Towards RAM-Based Variant Generation of Business Process Models

Ahmed Tealeb(✉), Ahmed Awad, and Galal Galal-Edeen

Information Systems Department, Faculty of Computers and Information, Cairo University, Giza, Egypt
ahtealeb@gmail.com, a.gaafar@fci-cu.edu.eg, galal@acm.org

Abstract. Variability management of process models is a major challenge for Process-Aware Information Systems. Process model variants can be attributed to any of the following reasons: new technologies, governmental rules, organizational context or adoption of new standards. Current approaches to manage variants of process models address issues such as reducing the huge effort of modeling from scratch, preventing redundancy, and controlling inconsistency in process models. Although the effort to manage process model variants has been exerted, there are still limitations. Furthermore, existing approaches do not focus on variants that come from change in organizational perspective of process models. Organizational-driven variant management is an important area that still needs more study that we focus on in this paper. Resource Assignment Matrix (RAM) is an important aspect of the organizational perspective that has different variations. This paper introduces an approach inspired by real life scenario to generate consistent process model variants that come from adaptations in the RAM.

Keywords: Organizational-driven variant · Responsibility Assignment Matrix (RAM) · RACI Variants · Business Process Management (BPM) · Process-Aware Information Systems (PAISs)

1 Introduction

In recent years, the increasing adoption of PAISs has resulted in large process model repositories [1]. One of the fundamental challenges of modeling business process is to deal with the multitude of variants that may exist for a particular process [2]. Each process variant constitutes an adjustment of a reference or basic process model to specific requirements. Efficient management of process model variants is a critical issue for organizations with the aim of helping them reduce the huge effort of modeling from scratch, prevent redundancy, and tackle inconsistency in process models.

Despite the efforts done in current approaches e.g., in Provop [3], C-EPCs [4], and PPM [5] to manage process model variants, there are still unaddressed issues. Current approaches focus on dealing with variants coming from change in

A. Norta et al. (Eds.): ICSOC 2015 Workshops, LNCS 9586, pp. 91–102, 2016.
DOI: 10.1007/978-3-662-50539-7_8

the control and the behavioral perspectives of process models. However, variants originating from the organizational perspective still need to be studied.

The organizational perspective is one of the different views integrated in the process model. It identifies the hierarchy of the organization within which the process will be executed. Russell et al. [6] introduced a set of Workflow Resource patterns (WRP) to capture the requirements for resource management such as representation and utilization in workflow environment [6]. Awad et al. [7] proposed an extension metamodel for BPMN to enable representation of resource assignment constraints using Object Constraints Language (OCL) [8] to WRP [6]. We, in a previous work, discussed organizational structures as an aspect of the organizational perspective based on general concepts such as abstraction and polymorphism [9].

Another important aspect of the organizational perspective is the Responsibility Assignment Matrix (RAM). RAM enables organization's stakeholders to understand the responsibilities of each person. RAM provides important information about resource assignments which till now is not supported completely by Business Process Model and Notation (BPMN) 2.0 [10]. The aim of this paper is to propose a context-based approach for generating consistent process model variants focusing on the different variations of RAM. We extract the RAM for a given base model. Then, we enable the user to adapt RAM and validate the different changes. Finally, we generate the variant of the base model in hand.

The rest of this paper is organized as follows: Sect. 2 introduces the basic concepts related to our approach and discusses a motivating scenario. Section 3 presents our RAM-based algorithms for generating consistent process model variants. Section 4 discusses related work. Finally, Sect. 5 concludes our approach and outlines directions for future research.

2 Background

In this section, we introduce in brief the background that is related to our approach. We present RAM in brief and the work of Cabanillas et al. [14] for mixing RAM with BPMN as follows in the Sects. 2.1 and 2.2 respectively. Then, we present our motivating scenario in Sect. 2.3.

2.1 RAM in Brief

RAM is a matrix-based chart represented in a grid in order to identify the roles (people or groups or departments) and their responsibilities for a given task. RAM is also known as RACI (pronounced 'racey') matrix or Linear Responsibility Chart (LRC). Project Management Institute defined RACI in [11] as a common type of RAM that uses responsible, accountable, consult, and inform statuses to define the involvement of stakeholders in project activities. RACI matrix has four association types or patterns as defined in [12]:

– *Responsible (R):* "The Doer" who is actually responsible for the execution of the work. There is typically one person responsible for a task but the responsibility can be shared in some cases. Moreover, the person who is *Responsible (R)* for a task in some cases is also *Accountable (A)*.

- *Accountable (A):* "The Buck Stops Here" who ultimately approves or authorizes the work performed by the person *Responsible (R)* for a task. There must be one and only one role/person accountable for any given task.
- *Consulted (C):* "In the Loop" who are the subject matter experts to be consulted before and during the task performed as a two-way communication.
- *Informed (I):* "Keep in the Picture" who need to be informed about the status of the task performed as a one-way communication.
- *Support (S):* is another association type that is sometimes used in addition to the above four types who provides the resources and hence support for a specific task. This is an optional type and if this category used along with the other four RACI categories, then matrix will be called RASCI matrix.

There are several variants for RACI matrix such as RASI, RASCI, RACI-VS, DACI, CAIRO [13]. We will make use of some of these alternatives as variants for the RACI matrix as will be explained in Sect. 3.

2.2 RASCI-aware BPMN

Cabanillas et al. [14] introduced an approach for mixing RAM information with business process models together in BPMN. The approach is based on *RASCI patterns* in order to make the process models RASCI-aware [14]. The five RASCI patterns are Responsible, Accountable, Support, Consulted, and Informed patterns. Moreover, they proposed an extension of BPMN with a set of RASCI tasks and collapsed subprocesses. They identified some specifications and constraints for these patterns along with process model in BPMN [14]. We use these patterns with some adaptations in our approach as in Sects. 3.1, 3.2 and the Appendix: RAM Patterns. Furthermore, they used Resource Assignment Language (RAL) which is defined with examples in [15,16] instead of XPath to build resource assignment expressions. RAL allows expressing both *direct assignments* extracted from RASCI matrices as well as *binding information*. Cabanillas et al. [17] implemented CRISTAL tool, which composed several tools such as RACI2BPMN, DT RAL Solver, and RT RAL Solver. CRISTAL is used for generating RACI-aware business process model automatically.

The aim of Cabanillas et al.'s [14–17] intensive work is introducing a novel approach to integrate RACI matrix information with process model using BPMN. However, our approach addresses the issue of generating consistent process model variants from adaptations in RACI matrix based on context.

2.3 Motivating Scenario

In this section, we introduce a real life scenario that motivated the development of our approach. We introduce the base model for "Purchase Request" business process enriched by Cabanillas et al. RASCI patterns.

Purchase request (PR) is one of the most frequently executed business processes in organizations. The process of PR starts when an employee creates a PR. Then, the employee may consult his Boss in filling the PR. Once the PR is

sent for approval, the Boss of the employee receives the request; the Boss either approves or rejects the PR. If the PR is rejected, the process sends a message with the rejection of PR to the Employee. If the PR is approved, the process sends a message with approval of PR to the Purchasing Department (PD). The PD gets quotations and makes a purchase order. Figure 1 represents a *base model* for the "Purchase Request" process.

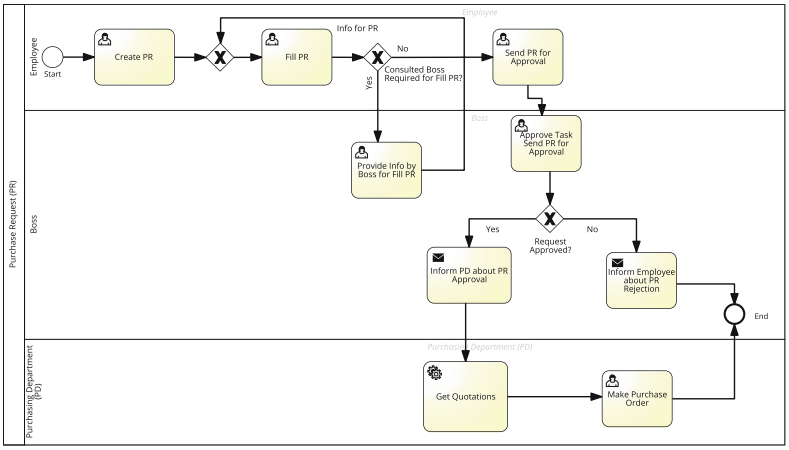


Fig. 1. Purchase request - base process model

The PR process may vary from one department to another as in the Ministry of Interior (MOI) — State of Qatar. We discuss in detail how these variants may be generated in Sect. 3.

3 RAM-Based Variants Generation

In this section, we introduce a solution for generating consistent process model variants based on RAM and RASCI-aware BPMN mentioned in Sect. 2. We present two RAM-based algorithms as follows: "*RAM Extraction*" and "*RAM-Based Variant Generator*" in the Sects. 3.1, and 3.2 respectively. Firstly, the RAM Extraction algorithm generates the corresponding RAM for a given base model. The base model has to be a RASCI-aware model. Secondly, we enable the user to adapt the Extracted RAM as in Sect. 3.1. Thirdly, RAM-Based Variant Generator validates the user's inputs in RAM. Finally, we generate the variant process model for the given base model as in Sect. 3.2.

3.1 RAM Extraction

The "RAM Extraction" algorithm is responsible for generating the RAM for a given base process model as in Fig. 1. RAM Extraction starts with reading the

base model either enriched with RASCI-aware BPMN or Not. Then, it extracts the corresponding RAM from the base model. If the base model is not enriched with RASCI patterns then only the 'Responsible' pattern is displayed.

Algorithm 3.1. RAM Extraction.

Inputs: BM is the base model of a process
Outputs: RAM is the responsibility assignment matrix of the base model
Constants:
A represents the Accountable pattern, C represents the Consulted pattern, I represents the Informed pattern, S represents the Support pattern
Variables:
LS is a set of different lanes in base of process model
L represents one lane
T represents a task
RAM initially is "Two Dimensional Table"

```
1  For each L in LS
2    RAM.addRoleToCoulmn(L)
3     For each T in L
4       RAM.addTaskToRow(T)
5       RAM.updateCell(L, T, R)
6     End For
7  End For
8  For each T in BM
9     L = BM.getLaneName(T)
10    If (T.Name Like 'Approve Task%') Then
11      RAM.updateCell(L, T, A)
12    Else
13      RAM.updateCell(L, T, R/A)
14    End If
15    If (T.Name Like 'Provide Info by '+L+'%') Then
16      RAM.updateCell(L, T, C)
17    End If
18    If (T.Name Like 'Inform '+L+' about %') Then
19      RAM.updateCell(L, T, I)
20    End If
21    If (T.Name Like 'Decide if Support from '+L+' Required for%') Then
22      RAM.updateCell(L, T, S)
23    End If
24 End For
```

Lines 1–7 are responsible for constructing a RAM Table with 'Responsible (R)' pattern for the given BM. Line 8 reads each task in the BM. Line 9 get the lane name for the current task. In case 'Accountable' pattern exists; Update intersected cell between 'Task Name' and the Role to 'A' as in Lines 10–11. Otherwise, Update intersected cell between 'Task Name' and its assigned role to 'R/A' as in Lines 12–14. In case 'Consulted' pattern exists; Update intersected cell between 'Task Name' and the Role to 'C' as in Lines 15–17. In case 'Informed'

pattern exists; Update intersected cell between 'Task Name' and the Role to 'I' as in Lines 18–20. In case 'Support' pattern exists; Update intersected cell between 'Task Name' and the Role to 'S' as in Lines 21–23. Line 24 exits while reach the end of tasks in the base model. For more details about RAM patterns, see "Appendix - RAM Patterns". Moreover, we will enable the user to configure other "Keywords" or "expressions" than the used in the algorithm for each RAM pattern.

Table 1 shows the corresponding RACI matrix for the base model in Fig. 1 based on RAM Extraction algorithm discussed in Sect. 3.1.

Table 1. RACI matrix of base process model

Tasks—Roles	Employee	Boss	Purchasing Department (PD)
Create PR	R/A		
Fill PR	R/A	C	
Provide Info by Boss for Fill BR		R/A	
Send PR for Approval	R	A	
Approve Task Send PR for Approval		R/A	
Inform PD about PR Approval		R/A	I
Inform Employee about PR Rejection	I	R/A	
Get Quotations			R/A
Make Purchase Order			R/A

We enable the user to configure and adapt the extracted RAM in Table 1 based on context. The users of different departments may apply different RAM patterns for the same task. Table 2 represents RASI as one of RACI's different variants. The assignment of consultation of "Boss" for "Fill PR" is replaced by another assignment. Firstly, we remove the 'C' from intersection cell Boss and "Fill PR"; colored with red. Based on this, the whole row for the task "Provide Info by Boss for Fill BR" will be removed also as this task is part of Consulted pattern; colored with red. Secondly, we add a new role called "Administrative Manager (AM)" with the orange color. Finally, we update the intersection cell between "AM" and "Fill PR" by 'S' for Support responsibility with green color. The new assignment means that the "Employee" delegates the task "Fill PR" to the "AM".

3.2 RAM-Based Process Variant Generator

The "RAM-Based Process Variant Generator" algorithm manages the issue of generating consistent process model variants caused by adaptations in RAM generated by "RAM Extraction" algorithm. RAM-Based Process Variant Generator validates the "Adapted RAM" as in Table 2. Then, it manipulates the base model using the "Adapted RAM" to generate consistent process model variant.

Table 2. RASI matrix of base model

Tasks—Roles	Employee	Boss	Purchasing Department (PD)	Administrative Manager (AM)
Create PR	R/A			
Fill PR	R/A			S
Provide Info by Boss for Fill BR		R/A		
Send PR for Approval	R	A		
Approve Task Send PR for Approval		R/A		
Inform PD about PR Approval		R/A	I	
Inform Employee about PR Rejection	I	R/A		
Get Quotations			R/A	
Make Purchase Order			R/A	

General Validations for RAM:

- For each task, there is one and only one role with Accountable (A).
- For each task, there is at least one role with Responsible (R); we assume also one and only one role with Responsible (R) for simplicity as configurable validation that may or may not validated based on user's selection.

Algorithm 3.2. RAM-Based Process Variant Generator.

Inputs: BM is the base model of a process; Validated Adapted RAM the output of RAM Extraction
Outputs: VPM variant process model
Variables:
RAMChanges[] is an array of changes to RAM
OPR represents the operation done for each changed cell in RAM Changes[]
CCV represents the value of each changed cell in RAM Changes[]
VPM initially is the base model

```
1  i = 0; VPM = BM
2  For each i in RAMChanges[]
3     OPR = i.getCellOperation()
4     CCV = i.getCellValue()
5     // OPR and CCV recognize the RAM patterns tasks relationships.
6     If OPR = Insert in Empty Cell Then
7        VPM.insertPattern(CCV)
8     Else If OPR = Delete from Existing Cell Then
9        VPM.deletePattern(CCV)
10    Else If OPR = Update Existing Cell Then
11       VPM.updatePattern(CCV)
12    Else If OPR = Insert New Role Then
13       VPM.insertNewLane(CCV)
14    Else If OPR = Insert New Task Then
15       VPM.insertNewTask(CCV)
16    End If
17    i = i + 1
18 End For
19 return VPM
```

Line 1 defines and initializes a counter for the array of RAM changes and set VPM initially to BM. Line 2 reads each change in the array of RAM changes. Line 3 retrieves the operation performed on the changed cell in RAM. Line 4 retrieves the value of the changed cell in RAM. Then, the algorithm checks the type of OPR. Lines 6–7 if correct, algorithm performs an insert operation for the desired pattern based on value in Line 4. Lines 8–9 if correct, it performs a delete operation for the desired pattern based on value in Line 4. Lines 10–11 if correct, it performs an update operation for the desired pattern based on value in Line 4. Lines 12–13 if correct, it performs an insert operation for a new lane based on value in Line 4. Lines 14–15 if correct, it performs an insert operation for a new task based on value in Line 4. Line 19 returns the generated VPM. These operations applied to the process base model using the RAM patterns as in "Appendix - RAM Patterns".

Figure 2 shows the generated variant of the base model in Fig. 1. Figure 3 shows the detailed subprocess of "Fill PR" with 'Support (S)' pattern and the added role "Administrative Manager".

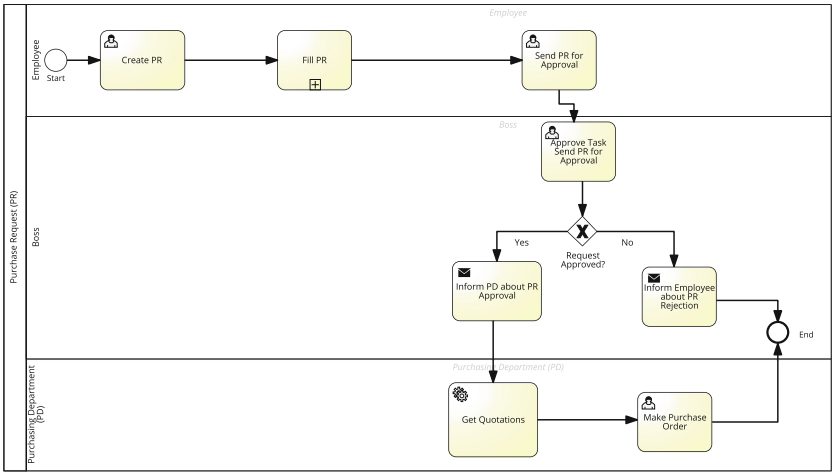


Fig. 2. Purchase request - RASI variant

So, we can conclude that the common source for *the variant* introduced for the process of "Purchase Request" before is the changes in RAM.

4 Related Work

Several approaches have been developed in recent years to manage the different variants of process models, Such as PROcess Variants by OPtions (Provop),

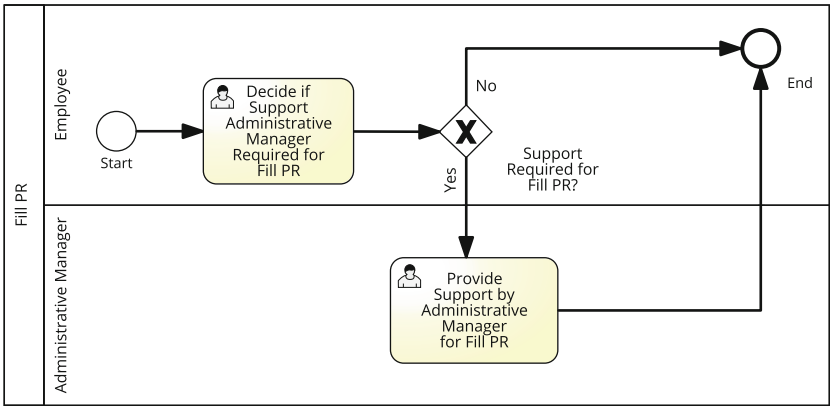


Fig. 3. Fill PR subprocess

Configurable Event-driven Process Chains (C-EPCs), and Partial Process Models (PPM). In this section, we state the pros and cons for each approach.

Provop is an approach for managing a set of related process variants throughout the entire Business Process Life Cycle (BPLC) [3]. In Provop, a specific variant is derived by adjusting the basic process model using a set of well-defined change operations [3]. Change Operations represent the difference between basic model and variant such as INSERT, DELETE, and MOVE process fragments, and MODIFY process elements attributes. Furthermore, Provop supports the context-aware process configuration either statically or dynamically [18]. The Provop lifecycle [19] consists of three major phases: the modeling phase, the configuration phase and the execution phase. Provop has been extended with a procedure to guarantee the correctness and soundness for a *family of configurable process variants* [20]. An extension has been developed for ARIS Business Architect to cope with variability in process models based on Provop [2]. Provop uses a bottom-up technique from process variants to the basic process model. Each variant is maintained through the base model only. So, the changes in any variant may not be consistent with other variants of the same process.

The concept of configurable process model has been defined by [4]. It merges variants of process models into a single configurable model. Configurable process models are integrated representations for variants of a process model in a specific domain. A framework to manage the configuration of business process models consists of three parts: *a conceptual foundation for process model configuration, a questionnaire-based approach for validating modeling, and a metamodel for holistic process configuration* [21]. C-EPCs are configurable version of EPCs, which provides a means to capture variability in EPC process models.

C-EPCs identify a set of variation points which are called *configurable nodes* in the model and constraints, which are called *configuration requirements* to restrict the different combinations of allowed variants in order to be assigned for variants called *alternatives* [21]. La Rosa et al. [22] proposed C-iEPC, that extends C-EPC notation with the notions of roles and objects associated to functions. C-iEPC supports variants from organizational perspective. C-EPCs uses a top-down technique from holistic or reference process model to process variants. Specifying all variants in a holistic reference model for a particular process is difficult to maintain.

PPM is a query-based approach that depends on defining process model views to maintain consistency among process variants [5]. These views are defined using a visual query language for business process models called BPMN-Q [23]. Based on BPMN-Q, a framework for querying and reusing business process models has been developed by [24]. PPM is using inheritance mechanisms from software engineering to make best use of the reusability as a concept of Object-Oriented Modeling of object orientation [5]. PPM approach provides support for consistency of process model variants, and allows handling issues for multiple inheritance levels. PPM uses both top-down and bottom-up techniques in handling process variants. Context issues related to variants of business process are not covered in the PPM approach.

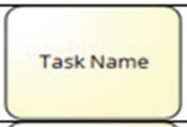
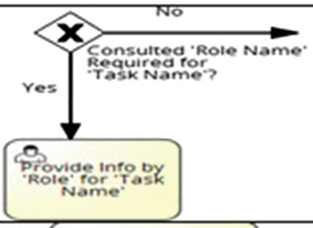
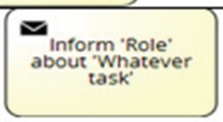
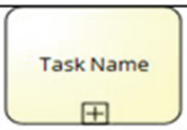
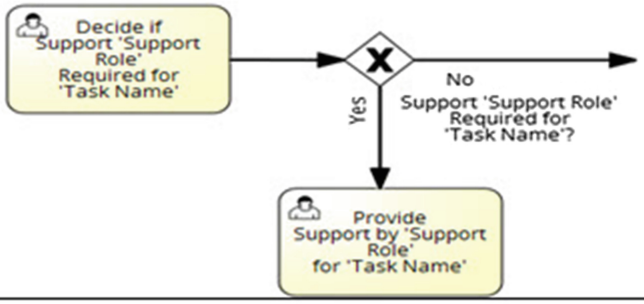
Despite the significant effort that has gone into the current approaches to manage process models variants, the organizational perspective has many aspects still need to be studied. So, Our approach focus on RAM as one important aspect of organizational perspective to manage generating the process model variants consistently.

5 Conclusions and Future Work

This paper introduces an approach to manage the generation of consistent process model variants that come from adaptations in RAM. The approach helps practitioners, such as process owners and/or designers in generating consistent variants of their process models depending on the case in hand. The most significant finding behind the approach is the importance of the organizational perspective's aspects such as RAM. In this paper, we presented two context-based algorithms to derive variants of process models based on RAM. We applied the approach to real life process models to further illustrate our ideas.

In future work, we seek to apply the approach for more real world cases in different domains. Furthermore, we look for other aspects from the organizational perspective to complete our approach. Finally, a proof-of-concept prototype that validates the concept behind approach will be implemented.

Appendix: RAM Patterns

No.	Pattern Name	Pattern Representation
1	Responsible Pattern (R)	
2	Accountable Pattern (A)	
3	Consulted Pattern (C)	
4	Informed Pattern (I)	
5	Support Pattern (S)	 

References

1. Dijkman, R., La Rosa, M., Reijers, H.A.: Managing large collections of business process models - current techniques and challenges. Comput. Ind. **63**(2), 91–97 (2012)
2. Reichert, M., Rechtenbach, S., Hallerbach, A., Bauer, T.: Extending a business process modeling tool with process configuration facilities: the provop demonstrator. In: BPMDemos, CEUR-WS, Ulm, Germany, vol. 489 (2009)

3. Hallerbach, A., Bauer, T., Reichert, M.: Managing process variants in the process life-cycle. In: ICEIS 2008, ISAS-2, Barcelona, Spain, pp. 154–161 (2008)
4. Rosemann, M., van der Aalst, W.M.P.: A configurable reference modelling language. J. Inf. Syst. **32**(1), 1–23 (2007)
5. Pascalau, E., Awad, A., Sakr, S., Weske, M.: Partial process models to manage business process variants. Int. J. Bus. Process Integr. Manage. **5**(3), 240–256 (2011)
6. Russell, N., van der Aalst, W.M.P., ter Hofstede, A.H.M., Edmond, D.: Workflow resource patterns: identification, representation and tool support. In: Pastor, Ó., Falcão e Cunha, J. (eds.) CAiSE 2005. LNCS, vol. 3520, pp. 216–232. Springer, Heidelberg (2005)
7. Awad, A., Grosskopf, A., Meyer, A., Weske, M.: Enabling Resource Assignment Constraints in BPMN. BPT Technical report 04–2009 (2009)
8. Object Constraint Language OCL 2.4 Specification. OMG (2014)
9. Tealeb, A., Awad, A., Galal-Edeen, G.: Context-based variant generation of business process models. In: Bider, I., Gaaloul, K., Krogstie, J., Nurcan, S., Proper, H.A., Schmidt, R., Soffer, P. (eds.) Enterprise, Business-Process and Information Systems Modeling. LNBIP, vol. 175, pp. 363–377. Springer, Heidelberg (2014)
10. OMG: BPMN 2.0 Specification, Technical report (2011)
11. PMI.: A Guide to the Project Management Body of Knowledge: PMBOK® Guide — 5th (edn.), Project Management Institute Inc., Pennsylvania, USA (2013)
12. Smith, M.: Role & Responsibility Charting (RACI). In: PM Forum (2005)
13. Responsibility Matrix —RACI, RASCI, and More, May-2015. http://www.bawiki.com/wiki/techniques/responsibility-matrix-raci-rasci-and-more
14. Cabanillas, C., Resinas, M., Ruiz-Cortés, A.: Mixing RASCI matrices and BPMN together for responsibility management. In: JCIS 2011, vol. 1, pp. 167–180 (2011)
15. Cabanillas, C., Resinas, M., Ruiz-Cortés, A.: Towards the Definition and Analysis of Resource Assignments in BPMN 2.0. ISA-11-TR-01, University of Seville (2011)
16. Cabanillas, C., Resinas, M., Ruiz-Cortés, A.: RAL: a high-level user-oriented resource assignment language for business processes. In: Daniel, F., Barkaoui, K., Dustdar, S. (eds.) Business Process Management Workshops. BPD 2011, vol. 99, pp. 50–61. Springer, Heidelberg (2011)
17. Cabanillas, C., del-Río-Ortega, A., Resinas, M., Ruiz-Cortés, A.: CRISTAL: collection of resource-centrIc supporting tools and languages. In: BPMDemos, CEUR-WS, vol. 940, pp. 51–56 (2012)
18. Hallerbach, A., Bauer, T., Reichert, M.: Context-based configuration of process variants. In: TCoB 2008, Barcelona, Spain, pp. 31–40 (2008)
19. Hallerbach, A., Bauer, T., Reichert, M.: Capturing variability in business process models: the provop approach. J. Softw. Maintenance Evol. Res. Pract. **22**(6–7), 519–546 (2010). Wiley InterScience
20. Hallerbach, A., Bauer, T., Reichert, M.: Guaranteeing soundness of configurable process variants in provop. In: IEEE Computer Society, CEC09, pp. 98–105 (2009)
21. La Rosa, M.: Managing Variability in Process-Aware Information Systems. PhD thesis, Queensland University of Technology, Brisbane, Australia (2009)
22. La Rosa, M., Dumas, M., ter Hofstede, A.H.M., Mendling, M.: Configurable multiperspective business process models. J. Inf. Syst. **36**(2), 313–340 (2011)
23. Awad, A.: BPMN-Q: a language to query business processes. In: EMISA. LNI, Germany, vol. P-119, pp. 115–128 (2007)
24. Sakr, S., Awad, A.: A framework for querying graph-based business process models. In: WWW, pp. 1297–1300. ACM (2010)