Available online at www.sciencedirect.com

Procedia Computer Science 1 (2010) 1513–1520

Procedia Computer Science

www.elsevier.com/locate/procedia

International Conference on Computational Science, ICCS 2010

Towards a model-driven transformation framework for scientific workflows

G. Scherp[a,*], W. Hasselbring[b]

[a]OFFIS Institute for Information Technology, Escherweg 2, D-26121 Oldenburg, Germany
[b]University of Kiel, Department of Computer Science, D-24118 Kiel, Germany

Abstract

Scientific workflows evolved to a useful means in computational science in order to model and execute complex data processing tasks on distributed infrastructures such as Grids. Many workflow languages and corresponding workflow engines and tools were developed to model and execute scientific workflows, without using established workflow technologies from the business domain. With the adoption of the service-oriented architecture (SOA) approach in modern Grid infrastructures, standardized and well-adopted workflow technologies from the business domain such as WS-BPEL are technically applicable to execute scientific workflows, too. In order to integrate business workflow technologies into the scientific domain, existing scientific workflow technologies for domain-specific modeling and established business workflows technologies for technical execution of scientific workflows can be combined. To do so, we propose an architecture for a transformation framework based on model-driven technologies that transforms a scientific workflow description at the domain-specific level to an executable workflow at the technical level.

Keywords: scientific workflows, Grid, WS-BPEL, model-driven technologies

1. Introduction

Many scientific domains as biology, chemistry or physics have an increasing demand to utilize computational power for their purposes, which is also called computational science (or scientific computing). Computational science creates a new field of science beside constructing theories and executing lab experiments, and deals with the construction, examination and optimization of mathematical models based on numerical analysis and simulation techniques. In practice, for example, data from lab experiments are collected and analyzed, or whole lab experiments are executed as simulations, which are both used to gain (scientific) knowledge from interpretation of the output data. Tools to support scientists in computational science are also referred to as problem-solving environments. Since knowledge acquisition and problem solving often consist of several computational steps, workflow technologies can be used to describe the execution order of these computational steps (or workflow activities). Such a workflow can be executed with a workflow engine. Workflows used in the scientific domain are also called scientific workflows.

*Corresponding author
Email addresses: guido.scherp@offis.de (G. Scherp), wha@informatik.uni-oldenburg.de (W. Hasselbring)

Currently, no widely accepted definition for scientific workflows is available analogous to the workflow definition, for a better distinction to scientific workflows we use the term business workflow, of the workflow management coalition (WfMC) [1]. In [2] a short description is given that clearly points two major goals of scientific workflows.

> “The main goals of scientific workflows, then, are (i) to save 'human cycles' by enabling scientists to focus on domain-specific (science) aspects of their work, rather than dealing with complex data management and software issues; and (ii) to save machine cycles by optimizing workflow execution on available resources.”

This description motivates an approach to distinguish different levels of abstraction to represent a scientific workflow [3], in this case a domain-specific and a technical level: The domain-specific level is an abstract and domain-specific view of a scientific workflow. It is intended to be used by scientists in order to ease the modeling of scientific workflows according to their scientific domain while abstracting from technical details of their execution (goal one). A workflow language representing the domain-specific level is usually not used for workflow execution. The technical level is a concrete and technical view of scientific workflows and used for their (optimized) execution (goal two). That means a scientific workflow is described with a workflow language that can be executed by a corresponding workflow engine, and all workflow activities including required data sets are mapped to appropriate resources (cluster resources, data storages) before or during workflow execution.

The separation of a domain-specific and a technical level is already well-established in the business domain. For example, the Business Process Modeling Notation (BPMN) [4] can be used to model a business workflow at the domain-specific level and WS-BPEL [5] at the technical level. In [6, 7] the generation of WS-BPEL code from BPMN is described. Thus, these two level of abstraction are also separated by different applied technologies. Another reason is a clear separation of concerns and roles within a company. A business workflow is usually modeled by business specialists (role of management department) at the domain-specific level and implemented by IT specialists (role of IT department) at the technical level. Each role needs an appropriate view of the business workflow (domain-specific or technical) that result in the use of appropriate workflow languages and tools for each level.

A clear separation of a domain-specific and a technical level is usually missing in scientific workflow technologies. Typically, scientific workflows regards the scientist as a single role that models, execute and monitors a scientific workflow execution. As the scientist uses the domain-specific level, the technical level is hidden from the user and the executable workflow has to be automatically generated. Thus, the separation of a domain-specific and a technical level in scientific workflows is not motivated by the separation of concerns regarding different roles analogous to business workflows. But the separation of appropriate workflow technologies for each level is beneficial for scientific workflows, too.

Many workflow languages and corresponding workflow engines and tools were developed for scientific workflows, whereas existing and established workflow technologies from the business domain were not used. One reason for this parallel development is that business workflow technologies have a different application focus (business processes) and could not fit the requirements for scientific workflows (scientific experiments). However, the introduction of service-oriented architectures (SOA) in the business domain and its later adoption in the scientific domain has reduced the gap between scientific and business workflows at the technical level. Grid infrastructures [8] have evolved to service-oriented architectures (SOA) [8] in which each access to resources are encapsulated by services. Standards as the Web Services Resource Framework (WSRF) [9] were developed and implemented by Grid middlewares such as Globus Toolkit[1] or UNICORE 6[2] in order to create service-based Grid infrastructures. WSRF enables stateless Web services to have a state, utilizing existing Web services standards as WS-Addressing[3] and WS-Notification[4]. As each workflow activity is executed by an appropriate service, scientific and business workflows executed in a SOA can both be called (Web) service orchestrations. The XML-based Web Services Business Process Execution Language (WS-BPEL or BPEL) [5] is an OASIS standard to describe such Web service orchestrations and is well-established in the business domain. Many open source and commercial workflow engines are available to execute so called WS-BPEL processes.

[1]http://www.globus.org/toolkit
[2]http://www.unicore.eu/
[3]http://www.w3.org/Submission/ws-addressing/
[4]http://docs.oasis-open.org/wsn/

Our approach is to use well-adopted business workflow technologies such as WS-BPEL for the execution of scientific workflows, focusing on SOA-based Grids as execution infrastructure. To hide the complexity of a workflow language like WS-BPEL we intend to generate the corresponding executable workflow source code from a domain-specific scientific workflow model, see Figure 1. The domain-specific workflow model is created by a scientist using a repository of available data and services without dealing with technical details. The repository contains technical descriptions which are hidden from the user. Based on these descriptions, needed input data and services can be mapped to appropriate resources during the transformation process or workflow execution. Therefore, we implement a transformation framework based on model-driven technologies that transforms a scientific workflow description at the domain-specific level to an executable workflow at the technical level. The design of the transformation framework allows the extension or modification of the transformation process. So, any workflow language can be supported in principle, but we focus on using existing scientific workflow languages and tools for the domain-specific level and well-established business workflow languages and engines, in our case WS-BPEL, for the technical level. We do not intend to create a new scientific workflow language.

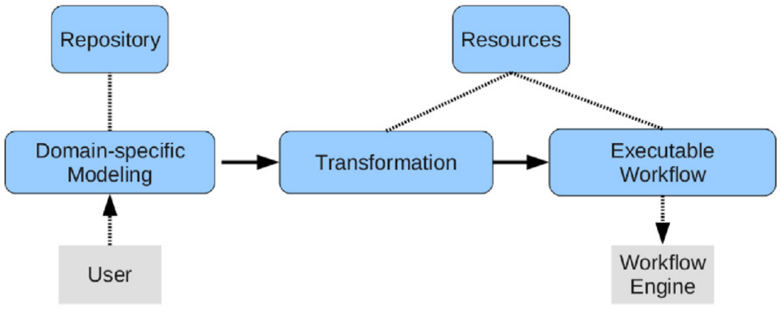

Figure 1: General approach

We expect that future scientific workflow technologies will increasingly utilize SOA-based Grid infrastructures. Our approach intends to enable existing scientific workflow technologies, that currently do not or rudimentary support SOA-based Grid infrastructures, to profit from using standardized and well-adopted business workflow languages as WS-BPEL and corresponding mature workflow engines for scientific workflow execution at the technical layer. The modeling of a scientific workflow at the domain-specific layer is untouched.

2. Transformation Framework

The transformation framework generates an executable workflow at the technical layer from a scientific workflow description at the domain-specific layer. The architecture of the transformation framework allows to support arbitrary workflow languages and to modify the transformation process. Finally, the following main goals are associated with the transformation framework.

- Combination of existing scientific workflow technologies for scientific workflow modeling (domain-specific layer) and existing business workflow technologies for scientific workflow execution (technical layer).
- Extensibility with further workflow languages at both level.
- Generation of an executable scientific workflow based on a well-adopted business workflow language such as WS-BPEL. All constructs needed for scientific workflow execution, which are often complex and error-prone in manual implementation, are generated automatically based on predefined transformation rules. Beside the generation of constructs representing scientific workflow activities of the domain-specific layer, the transformation framework also generates constructs as error handling that are usually not represented at the domain-specific layer.

- Utilization of well-established technologies from model-driven engineering.

The transformation framework consists of the following components and activities, see Figure 2.

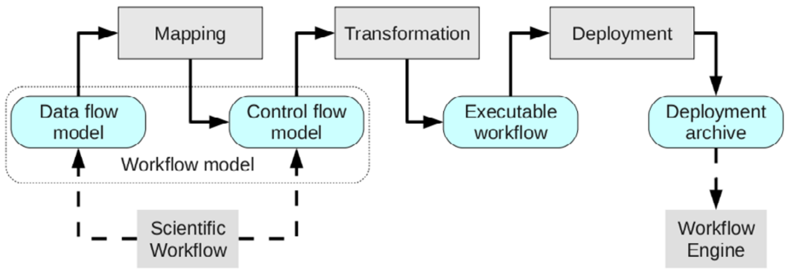

Figure 2: Architecture of the transformation framework

- *Workflow model*: The workflow model consist of a data flow-oriented model and a control flow-oriented model of the scientific workflow. It is solely used as an internal representation. As scientific workflows in the context of computational science are usually data-centric, the typically representation of a scientific workflow is data flow-oriented [10], while Business Workflow by contrast are control flow-oriented. But usually both workflow types may contain elements from the counterpart representation. In a first step all data flow and control flow information from the scientific workflow are mapped to the (internal) workflow model. Which restrictions we need for data and control flow (for example based on workflow patterns[5] is ongoing work.
- *Mapping*: This steps maps all information from the data flow-oriented model that are not regarded in the control flow-oriented model to corresponding activities in the control flow-oriented model.
- *Transformation*: This step generates an executable workflow from the control flow-oriented model based on a control flow-oriented workflow language such as WS-BPEL.
- *Executable workflow*: The executable representation of a scientific workflow based on a workflow language such as WS-BPEL.
- *Deployment adapter*: This step creates a deployment archive for the executable scientific workflow.
- *Deployment archive*: An archive to deploy the executable workflow to a vendor-specific workflow engine.

One issue for the mapping step is that many options exists to map data flow to a control flow representation. We illustrate this issue with an easy example, see Figure 3. A scientific workflow consists of one processor (P1) which needs input data from two data sources (DSo1 and DSo2). Afterwards, the output data is stored in one data sink (DSi1). We assume that data transfer services exists to transport data from a data source to a processor's location or from a processor's location to a data sink. Depending on the semantic of the data flow-oriented model the data transfers from DSo1 and DSo2 to P1 can be executed either in parallel or sequential. This interpretation of the data flow will be configurable in the mapping step. The order of data transfer executions may also depend on performance issues, which is currently out of scope in our work.

The transformation steps itself consists of two sub-steps. In the first sub-step the control flow-model is extended by typical aspects for the execution of a scientific workflow. These extensions concerns general aspects as error handling and monitoring as well as domain-specific aspects as job submissions and data transfers. For example, the activity for job submission (one processor at the domain-specific level) is expanded to the following typical sequence of activities to use a job submission service, see also Step 1 in Figure 4 and [11]. For simplicity monitoring, error handling, etc. is omitted.

[5]http://www.workflowpatterns.com/

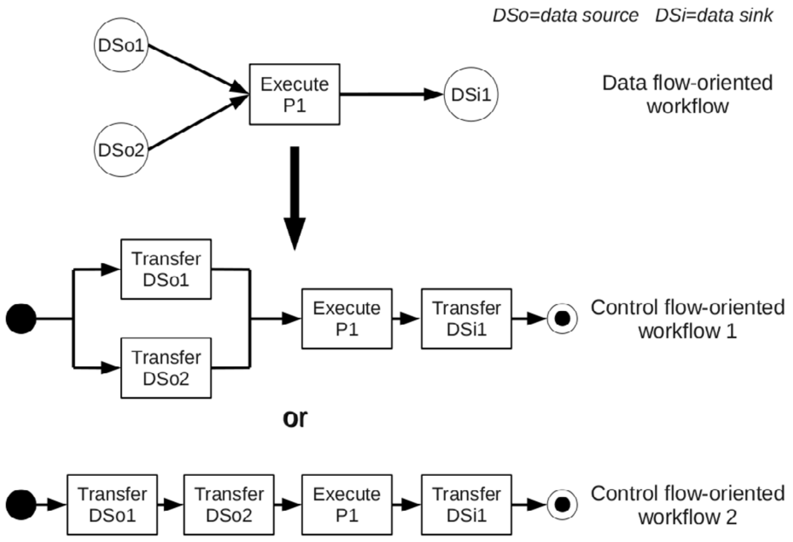

Figure 3: Example for possible mapping from data flow to control flow

1. *Prepare*: The job submission is prepared, e. g. a job submission description is created.
2. *Submit*: The job submission service is called.
3. *Wait*: Wait for the termination of job execution.
4. *Cleanup*: Perform some cleanup activities after job execution.

If no further transformation rules can be applied, the first sub-step is finished. In the second sub-step the extended control flow-oriented model is transformed to the target workflow language. That means, for each activity a corresponding code fragment is generated according to the executing infrastructure. The *wait* activity, for example, can transformed to either a pull model (periodically fetch job submission status; UNICORE 6) or push model (subscribe to a notification service and wait for status information; Globus Toolkit 4), see Step 2 in Figure 4. Regarding this sub-step we already developed code templates for WS-BPEL to invoke WSRF-based Web Services in Globus Toolkit 4 and UNICORE 6 [12, 11].

3. Related Work

There exist many scientific workflow technologies such as GWES[6], Kepler[7], Taverna[8], Triana[9], and Pegasus[10] for example, which are already successfully applied in the scientific domain. We do not intend to fully replace such technologies, but we rather want to offer a possibility to use well-adopted business workflow technologies such as WS-BPEL and corresponding mature workflow engines for the execution of scientific workflows in SOA-based Grid infrastructures. The other way round, we want to benefit from existing scientific workflow technologies to create a scientific workflow model at the domain-specific level.

[6]http://www.gridworkflow.org/gwes/
[7]https://kepler-project.org/
[8]http://www.taverna.org.uk/
[9]http://www.trianacode.org/
[10]http://pegasus.isi.edu/

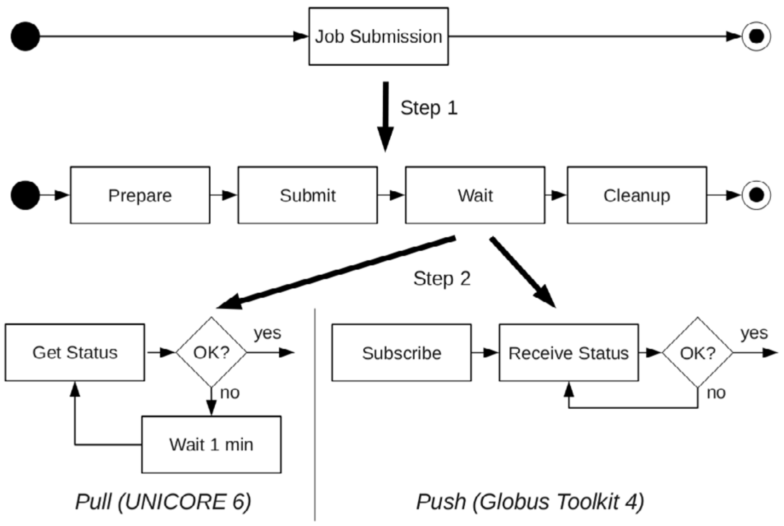

Figure 4: Two sub-steps of the transformation step

The general suitability to use WS-BPEL for the execution scientific workflows is shown in several publications [13, 14, 15, 16, 17, 18, 19, 20, 21]. As the invocation of Grid middleware services is very complex, some approaches [13, 17, 20, 21] suggest extensions to WS-BPEL (or the predecessor BPEL4WS). This creates language dialects and own workflow engines must be supported. We are using solely existing WS-BPEL constructs and hiding complexity by automatically generating the according code fragments. In [15] a Eclipse-based tool called Sedna is described that uses standard WS-BPEL without language-specific extensions. Sedna supports constructs as sub-workflows, macros and plugins to hide code complexity. Even if the modeling of a scientific workflow is simplified in Sedna, the user still needs technical background and has to use a control-driven representation, which is not suitable for scientists. We use a data flow-oriented and domain-specific representation for the scientific user.

The use of model-driven technologies in the scientific workflow domain is rather rudimentary. Approaches as [22] uses code generation, however, without using well-adopted model-driven technologies and concentrating on the mapping of abstract workflow activities to concrete resources with the focus of workflow optimization. The tool Sedna [15] support macros and plugins to generate WS-BPEL code, using an own code generation mechanism. The utilization of model-driven technologies is more popular in the business workflow domain. For example to transform a business workflow modeled with the Business Process Modeling Notation (BPMN; abstract level) to WS-BPEL code (technical level) [6, 7]. Approaches as [23] deals furthermore with the automatic generation of Web service interfaces. If these approaches from the business domain can be reused in our work is currently not clear. It has to be considered that the construction of business and scientific workflows have a different focus. A business workflow is noted as fully or partly technical representation of a so called business process of a company. The progress from a business process to an executable business workflow is a top down development process. That means several development steps may be necessary, for example the implementation of new Web services, to support a certain functionality of the business workflow. Current approaches try to semi-automatically support this progress. Scientific workflows, however, consists of the composition of existing data and processors and they should be executable directly after modeling without further development steps.

4. Conclusion and Future Work

In this paper we have shown that the execution of scientific workflows can benefit from standardized and well-adopted business workflow technologies such as WS-BPEL. Therefore, we motivated the separation of a domain-specific level for scientific workflow modeling and a technical level for scientific workflow execution. We presented an extendable transformation framework based on model-driven technologies in order to automatically transform a scientific workflow description at the domain-specific level to an executable scientific workflow at the technical level. Our approach intends to complement existing scientific workflow technologies instead of their replacement. Even though our approach is focusing on SOA-based Grid infrastructures, it can be applied to SOA-based infrastructures such as the emerging cloud technology, too.

Next, we plan to implement a prototype for the transformation framework based on the Eclipse Modeling Framework (EMF)[11]. The data flow-oriented model and control flow-oriented model will be represented as Ecore models based on the Ecore metamodel. Different technologies can be used to perform transformations on an Ecore model. Transformations from one Ecore model to another Ecore model are called model-to-model (M2M) transformations and can be build with the Query View Transformation (QVT)[12] or ATLAS Transformation Language (ATL)[13]. Transformations from one Ecore model to source code are called model-to-text (M2T) transformations and can be build with Xpand[14]. In our case, the mapping step uses a M2M transformation and the transformation step uses a M2M transformation for the first sub-step and a M2T transformation for the second sub-step. As workflow engine we use the BIS-Grid Workflow Engine[15] [24] from the BIS-Grid project[16][17]. The BIS-Grid Workflow Engine is a Grid wrapper for an arbitrary WS-BPEL workflow engine in the background.

References

[1] D. Hollingsworth, Workflow Management Coalition - The Workflow Reference Model, Tech. rep., Workflow Management Coalition (Jan. 1995).

[2] B. Ludäscher, M. Weske, T. McPhillips, S. Bowers, Scientific Workflows: Business as Usual?, in: U. Dayal, J. Eder, J. Koehler, H. Reijers (Eds.), 7th Intl. Conf. on Business Process Management (BPM), LNCS 5701, Ulm, Germany, 2009.
URL `http://daks.ucdavis.edu/ ludaesch/Paper/bpm09-ludaescher.pdf`

[3] Y. Gil, E. Deelman, M. Ellisman, T. Fahringer, G. Fox, D. Gannon, C. Goble, M. Livny, L. Moreau, J. Myers, Examining the Challenges of Scientific Workflows, Computer 40 (12) (2007) 24–32. doi:10.1109/MC.2007.421.

[4] OMG, Business Process Modeling Notation, V1.1, Business Process Modeling Notation, V1.1 (01 2008).

[5] OASIS, Web Services Business Process Execution Language Version 2.0, http://docs.oasis-open.org/wsbpel/2.0/OS/wsbpel-v2.0-OS.html (04 2007).

[6] C. Ouyang, M. Dumas, A. H. M. ter Hofstede, W. M. P. van der Aalst, From BPMN Process Models to BPEL Web Services, in: ICWS '06: Proceedings of the IEEE International Conference on Web Services, IEEE Computer Society, Washington, DC, USA, 2006, pp. 285–292. doi:http://dx.doi.org/10.1109/ICWS.2006.67.

[7] C. Ouyang, W. van der Aalst, A. t. H. M. Dumas, Translating BPMN to BPEL, Tech. rep. (2006).

[8] I. Foster, What is the grid? - a three point checklist, GRIDtoday 1 (6).
URL `http://www.gridtoday.com/02/0722/100136.html`

[9] OASIS, Web services resource framework (wsrf), http://www.oasis-open.org/committees/wsrf/.

[10] A. Barker, J. van Hemert, Scientific Workflow: A Survey and Research Directions, in: R. Wyrzykowski, et al. (Eds.), Seventh International Conference on Parallel Processing and Applied Mathematics, Revised Selected Papers, Vol. 4967 of LNCS, Springer, 2008, pp. 746–753.

[11] G. Scherp, A. Höing, S. Gudenkauf, W. Hasselbring, O. Kao, Using UNICORE and WS-BPEL for Scientific Workflow Execution in Grid Environments (to appear), in: Euro-Par 2009 Workshops - Parallel Processing, Vol. Lecture Notes in Computer Science of Lecture Notes in Computer Science, Springer, 2010.

[12] S. Gudenkauf, W. Hasselbring, A. Höing, O. Kao, G. Scherp, H. Nitsche, H. Karl, A. Brinkmann, Employing WS-BPEL Design Patterns for Grid Service Orchestration using a Standard WS-BPEL Engine and a Grid Middleware, in: The 8th Cracow Grid Workshop, Academic Computer Center CYFRONET AGH, Cracow, Poland, 2009, pp. 103 – 110.

[11] http://www.eclipse.org/modeling/emf/

[12] http://www.omg.org/spec/QVT/1.0/

[13] http://www.eclipse.org/m2m/atl/

[14] http://wiki.eclipse.org/Xpand

[15] http://bis-grid.sourceforge.net

[16] http://www.bisgrid.de

[17] This project is supported by the German Federal Ministry of Education and Research (BMBF) under grant No. 01IG07005 as part of the D-Grid initiative.

[13] Y. Wang, C. Hu, J. Huai, A New Grid Workflow Description Language, in: SCC '05: Proceedings of the 2005 IEEE International Conference on Services Computing, IEEE Computer Society, Washington, DC, USA, 2005, pp. 257–260. doi:http://dx.doi.org/10.1109/SCC.2005.14.

[14] W. Emmerich, B. Butchart, L. Chen, B. Wassermann, S. Price, Grid Service Orchestration Using the Business Process Execution Language (BPEL), Journal of Grid Computing 3 (3-4) (2005) 283–304. doi:10.1007/s10723-005-9015-3.
URL `http://dx.doi.org/10.1007/s10723-005-9015-3`

[15] B. Wassermann, W. Emmerich, B. Butchart, N. Cameron, L. chen, J. Patel, Sedna: A BPEL-Based Environment for Visual Scientific Workflow Modeling, Springer, 2006, Ch. 0, pp. 427–448.

[16] O. Ezenwoye, S. M. Sadjadi, A. Cary, M. Robinson, Orchestrating WSRF-based Grid Services, Tech. rep., School of Computing and Information Sciences, Florida International University (April 2007).

[17] F. Leymann, Choreography for the Grid: towards fitting BPEL to the resource framework: Research Articles, Concurr. Comput. : Pract. Exper. 18 (10) (2006) 1201–1217. doi:http://dx.doi.org/10.1002/cpe.v18:10.

[18] W. Tan, L. Fong, N. Bobroff, BPEL4Job: A Fault-Handling Design for Job Flow Management, in: ICSOC '07: Proceedings of the 5th international conference on Service-Oriented Computing, Springer-Verlag, Berlin, Heidelberg, 2007, pp. 27–42. doi:http://dx.doi.org/10.1007/978-3-540-74974-5_3.

[19] A. Akram, D. Meredith, R. Allan, Evaluation of BPEL to Scientific Workflows, in: CCGRID '06: Proceedings of the Sixth IEEE International Symposium on Cluster Computing and the Grid, IEEE Computer Society, Washington, DC, USA, 2006, pp. 269–274. doi:http://dx.doi.org/10.1109/CCGRID.2006.44.

[20] T. Dörnemann, T. Friese, S. Herdt, E. Juhnke, B. Freisleben, Grid Workflow Modelling Using Grid-Specific BPEL Extensions.
URL `http://edoc.mpg.de/316604`

[21] T. Dörnemann, M. Smith, B. Freisleben, Composition and execution of secure workflows in wsrf-grids, Cluster Computing and the Grid, IEEE International Symposium on 0 (2008) 122–129. doi:http://doi.ieeecomputersociety.org/10.1109/CCGRID.2008.74.

[22] E. Deelman, J. Blythe, Y. Gil, C. Kesselman, G. Mehta, K. Vahi, K. Blackburn, A. Lazzarini, A. Arbree, R. Cavanaugh, S. Koranda, Mapping Abstract Complex Workflows onto Grid Environments, Journal of Grid Computing 1 (1) (2003) 25–39. doi:10.1023/A:1024000426962.
URL `http://dx.doi.org/10.1023/A:1024000426962`

[23] X. Yu, Y. Zhang, T. Zhang, L. Wang, J. Zhao, G. Zheng, X. Li, Towards a Model Driven Approach to Automatic BPEL Generation, in: ECMDA-FA, 2007, pp. 204–218.

[24] S. Gudenkauf, W. Hasselbring, A. Höing, G. Scherp, O. Kao, Workflow Service Extensions for UNICORE 6 — Utilising a Standard WS-BPEL Engine for Grid Service Orchestration (2009) 103–112doi:http://dx.doi.org/10.1007/978-3-642-00955-6_13.