A Usage Control Model Extension for the Verification of Security Policies in Artifact-Centric Business Process Models

Ángel Jesús Varela-Vaca(✉), Diana Borrego, María Teresa Gómez-López, and Rafael M. Gasca

University of Seville, Seville, Spain
{ajvarela,dianabn,maytegomez,gasca}@us.es
http://www.idea.us.es/

Abstract. Artifact-centric initiatives have been used in business processes whose data management is complex, being the simple activity-centric workflow description inadequate. Several artifact-centric initiatives pursue the verification of the structural and data perspectives of the models, but unfortunately uncovering security aspects. Security has become a crucial priority from the business and customer perspectives, and a complete verification procedure should also fulfill it. We propose an extension of artifact-centric process models based on the Usage Control Model which introduces mechanisms to specify security policies. An automatic transformation is provided to enable the verification of enriched artifact-centric models using existing verification correctness algorithms.

Keywords: Artifact-centric business process model · Verification · Security · Declarative security policy · Usage control model

1 Introduction

Nowadays, organizations model their operations with business processes. To ensure the proper operation of the companies, it becomes necessary the verification of those processes to avoid unexpected errors at runtime, which may deal to inconsistent situations that cannot reach a business goal successfully. Therefore, it is more suitable to detect possible anomalies in the model at design-time before the processes are enacted, so preventing errors at runtime.

Traditionally, business processes are modeled as activity-centric business process models [1], where data are used as inputs and outputs of the activities. The activity-centric proposals describe at design-time the imperative workflow that an instance can follow. However, for some types of scenarios, it is very difficult to include the data state transitions into the activity point of view, specially for complex data models. For this reason, the artifact-centric methodology (data-centric approach) has emerged as a new paradigm to support business process management, where business artifacts appeared for the necessity of enriching the business process model with information about data [2], providing a way for

understanding the interplay between data and process. Artifacts are business-relevant objects that are created, evolved, and (typically) archived as they pass through a business, combining both data aspects and process aspects into a holistic unit [3].

Artifact-centric modeling establishes data objects (called artifacts) and their lifecycles as focus of the business process modeling. This type of modeling is inherently declarative: the control flow of the business process is not explicitly modeled, but follows from the lifecycles of the artifacts. The lifecycle represents how the state of an artifact may evolve over the time. The different activities change the state of the artifact and the values of the data associated to each artifact; these may be manual (i.e. carried out by a human participant of the process) or automatic (i.e. by a web service). The evolution of the artifacts implies a change of the state and the values of the data, until a goal state of an artifact is reached. One of the reasons why the artifact-centric paradigm facilitates the process description is the capacity to model the relations between objects with different cardinalities, not only 1-to-1 relations. This modeling capabilities are not entirely supported in activity-centric scenarios. For instance, BPMN 2.0 [4] (currently wide accepted activity-centric notation) allows to easily represent multi-instance activities and pools (processes), but with some limitations, such as the relations between different processes can only be expressed as hierarchies.

On the other hand, artifact-centric models allow 1-to-N and N-to-M relations between artifacts. Then, when more than one artifact is involved in the process, it is possible that a combination of services and data values violate the policies of the business. In order to avoid this situation at runtime, it is necessary to detect some of these possible errors at design time.

To our best knowledge, there are no pure works that consider security issues at artifact-centric business process models. However, the artifact-centric methodology needs for a way to express the security aspects that are not natively considered in the artifacts, such as *Subjects*, *Rights* and specific *Predicates*, and these security aspects need also to be included in the artifact verification since they can change the state of the artifacts.

The goal of this paper is, on the one hand, to provide an artifact-centric business process model specification of security features and constraints. On the other hand, this paper aims to address the automatic transformation of these enriched models into standard artifact-centric model where the design time verification techniques found in the literature can be applied including the security perspective.

The rest of the paper is structured as follows: Section 2 presents a brief review of the notions of artifact and artifact union as a formal model for artifact-centric business process models. Section 3 provides an extension of the existing artifact-centric model to enrich it with the security perspective. Section 4 explains the model transformation so that it is verifiable at design time by means of previous presented verification mechanisms. Section 5 presents an overview of related work found in the literature. Finally, conclusions are drawn and future work is proposed in Sect. 6.

2 Artifact-Centric Business Process Model in a Nutshell

Artifact-centric paradigm facilitates the process creation oriented to the description of the data object evolution during a process execution. The formalization of the artifact-centric business process model to be extended is presented and widely explained in [5]. Nevertheless, the main notions are listed below in order to facilitate the understanding, using an adaptation of the example in [5] to support the concepts. Although the complete example describes the handling of a conference by an organizing committee, in this paper we focus on the registration of participants, review of papers, and paper submission by means of artifacts. These three artifacts are shown in Fig. 1, where solid circles, squares and arrows represent states, services and flows within an artifact respectively, whereas dotted elements are included to represent structural dependencies between artifacts. Likewise, attributes are listed on the right of each artifact.

As reflected in [5], artifacts are represented as specified in the framework BALSA [6] as a basis. That way, the formalization of the model includes:

- **Structural perspective**, identified by the tuple $G = \langle St, Ser, E \rangle$, representing the set of states (St, circles in Fig. 1), the set of services (Ser, squares in Fig. 1) and the set of edges connecting them (E, arrows in Fig. 1), which form the lifecycle of each artifact;
- **Data perspective**, identified by the tuple $Data = \langle id, at, pre, post \rangle$, representing the identifier (id), the set of attributes (at), and the pre and postconditions (pre and $post$) of the services in Ser;

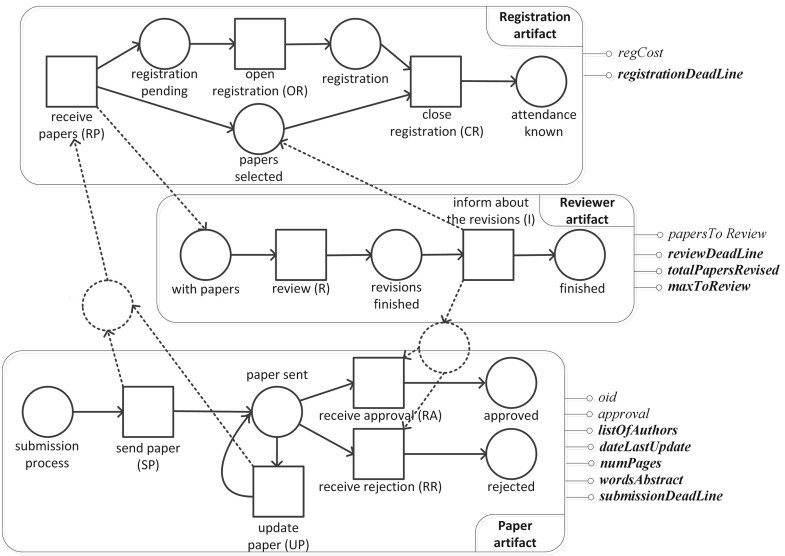

Fig. 1. Example of artifact union

– **Goal states**, identified by Ω, which is a set composed of subsets of St representing the end points in the lifecycle of the artifact.

Likewise, as also explained in [5], the global model is defined as the union of all artifacts composing it. That union is established by two types of policies, which limit the coordinated execution of artifacts lifecycles: (1) **Structural Policies**, expressing constraints on the relation between states and/or services of different artifacts (dotted elements in Fig. 1); and (2) **Data Policies**, expressing invariant conditions over the data (i.e. attributes) managed by the different artifacts in the complete model. For instance, the example in Fig. 1 counts on the data policy *All submitted papers should be reviewed*.

The constraints that should be satisfied during the execution of the services (that is, preconditions, postconditions and policies) are linear or polynomial equations or inequations over artifact instances and the attributes in At. That set of constraints is generated by the grammar presented in [5].

3 Usage Control Model (UCON$_{ABC}$) Extension for Artifacts

As previously commented, artifact-centric models do not provide a way to include security policies, a crucial aspect to ensure the artifact correctness. One way to incorporate them is following the UCON$_{ABC}$ model. UCON$_{ABC}$ model [7] has emerged as a generic formal model to represent complex, adaptable and flexible security policies in new environments such as Internet of Things (IoT). For instance, Digital Right Management (DRM) is an access control mechanism which can be modeled by UCON$_{ABC}$. Moreover, other traditional access control and trust management mechanisms can be defined by using this model. UCON$_{ABC}$ model consists of eight components: Subjects, Objects, both Subject and Object Attributes, Rights, Authorizations, Obligations and Conditions. These components can be divided into various groups:

1. **Components** are defined and represented by their attributes. There are two types of components:
   – *Subjects* is a component which holds or exercise certain rights on objects.
   – *Objects* is an entity which a subject can access or usage with certain rights.
2. **Rights** are privileges that a subject can hold and exercise on an object.
3. **Predicates** to evaluate for usage decision. These predicates can be represented without limitations using the same grammar (i.e. by constraints) proposed in [5]. UCON$_{ABC}$ model defines three types of predicates:
   – *Authorizations (A)* have to be evaluated for usage decisions and return whether the subject (requester) is allowed to perform the requested rights on the object.
   – *Obligations (B)* represent functional predicates that verify mandatory requirements a subject has to perform before or during a usage exercise.
   – *Conditions (C)* evaluate environmental or systems factors to check whether relevant requirements are satisfied or not.

All these predicates can be evaluated before or during the rights are exercised. In that case, UCON$_{ABC}$ model splits each predicate into two types of sub-predicates depending on when it must be evaluated: (1) pre-Authorization (preA) is evaluated before a requested right is exercised; and (2) on-Authorization (onA) is performed while the right is exercised. Likewise, obligations and conditions can be divided into pre- and on-predicates. Regarding attributes, UCON$_{ABC}$ model introduces the concept of mutability which indicates whether certain attributes can be modified or not during the usage decision process. This concept can be modeled using specific predicates as part of the usage decision predicates.

We have used UCON$_{ABC}$ components to extend the original artifact model in order to achieve a secure-extended artifact model which includes a news perspective called Security Perspective. The Security Perspective that extends the artifact models is formalized as follows:

– Security Perspective is identified by the tuple $Sec = \langle R, Sub, Pol \rangle$, where R represents the set of rights, Sub represents the assignments of subjects, and Pol is the set of predicates that define the security policy.

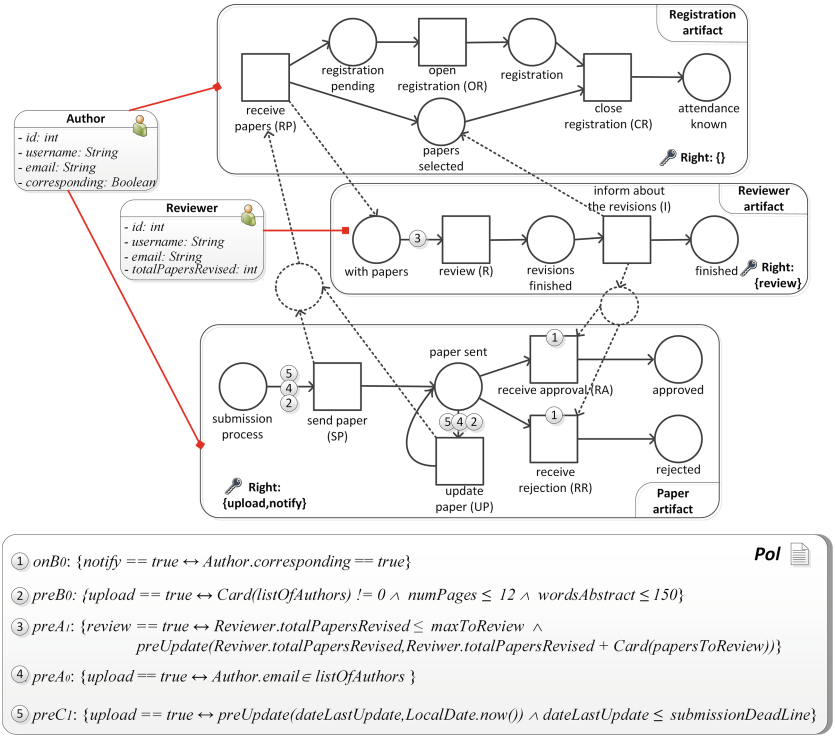

Fig. 2. Example of subjects, rights and security policy.

Objects are the artifacts or artifact unions in which security policies should be defined. Nevertheless, artifact models do not provide elements to define *Subject* concerns. The provided extension enables the specification of *Subjects* in the artifact model. Thus, artifacts can be performed by one or various subjects. When it comes to represent a subject, there are two possibilities: (a) a *Subject* is defined for the complete artifact; and (b) different *Subjects* are defined for each specific services within the artifact. In Fig. 2, the example shows two different *Subjects*: (a) 'Author' are assigned to the Paper and Registration artifact; (b) 'Reviewer' is assigned to reviewer artifact. The *Subjects* are formalized by a set of attributes in the same way than an artifact, as shown in Fig. 2. These attributes have an associate semantic that is used to the evaluation process of the predicates. An example of attribute description for the Author *Subject* is listed in Table 1.

Regarding *Rights*, a set of them has to be defined for each artifact. They represent the different *Rights* that a subject may exercise in the artifact. In Fig. 2, no *Rights* are defined in the registration artifact, the reviewer artifact has only one right called 'review', enabling a reviewer to carry out a review process. Likewise, in the paper artifact there are two *Rights*: '*upload*', enabling the files uploading; and '*notify*', which enables an author to be notified or not.

As it was aforementioned, *Pol* is a security policy represented by a set of *Predicates*. The UCON$_{ABC}$ model introduces multiple kinds of *Predicates* depending on the type of *Predicate* and where the *Predicate* has to be evaluated. Our extension enables the specification of *Predicates* throughout the different parts of the artifact model. There are several places where *Predicates* can be located: (1) transitions between states and services, where all types of *pre-Predicates* can be checked; and (2) in the services, where all type of *on-Predicates* can be checked; and (3) invariants defined for a complete artifact. Thus, these predicates do not require to be checked in a specific place but that have to be checked in every moment. In this case, all types of *Oblitations* and *Conditions* predicates can be defined and checked. For instance, an invariant could be an *Obligation* predicate which indicates *Author* subject must be the same for *Registration* and *Paper* artifact. This type of constraint may define as a invariant which indicates *Author.id* attribute in *Registration* artifact have to be equal to the *Author.id* in *Paper* artifact.

Table 1. Subject attribute description

Author	
id	The author's identification
username	The author's username
email	The author's email
corresponding	The author which submits the paper is the author to be notified

In Fig. 2, there is a unique security policy defined for the three artifacts. This policy is encompassed of five *Predicates* in order to illustrate the main *Predicates* of UCON$_{ABC}$:

1. An *Obligation* predicate (cf. onB_0) enables the notification *Right* whether the author to be notified is established as a corresponding author.
2. A *Conditional* predicate (cf. $preB_0$) enables the upload *Right* whether any author is established for the paper, its number of pages are less than twelve and the abstract is composed of less than one hundred and fifty words.
3. An *Authorization* predicate (cf. $preA_1$) enables the review *Right* whether the reviewer *Subject* has a '*totalPapersRevised*' less than or equal to a 'maxToReview' established as attribute of the artifact. This predicate introduces a *preUpdate* predicate that establishes an update for the attribute '*totalPapersRevised*'. The *preUpdate* predicate establishes an update prior to the usage by means of an increment of '*totalPapersRevised*' with the number of papers from the list of papers within '*papersToReview*' [5].
4. An *Authorization* predicate (cf. $preA_0$) enables the upload *Right* whether the author's email belongs to one of the authors in the list of author of the paper (cf. '*listOfAuthors*').
5. A *Condition* predicate (cf. $preC_1$) enables the upload options whether the local date when the paper is being uploaded or updated, is less than the submission date established as deadline in the conference. This predicate introduces a *preUpdate* predicate that establishes an update for the attribute '*dateLastUpload*'. This predicate attempts to establish a new value for the '*dateLastUpload*' with the local time.

The relation of predicates and the elements of the artifact model are depicted by circled-numbers attached to the transitions and services (notice that the numbers do not indicates the order of the policy). In Fig. 2, one *pre-Authorization* predicate (cf. $preA_1$) is included in reviewer artifact previous to the '*review(R)*' service is carried out. Other eight *Predicates* are included in *Paper* artifact: one *pre-Conditional*, *pre-Authorization* and *pre-Obligation* (cf. $preA_0$, $preB_0$, $preC_1$) are established previous the '*send paper(SP)*' and '*update paper(UP)*' are carried out. Two *on-Conditional* predicate are (cf. onB_0) established during the services of '*receive approval (RA)*' or '*receive rejection(RR)*'.

4 Transformation for the Verification of Security Policies in Artifacts

The inclusion of a security perspective in artifact models aims to provide mechanisms to verify at design time the correctness of security policies, as well as structural and data policies. The verification may consider many aspects, and we propose to follow the verification ideas and algorithms introduced in [5] where two types of correctness are carried out:

– Reachability, which checks whether there is a possible trace of execution where every state can be reached, so there is an evolution of the lifecycle in which

the state is available taken into account the pre and post conditions of the possible service executions.
– Weak-termination, which is a correctness criterion that ensures that a goal state is always reachable from every reachable state.

These existing design time verification algorithms are prepared to verify an artifact model that only contains structural and data policies. Nevertheless, security features are now included, therefore they have to be taken into consideration for the verification. Then, we propose an automatic transformation of the secure-extended artifact model to a simple artifact model as shown in Fig. 3. Thus, we propose to transform automatically the three types of components provided by UCON$_{ABC}$ model (*Subjects*, *Rights* and *Security Policies* (*Predicates*)) into artifact elements such as described in literature and summarized in Sect. 2.

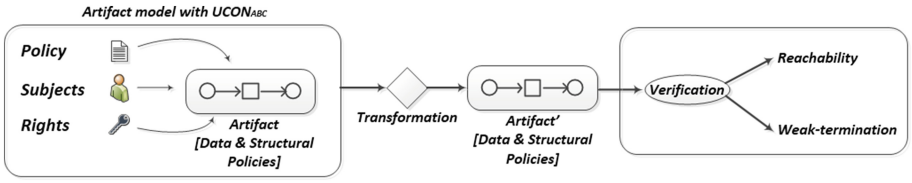

Fig. 3. Verification process of an extended UCON$_{ABC}$ artifact.

The transformations proposed for each component are as follows:

– *Subjects* are transformed into attributes of the related artifact. For instance, Author is linked to *Registration* and *Paper* artifact, hence it is transformed into an attribute inside of these artifacts.
– *Rights* are transformed into attributes of the related artifact. For instance, upload is a *Right* defined in the *Paper* artifact, hence it is transformed into an attribute within this artifact.
– *Security policy (Predicates)* are transformed into constraints to be checked along with the pre and postconditions of the artifact.

Likewise, it is necessary to consider when some *Predicates* have to be evaluated. That is, previous or after pre and postconditions:

1. *Predicates* to be evaluated before preconditions:
   $pre'(n) \rightarrow \langle preA(n) \wedge preB(n) \wedge preC(n) \wedge pre(n) \rangle$
2. *Predicates* that have to be evaluated just after preconditions and previous to postconditions:
   $post'(n) \rightarrow \langle onA(n) \wedge onB(n) \wedge onC(n) \wedge post(n) \rangle$

For instance, the *Reviewer* artifact at the '*review (R)*' service contains the next precondition (cf. Table 2 in [5]):

$pre(n) : Card(papersToReview) \geq 4$

The new precondition after the transformation looks like:

$$pre'(n) : upload == true \rightarrow \langle Reviewer.totalPapersRevised \geq maxToReview \wedge Reviewer.totalPapersRevised == Reviewer.totalPapersRevised' + Card(papersToReview)\rangle \wedge Card(papersToReview) \geq 4$$

The *preUpdate* predicate has been adapted to a constraint that compares the value of *Reviewer.totalPapersRevised* with its previous value updated, *Reviewer.totalPapersRevised'*. These new pre (cf. $pre'(n)$) and postconditions (cf. $post'(n)$) replace the pre (cf. $pre(n)$) and postconditions (cf. $post(n)$) through the artifact. That is, pre and postconditions of services are replaced by enriched constraints that also consider security concerns.

In summary, the formalization of the artifact model changes as follows:

- **Data perspective**, identified by the tuple $Data = \langle id, at, pre, post\rangle$, the set of attributes (at) are extended with the objects *Subjects* and *Rights*, and the pre and postconditions (pre and $post$) of the services in *Ser* are extended as aforementioned.
- **Data Policies**, the inclusion of new attributes in *Data* may require a set of new invariants related to those attributes. As aforementioned, there are some *Predicates* established as invariants. These invariants are transformed into invariants of the Data Policies.

With this guidelines, after the model is transformed, the algorithms presented in [5] can be applied, getting a complete verification covering the three perspectives (data, structural and security) in the artifact-centric process models. The perspectives are formulated into a Constraint Satisfaction Problem (CSP), and the algorithms add constraints to determine both correctness for each state in the process. Both algorithms are complete: neither false positives nor false negatives are generated. Moreover, the algorithms offer precise diagnosis of the detected errors, indicating the execution causing the error where the lifecycle gets stuck.

The execution time is linked to the complexity of the resolution of the CSP, as it was discussed in [8]. In general, no affirmation about the efficiency or scalability of our proposal can be given, mainly because the scalability could be affected by a large increase in the number of constraints and/or variables wrt the number of states. However, this is not usual in real life artifact-centric business processes [9]. Furthermore, owing to the search methods used by CSP solvers, and to the constraints limited by a grammar, the increase in the number of constraints and/or variables could not affect the execution time. As a concrete example, the verification of the reachability of the motivating example takes less than 3 s[1].

5 Related Work

The compliance of security issues in business process models is studied in [10]. Although the authors only focused on activity-centric process models,

[1] The test case is measured using a Windows 7 machine, with an Intel Core I7 processor, 3.4 GHz and 8.0 GB RAM.

they provide a LTL formalism to define security compliance rules for business processes. The authors also indicate the difficulty of using artifact-centric models since there exists no well-defined operational semantics for directly executing the defined models. They underlying the impossibility of artifact perspective to introduce rules that enable to establish conditional activities however our contribution enable to establish conditional execution of tasks based on a security policy.

Security has been considered in other several stages of business process management, [11]. A vast number of works provide several ways to represent and verify security requirements at the modeling stage, such as [12–14]. These works enable to generate security components from the business process models. Currently, monitoring and process mining techniques are new trends in order to detect whether certain security requirements are complied by analyzing event logs [15]. Nevertheless, these works are carried out taking into consideration just the activity-centric perspective skipping the artifact-centric perspective.

Regarding UCON$_{ABC}$, the UCON$_{ABC}$ model provides an advantage with regard to traditional access control models since it covers a wide spectrum of security issues such as access controls, trust management, and DRM in a systematic manner for protecting digital resources. However, the UCON$_{ABC}$ model also presents several limitations such as how the UCON$_{ABC}$ can handle the contextual information of the scenarios or the lacks of the UCON$_{ABC}$ to support the complex usage modes that are required in modern computing scenarios. These limitations have been detailed and discussed in [16].

Related to what artifact model is more appropriate to include security aspects, we realize that most previous works in the literature do not take into account numerical data verification in the artifact-centric model. The paper [17] performs a formal analysis of artifact-centric processes by identifying certain properties and verifying their fulfillment, such as persistence and uniqueness. Although it is [18] who performs a static verification of whether all executions of an artifact system satisfy desirable correctness properties. In that work services are also specified in a declarative manner, including their pre and postconditions.

However, they fail in the presence of even very simple data dependencies or arithmetic, both crucial to include security policies. This problem is addressed and solved in [19], where data dependencies (integrity constraints on the database) and arithmetic operations performed by services are considered. To verify the behavior of an artifact system, contribution [20] transforms the GSM model into a finite-state machine and systematically examines all possible behaviors of the new model against specifications. Likewise, the approach in [21] observes two deficiencies in the GSM approach, and resolves them. They also observe that GSM programs generate infinite models, so that they isolate a large class of amenable systems, which admit finite abstractions and are therefore verifiable through model checking.

The field of compliance for artifact-centric processes has been addressed in [22]. The authors extend the artifact-centric framework by including the modeling of compliance rules, and obtain a model that complies by design. This way, the runtime verification of compliance is not required. The contribution [23]

checks for conformance between process models and data objects at design time. They propose a notion of weak conformance, which is used to verify that the correct execution of a process model corresponds to a correct evolution of states of the data objects. Although is in [5] where the reachability and weak-termination is verified combined structural and data information. This verification approach integrates some requirements necessary for security perspective: pre and post-conditions defining the behavior of the services, numerical data verification when the model is formed by more than one artifact, and handling 1-to-N and N-to-M associations between artifacts.

6 Conclusions and Forthcoming Work

To ensure the correctness of artifact-centric business process models, their security perspective should be considered. Therefore, we propose an extension of a previous artifact-centric business process model related to Usage Control Model (UCON_{ABC}) that introduces mechanisms to specify modern security polices and constraints that help the coverage of the security perspective besides the aspects in the structural and data perspectives.

Since the existing techniques and algorithms do not provide the possibility to manage security policies, we transform automatically the enriched model into a artifact model to be verified avoiding its manual performance which is time-consuming and error-prone.

To the best of our knowledge, this paper presents the first approach for artifact-centric business process models that integrates security aspects, which define the behavior of the artifact dealing with security restrictions.

As future work, we plan to offer additional feedback in case of a violation, making easier the job of fixing the problem causing the error. Furthermore, we plan to deploy diagnosis algorithms in order to explain how and why violations are produced. This work is only focused on the design perspective of artifacts although the same ideas can be adapted to be applied at runtime. That is, we can extract runtime event logs that can be matched with the artifact policies in order to check the compliance of the three perspectives.

Acknowledgement. This work has been partially funded by the Ministry of Science and Technology of Spain (TIN2015-63502) and the European Regional Development Fund (ERDF/FEDER).

References

1. Weske, M.: Business Process Management: Concepts, Languages, Architectures. Springer, New York (2007)
2. Nigam, A., Caswell, N.S.: Business artifacts: an approach to operational specification. IBM Syst. J. **42**(3), 428–445 (2003)
3. Cohn, D., Hull, R.: Business artifacts: a data-centric approach to modeling business operations and processes. IEEE Data Eng. Bull. **32**(3), 3–9 (2009)

4. OMG: Object Management Group, Business Process Model and Notation (BPMN) Version 2.0. OMG Standard (2011)
5. Borrego, D., Gasca, R.M., Gómez-López, M.T.: Automating correctness verification of artifact-centric business process models. Inf. Softw. Technol. **62**, 187–197 (2015)
6. Hull, R.: Artifact-centric business process models: brief survey of research results and challenges. In: Meersman, R., Tari, Z. (eds.) OTM 2008, Part II. LNCS, vol. 5332, pp. 1152–1163. Springer, Heidelberg (2008)
7. Park, J., Sandhu, R.: The UCON ABC usage control model. ACM Trans. Inf. Syst. Secur. **7**(1), 128–174 (2004)
8. Gómez-López, M.T., Gasca, R.M., Pérez-Álvarez, J.M.: Compliance validation and diagnosis of business data constraints in business processes at runtime. Inf. Syst. **48**, 26–43 (2015)
9. Chinosi, M., Trombetta, A.: BPMN: an introduction to the standard. Comput. Stand. Interfaces **34**(1), 124–134 (2012)
10. Reichert, M., Weber, B.: Enabling Flexibility in Process-Aware Information Systems - Challenges, Methods, Technologies. Springer, Heidelberg (2012)
11. Leitner, M., Rinderle-Ma, S.: A systematic review on security in process-aware information systems - constitution challenges, and future directions. Inf. Softw. Technol. **56**(3), 273–293 (2014)
12. Salnitri, M., Brucker, A.D., Giorgini, P.: From secure business process models to secure artifact-centric specifications. In: Gaaloul, K., Schmidt, R., Nurcan, S., Guerreiro, S., Ma, Q. (eds.) BPMDS 2015 and EMMSAD 2015. LNBIP, vol. 214, pp. 246–262. Springer, Heidelberg (2015)
13. Wolter, C., Menzel, M., Schaad, A., Miseldine, P., Meinel, C.: Model-driven business process security requirement specification. J. Syst. Archit. **55**(4), 211–223 (2009)
14. Jürjens, J.: Developing secure systems with UMLsec — from business processes to implementation. In: Fox, D., Köhntopp, M., Pfitzmann, A. (eds.) Verlssliche IT-Systeme 2001. DuD-Fachbeiträge, pp. 151–161. Springer, Verlag (2001)
15. Accorsi, R., Wonnemann, C., Stocker, T.: Towards forensic data flow analysis of business process logs. In: 2011 Sixth International Conference on IT Security Incident Management and IT Forensics, Institute of Electrical & Electronics Engineers (IEEE), May 2011
16. Grompanopoulos, C., Mavridis, I.: Challenging issues of UCON in modern computing environments. In: Proceedings of the Fifth Balkan Conference in Informatics. BCI 2012, pp. 156–161. ACM, New York (2012)
17. Gerede, C.E., Bhattacharya, K., Su, J.: Static analysis of business artifact-centric operational models. In: SOCA, pp. 133–140. IEEE Computer Society (2007)
18. Deutsch, A., Hull, R., Patrizi, F., Vianu, V.: Automatic verification of data-centric business processes. In: ICDT, pp. 252–267 (2009)
19. Damaggio, E., Deutsch, A., Vianu, V.: Artifact systems with data dependencies and arithmetic. ACM Trans. Database Syst. **37**(3), 22 (2012)
20. Gonzalez, P., Griesmayer, A., Lomuscio, A.: Verifying GSM-based business artifacts. In: Goble, C.A., Chen, P.P., Zhang, J. (eds.) ICWS, pp. 25–32. IEEE Computer Society (2012)
21. Belardinelli, F., Lomuscio, A., Patrizi, F.: Verification of GSM-based artifact-centric systems through finite abstraction. In: Liu, C., Ludwig, H., Toumani, F., Yu, Q. (eds.) Service Oriented Computing. LNCS, vol. 7636, pp. 17–31. Springer, Heidelberg (2012)

22. Lohmann, N.: Compliance by design for artifact-centric business processes. In: Rinderle-Ma, S., Toumani, F., Wolf, K. (eds.) BPM 2011. LNCS, vol. 6896, pp. 99–115. Springer, Heidelberg (2011)
23. Meyer, A., Polyvyanyy, A., Weske, M.: Weak conformance of process models with respect to data objects. In: Proceedings of the 4th Central-European Workshop on Services and their Composition, ZEUS-2012, Bamberg, pp. 74–80, 23–24 February 2012