A Case Modelling Language for Process Variant Management in Case-Based Reasoning

Riccardo Cognini[2], Knut Hinkelmann[1,4], and Andreas Martin[1,3(✉)]

[1] School of Business, FHNW University of Applied Sciences and Arts Northwestern Switzerland, Olten, Switzerland
{knut.hinkelmann,andreas.martin}@fhnw.ch
[2] Computer Science Department, School of Science and Technology, University of Camerino, Camerino, Italy
riccardo.cognini@unicam.it
[3] School of Computing, University of South Africa, Florida Park, Roodepoort, Johannesburg, South Africa
[4] Department of Informatics, University of Pretoria, Pretoria, South Africa

Abstract. Conventional business process management has been very successful for routine work but has deficiencies in dealing with the flexibility of knowledge workers' work, since the tasks are hard to determine and highly dependent on the current situation. For knowledge workers it is useful to structure the processes just in part as process variants, which can be adapted, modified and even newly created at runtime by them. This paper describes an application of a case-based reasoning approach and introduces a process variant modelling language that supports the manual generation and refinement of generalized process variants. This approach is demonstrated in a public administration scenario.

Keywords: Case modelling · Modelling language · Process flexibility · Knowledge work · Case-based reasoning · Adaptive Case Management

1 Introduction

Knowledge work cannot be represented sufficiently in traditional business process management, where the work is structured and described in advance. It is especially difficult to predict upcoming tasks because knowledge work can deal with different requirements at the same time. Type and scope of tasks are hard to determine in advance, while sequence of tasks and even the tasks themselves may vary due to already achieved results and unforeseeable events. Knowledge work is not routine work, and "[...] the sequence of actions depends so much upon the specifics of the situation [...] necessitating that part of doing the work is to make the plan itself" [1, p. 8]. It is not always possible to define the whole structure including all elements of a knowledge-intensive process at build-time or just before instantiation.

For knowledge work it seems useful to take approaches that structure the BP just in part as process fragments since no fully defined models can be easily

© Springer International Publishing Switzerland 2016
M. Reichert and H.A. Reijers (Eds.): BPM Workshops 2015, LNBIP 256, pp. 30–42, 2016.
DOI: 10.1007/978-3-319-42887-1_3

adapted/modified at runtime by the users [2]. "Process fragments are reflecting the partial and intermittent knowledge one modeller [or a knowledge worker] has at a certain time about a specific situation" [3, p. 399]. Knowledge workers are required to make decisions based on process fragments, which can only be made by the knowledge workers themselves, given the process elements that can only be executed by humans (see e.g., case management model and notation human tasks [4]). In the past, related work introduced sophisticated similarity and adaptation mechanisms, such as case-based reasoning (CBR), which provides huge reasoning power to support process flexibility. Unfortunately, no significant attention has been paid to knowledge workers' need to model process fragments, which require human decision, using CBR during run-time. *Therefore, a CBR vocabulary and case content representation (modelling language) are needed that support the manual planning, modelling, generation and refinement of process fragments during run-time.*

Case-based reasoning (CBR) is a technologically independent methodology that uses the knowledge of previously experienced situations and its solution, to propose a potential solution to a new situation (the current problem) [5]. CBR has been applied in business process contexts, e.g., for workflow retrieval, adaptation, construction and monitoring (see [6–8]). Existing work mainly focuses on the management of structured and predictable business processes. Some related work supports ad-hoc changes of workflows; several sophisticated similarity and adaptation mechanisms and frameworks were developed recently. CBR is originally designed to retrieve, reuse, revise and retain concrete cases for a specific situation [9]. Based on the original notion of CBR, some researchers investigated the generalization and abstraction of cases [10]. This can reduce the complexity of the cases, increase the flexibility and reduce the size of the case base to enhance the retrieval efficiency [9]. Abstraction differs from generalization. According to Mueller and Bergmann [8, p. 396], "[...] abstraction [...] would require reducing the overall granularity of workflows (e.g. less tasks and data items) [...]". This differentiation is particular important when implementing an automatic algorithm.

This work focuses on the man-made modelling of cases without a requirement of reducing the granularity of process fragments depending on the variety of the situation of the knowledge workers. *Therefore, this paper describes the application of a CBR approach and introduces a case-based process fragment modelling language named Business Process Feature Model (BPFM) that supports the manual generation and refinement of generalized cases.* This approach can be appropriately demonstrated in a specific application scenario. In the following we describe a process fragment modelling language that differs from existing work for representing the case content that is appropriate to variants.

2 Application Scenario

In this section we present an application scenario in order to show how to use BPFM as case content representation (modelling language).

The application scenario is the master study admission process of the FHNW. Figure 1 shows the process model in BPMN 2.0. The admission process starts when an application has arrived. In a first activity the study assistant prepares the eligibility check. He collects and prepares all the information in order to allow the dean of the programme to check the eligibility of the candidate. In the case the candidate is eligible, she/he is invited for an oral interview. Otherwise, a rejection letter is sent. If after the interview the candidate is accepted, the administration department determines the tuition fee. At the end an acceptance letter is sent to the candidates. If the candidate is not eligible, a rejection letter is sent.

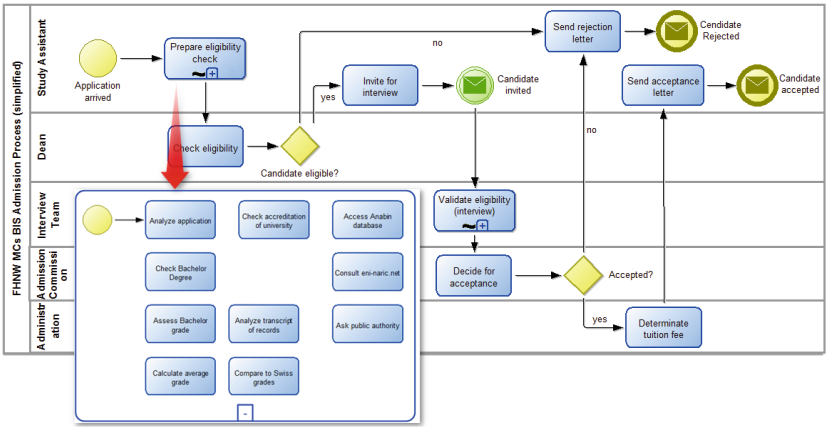

Fig. 1. Master study admission process including prepare eligibility check

Although this process looks like a structured process, the activity *Prepare eligibility check* is a complex knowledge intensive sub-process. In interviews with the stakeholder, the following activities are identified, which are modelled in Fig. 1 as sub-process: It is determined whether the bachelor degree qualifies for the master programme. Because candidates can come from different countries, there can be a huge variety of degrees and certificates. If the bachelor degree is unknown, the transcript of record is analysed. The university from which the candidate earned the bachelor degree is checked for accreditation. If the university is unknown to the study assistant, the study assistant can access a database called Anabin. Furthermore, many countries have a list of accredited universities on the Web. Access to them is provided via enic-naric.net. There are several other databases and online resources. The selection of the appropriate resource depends on the country. If the university cannot be found in any resource, the study assistant can ask a public authority for confirmation. The eligibility furthermore depends on the average grade of the bachelor degree, which must be at least “B”. If the average grade is not mentioned in the transcript of records,

it is calculated by the study assistant. For unknown grading systems one has to find out how it compares to Swiss grades.

It is not clear in advance which activities are required or in which order they are executed. By analysis of a number of cases of the application scenario, we derived research objectives for the CBR approach, especially (1) the modelling of the case content and (2) its characterization. In the terminology of CBR, the characterisation can be regarded as the problem description (of the situation) and the case content can be regarded as the solution or lesson [11], which consists of at least one knowledge item (e.g., documents, processes, etc.) [12].

Demanding Example for Current Modelling Languages. In the following we describe three possible instances of the activity *Prepare eligibility check*. They are real cases derived from the application scenario, pointing out specific aspects that challenge existing modelling approaches. The cases as shown in Figs. 2, 3 and 4 are different solutions to the problem involving a candidate coming from an unknown university.

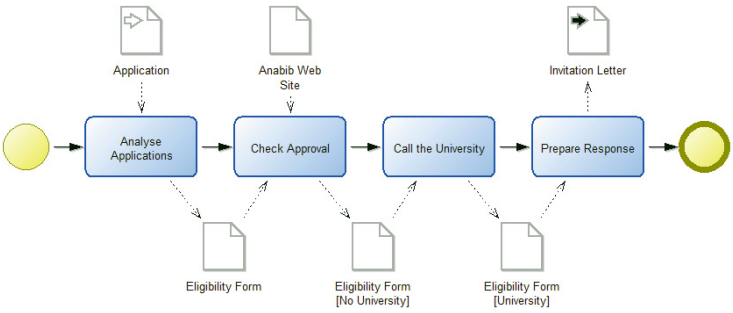

Fig. 2. Case A

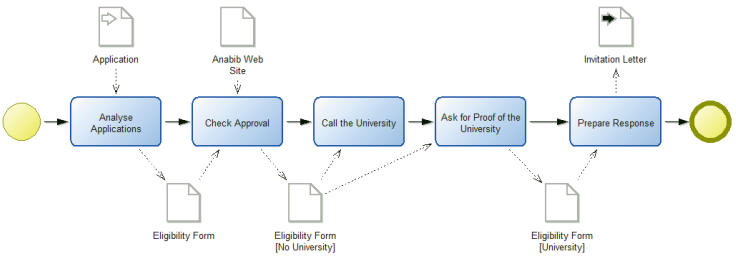

Fig. 3. Case B

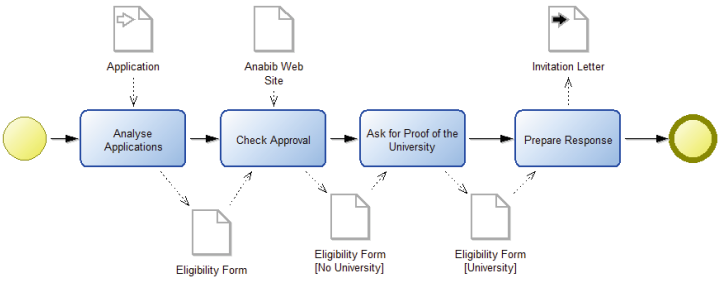

Fig. 4. Case C

In the first case (Fig. 2) the problem is solved by calling the university of the applicant student asking for some information. In the second case (Fig. 3) the university is called and then proof of the existence of the university. In the last case (Fig. 4) just the proof of the of existence of the university is requested.

The described cases are similar, and each one of them can be used to solve the problem. In all the cases there is some implicit knowledge missing. Even though the CBR case description (not shown in figures) can be considered, it is not possible, e.g., to recognize immediately the conditions that lead to a certain decision. To make decisions visible and to present alternative flows, a generalized case could potentially be modelled by the knowledge worker, which might be also abstracted at the same time.

3 Approach Comparison

From our experience BPMN or any other imperative or BPMN-like language seems to be unsuitable to represent cases in the CBR system. In [13], the author gives some motivations specifying why BPMN could not be used for BP modelling in Adaptive Case Management systems; this motivation is also valid in the CBR context.

The end users of a CBR system do not have enough knowledge and skills to model or update a BPMN model. Generally, end users are able to specify which activities should be performed, defining how one has to perform them, but they are not able to establish a temporal order of these activities since they are focused just on their own tasks. In addition, we also have to consider that modifying a BPMN diagram modelled by someone else can be a difficult task like modifying a software source code. Another issue of imperative languages in this context is that they are designed to express something that is fully defined including all the possible aspects of a BP.

In order to deal with BP that cannot be fully defined in recent years, the OMG designed the Case Management Model and Notation (CMMN) language [4]. In contrast to BPMN 2.0, CMMN is a declarative language designed to model no predefined, partially structured and no repeatable BPs. Mandatory and optional activities can be modelled without specifying an execution order or specifying the

Table 1. Comparison of modelling languages

	BPMN	CMMN	DECLARE	BPFM
For BP modelling	Yes	Yes	Yes	Yes
Language type	Imperative	Declarative	Declarative	Declarative
Defined activities flow	Full	In part	In part	In part
Complex constraints	Yes	No	No	Yes
Data representation	Yes	In part	No	Yes
Variants representation	No	No	No	Yes

situation in which an activity can be executed. In our opinion, the main issue of CMMN is that it is not possible to specify complex execution criteria. For instance, it is not possible to specify at least one of the activities in a set has to be executed. Constraints to specify such situation should be defined in order to model more detailed cases. Another issue of CMMN is that no complex data elements are provided; it implies that just little information about the type of data or document will be available to the performers during the execution of a case. This issue affect also other declarative languages [2] such as DECLARE [14], which is a notation designed to support loosely structured BPs.

To deal with these issues we propose the use of BPFM notation as a language for case representation in CBR. BPFM notation permits defining the BP activities that must or can be performed without including, or including only partially, an execution order of them considering complex constraints and different types of data objects. Furthermore, a BPFM model is a Configurable Process Model since it can encapsulate more than one BP variant. A BP variant can also be easily extracted via the process configuration step. Table 1 summarizes the comparison of BPFM with the other languages.

4 The Approach

4.1 Business Process Feature Model Notation

A Business Process Feature Model is constituted by a tree of related activities [15]. The root identifies the service under analysis as well as the family of the BPs behind of the service itself. Each internal (non-leaf) activity denotes a subprocess that can be further refined, and the external (leaf) activity represents an atomic task. To better specify how to execute tasks, BPFM allows one to type them using the same meaning and graphical representation given by BPMN 2.0. A BPFM model allows for the defining of constraints between activities in two adjacent levels of the tree. Constraints are used to express (i) if child activities can or have to be selected in the configuration to be included in the BP variant, and (ii) if they can or have to be included in each execution path of the BP variant, considering in this way the static and dynamic (run-time) inclusion of the activities. Depending on the type, each constraint has only one father

activity, and it can have one (binary constraints) or more (multiple constraints) child activities. Constraints are described as follows.

- A *Mandatory Constraint* requires that the connected child activity be inserted in each BP variant, and it must also be included in each execution path (Fig. 5A).
- An *Optional Constraint* allows for the connected child activity to be inserted (or not) in each BP variant, and it could be included (or not) in each execution path (Fig. 5B).
- A *Domain Constraint* requires that the connected child activity be inserted in each BP variant, but it could be included (or not) in each execution path (Fig. 5C).
- A *Special Case Constraint* allows for the connected child activity to be inserted (or not) in each BP variant. When it is inserted it has to be included in each execution path (Fig. 5D).
- An *Inclusive Constraint* requires that at least one of the connected child activities be inserted in each BP variant, and at least one of them must be included in each execution path (Fig. 5E).
- A *One Optional Constraint* requires that exactly one of the connected child activities be inserted in each BP variant, and it could be included (or not) in each execution path (Fig. 5F).
- A *One Selection Constraint* requires that exactly one of the connected child activities be inserted in each BP variant, and it has to be included in each execution path (Fig. 5G).
- An *XOR Constraint* requires that all the connected child activities be inserted in each BP variant, and exactly one of them has to be included in each execution path (Fig. 5H).
- An *XOR Selection Constraint* requires that at least one of the connected child activities be inserted in each BP variant, and exactly one of them has to be included in each execution path (Fig. 5I).

Finally, *Include* and *Exclude* relationships between activities are also considered according to the base definition of FM (Fig. 5J and K).

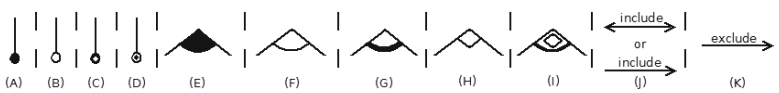

Fig. 5. BPFM constraints

BPFM notation gives also the possibility to model Data Objects since each BP variant could include completely different sets of Data Objects [16]. BPFM manages all types of Data Objects introduced by BPMN 2.0 and uses the same graphical representation. As well as in BPMN 2.0, Data Objects can be connected as inputs and outputs to one or more activities (Fig. 6A). In modelling Data

Objects we also give the possibility to include information concerning the state. Therefore an activity can require or can generate a Data Object in a specific state. A Data Object cannot be in two different states at the same time (Fig. 6B).

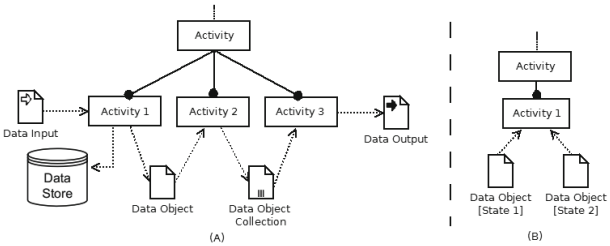

Fig. 6. Data object in BPFM.

4.2 BPFM in Case-Based Reasoning

In traditional CBR terminology, a case consists of a *problem* description that is used for describing a *solution* [11]. Based on Bergmann [11], we extended this CBR terminology in such a way that the solution is denoted as *case content*, which contains not only the solution itself but also information that is useful to find a solution. To describe the case content we are using the term *case characterisation* based on Bergmann [11, p. 50], which enriches the classical problem with additional information, e.g. "derived descriptions or properties that were not present in the problem solving situations from which the experience emerges".

Case Content Containing Process Knowledge. The introduced modelling language Business Process Feature Model (BPFM) provides the expressiveness to tackle the objectives for case content. In addition to the BPFM elements, the following case content elements are used to describe case inter-relationships:

- *ParentTask*: The parent task element is used to express a possible sub-task/case relationship.
- *ChildTask*: The child task element is the inverse of the parent task element.
- *RelatedTask*: This element is used to express that there exist related tasks.
- *ReusedTask*: The reused task element is used to list tasks, which have been (re)used in the adaptation phase of CBR.

Adapting Case Models. The BPFM modelling approach allows for a flexible adaptation of cases. Assume that during execution of the case model it turns out that the case model needs to be adapted. Standard CBR *revises* the case model and *retains* it as a new case. This new case is independent from all the

other cases. The information that it is a variant of an already existing case is lost. Using BPFM this dependency could be made explicit. Instead of storing a new case, the user can adapt the current case model by adding a child/parent task, related task or reused task.

Case Characterisation Describing Process Knowledge. Figure 7 shows a partition of the case characterisation configuration including elements of the process and domain knowledge using the ontology-based, case-based reasoning (OBCBR) approach of Martin et al. [17].

In brief, the reused approach of Martin et al. [17] provides a wide range on similarity functions for *retrieval* and *adaptation*. As shown in Fig. 7, the *similarity configuration* is made in an RDFS ontology using specific concepts that are *attached to the properties*.

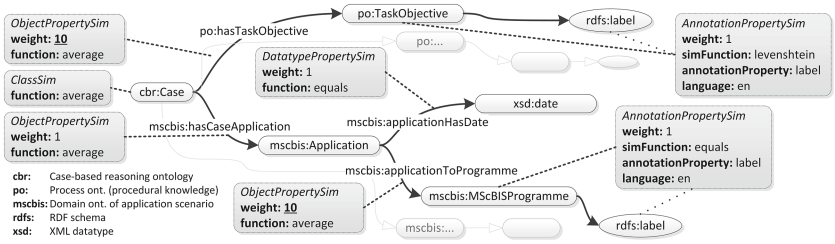

Fig. 7. Exemplary configuration of the case characterisation

The vocabulary for describing the cases is domain specific and therefore different from one application scenario to another. Considering the objectives from the application scenario and the results of the task management system KISSmir introduced by Martin and Brun [18] and Brander et al. [19], we could derive the following basic domain independent vocabulary elements:

- *TaskObjective*: The task objective element describes the goal of the task itself. This is similar to the name and/or description of an BPMN activity.
- *TaskRole*: The task role element is used to describe the role of the involved person of the task. Through the inclusion of an enterprise or domain ontology, it is possible to reuse an existing enterprise specific role/organisational model.
- *TaskUser*: The task user elements is used to indicate the person who described the case.

5 Demonstration and Evaluation

We represented all the cases of the application scenario described in Sect. 2 using BPFM. For the sake of space, in this paper we present only the two cases. In particular, Fig. 8A shows a case in which a student has all the hallmarks to

be eligible but graduates (bachelor degree) in a foreign university that is not stored in the Anabib website (it is related to the cases represented in Sect. 2). Conversely, Fig. 8B shows a case in which a student cannot be eligible since he has just a three-year degree in commerce taken in South Africa. The activities of the example case in Fig. 8A are described as follows. *Analysis Applications* is a mandatory atomic activity in which the application is analysed by the study assistant. *Things to Check* is a mandatory composed activity in which some checks are done in order to approve or reject the application. In this case, it is composed by two sub-activities, which are:

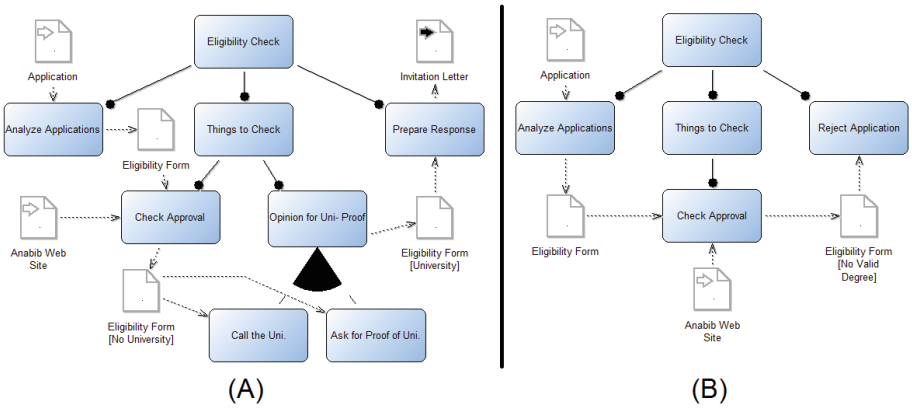

Fig. 8. BPFM model of cases related to the application scenario

- *Check Approval* is a mandatory atomic activity in which the study assistant checks if the final degree university is in the Anabin website and if it is acceptable. In this case, the university is not stored in the Anabib website.
- *Option for Uni. Proof* is a mandatory composed activity in which the study assistant has to have a proof of existence of the Final Degree University. It is composed by two sub-activities connected via an Inclusive Constraint, which means three possible solution variants can be applied. These sub-activities are: *Phone Call to University* in which the study assistant calls the university to be sure that it exists, and *Ask for Proof of University* in which the study assistant asks to the applicant student a proof of the existence of the university.

Prepare Response is a mandatory atomic activity in which the study assistant prepares the invitation letter for the meeting with the eligible student.

The activities of the example case in Fig. 8B are described as follows. *Analysis Applications* is a mandatory atomic activity in which the application is analysed by the study assistant. *Things to Check* is a mandatory composed activity in

which some check are done in order to approve or reject the Application. In this case, it is composed by just one sub-activity:

– *Check Approval* as described above.

Reject Application is a mandatory atomic activity in which the study assistant rejects the application.

Since the described cases are similar, using the BPFM notation can be encapsulated in a single BPFM model including their commonalities and variabilities. To do that, we include all the activities and data objects that the two previous BPFM models include. This is shown in Fig. 9. As the reader can see, the activity *Option for Uni. Proof* is now connected to the father activity via a *Special Case Constraint* since it has to be available even if the university is not in the Anabin website (the case in Fig. 8A). In the figure, the activity *Response* and the related *One Selection Constraint* were added in order to distinguish the two different types of outcomes. In fact, if the candidate is eligible, the activity *Prepare Response* has to be available (the case in Fig. 8A), otherwise the activity *Reject Application* must be available (the case in Fig. 8B).

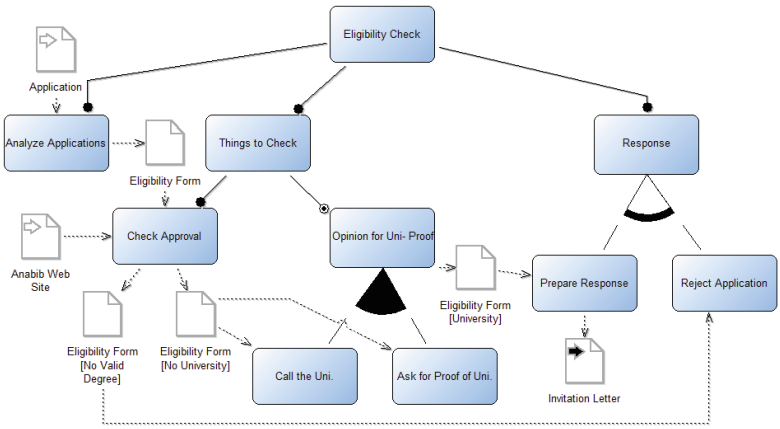

Fig. 9. Cases in Fig. 8 joined in a single BPFM model

The case characterisation contains the elements of the basic vocabulary as introduced in Sect. 4.2 and the domain specific elements of the application scenario ontology. In addition to the basic elements, the following domain specific concepts were used:

– *Person*: The person concept is used to identify the applicant and its *Role*. Apart from that it is linked to the following elements:
  – The *AcademicQualification* is divided into the *Degree* (e.g., Bachelor), *DegreeType* (e.g., Science), *DegreeSubject* (e.g., Information Systems) and the *FinalDegreeUniversity* (e.g., FHNW), where the degree has been awarded.

  - The applicant has to show its *LanguageCompetence* and adequate *ProfessionalExperience*.
  - Finally, the *Nationality* and *Residence* information is captured.
- *Application*: The application concept contains *AdditionalInformation* and the reference to the *Programme* where the applicant applies for.

6 Conclusion and Future Work

CBR case representation is an aspect that needs to be taken into account more in knowledge-intensive BPs. This paper presented an approach to model cases in knowledge-intensive BPs. The approach merges CBR with BPFM notation in order to represent cases. We applied the approach to a concrete case in a public administration scenario in order to show its suitability.

In the future, we will deal with the granularity of the BPFM case models. On one extreme, a manager could make only one bpFM model representing all the cases. In this case the BPFM is adapted, and then CBR is not needed. On the other hand, a manager could represent each case as a separate model. But, in this case, variants are not a need. To find the appropriate granularity we plan to make further evaluation in the application scenario and also test it in a new scenario.

References

1. Swenson, K.D., Palmer, N., Silver, B.: Taming the Unpredictable Real World Adaptive Case Management: Case Studies and Practical Guidance. Future Strategies Inc., New York (2011)
2. Reichert, M., Weber, B.: Enabling Flexibility in Process-Aware Information Systems: Challenges, Methods, Technologies. Springer, Heidelberg (2012)
3. Eberle, H., Unger, T., Leymann, F.: Process fragments. In: Meersman, R., Dillon, T., Herrero, P. (eds.) OTM 2009, Part I. LNCS, vol. 5870, pp. 398–405. Springer, Heidelberg (2009)
4. OMG: case management model and notation (CMMN), Version 1.0., May 2014
5. Aamodt, A., Plaza, E.: Case-based reasoning: foundational issues, methodological variations, and system approaches. AI Commun. **7**(1), 39–59 (1994)
6. Madhusudan, T., Zhao, J., Marshall, B.: A case-based reasoning framework for workflow model management. Data Knowl. Eng. **50**(1), 87–115 (2004)
7. Minor, M., Bergmann, R., Görg, S.: Case-based adaptation of workflows. Inf. Syst. **40**, 142–152 (2014)
8. Müller, G., Bergmann, R.: Generalization of workflows in process-oriented case-based reasoning. In: The 28th International FLAIRS Conference, At Hollywood, Florida, USA, pp. 391–396 (2015)
9. Bergmann, R., Wilke, W.: On the role of abstraction in case-based reasoning. In: Smith, I., Faltings, B.V. (eds.) EWCBR 1996. LNCS, vol. 1168, pp. 28–43. Springer, Heidelberg (1996)
10. Maximini, K., Maximini, R., Bergmann, R.: An investigation of generalized cases. In: Ashley, K.D., Bridge, D.G. (eds.) ICCBR 2003. LNCS, vol. 2689, pp. 261–275. Springer, Heidelberg (2003)

11. Bergmann, R.: Experience Management: Foundations, Development Methodology, and Internet-Based Applications. LNAI, vol. 2432. Springer, Heidelberg (2002)
12. Bergmann, R., Schaaf, M.: Structural case-based reasoning and ontology-based knowledge management: a perfect match? J. Univ. Comput. Sci. **9**(7), 608–626 (2003)
13. Swenson, K.D.: Position: BPMN is incompatible with ACM. In: La Rosa, M., Soffer, P. (eds.) BPM Workshops 2012. LNBIP, vol. 132, pp. 55–58. Springer, Heidelberg (2013)
14. Pesic, M., Schonenberg, H., Van der Aalst, W.M.: Declare: full support for loosely-structured processes. In: 11th IEEE International Enterprise Distributed Object Computing Conference (EDOC 2007), p. 287. IEEE (2007)
15. Cognini, R., Corradini, F., Polini, A., Re, B.: Extending feature models to express variability in business process models. In: Persson, A., Stirna, J. (eds.) CAiSE 2015 Workshops. LNBIP, vol. 215, pp. 245–256. Springer, Heidelberg (2015)
16. Cognini, R., Corradini, F., Polini, A., Re, B.: Using data-object flow relations to derive control flow variants in configurable business processes. In: Business Process Management Workshops - BPM 2014 International Workshops, Eindhoven, The Netherlands, September 7–8, 2014, Revised Papers, pp. 210–221 (2014)
17. Martin, A., Emmenegger, S., Wilke, G.: Integrating an enterprise architecture ontology in a case-based reasoning approach for project knowledge. In: Proceedings of the First International Conference on Enterprise Systems (ES 2013), pp. 1–12. IEEE, November 2013
18. Martin, A., Brun, R.: Agile process execution with KISSmir. In: 5th International Workshop on Semantic Business Process Management collocated with 7th Extended Semantic Web Conference, Heraklion, Greece (2010)
19. Brander, S., Hinkelmann, K., Hu, B., Martin, A., Riss, U.V., Thönssen, B., Witschel, H.F.: Refining process models through the analysis of informal work practice. In: Rinderle-Ma, S., Toumani, F., Wolf, K. (eds.) BPM 2011. LNCS, vol. 6896, pp. 116–131. Springer, Heidelberg (2011)