

METHOD AND APPARATUS FOR RECONCILING DIFFERENT VERSIONS OF A FILE

This is a continuation of application Ser. No. 08/061,674
filed on May 14, 1993, now abandoned.

FIELD OF INVENTION

This invention relates to distributed file systems and more
particularly to a method and system for reconciling different
versions of files, in which the files are stored in computers
at two or more separate locations or sites.

BACKGROUND OF THE INVENTION

There is a problem, especially with the portability of
computers and floppy disks that a given file, for instance, in
a lap top may not reflect the same information or data as the
same file at a desktop or fixed work station.

This is because work is frequently taken from location to
location. As frequently happens, a file created at a fixed work
station at the office may be modified at a remote location,
such as one's home, by merely transporting a disk or diskette
containing the file and modifying it at the remote location.
Multiple versions of the same file can also exist in distrib-
uted networks when files are modified or manipulated by
multiple users.

Problems thus arise when the versions of the file at two
sites, such as home and office, do not agree because they
have not been identically updated. This can occur by acci-
dent when one forgets to transport a floppy disk from one
location to the other; or when one forgets to load the disk
altogether.

It is of course desirable to have some synchronization
between versions of the same file when created or modified
at two different sites. For instance, it is possible to have the
same version of a file at two sites and only access one at a
time. When, however, versions of a file are created at two
sites, it is important to be able to update or reconcile the files
at both sites so as to appropriately update both files, or only
one file.

In the past, systems have compared the times that a file
was updated at different sites, have automatically selected
the most recent version, and have copied this version into the
appropriate file at both sites. Such systems include the
Novell, Netware, Sun Microsystems Network File System
(NSF) and Andrew File Systems. All of these systems have
problems with their automatic updating procedures.

It is also a feature of NFS, Andrew File System, and,
Netware that they automatically alter files immediately after
they are modified. This results in significant performance
problems as new versions of files are transmitted. Moreover
all updates are distributed throughout the network, exposing
raw work product to all on the system. It can also be an
embarrassment because of the automation process, where
those connected to the distributed system immediately have
knowledge of new unedited data and changes.

It will of course be appreciated that when there are
multiple users or contributors to a single file, such as in
writing software, or as in editing documents, it is very
important to alert all users of the same file as to what others
are doing so that at some point there is control in each of the
users as to what updating or reconciling of multiple versions
of the file will be permitted. It is particularly annoying for
the writer of software to have someone else edit his software

without his knowledge. Likewise, it is equally unfortunate
for the word processing public to have one user edit a work
without giving adequate notice to the other user.

More specifically, an inadequate solution to the problem
of multiple versions of the file at different locations exists in
distributed file system technology as represented by the
NFS, Andrew, Apple Share, Novell, and Research Systems
software such as Coda and Ficus. All of these systems give
the impression of being a single global file system. The
advantages of having a single global file system are auto-
matic updating, sharing, and familiar time sharing systems
semantics. However, the problems with such systems are
that they fail or degrade when disconnected, are unpredict-
able in performance, are unacceptable in that updates are at
the system's convenience and not at the user's, and that they
require a modified operating system, often requiring a single
vendor.

Another inadequate solution to the problem of multiple
revisions of a file is found in the explicit file transfer
technology associated with diskette/tape, E-mail, Lap-Link
and file transfer protocols. What these systems attempt to do
is copy files and carry or mail them. While the advantages
are complete user control, flexible transport, and conversion
between different systems, the disadvantages include com-
plicated and error-prone protocols, in which overwriting of
useful data can occur accidentally and in which there are no
"merges" of different versions.

In all these systems, the most recent version of the file in
one computer is automatically copied to the other. Thus,
current programs seek to establish which file is correct by
date and time, a technique called "time stamping". However,
these types of systems are far from failsafe. For instance,
assuming one wishes to delete a file on a lap top, deleting the
file at the lap top may not result in deleting the file at the
fixed work station, but rather in restoration of the obsolete
file found at the work station. Thus automatic reconciling
systems are error-prone.

More generally, if some work is to be accomplished on a
file in more than one place, then it is possible that neither
supercedes the other. Time stamp based reconciliation thus
will possibly result in over-writing relevant information. As
a result, user's work embodied in the older version may be
lost without any warning. It is also possible that this will
only happen when one forgets to hook up the computers for
the reconciliation between versions of the file.

What is important is to know when a file has been edited
in two places, what has been done, whether or not to
authorize a merge of the two versions, and on what basis. It
is therefore important to devise a system by which a merge
is done in a safe way. It is also important to provide a system
in which conflicts are recognized, with the conflict not
necessarily being resolved automatically, but rather at the
option of an individual operator who has been alerted to the
fact of a conflict.

Note that one prior art way of determining a conflict is the
so-called "journaling" technique which is to keep a record of
what has transpired at one central location. Using a single
centralized computer, a forward log or journal type of
reconciliation may be accomplished.

SUMMARY OF THE INVENTION

However, rather than keeping a centralized journal, it is a
feature of the present invention that each computer or
system keep its own journal. The journal, which is a history
of file versions, indicates the file which is edited and its