NEXT GENERATION INTERACTIVE SCIENTIFIC EXPERIMENTING BASED ON THE WORKFLOW TECHNOLOGY

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

ABSTRACT
In this paper we explore to what extent the conventional workflow technology and service-oriented architecture (SOA) principles can be applied to support scientist in their experiments. Based on the requirements imposed on systems for scientific computing, e-Science and simulations, and an extended workflow life cycle we introduce the architecture of an interactive system that reuses the conventional workflow technology. We advocate the realization of this workflow system with advanced adaptation and monitoring features because we identified that modeling of scientific applications and simulations can only be done the "scientists' way" if the traditional workflow modeling as well as design and run time adaptation are combined in a user-friendly solution.

KEY WORDS
Simulation tools and languages, Workflow management, Workflow adaptation, SOA

1. Introduction

The utilization of workflows for scientific applications and simulations is subject to current research efforts [1]. The features of workflows address many requirements of scientists on the IT support for their experiments, computations, and simulations [2, 3, 4]. Such features help scientists to design, conduct, monitor, and analyze experiments, to share and reproduce results, and to follow the trial-and-error approach that is a typical procedure in e-Science [3].

Nevertheless, just the first steps are taken to adopt the workflow technology (as described in [5] and by the WfMC[1]) in e-Science. The main reason is the difference in the definitions of the term "workflow" used in the business and scientific workflow communities, which results into a discrepancy of how the different communities view/understand workflows. Obviously, there is a need to reconcile and clarify the definitions of the two communities. Another reason is the set of missing features needed by scientists for complete and intuitive support of scientific simulations, complex computations and experiments [2, 3]. For example, business workflow management systems (WfMSs) are not specialized in handling huge amounts of data or data pipelines. Additionally, provenance tracking or adaptability mechanisms are insufficiently implemented.

Our goal is to develop a WfMS based on the traditional workflow technology tailored to the needs of scientists and scientific applications without limiting it to a particular scientific domain. In this paper, we therefore identify and address a sub-set of requirements, namely the need for enhanced adaptation and monitoring of workflows. While in business applications workflow adaptation [6] and monitoring are a well-known and investigated need, in scientific applications these features have not been addressed by research. In order to realize our vision we advocate the reuse of existing workflow technologies and techniques. Carefully designed extensions will be needed to meet the specific requirements of e-Science [7].

With this work we contribute a life cycle definition for workflows accommodating the needs of scientists for modeling and executing scientific simulations or experiments. Additionally, we extend the life cycle of traditional workflows to reflect the identified needs. It points out to what extent the existing technologies can be reused to support scientists in their work and what the unaddressed issues are. The vision and the architecture of an interactive framework for scientific experiments leveraging conventional workflow technology and service-oriented architecture (SOA) principles is the one additional major contribution of this work.

The rest of the paper is organized as follows: Section 2 outlines benefits of the conventional workflow technology when being applied to the field of scientific workflows. Section 3 identifies requirements on monitoring and adaptation of scientific workflows with the help of existing scientific workflow management systems (SWfMSs) and the life cycle of scientific workflows. Section 4 extends the life cycle of conventional workflows to accommodate the features needed in scientific experiments and simulations. This extension has the purpose of fostering mutual understanding of the two communities. Additionally, the vision and architecture of a workflow-based infrastructure with advanced monitoring and adaptability features is presented.

[1] Workflow Management Coalition, http://www.wfmc.org

Table 1: Comparison of SWfMSs according to monitoring and adaptability features. Available properties are indicated by "x", unavailable ones are denoted by "-".

		Kepler	Triana	Taverna	Pegasus	e-BioFlow
Monitoring	**Aggregated statistics**	-	-	-	x	-
	Suspend, resume workflow	x	x	-	-	-
	Progress display	x	x	x (separate view)	by Condor Job Monitor	x (state of activities in table)
	Display past runs	-	-	x	x	x (as activity states)
Adaptation	**Execution of workflow fragments**	-	-	-	-	x
	Manual adaptation of running (fragments of) workflows	parameters	parameters	-	-	re-execution of workflow fragments
	Automatic run time adaptations	-	-	retry, alternative services	workflow reduction, resource provisioning, retry	-

Section 5 outlines related work in the field of conventional and scientific workflows. The paper is closed by the conclusions section.

2. Benefits of applying the conventional workflow technology to the scientific domain

Our vision is driven by the number of advantages the workflow technology delivers to scientific applications [1]. Existing software with features that are hard to implement (e.g. scalability with respect to resources or workflows, robustness) can be used as fundament for an advanced SWfMS.

As opposed to most SWfMSs, traditional workflow technology distinguishes between workflow models and instances. This concept is beneficial for modeling scientific simulations. For example, a traffic simulation may comprise several roads, cars and traffic lights. Naturally, each type of objects has similar behaviour (e.g. all cars stop at red signals), but each object can be configured differently (e.g. different cars have different velocities). That means each type can be designed as a workflow model; each specific object is represented by a workflow instance. All instances together can be seen as the simulation context.

Often, simulations have to be steered by scientists if decisions cannot be made by machines (e.g. to assess the quality of a scientific result). The integration of humans in workflows (so-called human tasks) is a common use case in business workflow scenarios, while most scientific WfMSs lack this kind of feature.

Considering the concrete workflow language BPEL (Business Process Execution Language) [8] several additional advantages come into play. Some of them are already discussed in [4], for instance. BPEL enables an asynchronous, non-blocking invocation of Web services (WSs) and hence decreases idle time of resources. Late binding of WSs is facilitated since only the interfaces of used WSs need to be known at design time and not the concrete location. BPEL's fault handling and compensation mechanisms contribute to the fault tolerance of the system and the consistency of results. The recursive programming model fosters reusability of workflows and collaboration between scientists. Such features are hardly provided by existing scientific workflow systems (e.g. Kepler, Triana, Taverna).

3. Deriving requirements on monitoring and adaptability of scientific workflows

To the best of our knowledge, requirements on monitoring and adaptability of applications used for scientific experiments have not been collected so far. We derive these requirements from a set of representative SWfMSs and the life cycle of scientific workflows. The life cycle is inferred from the techniques scientists apply when conduction experiments or simulations.

3.1 Monitoring and adaptability features of existing scientific workflow systems

There is a wide variety of SWfMSs, each of which having specific characteristics, such as being created for a particular scientific domain or implementing a certain workflow language. We analyze popular, state-of-the-art SWfMSs with respect to their monitoring and adaptability capabilities (see Table 1).

Kepler [9] provides a monitoring component (Kepler Execution Monitoring, KEM) to visualize the overall workflow progress in general as well as the actor states in particular during experiment execution. Monitoring can be configured by customizable symbols assigned to actors (e.g. traffic lights, progress bar). It is not possible to inspect past experiment runs. Kepler's run time adaptation features are limited to modifying parameter values of suspended experiments only. Suspending of experiments is only possible after an experiment instance finishes and before the next one starts (in scenarios where experiment runs are iterated).

Triana's [10] monitoring capabilities are integrated in the modeling view of the system. Monitoring is limited to highlighting active tasks. Workflow instances can be

stopped and resumed at run time but only parameters can be modified on the fly.
Taverna [11] provides a monitoring view separated from the modeling view. Scientists are enabled to inspect the progress of running workflows and the history of past workflow runs. The system does not support manual adaptations during the execution of workflows. Taverna allows specifying retry parameters and alternative services for activities at build time to be followed automatically at run time in case of errors. This function is not yet supported by the GUI. Hence appropriate code has to be inserted by hand into the workflow file.
Pegasus [12] provides a monitoring tool for visualizing statistics over workflow instances, e.g. the number of jobs per hour. The progress of running workflows can be monitored with the separated Condor Job Monitor tool. Pegasus enables automatic adaptations of workflows at run time: redundant data management tasks are omitted if the data to be produced is already available, retry of faulted tasks, (re-)mapping of tasks on resources.
The e-BioFlow system [13] supports the execution of workflow fragments and to use the results for further processing. As adaptation mechanism the re-execution of workflow fragments is implemented. Monitoring the workflow progress is realized by a table showing activity states. Past workflow runs are shown in the same table as activities with completed state.

3.2 Life cycle for scientific workflows

A key property of business workflow technology is a life cycle arranging separated, repeatable management phases handled by different user groups (Figure 1).

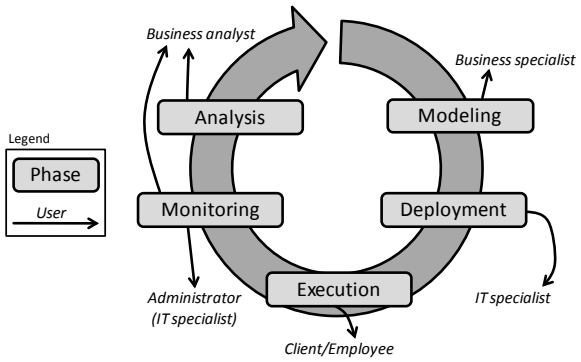

Figure 1: Life cycle of workflows in business process management (BPM).

After modeling, a workflow is explicitly deployed on an engine. The workflow execution is either explicitly started by an employee or implicitly by an incoming message sent from a client application. This may happen much later after deployment. Multiple instances of the same workflow model can be created and carried out. Monitoring collects, processes, and displays information over process instances enabling users to follow process execution. After workflow execution, an analysis phase may reveal the need for model changes (business process reengineering) so that the cycle repeats. Changes may also be done during workflow (instance) execution.
Although it is extremely important to know the life cycle of applications, in scientific workflow management, there is typically no workflow life cycle definition. To overcome this burden, we want to derive a life cycle for scientific workflows from typical and well-known properties of scientific experimenting, such as the trial-and-error approach [3]. The following example illustrates this.

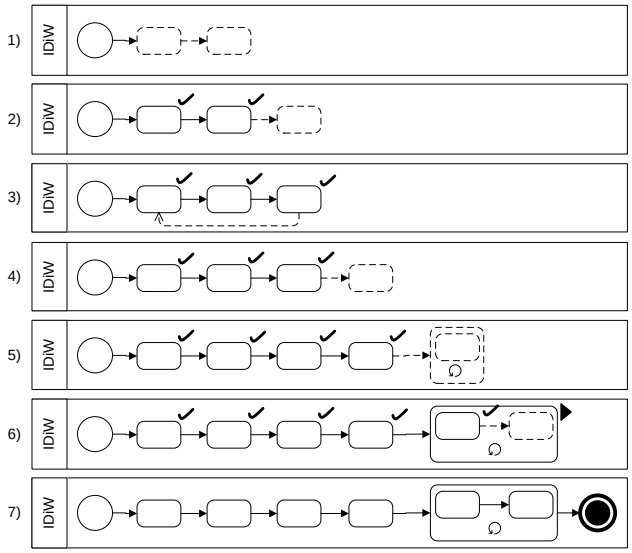

Figure 2: Evolving simulation workflow by the example of ink diffusion in water (IDiW). Dotted arrows or activities are newly added in the particular step. The arrow in step 3 denotes a re-execution of activities. Checks mark finished activities, the playback symbol marks running activities.

Imagine a scientist (with name Jim) that wants to create a simulation workflow to calculate and visualize the ink diffusion in a glass of water over a period of time. Such a simulation can be expressed as partial differential equation (PDE) and can be solved with the help of Dune, a C++ framework working with the finite element method (FEM). First, Jim selects activities to create and refine an FEM mesh that represents the glass of water (see Figure 2, step 1). He sets parameters to get an equidistant 3D FEM mesh and to refine the ink injection point as an area of increased importance. After that he runs the simulation workflow (fragment). Jim appends a visualization of the resulting mesh and resumes the simulation to get to know whether he is confident with the mesh (2). With the help of the visualization of the mesh he detects the mesh distance to be too large. Therefore, Jim adjusts the appropriate parameters and re-executes the three activities to eventually get the desired mesh (3). He appends an activity to set up the mesh with initial and boundary conditions (e.g. the velocity of the ink diffusion or the injection area) and resumes the workflow (4). The simulation setup is now complete. For the actual simulation Jim places an activity to solve the PDE of the scenario at a particular time step. To realize several different time steps he surrounds the solver with a loop

(5). Jim resumes the workflow and monitors the progress of the workflow instance. After a few iterations he decides to suspend the workflow to insert a result visualization activity (6). When resuming the workflow the new activity creates an image of the intermediate result of the investigated ink diffusion. Jim has now completed the workflow to simulate the ink diffusion in water (7).

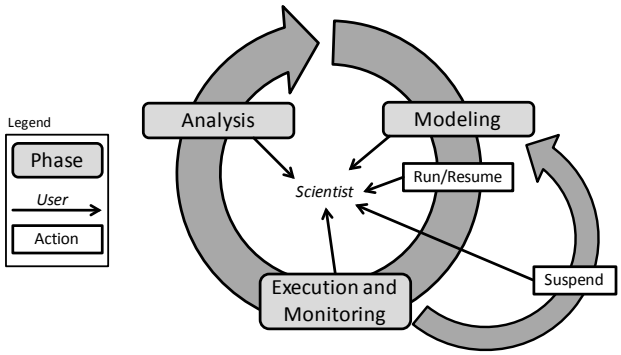

Figure 3: Life cycle of scientific workflows how scientists experience it

Based on this example we can derive a life cycle for scientific workflows that accounts for the typical actions of scientists when conducting experiments (Figure 3). Scientists are most often the only user group of SWfMSs (typically playing the roles of both user and administrator). They design parts of an experiment and start it before having completed modeling the whole experiment. Scientists do not cope with (and in fact, do not want to cope with) technical details such as a deployment mechanism. Instead a "run/resume" operation immediately starts a selected workflow. Scientists require the ability to suspend running experiments, conduct adaptive or completing actions and resume the experiments. From a scientist's perspective there is no difference between modeling and run time adaptation phases. Scientists experience these phases as modeling of experiments only. This is due to the fact that they are unaware of the separation between workflow models and instances. Furthermore execution and monitoring phases are combined into a single phase because from the scientist's point of view there is no difference between these two. Monitoring is de facto the visual representation of a running workflow. An analysis phase can follow subsequently to examine past workflow runs what can influence workflow models and trigger a re-execution. The life cycle reflects a concept we call *model-as-you-go*: the blending of modeling, execution, run time adaptation, and monitoring phases.

3.3 Identifying requirements

From the observed properties of existing SWfMSs and the life cycle of scientific workflows we derive requirements on monitoring and adaptability of scientific workflows. The considered workflow systems are developed for different application areas and hence only implement selected monitoring and adaptability features. Since we want to design a general SWfMS without a specialization to a particular domain, the identified requirements cover all features of Table 1.

R1. A customizable monitoring component should be integrated into the SWfMS's modeling tool. This reflects the blending of modeling and monitoring phases.
R2. The progress of single workflow instances needs to be observable by scientists in a graph-based visualization. This represents the blending of modeling, execution, and monitoring phases.
R3. Scientists should be able to inspect past workflow runs. This satisfies the analysis phase.
R4. Aggregated statistics over workflow instances and utilized software and hardware enable scientists to assess workflow execution in the employed infrastructure (see Table 1, Pegasus).
R5. Steering of workflow execution (run, suspend, resume) should be possible. This enables scientists to (unconsciously) switch between modeling and execution phases.
R6. Manual run time adaptation of workflows in all dimensions is needed in a straight forward manner.
R7. Automatic run time adaptability capabilities are needed to react on infrastructure changes without human intervention (see Table 1, Taverna and Pegasus).
R8. The execution of workflow fragments should be possible to allow scientists testing incomplete workflows (see Table 1, e-BioFlow).
R9. Tracking and displaying provenance information need to account for the changing nature of adapted workflows.

4. Framework for Interactive Scientific Experimenting

In order to comply with the observations of the previous section as well as to meet the gathered requirements on monitoring and run time adaptability, we envision an interactive framework for scientific workflows based on the conventional workflow technology.

4.1 Extending the life cycle of conventional workflows

To support a traditional software engineering approach, we extend the life cycle of business workflows to accommodate the model-as-you-go approach of creating and executing scientific workflows (Figure 4). It is geared towards the life cycle of Figure 3 but considers the technical details behind the "run/resume" operation and the adaptation mechanisms. It helps bringing the two communities together and creates a common base of understanding between them. The adaptation cycle is divided into two cycles. The lower one denotes

adaptations on the functions dimension; the right-hand side cycle stands for modifications on the logic dimension. In the latter case, a (re)deployment of parts of the experiment is entailed and may need to deal with migrating running experiment instances. SWfMSs typically abstain from a deployment mechanism because there is often only a single instance of a workflow model that is started directly by a scientist. Furthermore, GUI and engine are often tightly coupled so that there is no need for a translation of the model into an engine-optimized, executable format. After modeling the process model is already in the required format. Nevertheless, we advocate the adoption of the deployment phase because of its many advantages like reusability of workflow models or efficient execution of workflow instances [14]. Of course, actual deployment should be hidden for scientists behind the mentioned "run/resume" operations.

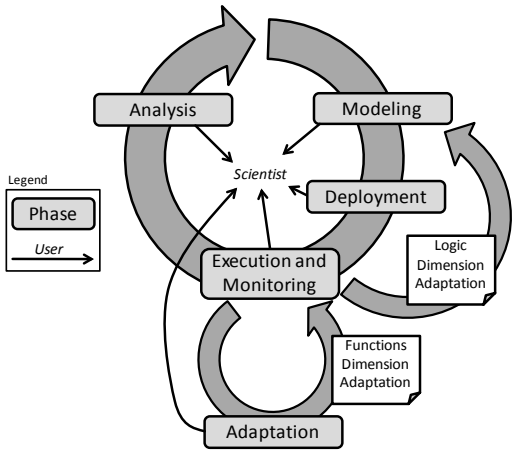

Figure 4: Life cycle of scientific workflows mapped on traditional workflow life cycle

The extended life cycle reveals that existing workflow technologies can be reused to support scientists in their work. But it also exhibits the need for advancements and extensions of the existing approaches: first, due to the fact that there is only a single user group in scientific workflow management, an integrated tool is needed that supports a scientist in all phases of the life cycle [14]. Second, logic dimension run time adaptation is crucial in SWFM, but not yet supported in established BPM solutions such as IBM WebSphere Process Server [15] or Oracle BPEL Process Server [16]. Third, an integrated monitoring tool is needed that enables scientists to follow the progress of a single workflow instance. In business WfMSs monitoring tools are typically standalone calculating and displaying data about several workflow instances [17].

There are reasons for the discrepancy between conventional workflow technologies and the way scientists work and think. Separation of roles and hence tools is wished and important due to different rights and domain knowledge of the users [14]. Structural adaptation of single process instances is not provided because processes often implement products offered by enterprises (e.g. a loan approval). These products do not change during their creation. In industry therefore process evolution is typically addressed by a versioning mechanism: multiple versions of a process model exist; running instances are executed pursuant to their (possibly old) model version; new instances are created according to a new model version. Monitoring of single workflow instances is of minor interest. Business analysts (see Figure 1) rather care about an overall snapshot of running processes.

4.2 Architecture of the proposed framework

The proposed interactive framework comprises an integrated monitoring component with novel characteristics. The monitoring solution must be an integral part of a SWfMS so as to prevent scientist from switching between tools (meeting requirement R1 of Section 3.3). Figure 5 presents a general high-level architecture of the envisioned SWfMS with its main components. The architecture is based on SOA principles. Resources (sensors, databases, etc.) are also available as services. These services are composed into workflows and accessed over a service bus. Since the interactive framework is the focus of this work, the details about other components are omitted. For the implementation of the architecture we want to pursue an engineering approach where we reuse as much as possible of existing concepts, technologies, and software.

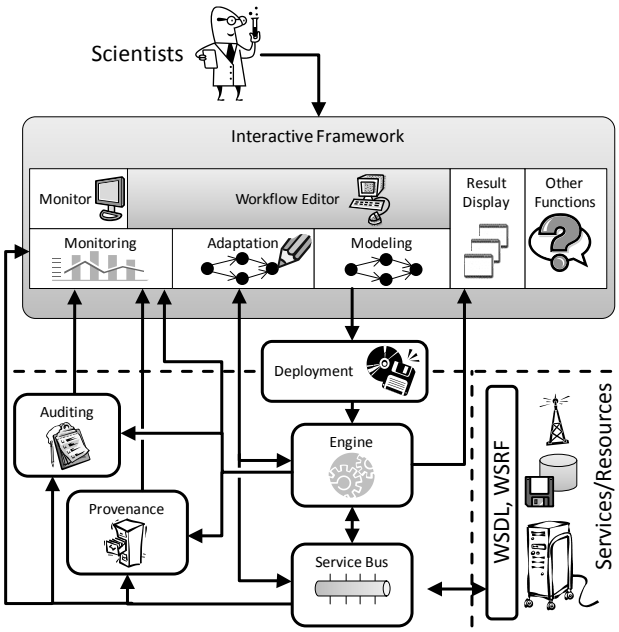

Figure 5: General high-level architecture of a SWfMS based on SOA principles. Gray components are novel contributions, white ones are based on existing concepts but need further extensions to meet the special requirements of scientific workflows.

The *monitor* component is a GUI for scientists mainly providing functionality known from business workflow monitoring. It visualizes statistics (R4) and additional information about workflow instances and instance groups with the help of appropriate diagrams [e.g.

distribution of used services among a network, key performance indicators (KPIs), or service level agreements (SLAs)]. Moreover, it facilitates monitoring of provenance information (R9) for scientific workflows which is not yet supported by the conventional workflow technology and infrastructures.

The *workflow editor* is a GUI that realizes the needed blending of modeling, execution, monitoring, and adaptation functionality to implement the model-as-you-go approach. It enables steering of workflows (R5). The underlying mechanisms are employed transparently for the users (e.g. the deployment mechanism or the difference between workflow models and instances) and their complexity is hidden. The mechanisms for the editor can mainly be reused from existing business WfMSs but definitely need to be adapted in order to meet the special requirements of scientific workflows, such as the specification of data flow [2, 3].

The *result display* component visualizes intermediary and final results of scientific computations and simulations, e.g. with the help of images or videos. Scientists should be enabled to step into the visualization of their results, e.g. by changing a perspective or a time interval. We are currently working on a result display component that makes use of visualization workflows to calculate and view images as result of simulation workflows. The used functions are provided by an appropriate visualization tool. A GUI will allow scientists to interactively tune parameters and thus change the view on their results.

The *other functions* component is an extension point of the foreseen framework. It represents functionality that is beyond the scope of this work, such as a catalogue of services where scientists can register their services to be later used for modeling of scientific workflows.

The *monitoring* component mainly implements functions known from BPM but needs at least extensions to be able to deal with provenance information. It uses auditing or historical data (i.e. data about events occurred during workflow execution) to deduce the current progress or past runs of single instances of workflow models (R2 and R3), to calculate statistics (e.g. server workloads, run time of tasks), or create diagrams for analysis. Auditing and provenance information can be used by scientists to reproduce workflow runs. In [18] additional requirements on monitoring in Grid environments are described which should be dealt with by the monitoring component of the proposed architecture: fault detection, different monitoring levels (workflows, services, resources, and infrastructure), policies, key performance indicators (KPIs), or service level agreements (SLAs).

The *modeling* component provides functions to enable the design of scientific workflows. Although any workflow language can be used, we advocate a language that implements SOA capabilities our architecture is based on. An eligible candidate is BPEL, which however needs extensions to satisfy the requirements on data flow and flexibility of scientific workflows. The design and deployment of partially designed workflows is necessary due to the evolving nature of scientific workflows (R8). Deploying partial process models is still an open research issue.

The *adaptation* component contains logic for full run time adaptability support in all workflow dimensions which is currently not covered by any existing SWfMS. These features help scientists to create their workflows incrementally and at the same time allow for process repair capabilities. Existing concepts from the business domain can be adopted, such as parameterized WS-flows with query strategy [19] or the WSDL-less BPEL dialect $BPEL^{light}$ [20] (R7), $ADEPT_{flex}$ [21] (R6), fragments in processes [22], or BPEL'n'Aspects [23] (R6). Enhancements of the existing approaches and novel concepts will be needed.

Figure 6 presents a deeper insight into the proposed framework. The *instance monitor* of the workflow editor visualizes the progress of running workflows in a graph-based manner. The needed auditing and provenance information are collected with the help of mining techniques [24].

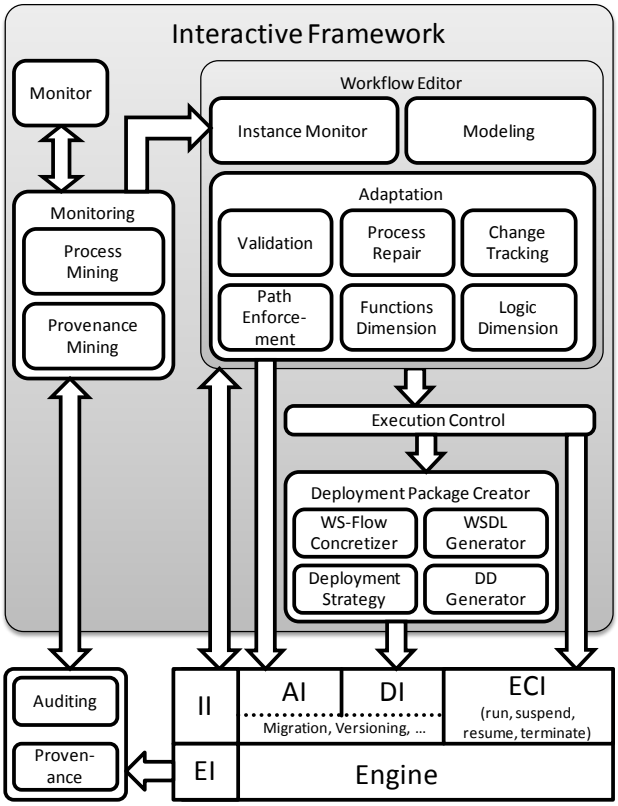

Figure 6: Detailed architecture of the proposed interactive framework including needed interfaces of the used workflow engine (EI – Event Interface, III – Instance Information Interface, ECI – Execution Control Interface).

Besides the already mentioned features the *adaptation* component allows scientists to enforce paths to be taken during workflow execution independent of evaluated transition or join conditions. Change tracking of a workflow is important for undoing modifications, for reproducing workflow runs, and compensation of adapted workflows. Additionally, it should not be possible to invalidly adapt workflows. For example, the past of a

workflow instance cannot be changed. This is ensured by a validation component.

The *execution control* component is used to manually start, suspend, resume, and terminate workflow instances from within the workflow editor which is typically not supported by traditional workflow tools. Additionally, since scientists should be oblivious to workflow deployment, it is hidden by the execution control. That means starting a new workflow causes a deployment step in the background.

The *pre-deployment* component prepares a workflow bundle for deployment. This can comprise the generation of a deployment descriptor, the generation of a WSDL document if the workflow is exposed as WS, and the concretization of abstract parts (e.g. if workflow templates [19] are used). The *deployment strategy* component addresses two major issues. First, resuming a workflow can result in a re-deployment of workflow parts if these parts are added or changed in an adaptation step. It must be ensured that finished or not modified workflow parts are not re-deployed. Second, a workflow can potentially be divided into several parts getting deployed on several machines. Such a scenario can be enabled, e.g. with a process space-based workflow enactment [25]. Different distribution strategies need to be implemented (e.g. distribution of activities near used data sources).

We identified a number of interfaces our foreseen workflow engine needs to provide. A workflow bundle can get deployed over the *deployment interface*. The interface must also be able to deal with the re-deployment of an already deployed workflow in case of an adaptation of its logic dimension. The *adaptation interface* is used to modify a workflow's function dimension where no re-deployment is required. Since both interfaces deal with adaptations, they must implement concepts to handle such modifications (e.g. instance migration or versioning). The *instance information interface* (III) is needed for the process repair mechanism to fetch and change workflow instance data (e.g. the content of variables). The *execution control interface* (ECI) allows steering workflow execution (i.e. to run, suspend, resume, and terminate a workflow). The *event interface* (EI) is used to fetch auditing and provenance information. The concrete realization of the interfaces is implementation dependent (e.g. as message queue or WS interface). ECI and EI are strongly connected. That means an ECI command causes an event to be published over the EI to be eventually visualized in the instance monitor.

Integrating the mentioned approaches will be challenging. We are currently working on a prototype for the proposed workflow editor with advanced monitoring and adaptability features. It will proof the applicability of the model-as-you-go concept.

5. Related Work

As mentioned, workflow adaptation is a well-known field in conventional WFM. There are several workflow systems that account for the modification of workflows at run time. $ADEPT_{flex}$ [21] introduces a new meta model with particular operations that can change the structure of running workflows. The focus is thereby on preserving structural correctness and consistency. InConcert [26] is a commercial system that allows modifying the task structure of a job to deal with deviations in the original process. The SWATS [27] system offers modification services to adapt workflow instances. Special attention is paid on user authorization, model consistency, and integrity rules. These workflow systems are tailored to an application in business scenarios and hence are not intended for the use in the area of scientific workflows with its unique properties [2, 3].

In e-Science, run time adaptation of workflows is currently insufficiently addressed by research efforts. The e-BioFlow system [13] contains an ad-hoc editor that allows (re-)executing workflow fragments and using the results for further processing. Kepler [9] and Triana [10] enable scientists to change parameter values during workflow execution. Pegasus [12] and Taverna [11] provide functionality for automatic workflow adaptation. As shown in Section 3.1, these systems implement a subset of monitoring and adaptation features since this subset seems to be sufficient for the respective application domains. Especially, none of the systems provides both manual and automatic adaptation features. Our vision and challenge is a scientific WfMS based on conventional workflow technology implementing all of the outlined features. The system is intended to be independent of a specific scientific domain.

6. Conclusion and Future Work

With this work we contribute an overview of the limitations of conventional service-based workflows when being used for scientific simulations and experiments. We show potential opportunities for enhancing them for an application in the scientific domain. A major contribution is the extension of the life cycle of conventional workflows towards scientific experiments, which will enormously foster mutual understanding between the two communities. The extended life cycle reflects the design and execution of scientific workflows in a model-as-you-go manner: it blends the phases modeling, execution, run time adaptation, and monitoring to accommodate the way scientists typically conduct experiments, simulations, and computations. The concept of workflow models and instances is thereby hidden from scientists.

In order to implement the proposed life cycle we presented the vision and architecture of an interactive infrastructure for e-Science based on conventional workflow technology and service-oriented computing. The infrastructure is designed to address the derived requirements towards adaptation and monitoring of scientific experiments, which results in specific but not yet addressed requirements towards the conventional

workflow infrastructures. To maintain industry and standardization relevance the solution reuses as much as possible of existing infrastructures and technologies. Nevertheless, the existing concepts need extensions to fit the special characteristics of scientific workflows. Additionally, to be of real value to the scientists, the framework is meant to hide the technology idiosyncrasies from scientists and to provide an intuitive tool for scientific experiments without increasing the learning curve.

Our future work will focus on designing and implementing the components of the presented infrastructure. In particular, devising adaptation operations on scientific workflows and monitoring for scientific applications will have the major focus.

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] Taylor et al., *Workflows for e-Science: Scientific workflows for grids* (Springer, 2007).

[2] Y. Gil et al., Examining the challenges of scientific workflows, *IEEE Computer*, 40(12), 2007.

[3] Barga & Gannon, Scientific versus business workflows. In: Taylor et al. (Eds.), *Workflows for e-Science: Scientific Workflows for Grids* (Springer, 2007).

[4] Akram et al., Evaluation of BPEL to scientific workflows, *Proc. of 6th IEEE International Symposium on Cluster Computing and the Grid*, 2006.

[5] Leymann & Roller, *Production workflow: Concepts and techniques* (Prentice Hall, 2000).

[6] Gehlert et al., Online testing, requirements engineering and service faults as drivers for adapting service compositions. *Service Wave 2008*, MONA+, 2009.

[7] Ludaescher et al., Scientific workflows: Business as usual?, *Proc. of 7th International Conf. on Business Process Management (BPM)*, 2009.

[8] OASIS, Web services business process execution language (BPEL) version 2.0, OASIS Standard, 2007.

[9] Altintas et al., Kepler: An extensible system for design and execution of scientific workflows, *Proc. International Conf. on Scientific and Statistical Database Management*, 2004.

[10] Churches et al., Programming scientific and distributed workflow with Triana services, In: *Concurrency and Computation: Practice and Experience*. Special Issue on Scientific Workflows, 2005.

[11] Oinn et al., Taverna: Lessons in creating a workflow environment for the life sciences, In: *Concurrency and Computation: Practice and Experience*, 2006, 18(10):1067-110. DOI: 10.1002/cpe.993.

[12] Deelman et al., Pegasus: Mapping scientific workflows onto the grid, *Proc. of 2nd European AcrossGrids Conf.*, Springer, 2004, pp. 11-20.

[13] I. Wassink et al, Designing workflows on the fly using e-BioFlow, *Proc. 7th Conf. on Service Computing (ICSOC)*, 2009.

[14] Sonntag et al., The missing features of workflow systems for scientific computations, *Proc. of Grid Workflow Workshop (GWW)*, 2010.

[15] IBM, WebSphere process server. [Online] `http://www-01.ibm.com/software/integration/wps/`

[16] Oracle, Oracle BPEL process manager. [Online] `http://www.oracle.com/appserver/bpel_home.html`
`http://www.cs.wisc.edu/condor/dagman/`

[17] Wagner et al., Vergleich von Business Activity Monitoring Werkzeugen, *Fachstudie Softwaretechnik No. 76*, University of Stuttgart, 2007.

[18] Mietzner et al., Business grid: Combining web services and the grid, In: *Transactions on Petri Nets and Other Models of Concurrency (ToPNoC)*, Special Issue on Concurrency in Process-aware Information Systems, Springer Verlag, 2009.

[19] D. Karastoyanova, Enhancing flexibility and reusability of web service flows through parameterization, PhD thesis, TU Darmstadt and University of Stuttgart, 2006.

[20] Nitzsche et al., $BPEL^{light}$, *Proc. 5th International Conf. on Business Process Management (BPM)*, 2007.

[21] Reichert and Dadam, $ADEPT_{flex}$ – Supporting dynamic changes of workflows without losing control, *Journal of Intelligent Information Systems*, 10(2), 1998.

[22] Eberle et al., Process fragments, *OTM Part I*, 2009.

[23] Karastoyanova and Leymann, Making scientific applications on the grid reliable through flexibility approaches borrowed from service compositions, In: Antonopoulos et al. (Eds.), *Handbook of research on P2P and grid systems for service-oriented computing: Models, methodologies and applications* (Information Science Publishing, 2009).

[24] Van Dongen et al., The ProM framework: A new era in process mining tool support, *LNCS*, volume 3536, pages 444-454, Springer-Verlag, 2005.

[25] Sonntag et al., Process space-based scientific workflow enactment, *International Journal of Business Process Integration and Management (IJBPIM)*, Special Issue on Scientific Workflows, Inderscience Publishers, 2010. (to appear)

[26] McCarthy and Sarin, Workflow and transactions in InConcert, *IEEE Bull. of the Tech. Com. on Data Engineering*, Vol. 16, No. 2, 1993.

[27] R. Siebert, An open architecture for adaptive workflow management systems, *Proc. International Workshop on Issues and Applications in Database Technology (IADT)*, 1998.