A Language for the Specification of Administrative Workflow Processes with Emphasis on Actors' Views

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

A Language for the Specification of Administrative Workflow Processes with Emphasis on Actors' Views

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] Departement 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. Administrative workflows refer to variable business processes in which all cases are known; tasks are predictable and their sequencing rules are simple and clearly defined. When such processes are collaboratively executed by several actors, it may be desirable, for security reasons (confidentiality), that each of them has at all times, only a partial perception (this is what we call "actor's view") of the current process state. This concern seems sufficiently important to be considered when specifying such workflows. However, traditional workflow specification languages (BPMN, BPEL, YAWL) only partially address it. This is why we present in this paper, a new language for specifying administrative workflows that allows us not only to simply model all of the processes tasks and their sequence, but also and especially to explicitly express the rights of the various actors with respect to each of them, in order to guarantee a certain degree of security. The proposed model is an executable grammatical specification that allows to express using decorated productions, the different types of basic flows (sequential, parallel, alternative and iterative) that are found in workflow specification languages; moreover, it also allows to specify the rights of each actor in each process and on its data in a formalism similar to that used in UNIX-like operating systems.

Keywords: Business Process · Workflow Language · Grammatical Model of Workflow · Artifact · Accreditation, View.

1 Introduction

Workflow technology is concerned with automating business processes. Since its emergence in the early 80s, it has continued to prove its worth in the computer-aided production industry by allowing companies to reduce the costs of their production, to quickly and easily develop new products and services, and thus to be more competitive [3]. Technically, workflow technology provides a clear technological framework, composed of two major entities: (1) a *process specification language* or *workflow language* which allows the description of such processes in a (graphical and/or textual)

format that can be interpreted by (2) an autonomous system called *Workflow Management System* (WfMS); the role of the latter is to facilitate collaboration and coordination between various actors involved in the (generally distributed) execution of processes, as well as to facilitate their ability to execute the tasks under their responsibility [2]: In this way, workflow technology reduces the automation of business processes to their specifications in *workflow languages*.

The growing reputation of workflow led to the creation, in 1993, of the *Workflow Management Coalition*[4] (WfMC) as the organization responsible for developing standards in this field. Since then, standards have been adopted, particularly for workflow languages. Through its standard *XML Process Definition Language* (XPDL), WfMC supports BPMN (*Business Process Model and Notation*[5]) [6] as a business process modelling standard. In addition to BPMN, several other process specification languages have been developed. Examples include YAWL (*Yet Another Workflow Language*) [7,8] which allows processes to be represented using a formalism derived from that of *Petri Nets*, and BPEL (*Business Process Execution Language*) [4] which allows to formalize the behaviour of business processes by choreographing web services.

Motivations of this work: one of the inherent characteristics of business processes is, the confidentiality that must sometimes be guaranteed on data and/or tasks that are executed. It is indeed easy to imagine administrative processes in which, various actors at any given time, have only a potentially partial perception of all the activities that have already and/or must be carried out: the perception that an actor has on the current state of a process is called his "view on the process". For example, in a peer-review process, a reviewer does not necessarily need to know if another reviewer has been contacted for the expertise of the article entrusted to him; and even if so, he should not necessarily know if the latter has already returned his report, etc. Similarly, when organising a journey for a Head of State, not all actors (secret services, civil office, doctor, presidential guard, etc.) have access to the same information which may include for example, tasks to be executed, their dates and states of execution, etc.

Administrative workflows are characterized by the fact that all cases (tasks and their sequences), all actors and the permissions they have on tasks, etc. are known in advance. When specifying such processes, it should also be possible to model confidentiality constraints; for example, it should be possible to explicitly express the permissions - called in the following *accreditations* - which each actor has on each task. Unfortunately, traditional worflow languages (BPMN, BPEL, YAWL, etc.), although well developed and very expressive (very high expressiveness), do not allow to simply address this problem by providing formalisms (notations) to model them. Indeed, the formalisms they offer generally only allow to specify tasks, their sequencing and their allocation to actors; they delegate the detailed management of possible accreditations to the WfMS [4].

Another important aspect of administrative processes is that they are inherently distributed. It is therefore natural to consider specifying them for execution on truly distributed architectures in order to take full advantage of the benefits (better fault toler-

[4] Official website of the WfMC: https://www.wfmc.org/.

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

ance, better performance, absence of congestion points, etc. [10]) that the latter provide over centralized architectures. On this aspect specifically, it can be noted that traditional workflow languages have been designed to write specifications to be executed on (distributed) WfMS built in the centralized architectural style standardized by the WfMC [9].

Paper contribution: considering the above-mentioned shortcomings of traditional workflow languages, we propose in this paper a new *Language for the Specification of Administrative Workflow Processes* (LSAWfP) allowing to simply express the standard characteristics (tasks, scheduling, etc.) of business processes as we would do with its predecessors. However, unlike these, LSAWfP makes it possible to specify the accreditations of the various actors of the process. With LSAWfP, the model of an administrative process is an executable grammatical specification given by a triplet $\mathbb{W}_f = \left(\mathbb{G}, \mathcal{L}_{\mathcal{P}_k}, \mathcal{L}_{\mathcal{A}_k}\right)$ in which:

- $\mathbb{G}$ is the *Grammatical Model of Workflow* (GMWf - a grammar -): its sorts (symbols) represent all tasks and its productions (decorated by sequencing operators) express their ordering;
- $\mathcal{L}_{\mathcal{P}_k}$ is the list of actors involved in the process;
- $\mathcal{L}_{\mathcal{A}_k}$ is the list of accreditations: it allows to define the *view of each actor* in a formalism inspired by the one used to specify user rights in UNIX-like systems.

Manuscript organization: after reminding some basic definitions and notions on workflows in section 2, we present more formally the proposed language (sec. 3.1) followed by an illustration of its use for modelling a peer-review process (sec. 3.2). A discussion on its expressiveness is conducted in section 3.3. Section 3.4 gives an overview of the recommended WfMS architecture on which instances of LSAWfP (ie. specifications made in LSAWfP) must be executed. Finally, section 4 is devoted to the conclusion.

2 Preliminaries

Workflow technology is full of many concepts. The presentation of some of them in this section aims at facilitating their understanding and especially, at motivating some of the choices made in this paper.

Definitions A *business process* is a set of tasks that follow a specific pattern and are executed to achieve a specific goal [3]. When such processes are managed electronically, they are called *workflows*. The WfMC [9] defines *workflow management* as the modelling and computer management of all the tasks and different actors involved in executing a business process. The peer-review validation [1] of an article in a scientific journal is a common example of business process.

Workflow typology In the literature, there are several approaches to workflow classification. However, it is the approach that classifies them by the nature and the behaviour of automated processes that is most commonly used. According to the latter, workflows are divided into three groups: production workflows, administrative workflows and ad-hoc workflows [7]. Production workflows are those automating highly structured processes that experience very little (or no) change over time. Administrative workflows apply to variable processes of which all cases are known; that means that tasks are predictable and their sequencing are simple and clearly defined (these are the ones that are of particular interest to the work we are doing). Ad-hoc workflows are more general; they automate occasional processes for which it is not always possible to define all the rules in advance.

Business process specification In the literature, the specification of a business process is commonly referred to as a *workflow model*. According to [2], a workflow model consists of three main conceptual models: the *organizational*, *informational* and *process* models.

The *organizational model* is used to express and classify the resources responsible for executing the tasks of the studied process. Generally, these are classified into *roles* to which tasks are assigned.

The *informational model* is used to describe the structure of consumed and produced data during processes execution.

Finally, the *process model* is used to describe the structure of each task, the coordination between them and consequently, the coordination between the various actors involved in their execution. The process model is generally expressed using a language and allows the expression of basic control flows (*sequential*, *parallel*, *alternative* and *iterative*) between tasks.

Ideally, a workflow language should be able to allow workflow model designers to express these three conceptual models.

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

In this section, we present the language LSAWfP. It is a new language that allows to specify administrative workflow processes with a particular emphasis on the consideration of accreditations.

3.1 Language Definition

In LSAWfP, each administrative process is specified using a triplet composed of: a grammatical model (called *Grammatical Model of Workflow* - GMWf - thereafter), a list of actors and a list of accreditations. The GMWf is used to describe all the tasks of the studied process and the precedence of execution between them, while the list of accreditations provides information on the role played by each actor involved in the process execution.

In the rest of this manuscript, any specification of a business process produced using the language LSAWfP will be called *a Grammatical Model of Administrative Workflow Process* (GMAWfP). A GMAWfP is therefore a triplet formally defined as follows:

Definition 1. *A **Grammatical Model of Administrative Workflow Process** (GMAWfP) $\mathbb{W}_f$ for a given business process, is a triplet $\mathbb{W}_f = \left(\mathbb{G}, \mathcal{L}_{\mathcal{P}_k}, \mathcal{L}_{\mathcal{A}_k}\right)$ wherein $\mathbb{G}$ is the studied process (global) GMWf, $\mathcal{L}_{\mathcal{P}_k}$ is the set of k actors taking part in its execution and $\mathcal{L}_{\mathcal{A}_k}$ represents the set of these actors accreditations.*

Concept of GMWf For a given process, the GMWf is the mathematical instrument that allows to specify all the tasks to be executed as well as the control flow (also called *routing*) that allows to schedule them. It is a grammatical model based on the observation that: the set of tasks of a given administrative process and their execution precedence orders can be described using a (finite) set of annotated trees (see fig. 1). Each of these trees, called *target artifact*, is a task graph representing one of the possible execution scenarios of the studied process. In fact, it is sufficient to consider in each target artifact that, nodes represent the different tasks to be executed and each hierarchical decomposition (a node and its sons) represents an ordering.

For a given set of tasks $\{X_0, X_{s1} \dots X_{sn}\}$, we consider two types of ordering simply specified using two types of decorated productions[6]: (1) *sequential ordering*, noted $X_0 \rightarrow X_{s1} \fatsemi X_{s2} \fatsemi \dots \fatsemi X_{sn}$, which specifies that task X_0 precedes (ie. must be executed before all) tasks $X_{s1}, \dots, X_{sn}$ which are to be executed in sequence (X_{s1} must precede X_{s2}, ...) and, (2) *parallel odering*, noted $X_0 \rightarrow X_{p1} \parallel X_{p2} \parallel \dots \parallel X_{pn}$, which specifies that task X_0 must be executed before tasks $X_{p1}, X_{p2}, \dots, X_{pn}$ which can be executed concurrently.

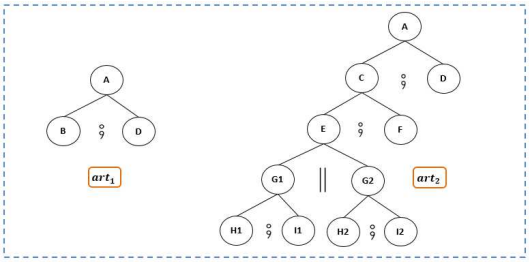

Fig. 1. Example of target artifacts for a given process (peer-review process)

From the above observations, it is easy to deduce that all the target artifacts of a given administrative process, form an algebraic tree language. It can therefore be defined by a grammar $\mathbb{G}$ (a GMWf) in which, each symbol (sort) corresponds to a task of the studied process and, each production (p) is of one of the two following forms: $p : X_0 \rightarrow X_1 \fatsemi \dots \fatsemi X_n$ or $p : X_0 \rightarrow X_1 \parallel \dots \parallel X_n$. Each target artifact t_i *is conform to* $\mathbb{G}$ and we note $t_i \therefore \mathbb{G}$. We can thus define a GMWf more formally in the following way:

[6] Decorations are made using the operators "⨟" (*is sequential to*) for sequential ordering and "||" (*is parallel to*) for parallel odering.

Definition 2. *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; $\mathcal{A} \subset \mathcal{S}$ is a finite set of particular symbols called **axioms**, representing tasks that can start an execution scenario, and $\mathcal{P} \subseteq \mathcal{S} \times \mathcal{S}^*$ is a finite set of **productions** decorated by the operators "$\fatsemi$" (is sequential to) and "$\|$" (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 $P : X_0 \to X_1 \fatsemi \ldots \fatsemi X_{|P|}$, or of the form $P : X_0 \to X_1 \parallel \ldots \parallel X_{|P|}$ and $|P|$ designates the length of P right-hand side. Each grammatical symbol $X \in \mathcal{S}$ is associated with an attribute called **status**, that can be updated when task X is executed; **X.status** provides access (read and write) to its content. A production with the symbol X as left-hand side is called a X-production.*

For some business processes, there may be particular cases where it would be impossible to strictly order all tasks using the (only) two retained production forms for GMWf. This is for example the case of a process with four tasks A, B, C and D such that: task A precedes all the others, tasks B and C can be executed concurrently and precede D. In these cases, the introduction of a given number of new symbols known as *(re)structuring* ones (not associated with tasks), can make it possible to produce a correct ordering that respects the form imposed on productions. For the previous example, introducing a new symbol S allows us to obtain the following productions: $p_1 : A \to S \fatsemi D$, $p_2 : S \to B \parallel C$, $p_3 : B \to \varepsilon$, $p_4 : C \to \varepsilon$ and $p_5 : D \to \varepsilon$ that model the proper ordering required for this process. To deal with such cases, we adjust the previously given definition of GMWf (definition 2) by integrating *(re)structuring symbols* into it; the resulting definition is as follows:

Definition 3. *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 2, $\mathcal{S} = \mathcal{T} \cup \mathcal{T}_{Struc}$ 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.*

Defined in this way, GMWf allow basic control flows (*sequential*, *parallel*, *alternative* and *iterative*) to be expressed between tasks as illustrated in section 3.3.

Concept of accreditation of an actor As business processes are generally executed collectively, it is necessary to set up mechanisms to ensure better coordination between the various actors and to guarantee the confidentiality of certain actions and data: this is the purpose of accreditation. With it, we propose to take these aspects into account during the workflow system design phase. The accreditation of a given actor provides information on its rights (permissions) relatively to each sort (task) of the studied process's GMWf. The nomenclature of rights that we handle and that we want simple, 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). An actor's ***view*** is 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/realize 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.*

3.2 Example of specification using LSAWfP

As an illustrative example, consider the process of validating an article in a peer-reviewed scientific journal commonly referred to as peer-review process. The latter can be briefly described as follows:

- The process is triggered when the editor in chief receives a paper for validation submitted by one of the authors who participated in its drafting;
- After receipt, the editor in chief 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.);
- If the submission is rejected, he writes a report then notifies the corresponding author and the process ends;
- In the other case, he chooses an associated editor and sends him the paper for the continuation of the validation;
- The associated editor prepares the manuscript, forms a referees committee (two members in our case) and then triggers the peer-review process;
- Each referee reads, seriously evaluates the paper and sends back a message and a report to the associated editor;
- After receiving reports from all referees, the associated editor takes a decision and informs the editor in chief who sends the final decision to the corresponding author.

Figure 2 shows the BPMN orchestration diagram corresponding to the graphical description of this peer-review process.

To specify this process using our language, we will proceed in four distinct steps during which we will produce each of the components of the triplet $\mathbb{W}_f = \left(\mathbb{G}, \mathcal{L}_{P_k}, \mathcal{L}_{\mathcal{A}_k}\right)$.

Step 1: *identification and ordering of process tasks* From the description of the peer-review process made previously, it is easy to identify all the tasks to be executed, all the actors involved as well as the tasks assigned to them. A summary of this assignment is presented in table 1.

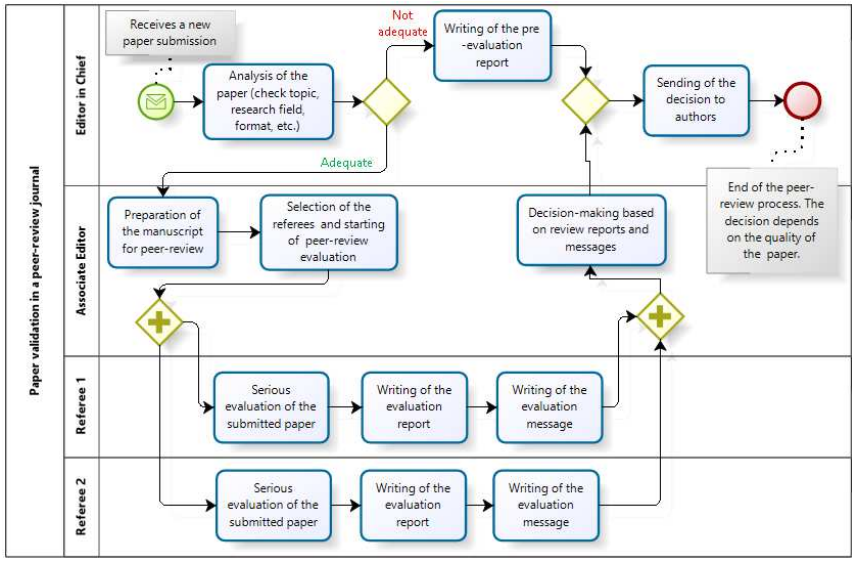

Fig. 2. BPMN orchestration diagram of the peer-review process.

From the analysis of the execution precedence constraints that exist between the highlighted tasks (see table 1), we obtain the target artifacts art_1 and art_2 of figure 1. For example, the target artifact art_1 in figure 1 shows how the "*Receipt and pre-validation of a submitted article*" task, executed by the editor in chief (EC) and to which sort A has been associated (for readability purposes - see table 1), must be executed before the two sequential tasks associated respectively to sorts B and D. This target artifact represents the scenario where the article received by the editor in chief is immediately rejected for form issues, research domain incompatibility, etc.

Step 2: *deduction of the Grammatical Model of Workflow* ($\mathbb{G} = (\mathcal{S}, \mathcal{P}, \mathcal{A})$) When analyzing this example's target artifacts, we deduce that the set of grammatical symbols $\mathcal{S}$ is: $\mathcal{S} = \{A, B, C, D, E, F, G1, G2, H1, H2, I1, I2\}$ (see table 1); the only initial task (axiom) is A (hence $\mathcal{A} = \{A\}$) and the set $\mathcal{P}$ of productions is:

$P_1 : A \to B \fatsemi D$	$P_2 : A \to C \fatsemi D$	$P_3 : C \to E \fatsemi F$	$P_4 : E \to G1 \parallel G2$
$P_5 : G1 \to H1 \fatsemi I1$	$P_6 : G2 \to H2 \fatsemi I2$	$P_7 : B \to \varepsilon$	$P_8 : D \to \varepsilon$
$P_9 : F \to \varepsilon$	$P_{10} : H1 \to \varepsilon$	$P_{11} : I1 \to \varepsilon$	$P_{12} : H2 \to \varepsilon$
$P_{13} : I2 \to \varepsilon$			

Step 3: *actors involved in the execution of the process* ($\mathcal{L}_{P_k}$) According to our description of the peer-review process, four ($k = 4$) actors participate in its execution: an editor in chief (EC), an associated editor (AE) and two referees ($R1$ and $R2$). So we deduce that $\mathcal{L}_{P_k} = \{EC, AE, R1, R2\}$. It should be noted that the notion of actor here does not

Tasks	**Associated Symbols**	**Executors**
Receipt, pre-validation of a submitted paper and possible choice of an associated editor to lead peer-review evaluation	A	EC
Drafting of a pre-validation report informing on the reasons for the immediate rejection of the paper	B	EC
Sending the final decision (acceptance or rejection of the paper) to the author	D	EC
Study, eventually formatting of the paper for the examination by a committee	C	AE
Constitution of the reading committee (selection of referees) and triggering the peer-review evaluation	E	AE
Decision making (paper accepted or rejected) from referees evaluations	F	AE
Evaluation of the manuscript by the first (resp. second) referee	$G1$ (resp. $G2$)	$R1$ (resp. $R2$)
Drafting of the after evaluation report by the first (resp. second) referee	$H1$ (resp. $H2$)	$R1$ (resp. $R2$)
Writing the message according to evaluation by the first (resp. second) referee	$I1$ (resp. $I2$)	$R1$ (resp. $R2$)

Table 1. Exhaustive tasks list of a paper validation process in a scientific journal and their respective performers.

necessarily refer to a specific natural person; it refers more precisely to a role that can be assumed by several natural persons with the same skills.

Step 4: *the accreditation of each participant ($\mathcal{L}_{\mathcal{A}_k}$)* From the assignment of tasks to actors (see table 1), it follows that the accreditation in writing of the editor in chief is $\mathcal{A}_{EC(w)} = \{A,B,D\}$, the one of the associated editor is $\mathcal{A}_{AE(w)} = \{C,E,F\}$ and that of the first (resp. the second) referee is $\mathcal{A}_{R_1(w)} = \{G1,H1,I1\}$ (resp. $\mathcal{A}_{R_2(w)} = \{G2,H2,I2\}$). Moreover, since the editor in chief can only execute task D if task C is already executed (see artifacts art_1 and art_2, fig. 1), for the editor in chief to be able to request this task execution from the associated editor, he must be accredited in execution on it; therefore, we have $\mathcal{A}_{EC(x)} = \{C\}$. In addition, in order to be able to access all the information on the progress of the peer-review evaluation (task C) and synthesize the right decision to be sent to the author, the editor in chief must be able to consult reports (tasks $I1$ and $I2$) and messages (tasks $H1$ and $H2$) of the various referees, as well as the final decision made by the associated editor (task F). These tasks, in addition to $\mathcal{A}_{EC(w)}$[7] constitute the set $\mathcal{A}_{EC(r)} = \mathcal{V}_{EC} = \{A,B,C,D,H1,H2,I1,I2,F\}$ of tasks on which he is accredited in reading. Doing so for each of the other actors leads to the deductions of the accreditations represented in the table 2 and we have $\mathcal{L}_{\mathcal{A}_k} = \{\mathcal{A}_{EC},\mathcal{A}_{AE},\mathcal{A}_{R1},\mathcal{A}_{R2}\}$.

[7] Remember that in our case we can only execute what we see.

Actor	**Accreditation**
Editor in Chief (*EC*)	$\mathcal{A}_{EC} = (\{A,B,C,D,H1,H2,I1,I2,F\},\{A,B,D\},\{C\})$
Associated Editor (*AE*)	$\mathcal{A}_{AE} = (\{A,C,E,F,H1,H2,I1,I2\},\{C,E,F\},\{G1,G2\})$
First referee (*R*1)	$\mathcal{A}_{R1} = (\{C,G1,H1,I1\},\{G1,H1,I1\},\emptyset)$
Second referee (*R*2)	$\mathcal{A}_{R2} = (\{C,G2,H2,I2\},\{G2,H2,I2\},\emptyset)$

Table 2. Accreditations of the different actors taking part in the peer-review process.

3.3 On the expressiveness of LSAWfP

In this subsection, we want to show that LSAWfP has all the expected characteristics of a workflow language. In particular, we show that each of its instances (i.e. a specification of a business process in this language) contains both an organizational model, an informational model and a process model.

Let's consider a specification $\mathbb{W}_f = \left(\mathbb{G}, \mathcal{L}_{\mathcal{P}_k}, \mathcal{L}_{\mathcal{A}_k}\right)$ of a given business process $\mathcal{P}_{op}$. The organizational model of $\mathcal{P}_{op}$ that expresses and classifies/assigns the resources that must execute its tasks is given by the couple $\left(\mathcal{L}_{\mathcal{P}_k}, \mathcal{L}_{\mathcal{A}_k}\right)$ of $\mathbb{W}_f$. Its informational model that describes the data structure being manipulated is given by the type of the attribute *status*[8]. LSAWfP does not impose any constraints on the type of this attribute and leaves the responsibility to the designer to specify it; by default it is a string type. The process model of $\mathcal{P}_{op}$ that provides information on the tasks and their sequencing (coordination) is then given by the GMWf $\mathbb{G}$ of $\mathbb{W}_f$.

Let's take a moment to look at the process model contained in a specification made in LSAWfP, to show that it effectively allows the designers to specify all the basic control flows (sequential, parallel, alternative and iterative) that they can find in traditional workflow languages. Figure 3 gives for each type of control flow its BPMN notation and the corresponding notations (tree 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 $q : S \rightarrow A\,;B$ 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 iterative flow on the task A (see fig. 3(d)).

Note that, when the process to be specified contains an iterative routing (modeled by a cycle in the task graph according to the BPMN notation (see fig. 3(d))), it is impossible to list exhaustively all the set of its target artifacts (execution scenarios) because

[8] Reminder: each task is represented by a grammatical symbol with an attribute named *status* (see definition 2)

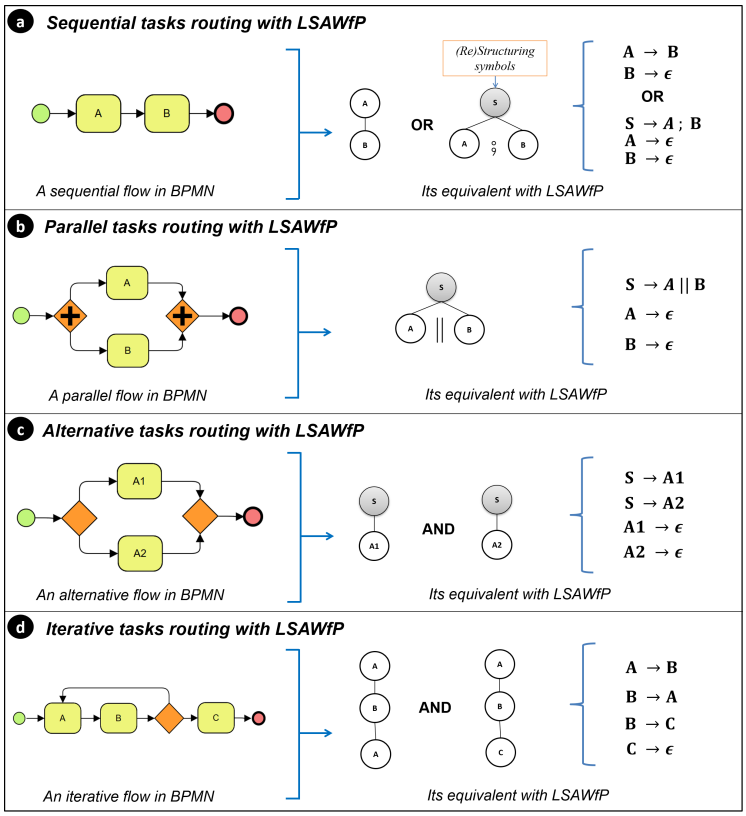

Fig. 3. Illustrating basic control flow with LSAWfP.

the latter is infinite. In this case, we propose to represent it by all its generators: generators are a finite and minimal set of artifacts allowing to represent each artifact as a combination/juxtaposition of generators; each artifact is therefore decomposable into a set of sub-artifacts all belonging to the set of generators. A generator is a target artifact for which each of its branches (from the root to a leaf) contains a given symbol only twice at most. Operationally, when designing the target artifacts of a given process, the designer must prune each branch as soon as he encounters a symbol for the second time. This will provide a finite set of target artifacts (generators) whose elements, combined with each other, represent the set of possible execution scenarios for the studied process. That is what was done to obtain the two target artifacts shown in figure 3(d).

3.4 Privileged WfMS architecture that must execute instances of LSAWfP

Process specifications in LSAWfP (GMAWfP) can be easily executed in a distributed way, by fully decentralized WfMS, offering an artifact-centric execution of business

processes. It is this type of WfMS, later called *P2P-WfMS-View*[9], that we describe in this section.

A *P2P-WfMS-View* is a set of components distributed on all the sites where various workflow actors operate. These different components (hereinafter referred to as *peers*) have the same architecture, execute the same protocols, communicate by service calls and cooperate in P2P to execute a GMAWfP. On each peer, a set of three (03) software components that manage the entire lifecycle (creation, storage, execution) of workflows is executed. These are (see fig. 4): a *local workflow engine* (LWfE), a *specialized graphic editor* and a *storage device*. The *local workflow engine* (LWfE) manages the life cycle of incoming requests on a given site. It communicates with engines of other peers via its communication interface which exposes four services: two input services or *provided services* (*returnTo* and *forwardTo* for processing requests/responses) connected to two corresponding output services or *required service* (*returnTo* and *forwardTo* for sending requests/responses) (see fig. 4). The *storage device* is a database (DB) of documents (a JSON[10] database for example) used by the LWfE to store the state of each workflow that it manages. The *specialized editor* allows the local actor to access process data, access and execute tasks assigned to him. It is important to note that on a given site, the specialized editor only gives access to information relevant to the local actor; i.e. those for which he has sufficient accreditation. It therefore guarantees that each actor has only a potentially partial perception of the executed processes.

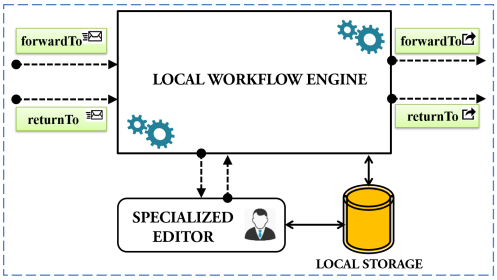

Fig. 4. Simplified peer architecture.

During the execution of a given GMAWfP, each peer keeps locally a copy of the (global) artifact representing the current execution state of the considered process. It is also the latter that serves as a medium for communication and coordination between actors: it is in this sense that the execution is *artifact-centric*.

The (global) artifact is in conformity with the GMWf of the considered GMAWfP and provides information on already executed tasks, on those ready to be immediately executed as well as on their executors. In fact, when a given actor acts on the workflow (by executing his tasks through the specialized editor for example), his local copy of

[9] Peer to Peer Workflow Management Systems with emphasis on actor's Views.
[10] JavaScript Object Notation, http://www.json.org, https://www.mongodb.com

the (global) artifact is updated accordingly. In order to synchronize, actors exchange (through service invocations) their local copies of the (global) artifact and these are merged each time, to obtain a coherent state of process execution before it is continued. In this way, we succeed in using the unique and simple artifact formalism, as a mechanism for the specification of process models and as a model of the exchange and coordination between actors mediums. It should be noted that existing solutions generally use at least two formalisms for the same needs.

For experimentation purposes, we have produced a P2P-WfMS-View prototype called *P2PTinyWfMS*[11] through which we can simulate the completely decentralized execution of processes. In accordance with P2P-SGWf-View architecture (see fig. 4), *P2PTinyWfMS* has a front-end for displaying and graphically editing artifacts handled when executing a business process (see fig. 5 and 6), as well as a communication module built using SON[12] (Shared-data Overlay Network) [5]; SON is a middleware offering several DSL (Domain Specific Language) to facilitate the implementation of P2P systems whose components communicate by services invocations.

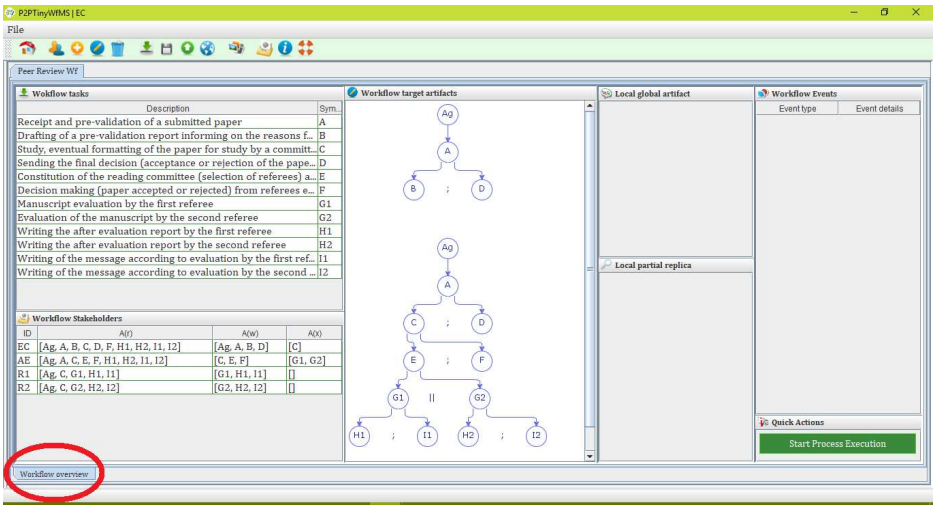

Fig. 5. P2pTinyWfMS on the editor in chief's site: presentation of the GMAWfP (tasks and their relationships, actors and their accreditations).

In order to execute our running example (the peer-review process), we deployed four instances of *P2PTinyWfMS* respectively identified by *EC*, *AE*, *R*1 and *R*2. Figures 5 and 6 are screenshots showing some highlights of the workflow's distributed execution. For example, on figure 5, the tab "*Workflow overview*" presents at the beginning of the execution, various tasks, actors, target artifacts etc., on the editor in chief's site. Figure

[11] *P2PTinyWfMS* is a tool developed in Java under Eclipse (*https://www.eclipse.org*).

[12] SON is available under Eclipse from SmartTools plugin family.

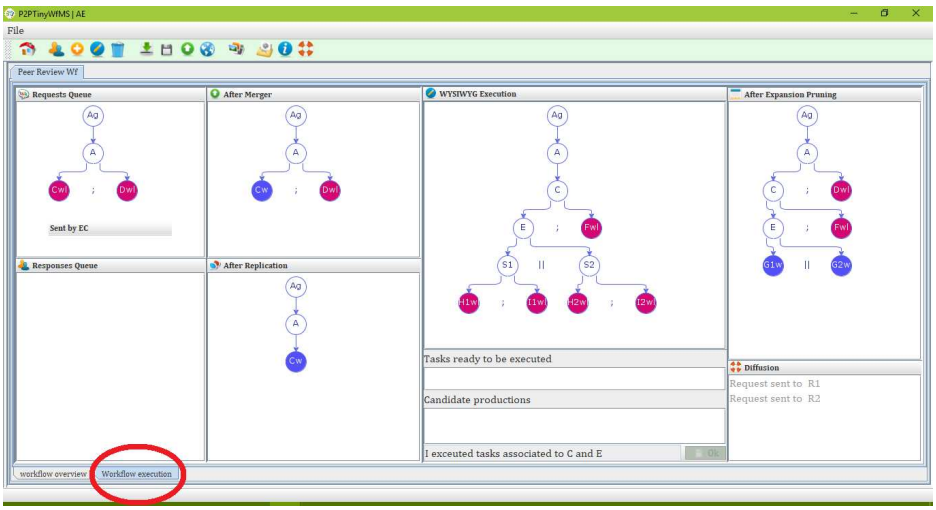

Fig. 6. Simulation of the execution of the peer-review process using P2PTinyWfMS.

6 is a screenshot of the tab "*Workflow execution*" made on the associated editor's site; it shows artifacts resulting from processing performed after the receipt of a request from the editor in chief. This figure 6 actually reveals that: the associated editor received an artifact under execution (fig. 6 (Requests Queue)) from the editor in chief's site; then, after the merging and replication operations performed by the LWfE, task *D* and its data were hidden to the associated editor (he does not have sufficient accreditations on the latter) while task *C* was proposed to him for execution. With the specialized editor, the associated editor has accessed and executed ready tasks one after the other until he could not continue; his partial copy of the global artifact was updated accordingly (fig. 6 (WYSIWYG Execution)). Finally, the LWfE has calculated the overall process execution state on the associated editor's site through an operation called *expansion-pruning* and has sent requests to referees' sites on which execution was supposed to continue concurrently.

4 Conclusion

In this paper, we have proposed a new workflow language called LSAWfP which allows, through a simple grammar-based formalism, to specify business processes. Like any workflow language, LSAWfP allows to specify basic flows (sequential, parallel, alternative and iterative) that are generally found in workflow models; particularly, it allows (unlike other languages) to address certain security aspects of administrative workflows. In fact, LSAWfP allows the workflow models designers, to simply express each actor's accreditations for each task in a process, by the means of a formalism inspired by that used in UNIX-like operating systems for the expression of users' rights.

The utility and usability of LSAWfP has been satisfactorily tested through an experiment of its use for the implementation of a distributed environment to execute a

peer-review process; this environment has been briefly presented in this paper. However, this experiment suggested that it would certainly be easier to handle LSAWfP if we had a (graphical) tool to assist in the design and validation of its instances. Moreover, it seems equally important to more precisely describe the model for executing business processes specified in LSAWfP. In our opinion, this is just a few of the many studies that must be carried out following the one presented in this paper.

References

1. Badouel, E., Hélouët, L., Kouamou, G.E., Morvan, C.: A Grammatical Approach to Data-centric Case Management in a Distributed Collaborative Environment. CoRR **abs/1405.3223** (2014), http://arxiv.org/abs/1405.3223
2. Divitini, M., Hanachi, C., Sibertin-Blanc, C.: Inter-organizational workflows for enterprise coordination. In: Coordination of Internet agents, pp. 369–398. Springer (2001)
3. Georgakopoulos, D., Hornick, M.F., Sheth, A.P.: An Overview of Workflow Management: from Process Modeling to Workflow Automation Infrastructure. Distributed and Parallel Databases **3**(2), 119–153 (1995). https://doi.org/10.1007/BF01277643, https://doi.org/10.1007/BF01277643
4. Jordan, D., Evdemon, J., Alves, A., Arkin, A., Askary, S., Barreto, C., Bloch, B., Curbera, F., Ford, M., Goland, Y., et al.: Web services business process execution language version 2.0. OASIS standard **11**(120), 5 (2007)
5. Lahcen, A.A., Parigot, D.: A Lightweight Middleware for Developing P2P Applications with Component and Service-Based Principles. In: 15th IEEE International Conference on Computational Science and Engineering, CSE 2012, Paphos, Cyprus, December 5-7, 2012. pp. 9–16 (2012). https://doi.org/10.1109/ICCSE.2012.12, https://doi.org/10.1109/ICCSE.2012.12
6. Model, B.P.: Notation (BPMN) version 2.0. OMG Specification, Object Management Group pp. 22–31 (2011)
7. Van Der Aalst, W.M.P.: The Application of Petri Nets to Workflow Management. Journal of Circuits, Systems, and Computers **8**(1), 21–66 (1998). https://doi.org/10.1142/S0218126698000043, https://doi.org/10.1142/S0218126698000043
8. Van Der Aalst, W.M.P., Ter Hofstede, A.H.: Yawl: yet another workflow language. Information systems **30**(4), 245–275 (2005)
9. WfMC: Wfmc Standards: the Workflow Reference Model, Version 1.1. http://www.aiim.org/wfmc/mainframe.htm (1995), http://www.aiim.org/wfmc/mainframe.htm
10. Yan, J., Yang, Y., Raikundalia, G.K.: SwinDeW-a P2P-Based Decentralized Workflow Management System. IEEE Trans. Systems, Man, and Cybernetics, Part A **36**(5), 922–935 (2006). https://doi.org/10.1109/TSMCA.2005.855789, https://doi.org/10.1109/TSMCA.2005.855789