University of Groningen

Provenance Holder

Stage, Ludwig; Karastoyanova, Dimka

Published in:

Business Process Management Workshops - BPM 2019 International Workshops, Revised Selected Papers

DOI:

[10.1007/978-3-030-37453-2_53](https://doi.org/10.1007/978-3-030-37453-2_53)

IMPORTANT NOTE: You are advised to consult the publisher's version (publisher's PDF) if you wish to cite from it. Please check the document version below.

Document Version

Publisher's PDF, also known as Version of record

Publication date:

2019

[Link to publication in University of Groningen/UMCG research database](#)

Citation for published version (APA):

Stage, L., & Karastoyanova, D. (2019). Provenance Holder: Bringing Provenance, Reproducibility and Trust to Flexible Scientific Workflows and Choreographies. In C. Di Francescomarino, R. Dijkman, & U. Zdun (Eds.), *Business Process Management Workshops - BPM 2019 International Workshops, Revised Selected Papers* (pp. 664-675). (Lecture Notes in Business Information Processing; Vol. 362 LNBIP). Springer. https://doi.org/10.1007/978-3-030-37453-2_53

Copyright

Other than for strictly personal use, it is not permitted to download or to forward/distribute the text or part of it without the consent of the author(s) and/or copyright holder(s), unless the work is under an open content license (like Creative Commons).

The publication may also be distributed here under the terms of Article 25fa of the Dutch Copyright Act, indicated by the "Taverne" license. More information can be found on the University of Groningen website: <https://www.rug.nl/library/open-access/self-archiving-pure/taverne-amendment>.

Take-down policy

If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.

Downloaded from the University of Groningen/UMCG research database (Pure): <http://www.rug.nl/research/portal>. For technical reasons the number of authors shown on this cover page is limited to 10 maximum.

Provenance Holder: Bringing Provenance, Reproducibility and Trust to Flexible Scientific Workflows and Choreographies

Ludwig Stage^(✉) and Dimka Karastoyanova

Information Systems Group, University of Groningen, Groningen, The Netherlands
bpm@ludwig-stage.de, d.karastoyanova@rug.nl

Abstract. Process adaptation has been in the focus of the BPM community also in the scope of interdisciplinary research on the interface with eScience. In eScience, scientists, software developers and analysts across a range of disciplines model, create and execute different types of data-driven experiments in an exploratory, trial-and-error style. Supporting this style of experimenting by software systems has been enabled in existing work by interactive workflow management systems. The main principles and techniques used are related to the adaptation of workflows and choreographies, as well as to systems integration and software architecture improvement. With the increasing need for collaboration in scientific explorations and adaptation remaining an essential part of the techniques used, the trust in the reproducibility of scientific research gains a most significant importance. To ensure trust in the provenance and reproducibility of collaborative and adaptable scientific experiments, in this paper we introduce a system architecture with focus on a new component that is responsible for ensuring trust in a generic and non-intrusive manner, independent of different process-aware technologies and storage platforms, including blockchain. We contribute the architecture of the Provenance Holder, its components, functionality and two main operations: recording provenance data and retrieving the trusted provenance information. We incorporate the Provenance Holder into the architecture of the interactive ChorSystem and identify the potential directions for future research.

Keywords: Flexible scientific choreographies · Reproducibility · Trust · Provenance · Collaboration · Blockchain

1 Introduction

Blockchain technology has had and still has a significant impact on Business Process Management (BPM) research. Many of these impact factors have been identified in [8]. In this paper, we focus on investigating one unexplored aspect, namely the interplay with and potential impact on the field of adaptable processes, both workflows and choreographies, in particular for the purposes of enabling trust and improving provenance and reproducibility of in-silico

scientific experiments, like e.g. simulations, scientific workflows and business analytics and data scientific experiments, that need to be adapted very often. Keeping an immutable trail of transactions without the need of a trusted third party and thus enabling of collaborations between “mutually untrusted parties” [7] is surely a promise of the blockchain technology that needs to be evaluated and exploited. In our previous work [6] we have suggested an approach that uses blockchain to allow for trusted collaborations among adaptable scientific experiments. The approach comprised an architecture of an enabling system and a number of alternative realizations depending on the role blockchain platforms might play. The most prominent and promising alternative realizations we suggested are (a) the use of blockchain platforms for storing purely workflow auditing/historical information or (b) the use of blockchain platforms and smart contracts (and potential extensions) to execute adaptive workflows directly on a blockchain platform. While we deem these alternatives appropriate and worth investigating, in this work we follow a more generic approach to fulfill the requirements of scientists and analysts. We mapped these requirements to properties that an enabling system has to provide in [6] and provide the list here as well:

- provenance to enable Findable Accessible Interoperable Reusable (FAIR) results [9],
- reproducibility for Robust Accountable Reproducible Explained (RARE) experiments [4],
- trust among collaborating parties and
- adaptability of the experiments.

Provenance is an important topic in the scientific community and extensively discussed in publications like [2, 5]. There are Scientific Workflow Management Systems (sWfMSs) [2], which already support provenance to a certain extent [5]. The sWfMS Apache Taverna [14], for instance, supports provenance of workflow execution by recording input and output of service calls during execution and provenance of data [1] which is done through annotating the workflow description at design time and generating origin-annotations at run time. However, changes to workflow definitions are not captured and have to be tracked by other means and reproducing changes is not at all in the focus of this or other works.

Since *changes* on workflow and choreography level are essential for collaborative scientific workflows, which represent in-silico experiments, as we have shown in e.g. [6] and [13], we need to establish a way to record information about the execution of choreographies, workflows and their activities, as well as their adaptations so that it is possible to reproduce every step taken and every adaptation made, and at the same time allows storage on a possibly untrusted medium. One prerequisite here is to ensure that all *adaptation* steps are coordinated among all participating partners (as in e.g. [13]). Given that it would be possible to rerun/reproduce choreographies on the basis of the recorded data. Note that runtime information about some activities of involved workflows or even whole parts of workflows might not be explicitly available because they are e.g. calls to service interfaces that may not be providing this information.

Despite that, the need of scientists is to be able to *retrace and reproduce* the executed flexible choreographies since they stand for the explorative way of experimenting. Therefore, in this paper we focus on introducing a system that combines these properties in a generic manner and thus allows for use of different types of technologies and realizations present and at the same time ensures that the collaboratively produced scientific results can be trusted by all participating parties. For this purpose we will mainly focus on introducing the functionality and architecture of a new component in the architecture of an overall sWfMS that brings these properties together and thus enables collaborative, adaptable and reproducible in-silico experiments. We call this new architectural component Provenance Holder (see Fig. 1).

As a basis for the work presented here we use the ChorSystem [12] that supports flexible choreographies of scientific workflows and which is based on a conventional Workflow Management Systems (WfMSs) rather than a system developed by the typical eScience software like the above mentioned Taverna [14] or Kepler¹. Please note that the work presented here can be applied with other WfMSs as long as they support the Model-as-you-go-approach [10] and are able to provide the required provenance data (cf. Sect. 2.3).

The rest of the paper is structured as follows: in Sect. 2 we introduce the architecture of our system, the *Provenance Holder*, discuss its capabilities, properties and requirements. In Sect. 3 we identify open research questions to be addressed in future work to fully implement and evaluate the new component. We conclude the paper in Sect. 4.

2 Provenance Holder

In this section we will introduce the Provenance Holder as an extension to the ChorSystem [12] we have presented in our previously published work. The ChorSystem allows for modelling, execution and adaptation of scientific workflows and choreographies and supports the trial-and-error manner of exploration scientists prefer. The system is our realization of the Model-as-You-Go approach [13]. The ChorSystem architecture is shown in Fig. 1 and consists of (i) a modeling and monitoring environment (ChorDesigner, Transformer and Process Designer) to model collaborative experiments, generate visible interfaces of participating workflows, refine internal workflow logic of each participating workflows and serve as monitoring tool, (ii) a special middleware (ChorMiddleware) which coordinates choreography adaptations and (iii) interactive WfMSs that support the execution, adaptations and monitoring of the workflows and choreographies together with the rest of the system components. All system components are provided as services and interact with each other using a service middleware, the Enterprise Service Bus (ESB).

In the current system, the adaptation steps on the choreography level or on individual workflows are recorded by the ChorSystem and the records are merely stored into a log file, since trust and provenance requirements were not in the

¹ <https://kepler-project.org/>.

primary focus of our research. As mentioned above, in [6] we introduced two possible approaches as alternative solutions towards enabling the required properties of provenance and trusted reproducibility of collaborative adaptive workflows. The first approach suggested the use of trusted decentralised ledger technology, e.g. blockchain, as the audit trail that can be used to prove the provenance of the adaptations in collaborative in-silico experiments. The second approach was aiming at integrating the properties into an adaptive, blockchain-based Business Process Management System (BPMS)/WfMS. As a natural continuation of work on these approaches, in this paper we contribute an architecture of a system that supports the collaborative adaptations in experiments and at the same time allows for the interchangeable use of different technologies to support trust without third parties through an additional component, the Provenance Holder (see Fig. 1).

The Provenance Holder component is responsible for collecting all information necessary to ensure provenance and reproducibility of and trust in the collaborative adaptations. We argue that this solution is generic and reusable across different scenarios and is less intrusive due to the separation of concerns principle we follow. As depicted in the high-level architecture of the system in Fig. 1, the Provenance Holder is not part of any other functional component of the architecture, but rather is a separate service whose responsibility is to collect all relevant information necessary for reproducibility and provenance in such a manner that the trust in the adaptive collaborations is ensured. Externalizing the functionality needed to support the requirements we identified in a separate service opens up the research questions of what the architecture of such a component is, what functions it has and how it should interact with the rest of the components of the ChorSystem.

Fig. 1. Architectural overview of the ChorSystem and the Provenance Holder

The *two main operations* the Provenance Holder needs to provide on its interface are: *collect provenance data* and *retrieve provenance information*. These operations can be realized by four *interaction scenarios* carried out by its components, which we also call *methods* and explain in Sect. 2.1. The *components*

of *Provenance Holder* are the controller, the adapter and one or more provenance providers - they are discussed in Sect. 2.2 and shown in Fig. 2. The impact on the components comprising the overall architecture, i.e. the Workflow Management Systems (WfMS), the ChorMiddleware and Modelling and Monitoring Environment will be identified in Sects. 2.3 and 2.4.

2.1 Provenance Holder Methods

The Provenance Holder performs four main methods that realize in combination the two main operations (see Fig. 2), namely collect provenance data and retrieve provenance information. The methods are:

1. Record
2. Retrieve (provenance information)
3. Validate
4. Migrate (to new type or instance of storage)

During the execution and adaptation of workflows and choreographies the Provenance Holder collects provenance data constantly. During normal execution of the experiments (or their runtime) only three out of the four methods are used regularly. The fourth one, the migration, is used when there is a change or addition of storage instances or storage technology is necessary or desired.

The data related to the experiment execution and changes also need to be authenticated. This can be done, for instance, by the executing WfMS environment by signing input, workflow version, and output (see Sect. 2.3 for more details). Here the input are the parameters handed over to the workflow upon invocation, the output is the returned message or the result of the workflow, and such information is made available on the ESB (Fig. 1) as it is the communication backbone of this service-oriented system.

The *Record* method selects the appropriate provider components (provider *a*, *b*, *c* and so on in Fig. 2) for a certain workflow type out of the available providers and uses them to store the provenance information. For the execution of workflows, this might be input data, the executed workflow version and its output data. The actual provenance information is paired with information about the corresponding choreography instance, such as a reference to the corresponding choreography instance, for later ease of retrieval. For an adaptation on workflow or choreography level, this information might consist of the description of the actual change performed, the new version and a reference to the preceding one. Depending on the technology employed to implement a provider, this data is then stored with the help of the selected provider(s).

The *Retrieve* method is used to fetch the desired provenance information from the provider components via their interfaces. The information is identified and retrieved using the corresponding choreography instance ID and/or workflow IDs. The actual data retrieval is done by each provider itself and returned to the *retrieve* method. After retrieval, the information is again validated before it is handed over to the adapter component, i.e. the Provenance Holder interface

Fig. 2. Provenance Holder components and methods

implementation. The validation is used to rule out storing errors or tampering on the data storage and to guarantee the validity of the data and the freshness of the validity check. The retrieved information should then be presented to the user, in our case by the ChorDesigner, however any other system will have to have its own way of presenting the information to its users.

During *Validation* the provided signature is verified in the controller component. The signed data is validated and the signee is identified. When the record method is called, the signature gets verified before the data is “recorded”. If the signature can not be verified or the signee not be identified, the information is rejected and not considered further, hence is not “recorded”. The failed signature verification is also communicated to the user, i.e. the experiment modeller, for instance in the ChorDesigner. When calling the retrieve method, the provenance information is fetched from the provenance provider and then validated. If two or more providers are present in a Provenance Holder, retrieved data is not only validated but the data of the different providers and each validation status is compared to one another to identify possible discrepancies. The status of the validation is again, communicated to the user. Especially if the data storage is not situated locally (relative to the Provenance Holder) an adversary might be able to change stored data. If this is done by a participant then he is also able to produce a “valid” signature of the changed data. To be able to validate signatures, the participants’ keys need to be exchanged beforehand and stored in the Provenance Holder.

The *Migrate* method is only used, if stored information has to be transferred to a new type or instance of storage, in case an addition or change of instances is desired or needed. It provides the ability to retrieve all stored provenance information from a provider component at once.

Addition of storage instances means that storage is expanded by a new instance of an already existing technology or a new instance of a not yet used storage technology, e.g. data is stored in an SQL database and now will be copied to a second one or will now also be stored in a flat file. Change of storage on the other hand means that one storage instance replaces another one. This can be done within a particular storage technology or done by replacing one technology

with another, e.g. data is stored in a flat file and will now be stored in an SQL database, hence it will be migrated. The employed technology has also implications on the complexity of such a migration because of the difference in features and of their characteristics. Migrations can be triggered both automatically or manually by an administrator; the actual procedure for migration is out of the scope of this paper as related work [11] is available.

It is important to note though that the cost of this method must be considered, especially when it comes to blockchain technology where the needed information can be spread virtually over the whole ledger. The migration step might also involve purging data from the source after a successful data transfer. Here, blockchain technology might also pose additional challenges. There are implications on the cost and complexity of such a migration (see Sects. 3.3 and 3.4 for more details).

2.2 Provenance Holder Components

The Provenance Holder consists of the controller, adapter and one or more provider components (see Fig. 2).

The *adapter* is the component ensuring the integration of the Provenance Holder into an existing system. In our case, as we want to enhance the ChorSystem [12], which is a service-based one, the component will enable the communication with the ESB and has two main operations to provide. The first one is to collect all necessary provenance information while collaborative experiments are being executed and changed, whereas the second one is to retrieve the provenance information from the Provenance Holder and relay it for presentation to the user. Since the employed WfMSs publish all necessary data to the ESB and the ChorDesigner communicates with the ChorMiddleware through the ESB, the Provenance Holder adapter ensures the communication with all components of the overall infrastructure.

Data collection and retrieval, which scientists commonly call publication, however, are done differently. The Provenance Holder subscribes to the relevant messages of the WfMS, ChorDesigner and ChorMiddleware and hence collects the needed information autonomously and continuously every time data is published by these systems. Collected provenance information on the other hand is only published/returned upon request to the users, for instance by the ChorDesigner and a user friendly presentation is required. The adapter component serves the request with only the collected provenance information, though. The adapter also captures change events of both choreography and workflow, so every time a change is made or deployed it is recorded by the Provenance Holder. To enable both provenance data collection and retrieval imposes additional requirements, especially on the WfMS (see Sect. 2.3) and the Modeling and Monitoring Environment (Sect. 2.4).

Providers or *provenance providers* have certain properties to fulfil and methods to implement. To be able to provide provenance support a provider has to use appropriate technology to fulfil the requirements. The implementation complexity can be arbitrarily high and is highly dependent on the employed technology:

writing into a log file, for example, is of low complexity, whereas employing blockchain technology is more on the high end of the complexity spectrum. The needs of different workflow types also come into play when deciding which technology to use. The provider components have to implement three out of the four methods explained in Sect. 2.1, namely *record*, *retrieve* and *migrate*.

The *controller* is in charge of enabling the interaction between the adapter and the provenance providers so that the Provenance Holder can provide the provenance service operations to each workflow and choreography. The controller combines the four methods we presented already into the realization of the two operations the Provenance Holder provides. For the *collect provenance data operation* the controller receives, validates and relays the provenance information to the providers. For the *operation retrieve provenance data*, it combines the methods *retrieve* and *validate*. In both cases the *validate* method is a crucial step as described in the previous section. The controller has a key storage integrated to be able to verify signatures and identify participants.

2.3 Requirements on Workflow Management Systems

The Provenance Holder as introduced above requires that the WfMSs are able to sign, for example, the provenance data they produce the executed workflow version, the input parameters given at invocation and the calculated result/output of the run workflow. The time when a workflow was started/executed might also be provenance data. The signature of changes made to the workflow model is also necessary. To produce a signature, a private/public key pair has to be generated beforehand. The public key also has to be made available to the Provenance Holder. All created signatures have to be published to the ESB, otherwise the Provenance Holder is not able to consider them.

The idea is that if a workflow is made available, hence its interface is published, the public key of the workflow or WfMS owner is also made available. Every time the workflow is run, the private key to this public key is used to sign the produced provenance data. By doing so the WfMS attests that a workflow instance of a certain version of the workflow model was successfully executed with a certain input and produced a certain output as explained above.

Every WfMS or its owner is interested in producing valid and reproducible results, since he can be held accountable because he signs them and everyone can verify the signature with the published public key. The incentive to produce valid and reproducible results here is, if a not reproducible result is published or a workflow/WfMS owner repudiates a result, he loses his credibility because the signature proves otherwise.

The key exchange and participant identification has to be done before participants can engage in collaborative scientific workflows. Since the key is made public alongside the workflow, the exchange is trivial. The participant identification on the other hand is not. Here principles such as trust on first use (TOFU) or trust upon first use (TUFU) and the web of trust are best applied. An identification through a different channel or even a personal identification on e.g. a conference is also possible. Every participant has to decide himself which method suites best.

2.4 Requirements on Modeling and Monitoring Environment

There are two requirements on the Modeling and Monitoring Environment, specifically the ChorDesigner, which need to be addressed to make use of the Provenance Holder in the ChorSystem. First, provenance data and the corresponding signatures need to be displayed to the users. Especially the validity of a signature is important for a user. A signature is valid if the signee is known, i.e. the public key, and the signature can be verified with help of said public key against the provided provenance data. In terms of visualization to the user, there are two possibilities, either only showing the provenance data to the user if the signature is valid or displaying alongside information about the validity. Pros and cons for both possibilities have to be identified and considered in future. Second, changes made to the workflows and choreographies need to be signed before they are enacted or deployed. Otherwise changes can not be attributed to a participant and authenticated properly later on.

3 Discussion of Open Issues

In this section we discuss the open issues we identify and that need to be addressed separately as part of the future research work and realization of the presented generic architecture.

3.1 Provenance Information

To be able to provide reproducibility of and trust in workflows and choreographies, certain information and data has to be collected and recorded during their execution. For instance, on workflow level input parameters, output/result and the used workflow version naturally come to mind. In practice, depending on many factors, more information may be needed in order to ensure the provenance and reproducibility of the experiments and their results. We will investigate what kind of information should actually be collected and recorded, what information may be omitted, and if and to what extent data and information needs to be pre-processed or enhanced to deliver the desired outcome. Factors influencing this work will be the type of experiments to be carried out, their realization (workflow vs. choreography and the technology used), the provenance type desired, the technology used for storing the provenance information, and others.

3.2 Workflow Types

There are different kinds of workflows, with different requirements, e.g., confidentiality or computational power. These might impose additional demands on the ChorSystem and especially on the Provenance Holder and the provenance providers used. In future work, we will investigate the different types of workflows, classify the distinguishing types for the ChorSystem in particular and the Provenance Holder and identify the technology and storage needs. We will

work towards providing a classification that maps the different types of experiments from different domains (business analytics, coupled simulations, data science pipelines, multi-scale and multi-physics simulations, etc.) and different technologies, paradigms and techniques towards their realization.

3.3 Complexity Implications

The available storage technologies are quite diverse, therefore switching between them or exchanging them, as the *migrate method* requires, is certainly a non-trivial task to be addressed. The complexity and multi-dimensionality of this kind of task is discussed for migrating enterprise applications to the cloud and a solution in form of a methodology is presented in [11]. The authors also emphasize on databases as part of such applications, and we consider this approach applicable here as well.

Especially when distributed ledger technology, such as blockchain, is involved, a change or switch from or to it not only brings a different feature-set but also a different set of characteristics. This difference might be explicitly desired and be the reason of the change in the first place. The actual migration might be costly though, for example because the implementation of a new provider is complex or retrieving data from one provider or initially storing a lot of data on the other might be expensive. Therefore, we will investigate in future what implications on complexity and costs might be involved, how they can be addressed and if and when they actually need to be addressed.

3.4 Blockchain Technology as Provenance Provider

Blockchain technology has characteristics that limit its applicability in our case and need to be addressed. The first characteristic is the eventual consistency property of blockchain and the second is that the access to entries is not efficient. These characteristics and issues arising from them are also briefly discussed in [3].

The eventual consistency could be addressed by, for example, adding a cache to a provider which implements the storage capabilities with blockchain. The cache holds the to-be-written data until it is written to the ledger or is part of the main chain and therefore distributed. This means on the other hand, if data is not written, or is not included in the chain, it will be resubmitted for storage until it is written.

To be able to retrieve data efficiently a lookup table could be created in the provider component. This table enables accessing data objects in a more direct manner without searching the whole ledger for individual entries. In future we will investigate what other approaches are possible and evaluate their feasibility.

Another property of blockchain technology is its immutability. This makes it impossible to remove once published data. We also will investigate if and when this is a problem and how to address it in the scope of collaborative and adaptive choreographies of experiments.

4 Conclusions

The focus of this paper is on addressing the requirements of scientists in the field of eScience. The need for adaptation of scientific workflows and choreographies due to the dynamic trial-and-error process of experiment modeling that (natural, data and business analytics) scientists follow and the need for reproducible research to establish trust in research results calls for innovative software systems. In our previous work towards supporting collaborative scientific experiments with exploratory nature, we have created an approach called “Model-as-You-Go for Choreographies” and a corresponding interactive software system, based on the workflow technology and runtime adaptation that allows for interleaving the modeling and execution life cycle phases of workflows and choreographies of scientific workflows. We also identified the additional requirements on these choreographies of scientific workflows such as provenance, reproducibility, trust and adaptability.

In this work, we presented a generic architecture of a software system which enables trusted and coordinated execution and adaption of scientific workflows, choreographies and in-silico experiments with particular focus on how provenance, reproducibility and trust in the collaborative adaptation can be enabled. The architecture is drafted in a generic way allowing implementations which fit the needs of scientists, and their experiments likewise, best. The resulting software system features a separate component, the Provenance Holder that is designed so that it is responsible for and capable of storing provenance data in a trusted manner and able to support the reproducibility of experiments based on the provenance data.

There are more open questions for future research. Some still stand from [6]. These are the user friendliness and technology transparency of the new component, performance characteristics and the access control mechanisms that will satisfy the scientists needs of minimum information disclosure and still allow for reproducibility. Performance characteristics need to be both analyzed in general and evaluated in particular for different implementations. We need to investigate which data and information actually needs to be recorded to be able to provide the desired properties (provenance, reproducibility, trust and adaptability). We also need to explore which technology suits which workflow types and requirements best. Migration of one storage technology to another poses complexity implications and in particular what concerns the blockchain technology and its characteristics that need to be paid special attention when it is employed in a provenance provider component.

References

1. Alper, P., et al.: Enhancing and abstracting scientific workflow provenance for data publishing. In: Proceedings of the Joint EDBT/ICDT 2013 Workshops, pp. 313–318. EDBT 2013. ACM, New York (2013). <https://doi.org/10.1145/2457317.2457370>

2. Atkinson, M., et al.: Scientific workflows: past, present and future. *Future Gener. Comput. Syst.* **75**, 216–227 (2017). <https://doi.org/10.1016/j.future.2017.05.041>. <https://hal.archives-ouvertes.fr/hal-01544818>
3. Dumas, M., et al.: Blockchain technology for collaborative information systems (dagstuhl seminar 18332). *Dagstuhl Rep.* **8**(8), 67–129 (2019). <https://doi.org/10.4230/DagRep.8.8.67>. <http://drops.dagstuhl.de/opus/volltexte/2019/10236>
4. Goble, C.: Results vary: the pragmatics of reproducibility and research object frameworks, keynote. In: *iConference* (2015). <https://www.slideshare.net/carolegoble/i-conference2015-goblefinalupload>
5. Herschel, M., et al.: A survey on provenance - what for? what form? what from? *Int. J. Very Large Data Bases (VLDB Journal)* **26**(6), 881–906 (2017)
6. Karastoyanova, D., Stage, L.: Towards collaborative and reproducible scientific experiments on blockchain. In: Matulevičius, R., Dijkman, R. (eds.) *CAiSE 2018. LNBP*, vol. 316, pp. 144–149. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-92898-2_12
7. López-Pintado, O., et al.: Caterpillar: a blockchain-based business process management system. In: Clarisó, et al. (eds.) *Proceedings of the BPM Demo Track at (BPM 2017)*, Barcelona, Spain, September 13, 2017. *CEUR Workshop Proceedings*, vol. 1920. CEUR-WS.org(2017). http://ceur-ws.org/Vol-1920/BPM_2017_paper_199.pdf
8. Mendling, J., et al.: Blockchains for business process management - challenges and opportunities. *ACM Trans. Manag. Inf. Syst.* **9**(1), 4:1–4:16 (2018). <https://doi.org/10.1145/3183367>
9. Mesirov, J.P.: Accessible reproducible research. *Science* **327**(5964), 415–416 (2010). <https://doi.org/10.1126/science.1179653>. <http://science.sciencemag.org/content/327/5964/415>
10. Sonntag, M., Karastoyanova, D.: Model-as-you-go: an approach for an advanced infrastructure for scientific workflows. *J. Grid Comput.* **11**(3), 553–583 (2013). <https://doi.org/10.1007/s10723-013-9268-1>
11. Strauch, S., et al.: Migrating enterprise applications to the cloud: methodology and evaluation. *Int. J. Big Data Intell.* **1**(3), 127–140 (2014)
12. Weiß, A., et al.: ChorSystem: a message-based system for the life cycle management of choreographies. In: *CoopIS 2016*, pp. 503–521. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-48472-3_30
13. Weiß, A., et al.: Model-as-you-go for choreographies: rewinding and repeating scientific choreographies. *IEEE Trans. Serv. Comput.* **PP**(99), 1 (2017). <https://doi.org/10.1109/TSC.2017.2732988>. ISSN 1939-1374
14. Wolstencroft, K.: The Taverna workflow suite: designing and executing workflows of web services on the desktop, web or in the cloud. *Nucleic Acids Res.* **41**(W1), W557–W561 (2013). <https://doi.org/10.1093/nar/gkt328>