A Grammatical Model for the Specification of Administrative Workflow Using Scenario as Modelling Unit

Milliam Maxime Zekeng Ndadji, Maurice Tchoupé Tchendji, Clémentin Tayou Djamegni, Didier Parigot

A Grammatical Model for the Specification of Administrative Workflow using Scenario as Modelling Unit

Milliam Maxime ZEKENG NDADJI[1,2][0000−0002−0417−5591], Maurice TCHOUPÉ TCHENDJI[1,2][0000−0002−9208−6838], Clémentin TAYOU DJAMEGNI[1][0000−0002−2231−3281], and Didier PARIGOT[3]

[1] Department of Mathematics and Computer Science
University of Dschang, PO Box 67, Dschang-Cameroon
{ndadji.maxime, maurice.tchoupe}@univ-dschang.org, dtayou@yahoo.com
[2] FUCHSIA Research Associated Team, https://project.inria.fr/fuchsia/
[3] Inria, Sophia Antipolis, France
didier.parigot@inria.fr

Abstract. Process modelling is a crucial phase of Business Process Management (BPM). Despite the many efforts made in producing process modelling tools, existing tools (languages) are not commonly accepted. They are mainly criticised for their inability to specify both the tasks making up the processes and their scheduling (their lifecycle models), the data they manipulate (their information models) and their organizational models. Process modelling in these languages often results in a single task graph; such a graph can quickly become difficult to read and maintain. Moreover, these languages are often too general (they have a very high expressiveness); this makes their application to specific types of processes complex: especially for administrative processes. In this paper, we present a new language for administrative processes modelling that allows designers to specify the lifecycle, information and organizational models of such processes using a mathematical tool based on a variant of attributed grammars. The approach imposed by the new language requires the designer to subdivide his process into scenarios, then to model each scenario individually using a simple task graph (an annotated tree) from which a grammatical model is further derived. At each moment then, the designer manipulates only a scenario of the studied process: this approach is more intuitive and modular; it allows to produce task graphs that are more refined and therefore, more readable and easier to maintain.

Keywords: Administrative Process Modelling · Workflow Language · Grammars · Artifact · Accreditation.

1 Introduction

Workflow technology aim at automating business processes[4]. To do so, it provides a clear framework composed of two major entities: (1) a *workflow language* for the description of such processes in a (generally graphical) format that can be interpreted

[4] A *business process* is a set of tasks that follow a specific pattern and are executed to achieve a specific goal [1]. When such processes are managed electronically, they are called *workflows*.

by (2) a software system called *Workflow Management System* (WfMS). The role of WfMS is to facilitate collaboration and coordination of various actors involved in the distributed execution of processes' tasks: in this way, workflow reduces the automation of business processes to their modelling in *workflow languages*; process modelling (specification) is therefore a crucial phase of workflow management[5].

Several tools have been developed to address process modelling. Among the most well-known are the BPMN standard (*Business Process Model and Notation*[6]) [13] based on statecharts, and the YAWL language (*Yet Another Workflow Language*) [1] which uses a formalism derived from that of Petri nets. Despite the significant research progress around these tools (often qualified as "*traditional tools*"), they are not commonly accepted. Indeed, they are often criticized for not being based on solid mathematical foundations [8], for having a much too great expressiveness compared to the needs of professionals in the field [22] and/or for not being intuitive [8].

Another important criticism often levelled at traditional workflow languages is the fact that they treat data (process *information model*) and users (part of process *organizational model*) as second-class citizens by highlighting tasks and their routing (process *lifecycle model*). To precisely remedy this, researchers have developed over the last two decades and under the initiative of IBM, the artifact-centric [15] approach to the design and execution of business processes. This one, revisited in several works [2,9,10,11,3,4,7], proposes a new approach to workflow management by focusing on both automated processes and data manipulated using the concept of "*business artifact*" or "*artifact*" in short. An artifact is considered as a document that conveys all the information concerning a particular case of execution of a given business process, from its inception in the system to its termination. A major shortcoming of artifact-centric models is that, after designing a given business process, it's difficult to manage it out of the context for which it was designed: specification and execution context (the WfMS on which it must be executed) are strongly coupled. In fact, in artifact-centric approaches the process specification is done with artifact modelling and artifacts are usually tailored to dedicated collaborative systems; process designers are then obliged to take into account certain details related to the workflow execution technique during the modelling phase: it is therefore difficult to consider these approaches exclusively as business process modelling tools since they are context dependant.

Another mentioned shortcoming of existing process modelling approaches is that they concentrate the modelling of a given process into a single task graph. Not only does this not allow designers to explicitly express the entire control flow of certain types of processes, but the resulting specifications are generally not easy to read, to maintain and to evolve. These concerns were first raised by Wil M. P. van der Aalst et al. [19,21]. They provide a solution to this by introducing the concept of *proclet*; they thus propose to deal with several levels of granularity assigned to lightweight workflow processes (proclets) in charge of orchestrating their execution. The modelling of each

[5] The *Workflow Management Coalition* (it is the organization responsible for developing standards in workflow) defines *workflow management* as the modelling and computer management of all the tasks and different actors involved in executing a business process [1].

[6] BPMN was initiated by the *Business Process Management Initiative* (BPMI) which merged with *Object Management Group* (OMG) in 2005.

level of granularity is therefore done using a smaller task graph. We find this vision very interesting. However, the notion of granularity manipulated in [19] is not very intuitive and seems, as for artifact-centric models, intimately linked to the execution model of proclets. In the case of an administrative process[7] $\mathcal{P}_{op}$, we think it would be more affordable to partition its task graph according to a characteristic that is natural to it. Knowing that such a process is naturally composed of a set of execution scenarios and can be represented by a finite set $\left\{\mathcal{S}_{op}^{1},\ldots,\mathcal{S}_{op}^{k}\right\}$ of *representative scenarios*[8] (see sec. 3.2) known in advance, we propose to use the "scenario" as the modelling unit".

All the above-mentioned shortcomings of traditional workflow languages confirm that there is still a need of scientific innovation in the field of business process modelling. This paper presents a new *Language for the Specification of Administrative Workflow Processes* (LSAWfP) based on the concept of attributed grammars (a specification of business processes by the means of attributed grammars is also presented in [5]). LSAWfP is built in a more traditional way and then, unlike the artifact-centric approaches, it allows process modelling independently of a workflow execution technique. Opposed to traditional workflow languages, LSAWfP provides coherent tools to model both processes' lifecycle model, information model and organizational model. Additionally, LSAWfP uses the "scenario" as the modelling unit: a given process modelling consists to the modelling of each of its representative scenario. Designers can thus focus on the modelling and the maintenance of process' parts rather than handling the whole process at a time: this seems to be more intuitive, modular and easier, and can also be further well composed with the level of granularity splitting approach proposed by [19].

LSAWfP is especially tailored for administrative processes modelling: its expressiveness is then built to fit the needs of such processes. Its modelling approach can be described as follow: from the observation that one can analyse the textual description of a given administrative process to exhibit all its possible representative execution scenarios leading to its business goals, LSAWfP propose to model each of these scenarios by an annotated tree called a *representative artifact* in which, each node corresponds to a task of the process, and each hierarchical decomposition (a node and its sons) represents a scheduling of these tasks. From these representative artifacts, are derived an attributed grammar $\mathbb{G}$ called the *Grammatical Model of Workflow* (GMWf). The symbols of a given GMWf represent the process tasks and each of its productions represents a scheduling of a subset of these tasks; intuitively, a production given by its left and right hand sides, specifies how the task on the left hand side precedes (must be executed before) those on the right hand side. Thus, the GMWf of a process contains both its *information model* (modelled by its attributes) and its *lifecycle model* (thanks to the set of its productions). Once the GMWf is obtained, LSAWfP propose to add organizational information (*organizational model*) modelled by two lists: $\mathcal{L}_{P_k}$ which contains actors involved in the process and $\mathcal{L}_{\mathcal{A}_k}$ which contains their *accreditations*. These lists aim at modelling actors, their roles and the different perceptions they have on a given

[7] According to the classification framework of [12], administrative processes are those for which all cases are known; tasks are predictable and their sequencing rules are simple and clear.

[8] We refer to a representative scenario as any execution scenario that, in combination with other representative scenarios, can generate a (potentially infinite) set of other scenarios.

process. Thus, with LSAWfP, the model (subsequently called *a Grammatical Model of Administrative Workflow Process* - GMAWfP -) of a given administrative process $\mathcal{P}_{op}$ is an executable grammatical specification given by a triplet $\mathbb{W}_f = \left(\mathbb{G}, \mathcal{L}_{P_k}, \mathcal{L}_{\mathcal{A}_k}\right)$.

The rest of this manuscript is organised as follows: after describing a running example (the peer-review process) in section 2, we present more formally and with illustrations, the proposed language in section 3. A discussion on its expressiveness and on some ongoing works is conducted in section 4. Finally, section 5 is devoted to the conclusion.

2 A Running Example: the Peer-Review Process

As running example, we will use the peer-review process. A brief description of it inspired by those made in [19], can be the following one:
- The process starts when the editor in chief (*EC*) receives a paper for validation;
- Then, the *EC* performs a pre-validation after which he can accept or reject the submission for various reasons (subject of minor interest, submission not within the journal scope, non-compliant format, etc.); let us call this **task** "A";
- If he rejects the submission, he writes a report (**task** "B") then notifies the corresponding author (**task** "D") and the process ends;
- Otherwise, he chooses an associated editor (*AE*) and sends him the paper for the continuation of its validation;
- The *AE* prepares the manuscript (**task** "C") and contacts simultaneously two experts for the evaluation of the paper (**tasks** "$E1$" and "$E2$"); if a contacted expert refuses to participate, the *AE* contacts another one (iteration on **task** "$E1$" or "$E2$"). Otherwise, the expert (referee) can start the evaluation;
- Each referee reads, seriously evaluates the paper (**tasks** "$G1$" and "$G2$") and sends back a report (**tasks** "$H1$" and "$H2$") and a message (**tasks** "$I1$" and "$I2$") to the *AE*;
- After receiving reports from all referees, the *AE* takes a decision and informs the *EC* (**task** "F") who sends the final decision to the corresponding author (**task** "D").

From the description above, one can identify all the tasks to be executed, their sequencing, actors involved and the tasks assigned to them. For this case, four actors are involved: an editor in chief (*EC*) which is responsible for initiating the process, an associated editor (*AE*) and two referees (*R*1 and *R*2). Figure 1 shows the orchestration diagrams corresponding to the graphical description of this peer-review process using the widely used process-centric notations BPMN (*Business Process Model and Notation*) and WF-Net (*Workflow Net*). As usual, tasks are ordered using *sequential flow*, *{And, Or}-Splits* and *{And, Or}-Joins*. Each diagram resumes the *main scenarios* of the studied process.

3 A Language for the Specification of Administrative Workflow Processes (LSAWfP)

In this section, we present the new language LSAWfP that allows to specify administrative workflow processes independently of a workflow execution technique.

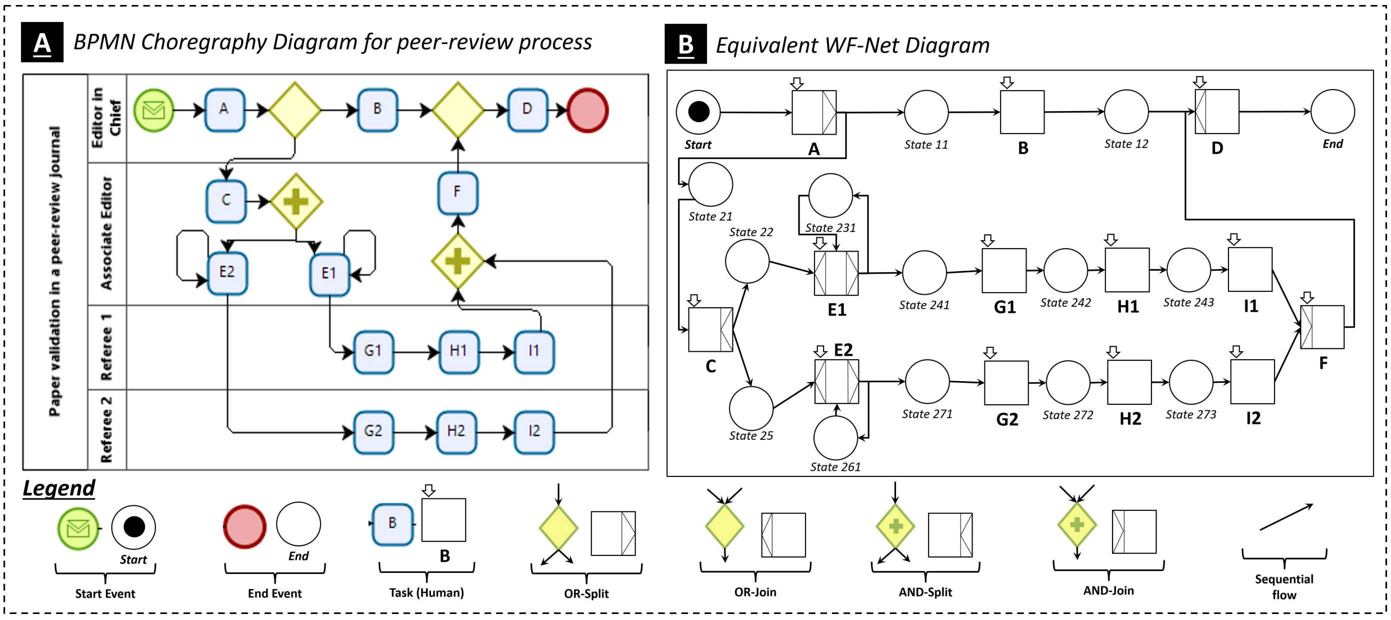

Fig. 1. Orchestration diagrams of the peer-review process.

3.1 Artifacts as Control Flow Graphs

Let's consider an administrative process $\mathcal{P}_{op}$ to be modelled. By definition (of administrative process), its set $\mathbb{T}_n = \{X_1, \ldots, X_n\}$ of tasks is known in advance. In traditional workflow languages like BPMN or WF-Net, the control flow between its tasks is represented using a directed graph that can contain cycles (see fig. 1). Such a graph allows the modelling of the potentially infinite set[9] of $\mathcal{P}_{op}$'s execution scenarios. Let's note however that each $\mathcal{P}_{op}$'s execution scenario can also be modelled using an annotated tree t_i called *artifact*. Indeed, starting from the fact that a given scenario $\mathcal{S}^i_{op}$ consists of a subset $\mathbb{T}_m \subseteq \mathbb{T}_n$ of $m \leq n$ tasks to be executed in a specific order (in parallel or in sequence), one can represent $\mathcal{S}^i_{op}$ as a tree t_i in which each node (labeled X_i) potentially corresponds to a task $X_i \in \mathbb{T}_m$ of $\mathcal{S}^i_{op}$ and each hierarchical decomposition (a node and its sons) corresponds to a scheduling: the task associated with the parent node must be executed before those associated with the son nodes; the latter must be executed according to an order - parallel or sequential - that can be specified by particular annotations "$\fatsemi$" (is sequential to) and "$\|$" (is parallel to) which will be applied to each hierarchical decomposition. The annotation "$\fatsemi$" (resp. "$\|$") reflects the fact that the tasks associated with the son nodes of the decomposition must (resp. can) be executed in sequence (resp. in parallel). To model iteration, nodes can be recursive in an artifact: i.e a node labelled X_i may appear in subtrees rooted by a node having the same label X_i.

[9] This is the case when there is one or more iterative routing (materialized by cycles in the task graph) on tasks.

Considering the running example (the peer-review process), two of its execution scenarios can be modelled using the two artifacts art_1 and art_2 in figure 2. In particular, we can see that art_1 shows how the task "Receipt and pre-validation of a submitted paper" assigned to the EC, and associated with the symbol A (see sec. 2), must be executed before tasks associated with the symbols B and D that are to be executed in sequence.

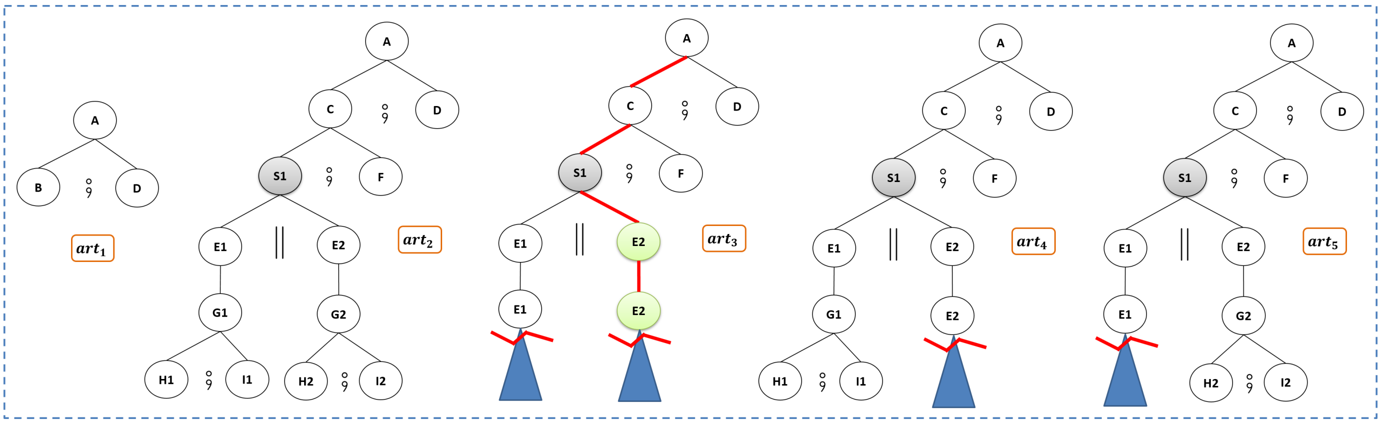

Fig. 2. Representative artifacts of a paper validation process in a peer-review journal.

3.2 Representative Artifacts and Grammatical Model of Workflow

Representative Artifacts As mentioned earlier (see sec. 3.1), the set of execution scenarios for a given administrative process can be infinite. This is the case of our running example process in which we can iterate on tasks $E1$ and $E2$ without limit and thus generate an infinite set of execution scenarios. In these cases, the designer cannot list this set of scenarios in order to model each of them. It is then necessary to substitute this one by a finite set $\{\mathcal{S}^1_{op}, \ldots, \mathcal{S}^k_{op}\}$ of scenarios said to be *representative*. Each representative scenario can then be modelled by a so called *representative artifact*.

For a given process, the set of its representative artifacts is obtained by adding to the finite set of artifacts modelling its nominal scenarios (those leading to its different business goals without iteration), those representing the modelling of its alternative scenarios (these are scenarios in which at least one iteration have been made). Operationally, when designing an alternative scenario artifact, the designer must prune it at each first iteration encountered: i.e the designer must prune each branch of an alternative scenario artifact as soon as he encounters a node labelled for the second time by a same label along a path starting from the root. In fact, one could assume that to design the representative artifacts of a given business process, the designer begins by identifying the initial tasks of it (i.e., the tasks that can start one of its execution scenarios); each of these tasks will thus constitute the root of several representative artifacts. To construct the set $arts_{X_{0_i}}$ of representative artifacts rooted in a given initial task X_{0_i}, the designer will:

(1) Construct an artifact *art* having X_{0_i} as the single node (root);
(2) Then, he will determine the set $follow = \left\{ \left(X_{1_{i_1}}, \ldots, X_{m1_{i_1}}\right), \ldots, \left(X_{1_{i_n}}, \ldots, X_{mn_{i_n}}\right) \right\}$ of task combinations (each combination is either sequential or parallel[10]) that can be immediately executed after the execution of X_{0_i}. For each combination $\left(X_{1_{i_j}}, \ldots, X_{mj_{i_j}}\right)$, the designer will create a new artifact art_j by expanding the node X_{0_i} of *art* such that in art_j, the tasks $X_{1_{i_j}}, \ldots, X_{mj_{i_j}}$ are the child nodes of X_{0_i}.
(3) It will then only remain to recursively develop (using the principle of (2)) each leaf node of the new artifacts until representative artifacts (those that describe an execution scenario in its entirety) are obtained.

This construction principle emphasizes the fact that one does not loose information by pruning an artifact when encountering a given node X for the second time in the same branch. In such a case, it is not necessary to develop X a second time since the designer has enumerated (in several artifacts) all the possibilities (scenarios) of continuing the execution of the process after the execution of the task associated with X. As we will see in section 3.2, these possibilities will be coded in a grammar and thus, the execution scenarios characterized by several iterations on X, will indeed be specified in the language. When constructing a representative artifact, the pruning of a branch is therefore systematic when a node is encountered for the second time; no matter how many nodes generate an iteration in the same branch.

Figure 2 presents the five representative artifacts of our running example process. The artifacts art_1 and art_2 model the two nominal scenarios: art_1 models the scenario in which the *EC* directly rejects the paper while art_2 models the case where the paper is evaluated by referees ($R1$ and $R2$) without the *AE* having to contact more than two experts (no iteration on tasks $E1$ and $E2$). The artifacts art_3, art_4 and art_5 represent the infinite set of alternative scenarios in this example. In art_3 in particular, we can see that the designer has pruned at node $E2$ which appeared for the second time in the same branch.

Grammatical Model of Workflow From the finite set of representative artifacts of a given process, it is possible to extract an abstract grammar[11] that represents the underlying process's lifecycle model : it is this grammar that we designate by the expression *Grammatical Model of Workflow (GMWf).*

Let's consider the set $\{t_1, \ldots, t_k\}$ of representative artifacts modelling the k representative execution scenarios of a given process $\mathcal{P}_{op}$ of n tasks ($\mathbb{T}_n = \{X_1, \ldots, X_n\}$). Each t_i is a derivation tree for an abstract grammar (a GMWf) $\mathbb{G} = (\mathcal{S}, \mathcal{P}, \mathcal{A})$ whose set of symbols is $\mathcal{S} = \mathbb{T}_n$ (all process tasks) and each production $p \in \mathcal{P}$ reflects a hierarchical

[10] If a given combination $\left(X_{1_{i_j}}, \ldots, X_{mj_{i_j}}\right)$ is sequential (resp. parallel), its tasks are to be (resp. can be) executed sequentially (resp. in parallel).

[11] It is enough to consider the set of representative artifacts as a regular tree language: there is therefore an (abstract) grammar to generate them.

decomposition contained in at least one of the representative artifacts. Each production is therefore exclusively of one of the following two forms: $p : X_0 \rightarrow X_1 \fatsemi \ldots \fatsemi X_n$ or $p : X_0 \rightarrow X_1 \parallel \ldots \parallel X_n$. The first form $p : X_0 \rightarrow X_1 \fatsemi \ldots \fatsemi X_n$ (resp. the second form $p : X_0 \rightarrow X_1 \parallel \ldots \parallel X_n$) means that task X_0 must be executed before tasks $\{X_1, \ldots, X_n\}$ and these must be (resp. that can be) executed in sequence (resp. in parallel). A GMWf can therefore be formally defined as follows:

Definition 1. *A* ***Grammatical Model of Workflow*** *(GMWf) is defined by $\mathbb{G} = (\mathcal{S}, \mathcal{P}, \mathcal{A})$ where :*

- *$\mathcal{S}$ is a finite set of* ***grammatical symbols*** *or* ***sorts*** *corresponding to various* ***tasks*** *to be executed in the studied business process;*
- *$\mathcal{A} \subseteq \mathcal{S}$ is a finite set of particular symbols called* ***axioms****, representing tasks that can start an execution scenario (roots of representative artifacts), and*
- *$\mathcal{P} \subseteq \mathcal{S} \times \mathcal{S}^*$ is a finite set of* ***productions*** *decorated by the annotations "$\fatsemi$" (is sequential to) and "$\parallel$" (is parallel to): they are* ***precedence rules****. A production $P = \left(X_{P(0)}, X_{P(1)}, \cdots X_{P(|P|)}\right)$ is either of the form $P : X_0 \rightarrow X_1 \fatsemi \ldots \fatsemi X_{|P|}$, or of the form $P : X_0 \rightarrow X_1 \parallel \ldots \parallel X_{|P|}$ and $|P|$ designates the length of P right-hand side. A production with the symbol X as left-hand side is called a X-production.*

Let's illustrate the notion of GMWf by considering the one generated from an analysis of the representative artifacts obtained in the case of the peer-review process (see fig. 2): the derived GMWf is $\mathbb{G} = (\mathcal{S}, \mathcal{P}, \mathcal{A})$ in which the set $\mathcal{S}$ of grammatical symbols is $\mathcal{S} = \{A, B, C, D, S1, E1, E2, F, G1, G2, H1, H2, I1, I2\}$ (see sec 2); the only initial task (axiom) is A (then $\mathcal{A} = \{A\}$) and the set $\mathcal{P}$ of productions is:

$P_1 : A \rightarrow B \fatsemi D$	$P_2 : A \rightarrow C \fatsemi D$	$P_3 : C \rightarrow S1 \fatsemi F$	$P_4 : S1 \rightarrow E1 \parallel E2$
$P_5 : E1 \rightarrow G1$	$P_6 : E2 \rightarrow G2$	$P_7 : E1 \rightarrow E1$	$P_8 : E2 \rightarrow E2$
$P_9 : G1 \rightarrow H1 \fatsemi I1$	$P_{10} : G2 \rightarrow H2 \fatsemi I2$	$P_{11} : B \rightarrow \varepsilon$	$P_{12} : D \rightarrow \varepsilon$
$P_{13} : F \rightarrow \varepsilon$	$P_{14} : H1 \rightarrow \varepsilon$	$P_{15} : I1 \rightarrow \varepsilon$	$P_{16} : H2 \rightarrow \varepsilon$
$P_{17} : I2 \rightarrow \varepsilon$			

There may be special cases where it is not possible to schedule the tasks of a scenario using the two (only) forms of production selected for GMWf. For example, this is the case for the peer-review process wherein task C precedes tasks $E1$, $E2$ and F, tasks $E1$ and $E2$ can be executed in parallel and precede F (see sec. 2). In such cases, the introduction of a few new symbols known as *(re)structuring symbols* (not associated with tasks) can make it possible to produce a correct scheduling. For the peer-review process example, the introduction of a new symbol $S1$ allows us to obtain the following productions: $P_3 : C \rightarrow S1 \fatsemi F$ and $P_4 : S1 \rightarrow E1 \parallel E2$ which properly model the required scheduling and avoid the usage of the malformed production $p : C \rightarrow E1 \parallel E2 \fatsemi F$ (see in fig. 2, art_2, the node $S1$ — in gray —). To deal with such cases, the previously given GMWf definition (definition 1) is slightly adapted by integrating the (re)structuring symbols; the resulting definition is as follows:

Definition 2. *A* ***Grammatical Model of Workflow*** *(GMWf) is defined by $\mathbb{G} = (\mathcal{S}, \mathcal{P}, \mathcal{A})$ wherein $\mathcal{P}$ and $\mathcal{A}$ refer to the same purpose as in definition 1, $\mathcal{S} = \mathcal{T} \cup \mathcal{T}_{Struc}$ is a finite set of* ***grammatical symbols*** *or* ***sorts*** *in which, those of $\mathcal{T}$ correspond to* ***tasks*** *of the studied business process, while those of $\mathcal{T}_{Struc}$ are (re)structuring symbols.*

3.3 Modelling the Information Model of Processes with GMWf

As formalized in definition 2, a GMWf perfectly models the tasks and control flow of administrative processes (lifecycle model). In this section we discuss the specification of processes-related data (*the information model*) in LSAWfP.

It is not easy to model the structure of business processes data using a general type as they differ from one process to another. For the current work, tackling the automated processes data structure has no proven interest because it does not bring any added value to the proposed model: a representation of these data using a set of variables associated with tasks is largely sufficient. However, it should be noted that in existing data-driven modelling approaches like the Guarded Attribute Grammar (GAG) model [5], these variables typically have two parts to allow designers to model each task's (1) preconditions or input data required for its actual execution, and (2) post-conditions or output data produced during its execution. In addition, dependency relationships between data are often modelled.

In this work, the potential manipulated data by a given process task is represented using an *attribute* embedded in the nodes associated with it. To materialise this adjustment, we update for the last time the definition of GMWf. We thus associate with each symbol, an attribute named *status* allowing to store all the data of the associated task; its precise type is left to the discretion of the process designer. The new definition of GMWf is thus the following one:

Definition 3. *A **Grammatical Model of Workflow** (GMWf) is defined by $\mathbb{G} = (\mathcal{S}, \mathcal{P}, \mathcal{A})$ wherein $\mathcal{S}$, $\mathcal{P}$ and $\mathcal{A}$ refer to the same purpose as in definition 2. Each grammatical symbol $X \in \mathcal{S}$ is associated with an attribute named **status**, that can be updated when tasks are executed; **X.status** provides access (read and write) to its content.*

3.4 An Organizational Model for LSAWfP

Because business processes are generally carried out collectively, it is important to model actors an to set up mechanisms to ensure better coordination between them and to eventually guarantee the confidentiality of certain actions and data: this is the purpose of *accreditation*. The accreditation of a given actor provides information on its rights (permissions) relatively to each sort (task) of the studied process's GMWf. We propose here, a simple but non-exhaustive nomenclature of rights. It is inspired by the one used in UNIX-like operating systems. Three types of accreditation are therefore defined: accreditation in reading *(r)*, writing *(w)* and execution *(x)*.
1. *The accreditation in reading (r)*: an actor accredited in reading on sort X must be informed of the execution of the associated task; he must also have free access to its execution state (data generated during its execution). We call an actor's ***view***, the set of sorts on which he is accredited in reading.
2. *The accreditation in writing (w)*: an actor accredited in writing on sort X can execute the associated task. To be simple, any actor accredited in writing on a sort must necessarily be accredited in reading on it.
3. *The accreditation in execution (x)*: an actor accredited in execution on sort X is allowed to ask the actor who is accredited in writing in it, to execute it (realization of the associated task). More formally, an accreditation is defined as follows:

Definition 4. *An **accreditation** $\mathcal{A}_{A_i}$ defined on the set $\mathcal{S}$ of grammatical symbols for an actor A_i, is a triplet $\mathcal{A}_{A_i} = \left(\mathcal{A}_{A_i(r)}, \mathcal{A}_{A_i(w)}, \mathcal{A}_{A_i(x)}\right)$ such that, $\mathcal{A}_{A_i(r)} \subseteq \mathcal{S}$ also called **view** of actor A_i, is the set of symbols on which A_i is accredited in reading, $\mathcal{A}_{A_i(w)} \subseteq \mathcal{A}_{A_i(r)}$ is the set of symbols on which A_i is accredited in writing and $\mathcal{A}_{A_i(x)} \subseteq \mathcal{S}$ is the set of symbols on which A_i is accredited in execution.*

The accreditations of various actors must be produced by the workflow designer just after modelling the scenarios in the form of representative artifacts. From the task assignment for the peer-review process in the running example (see sec. 2), it follows that the accreditation in writing of the EC is $\mathcal{A}_{EC(w)} = \{A,B,D\}$, that of the AE is $\mathcal{A}_{AE(w)} = \{C,S1,E1,E2,F\}$ and that of the first (resp. the second) referee is $\mathcal{A}_{R_1(w)} = \{G1,H1,I1\}$ (resp. $\mathcal{A}_{R_2(w)} = \{G2,H2,I2\}$). Since the EC can only execute the task D if the task C is already executed (see fig. 2), in order for the EC to be able to ask the AE to execute this task, he must be accredited in execution on it; so we have $\mathcal{A}_{EC(x)} = \{C\}$. Moreover, in order to be able to access all the information on the peer-review evaluation of a paper (task C) and to summarize the right decision to send to the author, the EC must be able to consult the reports (tasks $I1$ and $I2$) and the messages (tasks $H1$ and $H2$) of the different referees, as well as the final decision taken by the AE (task F). These tasks, added to $\mathcal{A}_{EC(w)}$[12] constitute the set $\mathcal{A}_{EC(r)} = \mathcal{V}_{EC} = \{A,B,C,D,H1,H2,I1,I2,F\}$ of tasks on which it is accredited in reading. By doing so for each of other actors, we deduce the accreditations represented in table 1.

Actor	Accreditation
EC	$\mathcal{A}_{EC} = (\{A,B,C,D,H1,H2,I1,I2,F\},\{A,B,D\},\{C\})$
AE	$\mathcal{A}_{AE} = (\{A,C,S1,E1,E2,F,H1,H2,I1,I2\},\{C,S1,E1,E2,F\},\{G1,G2\})$
$R1$	$\mathcal{A}_{R1} = (\{C,G1,H1,I1\},\{G1,H1,I1\},\emptyset)$
$R2$	$\mathcal{A}_{R2} = (\{C,G2,H2,I2\},\{G2,H2,I2\},\emptyset)$

Table 1. Accreditations of the different actors taking part in the peer-review process.

Since the (re)structuring symbols are not associated with tasks and were only introduced to adjust the control flow, their execution neither requires nor produces data. Therefore, the accreditation in writing and execution on them may be best left to the designer's appreciation; he will then make the assignment by referring to the execution model he will use later. To this end, he could use the same principle for the assignment of these accreditations in the case of concrete process' tasks. However, one could by default consider that all actors are accredited in reading on (re)structuring symbols; this would make these symbols visible to all of them and would guarantee that the adjustment of the control flow will be effective for all of them even if they have partial perceptions of the process.

[12] Recall that we consider that one can only execute what he see.

3.5 Summary: definition of LSAWfP

To summarise, we state that in LSAWfP, an administrative process $\mathcal{P}_{op}$ is specified using a triplet $\mathbb{W}_f = \left(\mathbb{G}, \mathcal{L}_{P_k}, \mathcal{L}_{\mathcal{A}_k}\right)$ called *a Grammatical Model of Administrative Workflow Process* (GMAWfP) and composed of: a GMWf, a list of actors and a list of their accreditations. The GMWf is used to describe all the tasks of the studied process and their scheduling, while the list of accreditations provides information on the role played by each actor involved in the process execution. A GMAWfP can then be formally defined as follows:

Definition 5. *A* ***Grammatical Model of Administrative Workflow Process*** *(GMAWfP) $\mathbb{W}_f$ for a given business process, is a triplet $\mathbb{W}_f = \left(\mathbb{G}, \mathcal{L}_{P_k}, \mathcal{L}_{\mathcal{A}_k}\right)$ wherein $\mathbb{G}$ is the studied process (global) GMWf, $\mathcal{L}_{P_k}$ is the set of k actors taking part in its execution and $\mathcal{L}_{\mathcal{A}_k}$ represents the set of these actors accreditations.*

4 Ongoing and Perspective Work on LSAWfP

In this section, we present some of the work being currently done on LSAWfP while assessing what has already been done and presented in this paper.

4.1 On the Expressiveness of LSAWfP

Let's consider a specification $\mathbb{W}_f = \left(\mathbb{G}, \mathcal{L}_{P_k}, \mathcal{L}_{\mathcal{A}_k}\right)$ of a given business process $\mathcal{P}_{op}$. As described above, its organizational model that expresses and classifies/assigns the resources that must execute its tasks is given by the couple $\left(\mathcal{L}_{P_k}, \mathcal{L}_{\mathcal{A}_k}\right)$ of $\mathbb{W}_f$. Its informational model that describes the data structure being manipulated is given by the type of the attribute *status* associated with each task. Its lifecycle model that provides information on tasks and their sequencing (coordination) is given by the GMWf $\mathbb{G}$ of $\mathbb{W}_f$. Thus, we can conclude that LSAWfP has the major expected characteristics of a workflow language according to [1].

The GMWf effectively allows the designers to specify all the basic control flows (sequential, parallel, alternative and iterative) which can be found in traditional workflow languages. Figure 3 gives for each type of basic control flow its BPMN notation and the corresponding notations (artifact and associated productions) in LSAWfP as described below:
- the sequential flow between two tasks A and B can be expressed either by a production p of the form $p : A \rightarrow B$, or by a production q of the form $q : S \rightarrow A \fatsemi B$ in which S is a (re)structuring symbol (see fig. 3(a));
- the parallel flow between two tasks A and B is expressed using a production p of the form $p : S \rightarrow A \parallel B$ (see fig. 3(b));
- the alternative flow (choice) between two tasks $A1$ and $A2$ is expressed using two productions $p1$ and $p2$ such that $p1 : S \rightarrow A1$ and $p2 : S \rightarrow A2$; S is a (re)structuring symbol expressing the fact that after "execution" of S, one must execute either task $A1$ or task $A2$ (see fig. 3(c)).
- iterative routing (repetition) is expressed using recursive symbols. Thus the productions $p1 : A \rightarrow B$, $p2 : B \rightarrow C$ and $p3 : B \rightarrow A$ express a potentially (transitive) iterative

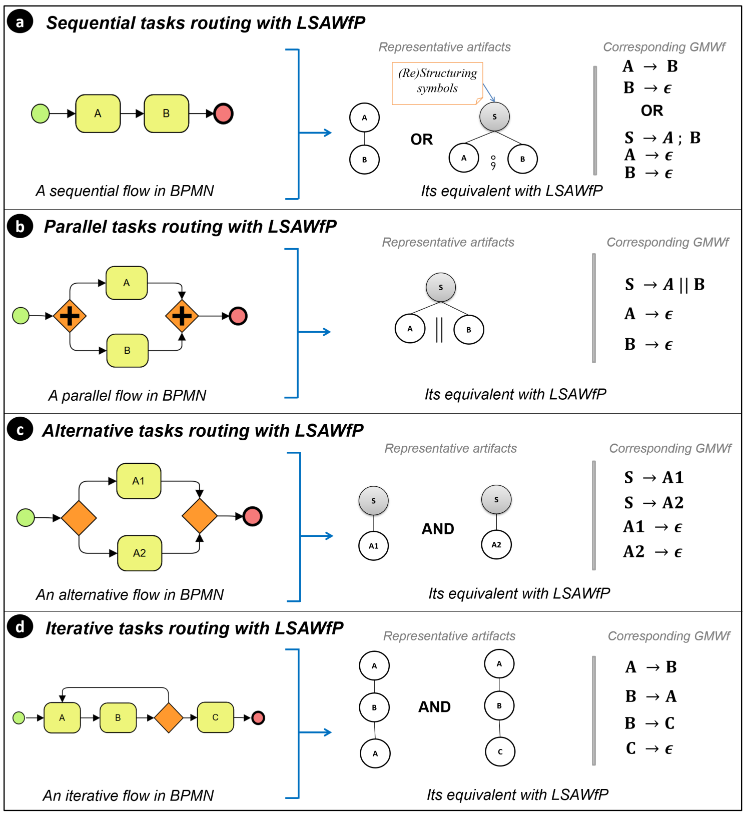
a Sequential tasks routing with LSAWfP
Representative artifacts
Corresponding GMWf
(Re)Structuring symbols
A sequential flow in BPMN
Its equivalent with LSAWfP
OR
A → B
B → ϵ
OR
S → A ; B
A → ϵ
B → ϵ
b Parallel tasks routing with LSAWfP
A parallel flow in BPMN
S → A || B
A → ϵ
B → ϵ
c Alternative tasks routing with LSAWfP
An alternative flow in BPMN
AND
S → A1
S → A2
A1 → ϵ
A2 → ϵ
d Iterative tasks routing with LSAWfP
An iterative flow in BPMN
A → B
B → A
B → C
C → ϵ

Fig. 3. Illustrating basic control flows with LSAWfP.

flow on the task A (see fig. 3(d)); $P_7: E1 \rightarrow E1$ in the running example also expresses a direct iterative flow on $E1$ (see fig. 2).

One avenue we are currently exploring is that of measuring the expressiveness of LSAWfP in relation to workflow patterns [20]. This will allow us to characterize precisely the class of processes that this language can model.

4.2 Towards an Artifact-Centric Model of Processes Design and Distributed Execution Based on Cooperative Edition of a Mobile Artifact

We are also working to produce an artifact-centric model of business process management. In this model inspired by the work of Badouel et al. on cooperative editing [6,16,17,18,14], the process tasks are executed by the various actors with the help of software agents that they pilot. These software agents are autonomous, reactive and communicate in peer to peer mode by exchanging an artifact (considered as "mobile") edited cooperatively. This mobile artifact is an annotated tree that represents the execution status of the process at each moment. For this purpose, it contains information on the tasks already executed, on the data produced during these executions and on the tasks ready to be executed.

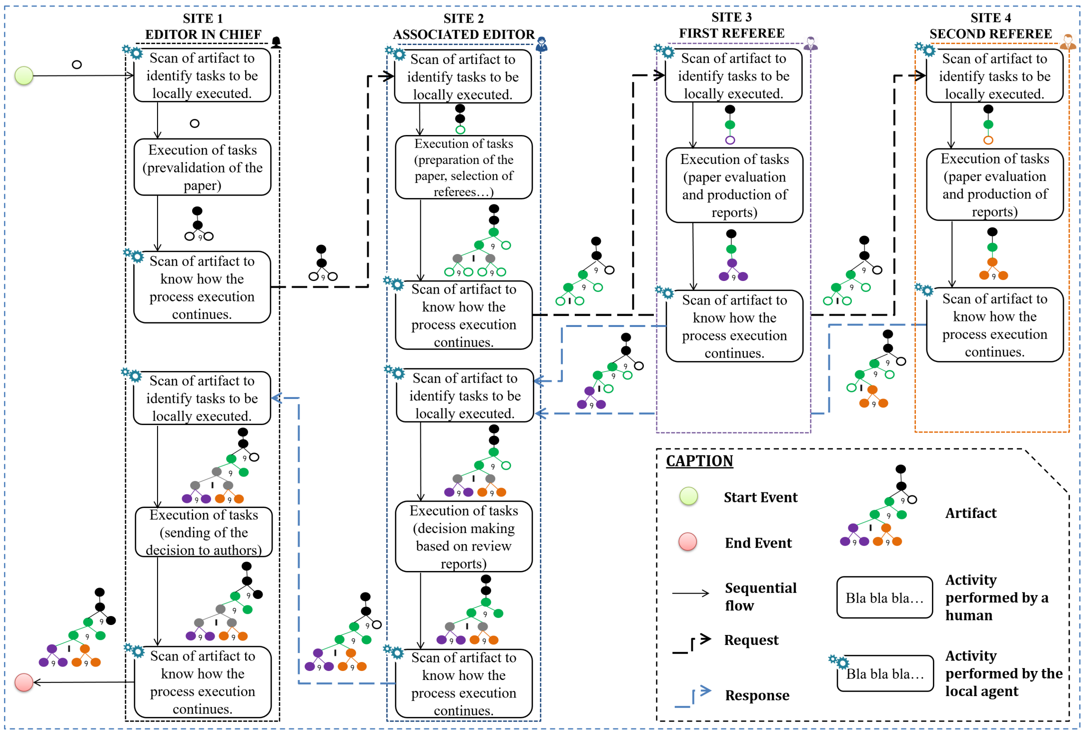
SITE 1
EDITOR IN CHIEF
SITE 2
ASSOCIATED EDITOR
SITE 3
FIRST REFEREE
SITE 4
SECOND REFEREE
Scan of artifact to identify tasks to be locally executed.
Execution of tasks (prevalidation of the paper)
Scan of artifact to know how the process execution continues.
Execution of tasks (preparation of the paper, selection of referees…)
Execution of tasks (paper evaluation and production of reports)
Execution of tasks (sending of the decision to authors)
Execution of tasks (decision making based on review reports)
CAPTION
Start Event
End Event
Sequential flow
Request
Response
Artifact
Bla bla bla…
Activity performed by a human
Activity performed by the local agent

Fig. 4. An overview of the artifact-centric execution of the peer-review process.

When the mobile artifact is received at a given execution site, the local agent executes an update protocol whose purpose is to reveal the tasks ready to be executed locally by the local actor. The execution of the tasks by the local actor is done using a specialized editor and can be assimilated to the edition of a structured document since its actions cause the received mobile artifact (the tree) to be updated, by expanding some of its leaf nodes into sub-trees and by assigning values to the "status" attributes of some other nodes. When all the tasks ready to be locally executed have been executed, the artifact is sent to other agents for further execution of the process if necessary.

To run the peer-review process described in section 2 with the artifact-centric model being built, four agents controlled by four actors (the *EC*, the *AE*, the *R*1 and the *R* agents) will be deployed. Figure 4 sketches an overview of exchanges that can take place between those four agents. The scenario presented there corresponds to the nominal one in which the paper is pre-validated by the *EC* and therefore, is analysed by a peer review committee. The artifact-centric execution is triggered on the *EC*'s site by introducing (in this site) an artifact reduced to its root node. During its transit through the system, this artifact grows. Note that there may be situations where multiple copies of the artifact are updated in parallel; this is notably the case when they are present on site 3 (first referee) and 4 (second referee).

5 Conclusion

In this paper, we have proposed a new workflow language called LSAWfP which allows, through a simple grammar-based formalism, to specify administrative business processes. Like any traditional workflow language, LSAWfP allows to specify basic flows (sequential, parallel, alternative and iterative) that are generally found in workflow models; particularly, it focuses on the modelling of each of the process scenario using an artifact. Moreover, LSAWfP allows to model the main characteristics of business processes (their lifecycle, their informational and their organizational aspects). We also presented some of the work associated with LSAWfP that are currently in progress.

In our opinion, an other work that can be done following the one presented in this paper, is the production of a software tool to assist in the specification of business processes in the LSAWfP language. Such a tool, in addition to providing interfaces for the graphic design of scenario graphs (representative artifacts), will allow designers to check the correctness of the produced specifications and ensure their conversion to other formats (BPMN and YAWL for example).

References

1. Van der Aalst, W.M.: Business process management: a comprehensive survey. ISRN Software Engineering **2013** (2013)
2. Abi Assaf, M.: Towards an integration system for artifact-centric processes. In: Proceedings of the 2016 on SIGMOD'16 PhD Symposium. pp. 2–6. ACM (2016)
3. Assaf, M.A., Badr, Y., Amghar, Y.: A continuous query language for stream-based artifacts. In: International Conference on Database and Expert Systems Applications. pp. 80–89. Springer (2017)
4. Assaf, M.A., Badr, Y., El Khoury, H., Barbar, K.: Generating database schemas from business artifact models. I.J. Information Technology and Computer Science **2**, 10–17 (2018)
5. Badouel, E., Hélouët, L., Kouamou, G.E., Morvan, C., Fondze Jr, N.R.: Active workspaces: distributed collaborative systems based on guarded attribute grammars. ACM SIGAPP Applied Computing Review **15**(3), 6–34 (2015)
6. Badouel, E., Tchendji, M.T.: Merging Hierarchically-Structured Documents in Workflow Systems. Electronic Notes in Theoretical Computer Science **203**(5), 3–24 (2008). https://doi.org/10.1016/j.entcs.2008.05.017, https://doi.org/10.1016/j.entcs.2008.05.017
7. Boaz, D., Limonad, L., Gupta, M.: Bizartifact: Artifact-centric business process management, june 2013 (2013), https://sourceforge.net/projects/bizartifact/,accessed12December2019
8. Börger, E.: Approaches to modeling business processes: a critical analysis of bpmn, workflow patterns and yawl. Software & Systems Modeling **11**(3), 305–318 (2012)
9. Deutsch, A., Hull, R., Vianu, V.: Automatic verification of database-centric systems. ACM SIGMOD Record **43**(3), 5–17 (2014)
10. Hull, R., Narendra, N.C., Nigam, A.: Facilitating workflow interoperation using artifact-centric hubs. In: Service-Oriented Computing, pp. 1–18. Springer (2009)

11. Lohmann, N., Wolf, K.: Artifact-centric choreographies. In: International Conference on Service-Oriented Computing. pp. 32–46. Springer (2010)
12. McCready, S.: There is more than one Kind of Workflow Software. Computerworld **2** (1992)
13. Model, B.P.: Notation (BPMN) version 2.0. OMG Specification, Object Management Group pp. 22–31 (2011)
14. Ndadji, M.M.Z., Tchendji, M.T.: A Software Architecture for Centralized Management of Structured Documents in a Cooperative Editing Workflow. In: Innovation and Interdisciplinary Solutions for Underserved Areas, pp. 279–291. Springer (2018)
15. Nigam, A., Caswell, N.S.: Business artifacts: An approach to operational specification. IBM Systems Journal **42**(3), 428–445 (2003)
16. Tchoupe Tchendji, M.: Une Approche Grammaticale pour la Fusion des Réplicats Partiels d'un Document Structuré: Application à l'Édition Coopérative Asynchrone. Ph.D. thesis, Université de Rennes I (France), Université de Yaoundé I (Cameroun) (2009)
17. Tchoupé Tchendji, M., Djeumen, R.D., Atemkeng, M.T.: A Stable and Consistent Document Model Suitable for Asynchronous Cooperative Edition. Journal of Computer and Communications **5**(08), 69 (2017)
18. Tchoupé Tchendji, M., Zekeng Ndadji, M.M.: Tree Automata for Extracting Consensus from Partial Replicas of a Structured Document. Journal of Software Engineering and Applications **10**(05), 432 (2017)
19. Van Der Aalst, W.M., Barthelmess, P., Ellis, C.A., Wainer, J.: Proclets: A framework for lightweight interacting workflow processes. International Journal of Cooperative Information Systems **10**(04), 443–481 (2001)
20. Van Der Aalst, W.M., ter Hofstede, A.H.: Workflow patterns put into context. Software & Systems Modeling **11**(3), 319–323 (2012)
21. Van Der Aalst, W.M., Mans, R., Russell, N.C.: Workflow support using proclets: Divide, interact, and conquer. IEEE Data Eng. Bull. **32**(3), 16–22 (2009)
22. Zur Muehlen, M., Recker, J.: How much language is enough? theoretical and practical use of the business process modeling notation. In: Seminal Contributions to Information Systems Engineering, pp. 429–443. Springer (2013)