

two or more views. Derived objects that have never been shared reside in view-private storage.

While this invention has been particularly shown and described with references to a preferred embodiment thereof, it will be understood by those skilled in the art that various changes in form and details may be made therein without departing from the spirit and scope of the invention as defined by the appended claims. For example, although described in terms of a UNIX programming environment, this invention is applicable to a wide variety of operating systems and environments, including for instance the MS-DOS, OS/2 and Microsoft Windows 3.X and MS Windows/NT environments commonly found on IBM compatible personal computers or the MacOS environment or Apple Computers. Furthermore, although specific command language has been described, other command language can also be used to invoke the underlying processes of this invention.

We claim:

1. A data processing system for controlling versions of objects, comprising
 - a storage device for storing a plurality of versions of a set of objects,
 - a processor for executing instructions and for retrieving a specific version of one or more target objects from and storing each version of such target objects to the storage device during a process, and
 - an object version selector (i) which evaluates version selection rules by performing one or more queries on each target object during the process without requiring prior knowledge of each target object, and (ii) which provides the processor with access only to a specific version of such target object as determined by version selection rules during the process.
2. The data processing system of claim 1, wherein the object version selector provides for viewing selected versions of objects as a transparent file system comprising directories, files and links.
3. The data processing system of claim 2, wherein the transparent file system provides access to a version of an object by accessing that object in-place.
4. The data processing system of claim 1, wherein each object has a pathname for accessing the object from the storage device, each pathname having a plurality of pathname components listed in sequential order, and the object version selector applies version selection rules to a target object determined by matching an N component wildcard pattern to the final N components of a pathname.
5. The data processing system of claim 1, wherein the object version selector applies the existing version selection rules to newly created objects.
6. The data processing system of claim 1, wherein the object version selector stores the identity of a selected object version, selected by applying the version selection rules, in a cache memory and the object version selector provides for invalidating a selected object version identity stored in cache memory upon detecting a change which could affect the selected object version.
7. The data processing system of claim 1, wherein the version selection rules comprise a rule for selecting that version of an object that was the most recent version of that object at a specific time in the past.
8. The data processing system of claim 1, wherein the version selection rules comprise a rule for selecting that

version of an object that was the most recent version of that object at the specific time that a process requiring that object began.

9. The data processing system of claim 8, wherein the time that the process began is adjusted to compensate for time skew among the storage devices storing the objects required by the process.

10. The data processing system of claim 9, wherein the process includes a system build.

11. The data processing system of claim 1, wherein the versions of objects are conceptually stored along branches of a tree structure, with each sequential version of an object being stored sequentially as the next version of that object in the same branch, and parallel versions of an object being similarly stored sequentially along a parallel branch, the parallel branch having its source at a version of the object on another branch.

12. The data processing system of claim 11, wherein the version selection rules comprise a rule for creating a new parallel branch when a version of an object is accessed by the processor for modification creating a new version of the object, the new version of the object being subsequently stored in the newly created parallel branch.

13. The data processing system of claim 11, wherein the object version selector provides for accessing the versions of an object as a file system structured as directories and sub-directories.

14. The data processing system of claim 1, wherein the storage device stores a plurality of attributes associated with each version of the objects, and the object version selector provides for searching for a version of an object determined by the state of the associated attributes.

15. The data processing system of claim 1, wherein versions of objects comprise compressed files.

16. The data processing system of claim 15, wherein a version of an object stored as a compressed file is uncompressed upon access to the object by the processor.

17. The data processing system of claim 16, wherein an uncompressed version of an object is stored in a cache memory.

18. The data processing system of claim 1, wherein the objects comprise directories.

19. The data processing system of claim 18, wherein a version of a directory object comprises a catalog of objects and their associated names, each cataloged object referring to all versions of that object.

20. The data processing system of claim 19, wherein the version selected of a directory object can be independent of the version selected of an object cataloged in the selected directory object.

21. The data processing system of claim 1, wherein the object version selector further comprises

a system builder for executing a system build process producing derived objects from versions of source objects selected by the object version selector;

an auditor for recording, as an audit record, which versions of the source objects are accessed through the object version selector during a system build process, the audit record being associated with the produced derived objects; and

wherein the system builder further comprises a comparator for comparing the audit record, associated with a derived object produced by a previous system build process and stored in the storage device, with the requirements of the currently executing system build