Received: 16 October 2016 | Revised: 11 October 2017 | Accepted: 6 November 2017
DOI: 10.1002/smr.1931

WILEY Journal of Software: Evolution and Process

RESEARCH ARTICLE

A hybrid approach for aspect-oriented business process modeling

Amin Jalali[1] | Fabrizio Maria Maggi[2] | Hajo A. Reijers[3,4]

[1]Stockholms Universitet, DSV, Postbox 7003, 164 07 Kista, Sweden
[2]University of Tartu, Tartu, Estonia
[3]VU University Amsterdam, Amsterdam, The Netherlands
[4]Eindhoven University of Technology, Eindhoven, The Netherlands

Correspondence
Amin Jalali, Stockholms Universitet, DSV, Postbox 7003, 164 07 Kista, Sweden.
Email: aj@dsv.su.se

Abstract

Separation of concerns has long been an important strategy to deal with complexity when developing a system. Some concerns (like security) are *scattered* through the whole system, and different modules are *tangled* to such concerns. These concerns are known as *cross-cutting* concerns. When the system in question is a business process, cross-cutting concerns are aimed at being encapsulated by Aspect-Oriented Business Process Modeling. However, the state-of-the-art techniques in this field lack efficient mechanisms that (1) support composition of cross-cutting concerns that can be defined in parallel to (a part of) a process model and (2) enable specifying both mandatory and optional cross-cutting concerns. To address these limitations, this paper proposes a new Aspect-Oriented Business Process Modeling approach. The approach is hybrid since it is based on *declarative* rules to relate *imperative* cross-cutting concerns and *imperative* business process models. The approach is explained, formally grounded with precise semantics, and used accordingly to implement the artifacts that support modeling and enactment of business processes in the proposed fashion as a proof of concept. In addition, the approach is evaluated on the basis of the Technology Acceptance Model during a workshop session. The result shows that participants perceived the approach usable and easy to use.

KEYWORDS

aspect orientation, business process modeling, cross-cutting concerns, declarative rules, hybrid models

1 | INTRODUCTION

Separation of concerns is an important strategy for people to deal with complexity. This strategy has been widely used and adopted in developing systems where designers can divide the specifications of systems into smaller individual modules. These modules can be managed, understood, and changed separately but integrated into a comprehensive, overall specification. The lack of a proper separation can lead to very intertwined complex systems, which are hard to be changed and managed.

Business processes are examples of such systems, which can be defined through sets of constraints to fulfill specific goals. These constraints specify how activities in a business process can be managed, and they are categorized into 2 major groups. One group consists of procedural flow-based constraints that specify how a business process should be performed step by step. The other group consists of very flexible models suitable for specifying processes characterized by high variability, which can hardly be specified as a predetermined procedure. We call the models in the first group imperative models while those in the second group declarative models.[1]

Imperative models specify the behavior of flow-oriented and rigid business processes. In these models, the behavior of a process is specified explicitly. Imperative models can be expressed using Business Process Model And Notation (BPMN),[2] Yet Another Workflow Language (YAWL),[3] Petri nets, etc. In contrast, *declarative models* allow all behavior to happen unless it is explicitly forbidden. Thus, they suit knowledge intensive business processes better, where people have a high degree of flexibility in the process enactment.[1] The border between imperative and declarative models is not very distinct. Indeed, there are variants of processes that cannot be specified solely by imperative or declarative constraints effectively. Instead, a combination of these constraints defines how those processes should work. The techniques that aim at specifying processes using a combination of imperative and declarative constraints are called *hybrid modeling* techniques.[4-10]

The selection of the right technique for specifying a business process is not always sufficient to enable process modelers to deal with the complexity of business processes. Therefore, there are available different modularization techniques, which are introduced for this purpose.[11] Some of these techniques have been defined considering that some concerns in organizations like security and privacy are not limited to 1 business process, but they can have effect in many different processes. These concerns are known as *cross-cutting* concerns, which support interoperation in organizations.[12] The Aspect-Oriented paradigm proposes a new modularization technique in which a developer can write unitary and separate statements to define the relations between different modules and these concerns.[13] In the process modeling area, Aspect-Oriented Business Process Modeling (AO-BPM) aims at encapsulating these concerns.[14]

Aspect-Oriented Business Process Modeling approaches specify how cross-cutting concerns can be managed to a large extent for imperative business process models[15,16] and can be applied to specific points of the process models, ie, before, after, or around an activity. Therefore, these approaches cannot specify cross-cutting constraints that can be executed in parallel to the main process and that cannot be attached to a specific activity and those that are optional. For example, as a security concern for the entry process into the United States, the arriving travelers should submit their passport information and customs declaration form prior to Customs and Border Protection inspection.* A traveler can submit the documents at any time after arrival and before Customs and Border Protection inspection. This sort of concerns cannot be encapsulated through existing AO-BPM approaches since it is not bounded to a single activity in a process model. Instead, a traveler can do it at any time in parallel to the remaining part of the process.

As a second example, we consider the different financial concerns that are running in an organization. In Stockholm University, any reimbursement can be applied after the payment but before the accounting department closes the fiscal year. Failing to submit the reimbursement does not prevent the process to continue (the accounting department will close the fiscal year at the right time), yet the applicant will not be reimbursed. This is a typical example of the concern that is not mandatory for a business process.

The current state-of-the-art techniques in AO-BPM lack efficient mechanisms that (1) support composition of cross-cutting concerns that can be defined in parallel to (a part of) a process model and (2) enable specifying both mandatory and optional cross-cutting concerns. The problem roots in the fact that current AO-BPM approaches are developed for imperative process modeling techniques, and they are not flexible enough to support articulation of such processes that requires more flexibility. Therefore, we propose to solve these limitations using a hybrid modeling approach that enables relating imperative cross-cutting concerns and business process models through declarative rules.[17] We assume that cross-cutting concerns and business process models are represented using an imperative notation since we want to support modelers in scenarios where processes are executed following well-defined procedures. Take for example a procedure for handling a loan application or a procedure for admitting a patient in a hospital. In these cases, representing these procedures in a declarative fashion would quickly hamper the understandability of the models (too many rules are generally needed to model well-structured processes). In addition, since nowadays workflow systems are mainly driven by procedural specifications, configuring these systems starting from a declarative model could become very problematic.

We give a formal definition of syntax and semantics for our approach, and we proof the soundness of the resulting models. This is an important property that needs to be satisfied when the models have to be executed to guarantee a proper completion of each process execution. We have also implemented artifacts, as proofs of concept, to support both modeling and enactment of business processes, specified using this approach. The approach has been applied to a case from the educational domain to demonstrate how it supports the execution of business processes and cross-cutting concerns. In addition, the usability of the approach has been studied through a workshop. The result of the workshop shows that the hybrid approach can enhance AO-BPM to specify both optional and mandatory concerns, which can be executed concurrently with other process functionalities.

The remainder of this paper is structured as follows. Section 2 provides basic terminologies and a summary of aspect orientation in business process management. Section 3 explains the proposed approach. Sections 4 and 5 introduce definitions and semantics. Section 6 describes the artifacts implemented to support the proposed approach. Section 7 reports the result of the workshop that studies the usefulness of the approach. Section 9 discusses the related work. Section 10 concludes the paper and explains future directions for research.

2 | BACKGROUND

Separation of concerns has long been an important issue in software development domain, and there are different modularization approaches that support it. In this section, we introduce modularization techniques in business process modeling. Then, we will introduce how an aspect-oriented approach can support encapsulation of cross-cutting concerns in business process models.

2.1 | Modularization techniques in business process modeling

Real business processes are usually very complex so that business process modeling is often a challenge for process designers. Modularization techniques are means to separate the different concerns of a business process so as to facilitate coping with its complexity. La Rosa et al[11]

*http://www.cbp.gov/newsroom/national-media-release/2014-08-11-000000/new-mobile-passport-control-app-available

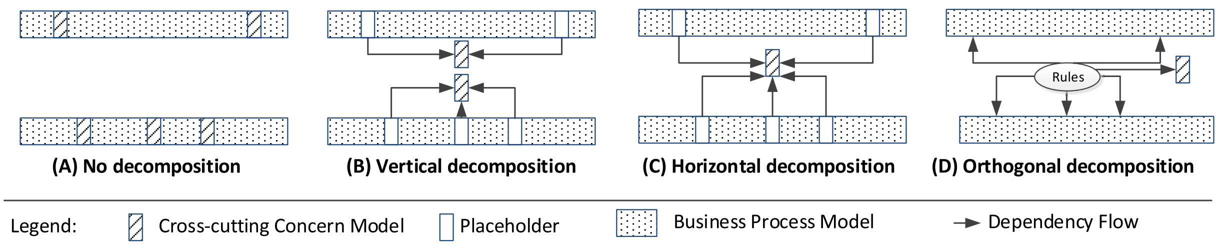

(A) No decomposition
(B) Vertical decomposition
(C) Horizontal decomposition
(D) Orthogonal decomposition
Rules
Legend:
Cross-cutting Concern Model
Placeholder
Business Process Model
Dependency Flow

FIGURE 1 An abstract representation of modularization techniques for business process modeling[14]

categorize the modularization techniques in business process modeling into 3 classes: vertical, horizontal, and orthogonal. Figure 1 shows an abstract representation of these techniques.

Figure 1A shows a scenario where no decomposition is applied. As can be seen, the concerns should be remodeled several times in each process model. This repetition adds costs in terms of process design and maintenance. To avoid such problems, the concerns should be encapsulated to make them reusable.

Figure 1B shows a scenario where vertical decomposition is applied. This decomposition suits a situation where the concerns are limited to 1 process model, and the process designer knows exactly the places in the process model where each concern should be considered. As can be seen, the concern is encapsulated in a separate module for each process model and is called by adding a placeholder in it. Although the encapsulated module can be reused several times within the scope of the same process model, the concern should be remodeled to be used for other process models (scattering problem). This still hinders reusability in cases where concerns should be shared by more than 1 process model.

Figure 1C shows a scenario where horizontal decomposition is applied. This decomposition suits a situation where the concerns are not limited to 1 process model, and, also, the process designer knows exactly the places in the process models where each concern should be imposed. The concern is encapsulated in a separate module and is called by adding placeholders in the process models. This is a very effective decomposition technique, but still not flexible because the process designer should explicitly specify the places in the process models where the concern should be applied. Therefore, a change in the regulations on how to apply a cross-cutting concern enforces modelers to investigate all placeholders and refine them accordingly (tangling problem).

Figure 1D shows a scenario where orthogonal decomposition is applied. This decomposition suits a situation where the concerns are not limited to 1 process model, and the process designer does not know the places in the process models where each concern should be considered beforehand. The concern is encapsulated in a separate module and is imposed in the process models when some conditions are satisfied. These conditions are articulated using some rules that specify the circumstances under which the concern should be applied. Neither the process models nor the cross-cutting concerns are aware of each other in this modularization technique. In this way, if the condition for applying a concern changes, the process designer only needs to change these rules. In this sense, this modularization technique is more flexible than all the others. Aspect-Oriented Programming (AOP) is an example of an orthogonal modularization technique.

2.2 | Modularizing aspects with aspect orientation

Aspect-Oriented Programming proposes to encapsulate the core functionalities of programs in modules called *core classes* and to encapsulate the cross-cutting functionalities in other modules called *advices*.[18] Advices encapsulate actions that should be performed to fulfill cross-cutting concerns, and they can later be related to core classes through asserting sentences like: "In program P, whenever condition C arises, perform action A."[19] These assertions are called pointcuts. The combinations of advices and related pointcuts are called aspects.

Similar terminologies are applied in AO-BPM. The left-side part of Figure 2 shows 4 processes and 4 cross-cutting concerns, symbolized with horizontal and vertical bars, respectively. It is visually expressed that a business process must address a concern if the concern crosses over it. AO-BPM aims at modeling the core functionalities of each of these business processes in a module called core concern (see the right-hand side of Figure 2). Each cross-cutting concern is encapsulated in a module called advice. Pointcuts are components that enable assertion of statements

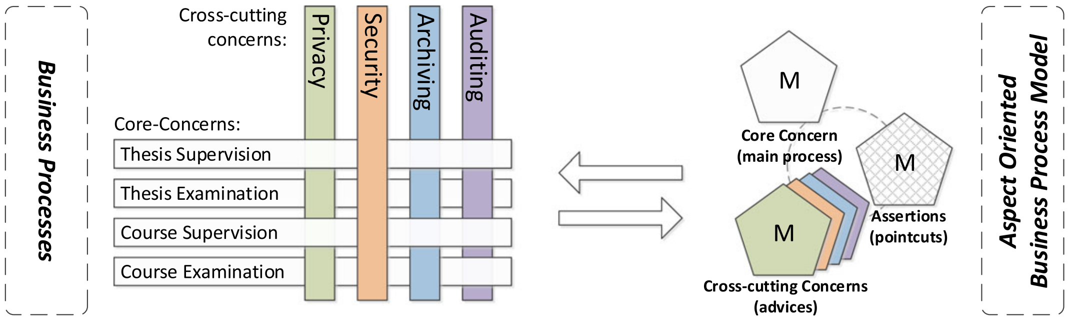


FIGURE 2 Aspect-Oriented Business Process Modeling

like “In process P, whenever condition C arises, perform advice A.” In Figure 2, main process, advices, and pointcuts are represented by pentagons (annotated by *M* as an acronym for “Model”).

The strength of AOP is the possibility of “programming by making quantified programmatic assertions over programs written by programmers oblivious to such assertions.”[19] The same advantages are applied to the business process modeling area, where *Obliviousness* and *Quantification* are considered as fundamental principles. Obliviousness refers to the fact that modelers can design the core concern without knowing about cross-cutting concerns. Quantification is the mechanism to recognize the points in the core concerns where a concern should be applied (join points).

To enact aspect-oriented models, they should be transformed into standard models, or the system that enacts them should be aware of their semantics.[20] Transforming these models into a standard notation is called *Static Weaving*,[21] and enacting them through a system that is aware of their semantics is called *Dynamic Weaving*.[22] Weaving is a composition process that interconnects core and cross-cutting concerns together based on the rules specified in pointcuts, thus resulting in a unified process specification.

3 | APPROACH

This section introduces a new approach to modularize cross-cutting concerns in the process modeling area to address the limitations mentioned in Section 1. The new approach supports 2 requirements, ie, (1) the definition of both mandatory and optional cross-cutting concerns and (2) the composition of cross-cutting concerns that can be defined in parallel to (a part of) a process model. We use one of the processes in Figure 2, the *Course Examination* process, as a running example to explain our approach.

3.1 | Core concern

The *Course Examination* process should comply with 3 cross-cutting concerns, ie, privacy, security, and archiving concerns in accordance with the concerns that cross over this process (see Figure 2). To specify this process, we use the Petri net shown in Figure 3. The *Course Examination* starts with the *Schedule The Exam* task performed by the education secretary. Then, students can register for the exam through the *Register Students* task. In parallel, the *Send Exam Questions* task enables the examiner to send the exam questions (at least 1 wk before the exam) to the secretary. After the registration deadline, the secretary prepares the exams through the *Print Exam Materials* task.

After the exam, the administrative personnel registers students who participated in the exam through the *Register Participant List* task. The examiner grades the exam and reports the result through the *Register Grades* task. The grades are A, B, C, D, E, Fx, and F. Grades Fx and F are failing grades; however, students who receive Fx have another chance to improve their grades to E. The examiner should provide instructions (as an assignment) for those who received the grade Fx through the *Send Fx instruction* task. Students can *Submit Assignments* before the specified deadline. Thereafter, the examiner grades the submissions and reports the result through the *Register Grades* task. Finally, the Course Leader (can be the same person as the examiner) writes the course summary.

3.2 | Cross-cutting concerns

Figure 4 shows cross-cutting concerns that are applicable for the *Course Examination* process. These concerns are encapsulated as advices and are modeled using Petri nets.

As a privacy concern, the *Grade Inform* advice specifies that the students should be informed about grades through emails. As a security concern, the *Grade Registration* advice specifies that the administrative staff should provide a grading registration form, which should be signed by the examiner. As an archiving concern, the *Archive Examination* advice specifies that the examined sheets should be scanned if they are not in digital format by the administrative staff. Subsequently, they should be uploaded to the system. Moreover, the *Archive Exam Materials* advice specifies that the examiner can optionally archive exam material like questions and correct answers.

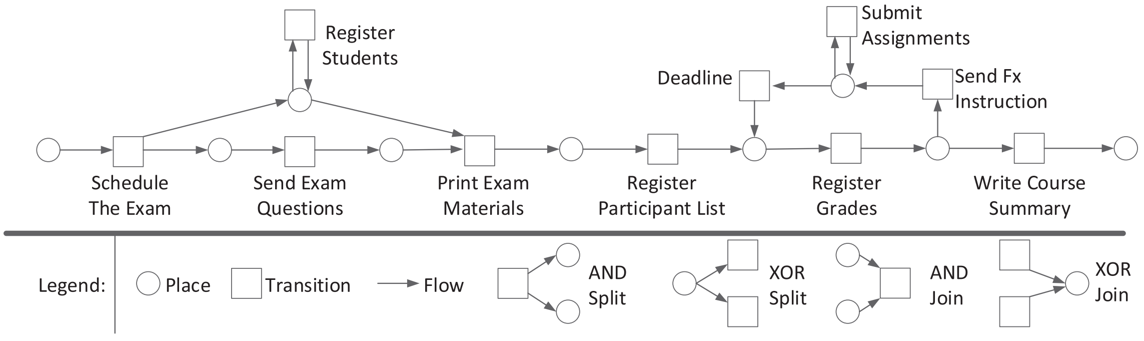


FIGURE 3 Course Examination process

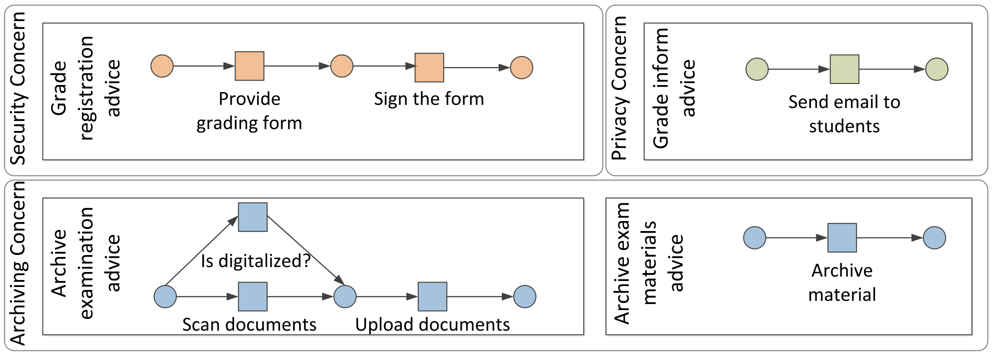


FIGURE 4 Cross-cutting concerns for the Course Examination process

3.3 | Pointcuts

Having both a core concern (the main process) and cross-cutting concerns, we can specify *pointcuts* to relate these parts. To do this, we need to introduce a new language to specify pointcuts that is able to cope with the imperative nature of core and cross-cutting concerns and, at the same time, it is declarative so that the 2 parts can be connected in a flexible way (like in any orthogonal decomposition). Three elements should be considered in the definition of the language, ie, (1) if quantification is clear-box or black-box, (2) if quantification is static or dynamic, and (3) if weaving is static or dynamic. Figure 5 shows the relation of these options with the issues they might introduce.

Quantification is the mechanism to identify the join points in the core concern, ie, the points in the process model where a cross-cutting concern should be applied. Quantification can be clear-box or black-box. A clear-box quantification is specific to 1 notation (see, for example, Reichert and Weber[23]). Therefore, since there are many different syntaxes to specify a process model, using a clear-box quantification makes it very difficult to end up with a general way to specify pointcuts. This issue is indicated by *S* in Figure 5. In contrast, a black-box quantification enables abstracting from a specific syntax, and elements of the core process model are quantified through general interfaces (eg, the activity names).

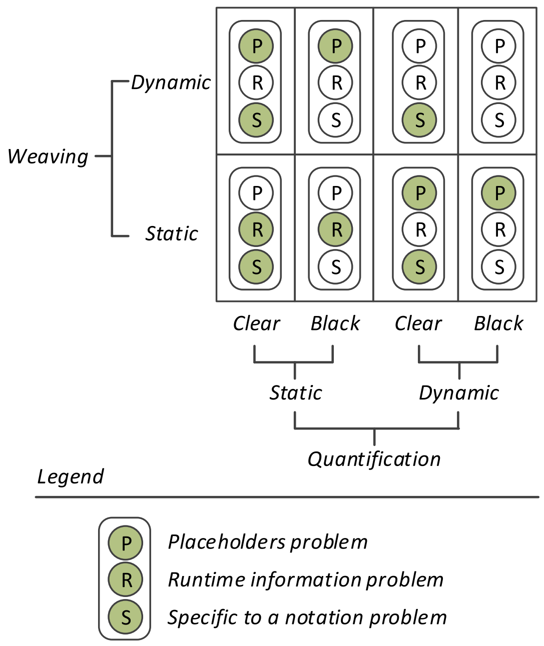


FIGURE 5 Design criteria for a pointcut language

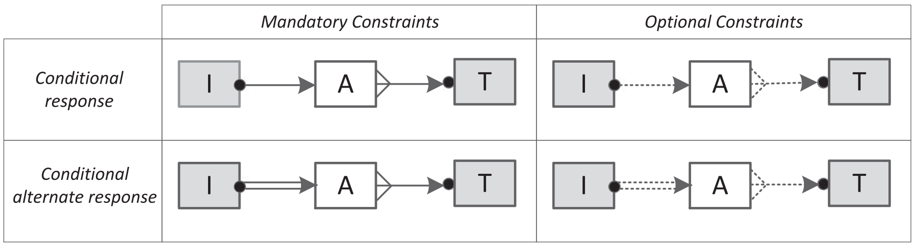


FIGURE 6 Proposed language for the specification of pointcuts

Quantification can be performed at design time (static) or at runtime (dynamic). We stress that this classification of quantification is not given on the basis of when the rules (ie, the pointcuts) are specified, but based on when the join points (ie, the exact points in the core concern where the cross-cutting concerns should be applied) are identified. Indeed, the pointcuts can be asserted at design time for both static and dynamic quantification. Sometimes, in dynamic quantification, the pointcuts can be specified at runtime. The assertion of pointcuts at runtime is called in the literature late modeling.[24]

Weaving can also be performed at design time (static) or at runtime (dynamic). Therefore, 3 combinations can be considered, ie, (1) both quantification and weaving are performed statically, (2) one of them is performed statically and the other one dynamically, and (3) both quantification and weaving are performed dynamically. If both quantification and weaving are performed statically, the language cannot take into consideration runtime information since none of them is performed at runtime. This is an issue (indicated by *R* in Figure 5) because certain information about a process could be available only at runtime when the user decides about data values. For example, information about the data perspective of a process like the amount of a loan in a process for handling loan applications is available only at runtime.

Runtime information can be predicted at design time and captured through placeholders—which can be bound at runtime—if either quantification or weaving is performed at runtime. This is a sort of underspecification technique commonly used to address flexibility in business process modeling.[24] If quantification is performed at design time and weaving at runtime, the process designer can define placeholders for predicted scenarios. For example, placeholders should be added after each task to check if the amount of a loan is exceeding a certain threshold at runtime. Note that several placeholders could need to be specified in the process model since the designer does not know exactly when the condition will be valid (this issue is indicated by *P* in the figure). If quantification is performed at runtime and weaving at design time, the process designer can define the rules only once. However, the weaving process will generate several placeholders anyway to capture these rules and check if the condition is valid.

To summarize, as it can be seen from Figure 5

- the clear-box quantification suffers from the problem of being specific to only 1 notation;
- if quantification and weaving are both performed at design time, then it is not possible to take runtime information into consideration;
- if either quantification or weaving is static and the other one is dynamic, then a large amount of placeholders needs to be added to the process models.

As it can be seen in the figure, the combination of black-box dynamic quantification with dynamic weaving is the best choice for process modeling and enactment with AO-BPM. This choice also enables to change the quantification rules at runtime, so this choice is followed in this paper. In particular, to define a language to specify pointcuts that support our requirements, we propose a new approach in which the relation between core concern and cross-cutting concerns can be defined using declarative rules. Quantification in this approach is based on 2 join points in a process model, called *initiator* and *terminator*. An *initiator* is a join point in a process model that specifies the start point after which a cross-cutting concern can be executed. A *terminator* is a join point in a process model that specifies the end point before which a cross-cutting concern should be terminated.

Moreover, it should be considered that we can define 2 possibilities when enabling instances of cross-cutting concerns, ie, (1) instances of concerns can be created and enabled *every time* an initiator is executed or (2) instances of concerns can be created and enabled *once* between the execution of an initiator and the execution of a terminator. We define 4 declarative rules to support the definition of pointcuts to relate advices to the main process, called mandatory conditional response (mcr), mandatory conditional alternate response (mcar), optional conditional response (ocr), and optional conditional alternate response (ocar):

- mcr indicates that an advice must be executed *once* in the period between the execution of the initiator and the execution of the terminator.
- mcar indicates that an instantiation of the advice must be started *every time* the initiator is executed and every instantiation must be terminated before the execution of the terminator.
- ocr indicates that an advice can be executed at most *once* in the period between the execution of the initiator and the execution of the terminator.
- ocar indicates that an instantiation of the advice can be started *every time* the initiator is executed and every instantiation must be terminated before the execution of the terminator.

The graphical representations of these rules are illustrated in Figure 6. Note that *I*, *A*, and *T* represent the initiator, the advice name, and the terminator, respectively.

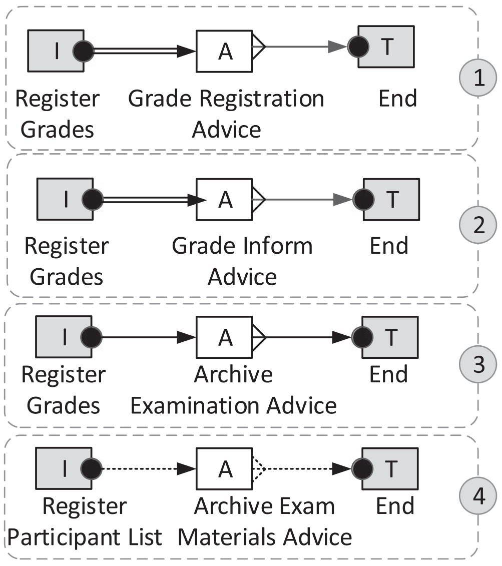

I
A
T
Register Grades
Grade Registration Advice
End
1
I
A
T
Register Grades
Grade Inform Advice
End
2
I
A
T
Register Grades
Archive Examination Advice
End
3
I
A
T
Register Participant List
Archive Exam Materials Advice
End
4

FIGURE 7 Specification of pointcuts for the running case

Figure 7 shows the application of these rules to our running case. Four declarative rules are defined to relate advices to the main process, which are annotated by numbers in the figure. The first rule indicates that the *Grade Registration* advice should be executed every time the grades are registered, and it should be completed before the process is ended. The second rule indicates that the *Grade Inform* advice should be executed every time the grades are registered, and it should be completed before the process is ended. The third rule indicates that the *Archive Examination* advice should be executed after registering the grades (once for all), and it should be completed before the process is ended. The fourth rule indicates that the *Archive Exam Materials* advice may be executed after registering the participant list, but—if so—it should be finished before the process is ended. The first 3 rules are mandatory rules, and the last rule is optional. Moreover, the first 2 rules are of type mcar, because both the security and the privacy concerns should be executed for every change in the grades. The third rule is of type mcr; the fourth rule is an ocr.

In this paper, we limit our pointcut language to support quantification over main concerns through names of activities. The language can easily be extended to support other perspectives like data and resource as defined in the literature.[16,25]

4 | WORKFLOW NETS

The proposed modeling approach requires a formal definition of syntax and semantics. The provided semantics will enable a Workflow Management System to interpret the models and weave them for an appropriate execution. In particular, we introduce the notion of Hybrid net, which is used to combine imperative core and cross-cutting concerns based on declarative rules. In this paper, we focus on the control-flow perspective while abstracting from other perspectives like data, resource, or time. Our formalism is based on the definition of workflow net provided in the literature.[26,27]

A workflow net is a particular type of Petri net. Petri nets consist of places and transitions, shown by circles and squares, respectively. Places show the substates of a system, and transitions show the actions that change the substates. Places can be connected to transitions, and transitions can be connected to places through directed arrows. The overall state of the net is specified by the distribution of tokens over places (a.k.a. marking). Each transition is enabled if there is 1 token in each input place, and the result of executing a transition is reflected by adding a token in each output place and removing a token from each input place (*firing rule*). For example, Figure 8 shows 2 Petri nets called Pc and Pcc, representing a core concern and a cross-cutting concern, respectively. The declarative rules specified in the lower right part of the figure indicate that we want this cross-cutting concern to be executed between B and D *and* between C and E of the main process.

The conventional definition of a Petri net does not allow for the desired execution semantics for these nets. In this example, if B and C are executed, they will produce tokens in their output places. We are also interested in producing a token in the input place of the cross-cutting concern, which will enable the execution of the concern. Moreover, we want to change the firing rules to not enable the terminator (D) until its corresponding cross-cutting concern has completed. However, it is not possible to distinguish between 2 tokens in Pcc, since they do not convey information about their initiator. This means that if one of these tokens is consumed by task S, producing a token in the output place, we do not know if we should enable D or not (the situation is shown in Figure 8). For this reason, we need to add information to tokens about their initiator. In addition, as discussed later, a unique id needs to be assigned to each token, which is used to ensure safeness and soundness properties.

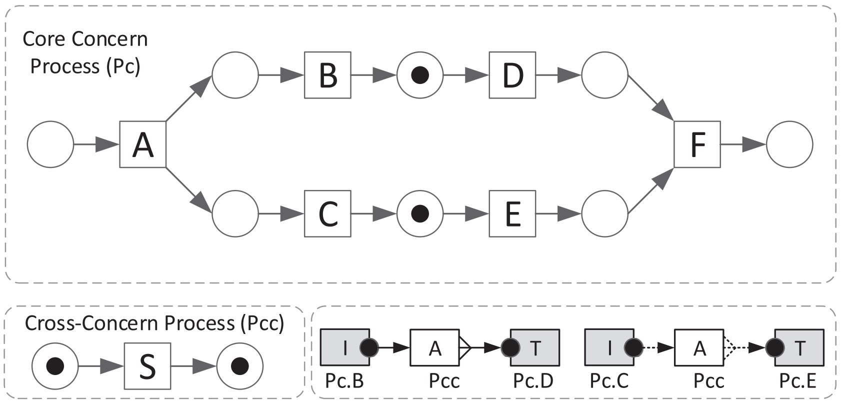


FIGURE 8 Petri net examples of core concern and a cross-cutting concern

4.1 | Colors in Petri nets

In the following definitions, we adapt the standard concepts related to Petri nets and workflow nets[26,27] to include the notion of colored tokens. These definitions will be used as background to introduce Hybrid nets in the following section.

Definition 1. P/T-nets: A Place/Transition net (P/T-net) is a tuple $N = (P^N, T^N, F^N)$,[†] where P^N is a finite set of *places*; T^N is a finite set of *transitions*, such that $(P^N \cap T^N = \emptyset)$; and $F^N \subseteq (P^N \times T^N) \cup (T^N \times P^N)$ is a set of *directed arcs*, called the flow relation.

A preset of an element $x \in P^N \cup T^N$ in the net N is defined as $\overset{N}{\bullet}x = \{y \in (P^N \cup T^N) | (y, x) \in F^N\}$. A postset of an element $x \in P^N \cup T^N$ in the net N is defined as $x\overset{N}{\bullet} = \{y \in (P^N \cup T^N) | (x, y) \in F^N\}$. P, T, and F represent the universe of all places, transitions, and flows, respectively.

Definition 2. Colored tokens: A colored token is a pair $\mathbb{T} = (\{\beta\} \cup T, \mathbb{N})$. The first value of the tuple represents the initiator. The value β will be used later for defining a base net. The second value represents an instance number for the token.

Definition 3. Marked P/T-nets: A marked P/T-net is a tuple (N, s^N), where $N = (P^N, T^N, F^N)$ is a P/T-net; and s^N is a function (called bag) over P^N denoting the marking of the net, ie, the distribution of tokens in the net. The set of all marked P/T-nets is denoted with $\mathcal{N}$.

A marking is a function from P and $\mathbb{T}$ to the set of natural numbers $\mathbb{N}$. The sequence brackets are used for specifying the bag, eg, $[(p_1, (\beta, 1))^1, (p_2, (t4, 2))^3]$ denotes the bag with 1 token $(\beta, 1)$ in place p_1 and 3 tokens $(t4, 2)$ in place p_2. Considering 2 bags X and Y, the sum of 2 bags $(X + Y)$, the difference $(X - Y)$, the presence of an element in a bag $(p_1, (\beta, n \in \mathbb{N})) \in X$, and the notion of subbags $(X \leq Y)$ are considered as well. Having s be the set of all markings, *assign* is a function $assign : \{\beta\} \cup T \rightarrow \mathbb{N}$, such that $\forall v_1 \in \{\beta\} \cup T, \quad \forall v_2 \in \mathbb{N}, \quad \forall p \in P : (p, (v_1, v_2)) \in s \rightarrow n \neq v_2$.

Definition 4. Firing rules: Let $(N = (P^N, T^N, F^N), s^N)$ be a marked P/T-net. The transition $t \in T^N$ is enabled for the token with the value $a \in \mathbb{T}$, denoted $(N, s^N)[t, a\rangle$, *iff* $\forall p \in \overset{N}{\bullet}t : (p, a) \in s^N$. The firing rule $-[-, -\rangle- \subseteq \mathcal{N} \times (T, a) \times \mathcal{N}$ is the smallest relation satisfying for any $(N, s^N) \in \mathcal{N}$ and any $t \in T^N$, $(N, s^N)[t, a\rangle \Rightarrow (N, s^N)[t, a\rangle(N, s^N) - \{(p, a) | \forall p \in \overset{N}{\bullet}t\} + \{(p, a) | \forall p \in t\overset{N}{\bullet}\})$.

Definition 5. Reachable markings: Let (N, s_0) be a marked P/T-net in $\mathcal{N}$, where s_0 is the initial marking. A marking s' is reachable from s_0 *iff* there exists a sequence of enabled transitions whose firing leads from s_0 to s'. The set of all reachable markings of (N, s_0) is denoted with $[N, s_0\rangle$.

Definition 6. Firing sequences: Let (N, s_0^N) with $N = (P^N, T^N, F^N)$ be a marked P/T-net. A sequence $\sigma \in T^{N^*}$ [‡] is called a firing sequence of (N, s_0^N) *iff*, for some natural number $n \in \mathbb{N}$ and a token with value a, there exist markings $s_1^N, \ldots, s_n^N$ and transitions $t_1, \ldots, t_n \in T^N$ such that $\sigma = t_1 \ldots t_n$ and, for all i with $0 \leq i < n$, $(N, s_i^N)[t_{i+1}, a\rangle$ and $s_{i+1}^N = s_i^N - \{(p, a) | \forall p \in \overset{N}{\bullet}t\} + \{(p, a) | \forall p \in t\overset{N}{\bullet}\}$

Definition 7. Connectedness: A net $N = (P^N, T^N, F^N)$ is weakly connected, or simply connected, *iff*, for every 2 nodes x and y in $P^N \cup T^N$, $x(F^N \cup F^{N^{-1}})^* y$. The net N is strongly connected *iff*, for every 2 nodes x and y, $xF^{N^*}y$.

Definition 8. Boundedness, safeness: A marked net $(N = (P^N, T^N, F^N), s^N)$ is bounded *iff* the set of reachable markings $[N, s^N\rangle$ is finite. It is safe *iff*, for any $s' \in [N, s^N\rangle$ and any $p \in P^N$ and any $a \in \mathbb{T}$, $s'(p, a) \leq 1$. Note that safeness implies boundedness.

[†] The superscript N for a set specifies the net that contains the elements of the set.
[‡] R^{-1} is the inverse and R^* is the reflexive and transitive closure of a relation R.

Definition 9. Dead transitions, liveness: Let $(N = (P^N, T^N, F^N), s^N)$ be a marked P/T-net. A transition t is dead in (N, s^N) *iff* there is no reachable markings $s' \in [N, s^N\rangle$ for a token $a \in \mathbb{T}$ such that $(N, s'^N)[t, a\rangle$. (N, s^N) is live *iff*, for every reachable marking $s' \in [N, s^N\rangle$ and $t \in T^N$, there is a reachable marking $s'' \in [N, s'\rangle$ such that $(N, s'')[t, a\rangle$.

In accordance with the above definitions, we also use a slightly different definition for workflow nets.

Definition 10. Workflow nets: Let $N = (P^N, T^N, F^N)$ be a P/T-net and $\bar{t}$ an element not in $P^N \cup T^N$. N is a workflow net (WF-net) *iff* (1) P^N contains an input place i such that $\overset{N}{\bullet} i = \emptyset$; (2) P^N contains an output place o such that $o \overset{N}{\bullet} = \emptyset$, and (3) $\bar{N} = (P^N, T^N \cup \{\bar{t}\}, F^N \cup \{(o, \bar{t}), (\bar{t}, i)\}$ is strongly connected.

Definition 11. Soundness: Let $N = (P^N, T^N, F^N)$ be a WF-net with input place i and output place o. N is sound for any $n \in \mathbb{N}$ *iff*$(N, [(i, (\beta, n))])$ is safe, for any marking $s \in [N, [(i, (\beta, n))]\rangle$, $o \in s$ implies $s = [(o, (\beta, n))]$, for any marking $s \in [N, [(i, (\beta, n))]\rangle$, $[(o, (\beta, n))] \in [N, s\rangle$, and $(N, [(i, (\beta, n))])$ contains no dead transitions.

We only extend the definition of P/T-nets and WF-nets with colored tokens. Therefore, the verification of soundness that is given in van der Aalst[26] is also valid for this extension, since tokens with different colors are treated separately from each other.

5 | HYBRID NETS

In this section, we introduce Hybrid nets, which enable us to define core and cross-cutting concerns as well as the pointcuts to connect them. Cross-cutting concerns can be defined through *workflow nets*, and core concerns are defined through *base nets*. The introduction of Hybrid nets allows us to relate these nets to each other through declarative rules. We also specify the execution semantics for these nets and discuss why they are sound.

Definition 12. Base nets: A base net (B-net) is a workflow net BN with 1 start and 1 end transition such that there exists a workflow net $WF = (P^{WF}, T^{WF}, F^{WF})$, $BN = (P^{WF} \cup \{i'^{BN}, o'^{BN}\}, T^{WF} \cup \{start^{BN}, end^{BN}\}, F^{WF} \cup \{(i'^{BN}, \mathrm{start}^{BN}), (\mathrm{start}^{BN}, i^{WF}(o^{WF}, \mathrm{end}^{BN}), (\mathrm{end}^{BN}, o'^{BN})\})$, where i^{WF} and o^{WF} are the input and output places of the workflow net WF, respectively, $end^{BN}, start^{BN} \notin T^{WF}$, and $i'^{BN}, o'^{BN} \notin P^{WF}$.

The start and the end transitions are introduced to enable the definition of cross-cutting concerns that can run in parallel to the main process from the start until the end of the process. Figure 9 shows an abstract representation of a base net. As it can be seen, it has a start transition that connects the input place of the base net to the input place of the workflow net and an end transition that connects the output place of the workflow net to the output place of the base net.

Definition 13. Hybrid nets: A Hybrid net (H-net) is a tuple $H = (BN, CN = \bigcup_{i=1..n} \{N_i = (P^{N_i}, T^{N_i}, F^{N_i})\}, \phi, \chi)$, where BN is a B-net with a start event ($start^{BN}$) and an end event (end^{BN}), CN is a set of WF-nets (called concern nets) such that $\{BN\} \cap CN = \emptyset$, $n \in \mathbb{N}$, ϕ is a function from transitions in BN to the set of concern nets, ie, $\phi : T^{BN} \rightarrow 2^{CN}$, and χ is a function from transitions in BN and their related concern nets to tuples composed of other transitions in BN and a member of the relation set $RS = \{mcr, mcar, ocr, ocar\}$, ie, $\chi : t_1 \in T^{BN} \times \phi(t_1) \rightarrow (t_2 \in T^{BN}, RS)$, where $t1 \neq end^{BN} \wedge t2 \neq start^{BN}$. $t1$ is called initiator, and $t2$ is called terminator. *mcr*, *ocr*, *mcar*, and *ocar* are acronyms that stand for *mandatory conditional response*, *optional conditional response*, *mandatory conditional alternate response*, and *optional conditional alternate response* relations, respectively.

H-nets enable to relate core and cross-cutting concerns through specific relation types. Figure 10 shows an abstract graphical representation of a H-net. It consists of a base net (B1) and a cross-cutting concern (C1). To connect these 2 nets, we need some rules. Imagine that we are interested in performing the cross-cutting concern in parallel to the process model, ie, starting after $Start^{B1}$ and ending before end^{B1}. This relation is illustrated using the dashed flows in the figure. An mcr rule relates $Start^{B1}$ to C1, ie, $\phi(Start^{B1}) = C1$. It enforces the concern to be finished before end^{B1}, ie, $\chi(Start^{B1}, C1) = (end^{B1}, mcr)$. The graphical representation of this rule is illustrated in the figure between the 2 nets.

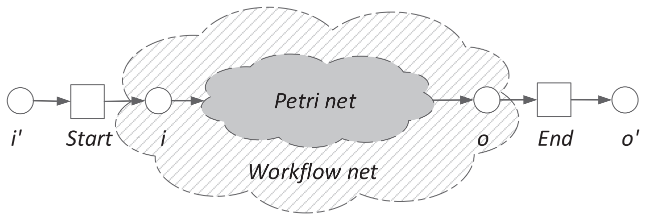


FIGURE 9 An abstract graphical representation of a B-net

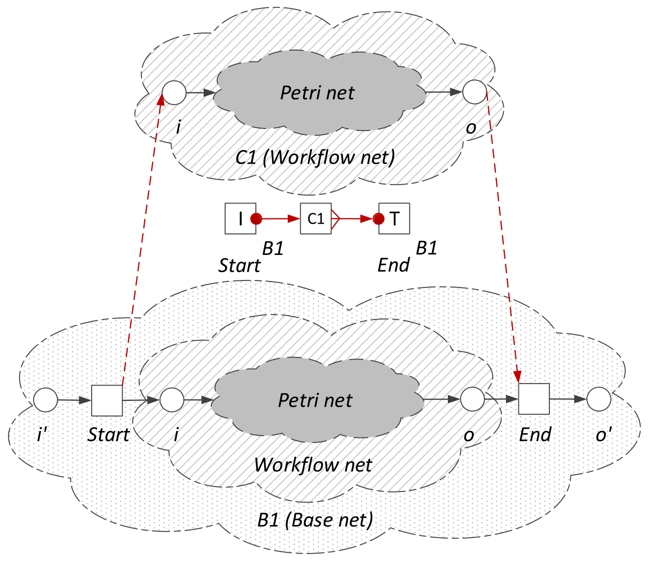


FIGURE 10 An abstract graphical representation of an H-net

Definition 14. **Marked H-nets:** A marked H-net is a tuple (HN, s), where HN is an H-net, ie, $H = (BN, CN, \phi, \chi)$, and s is a bag over all places of WF-nets in HN, ie, $s = \bigcup_{c \in CN} s^c \cup s^{BN}$, where (BN, s^{BN}) and $\forall c \in CN, (c, s^c)$ are marked P/T-nets.

Marked H-nets indicate the state of a system based on the state of its separated modules.

Here, we explain B-net enabling and firing rules informally with the example net introduced in Figure 8. The top net in Figure 11 shows the third marking of the net, where there is a token in the input places of transitions B and C. A transition in the B-net is enabled if (1) there is at least 1 token in each input place and (2) there is no incomplete mandatory concern for which this transition is defined as the terminator. The second condition is satisfied if there is no token (corresponding to the mandatory concerns for which the transition is defined as the terminator) left in any place (except the output place) of the concern nets. In our example, transitions B and C are enabled (represented with a bold border) because (1) there is 1 token in their input place and (2) these transitions are not terminators of any mandatory concern.

An enabled transition can be fired. If a transition in the B-net is fired, it consumes and produces a set of tokens. The set of consumed tokens includes (1) a token from each input place and (2) all tokens corresponding to the concerns for which the transition is defined as the terminator from all concern nets. The set of produced tokens includes (1) a token in each output place and (2) a token in the input places of the concern nets for which the transition is defined as an initiator.

The second net in the figure shows the 5th marking, where both transitions (B and C) are fired. Firing these transitions results in a token in each of their output places. However, they also produce a token in the input place of the concern net since B and C are both initiators of declarative rules corresponding to the concern. As it can be seen in the figure, the first value of the token is changed from β to the name of the transition, and the second value is a unique number. In this way, we can distinguish between these 2 tokens. Now, transition S is enabled in the concern. Transition E in the base net is also enabled. This transition is enabled because it is not defined as the terminator for a mandatory rule. Indeed, it is defined as the terminator for an optional rule. Transition D is not enabled because it is defined as the terminator of a mandatory rule, and there is a token in a place that is not the output place of the corresponding concern (ie, the input place of the concern).

The last net shows the next marking, where transition E is fired. As it can be seen, the token in its input place is removed. In addition, all tokens (corresponding to the optional concern for which E is the terminator) in the concern net are also removed. As a result, it is not possible to execute the optional concern anymore. Now, the only enabled transition is S in the concern net. If the transition is fired, it consumes the token from its input place and produces a token in its output place (see the 7th marking in Figure 12). Therefore, transition D becomes enabled since there is a token in its input place and only 1 token corresponding to the mandatory concern for which D is the terminator in the output place of the concern net. The firing of this transition results in enabling transition F (see the 8th marking). The formal definitions of enabling and firing rules are defined as follows.

Definition 15. **Concern nets enabling and firing rules:** for any concern net $c \in CN$, enabling and firing rules are the same as for P/T-nets.

Definition 16. **B-net enabling rule:** In the B-net BN, the transition $t \in T^{BN}$ is enabled for the token with value $a = (\beta, n)$, denoted $(BN, s^{BN})[t, a\rangle$, *iff*

- $\forall p \in \overset{BN}{\bullet} t : (p, a) \in s^N$, and
- $\forall t_1 \in T^{BN} \forall c \in \phi(t_1) \left((\chi(t_1, c) = (t, mcr) \vee \chi(t_1, c) = (t, mcar)) \rightarrow \nexists t'' \in T \forall n \in \mathbb{N} \left(c, [\bigcup_{p \in P^c} \{(p, (t_1, n))\} \cap s^c] \right) [t'', (t_1, n)\rangle \right)$

Definition 17. **B-net firing rule:** the firing rule $-[-,-\rangle- \subseteq \mathcal{N} \times (T^{BN}, a) \times \mathcal{N}$ is the smallest relation satisfying for any $(BN, s^{BN}) \in \mathcal{N}$ that

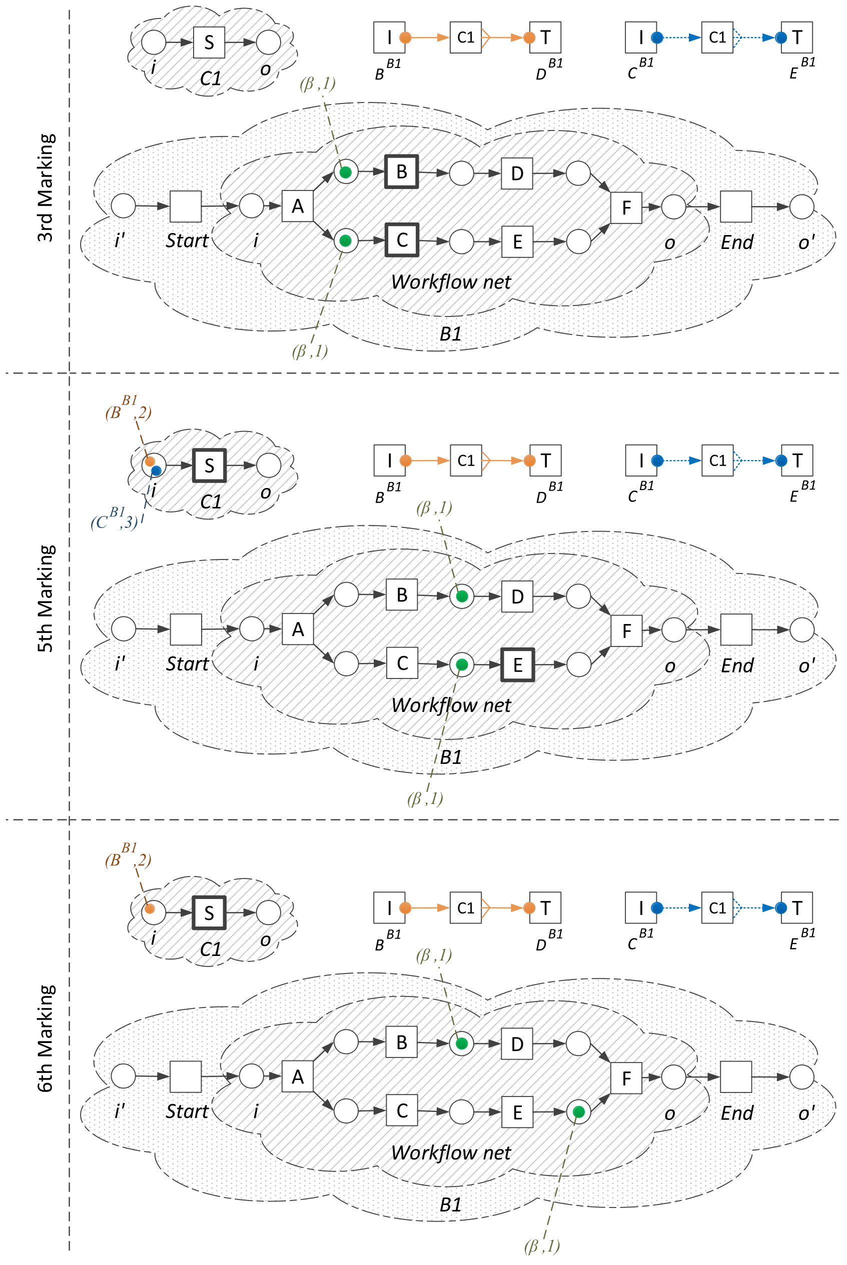


FIGURE 11 An example for enabling and firing rules in an H-net

- for any $t \in T^{BN} \backslash end^{BN}$:

$$(BN,s)[t,a\rangle \Rightarrow (BN,s^{BN})[t,a\rangle(BN,s^{BN}) - \left(BN,[\underset{p\in\bullet t}{\overset{BN}{\cup}}\{(p,a)\}]\right) + \left(BN,[\underset{p\in t\bullet}{\overset{BN}{\cup}}\{(p,a)\}]\right)$$
$$-\sum\nolimits_{n\in CN,\forall t_1\in T^{BN}\chi(t_1,n)=(t,m\in RS)\wedge\exists s^n\in s(n,s^n)\in\mathcal{N}}(n,s^n)^{¶}$$
$$+\sum\nolimits_{n\in\phi(t),\forall s^n\in s(n,s^n)\notin\mathcal{N}}\sum_{\frac{t_1\in T^{BN},}{\chi(t,n)=(t_1,mcr)\vee\chi(t,n)=(t_1,ocr)}}\sum\nolimits_{p\in P^n,\overset{n}{\bullet}p=\emptyset}(n,[(p,(t,assign(t)))])^{\|}$$
$$+\sum\nolimits_{n\in\phi(t)}\sum_{\frac{t_1\in T^{BN},}{\chi(t,n)=(t_1,mcar)\vee\chi(t,n)=(t_1,ocar)}}\sum\nolimits_{p\in P^n,\overset{n}{\bullet}p=\emptyset}(n,[(p,(t,assign(t)))])^{**}$$

- for a transition $t = end^{BN}$ with an output place o', the firing rule for the token with value $a = (\beta, n)$ is: $(BN,s)[t,a\rangle \Rightarrow (BN,s)[t,a\rangle(BN,s^{BN}) - \sum_{n\in\{BN\}\cup CN,\forall s'\in s}(n,s')\in\mathcal{N}^{††} + (BN,[(o',a)])$

Definition 18. H-net firing rules: The firing rule of a marked H-net $\left(H = \left(BN = (P^{BN},T^{BN},F^{BN}),CN,\phi,\chi\right), s = s^{BN}\cup s^{CN}\right)$ consists of the union of the firing rules for its B-net and concerns' nets.

¶ to remove all initiated tokens from related concerns
‖ to add tokens in mcr and ocr
** to add tokens in mcar and ocar
†† to remove all tokens from all nets

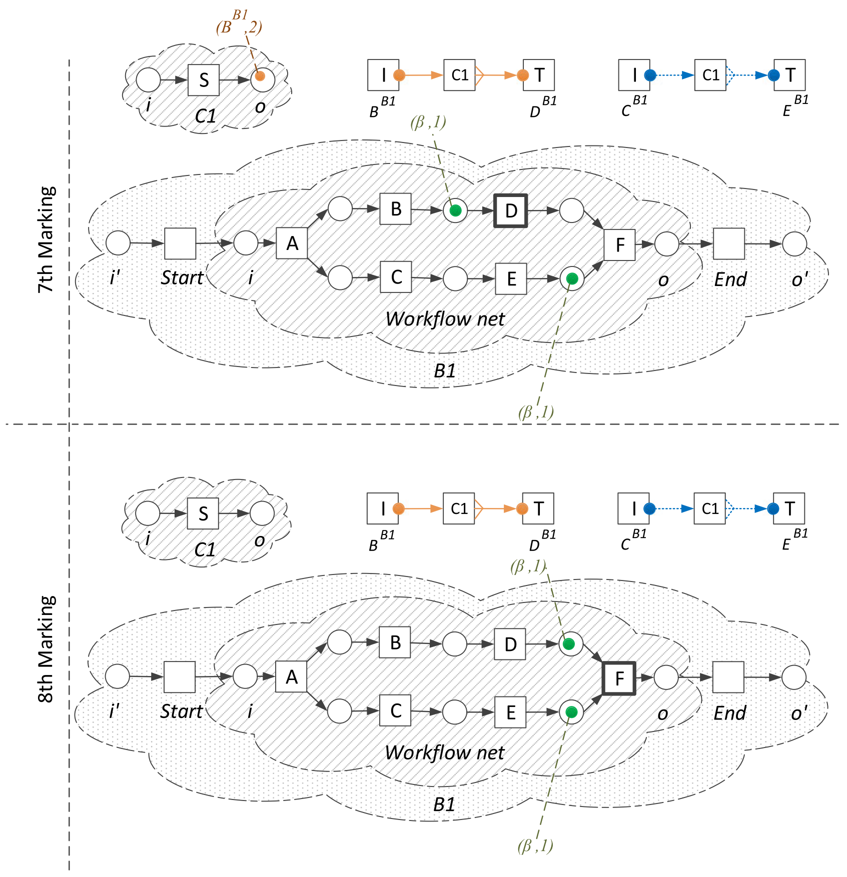


FIGURE 12 An example for enabling and firing rules in an H-net

Definition 19. Soundness: Let $H = \left(BN = (P^{BN}, T^{BN}, F^{BN}), CN, \phi, \chi\right)$ be an H-net, where i and o denote the input and output places of BN, respectively, and $start^{BN}$ and end^{BN} denote start and end event transitions of BN respectively, and a is a token with the value $(\beta, 1)$). H is sound *iff*

- safeness: $(HN, [(i, a)])$ is safe,
- proper completion: for any marking $s \in [H, [(i, a)]\rangle$, $(o, a) \in s$ implies $s = [(o, a)]$),
- option to complete: for any marking $s \in [H, [(i, a)]\rangle$, $[(o, a)] \in [H, s\rangle$, and
- absence of dead transition: $(H, [(i, a)])$ contains no dead transitions.

An H-net is always sound because

1. *it is safe* since (*a*) there will not be 2 tokens with the same value relating to the same case in any place in the concern nets because each marked concern net is initiated with a different token number by the *assign* function and each concern net is a WF-net, which is safe; (*b*) there will not be 2 tokens with the same value relating to the same case in any place in the B-net since other concerns do not add tokens in the places in the B-net, which is also a WF-net and safe.
2. *it finishes properly* since end^{BN} will remove all tokens from all places in all nets and will produce 1 token in the output place.
3. *it always has the option to complete* since (*a*) for optional concerns, they will not affect the execution semantics of the B-net, so the end marking is reachable from any marking in these nets; (*b*) for mandatory concerns, they are initiated with a marking that puts a token in their input place, and since they are WF-nets, they will finish with a token in the output place. This token will be consumed by the terminator or by the end transition in the B-net, so any marking in these nets will reach the end; (*c*) the end marking is reachable from any marking in the B-net, because the B-net is a WF-net that needs a proper completion of the mandatory concern nets to continue, and the mandatory concern nets are, in turn, WF-nets, which will be completed properly. Thus, the B-net will reach the end as well.
4. *it has no dead transition* since for any transition $t \in T^{BN}$, there is a marking that is reachable from the initial marking to enable it because (*a*) if there are no concern nets initiated by a transition in the B-net, there is a marking that will enable the transition because the B-net itself is a WF-net and all transitions have a reachable marking that enables them; (*b*) if there are concern nets initiated by a transition in the B-net, (*i*) if they are optional concerns, they do not affect the execution of the transitions in the B-net, and (*ii*) if they are mandatory concerns, they are also WF-nets, which are initiated by a token in their input places, so they will eventually finish and they will not make any reachable marking in the B-net unreachable. Thus, there will be no dead transitions in an H-net.

6 | IMPLEMENTATION

This section briefly describes the design choices for the implementation of the artifacts developed as a proof of concept to support the defined notation and semantics. As explained earlier, we selected a black-box dynamic quantification and a dynamic weaving to avoid problems that other choices introduce. This choice requires a system that (1) provides information about process instances at runtime and (2) enables altering process instances at runtime. These functionalities are supported by a standard called Workflow Reference Model,[28] which is developed by the Workflow Management Coalition.

6.1 | The architecture

Yet Another Workflow Language[3] is an example of Workflow Management System, which is designed and implemented according to the Workflow Reference Model. It is open-source and is designed on the basis of a Service-Oriented Architecture, which makes extending it easy. It also supports workflow patterns, which makes it a good choice for designing and enacting business processes. Therefore, we defined our artifacts on top of YAWL.

Figure 13 depicts the architecture of the whole system. In this figure, the *YAWL Workflow Engine* corresponds to the *Workflow Enactment Service* of the Workflow Reference Model. The *YAWL Editor* corresponds to the *Process Definition Tools*. The *Resource Service* and the *Weaver Service* are the *Workflow Client Applications*.

The user can design core and cross-cutting concerns using the YAWL Editor. The core concerns are stored in the Process Repository, and the cross-cutting concerns are stored in the Advice Repository. The user can also define the relations between these concerns (the pointcuts) using a Rule Editor, which stores the pointcuts in the Pointcut Repository. The YAWL Workflow Engine is responsible for enacting the core and the cross-cutting concerns. On top of it, we implemented the Weaver Service, based on the enabling rules and firing rules defined in Section 5, which enforces the correct execution of these models. The Resource Service provides an interface to the users for the execution of instances of each task at runtime. This service authorizes the user according to the access rights, which are defined in the Org(organizational) Model. It also stores the result of any execution in an Event Log. The tasks that can be performed by a user are shown to her or him through a worklist.

6.2 | The modeling toolset

The YAWL Editor supports the definition of both imperative core and cross-cutting concerns. To support the definition of pointcuts, we implemented a designer called *Rule Editor*. Figure 14 shows a screenshot of the rule editor and the YAWL Editor for our running case.** The top window shows the YAWL Editor through which the core and cross-cutting concerns are designed. The bottom window shows the Rule Editor that enables the definition of pointcuts.

**‡‡All artifacts can be downloaded fromhttp://people.dsv.su.se/~aj/WeaverService/

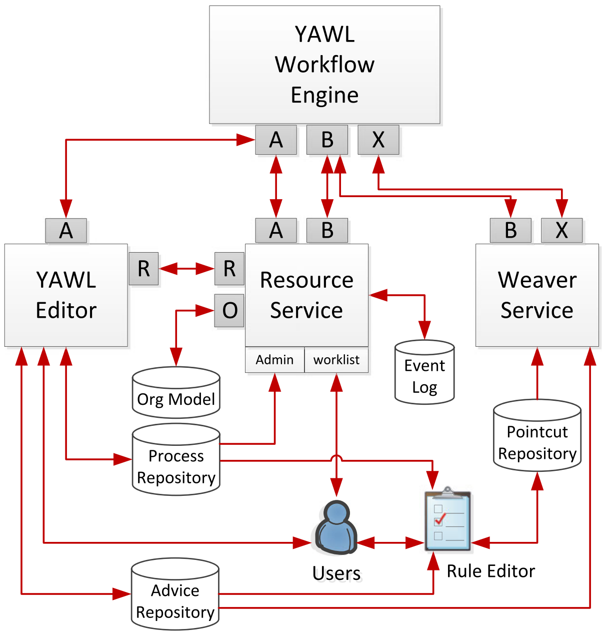


FIGURE 13 The architecture

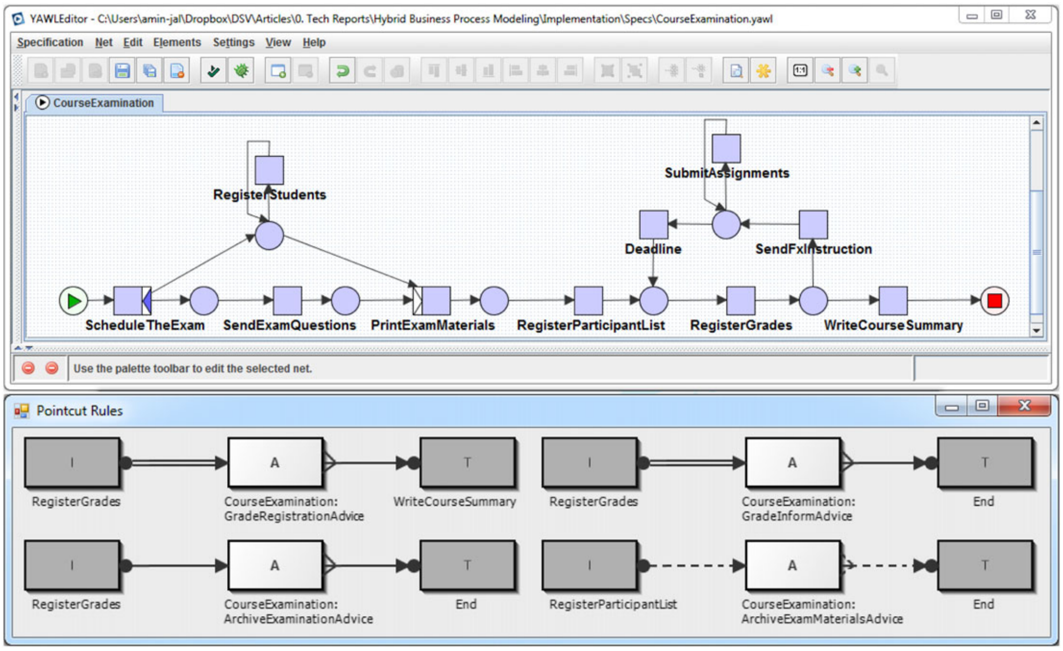
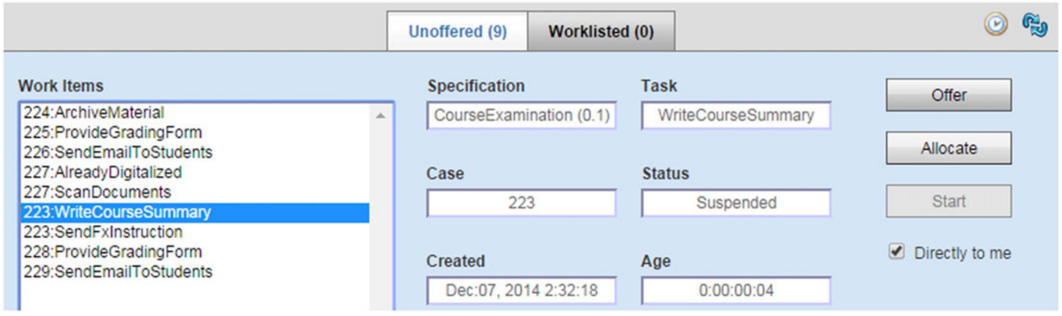

YAWLEditor - C:\Users\amin-jal\Dropbox\DSV\Articles\0. Tech Reports\Hybrid Business Process Modeling\Implementation\Specs\CourseExamination.yawl
Specification Net Edit Elements Settings View Help
CourseExamination
RegisterStudents
SubmitAssignments
Deadline
SendFxInstruction
ScheduleTheExam
SendExamQuestions
PrintExamMaterials
RegisterParticipantList
RegisterGrades
WriteCourseSummary
Use the palette toolbar to edit the selected net.
Pointcut Rules
I RegisterGrades
A CourseExamination: GradeRegistrationAdvice
T WriteCourseSummary
I RegisterGrades
A CourseExamination: GradeInformAdvice
T End
I RegisterGrades
A CourseExamination: ArchiveExaminationAdvice
T End
I RegisterParticipantList
A CourseExamination: ArchiveExamMaterialsAdvice
T End

FIGURE 14 The YAWL and Rule Editors



Work Items
224:ArchiveMaterial
225:ProvideGradingForm
226:SendEmailToStudents
227:AlreadyDigitalized
227:ScanDocuments
223:WriteCourseSummary
223:SendFxInstruction
228:ProvideGradingForm
229:SendEmailToStudents

Specification	CourseExamination (0.1)
Task	WriteCourseSummary
Case	223
Status	Suspended
Created	Dec:07, 2014 2:32:18
Age	0:00:00:04

[✓] Directly to me

FIGURE 15 Enacting the process models

6.3 | The enactment service

To support the enactment of the models, we implemented a service in YAWL, called Weaver Service. Figure 15 shows a worklist, in which all instances of the tasks that can be executed in the core and the cross-cutting concerns can be observed (the user administrator can see the tasks assigned to all the users of the system). The task *Write Course Summary* cannot be started in this case (it is suspended) because the Weaver Service disables it. This is because 2 instances of the mandatory advice *Grade Registration* (see tasks with ids 225 and 228) should be finished before this task (see the top left rule in Figure 14).

The models are enacted according to the provided semantics. In particular, enacting these models using the implemented artifacts allows us to (1) enable the separation of cross-cutting concerns such that these can be executed in parallel to parts of the main process model and (2) enable the definition of both optional and mandatory cross-cutting concerns.

7 | EVALUATION

This section reports the result of evaluating our approach. This evaluation aims at studying the potential of the proposed approach to be later used in real domains. Moody[29] proposed and investigated a method to evaluate the information systems' artifacts based on the technology acceptance model[30] and methodological pragmatism. In this method, the *actual usage* of an artifact is defined as a variable that depends on the *intention to use* of the artifact by users. The intention to use depends on the perceptions of the users for both *efficiency* and *effectiveness*, called *perceived ease of use* and *perceived usefulness*, respectively. The efficiency is defined as reducing the effort to complete a task, and the effectiveness is defined as improving the quality of the result.

The method is used for evaluating different artifacts in the business process management area (eg, the patterns to deal with complexity in business process models[11,31]). Therefore, we evaluated the *perceived ease of use* and *perceived usefulness* of our artifact using this method. To the best of our knowledge, this is the first evaluation about perceived ease of use and usefulness of an AO-BPM approach.

To evaluate the efficiency and effectiveness of our artifact, we set up a workshop with 28 students, of which 20 were bachelor and 8 were master level students. These students have been studying a course on business process management in which they learn about business process modeling, analysis, configuration, implementation, monitoring, and mining. The course is offered to both bachelor and master students. Bachelor students can take this course in their final year of study as an advance course in the area of Computer and Systems Sciences. They experienced tools like WoPeD for designing and analyzing business process models using Petri nets; Signavio for designing BPMN models; YAWL for modeling, analysis, and enacting business processes using YAWL Notation; and ProM for performing process mining activities. Therefore, we believe that they had a good background to participate in the workshop and experience our approach, and they can be considered good proxies for novice process analysts. Among these students, 11 participants were female, and 17 were male. The age of student varies. There were 15 students between 20 and 24, 8 students between 25 and 29, 3 students between 30 and 34, and 2 students between 35 and 39.

On the basis of our teaching experience, students learn more quickly when they perform process design activities in groups of 2 or 3. Thus, in the workshop, students were divided into 10 groups randomly chosen, ie, 8 groups of 3 and 2 groups of 2. We asked students to design a process model for our running case. Then, we asked them to apply a change in the model. Later, we introduced the new technique defined in this paper, and we asked them to design the process using this technique. Afterwards, we asked them to apply the change in the new model. Finally, we asked every participant to fill a survey individually. The survey questions are adapted from the questionnaire given in Davis.[30] This questionnaire was chosen with the aim of measuring the perceived usefulness and easy to use of our approach based on the framework defined in this paper. The details about questions can be found in the appendix. The survey contains 2 sets of questions to measure if our proposed approach improves usefulness (first set) and ease of use (second set) with respect to a standard modeling approach. Students could choose answers based on scales from *extremely likely* to *extremely unlikely*, which are mapped to numbers from 7 to 1, respectively.[30]

Figure 16 shows how our approach is perceived in terms of usefulness and ease of use. The medians for both measures are above 5, which shows that most participants found the approach useful and easy to use. There were also negative comments though. One comment on the approach was about model fragmentation in a way that "it [(this approach)] makes the models more sparse." This comment applies to the whole modularization techniques that encapsulate parts of models to support reusability and change management. One student mentioned that the models could be "a bit hard to understand." According to a comment raised by one of the participants, it is challenging to "learn all the different arrows." These comments are about the graphical notation of our approach that has sometimes been found not easy to be understood. The graphical notation can be changed in the future to make the modeling notation easier to understand. However, this problem might also be applied to declarative business process modeling in general, which requires a similar investigation to improve understandability.

Overall, the evaluation provides indications that the method is useful and easy to use for a novice audience. This raises our hopes that the method would be adopted by process analysts in practice. As shown in Jalali et al,[16] AO-BPM can help designers to deal with the complexity of business

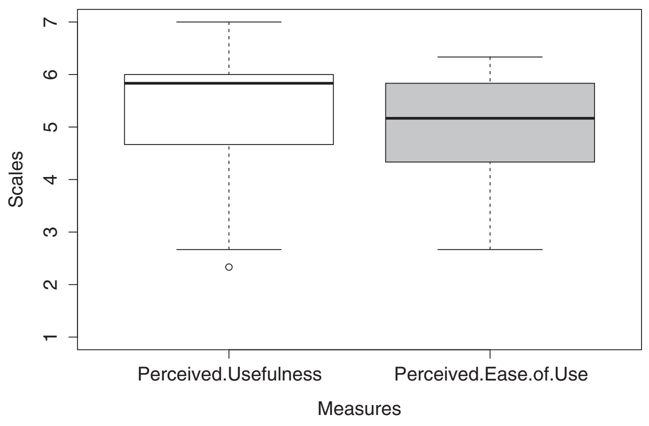


FIGURE 16 Perceived Usefulness and Ease of Use of our approach in General

process models. On the basis of our evaluation, we hope that the proposed modeling approach can provide an additional instrument to support the design of complex process models. Currently, there is a preliminary stage of a project at Stockholm University to find suitable subjects to assess the applicability of applying aspect mining[32] in the banking domain. Some of the identified process managers will be invited to assess other techniques in the context of AO-BPM and, in particular, the modeling approach presented in this paper.

8 | DISCUSSION

This section discusses the possibility of extending other business process modeling notations (particularly BPMN) based on our approach. In addition, it discusses threats to validity of our evaluation.

8.1 | The proposed solution based on BPMN

We present our modeling approach using Petri nets, as they provide the formal foundations of several imperative languages and are one of the standard ways to model and analyze business processes. However, since BPMN is a widely adopted business process modeling language in industry, we also show how our approach can be used when the process models are represented using BPMN.

Figures 17 and 18 show the *Course Examination* process and the related cross-cutting concerns expressed using BPMN 2.0. The core concern complies with the BPMN 2.0 specification, which means that the process designer does not need to know about the cross-cutting concerns when modeling the core concern (there are no elements referring to the cross-cutting concerns in the core concern). This shows that the obliviousness property is valid. The cross-cutting concerns in our approach are also imperative models, and they can be designed using AOBPMN 2.0.[33] Each cross-cutting concern has a *conditional start event* that triggers the concern if a given condition is evaluated to true. The conditions are specified through pointcuts. The pointcuts can be specified with the language we propose in this paper with the same semantics.

Note that modeling core and cross-cutting concerns by only using BPMN 2.0 can still introduce the scattering and tangling problems that were discussed at the beginning of the paper for vertical and horizontal decomposition techniques. For example, the use of event subprocesses with a non-interrupting conditional start event to represent cross-cutting concerns does not solve the scattering problem since all process models that are connected to a concern should incorporate the same subprocess that represents that concern. In case of callable subprocesses (ie, subprocesses that can be shared by different process models), the scattering problem is solved, but still the modeler should know at design time the places where the concern has to be incorporated and a change in how the subprocess should applies would enforce modelers to refine all invocations.

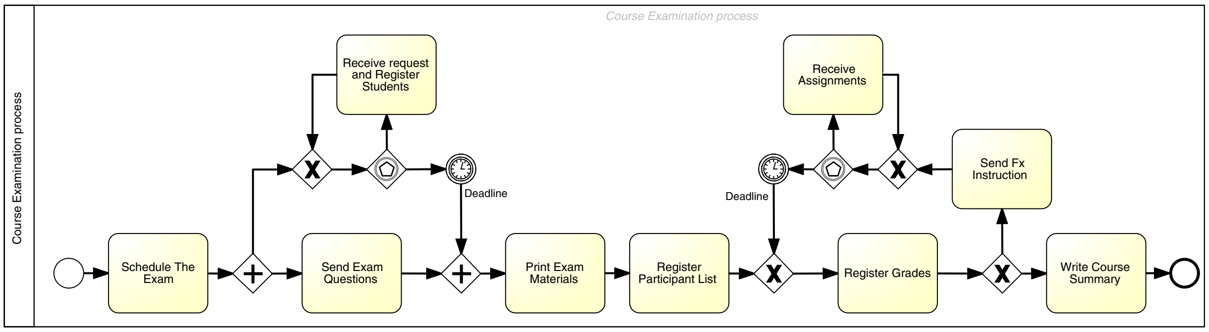


FIGURE 17 Course Examination process in BPMN Notation

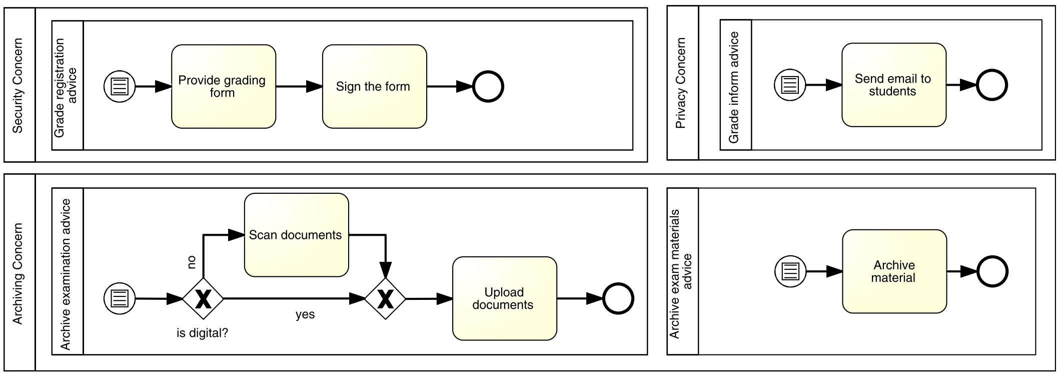


FIGURE 18 The cross-cutting concerns processes in AOBPMN Notation

8.2 | Threats to validity

The evaluation of usability and understandability of a modeling language is not trivial since it depends on the characteristics of the subjects involved in the evaluation like their competencies and experience. In our evaluation, this threat to validity is mitigated by the fact that, since hybrid approaches are new in the business process management field, there are very few people who have experience in modeling using these approaches. Therefore, it is reasonable to assume that all the subjects involved were not familiar with this type of approaches.

A second threat to validity of our evaluation is the fact that we only tested our approach on 1 group of subjects, ie, students. Thus, the evaluation cannot be generalized further to claim usability and understandability for other groups of subjects like process managers, business analysts, or industrial business designers. In spite of this, the probability of well-perceived usefulness due to the student's limited knowledge of other languages and modeling options is mitigated by the fact that students had, on average, good knowledge of different business process modeling notations like Petri nets, YAWL, EPCs, and BPMN.

Finally, we used a simple case in our evaluation, which could enhance the perceived ease of use of the new technique by participants. The ease of use could decrease when using more complex business processes due to the inherent complexity of the processes themselves. Therefore, it would be interesting to evaluate the ease of use of our approach when modelers are required to deal with more complex cases. In addition, the use of more sophisticated scenarios would also be valuable for further investigating the effectiveness of our approach to support modelers when dealing with complex process models.

9 | RELATED WORK

The idea of separation of cross-cutting concerns has been proposed and investigated in the programming area in the context of Aspect-Oriented Programming (AOP).[13,19] This idea has been growing and inspired 2 other paradigms, ie, requirement engineering[34] and service composition.[35-38]

Aspect-Oriented Service Composition inspired researchers to work on separation of concerns also in the context of business process models. Wang et al[39] propose an approach to encapsulate cross-cutting concerns of business processes into separated modules, but their approach does not support obliviousness property. Thus, the process modeler should know about cross-cutting concerns, and a placeholder should be considered for relating cross-cutting concerns to process models. The same approach has been followed by Shankardass.[40] Although these works could not propose an approach to separate cross-cutting concerns completely, they identified the problem and the need of solutions for such a problem.

Charfi et al.[41] proposed extensions of BPMN to support separation of cross-cutting concerns from business process models, called AO4BPMN. However, they defined neither a pointcut language nor a tool that identifies join points [Correction added on 12 February 2018, after first online publication: “quantifies” corrected to “identifies”]. Jabeen et al[42] extended AOBPMN by developing a pointcut language without developing a tool to support it. Cappelli et al[15] defined a pointcut language and developed a tool to support it by extending the Oryx Editor.[43] This extension is called CrossOryx.[15] None of these approaches defined the modeling notation formally, so Jalali et al[33] formalized the aspect orientation extension of BPMN.

Santos et al[44] discussed some open issues in modeling cross-cutting concerns, classified in 3 categories, ie, (1) how to model them (called method), (2) how to document them (called model), and (3) how to use them (called management). Jalali[45] proposed a framework based on which different approaches are compared.

Aspect-Oriented Business Process Modeling enables separation of cross-cutting concerns by encapsulating them into separated modules. Like any business process modeling technique, these models should be able to be analyzed by the designer in terms of soundness. Jalali formalized the modeling using Petri nets and proposed an algorithm to merge these models into traditional ones, called Static Weaving.[21] In this way, the merged model can be analyzed through existing business process modeling approaches, and process designers can investigate problems at design time.

To enact aspect-oriented business process models,[20] Jalali et al[22,46] defined the dynamic weaving. This approach has been extended to cover the whole Business Process Management lifecycle.[16] They also show how the enactment support for AOBPM can support agile business process development.[47] Patiniotakis et al[48] also implemented a tool to support dynamic weaving. Witteborg et al[49] extended AOBPMN and supported the enactment of those models through static weaving.

There are some other works that focus on different aspects of AO-BPM. Di Francescomarino et al.[50] and Ghidini et al[51] investigated how cross-cutting concerns can be identified by querying process models. Jalali[32] proposed a method to discover aspect-oriented business process models from event logs that contain 4 main steps, ie, (1) cross-cutting concerns discovery, (2) cross-cutting concerns elimination, (3) business process discovery, and (4) relation (pointcut) discovery. Brandão et al[52] also proposed an approach to discover cross-cutting concerns automatically.

Sutton[53] illustrates how to use Aspect Orientation to model software processes. Reis et al[54] also used this principle to support the design of high-level policies in software process models. Furthermore, Harrison et al[55] demonstrated how Aspect-Oriented software development could support software development processes in an open source project, called Concern Manipulating Environment. Mishali and Katz[56] used aspects to support software development modeling and management. To support development, they enforced desirable development practices using aspects. To support management, they used aspects to document software development processes. Bendraou et al[57] proposed a framework for process modeling, simulation, and execution by combining Aspect and Model-Driven Engineering approaches. This framework has an executable semantics that enables simulation and execution of models, and it is based on UML4SPM, which is a UML2.0-based Language for Software Process Modeling. UML2.0 activity diagrams are enriched in this approach to deal with software process modeling.

All these approaches focus on imperative approaches to address separation of cross-cutting concerns in process modeling, and they cannot enable the separation of cross-cutting concerns executed in parallel to parts of the main process model and the definition of both optional and mandatory cross-cutting concerns.

Finally, there are few initiatives similar to our work aimed at investigating the understandability of different imperative and declarative modeling languages in the business process management area. For example, La Rosa et al[11,31] investigated the perceived usefulness and easy to use of different workflow patterns. De Smedt et al[58] investigated how the understandability of declarative models can be improved by making hidden dependencies among them visible to users. To the best of our knowledge, ours is the first evaluation about perceived ease of use and usefulness of an AO-BPM approach.

10 | CONCLUSION

This paper introduces a new approach for separating cross-cutting concerns when modeling business processes. This approach facilitates this by defining declarative rules to relate process models with cross-cutting concerns. In particular, it supports the definition of concerns that can be executed in parallel to the main process. The new approach also enables the definition of optional cross-cutting concerns. A formal semantics of the proposed models is also defined in this paper to enable workflow management systems to execute the proposed models. The approach has been implemented to demonstrate its feasibility in the setting of a case study from the education domain. The usability of the approach is also studied through a workshop with the involvement of students who are familiar with modeling business processes. The result shows that participants perceived the approach usable and easy to use.

For future work, it would be interesting to extend and investigate our approach, which is limited to the control-flow perspective, toward other perspectives as well, like data, resource, and time. In addition, we only focused on using specific types of declarative rules, so it is open to investigate if other types of declarative rules can help enriching the separation of concerns on a more advanced level. Finally, it is interesting to perform a thorough evaluation to measure and quantify the improvement of maintainability of the new types of models in comparison to the use of traditional approaches, in particular in the context of industry-strength cases.

ACKNOWLEDGMENT

We would like to thank Chiara Di Francescomarino for her valuable help in the analysis of the results of our evaluation.

ORCID

Amin Jalali http://orcid.org/0000-0002-6633-8587

REFERENCES

1. Pesic M. Constraint-based workflow management systems: shifting control to users. *Ph.D. Thesis*: Eindhoven University of Technology; 2008.
2. Object Management Group, Inc. (OMG). Business process model and notation (BPMN). Technical report, Object Management Group, Inc. (OMG); 2013.
3. van der Aalst W, Aldred L, Dumas M, ter Hofstede A. Design and implementation of the yawl system. In: Persson A, ed. *CAiSE 2004*, LNCS, vol. 3084. Riga, Latvia: Springer; 142-159.
4. De Giacomo G, Dumas M, Maggi FM, Montali M. Declarative process modeling in BPMN. *Caise 2015*. Springer, Stockholm, Sweden: Lecture Notes in Computer Science; 2015:84-100.
5. Maggi FM, Slaats T, Reijers HA. The automated discovery of hybrid processes. In: Proceedings of the Business Process Management - 12th International Conference, BPM 2014; 2014; Haifa, Israel. 392-399.
6. Slaats T, Schunselaar DMM, Maggi FM, Reijers HA. The semantics of hybrid process models. In: CoopIS. Rhodes, Greece; 2016:531-551.
7. Smedt JD, Weerdt JD, Vanthienen J. Fusion miner: process discovery for mixed-paradigm models. *Decis Support Syst*. 2015;77:123-136.
8. Smedt JD, Weerdt JD, Vanthienen J, Poels G. Mixed-paradigm process modeling with intertwined state spaces. *Bus IS Eng*. 2016;58(1):19-29.
9. van der Aalst W, Adams M, ter Hofstede A, Pesic M, Schonenberg H. Flexibility as a service. In: Chen L, ed. *Database Systems for Advanced Applications*, LNCS, vol. 5667. Brisbane, Australia: Springer; 2009:319-333.
10. Westergaard M, Slaats T. Mixing paradigms for more comprehensible models. In: Daniel F, ed. *Business Process Management*, LNCS, vol. 8094. Beijing, China: Springer; 2013:283-290.
11. La Rosa M, Wohed P, Mendling J, ter Hofstede A, Reijers H, van der Aalst W. Managing process model complexity via abstract syntax modifications. *IEEE Trans Ind Inf*. 2011;7(4):614-629.
12. Rezaei R, Chiew TK, Lee SP. A review on e-business interoperability frameworks. *J Syst Software*. 2014;93:199-216.
13. Elrad T, Filman R, Bader A. Aspect-oriented programming: introduction. *Commun ACM*. 2001;44(10):29-32.
14. Jalali A. Aspect-oriented business process management. *Ph.D. Thesis*: Department of Computer and Systems Sciences, Stockholm University; 2016.
15. Cappelli C, Santoro F, do Prado Leite J, Batista T, Medeiros A, Romeiro C. Reflections on the modularity of business process models: the case for introducing the aspect-oriented paradigm. *BPM J*. 2010;16:662-687.
16. Jalali A, Ouyang C, Wohed P, Johannesson P. Supporting aspect orientation in business process management - From process modelling to process enactment. *Software Syst Model*. 2017;16:903-925.
17. Jalali A, Maggi F, Reijers H. Enhancing aspect-oriented business process modeling with declarative rules. In: 34th International Conference on Concep-

tual Modeling (ER), LNCS. Stockholm, Sweden: Springer; 2015:108-115.

18. Tarr P, Ossher H, Harrison W, Sutton J. N degrees of separation: multi-dimensional separation of concerns. In: Proceedings of the 21st International Conference on Software Engineering, ICSE '99. New York, NY, USA: ACM; 1999:107-119.
19. Filman R, Friedman D, Norvig P. Aspect-oriented programming is quantification and obliviousness; 2000.
20. Jalali A. Foundation of aspect oriented business process management. *Master's Thesis*; 2011.
21. Jalali A. Static weaving in aspect oriented business process management. In: 34th International Conference on Conceptual Modeling (ER), LNCS. Stockholm, Sweden: Springer; 2015:548-557.
22. Jalali A, Wohed P, Ouyang C, Johannesson P. Dynamic weaving in aspect oriented business process management. In: Meersman R, ed. *CoopIS*, LNCS, vol. 8185. Graz, Austria: Springer; 2013:2-20.
23. Reichert M, Weber B. *Enabling Flexibility in Process-Aware Information Systems: Challenges, Methods, Technologies*. Springer Science & Business Media; 2012.
24. Schonenberg H, Mans R, Russell N, Mulyar N, van der Aalst WM. Towards a taxonomy of process flexibility. *CaiSE Forum*, Vol. 344. Montpellier, France: CEUR Workshop Proceedings; 2008:81-84.
25. Jalali Amin, Johannesson Paul. Multi-perspective business process monitoring. In: Nurcan S, Proper H, Soffer P, Krogstie J, Schmidt R, Halpin T, Bider I, eds. *Enterprise, Business-Process and Information Systems Modeling*, Lecture Notes in Business Information Processing, vol. 147. Valencia, Spain: Springer Berlin Heidelberg; 2013:199-213.
26. van der Aalst W. Verification of workflow nets. In: Azéma P, ed. *Petri Nets*, LNCS, vol. 1248. Toulouse, France: Springer; 1997:407-426.
27. van der Aalst W, Weijters T, Maruster L. Workflow mining: discovering process models from event logs. *IEEE Trans Knowledge Data Eng*. 2004;16(9):1128-1142.
28. Hollingsworth D. Workflow management coalition—the workflow reference model. Technical report, Workflow Management Coalition; 1995.
29. Moody DL. The method evaluation model: a theoretical model for validating information systems design methods. ECIS 2003 Proc. Naples, Italy: 2003:79.
30. Davis F. Perceived usefulness, perceived ease of use, and user acceptance of information technology. *MIS Quarterly*. 1989;13(3):319-340.
31. La Rosa M, ter Hofstede A, Wohed P, Reijers H, Mendling J, van der Aalst W. Managing process model complexity via concrete syntax modifications. *IEEE Trans Ind Inf*. 2011;7(2):255-265.
32. Jalali Amin. Aspect mining in business process management. In: Johansson B, Andersson B, Holmberg N, eds. *Perspectives in Business Informatics Research*, Lecture Notes in Business Information Processing, Vol. 194. Lund, Sweden: Springer International Publishing; 2014:246-260.
33. Jalali A, Wohed P, Ouyang C. Aspect oriented business process modelling with precedence. In: Mendling J, Weidlich M, eds. *BPMN*, LNCS, vol. 125. Vienna, Austria: Springer; 2012:23-37.
34. Rashid A, Sawyer P, Moreira A, Araújo J. Early aspects: a model for aspect-oriented requirements engineering. In: Proceedings of the IEEE Joint International Conference on Requirements Engineering. Essen, Germany: IEEE; 2002:199-202.
35. Charfi A, Mezini M. Aspect-Oriented web service composition with AO4BPEL. In: Zhang L, Jeckle M, eds. *Web Services*, LNCS, vol. 3250. Erfurt, Germany: Springer Berlin Heidelberg; 2004:168-182.
36. Charfi A, Mezini M. Aspect-oriented workflow languages. In: Meersman R, Tari Z, eds. *On the Move to Meaningful Internet Systems 2006: CoopIS, DOA, GADA, and ODBASE*, Lecture Notes in Computer Science, vol. 4275. Montpellier, France: Springer Berlin Heidelberg; 2006:183-200.
37. Charfi A, Mezini M. Ao4bpel: an aspect-oriented extension to bpel. *World Wide Web*. 2007;10(3):309-344.
38. Charfi A. Aspect-oriented workflow languages: AO4BPEL and applications. *Ph.D. Thesis*: der Technischen Universitat Darmstadt, Darmstadt; 2007.
39. Wang J, Zhu J, Liang H, Xu K. Concern oriented business process modeling. In: IEEE International Conference on E-Business Engineering, Hong Kong, China; 2007:355-358.
40. Shankardass A. The dynamic adaptation of an aspect oriented business process in a service oriented architecture platform. *Master's Thesis*: Athabasca University, Athabasca, Alberta; 2009.
41. Charfi A, Müller H, Mezini M. Aspect-oriented business process modeling with AO4BPMN. In: Khne T, ed. *Modelling Foundations and Applications*, LNCS, vol. 6138. Paris, France: Springer; 2010:48-61.
42. Jabeen A, Tariq S, Farooq Q, Malik Z. A lightweight aspect modelling approach for BPMN. In: IEEE 14th International Multitopic Conference (INMIC), 2011. Karachi, Pakistan: 2011:255-260.
43. Decker G, Overdick H, Weske M. Oryx an open modeling platform for the BPM community. In: Dumas M, Reichert M, Shan M-C, eds. *Business Process Management*, Lecture Notes in Computer Science, vol. 5240. Milan, Italy: Springer Berlin Heidelberg; 2008:382-385.
44. Santos F, Cappelli C, Santoro F, do Prado Leite J, Batista T. Aspect-oriented business process modeling: analyzing open issues. *Bus Process Manage J*. 2012;18(6):964-991.
45. Jalali A. Assessing aspect oriented approaches in business process management. *BIR*, LNBIP, vol. 194. Lund, Sweden: Springer;2014:231-245.
46. Jalali A, Wohed P, Ouyang C. Operational semantics of aspects in business process management. In: Herrero P, ed. *OTM 2012 Workshops*, Vol. 7567. Rome, Italy: Springer; 2012:649-653.
47. Bider Ilia, Jalali Amin. Agile business process development: why, how and when-applying Nonaka's theory of knowledge transformation to business process development. *Inf Syst E-Bus Manage*. 2014;14:1-39.
48. Patiniotakis I, Papageorgiou N, Verginadis Y, Apostolou D, Mentzas G. An aspect oriented approach for implementing situational driven adaptation of BPMN2.0 workflows. In: Rosa M, Soffer P, eds. *BPM Workshops*, LNBIP, vol. 132. Tallinn, Estonia: Springer; 2013:414-425.
49. Witteborg H, Charfi A, Colomer Collell D, Mezini M. Weaving aspects and business processes through model transformation. In: Villari M, Zimmermann W, Lau KK, eds. *Service-Oriented and Cloud Computing*, LNCS, vol. 8745. Manchester, UK: Springer; 2014:47-61.
50. Di Francescomarino C, Tonella P. Crosscutting concern mining in business processes. *IET Software*. 2011;5(6):552-562.
51. Ghidini C, Di Francescomarino C, Rospocher M, Tonella P, Serafini L. Semantics-based aspect-oriented management of exceptional flows in business processes. *IEEE Trans Syst Man Cybern Part C Appl Rev*. 2012;42(1):25-37.
52. Brandão BCP, Santoro FM, Azevedo LG. Towards aspects identification in business process through process mining. In: Proceedings of the Annual Conference on Brazilian Symposium on Information Systems: Information Systems: A Computer Socio-Technical Perspective-Volume 1 Brazilian Computer Society. Goiânia, GO, Brasil: 2015:99.
53. Sutton SM. *Aspect-Oriented Software Development and Software Process*. Berlin, Heidelberg: Springer Berlin Heidelberg; 2006;177-191.

54. Reis RQ, Reis CL, Schlebbe H, Nunes DJ. Towards an aspect-oriented approach to improve the reusability of software process models. In: Workshop on Early Aspects: Aspect-Oriented Requirements Engineering and Architecture Design (AOSD-2002). Enschede, The Netherlands: 2002.
55. Harrison W, Ossher H, Sutton S, Tarr P. Supporting aspect-oriented software development with the concern manipulation environment. *IBM Syst J*. 2005;44(2):309-318.
56. Mishali O, Katz S. Using aspects to support the software process: XP over Eclipse. In: Proceedings of the 5th International Conference on Aspect-Oriented Software Development ACM. Bonn, Germany: 2006:169-179.
57. Bendraou R, Jezqul J-M, Fleurey F. *Achieving Process Modeling and Execution through the Combination of Aspect and Model-Driven Engineering Approaches*, Vol. 24. John Wiley & Sons, Ltd; 2012;765-781.
58. De Smedt J, De Weerdt J, Serral E, Vanthienen J. Improving understandability of declarative process models by revealing hidden dependencies. In: International Conference on Advanced Information Systems Engineering. Ljubljana, Slovenia: Springer; 2016:59-64.

How to cite this article: Jalali A, Maggi FM, Reijers H. A hybrid approach for aspect-oriented business process modeling. *J Softw Evol Proc*. 2018;30:e1931. https://doi.org/10.1002/smr.1931

APPENDIX A

This appendix describes the structure of the survey and other relevant contents to help a better understanding of the experimental material.

A.1 | Structure of the survey

Students could choose answers based on scales from *extremely likely* to *extremely unlikely*. Here is the range of the scale among which students could select 1 option.

- extremely likely
- quite likely
- slightly likely
- neither likely nor unlikely
- slightly unlikely
- quite unlikely
- extremely unlikely

These options (with the specified order) are mapped to an ordinal scale where the extremely likely option is mapped to 7 and extremely unlikely option is mapped to 1.

In the following sections, we describe the 2 sets of questions for measuring the *perceived usefulness* and the *perceived ease of use* of our approach.

A.2 | Questions for measuring perceived usefulness

These are the questions used to measure the *perceived usefulness* of our approach. The questions are designed on the basis of the technology acceptance model.[30]

- B.1.1. Using Hybrid AO-BPM in this experience enabled me to model business processes more quickly.
- B.1.2. Using Hybrid AO-BPM in this experience improved my performance when modeling business processes.
- B.1.3. Using Hybrid AO-BPM in this experience increased my productivity when modeling business processes.
- B.1.4. Using Hybrid AO-BPM in this experience enhanced my effectiveness on modeling business processes.
- B.1.5. Using Hybrid AO-BPM in this experience made it easier for me to model business processes.
- B.1.6. I found Hybrid AO-BPM useful in this experience.

A.3 | Questions for measuring perceived ease of use

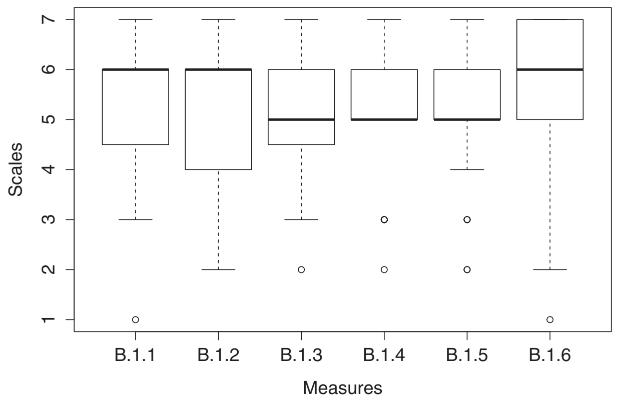
These are the questions used to measure the *perceived ease of use* of our approach. Also in this case, the questions are designed on the basis of the technology acceptance model.

- B.2.1. Learning how to use Hybrid AO-BPM was easy for me.
- B.2.2. I found it easy to get Hybrid AO-BPM to do what I wanted to do.

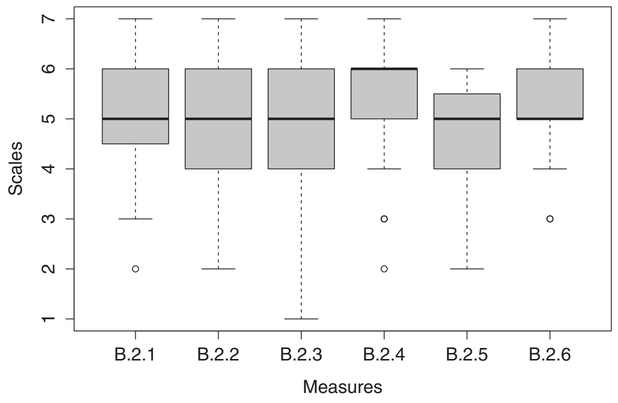
- B.2.3. I found Hybrid AO-BPM understandable and clear.
- B.2.4. I found Hybrid AO-BPM flexible to be used in modeling business processes.
- B.2.5. It was easy for me to become skilful at using Hybrid AO-BPM.
- B.2.6. I found Hybrid AO-BPM easy to use.

A.4 | Evaluation results in detail

Figure A1 shows the results of the evaluation in detail. In general, we received quite good feedback from participants for all measures. None of the median is below 5, which indicates good perceived usefulness and ease of use in general. Note that according to measure B.2.3, although the approach is well understood by most of the subjects, some of them found it not very easy to understand.



(A) Perceived Usefulness



(B) Perceived Ease of Use

FIGURE A1 Perceived Usefulness and Ease of Use of our approach in detail