PROFILE COMPILER: Ontology-Based, Community-Grounded, Multilingual Online Services to Support Collaborative Decision Making

Yan Tang, Stijn Christiaens, Koen Kerremans and Robert Meersman

Abstract—In today’s industries, ontologies and the modern semantic technologies are offered as the potential solutions to support mutual understanding in the collaborative settings. Currently, organizations are unable to cope with collaborative decision making in a particular domain. Mastering domain knowledge and sharing decision rules is a complex yet important issue. In this paper, we first design a generic framework for community-grounded, collaborative decision making enabled, ontology based systems. This framework is constructed with three main engineering tracks: the linguistic engineering track, the ontology engineering track and the group decision engineering track. Each engineering track has its own methodology supported by a tool. Then, we use Semantic Decision Tables to elaborate the knowledge blocks resulting from these three engineering tracks. Semantic Decision Tables are mainly used (1) to elaborate concepts of the domain ontologies, (2) to gather semantically grounded decision rules within a decision group and (3) to provide tabular reports to the users. Finally, an online multilingual knowledge platform – the PROFILE COMPILER is developed as the result. It allows small or medium-sized enterprises (SMEs) to retrieve and personalize existing domain ontology, design and share their decision rules. We illustrate in the domain of Human Resource Management (HRM).

Index Terms—Information systems, Intelligent databases, Knowledge base management systems

I. INTRODUCTION

Ontologies ([7], [8]) as knowledge resources that are shared and agreed upon within a user community, provide the means to establish mutual understanding in collaborative settings. A lot of work has been done by the Semantic Web community on formalizing, reasoning and querying ontologies. The research on socio-aspect of the ontology creation and application has been investigated recently, such as in [5], [6]. The authors point out that a key objective in next-generation computerized distributed working environments is to effectively leverage individuals working together to a community level. Relevant commonalities in the ontology and differences of using ontologies are not static. They keep evolving all the time.

From our experiences in the projects, such as the IWT POCEHRMOM1 and the EU PROLIX2 projects, this kind of community aspect has been stressed again and again. We have learned that the programmatic aspects of gradually enriching ontology and using ontology in a group setting are the keys to the success. The ontology based systems would significantly improve if users can define their individual organizational knowledge on the bases of relevant commonalities.

This paper provides a recent research effort and experiments on designing a community grounded framework, with which we sew linguistic engineering, ontology engineering and group decision engineering together. These three engineering tracks are executed in a dynamic and programmatic manner over long periods of time.

The rest of the paper is structured as follows. In section II, we present the use case background and design overview, which is explained in detail by focusing on three engineering tracks: the linguistic engineering track (section III.A), the ontology engineering track (section III.B) and the group decision engineering track (section III.C). Decision rules designed as SDT commitments are illustrated and discussed in section IV. Based on the design framework, we implement the PROFILE COMPILER – a tool to support ontology-based, community-grounded, multilingual, collaborative group

---

1 IWT POCEHRMOM project aims to develop ontologies for HRM. The purpose is to assist HRM activities designed by small or medium-sized enterprises. [http://cvc.ehb.be/PoCeHRMOM/FrameSET.htm](http://cvc.ehb.be/PoCeHRMOM/FrameSET.htm)

2 EU PROLIX project is to align learning with business processes in order to enable organisations to faster improve the competencies of their employees according to continuous changes of business requirements. [http://www.prolixproject.org/](http://www.prolixproject.org/)
knowledge blocks evolve over time amongst the community. Engineering tracks are recursively executed. For the ontology engineering and group decision engineering. Those development tracks of respectively linguistic engineering, in practice. In the following sections, we will discuss the discussion and the future work (section VII).

II. DESIGN OVERVIEW AND USE CASE BACKGROUND

The development cycle, shown in Fig. 1, covers three main engineering tracks: the linguistic engineering track (section III.A), the ontology engineering track (section III.B) and the group decision engineering track (section III.C). Each development track is supported by its own methodology and construction tools. The linguistic engineering track is executed by a group of linguistics. The ontology engineering track needs domain experts and ontology engineers to model the domain ontologies. The decision maker group is involved in the group decision engineering track.

Fig. 1 Design overview of the development tracks

This design overview is simple at a glance, yet complicated in practice. In the following sections, we will discuss the development tracks of respectively linguistic engineering, ontology engineering and group decision engineering. Those engineering tracks are recursively executed. For the knowledge blocks evolve over time amongst the community.

We implement the design in Fig. 1 in the domain of human resource management (HRM). Note that this design is not restricted to a specific domain or problem. We illustrate with the example from a real-world problem, such as the competence profile management in the HRM domain, in order to give a better image of Fig. 1.

In HRM domain, the term competence profile is defined as a well structured document that contains different types of competences and skills. It describes the ability of an employee or the requirement of a job. Each competence (or skill) has its level. A compiler rule is a decision rule, based on which the system proposes a matching between jobs, competencies and experts.

There are several definitions of competence and competency available. One definition given by the HR-XML consortium\(^3\) specifies a competence as “a specific, identifiable, definable, and measurable knowledge, skill, ability and/or other employment-related characteristic (e.g. attitude, behavior, physical ability) which a human resource may possess and which is necessary for, or material to, the performance of an activity within a specific business context”. Based on this definition, a competency is considered as the smallest unit of capability and a competence is a structured set of competencies. The authors in [2], for instance, adopt this definition.

Another definition of competency given in [15] specifies it as a combination of resources in a specific context (e.g. physical, social, organizational, cultural and/or economical aspects) for reaching an objective and fulfilling a mission. This definition highlights a singular dimension in which a competency is defined as a set of competences. A competence can be further defined and modeled. In this paper, we adopt the meaning of competences as it has been defined in [15].

In the following sections, we will use the above definitions as the domain background for every engineering track in Fig. 1.

III. ENGINEERING TRACKS

A. Linguistic Engineering

One of the purposes of the linguistic engineering track is to identify the multilingual terminology that is used to denote occupations and competences. The terminology is extracted from textual resources (a multilingual collection of existing taxonomies and occupation profiles) and structured according to a common, predefined abstract metamodel or categorization framework. In this project, the categorization framework visualizes the metacategories (such as ‘occupation’, ‘competence’, etc.) which are considered important for the development and realization of the Profile Compiler application. A metacategory (e.g. ‘occupation’) is used to cluster a set of categories on the basis of at least one overlapping property or a similar parent node. Categories are derived from the sample of terms (e.g. ‘software developer’, ‘police officer’, ‘teacher’) that are extracted from the textual resources. By using the MCFE software [4] for creating and managing categorization frameworks, the terms are either manually extracted and classified or semi-automatically structured on the basis of a number of routines.

The principle of extracting and immediately classifying terminology is adopted from the method of Termontography [10]. Termontography is a terminology engineering approach, combining theories and methods of sociocognitive terminological analysis [22] with methods in ontology engineering [18]. An important view in Termontography is that a knowledge analysis phase should ideally precede the methodological steps which are generally conceived as the starting-points in terminography: i.e. the compilation of a domain-specific corpus of texts [14] and the understanding

\(^3\) [http://ns.hr-xml.org/2.4/HR-XML-2.4/CPO/Competencies.html](http://ns.hr-xml.org/2.4/HR-XML-2.4/CPO/Competencies.html)
and analysis of the categories that occur in a certain domain [13]. Terminology engineering is in this sense not strictly confined to the identification and defining of terms but is also focused on knowledge modeling through the organization of terms in a categorization framework. We will show in this section how the preparatory work for ontology engineering (section III.B) is realized by adopting principles of Terminography in the linguistic engineering track.

Fig. 2 shows some results of the linguistic engineering track in the MCFE software. The metacategories that are considered relevant for the Profile Compiler application are (amongst others) ‘competence’ and ‘occupation’. From Fig. 2, it can be derived that the metacategory ‘competence’ is further divided into a number of submetacategories: ‘computer competence’ (i.e. general computer skills and knowledge), ‘domain competence’ (i.e. skills and knowledge pertaining to a specific domain), ‘internal skill’ (i.e. attitudes, characteristics of an individual) and ‘language competence’ (i.e. general language skills and knowledge). Selecting one of the metacategories in the tree structure shows all categories that are classified according to this metacategory in the right pane. In the example, the categories found under the metacategory ‘occupation’ were all derived from English terms that were extracted from O*NET, a large American resource providing occupational information.

The conceptual information about a category that is derived from the linguistic analysis of the multilingual texts, is translated into a list of intercategorial relations in the ‘Category relations’ pane. The example in Fig. 4 shows that the category ‘actor’ is linked to four competences, extracted from O*NET: ‘Oral Expression’, ‘Originality’, ‘Memorization’ and ‘Speech Clarity’. It is possible to further specify the terms that are used to denote all categories (and metacategories) in the framework by what could be considered ‘typical’ terminological information. Depending on the requirements specified for each project, examples of such information are: a definition in natural language, a part-of-speech tag, etymological information, gender information, etc.

By presenting the conceptual information about a specific category in the ‘Category relations’ pane, it should become clear that the linguistic engineering track is not just limited to the extraction of terms (and identification of categories or concepts). We have tried to structure terminology in a categorization framework and to deliver a first model in natural language which forms the basis of the work in the ontology engineering track, outlined in the next section.

B. Ontology Engineering

One of the aims of the ontology engineering track is to lift terms in multilingual sources (resulting from the linguistic engineering track in section III.A) to the conceptual level in order to tackle the problems of ambiguity and misunderstanding. It results in a domain ontology – a HRM ontology. Another aim of this track is to enrich the concepts in the domain ontology in a more fine-grained manner.

1) DOGMA approach to ontology engineering

DOGMA is introduced here as the basic ontology framework. A mapping from linguistic terms to ontologies is automatically processed by applying the lexon engineering method. As it is beyond the scope of this paper, we refer to the relevant report for the details.

Inspired by tried-and-tested principles from conceptual database modeling, DOGMA (Developing Ontology-Grounded Methods and Applications, [12], [17]) is designed as a methodological framework for ontology engineering. In the DOGMA framework, an ontology is composed of two layers: the lexons and the commitments.

---

The ‘Category terms’ pane in the MCFE software thus lists all possible terms that are used to denote a specific category.

---

[13] Terminology engineering is in this sense not strictly confined to the identification and defining of terms but is also focused on knowledge modeling through the organization of terms in a categorization framework. We will show in this section how the preparatory work for ontology engineering (section III.B) is realized by adopting principles of Terminography in the linguistic engineering track.

[14] Terminology engineering is in this sense not strictly confined to the identification and defining of terms but is also focused on knowledge modeling through the organization of terms in a categorization framework. We will show in this section how the preparatory work for ontology engineering (section III.B) is realized by adopting principles of Terminography in the linguistic engineering track.

---

http://www.onetcenter.org/

http://www.starlab.vub.ac.be/website/lexon.engineering

On the one hand, a conceptual database modeling approach to ontology engineering is promising and feasible; but on the other hand, ontology modeling is NOT equivalent to database modeling. Unlike data models, the fundamental asset of ontologies is their relative independence of particular applications.
1) Lexon. A lexon, which represents a simple fact, is a quintuple \(<γ, t_1, r_1, t_2, γ_2>\), where \(t_1\) and \(t_2\) are terms that represent two concepts in some language (e.g., English). \(r_1, r_2\) are roles (\(r_1\) corresponds to “role” and \(r_2\) is “co-role”) referring to the relationships that the concepts share with respect to one another. \(γ\) is a context identifier which serves to disambiguate the terms \(t_1, t_2\) into the intended concepts, and in which the roles \(r_1, r_2\) become “meaningful”.

For example, we have a lexon \(<γ, \text{employee}, \text{work for}, \text{is the working place of}, \text{enterprise}>\) indicating a fact – (an) employee works for (an) enterprise.

2) Commitment. The ontological commitment layer formally defines rules and constraints by which an application (or “agent”) may make use of lexons. A commitment uses application specific instantiation, integrity constraints and logical connections. It bridges the ontology base and the applications, aiming to find a balance between monolithic representation of generic and specific, declarative and operational semantics of the application domains.

A commitment needs to be expressed in a commitment language. A commitment is expressed as a constraint (or rule) applied on the semantic path. Suppose we have a lexon \(<γ, \text{employee}, \text{work for}, \text{is the working place of}, \text{enterprise}>\) with a constraint - “one employee works for at most one order manager”. We apply the uniqueness constraint UNIQ on the lexon \(p1\) written as: \(p1 = [\text{employee}, \text{work for}, \text{is the working place of}, \text{enterprise}]: \text{UNIQ}(p1)\).

2) Meaning Evolution Support Systems

DOGMA-MESS (DOGMA – Meaning Evolution Support System) is a methodology to support interorganizational ontology engineering [6] and to assist a community of domain experts to gradually enrich ontologies.

As the discussion of context is not the paper focus, we use \(γ\) as a specific context identifier for the lexon in this section. In the following sections, we omit the context identifier.

For instance, a template of “Competence” defines that a competence needs to have “Competence level”, “Action” and “Actor”. The “Action” has a certain “Quality”. When a domain expert gets the template, he may specify the “Competence” as “Speech clarity”, the “Competence level” as “Very good”, the “Action” as “Speak” and “Understand”, and the “Actor” as “Person actor”. Fig. 6 illustrates such a template and its specifications. The outer white boxes represent the template; and the inner blue boxes show the specialization.

The templates are constructed by analyzing well-structured linguistic resources that result from the linguistic engineering track (section III.A). DOGMA-MESS allows the domain experts to describe meanings of a concept from their own organizational perspectives. The models and specifications (such as Fig. 6) enrich the domain ontology by adding an extra level of detail, which allows analysis at a more fine-grained level of knowledge.
C. Group Decision Engineering

In Fig. 1, the track of group decision engineering contains the activities of teaming up a group (of decision makers), studying this group and designing decision semantics for Semantic Decision Table (SDT). The activities of teaming up a group and studying this group are specified in [11].

The SDT is a collaborative decision supporting system ([19], [20],) which is modeled based on McGrath’s schema (p. 13, [11]) and DOGMA (Section III.B). Semantics, a group of decision makers, mutual understanding and collaborative environment(s) are at the heart of SDT.

The model shown in Fig. 7 has been thoroughly discussed in [21]. The community aspect of the semantic decision tables is stressed and analyzed.

The semantics of SDT, which is derived from decision makers and stored together with domain ontologies, includes decision rules and decision semiotics. The decision rules are modeled as the SDT commitments (also called semantically grounded decision rules). The decision semiotics is modeled as SDT lexons (decision binary facts).

An SDT contains richer decision rules than a traditional decision table. In a traditional decision table, the conditions and decisions within a decision column are connected with the implication logic connective – if…then. The commitments of an SDT may carry more types than mere implication. In [19], different types of SDT rules are grouped and discussed. More specifically, an SDT commitment contains at least one constraint in the following categories:

- Qualifications, which include Cardinality, Quantification, Value constraint, Modification, Time, Aspect and Identification Schema.
- Set constraints, which contain Intersection, Complementation and Union.
- Parameters.
- Logical connectors, which are Negation, Conjunction, Disjunction, Implication, and Modality.
- Control operators, which cover Dependence, Precedence and Loop.

For instance, a decision lexon \(<r, \text{competence}, \text{has}, \text{is of}, \text{competence level}>\) can be constrained by the Modal Necessity\(^8\) as:

\[ P1 = [\text{competence, has, is of, competence level}]:L(P1(\text{has, is of})). \]

The commitment means that a Competence MUST have a Competence Level. A decision commitment is designed by the decision group through a formal commitment session, which depends strongly on its decision tasks in a specific moment.

The main goals of using SDT in the PROFILE COMPILER are (1) to elaborate concepts in available domain ontologies (which result from the ontology engineering track in section III.B), (2) to provide application semantics (or decision semantics) for the PROFILE COMPILER, and (3) to generate

\(^8\) L used in the commitment is a necessity constraint of Modal Logic. The SDT commitments can be modeled in a tool called SDT Constructor, which is recently developed as a plug-in in the DOGMA Studio Workbench V1.0.
IV. Design Decision Semantics

The definition of competency given in section II highlights a singular dimension of a competency, which adopts the traditional view of KSA (Knowledge, Skills and Behavior Aptitudes). Knowledge includes theoretical knowledge (e.g. educational background, diploma) or procedural knowledge (e.g. knowing how to program java code). Skills designate the know-how in both the senses of formalized (e.g. managing projects, convincing people) and empirical (e.g. tricks and talents). Behavior aptitudes describe the observed behavior of a person during a mission (e.g. patient, precise).

Accordingly, we design the constrained decision rules as:

\[
P_2 = [Ki, is, KPi]: P(P_2(is)) \]

Knowledge (Ki) is POSSIBLY theoretical knowledge (TKi).

\[
P_3 = [Ki, is, KPi]: P(P_3(is))\]

Knowledge (Ki) is POSSIBLY procedural knowledge (KPi).

\[
P_4 = [C, has, is of, Ci]: P_4(C) = \{Ci\}\]

Competency (C) is a SET OF competences (Ci).

\[
P_5 = [C, has, is of, CC]: P_5(C) = \{P_5(CC)\}\]

Competency (C) NECESSARILY has common competency (CC); 2) Common competency (CC) is a SUBSET OF competency (C)

\[
P_6 = [CC, has, is of, CCi]: P_6(CC) = \{P_6(CCi)\}\]

Common competency (CC) is a SET OF common competences (CCi).

\[
P_7 = [Ci, is, is, Bi]: P_7(is)\]

A competency (Ci) is POSSIBLY a behavior aptitude (Bi).

\[
P_8 = [B, has, is of, Bi]: P_8(B) = \{P_8(Bi)\}\]

Behavior aptitude set (B) is a SET OF behavior aptitudes (Bi).

\[
P_9 = [S, has, is of, Si]: P_9(S) = \{P_9(Si)\}\]

Skill set (S) is a SET OF skills (Si).

\[
P_{10} = [Ci, is, is, Si]: P_{10}(is)\]

A competency (Ci) is POSSIBLY a skill (Si).

These commitments at the meta-level are embedded in the PROFILE COMPILER. In addition, the community involved in our current HRM projects brings forward the following requirements:

- Common competency, such as linguistic skills and computer knowledge, is required by all kinds of expertise.
- Every competence has its own competence level, which is measured by a set of criteria. The criteria are given by O*NET and can not be modified.
- Competences are sometimes relevant to each other. Hence, proper connections need to be established.
- For each job type, all the competences are not equivalently important. The importance is defined by the domain experts and/or the end users.

1) With respect to the first requirement, the domain experts predefine the set of common competences manually. In the current version of PROFILE COMPILER, only linguistic skills and computer skills are included in the common competency set.

Accordingly, the SDT commitment on P5 is enriched as:

\[
P_5 = [C, has, is of, CC]: L(P_5(has, is of)), P_5(C) = \{P_5(CC)\}, CC = AND(CL, CK)\]

Where CL is the linguistic competency set and CK the computer competency set.

2) As the ontologies captured in the ontology engineering track (section III.B) contain the necessary description of competence levels for each competence, the second requirement is automatically fulfilled.

3) In order to fulfill the third requirement, we need to establish the dependencies properly for the competences. For instance, a decision rule – if a person has competency of speech clarity, then he has the competency of oral expression can be defined as:

\[
(P_{11} = [person, has competency, is competency of, speech clarity], P_{12} = [person, has competency, is competency of, oral expression]) \]

This kind of commitments expresses a dependency between several concrete competences.

4) With regard to the forth requirement, we need to set up a Score of Importance for every competence. The score of importance is predefined by HRM domain experts or set by the end users.

4.1) for the case of predefining the score of importance, domain experts need to manually set it for each competence

\[1\]

A sometimes is also known as Abilities or Attitude. Aptitudes describe what a person has the ability to do for the future mission. It implies the abilities and attitude.

\[1\]

The comment starts with ‘//’
that describes a job. E.g. an actor should have the highest
importance score (5) for oral expression. A commitment
statement is thus written as:

\[ (P13=\text{[actor, has, is of, competence]}, P14=\text{[oral expression, is a, is, competence]}, P15=\text{[competence, has, is of, importance score]} ) : \text{IMP(AND(P13,P14),P15(importance score))==5)}].

4.2) for the case of defining competence level by the end
users, we use the following method: Let the predefined
importance score of a competence be \( L_i \). We allow the end
users to define the importance score, the result of which is \( L'_i \).
The weights \( W_i \) and \( W'_i \), which are used to balance \( L_i \)
and \( L'_i \), are expressed as percents. We integrate the method
the following commitment statement:

\[ P16=\text{[Ci, has, is of, CLi]} : \text{L(P16(has, is of))}, P16(\text{CLi})\geq 0. \] // A competence (Ci)
NECESSARILY has a competence level (CLi), which has a NONNEGATIVE value.

\[ \text{IMP((P16CLi==0), INTERS(P16(Ci),P16(C)=NULL)}]. \] // If a competence level (CLi)
value is 0, then the competence (Ci)
does NOT BELONG TO the competency set (C) of a job (the intersection of
these two sets are empty).

\[ P17=\text{[Ci, has, is of, II]}:P17(\text{II})\geq 0, P17(\text{II})\leq M_i, II=Li*W_i+Li'*W'_i. \] // A
competence (Ci) has a score of
importance (II), which is MORE THAN OR
EQUAL TO 0 and LESS THAN OR EQUAL TO the
maximum level value (M_i).

The SDT commitments shown above need to be further
mapped into programming code for the execution. It is
important to know that, in the PROFILE COMPILER, SDT is
used for three purposes: (1) to elaborate concepts in available
domain ontologies, (2) to provide application semantics (or
decision semantics) for the PROFILE COMPILER, and (3) to
generate tabular reports inquired by end user. In the next
section, we will briefly present the current state of the
PROFILE COMPILER, which is a tool embedded with SDT.

V. PROFILE COMPILER: CURRENT STATE

The current version of PROFILE COMPILER\(^{14}\) supports
users to search and browse required competences of certain
jobs. It functions based on a query (e.g. search, browse)
provided by a user. The results are in English, Dutch and
French.

For instance, a user may browse the definition and
competences of “actor” by clicking ‘actor’ item in the
ontology tree. Fig. 8 illustrates part of the results in English.

The PROFILE COMPILER generates tabular reports based
on queries provided (or chosen) by users. The queries make
use of existing SDT commitments that are the results in
Section IV. There are two kinds of queries: 1) generate
specifications for a certain job; 2) generate matched jobs for
an expert.

Fig. 9 is a tabular report of proposed matched job list. A user
first has to provide his profile set – competences and
competence level. The PROFILE COMPILER analyzes the
profile and proposes matching jobs according to the decision
rules in the SDT commitment set, e.g. the score of importance
different competences and skill. Similarly, a tabular report
contain specifications of competences for a job – ‘actor’.

VI. RELATED WORK

In the domain of Human Resource Management (HRM),
promising related works on using ontologies have been
carrying on in both the research and industrial fields. For
instance, the authors in [15] have developed a generic
framework for developing competency referencing systems in
which ontologies are used to represent the semantics of the
elements that constitute individual competency profiles, such
as the knowledge, the behavior and the objectives of a job
candidate. Schmidt et al. [16] use ontologies extensively for
several different purposes\(^{16}\) (at different levels), e.g.
knowledge management, business process management,
competence management. There exist quite some existing

\(^{14}\)http://dogma.vub.ac.be:8080/ProfileCompiler/

\(^{15}\) Note that ‘actor’ contains more competences than shown in the
screenshot.

\(^{16}\) The design overview: http://professional-learning.eu/index.shtml
online systems, which try to support HRM activities such as recruitment by making use of competence concepts. For instance, in [9], ontologies are used to design a recruitment portal (WebDeEmpleo) and guide an end-user in his job searching process.

From our experiences in projects, such as the IWT POCEHRMOM and the EU PROLIX projects, we have learned that systems such as WebDeEmpleo would significantly improve if users can define competence profiles on the bases of common reference frameworks as the PROFILE COMPILER does.

In addition, we stress the importance of designing a job profile by a group in a collaborative decision making environment. In this paper, the cross interests of a community is extracted as the domain ontology. The organizational knowledge, such as the semantically rich decision rules, is coded into Semantic Decision Table (SDT) commitments. The SDT is coupled in the PROFILE COMPILER. The idea is to embrace both the group decision making knowledge and the individual decision tasks.

VII. CONCLUSION, DISCUSSION AND FUTURE WORK

Our objective in this paper is not to design a specific ontology-based application. Instead, we tend to study community-grounded application semantics in its own development lifecycle development in general. Ontology engineering, methodology, linguistic tracks and group decision support are tailored together. The communities of linguistics, ontology domain experts and decision group are respectively responsible for the tracks of linguistic engineering, ontology engineering and group decision engineering.

Thanks to the DOGMA-MESS methodology, domain experts are able to actively, gradually enrich the ontology at the concept level from their individually organizational perspective, hereby providing one extra level of details.

As the result of the engineering trucks in Fig. 1, the PROFILE COMPILER provides ontology-based, multilingual online services to support collaborative decision making for HRM. It embeds Semantic Decision Tables (SDT) in the modern Semantic Web technologies.

The SDT, which can be considered as the decision rule storage, allows companies to compile competence-based job profiles on the basis of a large, multilingual, ontology-based knowledge resource of competences and jobs. More specifically, SDT can be used in decision sessions to describe a job profile, find specific expertise and propose a matching job/expertise. The individual (e.g. a human resource manager) can gather the personalized competence profiles and contribute to the design of the compiler rule within a group (e.g. a community of small and medium-sized enterprises).

This paper brings out the importance of community aspect in the system development lifecycle. As early mentioned as the paper motivation, the idea embraces many advantages. For instance, the knowledge resources (terminologies, ontologies, group decision making tasks) are no longer static. They evolve gradually, systematically with the respect to the user communities. The decision rules, which make use of domain ontologies and terminologies, are well separated from the whole system. The community individuals (e.g. decision makers) are free to couple those rules by simply writing SDT commitments.

However, we discuss that the complexity and time consuming problem might be the main disadvantages. We have been busy assisting different communities from different domains for a very long time. It is not easy for the community individuals to learn how to specify concepts based on ontology templates and how to formalize SDT etc. In the future, we will focus more on the experimental setup and improve the design.

ACKNOWLEDGMENT

We’re pleased to thank Jan Vereecken for his help on the software development. This research is funded in the framework of IWT-TETRA (http://www.iwt.be) by the Flemish government and by seven Flemish enterprises: Actonomy (http://www.actonomy.com), Co:Inpetto (http://www.co-inpetto.be), Jobs & Careers (http://www.jobscareer.be), Linking (http://www.joining.be), Synergetics (http://www.synergetics.be), T-Groep (http://www.t-groep.be) and The Profile Group (http://www.profilegroup.com). The research performed in the context of this paper is also partly sponsored by the Brussels Capital Region and EU FP6 PROLIX project.

REFERENCES


