Towards supporting the life cycle of large scale scientific experiments

Marta Mattoso*, Cláudia Werner,
Guilherme Horta Travassos, Vanessa Braganholo,
Eduardo Ogasawara, Daniel de Oliveira,
Sérgio Manuel Serra da Cruz and Wallace Martinho

Programa de Engenharia de Sistemas e Computação
Cidade Universitária, Centro de Tecnologia,
Federal University of Rio de Janeiro,
Bloco H, Sala 319, Caixa Postal: 68511, Rio de Janeiro 21941-972, Brazil
E-mail: marta@cos.ufrj.br E-mail: werner@cos.ufrj.br
E-mail: ght@cos.ufrj.br E-mail: braganholo@dcc.ufrj.br
E-mail: ogasawara@cos.ufrj.br E-mail: danielc@cos.ufrj.br
E-mail: serra@cos.ufrj.br E-mail: wpereira@cos.ufrj.br
*Corresponding author

Leonardo Murta

Fluminense Federal University,
Rua Passo da Pátria, 156 - Bloco E, 3º andar, Sala 304,
São Domingos, Niterói, Rio de Janeiro 24210-240, Brazil
E-mail: leomurta@ic.uff.br

Abstract: One of the main challenges of scientific experiments is to allow scientists to manage and exchange their scientific computational resources (data, programs, models, etc.). The effective management of such experiments requires a specific set of cardinal facilities, such as experiment specification techniques, workflow derivation heuristics and provenance mechanisms. These facilities may characterise the experiment life cycle into three phases: composition, execution, and analysis. Works concerned with supporting scientific workflows are mainly concerned with the execution and analysis phase. Therefore, they fail to support the scientific experiment throughout its life cycle as a set of integrated experimentation technologies. In large scale experiments this represents a research challenge. We propose an approach for managing large scale experiments based on provenance gathering during all phases of the life cycle. We foresee that such approach may aid scientists to have more control on the trials of the scientific experiment.

Keywords: scientific workflow; scientific experiment; *in silico* experiment; scientific experiment life cycle; e-science.

Reference to this paper should be made as follows: Mattoso, M., Werner, C., Travassos, G.H., Braganholo, V., Ogasawara, E., Oliveira, D., Cruz, S.M.S., Martinho, W. and Murta, L. (2010) 'Towards supporting the life cycle of large scale scientific experiments', *Int. J. Business Process Integration and Management*, Vol. 5, No. 1, pp.79–92.

Biographical notes: Marta Mattoso is a Professor of the Department of Computer Science at the COPPE Institute from Federal University of Rio de Janeiro since 1994, where she leads the Distributed Database Research Group. Her current research interests include distributed and parallel databases and data management aspects of scientific workflows. She is the Principal Investigator in research projects in those areas, with funding from Brazilian agencies, including CNPq, CAPES, FAPERJ, FINEP and INRIA-France. She has published over 100 refereed international journal articles and conference papers. She has served in program committees and is a regular reviewer of international journals.

Cláudia Werner is a Professor of the Department of Computer Science at the COPPE Institute from Federal University of Rio de Janeiro (UFRJ) since 1994, where she leads the Software Reuse Research Group. She is a 1D CNPq – Brazilian Research Council and FAPERJ Researcher. She has received the Doctor of Science degree from UFRJ. She has been active in the software engineering research community for more than 15 years and her current research interests include software reuse, component based development and software product-line. Information regarding her research projects, publications and working activities can be found at http://www.cos.ufrj.br/~werner.

Guilherme Horta Travassos is a Professor of Software Engineering in the Systems Engineering and Computer Science Program at COPPE/UFRJ – Brazil and also a CNPq and FAPERJ Researcher. He leads the Experimental Software Engineering Group at COPPE/UFRJ. His current research interests include experimental software engineering, e-science and non-conventional applications, OO software quality, environments and VV&T. He has published over 200 refereed papers in conferences, book chapters and journals. He is a member of ISERN, ACM, SBC – Brazilian Computer Society, BKCASE Project and Elsevier - Information, Software and Technology editorial board. Further information can be found at http://www.cos.ufrj.br/~ght.

Vanessa Braganholo is a Professor of the Computer Science Department at the Federal University of Rio de Janeiro (UFRJ), Brazil. She has a Computer Science PhD (2004) from the Federal University of Rio Grande do Sul (UFRGS), Brazil. She has published several papers and served in program committees of both Brazilian and international conferences and journals. Her research interests include databases and e-science.

Eduardo Ogasawara is a DSc candidate at the Department of Computer Science at the Federal University of Rio de Janeiro. He received his BSc in 1997 and MSc in 2000, both from Federal University of Rio de Janeiro. His interests include e-science, workflow management, software reuse, HPC, P2P, databases and data mining.

Daniel de Oliveira is a DSc candidate at the Department of Computer Science Federal University of Rio de Janeiro. He received his BSc in 2005 and MSc in 2008, both from Federal University of Rio de Janeiro. His interests include e-science, cloud computing, workflow management, data mining, text mining and ontologies.

Sérgio Manuel Serra da Cruz is a DSc candidate at the Department of Computer Science at the Federal University of Rio de Janeiro. He works as a Researcher at Núcleo de Computação Eletrônica-UFRJ and as a Professor at Estácio de Sá University. His interests include provenance, distributed computing, e-science, bioinformatics, workflow management systems, databases, metadata and ontologies.

Wallace Martinho is an MSc student at the Department of Computer Science at the Federal University of Rio de Janeiro. He received his BSc in 2006 from State University of Rio de Janeiro. His interests include e-science, software engineering and simulation address.

Leonardo Gresta Paulino Murta is a Professor of the Computer Science Institute at Fluminense Federal University (UFF), Niterói, Brazil. He has received his DSc in Systems Engineering and Computer Science from the Federal University of Rio de Janeiro (UFRJ), Brazil, in 2006. As a result of his academic work at UFF and UFRJ, he has published over 70 papers. His current research interests include the adaption of existing configuration management and software reuse techniques to the e-science context. More information can be found at http://lattes.cnpq.br/1565296529736448.

1 Introduction

Modern scientific investigations are dependent on huge amounts of data and complex scientific apparatus. Such investigations, also known as large scale science, enable a new order of collaborative inter-disciplinary research based on shared expertise, instruments and computing resources. It has also driven the field of high performance computing (HPC) and yielded the design of novel tools and development of techniques that are dramatically different from the traditional ones available at personal computers or conventional installations, mainly with respect to performance. The paradigm of scientific workflows introduced a new model of interaction in the research community, which requires richer information than the one captured in the traditional script-based pipelines.

Scientific workflows are being used as an abstraction that models a flow of activities and data ready to be executed by a workflow engine. Scientific workflows are an attractive alternative to represent those pipelines or script-based applications. In scientific workflows, these activities are often programs or services that represent solid algorithms and computational methods. Current scientific workflow management systems (SWfMS) are focused on managing the execution of the workflow. However, executing a workflow is only part of a large scale scientific experiment.

A scientific experiment may be defined as ‘a test under controlled conditions that is made to demonstrate a known truth, examine the validity of a hypothesis, or determine the efficacy of something previously untried’ (Soanes and Stevenson, 2003). It may also be defined as ‘a situation, created in laboratory, which aims to observe, under controlled conditions, the relationship between phenomena of interest’ (Jarrard, 2001). The word ‘control’ is used to indicate that there are efforts to eliminate, or at least reduce, as much as possible, the occasional mistakes during a planned observation (Jarrard, 2001). Based on these definitions, we may conclude that a scientific experiment can be strictly associated to a set of controlled actions. These controlled actions include variations of tests, and their results are usually compared to each other to accept or refute a hypothesis.

In this work, the concept of a scientific experiment is seen as encompassing the concept of a workflow, and not as a synonym. A scientific workflow may be seen as a controlled action of the experiment. Hence, the workflow can be defined as one of the trials conducted in the context of a scientific experiment to evaluate a controlled action. The set of trials represented by each distinct workflow execution defines one scientific experiment. Therefore, a scientific workflow is only part of a scientific experiment. The life cycle of the scientific experiment involves several phases including the execution of one workflow trial.

Our goal in this paper is to contribute with a broader understanding of the life cycle of large scale *scientific* experiments. The term life cycle is generally used to describe a sequence of steps imposed to the development of a given study, it denotes the entire set of tasks over which scientists iterate throughout an experimental study (Deelman et al., 2009). In our view, the life cycle has three main phases: *composition*, *execution* and *analysis*, briefly presented as follows.

Ideally, during *composition* scientists should be able to define and look at the experiment definition in a high level of abstraction. This definition should allow them to make choices for particular methods to support their experiment's hypothesis.

Then, during *execution* scientists should be able to configure scientific workflows that correspond to their experiment definition. In the execution phase, workflow programs are chosen for the methods defined on the experiment so that the workflow can be executed by the SWfMS.

Additionally, all tests defined for the scientific hypothesis should be registered to represent the experiment itself, not for a single workflow execution. Thus, during the *analysis* phase, the scientist should be able to analyse these registers looking at decisions made during the composition phase and execution results for the different workflow trials of the experiment.

In large scale experiments, supporting the management of these phases of the experiment becomes a fundamental issue. The number of parameters being explored is usually large, as well as the variations of well known methods, algorithms, and programs used in the experiment trials.

As an example of this life cycle, during the *composition* phase of experiment *E*, the scientist decides to follow methodology *M*, which is based on a family of algorithms, *A1*, *A2*, and *A3*. Suppose that each algorithm has two different programs that implement them: *PA11*, *PA12*, *PA21*, …, *PA32*. Therefore, to evaluate these alternatives the scientist would have to specify and *execute* several workflows. If the scientist wants to register and *analyse* these experiment alternatives with current systems, several limitations are found. SWfMS are focused on one workflow definition and its execution is disconnected from the concept of the experiment as we present here. Systems like Askalon (Qin et al., 2007), Kepler (Altintas et al., 2004), Taverna (Hull et al., 2006) and VisTrails (Callahan et al., 2006) have powerful graphical interfaces to represent workflows. However, in these systems, workflows need to be defined based on activities that are associated to executable programs/services or any other available implementation. Current SWfMS do not support the definition and registration of a workflow based on algorithms. In the experiment *E*, of our previous example, the scientist should be able to define and execute the corresponding different workflow trials. Thus, in SWfMS, queries like:

1 which workflows use *PA11*

2 which data input was provided to *PA32*; can all be successfully answered.

However, a query like: which workflows belong to experiment *E* may not be trivially obtained. Moreover, queries like:

1 which workflows of experiment *E* use algorithm *A2*

2 which workflows of experiment *E* follow methodology *M*, are not supported.

Another limitation of current SWfMS is related to the lack of support in manually executed workflow activities. For example, consider an experiment that is dependent of results generated by two different scientific workflows. These two workflows may not be 'merged' in order to create a single workflow to represent the experiment because between them there are many manual activities (that needs human intervention) such as validation of produced data, or 3D modeling in external tools. Each one of these workflows can be modeled and executed by different research teams inside the same company. Since the manual activities may take weeks and need specific processing, SWfMS do not support workflow specifications with manual activities. However, the scientist may want to register the experiment with all its activities, independent from the workflow execution limitations. The analyses that the scientist may need to make can be related to the experiment, not to disconnected scientific workflows. For example, scientists may need to know which parameters of the 'first' workflow influenced the results of the 'second' workflow, or even relate independent trials of each workflow as a unique high-level trial, composed of the first workflow trial, the manual activities, and the second workflow trial. It is important to highlight that all the data that are needed to answer this question of the experiment is produced on the SWfMS, but by different executions of 'disconnected' scientific workflows.

Using current approaches, scientists are not able to make several analyses to draw conclusions of the experiment. The existing SWfMS only store, query, or visualise data resulting from each disconnected scientific workflow execution and not the entire experiment. They lack a support for all the three phases of the experiment life cycle.

An open issue in the composition phase is a high level model to define the experiment and the corresponding process to the modelling itself. The execution phase is reasonably well supported by current SWfMS, however there are open issues in supporting distributed executions and workflow configuration management. Another open issue is in the analysis phase, where this high level definition of the experiment must be registered in a way that the previous unanswered queries could be answered. For example, by associating models to executions, queries relating algorithms to workflow executions can be answered.

In summary, there are several tasks not fully supported by SWfMS when it comes to support the experiment life cycle in an integrated way. In order to meet these large scale experimentation needs, new frameworks, models, theories, and techniques to support a broader experiment life cycle must be investigated.

In this paper, differently from the current mainstream, we advocate a view of a novel large scale experiment life cycle. We foresee that supporting this life cycle may aid scientists to have more control on scientific experiment trials. We contribute by detailing this life cycle and the corresponding issues towards supporting the experiment life cycle. We also analyse current systems with respect to this lack of support.

The paper is organised as follows: Section 2 presents related work, explaining some important concepts and detailing the existing large scale experiment life cycles. Section 3 introduces our life cycle proposal. Section 4, 5, and 6 detail the phases of the proposed life cycle (composition, execution and analysis, respectively) and point some challenges and solutions related to each phase. Finally, Section 7 concludes the paper presenting some future directions.

2 Related work

This section introduces, in Section 2.1, some important concepts related to the experimentation process and scientific workflows. Section 2.2 analyses current views of the experiment life cycle. None of the related work presents systems that support the whole experiment life cycle.

2.1 Large scale experiments

Over the years, scientific literature considered experimental studies as *in vitro* or *in vivo*. *In vivo* experiments are those where subjects in their own environments are involved, while *in vitro* experiments are those where studies are executed in a controlled environment, such as a laboratory. However, the fast development of computational infrastructures in the last decades such as grids, clusters, web services, and cloud computing enabled two new classes of computer-based experiments named *in virtuo* and *in silico* (Travassos and Barros, 2003). The major characteristic of those experiments is the massive use of computer models, which represent an abstraction of the observable reality, and present, in a generic way, characteristics or relations between elements of such reality. *In virtuo* experiments are those where there is an interaction among subjects and a computerised model of reality (virtual reality), while *in silico* experiments are those where both the subjects and real world are described as computer models. Large scale scientific experiments can be categorised most of time as *in silico* experiments.

In silico experiments may be found in many different domains such as bioinformatics (Stevens et al., 2004; Dávila et al., 2008), meteorology (Gannon et al., 2007), ecology (Pennington et al., 2007), astronomy (Berriman et al., 2007), and deep water oil exploitation (Oliveira et al., 2009a). Large scale *in silico* experiments represent those studies that encompass multiple combinations of programs, services, disperse scientific apparatus, huge amounts of data resources, and intense usage of heterogeneous and distributed computing environments. In these cases, the management of such experiments ends-up too complex to be manually accomplished (Cavalcanti et al., 2005). In this work we refer the generic term of scientific experiments restricted to large *in silico* experiments.

A scientific experiment is conducted by a scientist or research team, responsible for designing, executing, controlling, and analysing it. In this context, scientific models, methods, algorithms, computer programs, services, and data became the most valuable scientific assets used by scientists during the experimentation life cycle.

In fact, the relationship between these scientific assets is a fundamental requirement for scientific experiments and it is mainly related to the concept of provenance. Provenance (also referred as lineage or pedigree) represents the ancestry of an object (Freire et al., 2008). Provenance of an object, such as a data product, contains information about the process used to derive the object, in this case the data product. It provides important documentation that is essential to preserve the data, to determine their quality and authorship, and to reproduce as well as to interpret and validate the associated scientific results generated by large scale scientific experiments. According to Freire et al. provenance can be categorised into prospective and retrospective. Prospective provenance may be defined as a computational specification of activities of a given workflow and corresponds to the steps that must be followed to generate a data product or class of data products (Freire et al., 2008). Retrospective provenance defines the actual steps taken during the execution of the workflow. As far as we are concerned, up to now there is no integrated solution that acts as a backbone, managing provenance data captured at distinct phases of the experiment life cycle.

2.2 Definitions of scientific experiment life cycles

In order to perform a large scale scientific experiment, a set of phases that are important through the course of an experimental study must be accomplished. These phases are common to experimental studies in many domains. An experiment life cycle may be represented as a model that describes a variety of phases that must take place during the experiment life period. The idea of representing a life cycle as a model is exhaustively used in other domains, such as software engineering to represent the software development life cycle (Pressman, 2004). There are several models for such experiment life cycles, each one describing different approaches.

Our model for the experiment life cycle has three phases: composition, execution and analysis, briefly described at the introduction and detailed on Section 3. This subsection details some of the existing approaches to model experiment life cycles and their particularities.

Livny et al. (1994) propose a life cycle based on a generic definition for experimental studies. By generic the authors mean that the proposed experiment life cycle may be applied to most of the experimental studies usually executed. They recognise that scale and scope of an experimental study is determined by the ability of a given research team to manage and generate the ever increasing amount of data. Thus, they focus their work on the data management of desktop experiments. However, their view of an experiment is strongly related to the workflow ready to be executed and the management of execution data. According to our life cycle proposal, the life cycle of Livny et al. (1994) is restricted to supporting the execution phase. It presents a workflow design step that is close to the execution, rather than a high level view of the experiment based on models and algorithms, for example.

The myGrid project (Goble et al., 2003) is one of the first works to define a scientific experiment life cycle. Their proposal has six phases: construction, discovery, personalisation, execution and management. In Goble et al. (2003) and Stevens et al. (2003) the authors draw attention to requirements and challenges of supporting this life cycle. However, their concept of an experiment is still close to the executable workflow. Thus, during the construction, discovery and personalisation phases, the scientist is guided to finding executable services and resources. In Goble et al. (2003) the myGrid team proposes the use of ontologies to add semantics to bioinformatics workflows, but they do not show how or the role of ontologies in their experiment life cycle.

Bose and Frew (2005) present a variation of the life cycle proposed by Livny et al. (1994) and highlight the importance of prospective lineage data of scientific experiments. Despite its contribution, it presents the same limitation of the life cycle proposed by Livny et al., and Bose and Frew do not make explicit the distinction between the workflow and the scientific experiment. Thus it only details the workflow execution phase.

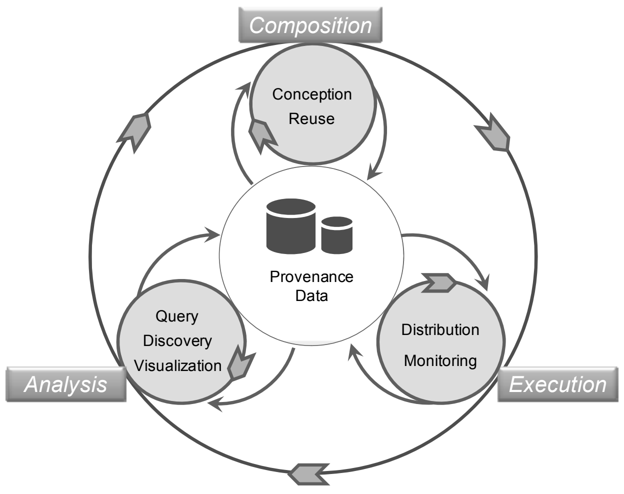
Oinn et al. (2007) propose a life cycle based on the work of Bose and Frew (2005), adding two new steps: sharing and reuse. These steps are focused on making the entire experiment public, so it may be reused by other scientists. These two new steps correspond to part of our view of the composition phase of the experiment. This life cycle adds more semantic to the workflow definition. However, they also present the concepts of workflow and the scientific experiments interchangeably. Thus, an experiment that has several workflow instances as trials are not explicitly represented.

All of these life cycles focus on a similar perspective. They do not share our view of two different concepts, i.e., the experiment as a high level definition and the workflow as one of the instances of the experiment trial. Consequently, the scientist cannot choose between working in the experiment on a high level of abstraction and its corresponding workflow execution. Queries like the ones presented at the Introduction remain unanswered. We believe that the first step towards this support is to present the life cycle and its challenges. Section 3 presents our approach to model a large scale scientific experiment life cycle.

3 Our model of a scientific experiment life cycle

In this section we introduce our model of a scientific experiment life cycle. Figure 1 presents the large scale scientific experiment life cycle, which essentially consists of multiple loops traversed by the scientist several times in the course of a scientific experiment. In Figure 1, the major phases may be identified: *composition*, *execution* and *analysis*. Each phase has an independent cycle, taking place at distinct moments of the experiment and handling explicit provenance metadata.

Figure 1 The proposed scientific experiment life cycle



The *composition* phase is responsible for structuring and setting up the whole experiment, establishing the logical sequence of activities, the type of input data and parameters that should be provided, and the type of output data that are generated. All this information is related to the prospective provenance. This phase may be further decomposed in two sub-phases: *conception* and *reuse*. The *conception* sub-phase is responsible for setting up the experiment. During conception the goal is to capture the abstract workflow, which is a representation of the scientific experiment protocol. Additionally, during conception, it is possible to setup a concrete workflow, which is obtained through the derivation of the abstract workflow. The *reuse* sub-phase is responsible for retrieving an existing experiment and adapting it for a new purpose.

The *execution* phase is responsible for materialising the experiment, thus defining exactly the input data and the parameters to be delivered to the SWfMS in order to execute the experiment, i.e., a concrete workflow instance. All the execution information is related to the retrospective provenance. This phase may also be further decomposed in two sub-phases: *distribution* and *monitoring*. Distribution is related to the need of executing workflow activities in heterogeneous environments, due to many different reasons, such as performance. Monitoring is related to need of checking the status of the workflow execution, since it may last for a long time.

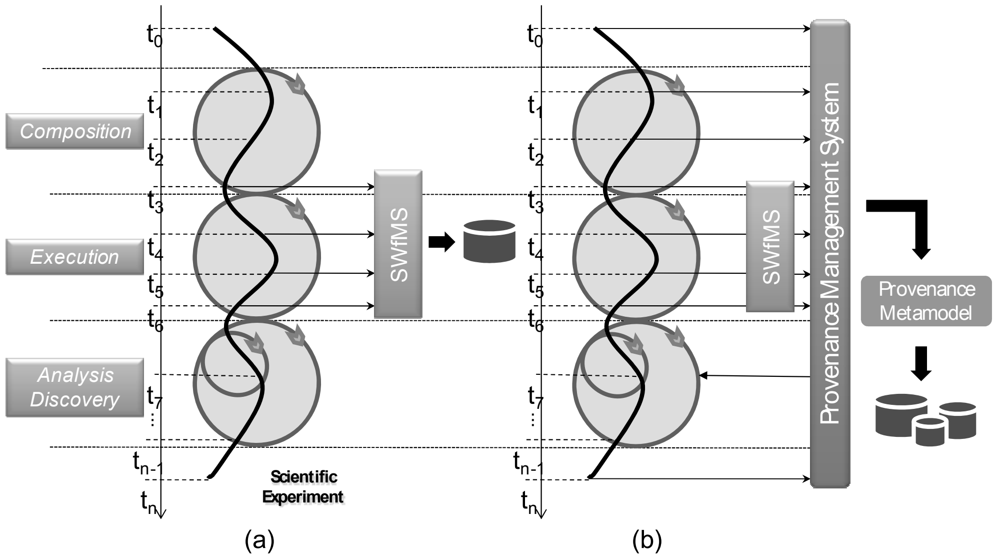
Finally, the *analysis* phase is responsible for studying the data generated by the composition and execution phases. This phase may be further decomposed into a sub-phase that supports the analysis of results, such as *query* and *visualisation*. In this phase we may also have a *discovery* sub-phase that is responsible for deciding on the conduction of the experiment. A scientist may face two different situations when analysing the results of an experiment:

1 the result is likely to be correct

2 the hypothesis is refuted. In both cases scientists will probably need new workflow executions to effectively validate the hypothesis or create a new one.

In the case of running several workflows to validate a hypothesis, all the workflows (with different parameters and data sets) must be connected to the same experiment. In fact, it is in this requirement that most of the existing SWfMS fails.

Figure 2 Current approach for provenance gathering for each cycle of the *in silico* experiment (a) an ideal provenance gathering approach for each cycle of the *in silico* experiment (b)



In Figure 2(a), we present an example of a scientific experiment timeline for each phase of the experiment cycle. During the timeline, each phase is represented in the time axis; where t_0 means the beginning of the cycle run and t_n its end. It shows the current support of SWfMS along the three phases. In Figure 2(b), we show an ideal system where provenance data is being collected and analysed along the three phase's independent from the SWfMS.

The following sections of this paper detail each one of the phases, thus analysing the requirements and listing the computational resources that currently support each one of them. Provenance data is presented as the link between the three phases.

4 Composition

As previously mentioned, the composition phase is divided in two main sub-phases: *conception* and *reuse*. Conception is responsible for setting up the experiment activities. It includes the specification and modeling of scientific experiments in different levels of abstraction and the registering of workflow variations in the context of the experiment. Reuse is a part of the composition phase related to the ability to model a scientific experiment by taking advantage of a previous elaborated workflow (or even an entire experiment) to build new workflows (or experiments). In the next subsections, conception and reuse are detailed, including information about the challenges in each area.

4.1 Conception

In the literature, some authors also propose the use of different levels of abstraction to compose scientific workflows. Those authors usually classify scientific workflows in two levels of abstraction: abstract and concrete (Cavalcanti et al., 2005; Yu and Buyya, 2005;

Deelman et al., 2009; Ogasawara et al. 2009b). An abstract workflow is a representation of a flow of activities that is specified without mapping to execution resources. In a simple way, an abstract workflow is the chaining of activities that specifies what should be done, but without specifying how. The abstract level itself may be further decomposed into different sublevels of abstraction, as needed. For example, we may define an abstract workflow in terms of its abstract generic activities, then decompose the activities into methods and, finally, in terms of algorithms. All these abstract levels do not bind to infrastructure resources, focusing strictly on concepts. On the other hand, a concrete workflow binds abstract workflow activities to specific execution resources. In the same way as in software development (Pressman 2004), an abstract workflow corresponds to an analysis model and a concrete workflow corresponds to a design model.

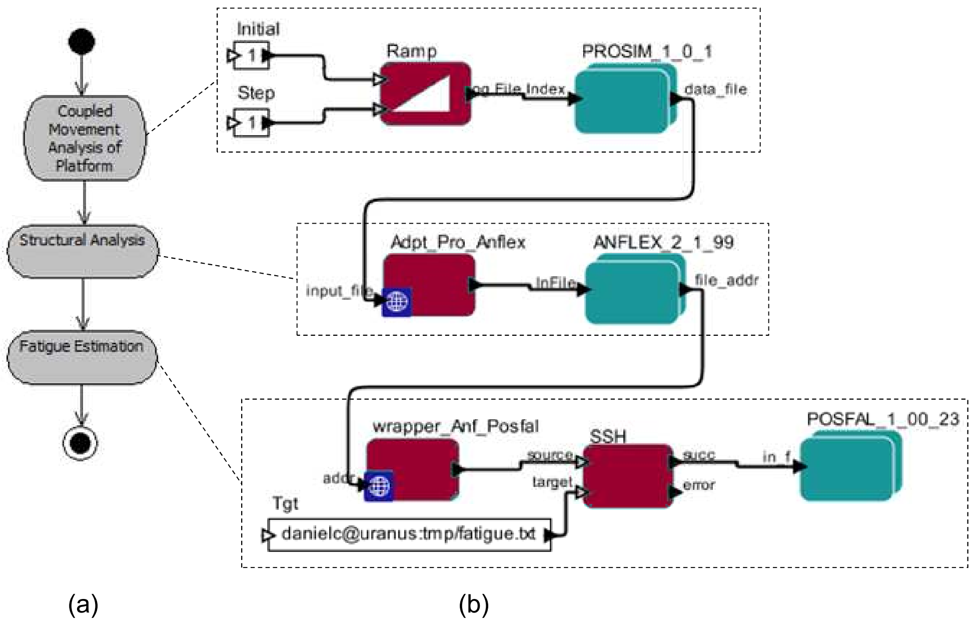
Scientists should be able to model the scientific experiment in any of those abstract levels. Figure 3(a) and Figure 3(b) presents a deep water oil exploitation (Oliveira et al. 2009a) workflow for fatigue estimation of a riser[1], at abstract and concrete levels, respectively. The abstract workflow was modeled using UML activity diagram, while the concrete workflow was built using Kepler (Altintas et al. 2004). By analysing the abstract workflow and its concepts shown in Figure 3(a), it is possible to identify that the workflow is composed by three activities that include the coupled movement analysis of the platform, the structural analysis of each riser and the fatigue estimation. We may also observe that the actual technique to be used in each activity is not selected yet. The choice of software or technological resources to be used during the scientific workflow execution is not done at this conceptual level.

The concrete workflow shown in Figure 3(b) is presented using the Kepler notation. In Kepler, actors are represented by dark gray and light gray squares. To bring a uniform vocabulary among all scientific workflow notations, Kepler actors will be called activities. These activities are directly linked to each other, and also represent relationship dependence. At concrete level, workflow activities are specific packages or programs of an application domain (light gray activities, for example, PROSIM, ANFLEX and POSFAL, represented in Kepler as sub-workflows, hiding other concrete details), special adaptor activities (dark gray activities, for example, wrappers), are used to manipulate input data and transform it into a required format in order to feed other activities, and infrastructure activities (dark gray activities, for example SSH and ramp). In this way, activities have input ports and output ports that are used to interconnect them.

The concrete workflow of Figure 3(b) is one possible derivation of the abstract workflow. Other UML diagrams and forms are used to register the different ways that the abstract activity of 'risers structural analysis' can be conducted. The algorithm behind the chosen ANFLEX program is just one of them. All this semantic information needs to be registered during the composition phase.

Due to the incremental nature and the presence of different levels of abstraction, the conception of a scientific experiment, like in software development, requires some methodological support. This is justified since the large scale of the experiments and the amount of elements involved are considerably increasing, thus demanding new ways to support the domain specialist in the conception of the workflow, starting from the requirements until design of the workflow at concrete level. Currently, some initiatives (Verdi et al. 2007, Martinho et al. 2009) are under development to define a methodology for conception.

Figure 3 Abstract workflow for fatigue estimation of a riser (a) concrete workflow for fatigue estimation using PROSIM, ANFLEX and POSFAL (b) (see online version for colours)



In order to empower experiment conception, SWfMS would need to support the specification and modeling of scientific workflows in these different levels of abstraction. However, no SWfMS provides an integrated development environment (IDE) (Pressman, 2004) that is able to model the entire experiment in terms of multiple levels of abstraction. Currently, well known SWfMS such as Kepler (Altintas et al. 2004), Swift (Zhao et al., 2005), Taverna (Hull et al., 2006) and VisTrails (Callahan et al., 2006) do not support the definition of workflows at abstract activity levels. The support for concrete workflows by itself may be considered as insufficient for the conception of workflows, especially because, for the domain specialist, it is more important to be concerned about the concept of the experiment rather than infrastructure issues. Considering the scientific experiment as a life cycle that encompasses many workflows or many variations of the same workflow, it is fundamental to support both levels of abstraction, from the abstract to the concrete level, and *vice-versa*. Nevertheless, there is no approach that fulfills this requirement.

The task specification language (TSL) (Lin et al., 2009) and ASKALON workflow composition tool (Qin et al. 2007) are initiatives to represent scientific workflows in different levels of abstraction. TSL is a task template model that encapsulates functional component and mapping of shims (i.e., wrappers that must be inserted into scientific workflows when the tasks are not compatible). The TSL language creates a level of abstraction that provides ways to model workflows without concerning about infrastructure problems. Nevertheless, they do not give higher abstraction to support alternative activities that may characterise the experiment. Askalon workflow composition tool provides a graphic representation of grid workflow applications using UML activity diagrams which are more amenable than pure-textual specifications. Askalon workflows are expressed as abstract grid workflow language (AGWL), which is an XML-based language. Although it uses UML, it does not support a high level of abstraction. This is due to the fact that it is used only to model concrete workflows, coupled to the grid environment, i.e., no experimental conceptual information, such as algorithms is indeed supported and neither alternative activity are represented. Additionally, it is also important to support the analysis on the experiments. Scientists must be able to represent semantics related to both abstract workflow and concrete workflow definitions. At this point, ontologies can play a major role while helping defining these workflows. The relationship of which software implements a certain algorithm used on an activity of the workflow can be an example. Thus, starting from a given abstract workflow definition, the composition of the concrete workflow definition may be supported by inference on the ontology. For instance, an ontology mechanism, such as the one presented by Oliveira et al. (2009b) may be capable of allowing workflow designers to reason about the programs or services to be used when they are composing a scientific workflow. Consequently, to support these sorts of commonsense reasoning, it is fundamental to represent programs, methods, algorithms, activities, and data involved in scientific workflow subject to obtain the desired semantic.

4.2 *Reuse*

Reuse is a sub-phase of the life cycle where both knowledge of software configuration management (Conradi and Westfechtel, 1998) and software reuse (Frakes and Kang, 2005) may be used in order to bring systematic processes for modeling workflows during the composition phase of scientific experiments life cycle.

The chaining of these programs is not a trivial task, and in many cases it becomes a barrier to build more sophisticated analyses or models. SWfMS like Taverna (Hull et al., 2006), Kepler (Altintas et al., 2004), and VisTrails (Callahan et al., 2006) offer rich graphic interfaces where previously registered components can be dragged and dropped to a workflow editing area. This directly results in the setup of a concrete workflow. However, there is no support to the previous steps of the workflow composition process. Indeed, the composition of a workflow includes the conception of activities, the selection of adequate programs or components to enact these activities, and also the setup of the activities flow. In current SWfMS, the action of discovering an activity is limited, since not all components are necessarily registered in the SWfMS. Also, the knowledge of which activities can be linked to each other is still tacit. It is necessary to run a large number of examples to gain some experience in the setup of the activity flow.

An alternative, often used by scientists, is to try to reuse a previously defined workflow or part of it. However, supporting reuse in SWfMS is also limited to a concrete level. Some initiatives have begun to support the setup step of an activity flow of the concrete workflow composition process, such as in VisTrails (Koop et al., 2008; Oliveira et al. 2008; Scheidegger et al., 2008). VisTrails suggests a list of related activities to the workflow being setup based on previously developed workflows stored in a repository. However, these suggestions are limited to previously executed workflows. WOODSS is workflow system that aims at reuse. WOODSS searches for existing solutions (i.e., concrete workflows), which are able to solve a specific problem. However, the WOODSS solution is based directly on concrete workflows, not allowing the reuse in higher levels of abstraction. The myExperiment (Goderis et al., 2008) initiative provides an interesting site with a repository of previously defined workflows. This workflow repository is very useful when scientists need perfect matches, but most of these workflows is defined under the Taverna workflow definition language and belong to the bioinformatics domain. Thus, adapting a workflow from this repository or using a different language for composition is far from trivial. In other words, we advocate the switching from individual components or the reuse of an entire workflow to experiment reuse.

4.3 Interplay between conception and reuse

It is observable that the two sub-phases of the composition phase are related and their frontier is tenuous, particularly when scientists are creating or adapting their experiments. As the experimental process tends to be established, this frontier becomes clear.

An experiment line can be characterised by abstract workflows that are joint/merged together in a way that they can derive multiple workflows at abstract and concrete levels. The experiment line has been inspired on software product lines[2]. Similarly to product lines, the experiment lines allow scientists to compose for reuse. The approach provides a way to compose experiments in a systematic way in order to be further reused.

Additionally, like in the software development domain, scientific experiments demand ways to control the evolution of scientific workflows. Based on the dynamic characteristics of the composition phase, it becomes fundamental to support it from a configuration management[3] perspective (Conradi and Westfechtel, 1998). Actually, configuration management techniques clarify the frontier of conception and reuse, by addressing the following requirements:

1 repositories for workflows with access control, in which it is possible to store workflows and register their stable and under development versions

2 mechanisms to represent and store versions for the activities being used in workflow composition

3 workspaces to support the modelling of a workflow during creation and maintenance phases.

The workspace must also support the workflow publication in the repository. Although no SWfMS provide this support (Ogasawara et al., 2009c), initiatives such as the GExpLine tool (GExp, 2009), provides configuration management support for handling scientific experiments and becomes a valuable asset to support prospective provenance (Freire et al., 2008) for the whole experiment.

5 Execution

At the end of the composition phase, the abstract workflow is then instantiated into a concrete workflow that can be executed under a certain computational infrastructure. This concrete definition is then submitted to an appropriate engine, which is responsible for enacting it and generating results. The action of enacting the concrete workflow and monitoring its execution is named *execution*.

The execution phase encapsulates the infrastructure activities that must be accomplished when running a scientific workflow modeled by a set of software components (e.g., local or remote programs, web services, and grid services). In other words, the execution phase is responsible for retrieving a concrete workflow definition, somehow executing this definition in an appropriate engine, and producing results to be analysed in the following phase of the life cycle. The infrastructure activities should include, for instance: executing, logging, and monitoring (Medeiros et al., 1995; McGough et al., 2007).

When we are dealing with large scale scientific experiments, a fundamental requirement is that a given workflow, or some activities within these workflows, may need to be executed in heterogeneous environments, such as clusters, grids, or even clouds. SWfMS must have infrastructure components that enable dealing with network transport, distribution of data, shared file systems and security issues related to these distributed resources. Actually, once the execution is started, it is important to monitor both the local and distributed execution of a given workflow or an activity within it.

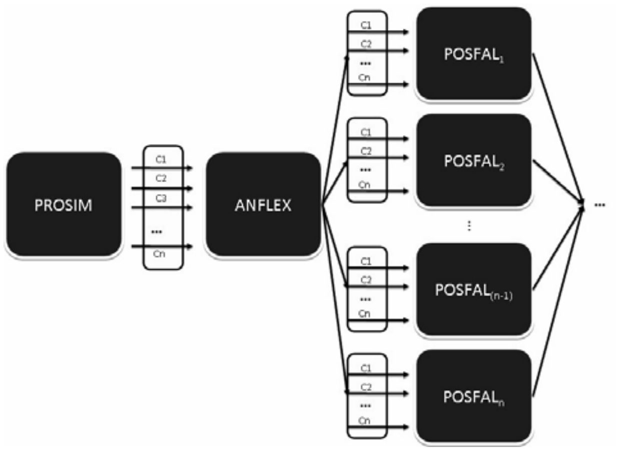
Regarding activity distribution mechanisms, according to Yu and Buyya (2005) it is possible to classify SWfMS in two different categories: the ones that run locally in the desktop of the scientists (i.e., the control is entirely local), which for simplification we are going to call 'local engines', and the ones that execute under distributed environments, which for simplicity we are going to call 'distributed engines'. In the first category we have SWfMS such as Kepler, Taverna and Vistrails, which focus more on provenance and semantics regarding distributed execution components. On the other hand, in the second category we have SWfMS such as Pegasus/Condor (Deelman et al., 2007), Askalon (Wieczorek et al., 2005), Triana (Taylor et al., 2007), and Swift (Zhao et al. 2005). These SWfMS focus on distributed executions regarding semantics and easy-to-use interfaces. Although a deeper discussion of their execution features transcends the scope of this paper, it is important to understand the current strengths and limitations of these two groups of SWfMS.

Local engines usually have graphical interfaces with provenance support, semantics, and monitoring mechanisms for the execution of activities that are being enacted in the desktop of the scientists. Nevertheless, these SWfMS do not provide primitives to distribute large scale or long term activities that are candidates to be parallelised and executed under HPC. On the other hand, distributed workflow systems are usually not focused on easy-to-use interfaces, semantics, and provenance support. Indeed, many of these systems have limited monitoring mechanisms and limited provenance support.

Nevertheless, due to the exploratory scenario of the scientific experiment, a new computing paradigm, named many task computing (MTC) (Raicu et al., 2008), was recently introduced. This paradigm consists on using various computing resources over short periods of time to accomplish many computational tasks, including both dependent and independent ones. Particularly, in this paradigm, two important types of parallelism need to be systematically addressed in scientific workflows (Barga et al., 2008): parameter sweep parallelism (Walker and Guiang, 2007) and data parallelism (Dean and Ghemawat, 2008). Data parallelism can be characterised by the simultaneous execution of one activity a_i of the workflow Wf, where each a_i execution processes a subset of the entire input data. Parameter sweep parallelism is characterised by the simultaneous execution of one activity of the workflow Wf (or all the activities of Wf), each execution using different values for the parameters Pt. For example, Figure 4 shows an example of activity distributed execution following the parameter sweep in MTC paradigm based on the example given on Figure 3. In this case the activity POSFAL runs under a HPC environment and each instance of POSFAL runs consuming different values of parameter C (C_1, C_2, C_3) and so on.

These two types of parallelism may represent a barrier for scientists to control, gather and register workflow provenance. It is important to have means to support these parallelisms implicitly during the workflow execution. Recent approaches implemented on top of Swift (Raicu et al., 2007) and Kepler (Abramson et al., 2009) addresses MTC for a particular engine, but an interesting approach is to bring distributed capabilities with provenance gathering features and monitoring support independent of SWfMS. SWfMS-independent initiatives such as Hydra (Ogasawara et al. 2009a) aim at bridging the gap between the SWfMS and the HPC, thus creating an environment for distributed executions with integrated provenance gathering.

Figure 4 An example of parameter sweep using MTC paradigm for a scientific workflow



Although it is not possible to establish beforehand that neither of these approaches is better than the others, large scale scientific experiment demands for both composing and executing with provenance gathering also while distributing activities under HPC environments.

6 Analysis

Once the scientific experiment is completely executed, scientists must select measurement data and analyse them to verify if their hypothesis is confirmed or refuted and draw conclusions about the scientific experiment.

When scientists discover that the initial hypothesis of an experiment is refuted, they (may) construct a new hypothesis for this experiment starting an entire new scientific workflow and the experimental life cycle continues. Nevertheless, if they find that their hypothesis was confirmed, they still may want to test it in a new scenario (using different configurations of parameters, for example) to assure that the experiment was correctly conducted.

Although the analysis phase is important for the scientists that are conducting the scientific experiments, it is also important to other scientists that intend to conduct similar experiments. Another possible scenario in this case is that other scientists may need to validate the conducted experiment. Thus, reproducibility emerges as the underpinning of the scientific practice, where scientists attempt to repeat the experiment to confirm the hypothesis. If a given experiment is not reproducible, it cannot be considered scientific. The term ‘scientific’ implicitly gives the idea that the experiment followed procedures, stored information that can be validated, reproduced and confirmed. In summary, reproducibility refers to the ability of an experiment to be accurately reproduced, or replicated, by someone else working independently, and producing the same results. At this point of the experiment life cycle, the analysis of provenance data plays a fundamental role. Retrieving and analysing provenance data are fundamental to guarantee reproducibility of the scientific experiment. This kind of analysis may also provide important documentation about the repeatability of the experiment (Simmhan et al., 2005; Cruz et al., 2009).

Provenance may help scientists to preserve the input data, to determine its quality and its authorship as well as to interpret and validate the associated scientific results. The next subsections present in detail the sub-phases that compose the analysis phase of the experiment life cycle: *query* and *visualisation*, and *discovery*.

6.1 Query and visualisation

Querying experiment results and provenance data offer insights about the scientific experiment. According to Ioannidis and Livny (1987), there are three types of queries that scientists may perform on results, which may also be extended to provenance data: preliminary, analysis and follow-up queries.

Preliminary queries – In principle, such queries need to deal with the full experimental frame of the study and must include the specification of the parameters values used in a given workflow, the version of an input data set, the version of the workflow, and some indication about the completeness of execution of the scientific experiment. In most cases, due to the huge amount of data that must be specified, such queries are time consuming.

Analysis queries – This kind of query is related to the relationships among data objects such as specified in the OPM model (Moreau et al., 2008). This includes both acyclic ancestry relationships and cyclic relationships of object identity (Holland et al., 2008). Such processing is invoked by applying domain specific actions, such as obtaining the difference between the runs of the same concrete workflow with data evaluating in time.

Follow-up queries – Due to the investigative nature of scientific experiments, scientists are stimulated to navigate through a multi-dimensional space of parameters that capture the behaviour of the observed phenomenon and compare their results.

Visualisation is another way of understanding scientific results. For example, instead of tediously tracing through large datasets, scientists may view results condensed as graphs or maps, and draw conclusions from these projected views. With visualisation, the visual human perception is used in place of complex data crunching algorithms, leading to easier detection of patterns hidden in huge amounts of data (Robertson et al., 1993).

Visualisation techniques together with provenance may allow scientific analysis and understanding of results generated from scientific experiment. Particularly when such tools are integrated in a single tool used to analyse and evaluate the final results, derivation processes, and any intermediate result derived during the processes (Del Rio and da Silva, 2007; Davidson and Freire, 2008).

In this paper we consider the visualisation as the result of queries over provenance data. Thus, we do not draw attention to visualisation techniques, but how to query and access the information stored as provenance.

6.2 Discovery

This sub-phase is one of the most important in the experiment life cycle, since it is responsible for deciding the conduction of the experiment as a whole. In this sub-phase, scientists have to evaluate queried and visualised provenance data (this data was already produced by the previous sub-phase) to draw conclusions about the entire experiment. These conclusions (as already mentioned in the introduction) may lead scientists to two different paths:

1 the result is likely to be correct

2 their hypothesis is refuted.

Although it may appear simple to draw these conclusions, scientists probably need a powerful computational infrastructure that allows them to analyse the scientific experiment as a whole, including all the scientific workflows that are part of the same experiment. For example, in the deep water oil exploitation domain there is an experiment that aims to calculate the fatigue of risers. Although the fatigue calculus is a self-contained scientific workflow by definition, it is dependent of results generated by two different scientific workflows: one to make the structural analyses of the platform and the other to make the structural analyses of each riser. These workflows may not be ‘merged’ in order to create a single workflow because between them there are many manual activities.

Although SWfMS see them as disconnected from each other, these workflows are strictly related since the results generated by the fatigue analysis workflow are based on results achieved by the previous workflows. When the concept of the scientific experiment is supported, these workflows can be related to each other, in a higher abstract level. In this fatigue analysis case, for example, scientists may need to know which parameters of the platform influenced a riser that generates a fatigue time bellow an acceptable value.

Other experiment based queries are: common research questions like: ‘which are the abstract workflows that could be reused to support a new scientific experiment?’, ‘what are the possible concrete derivations of an experiment?’, ‘which experiment trials have produced more discrepant results?’, ‘what are the alternatives for execution of a certain abstract activity?’, ‘how many workflows have been used to perform experiment Y?’, ‘what method was used to execute the activity X on day D’. This is only possible if queries can be submitted to the experiment level and not by querying different executions of disconnected scientific workflows.

Using current approaches, scientists are not able to make this type of provenance analysis to draw conclusions of the experiment. The existing SWfMS only stores, queries or visualises data provenance of each disconnected scientific workflow, thus related to executable resources and not the algorithms and other data from the entire experiment.

7 Discussions and conclusions

In this paper we proposed and described in detail a large scale *scientific* experiment life cycle. It contributes by presenting a better understanding of a large scale scientific experiment, decoupling the goals of an individual workflow execution from the goal of the whole experiment.

This paper clarifies some aspects of workflow representation, which demands support for different levels of abstractions, depending of the contexts of the experiment. The proposed life cycle was decomposed into three main phases, which are:

1 composition

2 execution

3 analysis.

One major difference of our life cycle is that we evidence the advantages of representing the scientific experiment in different levels of abstraction. The paper points out important facilities that are already provided by the existing SWfMS and many others that should be improved or even developed from scratch. Some of these aspects should be more carefully studied, including initiatives available worldwide for each scientific experiment life cycle.

The authors believe that by supporting provenance throughout all phases of the experiment life cycle, the provenance data will help scientists not only to analyse results to verify if their hypothesis can be confirmed or refuted, but also may assist them to protect the integrity, confidentiality, and availability of the relevant and fundamental information to reproduce the large scale scientific experiment and share its results (i.e., the complete history of the experiment).

Table 1 Challenges concerned with scientific experiments

Phase	*Sub-phase*	*Challenges*	*Existing solutions*	*Limitations*
Composition	Conception	Supporting different levels of abstraction	None	The levels of abstraction are mainly concrete and are not bound to the notion of the experiment. Its alternative workflows (variations and optional activities) are not registered as part of one experiment.
		Support for manual/ semi-automatic activities	None	The notion of the experiment is lost when an abstract workflow needs to be split into two or more concrete workflows, due to the existence of manual activities in the abstract representation.
		Providing semantics (ontologies)	myGrid	Concrete activities can be associated to ontologies. The notion of the experiment is not associated to ontological concepts.
	Reuse	Recommendation systems	VisTrails and Woods	Reuse is based only on concrete level. It lacks support for reuse of the experiment itself.
		Scientific experiment repository	myExperiment	The focus is on concrete level only. Executable workflows, mainly on Taverna, can be browsed, but it lacks knowledge of the experiment definition and its alternative workflows.
	-	Evolution control (config. management)	VisTrails	It is based on one current version. It lacks support for the configuration of the version management of the experiment and prospective provenance integration.
Execution	-	Implicit parallelism definition	Askalon, Swift	Data fragmentation and parameter sweep must be defined explicitly. These are characteristic of the experiment itself, not of a specific concrete workflow.
		Execution in heterogeneous environments	Swift, Pegasus, Askalon …	These systems can execute in multiple platforms, but lack support for provenance gathering as part of the notion of the experiment.
Analysis	-	Provenance analysis through the whole experiment life cycle	None	No provenance system is capable to query provenance from alternative workflows executions which are part of the same experiment. No system can query workflows based on the algorithmic definition of the workflow.

Note: *In this table we have not included our research team approaches that may assist these challenges.

The integration of all these scientific experiment phases into one abstraction is not a trivial task, and *ad-hoc* approaches do not seem to be acceptable. However, by using a set of systems and techniques that supports all phases of the life cycle for a scientific experiment we may answer some fundamental questions that transverse all the scientific workflows that compose the experiment. Table 1 presents a summary of the challenges inherent to large scale scientific experiments.

Although many interesting and innovative computational initiatives have been proposed in the last years regarding scientific experimentation, most of them focus on providing tools and mechanisms to deal with scientific workflows disconnected from the scientific experiment to which they are related. These initiatives are indeed an important step forward. However they do not solve the problem of dealing with the scientific experiment as a whole. New approaches are needed when the experiment is of large scale, and it will probably be one of the great challenges in research and a fertile field for the next years.

Acknowledgements

This work is partially sponsored by CNPq and CAPES.

References

Abramson, D., Bethwaite, B., Enticott, C., Garic, S., Peachey, T., Michailova, A., Amirriazi, S. and Chitters, R. (2009) 'Robust workflows for science and engineering', in *Proceedings of the 2nd Workshop on Many-Task Computing on Grids and Supercomputers*, Portland, Oregon, pp.1–9.

Altintas, I., Berkley, C., Jaeger, E., Jones, M., Ludascher, B. and Mock, S. (2004) 'Kepler: an extensible system for design and execution of scientific workflows', in *16th SSDBM*, Santorini, Greece, pp.423–424.

Barga, R.S., Fay, D., Guo, D., Newhouse, S., Simmhan, Y. and Szalay, A. (2008) 'Efficient scheduling of scientific workflows in a high performance computing cluster', in *6th International Workshop On Challenges Of Large Applications In Distributed Environments*, Boston, MA, USA, pp.63–68.

Berriman, G.B., Deelman, E., Good, J., Jacob, J.C., Katz, D.S., Laity, A.C., Prince, T.A., Singh, G. and Su, M. (2007) 'Generating complex astronomy workflows', *Workflows for e-Science*, Springer, pp.19–38.

Bose, R. and Frew, J. (2005) 'Lineage retrieval for scientific data processing: a survey', *ACM Computing Surveys*, Vol. 37, No. 1, pp.1–28.

Callahan, S.P., Freire, J., Santos, E., Scheidegger, C.E., Silva, C.T. and Vo, H.T. (2006) 'VisTrails: visualization meets data management', in *Proceedings of the 2006 ACM SIGMOD*, Chicago, IL, USA, pp.745–747.

Cavalcanti, M.C., Targino, R., Baião, F., Rössle, S. ., Bisch, P.M., Pires, P.F., Campos, M.L.M. and Mattoso, M. (2005) 'Managing structural genomic workflows using web services', *Data & Knowledge Engineering*, Vol. 53, No. 1, pp.45–74.

Conradi, R. and Westfechtel, B. (1998) 'Version models for software configuration management', *ACM Computing Surveys*, Vol. 30, No. 2.

Cruz, S.M.S.D., Campos, M. and Mattoso, M. (2009) 'Towards a taxonomy of provenance in scientific workflow management systems', in *IEEE International Workshop on Scientific Workflows*, Los Angeles, California, USA.

Dart, S. (1991) 'Concepts in configuration management systems', in *Proceedings of the 3rd International Workshop on Software Configuration Management*, Trondheim, Norway, pp.1–18.

Davidson, S.B. and Freire, J. (2008) 'Provenance and scientific workflows: challenges and opportunities', in *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, Vancouver, Canada, pp.1345–1350.

Dávila, A.M.R., Mendes, P.N., Wagner, G., Tschoeke, D.A., Cuadrat, R.R.C., Liberman, F., Matos, L., Satake, T., Ocaña, K.A.C.S., et al., (2008) 'ProtozoaDB: dynamic visualization and exploration of protozoan genomes', *Nucleic Acids Research*, January, Database issue, Vol. 36, pp.D547-D552.

Den, J. and Ghemawat, S. (2008) 'MapReduce: simplified data processing on large clusters', *Commun. ACM*, Vol. 51, No. 1, pp.107–113.

Deelman, E., Gannon, D., Shields, M. and Taylor, I. (2009) 'Workflows and e-science: an overview of workflow system features and capabilities', *Future Generation Computer Systems*, Vol. 25, No. 5, pp.528–540.

Deelman, E., Mehta, G., Singh, G., Su, M. and Vahi, K. (2007) 'Pegasus: mapping large-scale workflows to distributed resources', *Workflows for e-Science*, Springer, pp.376–394.

Del Rio, N. and da Silva, P. (2007) 'Probe-it! Visualization support for provenance', *Advances in Visual Computing*, pp.732–741.

Frakes, W. and Kang, K. (2005) 'Software reuse research: status and future', *IEEE Transactions on Software Engineering*, Vol. 31, No. 7, pp.529–536.

Freire, J., Koop, D., Santos, E. and Silva, C.T. (2008) 'Provenance for computational tasks: a survey', *Computing in Science and Engineering*, Vol. 10, No. 3, pp.11–21.

Gannon, D., Plale, B., Marru, S., Kandaswamy, G., Simmhan, Y. and Shirasuna, S. (2007) 'Dynamic, adaptive workflows for mesoscale meteorology', *Workflows for e-Science*, Springer, pp.126–142.

GExp (2009) *Large Scale Management of Scientific Experiments*, available at http://gexp.nacad.ufrj.br/.

Goble, C., Wroe, C. and Stevens, R. (2003) 'The myGrid project: services, architecture and demonstrator', *in Proc. of the UK e-Science All Hands Meeting*.

Goderis, A., De Roure, D., Goble, C., Bhagat, J., Cruickshank, D., Fisher, P., Michaelides, D. and Tanoh, F. (2008) 'Discovering scientific workflows: the myExperiment benchmarks', *IEEE Transactions on Automation Science and Engineering*.

Holland, D., Braun, U., Maclean, D., Muniswamy-Reddy, K. and Seltzer, M. (2008) 'Choosing a data model and query language for provenance', in *Second International Provenance and Annotation Workshop – IPAW*, Salt Lake City, UT, USA.

Hull, D., Wolstencroft, K., Stevens, R., Goble, C., Pocock, M.R., Li, P. and Oinn, T. (2006) 'Taverna: a tool for building and running workflows of services', *Nucleic Acids Research*, Web server issue, Vol. 34, pp.729–732.

Ioannidis, Y.E. and Wong, E. (1987) 'Query optimization by simulated annealing', in *Proceedings of the 1987 ACM SIGMOD International Conference on Management of Data*, San Francisco, California, USA, pp.9–22.

Jarrard, R.D. (2001) *Scientific Methods. Online book*, available at http://emotionalcompetency.com/sci/booktoc.html.

Koop, D., Scheidegger, C., Callahan, S., Freire, J. and Silva, C. (2008) 'VisComplete: automating suggestions for visualization pipelines', *IEEE Transactions on Visualization and Computer Graphics*, Vol. 14, No. 6, pp.1691–1698.

Lin, C., Lu, S., Fei, X., Pai, D. and Hua, J. (2009) 'A task abstraction and mapping approach to the shimming problem in scientific workflows', in *Proceedings of the 2009 IEEE International Conference on Services Computing*, Vol. 00, pp.284–291.

Livny, I., Ioannidis, Y., Livny, M., Haber, E., Miller, R., Tsatalos, O. and Wiener, J. (1994) 'Desktop experiment management', *IEEE Data Engineering Bulletin*, Vol. 16.

Martinho, W., Ogasawara, E., Oliveira, D., Chirigati, F., Santos, I., Travassos, G. and Mattoso, M. (2009) 'A conception process for abstract workflows: an example on deep water oil exploitation domain', in *5th IEEE International Conference on e-Science*, Oxford, UK.

McGough, A.S., Lee, W., Cohen, J., Katsiri, E. and Darlington, J. (2007) *'ICENI', Workflows for e-Science*, Springer, pp.395–415.

Medeiros, C., Vossen, G. and Weske, M. (1995) 'WASA: a workflow-based architecture to support scientific database applications', *Database and Expert Systems Applications*, pp.574–583.

Moreau, L., Freire, J., Futrelle, J., McGrath, R., Myers, J. and Paulson, P. (2008) 'The open provenance model: an overview', *Provenance and Annotation of Data and Processes*, pp.323–326.

Northrop, L. (2002) 'SEI's software product line tenets', *IEEE Software*, Vol. 19, No. 4, pp.32–40.

Ogasawara, E., Oliveira, D., Chirigati, F., Barbosa, C., Elias, R.N., Braganholo, V. and Mattoso, M. (2009a) 'Exploring many task computing in scientific workflows', in *2nd Workshop on Many-Task Computing on Grids and Supercomputers*, pp.1–10.

Ogasawara, E., Paulino, C., Murta, L., Werner, C. and Mattoso, M. (2009b) 'Experiment line: software reuse in scientific workflows', in *21th SSDBM*, New Orleans, LA, pp.264–272.

Ogasawara, E., Rangel, P., Murta, L., Werner, C. and Mattoso, M. (2009c) 'Comparison and versioning of scientific workflows', in *International Workshop on Comparison and Versioning of Software Models*, Vancouver, Canada, pp.25–30.

Oinn, T., Li, P., Kell, D.B., Goble, C., Goderis, A. and Greenwood, M., Hull, D., Stevens, R., Turi, D., et al. (2007) 'Taverna/myGrid: aligning a workflow system with the life sciences community', *Workflows for e-Science*, Springer, pp.300–319.

Oliveira, D., Cunha, L., Tomaz, L., Pereira, V. and Mattoso, M. (2009a) 'Using ontologies to support deep water oil exploration scientific workflows', in *IEEE International Workshop on Scientific Workflows*, Los Angeles, California, USA.

Oliveira, D., Ogasawara, E., Baião, F. and Mattoso, M., (2009b) 'Using ontologies to provide different levels of abstraction in scientific workflows', in *5th IEEE International Conference on e-Science*, Oxford, UK.

Oliveira, F., Murta, L., Werner, C. and Mattoso, M. (2008) 'Using provenance to improve workflow design', in *2nd International Provenance and Annotation Workshop – IPAW*, Salt Lake City, UT, USA, pp.136–143.

Pennington, D.D., Higgins, D., Peterson, A.T., Jones, M.B., Ludäscher, B. and Bowers, S. (2007) 'Ecological niche modeling using the Kepler workflow system', *Workflows for e-Science*, Springer, pp.91–108.

Pressman, R.S. (2004) *Software Engineering Software Engineering: A Practitioner's Approach*, 6th ed., McGraw-Hill.

Qin, J., Fahringer, T. and Pllana, S. (2007) 'UML based grid workflow modeling under ASKALON', *Distributed and Parallel Systems*, pp.191–200.

Raicu, I., Foster, I. and Zhao, Y. (2008) 'Many-task computing for grids and supercomputers', in *Workshop on Many-Task Computing on Grids and Supercomputers*, pp.1–11.

Raicu, I., Zhao, Y., Dumitrescu, C., Foster, I. and Wilde, M. (2007) 'Falkon: a fast and light-weight task execution framework', in *2007 ACM/IEEE Conference on Supercomputing*, Reno, Nevada, pp.1–12.

Robertson, G.G., Card, S.K. and Mackinlay, J.D. (1993) 'Information visualization using 3D interactive animation', *Communications of the ACM*, Vol. 36, No. 4, pp.57–71.

Scheidegger, C.E., Vo, H.T., Koop, D., Freire, J., Silva, C.T., (2008) 'Querying and re-using workflows with VsTrails', in *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, Vancouver, Canada, pp.1251–1254.

Simmhan, Y.L., Plale, B. and Gannon, D. (2005) 'A survey of data provenance in e-science', *ACM SIGMOD Record*, Vol. 34, No. 3, pp.31–36.

Soanes, C. and Stevenson, A. (2003) *Oxford Dictionary of English*, 2nd ed., Oxford University Press.

Stevens, R.D., Tipney, H.J., Wroe, C.J., Oinn, T.M., Senger, M., Lord, P.W., Goble, C.A., Brass, A. and Tassabehji, M. (2004) 'Exploring Williams-Beuren syndrome using myGrid', *Bioinformatics*, August, suppl_1, Vol. 20, pp.i303–310.

Stevens, R., Glover, K., Greenhalgh, C., Jennings, C., Pearce, S., Li, P., Radenkovic, M., Wipat, A., Tower, C., et al. (2003) 'Performing in silico experiments on the grid: a users perspective', in *Proc UK e-Science Programme All Hands Conference*, Nottingham, UK, pp.2–4.

Taylor, I., Shields, M., Wang, I. and Harrison, A. (2007) 'The Triana workflow environment: architecture and applications', *Workflows for e-Science*, Springer, pp.320–339.

Travassos, G.H. and Barros, M.O. (2003) 'Contributions of in virtuo and in silico experiments for the future of empirical studies in software engineering', in *Proc. of 2nd Workshop on Empirical Software Engineering the Future of Empirical Studies in Software Engineering*, Roma.

Verdi, K.K., Ellis, H.J. and Gryk, M.R. (2007) 'Conceptual-level workflow modeling of scientific experiments using NMR as a case study', *BMC Bioinformatics*, Vol. 8, p.31.

Walker, E. and Guiang, C. (2007) 'Challenges in executing large parameter sweep studies across widely distributed computing environments', in *5th IEEE Workshop on Challenges of Large Applications in Distributed Environments*, Monterey, California, USA, pp.11–18.

Wieczorek, M., Prodan, R. and Fahringer, T. (2005) 'Scheduling of scientific workflows in the ASKALON grid environment', *SIGMOD Rec.*, Vol. 34, No. 3, pp.56–62.

Yu, J. and Buyya, R. (2005) 'A taxonomy of workflow management systems for grid computing', *Journal of Grid Computing*, Vol. 34, Nos. 3–4, pp.171–200.

Zhao, Y., Dobson, J., Foster, I., Moreau, L. and Wilde, M. (2005) 'A notation and system for expressing and executing cleanly typed workflows on messy scientific data', *ACM SIGMOD Record*, Vol. 34, No. 3, pp.37–43.

Notes

1 A riser is a rigid or flexible duct that connects a section of sub-sea piping to the platform (Oliveira et al., 2009a).

2 Software product lines are software engineering methods, tools, and techniques for creating a collection of similar software systems from a shared set of software assets using common means of production (Northrop, 2002).

3 Software CM is a discipline for controlling the evolution of software systems (Dart, 1991).