Methodology to Extend RAL*

Cristina Cabanillas[1], Manuel Resinas[2],
Antonio Ruiz-Cortés[2], and Jan Mendling[1]

[1]Vienna University of Economics and Business, Austria
{cristina.cabanillas,jan.mendling}@wu.ac.at
[2]University of Seville, Spain {resinas,aruiz}@us.es

Abstract. Resource Assignment Language (RAL) is a language for the selection of organisational resources that can be used, for example, for the assignment of human resources to business process activities. Its formal semantics have allowed the automation of analysis operations in several phases of the business process lifecycle. RAL was designed considering a specific organisational metamodel and pursuing specific purposes. However, it can be extended to deal with similar problems in different domains and under different circumstances. In this paper, a methodology to extend RAL is introduced, and an extension to support another organisational metamodel is described as a proof-of-concept.

Keywords: business process management, description logics, RAL, resource assignment, W3C Organisation Ontology

1 Introduction

Resource Assignment Language (RAL) is a language for the assignment of performers to Business Process (BP) activities. Its formal semantics based on Description Logics (DLs) enable the automation of analysis operations in several phases of the BP lifecycle, so that questions such as who can take part in a specific BP activity, or who is never involved in a certain BP, can be automated.

RAL was developed to address specific gaps existing in Business Process Management (BPM). Specifically, it was designed following a rationale grounded on an already defined organisational metamodel and on the Workflow Resource Patterns (WRPs) [1], which capture the way in which resources can be managed in BPs. Thus, one of the limitations of the language is that it can only be used in organisations that implement certain organisational structures supported by the metamodel. However, RAL was designed as a modular language to make it extensible, due to several reasons, a.o.: (i) the selection of human resources might be required in other domains or for further purposes, e.g., to select people for the composition of teams; (ii) different organisations might have different organisational structures, which, besides, might evolve over time; (iii) the selection of non-human resources might also be useful in BPM and other domains.

* This work was partially supported by the European Union's Seventh Framework Programme (FP7/2007-2013), the European Commission (FEDER), the Spanish and the Andalusian R&D&I programmes (grants 318275 (GET Service), P12-TIC-1867 (COPAS), TIN2012-32273 (TAPAS), TIC-5906 (THEOS)).

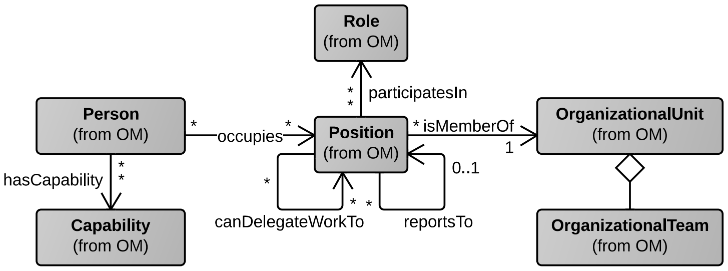

Fig. 1: Excerpt of the organisational metamodel described by Russell et al. [1]

In this paper, a methodology to extend RAL is introduced, describing guidelines and tips on how to perform each step. In addition, as a proof-of-concept of the application of the methodology, the language is extended to support the World Wide Web Consortium (W3C) Organisation Ontology (OO) in order to show how it can be made compatible with other organisational metamodels.

The rest of this paper is structured as follows. In Section 2, RAL is outlined. In Section 3, the methodology to extend it is described. In Section 4, a concrete example of the implementation of the methodology is detailed. Finally, in Section 5 the conclusions drawn from this work are presented.

2 Background. RAL

Resource Assignment Language (RAL) is a Domain Specific Language (DSL) explicitly developed to define conditions to select which resources can be potential participants of a BP activity, i.e., to assign resources to activities. It was first introduced in [2], extended in [3] and revised in [4]. Its specification and its implementation within Collection of Resource-centrIc Supporting Tools And Languages (CRISTAL) can be found at `http://www.isa.us.es/cristal/`.

The language allows formulating expressions that make reference to elements of an organisational model and to elements of a BP model. In particular, in the current version, the organisational metamodel has to (at least partially) implement the organisational metamodel described by Russell et al. in [1], which consists of persons, capabilities, positions, roles and organisational units, as depicted in Fig. 1. The BP metamodel used in RAL is highly inspired in Business Process Model and Notation (BPMN) [5], abstracting the most important characteristics of the current approaches of such types of metamodels and adapting some parts for resource analysis purposes. With these elements, RAL specification and semantics were defined as summarised next.

2.1 RAL Specification

RAL is a modular language composed of *expressions* that may contain different types of *constraints*. It comprises RAL Core, which allows defining basic resource

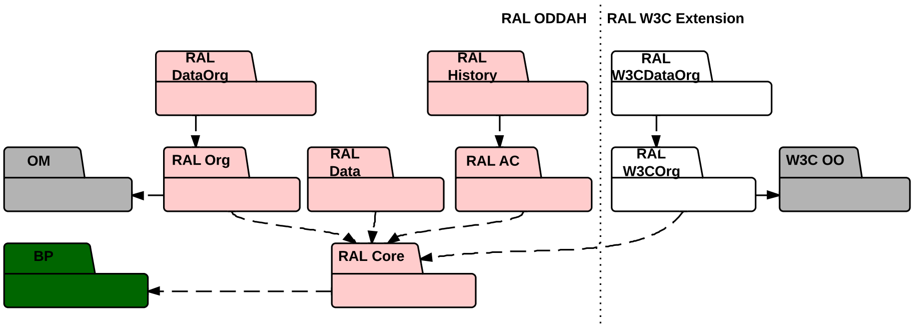

Fig. 2: RAL ODDAH and extension to support the W3C Organisation Ontology

assignments; and several extensions that add new types of expressions and/or constraints. In particular, RAL ODDAH is made up of the modules depicted on the left hand side of Fig. 2[1] and outlined next[2]. Their Extended Backus-Naur Form (EBNF) syntax can be found in [4].

RAL Core contains four types of expressions (*PersonExpr*, *IsAssignmentExpr*, *NegativeExpr*, *CompoundExpr*) and one type of constraint (*PersonConstraint*) to define conditions based on people's characteristics. Among others, it allows specifying conditions such as `IS David`, and `(NOT (IS Anna)) AND (NOT (IS Alex))`.

RAL Org extends RAL Core with four types of expressions (*GroupResourceExpr*, *CommonalityExpr*, *CapabilityExpr*, *HierarchyExpr*) to select people according to their organisational information on the ground of the metamodel depicted in Fig. 1, and four types of constraints (*PositionConstraint*, *RoleConstraint*, *UnitConstraint*, *CapabilityConstraint*) to tune the selection conditions. It allows specifying conditions such as `HAS ROLE PhDStudent`, `HAS CAPABILITY Degree`, and `CAN DELEGATE WORK TO POSITION HrmsPhDStudent`, among others.

RAL Data** and **RAL DataOrg allow selecting individuals or group resources[3] indicated in a data field, by extending *PersonConstraint* (*RAL Data*), and *PositionConstraint*, *RoleConstraint* and *UnitConstraint* (*RAL DataOrg*). They allow specifying conditions like `IS PERSON IN DATA FIELD Form.Purchaser`, and `IS UNIT IN DATA FIELD Application.DepartmentName`.

RAL AC stands for RAL Access-Control and it extends *PersonConstraint* to enable the specification of constraints related to the resources allocated to

[1] OM and BP represent the organisational and the BP metamodels used in RAL, respectively.

[2] Due to space restrictions, the extension of RAL ODDAH described in Section 4 is also depicted in Fig. 2.

[3] *Group resource* is used to jointly refer to positions, roles and organisational units.

other activities, specifically Binding of Duties (BoD) or Segregation of Duties (SoD) with the actual performer of another activity of *the same* BP instance, e.g., `IS ANY PERSON responsible for ACTIVITY SubmitCRV`, and `NOT (IS ANY PERSON accountable for ACTIVITY SignAuthorisation)`.

RAL History extends RAL AC's *PersonConstraint* to enable referencing previous execution instances of a BP, giving rise to conditions like `IS ANY PERSON accountable for ACTIVITY SubmitCRV IN ANOTHER INSTANCE`, and `IS ANY PERSON responsible for ACTIVITY BuyTickets FROM 01/01/2014 TO 01/07/2014`.

The RAL modules can be composed with each other to define more complex conditions. For instance, `(HAS UNIT Hrms) AND (SHARES SOME POSITION WITH ANY PERSON responsible for ACTIVITY SubmitCRV)` combines RAL Org and RAL AC. In order to make RAL expressive, the design goal was to provide as much support as possible for the following selection criteria: (i) the relationships represented in the organisational model of the company; (ii) the Creation Patterns, a subset of the WRPs [6] that capture behaviour related to resource selection; and (iii) the different degrees of responsibility a person can have for an activity, i.e., the *task duties* associated to the activities. Task duties have been considered in recent releases of languages such as WS-HumanTask [7] and BPEL4People [8], as well as in the so-called RACI matrices [9]. In RAL, they are introduced in RAL AC.

As a result, RAL provides capabilities to define all the relationships of an organisational model that implements totally or partially the metamodel used in the language, full support for the Creation Patterns, and the opportunity to assign resources with different degrees of responsibility to a single activity.

2.2 RAL Semantics

RAL semantics were defined considering the formalisation principles defined in [10]. The formalisation is based on a semantic mapping to DLs [11] with the primary objective of establishing a sound basis for sophisticated automated support, supported by two reasons. First, RAL expressions can be seen as a way to specify a subset of the people of an organisation by defining conditions they must satisfy. This way of defining RAL expressions fits nicely into the way DLs express their concepts and, hence, they provide a very natural way to describe the problem, which allows following the *Semantics Priority Principle* and makes it easier to avoid unnecessary representational choices as suggested by the *Conceptualisation Principle*. As a consequence, we can define RAL Core semantics, and then extend them for RAL Org, RAL Data, RAL DataOrg, RAL AC and RAL History without modifying the essence. The second reason is that there is a plethora of off-the-shelf DL reasoners that can be used to automatically analyse RAL expressions and, thus, to automatically infer information from them.

In order to formalise RAL using DLs, every model related to RAL has to be mapped into DL elements either in the TBox or in the ABox of a Knowledge Base (KB). Although there is a significant number of concepts in the problem domain, we tried to keep the number of concepts in the formalisation as small as possible, as suggested by the *Orthogonality Principle*. The detailed description of RAL semantics can be found in [12].

3 Methodology to Extend RAL

Two different approaches can be followed to extend RAL. The first one is applicable when a new organisational metamodel needs to be used, and consists of mapping the concepts and relationships of such metamodel to the ones depicted in Fig. 1, and then use RAL as defined in Section 2. Query View Transformation (QVT) techniques could be used to this end. However, this option may not be feasible if the new organisational metamodel highly differs from the one used in RAL. If this is the case, or the purpose of the extension is to apply RAL to a different domain, the second option should be used, which consists of extending RAL specification and semantics as is explained next.

3.1 Define Expressions and Constraints

First of all, RAL specification has to be extended. For instance, in case of extending RAL to support a new organisational metamodel, new concepts and new organisational relationships may be considered to define resource selection conditions. We next describe the principles taken into account for the design of RAL expressions and constraints, which should be used as guideline for extensions:

1) *Keep the EBNF specification compact.* The goal is to learn to use the language by following similar patterns for the definitions of the selection conditions. For instance, for all the organisational entities, the pattern `HAS + orgEntityName + id` has been used, so that the user can define conditions such as `HAS ROLE Researcher`, `HAS POSITION HrmsClerk` and `HAS UNIT Hrms`.

2) *Keep syntax close to natural language.* The aim is to increase understandability and readability.

3) *Keep the minimal complete subset of coherent selection conditions.* For that purpose, it is also necessary to consider the rationale used to define RAL expressions and constraints (cf. Section 2). In our design of RAL, this implied, e.g., limitting the types of expressions that can be negated. As a consequence, we found expression type `NOT (IsAssignmentExpr)` very generic, and expression type `NOT (CompoundExpr)` unnecessarily hard to understand and automate; hence, they were left aside the language specification.

4) *Keep a distinction between expressions and constraints.* An expression is a type of selection conditions, whereas a constraint specifies restrictions that can be defined for an expression type, i.e., variations regarding the conditions bounded to a type of selection expression. In general, all the conditions related to an expression type begin the same way or follow some pattern, but specific parts can be configured with constraint types for different selection criteria. Therefore, constraints are defined to be reused in different expression types. For instance, the pattern shown in the previous point is used for the expression type *GroupResourceExpr*. Another example is pattern `IS PersonConstraint` defined for *PersonExpr*, which leads to expressions such as `IS David` and `IS PERSON IN DATA FIELD Form.Purchaser`; the former specifies a particular person and belongs to RAL Core, and the latter indicates that the person is written on a field of a data object and belongs to RAL Data. *PersonConstraint* is, in turn, used in other expression types, e.g., *CommonalityExpr*.

5) *Keep module decomposition reasonable.* New RAL expressions and constraints may give rise to new RAL modules, whose scope must be well defined.

It may not be possible to meet 100% of the five aforementioned recommendations simultaneously, so a balance among them should be pursued, instead. For instance, the syntax of the language may be affected by a compact EBNF specification, but reasonable readability levels must be ensured. In the extension presented in Section 4, we follow the order proposed above. However, it may change depending on the purpose of the extension. For instance, if it is intended to be used by developers mainly, the second guideline might not be so important.

3.2 Define DL-Based Semantics

RAL DL-based KB must be extended by adding elements to the TBox and the ABox, and doing all the configurations required following a similar procedure as the one used to define RAL semantics, which is summarised next.

1) *Define a mapping from the expressions of the extension to DLs.* Since all expressions define resource selection conditions, the new RAL expressions must be mapped to DL concepts that are subconcepts of Person so that all people included in a DL concept meet the conditions of an expression.

2) *Define a mapping from the constraints of the extension to DLs.* The new RAL constraints must be mapped to DL concepts according to the type defined by the constraint. For instance, a role constraint would be mapped as a subconcept of DL concept Role.

In addition, if the expressions and constraints introduced in the extension refer to concepts of a metamodel that has not been included by other RAL extension yet, it is necessary to map them according to the following guidelines.

3) *Map the new metamodel(s) used by the extension in the TBox.* This involves mapping the classes and relationships of the new metamodel(s) into concepts and properties in the TBox, respectively, together with the required configuration, e.g., cardinality restrictions.

4) *Map the instances of the metamodel(s) in the ABox.* This involves mapping the elements of specific models into individual assertions in the ABox as instances of the concepts introduced in the previous step.

5) *Configure the KB with additional DL axioms.* Specifically, DL axioms must be added to deal with the *open world assumption*, which involves defining every concept and property as different from each other with the *disjoint with* axiom, and stating that there are no relationships between individuals other than those explicitly defined.

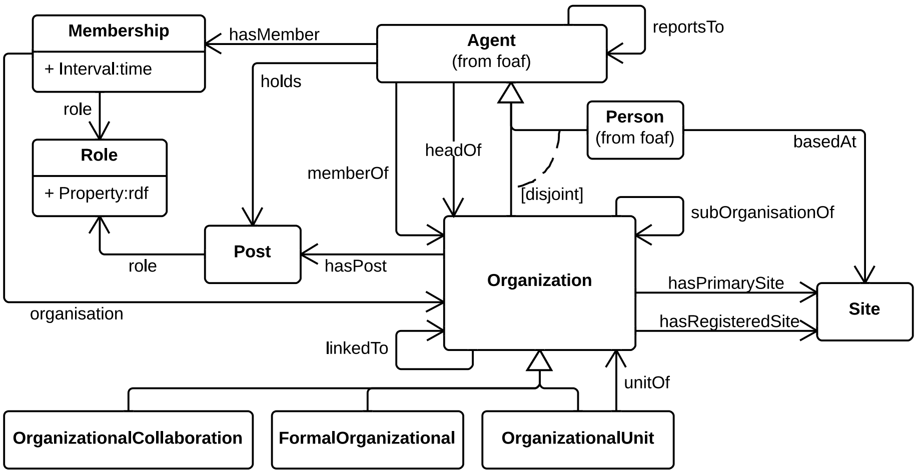

Fig. 3: W3C Organisation Ontology

4 Using RAL with the W3C Organisation Ontology

As a proof-of-concept, in the following we describe the extension of RAL to make it compatible with the OO defined by the W3C as recommendation to represent organisational structures[4], whose metamodel is depicted in Fig. 3.

In short, an *Agent* (an *Organisation* or a *Person* as defined in another W3C ontology) is member of one or more organisations that she can head. The metamodel is intended to be used in many different types of organisations, so particular sub-classes of organisation are distinguished. Organisations can be structured hierarchically and exist in locations (class *Site*). The primary site indicates the default means by which an organisation can be contacted, and the registered site indicates a legally registered site for the organisation. A person is also located at a specific site. Agents can play *Roles* in organisations during a specific period of time, which is represented with class *Membership*. Roles are described by a set of properties that define the associated rights and duties. Furthermore, a set of positions (class *Post*) can be defined as basis to define the reporting line. Unlike roles, positions exist regardless of whether they are currently held by some agent or not. The W3C OO also includes the concept of organisation evolution, but it is disregarded in this section due to space limitation.

4.1 Adding New Expressions and Constraints to RAL

The extension of RAL specified in Language 1 has been defined following the principles described in Section 3.1. It is the result of adding two new modules (cf. right hand side of Fig. 2). RAL W3COrg enables assignments based on the

[4] `http://www.w3.org/TR/vocab-org/`

Language 1 Specification of modules RAL W3COrg and RAL W3COrgData

```
1   RALExpression := ...  |  W3CExpr
2
3   PersonConstraint := personName
4                     |  LOCATED [IN | AT] address
5
6   W3CExpr  :=  W3CGroupResourceExpr
7             |  W3CReportingExpr
8
9   W3CGroupResourceExpr  :=  IS HEAD OF OrgConstraint
10     | HAS ( OrgConstraint  |  PostConstraint )
11     | (HAS  |  HAD)  RoleConstraint
12
13  W3CReportingExpr  :=  REPORTS TO ( PersonConstraint  |  PERSON WHO W3CExpr)
14     | IS REPORTED BY ( PersonConstraint  |  PERSON WHO W3CExpr)
15
16  OrgConstraint  :=   orgName
17     | [OrgType] [WITH SiteType LOCATION address]
18           [WHICH IS A (SUBORGANISATION  |  SUPERORGANISATION) OF orgName]
19     | OrgType IN DATA FIELD dataObject.fieldID
20
21  PostConstraint := POST (positionName  |  WITH ROLE roleName+
22     | IN DATA FIELD dataObject.fieldID) IN OrgConstraint
23
24  RoleConstraint := ROLE (roleName
25     | IN DATA FIELD dataObject.fieldID) IN OrgConstraint [TimeInterval]
26
27  OrgType := ORGANISATION  |  ORGANISATIONAL COLLABORATION
28     | FORMAL ORGANISATION  |  ORGANISATIONAL UNIT
29
30  SiteType := PRIMARY  |  REGISTERED  |  λ
```

W3C OO (cf. Fig. 3) by extending RAL Core with a new type of expression (*W3CExpr*, line 1) that involves two subtypes of expressions, and a new *PersonConstraint*[5] that allows selecting people depending on their location (line 4). RAL W3CDataOrg extends RAL W3COrg to support Deferred Allocation (cf. WRPs [6]). The expressions and constraints of both modules are described next:

- **W3CGroupResourceExpr (line 6):** It involves all the conditions for the selection of people according to characteristics of the group resources they are related to, i.e., (i) the organisations they head (line 9), (ii) the organisations they are member of (line 10), (iii) the positions they hold (line 10), and (iv) the roles they play (line 11). Notice that, in order to keep the EBNF specification compact and, thus, ease learnability, the pattern `HAVE + restOfExpr` has been used instead of each specific relationship of the metamodel. That pattern could not be applied to point (i), but the name of the relationship defined in the organisational metamodel has been used, instead.
- **W3CReportingExpr (line 7):** It allows defining restrictions according to the reporting line defined in the organisation (lines 13-14), either by specifying a person given by a *PersonConstraint*; or by defining a new restriction with the help of the W3C OO extension itself.

[5] *PersonConstraint* was introduced in RAL Core and extended in RAL Data and RAL AC [4].

- **OrgConstraint:** It allows defining restrictions with regard to the organisations of which the potential performers are member. Specifically, RAL W3COrg allows specifying (i) a concrete organisation (line 16); and (ii) an organisation of a determined or undetermined type with specific characteristics regarding its location (line 17), and its hierarchical relationship with other organisations (line 18). RAL W3CDataOrg allows indicating that the organisation of which the potential performers are member are specified in a data field of a BP (line 19). Note that relationship *linkedTo* of the W3C OO has been excluded following principle 3 defined in Section 3.1, since it is a very generic relationship for which applicability has not been found without performing a refinement, e.g., to provide details about the connection between the organisations. However, that is out of the scope of this paper.
- **PostConstraint:** It allows defining restrictions with regard to the positions held by the potencial performers (line 21), specifically a position in particular, or a position related to a specific (set of) role(s) in an organisation, in the case of RAL W3COrg; and a position indicated in a data field within an organisation, in the case of RAL W3CDataOrg.
- **RoleConstraint:** It allows defining restrictions with regard to the roles played by the potential performers in an organisation for a specific period of time, specifically a concrete role, in the case of RAL W3COrg (line 24); and a role indicated in a data field, in the case of RAL W3CDataOrg (line 25).

4.2 Extending RAL Semantics

Since the expressions and constraints of the extension refer to elements of a new metamodel, the W3C OO, the extension of the RAL semantics involves mapping the metamodel together with the expressions and constraints. According to Section 3.2, three steps are necessary to include the metamodel in the DL-based KB. However, in this case the first and second steps are not necessary since the W3C OO is already defined as a DL-based KB itself. Therefore, only the step to deal with the open world assumption is necessary. For instance, if there is an Agent a that just holds one Post p, it is necessary to add to the DL-based KB that $holds(a, p)$ and that $\{a\} \sqsubseteq \forall holds.(\{p\})$, i.e., that agent a only holds position p.

Concerning the mapping of the expressions and constraints, it is necessary to define a mapping μ from expressions and constraints to DL concept descriptions. Next, we detail some of them.

$$\begin{aligned} \mu(\texttt{IS HEAD OF OrgConstraint}) &\equiv \exists headOf.(\mu(OrgConstraint)) \\ \mu(\texttt{HAS RoleConstraint}) &\equiv \exists hasMember.(\exists role.(\mu(RoleConstraint))) \\ \mu(\texttt{POST WITH ROLE r IN OrgConst}) &\equiv \exists role.(\{r\}) \sqcap \exists hasPost^{-}.(\mu(OrgConst)) \end{aligned}$$

The first two mappings map expressions included in *W3CGroupResourceExpr*. Note that the DL concepts in which the expressions are mapped are subconcepts of Person and both of them use the mapping again to map the constraints referenced in the expression. The third mapping maps a *PostConstraint* into a DL concept that is subconcept of Post.

5 Conclusions

In this paper, a methodology to extend RAL has been introduced. RAL is a language for the selection of human resources to be assigned to BP activities, which may require an adaptation or extension to be used for other purposes. As an example of the use of the methodology, RAL has been extended to support a new organisational metamodel, specifically the W3C OO. Extensions to apply RAL to other domains could also be useful, e.g., in the context of document management in order to define permissions for data access, thus replacing the BP-related part of RAL for document management aspects.

The aim of this work is to make it easier for any user to design and use ad-hoc versions of RAL following the same rationale and principles used to define the specification and semantics of the language. RAL is currently being extended applying such methodology for the selection of teams for collaborative work.

References

1. N. Russell, A. ter Hofstede, D. Edmond, and W. M. P. van der Aalst, “Workflow Resource Patterns,” tech. rep., BETA, WP 127, Eindhoven University of Technology, 2004.
2. C. Cabanillas, M. Resinas, and A. Ruiz-Cortés, “RAL: A High-Level User-Oriented Resource Assignment Language for Business Processes,” in *Business Process Management Workshops (BPD'11)*, pp. 50–61, 2011.
3. C. Cabanillas, M. Resinas, and A. Ruiz-Cortés, “Designing Business Processes with History-Aware Resource Assignments,” in *BPM 2012 Workshops (BPD'12)*, vol. 132, pp. 101–112, 2012.
4. C. Cabanillas, M. Resinas, A. del Río-Ortega, and A. Ruiz-Cortés, “Automated Design-Time and Run-Time Analysis of the Business Process Resource Perspective,” *Information Systems*, p. Under review, 2013.
5. OMG, “BPMN 2.0,” recommendation, OMG, 2011.
6. N. Russell, W. M. P. van der Aalst, A. H. M. ter Hofstede, and D. Edmond, “Workflow Resource Patterns: Identification, Representation and Tool Support,” in *CAiSE*, pp. 216–232, 2005.
7. “Web Services-Human Task (WS-HumanTask) v1.1,” tech. rep., OASIS, 2010.
8. “WS-BPEL Extension for People (BPEL4People),” tech. rep., OASIS, 2009.
9. M. Smith, “Role And Responsibility Charting (RACI),” in *Project Management Forum (PMForum)*, p. 5, 2005.
10. A. H. M. T. Hofstede and H. Proper, “How to Formalize It? Formalization Principles for Information System Development Methods,” *Information and Software Technology*, vol. 40, pp. 519–540, 1998.
11. B. Motik and R. Rosati, “Reconciling description logics and rules,” *J. ACM*, vol. 57, pp. 30:1–30:62, June 2008.
12. C. Cabanillas, M. Resinas, and A. Ruiz-Cortés, “Defining and Analysing Resource Assignments in Business Processes with RAL,” in *ICSOC*, vol. 7084, pp. 477–486, 2011.