Integrated semantic-based composition of skills and learning needs in knowledge-intensive organizations

S. Colucci, T. Di Noia, E. Di Sciascio, F.M. Donini*, A. Ragone
SisInfLab
Politecnico di Bari, Bari, Italy
* Università della Tuscia, Viterbo, Italy

1 Introduction

Creation and maintenance of competencies is one of the most strategic activities in organizations, especially in knowledge intensive ones, as is the case of consulting companies. The role of core competencies in making a company achieve competitive advantage has been widely investigated in (Hamel and Prahalad, 1990). Other studies (Gronau and Uslar, 2004) show that the return on investment is significantly impacted by enriching knowledge management systems companies use, with components for the specific management of skills(*Skill Management Systems*, SMS).

One of the services SMS should provide is the search for competencies inside the organization: an organization dealing with a task to perform will firstly check for the required skills among the available personnel, an activity whose complexity grows up with the size of the company. If such a search process leads to discover lack of needed competencies, organizations may hire external personnel, or encourage internal personnel to learn new competencies on the unavailable skills. The current availability of several, well-organized, e-learning modules, makes such possibility appealing and economically advantageous.

A skill management system performing both the process of searching among available skills and facilitating the creation of missing ones can hence be a noteworthy source of competitive advantage for a knowledge-intensive organization.

We present here an approach and a system for supporting the whole process of skills retrieval and creation inside a company: the proposed framework exploits recent advances in semantic-based inference services and technologies.

It is noteworthy that the terms skill and competence are not kept distinct in the rest of the chapter: they both are meant to describe any ability or sort of knowledge held by individuals.

The approach exploits the formalism and the reasoning services provided by Description Logics(DLs) and is fully in the Semantic Web initiative mainstream.

It is well known that standard reasoning services from DLs can be used to evaluate if the individual profile and the task–both described in DLs formalism–completely match, *e.g.*, the profile is classified by the task. Yet, usually one is not only interested in perfect matches, which can be rare when complex and expressive descriptions are used. If we revert to classical unstructured-text Information Retrieval systems, we may obtain a similarity-based match, but such matches are only probabilistic and, for example, two fragments of descriptions such as "C++ programmer" and "experienced analyst and developer using Object Oriented languages" would simply get no similarity. Nevertheless, it is obvious, from the above example, that a human user would infer that the two descriptions are at least a potential match, and would ask which are the Object Oriented languages known to the individual. Hence, also potential matches have to be taken into account; we still care about the reasons why the descriptions do not exactly match with each other, in order to rank all the individuals potentially matching the task to perform, and individuate further competencies needed.

We employ two non standard reasoning services proposed and studied for Description Logics(Di Noia et al., 2003) for evaluating such reasons: Concept Contraction for discovering the characteristics in the individual profile that are in conflict with the ones specified in the task description – this process is performed in order to suggest a belief revision; Concept Abduction for explaining the characteristics required for carrying out the task and missing in the individuals profile. Let "experienced analyst and developer using Object Oriented languages" and " C++ novel programmer" be respectively a required task and an individual profile: a contraction process will return the information that in order to gain a potential match the requirement about experience has to be given up and the rest of the demand may be kept. If we consider instead the two fragments before "C++ programmer" and "experienced analyst and developer using Object Oriented languages" respectively as task and individual profile the abduction process would return the concept "C++" cause it is not explicitly said in the profile the individual knows about C++, while we can state that the individual knows object oriented programming, which is part of the background of a C++ programmer.

The choice of the management in the assignment process may revert on a single individual(*One to One Skill Matching*) or on an ad-hoc composed team(*Many to One Skill Matching*). When a team has to be composed, a Concept Covering Problem (Colucci et al., 2005a) is solved. The Problem can be seen as analogous - yet with noteworthy peculiarities- to a classical set covering problem in a semantic-based framework, because of its definition in terms of Concept Abduction.

The approach so far outlined performs the process of searching among available skills; when the organization personnel is not endowed of all the skills required for needed tasks, missing competencies are individuated by exploiting the abduction service and can become part of the learning process employees may be asked to start.

The nice property of the proposed approach is that the e-learning modules discovery and composition process can be basically carried out iterating the formerly outlined process, as the unavailable competencies become a specification of required courseware and are used to discover and compose, using semantics of descriptions, learning resources covering as much as possible the learning need, and orchestrate resources according to specified prerequisites. The current implementation of our approach extends to this aim LOM standard header (IEEE, 2002), embedding a structured module description expressed in an OWL-DL (OWL, 2004) subset.

The rest of this work is organized as follows: in Section 2 a literature review on skill management and e-learning systems is provided; an overview on Description Logics and their relationship with Semantic Web languages is also given to make the proposed semantic-based approach understandable. The issues, problems and controversies the chapter aims to solve are presented in Section 3 to motivate our solution proposal. The system is then outlined with the help of an example. Conclusions and future trends close the chapter.

2 Background

2.1 Skill Management Systems

Systems and techniques for Skills management have recently become the object of a growing interest, as knowledge and expertise of individuals have been acknowledged strategic assets of knowledge-intensive companies (O'Leary, 1998a; Hamel and Prahalad, 1990).

In (Gronau and Uslar, 2004) the role of skill management systems in organizational activities such as expert finding, personnel recruitment, personnel development and project management has been underlined. All these activities need the definition of the skills individuals are endowed with. Such an activity traditionally involves human judgment in individuating and classifying skills hold by individuals, in evaluating the degree of competence and in keeping up-to-date individuals profiles. IT-supported system use is then suggested in managing companies competencies in order to downsize the subjectivity of human evaluation. Matching personal profiles is an activity required in an arising number of scenarios, ranging from recruitment agencies and human resource organizational units to dating services. Those contexts, even in their deep dissimilarity, share the need to satisfy a list of individual requests, by matching them with available individual profile offers. The match between individual profiles we are interested in is, obviously, not an exact one, which is both quite simple and rare. Given a task description and individual profile descriptions, the matchmaking process has to return one or more best possible matches among the available ones.

The problem of expert finding is also modeled in (Yimam, 2000), in which the requirements of a skill finding approach are formally outlined in a domain analysis. A server architecture for expert finding, based on the previously mentioned domain analysis is also proposed.

The use of ontologies as knowledge repositories has now become almost common in novel knowledge management architectures, in order to give a common vocabulary and to use inference services on elicited knowledge(O'Leary, 1998a; O'Leary, 1998b).

Skill management systems presented in literature, almost all embedding skill searching facilities, may be classified in two categories including respectively non ontology-based and ontology-based systems.

Among non ontology-based approaches database querying and similarity between weighted vectors of stemmed terms, typical of text-based Information Retrieval, have been used to evaluate possible matches (Veit et al., 2001). Obviously, forcing profiles to be expressed by data structures or vectors of terms does not allow to deal with incomplete information, always present in matchmaking context in the form of either unavailable or irrelevant information.

Skill matching has been also modeled as a bipartite graph in which the first set of vertices includes assignees and the second one includes tasks to be performed(Saip and Lucchesi, 1993). Edges belonging to this graph link people to tasks. By determining a cost function that associates each edge with a real value, a weighted bipartite graph ensues, which results in a well known problem in Operational Research area, the Assignment Problem (Kennington and Wang, 1991; Galil, 1986; Hillier and Lieberman, 1995).

Among proposals on the subject, in (Sure et al., 2000) two skill matching systems, *ProPer* and *OntoProper*, were presented, both storing in a database skill profiles represented as vectors and using approaches from decision theory to allow for approximate match, not obtainable with plain database queries. *OntoProper* embeds also an ontology, reducing skill database maintenance effort by enriching the database with ground and inferred facts from secondary information, such as project documents. But both systems lack of an ontology as skill repository, allowing to infer on previously introduced profiles.

In (Becerra-Fernandez, 2000) two People Finder Knowledge Management Systems are proposed: the Searchable Answer Generating Environment(SAGE) and the Expert Seeker. Both systems use DBMSs as skill repositories and query engines performing a keyword search for expertise, even if the second one provides more search options. Even though proposing a database approach, the paper underlines the need to employ artificial intelligence technologies in People Finder Knowledge Management Systems in order to infer new knowledge from elicited skills and to keep automatically up-to-date profiles employing data mining techniques.

Also agent technologies have been employed to support the search for the right expert: in (Garro and Palopoli, 2003) it is proposed an XML multi-agent system providing support to management in searching the most suitable employee for a specific job, together with many other facilities. In (Sugawara, 2003) an agent-based application for supporting job matchmaking is proposed, focusing on the telework scenario.

In (Lau and Sure, 2002) an ontology based skill management system is proposed, allowing employees to elicit their skills and providing an advanced expert search within a company intranet.

In (Hefke and Stojanovic, 2004) a semantic based portal is proposed. The portal answers users queries about tasks to perform by providing ad-hoc organizational teams. The user request is formalized as a query searching the competences required for the task in the ontology used as skill repository. The system returns a set of one or more workers able to cover all the competences required for the task. All the available sets are ranked on the basis of the ontological closeness of query concepts to concepts formalizing skills held by proposed people.

In (Liu and Dew, 2004) a system integrating the accuracy of concept search with the flexibility of keyword search is proposed to match expertise within academia. The system is based on the use of semantic web technologies and in particular on RDF and XML in order to extract expertise integrated profiles from heterogeneous information sources.

Our approach takes full advantage of structured, ontology-based, descriptions, adopts an open world assumption –the absence of a characteristic in a description is not interpreted as a constraint of absence; instead, it is considered as a characteristic that could be either refined later, or left open if it is irrelevant– typical of knowledge representation. It obviously allows to find a set of one or more individuals that, based on provided skills descriptions, cover the requested task, but also, when a completely satisfactory set cannot be retrieved due to lack of requested skills, provides a logic-based answer to what is missing, or what should be revised in the task request to cover it based on available skills.

Skill management can be characterized in terms of multiplicity relationships between individuals skills and tasks to be accomplished (Colucci et al., 2003b):

- *one to one*: *one* task or job profile has to be matched with *one* individual;
- *many to one*: *one* task has to be assigned to *several* individuals that, together, are endowed of all skills requested for task realization;
- *one to many*: several tasks have to be matched with the skills of an individual able to accomplish them;
- *many to many*: several tasks have to be assigned to several available individuals;

In (Colucci et al., 2003c) a semantic based approach was presented to the problem of skills finding in an ontology supported framework. The framework is devoted to *one to one* skill matching and considers skill management as an electronic marketplace of knowledge in which skills are a peculiar kind of goods that have distinguishing characteristics with respect to traditional assets; buyers are entities that need the skills of people, such as projects, departments and organizations. On the other hand, knowledge sellers are individuals that offer their own skills. Obviously, descriptions of profiles share a common skills ontology.

Although semantic facilitators have been proposed in the literature for several scenarios (Trastour et al., 2002), (Sure et al., 2000), (Staab et al., 2001), they do not take full advantage of the ontological structure and limit their search to simple subsumption/classification matching.

Our approach, based on Description Logics formalization and reasoning, is oriented to finding the best individual for a given task or project, based on profile descriptions sharing a common ontology. The approach is able to cope with cases in which no perfect matches exist, *i.e.*, finding those available profiles that, for a given skill request best match, also if not identical, and vice versa. In particular we logically distinguish cases in which some skills in a request profile are not specified in the offered one, yet there is no contradiction and *e.g.*, further inquiries can be done (what is called a *potential match*); cases in which some skills in the request are in contrast with the given profile (what is called a *partial match*); in this case the one who is carrying out the search may check for unsatisfiable requests and eventually retract them if no better choice is at hand. It is noteworthy that our approach allows not only a logical categorization, but also a ranking of matches within each category. Notice that a *full match* is hence just a special case of a *potential* match.

In (Colucci et al., 2004) an approach was presented to the contemporary optimization of several *one to one* skill matching, endowing with semantics the process of searching solutions to task assignment. The problem classical application is to assign jobs to employees minimizing an objective function measuring the total cost of assignment. We may think of the cost function used for weighting arcs in term of suitability of persons to tasks. This assumption causes the objective function to measure quantitatively the effectiveness of performing all the tasks instead of the total cost of the assignment.

Evaluating the suitability of an individual to a job is a task traditionally performed by companies management on the basis of personal knowledge of workers. As a result, knowledge about coefficients measuring suitability of different matches is subjective and implicit, not allowing end users to clearly determine the reasons for match suggestions and to eventually revise them. Such an approach makes this process objective and explicit by using algorithms (Colucci et al., 2003b) exploiting reasoning services from DLs.

All the systems and approaches so far outlined deal with the search for skills among the internal personnel. It has been underlined earlier in the chapter the importance of creating new competencies when the available ones are not enough to perform all the needed tasks. In order to achieve such knowledge creation SMS may integrate components supporting the training process of employees, exploiting e-learning technologies.

The term e-learning has become common, describing several concepts, from complete web-based courses to distance learning and tutoring.

Recently, also thanks to various standardization efforts (IEEE, 2003), emphasis has been placed on the concept of learning object *i.e.*, small and easily reusable educational resources to be composed to allow personalized instruction and courseware creation (Ip et al., 2002; Cabezuelo and Beardo, 2004; Ajami, 2004; Vossen and Jaeschke, 2003).

Obviously, discovery and composition of such learning objects in an automated way requires the association of unambiguous and semantically rich metadata, defined in accordance with shared ontologies. The LOM –Learning Object Metadata (IEEE, 2002)– standard, though limited in the basic annotation items, allows to freely define annotated metadata describing a learning resource.

The semantic-based annotation of educational resources is hence fully in the stream of the Semantic Web initiative (Berners-Lee et al., 2001), and it can share with it both techniques and approaches (Sanchez and Sicilia, 2004; Bennacer et al., 2004; Gasevic et al., 2004). In particular, as more and more learning objects become available on the Web as services with well-defined machine interpretable interfaces as described *e.g.*, in OWL-S (OWL-S, 2004; Sycara et al., 2003), personalized learning units can be built by scratch, by retrieving learning resources. Automated composition of learning resources, exposed as web services for example, can then match a personalized learning need.

2.2 Description Logics and OWL Basics

Description Logics (DLs) (Baader et al., 2002) are a family of logic formalisms for knowledge representation whose basic syntax elements are *concept* names and *role* names. Concepts stand for sets of objects, *e.g.*, `ProcessEngineer`, `Graduate`, `BusinessApplication`, while roles, *e.g.*, `hasAbility`, `specialized`, link objects in different concepts.

Basic elements can be combined using *constructors* to form concept and role *expressions*. Based on the set of constructors adopted different DLs can be defined. Every DL allows one to form a *conjunction* of concepts denoted as $\sqcap$; some DLs include also disjunction $\sqcup$ and complement $\neg$ to close concept expressions under boolean operations. Roles can be combined with concepts using *existential role quantification* ($\exists$) , *e.g.*, $\texttt{Graduate} \sqcap \exists \texttt{hasAbility.NegotiationSkills}$, which describes the set of graduated people with negotiation skills, and *universal role quantification* ($\forall$), *e.g.*, $\texttt{Programmer} \sqcap \forall \texttt{hasMasterDegree.Engineering}$, which describes programmers having only an engineering degree.

Other constructs may involve counting, as number restrictions: $\texttt{Graduate} \sqcap (\leq\ 3\ \texttt{hasAbility})$ expresses graduates having at least three abilities, and $\texttt{AccountManager} \sqcap (\geq\ 2\ \texttt{hasTechnicalSkills})$ describes account managers endowed of at least two skills belonging to the technical area.

The representation of knowledge is achieved in DLs formalism by using concepts expressions to structure *inclusion assertions* and *definitions.* For example we could impose that programming may be partitioned into structural and object oriented using the two inclusions $\texttt{Programming} \sqsubseteq \texttt{StructuralProgramming} \sqcup \texttt{ObjectProgramming}$ and $\texttt{StructuralProgramming} \sqsubseteq \neg \texttt{ObjectProgramming}$.

We can state also that working teams have to be composed by at least two members as $\texttt{Team} \sqsubseteq (\geq\ 2\ \texttt{hasTeamMember})$. Historically, sets of such inclusions are called TBox (Terminological Box).

The basic reasoning problems for concepts in a DL are satisfiability, which accounts for the internal coherency of the description of a concept (no contradictory properties are present), and subsumption, which accounts for the more general/more specific relation among concepts, that forms the basis of a taxonomy. More formally:

- a concept C is satisfiable if there exists an interpretation in which C is mapped into a nonempty set, unsatisfiable otherwise. If a TBox $\mathcal{T}$ is present, satisfiability is relative to the models of $\mathcal{T}$, that is, the interpretation assigning C to a nonempty set must be a model of the inclusions in $\mathcal{T}$.

- A concept$C$ subsumes a concept D if every interpretation assigns to C a subset of the set assigned to D. Also subsumption is usually established relative to a TBox, a relation that we denote $\mathcal{T} \models C \sqsubseteq D$.

Also a TBox can be said satisfiable if there exist at least one model (i.e., an interpretation fulfilling all its inclusions in a nontrivial way).

It is easy to see that $\mathcal{T}$ in DLs represents what is called an ontology in a knowledge representation system. In the rest of the paper we refer to the $\mathcal{ALN}$(**A**ttributive **L**anguage with unqualified **N**umber restrictions) DL. The choice of such a DL is due to a trade off between language expressiveness and computational complexity of inference services(Brachman and Levesque, 1984).

Constructs allowed in an $\mathcal{ALN}$ DL are:

- $\top$ *universal concept.* All the objects in the domain.

- $\bot$ *bottom concept.* The empty set.

- A *atomic concepts.* All the objects belonging to the set represented by A.

- $\neg A$ *atomic negation.* All the objects not belonging in the set represented by A.

- $C \sqcap D$ *intersection.* The objects belonging both to C and D.

- $\forall R.C$ *universal restriction.* All the objects participating to the R relation whose range are all the objects belonging to C.

- $\exists R$ *unqualified existential restriction.* There exists at least one object participating in the relation R.

- $(\geq\ n\ R)$,$(\leq\ n\ R)$,$(=\ n\ R)$. Respectively the minimum, the maximum and the exact number of objects participating in the relation R.

We use a *simple-TBox* in order to express the relations among objects in the domain. With a *simple-TBox* in all the axioms (for both inclusion and definition) the left side is represented by a concept name.

Ontologies using the above logic can be easily modeled using languages for the Semantic Web (DAML+OIL, 2001; McGuinness et al., 2002; OWL, 2004). These languages have been conceived to allow for representation of machine understandable, unambiguous, description of web content through the creation of domain ontologies, and aim at increasing openness and interoperability in the web environment. The strong relations between DLs and the above introduced languages for the Semantic Web (Baader et al., 2003) is also evident in the definition of the three OWL sub-languages:

OWL syntax	**DL syntax**
$< owl:Thing/ >$	$\top$
$< owl:Nothing/ >$	$\bot$
$< owl:Classrdf:ID = "C"/ >$	C
$< owl:ObjectPropertyrdf:ID = "R"/ >$	R
$< rdfs:subClassOf/ >$	$\sqsubseteq$
$< owl:equivalentClass/ >$	$\equiv$
$< owl:disjointWith/ >$	$\neg$
$< owl:intersectionOf/ >$	$\sqcap$
$< owl:allValuesFrom/ >$	$\forall$
$< owl:someValuesFrom/ >$	$\exists$
$< owl:maxCardinality/ >$	$\leq$
$< owl:minCardinality/ >$	$\geq$
$< owl:cardinality/ >$	$=$

Table 1: Correspondence between OWL-DL and $\mathcal{ALN}$ DL syntax

OWL-Lite: allows class hierarchy and simple constraints on relation between classes;

OWL-DL: is based on Description Logics theoretical studies, it allows a great expressiveness keeping computational completeness and decidability;

OWL-Full: using such a language, there is a huge syntactic flexibility and expressiveness. This freedom is paid in terms of no computational guarantee.

The subset of OWL-DL Tags allowing to express an $\mathcal{ALN}$ DL is presented in Table 1. In the rest of the chapter we will use DL syntax instead of OWL-DL syntax, to make expressions much more compact.

3 Integrated Semantic-based Composition of Skills and Learning Needs

The skill management approach presented here is aimed at supporting decisions of company management in the whole process of finding competencies required for tasks to be performed. When a task has to be performed, it is reasonable that company management tries as a first attempt to find the skills needed among the internal personnel. It is up to the management either to assign the task to a single person(one to one matching) or to an ad-hoc created team(many to one matching). In both cases the match we are interested in providing is not just a perfect one: in case neither an employee nor a team is able to cover all the skills required for the task, we still search for the best available match, as humans would do. The proposed approach allows the management to know the reasons why the match is not full: which skills either the employee or the team is missing to perfectly cover the task. In other words the approach provides explanation hypothesis, exploiting the semantics of both the task and individuals profiles descriptions. On the basis of such explanation a personnel training process on lacking skills may take place. In case of one to one matching the employee assigned to the task has to effort the learning process. In case of a team ad-hoc created for performing the task, each member of the team has to effort a different learning process dependent on the background knowledge expressed by her profile. The courseware employees are asked for attending is automatically composed on the basis of the learning need resulting by the previous searching process.

3.1 The semantic base of the composition process

Standard inference services in DLs include subsumption and satisfiability. These basically return a yes/no answer. The scenario we outlined requires instead both explanation and belief revision in order to cope with cases in which no perfect march exists. Hereafter we recall basic definitions of two non-standard inference services that will be used to overcome highlighted limitations of classical ones. For the readability of the following definitions it can be helpful thinking of the concept S(supply) in terms of provided individual profile and D(demand) in terms of required task.

3.1.1 Non-standard Inferences for Semantic-based Belief Revision and Explanation

Let us consider two concepts $\mathcal{S}$ and $\mathcal{D}$, if their conjunction $\mathcal{S} \sqcap \mathcal{D}$ is unsatisfiable in the TBox $\mathcal{T}$ representing the ontology, *i.e.*, they are not compatible with each other, we may want, as in a belief revision process, to retract requirements in $\mathcal{D}$, G (for *Give up*), to obtain a concept K (for *Keep*) such that $K \sqcap \mathcal{S}$ is satisfiable in $\mathcal{T}$.

Definition 1 *Let $\mathcal{L}$ be a DL, $\mathcal{S}$, $\mathcal{D}$, be two concepts in $\mathcal{L}$, and $\mathcal{T}$ be a set of axioms in $\mathcal{L}$, where both $\mathcal{S}$ and $\mathcal{D}$ are satisfiable in $\mathcal{T}$. A* Concept Contraction Problem *(CCP), identified by $\langle \mathcal{L}, \mathcal{D}, \mathcal{S}, \mathcal{T} \rangle$, is finding a pair of concepts $\langle G, K \rangle \in \mathcal{L} \times \mathcal{L}$ such that $\mathcal{T} \models \mathcal{D} \equiv G \sqcap K$, and $K \sqcap \mathcal{S}$ is satisfiable in $\mathcal{T}$. We call K a* contraction *of $\mathcal{D}$ according to $\mathcal{S}$ and $\mathcal{T}$.*

Obviously, there is always the trivial solution $\langle G, K \rangle = \langle \mathcal{D}, \top \rangle$ to a CCP, that is give up everything of $\mathcal{D}$. In our skill matching framework, it models the (infrequent) situation in which, in front of some very appealing profile $\mathcal{S}$, incompatible with the requested task, a recruiter just gives up completely her specifications $\mathcal{D}$ in order to meet $\mathcal{S}$. On the other hand, when $\mathcal{S} \sqcap \mathcal{D}$ is satisfiable in $\mathcal{T}$, the "best" possible solution is $\langle \top, \mathcal{D} \rangle$, that is, give up nothing — if possible. Hence, a *Concept Contraction* problem is an extension of a satisfiability one. Since usually one wants to give up as few things as possible, some minimality in the contraction must be defined (Gärdenfors, 1988). In most cases a pure logic-base approach could be not sufficient to decide between which beliefs to give up and which to keep. There is the need of modeling and defining some extra-logical information. One approach is to give up minimal information (Colucci et al., 2003a). Another one considers some information more important than other and the information that should be retracted is the least important one, that is negotiable and strict constraints are introduced (Di Noia et al., 2004a).

When subsumption does not hold *i.e.*, a full match is unavailable, one may want to hypothesize some explanation on which are the causes of this result. In (Di Noia et al., 2003) the Concept Abduction Problem (CAP) was introduced and defined as a non standard inference problem for DLs to provide a logic-based answer to such question.

Definition 2 *Let S, D, be two concepts in a Description Logic $\mathcal{L}$, and $\mathcal{T}$ be a set of axioms, where both S and D are satisfiable in $\mathcal{T}$. A* Concept Abduction Problem *(CAP), denoted as $\langle \mathcal{L}, C, D, \mathcal{T} \rangle$, is finding a concept H such that $\mathcal{T} \not\models S \sqcap H \equiv \bot$, and $\mathcal{T} \models S \sqcap H \sqsubseteq D$.*

$\mathcal{P}$ is a symbol for a CAP, and $SOL(\mathcal{P})$ denotes the set of all solutions to a CAP $\mathcal{P}$.
In (Di Noia et al., 2003) also a minimality criteria for H and a polynomial algorithm to find solutions which are irreducible, for an $\mathcal{ALN}$ DL, have been proposed.

Given a CAP, if H is a conjunction of concepts and no sub-conjunction of concepts in H is a solution to the CAP, then H **is an irreducible solution**. The *rankPotential* algorithm (Di Noia et al., 2004b) allows to numerically compute the *length* of H. Obviously such length, for a complex ontology, is much more than a trivial depth in a hierarchy.

The solution to a CAP can be interpreted as what has to be hypothesized in S, and in a second step added to, to make S more specific than D, which would make subsumption result true. So, as Concept Contraction extends satisfiability, Concept Abduction extends subsumption.

The two inference services introduced above are athe basis of the logic-based approach we propose and detail in the following.

3.1.2 One to One Skill Matching

The process of assigning one employee to one task we propose here involves both belief revision and explanation. The request for assignment is supposed to be revisable if no individual profile potentially matching it is at hand. At the same time we are still interested in an explanation on the skills uncovered by each profile, either in case the request is contracted or not. Such information are used to rank all the possible candidates w.r.t. the job description, on the basis of both conflicting and hypothesized skills in profiles.

In particular we define here an algorithm $Assign(T, P, \mathcal{T})$ evaluating the space of possible matches between one needed task T and a set of available profiles $P = \{P_i\}, i = 1...n$, w.r.t. a TBox $\mathcal{T}$defined in $\mathcal{ALN}$.

Algorithm $\mathit{Assign}(T, P, \mathcal{T})$
input $P, T \equiv K \sqcap G$ concepts in $\mathcal{L}$ such that both $\mathcal{T} \models P \not\equiv \bot$ and $\mathcal{T} \models T \not\equiv \bot$
output $\langle P_{assigned}, H, G \rangle$
begin algorithm
    $P_{assigned} \equiv \top$;
    $U_{min} = \infty$;
    $N = rankPotential(\top, T, \mathcal{T})$;
    for each $P_i \in P$
        if $\mathcal{T} \models T \sqcap P_i \equiv \bot$
          then
          $\langle G_i, K_i \rangle = \mathit{contract}(P_i, T, \mathcal{T})$;
          else
          $K_i = T$;
        end if
        $H_i = \mathit{abduce}(P_i, K_i, \mathcal{T})$;
        $k_i = rankPotential(\top, K_i, \mathcal{T})$;
        $h_i = rankPotential(P_i, K_i, \mathcal{T})$;
        $g_i = rankPotential(T_i, K_i, \mathcal{T})$;
        $U_i = u(N, k_i, h_i, g_i)$;
        if $U_i < U_{min}$;
          then
          $U_{min} = U_i$;
          $P_{assigned} = P_i$;
          $H = H_i$;
          $G = G_i$;
        end if
    end for each
    return $\langle P_{assigned}, H, G \rangle$;
end algorithm

Notice that $H_i = \mathit{abduce}(P_i, K_i, \mathcal{T})$ determines that H is a solution for the $CAP\langle \mathcal{L}, T, P_i, \mathcal{T} \rangle$ while $\langle G_i, K_i \rangle = \mathit{contract}(P_i, T, \mathcal{T})$ determines $\langle G_i, K_i \rangle$ is a solution for the $CCP\langle \mathcal{L}, T, P_i, \mathcal{T} \rangle$.

The algorithm returns an assignee $P_{assigned}$ among the elements of P, together with an explanation H of skills to be hypothesized in $P_{assigned}$ to completely cover T and the concept G in case a preliminary contraction process on T is needed. The choice among P is made by minimizing the function U proposed in (Colucci et al., 2005c) computing a measure, indicating how good a match is, according to the simple closed form:

$$U(N, k, h, g) = \left| 1 - \frac{N}{N-g} * (1 - \frac{h}{k}) \right|$$

with the following meaning for parameters:

- k : length of K that belongs to the solution of a concept contraction problem between P_i and T

- h : length of H solution of a concept abduction problem between $K(P_i$ if no contraction is needed) and T
- g : length of G that belongs to the solution of a concept contraction problem between P_i and T
- N : length of T.

Such lengths are computed by applying the algorithm *rankPotential* as follows:

- $k = rankPotential(\top, K, \mathcal{T})$
- $h = rankPotential(P, K, \mathcal{T})$
- $g = rankPotential(T, K, \mathcal{T})$
- $N = rankPotential(\top, T, \mathcal{T})$

The rational of the closed form is given in (Colucci et al., 2005c). By choosing the candidate minimizing U, the algorithm takes into account both g and h, *i.e.*, a numerical measure of how much it has to be given up in the request T and how much to hypothesize in the profile analyzed at each stage.

3.1.3 Many to One Skill Matching

When an ad-hoc team has to be created for performing task T the process of team composition can be carried out by solving an extended Concept Covering Problem defined in terms of abduction and contraction.

We recall the definition of a Concept Covering Problem in terms of Concept Abduction, independently on the employed DL $\mathcal{L}$, originally given in (Colucci et al., 2005a):

Definition 3 *Let D be a concept, $\mathcal{R} = \{S_1, S_2, ..., S_k\}$ be a set of concepts in a Description Logic $\mathcal{L}$, and $\mathcal{T}$ be a set of axioms, where D and $S_i, i = 1..k$ are satisfiable in $\mathcal{T}$.*

1. *A* Concept Covering Problem *(CCoP), denoted as* CCoP$(\langle \mathcal{L}, \mathcal{R}, D, \mathcal{T} \rangle)$, *is finding, if it exists, a set $\mathcal{R}_c \subseteq \mathcal{R}$, such that both for each $S_j \in \mathcal{R}_c$, $\mathcal{T} \not\models \sqcap S_j \equiv \bot$, and $H \in SOL(\langle \mathcal{L}, \sqcap S_j, D, \mathcal{T} \rangle)$ is such that $\mathcal{T} \not\models H \sqsubseteq D$.*
2. *We call $\langle \mathcal{R}_c, H \rangle$ a* solution *for the CCoP $\langle \mathcal{L}, \mathcal{R}, D, \mathcal{T} \rangle$.*

In the same paper the algorithm $GREEDY solveCCoP$ exploited concept abduction to extend a tractable greedy set-covering algorithm (Cormen et al., 1990). We do not delve into details but notice that a Concept Covering Problem has noteworthy peculiarities w.r.t. a set covering one(Colucci et al., 2005a).

Here we propose the algorithm *TeamComposer*$(P, T, \mathcal{T})$ –where T is the needed task and $P = \{P_i\}, i = 1...n$ is a set of individual profiles both described w.r.t. a TBox $\mathcal{T}$– to take also concept contraction into account during the process of selection of team members.

1: **Algorithm** $\mathit{TeamComposer}(P, T, \mathcal{T})$
2: **input** concepts T, $P_i \in P$, where T and P_i are satisfiable in $\mathcal{T}$
3: **output** $\langle P_c, H, G_{contraction} \rangle$
4: **begin algorithm**
5: $\quad P_c = \emptyset$;
6: $\quad T_{uncovered} = T$;
7: $\quad G_{contraction} = \top$;
8: $\quad H_{fin} = \top$;
9: $\quad G_{fin} = \top$;
10: $\quad U_{min} = \infty$;
11: $\quad N = rankPotential(\top, T, \mathcal{T})$;
12: $\quad$ **do**
13: $\qquad P_{max} = \top$;
14: $\qquad$ **for each** $P_i \in P$ **such that** $P_c \cup \{P_i\}$ covers $T_{uncovered}$
15: $\qquad\quad$ **if** $\mathcal{T} \models T_{uncovered} \sqcap P_i \equiv \bot$ **then**
16: $\qquad\qquad \langle G, K \rangle = \mathit{contract}(P_i, T_{uncovered}, \mathcal{T})$;
17: $\qquad\quad$ **else**
18: $\qquad\qquad K = T_{uncovered}$;
19: $\qquad\qquad G = \top$;
20: $\qquad\quad$ **end if**
21: $\qquad\quad H = \mathit{abduce}(P_i, K, \mathcal{T})$;
22: $\qquad\quad k = rankPotential(\top, K, \mathcal{T})$;
23: $\qquad\quad h = rankPotential(P_i, K, \mathcal{T})$;
24: $\qquad\quad g = rankPotential(T, K, \mathcal{T})$;
25: $\qquad\quad U = u(N, k, h, g)$;
26: $\qquad\quad$ **if** $U < Umin$
27: $\qquad\qquad P_{max} = P_i$;
28: $\qquad\qquad H_{fin} = H$;
29: $\qquad\qquad G_{fin} = G$;
30: $\qquad\quad$ **end if**
31: $\qquad$ **end for each**
32: $\qquad$ **if** $(P_{max} \not\equiv \top$ **and** $K \not\equiv \top)$ **then**
33: $\qquad\quad P = P \backslash \{P_i\}$;
34: $\qquad\quad P_c = P_c \cup \{P_i\}$;
35: $\qquad\quad T_{uncovered} = H_{fin}$;
36: $\qquad\quad G_{contraction} = G_{contraction} \sqcap G_{fin}$;
37: $\qquad$ **end if**
38: $\quad$ **while** $(P_{max} \not\equiv \top$ **and** $K \not\equiv \top)$;
39: $\quad$ **return** $\langle P_c, T_{uncovered}, G_{contraction} \rangle$;
40: **end algorithm**

The algorithm tries to cover T "as much as possible", using the concepts $P_i \in P$. If a new individual profile P_i can be added to the already composed team P_c, *i.e.*, $\mathcal{T} \not\models (\sqcap_{P_k \in P_c} P_k) \sqcap P_i \equiv \bot$, then an extended matchmaking process is performed (rows 14–32). If P_i is not consistent with the uncovered part of the task $T_{uncovered}$, the latter is contracted and subsequently an abduction process is performed between the contracted uncovered task and P_i (rows 15–17). If P_i is consistent with $T_{uncovered}$, only a concept abduction problem is solved (rows 19–21). Based on the previously computed concepts G, K and H a global score is used as a metric to evaluate how good P_i is with respect to the covering set (rows 23–26).

We do not need to change the definition of Concept Covering Problem provided in (Colucci et al., 2005a): the problem we solve is still the one detailed in Definition 3. The distinguishing element between the two solving algorithms $GREEDY solveCCoP$ and *TeamComposer* is in the choice criterion among the P_i greedy selected to compose the team. In $GREEDY solveCCoP$ such choice was made on the basis of a minimality criterion on H, the solution of an abduction problem on $T_{uncovered}$– the part of the needed task yet to cover at each stage of the algorithm– and the selected profile P_i. In *TeamComposer* instead the choice criteria is the minimization of the function U explained in Section 3.1.2, which takes into account also a measure of how much the task request should be contracted before the members selection, thus improving the selection process.

The outputs of *TeamComposer* are:

- P_c : the set of employees composing the team
- $T_{uncovered}$: the part of the task description not covered by the ad-hoc created team
- $G_{contraction}$: the part of the task description given up to at the end of the whole team composition process

3.1.4 Composition of the training process

The matching process so far outlined returns an explanation on missing skills both in the case of assignment to one employee and of team composition. The formal representation of this explanation is given by H returned by *Assign* in one to one skill matching and by $T_{uncovered}$ returned by *TeamComposer* in many to one skill matching.

H or $T_{uncovered}$ represent the learning need to be covered by a courseware to be automatically generated by composing learning objects through a process exploiting the DL standard and non-standard inference services.

In (Colucci et al., 2005b) we proposed a general framework based on OWL technologies for composition; such framework can be easily integrated in existing metadata specifications, such as SCORM (SCORM, 2004), LOM (IEEE, 2002), IMS (IMS, 2001), Dublin Core (DublinCore, 1999), although we currently use a LOM extended header. The courseware composition was there considered as a learning objects (λ) retrieval problem. In this perspective, if there is a *learning need* and a repository of learning objects potentially satisfying the learner specifications, a solution to a λ-retrieval problem is:

retrieve (a sequence of) some λs from the repository such that their composition satisfies the learning need as far as possible.

In case a perfect covering of the learning need is not found, an approximate solution has to be taken into account, together with explanation hypothesis of what remains missing.

Formally learning objects and learning needs are defined as follows:

Learning Object[1] : $\lambda = \langle \lambda_D, \lambda_{\mathcal{BK}} \rangle$. *$\lambda_D$ describes the knowledge the user will acquire after she/he uses λ. Using a language endowed with a well-defined syntax and semantics, it models the* offered knowledge. *$\lambda_{\mathcal{BK}}$ is a representation of prerequisites in order to benefit from λ.*

Learning Need: $\rho = \langle \rho_D, \rho_{\mathcal{BK}} \rangle$, *where ρ_D is the description of the requested learning need and $\rho_{\mathcal{BK}}$ represents the background knowledge owned by the requester before looking for the courseware.*

According to the framework we propose in this chapter the learning need is the one returned by the previous searching process: ρ_D is exactly the explanation H or $T_{uncovered}$ returned in case of one to one or many to one skill matching, respectively.

As concerns the background knowledge $\rho_{\mathcal{BK}}$ held by the assignees when the training process starts we have to distinguish the multiplicity cases:

- *one to one skill matching*: $\rho_{\mathcal{BK}}$ is represented by the profile P_i returned by *Assign* ,*i.e.*, the profile of the individual selected for performing the task

- *many to one skill matching*: each member P_i of the ad-hoc created team P_c returned by *TeamComposer* has to start a training process in which $\rho_{\mathcal{BK}}$ is represented by her profile P_i.

Notice that we need to search for one courseware for each member of the ad-hoc created group because the background knowledge is typical of each individual and the learning process to initiate depends on the skills initially held by the members.

The algorithm *teacher* presented in (Colucci et al., 2005b) automatically computes a composite courseware. The algorithm takes as input a set of learning objects $\mathcal{R} = \{\lambda^i = \langle \lambda^i_D, \lambda^i_{\mathcal{BK}} \rangle\}$, the learning need $\rho = \langle \rho_D, \rho_{\mathcal{BK}} \rangle$, and an ontology $\mathcal{T}$ and returns the composite courseware $\Lambda(\rho, \mathcal{R})$ and the uncovered part, $\rho_{D_{uncovered}}$, of the request description ρ_D.

A composite courseware is a sequence of learning objects such that both the following conditions hold: it can be started using some background knowledge the requester owns ($\rho_{\mathcal{BK}}$) and the provided composite courseware covers the user request description (ρ_D).

More formally we need the following definition of couserware flow to define a composite courseware.

[1]Without loss of generality here we consider only the information needed for a semantic discovery and composition.

A **courseware flow** with respect to some initial background knowledge $\rho_{\mathcal{BK}}$, denoted as $\overline{\Lambda}(\rho_{\mathcal{BK}})$, is a finite sequence of learning objects $(\lambda^1, \lambda^2, ..., \lambda^i, ..., \lambda^n)$, where for each learning object λ^i belonging to to the courseware, all the following conditions hold:

1. the initial background knowledge, $\rho_{\mathcal{BK}}$, is at least $\lambda^1_{\mathcal{BK}}$, that is the background knowledge required by λ^1, the first Learning Object of the sequence. In order to learn from a sequence of learning objects (LOs), the user must have at least the prerequisites needed to learn from the starting LOs.

2. after using λ^{i-1}, the user has a background knowledge which is at least $\lambda^i_{\mathcal{BK}}$, *i.e.*, the one required by the *i-th* Learning Object. While benefiting of the composite LOs, the user acquires new knowledge which becomes part of her background. Such an *updated* background knowledge must satisfy the λ^i requirements.

The background knowledge of the learner before the fruition of λ^i is the conjunction of all the knowledge provided by λ^j_D, with $j < i$, and the initial background knowledge $\rho_{\mathcal{BK}}$.

Indicating with $\mathcal{BK}_i$ the background knowledge before using λ^i, using DL syntax, the following relation ensues:

$$\mathcal{BK}_i = \rho_{\mathcal{BK}} \sqcap \lambda^1_D \sqcap \lambda^2_D \sqcap ... \sqcap \lambda^{i-1}_D$$

We can now define formally a **courseware flow**.

Definition 4 *A* **courseware flow** *with respect to some initial background knowledge $\rho_{\mathcal{BK}}$ is a finite sequence of learning objects $\overline{\Lambda}(\rho_{\mathcal{BK}}) = (\lambda^1, \lambda^2, ..., \lambda^i, ..., \lambda^n)$ with $i = 1..n$, where for each $\lambda^i \in \overline{\Lambda}(\rho_{\mathcal{BK}})$ all the following conditions hold:*

1. $\rho_{\mathcal{BK}} \sqsubseteq \lambda^1_{\mathcal{BK}}$.

2. $\mathcal{BK}_i \sqsubseteq \lambda^i_{\mathcal{BK}}$.

We indicate with $\mathcal{D}_{\overline{\Lambda}}$, the set of learning objects descriptions in $\overline{\Lambda}(\rho_{\mathcal{BK}})$. $\mathcal{D}_{\overline{\Lambda}} = \{\lambda^i_D | \lambda^i \in \overline{\Lambda}(\rho_{\mathcal{BK}})\}$.

Based on the previous definition of **courseware flow**, it possible to define a **composite courseware** with respect to a request ρ.

Definition 5 *Let $\mathcal{R} = \{\langle \lambda^i_D, \lambda^i_{\mathcal{BK}} \rangle\}$, with i=1..k, be a set of learning objects λ^i, and $\langle \rho_D, \rho_{\mathcal{BK}} \rangle$ be a request for a courseware, such that λ^i_D, $\lambda^i_{\mathcal{BK}}$, ρ_D and $\rho_{\mathcal{BK}}$ are modeled as concept descriptions in a DL w.r.t. an ontology $\mathcal{T}$.*

A **composite courseware** *for $\rho = \langle \rho_D, \rho_{\mathcal{BK}} \rangle$ with respect to $\mathcal{R}$, denoted $\Lambda(\rho, \mathcal{R})$, is a courseware flow such that for each λ_j in the courseware flow, $\mathcal{D}_\Lambda = \{\lambda^j_D | \lambda^j \in \Lambda(\rho, \mathcal{R})\}$,* covers ρ_D.

At each stage of the algorithm *teacher* the learning objects to be added to the lesson flow are searched for only within the current *usable learning objects*. In particular we define an *usable learning object* as follows:

Definition 6 *Given a courseware flow $\overline{\Lambda}(\rho_{\mathcal{BK}}) = (\lambda^1, \lambda^2, ..., \lambda^n)$, we say that a learning object is a* **usable learning object** λ_u *for $\overline{\Lambda}(\rho_{\mathcal{BK}})$ if and only if*

1. $\lambda_u \notin \overline{\Lambda}(\rho_{\mathcal{BK}})$.

2. $\overline{\overline{\Lambda}}(\rho_{\mathcal{BK}}) = (\lambda^1, \lambda^2, ..., \lambda^n, \lambda_u)$ *is a courseware flow.*

A **usable learning object** λ_u for $\overline{\Lambda}(\rho_{\mathcal{BK}})$ is a learning object which can be used after the user benefits from $\overline{\Lambda}(\rho_{\mathcal{BK}})$, *i.e.*, its required background knowledge is provided by $\overline{\Lambda}(\rho_{\mathcal{BK}})$.

Actually, given a courseware flow, several usable learning objects exist.

Definition 7 *Given a courseware flow $\overline{\Lambda}(\rho_{\mathcal{BK}})$ and a set of learning objects $\mathcal{R} = \{\lambda^i\}$ we call* **usable set** *for $\overline{\Lambda}(\rho_{\mathcal{BK}})$, the set of all the $\lambda^i \in \mathcal{R}$ such that λ^i is a usable learning object for $\overline{\Lambda}(\rho_{\mathcal{BK}})\}$. $\mathcal{U}_{\overline{\Lambda}(\rho_{\mathcal{BK}})} = \{\lambda_u^i | \lambda_u^i$ is a usable learning object for $\overline{\Lambda}(\rho_{\mathcal{BK}})\}$*

The *usable set* is hence the set of all the learning objects that can be used after the user benefits from a courseware flow.

3.2 A System for integrated semantic-based composition

The described framework has been integrated in a prototype system supporting the whole process, sketched in Figure 1.

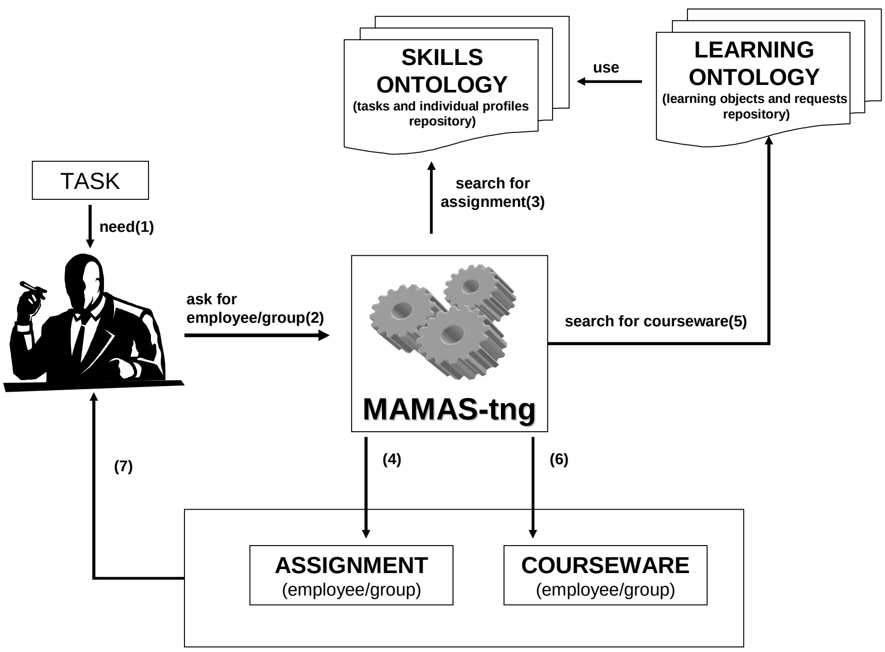

Figure 1: System Architecture

A *Skills Ontology* is used as vocabulary through which internal personnel profiles and needed tasks are formally described. A different *Learning Ontology* is instead used to describe the domain of learning and learning objects and the requests for coursewares descriptions. Both of the ontologies are about skills domain, but they look at it from different perspectives: the skills ontology stores information about what an employee "knows"; the learning ontology describes what learning objects "teach". After the explanation process it is possible to know which knowledge the assigned employees lack of: such information represents what they should learn through the learning process.

Consider, for example, the need for a *Process Engineer endowed with communication skills* and the available employee *John: Engineer, specialized in Plant Management and inclined to communication*, both stored in the skills ontology according to the following formalization:

Task $T = \texttt{ProcessEngineer} \sqcap \exists\texttt{hasAbility}$
$\sqcap\forall\texttt{hasAbility.CommunicationSkills}$

John $P = \texttt{Engineer} \sqcap \exists\texttt{hasAbility} \sqcap \forall\texttt{hasAbility.CommunicationSkills}$
$\sqcap\exists\texttt{specialized} \sqcap \forall\texttt{specialized.PlantManagement}$

T and P refer to the example skills ontology in Figure 2.

By applying the algorithm *Assign(P,T,$\mathcal{T}$)* we obtain the explanation on lacking skills:
$H_i = \exists\texttt{specialized} \sqcap \forall\texttt{specialized.ProcessControl}$.

Such learning need has to be mapped, according to the vocabulary of the example learning ontology in Figure 3, in a learning request for an automatic courseware composition:
$\rho_D = \exists\texttt{givesSpecialization} \sqcap \forall\texttt{givesSpecialization.ProcessControl}$.

In Figure 4 examples of the learning objects stored in the learning ontology are shown. According to such formalization the description component of each learning object (the filler of the role `hasDescription`) is evaluated for the creation of the courseware satisfying the learning request ρ_D. The background knowledge component (the filler of the role `requiresBackgroundKnowledge`) is instead used to evaluate whether the individual profile at hand is allowed to learn that module.

By looking at the given example it is easy to see that we need some mapping rules to convert the properties expressing knowledge needs in the skills ontology in the properties expressing teaching actions in the learning ontology. In particular we need mapping rules, which map patterns of expressions written according to the skills ontology into corresponding patterns in the learning ontology. Those mapping rules could be modeled as axioms in a "mapping ontology", taking our approach out of the $\mathcal{ALN}$ OWL-DL subset in which the algorithms we presented so far work.

Such language, though limited in expressiveness, has on the other side the advantage to keep acceptable the computational complexity of the proposed algorithms. In order not to lose such advantage we preferred to keep both the

$$
\begin{array}{l}
\texttt{Graduate} \equiv \exists\texttt{hasMasterDegree} \\
\texttt{NewGraduate} \equiv \texttt{Graduate} \sqcap \forall\texttt{hasExperience}.\bot \\
\texttt{Engineer} \equiv \texttt{Graduate} \sqcap \forall\texttt{hasMasterDegree.Engineering} \\
\texttt{AccountManager} \sqsubseteq \texttt{Graduate} \\
\\
\texttt{ProcessEngineer} \equiv \texttt{Engineer} \sqcap \exists\texttt{specialized} \\
\qquad\qquad\qquad \sqcap\forall\texttt{specialized.}(\texttt{ProcessControl} \sqcap \texttt{PlantManagement}) \\
\\
\texttt{Programmer} \sqsubseteq \exists\texttt{hasTechnicalSkills} \sqcap \forall\texttt{hasTechnicalSkills.Programming} \\
\texttt{WebProgrammer} \sqsubseteq \texttt{Programmer} \\
\texttt{Lead-role} \sqsubseteq \texttt{Ability} \\
\texttt{TeamCoordinator} \sqsubseteq \texttt{Ability} \\
\texttt{NegotiationSkills} \sqsubseteq \texttt{Ability} \\
\texttt{CommunicationSkills} \sqsubseteq \texttt{Ability} \\
\texttt{MeetingsCoordination} \sqsubseteq \texttt{CommunicationSkills} \sqcap \texttt{NegotiationSkills} \\
\texttt{ERPsystem} \sqsubseteq \texttt{BusinessApplication} \\
\texttt{FinancialSystem} \sqsubseteq \texttt{BusinessApplication} \\
\texttt{WebBasedTechnology} \sqsubseteq \texttt{Technology} \\
\texttt{ProgrammingLanguage} \sqsubseteq \texttt{Technology} \\
\texttt{Java} \sqsubseteq \texttt{ProgrammingLanguage} \\
\texttt{C++} \sqsubseteq \texttt{ProgrammingLanguage} \\
\texttt{Programming} \sqsubseteq \texttt{TechnicalSkills} \\
\texttt{ITConsulting} \sqsubseteq \texttt{TechnicalSkills} \\
\texttt{TechnicalWriting} \sqsubseteq \texttt{TechnicalSkills} \\
\texttt{English} \sqsubseteq \texttt{Language} \\
\texttt{German} \sqsubseteq \texttt{Language}
\end{array}
$$

Figure 2: An excerpt of the skills ontology

$$
\begin{array}{l}
\texttt{learningObject} \sqsubseteq \exists\texttt{hasDescription} \sqcap \exists\texttt{requiresBackgroundKnowledge} \\
\texttt{ProgrammingCourse} \equiv \texttt{learningObject} \sqcap \\
\qquad\qquad \forall\texttt{hasDescription.}(\exists\texttt{teachProgrammingLanguage} \sqcap \\
\qquad\qquad \forall\texttt{teachProgrammingLanguage.ProgrammingLanguage}) \\
\texttt{EnablingCourse} \equiv \texttt{learningObject} \sqcap \forall\texttt{hasDescription.}(\exists\texttt{teachAbility} \sqcap \\
\qquad\qquad \forall\texttt{teachAbility.Ability}) \\
\texttt{SpecializationCourse} \equiv \texttt{learningObject} \sqcap \\
\qquad\qquad \forall\texttt{hasDescription.}(\exists\texttt{givesSpecialization})
\end{array}
$$

Figure 3: An excerpt of the learning ontology

$$\begin{aligned}
&\texttt{JavaCourse} = \texttt{ProgrammingCourse}\\
&\qquad \sqcap\forall\texttt{requiresBackgroundKnowledge.Programming}\\
&\qquad \sqcap\forall\texttt{hasDescription.}(\forall\texttt{teachProgrammingLanguage.Java})\\
&\texttt{ProcessControlCourse} = \texttt{SpecializationCourse}\\
&\qquad \sqcap\forall\texttt{requiresBackgroundKnowledge.ProcessManagement}\\
&\qquad \sqcap\forall\texttt{hasDescription.}(\forall\texttt{givesSpecialization.ProcessControl})\\
&\texttt{MeetingCoordinationCourse} = \texttt{EnablingCourse}\\
&\qquad \sqcap\forall\texttt{requiresBackgroundKnowledge.}\top\\
&\qquad \sqcap\forall\texttt{hasDescription.}(\forall\texttt{teachAbility.MeetingsCoordination})\\
&\texttt{ProcessPlanningCourse} = \texttt{SpecializationCourse}\\
&\qquad \sqcap\forall\texttt{requiresBackgroundKnowledge.MeetingsCoordination}\\
&\qquad \sqcap\forall\texttt{hasDescription.}(\forall\texttt{givesSpecialization.ProcessPlanning})
\end{aligned}$$

Figure 4: Learning objects descriptions

skills and the learning ontology in an $\mathcal{ALN}$ DL and to write the rules out of the ontologies.

When the need for performing a task arises, the management starts either the process of assignment to one individual or of creation of an ah-hoc team.

The system makes calls to MAMAS (http://dee227.poliba.it:8080/MAMAS-tng)in order to exploit the inference services. MAMAS provides an assignment(either employee or group) together with a formal explanation of the reasons why the match is not perfect. Such explanation represents the lacking skills to be covered by a learning process the assignees (one or more) have to start; this explanation is then translated into a learning request according to the mapping rules. On the basis of the formalized request, MAMAS is called again to automatically compose the personalized coursewares endowing the assignees with the skills needed for the task. If a courseware completely fulfilling the learning need is not found, MAMAS will return an explanation about the skills still missing.

3.3 System behavior

In this section the whole process of composition of skills and learning objects is detailed with the help of an example.

Let us suppose a Company has the problem to assign the realization of a task according to the following, simple, specifications: *engineers, with lead-role, negotiation and communication skills and at least two years experience. The task requires Process Control, Web Technology, Process Plan and ERP systems knowledge and orientation to team work.* Such a request can be formalized in *DL* as:

$T = \texttt{Engineer} \sqcap \exists\texttt{specialized} \sqcap \forall\texttt{specialized.}(\texttt{ProcessControl}$

$\sqcap$`WebBasedTechnology` $\sqcap$ `ERPsystem` $\sqcap$ `ProcessPlanning`)
$\sqcap\exists$`hasAbility` $\sqcap$ $\forall$`hasAbility.`(`NegotiationSkills`
$\sqcap$`CommunicationSkills` $\sqcap$ `TeamCoordinator` $\sqcap$ `Lead-role`)
$\sqcap\exists$`hasExperience` $\sqcap$ $\forall$`hasExperience.`($\geq$ 2 `years`)

Suppose now the four individuals described in the following are available:

Julia : *Julia is a Process Engineer specialized in Business application, able to lead a team and coordinate meetings. She got a master degree recently.*

Tom : *Tom is a Web oriented Programmer. His favorite programming languages are Java and C++, used for web based application.*

Richard : *Richard is an engineer specialized in ERP systems. He has leading ability, particularly for working teams. He has a 3 years work experience.*

Alison : *Alison is an account manager specialized in IT-consulting and writing technical documents. She has a good knowledge of both English and German.*

The above employee are candidates to solve the task and their profile can be formalized as DL concepts:

$P_1(Julia)$ =`ProcessEngineer` $\sqcap$ $\forall$`specialized.BusinessApplication`$\sqcap$
$\forall$`hasAbility.`(`MeetingsCoordination` $\sqcap$ `TeamCoordinator`)
$\sqcap$`NewGraduate`
$P_2(Tom)$ =`WebProgrammer` $\sqcap$ $\exists$`programmLanguageKnowledge`$\sqcap$
$\forall$`programmLanguageKnowledge.`(`Java` $\sqcap$ `C++`)
$\sqcap\forall$`specialized.WebBasedTechnology`
$P_3(Richard)$ =`Engineer` $\sqcap$ $\forall$`specialized.ERPsystem`
$\sqcap\forall$`hasAbility.`(`TeamCoordinator` $\sqcap$ `Lead-role`)
$\sqcap\exists$`hasExperience` $\sqcap$ $\forall$`hasExperience.`($\geq$ 3 `years`)
$P_4(Alison)$ =`AccountManager` $\sqcap$ $\exists$`specialized`
$\sqcap\forall$`specialized.`(`ITConsulting` $\sqcap$ `TechnicalWriting`)
$\sqcap\exists$`hasLanguageKnowledge`
$\sqcap\forall$`hasLanguageKnowledge.English` $\sqcap$ `German`

In case the management decide to assign the task T only to one employee the algorithm $Assign(T, P, \mathcal{T})$ is applied, with $P = \{P_1, P_2, P_3, P_4\}$. *Assign* choses the candidate minimizing U: *Richard.*

Richard is characterized by the following solution to the contraction and the abduction problem, respectively:

G_{P_3} =$\top$
K_{P_3} =`Engineer` $\sqcap$ $\forall$`specialized.ERPsystem`
$\sqcap\forall$`hasAbility.`(`TeamCoordinator` $\sqcap$ `Lead-role`)
$\sqcap\exists$`hasExperience` $\sqcap$ $\forall$`hasExperience.`($\geq$ 3 `years`)
H_{P_3} =$\forall$`specialized.`(`ProcessPlanning`)
$\sqcap\forall$`hasAbility.`(`NegotiationSkills` $\sqcap$ `CommunicationSkills`)

Such results show that Richard profile is consistent with the required task, so that no contraction is needed and the concept K_{P_3} still equals P_3. H_{P_3} shows instead that, though Richard represents the best individual able to cover the task, he lacks of negotiation and communication skills and of a specialization in Process Planning. According to the definitions given in Section 3.1.2 the computed utility for such an assignment is $U_{P_3} = 0.23076922$. The analogously computed utilities of the other profiles are: $U_{P_1} = 0.25$ $U_{P_2} = 0.9230769$ $U_{P_4} = 0.84615386$.

H_{P_3} represents the learning need to be covered by an automatically composed courseware but it needs then to be rewritten in the learning ontology vocabulary, according to the rules contained in the mapping ontology, in the following learning request:

$$\rho = \forall \texttt{givesSpecialization.}(\texttt{ProcessPlanning}) \sqcap \\ \forall \texttt{teachAbility.}(\texttt{NegotiationSkills} \sqcap \texttt{CommunicationSkills})$$

As a result of the automated courseware composition process Richard is asked to learn `MeetingCoordinationCourse` and `ProcessPlanningCourse`, to be learned in the specified order. Richard holds the background knowledge needed to attend `MeetingCoordinationCourse` and thanks to such fruition acquires the background knowledge needed to attend `ProcessPlanningCourse`. After the fruition of both courses Richard covers all the skills he missed for performing the task he was selected for.

4 Future trends

The Semantic Web initiative (Berners-Lee et al., 2001) put the tricky challenge of endowing the world wide web with a shared interpretation of data. Many steps have been made in this direction and, even if the complete realization of such an ambitious project is far to be reached, the possibilities opened by the objectives achieved so far are noteworthy.

In particular the vision of knowledge intensive companies is strongly affected by the chance to exploit novel techniques for the management of their most strategic asset: competencies.

The standardization process in knowledge representation carried out on the Semantic Web mainstream puts the basis for collaborative environments representing a new frontier for global economy.

By sharing a common knowledge on competencies and exploiting the semantic-web enabled technologies to exchange or produce such an economically appealing good, companies could give up to their traditional borders and take advantage from the resulting global knowledge market.

5 Conclusions and future work

In this chapter a system and an approach to integrated semantic-based composition of skills and learning needs have been proposed to help knowledge intensive companies in holding, retrieving and creating competencies. When a task has to be assigned to one or more individuals, the system supports both the search for skills among internal personnel and the composition of personalized training processes about the skills unavailable in the company. The approach exploits non-standard reasoning services from Description Logics in order to explain the skills still uncovered after the search process(Concept Abduction) and to revise the request for skills in case all the available employees have profiles conflicting with the required one(Concept Contraction). A minimality criterion on a function taking both belief revision and explanation into account is proposed to carry out the selection among internal personnel. On the basis of the explanation on lacking skills a courseware is automatically generated by retrieving a sequence of learning objects such that their composition satisfies the learning need as far as possible. The improvement of the learning component of the proposed system is object of our current studies. In particular we are evaluating methods for taking into account belief revision in the automatic composition of coursewares and for making the learning process personalized inside the selected team, in case an ad-hoc team is created for performing the needed task. Finally the proposed function is continuously under test with real data from human resource department of knowledge intensive companies.

References

Ajami, K. (2004). Specifying and implementing interoperable and reusable learning objects: one step beyond. In Proc. of Intl. Conf. on Information and Communication Technologies: From Theory to Applications, pages 111–112.

Baader, F., Calvanese, D., Mc Guinness, D., Nardi, D., and Patel-Schneider, P., editors (2002). *The Description Logic Handbook*. Cambridge University Press.

Baader, F., Horrocks, I., and Sattler, U. (2003). Description logics as ontology languages for the semantic web. In Hutter, D. and Stephan, W., editors, *Festschrift in honor of Jörg Siekmann*, Lecture Notes in Artificial Intelligence. SV.

Becerra-Fernandez, I. (2000). The role of artificial intelligence technologies in the implementation of people-finder knowledge management systems. *Knowl.-Based Syst.*, 13(5):315–320.

Bennacer, N., Bourda, Y., and Doan, B. (2004). Formalizing for querying learning objects using OWL. In *Proc. of Intl. Conf. on Advanced Learning Technologies*, pages 321–325. IEEE.

Berners-Lee, T., Hendler, J., and Lassila, O. (2001). The semantic web. *Scientific American*, 248(4):34–43.

Brachman, R. and Levesque, H. (1984). The tractability of subsumption in frame-based description languages. In *Proc. of the Fourth National Conference on Artificial Intelligence (AAAI-84)*, pages 34–37. MK.

Cabezuelo, A. and Beardo, J. (2004). Towards a model of quality for learning objects. In Proc. of Intl. Conf. on Advanced Learning Technologies, pages 822–825. IEEE.

Colucci, S., Di Noia, T., Di Sciascio, E., Donini, F., and Mongiello, M. (2003a). Concept Abduction and Contraction in Description Logics. In *Proc. of the 16th Intl. Workshop on Description Logics (DL'03)*, volume 81 of *CEUR Workshop Proceedings*.

Colucci, S., Di Noia, T., Di Sciascio, E., Donini, F., Mongiello, M., and Mottola, M. (2003c). Finding skills through ranked semantic match of descriptions. *Journal of Universal Computer Science*, pages 168–173. Springer. Issue on proc. of I-Know'03.

Colucci, S., Di Noia, T., Di Sciascio, E., Donini, F., Mongiello, M., and Mottola, M. (2003b). A formal approach to ontology-based semantic match of skills descriptions. *Journal of Universal Computer Science (J.UCS)*, 9(12):1437–1454.

Colucci, S., Di Noia, T., Di Sciascio, E., Donini, F., Mongiello, M., and Piscitelli, G. (2004). Semantic-based approach to task assignment of individual profiles. *Journal of Universal Computer Science*, 10(6):723–730.

Colucci, S., Di Noia, T., Di Sciascio, E., Donini, F., Piscitelli, G., and Coppi, S. (2005a). Knowledge Based Approach to Semantic Composition of Teams in an Organization. In *Proceedings of the 20th Annual ACM (SIGAPP) Symposium on Applied Computing SAC-05*, pages 1314–1319. ACM.

Colucci, S., Di Noia, T., Di Sciascio, E., Donini, F., and Ragone, A. (2005b). Semantic-based automated composition of distributed learning objects for personalized e-learning. In *The Semantic Web: Research and Applications. 2nd European Semantic Web Conf.*, volume 3532 of *LNCS*, pages 633–648. SV.

Colucci, S., Noia, T. D., Sciascio, E. D., Donini, F., and Mongiello, M. (2005c). Concept Abduction and Contraction for Semantic-based Discovery of Matches and Negotiation Spaces in an E-Marketplace. *Electronic Commerce Research and Applications*, 4(4).

Cormen, T. H., Leiserson, C. E., and Rivest, R. L. (1990). *Introduction to Algorithms*. The Massachusetts Institute of Technology.

DAML+OIL (2001). DAML+OIL Specifications. www.daml.org/2001/03/daml+oil-index.html.

Di Noia, T., Di Sciascio, E., and Donini, F. (2004a). Extending Semantic-Based Matchmaking via Concept Abduction and Contraction. In *Proceedings of the 14th International Conference on Knowledge Engineering and Knowledge Management (EKAW 2004)*, LNAI. Springer.

Di Noia, T., Di Sciascio, E., Donini, F., and Mongiello, M. (2003). Abductive matchmaking using description logics. In *Proc. of the 18th Intl. Joint Conf. on Artificial Intelligence (IJCAI 2003)*, pages 337–342. MK.

Di Noia, T., Di Sciascio, E., Donini, F., and Mongiello, M. (2004b). A system for principled Matchmaking in an electronic marketplace. *Intl. Journal of Electronic Commerce*, 8(4):9–37.

DublinCore (1999). Dublin core metadata element set, version 1.1: Reference description. http://dublincore.org/documents/1999/07/02/dces/.

Galil, Z. (1986). Efficient algorithms for finding maximum matching in graphs. *ACM Comput. Surv.*, 18(1):23–38.

Gärdenfors, P. (1988). *Knowledge in Flux: Modeling the Dynamics of Epistemic States.* Bradford Books, MIT Press, Cambridge, MA.

Garro, A. and Palopoli, L. (2003). An xml multi-agent system for e-learning and skill management. In *Agent Technologies, Infrastructures, Tools, and Applications for E-Services*, volume 2592 of *LCNS*, pages 283–294. Springer.

Gasevic, D., Jovanovic, J., and Devedzic, V. (2004). Enhancing learning object content on the semantic web. In *Proc. of Intl. Conf. on Advanced Learning Technologies*, pages 714–716. IEEE.

Gronau, N. and Uslar, M. (2004). Integrating knowledge management and human resources via skill management. In *Proc. I-KNOW '04*, pages 135–142. J.UCS.

Hamel, G. and Prahalad, C. K. (1990). The core competence of the corporation. *Harvard Business Review*, May-June:79–91.

Hefke, M. and Stojanovic, L. (2004). An ontology-based approach for competence bundling and composition of ad-hoc teams in an organization. In *Proc. I-KNOW '04*, pages 126–134. J.UCS.

Hillier, F. S. and Lieberman, G. J. (1995). *Introduction to Operations Research.* MGH.

IEEE (2002). *IEEE Standard for Learning Object Metadata*, std 1484.12.1-2002 edition.

IEEE (2003). *IEEE standard for learning technology-learning technology systems architecture (LTSA)*, std 1484.1-2003 edition.

IMS (2001). Ims, learning resource meta-data best practices and implementation guide version 1.1 - final specification. http://www.imsproject.org/metadata/mdbestv1p1.html.

Ip, A., Young, A., and Morrison, I. (2002). Learning objects - whose are they? In Proc. of 15 th Conf. of the National Advisory Committee on Computing Qualifications, pages 315–320.

Kennington, J. and Wang, Z. (1991). An empirical analysis of the dense assignment problem: Sequential and parallel implementations. *ORSA Journal on Computing*, 3(4):299–306.

Lau, T. and Sure, Y. (2002). Introducing ontology-based skills management at a large insurance company. In *Proc. of the Modellierung*.

Liu, P. and Dew, P. (2004). Using semantic web technologies to improve expertise matching within academia. In *Proc. I-KNOW '04*, pages 370–378. J.UCS.

McGuinness, D., Fikes, R., Hendler, J., and Stein, L. (2002). DAML+OIL: An Ontology Language for the Semantic Web . *IEEE Intelligent Systems*, 17(5):72–80.

O'Leary, D. (1998a). Enterprise knowledge management. *IEEE Computer*, 31(3):54–61.

O'Leary, D. (1998b). Using AI in knowledge management: Knowledge bases and ontologies. *IEEE Intelligent Systems*, 13(3):34–39.

OWL (2004). www.w3.org/TR/owl-features/.

OWL-S (2004). The owl services coalition. www.daml.org/services/owl-s/1.0/owl-s.html.

Saip, H. B. and Lucchesi, C. (1993). Matching algorithms for bipartite graph. Technical Report DCC -03/93. Departamento de Cincia da Computao, Universidade Estudal de Campinas.

Sanchez, S. and Sicilia, M. (2004). On the semantics of aggregation and generalization in learning object contracts. In *Proc. of Intl. Conf. on Advanced Learning Technologies*, pages 425–429. IEEE.

SCORM (2004). Advanced distributed systems (adl) lab, sharable content object reference model (scorm). http://www.adlnet.org/index.cfm?fuseaction=scormabt.

Staab, S., Schnurr, H., Studer, R., and Y. (2001). Knowledge Processes and Ontologies. *IEEE Intelligent Systems*, 16(1).

Sugawara, K. (2003). Agent-based application for supporting job matchmaking for teleworkers. In Patel, D., Patel, S., and Wang, Y., editors, *Proc. of the 2nd IEEE Intl. Conference on Cognitive Informatics (ICCI 2003), 18-20 August 2003, London, UKIEEE ICCI*, pages 137–142. IEEE Computer Society.

Sure, Y., Maedche, A., and Staab, S. (2000). Leveraging corporate skill knowledge - from proper to ontoproper. In *Proc. of the Third Intl. Conf. on Practical Aspects of Knowledge Management.*

Sycara, K., Paolucci, M., Ankolekar, A., and Srinivasan, N. (2003). Automated discovery, interaction and composition of semantic web services. *Journal of Web Semantics*, 1.

Trastour, D., Bartolini, C., and Priest, C. (2002). Semantic Web Support for the Business-to-Business E-Commerce Lifecycle. In *Proc. International World Wide Web Conference (WWW) '02*, pages 89–98. ACM.

Veit, D., Muller, J., Schneider, M., and Fiehn, B. (2001). Matchmaking for Autonomous Agents in Electronic Marketplaces. In *Proc. Intl. Conference on Autonomous Agents '01*, pages 65–66. ACM.

Vossen, G. and Jaeschke, P. (2003). Learning objects as a uniform foundation for e-learning platforms. In Proc. of Intl. Symp. on Database Engineering and Applications Symposium, pages 278–287.

Yimam, D. (2000). Expert finding systems for organizations: Domain analysis and the demoir approach. In *Beyond Knowledge Management: Sharing Expertise.* MIT Press.