Automated Team Selection and Compliance Checking in Business Processes[*]

Cristina Cabanillas
Vienna University of Economics and Business
Vienna, Austria
cristina.cabanillas@wu.ac.at

Manuel Resinas
Universidad de Sevilla
Sevilla, Spain
resinas@us.es

Jan Mendling
Vienna University of Economics and Business
Vienna, Austria
jan.mendling@wu.ac.at

Antonio Ruiz-Cortés
Universidad de Sevilla
Sevilla, Spain
aruiz@us.es

ABSTRACT

Plenty of activities in many business contexts must be performed collaboratively, e.g., in a hospital or when organising a conference. Tasks such as team composition and allocation are usually performed manually and on the ground of limited criteria such as individual skills, a.o. because adequate automatic support is missing. This paper addresses this shortcoming. We present an approach for team selection and compliance checking in business processes, which includes (i) a language for describing teams; (ii) a way to define team selection conditions and policies related to team composition; and (iii) a mechanism for the automatic resolution of the team selection conditions and for team-related compliance checking based on formal ontologies.

Keywords

business process management, RALTeam, resource assignment, team selection, compliance checking

1. INTRODUCTION

Although resource management in Business Processes (BPs) has gained increasing attention in recent years, there is hardly any approach that supports the assignment and allocation of a work item to teams that take part in a Business Process (BP). This comes as a surprise as there are areas such as healthcare where daily activities like surgeries typically require the availability of more than one person. As a consequence, standard workflow concepts cannot be directly applied since they assume a 1:1 relation between work item and worker.

The management of teams relates to a broad spectrum of issues, which are partially discussed in the area of agent and multi-agent systems [37], distributed systems [16] and social computing [21]. These include, e.g., team composition considering availability and preferences, constraints on team selection in relation to a task, conflicts of interest, optimal scheduling, or strategies to improve team performance. The problem is, however, that automatic support for the allocation of *suitable* teams to BP activities in process-oriented organisations is missing.

In this paper, we address this research problem by tackling team selection and automatic checking of compliance rules related to the composition of teams. Our contribution is a language for the description of teams called RALTeam, grounded on a team-aware organisational metamodel. Extending that language it is possible to define team selection conditions for assigning teams to process activities, and team-aware policies that specify constraints over the composition of teams in an organisation. The semantics of RALTeam is then formalised with Description Logics (DLs), which facilitate the automatic resolution of the selection conditions during process execution for team allocation, as well as the automatic checking of such conditions against the team composition policies defined in the company in order to ensure compliance. All constructs of the language are motivated by projects that we have been involved in or which are discussed in the literature.

Against this background, the rest of the paper is structured as follows. Section 2 describes the research problem using a scenario from the healthcare domain. Section 3 presents an organisational metamodel that explicitly captures team-related concepts. Section 4 defines RALTeam as a language for team description. Section 5 explains how the language

[*] This work was partially supported by the Austrian Research Funding Association (FFG), the European Commission (FEDER), and the Spanish and the Andalusian R&D&I programmes (grants 845638 (SHAPE), TIN2009-07366 (SETI), TIN2012-32273 (TAPAS), TIC-5906 (THEOS)).

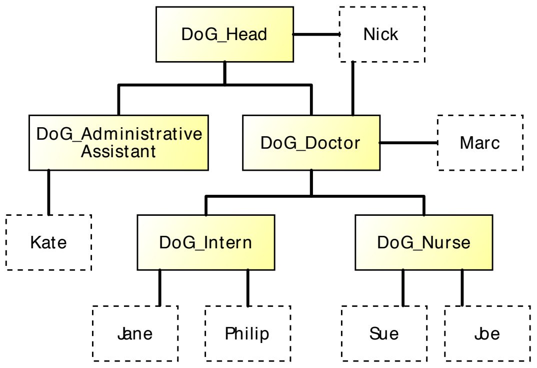
can be extended to define team selection conditions and team-related policies. Section 6 defines the semantics of the language and the mechanism for team selection and compliance checking. Section 7 outlines our proof-of-concept implementation. Section 8 discusses related work before Section 9 concludes the paper.

2. MOTIVATING SCENARIO

One domain in which team work is common is healthcare, from which we adapt a scenario presented in [24]. Figure 1 shows a BP for patient diagnosis modelled with Business Process Model and Notation (BPMN) [29]. Resource assignments are defined in terms of organisational roles, along with conditions that must hold for department members to participate in the activities. First, the patient is registered by a clerk. Then, a doctor and an assistant conduct a physical examination while in parallel a nurse prepares the required documents. Sometimes, further tests must be performed by *the same doctor and assistant* with the help of a nurse. When these activities are completed, the doctor assesses the results of the test(s) and decides which information the nurse has to give to the patient.

Let us assume that the process is executed within the Department of Gynaecology (DoG), whose organisational structure is shown in Figure 2. It is organised based on a hierarchy of positions that are occupied by the members of the department. The head is a doctor called Nick, who can delegate work to all the resources occupying lower positions in the hierarchy, i.e., to all the members of the department. Below, there is an administrative assistant (Kate) and another doctor (Marc), who report to the department head. Subordinates of doctors are interns (Jane and Philip) and nurses (Sue and Joe). The table attached to the hierarchy shows the roles associated to each of the positions in this organisational unit, which typically establish the privileges for the execution of activities and the access to data. Furthermore, as many activities in the department are collaborative, there are some work teams already composed which are usually directly used for assignment. Jane and doctor Nick form a team called Perm_RE_1, where RE stands for Routine Examination. There is a rule in the hospital stating that there must be at least one doctor in each RE team. In that team, Nick plays the role of a coordinator and both of them are implementers. Similarly, Philip is supervised by doctor Marc, such that they form another team called Perm_RE_2. There are also two teams of three members. Nick, Jane and Sue form team Perm_AT_1, and Marc, Philip and Joe for team Perm_AT_2. AT stands for Advanced Tests, and all AT teams must have at least three members, including a doctor and a nurse.

The only accurate way to model resources in BPMN is by means of its XML syntax, by default with XPath. Teams are neither captured in the BPMN metamodel nor in other widely used BP modelling notations. We have used the BPMN Group and Text Annotation artifacts for the sake of clarity. Nonetheless, there are constraints specified in the description of the BP that could not be represented in the model, and cannot be defined with XPath, e.g., the fact that the doctor that takes part in the performance of the advanced tests is the same as the doctor conducting the physical examination, or the fact that the nurse delivering



Position	Role
DoG_Head	Doctor
	Coordinator
DoG_Doctor	Doctor
	Assistant
DoG_Intern	Assistant
DoG_Nurse	Nurse
DoG_AdministrativeAssistant	Clerk

Figure 2: Department of Gynaecology (DoG)

information to the patient is the same who made the documents. Such constraints are fundamental in order to select appropriate teams or individuals. For instance, with the organisational structure and the teams in the DoG, there are two possible teams for activity *Conduct physical examination*, and two possible teams for activity *Conduct advanced tests*. However, selecting a proper team for activity *Conduct advanced tests* depends on the team that conducted the examination in that specific BP instance.

Another domain in which collaborative work is often found is Software (SW) development, where several teams are usually involved in the different SW development phases. For example, the company responsible for the music service system Spotify published its team-based structure [20]. Team composition and selection is also fundamental (and critical) in spaceflight and military missions. The NASA HRP BHP is in charge of managing the risks related to team performance and effectiveness in spaceflight missions [33]. The Team Integrated Design Environment (TIDE) is a tool for the design of teams for military missions [22]. Emergency services also require team work. For instance, temporary teams are ordinary for police and firefighters which, furthermore, sometimes must also cooperate with teams from other organisations, e.g., to battle a blaze distributed over a canyon ridge [8]. We use these domains as reference in the design of the team-aware organisational metamodel presented next.

3. TEAM-AWARE ORGANISATIONAL METAMODEL

Elements related to teams must be part of the organisational metamodel of the company, such that the assignment of teams to activities can be easily managed. To this end, we take the organisational metamodel described by Russell et al. [30] as a starting point, which covers people, capabilities, positions, roles and organisational units. It can thus be used to describe the entities of the DoG.

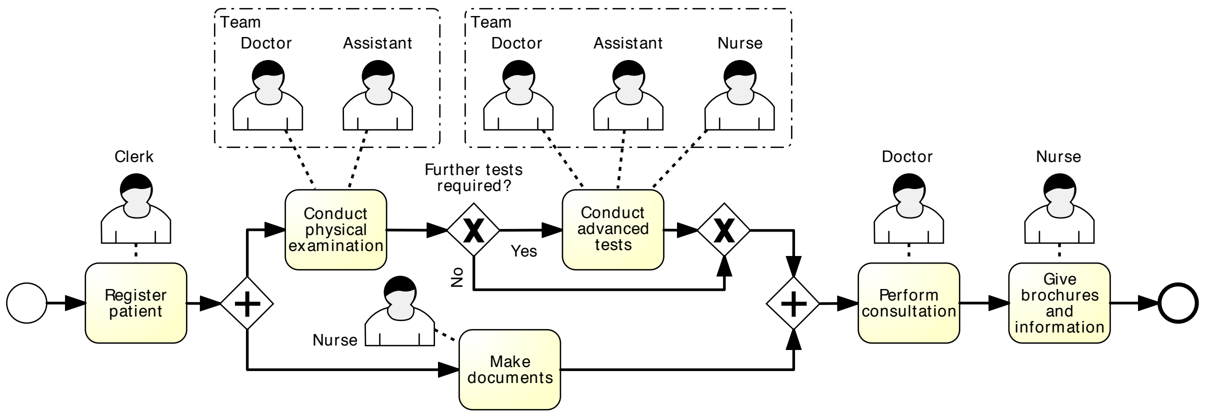


Figure 1: Process for patient diagnosis

The metamodel must be extended based on observations from the aforementioned domains, giving rise to the team-aware organisational metamodel depicted in Figure 3, where former entities are coloured in grey and team-related concepts in white. There is now an *Agent* class that enables assigning BP *Activities* not only to persons but also to teams. A *Team* is a set of people collaborating in the completion of a specific activity with a common objective. A person is a member of a team due to playing a role in the team (class *TeamRole*), e.g., Coordinator of Perm_RE_1. Each team role has a specific *TeamRoleType* according to types explicitly defined in the company, such as Investigator, Coordinator, Implementer or Specialist [4]. Please, note that team roles are fundamentally different from organisational roles. In our example, Nick occupies the position DoG_Doctor and participates in organisational roles Doctor and Assistant, but within team Perm_RE_1 he has the team role of Coordinator of Perm_RE_1 (of type Coordinator) and the team role of Implementer of Perm_RE_1 (of type Implementer). Please, note that one person can be a member of several teams.

A team can have a type (class *TeamType*) that is associated with a specific configuration of the organisational roles. For example, in the motivating scenario there are teams Perm_RE_1 and Perm_RE_2 of type Routine Examination, composed of a doctor and an assistant. Team type Advanced Tests is made up of a doctor, an assistant and a nurse. More teams of these types could be created with the same role configuration. There could also be a team type Heart Surgery made up of two doctors, two assistants and one nurse, for instance. In this way, team types provide templates for the composition of teams.

Teams can also be structured hierarchically. For example, in SW development, there are often teams of SW Analysts (composed of persons with role Analyst), teams of SW Developers and teams of SW Testers. The teams of analysts delegate work to the teams of developers, which report issues and results to the former and, in turn, delegate work to the teams of testers. These report the results to the developers. In this context, modes of communication between teams have to be established, which we do not directly address here.

Finally, teams are also classified according to their temporality. A *PermanentTeam* is defined without an expiry date, e.g., all the teams defined in Section 2. Permanent teams can be referenced by their identifier at any moment. However, in certain occasions new teams are composed for specific purposes. For instance, in emergency surgeries teams are created, modified and broken up constantly depending on the requirements of the surgery. Such teams are called *TemporaryTeams* because they have an expiry date defined as a specific scope, which can be (i) a specific period of time, e.g, a team active from August 1^{st} to August 31^{st} to provide support during the summer holiday break; (ii) it can be associated to a single activity instance, e.g., the execution of a single surgery; or (iii) it can be related to a process instance, so that the team can be treated as a single entity during the execution of the process instance because its participation could be required at any moment. Further team classifications are proposed in literature that mostly focus on how teams organise themselves (e.g., their coordination mechanism [26]). However, they are not included in the metamodel because our focus is on those aspects that are relevant for team selection in the context of resource assignment.

A person is a team creator if they are in charge of its setting it up and of recruiting its members. They are not necessarily a member of the team, though. The figure of team creator is not mandatory, as teams may be automatically composed by a system according to some pre-defined properties.

It is important to remark the difference between an *OrganisationalUnit* and a *PermanentTeam*. Although both are groups of people with an indefinite duration, the former is not an entity of collaborative work with a single goal by nature, but is composed of members that participate in different activities, each of which has a specific objective. In case of assigning concrete work to an organisational unit, it is because the unit is working as a team in the context of an activity or process, i.e., there is a new team made up of the members of the organisational unit. For instance, if a hospital is organising an event, each department (i.e., organisational unit) could form a team working on the preparation of a specific issue (i.e., in that moment all their members have a common goal). Such a distinction has been described before [9].

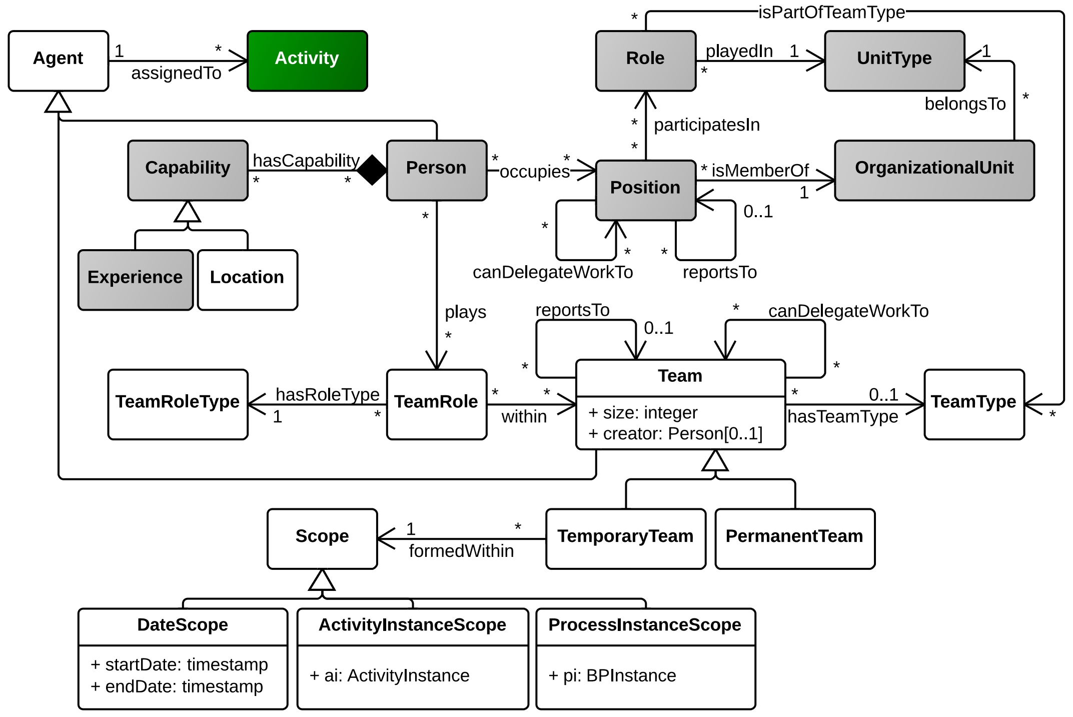


Figure 3: Team-Aware Organisational Metamodel

4. RALTeam FOR TEAM DESCRIPTION

Resource Assignment Language (RAL) is a Domain Specific Language (DSL) for the definition of resource selection conditions, currently focused on human resources [6]. Its current version [5] uses the grey excerpt of the metamodel in Figure 3 and allows expressing a great variety of conditions with a syntax similar to natural language, such as:

```
RAL1: IS Samuel
RAL2: NOT (IS ANY PERSON responsible for ACTIVITY
      RegisterP)
RAL3: (HAS ROLE Assistant) OR (HAS POSITION
      DoG_Doctor)
RAL4: SHARES SOME ROLE WITH PERSON IN DATA FIELD
      Test.Doctor
RAL5: (HAS UNIT DoG) AND (IS ANY PERSON accountable
      for ACTIVITY MakeDocs)
```

We have defined an extension for RAL called RALTeam to describe teams according to the team-aware organisational metamodel described in Section 3. Similarly to RAL, RALTeam consists of *expressions* and *constraints*, whose Extended Backus-Naur Form (EBNF) syntax is shown in Language 1. In particular, it includes eight types of expressions (*RALTeamExpr*), whose configuration is supported by three types of constraints (*TCardinalityConstraint, ScopeConstraint, MembershipConstraint*). The language has been designed using a similar rationale as for the design of RAL [5] and following the methodology described in [7].

TeamIDExpr (line 7) allows indicating directly a team ID.

TeamSizeExpr (line 9) allows specifying the number of team members with a *TCardinalityConstraint* (line 30), e.g., `WITH AT MOST 3 MEMBERS`.

TeamRoleExpr (line 11) allows specifying a set of team role types for a team, i.e., team role types played by some of its members.

TeamTypeExpr (line 13) allows specifying the type of a team among three options: (i) a specific type, (ii) the same type as another team defined by a *RALTeamExpr* (line 14), or (iii) a type different than the type of another team defined by a *RALTeamExpr* (line 15).

TeamCreatorExpr (line 17) specifies the creator of a team using options similar to the *TeamTypeExpr*, plus one option described below.

TeamScopeExpr (line 22) allows specifying the scope of a team using options similar to the *TeamTypeExpr*, the only difference being that the specific scope can be defined according to the three types of scopes described in the metamodel (cf. Fig. 3) with a *ScopeConstraint*.

TeamCompoundExpr (line 24) allows combining the aforementioned expressions with the AND and OR operators.

TeamMemberExpr (line 4) uses a *MembershipConstraint* to provide information about the team members, and optionally the team role type(s) that they play (line 28). Specifically, it allows specifying (i) a concrete person (line 37), resulting in sentences such as `WHOSE MEMBERS INCLUDE Marc` or `WHOSE MEMBERS INCLUDE Marc PLAYING TEAM ROLE TYPE Coordinator`; (ii) an amount of people, e.g., `WHOSE MEMBERS INCLUDE EXACTLY 2 PEOPLE PLAYING TEAM ROLE TYPE Implementer`; or (iii) a certain number of people with specific characteristics defined with *PeopleSelection* (line 38), which include:

Language 1 RALTeam for team description

```
 1  RALTeamExpr := TeamIDExpr          | CREATED BY TeamCreatorExpr
 2       | WITH TeamSizeExpr           | WITH SCOPE TeamScopeExpr
 3       | CONTAINING TeamRoleExpr     | TeamCompoundExpr
 4       | OF TYPE TeamTypeExpr        | WHOSE MEMBERS INCLUDE TeamMemberExpr
 5
 6
 7  TeamIDExpr := teamID
 8
 9  TeamSizeExpr := TCardinalityConstraint (MEMBER | MEMBERS)
10
11  TeamRoleExpr := TEAM ROLE (TYPE | TYPES) teamRoleTypeList
12
13  TeamTypeExpr :=  teamTypeID
14                   | LIKE TEAM '(' RALTeamExpr ')'
15                   | UNLIKE TEAM '(' RALTeamExpr ')'
16
17  TeamCreatorExpr := personID
18                   | THE SAME PERSON AS TEAM '(' RALTeamExpr ')'
19                   | A DIFFERENT PERSON THAN TEAM '(' RALTeamExpr ')'
20                   | SOMEONE WHO PeopleSelection
21
22  TeamScopeExpr := ScopeConstraint | ...
23
24  TeamCompoundExpr := '(' RALTeamExpr ')' AND '(' RALTeamExpr ')'
25          | '(' RALTeamExpr ')' OR '(' RALTeamExpr ')'
26
27  TeamMemberExpr := '(' MembershipConstraint ')'
28                                      [PLAYING TEAM ROLE TYPE teamRoleTypeID]
29
30  TCardinalityConstraint := EXACTLY num | AT LEAST num | AT MOST num
31                   | BETWEEN num AND num
32
33  ScopeConstraint := ACTIVE BETWEEN timestamp AND timestamp
34          | ACTIVE DURING THE EXECUTION OF ACTIVITY activityInstanceID
35          | ACTIVE DURING THE EXECUTION OF PROCESS processInstanceID
36
37  MembershipConstraint := personID
38    | (ONLY | TCardinalityConstraint) (PERSON | PEOPLE) [WHO PeopleSelection]
39
40  PeopleSelection := PersonExpr | GroupResourceExpr | CommonalityExpr
41          | CapabilityExpr | HierarchyExpr | NegativeExpr | CompoundExpr
42          | (IS | ARE) [NOT] (MEMBER | MEMBERS) OF TEAM '(' RALTeamExpr ')'
```

Language 2 RALTeam for team selection and rule definition (EBNF)

```
1  RALTeamSelection := TEAM RALTeamExpr
2
3  RALTeamPolicy := TEAMS RALTeamExpr MUST [NOT]
4                   BE TEAMS RALTeamExpr
```

- properties specified with a RAL expression (line 40), e.g., `WHOSE MEMBERS INCLUDE ONLY PEOPLE WHO HAVE UNIT DoG` specifies that all the member of the team belong to DoG, where `HAVE UNIT DoG` comes from RAL and unit refers to an organisational unit. The link with RAL involves all the RAL expressions but one (*IsAssignmentExpr*).
- people that do (or do not) belong to other teams defined by a *RALTeamExpr* (line 42).

It also introduces an option in *CreatorConstraint* (line 20) to provide more details about the team creator by means of a RAL expression.

5. RALTeam FOR TEAM SELECTION AND RULE DEFINITION

The concepts of RALTeam as defined in Section 4 offer a mechanism for team description. Let us now consider the required language concepts for team selection and rule definition. To this end, Language 2 introduces two additional expressions:

RALTeamSelection (line 1) allows defining conditions for the selection of teams. Hence, it allows assigning teams to BP activities by specifying the conditions that they must fulfil, for instance:

`TEAM CONTAINING TEAM ROLE TYPE Coordinator` would return at least teams Perm_RE_1 and Perm_RE_2 in our scenario.

`TEAM OF TYPE LIKE TEAM ((Perm_RE_1) OR (Temp_AT_2))` selects teams Perm_RE_2 and team Temp_AT_1 according to our scenario.

`TEAM WITH SCOPE ACTIVE DURING THE EXECUTION OF PROCESS bp1` selects all the permanent teams and the temporary teams whose scope fits with the one specified in the expression, i.e., teams active while *bp*1 is running.

`TEAM (WITH AT LEAST 4 MEMBERS) OR (OF TYPE Advanced Tests)` selects teams Perm_AT_1 and Temp_AT_2.

RALTeamPolicy (line 4) allows defining policies related to teams that must hold in the organisation, such as:

`TEAMS WITH BETWEEN 5 AND 10 MEMBERS MUST BE TEAMS CONTAINING TEAM ROLE TYPE Coordinator.`

`TEAMS CREATED BY SOMEONE WHO HAS ROLE Assistant MUST NOT BE TEAMS OF TYPE Routine Examination.`

Applying formal semantics to all the expressions described above, the resolution of the conditions for resource selection, which return a set of teams that are potential performers of a BP activity; as well as the checking of compliance between the existing teams and the policies defined by the company, can be automated.

6. AUTOMATIC TEAM SELECTION AND COMPLIANCE CHECKING

Following the same approach as in RAL [5], RALTeam semantics is defined by means of a mapping to DLs [2]. Knowledge representation systems based on DLs involve two components: TBox and ABox. The TBox describes terminology, i.e., the ontology in the form of *concepts* and *properties* (relations between the concepts) and their relations, while the ABox contains *assertions* about individuals (instances of concepts) using the terms from the ontology [2]. The mapping is a function $\cdot^{\mathcal{T}}$ that maps the team-aware organisational metamodel, its instantiation for a specific organisation and the RALTeam expressions to DL axioms and concept descriptions.

The mapping of the team-aware organisational metamodel is straightforward: metamodel classes and associations are mapped as concepts and properties in the Knowledge Base (KB), respectively, and cardinality restrictions are mapped as axioms such as $Team \sqsubseteq \geq 1\, hasTeamType.(TeamType)$. There is only one consideration to this mapping. In the metamodel, the relation between Person, Team and TeamRoleType is modelled with class TeamRole. However, DLs allow a more convenient way of expressing such a relation by using hierarchies of properties. In this case the mapping involves adding a property *hasMember* from Team to Person and defining each TeamRoleType as a new subproperty of *hasMember*. In addition, a new property *roleType* is added from Team to TeamRoleType. This avoids introducing an "artificial" concept to define the ternary relation of the metamodel, hence minimising the number of constructs as suggested in the *Conceptualisation Principle* described by ter Hofstede and Proper [38].

The instantiation of the metamodel is mapped as follows. Class instances and their relations are mapped as individuals and relations between them except for *TeamRole* instances, which are mapped by means of *hasMember* subproperties as described above in order to build DL expressions with them in an easier way. In addition, we assume that we have complete knowledge about the organisational model. Therefore, a mechanism to deal with the open world assumption of DLs should be provided. The *open world assumption* means that DLs assume that the knowledge may be incomplete, and hence, the absence of a property assertion stating that $hasMember(Perm\_RE\_1^{\mathcal{T}}, Jane^{\mathcal{T}})$ does not mean that Jane does not belong to team Perm_RE_1. The solution proposed is an usual way to deal with the open world assumption, which implies that the mapping must include assertions that explicitly state that each individual has exactly the properties specified and no more (e.g., Team Perm_RE_1 has exactly two *hasMember* relations: $Perm\_RE\_1^{\mathcal{T}} \in = 2\ hasMember.Person$).

Finally, RALTeam expressions are mapped into DL concept descriptions, which are all subconcepts of Team and are defined in a way such that for every team t that satisfies a RALTeam expression *expr*, it holds that $t^{\mathcal{T}} \in expr^{\mathcal{T}}$. Table 1 details the mapping for most RALTeam expressions. Expressions that involve *TeamMemberExpr* require an additional mapping (cf. Table 2) to obtain DL concepts from people selection expressions (cf. PeopleSection in Language 1).

All the DL concept descriptions used in this mapping belongs to the direct model-theoretic semantics of OWL 2, which extends the semantics of the DLs $\mathcal{SROIQ}$ with data types and punning [27]. In particular, note that the kind of reasoning used for date scopes in temporary teams does not require the use of temporal DLs. Instead, dates are used as if they were integer numbers, i.e., simple datatypes. This means that any DL reasoner that can handle OWL 2 semantics can be used to reason about teams.

In fact, let $\mathcal{K}$ be a KB obtained after mapping the elements of the team-aware organisational metamodel, its instantiation for a specific organisation and the RALTeam expressions using mapping $\cdot^{\mathcal{T}}$. Both team selection and team compliance checking can be formulated in terms of standard DL reasoning tasks on $\mathcal{K}$ that are implemented by most DL reasoners. In particular, two DL reasoning tasks are used, namely, concept retrieval, which is the problem of computing the set containing exactly every instance of a concept with respect to a KB $\mathcal{K}$; and consistency, which is the problem of deciding whether a KB $\mathcal{K}$ is consistent. We denote the former reasoning task as $individuals_{\mathcal{K}}$ and the latter as $consistent_{\mathcal{K}}$.

Team selection. This operation involves obtaining all teams defined in the organisation that satisfy a given RALTeam expression *expr*. Therefore, it can be expressed using the $individuals_{\mathcal{K}}$ reasoning task on the DL mapping of the RALTeam expression, $individuals_{\mathcal{K}}(expr^{\mathcal{T}})$.

Team compliance checking. This operation involves checking whether the teams in the company are compliant with a set of policies for team composition specified with RALTeam. In this case, the policies should first be mapped to DLs, and then the $consistency_{\mathcal{K}}$ reasoning task can be used to check compliance. The mapping of the policies is done as follows. Given a policy of the form `TEAMS expr1 MUST BE TEAMS expr2`, an axiom $expr_1^{\mathcal{T}} \sqsubseteq expr_2^{\mathcal{T}}$ is added to the DL KB. This axiom states that all teams that satisify $expr_1$ must also satisfy $expr_2$. After all policies are mapped, the $consistency_{\mathcal{K}}$ reasoning operation checks whether these axioms hold for all teams in the DL KB. Furthermore, one could use the explanation facilities integrated in many DL reasoners to find out the reason why a team is not compliant with the policies.

7. PROOF OF CONCEPT

We have evaluated the viability of the concepts covered by RALTeam with a prototypical application implemented with

Table 1: Excerpt of the mapping of the RALTeam expressions to DL concepts

Type	**RALTeam Expr** ($expr$)	**DL Concept Description** ($expr^{\mathcal{T}}$)
Team	teamID	$\{teamID\}$
Size	AT LEAST n MEMBERS	$Team \sqcap \geq n\, hasMember$
	EXACTLY n MEMBERS	$Team \sqcap = n\, hasMember$
	BETWEEN n AND m	$Team \sqcap \geq n\, hasMember \sqcap \leq m\, hasMember$
Role	TEAM ROLES typeList	$\exists roleType(\{typeList\})$
Type	teamTypeID	$\exists hasType.\{teamTypeID\}$
	LIKE (expr)	$\exists hasType.(\exists hasType^{-}.expr^{\mathcal{T}})$
	UNLIKE (expr)	$Team \sqcap \neg\exists hasType.(\exists hasType^{-}.expr^{\mathcal{T}})$
Creator	personId	$\exists hasCreator.\{personId\}$
Scope	ACTIVE BETWEEN start AND end	$PermanentTeam \sqcup \exists formedWithin.(TemporalScope \sqcap \exists(startDate \leq start) \sqcap \exists(endDate \geq end))$
	ACTIVE DURING THE EXECUTION OF PROCESS pId	$PermanentTeam \sqcup \exists formedWithin.(ProcessInstanceScope \sqcap \exists pi.\{pId\})$
	LIKE (expr)	$\exists formedWithin.(\exists formedWithin^{-}.(expr^{\mathcal{T}}))$
Comp.	expr1 AND expr2	$expr_1^{\mathcal{T}} \sqcap expr_2^{\mathcal{T}}$
	expr1 OR expr2	$expr_1^{\mathcal{T}} \sqcup expr_2^{\mathcal{T}}$
Member	personId	$\exists hasMember.(\{personId\})$
	ONLY PEOPLE WHO ps	$Team \sqcap \forall hasMember.(ps^{\mathcal{P}})$
	AT LEAST 1 PERSON WHO ps PLAYING TEAM ROLE TYPE teamRoleTypeId	$Team \sqcap \geq 1\, teamRoleTypeId.(ps^{\mathcal{P}})$

Table 2: Mapping of RALTeam PeopleSelection to DL concept descriptions. Function $\cdot^{\mathcal{R}}$ stands for the RAL mapping detailed in [5]

People selection (ps)	**DL Concept Description** ($ps^{\mathcal{P}}$)
SelectionExpr	$SelectionExpr^{\mathcal{R}}$
IS MEMBER OF expr	$\exists hasMember^{-}.(expr^{\mathcal{T}})$
IS NOT MEMBER OF expr	$Person \sqcap \neg\exists hasMember^{-}.(expr^{\mathcal{T}})$

Java and the OWL API, and tested with the HermiT OWL reasoner. The mappings for all RALTeam expressions and the Java application can be found at `http://www.isa.us.es/cristal`. Using the implemented concepts, we are able to express the team assignments of our motivating scenario (cf. Section 2) as detailed next.

Activity *"Conduct physical examination"* must be performed by an RE team can be defined using RALTeam as follows:

```
TEAM OF TYPE RoutineExamination
```

The selection of teams that fulfil this RALTeam expression can be done by means of the following DL reasoning task:

$$individuals_{\mathcal{K}}(\exists hasType.\{RoutineExamination\})$$

Activity *"Conduct advanced tests"* must be done by an AT team whose doctor took part in activity *Conduct physical examination* in that BP instance can be defined using RALTeam as follows:

```
(TEAM OF TYPE AdvancedTests) AND
(TEAM (WHOSE MEMBERS INCLUDE (AT LEAST 1 PERSON WHO
  ((IS ANY PERSON involved in ACTIVITY
       ConductPhysicalExamination)
    AND (HAS ROLE Doctor)))))
```

whose team selection can be done by means of the following DL reasoning task:

$$\begin{aligned}&individuals_{\mathcal{K}}(\exists hasType.\{AdvancedTests\} \sqcap \\ &((Team \sqcap \exists hasMember.(IS\ ANY\ PERSON...^{\mathcal{R}})) \sqcap \\ &(Team \sqcap \exists hasMember.(HAS\ ROLE\ Doctor^{\mathcal{R}}))))\end{aligned}$$

These assignments can then be used by the hospital to support team selection and scheduling at run time. The rules mentioned in Section 2 are defined as follows.

There must be at least one doctor in each routine examination team:

```
TEAMS OF TYPE RoutineExamination MUST BE TEAMS
WHOSE MEMBERS INCLUDE AT LEAST ONE PERSON
      WHO HAS ROLE Doctor
```

In DLs, this involves adding the following axiom to the KB $\mathcal{K}$:

$$\begin{aligned}&\exists hasType.\{RoutineExamination\} \sqsubseteq \\ &(Team \sqcap \exists hasMember.(HAS\ ROLE\ Doctor^{\mathcal{R}}))\end{aligned}$$

All advanced tests teams must have at least three members[1].

```
TEAMS OF TYPE AdvancedTests MUST BE
```

[1]This rule has been shortened for the sake of brevity (cf. Section 2).

```
TEAMS WITH AT LEAST 3 MEMBERS
```

In DLs, this involves adding the following axiom to the KB $\mathcal{K}$:

$$\exists hasType.\{AdvancedTests\} \sqsubseteq (Team \sqcap \geq 3 hasMember)$$

Finally, compliance between the teams defined in the company and these rules can then be checked with the DL query $consistency_{\mathcal{K}}$.

8. RELATED WORK

The necessity to deal with individual and collaborative tasks in the same environment has been identified and some partial solutions have been proposed in Computer Supported Collaborative Work (CSCW) [28, 18]. In the OSSAD method, collaborative tasks are supported by the concept of "horizontal macro-operation" [28]. None of the approaches found in this field pursues our goal. We tackle the challenge identified in a survey on team work over the past fifty years [31] related to the assignment of teams to activities, i.e., team selection. Their notion of adaptive teams is closely related to our concept of temporary team (cf. Section 3). Next, we discuss approaches related to our work from several domains.

Team Modelling: STEAM [36] defines an organisational metamodel to support hierarchies of teams, composed of individuals. Both teams and people can be associated to roles according to their capabilities. Roles can be persistent or task-specific. Tambe et al. [37, 19] investigated how that metamodel worked in building agent-teams in the simulation league for Robocup, and how agents learn specific skills. Van der Aalst and Kumar focused on modelling organisational structures and work distribution in the context of team work [39]. Their *team_type* is our *TeamRole*, their *team_position* is our *Role*, and their *role* is our *Position*. Temporality in teams is not considered in their approach. Dustdar developed Caramba [14], a process-aware information system to integrate artifacts, resources and processes [15] that emphasises communication and interaction but disregards teams.

Team Composition and Selection: Most approaches dealing with team composition and selection address the problem of finding the best match of experts to required skills [14, 17, 3, 1]. In this context, several approaches study connectivity and social aspects for team composition, e.g., social distance between people [40]. Dorn et al. [12] highlight physical location and communication capabilities between team members as relevant. They present an approach for deriving user profiles from social networks and create virtual teams in which there is balance between skills and connectivity. This is extended towards a skill-dependent recommendation model for team composition [13]. Some other approaches considering both skills and connectivity are [21, 34, 10]. RALTeam takes into account skills and geo-positions of people. Social connectivity is not considered due to its intra-organisational focus, but it could be extended to deal with social aspects as well. Some of the Advanced Resource Patterns (ARPs) described by Meyer [25] are related to team selection, namely Single Entity and Restricted Team Size. Both are supported by RALTeam, as it treats teams as a single entity for resource assignment and allows defining the team size with the *TeamSizeExpr*.

Team Allocation: Partially orthogonal to our work is team allocation. Mans et al. introduced an approach [24] that allocates people to BP activities considering their calendars, the calendars of the people they have to collaborate with in the BP activities, and the ongoing execution of the BP, so that everything is completed on-time. This approach is also used in Proclets [23], a framework that provides support for the modelling and execution of "non-monolithic" processes, i.e., unstructured processes with complex interactions between participants, where activity execution is sometimes collaboratively performed by several people. Such features are not supported by most of the current Process-Aware Information Systems (PAIS). Our approach could be combined with schedule-based allocation approaches.

Team Cooperation and Performance: Several literature reviews and surveys have been conducted on this topic [9, 35, 32]. Moe et al. argued that traditional teams follow a plan-driven model, whereas self-managing agile teams face change-driven development. They studied work cooperation and performance in self-managed agile teams [26], applying the Dickinson and McIntyre's team work model [11] to a real case where teams used Scrum. Caramba [14] supports the collaboration of virtual teams in adaptive workflow management systems, i.e., processes that are not perfectly defined from the beginning but are reconfigured on-the-fly. The inContext Pervasive Collaboration Services Architecture (PCSA) [16] aims at supporting highly dynamic forms of human collaboration such as Nimble (short-lived collaboration), Virtual (spanning different geographical places) and Mobile (collaboration with mobility capabilities) teams.

9. CONCLUSIONS AND FUTURE WORK

The integration of team work in business processes is still limited. In this paper, we have addressed this research problem by introducing a language to describe teams and its applicability for the definition of team selection conditions and policies related to the composition of teams. The DL-based semantics of the language has been used to automate the resolution of team selection conditions and for compliance checking with team-related policies. The feasibility of the approach has been tested with an implementation.

We deem our approach not only relevant from a research angle, but it paves the way for automatically resolving higher level queries with strong practical applications such as "do we have the necessary human resources to conduct a surgery on trauma?". We aim to conduct case studies in different domains to identify those higher level queries and to further validate RALTeam expressiveness.

Furthermore, extending RALTeam to support on-the-fly team composition at run time, the composition and selection of virtual or distributed teams, and the integration of these results with other approaches such as schedule-aware workflow management systems [24] are part of our planned future work as well.

10. REFERENCES

[1] A. Anagnostopoulos, L. Becchetti, C. Castillo, A. Gionis, and S. Leonardi. Power in Unity: Forming Teams in Large-scale Community Systems. In *CIKM*, pages 599–608, 2010.

[2] F. Baader, D. Calvanese, D. McGuinness, D. Nardi, and P. Patel-Schneider. *The Description Logics Handbook: Theory, Implementations, and Applications*. Cambridge University Press, 2003.

[3] A. Baykasoglu, T. Dereli, and S. Das. Project Team Selection Using Fuzzy Optimization Approach. *Cybern. Syst.*, 38(2):155–185, 2007.

[4] M. R. Belbin. *Team Roles at Work*. A Butterworth-Heinemann Title, 1996.

[5] C. Cabanillas, M. Resinas, and A. R. Cortés. Specification and Automated Design-Time Analysis of the Business Process Human Resource Perspective. *Inf. Syst.*, page In press., 2015.

[6] C. Cabanillas, M. Resinas, and A. Ruiz-Cortés. RAL: A High-Level User-Oriented Resource Assignment Language for Business Processes. In *BPM Workshops (BPD'11)*, pages 50–61, 2011.

[7] C. Cabanillas, M. Resinas, A. Ruiz-Cortés, and J. Mendling. Methodology to Extend RAL. In *Jornadas de Ciencia e Ingeniería del Software*, 2014.

[8] J. M. Carroll, M. B. Rosson, G. Convertino, and C. H. Ganoe. Awareness and teamwork in computer-supported collaborations. *Interacting with Computers*, 18(1):21–46, 2006.

[9] S. G. Cohen and D. E. Bailey. What makes teams work: Group effectiveness research from the shop floor to the executive suite. *Journal of Management*, 23(3):239–290, 1997.

[10] A. Datta, J. T. T. Yong, and A. Ventresque. T-RecS: team recommendation system through expertise and cohesiveness. In *WWW*, pages 201–204, 2011.

[11] T. L. Dickinson and R. M. McIntyre. A conceptual framework for teamwork measurement. In *Team performance assessment and measurement: Theory, methods, and applications*, pages 19–43. Lawrence Erlbaum Associates Publishers, 1997.

[12] C. Dorn and S. Dustdar. Composing near-optimal expert teams: a trade-off between skills and connectivity. In *OTM 2010*, volume 6426, pages 472–489. Springer-Verlag, Oct. 2010.

[13] C. Dorn, F. Skopik, D. Schall, and S. Dustdar. Interaction mining and skill-dependent recommendations for multi-objective team composition. *Data & Knowledge Engineering*, 70(10):866–891, 2011.

[14] S. Dustdar. Caramba: Process-Aware Collaboration System Supporting Ad hoc and Collaborative Processes in Virtual Teams. *Distributed and Parallel Databases*, 15(1):45–66, 2004.

[15] S. Dustdar. Process-Aware Information Systems for Virtual Teamwork, 2005.

[16] S. Dustdar and H.-L. Truong. A Pervasive, Service-Oriented Architecture for Supporting Teamwork. *ERCIM News*, pages 47–48, 2007.

[17] E. L. Fitzpatrick and R. G. Askin. Forming Effective Worker Teams with Multi-functional Skill Requirements. *Comput. Ind. Eng.*, 48:593–608, 2005.

[18] N. Guimarães, P. Antunes, and A. Pereira. The Integration of Workflow Systems and Collaboration Tools. In *Workflow Management Systems and Interoperability*, volume 164 of *NATO ASI Series*, pages 222–245. Springer Berlin Heidelberg, 1998.

[19] H. Kitano, M. Asada, Y. Kuniyoshi, I. Noda, and E. Osawa. RoboCup: The Robot World Cup Initiative. In *International Conference on Autonomous Agents*, AGENTS '97, pages 340–347, 1997.

[20] H. Kniberg and A. Ivarsson. Scaling Agile at Spotify. Crisp's Blog, 2012. `http://blog.crisp.se/2012/11/14/henrikkniberg/scaling-agile-at-spotify`.

[21] T. Lappas, K. Liu, and E. Terzi. Finding a Team of Experts in Social Networks. In *KDD*, pages 467–476, 2009.

[22] J. MacMillan, M. Paley, Y. Levchuk, E. Entin, D. Serfaty, and J. Freeman. Designing the Best Team for the Task: Optimal Organizational Structures for Military Missions. In *New Trends in Cooperative Activities: System Dynamics in Complex Settings*, 2002.

[23] R. S. Mans, N. C. Russell, W. M. P. van der Aalst, P. J. M. Bakker, A. J. Moleman, and M. W. M. Jaspers. Proclets in Healthcare. *J. of Biomedical Informatics*, 43(4):632–649, 2010.

[24] R. S. Mans, N. C. Russell, W. M. P. van der Aalst, A. J. Moleman, and P. J. M. Bakker. *Transactions on Petri Nets and Other Models of Concurrency IV*. Springer-Verlag Berlin Heidelberg, 2010.

[25] A. Meyer. Resource Perspective in BPMN - Extending BPMN to Support Resource Management and Planning. Master's thesis, Hasso Plattner Institute, 2009.

[26] N. B. Moe, T. Dingsø yr, T.yr, and T. Dybå. A teamwork model for understanding an agile team: A case study of a Scrum project. *Information and Software Technology*, 52(5):480–491, 2010.

[27] B. Motik, P. F. Patel-Schneider, and B. C. Grau. OWL 2 Web Ontology Language Direct Semantics. `http://www.w3.org/TR/2009/REC-owl2-direct-semantics-20091027/`, 2009.

[28] S. Nurcan. Analysis and design of co-operative work processes: a framework. *Information and Software Technology*, 40(3):143–156, 1998.

[29] OMG. BPMN 2.0. Recommendation, OMG, 2011.

[30] N. Russell, A. ter Hofstede, D. Edmond, and W. M. P. van der Aalst. Workflow Resource Patterns. Technical report, BETA, WP 127, Eindhoven Univ. of Tech., 2004.

[31] E. Salas, K. C. Stagl, C. S. Burke, and G. F. Goodwin. Fostering team effectiveness in organizations: toward an integrative theoretical framework. *Nebraska Symposium on Motivation*, 52:185–243, 2007.

[32] C. Savelsbergh, B. v. d. Heijden, and R. Poell. Attitudes towards factors influencing team performance: A multi-rater approach aimed at establishing the relative importance of team learning behaviors in comparison with other predictors of team performance. *Team Performance Management*, 16:451–474, 2010.

[33] L. L. Schmidt, K. Keeton, K. J. Slack, L. B. Leveton, and C. Shea. Risk of Performance Errors due to Poor Team Cohesion and Performance, Inadequate Selection/Team Composition, Inadequate Training, and Poor Psychosocial Adaptation. In *Human health*

and performance risks of space exploration missions: evidence reviewed by the NASA Human Research Program. National Aeronautics and Space Administration, Lyndon B. Johnson Space Center.

[34] P. V. Singh. The Small-world Effect: The Influence of Macro-level Properties of Developer Collaboration Networks on Open-source Project Success. *ACM Trans. Softw. Eng. Methodol.*, 20(2):6:1–6:27, 2010.

[35] K. Sycara and G. Sukthankar. Literature Review of Teamwork Models. Technical Report CMU-RI-TR-06-50, Robotics Institute, November 2006.

[36] M. Tambe. Teamwork in real-world, dynamic environments. In *International Conference on Multi-Agent Systems (ICMAS-96)*. AAAI Press, 1996.

[37] M. Tambe, J. Adibi, Y. Al-Onaizan, A. Erdem, G. A. Kaminka, S. C. Marsella, and I. Muslea. Building agent teams using an explicit teamwork model and learning. *Artificial Intelligence*, 110(2):215–239, 1999.

[38] A. ter Hofstede and H. Proper. How to Formalize It? Formalization Principles for Information System Development Methods. *Information and Software Technology*, 40:519–540, 1998.

[39] W. M. P. van der Aalst and A. Kumar. A Reference Model for Team-enabled Workflow Management Systems. *Data Knowl. Eng.*, 38(3):335–363, 2001.

[40] D.-N. Yang, Y.-L. Chen, W.-C. Lee, and M.-S. Chen. On Social-temporal Group Query with Acquaintance Constraint. *Proc. VLDB Endow.*, 4(6):397–408, 2011.