Generation of personalized documents using DOM-objects in situation-oriented databases

Gusarenko A. S.
Ph. D., senior researcher.
FSEE HPE Ufa State Aviation
Technical University
450000, Ufa, st. Karl Marks, h. 12
phone +79174290226,
e-mail: artyomgusarenko@gmail.com

Submitted: August 28, 2013; Accepted: September 28, 2013;
Published: December 9, 2013. UDC 004.65
Subject MSC mathematical classification: 90C35, 90C39
Subject ACM computer classification: I.6.5

Abstract

The article uses the processing documents in XML-situation-oriented databases based on dynamic DOM-object for the task of generating personalized documents. The paper proposes a dynamic model SODB providing specifications and new elements to create the finished electronic documents based on the Office Open XML blanks prepared by means of a word processor. The dynamic model of the workpiece states are generated using data from the XML documents SODB on the web application server. The model in this article allows you to specify a routine operation to generate documents on a higher level of abstraction with model syntax SOBD. The article shows the graphical and textual equivalent assets put into a dynamic model and interpreted with the help of an interpreter embedded dynamic models for the solution of problem-chi document generation. Interpreter built dynamic models work with documents Office Open XML, using the technology of dynamic DOM-objects, with the resulting Document to-have options and presentation docx vdx. The paper discusses ways of specifying the source and processing of XML-based data dynamic DOM-place is a state of dynamic models. It is proposed to implement the model on a platform of PHP using blanks docx and vdx.

Keywords: NoSQL, interpretation, XML databases, dynamic model.

1 Introduction

Developing the concept of processing XML-documents situation-oriented databases (SODB) based on dynamic DOM-objects proposed in (Mironov & Gusarenko, 2012a; Mironov & Gusarenko, 2012b) it is required to solve the problem of generating electronic documents. Electronic documents are the basis for document-oriented information systems shall contain personalized data for use in the information system and issue them to the user in the form of the resulting document. The document-oriented systems based SODB often required to solve the problem of generating ready documents, indirectly through a web application (Benzaken et al., 2013; Takahashi et al., 2013; Kobayashi et al., 2011; O'Connor, Merriman & Nelson, 2010) of XML-data contained therein and stored on the server.

And to create such documents (Mironov & Gusarenko, 2012; Mironov & Gusarenko, 2013a) need not entirely, from the stage of placing it in the server's file system with the establishment of registration and alignment of its internal structure, but only the content data and the enrichment of meaningful content with the personalized user information. Thus, the problem of generating (Mironov & Gusarenko, 2012) a simplified looks like the creation of an electronic document previously prepared by conventional means, such as a word processor, and subsequently filling it in the information system means SODB (Strauch & Kriha, n.d.; Mironov & Gusarenko, 2012b; Mironov & Gusarenko, 2012a; Gusarenko, 2013; Mironov & Gusarenko, 2013a; Mironov & Gusarenko, 2013b).

Harvesting is adjusted during the generation of the document not being completed design, as already incorporated in the design of the workpiece, and personalized user data needed. The blank document is a file format Office Open XML, it is an archive in which the contents of the document is divided into several files in XML, each of which contains the specification design document and the information contained in the document. Examples of data sources for SODB format Office Open XML documents are docx and vdx. In its structure, it is an archive containing a file XML, which can be processed in SODB with technology of dynamic DOM-objects.

Dynamic DOM-objects are filled XML-data and loading them into the XML-documents for processing SODB. With such data, the paper describes the objects and hierarchical data structures. To solve the problem of generating documents interpreter embedded dynamic models requires a specially programmed functionality, and dynamic model of special linguistic software for the dynamic model SODB.

In (Mironov & Gusarenko, 2013a) deals with the use of data sources to create content and its transformation to issue results to the user. It was mostly the contents of web pages transformed with XSLT, issues of creating the resulting documents were not considered, so you need a special tool SODB implements the declared functionality.

This article discusses the concept of automatically generating content from XML-database data and documents blanks Office Open XML, associated with the states of the dynamic model SODB, as well as specifications to work with blanks necessary to equip a dynamic model of DOM-elements.

2 Using structure of Open Office XML documents

This article is an extension of work on the creation of tools, linguistic tools SODB (Mironov & Gusarenko, 2013a), which were developed by means of the specification of dynamic DOM-objects. When generating a structure of Office Open XML document should be considered hierarchical contents of the package archive document. The content of the archive can be seen if the newly created document file extension to assign zip, and then open the archiver. Open the file containing the hierarchical structure of the files Fig. 2.1 responsible for its content. Main content of the text file is placed in the XML-file document.xml. The structure of the document shown in Fig. 2.1 in the form of a graphical model has a standard set of files and directories. All content is divided into parts, according to functional purpose. Element connections package contains XML-record communications with other parts of the document, such as file properties of the core document and file core.xml software-defined properties app.xml. The master document and its contents are represented in a file document.xml, mainly text contained in the document. For constructing document footers and headers you want to edit files and footer1.xml header1.xml respectively, as they are all numbered files with the name of the header and footer will have serial numbers.

Part numbering.xml archive contains a definition of numbered paragraphs of the document, as well as styles for them. Document settings, including stored in the preset file settings.xml. All the files inside the archive documents have XML-only content so there is a possibility of using the technology of processing XML-documents archive with dynamic DOM-objects are loaded to the workpiece by parts and enriching it with personalized data from SODB.

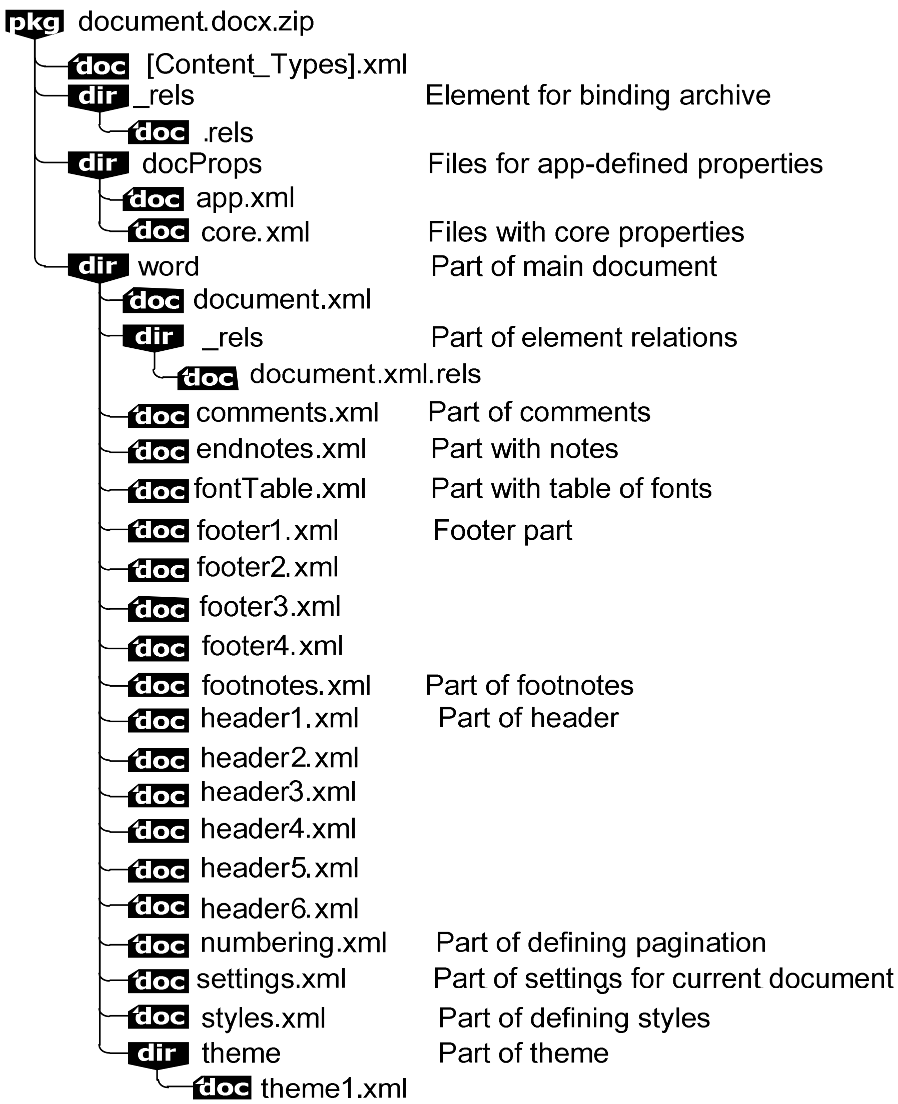

Fig. 2.1 Internal structure of Office Open XML

Moreover, using the same technology (Mironov & Gusarenko, 2012b) will be possible to generate both documents docx, and documents vdx. Widely used method of programming such documents manually (Mironov & Gusarenko, 2013a), that is, for each document operations are programmed separately in a programming language application. In this situation, to reduce the amount of routine processing of parts of the document is required to equip the model of DOM-elements special means constructing parts of documents (Figure 2.1) data from SODB processed in dynamic DOM-objects.

Thus, in the initial preform provides special bookmarks w:bookmarkStart and w:bookmarkEnd to contain personalized data. Bookmark w:bookmarkStart used to indicate the beginning of data, and w:bookmarkEnd end of the field placement. Bookmarks from the workpiece are specified in the model in order to fill them with content. The model is required to specify the linguistic expressions to modify the contents of bookmarks and other areas in which to place the data. It also requires the specification of variables, which will be personalized data stored and DOM-objects in which they are processed. The approach focuses on the use of models to generate blanks involves the increased use of data sources, as well as the expansion of specifications for their determination in the model update method.

The dynamic model of the specification attached to the states DOM-objects and nested definition of data sources, along with the specifications of operations for generating units of the workpiece. Also provided within the state variables for personalized user data used in the DOM-objects. These specifications automatically processes the interpreter embedded dynamic models (He & Zhai, 2013; Han, et al., 2014; Pokorný, 2013; Kaur & Rani, 2013) on the basis of this DOM-objects and processed blanks in them are being completed and is automatically personalized based on the data stored in the variables associated with the state of the model.

Manual programming does not allow to use the model to improve the level of abstraction using a typical operations generate, retrieve and filter data when processing Office Open XML-blanks, and output the result in the form of the resulting document. In this case it is reasonable to equip these functions SODB model.

The proposed approach to the states of the dynamic model are related:

• DOM-elements in a dynamic model has children of sources that regulate the use of the method of data processing, as well as set the target node and the updated value, and the children of the receivers - the method of preparation and preservation of the archive in the target file of the archive;

• in the course of interpreting the dynamic model of the interpreter automatically DOM-generation facilities and the implementation of specifications for embedded sources and loading XML-data in accordance with a given query in their recovery, delete, update, maintenance and filtration.

DOCX-documents. For the collection and processing of personal data required global variables to define them using the symbol glb, which indicates the value specification interpreter which data session variable to use in a state of dynamic model SODB. Blanks docx Fig. 2.2 loaded in the interpretation of the model and filled with personalized data from the DOM-object student who requested from SODB Fig. 2.2, and then sent to the user Fig. 2.3, is used for this transition denoted jmp after pressing the button, the generation of the resulting document and send it to the browser.

VDX-documents. They have an internal representation of XML, but are not an archive with the structure similar to Fig. 2.1. However, it is the same XML-content you want to fill in personalized data. Billets documents vdx Fig. 2.3 is also associated with a state of dynamic models SOBD for their generation requires loading the internal structure of the workpiece in the DOM-object (Mironov & Gusarenko, 2012a; Mironov & Gusarenko, 2012b). Models are required for the specification means blanking operations vdx, such specification uses a model figure. 2.3. In the pre-prepared documents placed bookmarks. Bookmark, which is similar to docx documents are containers for user data. The example model in Fig. 2.3 finished document is sent to the client browser when you click on the button Result. The new format Visio 2013, vsdx is an archive from XML-files, but in this article are not considered.

Bookmarks. Bookmarks in the document-procurement must be prepared in advance in the media Word to paste in their place user data. In order to manually process each document you want to use a bookmark specification model to automatically populate their personalized data. When processing model interpreter dynamic models SODB bookmarks associated with a data (Kobayashi, et al., 2011) source are automatically filled. Attributes describe the bookmarks in the source of data and have the form

<w:bookmarkStart w:id="0" w:name="Name bookmark" />

Fig. 2.4. To handle these specifications should be added to the interpreter functionality to fill in the DOM-objects tab, which allows you to create these objects are filled when the parent becomes the current state and delete the contents of bookmarks, when the state ceases to be current. This will reduce the complexity of the preparation of documents in the DOM-objects when processing XML, and set the standard transformation operations XML-data for further processing to specify procedures for processing XML.

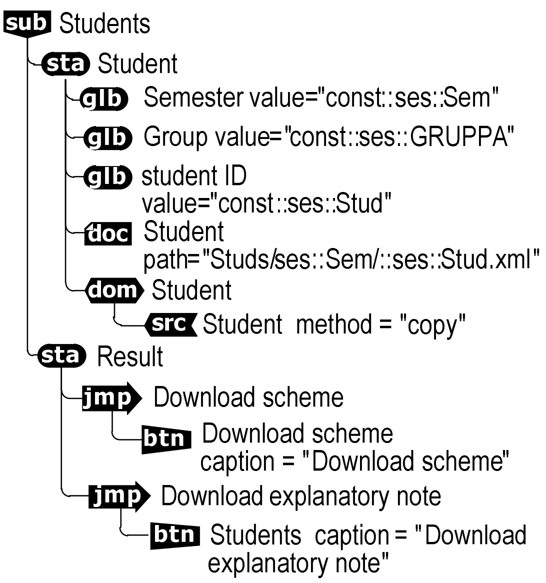

Fig. 2.2. diagram of dynamic model with control elements in which documents are generated

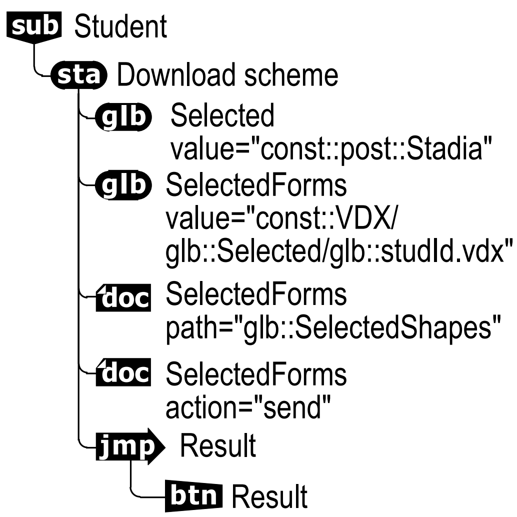

Fig. 2.3. The diagram of the dynamic model with the states, which are associated DOM-objects with the specifications generate Office Open XML documents

Fig. 2.4. Diagram of the dynamic model in the "Upload explanatory note", which are associated dynamic DOM-objects with the specifications generate personalized Office Open XML document "Explanatory Note"

3 Concept generation of personalized documents using DOM-objects

Target nodes targNode and values updateValue. The models describe the source, previously introduced for the sources of dynamic DOM-objects (Mironov & Gusarenko, 2012b) add construction jobs target node of the resulting document, which appeal to the tab by name,

targNode = "//w:r[preceding-sibling::w::bookmarkStart/@w:name='тема'][1]/w:t"

It gives access to it, and the value

updateValue = "dom::Stud#//TOPIC/NAME"

the value is inserted in its place, in which the interpreter inserts personalized content. Further implementation of the idea to reduce the complexity of the issues require further study to ensure that the data sources(Dekeyser, Hidders & Paredaens, 2004; Herskovits & Chen, 2008; Jea, Chang & Chen, 2009; Kudrass & Conrad, 2002; Nassis, Dillon, Rajagopalapillai & Rahayu, 2006; Batory, 2006; Dejanovic, Milosavljevic, Perisic & Tumbas, 2010).

Methods of modifying the workpiece and updateNode removeNode. In order to be able to modify and adjust (Mironov & Gusarenko, 2012b) the workpiece models require special machining techniques. These methods define the mode of operation of the interpreter with XML-content. Minimum necessary methods introduced in the model, referred to by functional purpose. Method updateNode updates the target node, the value specified in targNode updateValue.

For disposal or purification required nodes preform is possible that the resultant preform requires not only that a paste, and delete the document to bring its original state (Dekeyser, Hidders & Paredaens, 2004; Herskovits & Chen, 2008; Jea, Chang & Chen, 2009; Kudrass & Conrad, 2002; Nassis, Dillon, Rajagopalapillai & Rahayu, 2006; Batory, 2006; Dejanovic, Milosavljevic, Perisic & Tumbas, 2010).

The result is output and the save zipEntry resulting document. To specify the interpreter of the archives, which should be updated, it requires specification of the archive to display the XML-content. The model indicated attribute zipEntry and prescribed path to an XML file in the archive workpiece you want to update. To adjust the blanks, a special buffer server that is cleaned and updated using methods and updateNode targNode.

With such zipEntry buffer can make the blank document. In such circumstances, SOBD required to equip these specifications toolbox item Wola process and store XML-documents into dynamic DOM-objects are in memory and external memory (Djukic, Lukovic, Popovic & Ivancevic, 2013; Panach, Juristo & Pastor, 2013; Strosnider, Nandi, Kumaran, Ghosh & Arsanjani, 2008).

4 Example a generation of personalized documents

To illustrate the proposed concept is invited to consider a practical example of the use of technology of dynamic DOM-objects to solve the problem of generating personalized documents explanatory note, similar to that given in (Mironov & Gusarenko, 2012a; Mironov & Gusarenko, 2012b) with information about the student from SODB.

Initial data. Data on students are stored in numbered files, XML - in our example would be a document without Studenty.xml rooms that are located in the XML server SOBD. Fig. 4.5 shows the conceptual models of these documents and test instances of the content. Document Studenty.xml (Fig. 4.5a) in the root object "Students" can contain multiple nested instances of "student" with the attributes of the "code" (Student ID) and "name" (first name, middle name of the student), and attribute "Code" is an identifier. Document (Fig. 4.5a) contains information about the student. Similarly, the XML-document Temy.xml (Fig. 4.5b) in the root object "Themes" can contain multiple nested instances of "theme" with attributes "theme room" (code theme) and "Item Name" (the name of the topic) , the attribute "room theme" is an identifier. The model of the document fragment (Fig. 4.5b) contains information about the functions and units in accordance with the job. Document zagotovka.xml (Fig. 4.5c) in the root object, "the document body" may contain multiple copies of attachments with personalized data corresponding to the student with the attributes of "Artist name", "Group" and "specialty code", as well as personalized information issued by the students asked, of. Model copy of the document (Fig. 4.5c) contains information tab identifier rsidRpr, these data are the building and include a year course, semester, specialty code, the name of the subject, type of work, group and student record book. This example is the formation of the document blank (Fig. 4.5) on the basis of information and documents about the student job loaded in the DOM-objects from the XML-documents SOBD. The initial state of the request form displays the names of students for whom using the DOM-object data is requested for harvesting. The result is displayed in the browser as a document of the explanatory note. The test specimen (Fig. 4.5) shows how the interpreter enriches the blank with the name of university theme. Department of Labour Protection and date of issue of the document setting student Studenty.xml.

Dynamic model. The basis of the application is a model of a new level of abstraction, which is equipped, the new specifications. The dynamic model of the application was considered in, the main distinguishing features of a web application model considered in this paper is to set machining specifications of XML documents using elements "dom", before using this structure described just XML-encoded into external memory server SODB. Introduced new specifications in the blanks of type bookmark to insert personalized user data DOM-objects. Examples of such tools are shown in Fig. 4.5 in the "Upload an explanatory note" Structure denoted bookmark, tells the interpreter that in the preform, the value "//w:r[preceding-sibling:: w::bookmarkStart/@ w:name = 'theme'] [1]/w: t" in substate " Student "submodels" Students "requires the insertion of DOM-object values updateValue" dom::Stud#//TOPIC/NAME "using the method updateNode, which is in memory in the DOM-object" target document "adds DOM-object of" Student "student group number and the name of the theme.

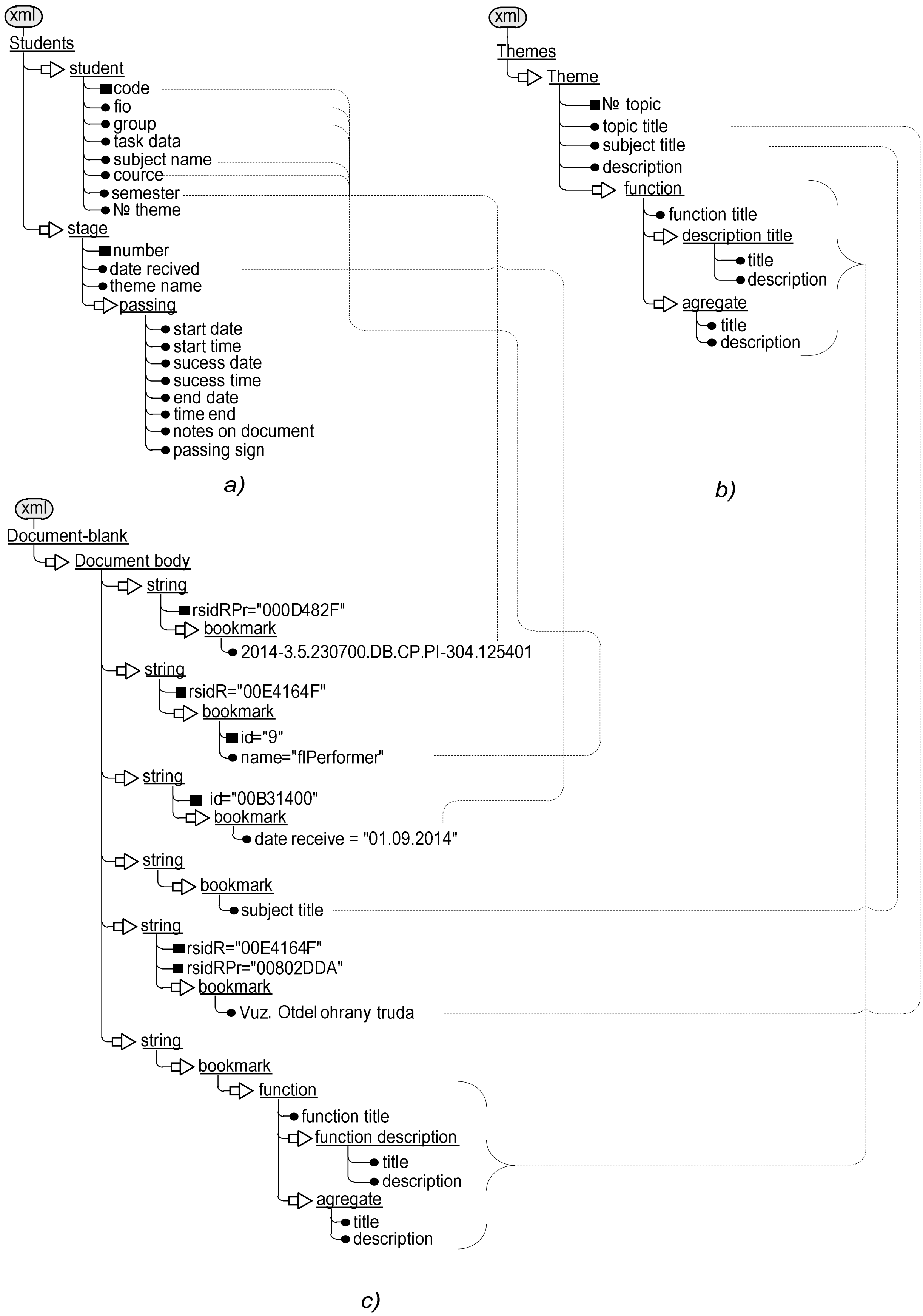

Fig. 4.5. The diagram of data structures dynamic DOM-objects in the "Upload explanatory note": *a* - a data document a student; *b* - information about the job; *c* - Office Open XML personalized blank document

Each state of the dynamic model SOBD global variables can be denoted glb, in addition, they may be of different origin, such as requested from DOM-object "Student" variable "Identifier". To obtain the results required user controls specifiable in the model, so-called transitions jmp take advantage of them inside buttons and widgets. Further work will consist in the development of user interfaces for the resulting documents.

Conclusion

In this article we consider the problem of generating documents in Office Open XML situationally-oriented databases of pre-prepared pieces created by conventional means Word and Visio, and also introduced a dynamic model means the job processing operations of the internal contents of the document class, workpieces required for their personalization. It formed a concept that is to use the model of a higher level of abstraction to equip applications using state of technology in the dynamic DOM-objects for processing XML-data from data sources SODB. The examples used in the model and are considered without the specific implementation of programming technologies.

The concept is based on the technology of dynamic DOM-objects associated with the states of the dynamic model SODB, with the creation of Office Open XML documents in the course of the interpretation given to the loading and processing is carried out with the conditions set automatic. The final document is sent to the user's web browser.

The main distinguishing features of this concept are:

1) the elements of the state of the dynamic model of situation-oriented database containing the associated DOM-objects with the specifications of processing bookmarks in a prepared blank document;

2) in the process of interpretation of the dynamic model of the objects are automatically loaded XML-content, then in accordance with the specified processing conditions formed the resulting personalized documents;

3) dynamic model supplemented with additional means for connecting and processing Office Open XML blanks in situationally-oriented database;

4) the model uses two kinds of methods to modify and customize the document-blank updateNode and removeNode;

5) the model incorporated the specifications and targNode updateValue necessary to specify the target node to automatically update the values of dynamic models SOBD interpreter;

6) introduced a model specification zipEntry to enrich a particular part of the Office Open XML archive of personalized data.

In accordance with this concept, a specialist will be possible without any manual programming to create applications that connect in a state of dynamic models Office Open XML sources and specify the conditions for their processing to produce personalized documents. Further development and implementation of the concept is related to software platform HSMI, to handle dynamic model HSM SODB.

Acknowledgments

This work was supported by RFBR grant 10-07-00167.

References

Batory, D. (2006). Multilevel models in model-driven engineering, product lines, and metaprogramming. *Ibm Systems Journal*, *45*(3), 527–539.

Benzaken, V. (2013). Static and dynamic semantics of NoSQL languages (pp. 101–113). Presented at the 40th Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages POPL’ 2013, Rome, Italy.

Chen, Y. J., & Huck, G. (2001). On the evaluation of path-oriented queries in document databases. *Database and Expert Systems Applications*, *2113*, 953–962.

Dejanovic, I., Milosavljevic, G., Perisic, B., & Tumbas, M. A. (2010). Domain-Specific Language for Defining Static Structure of Database Applications. *Computer Science and Information Systems*, *7*(3), 409–440.

Dekeyser, S., Hidders, J., & Paredaens, J. (2004). A transaction model for XML databases. *World Wide Web-Internet and Web Information Systems*, *7*(1), 29–57.

Djukic, V., Lukovic, I., Popovic, A., & Ivancevic, V. (2013). Model Execution: An Approach based on extending Domain-Specific Modeling with Action Reports. *Computer Science and Information Systems*, *10*(4), 1585–1620.

Gusarenko, A. S. (2013). *Handling XML-documents in situationally-oriented databases based on dynamic DOM-objects* (thesis abstract). Ufa State Aviation Technical University, Ufa.

Gusarenko, A. S., & Mironov, V. V. (2013a). Algorithmic and and linguistic support of situationally-oriented databases based on dynamic DOM-objects (Vol. 1, pp. 119–122). Presented at the Proc. 8th Workshop of winter school postgraduates and young researchers, Ufa: USATU.

Gusarenko, A. S., & Mironov, V. V. (2013b). Profiling algorithm of situationally-oriented database based on dynamic DOM-objects (Vol. 1, pp. 115–118). Presented at the Proc. 8th Workshop of winter school postgraduates and young researchers, Уфа: УГАТУ.

Han, W. (2014). Leveraging spatial join for robust tuple extraction from web pages (Vol. 261, pp. 132–148). Presented at the Information Sciences.

Herskovits, E. H., & Chen, R. (2008). Integrating Data-Mining Support into a Brain-Image Database Using Open-Source Components. *Advances in Medical Sciences*, *53*(2), 172–181.

He, W., & Zhai, J. (2013). Application of the indent conversion based on XML and DOM (Vol. 1, pp. 411–413). Presented at the Proceedings - 2013 International Conference on Computational and Information Sciences (ICCIS’ 2013).

Jea, K. F., Chang, T. P., & Chen, S. Y. (2009). A Semantic-Based Protocol for Concurrency Control in DOM Database Systems. *Journal of Information Science and Engineering*, *25*(5), 1617–1639.

Kaur, K., & Rani, R. (2013). Modeling and querying data in NoSQL databases (pp. 1–7). Presented at the Proc. of IEEE International Conference on Big Data 2013, Santa Clara, CA, United States.

Kobayashi, N., Ishii, M., Takahashi, S., Mochizuki, Y., Matsushima, A., & Toyoda, T. (2011). Semantic-JSON: a lightweight web service interface for Semantic Web contents integrating multiple life science databases. *Nucleic Acids Research*, *39*,

W533–W540.
Kudrass, T., & Conrad, M. (2002). Management of XML documents in object-relational databases (Vol. 2490, pp. 210–227). Presented at the Xml-Based Data Management and Multimedia Engineering-Edbt 2002 Workshops.
Mironov, V. V., & Gusarenko, A. S. (2012a). Dynamic DOM-objects in the situationally-oriented databases: a linguistic and algorithmic support of data sources. *Vestnik UGATU*, *16*(3), 167–176.
Mironov, V. V., & Gusarenko, A. S. (2012b). Situationally-oriented databases: the concept of managing XML-data based on dynamic DOM-objects. *Vestnik UGATU*, *16*(3), 159–172.
Nassis, V., Dillon, T. S., Rajagopalapillai, R., & Rahayu, W. (2006). An XML Document Warehouse model (Vol. 3882, pp. 513–529). Presented at the Database Systems for Advanced Applications, Proceedings.
O'Connor, B. D., Merriman, B., & Nelson, S. F. (2010). SeqWare Query Engine: storing and searching sequence data in the cloud. *Bmc Bioinformatics*, *11*, 9.
Panach, J. I., Juristo, N., & Pastor, O. (2013). Including Functional Usability Features in a Model-Driven Development Method. *Computer Science and Information Systems*, *10*(3), 999–1024.
Pokorný, J. (2013). New database architectures: Steps towards big data processing (pp. 3–10). Presented at the Proceedings of the IADIS International Conference Intelligent Systems and Agents 2013 ECDM' 2013, Prague, Czech Republic.
Sladic, G., Milosavljevic, B., Konjovic, Z., & Vidakovic, M. (2011). Access Control Framework for XML Document Collections. *Computer Science and Information Systems*, *8*(3), 591–609.
Strauch, C., & Kriha, W. (n.d.). *NoSQL databases*. Retrieved from http://www.christof-strauch.de/nosqldbs.pdf
Strosnider, J. K., Nandi, P., Kumaran, S., Ghosh, S., & Arsanjani, A. (2008). Model-driven synthesis of SOA solutions. *Ibm Systems Journal*, *47*(3), 415–432.
Sudarsan, R., & Gray, J. (2006). Metamodel search: Using XPath to search domain-specific models. *Journal of Research and Practice in Information Technology*, *38*(4), 337–351.
Takahashi, K. (2013). Design and evaluation of lifelog mashup platform with NoSQL database (pp. 133–139). Presented at the Proc. of 15th International Conference on Information Integration and Web-Based Applications and Services, iiWAS'2013, Vienna, Austria.
Tang, N., Yu, J. X., Wong, K. F., & Li, J. X. (2008). Fast XML structural join algorithms by partitioning. *Journal of Research and Practice in Information Technology*, *40*(1), 33–53.