# Provenance Implementation in a Scientific Simulation Environment

Guy K. Kloss and Andreas Schreiber

Simulation and Software Technology
German Aerospace Center
51147 Cologne, Germany
{Guy.Kloss, Andreas.Schreiber}@dlr.de
http://www.dlr.de/sc/

**Abstract.** Many of today's engineering applications for simulations are lacking machanisms to trace the generation of results and the underlying processes. Especially computations conducted in distribued computing environments as Grids are lacking suitable means to keep track of used resources. Trust of engineers in results produced within distribued simulation environments is very limited without this information.
This paper will demonstrate how trust and confidence in simulation results could be achived for engineering applications. It will highlight the backgrounds of the application, of provenance recording, the mapping to the application, and finally the implementation of provenance awareness for the application. Additionally it will present examples of analyzing the information stored to be of further use to the engineer.

## 1 Introduction

Complex numerical simulation plays an important role in today's industrial development. During the design processes it reduces the amount of expensive validation using real world models for physical examination. This again reduces the time for obtaining high quality designs, and thus gives the engineer the opportunity to evaluate more competing designs.

In this engineering process vast amounts of data are generated. Managing this data in the first place is a problem by itself, but it is subject to different solutions. The specific task to be discussed in this paper is to give the engineer tools at hand to keep track of the history of it's generation, which is important to trust the generated data and to analyze data with respect to changes in the simulation process.

### 1.1 Provenance

If an organization wants to prove compliance, they must establish the origin and authenticity of the information produced by their processes. This type of documentation needs to be recorded as the process takes place. To have proven integrity, it must be transparent and auditable. A documentation with

the mentioned properties reveals the full history of the creation some information: it's *provenance.* The goal is to conceive a *computer based representation* of provenance, that allows us to perform useful analysis and reasoning. For inter-operability with other applications and adaptability to new regulations, the provenance implementation needs to be open and standardized, rather than closed and monolithic.

The implementation of a provenance architecture demands a high flexibility and robustness to different scenarios. In the *Grid Provenance* project [1] two sample applications with complementary requirements regarding performance and scalability are to be implemented. The organ transplant management (OTM) application relies on a wide geographical distribution of the system, extremely high standards for security and privacy, multiple provenance stores, a high degree of human interaction, and long periods of time until a management process can be considered completed. On the other hand the aerospace engineering application presents high requirements in computation performance within secured networks. A complete set of requirements [2] for the reference implementation has been identified.

### 1.2 Application

Our goal is to add Provenance awareness to an engineering application which is composed of several individual numerical codes which are integrated in an interactive simulation environment.

The application is used by engineers in the simulation of complex flight maneuvers. This simulation implies the unsteady aerodynamics of a free flying, fully configured, elastic combat aircraft. In this application, two reasons for provenance recording are essential:

- Seamless process documentation for compliance and liability reasons.
- Providing better insight into archived simulational data through analysis by means of complex query methods.

In the following we will first give a very brief overview of the application. After that, an introduction to computer based provenance concepts and the provenance software infrastructure being used is given. Then in Sect. 4.1 we will describe the deployment of provenance services to our application.

## 2 Engineering Application

Aerospace engineering design problems are often being treated by complex simulations using a workflow of a variety of numerical codes and supporting tools. Usually, these simulation workflow does also contain loops, conditional constructs and parallelization of tasks.

For practically perform such complex simulations, all codes are being integrated into an integration system which provides a graphical user interface, starts all codes in the distribution environment and manages all involved data.

### 2.1 Simulation of a Maneuvering Combat Aircraft

Our example application for simulation of a maneuvering combat aircraft consist of three major numerical codes (see [3] for details):

- The computational fluid dynamics solver TAU [4] for aero-dynamics,
- the structure mechanics solver NASTRAN for aero-elasticity, and
- the flight mechanics solver SIMULA [5].

Each of these codes have different and very specific constraints and requirements on the underlaying hardware and software infrastructure, which requires to start each code on a different machine (see Fig. 1).

### 2.2 Integration System

An integration system provide a comfortable execution environment for complex simulations. Via a GUI on a desktop computer, the engineer graphically constructs the workflow, sets input parameters for individual components, and controls the workflow execution.

We are using the integration system TENT [6, 7] which allows one to integrate all necessary computational components and to construct workflows with these components. It allows engineers to steer and monitor running simulations interactively on the Grid and has an integrated data management systems based on XML and WebDAV.

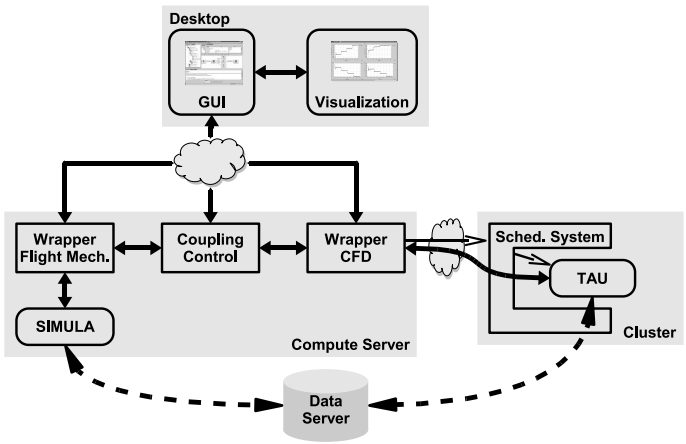


**Fig. 1.** Distribution of workflow components on the network.

## 3 Implementing Provenance

Sect. 1.1 gives an idea for the requirements for the provenance implementation in engineering tasks. Data history needs to be documented *as events take place*. This needs to be dealt with while implementing the compound application. Analysis of the provenance information is supported by a set of query tools. Not essential, but finally important as well, are management tools to access information within the provenance store and setup/configure the store.

Processes are executed in a distributed environment. Result files, log files, do not provide enough information to reflect the exact history of processes. Interactions between different actors need to be tracked. The provenance system needs to be capable of handling several origins for provenance record submissions, and it should support multiple locations of provenance stores as well.

An effective organizational method to ensure consistency is to avoid redundancy. Records on provenance thus should not contain all the information the application provides in copy. Instead, references to the original data will be embedded into the records. This is achieved by using a *documentation style* with the content referenced. Access to the data itself is provided through a plug-in architecture. By this many types of storage locations may be used by developing appropriate plug-ins.

The integration system is turned into a "provenance enabled" system. In many cases the overhead of provenance recording is neither needed nor desired for its use, thus the fundamental architecture is not changed. Because of this, all data and meta-data for the actual engineering task is stored on the WebDAV server. Additional information for provenance tracking is managed using the provenance system. Through the mentioned plug-ins all information can be aggregated through specific provenance tools in development for provenance analysis. For the engineering task all information can still be retrieved by means of the integration system.

Unfortunately this forked storage of content may cause inconsistencies due to changes in referenced data. The most effective way to solve this problem would be to store all information in one system only. This approach would be counter productive, as the (remote) provenance store is not designed to cope with high volume data commonly found in the engineering tasks in question. Thus consistency measures have to be ensured on the organizational level.

### 3.1 Provenance Assertions

To capture the complete process history, three types of p-assertions (assertions on provenance) have been identified [8]: interaction, actor state, and relationship p-assertions.

Interaction p-assertions form the bonding links between components in a process. They capture messages received from or sent by the components or the actors.

As the name implies, the actor state p-assertions provide information on the internal state in the context of a specific interaction. So they may be considered

as snapshots of distinguished states of the actor just before or just after it has received a message.

Finally the relationship p-assertions allow relationships between messages and data. They consist of a subject, one/many objects, and a designated relationship between them. Some examples of possible relationships are as follows: interaction A *is before/after* interaction B; data item C *was zipped to produce* data item D; data item D *is a combination of* data item C and data item B; interaction A *is in reply to* interaction B; interaction A *causes* interaction B. Since, we do not limit what relationships can be expressed, this allows asserting actors to express application specific relationships.

### 3.2 Provenance Store Access

Access to all interactions with the provenance store are abstracted through standardized WSDL interfaces (see Fig. 2). Enabling provenance awareness in existing applications should be as easy as possible, so recording of p-assertions is handled through a client side library. More complex and interactive functionality of querying the store and managing data inside is provided by external tools. The library supports recording interactions and communication of the components, the user interface, and the data server.

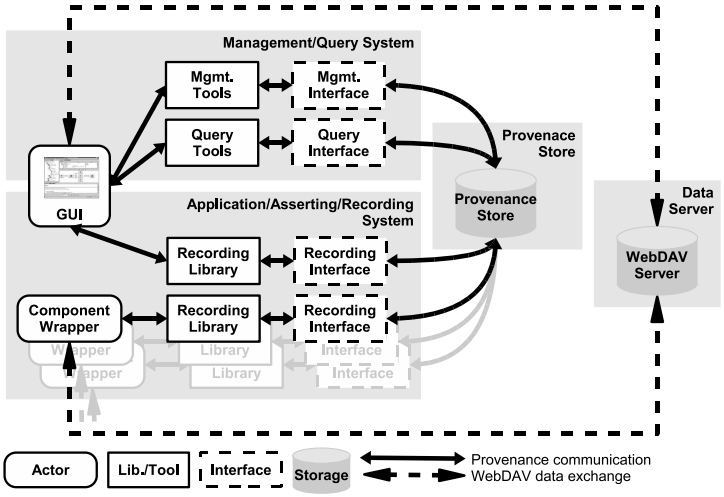


**Fig. 2.** Access of provenance store and data server by software components.

## 4 Applying Provenance

To implement provenance awareness into the application, and to make use of the recorded provenance information, first the application has to be analyzed me-

thodically. Given that information, the client side library has been implemented in TENT, and p-assertion recording in the appropriate places have been added.

The methodical analysis will be described in the following for a simpler application which implies the same workflow complexity and all required features for provenance recording as the aerospace application.

In the terminology of the application the different functional parts are called workflow *components*. From the point of view of the provenance system, the are called *actors*. Both terms are used as synonyms from different perspectives.

### 4.1 Implementation for Parameter Study

We are dealing with a workflow (see Fig. 3) that is set up and controlled through the GUI acting as the central process control instance. The data to initialize the simulation will go through some pre-processing to yield a numerical model for a flow simulation (*i1*). A variation component iterates a single parameter. The parameter is inserted into a configuration file that is passed to the simulation (*i2*). During the individual simulation the code writes information on numerical convergence to the standard output stream. This is intercepted and parsed for information to be monitored in the GUI ($m$). At the end of a single simulation step three things happen: The result set is transferred for storage to the data server (*d1*); the result file is sent to the visualization component for direct feedback on the results (*i4*); and finally the parameter variation component is notified to signal the end of the computations (*i3*). When all simulations have been computed, the process control component (GUI) stores unsaved results, the configuration, and the monitoring data on the data server (*d2*). The completed processing of the workflow spanning from *i0* to *d2* defines a complete *simulation*.

As described in Sect. 3, a process's provenance needs to be tracked as events occur. So the different actors need to be provenance aware, and record provenance relevant information making p-assertions. For some problems this may be achieved by simply intercepting the communication between the components. Due to the internal complexity of the different actors, those p-assertions would not document the process properly without further knowledge of the components' internal actions.

**Interaction P-Assertions.** All interactions between the actors form the linked documentation. Whenever an interaction occurs, an *interaction p-assertion* is submitted. This joins independent actions to a linked process representation. As seen from Fig. 3 the workflow's relevant interactions are *i0 – i4, m1, d1,* and *d2.*

As the content for p-assertions for example, *i2* would state the parameter set for the simulation, and $m$ would contain the monitoring information.

Additionally *tracers* are introduced here. A tracer is an identifier to tag certain groups of p-assertions within the provenance store (for a subset of a workflow conducted). One tracer for example would identify a complete workflow. It would be introduced with *i0*. Another tracer would be introduced (and changed) with every *i2*, to identify all p-assertions for that specific inner loop.

**Actor State P-Assertions.** The interactions between actors define the linked application. The state of an actor before or after such interactions may be of importance for a complete provenance representation. To account for this, *actor state p-assertions* have been identified to document an instance snap shot of the actor. All ("boxed") actors from the figure record their state at various points of time. For the GUI, this p-assertion may record the workflow's name and configuration. For the simulation, it could be the computation's completion state (converged, crashed, interrupted, etc.).

**Relationship P-Assertions.** Various causalities within the workflow can be found. They will be recorded through *relationship p-assertions*. Some of these relationships would be as follows: A single interaction *i1* causes *i2*; an interaction *i2* causes all of *r2, r3, r5,* and *r6*; and in a wide spanning relation the initial *i0* causes the final *d2*.

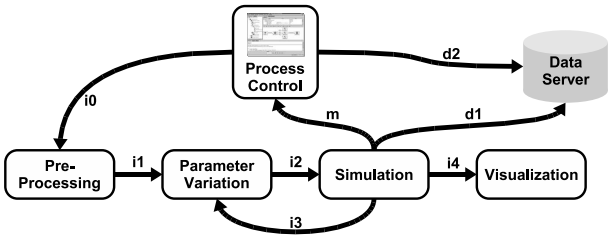


**Fig. 3.** Mapping of a simple optimization application to provenance architecture.

### 4.2 Aerospace Engineering Application

The implementation for the aerospace application consists of the functional components as described in Sect. 2, with additional actors as the GUI, a coupling manager, a data server, and possibly visualization as shown in Fig. 3. The mapping is performed in the same way as for the parameter study. So it will not be discussed here in further detail.

The aerospace engineering scenario is used for research, and terminally the construction of aircrafts. Specific regulations for reproducibility need to be considered. Thus documentation of conducted processes for legal reasons is important. Additionally the engineer can take advantage of the provenance information, as it will give a better overview and means for analyzing the previously conducted simulations. This may lead to a better insight into similarities and differences previously unnoticed.

This can be obtained by asking questions on the data within the provenance store using the query tool.

– Given some data item, what was the simulation case?

- Given some parameter, in what simulation(s) has it been used?
- What data has been recorded in a simulation with a specific parameter?
- What simulations have been run using a given model (aircraft design)?
- Given two/more simulations with the same setup, what is the result and the difference in provenance?

## 5 Alternate Approaches

Various approaches for tracking provenance information are analyzed relative to the application discussed.

Embedded within the operating system a provenance enabled file system PASS [9] is developed. For the benefit of not provenance enabling actors within a system, the "observed" information may be very low level and fine granular. The amount of detail can obscure the view onto desired information. Additionally this approach does not capture distributed processes very well, it is not aware of actor states, and thus may not be able to capture the complete process provenance. More interactively CAVE and CODESH [10] facilitate the user with a UNIX like shell. Operations are conducted through it within a session to narrow down the information more usably.

The problem solving environment WinFX has been provenance-enabled by Microsoft [11]. It currently does not allow workflows in inhomogenious platform environments. It provides a closed provenance system, it caches all provenance data during the process, and persists it at process finalization. More open approaches provide the extensible and flexible frameworks Kepler [12] and Karma [13]. Both can be customized flexibly for scientific domains in inhomogenious and distributed environments like TENT. Kepler provides its own provenance tracking system, as does Karma. Additionally Karma has been shown to be able to connect to more generic provenance systems as the ones developed within the PASOA [14] and Grid Provenance projects [1].

## 6 Conclusion

The need for provenance recording arises in many processes. Reasons for undertaking the necessary steps – that may be considered tedious – can be very different: trust in results, regulations, liability, or just to get better tools for process analysis.

Even though parts of the provenance system are still in development, provenance recording from a smaller scale sample application has been demonstrated already. As the libraries for recording will become usable more easily (due to the ongoing implementation and tools support), it will become easier to implement provenance awareness into the aerospace engineering target application, as well as other possible applications. Furthermore, with availability of the supporting tools and user interfaces for query functionality, inspection possibilities will be provided. These are the necessary tools to start harvesting information available in the provenance store for enhanced insight into engineering processes.

## Acknowledgment

This work was a part of the Grid Provenance project, founded by EU IST 511085.

## References

1. The Grid Provenance Website. [Online]. Available: http://www.gridprovenance.org/
2. Publications of Grid Provenance Project. [Online]. Available: http://twiki.gridprovenance.org/bin/view/Provenance/ProjectPublications
3. A. Schütte, G. Einarsson, B. Schöning, T. Alrutz, W. Mönnich, J. Neumann, and J. Heinecke, "Numerical simulation of maneuvering combat aircraft," in *Proc. 14th DGLR Symposium of STAB*, STAB, Ed. Bremen, Germany: Springer-Verlag, nov 2004.
4. T. Gerhold, "The DLR TAU Code – an Overview," in *ODAS 99, ONERA–DLR Aerospace Symposium*, 1999.
5. W. Mönnich and J. J. Buchholz, "SIMULA – Ein Programmpaket für die Simulation dynamischer Systeme," DFVLR, Tech. Rep., 1991.
6. A. Schreiber, "The Integrated Simulation Environment TENT," *Concurrency and Computation: Practice and Experience*, 2002.
7. A. Schreiber, T. Metsch, and H.-P. Kersken, "A Problem Solving Environment for Multidisciplinary Coupled Simulations in Computational Grids," *Future Generation Computer Systems*, 2005.
8. P. Groth, S. Miles, V. Tan, and L. Moreau, "Architecture for Provenance Systems," ECS, University of Southampton, Tech. Rep., oct 2005. [Online]. Available: http://eprints.ecs.soton.ac.uk/11310/
9. U. Braun, S. Garfinkel, D. A. Holland, K.-K. Muniswamy-Reddy, and M. I. Seltzer, "Issues in Automatic Provenance Collection," in *Proceedings of the International Provenance and Annotation Workshop (IPAW)*, Chicago, Illinois, USA, May 2006.
10. D. Bourilkov, V. Khandelwal, A. Kulkarni, and S. Totala, "Virtual Logbooks and Collaboration in Science and Software Development," in *Proceedings of the International Provenance and Annotation Workshop (IPAW)*, Chicago, Illinois, USA, May 2006.
11. R. Barga, "Automatic Generation of Workflow Execution Provenance," in *Presentation at the International Provenance and Annotation Workshop (IPAW)*, Chicago, Illinois, USA, May 2006.
12. I. Altintas, O. Barney, and E. Jaeger-Frank, "Provenance Collection Support in the Kepler Scientific Workflow System," in *Proceedings of the International Provenance and Annotation Workshop (IPAW)*, Chicago, Illinois, USA, May 2006.
13. Y. L. Simmhan, B. Plale, D. Gannon, and S. Marru, "Performance Evaluation of the Karma Provenance Framework for Scientific Workflows," in *Proceedings of the International Provenance and Annotation Workshop (IPAW)*, Chicago, Illinois, USA, May 2006.
14. The PASOA Website. [Online]. Available: http://www.pasoa.org/