HERMES: a Tool for Querying Heterogeneous Data Stores

Euclid Keramopoulos[1], Dimitris Dervos[1], Ignatios Deligiannis[1]
[1]Dept. of Information Technology
Alexander TEI of Thessaloniki,
Thessaloniki, Greece
(euclid, dad, ignatios)@it.teithe.gr

Georgios Evangelidis[2], Philippos Pouyioutas[3]
[2]Dept. of Applied Informatics, University of Macedonia,
Thessaloniki, Greece
gevan@uom.gr
[3]Dept. of Computer Science
University of Nicosia Nicosia, Cyprus
pouyioutas.p@unic.ac.cy

Abstract — **Data are organized/stored in a plethora of different databases, locations and data models. In many cases, companies and organizations utilize data stored in a variety of data models at remote database servers. A challenge would be to construct queries that join data from such disparate sources. In this paper, we report on research work relating to graphical query languages and data modeling. More specifically, we report on a new software tool for the novice user, one that transforms databases of different data models (Relational, Object-Relational, Object-Oriented) into an equivalent XML model, plus on a new graphical XQuery type query language that facilitates the construction of user queries that join data from different data models.**

Keywords — Graphical Query Languages, Relational Database Model, Object-Relational Database Model, Object-Oriented Database Model, Semi-structured Database Model, XQuery

I. INTRODUCTION

In the last three decades, data are organized and stored in various types of data stores, like: Relational [1], Object-Relational [2], Object-Oriented [3], XML (Extensible Markup Language) [4], RDF (Resource Description Framework) [5] or even in free text format. Moreover, real life applications call for the integration of data stored in different types of remote database servers, implementing different types of database models. In this respect, we have not been able to identify works in the international literature reporting on query languages that facilitate the construction of queries which process data from heterogeneous data sources, like Relational, Object-Relational, Object-Oriented and XML databases.

XML [4] is a markup language developed by the World Wide Web Consortium (W3C) [6] aimed to deliver structured content over the web and to provide a competitive way of storing data. Also, XML tends to become a standard way of data interchanging on the Web. Thus, XML can be used as a solution in joining data from different sources.

Moreover, XQuery [7] is the W3C standard to retrieve information from XML documents. Nevertheless, its syntax is complicated and demanding especially for novice users. The majority of computer users need only learn how to complete simple tasks, whereas the problems they have to solve are usually expressed in non-computing terms. Nowadays, the typical user has changed from the skilled professional to the computer literate (unskilled or novice) user, and thus there is a need, the user interface to be simpler and friendlier. The initiation of graphical user interfaces utilizing cognitive skills, advances in graphics technology, and increased computing power, has simplified and improved the way humans interact with computers. Consequently, computer systems accessibility has improved. Today, graphical user interfaces comprise an essential part of any computer system and system designers have come to accept that in order to improve users' productivity it is essential for a user interface to address users' skills [8]. Thus, a number of graphical interfaces for XML query languages have been developed.

In this paper, we report on the design of a new tool, HERMES which is based on XQuery and facilitates the construction of queries targeting Relational, Object-Relational, Object-Oriented and XML data sources. Queries may involve joining data from heterogeneous data sources. HERMES consists of (a) a mechanism to transform a database of Relational, Object-Relational and Object-Oriented database system into an XML database, and (b) a graphical query language for expert and novice users to XQuery, namely KINISIS [10].

Section II comprises an overview on Query Languages (textual and graphical) for XML data. In section III, we present the design, implementation and evaluation of HERMES. Finally, in section IV we conclude and comment on what is next to be done.

II. OVERVIEW OF QUERY LANGUAGES FOR XML DATA

We are aware of at least twenty different query languages for XML data that have been proposed. The first one to be introduced was a query language for semistructured data called Lorel [11,12], later extended for XML data. Lorel was designed for the Lore DataBase Management System. Another language originally developed for semi-structured data is Tree Query Language (TQL) [13, 14], which utilizes match-filter operations to describe XML documents as labeled trees. XQL [15] is based on path expressions similar to XPath where XML-QL [16] uses a "WHERE" construct syntax. XML Matching and Structuring Language (XMAS) [17] deals with semi-structured data and uses a "WHERE" construct syntax. CDUCE [18], which is an extension of XDuce [19], is a programming language based on matching patterns using a "SELECT-FROM-WHERE" syntax. Another interesting language for retrieving XML data is CQL [20]: one with a rather strict and complicated syntax. Tequyla-TX [21] is a typed text retrieval query language for XML documents. DQL [22] is an extension of OQL for the manipulation of tree-like and forest-like data. Quilt [23] is an XML Language for heterogeneous data sources. In fact, it is

a functional language in which a query is represented as an expression. XML-SQL [24] is an XML Query Language based on SQL and path table. XJ [25] and LINQ [26] are languages for searching XML data in a Java environment. An interesting approach is the one presented in [27, 28, 29, 30], where a query is defined using XSemantics Nets on an XML Document Warehouse. World Wide Web Consortium created (a) XQuery 1.0 [7], as the standard query language for XML data which is also used by a number of DBMS's [31], and (b) XSLT 2.0 [32], as the standard transformation language for XML data. Both languages embed XPath 2.0 [33], which utilizes path expressions to retrieve data from XML Documents. XQuery 1.0 and XSLT 2.0 are supersets of XPath 2.0 and all three languages share the same data model, having a common set of functions and operators. XML-GL [34, 35] is a graphical query language that depicts documents and their related DTDs with the use of graphical representations. XQuery By Example (XQBE) [36] is a graphical query language based on XML-GL. In fact, XQBE implements XQuery 1.0 queries with graphical representations. Xing [37] is yet another graphical query language, as well as visXcerpt [38, 39, 40, 41], which is an extension of Xcerpt [39, 40]. In [9], an analytical review of graphical interfaces for XML query languages is presented.

III. THE HERMES APPROACH

HERMES is graphical user interface based tool used for constructing queries that join data across heterogeneous data sources and models. HERMES consists of:

(a) A mechanism that transforms non-XML (e.g. Relational, Object-Relational and Object-Oriented) database model instances into an XML one, and

(b) A graphical query language interface called KINISIS that utilizes road traffic sign metaphors in order to facilitate the construction of XQuery syntax by the novice user.

A. Transforming Data Models into a unified XML type

Although the evolution of the Internet has been remarkable, large volumes of data are still organized in traditional data stores, like relational and object-relational DBMS's. Moreover, during last ten years object-oriented databases are increasingly used for storing information.

Our research effort focuses on the creation of a query mechanism that can be applied to data stored in a relational, object-relational, object-oriented, or a semi-structured (XML) database as well as in distributed realizations involving more than one of the stated DB models. We chose to exploit the potential of XML as a data transfer tool. Thus, we created mechanisms, one for each model, to convert a data model (relational, object-relational, object-oriented) to the corresponding data in XML format, without losing information.

The relational model involves entities and relationships implemented as tables (relations) in the corresponding database schema. There are several algorithms in the literature that have been developed for mapping the relational model to XML, for example: ConvRel [42], Holistic Constraint-Preserving Transformation Algorithm [43], evolution [44], SQL-based algorithms [45], and SilkRoute [46]. In HERMES, an XML file is created for each type of table. The *root* tag of the XML file is the name of the table and in separate tags come the names of the columns of root table.

The object-relational model extends the relational model by employing object-oriented features. Main elements of the object-relational databases (ORDB) are the support of creation of new user types (class) and objects. Also, ORDBs support inheritance, collections and references (relationships). All major DBMS (Oracle, IBM DB2, Microsoft SQL Server, Postgres, etc) support SQL3, so they are characterized as object-relational. In the literature, only a few algorithms have been proposed relating to the transformation of an ORDB schema to XML. The key algorithm is XPERANTO [47]. XPERANTO supports the transformation of relational and object-relational data structures into XML format; structures like tables (Typed tables - objects), columns (Typed columns) references (relationships), inheritance and collections. In HERMES, a separate XML file is created for each one object (typed table). The *root* tag of the XML file is the name of the table and in separate tags come the names of the columns of root table.

The object-oriented database model has its origins in object-oriented programming paradigm, while offering all the basic features of the traditional databases. In the literature only a few algorithms have been proposed regarding the conversion of an OODB to XML. The most important one is [48], where in the first step an object graph is created which consists of the object-oriented schema metadata. Next follows the mapping of the actual data from the object-oriented database into corresponding XML document(s). In HERMES, we have decided to use XSTREAM API [49] for the transformation of an OODB to XML files.

Having transformed any one of the aforementioned DB models into the corresponding XML file(s), we can proceed and formulate an XQuery construct, utilizing the graphical query interface of KINISIS. In the next section we present KINISIS, a graphical XQuery language, designed to be used by expert and novice users.

B. KINISIS, a Graphical XQuery Language

KINISIS [10] is a graphical query language which is designed on top of XQuery. It supports all the XQuery features and it is represented graphically by a set of basic road traffic sign metaphors. The latter have been chosen because road traffic rules are well known so one can easily extend the paradigm to imply XQuery command syntax semantics.

We have completed the implementation of KINISIS as an end-tool using Java and the Java Swing API. KINISIS supports query construction via a Graphical User Interface (GUI) with a user-friendly approach utilizing drag and drop operations. The GUI was designed for users to work on an XQuery Flower philosophy, with one difference: all the

tricky syntax of XQuery is replaced/hidden by a set of Basic road sign metaphors.

Moreover, a compiler was developed. One which transforms the user-constructed graphs into the equivalent XQuery syntax. The latter, once constructed, is directed to the underlying IBM DB2 RDBMS XQuery parsing and execution manager.

KINISIS functionality is best explained by means of two examples. The first example represents a simple and straightforward case. The second example is a bit more involved since it represents a case where two XML files are joined to produce the sought for answer: XML files of the type HERMES constructs easily.

Proceeding with the two cases outlined in the previous paragraph, we used the following running example, taken from [50]. The running example makes use of two XML files: (a) "catalog.xml" which lists product details for every department of the catalog, and (b) "prices.xml" which lists all product prices on a certain date.

catalog.xml

```
<catalog>
 <product dept="WMN">
  <number>557</number>
  <name language="en">Fleece Pullover</name>
  <colorChoices>navy black</colorChoices>
 </product>
 <product dept="ACC">
  <number>563</number>
  <name language="en">Floppy Sun Hat</name>
 </product>
 <product dept="ACC">
  <number>443</number>
  <name language="en">Deluxe Travel Bag</name>
 </product>
 <product dept="MEN">
  <number>784</number>
  <name language="en">Cotton Dress Shirt</name>
  <colorChoices>white gray</colorChoices>
  <desc>Our <i>favorite</i> shirt!</desc>
 </product>
</catalog>
```

prices.xml

```
<prices>
 <priceList effDate="2006-11-15">
  <prod num="557">
   <price currency="USD">29.99</price>
   <discount type="CLR">10.00</discount>
  </prod>
  <prod num="563">
   <price currency="USD">69.99</price>
  </prod>
  <prod num="443">
   <price currency="USD">39.99</price>
   <discount type="CLR">3.99</discount>
  </prod>
 </priceList>
</prices>
```

Example 1.

The case involves a query that retrieves all the product names associated with Accessory (ACC) departments, listed alphabetically. Fig. 1 presents the graphical version of the query in the KINISIS GUI.

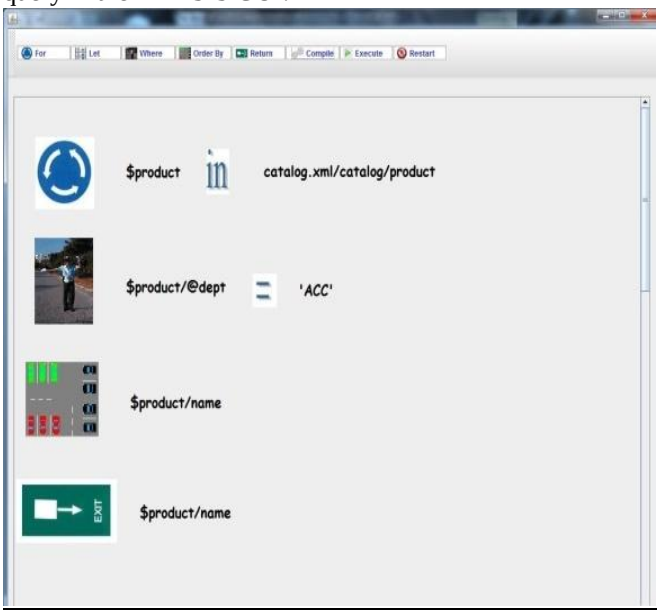

Figure 1. Example 1 in KINISIS

The graphical query in Figure 1 corresponds to the following XQuery syntax:

```
for $product in doc("catalog.xml")/catalog/product
where $product/@dept='ACC'
order by $product/name
return $product/name
```

Example 2.

Case number 2 involves the joining of two files in order to calculate and list each one product number alongside with its price. Product data are registered with "catalog.xml" and product prices appear in "prices.xml". The two XML files are joined on the clause: "$product/number = $price/@num" to produced the desired output.

Fig. 2 presents the graphical version of the query in the KINISIS GUI.

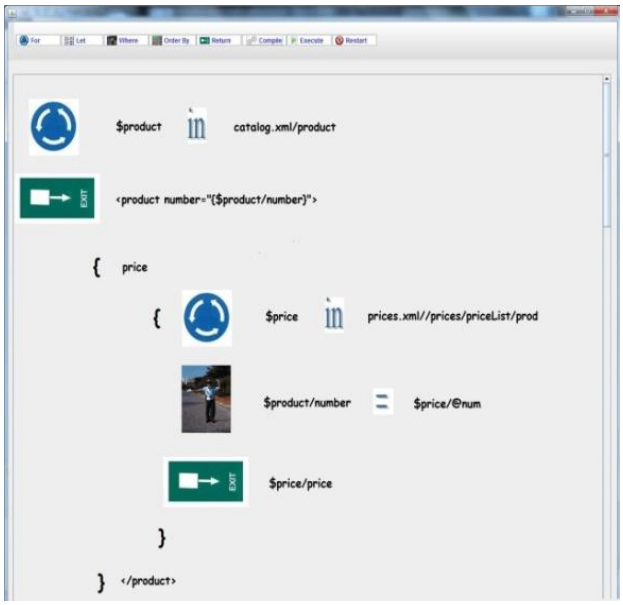

Figure 2. Example 2 in KINISIS

The graphical query in Figure 2 corresponds to the following XQuery syntax:

```
for $product in doc("catalog.xml")//product
return <product number="{$product/number}">
{ attribute price
 { for $price in doc("prices.xml")//prices/priceList/prod
  where $product/number = $price/@num
  return $price/price}
} </product>
```

In KINISIS, in order to represent the *join* operation, the XQuery structure is followed by using two FLWOR's, one embedded in the return clause of the other. The outer FLWOR returns the list of products, regardless of the availability of price information. The inner FLWOR selects the price, if it is available.

C. Evaluation of the language

We conducted a controlled experiment [51] in order to compare the quality factor understandability between KINISIS and XQuery. Twenty two (22) undergraduate students of the Department of Information Technology at the Alexander Technology Educational Institute (TEI) were asked to construct a number of queries using both XQuery and the KINISIS.

The execution of the controlled experiment completed with no problems. The conclusion was that (a) the KINISIS GUI query formulation procedure was found to be noticeably easier to comprehend and be used by the students, as compared to native W3C XQuery command syntax, and (b) query formulation in the KINISIS GUI was found to be less error prone at query construction time. More specifically, the students responded with significantly more correct answers when they were using KINISIS (80.91%, 89 correct answers from 110 queries in total) as compared to when they used the native XQuery command syntax (51.82% correct answers). The students have found the KINISIS application to be friendlier to use since they had to "draw" the query on the GUI, instead of having to construct its textual version in the IBM DB2 provided command editor window. KINISIS has made it easier for the students to compose their queries, making fewer mistakes (e.g. spelling, or entering an invalid path expression). As a consequence, KINISIS queries turned out to represent an 80.91% success ratio, next to only 51.91% of the native W3C XQuery queries.

IV. CONCLUSION

In this paper, we presented a new software tool, HERMES, which makes possible the querying of Relational, Object-Relational, Object-Oriented and semi-structuded (XML) databases, using one single user interface. The latter facilitates the construction of queries that involve data joins from heterogeneous data stores. HERMES consists of (a) a mechanism to transform a Relational, Object-Relational, or Object-Oriented database into an equivalent XML database, and (b) a user friendly graphical query language (KINISIS) which makes possible the construction of the query via a GUI that utilizes basic traffic rule sign metaphors to represent instances (clauses) of the native W3C XQuery command syntax.

The future stages of our research will focus on the following three objectives: (a) establish a new paradigm for the GUI-based query construction stage, namely one whereby the user will be able to navigate himself in a graph representing the XML (meta)data content, expanding and collapsing selected nodes at will, (b) expand HERMES in the direction of supporting the semantic web data model (linked data), and (c) improve further the efficiency and the representation power of the algorithm that transforms a Relational, Object-Relational or Object-Oriented database into XML files.

ACKNOWLEDGMENTS

The work presented in this paper has been supported by the Research Committee of the Alexander Technology Educational Institute of Thessaloniki, under the Research Support Program 2009 (Π.Ε.Ε. 2009).

REFERENCES

[1] H. Garcia-Molina, J.D. Ullman and J. Widom. “Database Systems The Complete Book (Second Edition)”, Pearson Prentice Hall, 2008.

[2] R. Elmasri R and S. Navathe. “Fundamentals of Database Systems, 6th Edition”, Addison-Wesley, 2011.

[3] R.G. Cattell and D.K. Barry. “The Object Data Standard: ODMG 3.0”, Morgan Kaufmann Publishers, 1999.

[4] XML 1.0. Extensible Markup Language (XML) 1.0 (Fifth Edition), 2008 [online] Available at: http://www.w3.org/TR/REC-xml/ (Accessed at 8 June 2012).

[5] RDF Primer. Resource Description Framework W3C Recommendation 10 February 2004 [online] Available at: http://www.w3.org/TR/rdf-primer/. (Accessed at 8 June 2012).

[6] World Wide Web Consortium (W3C). [online] Available at: http://www.w3.org/ (Accessed at 8 June 2012).

[7] XQuery 1.0: An XML Query Language (Second Edition), W3C Recommendation 14 December 2010. Available through the internet: http://www.w3.org/TR/xquery/ (Accessed at 8 June 2012).

[8] A. Dix, J. Finlay, G. Abowd and R. Beale. “Human Computer Interaction, 3nd Edition”, Europe: Prentice Hall, 2004.

[9] K. Bekiropoulos, E. Keramopoulos, O.Beza and P.Mouratidis. “A list of features that a graphical XML Query language should support”, in International Journal of Computer Systems Science and Engineering (IJCSSE), vol. 26, no. 1 pp 13-21, 2010.

[10] E. Keramopoulos, A. Pliakas, K. Tsekos and I. Deligiannis. “KINISIS, a Graphical XQuery Language”, in Proceedings of International Conference on Integrated Information (IC-ININFO’2011), Kos, Greece, 29 September - 3October 2011.

[11] Y. Papakonstantinou, H. Garcia-Molina, and J. Widom. “Object Exchange across Heterogeneous Information Sources”, in ICDE Conference, 251-260 Taipei, Taiwan, 1995.

[12] S. Abiteboul, D. Quass, J. McHugh and J. Widom and J.L. Wiener. “The Lorel query language for semistructured data”, in International Journal of Digital Libraries, vol. 1, no. 1, pp. 68-88, 1997.

[13] L. Cardelli and G. Ghelli. TQL: “A Query Language for Semistructured Data Based on the Ambient Logic”, in Journal of Mathematical Structures in Computer Science vol. 14, pp. 285-327, Cambridge University Press, 2004.

[14] Giovanni Conforti, Giorgio Ghelli, Antonio Albano, Dario Colazzo, et al. (2002). The Query Language TQL. 5th International Workshop on Web and Databases (WebDB 2002), pp. 13-18.

[15] R. Baeza-Yates and G. Navarro. “XQL and proximal nodes Source”, in Journal of the American Society for Information Science and Technology, vol. 53, no. 6, pp. 504 – 514, 2002.

[16] A. Deutsch, M. Fernandez, Daniela Florescu, Alon Levy and D. Suciu. “XML-QL: A Query Language for XML”, in 8th International World Wide Web Conference, Toronto, May 1999, pp. 1155-1169.

[17] C. Baru, A. Gupta, B. Ludäscher, R. Marciano, Y. Papakonstantinou and P. Velikhov. “XML-based information mediation with MIX”, in International Conference on Management of Data of the 1999 ACM SIGMOD, Philadelphia, Pennsylvania, United States, pp. 597-599.

[18] V. Benzaken, G. Castagna, and A. Frisch. “CDuce: An XML-Centric General-Purpose Language”, in Proceedings of the ACM International Conference on Functional Programming, Uppsala, Sweden 2003, pp. 51-63.

[19] H. Hosoya and B. Â C. Pierce. “XDuce: A typed XML processing language” in ACM Transactions on Internet Technology, vol 3, no. 2, 2003, pp. 117-148.

[20] V. Benzaken, G. Castagna and C. Miachon. “CQL: a pattern-based query language for XML”, in BDA 2004, pp. 469-490.

[21] D. Colazzo, C. Sartiani, P. Manghi, A. Albano, G. Ghelli, L. Lini et.al. “A Typed Text Retrieval Query Language for XML Documents”, in Journal of the American Society for Information Science and Technology, vol. 53, no. 6, 2002, pp. 467-488.

[22] E. Bruno, J. Maitre and E. Murisasco. “Querying XML data : the DQL language”, in 10th International World Wide Web Conference (WWW10), Hong Kong 2001, pp. 180-181.

[23] D. Chamberlin, J. Robie and D. Florescu. “Quilt: An XML Query Language for Heterogeneous Data Sources”, in WebDB 2000 Conference, in Lecture Notes in Computer Science, Springer-Verlag, 2000, pp. 53-62.

[24] T. Pankowski. “XML-SQL: An XML query language based on SQL and path tables”, in Lecture Notes in Computer Science vol. 2490, 2002, pp. 184-209

[25] M. Harren, M. Raghavachari, O. Shmueli, M. Burke, R. Bordawekar, I Pechtchanski et al. “XJ: Facilitating XML Processing in Java”, in Fourteenth International World Wide Web Conference, Chiba, Japan, ACM Press 2005, pp. 278-287.

[26] E. Meijer, B. Beckman and G. Bierman. “LINQ: Reconciling objects, relations and XML in the .NET framework”, in International Conference on Management of Data, Proceedings of the 2006 ACM SIGMOD international conference on Management of data, pp. 706.

[27] V. Nassis, T. S. Dillon, J. W. Rahayu and R. Rajugan. “Goal-Oriented Requirement Engineering for XML Document Warehouses”, in Processing and Managing Complex Data for Decision Support, J. Darmont and O. Boussaid, Eds.: Idea Group Publishing, 2006, pp. 28-62.

[28] R. Rajugan, E. Chang, L. Feng, and T. S. Dillon. “Visual Modeling of Behavioural Properties in the LVM for XML Using XSemantics Nets”, in 10th International Database Engineering & Applications Symposium (IDEAS 2006), Delhi, India, pp. 209-219.

[29] R. Rajugan, E. Chang, L. Feng, and T. S. Dillon. “Modeling Dynamic Properties in the Layered View Model for XML Using XSemantic Nets”,in International Workshop on XML Research and Applications (XRA 2006) Harbin, China, pp. 142-147.

[30] V. Nassis, T. S. Dillon, R. Rajugan, and W. Rahayu. “An XML Document Warehouse Model”, in 11th International Conference on Database Systems for Advanced Applications (DASFAA 2006), Singapore, pp. 513-529.

[31] O. Beza, M. Patsala and E. Keramopoulos. “Comparison of XML Support in IBM DB2, MICROSOFT SQL SERVER and ORACLE”, in 2st International Scientific Conference, eRA-2: The Contribution of Information Technology to Science, Economy, Society and Education, Athens, Greece, 2007, pp. 22-23.

[32] XSL Transformations (XSLT) Version 2.0, W3C Recommendation, (23 January 2007) Webite: http://www.w3.org/TR/xslt20/

[33] XPath Version 2.0, W3C Recommendation (23 January 2007), Website: http://www.w3.org/TR/xpath20/

[34] S. Ceri, S. Comai, E. Damiani, P. Fraternali, S. Paraboschi and L. Tanca. “XML-GL: a Graphical Language for Querying and Restructuring XML Documents”, in 8th International World Wide Web Conference 199, pp. 151-165.

[35] M. Erwig. “A Visual Language for XML”, In IEEE International Symposium on Visual Languages 2000, pp. 47-54.

[36] D. Braga, A. Campi and S. Ceri. “XQBE (XQuery By Example): a visual interface to the standard XML query language”, in ACM Transactions on Database Systems, vol. 30, no. 2, 2005, pp. 398 - 443

[37] M. Erwig. “Xing a visual XML query language”, in Journal of Visual Languages and Computing, vol. 14, no. 1, 2003, pp. 5–45.

[38] N Fuhr and K Grojohann. “visXcerpt: A Query Language for Information Retrieval in XML Documents”, in the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, New Orleans, 2001, pp. 172-180.

[39] F Bryand S. Berger. “Xcerpt and visXcerpt: From Pattern-Based to Visual Querying of XML and Semistructured Data”, in 29th International Conference on Very Large Databases, 2003, pp. 1053 – 1056.

[40] S. Berger, F. Bry, O. Bolzer, T. Furche, S. Schaffert and C. Wieser.“Xcerpt and visXcerpt: Twin query languages for the Semantic Web”, in 3rd International Semantic Web Conference (ISWC2004), Hiroshima, Japan, November 2004.

[41] S. Berger. “Conception of a Graphical Interface for Querying XML”, Available at: http://www.pms.ifi.lmu.de/publikationen /diplomarbeiten/Sacha.Berger/THESIS/HTML-view/index.html (Accessed at 8 June 2012)

[42] A. C. Duta, K. Barker K.and R. Alhajj R.”ConvRel: relationship conversion to XML nested structures”, in Proceedings of the 2004 ACM symposium on Applied computing, Nicosia, Cyprus, 2004, pp. 698 – 702

[43] R Zhou, C. Liu and J. Li, “Holistic constraintpreserving transformation from relational schema into XML schema”, in Proceedings of the 13th international conference on Database systems for advanced applications, March 19-21, 2008, New Delhi, India, pp. 4-18

[44] C. Liu, M.W. Vincent, and J. Liu. “Constraint preserving transformation from relational schema to XML schema”, in Journal of World Wide Web, vol 9, no 1, 2006, pp. 93 - 110 .

[45] J. Shanmugasundaram, E. Shekita, R. Barr, M. Carey, B. Lindsay, H. Pirahesh H., et. al.. "Efficiently publishing relational data as XML documents", in the VLDB Journal, 2001, vol. 10, pp.133-154

[46] M. Fernandez, Y. Kadiyska, D. Suciu, A. Morishima and W.C. Tan. "SilkRoute: A framework for publishing relational data in XML", in ACM Data engineering 2002, pp.12-19.

[47] M. Care, D. Florescu, Z. Ives, Y. Lu, J. Shanmugasundaram, E. Shekita, et. al., "XPERANTO: Publishing object-relational data as XML", in WebDB 2000, pp.105-110

[48] T. Naser, R. Alhajj, M. J. Ridley. "Two-Way Mapping between Object-Oriented Databases and XML", in Informatica, vol 33, 2009, pp. 297–308

[49] ODABA. Available at: http://odaba.com/content/start/ , (Accessed at 8 June 2012)

[50] P. Walmsley. XQuery. O'Reilly, 2007.

[51] E. Keramopoulos, I. Deligiannis, G. Spanos, A. Pliakas and K. Tsekos. "A Controlled Experiment Investigation of Understandability and Correctness on Two Query Languages for XML data", in International Journal of Web Engineering and Technology (in press)