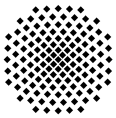

Universität Stuttgart

SimTech Cluster of Excellence

M. Sonntag[1] D. Karastoyanova[1] E. Deelman[2]

Bridging The Gap Between Business And Scientific Workflows

Stuttgart, December 2010

[1] Institute of Architecture of Application Systems (IAAS)
University of Stuttgart,
Universitaetsstrasse 38
70569 Stuttgart, Germany
{sonntag, karastoyanova}@iaas.uni-stuttgart.de
http://www.iaas.uni-stuttgart.de

[2] Information Science Institutes (ISI)
University of Southern California
Admiralty Way 4676
90292 Marina Del Rey, California, USA
deelman@isi.edu
http://www.isi.edu

Abstract Due to their different target applications business and scientific workflow systems provide different sets of features to their users. Significant amount of research is currently being done to employ the business workflow technology in the scientific domain. This usually means extending the workflow language and thus the modeling tool and execution engine. In this paper we aim to bring business and scientific workflows together in order to exploit the advantages of both. We explore the interplay between business and scientific workflows in the context of human interactions with the management of workflow execution. We present an approach and implementation based on BPEL and Pegasus and show that the approach can be beneficial to scientists.

Keywords Scientific workflows, Business workflows, Human tasks, Pegasus, BPEL.

Reference Sonntag, M., Karastoyanova, D., and Deelman, E. (2010) Bridging The Gap Between Business And Scientific Workflows. In: Proceedings of the 6th IEEE International Conference on e-Science, IEEE Computer Society.

Stuttgart Research Centre for Simulation Technology (SRC SimTech)

SimTech – Cluster of Excellence
Pfaffenwaldring 7a
70569 Stuttgart
publications@simtech.uni-stuttgart.de
www.simtech.uni-stuttgart.de

Bridging The Gap Between Business And Scientific Workflows

Humans In The Loop Of Scientific Workflows

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

Ewa Deelman
Information Science Institutes
University of Southern California
Marina Del Rey, USA
deelman@isi.edu

***Abstract*—Due to their different target applications business and scientific workflow systems provide different sets of features to their users. Significant amount of research is currently being done to employ the business workflow technology in the scientific domain. This usually means extending the workflow language and thus the modeling tool and execution engine. In this paper we aim to bring business and scientific workflows together in order to exploit the advantages of both. We explore the interplay between business and scientific workflows in the context of human interactions with the management of workflow execution. We present an approach and implementation based on BPEL and Pegasus and show that the approach can be beneficial to scientists.**

Scientific workflows, business workflows, human tasks, Pegasus, BPEL

I. INTRODUCTION

The success of workflow management systems (WfMS) in business scenarios recently resulted in the introduction of workflows to scientific calculations, simulations, and experiments. Scientists and scientific applications impose new requirements on the employed workflow technology. That is the main reason why scientific WfMSs do not utilize existing workflow systems of the business domain [1] and why these two categories of WfMSs developed. Both have their strengths and weaknesses and right to exist. *Business WfMSs* are usually based on agreed-upon standards in order to facilitate communication between different software systems and companies. The workflow logic is control flow-driven and includes constructs to specify paths and conditions. Typically, a business workflow implements a company's product or service. That means a robust execution is of utmost importance because a customer pays for it. The trend is to attempt to include as many automated steps as possible in workflows, however only some services can be offered fully automatically. An integration of human tasks (HTs) into a business workflow is therefore common. Business WfMSs do not natively support the specification of explicit data flow, the exact reproducibility of workflows, or processing of data streams—to name some of the disadvantages when being applied in the scientific domain. *Scientific WfMSs* often deal with huge amounts of data and/or complex calculations and hence utilize large storage capacities and computing resources. That is why scientific workflows are often executed in a cluster or Grid environment. A scientific workflow solves a particular scientific problem and is often itself subject to scientific research. Since data are at the center of scientific applications, scientific workflows are typically data flow-driven and do not possess rich control flow structures. They resemble batch processing programs and hence do not distinguish between workflow models and workflow instances. Moreover scientific workflows are time consuming. Usually scientific WfMSs are limited in functionality for HTs in workflows, fault handling, transactions, or quality of service features.

Recent efforts are carried out to introduce the business workflow technology to the scientific domain [2, 3, 4, 5]. This approach is only natural because of the numerous advantages this mature technology brings. But there is obviously a gap between the features business workflows provide and the requirements scientists and scientific applications have [1]. Business WfMSs therefore need a set of thorough extensions when being applied to the scientific domain, which involves great development efforts. In this paper, we therefore want to investigate another, different approach. We combine business and scientific workflows to harness the advantages of both of them and to eventually close the gap between them. Our work is driven by the fact that despite the employment of scientific WfMSs the scientists still have to conduct manual tasks. These tasks can be tedious and often require a specific execution order. Thus, they introduce many sources of failures, e.g. setting up working directories for the computations, choosing runtime properties and parameters for experiments/simulations, setting environment variables, or starting servers. Often, these tasks are carried out on a Linux command line, which is especially difficult for users untrained in Linux. There is a great potential to automate and supervise such tasks by a workflow. Of course, human intervention is still needed for information that cannot be automatically derived, e.g. the configuration of experiment properties. But even during experiment execution HTs may be needed, e.g. to control the convergence of results and to decide about the proceedings of the experiment accordingly. To the best of our knowledge none of the scientific WfMSs supports the integration of humans in workflows in an automated fashion, with the exception of uni-

The authors M.S. and D.K. 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. E.D.'s work was supported by the National Science Foundation under grant #OCI-0722019.

directional notifications, such as email to the user. This renders business workflows with their HT capabilities a perfect candidate for the supervision of these tasks. HTs are only one of the advantages that business workflows bring into this setting. Another benefit is the specification of fault handling actions within a workflow. It can be used to model behavior that is executed when an error occurs. Typically, different fault handling behavior can be defined for different faults. Furthermore, it is common that (parts of) business processes are carried out as transactions (units of work) where either all or none of the activities are executed successfully. In case of a failure within a transaction already completed work is undone (or *compensated*, in long-running transactions). Scientific workflows can also benefit from these transaction concepts. Compensating behavior could be to clean up the execution environment by moving or deleting files/directories for a safe re-execution of an experiment.

In this paper we present a concept for the interplay of business and scientific workflows. We show that such an engineering approach can cover a broader set of requirements of scientists and scientific applications and also what it means for a user to employ two (or more) workflow systems in one setting. We discover and discuss advantages, challenges and limitations of the approach. In order to show its practical relevance we have developed a prototype that implements the presented concept. It is based on the Business Process Execution Language (BPEL) [6] as business workflow technology and on Pegasus [7] as scientific WfMS. Although the prototype is based on these two concrete technologies, the concepts are broadly applicable.

The paper is organized as follows. Section 2 shows related work on HT capabilities of scientific and business WfMSs. Section 3 presents considerations that need to be taken into account when combining business and scientific workflows. Section 4 introduces the conceptual architecture to realize this approach. Section 5 presents the prototype for the concept; Section 6 summarizes the work and draws conclusions.

II. Related Work

Pegasus, Triana [8], Kepler [9], Taverna [10], and GriCoL [11] are popular and representative scientific WfMSs. Pegasus is a compiler for scientific workflows that optimizes the workflows for an execution in Grid environments by Condor DAGMan [12]. GriCoL is a scientific workflow system specialized for an execution in Grids and for a parallel processing of tasks in a pipeline mode. Triana, Kepler and Taverna are modeling tools and execution engines for scientific workflows. None of these systems provides features to specify HTs in workflows. Triana, Kepler and Taverna workflows can invoke Web services (WSs) and hence it would be possible to wrap human behavior by WSs. But they enable only synchronous WS invocations and do not allow specifying callback operations. A HT wrapped by a synchronous WS operation is not viable in practice because scientists would have to complete a HT before a connection timeout occurs (which is usually 60 seconds). In business workflow management, integration of humans into workflows is common. BPEL4People [13] is a BPEL extension that prescribes how to incorporate human activities into BPEL workflows. BPMN [14] also foresees activities that are carried out by humans (manual and user tasks). However, neither BPEL nor BPMN natively fulfil requirements important for the execution of scientific applications, such as explicit data flow or pipeline/stream processing. IBM's BPM Suite [15] and Oracle's SOA Suite [16] also support the integration of human behavior into workflows (in fact, they rely on BPEL as the workflow language). In case handling [17] and declarative workflow systems [18] humans play an important role to carry out tasks and steer case/workflow execution. However, these systems lack native support of mechanisms crucial for scientific applications such as an execution in distributed environments or exact reproducibility of workflows.

III. Combining Business And Scientific Workflows

Scientific workflow systems support scientists in the development and execution of scientific applications. The main benefit is gained through automated and parallel execution of tasks. Another advantage of many scientific workflow systems is the built-in ability to execute jobs on Grids or clusters. This is far beyond the capabilities of typical scientific applications implemented by scripting languages like Ant and Make or by programming languages like Fortran where Grid support must be integrated by the programmer himself. *Business workflow systems* focus on different aspects. Rich control flow structures are needed to cover all possible eventualities. This incorporates methods for handling faults and spanning transactions. It is also common to have tasks in the course of action that can only be carried out by humans. Usually, business processes incorporate the execution of manual tasks, even though the workflow itself is executed and managed automatically.

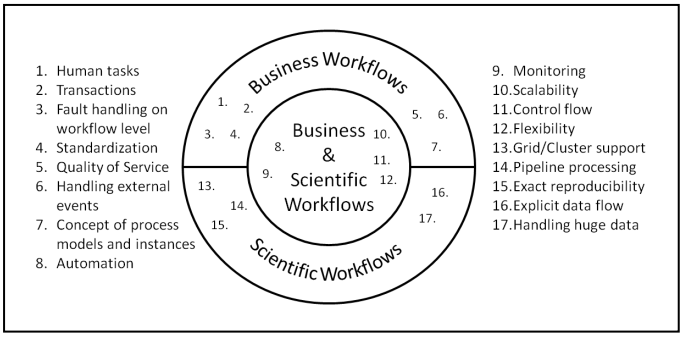

Figure 1. Covering the requirements of scientific applications

Obviously, scientific applications can benefit from the features provided by business workflow technology. In e-Science, there is a need to allow humans to steer their experiments and at the same time to increase robustness by means of appropriate fault handling and compensation mechanisms. Figure 1 illustrates a set of requirements of scientific applications (mainly taken from [19]) and how it is covered by scientific and business workflows. Note that we do not claim the set to be complete. Additionally, how the requirements are covered by business and scientific WfMSs shows only trends and cannot be generalized for all existing systems. The mapping in Figure 1 is based on our knowledge about BPEL, BPMN, Triana, Kepler, Taverna, Pegasus, and GriCoL. It reveals the gap between the capabilities of business

and scientific workflows and the discrepancy between the available support for those requirements and the support required by scientific applications.

There are two ways to leverage the strengths of the different technologies. One way is to concentrate either on a scientific or business workflow management system and incorporate extensions that cover the missing features. This is the most common path of existing work [2, 3, 4, 5, 20]. Although this would be a holistic approach, it would entail great efforts in re-implementing features already realized in the other domain. In this paper, we focus on the second possibility, namely to investigate an engineering approach that combines existing scientific and business workflow technologies in order to harness their full set of advantages. That means we try to bridge the gap between business and scientific workflows by bringing the advantages of both ways together.

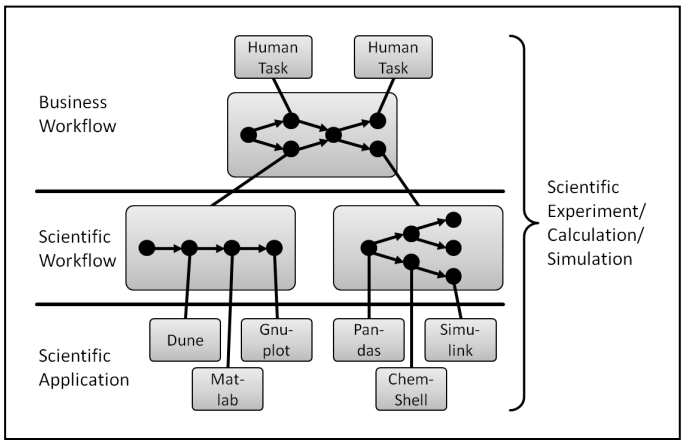

Figure 2. Relationship of business workflows, scientific workflows, and scientific applications in the proposed approach

The design is to have a business workflow system on top of one or more scientific workflow systems (Figure 2). That means this approach makes use of a business workflow to supervise the execution of scientific workflows that make use of one or more scientific applications. All three levels together can be seen as a scientific experiment or simulation. The business workflow reflects the life cycle of scientific experiments [21], i.e. setup of the environment; modeling, preparation, execution and monitoring as well as redesign of the scientific workflow; collecting and presenting results; and cleanup of the environment. Therefore the business WfMS executes and thus steers the whole process of scientific experimenting. The scientific WfMS only serves the execution phase of this life cycle and carries out the scientific workflow.

A. *Benefits*

The approach of incorporating scientific workflows into business workflows can yield a number of advantages. (1) Several steps in the life cycle of scientific workflows are conducted by humans. Integrating this human behavior into the supervising business workflows as HTs is a major benefit of the approach. HTs exactly prescribe what kind of information is needed from the scientist at what time. This promises to generate fewer failures due to human behavior, e.g. wrong configuration of the execution environment. (2) The supervising business workflow can automate manual tasks such as a tedious setup of the execution environment (e.g. installation of Grid clients, creation of directories, or staging files). (3) It is possible to add a GUI for scientists on top of the scientific workflow system which is especially useful for systems without GUI. (4) In business workflow management a main feature is the concept of workflow models and their instances/executions that follow the model. A workflow is modeled once and can be executed several times, even in parallel on a single engine. For communication or administration of workflow instances correlation mechanisms are applied that uniquely identify a workflow instance with the help of properties or IDs. In the scientific domain, this model/instance concept can be used to realize parameter sweeps. Existing scientific WfMSs (e.g. Kepler, Triana) usually do not distinguish between models and instances (except Wings [22], e.g.). When combining business and scientific workflows, this disadvantage can be discarded by the instance management and correlation mechanism of business WfMSs. One steering business workflow may compose many scientific workflows, which would support multi-scale or multi-physics simulations. (5) Fault handling is part of the business workflows and there are multiple best practices in terms of modeling tool and execution engine support. These can be utilized for scientific experiments by employing them into the supervising workflows—an enormous benefit for the scientists. In case of a failure fault handling behavior is triggered automatically and can carry out steps to retry erroneous activities, to undo their effects, or to achieve the desired goal another way. Other advantages of the business workflows are (6) rich auditing mechanisms that can be used as provenance information and (7) the persistent storage of execution data that yields a robust execution of workflows.

B. *Interactions With Scientific Applications*

The employment of a scientific workflow system shifts the interactions of scientists with a scientific application towards the scientific workflow system. That means the workflow system wraps the target applications by using their interfaces and provides its own interface to scientists. Scientific applications are no longer accessed directly by the user but indirectly through the scientific workflow. Similarly, when bringing business workflows into play, they are acting as a wrapper for the scientific WfMSs. Again, interactions of users with the overall system are shifted another layer up and are controlled and steered by the business workflow. This effect can simplify the work with the system from the user perspective: details about the interaction with scientific applications/WfMSs can be hidden (e.g. creation/copying of files/directories), new features can be introduced (e.g. automatic fault handling), and it is possible to orchestrate (and thus work with) many scientific applications or scientific workflows while dealing with a single business WfMS only.

Figure 3 illustrates this setup. There are three categories of interactions according to the three types of employed applications. The *scientific application* contains the domain logic of the respective scientific experiment/simulation. It implements formulae, solver, or visualization of (intermediate) results, i.e. it calculates and processes the relevant scientific

data. Main interactions with a scientific application are its installation, its invocation with the input data and the inspection of created results. Examples of the latter can be examining the final results after execution or investigating the convergence of results during execution. Depending on the concrete application it is also possible to infer a runtime status from the quantity of results, i.e. to monitor the application. This technique can be applied to scientific applications that produce intermediate results (e.g. for each time step of a calculation).

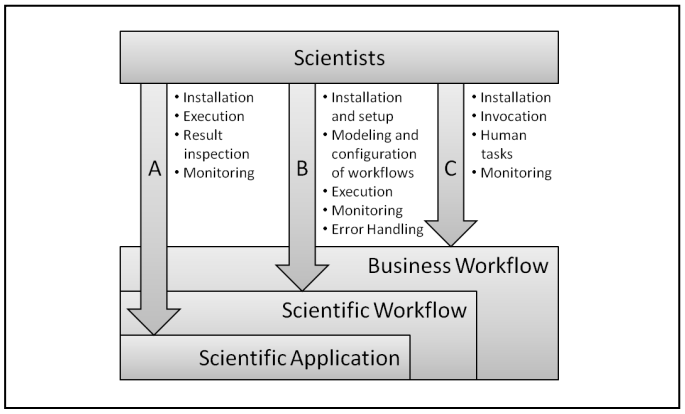

Figure 3. Categories of interactions between users and the scientific application

Scientific workflows steer the execution of a scientific calculation and can incorporate several applications, such as an FEM solver or a filter for images to detect shapes of objects. Scientific workflows typically deal with invocation of and data provision to scientific applications, collecting results, and transforming data. There are cases where scientific workflows even provision needed applications on the fly (usually in Grid environments) [23]. Installation and setup of the scientific workflow system can be seen as part of the user interaction with it. Additionally, workflows need to be modeled and configured (e.g. specifying the Grid resources to use). Configuration happens prior to execution and hence is independent of scientific results. Other interactions are triggering workflow execution, monitoring of the workflow (e.g. which step is currently carried out on which resource?) and manually correcting faults/problems.

Four main types of interaction can be observed between scientists and an employed *business workflow* system. First of all, the system needs to be installed. However, it is also possible to rely on an existing installation and to simply deploy the needed workflow on the engine. This setting may be the case in virtual environments like Grids or Clouds where a virtual machine with an installed workflow engine is already available. Additionally, the scientist can start the execution of the business workflow (that serves as a supervisor to the scientific workflow(s)). This is usually done by providing input or simply starting the workflow (which is essentially sending a message to the workflow system) which creates a new instance of the business workflow. Monitoring of the workflow status enables the user to see the current status of the overall simulation or experiment (i.e. the current phase in the life cycle of the scientific workflow). Monitoring of business workflows is typically realized in a graph-based fashion that reflects the control flow logic of the workflow. That way even inexperienced users can follow the progress easily. Finally, the scientists deal with work items issued by the business workflows. These HTs are used to incorporate human intelligence into the flow where an automatic processing is not possible or appropriate, e.g. the specification of parameters for the execution of scientific applications, the decision about convergence of results, or appropriate reactions on failures.

IV. Conceptual Architecture

So far, we have presented our theoretical consideration about combining business and scientific workflow technology in order to cover the full set of requirements of scientific applications. This section introduces the concept for a practical realization of the idea (Figure 4). A scientist usually unifies a domain specialist, programmer, and administrator in a single role [2]. That is why we foresee a single GUI as part of the overall architecture that assists scientists in the tasks to be conducted around a simulation. These tasks are installation and configuration of the target runtime environment (1), modeling scientific workflows (2), configuration of workflow models (3), specification of parameters (4) and input data (5) for single workflow runs, execution of workflows (6), visualization of result files (7), visualization of the runtime status of the experiment (8), making decisions about the proceedings of the experiment, for example based on the convergence of (intermediate) results (9), and deciding on the actions to handle faults (10). Note that the ingredients of the GUI are not components but rather actions of the scientist that are supported by the GUI. Many of these actions are HTs that require input data or decisions by a human being. The GUI needs to provide functionality to feed a workflow to the scientific workflow system. Although the GUI can assist the scientist in setting up the environment, the concept cannot relieve the scientist from all technical tasks. At least the GUI and workflow engine need to be installed or configured. But there are implementations where a pre-configured GUI and workflow engine with deployed workflows can be shipped in an easy-to-use installation bundle that only needs to be unpacked and started. This is by far simpler than setting up a scientific workflow system and additional Grid clients (e.g. Globus Toolkit).

The business workflow engine runs the processes coordinating the execution of scientific experiments and simulations. Several different coordination processes are needed. Each is tailored to an employed scientific workflow system because the systems have different interfaces, parameters, configuration options, etc. The processes coordinate the steps that need to be conducted to achieve the scientific goal. That means they prescribe the order and dependencies of actions, or the format of data, they automate tasks, and they catch and handle faults that occur during scientific workflow execution.

The employed scientific workflow system and scientific applications are leveraged to solve a more complex and holistic problem. That means the scientific applications solve partial issues of the overall problem and read and produce result files. The scientific workflows control the execution order of the applications and transfer the needed data to and from executables (i.e. they need reading access to result files). For

the coordination of scientific experiments by business processes it is helpful that a scientific workflow system provides an API that allows a business process to control it, e.g. with operations to load and start workflows and that signal failures during execution. However, if such an API is not offered, it is possible to integrate a scientific workflow system with an integration technique of any kind suitable for the available infrastructure, e.g. WSs or messaging adapters [24].

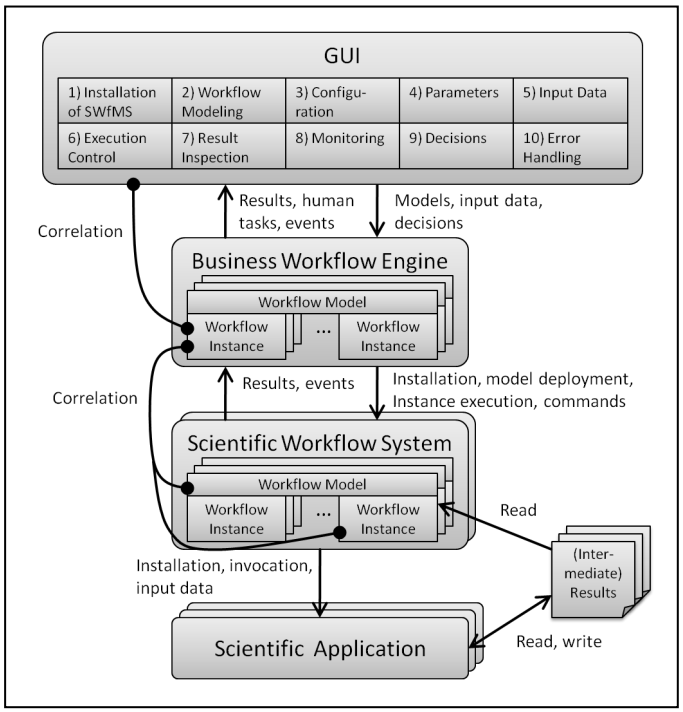

Figure 4. Conceptual architecture of the system

It is important to note that correct correlation between GUI, business workflows, scientific workflows, scientific applications, and result files is a must (Figure 4, correlation is shown by lines with round endings). The GUI can control several business workflows in parallel and hence needs to store correlation information in order to pass data to the correct instance. As mentioned earlier, the business workflow reflects the life cycle of scientific workflows. Each business workflow instance is therefore associated with one or more scientific workflow models (i.e. with a particular scientific problem), which reflects the modeling phase of the life cycle. Additionally, a business workflow instance can correspond to several workflow instances (i.e. experiment executions) in the execution and monitoring phase. Over the scientific workflow instances the business workflow instance is able to access information about an experiment run, such as execution time, errors, or results. The correlation between business and scientific workflows strongly depends on the employed scientific workflow system and its mechanism to uniquely identify workflow models and instances (to use the terms of conventional workflow technology). This might be by means of a model name and instance identifiers, or by names of files and directory paths, for example. A scientific workflow instance has to know where and how to invoke scientific applications and where the result files are stored. In some cases these applications may even be downloaded and installed on-the-fly. Since scientific workflows are usually long-running, the business and scientific workflow system may communicate asynchronously. This imposes the requirement on the scientific workflow adapter to store correlation tokens of the invoking business workflow in order to respond to the correct instance.

V. Coordinating Pegasus With BPEL Processes

We implemented the presented concept with two popular representatives for the domain of business and scientific workflow management, namely BPEL and Pegasus. The developed prototype proves the applicability of the concept in practice. However, the concept's generality allows an implementation with different technologies and systems, e.g. Kepler or Taverna. The main difference of Pegasus to the mentioned systems is that these are both a modeling tool and an execution engine rolled into one. Pegasus provides distinct tools for modeling and execution of scientific workflows. For a demonstration of the prototype consider [25]. The prototype can deal with arbitrary Pegasus workflows. Our tests are based on the black-diamond workflow of a Pegasus tutorial.

A. *Why BPEL?*

There are several reasons that make BPEL a good choice for the business workflow domain representative: (1) With the BPEL extension BPEL4People [13] and the specification WS-Human Task (WS-HT) [26] the integration of HTs and notifications into a workflow is enabled. Humans can interact with the workflow over a GUI, the HT client. In our scenario this will relieve scientists from typing commands in a Linux console as is required by Pegasus. Another possibility to implement HTs in BPEL is via usual WS calls. This approach suffers from the lack of features, e.g. a standard API for the HT client or the definition of user roles, but would be sufficient for our use case. (2) BPEL relies on WSs as activity implementation. That means the coordinating BPEL process can run on an arbitrary machine and communicate with the Pegasus (or Kepler, etc.) server over the network. Installation and configuration of the server can be carried out from remote sites. Even different scientific WfMS servers can be used for different scientific workflow runs. Additionally, the HT client used for the communication between scientists and the BPEL engine can be hosted on any machine in the network. This decreases the installation effort for a scientist to the HT client only. (3) Since BPEL also supports an asynchronous communication model, the number of exchanged messages in the system is minimal. No polling is needed to query the status of Pegasus. A scientific workflow is triggered with a one-way message and the BPEL process is then waiting for the response. Moreover, an asynchronous communication does not limit the runtime of an invoked operation through a timeout of an established session. This is especially useful because the triggered scientific workflows can run for a long time. Other advantages of BPEL are (4) the rich set of control flow structures such as the `if` or `repeat` activity, (5) its ability to handle faults on the workflow level by fault handlers, or (6) its transaction concept to carry out tasks in an all-or-nothing manner with the help of compensation handlers. Finally, (7) BPEL engines are usually shipped with a monitoring tool or

publish events that can be used to implement a monitor. Such a monitor is useful for scientists because it visualizes the current status of the workflow at a glance.

B. System Description

Figure 5 illustrates the architecture of our prototype that coordinates the Pegasus workflow system with the help of a BPEL process. The four parts that are distinguished by dotted lines can represent different machines. As these machine borders are only logical, the components can run on the same machine (which is not recommended due to memory needs). The Grid resources execute DAG jobs. These jobs are submitted to the sites via Condor. Pegasus, DAGMan and Condor are running on another machine (usually termed the *submit host*). Additionally, we need a WS wrapper that makes Pegasus' API accessible to BPEL. Although Pegasus' API is rich enough to trigger scientific workflow planning and execution, another WS is needed that offers operations on the file system, e.g. to read result files or to prepare the workflow execution directory. A BPEL engine executes the coordinating BPEL process. All communication between the GUI and the Pegasus site is routed through the engine. The running BPEL process instances "know" the current system or experiment state. The GUI is the component the scientists interact with. It mainly provides scientists with information about the system status and asks for input (see Section IV). Part of the GUI is a callback WS for an asynchronous communication with the BPEL process. It receives messages for new HTs or for the status of the process instance. Both GUI and BPEL engine hold their runtime status persistently in a database.

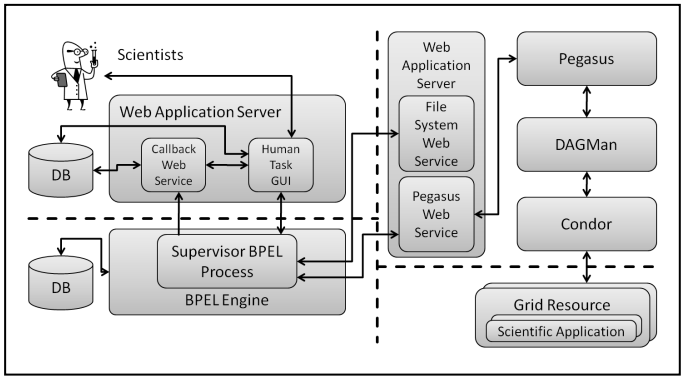

Figure 5. Main components of the system and their logical distribution among machines

C. BPEL process

Figure 6 illustrates a simplified view of the BPEL process used in our prototype. This "scientist's view" omits data transformation activities and activities of little importance for the scientist. The workflow is geared towards the two main phases of the Pegasus workflow life cycle, planning and execution. During planning Pegasus automatically transforms the DAX (an abstract workflow where tasks are not bound to executables and execution sites) to a concrete, executable DAG. This transformation is based on information provided by a user or prior workflow runs: the DAX, properties, parameters, a catalog of participating sites, of files, and of executables. Currently, the BPEL workflow supports the specification of the three catalogs by the scientist which is reflected by the three parallel HTs. After that the workflow invokes the Pegasus planning service. The result of the planning—either the created DAG or an error message in case of an unsuccessful planning—is passed to the GUI where the scientist can decide about running, re-planning, or aborting the workflow.

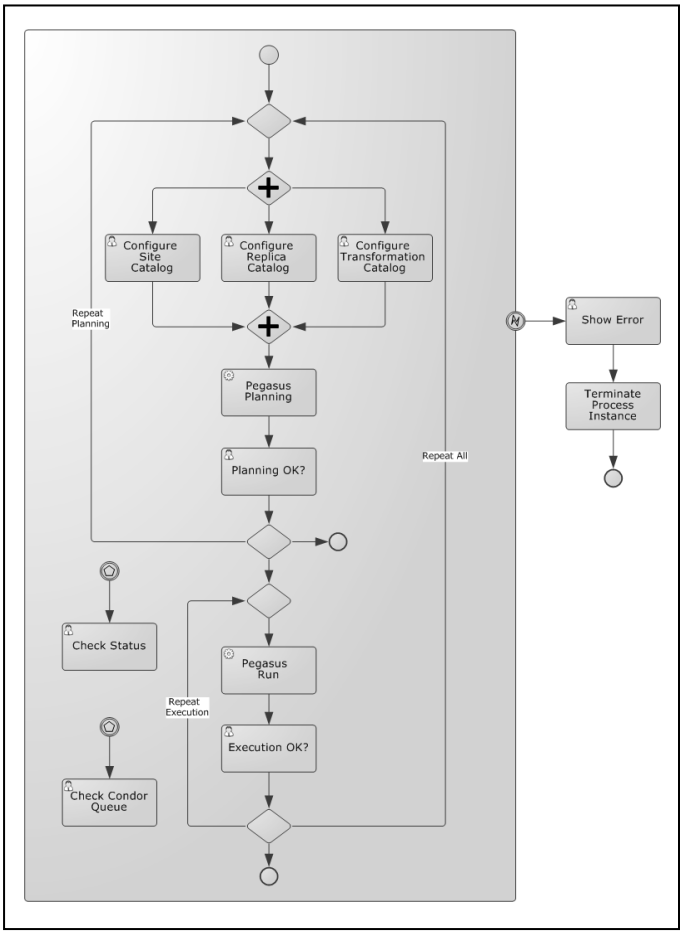

Figure 6. BPEL workflow for coordination of Pegasus planning and execution in BPMN notation

In order to execute a workflow Pegasus submits the DAG to DAGMan, which executes the jobs according to the workflow logic. Since the executed workflows may be long-running, we designed the BPEL workflow to invoke Pegasus asynchronously and to provide a callback operation for the result notification. After the workflow execution the scientist can decide by means of a HT to finish the experiment, to repeat execution, or to re-plan the workflow. The decision is based on the result files that were created. That means the scientist controls the convergence of the calculated results. For a repeated execution Pegasus submits the rescue DAG to DAGMan. This is Pegasus' built-in mechanism to handle runtime faults. With two different events the scientist can request further runtime information. "Check Condor Queue" delivers the content of condor execution queue. "Check Status" shows the status of the job that is currently executed. There is a global fault handler definition that catches all kinds of faults. It notifies the scientist about the fault. In the future, more

sophisticated fault handling logic can be added such as re-planning or re-executing the scientific workflow.

The current BPEL workflow requires a correct setup for the Pegasus server, workflow directory, and properties file as well as already staged input files. However, these and other tasks can be easily integrated into the workflow later on. This is actually one of the advantages of implementing the supervision logic as a workflow. Nevertheless, the current workflow is an example of how to combine business and scientific workflows and covers some of the interactions of scientists with scientific applications as shown in Figure 3: execution and result inspection of category A; configuration (in parts), execution, monitoring, error handling (in parts) of category B; and invocation, HTs, and monitoring of category C.

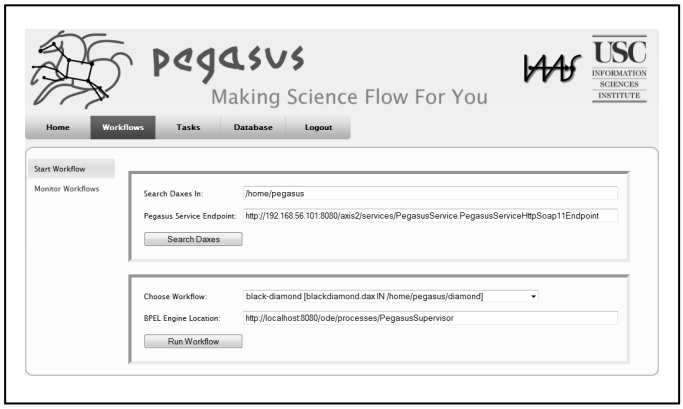

Figure 7. Start scientific workflow menu. Pegasus and BPEL engine location can be chosen.

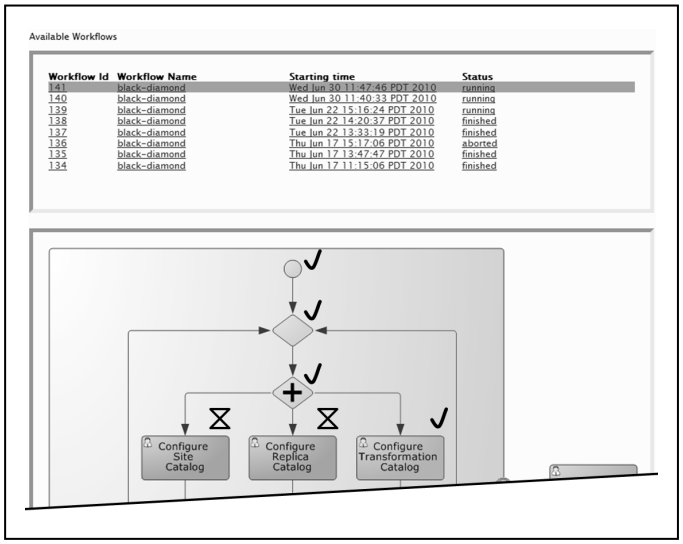

Figure 8. Monitoring view of workflows. Status of activities is signaled by different colors. For legibiliy of this gray-scale image icons where added to activities (no icon/gray = inactive, sand clock/blue = running, check/green = successful). The scientist can choose a workflow from the list above.

D. Implementation Details

We implemented the GUI as a Web application based on Java Server Pages and Servlets. That way it is possible to setup the GUI and the BPEL engine once on a server and use it from different desktops without installation of client software other than a browser. The GUI provides user management functionality so that it can be used by different users in parallel. A MySQL database (DB) is used as persistence layer to store user data, HTs, and workflow information. Apache Tomcat is taken as Web application server to host the GUI. The callback and Pegasus WS are implemented based on the WS engine Apache Axis2. For execution of BPEL processes we chose the Apache Orchestration Director Engine (ODE) 2.0, an open source BPEL engine. Communication between the HT client, BPEL engine, and Pegasus WS is conducted with SOAP messages over HTTP. The scientist can start a new experiment using the GUI. He/She can choose between different workflows that are already available on the Pegasus server (Figure 7). After starting a scientific workflow the monitoring view of the coordinating BPEL process is displayed. This monitoring enables scientists to see which tasks are running and whether user input is required (see Figure 8). By clicking on a task, the GUI opens it to be dealt with by the scientist (Figure 9). The GUI already takes care of the input that can be provided by the scientist, e.g. checking whether the execution site names in the site and transformation catalog coincide.

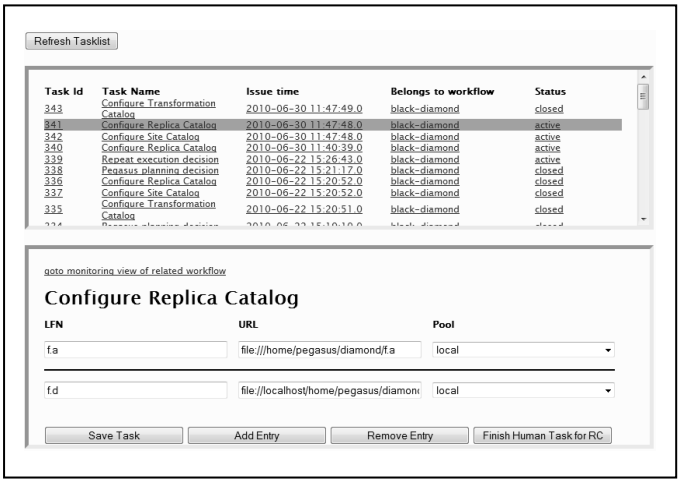

Figure 9. Human task view. The scientist can choose a task to work on from the list of assigned tasks above.

In Section IV we explained that a multi-level correlation mechanism is needed. The prototype implements unique identification of a BPEL workflow instance via the creation timestamp of a new instance and the name of the user that started the instance. Messages from the GUI to the BPEL engine have to carry these properties to be routed to the correct workflow instance. Correlation between BPEL workflow instance and the DAX is achieved via the DAX's directory and filename. The BPEL workflow instance addresses the DAG with the directory it is created in during planning. Although it may not be running yet, this DAG can be considered a workflow instance from the Pegasus perspective because (1) its directory is also used for storing runtime information, and (2) re-planning the DAX creates a new directory for the DAG. Realizing the correlation between BPEL and Pegasus workflows was straight-forward due to Pegasus' concept of unique directories for workflows. We realized Pegasus' run method as one-way operation in the WS wrapper. The invoking

BPEL engine does not have to keep an open HTTP session until the Pegasus workflow finishes. The WS wrapper instead listens on the Condor queue and sends a notification to the BPEL process if the queue is empty of particular workflow job ids (i.e. if the scientific workflow is finished).

E. Restrictions of the Prototype

Observing the convergence of results by the user is currently restricted to final results. It is desirable to enable users to inspect intermediary results and based on these decide about the proceedings of the experiment. Apache ODE does not implement the extension BPEL4People yet. We solved this problem by realizing the assignment of work items to users via WS invocations. The GUI's callback WS provides an appropriate operation. For our purposes this solution is sufficient. Additionally, we had to create a workaround for requesting Pegasus' status and Condor's queue status from the GUI. ODE's event handler does not run in version 2.0 so that we could not route these requests through the BPEL process. The GUI therefore communicates directly with the Pegasus server for these requests. Propagation of events of the BPEL workflow from the engine to the GUI is realized by WS calls within the workflow. This is a simple and working solution but should be substituted by native BPEL engine events in future in order to decouple the monitoring tool from this specific BPEL workflow. In total, the implemented prototype covers several of the GUI features as shown in Figure 4 like parts of aspects 3 and 6 through 10. The rest is open for future work.

VI. CONCLUSIONS

Business and scientific workflow systems were developed for completely different application areas and provide different functionality. Hence, they cover different requirements of scientists and scientific applications when being employed for scientific simulations and experiments. In many prior works the capabilities of business workflows to implement scientific applications were investigated. In this paper we followed a novel approach, namely the integration of business and scientific workflow systems. This approach is characterized by a good price performance ratio compared to the way of extending one of the technologies by features of the other. We presented a concept that places business on top of scientific workflows supervising the scientific workflow life cycle. In particular this introduces HTs, fault handling and transaction features to scientific applications and is therefore beneficial to the scientific community. A prototypical implementation with BPEL and Pegasus showed the feasibility of the concept in practice. But the approach also has weaknesses. It is strongly dependent on the functionality of the API of the employed scientific workflow system. Limited functionality could be extended with the help of appropriate adapters but in some cases requires a lot of additional effort. HTs are difficult to be integrated into the execution phase of the scientific workflows as they are typically not part of the scientific workflow. This again depends on the richness of the scientific workflow's API. In summary we can say that bringing business and scientific workflow technology together can create a benefit to scientists and scientific applications. It may prevent from re-implementing functionality of the other workflow domain.

REFERENCES

[1] M. Sonntag et al., "The Missing Features of Workflow Systems for Scientific Computations," Proceedings of the 3rd Grid Workflow Workshop (GWW), Paderborn, Germany, 2010.

[2] M. Sonntag and D. Karastoyanova, "Next Generation Interactive Scientific Experimenting Based On The Workflow Technology," 21st IASTED International Conference on Modelling and Simulation, Banff, Canada, July 2010.

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

[4] B. Wassermann et al., "Sedna: A BPEL-based environment for scientific workflow modeling," in: Workflows for e-Science: Scientific Workflows for Grids, I. Taylor et al., Eds. Springer, 2007.

[5] I. Wassink et al. , "Designing workflows on the fly using e-BioFlow," Int'l Conf. on Service Oriented Computing, Stockholm, Sweden, 2009.

[6] OASIS, "Web services business process execution language (BPEL) version 2.0," OASIS Standard, April 11th, 2007, http://docs.oasis-open.org/wsbpel/2.0/OS/wsbpel-v2.0-OS.html.

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

[8] Taylor et al.: "The Triana workflow environment: architecture and applications," in: Workflows for e-Science: Scientific Workflows for Grids, I. Taylor et al., Eds. Springer, 2007.

[9] Ludaescher et al, "Scientific workflow management and the Kepler system," Concurrency and Computation: Practice and Experience, vol. 18, 2006.

[10] et al., "Taverna: a tool for building and running workflows of services," Nucleic Acids Research, vol. 34, Web Server issue, 2006.

[11] Currle-Linde et al., "GriCoL: a language for scientific grids," Proc. 2nd IEEE Int'l Conf. on e-Science and Grid Computing, 2006

[12] Condor DAGMan, http://www.cs.wisc.edu/condor/dagman.

[13] A. Agrawal et al., "WS-BPEL extension for people (BPEL4People)," Version 1.0, 2007.

[14] OMG, "Business process modeling notation (BPMN) Version 2.0 Beta 2," OMG Document, Number dtc/2010-05-03, http://www.omg.org/spec/BPMN/2.0/Beta2/PDF/.

[15] IBM BPM Suite, http://www-01.ibm.com/software/info/bpm/.

[16] Oracle SOA Suite, http://www.oracle.com/us/technologies/soa/soa-suite/.

[17] W.M.P. v. d. Aalst et al., "Case handling: a new paradigm for business process support," Data and Knowledge Engineering, 53(2), 2005.

[18] W.M.P. van der Aalst et al., "Declarative workflows: balancing between flexibility and support," Computer Science – R&D 23(2), 2009.

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

[20] R. Barga and D.B. Gannon, "Scientific versus business workflows," in Workflows for e-Science: Scientific Workflows for Grids, I. Taylor, E. Deelman, D.B. Gannon, M. Shields, Eds. Springer, 2007.

[21] B. Ludaescher et al., "Scientific workflows: business as usual?," 7th Int'l Conf. on Business Process Management (BPM), 2009.

[22] Y. Gil et al., "Assisting scientists with complex data analysis tasks through semantic workflows," AAAI Fall Symposium Series on Proactive Assistant Agents, Arlington, VA, November 2010.

[23] G. Juve and E. Deelman, "Resource provisioning options for large-scale scientific workflows," 3rd Int'l Workshop on Scientific Workflows and Business Workflow Standards in e-Science, Indianapolis, USA, 2008.

[24] G. Hohpe and B. Woolf, "Enterprise integration patterns: designing, building, and deploying messaging solutions," Addison-Wesley, 2003.

[25] M. Sonntag et al., "BPEL4Pegasus: Combining Business and Scientific Workflows," Prototype demo, http://www.iaas.uni-stuttgart.de/institut/mitarbeiter/sonntag/indexE.php

[26] A. Agrawal et al., "Web services human task (WS-HumanTask)," Version 1.0, 2007.