

version to write the file, as well as data that is unknown but carried in the file.

The object version watermark can provide specific information for a particular object, whereby the particular object can be loaded, saved, or converted based on that information. The object version watermark can provide information for the particular object and can indicate the highest file version of any property that a specific object contains. As an example, suppose that a hyperlink property was added to object X in version 2.0, and no new properties were added to object X in version 3.0. If object X contains a hyperlink written in version 2.0 or a later version, its version watermark would be set to version 2.0. If object X does not contain a hyperlink (and therefore no hyperlink property is written) or was written by version 1.0, the version watermark would be set to version 1.0. This information can be used to further optimize loading and saving operations with respect to each individual object and the highest version that would understand all properties in that object. For example, object X that was written by version 3.0 and did not contain a hyperlink (and would therefore have a version 1.0 object watermark) could be written and read by version 1.0 without being concerned about future version properties existing, even though the file high version watermark would reflect version 3.0.

Generally, the most current version of the application program to save a file is best suited to determine how to handle any information from that version or previous versions. The most current version can read and understand everything in the file. The most current version of the application program determines which information to keep, convert, or discard. Accordingly, the highest version to write a file determines whether information is discarded or converted. All lower versions can simply propagate any unknown information.

Finally, all steps of methods **500** and **600** shown in FIGS. **5** and **6** are not required for the invention to be operable. Additionally, some steps may be performed in a different order than illustrated.

With reference to FIGS. **7-9**, specific exemplary embodiments of methods **500** and **600** according to the present invention will be described. In each of FIGS. **7-9**, method **500** or **600** has determined that the last version watermark is less than the active version of the application program (i.e., that the last version of the application program to save the original data file corresponds to a previous version of the application program with respect to the active version) (see step **595** of FIG. **5**). Additionally, the embodiments described below with reference to FIGS. **7-9** are also operable when method **500** or **600** has determined that the high version watermark is less than the active version of the application program.

FIG. **7** is a flow chart depicting a method **700** for loading and saving an original data file according to the present invention, where the active version has an additional object property that is not included in a previous version. In step **710** of method **700**, a default value for a new property is defined in an OPL for the active version of the application program. The process proceeds to step **725** where the original data file is loaded without changes. After the original data file has been modified, it is saved in step **730**. Accordingly, the new property is saved for use with the active and future versions of the application program. Previous versions will simply ignore the new property. In step **740**, the high version watermark is updated to the active version.

FIG. **8** is a flow chart depicting a method **800** for loading and saving an original data file according to the present invention, where the active version has deleted an object property that was used in a previous version. In step **820**, the deleted property is ignored when the original data file is loaded in the active version. When a modified version of the original data file is saved in step **830**, the file is saved with the deleted property available to older versions. While the active version did not use the deleted property, the deleted property is returned to the saved file for use by previous versions. The deleted property can be returned to the saved file by leaving the deleted property in the original data file, i.e., by not writing the deleted property out of the file. Alternatively, the deleted property can be returned to the saved file by propagating the deleted property (from the original data file) back into the saved file. Because the deleted property is known to the active version, the value of the deleted property can be derived from other property values, or a default value representing the deleted property can be generated, to allow the deleted property to be returned to the file. In step **840**, the high version watermark can be updated to the active version.

For example, suppose that the previous version included a drop shadow property on an object. The active version may no longer support this property, i.e., the drop shadow property has been deleted from the active version. The newer version of the application program recognizes that the drop shadow property has been deleted. However, the drop shadow property can still be written to the file by the active version. In step **820**, the original data file can be loaded, and the deleted drop shadow property can be ignored. When a modified version of the original data file is saved in step **830**, the file can be saved with the deleted drop shadow property available to older versions. While the active version did not use the deleted drop shadow property, the deleted drop shadow property can be returned to the saved file for use by previous versions. The drop shadow property value can be determined from the value of other properties, or a default value can be used. In step **840**, the high version watermark can be updated to the active version.

FIG. **9** is a flow chart depicting a method **900** for loading and saving an original data file according to the present invention, where the active version has modified an old object property of a previous version. Rather than replacing the old property with the modified property, a new property that represents the modified property is provided in the active version. In step **910** of method **900**, a default value for a new property is defined in an OPL of the active version of the application program. The old property is left unchanged. Then in step **920**, the original data file is loaded into the active version, and the old property is converted to the new (modified) property. After the original data file has been modified, it is saved in step **930**. In step **930**, the new property is saved without reference to the old property. Additionally in step **930**, the new property is converted to the old property and saved for use by a previous version. In step **940**, the high version watermark is updated to the active version.

For example, suppose that the previous version of the application program used the RGB (Red, Green, Blue) color model to define colors of objects. The active version of the application program could include a new property to allow defining the colors of objects in the CMYK color model (Cyan, Magenta, Yellow, Black) used by professional printers. The new CMYK color property can be based on the old RGB color property, and an equivalent CMYK value can be determined for each RGB value. A new opyid can be added