Mayflower—Explorative Modeling of Scientific Workflows with BPEL

Mirko Sonntag, Michael Hahn and Dimka Karastoyanova

Institute of Architecture of Application Systems, University of Stuttgart,
Universitaetsstrasse 38, 70569 Stuttgart, Germany
{sonntag, hahn, karastoyanova}@iaas.uni-stuttgart.de

Abstract. Using workflows for scientific calculations, experiments and simulations has been a success story in many cases. Unfortunately, most of the existing scientific workflow systems implement proprietary, non-standardized workflow languages, not taking advantage of the achievements of the conventional business workflow technology. It is only natural to combine these two research branches in order to harness the strengths of both. In this demonstration, we present Mayflower, a workflow environment that enables scientists to model workflows on the fly using extended business workflow technology. It supports the typical trial-and-error approach scientists follow when developing their experiments, computations or simulations and provides scientists with all crucial characteristics of the workflow technology. Additionally, beneficial to the business stakeholders, Mayflower brings additional simplification in workflow development and debugging.

Keywords: Scientific workflows, Model-as-you-go, SOA, BPEL.

1 Scientific Workflows

The introduction of workflows to scientific computations and simulations has proven beneficial for scientists in many domains, e.g. image processing in physical astronomy [1], earthquake simulations in geology [2], or simulation regarding the biodiversity of species [3]. Workflows speedup scientific computations through automation and straightforward parallelization of tasks, reduce the programming effort for scientists, and improve traceability of scientific results. There is a broad spectrum of scientific workflow systems available, such as Kepler[1], Triana[2], Taverna[3] and Pegasus[4]. Most of these systems have been developed from scratch, implement proprietary, non-standardized workflow languages, and serve specific scientific application domains.

There are also approaches to enhance tools and concepts of the business workflow technology in order to facilitate modeling and execution of scientific computations

[1] https://kepler-project.org/

[2] http://www.trianacode.org/

[3] http://www.taverna.org.uk/

[4] http://pegasus.isi.edu/

and experiments [4, 5, 6]. We are convinced that the well-established conventional workflow technology brings many advantages compared to existing solutions for scientific workflows, namely: (1) the technology is generic and thus independent of the scientific domain and can be applied to almost every scenario, (2) the concept of workflow models and instances can be used to conduct scientific parameter sweeps and parallelize computations, (3) the human tasks features are helpful to integrate human decision points and steering, (4) existing concepts for adaptation of workflows increase the flexibility of scientific workflows, and (5) conventional workflows are standard-based, which facilitates collaboration between scientists and reuse.

The life cycle of business workflows differs from that of their scientific counterpart [4]. That is one of the reasons why the current business workflow technology needs profound extension to be applicable in the scientific domain. Scientists develop software and workflows much more explorative and experimental in a trial-and-error manner [4, 6, 7]. They know the goal of their research but often not the exact way towards this goal or the exact (intermediary) results. That means the scientists approach their goals by trying out different parameter values, by adding or removing activities, by repeating steps of an experiment, or by using different solvers for equations. From the viewpoint of the conventional workflow technology the workflow modeling and runtime phases are not strictly separated, but they are experienced by scientists as a single phase because they can alternate arbitrarily. Furthermore, the workflows are not deployed by scientists. Instead, they are simply started and the workflow deployment is hidden behind a *run* operation. The scientists are the driver of all life cycle phases. They need a single, integrated, easy-to-use tool to deal with workflows in their respective phases without switching between the tools.

We call this flexible development of scientific workflows *Model-as-you-go* since workflows are modeled on the fly during execution [4]. The approach simplifies workflow modeling and increases the robustness of workflows because the user can fix structural failures, repair the workflow context or handle runtime faults without restarting the workflow. Model-as-you-go also integrates approaches for workflow flexibility such as ad hoc adaptations, instance migration, versioning and ad hoc backward loops. The challenges are (1) the start and configuration of workflows via a simple *run* operation instead of a full-blown deployment mechanism; (2) the correlation between processes in the engine and the modeling tool; (3) the combination of a process modeling tool and an instance monitor as entities originally designed for different life cycle phases; (4) the management of process models and instances in the modeling/monitoring tool; (5) the impact of semantically dependent activities on the adaptation of processes; and (6) the stepwise execution of workflows using a non-intrusive event model [8].

In this demo, we present an implementation of the Model-as-you-go concepts: *Mayflower*, the Model-as-you-go Workflow Developer. Mayflower uses BPEL as workflow language and is built upon existing BPEL implementations, namely the Eclipse BPEL Designer[5] as modeling tool and the Apache ODE[6] as workflow engine.

[5] http://eclipse.org/bpel/

[6] http://ode.apache.org/

The software makes development and debugging of BPEL workflows easier and hence can also be used in business scenarios.

2 Architecture Walkthrough

Mayflower consists of five major parts (Fig. 1): (1) The scientist interacts with the Eclipse-based *Modeling Framework*. It consists of the Eclipse BPEL Designer as workflow editor. We have extended the BPEL Designer with functions to (a) control workflow execution (run/resume, suspend and terminate), (b) monitor workflow instances using state information of the workflow engine, (c) adapt the logic and functions dimensions of running workflows as well as the workflow context (e.g. the content of variables, activity markings in order to enforce backward loops), (d) manage different versions of workflow models, (e) specify breakpoints, and (f) track changes made by the user to fill the provenance record. Further, there are plug-ins to access and administrate the Resource Manager and the Workflow Engine.

(2) The *Workflow Engine*, an extended Apache ODE, (a) instantiates workflow models, navigates through the workflow graphs and (b) invokes scientific computations exposed as Web services. It also contains components to (c) handle process models and instances (e.g. resume, suspend, terminate), (d) deploy and undeploy process models, (e) adapt the logic of running workflows, (f) publish execution events, and (g) access and modify the context of workflow instances.

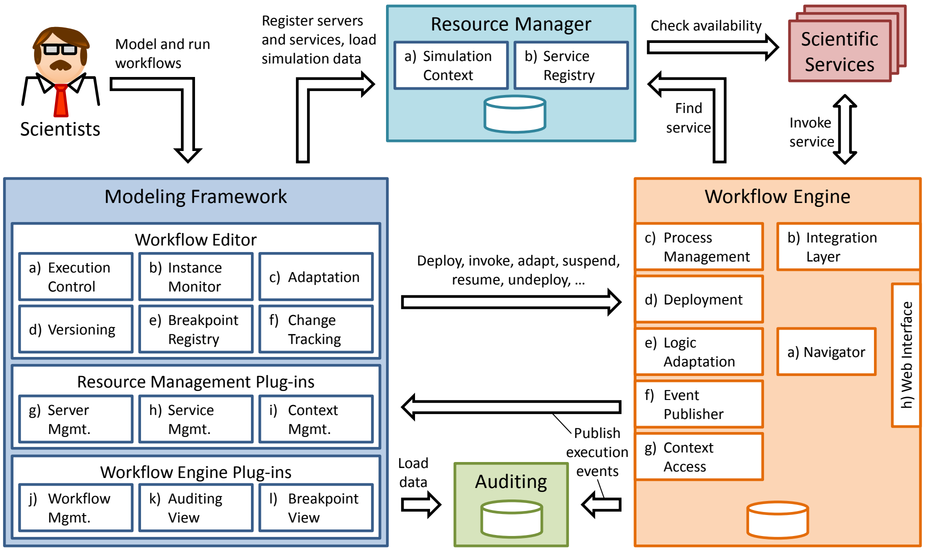


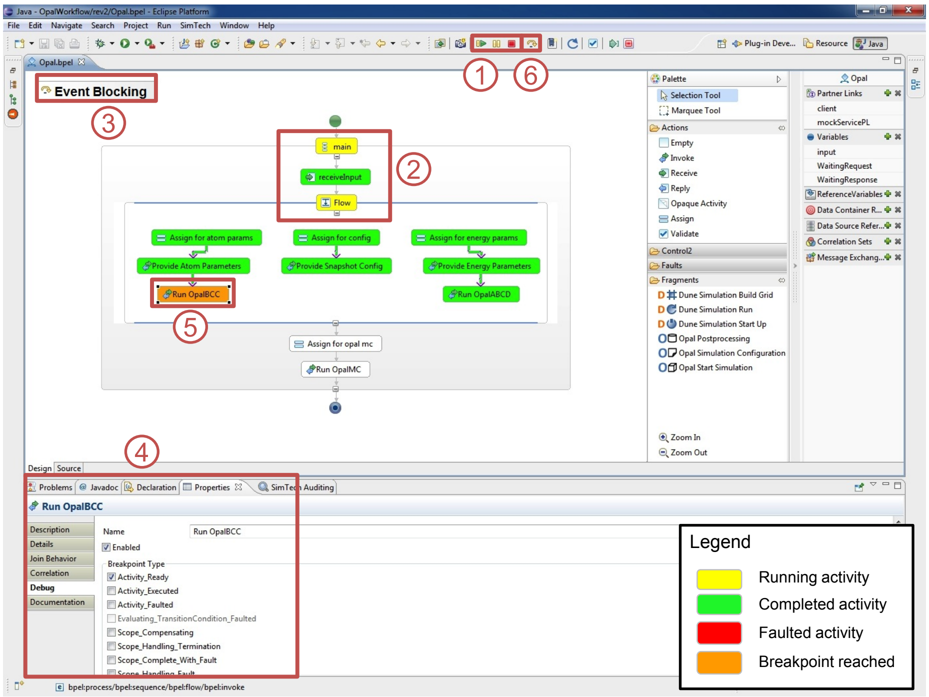
Fig. 1. Overview of the Architecture.

(3) The *Auditing* component stores execution events for workflow instances published by the engine. This information allows loading the state of all workflow instances into the instance monitor of the workflow editor, even of those that were not

started by the editor. It also correlates the workflow model in the engine with that in the modeling framework, since they have different representations and identifiers. (4) The *Resource Manager* (a) offers a logically centralized storage for simulation data and (b) works as registry for the scientific services that participate in the simulations and calculations. (5) The *Scientific Services* provide the domain-specific logic that is orchestrated by the scientific workflows running on the workflow engine.

Fig. 2 shows the user interface of Mayflower and some of its components:

1. Workflows can be started, suspended, resumed and terminated via a toolbar extension.
2. The instance monitor is an extension of the editor pane. Workflow models are enriched with instance information and the activities are colored according to their execution state (see the legend in Fig. 2).
3. The process instance state is displayed in the upper left corner of the editor pane.
4. Breakpoints can be specified in the properties of activities via the new *debug* tab. One or more execution events of an activity can be registered as breakpoint.
5. A reached breakpoint is signaled via a highlighted activity.
6. The user can skip the breakpoint with the help of a toolbar function.



Run OpalBCC

Name Run OpalBCC

☑ Enabled

Breakpoint Type

☑ Activity_Ready
☐ Activity_Executed
☐ Activity_Faulted
☐ Evaluating_TransitionCondition_Faulted
☐ Scope_Compensating
☐ Scope_Handling_Termination
☐ Scope_Complete_With_Fault
☐ Scope_Handling_Fault

bpel:process/bpel:sequence/bpel:flow/bpel:invoke

Legend

🟨	Running activity
🟩	Completed activity
🟥	Faulted activity
🟧	Breakpoint reached

Fig. 2. Screenshot of Mayflower's Modeling Framework with its main components.

3 Demonstrated Features

We demonstrate Mayflower with the help of a use case for the simulation of the ageing process in copper-alloyed steel, an example of a solid-body simulation. The simulation computes how the atomic structure of steel changes when being operated over many years. The atoms exchange their positions and build precipitations or clusters that negatively influence the material properties (Fig. 3). We have re-engineered the simulation tool with the help of BPEL and Web services in order to speed up the simulation runtime through automation of formerly manual tasks and parallelizing post-processing steps [9]. The workflow will be the basis for the demonstration.

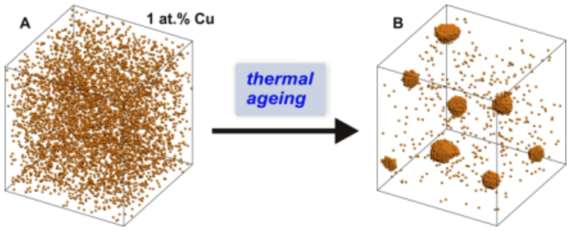


Fig. 3. Simulation of the ageing process in copper-alloyed steel [10]

The following aspects of Mayflower are shown:

1. *Workflow modeling, execution and monitoring.* The user can drag activities from the palette and drop them on the editor pane in order to specify the logic of the workflow. When a valid workflow model (fragment) is created, the user starts the workflow and puts in parameter values he is requested for by the tool. Monitoring of the running workflow starts automatically; the state is displayed by changing the activities' color based on workflow execution events.
2. *Usage of breakpoints.* Breakpoints are specified in the properties of activities. Workflow execution is paused when a breakpoint is reached. It is possible in parallel branches to pause one path while the other path continues.
3. *Adaptation and Versioning.* Modification of running workflows is experienced by scientists as workflow modeling: activities can be added, removed, or changed, while the process instance is already being executed. The changes are propagated to the engine with versioning and instance migration techniques. Furthermore, the user can inspect and change variable values and conduct ad hoc backward loops.
4. *Ad hoc rerun of activities.* The user can jump backwards to an arbitrary activity in the past of the workflow instance. He can select to compensate already completed work in the iteration body. As input for the next run of the activities it is possible to take either the current variable values or variable values that were valid at a former time step during workflow execution [11].
5. *Resource Management.* Via the resource manager plug-in the user provides the servers and scientific services that should be taken for a simulation. He can monitor the workload in the system by having a look at the number of service requests and the number of granted service usage tickets. Furthermore, the user can inspect intermediary results in the simulation context [9].

4 Maturity of the Software and Screencast

The software is a prototypical implementation of the Model-as-you-go concept developed in the past three years in the scope of our work in the research cluster SimTech[7]. A demonstration video of the tool is available online[8].

Acknowledgements. The authors would like to thank the German Research Foundation (DFG) for financial support of the project within the Cluster of Excellence in Simulation Technology (EXC 310/1) at the University of Stuttgart.

References

1. G. B. Berriman, E. Deelman, J. Good et al.: Generating Complex Astronomy Workflows. In: I. Taylor, E. Deelman, D. B. Gannon, M. Shields (Eds.): Workflows for e-Science. Springer, 2007
2. P. Maechling, E. Deelman, L. Zhao et al.: SCEC CyberShake Workflows—Automating Probabilistic Seismic Hazard Analysis Calculations. In: I. Taylor, E. Deelman, D. B. Gannon, M. Shields (Eds.): Workflows for e-Science. Springer, 2007
3. A. Jones: Workflow and Biodiversity e-Science. In: I. Taylor, E. Deelman, D. B. Gannon, M. Shields (Eds.): Workflows for e-Science. Springer, 2007
4. M. Sonntag and D. Karastoyanova: Next Generation Scientific Experimenting Based On the Workflow Technology. Proceedings of the 21st IASTED International Conference on Modeling and Simulation (MS'10), 2010
5. A. Akram, D. Meredith, and R. Allan: Evaluation of BPEL to scientific workflows. In: CCGRID '06: Proc. of the 6th IEEE International Symposium on Cluster Computing and the Grid, IEEE Computer Society, 2006, pp. 269–274
6. I. Wassink, M. Ooms, and P. van der Vet: Designing workflows on the fly using e-BioFlow. Lecture Notes in Computer Science (LNCS), vol. 5900, 2009, pp. 470–484
7. G. Vossen and M. Weske: The WASA approach to workflow management for scientific applications. In: Dogac et al. (Eds.): Workflow Management Systems and Interoperability, NATO ASI Series F: Computer and System Sciences, vol. 164, Springer-Verlag, Berlin, 1998, pp. 145–164
8. O. Kopp, S. Henke, D. Karastoyanova et al.: An event model for WS-BPEL 2.0. Technical Report No. 2011/07, University of Stuttgart, Germany, 2011
9. M. Sonntag, S. Hotta, D. Karastoyanova, D. Molnar, and S. Schmauder: Using services and service compositions to enable the distributed execution of legacy simulation applications. In: Proceedings of the 4th European Conference ServiceWave 2011, Poznan, Poland, 2011.
10. D. Molnar, P. Binkele, S. Hocker, and S. Schmauder: Multiscale modelling of nano tensile tests for different Cu-precipitation states in α-Fe. In: Proceedings of the 5th International Conference on Multiscale Materials Modelling, Fraunhofer Verlag, 2010, pp. 235–239
11. M. Sonntag and D. Karastoyanova: Ad hoc Iteration and Re-execution of Activities in Workflows. In: IARIA (Hrsg): International Journal On Advances in Software, ISSN 1942-2628, vol. 5, no. 1 & 2, Xpert Publishing Services, 2012, pp. 91–109

[7] http://www.simtech.uni-stuttgart.de/index.en.html

[8] http://www.iaas.uni-stuttgart.de/institut/mitarbeiter/sonntag/indexE.php#mayflowerVideo