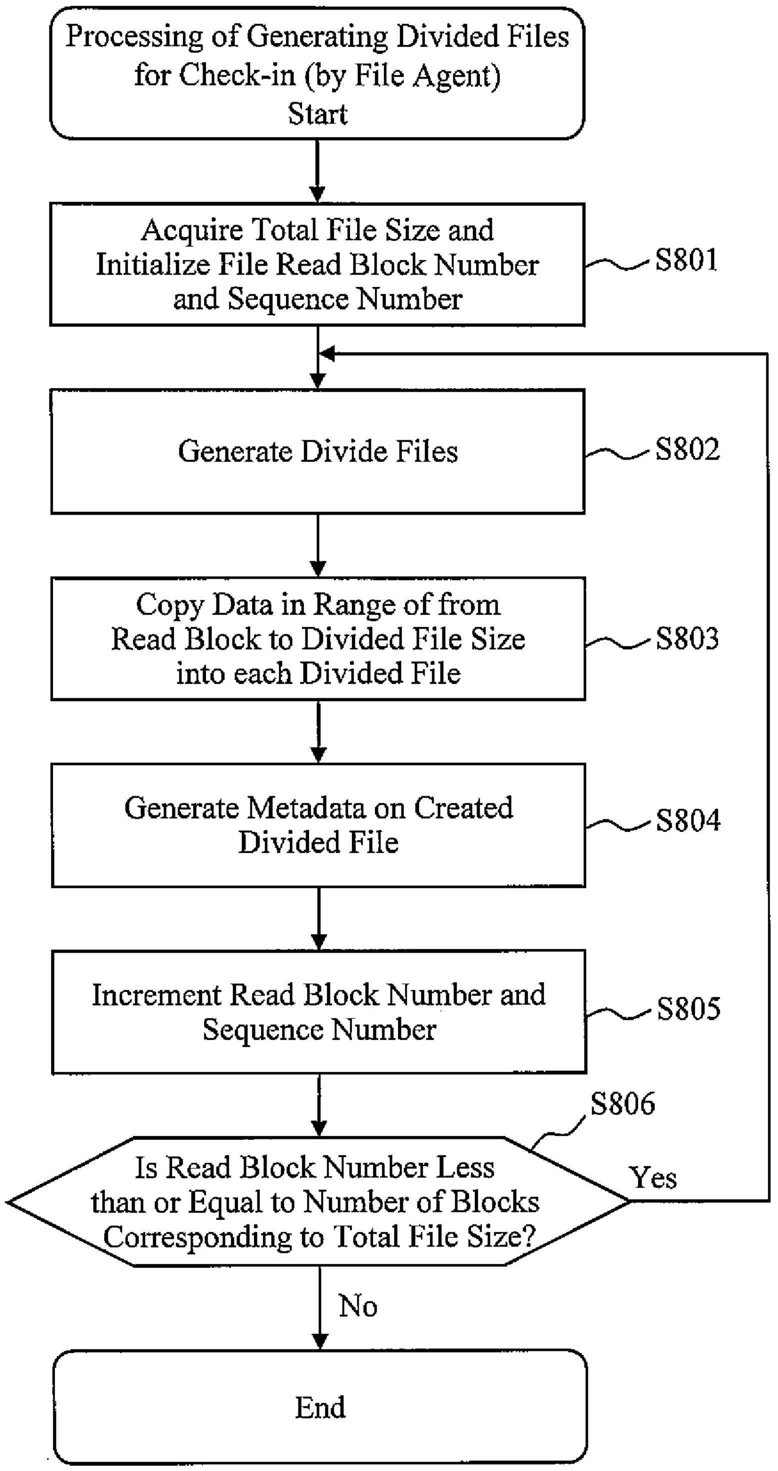
Metadata File Name (FileA-1.met) 601

Original File Name: FileA.txt 602
Sequence Number: #1 603
Last Sequence Number: #200 607
Divided-File Name: FileA-1.div 604
Storage Server: xxx.org, zzz.org, aaa.org, bbb.org 605
File Size: 10737418240 606

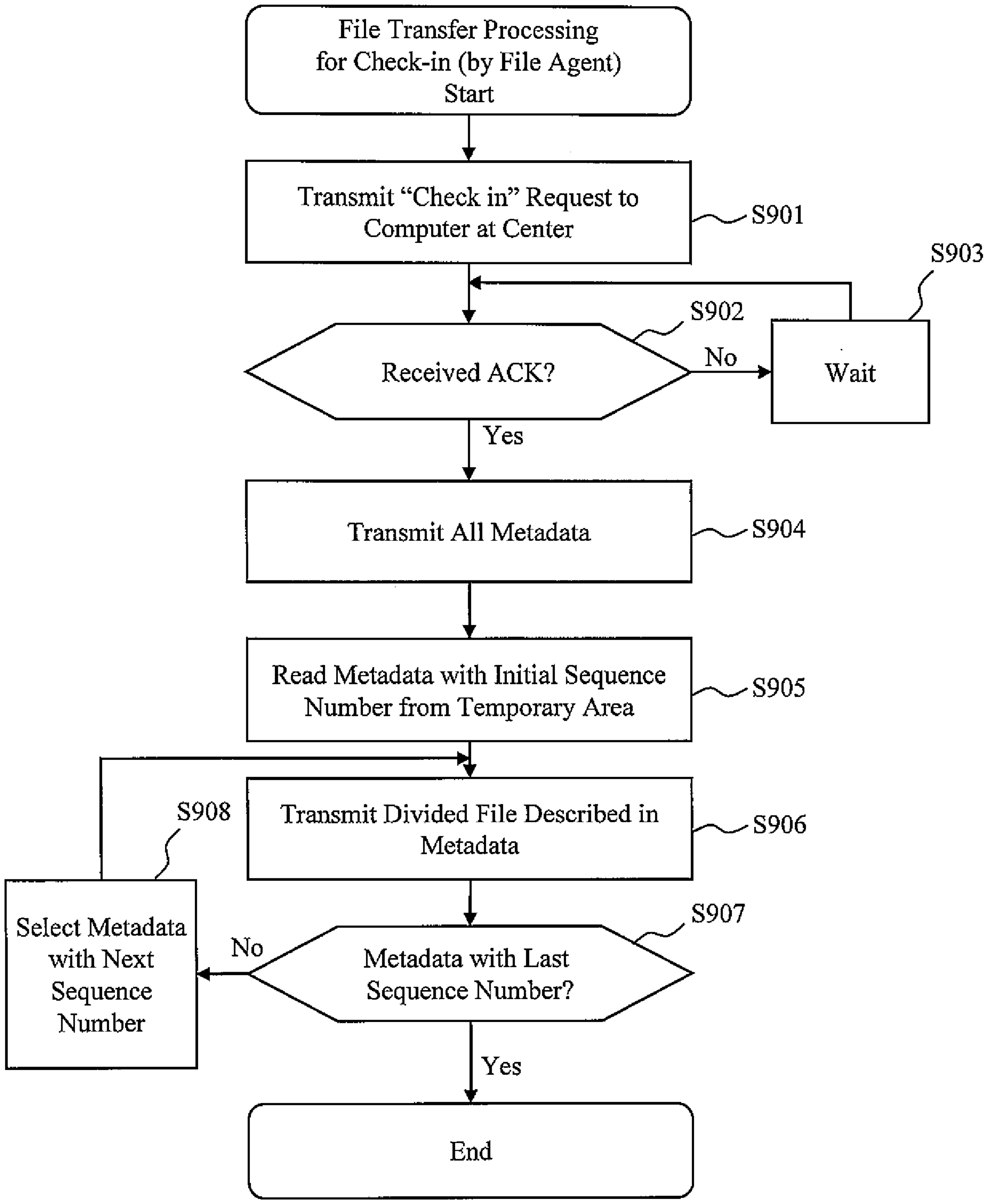
[Fig. 7]


Computer at Site A 100
Computer at Center 120
Computer at Site B 110
Check In
Divide File (701)
Acquire W Lock (703)
Check In (702)
Check Out
Ack (704)
Check Out (711)
Transmit Metadata (705)
Copy Metadata (712)
Transmit Divided File (706)
Request for Acquisition of Divided File (713)
Append (707)
Transmit Divided File (714)
Append (715)
Release W Lock (708)
Complete (709)
Complete (716)

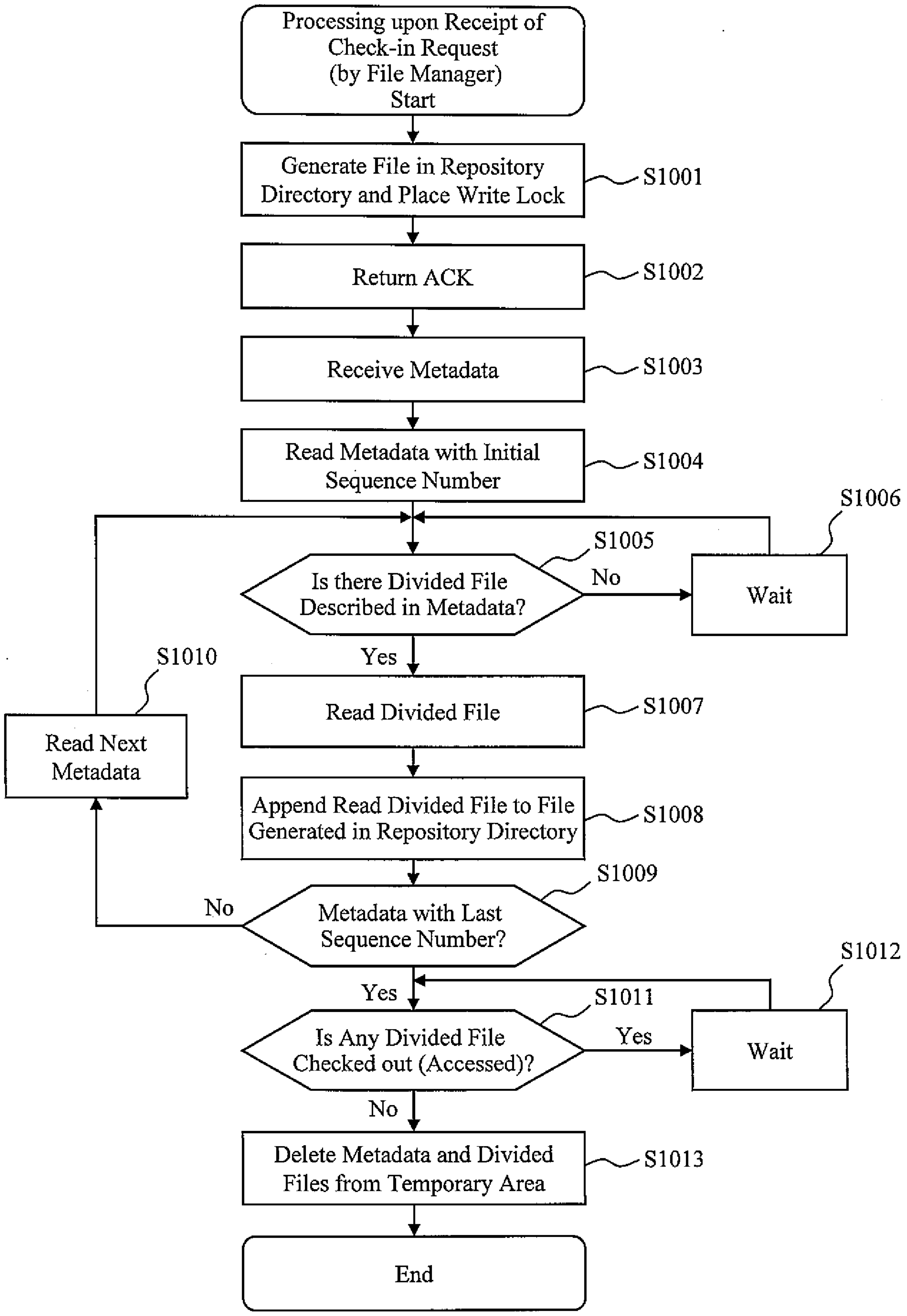
[Fig. 8]


Processing of Generating Divided Files for Check-in (by File Agent) Start
Acquire Total File Size and Initialize File Read Block Number and Sequence Number
S801
Generate Divide Files
S802
Copy Data in Range of from Read Block to Divided File Size into each Divided File
S803
Generate Metadata on Created Divided File
S804
Increment Read Block Number and Sequence Number
S805
S806
Is Read Block Number Less than or Equal to Number of Blocks Corresponding to Total File Size?
Yes
No
End

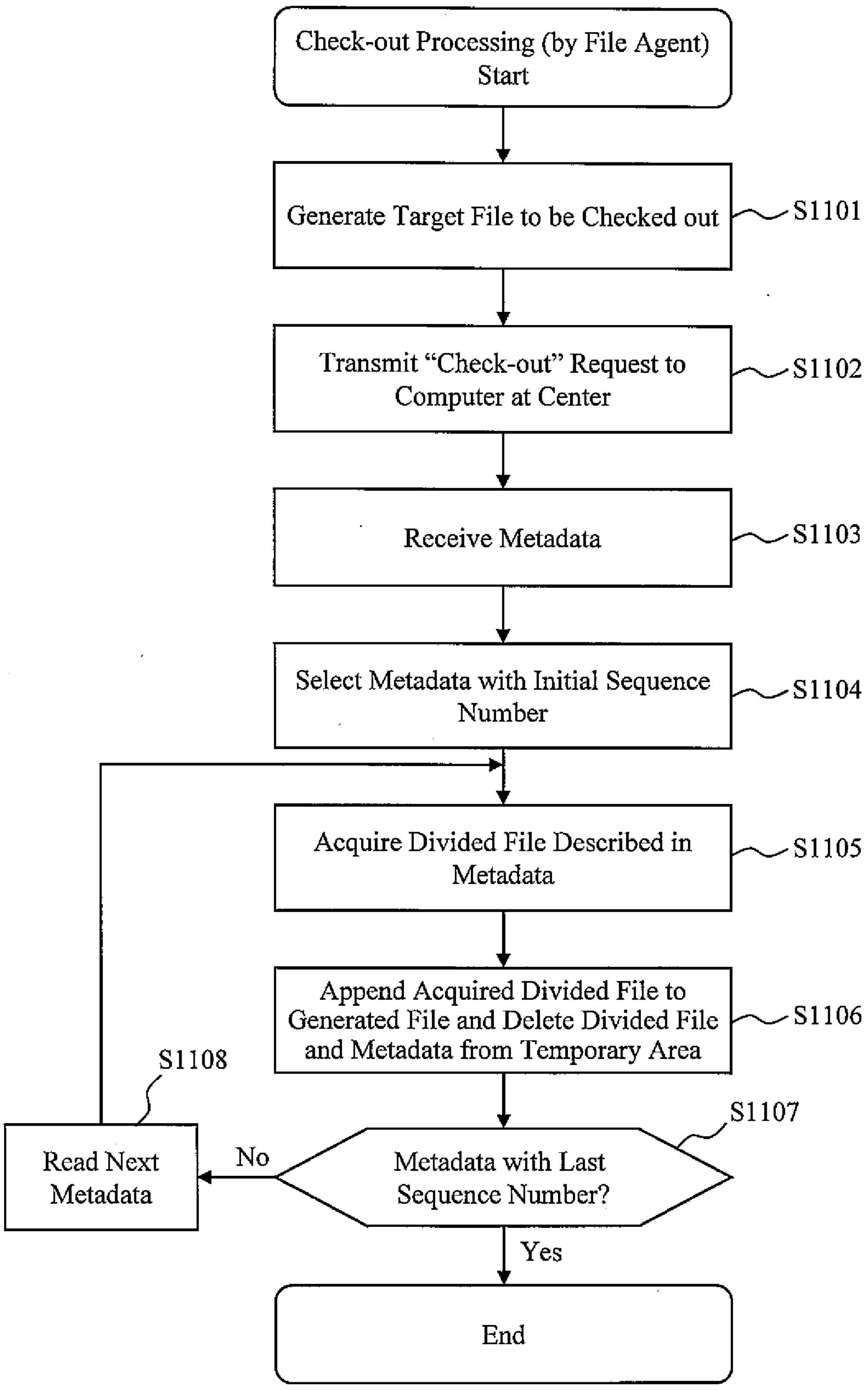
[Fig. 9]


File Transfer Processing for Check-in (by File Agent) Start
Transmit "Check in" Request to Computer at Center
S901
S903
S902
Received ACK?
No
Wait
Yes
Transmit All Metadata
S904
Read Metadata with Initial Sequence Number from Temporary Area
S905
S908
Transmit Divided File Described in Metadata
S906
S907
Select Metadata with Next Sequence Number
No
Metadata with Last Sequence Number?
Yes
End

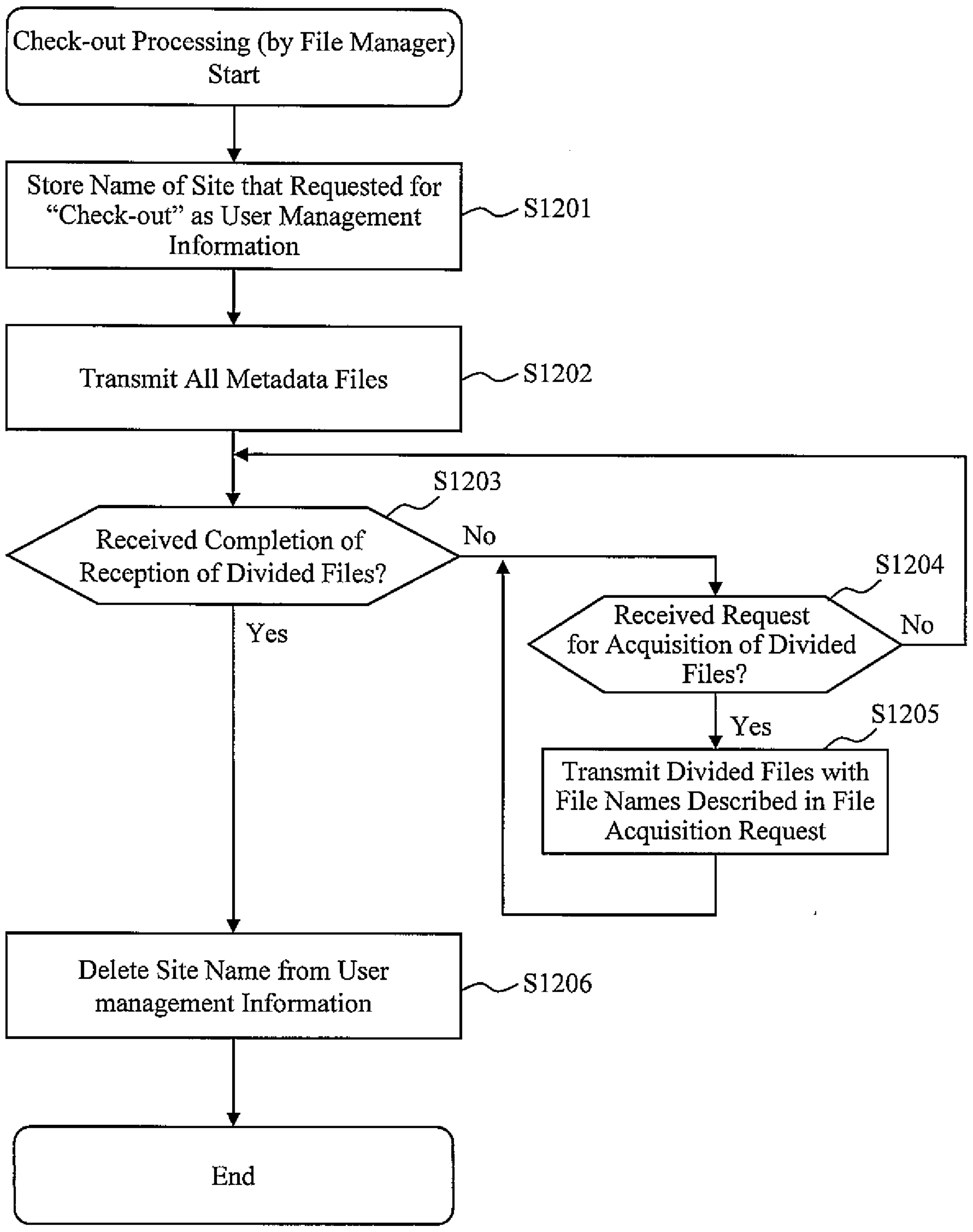
[Fig. 10]


Processing upon Receipt of Check-in Request (by File Manager) Start
Generate File in Repository Directory and Place Write Lock
S1001
Return ACK
S1002
Receive Metadata
S1003
Read Metadata with Initial Sequence Number
S1004
S1006
S1005
No
Is there Divided File Described in Metadata?
Wait
S1010
Yes
Read Divided File
S1007
Read Next Metadata
Append Read Divided File to File Generated in Repository Directory
S1008
S1009
No
Metadata with Last Sequence Number?
S1012
Yes
S1011
Yes
Is Any Divided File Checked out (Accessed)?
Wait
No
Delete Metadata and Divided Files from Temporary Area
S1013
End

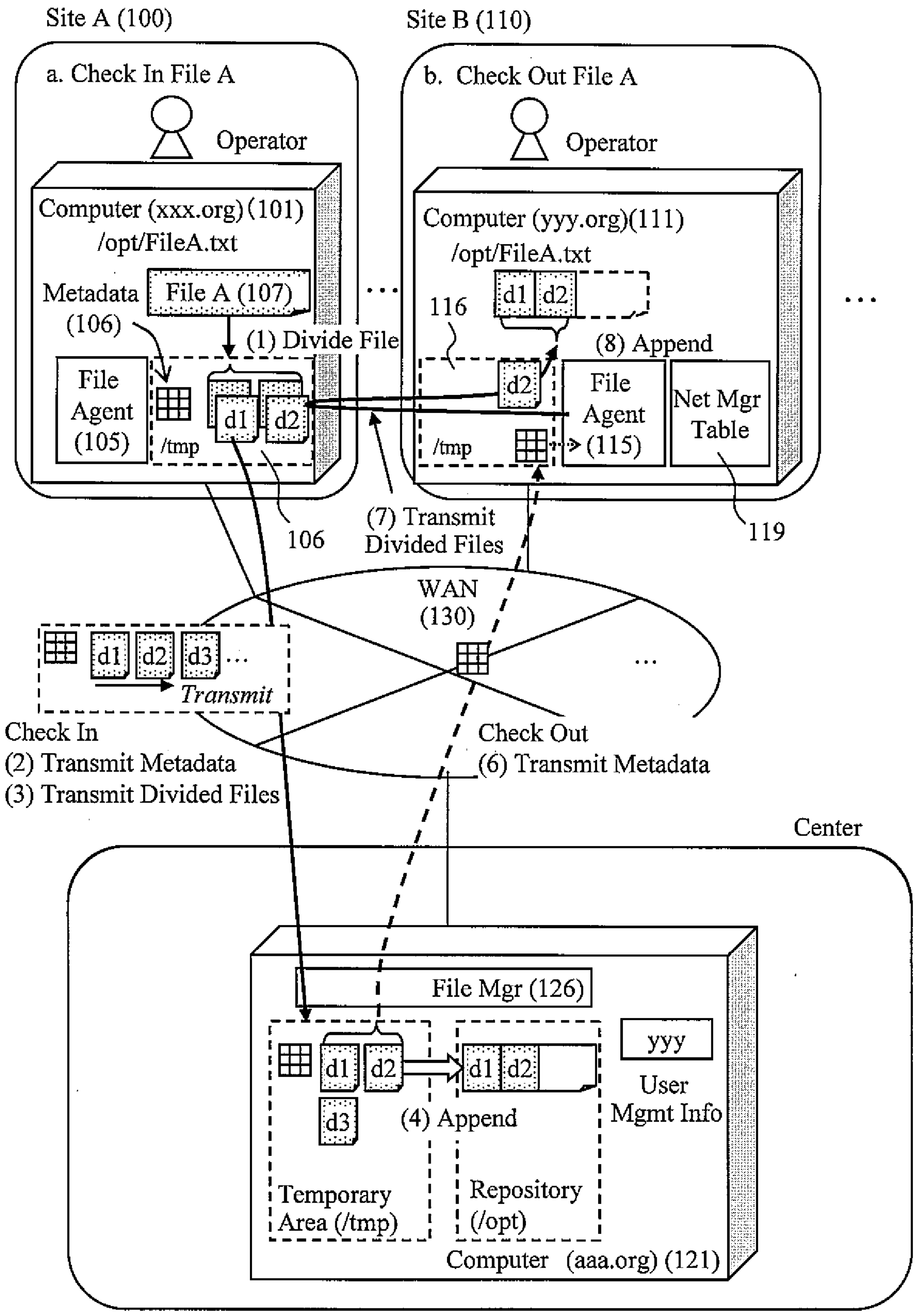
[Fig. 11]


Check-out Processing (by File Agent) Start
Generate Target File to be Checked out
S1101
Transmit "Check-out" Request to Computer at Center
S1102
Receive Metadata
S1103
Select Metadata with Initial Sequence Number
S1104
Acquire Divided File Described in Metadata
S1105
Append Acquired Divided File to Generated File and Delete Divided File and Metadata from Temporary Area
S1106
S1108
S1107
Read Next Metadata
No
Metadata with Last Sequence Number?
Yes
End

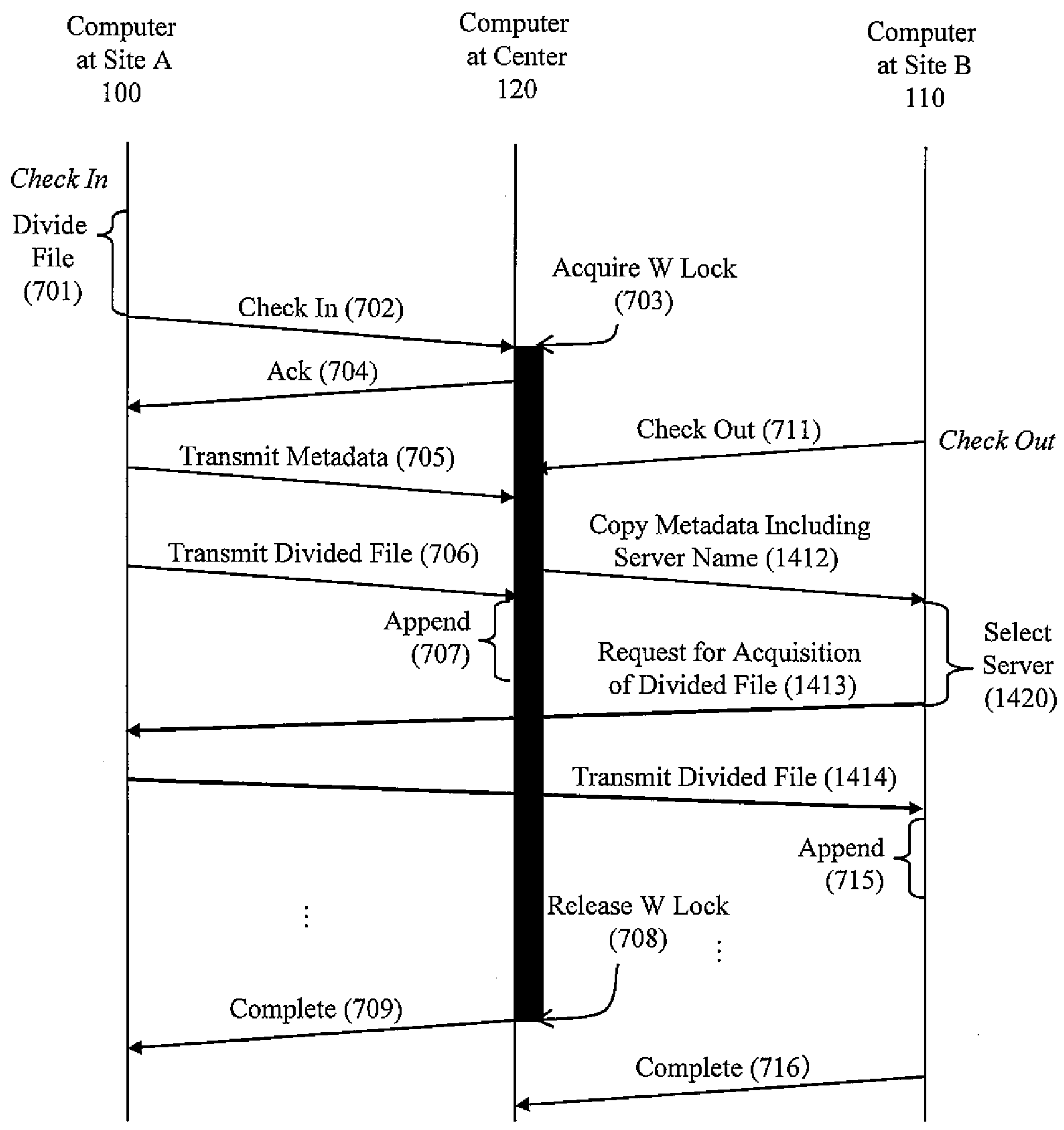
[Fig. 12]


Check-out Processing (by File Manager) Start
Store Name of Site that Requested for "Check-out" as User Management Information
S1201
Transmit All Metadata Files
S1202
S1203
Received Completion of Reception of Divided Files?
No
Yes
S1204
Received Request for Acquisition of Divided Files?
No
Yes
S1205
Transmit Divided Files with File Names Described in File Acquisition Request
Delete Site Name from User management Information
S1206
End

[Fig. 13]


Site A (100)
a. Check In File A
Operator
Computer (xxx.org) (101)
/opt/FileA.txt
Metadata (106)
File A (107)
(1) Divide File
File Agent (105)
/tmp
d1
d2
106
Site B (110)
b. Check Out File A
Operator
Computer (yyy.org)(111)
/opt/FileA.txt
d1
d2
116
(8) Append
File Agent (115)
Net Mgr Table
/tmp
119
(7) Transmit Divided Files
WAN (130)
d1 d2 d3 ...
Transmit
Check In
(2) Transmit Metadata
(3) Transmit Divided Files
Check Out
(6) Transmit Metadata
Center
File Mgr (126)
yyy
User Mgmt Info
d1 d2 d3
(4) Append
Temporary Area (/tmp)
Repository (/opt)
Computer (aaa.org) (121)

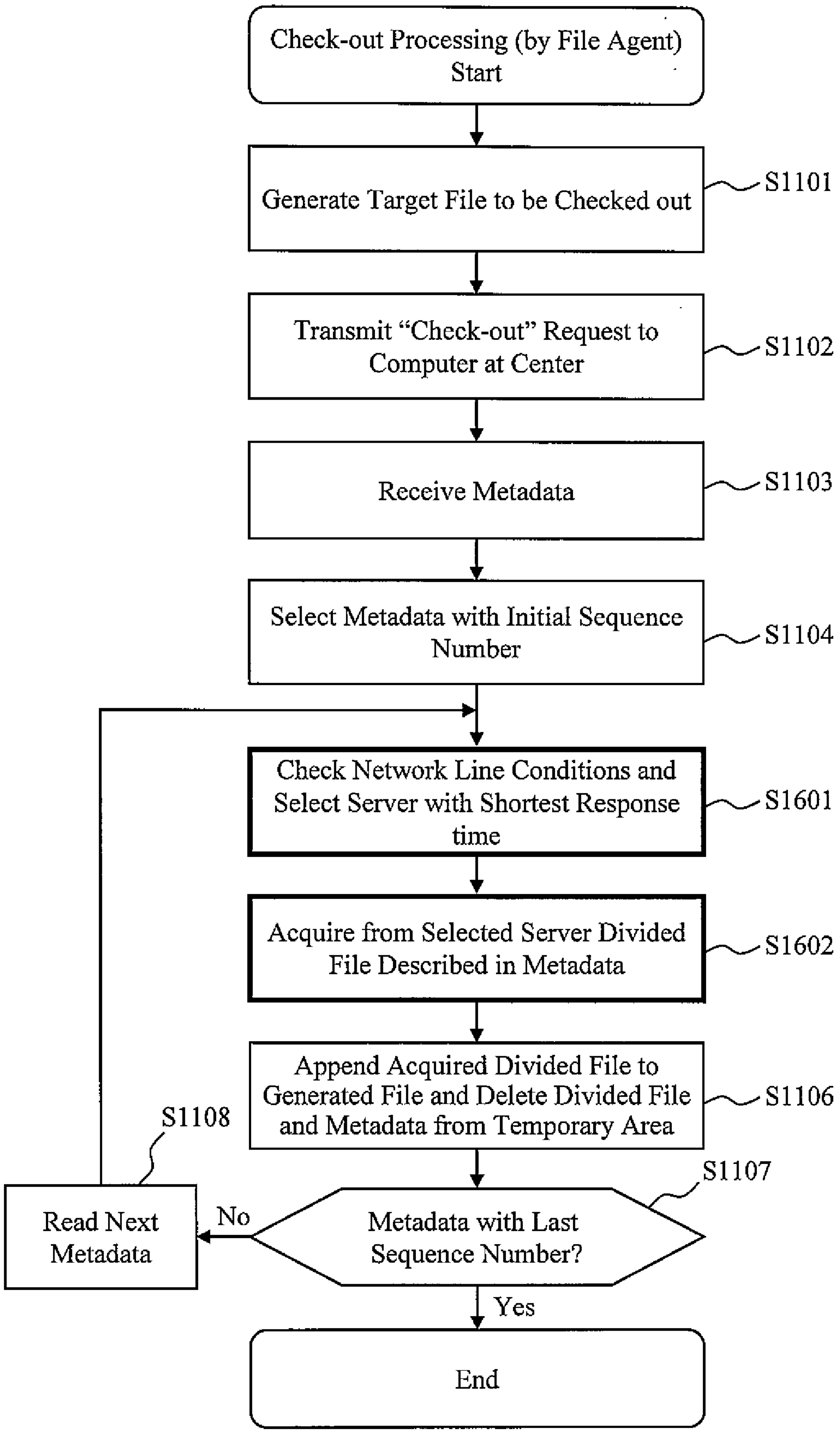
[Fig. 14]


Computer at Site A 100
Computer at Center 120
Computer at Site B 110
Check In
Divide File (701)
Check In (702)
Acquire W Lock (703)
Ack (704)
Check Out (711)
Check Out
Transmit Metadata (705)
Copy Metadata Including Server Name (1412)
Transmit Divided File (706)
Append (707)
Request for Acquisition of Divided File (1413)
Select Server (1420)
Transmit Divided File (1414)
Append (715)
Release W Lock (708)
Complete (709)
Complete (716)

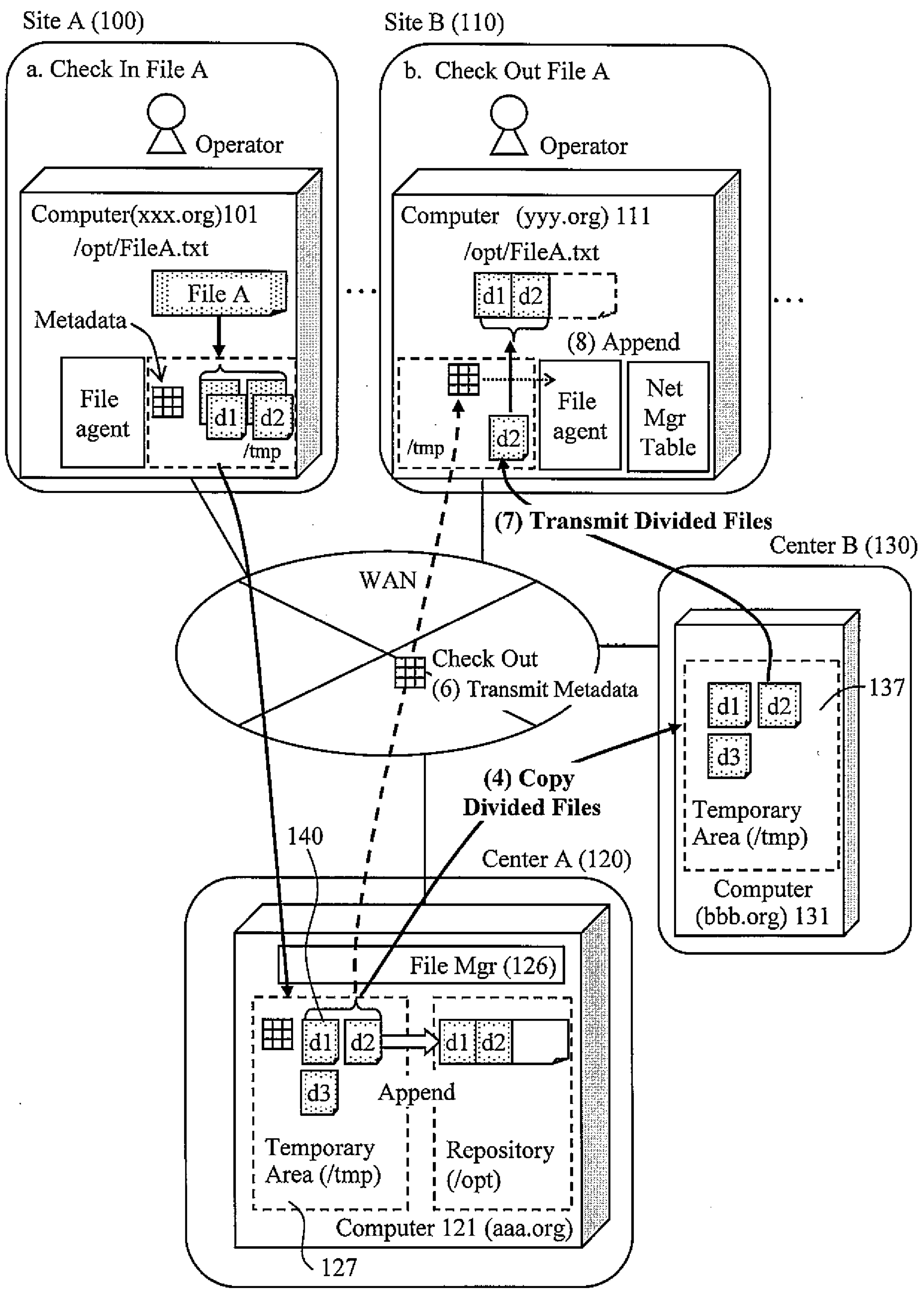
[Fig. 15]

Server (Computer) Name (1501)	Response Time (1502)	Hop Count (1503)	Load (1504)	...
xxx.org	1ms	1	10%	...
aaa.org	10ms	3	20%	...
bbb.org	13ms	2	10%	...
...	...	...	...	...

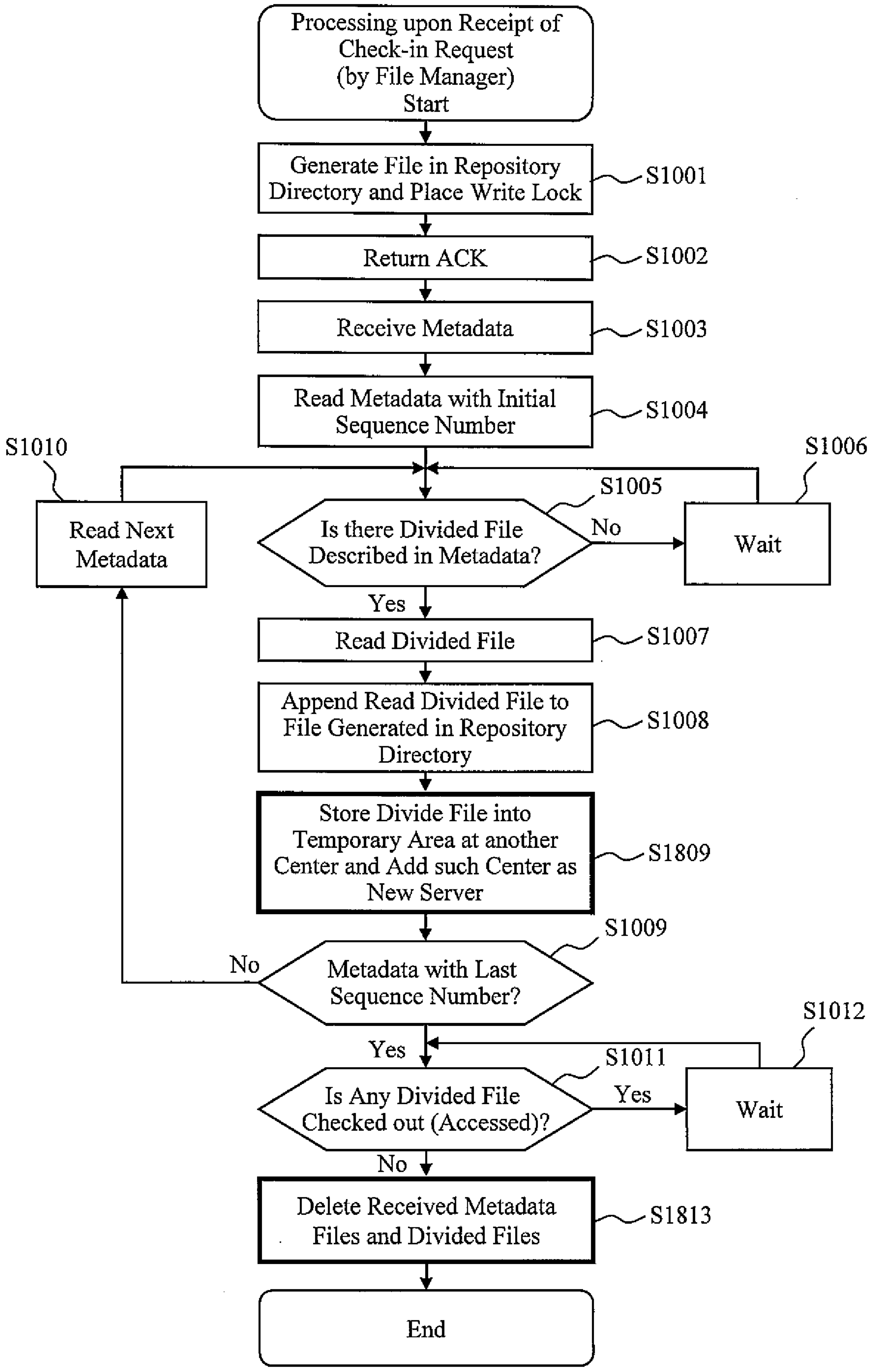
[Fig. 16]


Check-out Processing (by File Agent) Start
Generate Target File to be Checked out
S1101
Transmit "Check-out" Request to Computer at Center
S1102
Receive Metadata
S1103
Select Metadata with Initial Sequence Number
S1104
Check Network Line Conditions and Select Server with Shortest Response time
S1601
Acquire from Selected Server Divided File Described in Metadata
S1602
Append Acquired Divided File to Generated File and Delete Divided File and Metadata from Temporary Area
S1106
S1108
S1107
Read Next Metadata
No
Metadata with Last Sequence Number?
Yes
End

[Fig. 17]

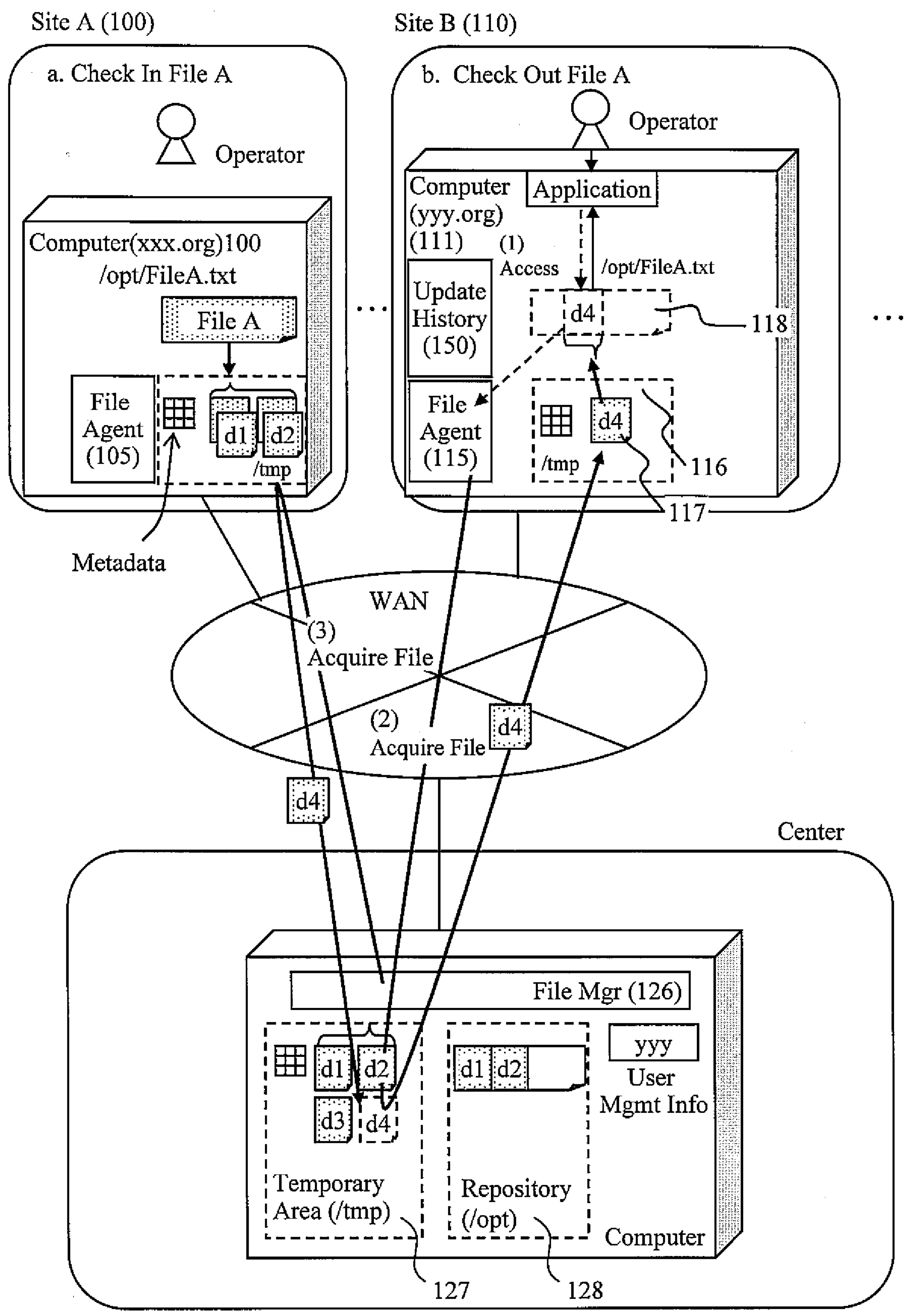

Site A (100)
a. Check In File A
Operator
Computer(xxx.org)101
/opt/FileA.txt
File A
Metadata
File agent
d1
d2
/tmp
Site B (110)
b. Check Out File A
Operator
Computer (yyy.org) 111
/opt/FileA.txt
d1
d2
(8) Append
File agent
Net Mgr Table
/tmp
d2
(7) Transmit Divided Files
Center B (130)
WAN
Check Out
(6) Transmit Metadata
d1
d2
d3
137
Temporary Area (/tmp)
Computer (bbb.org) 131
(4) Copy Divided Files
140
Center A (120)
File Mgr (126)
d1
d2
d3
Append
d1
d2
Temporary Area (/tmp)
Repository (/opt)
Computer 121 (aaa.org)
127

[Fig. 18]

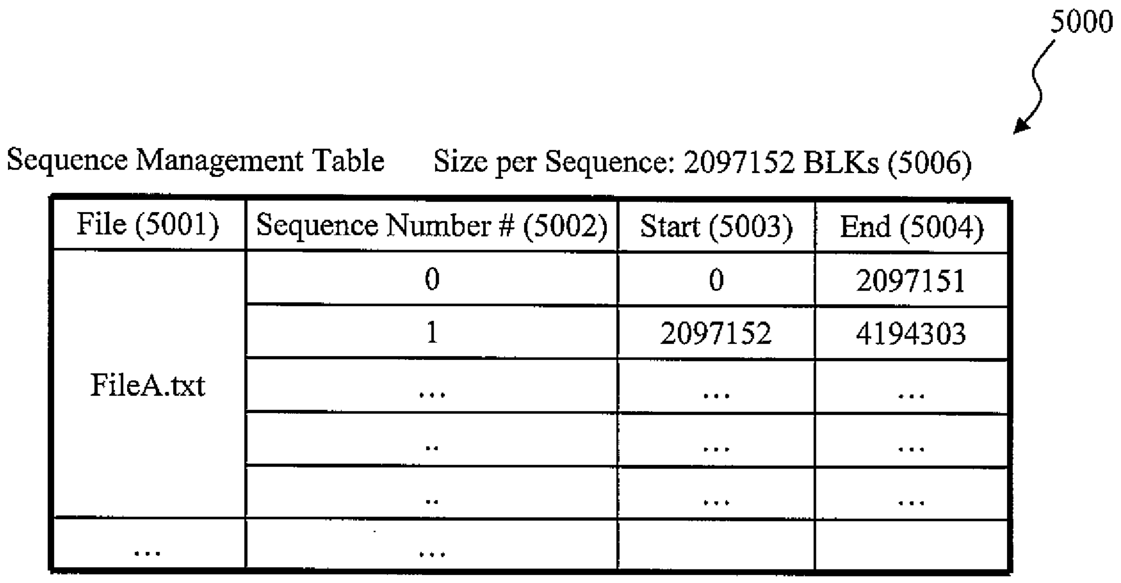


Processing upon Receipt of Check-in Request (by File Manager) Start
Generate File in Repository Directory and Place Write Lock
S1001
Return ACK
S1002
Receive Metadata
S1003
Read Metadata with Initial Sequence Number
S1004
S1010
S1006
S1005
Read Next Metadata
Is there Divided File Described in Metadata?
No
Wait
Yes
Read Divided File
S1007
Append Read Divided File to File Generated in Repository Directory
S1008
Store Divide File into Temporary Area at another Center and Add such Center as New Server
S1809
S1009
No
Metadata with Last Sequence Number?
S1012
Yes
S1011
Is Any Divided File Checked out (Accessed)?
Yes
Wait
No
Delete Received Metadata Files and Divided Files
S1813
End

[Fig. 19]


Site A (100)
a. Check In File A
Operator
Computer(xxx.org)100
/opt/FileA.txt
File A
File Agent (105)
d1
d2
/tmp
Metadata
Site B (110)
b. Check Out File A
Operator
Computer (yyy.org) (111)
Application
(1) Access
/opt/FileA.txt
Update History (150)
d4
118
File Agent (115)
d4
/tmp
116
117
WAN
(3) Acquire File
(2) Acquire File
d4
d4
Center
File Mgr (126)
d1
d2
d3
d4
Temporary Area (/tmp)
d1
d2
Repository (/opt)
yyy
User Mgmt Info
Computer
127
128

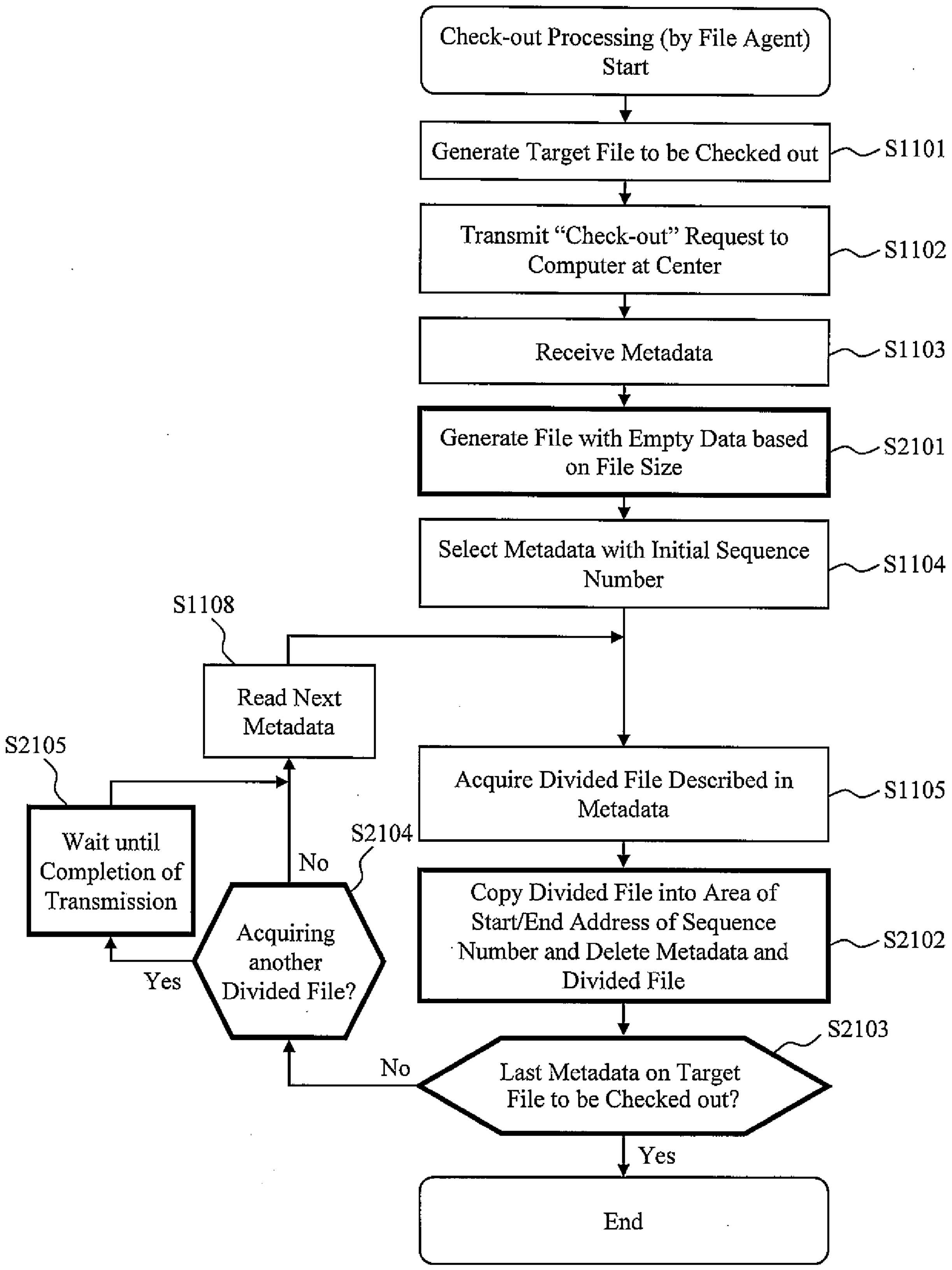
[Fig. 20]



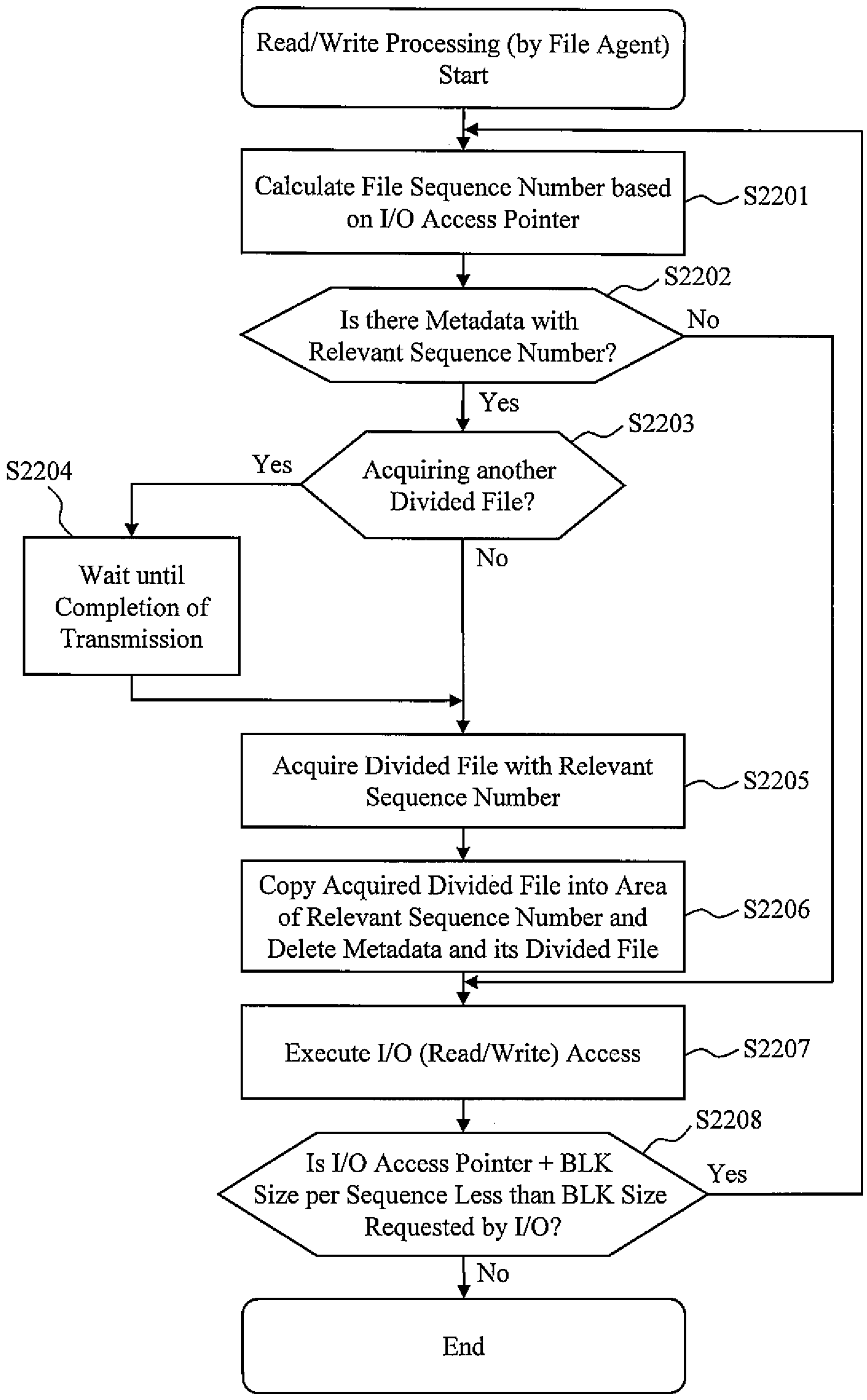
Sequence Management Table Size per Sequence: 2097152 BLKs (5006)

File (5001)	Sequence Number # (5002)	Start (5003)	End (5004)
FileA.txt	0	0	2097151
	1	2097152	4194303
	...	...	...
	..	...	...
	..	...	...
...	...		

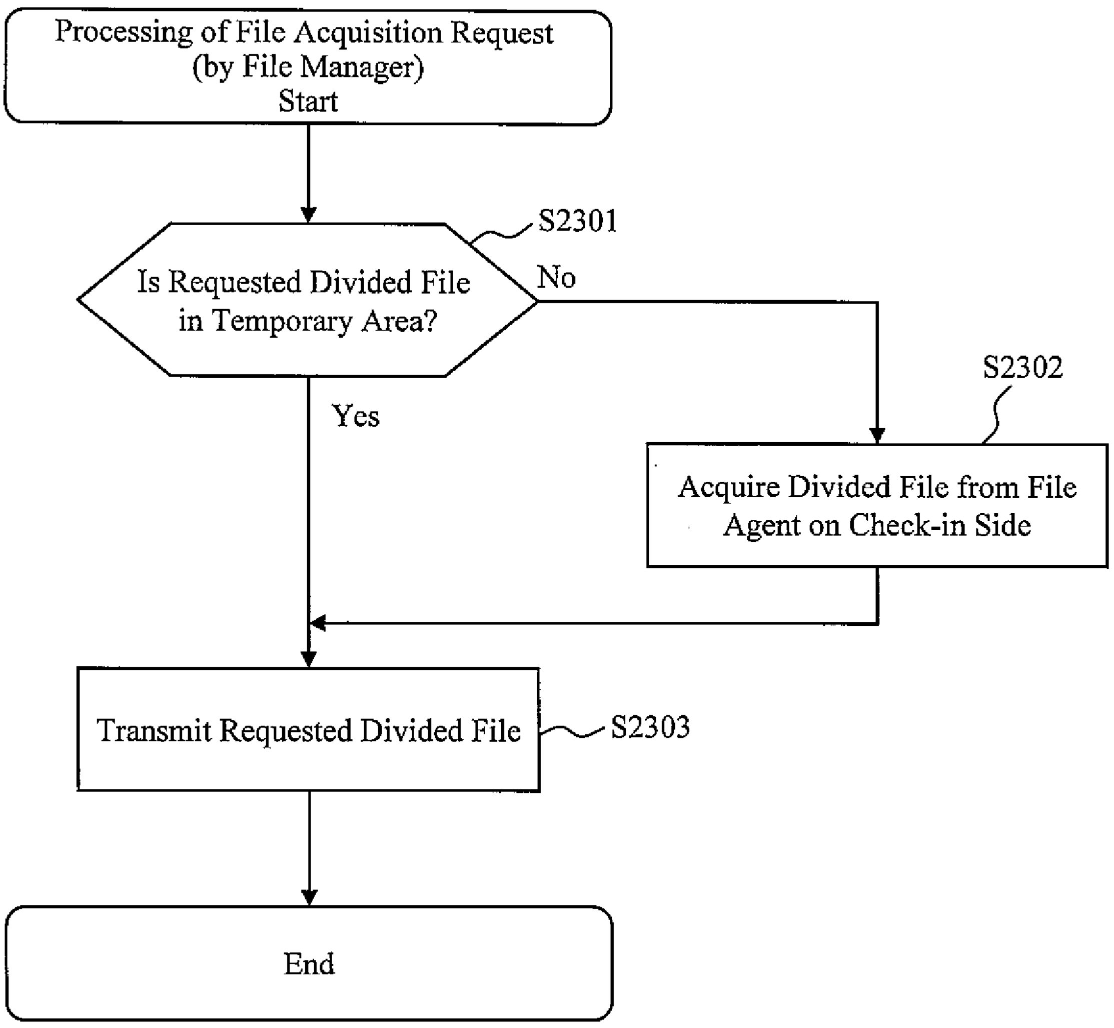
[Fig. 21]


Check-out Processing (by File Agent) Start
Generate Target File to be Checked out
S1101
Transmit "Check-out" Request to Computer at Center
S1102
Receive Metadata
S1103
Generate File with Empty Data based on File Size
S2101
Select Metadata with Initial Sequence Number
S1104
S1108
Read Next Metadata
S2105
Wait until Completion of Transmission
Acquire Divided File Described in Metadata
S1105
S2104
No
Acquiring another Divided File?
Yes
Copy Divided File into Area of Start/End Address of Sequence Number and Delete Metadata and Divided File
S2102
S2103
No
Last Metadata on Target File to be Checked out?
Yes
End

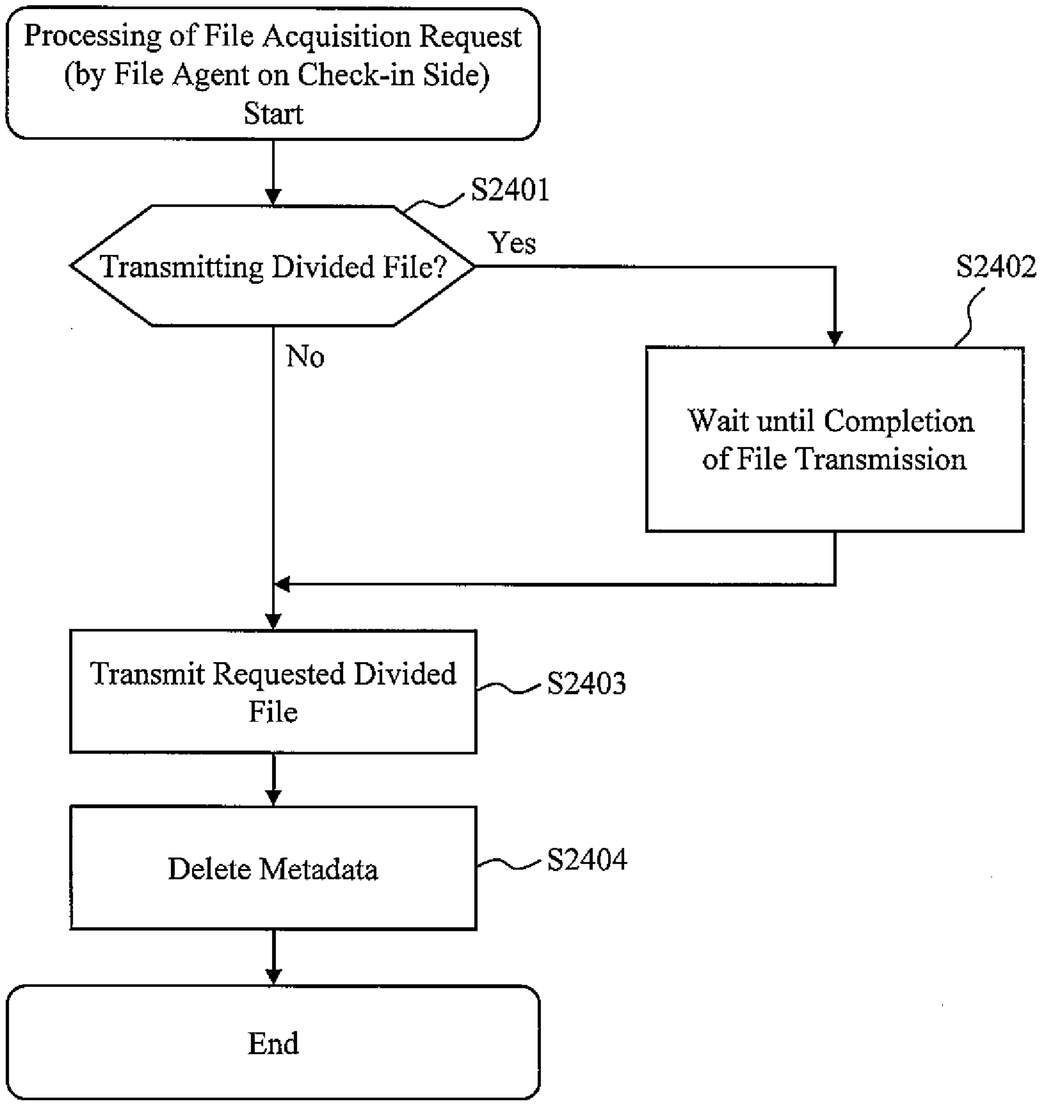
[Fig. 22]


Read/Write Processing (by File Agent) Start
Calculate File Sequence Number based on I/O Access Pointer
S2201
Is there Metadata with Relevant Sequence Number?
S2202
No
Yes
Acquiring another Divided File?
S2203
Yes
No
S2204
Wait until Completion of Transmission
Acquire Divided File with Relevant Sequence Number
S2205
Copy Acquired Divided File into Area of Relevant Sequence Number and Delete Metadata and its Divided File
S2206
Execute I/O (Read/Write) Access
S2207
Is I/O Access Pointer + BLK Size per Sequence Less than BLK Size Requested by I/O?
S2208
Yes
No
End

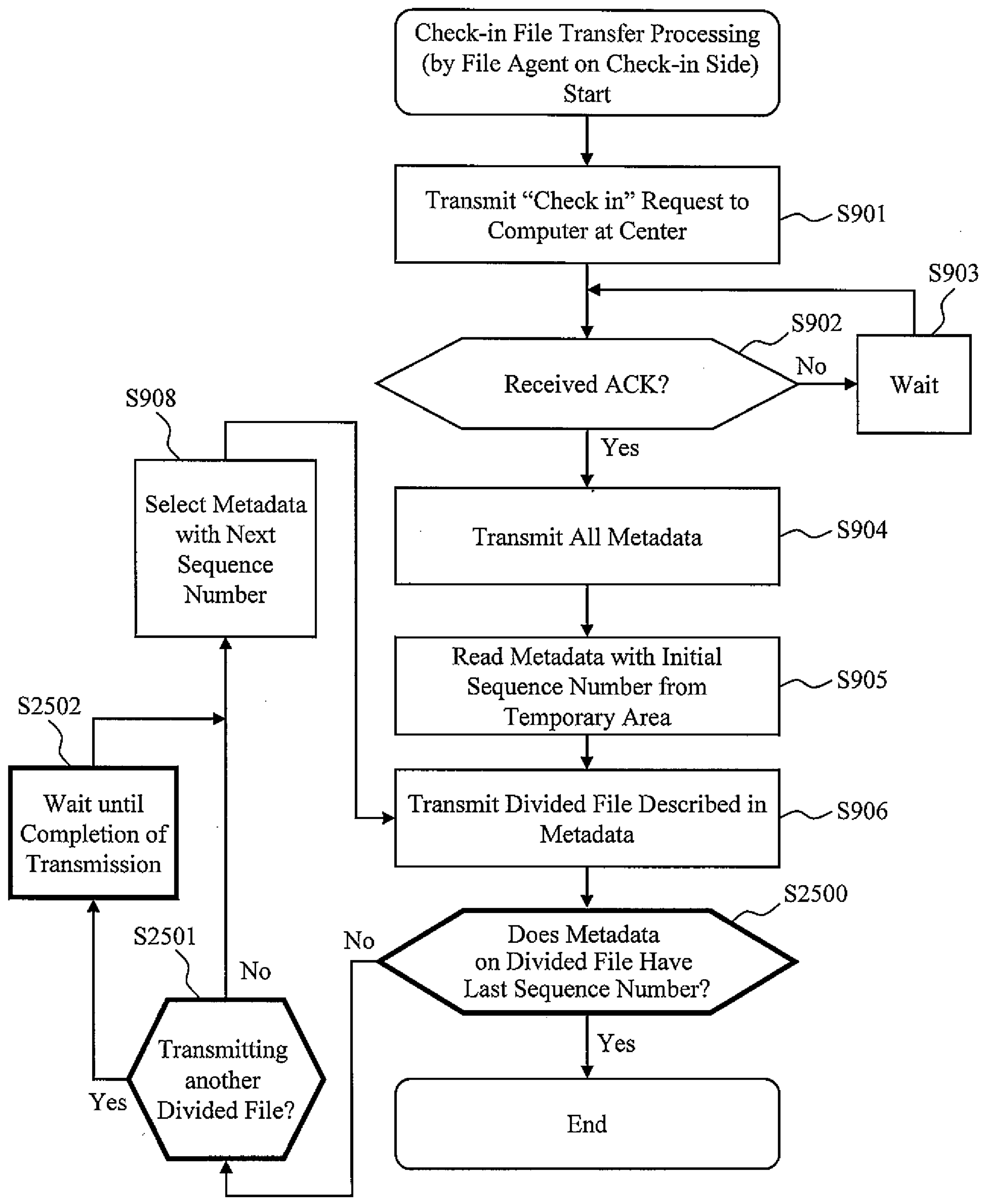
[Fig. 23]


Processing of File Acquisition Request (by File Manager) Start
S2301
Is Requested Divided File in Temporary Area?
No
Yes
S2302
Acquire Divided File from File Agent on Check-in Side
Transmit Requested Divided File
S2303
End

[Fig. 24]


Processing of File Acquisition Request (by File Agent on Check-in Side) Start
S2401
Transmitting Divided File?
Yes
No
S2402
Wait until Completion of File Transmission
Transmit Requested Divided File
S2403
Delete Metadata
S2404
End

[Fig. 25]


Check-in File Transfer Processing (by File Agent on Check-in Side) Start
Transmit "Check in" Request to Computer at Center
S901
S903
S902
Received ACK?
No
Wait
Yes
Transmit All Metadata
S904
Read Metadata with Initial Sequence Number from Temporary Area
S905
Transmit Divided File Described in Metadata
S906
S2500
Does Metadata on Divided File Have Last Sequence Number?
No
Yes
End
S908
Select Metadata with Next Sequence Number
S2502
Wait until Completion of Transmission
S2501
No
Transmitting another Divided File?
Yes

INTERNATIONAL SEARCH REPORT

International application No
PCT/JP2010/002103

A. CLASSIFICATION OF SUBJECT MATTER
INV. G06F17/30
ADD.

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
G06F

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)

EPO-Internal

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	US 2003/187860 A1 (HOLLAND MARK C [US]) 2 October 2003 (2003-10-02) page 1 - page 10	1-14
X	EP 0 890 915 A2 (IBM [US]) 13 January 1999 (1999-01-13) page 1 - page 35	1-14
X	US 2004/133652 A1 (MILOUSHEV VLADIMIR [US] ET AL) 8 July 2004 (2004-07-08) paragraph [0362] - paragraph [0363]	1-14
A	EP 1 544 735 A2 (IBM [US] LENOVO SINGAPORE PTE LTD [SG]) 22 June 2005 (2005-06-22) page 1 - page 5	1-14

-/--

[X] Further documents are listed in the continuation of Box C.

[X] See patent family annex.

* Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.

"&" document member of the same patent family

Date of the actual completion of the international search 10 November 2010	Date of mailing of the international search report 01/12/2010
Name and mailing address of the ISA/ European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Fax: (+31-70) 340-3016	Authorized officer Korkuzas, Valdas

INTERNATIONAL SEARCH REPORT

International application No
PCT/JP2010/002103

C(Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	US 2006/259710 A1 (GIMPL DAVID J [US] ET AL GIMPL DAVID JOSEPH [US] ET AL) 16 November 2006 (2006-11-16) page 1 - page 5 -----	1-14

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No
PCT/JP2010/002103

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 2003187860 A1	02-10-2003	NONE	
EP 0890915 A2	13-01-1999	TW 440769 B	16-06-2001
		US 5950199 A	07-09-1999
US 2004133652 A1	08-07-2004	US 2009234856 A1	17-09-2009
EP 1544735 A2	22-06-2005	CN 1667608 A	14-09-2005
		WO 2005059673 A2	30-06-2005
		JP 4149434 B2	10-09-2008
		JP 2005182800 A	07-07-2005
		KR 20050059989 A	21-06-2005
		US 2005131960 A1	16-06-2005
US 2006259710 A1	16-11-2006	US 2008104077 A1	01-05-2008