

more modified documents of the intermediate versions and the modified document of the higher version.

[0046] FIG. 6 shows an exemplary operation process by which the apparatus 200 provides inter-version document compatibility, wherein, the apparatus 200 is in applications of each version. As shown, an application of version 3.0 generates or obtains in other manners a document M_3 of version 3.0. The conversion module 202 in the apparatus 200 of the present invention automatically converts the document M_3 of version 3.0 into a document M_2 of version 2.0 by using a differentiation model between the data model of version 2.0 and the data model of version 3.0 (supposing version 3.0 and version 2.0 are adjacent versions) in the conversion stack 201.

[0047] Optionally, the optional verification module 203 in apparatus 200 of the present invention can verify the converted document M_2 so as to make the document M_2 conform to the constraints in the data model of version 2.0. Since the data model of version 2.0 may lack some data elements as compared with the data model of version 3.0, thus when the document M_3 of version 3.0 is converted into the document M_2 of version 2.0, some data elements in the document M_3 of version 3.0 may be lost.

[0048] Therefore, the verification module 203 or the merging module 204 or any other module in the apparatus 200 of the present invention may store the document of version 3.0 or the relevant information therein (for example, the information lost when it is converted into the document of version 2.0) for a later merging operation, so as to restore the lost data elements.

[0049] Next, the conversion module 202 converts the document M_2 of version 2.0 into a document M_1 of version 1.0 (supposing version 2.0 and version 1.0 are adjacent versions) by using the differentiation model between the data model of version 1.0 and the data model of version 2.0.

[0050] Also optionally, the optional verification module 203 in the apparatus 200 of the present invention may verify the converted document M_1 so as to make it conform to the constraints in the data model of version 1.0; the verification module 203 or the merging module 204 may store the document M_2 of version 2.0 or the relevant information therein (for example, the information lost when it is converted into the document of version 1.0) for a later merging operation.

[0051] Then, the conversion module 202 or any other module in the apparatus 200 of the present invention sends the document M_1 of version 1.0 to an application of version 1.0 through a network, to allow the user who use the application of version 1.0 to modify the document M_1 , so as to generate a modified document M_1' of version 1.0, and send back the document M_1' to the application of version 3.0 through the network.

[0052] Next, the conversion module 202 automatically converts the modified document M_1' of version 1.0 into a document M_2' of version 2.0 by using the differentiation model between the data model of version 1.0 and the data model of version 2.0 in the conversion stack 201.

[0053] Also optionally, the optional verification module 203 in the apparatus 200 of the present invention can verify the automatically converted document M_2' to make it conform to the constraints in the data model of version 2.0. The merging module 204 merges the stored document M_2 of version 2.0 or the relevant information therein with the document M_2' of version 2.0, which was converted or further modified and verified, to obtain a modified complete document M_2'' of version 2.0.

[0054] Then the conversion module 202 converts the modified complete document of version 2.0 into a document M_3' of version 3.0 by using the differentiation model between the data model of version 2.0 and the data model of version 3.0 in the conversion stack 201.

[0055] Also optionally, the optional verification module 203 in the apparatus 200 of the present invention can verify the automatically converted document M_3' to make it conform to the constraints in the data model of version 3.0. And the merging module 204 can merge the stored document M_3 of version 3.0 or the relevant information therein with the converted or further modified or verified document M_3' of version 3.0 to obtain a modified complete document M_3'' of version 3.0, and provide the document M_3'' to the user as a final document. In this way, the entire operation process of the apparatus 200 providing inter-version document compatibility of the present invention is accomplished.

[0056] Of course, although the above example shows a scenario where one application version 2.0 exists between two application versions 3.0 and 1.0 which are used by two users respectively, the apparatus 200 of the present invention is applicable to situations where there exist any more or zero intermediate versions between the two application versions used by two users respectively.

[0057] The conversion module 202 can use any model conversion method known in the art, for example, QVT, ATL, VIATRA, GReAT, Tefkat, Kermeta, MT, SiTra, etc., to convert a document conforming to the data model of one version into a document conforming to the data model of another version according to corresponding inter-version differentiation models.

[0058] As for class elements in data models, if a differentiation model indicates a class element is added in a new data model as compared with an old data model, since the old document does not have the class element, the class element will not be taken into consideration when the old document is converted into a new document; if the differentiation model indicates that a class element has been deleted from the new data model as compared with the old data model, then when the old document is converted into the new document, the data segments of which the type is the deleted class in the old document need to be deleted; if the differentiation model indicates that the name of a class element in the new data model change as compared with the name of the class element in the old data model, then when the old document is converted into the new document, the class name of the old document needs to be replaced by the corresponding new class name.

[0059] As for attribute elements in the data models, if the differentiation model indicates that an attributes element is added in the new data model as compared with the old data model, since the attribute element does not exist in the old document, the attribute element needs not to be considered when the old document is converted into the new document; if the differentiation model indicates that an attribute element has been deleted from the new data model as compared with the old data model, when the old document is converted into the new document, the data segments of which the type is the attribute deleted in the old document need to be deleted; if the differentiation model indicates that the name of an attribute element in the new data model changes with respect to the name of an attribute element in the old data model, then when the old document is converted into the new document, the