

Recommendations for Medical Treatment Processes: The PIGS Approach

Marcin Hewelt^(✉), Aaron Kunde, Mathias Weske, and Christoph Meinel

Hasso Plattner Institute at the University of Potsdam, Potsdam, Germany
{marcin.hewelt,aaron.kunde,mathias.weske,christoph.meinel}@hpi.de

Abstract. Medical treatment processes in hospitals are complex interactions of various actors, in the course of which treatment decisions have to be made. Clinical practice guidelines and clinical pathways provide guidance for practitioners for certain diseases. Especially for multi-morbid patients several guidelines and pathways might apply. Current process-oriented IT support in hospitals, however, does not consider multiple models for treatment recommendations.

In this contribution we define Treatment Cases (TCs) and give an operational semantics for their evolution based on Business Process Model and Notation (BPMN) models, which we use to model guidelines, pathways and standard operating procedures. This work is part of the Process Information and Guidance System (PIGS) approach, which aims at providing an overview of treatment history and give recommendations based on the current treatment state and multiple process models.

1 Introduction

The treatment of a patient in the hospital is a complex interaction of medical actors in different departments, involving both organizational and medical activities, e.g. scheduling of radiology appointments and prescription of drugs. Business Process Management (BPM) has been devised to deal with exactly this kind of interaction [1]. Decisions about the next treatment steps have to be made, based on the current patient condition and the treatment history. To support these decisions, Clinical Practice Guidelines (CPGs) were introduced as sets of evidence-based best practices for diagnosis and treatment of certain diseases, published by medical societies and updated regularly [2]. Since CPGs consist of free form text with some flow charts, their formalization in so called Computer Interpretable Guidelines (CIGs) has been thoroughly researched, leading to a variety of different formalisms, e.g. PROforma [3], GLIF3 [4] or GUIDE [5]. Based on these CIGs clinical decision support tools have been proposed to guide practitioners through diagnosis and treatment of a disease, e.g. *Tallis*¹. Because CPGs only cover medical aspects of a treatment, they need to be adapted to concrete hospital settings and enriched with organizational aspects, resulting in Clinical Pathways (CPs), which depict the complete path of a patient through a

¹ <http://www.cossac.org/tallis>, built on PROforma.

specific institution [6]. Only then can the path be supported by IT systems. For certain, often-occurring or critical activities hospitals implement Standard Operating Procedures (SOPs), which in detail describe the order of steps to take, e.g. for the reanimation of a patient. SOPs can be seen as the building blocks of CPs.

Hence, in reality several process models govern the treatment of a patient and inform treatment decisions. A special case are multi-morbid patients, whose treatment relies on multiple CPGs, which can contradict or prescribe redundant medical procedures. We approach this multi-model problem by extending the usual semantics of BPMN, which are defined in [7], to allow the integrated execution of multiple process models at the same time. To this end we formally define Treatment Cases (TCs) that were introduced in [8] to capture the complete treatment history of a patient, and propose an operational semantics based on partially ordered events. This allows us to recommend treatment steps based on the current treatment state taking into account multiple process models, which we assume to be modeled using BPMN. Medical actors interact with the TC, e.g. by adding recommended steps or skipping respectively executing certain activities, thus progressing the state of the treatment. This contribution details central aspects of the PIGS approach introduced in [8], which aims at process-oriented IT support for organizational and treatment processes in the hospital. We exemplify our approach with an usecase.

2 Related Work

Milla-Millán et al. [9] and Sánchez-Garzón et al. [10] use a multi-agent planning approach to deal with the generation of a personalized treatment plan for co-morbid patients. The CPGs are modeled in HTN Planning Domain Language (HPDL) which is an extension of Planning Domain Definition Language (PDDL) for Hierarchical Task Networks (HTNs). In contrast, we want to use BPMN in order to include business processes, which have already been modeled.

Alexandrou et al. [11] describe the challenge of personalizing CP to a specific patient. To this end, a software environment (SEMPATH) is presented, which adapts multiple CPs during execution time and creates a personalized CP for the patient on the fly. A CP is defined as a peculiar case of business process, which means, it can be modeled with BPMN. However, it is not described, if CPGs are modeled and used as well.

Pryss et al. [12] investigate four ward rounds in different clinical departments to determine requirements for tasks management during these ward rounds. The ward rounds have been modeled as BPMN processes to ease discussion with the doctors and identify workflows. That shows, that BPMN is sufficient to model healthcare processes.

3 Usecase

This section introduces a clinical treatment scenario, exemplifying the process-oriented support of medical treatment processes within the PIGS approach.

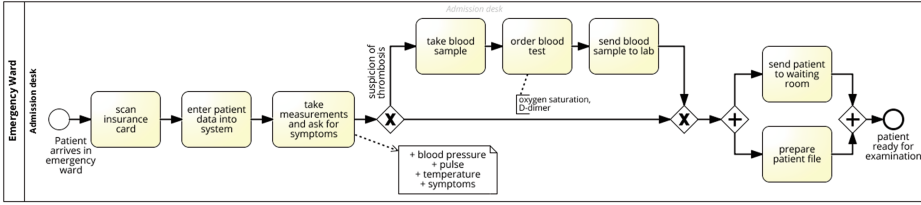


Fig. 1. Patient admission process

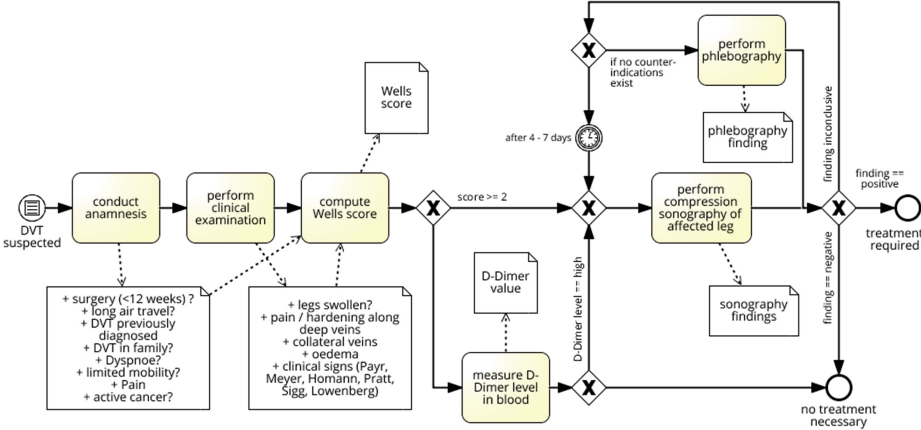


Fig. 2. Diagnosis of Thrombosis

The patient in our scenario is a 65 year old woman, who is transferred from her general practitioner with thrombosis suspicion and overall indisposition.

Figures 1, 2, 3 and 4 show a number of process models of a hospital. The patient admission process in Fig. 1 captures the organizational process of admitting a patient to the hospital, starting with the arrival of the patient in the emergency ward. Figure 2 shows the diagnostic part of the German S2 guideline for deep vein thrombosis (DVT), modeled with BPMN. It is activated when thrombosis is suspected. Based on information acquired during anamnesis and clinical examination, the Wells score needs to be computed to determine the likelihood of thrombosis. If the score value is below two, a blood test for D-dimers is performed, which due to its high specificity is adequate to exclude the possibility of thrombosis with high probability. If the test yields a high D-value or the Wells score was greater than one, a sonography of the affected leg is conducted by an experienced doctor to make the conclusive diagnosis of DVT. Figure 3 displays the guideline for treatment of DVT, which is triggered when the thrombosis diagnosis has been made. It consists of three different therapies, two of which are started immediately. Prior to vitamin K antagonist treatment, however, a gastroscopy is performed to determine the risk of stomach bleeding. The option

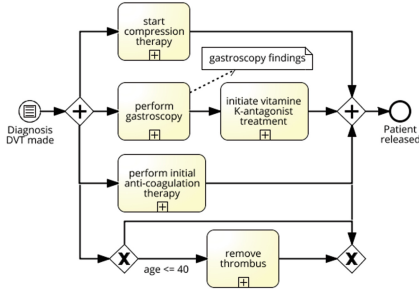


Fig. 3. Therapy of thrombosis

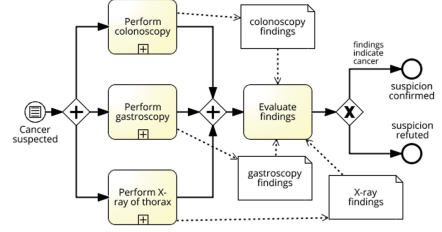


Fig. 4. SOP for cancer elimination

of removing the thrombus is only considered for patients below the age of 40 (the real condition is more complicated). To save space, the therapies are modeled as subprocesses. Figure 4 shows a hospital-specific SOP used to exclude cancer, which is triggered when the doctor suspects cancer as reason for the patient's condition. During anamnesis the doctor asks for the patient's overall condition (appetite, nausea, weight change, etc.) per default and based on the answers might suspect a cancer illness, which often acts as the originator of thrombosis, especially in elderly people. Together these process models form the model level of a PIGS.

We now turn to the execution level, on which the TC for one patient is located. Initially the TC is empty. When a patient arrives, the condition of the admission process becomes true and its start event 'patient arrived' is added to the TC. The first activity is added as recommendation to the TC, yielding the state displayed in Fig. 5.

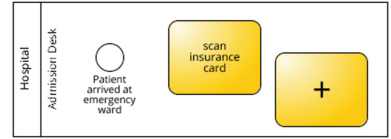


Fig. 5. Initial TC

A later state of the TC is displayed in Fig. 6. At this point in time, already several examinations have been performed, the diagnosis of DVT was confirmed, and therapy started with a Heparin injection and compression of the leg. The patient also was transferred from the emergency ward to a stationary ward of the hospital. Because of the cancer suspicion that arised during the anamnesis the SOP for cancer exclusion is active, as can be seen by the event 'cancer suspected' in Fig. 6. Therefore the three activities 'perform gastroscopy', 'perform colonoscopy', 'order X-ray of thorax' from the cancer exclusion SOP are recommended. However, the activity 'perform gastroscopy' is also part of the DVT treatment CPG, and thus executing it progresses both process models. Figure 6 displays a possible visualization of treatment history and recommendations. Data objects are used to denote the outcome of examinations. The lane construct of BPMN is used to illustrate who performed which task. Lanes are a good starting point to define role-specific views on the TC, i.e. a medical actor would only see the lane of her role or could filter out specific lanes. Furthermore, process

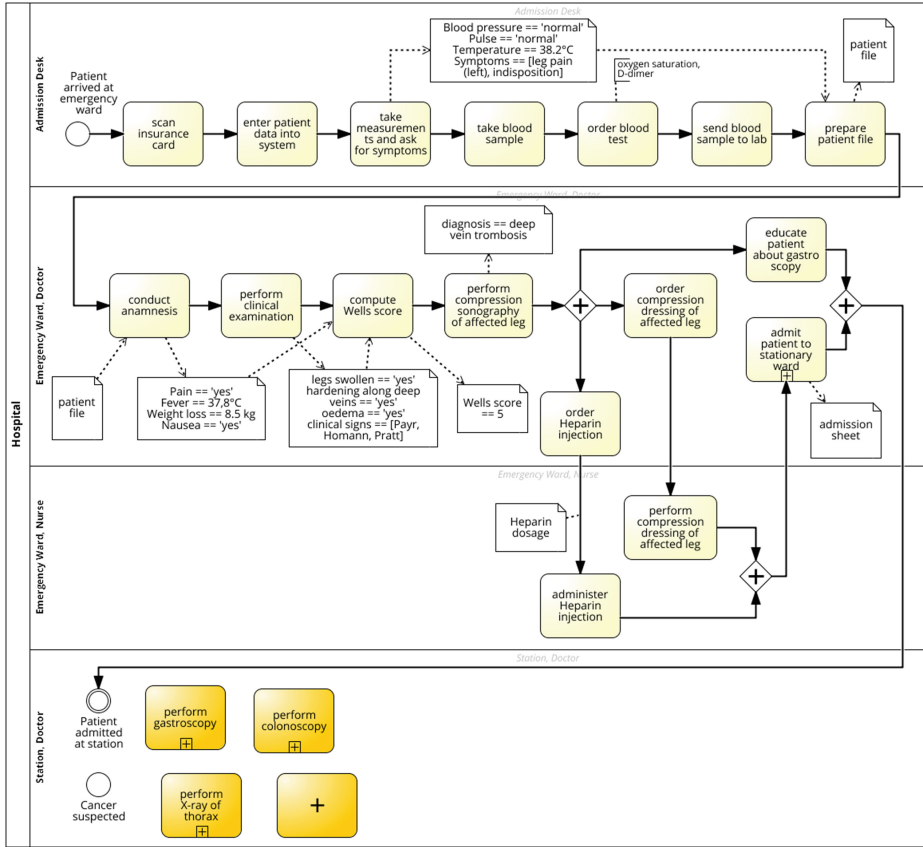


Fig. 6. Advanced state of TC

model abstraction could be used to reduce the number of displayed activities, by summarizing a set of activities into a collapsed subprocess. This is indicated by the collapsed subprocess ‘admit patient to stationary ward’, as well as by the recommended activities, which are also subprocesses. An additional task with the label ‘+’ is shown in Figs. 5 and 6. This modeling element represents an arbitrary activity that can be added to the treatment case. It thus allows medical users to add activities that are not part of a process model.

4 Formalizing Treatment Cases

As was described in the usecase in Sect. 3 several process models and rules govern the recommended behavior of hospital actors during patient treatment. The PIGS approach distinguishes between the model level, where all these models reside, and the execution level, which contains the TC. We first give a formalization of the model level and then turn to how those process models contribute to the actual TC.

4.1 The Model Level

Although CIGs, CPs, SOPs all have a different scope, our approach treats them alike and assumes that they are represented with BPMN. Therefore, we refer to all these different models on model level simply as process models. For this contribution we restrain ourselves to a small subset of BPMN constructs, leaving out timer events, exception handling, complex gateways among others. As the goal of the PIGS approach is to support hospital actors by providing an overview of and guiding them through the treatment process, we need to represent these processes formally in a computer system. A process model is a directed graph consisting of activity, event, gateway and data object nodes, connected by (control and data) flow edges.

Definition 1 (Process Model). A *process model* P is a tuple $(A, E, G, D, F, \varepsilon)$ where A is a set of activities, E a set of events, $G = G^\wedge \cup G^\times$ is a set of parallel resp. exclusive gateways, D is a set of data objects used in the model, $F \subseteq (A \times D) \cup (D \times A) \cup (A \cup E \cup G)^2$ is the flow relation, and ε is a partial function that assigns an expression to some activities and events.

Data objects are connected to activities by the flow relation F , which also connects activities, events and gateways with each other. The set of predecessors resp. successors is denoted by $\bullet N$ resp. $N \bullet$ for $N \in A \cup E \cup G$; the set of data preconditions resp. postconditions is denoted by $\blacksquare N$ resp. $N \blacksquare$ for $N \in A$. We assume that activities have at most one predecessor ($\bullet A \leq 1$) and at most one successor ($A \bullet \leq 1$). Furthermore, if a gateway has multiple successors it can only have one predecessor and vice versa. In the first case the gateway is called a split gateway (denoted $G^<$), in the latter case it is called a join gateway (denoted $G^>$). Expressions are used to specify conditions for activities and events, e.g. $\varepsilon(\text{'remove thrombus'}) = \text{'age} \geq 40$ ' in Fig. 2, stating that 'remove thrombus' activity can only be performed if the patient is younger than 40 years old. In the same way the start event of the admission process in Fig. 1 can only occur, when a patient arrives at the emergency ward. We leave the formal specification of the expression language for future work and just state here that expression terms can be formed over patient data and event of TCs.

Definition 2 (Model Level). The *model level* of a PIGS consists of set of process models \mathcal{P} and an universe for activities \mathcal{A} , events \mathcal{E} , and data objects \mathcal{D} , such that $\forall P_i \in \mathcal{P} : A_i \in \mathcal{A}, E_i \in \mathcal{E}, D_i \in \mathcal{D}$.

The universes hence encompass all activities, events, and data objects that occur in the process models. However, \mathcal{A} can contain additional activities that can be added manually to the TC by medical actors. The above definition allows for process models to share activities, events, and data objects, e.g. in our usecase the activity 'perform gastroscopy' is used both in the cancer exclusion SOP and the DVT treatment CPG.

In BPMN, activities and gateways are not atomic, but rather have internal states, e.g. they can be **running** or **terminated** [1]. These states are necessary

to faithfully capture the progress of the activities' execution and to define an operational semantics for process models. For example, an activity can only terminate if it is in state **running**. These fine-grained activity states are formalized as transition system called *lifecycle*, which depicts possible states an activity can be in and viable state transitions. We use a simplified lifecycle, similar to the one stated in [1], which can be seen as an abstraction of what is used in actual process engines, like jBPM² (described in Chap. 7.4 of [13]). Lifecycles are represented graphically with dots (states) and arrows (state transitions); state transitions are annotated with the event that triggers the transition. The lifecycle of an activity is shown in Fig. 7a and the one for gateways in Fig. 7b.

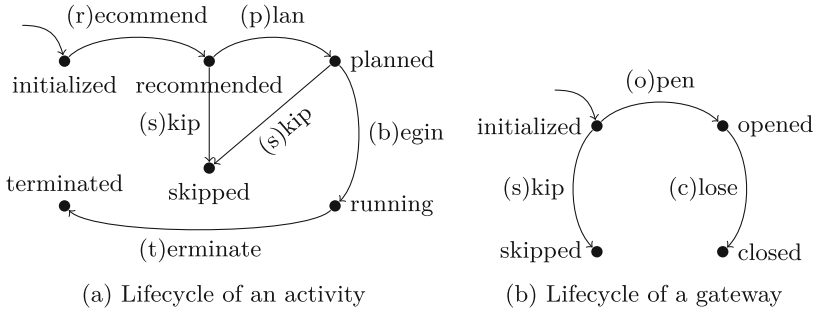


Fig. 7. Lifecycles of activities and gateways

After creation, an activity is in the state **initialized**. Once the PIGS determines the activity's preconditions are fulfilled, the system puts the activity into the state **recommended**, meaning that it is displayed to the medical user. Now the user can either add it to the TC, leading to state **planned** or skip its execution (state **skipped**). A planned activity can begin execution, resulting in state **running**. The final state **terminated** indicates that the activity finished execution. States **recommended** and **planned** replace the state **enabled** usually used in activity lifecycle [1]. This allows to distinguish between activities recommended by the system and planned for execution by the medical user. Gateways are initialized in the beginning and can transition to state **opened**. From there they can transition to state **closed** or be skipped. The rules governing the transitions are described in Sect. 5.

4.2 The Execution Level

The central artefact on the execution level is the TC, which represents the running treatment process for one specific patient. Contrary to a process instance in BPMN, a TC does not belong to a single process model, but to a set of

² <http://jbpm.jboss.org>.

models on the model level, that contributed some treatment steps to this particular TC. Whenever an activity or gateway transitions to a new lifecycle state, the occurrence of that transition is captured in the TC by adding an corresponding event to E_c and extending the order $<_c$. Hence, the TC records the history of one patient's treatment. Formally, the TC can be defined as follows.

Definition 3 (TC). A TC c is a tuple $(E_c, <_c)$, where E_c is a set of events corresponding to occurrences of lifecycle transitions of activities, gateways, and BPMN events; and $<_c$ is a partial order on E_c , representing causal dependencies of events and respecting activity and gateway lifecycles.

$<_c^\circ := \{x \in E_c \mid \neg \exists y \in E_c : (x, y) \in <_c\}$ denotes the set of maximal elements of the partial order.

While a TC c represents the whole treatment history, the set $<_c^\circ$ of maximal elements represents the current state, i.e. only the last events that occurred. Figure 8b shows a TC for the BPMN model in Fig. 8a. We use the shorthand notation $X.y$ to denote the event that activity X performed the transition y . $D := 7$ represents the event that A produced the data object D with the result 7. Therefore, $\varepsilon(B) = "D \geq 5"$ was satisfied and B recommended, as well as C , which has no attached expression. This particular TC displays the state after C has been planned. At the same time B is skipped and the gateway is closed. Init transitions are not displayed.

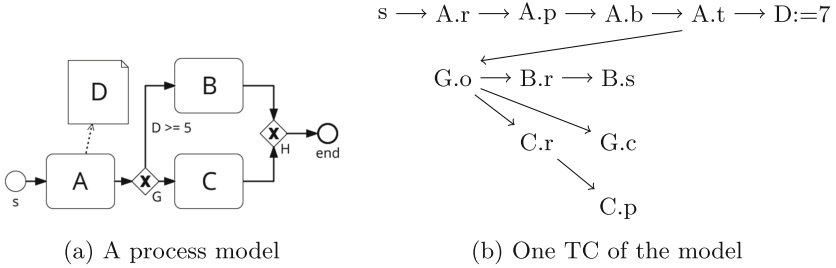


Fig. 8. Example of a process model and a possible TC

5 Generating Treatment Recommendations

Having introduced the model level and TCs, we now turn to explaining how the PIGS approach recommends treatment steps. We approach this problem by giving an operational semantics to TCs, which formally specifies how they evolve. Thus, recommendation of treatment steps reduces to determining those activities from active process models, for which the preconditions are fulfilled. In the case of one active process model this coincides with the usual execution semantics of

BPMN, given by Weske [1] through event diagrams. This contribution formalizes and extends the usual semantics for the case of multiple active process models.

The semantics is defined as a set of inference rules (antecedent \Rightarrow consequent), which are applied by the recommendation algorithm to the current treatment state, thus progressing the TC. The antecedent of a rule is to be read as follows: $X.y$ is a shorthand notation for $X.y \in <_c^\circ$, i.e. the event y of activity or gateway X belongs to the maximal elements of the partial order of the TC. Therefore, the evolution of a TC depends only on the current state of the TC as given by $<_c^\circ$. When the antecedent of a rule is fulfilled its consequent alters the TC by adding events. A consequent $W.z$ is shorthand for $E'_c = E_c \cup \{W.z\}$ and in addition $<'_c = <_c \cup \{(X.y, W.z)\}$ if $X.y$ has been given in the antecedent. We now discuss the rules in detail, and afterwards discuss decisions.

Activation Rules. Initially the TC for a patient is empty. A process model is activated, if the condition of its start event is fulfilled. The start event is added to the TC and all elements of the respective process model get initialized as stated by rule *Init*. During the course of a treatment, other process models can become activated, resulting in their start events being added to the TC.

The rule *Rec* specifies the conditions for activities to be recommended. An activity must be initialized, its predecessor in one of the process models must have terminated, and the expression attached to the activity must be fulfilled. Since several process models might be active and therefore many activities can be recommended, the system recommends only single activities rather than a sequence of activities, to avoid confusion. However, each recommendation is linked to the process model it is derived from, thus allowing to check the further course of action. The initialization condition $X.i$ prevents the recommendation of activities that succeed manually added activities Table 1 (using rule *Manu* in Table 2).

Table 1. Activation of process models and recommendations

<i>Init</i>	$s \in E_i \wedge \varepsilon(s) \Rightarrow s \wedge \forall X \in G_i \cup A_i : X.i$
<i>Rec</i>	$X.i \wedge Y.t \wedge (Y, X) \in F_i \text{ for some } i \wedge \varepsilon(X) \Rightarrow X.r$

Lifecycle Rules. The rules in Table 2 describe the (straightforward) progress of activities according to the lifecycle in Fig. 7a. The *Plan* rule states that an activity in state **recommended** can transition via the **plan** action into state **planned**. In this case the event $X.p$ is added to the TC event set E_c and the arc $(X.r, X.p)$ is added to the partial order $<_c$. This is analogous for the rules *Skip* and *Begin*. The rule *Term* states that when an activity terminates,

Table 2. Lifecycle transition rules

<i>Plan</i>	$X.r \Rightarrow$	$X.p$
<i>Skip</i>	$X.r \Rightarrow$	$X.s$
<i>Start</i>	$X.p \Rightarrow$	$X.b$
<i>Term</i>	$X.b \Rightarrow$	$X.t \wedge D := val$
<i>Manu</i>	\Rightarrow	$X.p$

the rules *Skip* and *Begin*. The rule *Term* states that when an activity terminates,

it can produce a data event when it has data postconditions, i.e. $X_{\blacksquare} \neq \emptyset$. This data event refers to a data object and records its value. In the partial order it is connected to $X.b$. Finally, the *Manu* rule states that users can add arbitrary activities from \mathcal{A} to the TC, corresponding to the blank activity labeled with ‘+’ discussed in Sect. 3.

The state transitions are triggered by the user of PIGS. For activities that can be executed automatically without the intervention of a medical user the states **planned**, **running**, and even **terminated** might coincide, e.g. for an activity ‘send release letter to general practitioner’.

Gateway Rules. The rules in Table 3 describe how gateways are opened and closed. The first rule, *Open*, states that initialized gateways are opened when their predecessor terminates (or one of their predecessors in case of a join). The second rule, $Rec^<$, states that open split gateways recommend all of their succeeding activities, the conditions of which are fulfilled. However, only one of those activities can be chosen by the medical user, because they are exclusive. This is ensured by the rule $SClose^{\times}$, which skips all other successors of a XOR split when one activity transitions to state **planned**. The third rule, $Rec^>$, states that join gateways recommend their only successor unconditionally when they close. The following four rules define closing of gateways. AND splits ($SClose^{\wedge}$) close when no successor was skipped or is still recommended, while XOR splits ($SClose^{\times}$) close when one successor is added to the TC, simultaneously skipping all other successors. AND joins ($JClose^{\wedge}$) close when all predecessors terminated, while for XOR joins ($JClose^{\times}$) one terminated predecessor suffices to close.

Table 3. Gateway transition rules

Open	$G.i$	\wedge	$X.t$	\wedge	$X \in \bullet G$	\Rightarrow	$G.o$	
$Rec^<$	$G^<.o$	\wedge	$X \in G^<\bullet$	\wedge	$\varepsilon(X)$	\Rightarrow	$X.r$	
$Rec^>$			$G^>.t$	\wedge	$X \in G^>\bullet$	\Rightarrow	$X.r$	
$SClose^{\times}$			$G^{\times}.o$	\wedge	$\exists! Y_k \in G^{\times}\bullet : Y_k.p$	\Rightarrow	$G^{\times}.c$	$\wedge \forall l \neq k Y_l.s$
$SClose^{\wedge}$			$G^{\wedge}.o$	\wedge	$\forall X_k \in G^{\wedge}\bullet : \neg(X_k.r \vee X_k.s)$	\Rightarrow	$G^{\wedge}.c$	
$JClose^{\times}$			$G^{\times}.o$	\wedge	$\exists X \in \bullet G^{\times} : X.t$	\Rightarrow	$G^{\times}.c$	
$JClose^{\wedge}$			$G^{\wedge}.o$	\wedge	$\forall X \in \bullet G^{\wedge} : X.t$	\Rightarrow	$G^{\wedge}.c$	

Decisions. In general, activities have three different types of preconditions: control flow, i.e. another activity has to have happened before; data flow, i.e. a certain data object needs to be present in a particular state; and decision conditions. Decisions, which are part of all process modeling approaches, describe which of several activities to choose depending on data. A simple example would be to administer a blood thinner if the blood pressure of a patient is above a certain threshold. Such decisions are specified by attaching conditions to the successor nodes of a XOR gateway using the ε function.

The rules capture control flow conditions by referring to the flow relation F_i of process models. For data preconditions and the simple decisions described

above, the rules refer to the function ε defined in Definition 1, mapping activities to an expression. The condition $\varepsilon(X)$ is part some rules' antecedent, which thus is only fulfilled if the expression evaluates to true for X . However, this evaluation is outside the scope of this contribution.

In many cases decisions are more complex and have several options on which to decide, and every option has a set of supporting and opposing arguments, the evaluation of which leads to a ranking of options. In such cases the outcome of evaluating $\varepsilon(X)$ is not boolean, but rather a numerical value, quantifying the support for an option. Technically, all options above a certain threshold should be recommended, weighted by their support. This would require to add the confidence of recommendations to the TC formalism and will be part of future work.

6 Conclusion

In this contribution we presented further details of the PIGS approach for process-oriented IT support of medical and organizational processes in hospitals. A set of BPMN process models capturing CPGs, CPs, SOPs and organizational processes is used to derive recommendations for treatment steps based on the current state of a TC. Therefore, we extended the usual lifecycle of an BPMN activity by an additional state. We gave a formal definition of a TC and a set of inference rules, which specify the evolution of the TC. Redundant and conflicting execution of activities can thus be avoided, since the treatment history is available to the user and the system. Since we only work with atomic tasks in the sense of BPMN in the Treatment Case, subprocesses in the model are handled by unfolding its internal to an atomic level. One limitation is the strict interpretation of process models, i.e. if activities from one guideline are performed in a different order or are skipped, the user “leaves” the process and therefore the guideline. This means she gets no further support for this model. A less strict interpretation should be considered in the future, especially if a step was skipped, because it has already been done in the nearer past or it would contradict to another recommendation. Another aspect, we haven't considered yet are tasks, which have to be scheduled at a specific point in time or that occur repetitively. Data-binding is out of scope in this work, but a data model (or ontology) has to be specified, when we implement the system. For now, we assume a simple key-value-store for the process models and treatment cases, which then has to be mapped to a specific ontology.

References

1. Weske, M.: Business Process Management: Concepts, Languages, Architectures, 2nd edn. Springer, Heidelberg (2012)
2. Grimshaw, J.M., Russel, I.T.: Effect of clinical guidelines on medical practice: a systematic review of rigorous evaluations. *Lancet* **342**(8883), 1317–1322 (1993)
3. Sutton, D.R., Fox, J.: The syntax and semantics of the PROforma guideline modeling language. *J. Am. Med. Inform. Assoc.* **10**(5), 433–443 (2003)

4. Boxwala, A.A., Peleg, M., Tu, S., Ogunyemi, O., Zeng, Q.T., Wang, D., Patel, V.L., Greenes, R.A., Shortliffe, E.H.: GLIF3: a representation format for sharable computer-interpretable CPGs. *J. Biomed. Inform.* **37**(3), 147–161 (2004)
5. Quaglini, S., Stefanelli, M., Cavallini, A., Miceli, G., Fassino, C., Mossa, C.: Guideline-based careflow systems. *Artif. Intell. Med.* **20**(1), 5–22 (2000)
6. Scheuerlein, H., Rauchfuss, F., Dittmar, Y., Molle, R., Lehmann, T., Pienkos, N., Settmacher, U.: New methods for clinical pathways - business process modeling notation (BPMN) and tangible business process modeling (t.BPM). *Langenbecks. Arch. Surg.* **397**(5), 755–761 (2012)
7. ISO/IEC: Information technology - object management group business process model and notation. Technical Report 19510, ISO/IEC, July 2013
8. Hewelt, M.: Process information and guidance systems in the hospital. In: KR4HC 2014 workshop proceedings (2014)
9. Milla-Millán, G., Fdez-Olivares, J., Sánchez-Garzón, I.: A common-recipe and conflict-solving MAP approach for care planning in comorbid patients. In: Bielza, C., Salmerón, A., Alonso-Betanzos, A., Hidalgo, J.I., Martínez, L., Troncoso, A., Corchado, E., Corchado, J.M. (eds.) CAEPIA 2013. LNCS, vol. 8109, pp. 178–187. Springer, Heidelberg (2013)
10. Sánchez-Garzón, I., Fdez-Olivares, J., Onaindía, E., Milla, G., Jordán, J., Castejón, P.: A multi-agent planning approach for the generation of personalized treatment plans of comorbid patients. In: Peek, N., Marín Morales, R., Peleg, M. (eds.) AIME 2013. LNCS, vol. 7885, pp. 23–27. Springer, Heidelberg (2013)
11. Alexandrou, D.A., Skitsas, I.E., Mentzas, G.N.: A Holistic Environment for the Design and Execution of Self-Adaptive Clinical Pathways. *IEEE Trans. Inform. Technol. Biomed.* **15**(1), 108–118 (2011)
12. Pryss, R., Langer, D., Reichert, M., Hallerbach, A.: Mobile task management for medical ward rounds – the MEDo approach. In: La Rosa, M., Soffer, P. (eds.) BPM Workshops 2012. LNBIP, vol. 132, pp. 43–54. Springer, Heidelberg (2013)
13. jBPM team, T.J.: jBPM Documentation - User Guide. jboss.org. 6.0.1.Final edn., November 2013

Business Process Management Workshops
BPM 2014 International Workshops, Eindhoven, The
Netherlands, September 7-8, 2014, Revised Papers
Fournier, F.; Mendling, J. (Eds.)
2015, LV, 588 p. 220 illus., Softcover
ISBN: 978-3-319-15894-5