USING XML TO REPRESENT KNOWLEDGE BY FRAMES

Milko Marinov

Abstract: *Recent fashion in knowledge representation languages is to use XML as the low-level syntax. This tends to make the output of these knowledge representation languages easy for machines to parse, at the expense of human readability. The strength of XML lies in its simplicity to represent data and knowledge. It is used to develop modeling languages that are tailored to specific knowledge, and specific data structures and hierarchies. This paper considers an approach to frame knowledge representation using XML. A generalized XML model of the frame-based representation is proposed. The implementation of the XML generator, which is built into the interactive tool for frame representation is described.*

Key words: *Knowledge representation; Frames; XML model; XML Generator.*

INTRODUCTION

The World Wide Web currently contains a lot of data, more and more structured data (on-line databases, structured documents) and simple metadata but very little knowledge [2,4]. Recent fashion in knowledge representation languages is to use XML as the low-level syntax. This tends to make the output of these knowledge representation languages easy for machines to parse, at the expense of human readability.

The frames and their isomorphisms are suitable representation for complex knowledge [6,9]. The concept of frame was proposed in the 1970's [8], and frame systems subsequently gained ground as basic tools for representing knowledge [2,9]. In comparison to OOP systems, frame systems typically embody some notion of reasoning. Frame system reasoning may sometimes be incomplete (i.e., there is no guarantee that everything that could be deduced from a given set of information may be deduced) and frame systems do not typically make guarantees about the computational tractability of their inference [4,6].

The eXtensible Markup Language (XML) is a new powerful technique for Internet development. XML is a programming language standard that was released by the World Wide Web Consortium (W3C) in 1996. It is a method of defining structured data in a text file. The strength of XML lies in its simplicity to represent data and knowledge. It is used to develop modeling languages that are tailored to specific knowledge, and specific data structures and hierarchies. Before the Web, knowledge modeling languages were normally characterized by high computational complexity and logic-based formats with specific syntax and a function of mapping knowledge from and to computer languages. A typical example is KIF – Knowledge Interexchange Language [5]. The new generation of the Web-oriented knowledge representation languages are now being developed. DAMPL – The DARPA Agent Markup Languages, OIL – Ontology Inference Layer and SHOE – Simple HTML Ontology Extension are examples of such languages aiming to achieve the status of standards [1,3]. OIL aims to combine the most attractive features of frame based languages with the expressive power and formal rigour of a much expressive description logic. The knowledge representation languages are system independent and Web compatible through their adherence to XML and Resource Description Framework (RDF). Even languages like UML which is primarily a general modeling language for object-oriented analysis and design, is successfully applied in knowledge modeling when used with RDF syntax [1,6]. The fact that tools for knowledge modeling are now made available and understandable even for those without programming and artificial intelligence expertise opens up new opportunities for the application of Web-oriented frame-based knowledge representation.

The present paper considers an approach to frame knowledge representation using XML. The paper is structured as follows. Section 2 briefly summarizes the currently available standard for semantic interoperability, namely XML. Section 3 presents a formal

definition of the frame-based representation. Section 4 provides generalized XML model of the frame-based representation. Section 5 describes the implementation of the XML generator. Section 6 summarises the author's contributions and his future research intentions.

XML TECHNOLOGY

XML is a metalanguage [1,5,10]. XML is simply a set of rules for creating new document types. XML documents define data objects. XML documents provide access to the content and structure of the data objects. The structure of the data defines the object hierarchy. XML tags can be nested. They may have attributes. Attributes may have values. The logical structure of an XML document consists of declarations, tags (elements), comments, characters and processing instructions. The well formedness and validity requirements are used by XML parsers (processors) during processing XML documents. It is much easier to check the well formedness of a document than checking its validity. Checking the document validity requires to establish the semantics rules that are used as a reference to measure the document semantics against. These semantics are included in the DTD (document type definition). DTDs are used by XML parsers to validate XML documents. A DTD contains statements that define to the parser what is possible to do in a valid XML document that uses this DTD. The complete XML model requires two documents: the XML document and the DTD. When the parsing process is successful, the parser may create a data object model (DOM). This model has the object tree of the XML document. From the point of view of a programmer an XML document is either a stream of events or a tree. The stream of events approach is faster since it can be implemented along with parsing and it requires less memory. The tree view is available only after parsing an XML document, it is more versatile but also requires more memory for the data structure.

FORMAL DEFINITION OF THE FRAME-BASED REPRESENTATION

A frame represents an object or a concept. The frame is a collection of attributes (slots), potentially having types (or value restrictions) and potentially filled initially with values. When a frame is being used the values of slots can be altered to make the frame correspond to the particular situation at hand. The frame is described as a network of nodes and relations. It is straightforward to translate between semantic networks and frame-based representation. Both slot values and slots may themselves be frames. In fact, the most basic kind of facet a slot can have is the value facet. The value facet is the facet of a slot used to hold the data for the slot.

Frame hierarchy architecture is based on production-frame knowledge representation that exploits classification hierarchy with inheritance relation and active slots with query procedures and daemons, around which production rules are clustered [9]. Such an approach allows combining in one model static knowledge about the problem in the form of slot values, structural knowledge of the problem domain in the frame hierarchy, and dynamic knowledge in the form of attached procedures that drive logical inference.

The frame system state can be represented as $\mathcal{W}: I^2 \rightarrow \boldsymbol{S}$, where I - a set of identifiers, $\boldsymbol{S}$ – set of slots of the form $\boldsymbol{<v, d, \{Q_i\}, \{\mathcal{D}_j\}, \{C_k\}>}$ that include current slot value $v \in \mathbb{T}$, default slot value $d \in \mathbb{T}$, set of query procedures $\boldsymbol{\{Q_i\}}$ and set of daemons. Query procedures Q_i are expressions constructed according to some defined syntax, and daemons are represented by functions that change system state $\mathcal{D}_j : \mathbb{W} \rightarrow \mathbb{W}$. A set of slot values $\mathbb{T}$ can be of arbitrary structure.

Integration of relational databases into the frame model is based on implicit definition of frame set $\mathcal{F}(\mathcal{R})$ by a relational table $\mathcal{R}$, where each table row $\mathcal{R}=\{<x_1^{(i)}, x_2^{(i)}, \ldots x_n^{(i)}>\}$

defines a frame named $x_1^{(i)}$ (we can without loss of generality consider first key column or $\mathcal{R}$ to be a naming attribute) and slots $s_2, \ldots s_n$, with values $x_2^{(i)}, \ldots x_n^{(i)}$ accordingly.

Suggested approach for integration of relational databases and reasoning is in a way of a compromise between classical models of active databases, where relational model is extended by forward inference triggers and deductive databases, that are rather first-order logic representation of relational structures. Presented approach allows using existing relational DBMS for reasoning over data contained there by inheriting implicit database frames from parent frames that encapsulate dynamic knowledge used in inference.

XML MODEL

In order to create a frame database the requirements in many cases do not correspond to all requirements for development of a standard database. This is due to the fact that frame database can be neither included in the relational model, nor in the hierarchical model. The frame database is closest to the object data model. Each frame database supports four tables. The general conceptual schema of the frame database is presented on figure 1.

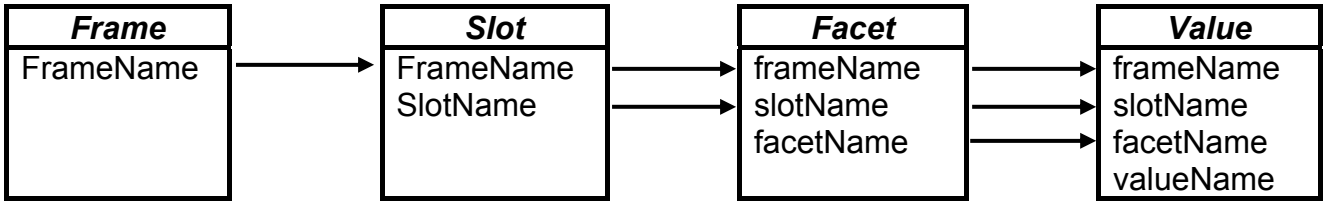

Figure 1: ***Conceptual schema of the frame database***

A schema conforms to the XML model can easily be derived from a conceptual Entity-Relationship schema. The main steps of the transformation method are:

- transformation of complex and generalisation/specification relationship types;
- creation of a tree structure by cycles breaking and father conflicts resolution (such a conflict occurs when an element has several potential fathers);
- modification of some non-allowed cardinalities;
- specification of sequence or choice between the children of a same father;
- transformation of attributes into XML elements or attributes.

Here is the order in which the XML DTD is created from the relational schema of the frame database: add the defined entities; add the elements to the entities; create the relationships.

This gives us the following structure of the XML DTD:

```
< !ELEMENT Frame (Slot +) >
< !ATTLIST Frame
        frameName CDATA #REQUIRED >
< !ELEMENT Slot (Facet +) >
< !ATTLIST Slot
        slotName CDATA #REQUIRED
        frameName CDATA #REQUIRED >
< !ELEMENT Facet (Value +) >
< !ATTLIST Facet
        facetName CDATA #REQUIRED
        slotName CDATA #REQUIRED
        frameName CDATA #REQUIRED >
< !ELEMENT Value EMPTY >
< !ATTLIST Value
        valueName CDATA #REQUIRED
```

```
                facetName CDATA #REQUIRED
                slotName CDATA #REQUIRED
                frameName CDATA #REQUIRED >
```

Here is a sample of the XML document:

```
<? xml version="1.0" ?>
<!DOCTYPE Frame SYSTEM "http:// …/…/filename.dtd"
<Frame>
        frameName = " ….. " >
        <Slot>
                slotName = " ….. " >
                <Facet>
                        facetName = " ….." >
                        <Value>
                                valueName = " ….." \>
                </Facet>
        </Slot>
</Frame>
```

IMPLEMENTATION OF THE XML GENERATOR

The XML generator is built into a frame tool [7]. The architecture of the tool is presented on figure 2.

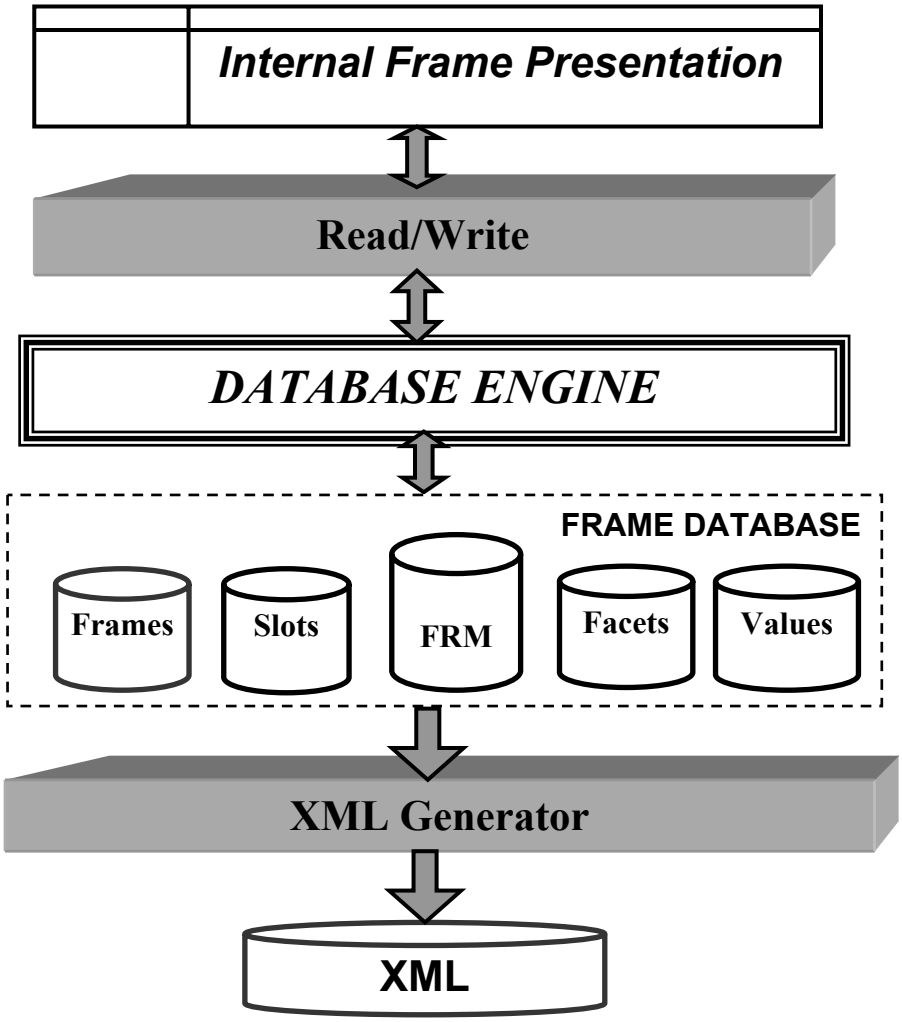

Figure 2: ***Architecture of the tool***

The frame tool implements most of the basic frame manipulation functions required for building databases of frames. These include creating a new frame database, opening an existing frame database, writing and deleting frame information. It is possible to add slots, facets, and values to the new frame definition, classify it, or edit other frames.

The XML generator lets the user to grab data from the frame database and create an XML document using the existing database schema. This makes building dynamic XML documents quick and easy. The XML generator uses the existing frame database schema to identify table relationships, constructing a query by setting field values for each table from the existing table schema. The generator then creates an XML document, absolving the user of the need to understand XML tags and other low-level details. The Field Manager is the heart of the XML Generator. Reading from the input file, each record contains data fields corresponding to the frame element. When the table is displayed in the XML generator, it shows a list of all the fields that belong to that table. The user is simply required to specify the order in which these fields occur. DTDs accompany XML documents and incorporate information about the allowed sequence and nesting tags, formats, and attributes, including their types and defaults. The generator creates DTD by identifying the participating tables and their relationships according to the proper DTD format. It constructs the XML document by iterating through the tree view and reading the attributes of every table and participating fields.

XML's ability to generate its own tags makes it much more flexible than HTML and its static tags, but it is necessary a way to describe what these tags do before someone else can manipulate the frame data. Document Type definitions (DTDs) solve this problem. DTDs accompany XML documents and incorporate information about the allowed sequence and nesting of tags, formats, and attributes, including their types and defaults. DTDs and XML make it possible for the user manipulating XML to understand the data in XML.

XML Generator algorithm

```
Retrieve the table and column information from the database information;
Retrieve table relationships;
String XMLGenerator :: GenerateXML( Pointer to DB, RootNode, ChildNode)
{
        Append Root Node in the string;
        Get the Root Node Name from whole string with Attributes;
        Add more XML data if required before the XML is generated;
        for ( i=0; i < FieldCount; i++ )
                        Store all the FieldName in an Array;
        For each Record in the RecordSet {
                Get the Attribute of the Child Node;
                Do some pre processing before the content of the XMLElement is generated;
                for( i=0; I < FieldCount; i++ ) {
                        Get the Name and Attribute List;
                        Add to Final String;
                }
                Do some post processing after the content of the XML Element is generated;
                Create the Child Node & add it in Final XML String;
        }
        Add more XML data if required after the XML is generated;
        Append the Closing Root Node;
}
```

The above described generalized algorithm of the XML Generator does not include all possible branches in it. The author provides this part of the algorithm which might be of interest to a wider range of readers.

CONCLUSION

XML has been proposed in this paper as a technique to represent knowledge by frames. Although XML allows the use of any tags as long as they are properly nested in the document, it is possible to define restrictions on the set of tags that can be used in the document. This is done in a DTD, which expresses in a grammar-like formalism which allowed sequences and nesting of tags are allowed in a document. The technique will allow developing Web-based knowledge-managed information systems.

Frames provide a fairly simple and clear way of representing properties of objects and categories of objects. There are many things that cannot easily be represented when using frames. It is hard to express negation, disjunction and certain types of quantification, that is, representing that something is true for *'all'* or *'some'* of the category of objects.

Further research is still needed to fully implement and develop the proposed XML model. Further efforts of the author will be aimed at building the XML generator into a knowledge-based information system to support the meta-knowledge and to refining the user interface.

REFERENCES

[1] Bertino E., G. Guerrini, M. Mesiti, A matching algorithm for measuring the structural similarity between an XML document and a DTD and its applications, Information Systems, Vol. 29, pp. 23-46, 2004.

[2] Harlemen F., D. Fensel, Practical Knowledge Representation for the Web, IJCAI'99 Workshop on Intelligent Information Integration, 1999. http://www.cs.vu.nl/~frankh/postscript/IJCAI99-III.pdf

[3] Jolma, A. and Rizzoli, A.E., A review of interoperability technicques for models, data, and knowledge in environmental software. In Post, D.A. (ed) MODSIM 2003 International Congress on Modelling and Simulation, Volume 4, pp. 1745--1750. 2003. http://www.idsia.ch/~andrea/papers/modsim03_aj.pdf

[4] Lassila O., Web Metadata: A Matter of Semantics, IEEE Internet Computing, Vol.2, No 4, pp. 30-37, 1998.

[5] Liu J., M. Vincevt, Querying relational databases through XSLT, Data & Knowledge Engineering, Vol. 48, pp. 103-128, 2004.

[6] Martin P., Conventions and Notations for knowledge Representation and Retrieval, in Proc. of ICCS 2000, 8th Int. Conference on Conceptual Strutures, pp. 41-54, 2000. http://www.webkb.org/doc/papers/iccs00/iccs00.pdf

[7] Marinov, M., An interactive tool for frame representation, Information Technologies & Control, No. 1, pp. 16-19 (2003).

[8] Minsky M., A framework for representing knowledge, in Patrick Henry Winnston (ed.), The Psychology of Computer Vision, McGraw-Hill, New York, 1975.

[9] Soshnikov D., An Architecture of Distributed Frame Hierarchy for Knowledge Sharing and Reuse in Computer Networks, In Proceedings of the 2002 IEEE International Conference on Artificial Intelligence Systems, IEEE Computer Society Press, pp. 115-119, 2002.

[10] Williams K., Professional XML Databases, Wrox Press Ltd, 2000.

ABOUT THE AUTHOR

Assist. Prof. Milko Marinov, PhD, Department of Computer Systems & Technologies, University of Rousse, Phone: (+359 82) 888 356, E-mail: MMarinov@ecs.ru.acad.bg.