Round-Tripping Biblical Hebrew Linguisitic Data

Jan H. Kroeze, University of Pretoria, Pretoria, 0002 South Africa; E-mail: jan.kroeze@up.ac.za

1. INTRODUCTION

In processing language electronically, one can either concentrate on the digital simulation of human understanding and language production, or on the most appropriate way of storing and using existing knowledge. Both are valid and important. This paper falls in the second category, by assuming that it is useful to capture the results of linguistic analyses in well-designed, exploitable, electronic databanks. The paper focuses on the conversion of linguistic data of Genesis 1 between an XML data cube and a multidimensional array structure in Visual Basic 6 in order to facilitate data access and manipulation.

2. RELATIONAL DATABASE TECHNOLOGY VS. XML TECHNOLOGY

If the linguistic data of Genesis 1 were to be captured in a typical relational database, organised according to clauses as records in rows, and according to linguistic categories as logical attributes in columns, the word order of the elements in the clauses would be lost. One could, of course, store the positions of the words or phrases in a separate table and use a system of primary keys, foreign keys and joins to reassemble the original text, but this would cause much overhead (cf. Bourret, 2003).

Since the tables in traditional two-dimensional relational databases consist of rows and columns implying a maximum depth of two levels (Jeong & Yoon, 2001), one could only capture information of one linguistic module in a row without losing coherence. However, even if only the analysis of one linguistic module were captured in a table, it would be sparsely populated. One would therefore need multidimensional data storage facilities in order to find a better solution.

This paper, therefore, investigates XML technology to mark up free text, because it allows the hierarchical and multidimensional organisation of data. This data can be transformed into a three-dimensional array, processed by a computer program and saved back to XML for storage or transfer. Furthermore, XML was chosen because it is itself text-based and intuitively provides a suitable means to capture linguistic, textual data. It is very simple to use and does not require a high level of programming skills. An XML file is also platform independent and can be used by various programming languages and other software packages.

3. THE STRUCTURE OF THE DATABANK

3.1 The Structure of the Databank in XML

An XML document comprising of the text and markup of Genesis 1 may be regarded as a native XML database while a VB6 program that manipulates the data may be regarded as a content management system (cf. Bourret, 2003). The structure of the databank in XML for this project is built on the hierarchy as shown in Figure 1.

A complete discussion of the use of XML to build the Genesis 1 linguistic data cube may be found in Kroeze (2006a).

3.2 The Structure of the Databank in VB6

Although a lot of processing, such as indexing and searching, can be done directly on XML documents, one often needs a program to do more ambitious analyses (cf. Burnard, 2004). Visual Basic 6 (VB6) was chosen to perform this role with a view to this project, because XML is essentially a hierarchical system which fits the three-dimensional array data structure facilitated by VB6 perfectly.

Figure 1. Hierarchy underlying the XML databank containing linguistic data of Genesis 1

Hebrew Bible	- not used in this study
Bible Book	- not used in this study
Chapter	- root element in this study: <Genesis1>
Clause	- each clause represented by one table: <clause>
Clause Number	- each clause's ID: <clauseno>
Table Headers	- headings for each column: <headers><header>
Language Levels 1-5	- the various modules of analysis: <level1> ...
Level Description	- description of module per row: <leveldesc>
Phrases 1-5	- the word groups in a clause: <phrase1> ...

When converted into VB6, the databank module consists of a multidimensional array data structure. Multi-array data storage is one of the two main ways of implementing data warehouses (Wang & Dong, 2001). Typical data warehousing processing that may be performed on the clause cube using arrays and loops has been discussed in Kroeze (2004b). A multidimensional array structure is very suitable for a limited data set, such as the data in this project, due to its built-in indexing. Multidimensional online analytical products (MOLAP) "typically run faster than other approaches, primarily because it's possible to index directly into the data cube's structure to collect subsets of data" (Kay, 2004). The VB6 program discussed in this paper may be regarded as a simple MOLAP tool.

The three-dimensional array in VB6 contains records of 108 clauses. Each clause has five or less phrases. Each phrase potentially has five levels of analysis. One

Figure 2. The first unit in the three-dimensional array populated with linguistic data from the first clause in Genesis 1

```
Option Explicit
Public Clause(1 To 108, 1 To 5, 1 To 6) As String
Sub Main()
Clause(1, 1, 1) = "Gen01v01a"
Clause(1, 1, 2) = "bre$it"
Clause(1, 1, 3) = "in the beginning"
Clause(1, 1, 4) = "PP"
Clause(1, 1, 5) = "Adjunct"
Clause(1, 1, 6) = "Time"
Clause(1, 2, 1) = "-"
Clause(1, 2, 2) = "bara"
Clause(1, 2, 3) = "he created"
Clause(1, 2, 4) = "VP"
Clause(1, 2, 5) = "Main verb"
Clause(1, 2, 6) = "Action"
Clause(1, 3, 1) = "-"
Clause(1, 3, 2) = "elohim"
Clause(1, 3, 3) = "God"
Clause(1, 3, 4) = "NP"
Clause(1, 3, 5) = "Subject"
Clause(1, 3, 6) = "Agent"
Clause(1, 4, 1) = "-"
Clause(1, 4, 2) = "et ha$amayim ve'et ha'arets"
Clause(1, 4, 3) = "the heaven and the earth"
Clause(1, 4, 4) = "NP"
Clause(1, 4, 5) = "Object"
Clause(1, 4, 6) = "Patient"
End Sub
```

level of analysis is added to record the verse number as primary key for reference and searching purposes. An array of 108 x 5 x 6 is used to implement this data structure. If manually populated with data, the first clause could be coded as shown in Figure 2.

A complete discussion of this structure may be found in Kroeze (2004a). The same structure is used in this project to convert the data captured in the XML document into the VB6 array.

4. CONVERSION BETWEEN VB6 AND XML (ROUND-TRIPPING)

One of the advantages of an XML databank is the separation of the data and the manipulation thereof. The same data can be used for various purposes. An XML document in itself is not very reader-friendly. Therefore, one needs other software to efficiently process the data in such a repository (Kumar et al., 2005). Conversion is often necessary to make the data accessible for algorithms that implement efficient retrieval and human-friendly interfaces (cf. Ramsay, n.d.; cf. Witt, 2005).

If data is represented in a different format, it should first be parsed by an application. In our experiment, the data should be represented in an interlinear format which is easier to read. This necessitates that the VB6 program reads the data into an array in order to be printed as a series of linear tables on the screen. Removal of the XML tags restores the original text so that the layers of analysis become more comprehensible.

The conversion to and from XML format is called round-tripping. Round-tripping is the circular process of storing document data in a (XML) database and recreating the document from the database, a process which could result in a different version of the original document (Bourret, 2003). In this experiment round-tripping refers to the process of converting the Genesis 1 XML document to the three-dimensional array structure in VB6 and saving it again in XML format. If no changes are made while the data resides in the array, the resulting XML document should be an exact copy of the original (*ideal* round-tripping - Smiljanić, Blanken, Van Keulen, & Jonker, 2002). However, the array phase should facilitate updates to be made, which should also be reflected in the resulting XML document. Due to the limited scope and length of this paper, no program code is presented here.

4.1 From XML to VB6

All data in an XML document is text. The markup itself is also text only (Huitfeldt, 2004). For a linguistic database this creates no problem, since it also contains text data only. Therefore, in VB6 all the variables of the three-dimensional array are also of type *string* only. The limitation of arrays that all the elements should be of the same type (string, integer, boolean, etc.) therefore poses no problem. To strip the XML code from its tags, much string processing is performed (cf. Petroustos, 1999).

The easiest way to convert the Genesis 1 data would be to ensure that empty elements (e.g. where a clause has less than five noun phrases) are represented by a dash (-).The loop that reads the data cube elements into the three-dimensional array can then simply assume that the next line in the XML document will be the next element in the data structure. Not all phrases have syntactic or semantic functions and these missing elements may also be rendered by a dash. This simple implementation is used in this experiment, because it also ensures that the XML document is an exact copy of the original document after ideal round-tripping.

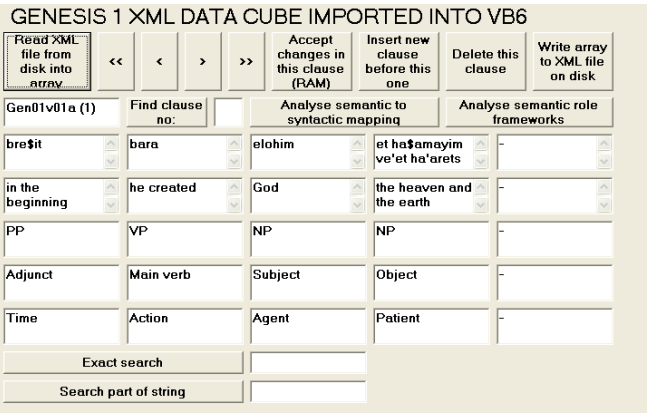
Although validation is usually done by means of a schema, it could also be performed here during the conversion process. In a subroutine the tags are first stripped and the remaining data tested against a standardised list of valid entries (e.g. syntactic and semantic functions) before it is transferred to the array. To show the contents of the array, the elements of each clause are displayed in a series of textboxes. When the program is run, the interface looks as shown in Figure 3.

Advanced data mining may now be performed on linguistic data stored in the three-dimensional array. Compare, for example, Kroeze (2006b) for a study of the semantic role frameworks extracted from Genesis 1.

4.2 From VB6 to XML

The conversion of the content of the three-dimensional array in VB6 to XML is more or less the reversal of the above process. While it is not necessary to perform validation again, string processing is used to convert the variables to lines of text

Figure 3. Interface used to render, process and round-trip linguistic data stored in a three-dimensional array and XML data cube



wrapped in applicable XML tags. The structure of the XML schema must be strictly adhered to in order to create a file that can again be read into VB6. In order to keep the test data intact, the current date and time could be added to the name of the output file so that a different XML file is created each time the button "Write array to XML file on disk" is pressed. If one wants the output file to have the same name as the input file (as in ideal round-tripping) that part of the code should be changed by replacing the name of the output file with the name of the input file.

5. CONCLUSION

This project proposed a database solution for the capturing and use of linguistic data. The paper focused on the conversion of linguistic data between an XML document and an array structure in VB6. After referring to the data structures in both XML and VB6, a method was discussed to round-trip the linguistic data between these software formats. One may conclude that these technologies are suitable for the efficient storage, transfer and processing of linguistic data.

REFERENCES

Bourret, R. (2003). *XML and databases.* Retrieved October 20, 2003, from http://www.rpbourret.com/xml/XMLAndDatabases.htm

Burnard, C. (2004). A gentle introduction to XML. *Essays in Humanities Computing.* Retrieved November 23, 2005, from http://www.digitalhumanities.org/Essays/

Huitfeldt, C. (2004). Scholarly text processing and future mark-up systems. *Essays in Humanities Computing.* Retrieved November 23, 2005 from http://www.digitalhumanities.org/Essays/

Jeong, H. W., & Yoon, A. (2001). Linguistic database handling using XML in web environment. In *ISIE 2001: 2001 IEEE International Symposium on Industrial Electronics Proceedings, 12-16 June 2001: Vol. 2* (pp. 833-838). Piscataway, NJ: IEEE.

Kay, R. (2004, March 29). Data cubes. *Computer World, 3*(13), 32. Retrieved December 8, 2006, from http://www.computerworld.com/action/article.do?command=viewArticleBasic&articleId=91640

Kroeze, J. H. (2004a). Towards a multidimensional linguistic database of Biblical Hebrew clauses. *Journal of Northwest Semitic Languages, 30*(2), 99-120.

Kroeze, J. H. (2004b). Slicing and dicing cyber cubes of Biblical Hebrew clauses. Paper read at the *SASNES Conference, Rand Afrikaans University, Johannesburg, August 2004.*

Kroeze, J. H. (2006a). Building and displaying a Biblical Hebrew linguistics data cube using XML. Paper read at the Israeli Seminar on Computational Linguistics (ISCOL), Haifa, Israel, June 29, 2006. Retrieved December 8, 2006, from http://mila.cs.technion.ac.il/english/events/ISCOL2006/ISCOL20060629_KroezeJH_XML_Paper.pdf

Kroeze, J.H. (2006b). Semantic role frameworks extracted from a multidimensional database of Gen. 1. Paper read at the SASNES Conference, University of South Africa, Pretoria, September 11-12, 2006.

Kumar, A., Schreibman, S., Arneil, S., Holmes, M., Bia, A., & Walsh, J. (2005). *<teiPublisher>:* a repository management system for TEI documents [Electronic version]. *Literary and Linguistic Computing, 20*(1), 117-132.

Petroustos, E. (1999). The complete Visual Basic 6 language reference. In J. Crawford (Ed.), *Visual Basic 6 complete* (pp. 744-904). San Francisco, CA: Sybex.

Ramsay, S. (n.d.). Databases. *Essays in Humanities Computing.* Retrieved November 23, 2005, from http://www.digitalhumanities.org/Essays/ (Reprinted from *A companion to digital humanities.*)

Smiljanić, M., Blanken, H., Van Keulen, M., & Jonker, W. (2002, October). *Distributed XML database systems.* Retrieved July 27, 2005 from http://www.purl.org/utwente/38064

Wang, X., & Dong, Y. (2001, October 29 - November 1). XML-based data cube. In *Proceedings of the Fifth International Conference on Info-tech and Info-net, Beijing, China, Vol. 5* (pp. 48-53). [Electronic version]. IEEE.

Witt, A. (2005). Multiple hierarchies: new aspects of an old solution. In S. Dipper, M. Götze, & M. Stede (Eds.), *Heterogeneity in focus: creating and using linguistic databases* (pp. 55-85). Universitätsverlag Potsdam, Germany. (Interdisciplinary Studies on Information Structure (ISIS), 2, Working Papers of the SFB 632.)