Improving the Projection-Based and Artifact-Centric Decentralized Execution of LSAWfP Models

Milliam Maxime Zekeng Ndadji, Baudouin Nankeng Meli, Maurice Tchoupé Tchendji

Improving the Projection-Based and Artifact-Centric Decentralized Execution of LSAWfP Models

M. M. ZEKENG NDADJI[*1,2], B. NANKENG MELI[1], M. TCHOUPÉ TCHENDJI[*1,2]

[1]Department of Mathematics and Computer Science, University of Dschang
[2]FUCHSIA Research Associated Team, https://project.inria.fr/fuchsia/

*E-mail : {ndadjimaxime, ttchoupe}@yahoo.fr

Abstract

With the development of artifact-centric Business Process Management (BPM) models, the search for fully decentralized and less complex BPM techniques re-emerged in the last decade. In this context, the Language for the Specification of Administrative Workflow Processes (LSAWfP) and a fully decentralized execution model of its specifications have been proposed between 2019 and 2021. These new tools make it possible to specify business processes by means of a grammatical model and, to execute them in a decentralized mode via a protocol whose key algorithms (called projections) are intended to guarantee the confidential execution of certain tasks. In this paper, improvements to this execution model are proposed in order to overcome two previously formulated (disabling) assumptions. The resulting execution model allows the use of iterative routing in the specification of processes to be executed (increase of expressiveness); furthermore, the performance of the execution model's key algorithms are significantly improved while preserving properties of the original model as well as possible.

Keywords

Decentralized BPM; LSAWfP; Projection; Grammars; Views

I INTRODUCTION

During the last two decades, many research works on Business Process Management (BPM) have focused on the artifact-centric paradigm. This paradigm was first developed in the work of Nigam and Caswell [1]. The artifact-centric paradigm maintains the traditional BPM modus operandi which is that, automating a business process requires two phases: the *modelling phase* during which the process is studied and then specified according to a so-called workflow language, and the *execution phase* during which the obtained specification is used in a BPM System (BPMS); the said BPMS is then in charge of orchestrating process tasks' execution by the different actors (the BPMS facilitates exchanges and coordination between them) [2]. However, the artifact-centric paradigm brings the novelty whereby, a given process modelling is equivalent to the modelling of a data structure called *artifact*; this artifact can be seen as a structured document cooperatively edited by the actors, and containing details on the tasks to be executed, the data necessary for the execution as well as the data produced during the execution of these tasks [3].

The advent of the artifact-centric paradigm has boosted research on process modelling and

their execution in a completely decentralized mode. It is in this context that Zekeng et al. [4] have proposed a new workflow language called LSAWfP (a Language for the Specification of Administrative Workflow Processes), allowing to specify processes in the form of context-free grammars called *Grammatical Models of Workflow* (GMWf). They also developed a completely decentralized execution model [5], of processes specified with a version of LSAWfP and whose GMWf do not admit recursivity[1]. The said execution model treats the execution of a given process as the cooperative edition by the actors, of an artifact (an annotated tree considered as a structured document) which circulates from site to site according to its state. In order for the execution to proceed correctly while respecting constraints initially defined, such as the confidentiality regarding the execution of certain tasks, the authors have proposed several algorithms for the projection of artifacts and their models. Two assumptions (the *axiom's visibility assumption* and the *non-recursive GMWf assumption* [5]) that limit the expressiveness of the considered workflow language, have also been formulated so that the algorithms can succeed.

The work in this paper overcomes the axiom's visibility and non-recursive GMWf assumptions formulated in the work of Zekeng et al. [5]. New behaviors are therefore provided for the different projection algorithms. As a result, a new completely decentralized execution model (an improved version of the previous one) of processes specified using LSAWfP is obtained. The LSAWfP models[2] are actually executed even if they are recursive: the new execution model allows a more expressive processes' specification. Moreover, the obtained new versions of projection algorithms are more efficient than the previous ones as revealed by the study of their theoretical complexity.

In the rest of this paper, the model proposed by Zekeng et al. for the specification and the completely decentralized execution of business processes is presented in section II. In section III, the contribution of this paper and an illustrative example are presented. Sections IV and V are respectively dedicated to the discussion of the conducted work and to the conclusion.

II OVERVIEW OF THE MODELLING AND THE DECENTRALIZED EXECUTION OF BUSINESS PROCESSES USING LSAWFP

2.1 Business process modelling with LSAWfP

A workflow is generally composed of a collection of tasks, a set of actors, and dependencies between tasks. Tasks correspond to individual steps in a business process, actors are responsible for the enactment of tasks, and dependencies determine the execution sequence of tasks and the data flow between them. The constituent units of processes (tasks, actors, etc.) are often referred to as *process elements*. Generally, the main purpose of a workflow language is to provide tools (graphical or not) to represent process elements and the relations between them.

LSAWfP is a workflow language wherein process elements are specified using three mathematical tools: a GMWf, a list of actors, and a list of accreditations. More precisely, with LSAWfP, the specification of a given business process $\mathcal{P}_{op}$, results in a triplet $\mathbb{W}_f = (\mathbb{G}, \mathcal{L}_{P_k}, \mathcal{L}_{\mathcal{A}_k})$ called *a Grammatical Model of Administrative Workflow Process* (GMAWfP) and composed of a GMWf $\mathbb{G}$, a list of actors $\mathcal{L}_{P_k}$ and a list of their accreditations $\mathcal{L}_{\mathcal{A}_k}$ [4]. The GMWf is used to describe all the tasks of the studied process and their scheduling (the process control flow),

[1] Informally, a non-recursive grammar is a grammar in which there is no non-terminal symbol whose expansion using productions allows to obtain a string containing this same symbol.

[2] The obtained specification after modelling a process with LSAWfP is called a LSAWfP model.

while the list of accreditations provides information on the role played by each actor involved in the process execution. The GMWf is a grammar $\mathbb{G} = (\mathcal{S}, \mathcal{P}, \mathcal{A})$ for a tree language, in which:

- $\mathcal{S}$ is a finite set of **grammatical symbols** or **sorts** corresponding to various process **tasks**;
- $\mathcal{A} \subseteq \mathcal{S}$ 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 annotations "$\wp$" (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 $P : X_0 \to X_1 \wp \cdots \wp X_{|P|}$, or of the form $P : X_0 \to X_1 \| \cdots \| X_{|P|}$. The first form $P : X_0 \to X_1 \wp \cdots \wp X_{|P|}$ (resp. the second form $P : X_0 \to X_1 \| \cdots \| X_{|P|}$) means that task X_0 must be executed before tasks $\left\{X_1, \cdots, X_{|P|}\right\}$ that must be (resp. can be) executed in sequence (resp. in parallel) from the left to the right. A production with the symbol X as left-hand side is called a *X-production*. Given a production P, $|P|$ designates the length of its right-hand side.

Regarding the list of accreditations, each of its elements is a triplet $\mathcal{A}_{A_i} = \left(\mathcal{A}_{A_i(r)}, \mathcal{A}_{A_i(w)}, \mathcal{A}_{A_i(x)}\right)$ of grammatical symbols lists indicating for a given actor A_i, its rights (permissions) relatively to each sort (task) of the studied process's GMWf. In $\mathcal{A}_{A_i}$:

- $\mathcal{A}_{A_i(r)} \subseteq \mathcal{S}$ also called **view** of actor A_i, is the list of symbols on which A_i is accredited in reading; i.e. A_i has free access to the execution states (data generated during execution) of each symbol in $\mathcal{A}_{A_i(r)}$. Any sort of $\mathcal{A}_{A_i(r)}$ is said to be *visible* by A_i, and those not belonging to $\mathcal{A}_{A_i(r)}$ are said to be *invisible*.
- $\mathcal{A}_{A_i(w)} \subseteq \mathcal{A}_{A_i(r)}$ is the list of symbols on which A_i is accredited in writing; i.e. A_i is the actor in charge of executing each symbol of $\mathcal{A}_{A_i(w)}$.
- $\mathcal{A}_{A_i(x)} \subseteq \mathcal{S}$ is the set of symbols on which A_i is accredited in execution; i.e. A_i is allowed to request that the execution of each symbol in $\mathcal{A}_{A_i(x)}$ be carried out.

Examples of processes modelled with LSAWfP as well as additional details for understanding this language are given in [4, 5]; the reader is invited to consult these works if necessary.

2.2 Key steps of the decentralized execution of LSAWfP models and issues

Inspired by the previous work of Badouel and Tchoupé [6], Tchoupé and Zekeng [7] on the cooperative edition of structured documents, Zekeng et al. laid in [5], the foundations of a decentralized artifact-centric execution model of processes modelled with LSAWfP; they also formalized their model and studied its properties. To execute a given LSAWfP model (a GMAWfP), they consider a completely decentralized BPMS whose instances (the peers) are installed on the sites of the various actors. During the execution, these peers communicate (sending/receiving requests/responses) by exchanging copies of a (global) artifact said to be under execution. Such an artifact provides information on already executed tasks and on those ready to be executed. An artifact under execution is represented by a tree that contains *buds*. Buds indicate at a moment, the only tasks that will be executed. A bud can be either *unlocked* or *locked* depending on whether the corresponding task (node) is ready to be executed or not.

An important requirement of the model is to allow the confidential execution of certain tasks and to ensure a "*know what you should know*" policy [8] for the different actors. For this purpose, each actor acts on a potentially partial replica of the artifact: this partial replica contains only the information to which the concerned actor can have access. Technically, a partial replica $t_{\mathcal{V}_i}$ of an artifact t is obtained by projection (using an operator π said of ***artifact projection***) of t according to the view $\mathcal{V}_i$ of the concerned actor: it is noted $t_{\mathcal{V}_i} = \pi_{\mathcal{V}_i}(t)$. More precisely, according to the operator proposed by Zekeng et al., the projection $t_{\mathcal{V}_i}$ of an artifact t according to the view $\mathcal{V}_i = \mathcal{A}_{A_i(r)}$ is obtained by deleting in t all the nodes whose types do not belong to

$\mathcal{V}_i$ (all invisible nodes). However, this is done while ensuring that: (i) the nodes of $t_{\mathcal{V}_i}$ preserve the previously existing execution order between them in t, (ii) $t_{\mathcal{V}_i}$ is built by using exclusively the only two forms of production retained for GMWf and, (iii) $t_{\mathcal{V}_i}$ is unique in order to ensure the continuation of process execution.

To achieve the conditions (i), (ii), and (iii), two measures are taken. The first one is the creation and the use of new so-called (re)structuring symbols when necessary, in order to preserve the hierarchy between the nodes and to ensure that the form of productions is not altered. The second one is the restriction of the use of the proposed operator only in the case of LSAWfP models for which all axioms are visible by all actors (*axioms' visibility assumption*). This assumption ensures that in no case, the projection erases an axiom that is the root node of the projected artifact and produces a forest.

Axioms' visibility assumption issue. Knowing that axioms are tasks that can start one or several process execution scenarios, this second measure taken by Zekeng et al. is not very judicious for several real-life processes; in fact, this measure implies that all actors can be aware of the execution of each axiom and moreover, they have free access to the generated data during these executions. To overcome this, the authors propose for any given process, to extend its GMWf by adding a new single axiom whose only role will be to start the process execution. Once again, the problem of choosing the actor in charge of starting the process (i.e. executing the task associated with the new axiom) arises. According to the constraints of LSAWfP, only one actor must be able to do this; this contradicts again the reality of processes since there are some processes that can be started by various tasks executed by various actors.

With the artifact projection operation, the local actions of a particular actor depend on his view of the process. It is then necessary to control each actor's local actions, in order not only to preserve the possible confidentiality of certain tasks but also to ensure the consistency of local updates. To do this, Zekeng et al. propose to use a local GMWf on each site, to provide information on how the local executions should be carried out. A local GMWf $\mathbb{G}_{\mathcal{V}_i}$ is obtained, by projecting the global GMWf $\mathbb{G}$ according to the view $\mathcal{V}_i$ of the local actor (***GMWf projection***). This projection is carried out using Π operator and the GMWf obtained is noted $\mathbb{G}_{\mathcal{V}_i} = \Pi_{\mathcal{V}_i}(\mathbb{G})$. Concretely, the algorithm proceeds as follows: it generates the set of artifacts denoted by $\mathbb{G}$ then, it simply projects each artifact according to the view $\mathcal{V}_i$, and finally collects the productions in the obtained partial replicas while removing the duplicates. However, this GMWf projection algorithm only works for GMWf that do not allow recursive symbols (*non-recursive GMWf assumption*): it is only in that context that all the artifacts can be enumerated.

Non-recursive GMWf assumption issue. With this assumption, it is obvious that Zekeng et al. have reduced the expressiveness of the language used to specify the processes: it is no longer possible to use the iterative routing that is expressed by the presence of recursive symbols in the GMWf. One could think of applying their algorithm by generating only the set of representative artifacts (this is a finite set, even in the presence of recursive symbols) [4]. However, the projections of the obtained representative artifacts may not contain the exhaustive patterns to build the desired language (local GMWf) in some cases. Therefore, it is necessary to think of doing things differently.

To ensure system convergence, the contributions made by a given actor and contained in an updated partial replica $t_{\mathcal{V}_i}^{maj}$ at a given moment must be integrated into the (global) artifact before any synchronization with other peers. It is therefore, necessary to be able to merge these two artifacts, which are based on two different models. Zekeng et al. have proposed an

expansion operator to solve this issue; the said operator generates a merging guide (an artifact) and performs a three-way merge to obtain the new version of the (global) artifact. As for the efficiency, the study of the theoretical complexity of their algorithms shows that [5]: (a) the time complexity of the artifact-projection algorithm is $O(m^2)$ where m is the number of nodes of the artifact to be projected, (b) the necessary time to project a GMWf whose generated artifacts each have at most n nodes is about $O(mn^2)$ where m is the number of generated artifacts and, (c) the time complexity of the expansion algorithm is $O(mn^2)$ where m is the number of generated artifacts and n is the maximum number of nodes contained in those artifacts. These algorithms are therefore quite fast but can be improved.

III HIDING RATHER THAN DELETING NODES IN THE DECENTRALIZED EXECUTION OF LSAWFP MODELS

This paper's contribution has its roots in the following observation: in order to guarantee the confidential execution of certain tasks, Zekeng et al. made the "radical" choice of deleting nodes (during the projection) in the artifacts handled by the actors; it is this choice that subsequently "imposed" disabling assumptions in order to find a good balance. Instead of deleting nodes, we propose to simply hide them. More concretely, we propose to replace them with other nodes that do not carry any information that could possibly endanger the confidentiality of the tasks associated with the initial nodes. In this way, the structure of artifacts and models is not altered by the projection operations; this greatly simplifies these operations while guaranteeing an advantageous result. In the rest of this section, the new behavior of the execution model when the proposed change is applied will be presented.

3.1 Running example: the peer-review process

To facilitate the understanding of the concepts outlined in this section, we will illustrate them using the LSAWfP model for the peer-review process proposed in [4]. The tasks of the considered peer-review process are executed by an editor in chief (EC), an associate editor (AE) and two referees ($R1$ and $R2$). The accreditations of the latter are summarized in table 1. The

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.

productions of the GMWf $\mathbb{G} = (\mathcal{S}, \mathcal{P}, \mathcal{A})$ from this process are as follows:

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

In this example, $S1$ is a (re)structuring symbol introduced by the designer to guarantee the general form of the productions; $E1$ and $E2$ are recursive symbols. The only initial task (axiom) is A ($\mathcal{A} = \{A\}$).

3.2 Overview of the execution model and peer activity

Globally, the activity of a peer during the execution of a given process is slightly different from the one proposed in [5]. In fact, before the execution of a given process, each peer is configured using the process' GMAWfP ($\mathbb{W}_f = (\mathbb{G}, \mathcal{L}_{P_k}, \mathcal{L}_{\mathcal{A}_k})$). From the global GMWf $\mathbb{G}$ and the view $\mathcal{V}_i$ of the local actor, each peer derives by projection, a local GMWf $\mathbb{G}_{\mathcal{V}_i} = \Pi_{\mathcal{V}_i}(\mathbb{G})$. Then, the execution of a process case is triggered when an artifact t is introduced into the system (this can be done on any site where the actor is in charge of executing a task associated with an axiom); this artifact is an unlocked bud of the type of one axiom $A_{\mathbb{G}} \in \mathcal{A}$ (initial task) of the (global) GMWf $\mathbb{G}$ (see fig. 1). During execution, peers synchronize themselves by exchanging their local copies of the artifact being executed.

After receiving an artifact $t \therefore \mathbb{G}$ (i.e. t is conform to $\mathbb{G}$) on a given peer, the latter projects it (see Peer i in fig. 1) according to the local view $\mathcal{V}_i$. The obtained partial replica $t_{\mathcal{V}_i} \therefore \mathbb{G}_{\mathcal{V}_i}$ is then completed when needed: the result of this edition is an artifact $t^{maj}_{\mathcal{V}_i} \therefore \mathbb{G}_{\mathcal{V}_i}$ such as $t^{maj}_{\mathcal{V}_i}$ is an update of $t_{\mathcal{V}_i}$ ($t^{maj}_{\mathcal{V}_i} \geq t_{\mathcal{V}_i}$). At the end of the completion, the expansion of the obtained updated partial replica $t^{maj}_{\mathcal{V}_i} \therefore \mathbb{G}_{\mathcal{V}_i}$ is made in order to obtain the updated configuration $t_f \therefore \mathbb{G}$ of the (global) artifact local copy (see Peer i in fig. 1). If the resulting configuration shows that the process should be continued at other sites[3], then replicas of the artifact are sent to them. Else[4], a replica is returned to the peer from whom the artifact was previously received.

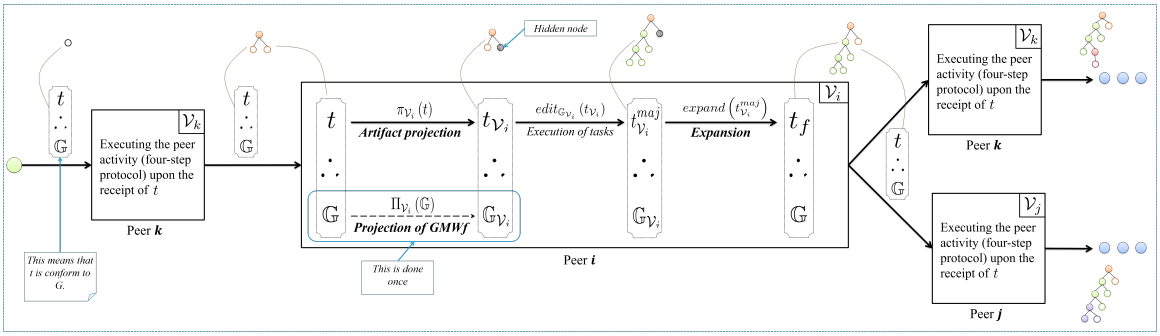

Figure 1: Overview of the distributed execution of a given process.

In addition to the differences that will be presented in the key algorithms of this execution model with the one proposed in [5], it can be noticed that the activity of a peer no longer takes place in five steps but rather in four steps: the merger activity that took place as soon as an artifact was received [9] is no longer necessary. Its role was to integrate nodes that could have been pruned during a previous expansion-pruning activity; in this case, there is no more pruning during/after the expansion.

3.3 The GMWf projection algorithm

Let us recall that this algorithm allows deriving from a global GMWf $\mathbb{G} = (\mathcal{S}, \mathcal{P}, \mathcal{A})$ and from a view $\mathcal{V}_i$, a local GMWf $\mathbb{G}_{\mathcal{V}_i} = (\mathcal{S}_{\mathcal{V}_i}, \mathcal{P}_{\mathcal{V}_i}, \mathcal{A}_{\mathcal{V}_i})$ to check the conformity of artifacts' partial replicas obtained after their projection. Since the main idea for artifacts projection is to avoid

[3]This is the case when the artifact contains buds created on the current peer and whose actors accredited in writing are on distant peers.

[4]The artifact is complete (it no longer contains buds), or semi-complete (it contains buds that were created on other peers and on which, the actor on the current peer is not accredited in writing).

deleting nodes in them, it is obvious that $\mathbb{G}_{\mathcal{V}_i}$ must be equivalent (structural equivalence[5] [10]) to $\mathbb{G}$ having renamed some symbols. Indeed, the renamed symbols are those which do not belong to the considered view $\mathcal{V}_i$. In this context, the local GMWf $\mathbb{G}_{\mathcal{V}_i}$ is useful for three goals: (1) indicating the precise type of node that should be created as a replacement for another node when projecting a given artifact, (2) providing the correct productions to be used when making local updates to partial replicas (executing tasks locally), and, (3) identifying nodes that have been replaced during projection and allowing them to be restored during expansion.

In the local GMWf, the renamed symbols are not associated with any task; in practice, they are just like (re)structuring symbols even though they communicate a completely different semantics. If it is derived from a recursive GMWf, then the local GMWf is recursive; hence its ability to guarantee consistency of local actions on recursive symbols. Moreover, considering that the action of renaming a symbol is done in constant time, the projection of a GMWf consisting of n symbols and p productions can be done in $O(np)$.

As an example, the following productions are those of the local GMWf at the editor in chief's site. The symbols $E1$, $E2$, $G1$, $G2$ and $S1$ not in the view $\mathcal{V} = \{A, B, C, D, H1, H2, I1, I2, F\}$ of the editor in chief, have been respectively renamed $Hi1$, $Hi2$, $Hi3$, $Hi4$ and $Hi5$.

$P_1 : A \rightarrow B \fatsemi D$	$P_2 : A \rightarrow C \fatsemi D$	$P_3 : C \rightarrow \underline{Hi5} \fatsemi F$	$P_4 : \underline{Hi5} \rightarrow \underline{Hi1} \parallel \underline{Hi2}$	$P_5 : \underline{Hi1} \rightarrow \underline{Hi3}$
$P_6 : \underline{Hi2} \rightarrow \underline{Hi4}$	$P_7 : \underline{Hi1} \rightarrow \underline{Hi1}$	$P_8 : \underline{Hi2} \rightarrow \underline{Hi2}$	$P_9 : \underline{Hi3} \rightarrow H1 \fatsemi I1$	$P_{10} : \underline{Hi4} \rightarrow H2 \fatsemi I2$
$P_{11} : B \rightarrow \varepsilon$	$P_{12} : D \rightarrow \varepsilon$	$P_{13} : F \rightarrow \varepsilon$	$P_{14} : H1 \rightarrow \varepsilon$	$P_{15} : I1 \rightarrow \varepsilon$
$P_{16} : H2 \rightarrow \varepsilon$	$P_{17} : I2 \rightarrow \varepsilon$			

3.4 The artifact projection algorithm

Technically, the projection $t_{\mathcal{V}_i}$ of an artifact t according to the view $\mathcal{V}_i = \mathcal{A}_{A_i(r)}$ is obtained by replacing in t all nodes whose types do not belong to $\mathcal{V}_i$ (all invisible nodes), with new nodes of corresponding types in the local GMMWf. Newly created nodes do not contain data even if they replace nodes that contained data; in this way, when manipulating the resulting partial replica, the local actor only has access to the process data he ought to have access to. The "know what you should know" policy intended by the authors of [5] is thus preserved. Moreover, the projection algorithm presented here satisfies the same three constraints imposed on their projection algorithm (see the second paragraph of section 2.2, page 4): i.e. (1) the nodes of $t_{\mathcal{V}_i}$ preserve the previously existing execution order between them in t, (2) $t_{\mathcal{V}_i}$ is built by using exclusively the only two forms of production retained for GMWf and, (3) $t_{\mathcal{V}_i}$ is a unique tree. Concerning its theoretical complexity, the projection of an artifact containing n nodes requires visiting each node only once; considering that the replacement operation of a given node is performed in constant time, the time required for this projection is of the order of $O(n)$. Figure 2 illustrates the projection of an artifact of the peer-review process relatively to the editor in chief's view. The hidden nodes have been represented in the partial replica with a red border.

3.5 The expansion algorithm

The considerations made for the artifact and the GMWf projection algorithms, impose changes on the expansion algorithm. Indeed, let us consider an artifact under execution t, and $t_{\mathcal{V}_i} = \pi_{\mathcal{V}_i}(t)$ its partial replica on the site of an actor whose view is $\mathcal{V}_i$; the expansion of the updated

[5]Two context-free grammars are defined as being structurally-equivalent if they generate the same sentences and assign similar parse trees (differing only in the labelling of the nodes) to each [10].

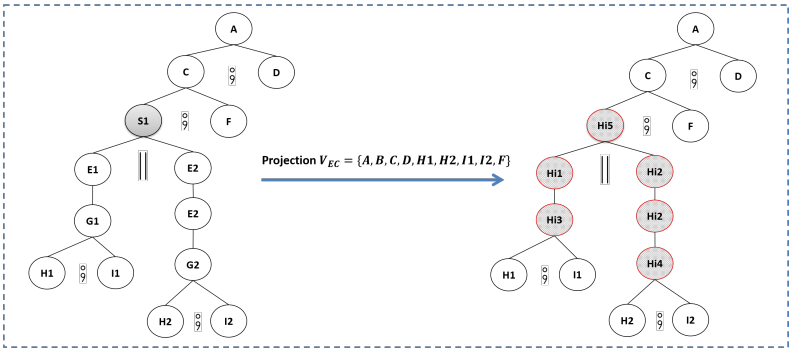

Figure 2: Example of projection made on an artifact and obtained partial replica.

partial replica $t^{maj}_{\mathcal{V}_i} \geq t_{\mathcal{V}_i}$ obtained by developing some unlocked buds of $t_{\mathcal{V}_i}$, now consists in merging t and $t_{\mathcal{V}_i}$. It is no longer necessary to look for a third artifact to guide the merger because, t and $t_{\mathcal{V}_i}$ are in conformity with two equivalent GMWf (the GMWf of $t_{\mathcal{V}_i}$ is the projection of the one of t according to the view $\mathcal{V}_i$).

In order to merge the two artifacts, a simultaneous deep path of them is performed. Depending on the nature, the state, and the address[6] of nodes $n_{t_i} \in t$ and $n_{t^{maj}_{\mathcal{V}_i}} \in t^{maj}_{\mathcal{V}_i}$ being visited, one of them is retained and inserted in the resulting artifact:

- If both nodes are located at the same address in their respective artifacts then, n_{t_i} is retained in the following cases: (i) n_{t_i} is closed or, (ii) the type of $n_{t^{maj}_{\mathcal{V}_i}}$ is a symbol created to hide another. In any other case, $n_{t^{maj}_{\mathcal{V}_i}}$ is retained. In all cases, the depth path of the two artifacts is continued.
- Otherwise, $n_{t^{maj}_{\mathcal{V}_i}}$ is retained; but if its type is a symbol created to hide another, then this type is changed by the original one before the node is inserted in the resulting artifact. This time, only the depth path of the $t^{maj}_{\mathcal{V}_i}$ artifact is continued.

This expansion algorithm takes about $O(n)$ time, where n is the number of nodes in $t^{maj}_{\mathcal{V}_i}$.

IV DISCUSSION

The supported idea in this paper, of hiding nodes rather than deleting them when projecting artifacts in the decentralized execution of processes specified with LSAWfP, significantly improves the execution model proposed by Zekeng et al. Indeed, in terms of expressiveness, the new execution model makes it possible to overcome the two assumptions made by these authors and consequently, to execute processes specified with LSAWfP regardless of their GMWf form (recursive or not, admitting several axioms associated to various tasks executed by various actors or not). In terms of simplicity, the new model decreases the activity of a peer to four key steps rather than five. In addition, the concept of upstair buds manipulated by Zekeng et al. is obsolete: i.e. after the expansion, there are no more nodes appearing in the resulting artifact but not present in the artifact under execution and its updated partial replica. Therefore, it is no longer necessary to prune the obtained artifact. In terms of efficiency, the main algorithms are faster: the artifact projection is done in linear rather than quadratic time, the GMWf projection is done in quadratic rather than cubic time and, the expansion is done in linear rather than cubic time.

[6]We consider here, addressing the artifacts' nodes with variable length strings inspired by Dewey's notation as presented in [7].

The new execution model also guarantees a "know what you should know" policy as desired by Zekeng et al. However, in their model, it is quite difficult to be able to understand the semantics denoted by a GMWf from any of its projections (this is due to the choice of deleting types). In the new model, the projection of a GMWf denotes the same semantics as it does; if someone has access to it, then it becomes easier for him/her to understand how the process works: the level of confidentiality is therefore decreased.

V CONCLUSION

In this paper, we were interested in the completely decentralized execution of processes specified with the LSAWfP language. We suggested changes to the execution model proposed by Zekeng et al. in order to overcome two assumptions made by them. As a result, the execution model has been significantly improved in terms of expressiveness, simplicity and efficiency. However, the new model slightly decreases the confidentiality level of the previous one. An interesting next step to the work presented in this paper, would be to propose mechanisms to verify the specifications made with LSAWfP, in order to help designers to produce process models that do not execute infinitely (because of recursivity) or, process models that have unreachable tasks.

REFERENCES

[1] A. Nigam and N. S. Caswell. “Business artifacts: An approach to operational specification”. In: *IBM Systems Journal* 42.3 (2003), pages 428–445.

[2] M. Dumas, M. La Rosa, J. Mendling, and H. A. Reijers. *Fundamentals of Business Process Management, Second Edition*. Springer, 2018. ISBN: 978-3-662-56508-7.

[3] E. Badouel, L. Hélouët, G.-E. Kouamou, C. Morvan, and N. R. Fondze Jr. “Active workspaces: distributed collaborative systems based on guarded attribute grammars”. In: *ACM SIGAPP Applied Computing Review* 15.3 (2015), pages 6–34.

[4] M. M. Zekeng Ndadji, M. Tchoupé Tchendji, C. Tayou Djamegni, and D. Parigot. “A Language and Methodology based on Scenarios, Grammars and Views, for Administrative Business Processes Modelling”. In: *ParadigmPlus* 1.3 (2020), pages 1–22.

[5] M. M. Zekeng Ndadji, M. Tchoupé Tchendji, C. Tayou Djamegni, and D. Parigot. “A projection-stable grammatical model for the distributed execution of administrative processes with emphasis on actors’ views”. In: *Journal of King Saud University - Computer and Information Sciences* (2021). ISSN: 1319-1578.

[6] E. Badouel and M. Tchoupé Tchendji. “Merging Hierarchically-Structured Documents in Workflow Systems”. In: *Electronic Notes in Theoretical Computer Science* 203.5 (2008), pages 3–24.

[7] M. Tchoupé Tchendji and M. M. Zekeng Ndadji. “Tree Automata for Extracting Consensus from Partial Replicas of a Structured Document”. In: *Journal of Software Engineering and Applications* 10.05 (2017), pages 432–456.

[8] J. Yan, Y. Yang, and G. K. Raikundalia. “SwinDeW-a P2P-Based Decentralized Workflow Management System”. In: *IEEE Trans. Systems, Man, and Cybernetics, Part A* 36.5 (2006), pages 922–935.

[9] M. M. Zekeng Ndadji. “A Grammatical Approach for Distributed Business Process Management using Structured and Cooperatively Edited Mobile Artifacts”. PhD thesis. University of Dschang, Cameroon, 2021.

[10] M. C. Paull and S. H. Unger. “Structural equivalence of context-free grammars”. In: *Journal of Computer and System Sciences* 2.4 (1968), pages 427–463. ISSN: 0022-0000.