11. Apparatus according to claim 7 wherein each of the object pointers comprises an object which includes a pointer that points to a demand loader object.

12. A method for constructing a document from a plurality of objects in a computer system having a memory and a non-volatile storage, a document object being accessed whenever an invocation of one of the document object's methods or the manipulation of one of the document object's attributes is required, the method comprising the steps of:

A. creating a demand loader object associated with each of the plurality of objects, each of the demand loader objects including program logic for loading the associated object into the memory from the storage when the associated object is accessed; and

B. connecting two of the plurality of objects with object pointer objects, each object pointer object being a member of a first one of the two objects and having a pointer therein pointing to a demand loader object associated with a second one of the two objects.

13. The method according to claim 12 wherein step A comprises the step of:

A1. creating each demand loader object with a reference count attribute having a reference count stored therein which reference count indicates the number of object pointers which point to the each demand loader object.

14. The method according to claim 13 wherein step B comprises the step of:

B1. creating each object pointer object with program logic for updating a reference count attribute in a demand loader object to which the object pointer points.

15. The method according to claim 12 wherein step A further comprises the step of:

A2. creating each demand loader object with an object ID for locating an associated object in the non-volatile storage.

16. The method according to claim 12 wherein step A further comprises the step of:

A3. creating each demand loader object with a pointer to an associated object.

17. A method for use with a computer system having a memory and a non-volatile storage, the method creating a new version of a document, which is composed of a plurality of objects which are interconnected by object pointers, a document object being accessed whenever an invocation of one of the document object's methods or the manipulation of one of the document object's attributes is required, in response to a user request to change one of the plurality of objects, and comprising the steps of:

A. creating a demand loader object associated with the one object, the demand loader object including program logic for loading the one object into the memory from the storage when the one object is accessed;

B. creating a list of object versions for the one object, the list including a version ID and a pointer to a copy of the one object for each object version; and

C. making a copy of the one object and inserting a version ID and a pointer to the one object copy into the list in response to the request.

18. The method according to claim 17 wherein the new document version has a document version ID and step C comprises the step of:

C1. inserting the document version ID into the list in response to the request.

19. The method according to claim 18 wherein step C further comprises the step of:

C2. checking the list to determine whether the document version ID is in the list in response to the request.

20. The method according to claim 19 wherein step C further comprises the step of:

C3. making a copy of the one object and inserting the document version ID and a pointer to the one object copy into the list if the document version ID is not in the list.

21. The method according to claim 17 wherein step A comprises the step of:

A1. creating a demand loader object with an object ID for retrieving the one object from the storage into the memory.

22. A computer program product for constructing a document from a plurality of objects in a computer system having a memory and a non-volatile storage, a document object being accessed whenever an invocation of one of the document object's methods or the manipulation of one of the document object's attributes is required, the computer program product comprising a computer usable medium having computer readable program code thereon, the computer program code including:

means for creating a demand loader object in the memory, the demand loader object being associated with each of the plurality of objects, each of the demand loader objects including program logic for loading the associated object into the memory from the storage when the associated object is accessed; and

means for creating object pointer objects in the memory, each of the object pointer objects connecting two of the plurality of objects, each object pointer object being a member of a first one of the two objects and having a pointer therein pointing to a demand loader object associated with a second one of the two objects.

23. A computer program product for use with a computer system having a memory and a non-volatile storage, the computer program product creating a new version of a document, which is composed of a plurality of objects which are interconnected by object pointers, in response to a user request to change one of the plurality of objects, a document object being accessed whenever an invocation of one of the document object's methods or the manipulation of one of the document object's attributes is required, and comprising a computer usable medium having computer readable program code thereon, the computer readable program code including:

means for creating a demand loader object in the memory, the demand loader object being associated with the one object, the demand loader object including program logic for loading the one object into the memory from the storage when the one object is accessed;

means for creating in the demand loader object a list of object versions for the one object, the list including a version ID and a pointer to a copy of the one object for each object version; and

means responsive to the request for making a copy of the one object and inserting a version ID and a pointer to the one object copy into the list.

* * * * *