Keywords: semantic web, rule-based systems, visual design, RuleML, XTT

Grzegorz Jacek NALEPA[*], Maria Antonina MACH[**]

INTELLIGENT BUSINESS WEB APPLICATIONS DESIGN USING THE XTT APPROACH[***]

Practical development of the intelligent business applications for the *Semantic Web* requires advanced methods and tools for knowledge processing and representation. XML-based languages allow for knowledge specification, however they do not support inference which is the core activity. In the paper it is proposed to apply a new knowledge representation for expert systems, the eXtended Tabular Trees (XTT) for the design, analysis, verification and processing of knowledge for Web applications. The technology integrates with RuleML, a language for specification of rules and assures portability and quality of the rule-base.

1 INTRODUCTION

Modern economies are based on knowledge to a great extent. Both knowledge and intellectual capital are nowadays considered the most valuable assets of any enterprise. It is so because dynamically changing markets force entrepreneurs to make rational decisions in a possibly short time, and these decisions are based on knowledge, which becomes the most important value of modern enterprise. Knowledge is the more and more considered one of key information resources of an enterprise, that allows for gaining competitive advantage. Knowledge creates enterprise's value. Knowledge, together with an ethic system influencing goals of an enterprise and ways to achieve them, both create the richness of the enterprise.

Since the role of knowledge is now recognized and widely appreciated, methods of knowledge management are being elaborated and improved. Yet before the enterprise starts managing knowledge, knowledge has to be acquired from both external and internal sources. One of the most important knowledge sources are enterprise employees. Not all employees' knowledge is explicit. Michael Polanyi defined the notion of a tacit knowledge [8]. Following his idea, Nonaka and Takeuchi distinguish between two types of knowledge: the tacit and the explicit one [10].

The second one is relatively easy to acquire, since it is systematic, codified, easy to communicate, transmittable in a formal language. Problems arise while trying to acquire the tacit knowledge, because

[*] AGH University of Science and Technology, Al. Mickiewicza 30, 30-059 Kraków, Poland; Institute of Physics, Kielce Pedagogical University, ul. Swietokrzyska 15, 25-406 Kielce, Poland; gjn@agh.edu.pl
[**] University of Economics, Komandorska 118/120, 53-345 Wrocław, Poland; maria.mach@ae.wroc.pl
[***] Research supported from the HEKATE Research Project.

it is personal, consists of mental models, beliefs, and perspectives that are difficult to be articulated or shared [11]. Briefly, problems of tacit knowledge acquisition concern:

1. knowledge expression - this kind of knowledge is often unconscious, therefore difficult to express,

2. mental barriers or negative emotions - some employees are not willing to share their knowledge; Moczydłowska [11] proved that approximately 33.6 % of employees present such barriers,

3. communication barriers - knowledge engineers are not always domain experts, therefore they suffer from problems with capturing people's knowledge; on the other hand, employees hardly ever understand formal notations used by knowledge engineers.

As for mental barriers, they are beyond the scope of this paper. Yet the other two problems may be solved or diminished with the aid of integrated computer tools, using visual knowledge representation.

Nowadays, Web technologies play a key role in practical engineering of global business systems. Applications of knowledge based systems in the area of Web-based business software is studied in the fields of *business intelligence* and *web intelligence*. In these areas number of both concepts and technologies are being developed. These technologies support the solving of the problems outlined above, namely knowledge acquisition, representation, and encoding.

Recent developments in the area of *The Semantic Web* proposal [9], turned the attention of the research community to application of the well-developed *Artificial Intelligence* methods to the Web technologies, including expert systems. One of the core features required by the Web application is the possibility of automatic, „*intelligent*" processing of *knowledge*, which in turn requires *inference* mechanisms. In contrast to the current Web architecture which merely stores data and provides simple *search* engines, the Semantic Web should offer advanced knowledge representation (covering semantic aspects) as well as advanced knowledge processing. These trends led to the development of number of technologies for knowledge *representation* and *processing*. These technologies have been founded on the rule-based systems (RBS) theory and practice.

Rule-based expert systems (RBS) constitute a powerful and very universal technology for information processing. Contrary to database technologies (both relational and XML-based), rules allow for *inference* of new knowledge, and not only information retrieval and simple question answering. In fact, rule-based expert systems technologies offer incomparably wider capabilities with respect to decision making, problem solving and generating parametric and structural solutions to complex problems. Some most common applications include process monitoring and intelligent control, system diagnosis, decision making and decision support, computer-aided activities such as design, analysis, prognosis, evaluation and control (see [1]).

With respect to Web applications one of the most important and matured technologies for the rule representation and low-level encoding is *RuleML* [2]. It has been accepted as an industry standard, for describing general rules in the Semantic Web, as well as business logic. There are already a number of tools for processing rules for the Web, including the RuleML format, such as jDrew (`www.jdrew.org`) which is a deductive reasoning engine for clausal first order logic. There are some simple design editors available, such as Oryx which is a graphical user interface application to design Mandarax (`mandarax.sf.net`, an open source library for deduction rules) knowledge bases.

In order to implement an efficient and consistent RBS, both the knowledge base and the inference engine must be developed and *integrated*. Some most important problems to be solved include the following: 1) defining an appropriate *language* for knowledge representation, 2) designing the *structure* of the knowledge-base, 3) defining the *inference control* mechanism, 4) assuring *quality* of the knowledge-base, and 5) preventing *portability* of knowledge base. These are well known problems in the field of *knowledge engineering*.

Unfortunately, RuleML itself addresses only the first problem (and partially problem 5), while tools like jDrew are related to the third problem. The area of the rule design (problem 2) still requires a lot of effort. The problem of rule verification and evaluation seems to be neglected and underestimated.

This paper is concerned with some conceptual and engineering aspects of efficient rule-based knowledge representation, that supports the knowledge acquisition problems, while supporting practical development of Web-base applications. In this paper a modular and *integrated* approach to solving the above problems at the logical level is presented. The proposed solution covers knowledge representation and structuring, knowledge elicitation, inference control, and quality verification. The approach is based on a novel concept for modular and structural knowledge representation: the *XTT* [4] approach. Here XTT stands for *eXtended Tabular Trees* – specific attributive decision tables of extended structure composed into a decision-tree-like graph. In the paper some details of the following concepts are presented: attributive language with non-atomic attribute values for knowledge representation, (Sect. 2) CASE supported design of the knowledge-base structure with XTT in Sect.3, evaluating the quality of the knowledge base, through the on-line analysis and assuring its portability through XML-based representation, where RuleML is one of the several supported solutions, see Sect. 4. The proposed approach constitutes the central part of the *Mirella* project (see `mirella.ia.agh.edu.pl`). Concluding remarks are given in Sect. 5.

2 EXTENDED LANGUAGE FOR KNOWLEDGE REPRESENTATION

Numerous RBS use simple knowledge representation logic based on attributes. Unfortunately, most of the systems allows for use of very simple atomic formulae only. Two most typical examples are of the form $A = d$ and $A(o) = d$, where A is an attribute, o is an object, and d is an *atomic* value of the attribute. In this way specification of attribute values is restricted to *atomic values only*.

In the proposed approach an extended attributive language is used. In fact we use *SAL*, the *Set Attributive Language*, as described in [3]. In SAL the atomic formulae are of two basic form, i.e. $A(o) = t$ and $A(o) \in t$, where t is an arbitrary set of values (a subset of the domain of attribute A). For intuition, $A(o) = t$ allows to say that attribute A takes *all* the values of t for object o (the so-called *internal conjunction*); in fact, A is a generalised attribute taking set values [3]. Expression $A(o) \in t$ says that attribute A takes *some* value of t (at least one) for object o (the so-called *internal disjunction*). In fact, the first case defines the so-called *internal conjunction* while the second one defines the so-called *internal disjunction*. Facts of the form $A(o) = t$ are used in the fact base and in conclusion part of rules while facts such as $A(o) \in t$ are used in preconditions part of rules. Further details on SAL and other attributive languages, their syntax, semantics and specific inference rules can be found in [3].

EXtended Tabular Trees (XTT), originally proposed in [4] constitute a new visual knowledge representation and design method. XTT incorporate SAL, and extended format of rules covering classical *if-then* ones, *retract-assert* part and inference control details. They aim at combining some of the existing approaches such as decision-tables and decision-trees by building a special hierarchy of Object-Attribute-Tables [3]. Some ideas used in its development were previously presented in [5,6]; a most complete description can be found in [3]. Below some key concepts of XTT are recapitulated in brief.

XTT *syntax* is based on some elementary concepts, allowing for building the hierarchy, these are: attribute, cell, header, row, table, connection, and tree. The header can be interpreted as the *table scheme*. It contains the list of attributes and their operating contexts: *conditional, assert, retract*, and *decision*.

An XTT table is interpreted as a set of rules, where rule $j + 1$ is processed after rule j. Rules grouped in one table share the same attributes. From operational point of view, the rules grouped in a single XTT-table are designed to operate in the same *context* or *state*. A row of a table is interpreted as a production rule, incorporating non-atomic attribute values. It is an *extended rule* format, allowing for *non-monotonic* reasoning, with explicit *control* statements represented through links between tables. Tables can be composed in a tree-like hierarchy. Each row x of a table w can have a right connection to another row z in another table y. Such a connection implies logical AND relation in between. Rule processing is then transferred from row j in table x to row k in table y.

The visual representation of XTT is crucial from the rule-based system design point of view. An example of an XTT structure is shown in Fig. 1.

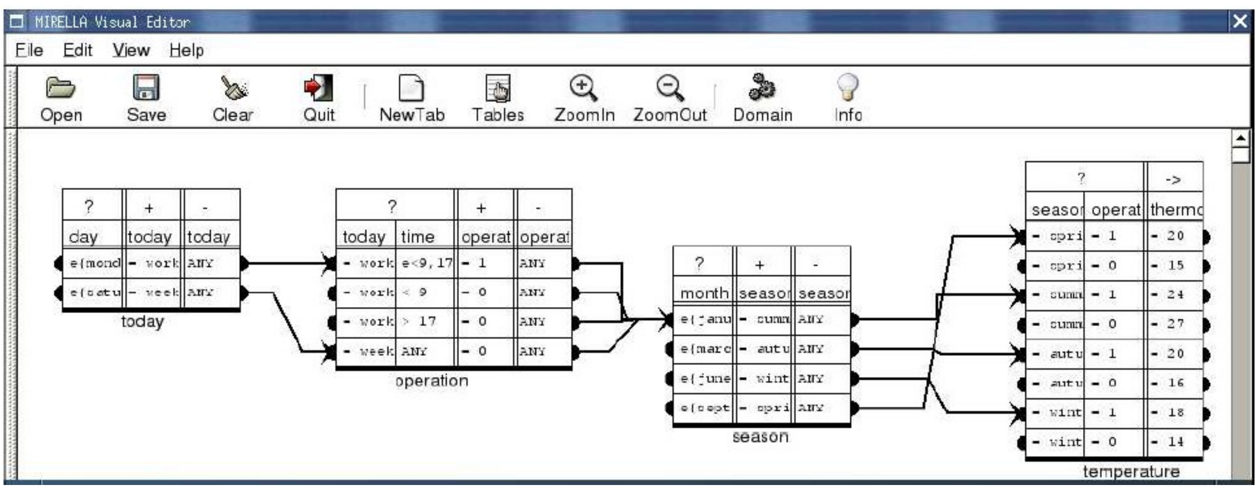

Fig. 1. An example of XTT structure.

In order to support the process of specifying RBS attributes for the XTT, as well as constructing the XTT table headers a conceptual design method has been developed. The *conceptual design* of the RBS aims at modelling the most important features of the system, i.e. attributes and functional dependencies among them. During this design phase the ARD modelling method is used. ARD [3,7] stands for *Attribute-Relationship Diagrams*. It allows for specification of functional dependencies of system attributes using a visual representation. The key underlying assumption in design of rule-based systems with knowledge specification in attributive logics is that the attributes are *functionally dependent*.

The concept of XTT as rule design, representation and transformation language provides the foundation for an integrated design methodology for rulebased systems, which is outlined in the next section.

3 DESIGNING EXPERT SYSTEMS USING XTT

The presented methodology can be seen as a hierarchical, top-down knowledge engineering process oriented towards more and more detailed specification of the system under design. It covers *conceptual modelling* (with ARD), *logical design* (with XTT), and *physical implementation* (with Prolog for knowledge verification and inference and RuleML for assuring portability) [4,3,7].

It also addresses three important aspects of the design models used, that is: *visual representation*, which is valuable from the design support and human interaction, *knowledge encoding*, which is based on XML (RuleML, XTTML, and ARDML) and is useful for automatic models transformations, and *executable code*, which is based on Prolog representation of RBS. Subsequent aspects of this approach are presented in Fig. 2.

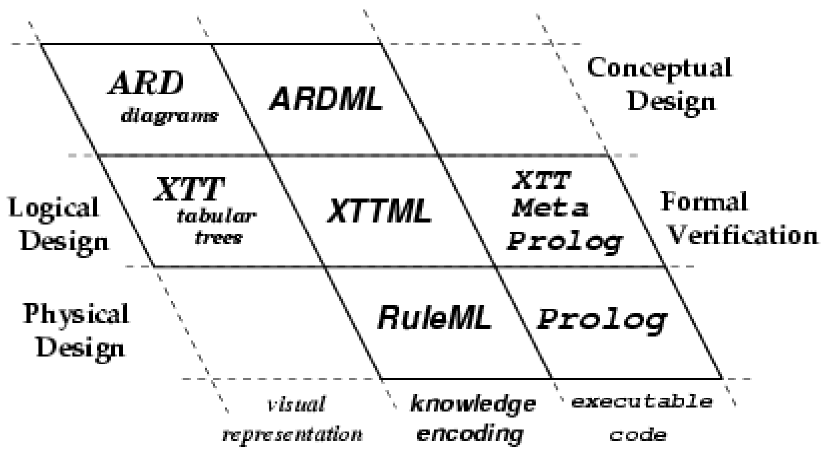

Fig. 2. Hierarchical design methodology.

4 XTT MARKUP REPRESENTATION

XML is becoming a *de facto* standard for flexible, *machine-readable*, knowledge representation and exchange meta-language. In order to provide means for XTT-based tools to be integrated with other rule-based tools two XTT→XML transformations have been originally described in [4].

The knowledge representation and design language XTT has two main aspects: *visual*, during the design the XTT structure is presented as a tree-table hierarchy, *logical*, during the implementation a logical, Prolog-based form is automatically generated. There is clearly a need to provide two distinct, domain specific, XML-based markup languages. For this purpose the following solutions have been chosen: *RuleML* [2], and *XTTML* – a new language, designed to represent visual aspect of XTT. A preliminary XTTML DTD has been included in [4]. An excerpt of XTTML code representing diagram in Fig. 1 is shown below.

```
<xttml:table_list>
  <tab tid="1" parent="0" state="0">
    <label>today</label> <type>0</type>
    <header>
      <hdratt context="0">day</hdratt>
      <hdratt context="1">today</hdratt>
      <hdratt context="2">today</hdratt></header>
    <rows>
      <row rid="1" parent="1" state="0">
        <in_connect/>  <out_connect>1</out_connect>
        <cells>
          <cell cid="1" parent="1" state="0">
            <att>day</att>  <oper>6</oper>
            <val atomic="0" ignore="0">
              <nav lobrace="2" hibrace="2">
                <navlo>monday</navlo>  <navhi>monday</navhi>
                . . . . .
```

RuleML [2] representation is related to logical XTT interpretation. However, XTT uses an *extended* rule form, which is not supported by RuleML. This format has to be encoded using the facilities RuleML provides. This has been accomplished using the *Hornlog* sublanguage and the cterm (*compound term*) markup [4]. Both the control specification and dynamic operation specification have to be encoded. The extended rule could be encoded in the SCLP (a rule format of *Common Rules*, the project was integrated into RuleML) syntax, as seen below. Using this approach it is easy to give a RuleML representation.

```
IF   att(val) AND att(val) ... AND att(val)
THEN conclusion(retract(att(val)), ... ,retract(att(val)),
                assert(att(val)), ... ,assert(att(val)),
                decide(att(val)), next(table,rule))
```

The extended rule, encoded in SCLP syntax, can be represented in RuleML:

```
<imp> <_body> <and>
    <atom> <_opr>attribute</_opr>    <ind>val</ind></atom>
    <atom> <_opr>attribute</_opr>    <ind>val</ind> </atom>
  </and> </_body>
  <_head> <atom>
    <_opr>conclusion</_opr>
    <cterm>
      <_opc>retract</_opc>
      <cterm> <_opc>attribute</_opc> <ind>val</ind> </cterm>
      <cterm> <_opc>assert</_opc>    </cterm>
      <cterm> <_opc>attribute</_opc> <ind>val</ind> </cterm>
      <cterm> <_opc>decide</_opc>    </cterm>
      <cterm> <_opc>attribute</_opc> <ind>val</ind> </cterm>
      <cterm> <_opc>next</_opc>      <ind>table</ind>
                                     <ind>rule</ind></cterm>
    </cterm> </atom> </_head> </imp>
```

Using XTT to design a RuleML rulebase provides high-level visual knowledge representation as well as logical analysis possibilities.

Transformation from XTT tables to a Prolog-based representation allows for obtaining a *logically equivalent* code that can be executed, analysed, verified, optimised, translated to another language, transferred to other system, etc.

The visual knowledge representation offered by XTT has two important advantages in the context of knowledge acquisition problems, outlined in Sect. 1.

First, it is easier to express knowledge with a visual tool. This concerns both tacit and explicit knowledge. This advantage refers to a psychological observation: while ”drawing” his knowledge, an employee immediately sees its structure and may control whether all important information has been expressed. If the tool is used by a knowledge engineer, an employee may co-operate with him, during knowledge acquisition, having the possibility to control advances in the process.

Second, visual representation allows for an easy formalization of acquired knowledge, since XTT tables are automatically transformed into Prolog-based code. Moreover, an employee may easily make corrections in the visual representation, diminishing the risk of misunderstandings between him and the knowledge engineer.

5 CONCLUDING REMARKS

The paper presents a novel approach to design of intelligent Web applications through a hierarchical, well-structured process composed of *conceptual*, *logical*, and *physical* design. It incorporates (automatically generated) RuleML code for assuring portability and XTT for visual design of the structure and control.

The practical advantages of the XTT representation influence knowledge management in an enterprise: more knowledge may be acquired, knowledge is of a good quality, and its formalization is quicker. This leads to a better use of information resources of an enterprise.

This approach has been already successfully applied in several areas including web security [6]. An experimental tool named Mirella integrating visual design, implementation, and verification has been developed. Future work on Mirella (`mirella.ia.agh.edu.pl`) will include full RuleML support with direct RuleML processing in Prolog.

REFERENCES

[1] LIEBOWITZ J., ed., *The Handbook of Applied Expert Systems*. CRC Press, Boca Raton 1998.

[2] BOLEY H., TABET S., WAGNER G., *Design rationale of ruleml: A markup language for semantic web rules*. In: SWWS'01, Stanford 2001.

[3] LIGĘZA A., *Logical Foundations for Rule-Based Systems*. Volume KU 0146. Scientific Publishers of AGH – University of Science and Technology, Cracow 2005.

[4] NALEPA G. J., *Meta-Level Approach to Integrated Process of Design and Implementation of Rule-Based Systems*. PhD thesis, AGH University of Science and Technology, AGH Institute of Automatics, Cracow, Poland 2004.

[5] LIGĘZA A., WOJNICKI I., NALEPA G., *Tab-trees: a case tool for design of extended tabular systems*. In: MAYR H. C. et al., eds.: *Database and Expert Systems Applications*. Volume LNCS 2113 of Lecture Notes in Computer Sciences. Springer-Verlag, Berlin, 2001, p. 422–431.

[6] NALEPA G. J., LIGĘZA A., *Security systems design and analysis using an integrated rule-based systems approach*. In SZCZEPANIAK P.S., KACPRZYK J., NIEWIADOMSKI A., eds., *Advances in Web Intelligence*: 3rd international Atlantic Web Intelligence Conference AWIC 2005: Lodz, Poland, June 6-9, 2005. Volume LNAI 3528 of Lecture Notes in Artificial Intelligence., Berlin, Heidelberg, New York, Springer-Verlag, 2005, p. 334–340.

[7] NALEPA G. J., LIGĘZA A., *Conceptual modelling and automated implementation of rule-based systems*. In ZIELIŃSKI K., SZMUC T., eds., *Software engineering: evolution and emerging technologies*. Volume 130 of Frontiers in Artificial Intelligence and Applications., IOS Press, 2005, p. 330–334.

[8] POLANYI M., *Personal Knowledge: Towards a Post-Critical Philosophy*. University of Chicago Press, 1974.

[9] GRONAU N., MUELLER C., USLAR M., *The KMDL Knowledge Management Approach: Integrating Knowledge Conversions and Business Process Modeling*. Proc. 5th International Conference on Practical Aspects of Knowledge Management, Vienna, Austria, 02.12.2004. LNCS vol. 3336, Springer, 2004, pp. 1-10.

[10] NONAKA I., TAKEUCHI H., *The Knowledge-Creating Company: How Japanese Companies Create the Dynamics of Innovation*. Oxford University Press, 1995.

[11] MOCZYDŁOWSKA J., *Zarządzanie wiedzą - dylematy psychologiczne*. In: NYCZ M., OWOC M. L. (eds.), *Pozyskiwanie wiedzy i zarządzanie wiedzą*. PN 1064, AE Wrocław, 2005, pp. 170-179.