Incorporating Organizational Aspects into Fragment-based Case Management

Simon Remy

Hasso Plattner Institute, University of Potsdam, Potsdam, Germany
simon.remy@hpi.de

Abstract. Business process management (BPM) enables organizations to model and analyze their business processes, for example, with the help of the Business Process Model and Notation (BPMN). Concerning knowledge-intensive and flexible processes, recent research identified a gap between implemented processes and modeled ones. To close this gap, several approaches have been developed. One of them is fragment-based case management (fCM). However, these approaches share a data-centric view on processes. This work presents an approach to enrich process fragments with organizational aspects. For this purpose, a meta-model that describes the utilization of process participants in fragment-based case management will be introduced. Further, it will be demonstrated how to apply the approach to BPMN models to derive organizational aware process fragments.

Keywords: Business Process Management · Fragment-based Case Management · Roles.

1 Introduction

As an interdisciplinary research field between computer science and business administration, *business process management* (BPM) enables organizations to design, administrate, configure, and analyze their processes [17]. Since the outcome of most business processes is the result of the execution of subsequent activities, BPM provides methods to analyze and to understand the relationships between them [17]. One way to represent these relations and interactions are process models using the *Business Process Model and Notation* (BPMN) standard. Among others, BPMN provides basic elements to model activities, events, and control-flow. Further, BPMN aims to close the gap between process modeling and implementation [14]. Because of this, process models can also be executed using process engines.

Lately, BPM has been applied in many different enterprises and industries. However, it became clear that there exists a gap between some real-life business processes and initially modeled ones. Many processes require a certain amount of flexibility and are limited by traditional workflow management systems [3]. Especially knowledge-intensive processes are affected by this, like treatment processes in healthcare. Those processes are rather unstructured compared to e.g.,

J. Manner, S. Haarmann, S. Kolb, O. Kopp (Eds.): 12th ZEUS Workshop, ZEUS 2020, Potsdam, Germany, 20-21 February 2020, published at http://ceur-ws.org/Vol-2575

a production process [6]. To better support knowledge-intensive processes, a new paradigm was introduced, namely case handling or also called case management [3, 11]. Based on this, other concepts like artifact centric models, *Guard Stage Milestone* models (GSM), *Adaptive Case Management* (ACM), and *Production Case Management* (PCM) were developed [5, 10, 12, 13].

These approaches have in common that they especially focus on the data perspective. The states of data-objects indicate the state of a case and enable knowledge-workers to make decisions about future steps in a case. However, none of the approaches explicitly incorporate organizational perspectives, like roles.

In this paper, we will present a meta-model for process fragments, which can be used in *fragment-based Case Management* (fCM), a specification of PCM. Besides data objects, the model considers the organizational perspective of business processes. Since knowledge-workers play an essential role in such processes, we aim to provide a way to explicitly include them in the modeling process as well as the relations between them. Further, we will demonstrate how a standard BPMN model can be transformed into role-specific fragments.

The remainder of this paper is organized as follows. In section 2, we introduce a running example, followed by related work in section 3. Section 4 presents a meta-model to formally describe the usage of roles in process fragments and a demonstration of deriving fragments from BPMN models based on it. Results, limitations, and future work are discussed in section 5.

2 Running Example

Figure 1 depicts a sample BPMN process model. The model shows a simplified treatment process of patients visiting the cardiology ward of a hospital. While the patient is only modeled implicit via the activity labels, the model consists of three lanes: nurse, physician, and both. While the BPMN standard does not specify the usage of lanes [14], they are commonly used to assign activities to a specific resource, in this example, roles.

Whenever a new patient enters the ward, a new process instance will be instantiated. First, a nurse admits the patient, collects her medical history and updates the patient's record. Next, a blood sample is drawn from the patient. This can either be done by the nurse or the physician, depending on who is available. After that, both have to examine the patient together, followed by an activity to prescribe a treatment by the physician. Before the process ends, the nurse releases the patient and updates the patient's record concurrently.

While executing activities, data objects will be read and written. Changes to them are indicated by changes in the data objects' state, e.g., after the patient has been admitted the state of the *Patient* data object changes from *init* to *admitted*. The state-space of each data object is defined by its lifecycle, which is not part of the process model (see [11] for details). While data objects are bound to specific process instances, they do not only persist information during its lifetime but also define *InputSets* and *OutputSets* of activities. Those sets can be seen as preconditions and postconditions of the respective activities. In other

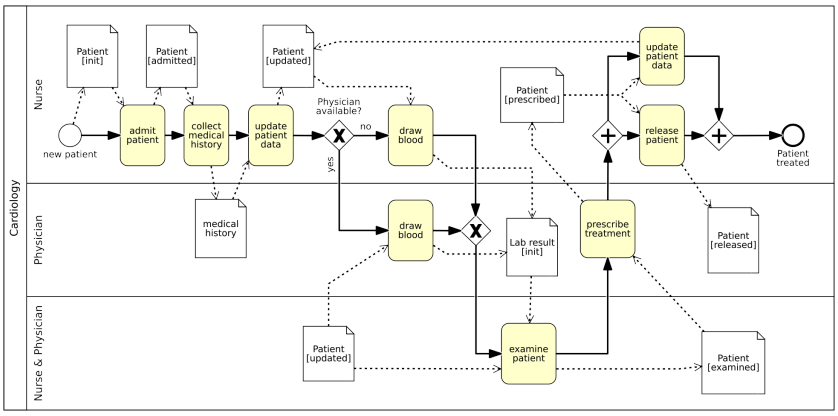

Fig. 1. Sample process that depicts the treatment process of patients in a cardiology department of a hospital. Indicated by the lanes, participants of two roles are involved. However, one activity requires participants of both roles to be executed.

words, an activity can be control-flow enabled, but not data-flow enabled. This is the case if not all data objects in its *InputSet* are in the required state [11].

Even if the presented example depicts a simple process, and therefore the use of fCM is limited, our findings can be applied to more complex models. We will use the example to illustrate how to derive process fragments for business processes based on roles according to the meta-model, described in section 4. For the remainder, we lift the assumption, that only explicitly modeled roles, like lanes in process models, will be considered.

3 Related Work

As described in the previous section, traditional process management approaches are limited in their capability to support knowledge-intensive processes. To overcome these limitations, several approaches have been proposed in the past.

As one of the first approaches case handling has been developed [3]. This approach not only focuses on the order of activities but mainly on data-objects. Since then, several other ideas were introduced. Business artifact centric approaches focus on the life-cycles of business objects to describe the context and structure of processes [13]. GSM follows another data-driven approach using guards, stages, and milestones to structure processes [5]. With the *Case Management Model and Notation* (CMMN), a new modeling standard has been introduced to support case management [15]. Process fragments were introduced by PCM to model small parts of a process to maintain a certain degree of the structure without limiting its flexibility too much [4, 11, 12]. Lastly, ACM aims to enable knowledge-worker to adopt processes at run-time[10].

In their literature review, Hauder et al. identify several research questions for ACM[8], where some can also be applied to the previously presented approaches. According to their work, successful case management requires collaborations between different roles and clear rules for interactions [9, 16]. Further, the authors understand roles as a powerful tool to restrict data access and to ensure data privacy [8]. One approach to model communication and interactions in business processes is the *Design & Engineering Methodology for Organizations* (DEMO) [7]. Other approaches are data-driven and based on historical process data. They aim to model social networks from recorded process data. Those networks visualize interactions between process participants and roles [1, 2].

4 Organizational Perspective on Process Fragments

To provide a formal basis, to discuss the usage of roles in fCM, we introduce a meta-model for process fragments in the following. Further, we show an application of the approach to derive fragments from BPMN process models.

4.1 Meta-Model

The meta-model, depicted in Figure 2, is based on the definition of process models, presented in [17]. *Process Fragment* is the central class of the meta-model. A fragment consists of *Edges*, *Nodes*, *Data Objects*, and *Roles*, where each edge connects exactly two nodes. However, nodes can be connected to multiple edges. Therefore edges express the control-flow relationship between nodes.

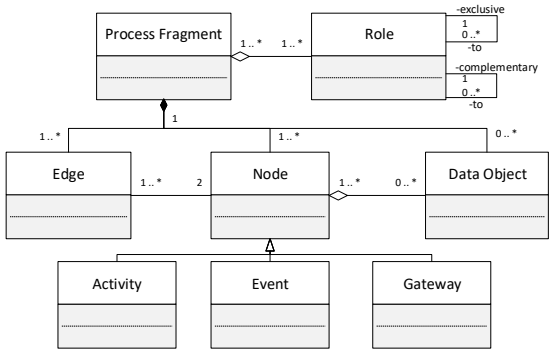

Fig. 2. Meta-model for process fragments

Further, a *Node* can be an *Activity*, an *Event*, or a *Gateway*. In difference to the process meta-model, a fragment can consist of a single activity. Since fragments must not have empty start events, a fragment is considered as enabled

if all preconditions of the first activity are fulfilled, in other words, as soon as it is dataflow-enabled [11].

Data Objects play an essential role in fCM and are therefore included in the meta-model. Multiple data objects can be associated with a set of nodes. However, not every node has to be associated with a data object. Thus, a fragment does not require data objects at all. While this seems to contradict the purpose of fCM, it allows the process modeler, to design process fragments, which are enabled at any time during the execution of an instance, like escalating the case to a higher level, e.g. a manager.

Lastly, each process fragment is associated with at least one *Role*. Only participants, who belong to the associated roles are able to execute instances of the fragment. However, multiple roles can be associated with the same fragment. In this case, roles can be either mutual exclusive to each other or complementary. In the first case, only participants of one role can be involved during run-time, while in the second case, participants of all roles have to participate in its execution.

4.2 Application

The first step to derive fragments is to split the process model horizontally based on each lane. As a result, activities in one lane will be disconnected whenever a handover between two lanes takes place. Those disconnected activity sequences are *fragment candidates*. However, fragments have to satisfy two conditions (*i*) be free of open *(X)OR-Joins/Splits*, and (*ii*) no shared activities with any other fragment. A join or split will be considered as open if its respective counterpart is not part of the same fragment.

In order to satisfy the first condition, the control-flow of a candidate will be cut before or after one of the respective gateways. In our running example, this is the case for the activity *draw blood* in the upper lane.This activity also violates the second condition, as it is part of an other fragment, that belongs to the *Physician*. After all fragment candidates satisfy the first condition, they have to be checked for shared activities. Depending on the control-flow structure, shared activities need to be handled differently. In the simplest case, a shared activity A is part of a sequence, without any exclusive and parallel gateways. In this case, the sequence is split into two or three fragments, depending on the position of A. If A belongs to a branch after an AND-Split, three scenarios, depending on the total number of branches and on the number of branches A belongs to, are possible. Given two branches, where A only belongs to one of them, a new fragment is created for the branch, that contains activity A. The other branch will be preserved as a sequence of the original fragment. If more branches exist and A still belongs to only one branch, only the effected branch needs to be removed, and a new fragment will be created. If activity A is part of multiple branches, a new fragment for each of them will be created. Depending on the number of not effected branches, the split can be preserved or not. Independent of the applicable scenario, all newly created fragments might need to be split up further in order to satisfy the second condition. Also, in order to keep the

semantic of the AND-join, its conditions need to be reflected in the data-flow. Regarding the running example, the activity *update patient data* occurs in two fragments, and the first described scenario applies. Since the control-flow is only split into two concurrent branches, both will be transformed into a fragment.

Further, if activity A is part of an XOR/OR-Split, the following steps need to be performed. All branches that contain A will be removed, and for each, a new fragment will be created. If at least one branch does not contain A, a new edge from the split node, to the join node will be inserted. Again, all newly created fragments might need to be split up further in order to satisfy the second condition. After all fragments have been derived and comply with both conditions, the corresponding roles will be associated with the fragments. Since multiple roles can be associated with one fragment, logical expressions are used to express the relations between them. Roles, which are mutually exclusive to each other, are joined by the $\vee$ operator, while complimentary roles are connected using the $\wedge$ expression. Fragments that share the same set of roles, including the same relations, are grouped as a collection.

Figure 3 depicts six fragments that are derived from the process model presented in section 1. The graphical presentation of the fragments is loosely based on the BPMN standard. Single fragments are modeled using core BPMN elements, like activities, gateways, and events. The fragments are grouped based on their associated role, which is located in the upper left corner. If one collection is associated with multiple roles, all are listed, including their relation operator.

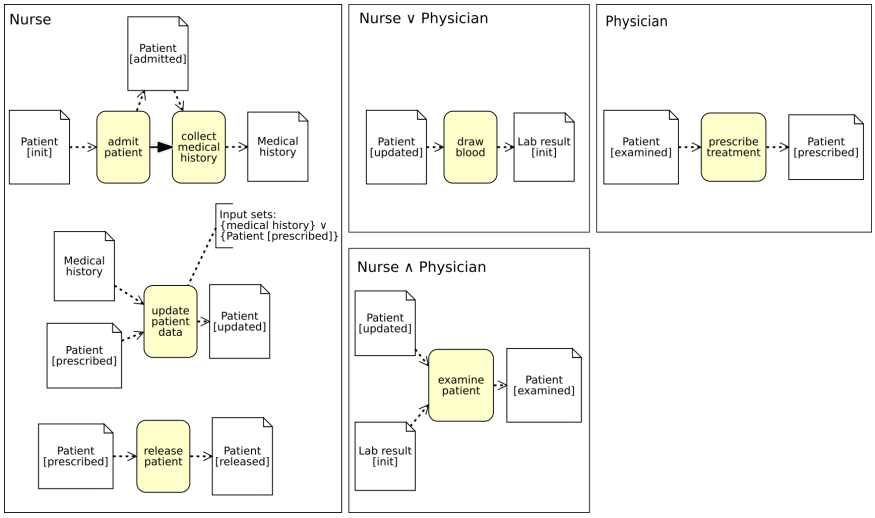

Fig. 3. Six fragments, derived from the BPMN model presented in section 1. The fragments are organized in collections according to their associated roles and their relations.

5 Discussion and Future Work

In this paper, we introduced a novel approach on how to integrate roles into process fragments to support fCM at design time. We presented a meta-model to define the components and provided a brief example of how to derive process fragments based on an existing process model.

Following this approach, adding new roles to an existing business process can easily be done by introducing a new collection of fragments or by adding role identifiers to existing ones, instead of editing a whole process model. Since fragments are organized in role-specific collections, it is also easier to remove them from the process. The compact representation provides a good overview in which parts of the process a role is involved and, therefore, where deadlocks or other inconsistencies may occur. This also goes along with better privacy protection, since fragments clearly show interactions between roles and data objects. In difference to BPMN, this approach provides a clear semantic of the relationships between roles. While in BPMN the behavior of, shared lanes or grouped activities, is not ultimately defined [14].

Like in BPMN, our approach does not specify any resource allocation method. Hence it would be possible that different participants of the same role are involved in different activities of the same fragment instance. Further, deriving process fragments from BPMN models can be challenging, regarding the semantic of the original control-flow. While the concurrent execution of multiple activities can be easily modeled using the respective BPMN elements, this is more complex with concurrent process fragments. The existing join condition has to be projected on the data-flow, using dedicated input sets of the subsequent fragments.

In this paper, we investigated a model-driven perspective to derive process fragments. In future work, we will explore a data-driven method based on event logs. Further, we will evaluate our approach based on real-life event logs concerning usability and interpretability.

References

1. van der Aalst, W.: Process Mining - Data Science in Action. Springer, Berlin, Heidelberg (2016)
2. van der Aalst, W., Reijers, H., Song, M.: Discovering social networks from event logs. Computer Supported Cooperative Work (CSCW) 14(6), 549–593 (2005)
3. van der Aalst, W., Weske, M., Grünbauer, D.: Case handling: a new paradigm for business process support. Data & Knowledge Engineering 53(2), 129–162 (2005)
4. Beck, H., Hewelt, M., Pufahl, L.: Extending fragment-based case management with state variables. In: Dumas, M., Fantinato, M. (eds.) Business Process Management Workshops, vol. 281, pp. 227–238. Springer International Publishing, Cham (2017)
5. Cohn, D., Hull, R.: Business artifacts: A data-centric approach to modeling business operations and processes. IEEE Data Engineering 32(3), 59 (2009)
6. Di Ciccio, C., Marrella, A., Russo, A.: Knowledge-intensive processes: Characteristics, requirements and analysis of contemporary approaches. Journal on Data Semantics 4(1), 29–57 (Mar 2015)

7. Dietz, J.L.G.: Designing technical systems as social systems. in: The language action perspective on communication modelling. In: Proceedings of the 8th International Working Conference on the Language-Action Perspective on Communication Modelling (LAP 2003), 2003 (2003)
8. Hauder, M., Pigat, S., Matthes, F.: Research challenges in adaptive case management: A literature review. In: 2014 IEEE 18th International Enterprise Distributed Object Computing Conference Workshops and Demonstrations. pp. 98–107. IEEE (2014)
9. Heil, S., Wild, S., Gaedke, M.: Collaborative adaptive case management with linked data. In: Proceedings of the 23rd International Conference on World Wide Web - WWW '14 Companion. pp. 99–102. ACM Press (2014)
10. Herrmann, C., Kurz, M.: Adaptive case management: Supporting knowledge intensive processes with IT systems. In: Schmidt, W. (ed.) S-BPM ONE - Learning by Doing - Doing by Learning, vol. 213, pp. 80–97. Springer Berlin Heidelberg (2011)
11. Hewelt, M., Weske, M.: A hybrid approach for flexible case modeling and execution. In: La Rosa, M., Loos, P., Pastor, O. (eds.) Business Process Management Forum, vol. 260, pp. 38–54. Springer International Publishing (2016)
12. Meyer, A., Herzberg, N., Puhlmann, F., Weske, M.: Implementation framework for production case management: Modeling and execution. In: 2014 IEEE 18th International Enterprise Distributed Object Computing Conference. pp. 190–199. IEEE (2014)
13. Nigam, A., Caswell, N.S.: Business artifacts: An approach to operational specification. IBM Systems Journal 42(3), 428–445 (2003)
14. Object Management Group: Business process model and notation (bpmn) (2011), `https://www.omg.org/spec/BPMN/2.0/PDF`
15. Object Management Group: Case management model and notation (2016), `https://www.omg.org/spec/CMMN/1.1/PDF`
16. Reinhardt, W., Schmidt, B., Sloep, P., Drachsler, H.: Knowledge worker roles and actions—results of two empirical studies. Knowledge and Process Management 18(3), 150–174 (2011)
17. Weske, M.: Business Process Management - Concepts, Languages, Architectures. Springer-Verlag Berlin Heidelberg, 3 edn. (2019)