USING THE BUSINESS PROCESS EXECUTION LANGUAGE FOR MANAGING SCIENTIFIC PROCESSES

Anna Malinova, Snezhana Gocheva-Ilieva

Abstract: This paper describes the use of the Business Process Execution Language for Web Services (BPEL4WS/BPEL) for managing scientific workflows. This work is result of our attempt to adopt Service Oriented Architecture in order to perform Web services – based simulation of metal vapor lasers. Scientific workflows can be more demanding in their requirements than business processes. In the context of addressing these requirements, the features of the BPEL4WS specification are discussed, which is widely regarded as the de-facto standard for orchestrating Web services for business workflows. A typical use case of calculation the electric field potential and intensity distributions is discussed as an example of building a BPEL process to perform distributed simulation constructed by loosely-coupled services.

Keywords: BPEL, scientific workflows, Web services, SOA.

ACM Classification Keywords: D.2.12 Interoperability - Distributed objects

Introduction

There is growing interest in the use of Web services infrastructures for scientific computing. Web services provide interoperability of various applications running on heterogeneous platforms. They enable dynamic connections and automation of business processes within and across scientific collaborations for application integration, reusability, and flexibility which is motivated mainly by the loosely coupled nature of the Web services. Web services became the preferred way to realize Service Oriented Architectures (SOA). SOA is architecture that represents software functionality as discoverable services on the network. All program functions and methods are exposed as services described by some universal description language (WSDL [1] in the case of Web services-based SOA). These interfaces can be invoked by other services to perform business processes.

The Business Process Execution Language (BPEL) [2] is an XML-based language used for integration of a number of Web services into more complex composite services. Thus BPEL enables the top-down realization of Service Oriented Architecture through composition, orchestration, and coordination of Web services. The BPEL composite services are called business processes and are managed by a workflow engine. BPEL processes orchestrate the interactions between the Web services using standard XML (SOAP) messages for communication. BPEL processes can be executed on any platform or product that compiles with the BPEL specification. BPEL supports the Web services technology stack, including SOAP, WSDL, UDDI, WS-Reliable messaging, WS-Coordination, and WS-Transaction.

This paper describes the use of BPEL for managing scientific workflows. The complex, unpredictable, and inter-dependent nature of the components in a scientific workflow leads to such requirements, concerning workflow language, as exception handling, recovery from uncertain situations, user interactions to facilitate interactive steering and monitoring, flexibility to support dynamic selection of services at runtime, etc. In the context of addressing these requirements, the BPEL specification features are discussed in the following sections.

We use BPEL to orchestrate Web services in order to perform simulation of metal vapor lasers. This includes providing of Web services wrappers of legacy scientific applications. We begin our workflow solution by defining a number of inter-related patterns that match the basic requirements of the users of the system. A typical use case of the calculation the electric field potential and intensity distributions is also provided as an example of building a BPEL process to perform distributed simulation constructed by loosely-coupled services. The BPEL processes

we build are realized with the Oracle BPEL Process Manager and Designer and are deployed to the Oracle Application Server.

Scientific Workflows and BPEL4WS

The spectrum of what might be called scientific workflow is wide and includes scientific discovery workflows, workflows that automate manual procedures or reengineer custom tools, and data and compute-intensive workflows. Scientific workflow support is needed for practically all information-oriented scientific disciplines, including bioinformatics, chemistry, ecology, geology, physics, etc. In this section we provide a number of common requirements of scientific workflows: service composition and reuse, scalability, detached execution, reliability and fault tolerance, user interaction, monitoring, "smart" re-runs, data provenance, etc. ([5], [6], [7]).

In general, the BPEL vocabulary is tailored more to the requirements of business processes, which often have different requirements compared to scientific workflows. For example, in [5] is outlined that business workflow approaches focus on control-flow patterns and events, whereas dataflow is often a secondary issue. Scientific workflow systems, on the other hand tend to have execution model that are much more dataflow-oriented.

In this section we provide an analysis of the BPEL specification in the context of the above listed requirements. We do this also in the context of the implementation technology we have adopted, particularly the Oracle's BPEL Process Manager as part of the Oracle SOA Suite.

Service composition and reuse.

Web services can be combined in two ways: orchestration and choreography. In orchestration a central process (which can be another Web service) takes control of the involved Web services and coordinates the execution of different operations. The involved Web services do not "know" (and do not need to know) that they are taking part in a composition process. Choreography, in contrast, does not rely on central coordinator and occurs in pear-to-pear style workflows where communications occur directly between partners. All participants in the choreography need to be aware of the business process, operations to execute, messages to exchange, and the timing of message exchanges.

BPEL supports two different ways of describing business processes that support orchestration and choreography: Executable processes - they follow the orchestration paradigm and can be executed by an orchestration engine; Abstract business protocols - they allow specification of the public message exchange between parties only. They do not include the internal details of process flows and are not executable. They follow the choreography paradigm.

From the perspective of composing Web services to execute scientific processes, orchestration is a more flexible paradigm and has the following advantages over choreography:

- The coordination of component process is centrally managed by a known coordinator.
- Web services can be incorporated without their being aware that they are taking part of a larger process.
- Alternative scenarios can be put in place in case faults occur.

Scalability.

Some scientific workflows involve large volumes of data and/or require high-end computational resources, e.g. running a large number of parallel jobs on a cluster computer. To support such data-intensive and compute-intensive workflows, suitable interfaces to Grid middleware components are necessary.

Parallel flows enable a BPEL process to perform multiple tasks at the same time, which is useful when we need to perform several time-consuming and independent tasks. Concurrency is provided with the <flow> activity which causes all the activities nested within it to be executed concurrently. Control exits from <flow> when all nested activities terminate.

BPEL also provides features to support handling multiple requests. Multiple customer interactions can be handled concurrently by creating multiple instances of the process, one for each interaction. This is not a problem if the

interaction consists of a single, synchronous invocation of an operation on a server, and the server does not invoke other servers in the process of handling the request. Scientific processes, however, often require long running conversations and transactions between partners. For workflows, this involves the addition of process identifiers that are embedded and exchanged between partners during a conversation. For handling such situations, BPEL allows a process to declare a *correlation set* local to a scope. This is a set of properties such that all messages having the same values of all the properties in the set are part of the same interaction and hence are handled by the same instance. Thus, a correlation set identifies a particular instance of among a set of instances of that process, and a correlation set and a port together uniquely identify a process instance among all process instances at a host machine.

Detached execution.

Long running scientific workflows require an execution node that allows the workflow control engine to run in the background on a remote server, without necessarily staying connected to a user's client application that has started and is controlling workflow execution.

In a BPEL process a web service can be invoked as a synchronous or asynchronous operation. Synchronous web services provide an immediate response to a query, and block the BPEL process for the duration of the operation. Asynchronous web services do not block the BPEL process, and are useful for environments in which a service can take a long time to process a client request. Asynchronous services also provide a more fault-tolerant and scalable architecture than synchronous services.

Reliability and fault tolerance.

A scientific workflow might incorporate a service that often "fail", change its interface, or just become unacceptably slow. Thus the workflow definition should support the definition of failure handling mechanisms.

BPEL provides a flexible structure for dealing with failures. Fault and compensation handlers are used to reverse the effects of partially completed interactions. The execution of these handlers is tied in with the concept of scopes. A scope serves to define the execution context of an activity. A scope can have a name and local declarations and encloses (possibly complex) activity to be executed. Declarations include, among other items, local variables, fault handlers, and a compensation handler. Compensation applies only to scope's external effects – the effects it has invoked at other sites. On entry, a scope's compensation handler is given a snapshot of the process's state at the time control exited (normally) from the scope. Since it can access only the snapshot and not the variables themselves, compensation cannot affect the state of the process and applies only to external activities. Faults signal failure and start the process of reversing the effects of an interaction. A fault might be raised in a process if it gets a fault response to an operation that it has invoked synchronously. Alternatively, a process might explicitly execute a <throw> activity if it recognizes that an anomalous situation has arisen.

User interaction.

Many scientific workflows require user decisions and interactions at various steps. An interesting challenge is the need for user interaction in a detached execution. Using a notification mechanism the user might be asked to reconnect to the running instance and make a decision before the paused (sub-) workflow can resume.

BPEL 1.1 and 2.0 do not include human interactions and are limited to service orchestration. Oracle BPEL Process Manager provides manual task Web service to integrate people and manual tasks into BPEL processes [8]. By implementing this as a true BPEL service, the interface to the task service is described with WSDL and people can be included in 100% standard BPEL processes – to the BPEL process, the person/manual task looks like any other asynchronous Web service. User notification is also supported – anything that can be done in BPEL (invoking a Web service, sending an e-mail message or JMS message, executing some Java code, and so on) can be done to notify a user of a task-related event.

Monitoring.

Scientific workflows are potentially long running activities and it is of importance to scientists to be able to observe and monitor the ongoing execution of a workflow.

Oracle BPEL Manager provides sensors to monitor BPEL activities, variables, and faults during runtime [8]. The following types of sensors can be defined, either through the BPEL Designer or manually by providing sensor configuration files:

- Activity sensors: Used to monitor the execution of activities within a BPEL process. For example, activity sensors can be used to monitor the execution time of an invoke activity or how long it takes to complete a scope. Along with the activity sensor, the variables of the activity might be also monitored.
- Variable sensors: Used to monitor variables (or parts of a variable) of a BPEL process. For example, variable sensors can be used to monitor the input and output data of a BPEL process.
- Fault sensors: Used to monitor BPEL faults.

The following two requirements are much desirable for scientific workflow systems, although difficult to implement:

"Smart" re-computations.

A special kind of user interaction is the change of a parameter of a workflow. A "smart" re-computation (re-run) would not execute the workflow from scratch, but only those parts that are affected by a parameter change. Another useful technique in this context is the ability to backtrack (in the case or parameter change or even a system failure) to a previously saved state without starting over from scratch.

Data provenance.

Computational experiments and runs of scientific workflows should be reproducible and indicate which specific data products and tools have been used to create a derived data product. A scientific workflow system should be able to automatically log the sequence of applied steps, parameter settings and intermediate data products. A related requirement is automatic report generation: The system should allow the user to generate reports with all relevant provenance and runtime information, e.g., in XML format for archival and exchange purposes, and in HTML (generated from the former, e.g., via a XSLT script) for human consumption.

While the above list of requirements for scientific workflow systems is by no means complete, it should be sufficient to capture many of the core characteristics. Other requirements include the use of an intuitive GUI to allow the user to compose a workflow visually from smaller components to animate workflow execution, to inspect intermediate results, etc., although these requirements are not related with the BPEL specification itself. Rather, they depend on the BPEL Designer and engine vendor.

Basic Patterns of the Workflow Solution

In this section we define a number of inter-related patterns that match the basic requirements of the users of the Web services-based system for simulation of metal vapor lasers we are in process of developing. Our approach is to decompose the simulation processes into a collection of basic patterns that can be fully automated. These are hierarchically organized, as they are co-dependant, as presented in Figure 1:

- Simulation execution and monitoring pattern: This is the most basic workflow pattern. It presents the ability to submit a simulation, monitor its execution, and handle any potential faults (which may include resubmitting the job if needed).
- Data retrieval and storage pattern: Adds to the above the capability of retrieving data from the data storage before job submission and uploading results to the storage upon completion.
- Metadata management: Generated metadata for the created file, and uploading these to the metadata storage.

This hierarchical organization maps well to BPEL-base workflows. Because these are themselves presented as Web services, it is possible for a workflow to incorporate other workflows into its structure. Not all capabilities presented, such as metadata capture, may be required for every job of the scientific process that the user wishes

to execute. This hierarchical approach also favors reusability and enables the potential addition of new services and its incorporation into larger workflows.

A data/metadata management tools will be integrated with the workflow tools using web services technology. This integration is essential to allow the automatic collection and publishing of metadata relating to the simulations along with efficiently handling the data files themselves.

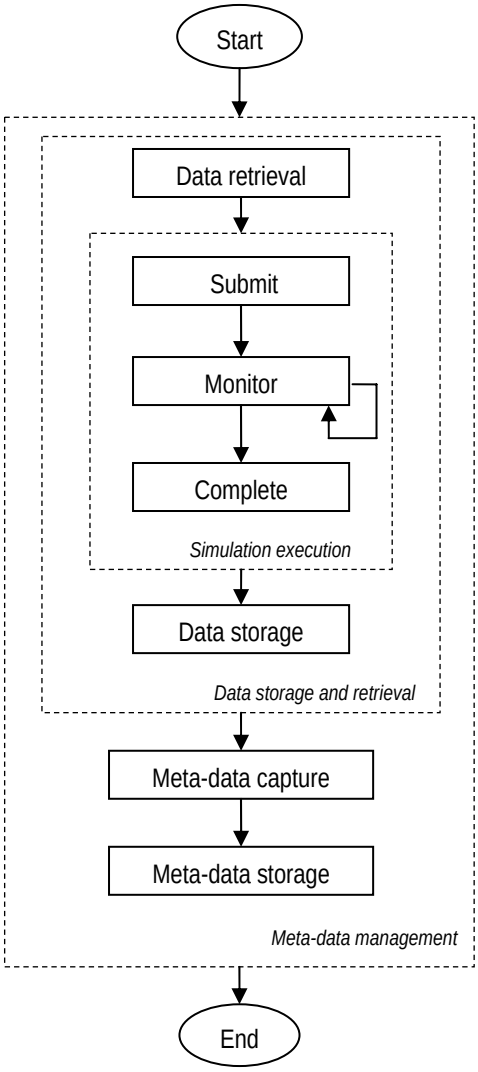

Figure 1. Basic simulation workflow patterns.

A Typical Use Case

In this section we describe an example workflow for calculation of the electric field potential and intensity distributions of the radio-frequency discharge in He-Cd laser. The obtained results are necessary for further investigation of the dependence between different laser characteristics, oriented for practical engineering purposes in order to improve the total efficiency of the real He-Cd laser.

We have wrapped legacy FORTRAN codes as Java modules trough the use of the Java Native Interface [3]. Then we have transformed these modules into Web services. These are:

- "Electrode" Web service – for a predefined geometry of the 2D cross-section of the laser design this module generates the appropriate mesh for discretization and classifies the points in the different sub-

regions: outer electrodes, laser tube, etc. Basic input parameter is the applied to the electrodes voltage, which is then used to determine the boundary conditions.

- "Poisson" Web service – This module provides numerical solution of the Poisson equation with which we model the potential equation.

The BPEL process first invokes a synchronous "GetData" service to retrieve data from the data storage. The data retrieved is a number of parameters predefined for the chosen laser type. These parameters are stored natively in XML files. An XML schema was created to describe these parameter sets. It consists of two types: *parameterset* and *parameter*. The type *parameter* is an abstract type which all parameters extend. Thus we can manage different parameter sets relevant to different laser types, as well to have different parameter set instances of a particular laser type. We have also created an Oracle's BPEL Manager Human Workflow Task service to incorporate a user task in the BPEL process and particularly the parameter approval. This enables the user to change some of the predefined values of physical constants, geometrical parameters, etc.

Then the "Electrode" service is invoked synchronously. The service creates output file which is next used as input for the "Poisson" service to calculate the electric field intensity and to save the output in a file.

Conclusion

From our work so far we conclude that BPEL is fully applicable for orchestrating scientific services. Furthermore BPEL could serve as a standard representation for scientific workflows and hence aid reproducibility. In addition the Oracle BPEL Process Manager, which we use, has built in support for the use of Web Services Invocation Framework (WSIF) [4]. Thus a direct Java code injection into a workflow script is enabled. This is a possible way to overcome some of the limitations of BPEL specification.

Acknowledgments

This work is supported by the National Science Fund, Project VU-MI-205/2006.

Bibliography

[1] Web Services Description Language (WSDL). http://www.w3c.org/TR/wsdl

[2] Business Process Execution Language (BPEL). http://www-128.ibm.com/developerworks/library/specification/ws-bpel/

[3] Java Native Interface (JNI). http://java.sun.com/j2se/1.5.0/docs/guide/jni/index.html

[4] Web Services Invocation Framework (WSIF). http://ws.apache.org/wsif

[5] B.Ludascher, I.Altintas, C.Berrkley, D.Higgins, E.Jaeger, M.Jones, E.Lee, J.Tao Y.Zhao. Scientific workflow management and the Kepler system. In: Concurrency and Computation: Practice & Experience.Vol. 18, 2006., p. 1039 - 1065.

[6] A.Akram, D.Meredith, R.Allan. Application of Business Process Execution Language to scientific workflows. In: International Transactions on Systems Science and Applications (accepted 2006).

[7] N.Joncheere, W.Vanderperren, R.Straeten, Requirements for a Workflow System for Grid Service Composition. In:Proceedings of the 2nd International Workshop on Grid and Peer-to-Peer Based Workflows (GPWW 2006), Vienna, Austria, September 2006. LNCS Springer-Verlag.

[8] D.Bradshaw, M.Kennedy. Oracle® BPEL Process Manager Developer's Guide10g (10.1.3.1.0). http://download-east.oracle.com/docs/cd/B31017_01/integrate.1013/b28981/toc.htm

Authors' Information

Anna Malinova – University of Plovdiv "Paisii Hilendarski", 24 Tzar Asen St., Plovdiv-4000, Bulgaria; e-mail: malinova@pu.acad.bg

Snezhana Gocheva-Ilieva – University of Plovdiv "Paisii Hilendarski", 24 Tzar Asen St., Plovdiv-4000, Bulgaria; e-mail: snow@pu.acad.bg