Using Patterns for Communicating About Flexible Processes

Ralf Laue[1] and Kathrin Kirchner[2]

[1] ✉University of Applied Sciences of Zwickau, Department of Computer Science
Dr.-Friedrichs-Ring 2a, 08056 Zwickau, Germany
Ralf.Laue@fh-zwickau.de
[2] Berlin School of Economics and Law, Alt-Friedrichsfelde 60, 10315 Berlin, Germany
kathrin.kirchner@hwr-berlin.de

Abstract. We describe the experiences from a project in the medical domain where processes were modeled in modeling sessions in close cooperation with physicians. In this project, we experienced difficulties in modeling flexibilities within processes. Flexible and knowledge-intensive processes do not follow a fixed sequence of steps, but rely on knowledge and experience of the medical staff. During the process execution process, the actors can decide for additional steps, change the execution order or skip a task. In standard business process modeling languages, it is not clear how to model such flexible situations.
We observed, however, that many flexible situations can be described by recurring patterns. We argue that those patterns can be used as building blocks for communication among the stakeholders. The advantage is that those building blocks are close to the vocabulary that is used when domain experts describe a process. The patterns can support the process modeler to recognize flexible situations in processes. In addition, a pattern catalog can recommend a way to model such situations in a suitable way in a standardized modeling language.

1 Introduction

In the research project Process Intelligence in Healthcare (PIGE, see www.pige-projekt.de) four different complex medical processes were modeled in several levels of detail (liver transplantation, living donor liver transplantation, hepatocellular carcinoma and colon rectal carcinoma). The project ran from 2010 to 2013 at the University Hospital Jena, Germany. A modeling session involved typically 2-3 physicians with experiences in the required process as well as 1-2 process modeling experts. Depending on the process and the required level of detail, further participants, e.g., nurses or administrative personal, were invited to the session or interviewed. From our experience with process modeling projects [2] the PIGE team (including the second author of this paper) learned that a close collaboration with process participants is useful in modeling. For this purpose, there is a need for a notation that describes process models in a way that is understandable also for non-modeling experts. Such a notation helps to elicit a process together with all stakeholders and to achieve a high model quality [4].

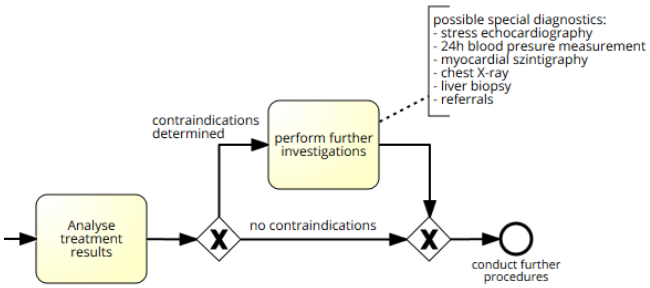


Fig. 1. Example of modeled BPMN process with annotation

In this paper, we build upon our practical experiences in modeling processes in the medical domain. We introduce patterns for the early stage of business process modeling. The aim of our patterns is to speak about variable parts of a process (model) using a vocabulary that is close to domain experts' understanding.

2 Lessons Learned from Practice

A medical treatment process is a typical scenario for a flexible process. Physicians decide dynamically which treatment is needed depending on the health status of the patient. A certain order of steps is known advance, e.g., that some preliminary examinations have to be carried out before an operation. The sequence of these examinations might change due to availability of medical devices and physicians at a certain moment.

In our modeling projects, we used the language BPMN for documenting medical processes. The reason for this decision was that BPMN is the current industry standard for business process modeling. If a process was (partly) flexible, the PIGE team experienced the following ways to model variability:

1. Only the typical process was modeled (covering only the control flow used in $\approx 80\%$ of all cases) or the process was modeled at a low level of detail. This might be all right for domain experts who are already involved into the process. However, it is not easy for new employees to understand the process and possible options in full detail because some information is missing.
2. Text annotations were used to write down possible additional steps or alternatives in natural language (see Fig. 1)
3. All possible different pathways were modeled as shown in Fig. 2(a). Here, task A is an optional step, while B and C can be carried out in an arbitrary order (but not in parallel). In case of a lot of alternative pathways, the number of branches becomes very high and thus, make it confusing to understand the process. Therefore, this option was used very rarely in our modeling sessions.

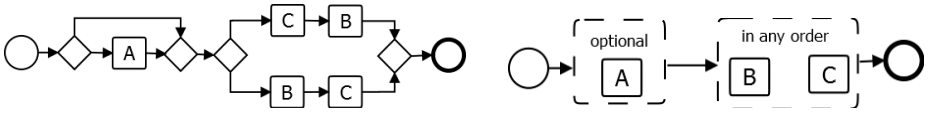
One problem in modeling flexible processes is the fact that there is no 1:1 correspondence between constructs of modeling languages such as BPMN and

the vocabulary that is frequently used by domain experts. For example, in our project the physicians often use phrases such as "if executed successfully" or "if executed successfully in time". This applies to activities which can have an outcome that is interpreted as "positive" or "negative". An example of a positive outcome would be that some antibody is found in a blood-test. Note that this notion of "executed successfully" is different from the fact that the blood-test has been completed properly (without exception). Of course, BPMN allows to model the different outcomes of an activity by using an XOR split gateway followed by two labeled sequence flow arcs. However, using such constructs in a modeling workshop quickly leads to relatively large models for situations that can be expressed very shortly as e.g. "If needed, activity A is executed, afterward B and C are performed in arbitrary order". A possible notation that can replace Fig. 2(a) is shown in Fig. 2(b). This model would be easily understandable by domain experts, but is not in line with syntactically correct BPMN models that are needed by workflow engines.

3 Patterns for the Early Stage of Business Process Modeling

The concept of patterns means to document known solutions to recurring problems which are known to work well in a given context. Before the concept of patterns was introduced in the field of business process modeling, semantic patterns have already been used for for creating unambiguous textual use case specifications. [5] discusses four patterns "Sequence", "Constraint", "Concurrency" and "Repetition" which can be regarded as patterns occurring between activities in a process. In the area of business process modeling, the the most popular work on patterns are the workflow patterns [9]. These patterns have been widely used by practitioners, vendors and academics. They are helpful for selecting a workflow system that corresponds to the needs of an organization and for evaluating the expressiveness of modeling languages. Workflow patterns provide an answer to the question which elements a workflow engine or modeling language should have in order to be useful in a given context. While this is a very important question, it is a rather technically-oriented point of view.

We brainstormed about typical situations in a process that are described by the stakeholders verbally *as one fact*. This means that the object of investigation was which building blocks are frequently used when people communicate about a process. An example for such a building block is when stakeholders



(a) Process in BPMN (b) Same process

Fig. 2. Process Example in BPMN and Using Pattern Blocks

say that activities are "performed at the same time / in parallel", etc. This is clearly different from the technically-oriented terminology of the workflow patterns which describe this situation as a combination of the patterns "Parallel Split" and "Synchronization". The human-oriented perspective of our patterns also means that it is important to use pattern names that can be understood and used intuitively by domain experts.

Based on our modeling experience, we identified a set of linguistic patterns that are frequently used to express common situations (and in particular variability) in processes. Using those patterns as "process building blocks" in a workshop with practitioners avoids large graphical models that might be difficult to read for some stakeholders. On the other hand, we do not lose preciseness, because each pattern can be transformed into formal modeling languages.

In addition to the most basic patterns (such as PARALLEL EXECUTION, see [4]), the following patterns have been identified: OPTIONAL EXECUTION and the very similar pattern EXECUTE ONLY IF..., REPEAT ACTIVITY UNTIL SUCCESS (with a variant where the number of repetitions is limited by some number), ACTIVITY MUST SUCCEED IN A GIVEN TIME, PERFORM IN ANY ORDER, TRY ALTERNATIVES UNTIL SUCCESS (with two variants "ordered" and "in any order"), PASS ALL TESTS (with three variants which will be discussed below) and ADDITIONAL ACTIVITY NECESSARY (with several variants describing the time when this additional activity should be executed). Using those patterns allows using terse models such as Fig. 2(b) in modeling workshops.

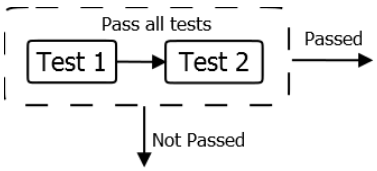
Due to space limitation, we can only give examples of three patterns here. We selected patterns that relate to a series of tests. Such situations occur quite often in medical processes, but also in other domains (e.g., approval procedures).

Our pattern catalog is organized by pattern templates. Each template contains the pattern name, the visual representation, a description of the situation where it is used and an example where this pattern occurred in a real process. For the purpose of this paper, the examples have been taken from the living liver donation process modeled in the PIGE project. Here, a healthy person who is closely related to the patient, donates a part of liver to that patient. Before the transplantation process, the donor undergoes an evaluation process to make sure that she/he is healthy enough and aware of procedures and consequences. Next, the pattern template contains a paragraph "Related Patterns" with references to sources where variants of this pattern have been discussed.

In addition to the extract published in this paper, the full pattern template also contains guidance how this pattern can be expressed in BPMN. This part can contain a discussion about modeling variants. In addition, some of the patterns contain a paragraph "Considerations for Optimization" which discusses challenges for process improvement in the given situation. It suggests questions and recommendations that can be worth discussing in a process workshop.

In the following subsections, we present the three variants of PASS ALL TESTS. Please note that every time when we use the term "activity", it can refer either to a single (atomic) task or to a sequence of several tasks. Without limiting generality, in our diagrams we restrict ourselves to two tests.

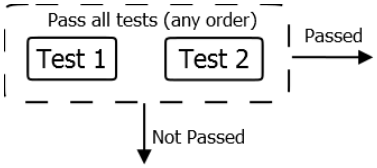
3.1 Pass All Tests (Ordered)



Problem: Some object has to undergo a series of tests. The order in which the tests should be executed is known beforehand. The process can proceed by executing the next activities only if all tests have been passed successfully. As soon as the first test fails, another path in the process flow is taken, and no further tests are necessary.
Example: A prospective liver donor needs to undergo a blood test to investigate the compatibility to the patient's blood. In case the blood is not compatible, the donor is not suitable, and no further tests are necessary. Otherwise, an investigation by a psychologist is planned. Here, too, the donor can be found to be not suitable if there are any contraindications.

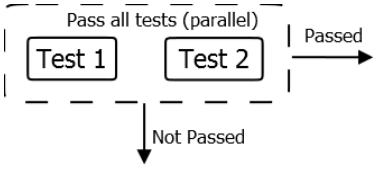
Related Patterns: The iterative approval of an object (such as a document) by multiple organizational roles is discussed as Iterative Approval pattern in [8]. In [7], the iterative approval of a document by multiple organizations has been discussed. However, the solution from [7] works only if all approval tasks and all roles who are responsible for the approvals can be regarded as being essential the same task/role.

3.2 Pass All Tests (Any Order)



Problem: Some object has to undergo a series of tests. The order in which the tests should be executed has to be decided at run time. The process can proceed by executing the next activities only if all tests have been passed successfully. As soon as the first test fails, another path in the process flow is taken, and no further tests are necessary.
Example: After the blood compatibility and psychological testing, a number of further investigations and tests are necessary. Their sequence can be decided by the medical personal, depending on the free time of specialists and the availability of CT and MRI scan devices, etc. If a mayor contraindication is found, the donor is not suitable, and no further tests are necessary. A transplant operation can only be planned if the whole evaluation procedure is completed successfully.

3.3 Pass All Tests (Parallel)



Alternative names: All-or-nothing [1]
Problem: Some object has to undergo a series of tests. These tests can be done in parallel (to be executed by different actors).
Example: While the prospective liver donor is undergoing medical investigations, medical consultations (meetings among physicians from different disciplines) can take place to discuss the case and discuss possible contraindications.

4 Discussion and Outlook

Our proposed patterns can be directly used in a modeling session and support the modeling of flexible process parts. While discussing the process, domain experts explain their activities and use phrases for expressing flexibility, e.g.,

- *The number of investigations on this list can usually be carried out in any order*: This corresponds to our pattern Pass All Tests (Any Order).
- *The first important step is ... the next step is ... If it fails, the whole procedure is canceled*: This calls for our pattern Pass All Test (Ordered).
- *While the patient is still ... other investigations...*: This leads to our pattern Pass All Tests (Parallel).

The application of our flexible patterns as building blocks can help to transfer the process description of a domain expert to a valid BPMN model. We will illustrate this with an example from the living liver donor evaluation process. In one modeling session, the evaluation process was discussed together with the physicians. During the evaluation, several investigations have to me made to decide whether the person can donate a part of her/his liver. The physicians explained the rough process of donor evaluation: After drawing blood to test whether it is compatible with the receiving patient, the person has to pass a psychological test. Here, the motives for the donation are discussed and it has to be evaluated whether the donor can face consequences and risks. Afterward, physical investigations and some other investigations are undertaken. If at any point of these investigations a contraindication occurs, the evaluation process ends, because the person cannot donate a part of her/his liver. Using our patterns, this high level description leads to our Pass All Tests (Ordered) pattern. This information leads to the overall picture shown in Fig. 3. The + sign at the bottom center of some tasks shows that the details are elaborated in a more specific diagram.

Next, for the *Physical examination* subprocess, the physicians explained that a number of investigations have to be done. For some of them, a special device is needed, that might be busy at the moment. Furthermore, a medical doctor from another department might be needed. Depending on the availability of those resources, an investigation might be postponed and another one can be done earlier to avoid waiting time. Needed investigations are written on on a paper evaluation list. If one investigation is planned with a concrete time, the date (and later also the result) is written down on the list. The nurses and physicians can see at a glance which investigations are still open, and whether a contraindication occurred. The possible donor has to undergo a number of investigations in any order, thus our pattern Pass All Tests (Any Order) can be used. All investigation results need to be positive in a way that the patient can be considered as a donor.

In the PIGE project, it turned out to be too confusing to model the execution of six tests in arbitrary order using only BPMN constructs. Therefore, a BPMN model with text annotations has been suggested (Fig. 4(a)). While the involved physicians understood the model thanks to their domain knowledge, the disadvantage is that such models still tend to be large and confusing. Furthermore, no transformation to a formal language would be possible. In contrast to Fig. 4(a), Fig. 4(b) is easy to understand and can additionally also be equipped with a formal syntax by transforming it to a formally defined language such as BPMN.

One of the patterns discussed in Sect. 3 (Pass All Tests (Parallel)) was discussed by Simon [6]. With respect to a model depicting this pattern he argued that graphical business process modeling languages "either ... leave room for interpretation, of if they are precise, they reach such a degree of complexity that they are difficult to read and understood by somebody who has not developed the model in his/her own". This statement illustrates the advantage of our approach: Instead of complex models, we use rather simple building blocks for a discussion among the stakeholders while still providing the possibility to transform the model into a formal modeling language. An approach to transform such building blocks automatically to BPMN has been presented in a previous paper [4].

The usefulness of "building blocks" that are close to the usual vocabulary of the domain experts has been confirmed in nearly 20 interviews with process experts in various domains (see [3] for an overview). It follows the basic idea

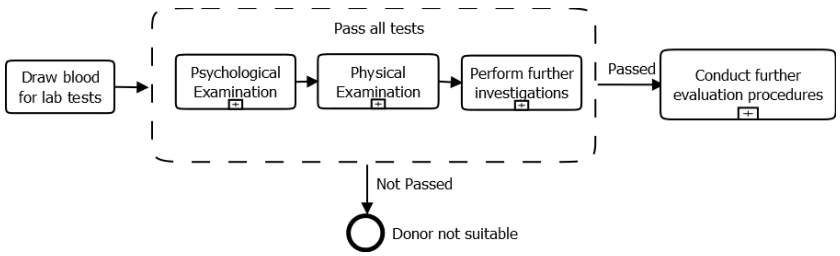
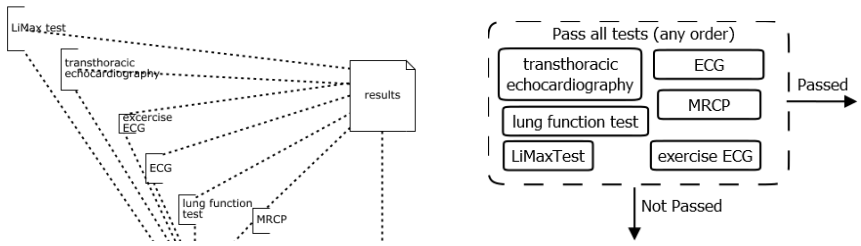


Fig. 3. Living Liver Donor Evaluation: Overall Picture



(a) BPMN (Detail from a Larger Model) (b) Pattern Block

Fig. 4. Subprocess: Physical Investigations of Living Liver Donor

that the modeling language should be adapted to the language that a domain expert uses in a modeling workshop - and not the other way round.

While the patterns we have already identified can be used to describe a considerable amount of situations, we do not claim that our pattern catalog is complete. Instead, we are sure that in order to cover more flexible situations in business processes, further patterns have to be identified in future. Furthermore, the usefulness of our patterns needs to be evaluated in more modeling sessions.

References

1. Gruhn, V., Laue, R.: Good and bad excuses for unstructured business process models. In: European Conference on Pattern Languages of Programs (2007)
2. Kirchner, K., Malessa, C., Scheuerlein, H., Settmacher, U.: Experience from collaborative modeling of clinical pathways. In: Modellierung im Gesundheitswesen: Tagungsband des Workshops im Rahmen der Modellierung 2014. pp. 13–24 (2014)
3. Kirchner, K., Nešković, S.: Using CUTA4BPM to support participative development of expert-driven processes. In: 2nd International Conference on Information Society Technology (2012)
4. Kirchner, K., Nešković, S., Stojimirović, D.: From user-understandable to technical process model: A model-driven approach using CUTA4BPM. In: 3nd International Conference on Internet Society Technology and Management (2013)
5. Rolland, C., Achour, C.B.: Guiding the construction of textual use case specifications. Data Knowl. Eng. 25(1-2), 125–160 (1998)
6. Simon, C.: Incremental development of business process models. In: Workshop Enterprise Modelling and Information Systems Architectures. pp. 222–235 (2005)
7. Thom, L.H., Lazarte, I.M., Iochpe, C., Priego-Roche, L., Verdier, C., Chiotti, O., Villarreal, P.D.: On the capabilities of BPMN for workflow activity patterns representation. In: BPMN Workshop. LNBIP, vol. 95, pp. 172–177. Springer (2011)
8. Thom, L.H., Reichert, M., Iochpe, C.: Activity patterns in process-aware information systems: basic concepts and empirical evidence. Int. Journal of Business Process Integration and Management 4(2), 93–110 (2009)
9. van der Aalst, W.M.P., ter Hofstede, A.H.M., Kiepuszewski, B., Barros, A.: Workflow patterns. Distributed and Parallel Databases 14(3), 5–51 (2003)