Supporting Knowledge Maintenance Through Knowledge Artifacts

Stefania Bandini, Ettore Colombo, Fabio Sartori and Giuseppe Vizzari

Complex Systems and Artificial Intelligence Research Center, University of Milano-Bicocca, Bld. U14, Viale Sarca 336, 20126 Milan, Italy

ABSTRACT

A Knowledge Artifact (KA)-based approach to the modeling of Knowledge Based Systems is presented. Its main feature is to consider a KA as a conceptual and computational knowledge model of decision making. During knowledge engineering activities, a domain and problem dependent language defined according to the KA is adopted to represent the knowledge model. This approach aims to support expert users in Knowledge Maintenance, a crucial activity to prevent the decrease of overall system effectiveness due to application domain dynamism and knowledge base aging. The compounding problem is adopted as a case study to exemplify the approach.

KEYWORDS

knowledge maintenance, knowledge artifacts, compounding problem, adaptation knowledge

1. INTRODUCTION

A Knowledge Based System (KBS) can be used to support decision making (Holsapple & Joshi, 2001) both of experts and of beginners by sharing a

Correspondence to: Ettore Colombo, CSAI – University of Milano-Bicocca, Bld. U14, Viale Sarca 336, 20126 Milan - Italy; ettore.colombo@disco.unimib.it

knowledge model among them and suggesting solutions to specific problems. The effectiveness of a KBS and the quality of its support depend mainly on the internal (e.g. correctness and integrity) and the external aspects (e.g. adherence to the application domain and maintainability level) of the knowledge base (KB) that implements a knowledge model. Internal properties are studied by the Validation, Verification, and Integrity area (Coenen et al, 2001), whereas external aspects are related to Knowledge Maintenance (KM): software maintenance is the science of controlling software system changes, while KM is the science of controlling changes in KBS. More precisely, KM is the process of reflecting over some KBS in order to handle a new situation (Debenham & Menzies, 2000).

Thus, KM concerns the identification of the changes occurring in the application domain and in the knowledge involved in the problem solving. It aims to contrast the aging of the knowledge represented in the KB and a possible consequent abandonment of the KBS itself, due to its decreased effectiveness.

The proposed approach adopts the Knowledge Artifact (KA) notion to tackle KM related issues. Generally, KAs are classified as organizational knowledge resources (e.g. books, individual knowledge) and defined as "objects that convey and hold usable representation of knowledge" (Holsapple & Joshi, 2001). For instance, KAs are considered particular instances of knowledge assets (Finneran, 1999): thus, a generic KA can be considered one of the intangible assets of an organization.

The KAs are also defined as pieces of knowledge existing in a reusable format; thus, they can be retrieved and used to facilitate communication (Seiner, 2001); furthermore, organization knowledge flows can be described as KAs variations (Newman, 2002).

In Artificial Intelligence, KAs are used in the lessons learned framework (Aha el al., 2001) to define a computational model in which exploited KAs have a knowledge based structure and contain lessons-learned representations.

In this paper, a KA-based methodological proposal for KBS development to support KM is presented. Within an organization, the existing consensus on the modus operandi can be reified into an intellectual and common object: a conceptual and computational model capturing the knowledge involved in problem solving. This is the agreed and shared KA to be involved in the KBS

development. The latter must provide both *expert* and *machine accessibility*. The first one is crucial to maintain sharing properties of the knowledge model and allows constructing a KM tool for managing model sentences by experts. In turn, *machine accessibility* (i.e. *processability)* is required to support automatic interpretation and computation of model sentences by KBS.

2. KNOWLEDGE MAINTENANCE OVERVIEW

Generally, a KBS is developed for specific problem solved by expert practitioners. During the knowledge acquisition, phase experts explain knowledge engineers how they face the problem (i.e. their *expert model*). Knowledge engineers aim to translate experts' sentences into a *knowledge model* that is designed using an artificial general purpose language, either formal (e.g. first-order logic) or semi-formal (e.g. semantic nets). An effective KBS is obtained after the software design and implementation: this step involves the choice of a programming language (e.g. Prolog, Lisp, Jess) to transfer the knowledge model into the KB.

A KM activity is performed by a *knowledge maintainer* and brings variations either directly on the KB or operating on its knowledge model. The KM tools are thus developed considering a specific kind of maintainer and the model the latter works on. A knowledge engineer can handle a KB working on its knowledge model using a *Computer Aided Knowledge Engineering* tool (Akkermans et al, 1994)(Angele et al, 1998)(Cairò, 1998). In turn, a software maintainer can work directly on the KB as in first generation expert systems, where problem solving expertise was transferred from knowledge sources (e.g. experts) to programs (Studer et al, 1998).

Although these approaches require manual maintenance operations, some automatic maintenance methods are available in the case of KBS adopting those *computational models* (e.g. Case-based Reasoning (Kolodner, 1993)) that provide learning capabilities (retaining cases is a sort of automatic maintenance). Instead, when a maintained KBS is managed by a *Meta–KBS* developed in accordance to an explicit maintenance model, an even more automatic form of maintenance can be performed. However, also maintenance knowledge is subject to aging problems, leading to the *recursive maintenance problem* (Menzies, 1999).

Even if domain experts are the actual holders of the knowledge whose change should be reified in knowledge model variations, only in few cases (e.g. Bultman et al, 2000) maintenance activities are demanded to them. To support expert maintainers, this paper proposes a way to develop tools to allow them to manage model variations.

Performing KM can require modifications along both the model dimensions: the intensional and extensional (Guida & Berini, 2000). The first dimension is related to the general nature, the abstract and persistent domain structure, it concerns the ontological analysis defining entities, properties and relationships. Additionally, the domain dynamic aspects (i.e. how changes can happen in the domain and which processes can take place in it) belong to the intensional dimension. No specific element description is made in it. Contents concerning the specific domain (i.e. instances of entities, properties, and relationships) build up the extensional dimension.

The main feature of this proposal is to design a KA based grammar to define a language to write sentences of the knowledge model (i.e. the *KA based model*). This grammar is designed after ontological and epistemological domain analysis and derived by the non-formal and domain dependent experts' jargon. Thus, for expert maintainers, it is easier to manage sentences written in this language rather than in a general purpose one.

Variations in the knowledge model along the extensional dimension are accepted as sentences' variations (i.e. insertion, deletion, and modification). In turn, variations along the intensional dimension require a *language re-engineering* because of consequent interventions on the language definition and the re-engineering and implementation of its interpreter (i.e. the KBS). Thus, maintenance by domain experts is to be considered as an extensional modification of the KA based model.

3. THE COMPOUNDING PROBLEM CASE STUDY

The *compounding problem* is the research of the most suitable way of combining ingredients to design a product as a *recipe*. This recipe indicates which ingredients are involved in the product construction by the indication of their amounts. The problem has been tackled exploiting the KBS approach

in several domains (e.g. tablet (Craw et al, 1998), color (Cheetham & Graf, 1997) and rubber compound (Bandini & Manzoni, 2002) design).

The chemical formulation of rubber blends has been considered to develop a system to support the design and manufacturing of truck tyres in the P-Truck project in collaboration with Pirelli Tyres. It concerns the definition of the raw materials (i.e. the ingredients) to be used and the related amounts (measured in Parts per Hundred Rubber - PHR). Domain experts are chemical technologists (Bandini et al, 2003), skilled in managing raw materials (e.g. carbon black, sulfur).

Because compounding is performed through the adaptation of a prior recipe, the knowledge acquisition campaign was focused on the *adaptation knowledge* modeled in *Design Relations* (between tyre performances and blend features) and *Compounding Relations* (between blend features and recipe modifications). In the following, the grammar related to truck tyre compounding is described by means of a BNF notation.

3.1 Ontological Knowledge

Basic domain elements and *Relations* among them have been identified during the ontological analysis. A particular kind of relation is represented by *Functions*: specific grammar elements that describe possible recipe transformations.

Terms. Basic elements are represented by symbols describing recipe names, low and high level properties (blend features and tyre performances).

<low level property> ::= "llp1" | "llp2" | ...
<high level property> ::= "hlp1" | "hlp2" | ...
<recipe> ::= "RECIPE_1" | "RECIPE_2" | ...

As previously reported, recipes are made up of ingredients, each one belonging to a family (e.g. Natural Rubber–NR or Carbon Black–CB).

<ingredient> ::= "I1" | "I2" | "I3" ...
<family> ::= "F1" | "F2" | "F3" | ...

Within a recipe, it is possible to identify blends of ingredients made up of elements of the same family (e.g. blends of CB) and systems (e.g. Polymeric Matrix, Fillers). Each system is a functional group of ingredients, inserted into a recipe to provide the rubber blend with a particular feature. An example of system is the Curing one, made up of sulfur and other elements playing a fundamental role during tyre curing process which gives the tyre required thermal-mechanical properties.

<system> ::= "S1" | "S2" | "S3" | ...

A family is a set of ingredients with common chemical and physical properties. Two elements of a family are distinguished by means of attributes (e.g. for CB, the *structure* and the *superficial area*).

<attribute> ::= <discrete attribute> | <continuous attribute> | <type>
<discrete attribute> ::= "DA1" | ...
<discrete attribute value> ::= "DAV1" | "DAV2" | ...
<continuous attribute> ::= "CA1" | "CA2" | "CA3" | ...
<type> ::= "T1" | "T2" | ...
<type value> ::= "TV1" | "TV2" | "TV3" | ...

Considering the truck tyre domain, the product is made up of several components (e.g. tread, liner). The component and the tyre usage (i.e. axis, market segment, and area) define the blend usage context (e.g. the tread for the directional truck axis used in Turkish quarries). This is important to define constraints on recipe formulation depending on its usage.

<usages> ::= "usage1" | "usage2" | ...

Relations. Recipe formulations are described by sentences like "*(RECIPE_1, NR01, 50)*" and "*(RECIPE_1, CB01, 35)*". These indicate that the ingredients *NR01* and *CB01* are in the recipe *RECIPE_1* with amounts of *50 PHR* and *35 PHR*, respectively.

<recipe element> ::= '(' <recipe> ',' <ingredient> ',' <amount> ')'
<amount> ::= <numeral-list> | <numeral-list>','<numeral-list>

<numeral-list> ::= <numeral> | <numeral><numeral-list> | empty
<numeral> ::= '0' | '1' | '2' | '3' | '4' | '5' | '6' | '7' | '8' | '9'

Families are sets of ingredients and their elements are described by attributes. For instance, "*CB01 belongs to CB*" and "*NR01 belongs to NR*" express that *CB01* and *NR01* respectively belong to Carbon Blacks and Natural Rubbers. Beside, "*CB is described by structure*" and "*NR is described by NR Type*" associate the *structure* attribute to *CB* and the *NR Type* attribute to *NR*.

<family ingredient> ::= <ingredient> 'belongs to' <family>
<family attribute> ::= <family> 'is described by' <attribute>

Moreover, each family is related to a functional system indicating the role that ingredients play in the compound. Thus, sentences like "*CB is involved in the Fillers system*" and "*NR is involved in the Polymeric Matrix*" state that Carbon Blacks must be included in the *Fillers* system and that Natural Rubbers are in the Polymeric Matrix.

<family system> ::= <family> 'is involved in' <system>

Beyond ⟨*family ingredient*⟩ and ⟨*family attribute*⟩, also attribute value description is necessary.

<attribute value> ::= <ingredient> 'has value' <amount> 'for' <continuous attribute>
| <ingredient> 'has value' <type values> 'for' <type>
| <ingredient> 'has value' <discrete attribute values> 'for' <discrete attribute>

"*CB1 has value 30 for structure*", "*NR01 has value RSS for NR Type*" and "*BR01 has value high for cis*" are some examples, where "*structure*", "*NR Type*" and "*cis*" are names of a continuous attribute, a type and a discrete attribute respectively.

Certain constraints defining recipe construction boundaries in the considered domain can be described referring to the blend usage. These are constraints on combination cardinality, combination amount, ingredient amounts, and continuous attribute value limits.

<combination> ::=
<family> ' has maximum blend cardinality ' <amount> 'for' <usage>
<ingredient constraint> ::=
<amount> ' is the upper bound for ' <ingredient> ' for ' <usage>
| <amount> ' is the lower bound for ' <ingredient> ' for ' <usage>
<family constraint> ::= <amount> ' is the upper bound for ' <family> ' for ' <usage>
| <amount> ' is the lower bound for ' <family> ' for ' <usage>
<continuous attribute constraint> ::=
<amount> ' is the upper bound for ' <continuous attribute> ' for ' <usage>
| <amount> ' is the lower bound for ' <continuous attribute> ' for ' <usage>

Hence, it is possible to indicate that A is the maximum number of elements for Carbon Black combinations for a specific usage (i.e. "*CB has maximum blend cardinality* A *for usage1*"). Furthermore, given L and U, the values of the lower and upper bound of a generic interval respectively, the amount of a particular ingredient (e.g. *Sulfur*) could be limited for *usage1* (i.e. "L *is the lower bound for Sulfur for usage1*" and "U *is the upper bound for Sulfur for usage1*"). Similarly, also intervals on families (e.g. *CB*) could be specified for a usage (e.g. *usage1*) through the sentences "L *is the lower bound for CB for usage1*" and "U *is the upper bound for CB for usage1*". Sentences "L *is the lower bound for structure for usage1*" and "U *is the upper bound for structure for usage1*" indicates the interval of admissible values for the continuous attribute *structure*.

Additionally, system constraints not depending on the usage exist: for instance, "*Polymeric Matrix has constant quantity*" means that the amount augmentation of a NR ingredient is forbidden as the associated system (i.e. the Polymeric Matrix) must have a constant quantity. A modification that provides substitutions not modifying the total ingredient amounts is alowed.

<system constraint> ::= <system> 'has constant quantity'

Functions. Possible recipe modifications have been identified as functions. These stand for the possible Recipe Interventions (RIs) to adapt a recipe.

<modification action> ::= <shift> | <augmentation> | <substitution for family type>
| <substitution for discrete attribute> | <substitution for continuous attribute>
<shift> ::= '+' <family> ' -' <family>
<augmentation> ::= '+' <family>
<substitution for discrete attribute> ::= <family> ' for ' <discrete attribute> ' from '
<discrete attribute value> ' to ' <discrete attribute value>
<substitution for continuous attribute> ::= <family> ' for ' <continuous attribute>
<substitution for family type> ::= <family> ' for ' <type> ' from ' <type value> ' to ' <type value>

For example the augmentation of Carbon Black blend amount is "*+CB*", while a Natural Rubber substitution taking care of their type is specified by "*NR for NR Type from RSS to SMR*", where "*RSS*" and "*SMR*" are two possible NR types.

3.2 Expert Knowledge

The knowledge acquisition campaign also concerns the epistemological analysis of the compounding decision making process. An important result is the T–Matrix (see Fig. 1), which shows the relations between Tyre Performances (TPs) and Blend Features (BFs) and between BFs and Recipe Interventions (RIs). Each T-Matrix is related to a particular usage and shows the recipe modification effects in terms of blend feature variations (i.e. as Compounding Relations) and the blend feature change effects on tyre performance (i.e. as Design Relations).

	BF 1	BF 2	BF 3	BF 4
TP 1	⊙↑	○↑	⊙↓	⊠
TP 2	⊠	⊙↑	▲↓	○↑
TP 3	○↑	⊠	▲↓	⊙↓
TP 4	⊙↓	▲↑	⊠	▲↑
TP 5	⊠	⊙↓	○↑	⊠
RI 1	⊙↓	▲↓	⊠	⊠
RI 2	○↑	⊙↓	▲↑	⊙↑
RI 3	⊙↓	⊠	⊠	▲↑
RI 4	▲↓	⊙↑	▲↑	⊠

Correlation	⊙	Strong
	○	Good
	▲	Weak
	⊠	No corr.
Proportionality	↑	Direct
	↓	Inverse

Fig. 1: A T-Matrix and the related dictionary

For example, the strong correlation with a direct proportionality between RI2 and BF4 stands for the sentence: RI2 variation produces a strong variation on BF4.

In the grammar, "*is related in compounding to*" and "*is related in design to*" and some *additional terms* to describe the grade of correlation and proportionality have been identified.

<correlation> ::= "strong" | "good" | "weak"
<proportionality> ::= "direct" | "inverse"
<correlation description> ::= "no" | `(' <correlation> ` correlation and '<proportionality> ` proportionality)'

For instance, calling *C* and *P* the value of correlation and proportionality, "*+CB is related in compounding to tensile strength with (C correlation and P proportionality)*" describes how the CB amount increase and the tensile strength feature (i.e. a low level property) are correlated and with which proportionality.

<design relation> ::= <low level property> ` is related in design to '
<high level property> ` with ' <correlation description> ` correlation'
<compounding relation> ::= <modification action> ` is related in compounding to '
<low level property> ` with ' <correlation description> ` correlation'

Moreover, given a usage, it is possible to weight the relevance of a blend feature or of a tyre performance. For each blend feature (or tyre performance) it is also possible to indicate the preferred modification direction.

<low level property weight> ::= <amount> `is the weight of' <low level property> `for' <usage> | <direction> `is the preferred direction of' <low level property> `for' <usage>
<high level property weight> ::= <amount> `is the weight of' <high level property> `for' <usage> | <direction> `is the preferred direction of' <high level property> `for' <usage>
<direction> ::= "augmentation" | "reduction"

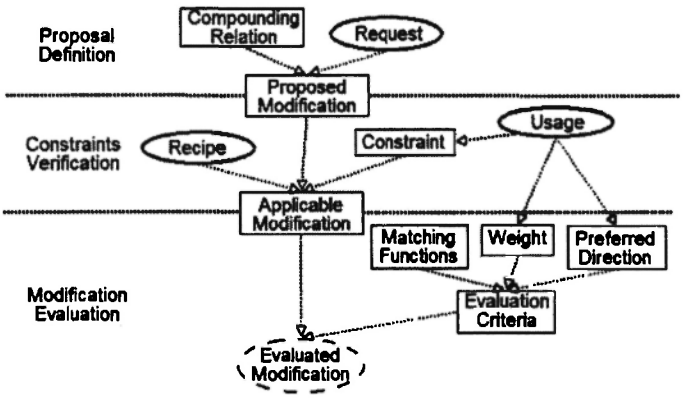

Fig. 2: The Compounding Problem Solving Method

3.3 Knowledge Artifact Based Support

Adaptation knowledge can be exploited to support the research of interventions that could generate a new recipe starting from an existing one and fitting given requests on property variations. Considering these instances of Compounding and Design Relations, "RI2 is related in compounding to BF4 with (strong correlation and direct proportionality)" and "BF4 is related in design to TP4 with (weak correlation and direct proportionality)", the developed KBS must be able to treat them: if RI2 is an augmentation function and a recipe intervention able to decrease weakly TP4 is needed, an interpretation of the introduced sentences can suggest an inverse application of RI2 (i.e. the reduction of the ingredient involved in RI2).

The problem solving method (PSM) defined in the P–Truck project (see Figure 2) gives an evaluation of the possible and applicable recipe interventions to meet requests of blend feature variations.

The PSM requires specific inputs (i.e. a recipe and a usage) and a list of variations described by a blend feature specification, a direction and a variation extent. The direction can be *to increase*, *to decrease* or *to maintain* a blend feature, while managed variation extents are *strong*, *good* and *weak*.

The PSM is divided into three rule sets: *Proposal Definition*, *Constraints Verification* and *Modifications Evaluation*. *Proposed modifications* are defined to change blend features in accordance to the input requests. The extent of each one of them is used to qualify the releted proposed modification. The recipe and the usage allow the setting of which constraints the recipes must satisfy. Each *proposed modification*, if applied, can bring to a recipe that violates or complies with these constraints. In the latter case, the modification is an *applicable modification* because it stands for an admissible solution.

Because the PSM goal is to rank *applicable modifications*, the third rule set involves an evaluation criterion to calculate the modification votes. In doing so, the blend feature importance expressed by the weight related to the input usage is considered. This is combined with the matching function result adopting a weighted average function. The matching function gives an estimate of how much a sub-effect on a specific blend feature fits the related request.

CONCLUSIONS

A methodological approach for Knowledge Based System development that provides the exploitation of the Knowledge Artifact notion in the KBS life cycle has been described. The main idea underlying this approach is the definition of a domain and problem dependent language to support experts in the direct maintenance of the knowledge model encapsulated in a KBS. Thus, a new perspective to KAs is introduced and put in relation with existing definitions and approaches. According to this standpoint, a KA is a computational model coming from the consensus on the modus operandi concerning a particular problem within a specific domain. This cognitive object can lead the definition of the domain and problem dependent language to represent the knowledge model.

The proposal to consider a KA based language allows an explicit representation of ontological and epistemological aspects of the considered domain. This means that the knowledge represented in the grammar captures the knowledge model intensional dimension. In turn, the extensional dimension is expressed by the sentences. The latter can be involved in knowledge sharing and dissemination when distributed by means of suitable

tools. Moreover, the KBS must be able to understand and exploit the knowledge model to support users' decision making. Since it is possible to work along the extensional dimension by managing KB sentences, domain experts are provided with specific tools to create, modify, and remove KB contents. For them, it is easier to manage sentences written in the KA based language rather than in a general purpose one because the first one reflects their viewpoint on the specific domain.

This paper does not discuss the part of this approach concerning the software system design and implementation. To complete this methodological proposal, it is important to consider a KBS architecture providing these functionalities: *knowledge storage* to persistently record the model in the KB; *deployment* and *capture* to access and maintain the KB by expert maintainers; *processing* to interpret and compute KB contents. A more detailed description of these functionalities in the P-Truck architecture can be found in (Bandini et al, 2004).

REFERENCES

Aha, D.W., Weber, R., Muñoz-Avila, H., Breslow, L. and Gupta, K.M. 2001. Bridging the Lesson Distribution Gap, in *IJCAI'01*, Morgan Kaufmann 987-992.

Akkermans, H., de Hoog, R., Schreiber, A., van de Velde, W. and Wielinga, B. 1994. *CommonKADS: a Comprehensive Methodology for KBS Development.* IEEE Expert 9(6):28-37.

Angele, J., Fensel, D. and Studer, R. 1998. *Developing Knowledge-Based Systems with MIKE.* Journal of Automated Software Engineering, 5(4):389-418.

Bandini, S., Colombo, E., Colombo, G., Sartori, F. and Simone, C. 2003. The Role of Knowledge Artifacts in Innovation Management: the Case of a Chemical Compound Designer COP, in *Proc. of C&T'03.* Kluwer Academic Publishers, 327-345.

Bandini, S., Colombo, E., Mereghetti, P. and Sartori, F. 2004. A General Framework for Knowledge Management System Design, in: *Proc. of ISDA'04*, 61-66.

Cairò, O. 1998. The KAMET Methodology: Contents, Usage and Knowledge Modeling, in: *Proc. of KAW'98.* SRGD Publications, 1-20.

Bandini, S. and Manzoni, S. 2002. Modeling Core Knowledge and Practices in a Computational Approach to Innovation Process, in: *Model-Based*

Reasoning: Scientific Discovery, Technological Innovation, Values. Kluwer Academic/Plenum Publishers, 369-390.

Bultman, A., Kuipers, J. and van Harmelen, F. 2000. Maintenance of KBS's by Domain Experts: the Holy Grail in Practice, in: *IEA/AIE'00*, LNAI, vol. 1821 Springer Verlag, 139-148.

Coenen, F., Eaglestone, B. and Ridley, M. 2001. Verification, Validation, and Integrity Issues in Expert and Database Systems: Two Perspectives, *IJIS*, **16(3)**:425-447.

Craw, S., Wiratunga, N. and Rowe, R. 1998. Case Based Design for Tablet Formulation, in: *EWCBR'98*, LNCS, vol. 1488, Springer-Verlag, 358-369.

Cheetham, W. and Graf, J. 1997. Case-Based Reasoning in Color Matching, in: *ICCBR'97*, LNCS, vol. 1266, Springer Verlag, 1-12.

Debenham, J. and Menzies, T. 2000. Expert System Maintenance. *Encyclopedia of Computer Science and Technology*, **42**, 35-53.

Finneran, T. 1999. *A Component Based Knowledge Management System* Available at www.tdan.com. Issue 9.0

Guida, G. and Berini, G. 2000. *Ingegneria della Conoscenza: Strumenti per Innovare e per Competere.* EGEA.

Holsapple, C. and Joshi, K. 2001. Organizational Knowledge Resources. *Decision Support Systems*, **31(1)**:39-54.

Kolodner, J. 1993. *Case-Based Reasoning.* Morgan Kaufmann Publisher Inc.

Menzies, T. 1999. Knowledge Maintenance: The State of the Art. *The Knowledge Engineering Review,* **14(1)**:1-46.

Newman, B. 2002. *Knowledge Flow Fundamentals.* Knowledge Management Handbook. Springer-Verlag.

Seiner R. 2001. *Metadata as a Knowledge Management Enabler.* Available at www.tdan.com. Issue 15.0

Studer, R., Benjamins, V.R. and Fensel, D. 1998. *Knowledge Engineering: Principles and Methods.* Data Knowledge Engineering **25(1-2)**:161-197.