

[0022] FIG. 7 shows an exemplary operation process that an apparatus according to another embodiment of the present invention provides inter-version document compatibility, wherein the apparatus is at a network server and is for providing inter-version conversion of documents as a network server;

[0023] FIG. 8 illustrates a method for providing inter-version document compatibility according to an embodiment of the present invention;

[0024] FIG. 9 shows the sub-steps included in the step of conversion between documents of different versions of an application by using a conversion stack according to an embodiment of the present invention; and

[0025] FIG. 10 shows the sub-steps included in the step of conversion between documents of different versions of an application by using a conversion stack according to another embodiment of the present invention.

#### DETAILED DESCRIPTION OF PREFERRED EMBODIMENTS

[0026] FIG. 2 shows an apparatus for providing inter-version document compatibility according to an embodiment of the present invention. The apparatus 200 includes: a conversion stack 201 which includes differentiation models between different versions of the data model of an application; and a conversion module 202 for converting between documents of different versions of the application by using the conversion stack 201, so as to provide compatibility between documents of different versions of the application.

[0027] According to an embodiment of the present invention, the differentiation models between data models of different versions of an application are differentiation models between the data models of adjacent versions of the application. By including only the differentiation models between the data model of adjacent versions, instead of differentiation models between the data models of any two versions, especially when the application has many versions, the number of the differentiation models that must be maintained is reduced significantly; and the conversion stack 202 can still realize conversion between documents of any two versions by performing one or more conversions using one or more differentiation models between the data models of the two versions. The adjacent versions usually means consecutive versions in the evolution process of an application, that is, if one version is developed based on another version, the version and the other version are adjacent versions.

[0028] For example, the conversion stack 201 shown in FIG. 2 includes differentiation models between the adjacent versions 1.0 and 1.1, 1.1 and 1.2, 1.0 and 2.0, 2.0 and 2.1, 2.1 and 2.2, 2.1 and 3.0, 3.0 and 3.1, 3.1 and 3.2. The conversion stack 201 shown in FIG. 2 reflects a relatively complex version evolution process which includes branches, wherein versions 1.0, 1.0, 1.2 are one version evolution branch, versions 1.0, 2.0, 2.1 and 2.2 are another version evolution branch, and versions 1.0, 2.0, 2.1, 3.0, 3.1, 3.2 are still another version evolution branch. Of course, the conversion stack shown in FIG. 2 is only exemplary, rather than limitation to the conversion stack of the present invention.

[0029] For example, the conversion stack 201 may include differentiation models between adjacent versions in a relatively simple version evolution process (for example, a linear evolution process without branches). In addition, in some other embodiments of the present invention, the conversion stack 201 may include the differentiation models between any

two versions in a plurality of versions. In still some other embodiments of the present invention, the conversion stack 201 may include differentiation models between some selected versions in a plurality of versions.

[0030] The conversion stack 201 further includes optional verification models which include rules for verifying whether a document conform to constraints of data models, and the apparatus 200 further includes an optional verification module 203 for verify a converted document according to a verification model, so as to enable the converted document to conform to the constraints of the data model of the version as the target of the conversion. The rules can be specified by a user according to a data model and stored in a verification model.

[0031] For example, the data model of version 2 requires that a teacher teaches at most 50 students, while the data model of version 3 eliminates the constraint. As such, when converted from version 3 to version 2, the document obtained may not meet the constraint of the data model of version 2. Therefore, a verification model corresponding to version 2 may include the rule requiring that a teacher teaches at most 50 students, and the verification module 203 may check the document converted from version 3 to version 2 according to this rule in the verification model, and inform the user to process when it detects that the document does not conform to the rule.

[0032] The apparatus 200 is in an application of each version, and the conversion stack 201 may include differentiation models between data models of this version and lower versions. For example, if the version of the application is V2.1, then the conversion stack 201 in the application may include differentiation models between data models of the versions prior to version V2.1 (for example, between data models of the adjacent versions); if the version of the application is V3.2, then the conversion stack 201 in the application may include differentiation models between data models of the versions prior to version V3.2 (for example, between data models of the adjacent versions), and so on.

[0033] According to another embodiment of the present invention, the apparatus 200 is at a network server, and provides document inter-version conversion as a networked service. Thus, the conversion stack 201 is at the network server and includes differentiation models between data models of versions (for example, the adjacent versions) prior to a specific version (for example, the currently latest version) of the application.

[0034] In addition, the apparatus 200 may further include an optional user interface (UI) module through which the user can manipulate and control the document conversion process. For example, the UI module may provide a dialog box allowing the user to select a target version of document conversion, and provide another dialog box allowing the user to select the storage location of the data generated during the conversion process, such as documents of intermediate versions.

[0035] The apparatus 200 further includes the following optional modules which are not shown: a model extractor for acquiring data models of different versions of an application; a model comparator for generating differentiation models between the data models of different versions of the application according to the data models of different versions of the application, and for storing the differentiation models into the conversion stack 201.

[0036] A data model of an application is also referred to as a meta-model or schema of the application and it defines the