Standardisation of Provenance Systems in Service Oriented Architectures White Paper

Authors: Luc Moreau and John Ibbotson
Type: Deliverable
Version: 1.0
Version: March 29, 2006
Status: public

Abstract

This White Paper presents *provenance* in computer systems as a mechanism by which business and e-science can undertake compliance validation and analysis of their past processes. We discuss an open approach that can bring benefits to application owners, IT providers, auditors and reviewers. In order to capitalise on such benefits, we make specific recommendations to move forward a standardisation activity in this domain.

The PROVENANCE project receives research funding from the European Commission's Sixth Framework Programme

Members of the PROVENANCE consortium:

IBM United Kingdom Limited	United Kingdom
University of Southampton	United Kingdom
University of Wales, Cardiff	United Kingdom
Deutsches Zentrum fur Luft- und Raumfahrt s.V.	Germany
Universitat Politecnica de Catalunya	Spain
Magyar Tudomanyos Akademia Szamitastechnikai es Automatizalasi Kutato Intezet	Hungary

The PROVENANCE project receives research funding from the European Commission's Sixth Framework Programme

Contents

The PROVENANCE project receives research funding from the European Commission's Sixth Framework Programme

1 Introduction

Provenance is already well understood in the study of fine art where it refers to the trusted, documented history of some art object. Given that documented history, the object attains an authority that allows scholars to understand and appreciate its importance and context relative to other works. Art objects that do not have a trusted, proven history may be treated with some scepticism by those that study and view them.

This same concept of provenance may also be applied to data and information generated within computer systems. Generally, in computer systems, applications produce data. Our vision is to transform applications into so called *provenance-aware applications*, so that the provenance of the data they produce may be retrieved, analysed and reasoned over as is needed by today's complex requirements of both e-Science and business.

In e-Science, end-users have to reproduce their results by replaying previous computations; they have to understand why two seemingly identical runs with the same inputs produce different results; they have to find out which data sets, algorithms or services were involved in the derivation of their results.

In business and e-Science, users, third-party reviewers, auditors or even regulators have to verify that the process that led to some result is compliant with specific regulations or methodologies; they have to prove that results are derived independently of services or databases with given license restrictions; they have to establish that data was captured at source by instruments that possess some precise technical characteristics.

Unfortunately, a data result typically does not contain the necessary historical information that would help the end-user, reviewer or regulator make the necessary verifications. Hence, extra information describing what actually occurred at execution time needs to be captured and stored; we refer to such extra information as *process documentation*. Using process documentation, the provenance of data results can be retrieved and analysed to suit the user's needs.

However, data sets in multiple application sectors, such as in high energy physics, pharmaceutical industry or aerospace engineering, are long lived, e.g., 20 to 100 years. Thus, the kind of validation and compliance verification that has to take place may not be known at the time applications are run to produce such data sets. In such circumstances, it is important that as much process documentation as possible is captured (because we may not necessarily be able to regenerate it in the future); furthermore, it is essential that such documentation is organised according to a well-specified and agreed upon structure, so that it can be analysed meaningfully in the future.

According to the service-oriented architectural style, data and services can be discovered and opportunistically reused, producing new results, which may themselves be published, shared, discovered and reused in a similar manner. This being so, any such discovered and reused result may have an impact on the final outcome of a computation, but such a result may have been produced by an application or service that uses a different programming paradigm, an alternate architectural style, a distinct technology or even a proprietary solution. For the necessary verification of execution to be possible, it is essential that the process documentation remains independent of technology, architectural style or proprietary nature.

The PROVENANCE project receives research funding from the European Commission's Sixth Framework Programme

The previous discussion indicates that there is a need for inter-operable solutions, by which documentation of actual execution can be made available in the long term, so that the provenance of data can be retrieved and tools can be built in order to analyse how results were produced. The purpose of this document is to promote a standardisation activity to support such a vision.

In order to do so, we identify some important concepts underpinning such a vision (Section 2). We then discuss a number of benefits of a standard solution (Section 3). Afterwards, we examine which elements could be the focus of standardisation (Section 4). Finally, we propose some recommendations to move such an activity forward (Section 5).

2 Some Provenance Concepts

Without anticipating the outcome of a standardisation activity in this context, it is necessary to introduce some terminology pertaining to provenance, and elements of a possible architectural solution [GJM+06]. Such terminology is useful to characterise what aspect a standardisation activity should focus on. In Figure 1, we present different roles and their relationships in a provenance-aware application.

Central to the architecture is the notion of a *provenance store*, an entity (whether independent service, component, or port) designed to facilitate the storage and maintenance of process documentation beyond the lifetime of an application run. In a given application, one or more provenance stores may be used in order to act as storage for process documentation: multiple provenance stores may be required for scalability reasons or for dealing with the physical deployment of a given application, possibly within different security domains.

In order to accumulate process documentation, a provenance store provides a *recording interface* that allows applications to submit descriptions about their execution. A provenance store is not just a sink for such process documentation: it must also support some query facility that allows, in its simplest form, browsing of its contents and, in its more complex form, retrieving the provenance of some specific data among all the documentation contained in the store. To this end, we introduce *query interfaces* that offer multiple levels of query capability. Finally, since provenance stores need to be configured and managed, we also require an appropriate *management interface*.

Client-side libraries facilitate the tasks of recording process documentation in a secure, scalable and coherent manner and of querying and managing provenance stores. They are also designed to ease integration with legacy applications, helping the task of converting applications into provenance-aware applications.

Once process documentation has been recorded in a provenance store, provenance of data can be retrieved and used by *processing services* and *presentation user interfaces*. The former provide added-value to the query interfaces by further searching, analysing, reasoning, or verifying compliance over the provenance of data, whereas the latter essentially visualise query results and processing services' outputs. Another kind of user interface to the provenance store is also identified in the architecture. This is the *management user interface*, which allows users to manage the contents of the provenance store.

The PROVENANCE project receives research funding from the European Commission's Sixth Framework Programme

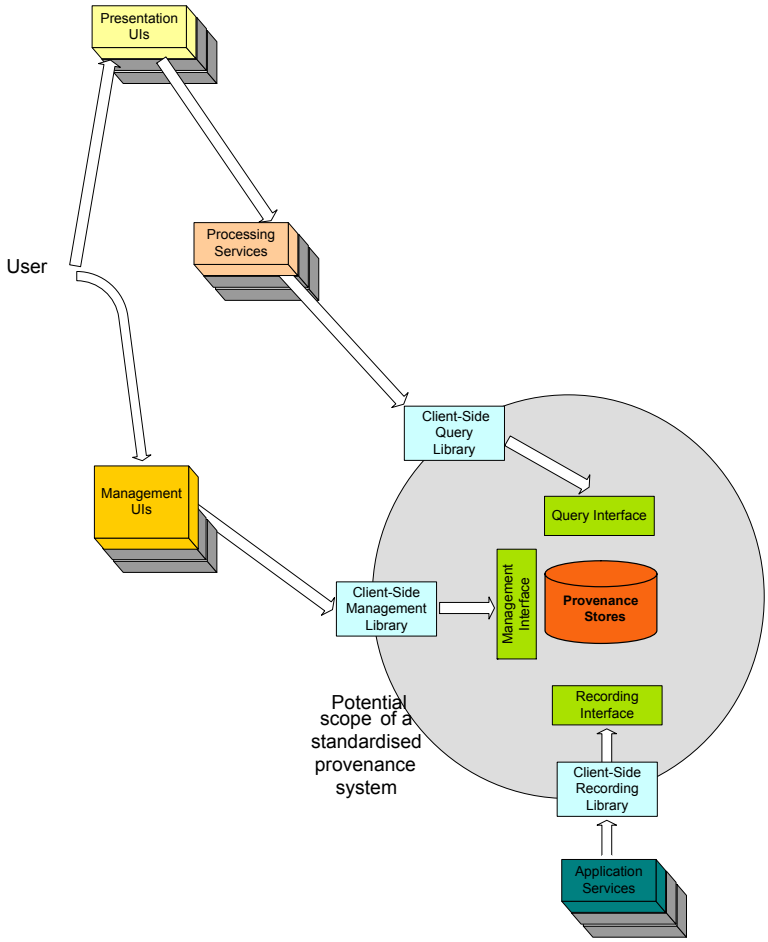

Figure 1: Key Components of a Provenance Architecture

In Figure 1, a circle identifies the potential scope of a standardisation activity in a first instance. It includes the provenance stores, their interfaces, and the libaries to interact with them. In a second instance, it would also be very valuable to standardise some specific processing services (e.g., compliance verification). However, such focus is beyond the motivation of the current document.

Having introduced these concepts, we now see how they apply to two concrete examples involving existing systems.

Example 1 While they do not all adopt the same terminology or architectural style, a number of existing systems follow the approach we describe in this document. Consider two systems: myGrid and the Virtual Data System. In myGrid [GGS+03], the workflow enactment engine (Taverna/FreeFluo) produces descriptions of the service invocations it performs, and stores them in an RDF triple store, which can be queried and mined. Similarly, in the Virtual Data System [FVWZ02], documentation of execution is stored in the metadata catalog, and

can be queried through the SQL query language. While myGrid and VDS workflows can be integrated in order to build a single application, their documentation of execution does not follow a common format, the method by which it is recorded is specific to each environment, and a single query cannot be run over documentation produced by both applications.

Example 2 A number of document formats (cf. Microsoft Word, Adobe PDF) are now embedding historical and versioning information, which can be seen as a form of documentation of the process documents undergo. Such documents (and the information they contain) can be used as input to a complex application already made-provenance aware. While querying the provenance of results produced by the application, we may be able to trace back to the input documents, but not back to their respective origins because their process documentation is structured differently.

3 Benefits of Standardisation

In this section, we discuss the benefits that standardisation would bring to application owners, software designers, IT solutions providers and regulators. Given that the functionality of a provenance store is exposed through its interfaces, we begin our discussion with the benefit of interface standardisation.

Standardisation of the recording interface results in one and only one agreed way by which applications have to structure and record their documentation of execution. Such an approach can be followed by all application components (cf. Figure 2, left), whether or not they are deployed at and managed by different institutions, whether or not they adopt Web Services technology, or whether their components are statically hardwired or dynamically discovered and composed. A standard way of recording process documentation also extends to the case where a complete subset of a process is outsourced to a contractor: all parties just need to follow the same recording conventions.

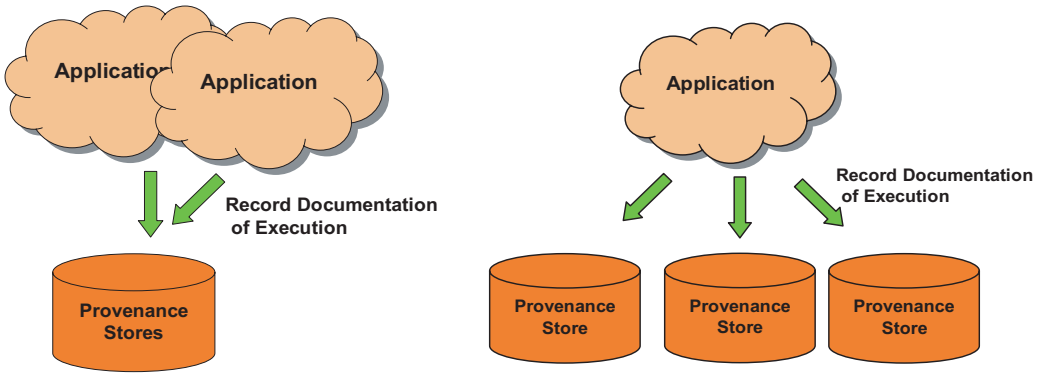

Figure 2: Standard Recording Interfaces

The PROVENANCE project receives research funding from the European Commission's Sixth Framework Programme

While it is beneficial for applications to have a single approach to follow, it is also very useful for IT providers, and provenance store hosts. They only need to offer provenance stores that implement the specified recording interface, and these will be guaranteed to interoperate with provenance-aware applications (cf. Figure 2, right). The differentiator for products or live services thus becomes their non-functional characteristics: implementations may offer differing "quality of service" that can be marketed (such as robustness, responsiveness, long-term archiving, curation capability, etc.)

Standardising the query interface by which provenance of data can be retrieved is also beneficial, as illustrated by Figure 3 (left). Whether or not process documentation is distributed in multiple stores, whether the application uses a single technology or whether it is the result of assembling different technologies, standardisation offers an agreed way of retrieving the provenance of data.

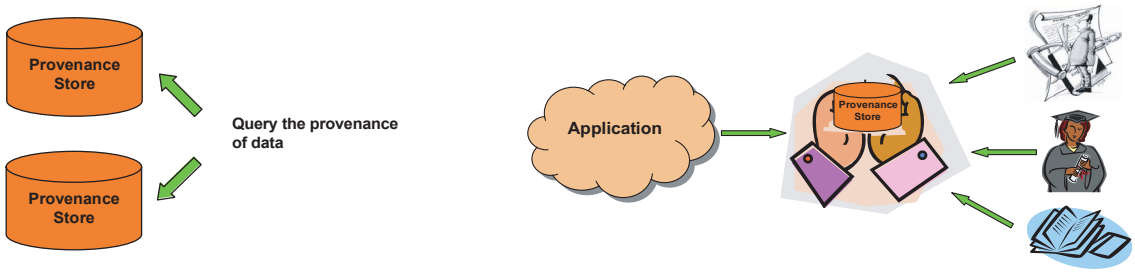

Figure 3: Agreed Querying Interfaces

Adopting standardised management interfaces allows hosts of provenance stores to seamingly manage their stores and the rest of their infrastructure.

Standardisation is also helpful for IT providers/developers, since they can focus on core robust mass-market middleware infrastructures, with the desired reliability, while leaving the development of higher-level niche functionality to specialised consultancy units. For these, new value-added services can be designed (e.g. compliance[1] oriented services), making use of the provenance query interface, and hence focusing on the business logic of the new functionality, rather then the issues of capturing process documentation and querying provenance.

There is also a net benefit for businesses who are not locked in a single vendor solution. Open interfaces allows new functionality, possibly developed by third-party developers, to be plugged into their existing systems. Finally, an open approach is also particularly beneficial in the context of mergers, which typically result in the integration of multiple IT systems and compliance verification having to take place across such merged systems.

[1]The proposed open, standardised infrastructure presents a solid foundation for building higher-level functionality, which in a second phase could itself become standardised. In particular, systems verifying compliance in the business sector (e.g., Sarbanes-Oxley and Basel II) can be built on the proposed open platform and standardised in the future.

The PROVENANCE project receives research funding from the European Commission's Sixth Framework Programme

Figure 3 (right) also illustrates the combined effect of standard recording and querying interfaces. In some context, application owners and regulators (or reviewers) may agree on a third party they trust to host provenance stores. Application owners entrust such parties with details of their processes and data, and therefore want some guarantee that their private information not be leaked to unauthorised users. Symmetrically, auditors trust that a reputable provenance store does not corrupt the data it holds, and therefore, even in the long term, the documentation represents an accurate description of the execution as initially provided by the application owner. Such a model of a trusted provenance store host is only viable if the relevant interfaces have been standardised.

4 Standardisation Options

So far, we have identified key architectural elements of provenance systems, and have argued for standardisation in this domain. We now discuss which aspects of provenance systems should be the primary focus of a standardisation activity. Figure 4 summarises some options, which we now present.

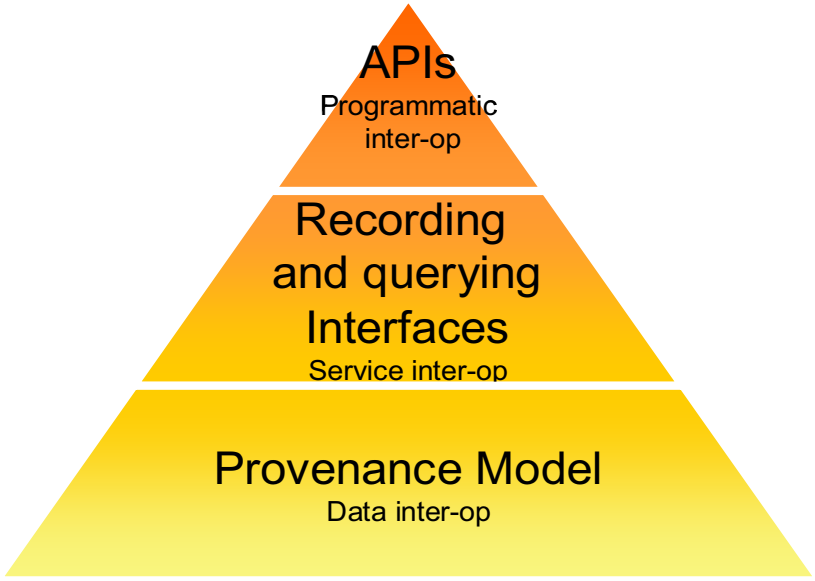

Figure 4: Standardisation Options

For process documentation to be meaningfully used in queries aimed at retrieving the provenance of some data, it needs to provide an application-independent top-level structure; such a structure must be *shared* between applications that describe details of their execution in a technology-independent manner and queriers who derive how results were arrived at. Hence, standardising a *data model* is crucial to offer some *data inter-operability* between applications that need to structure the description of their execution and provenance queriers

who need to express queries, both in terms of this shared data model. Given that, at execution time, the queries to be performed may not be known to the application owner, sharing a data model can only be effective if this is performed in an open manner, through a standardised approach.

A standard data model has an interest of its own, independently of the architectural solution sketched in Figure 1. Existing document formats (cf. Example 2) or metadata schema (e.g. Dublin Core) could be extended with such a data model so as to encompass process documentation in an inter-operable manner.

While a data model is a good step forward, it does not provide the necessary flexibility that underpins the service-oriented architectural style. If potential provenance stores are to be discovered dynamically, there must be agreed ways to interact with them. Hence, standardised *interfaces and protocols* for recording process documentation and querying the provenance of data allow *services to inter-operate*.

Finally, applications will need to be modified in order to become provenance aware; it is desirable that such modifications have minimum impact on code maintenance and performance. Over time, we anticipate that libraries will be developed, offering necessary functionality to upgrade an application to provenance-aware status; the *application programming interfaces* offered by such libraries would also benefit from being standardised, hereby offering *code inter-operability*.

The three previous options for standardisation are not orthogonal but can be viewed hierarchically as illustrated in Figure 4. Interfaces rely on data models, and APIs (and their implementations) are dependent on data models and interfaces. Finally, these options were presented in an application-independent context. But specific sectors (e.g., food industry, automotive industry, or bioinformatics research) may also benefit from refining these application-independent models with domain specific features that are of interest in their context. Thus, domain-specific standard profiles could refine and customise domain-independent standards.

5 Recommendations

There is wide range of fora in which standards for Web Services are being developed, and many of those could potentially be good venue for a standardisation of provenance systems.

Among these, we note that the Global Grid Forum's Open Grid Services Architecture [FKS+05] already identifies provenance as a functionality of data services. Quoting Section 3.5 on Data Services from [FKS+05]:

> OGSA data services are concerned with the movement, access and update of data resources. . . . Data services . . . provide the capabilities necessary to manage the metadata that describes OGSA data services or other data, in particular the *provenance* of the data itself.
>
> Metadata for data services may also include information about the *provenance* and quality of the data. This may be at the level of the whole resource or of its component parts, sometimes to the level of individual elements. This in turn requires the services or other

The PROVENANCE project receives research funding from the European Commission's Sixth Framework Programme

processes that generate the data to also maintain the consistency of the metadata. Complete *provenance* information can allow the data to be reconstructed by following the workflow that originally created it.

This being so, it is appropriate to specify and standardise provenance systems in the context of the Open Grid Services Architecture because it already introduces provenance in its vision. Thus, we are making the following recommendations:

1. Define a scenario making use of provenance for the OGSA Data Scenarios document.
2. Identify users and system designers interested in such a scenario.
3. Define the focus of a standardisation activity, which we proposed to be centered on a provenance data model and store interfaces.
4. Propose a BOF at the Global Grid Forum in the view of creating a working group on provenance.

6 About the Authors

Luc Moreau is a professor of Computer Science, in the School of Electronics and Computer Science at the University of Southampton. His research is concerned with large-scale open distributed systems not subject to centralised control. He has published over 100 articles in the domain of Grid computing, distributed systems, agent-based systems, and distributed information management. His investigation covers the spectrum of software engineering: design, specification, proof of correctness, implementation, performance evaluation and application. He is lead architect of the EU Provenance project and Principal Investigator of he UK Provenance-Aware Service Oriented Architecture projec. He serves on the programme committee of many conferences and workshops in the area of distributed systems. In particular, he is co-chair of the International Provenance and Annotation Workshop (IPAW'06), vice-chair of the topic Grid and Cluster Computing: Models, Middleware and Architectures at Europar'06 and an editor of the journal Computation and Concurrency: Practice and Experience. He can be contacted at `L.Moreau@ecs.soton.ac.uk`.

John Ibbotson is a member of IBM's Emerging Technology Services group based at the Hursley Development Laboratory near Winchester in the UK. He is coordinating the EU PROVENANCE project in addition to providing technical input as part of IBM's collaboration. From 1999 to 2004 he was IBM's representative on the W3C XML Protocol Working Group that was standardising SOAP - a key component of the Web Services Architecture. He is also a joint author of the WS-ReliableMessaging specification. Previously he was an active member of the joint OASIS/UN-EFACT ebXML initiative where he was a member of the Transport and Packaging working group. In addition he was a joint author of the tpaML specification contributed by IBM to ebXML which formed the basis of the CPP/CPA specifications. Earlier in his career, John has developed scientific image processing systems, digital libraries and has interests in multimedia databases and knowledge management. He

is a Chartered Engineer and Fellow of the Institution of Electrical Engineers. He can be contacted at john_ibbotson@uk.ibm.com

References

[FKS+05] I. Foster, H. Kishimoto, A. Savva, D. Berry, A. Djaoui, A. Grimshaw, B. Horn, F. Maciel, F. Siebenlist, R. Subramaniam, J. Treadwell, and J. Von Reich. The open grid services architecture, version 1.0. Technical report, Global Grid Forum, January 2005.

[FVWZ02] I. Foster, J. Voeckler, M. Wilde, and Y. Zhao. Chimera: A virtual data system for representing, querying and automating data derivation. In *Proceedings of the 14th Conference on Scientific and Statistical Database Management*, Edinburgh, Scotland, July 2002.

[GGS+03] Mark Greenwood, Carole Goble, Robert Stevens, Jun Zhao, Matthew Addis, Darren Marvin, Luc Moreau, and Tom Oinn. Provenance of e-science experiments - experience from bioinformatics. In *Proceedings of the UK OST e-Science second All Hands Meeting 2003 (AHM'03)*, pages 223–226, Nottingham, UK, September 2003.

[GJM+06] Paul Groth, Sheng Jiang, Simon Miles, Steve Munroe, Victor Tan, Sofia Tsasakou, and Luc Moreau. D3.1.1: An architecture for provenance systems. Technical report, University of Southampton, February 2006.

[PAS04] Pasoa: Provenance aware service oriented architecture. www.pasoa.org, 2004.

[PRO05] Enabling and supporting provenance in grids for complex problems. www.gridprovenance.org, 2005.