Leveraging Semantics to Improve Reproducibility in Scientific Workflows

Idafen Santana-Perez*, Rafael Ferreira da Silva‡, Mats Rynge‡, Ewa Deelman‡,
María S. Pérez-Henández*, Oscar Corcho*

* Ontology Engineering Group, Universidad Politécnica de Madrid, Madrid, Spain
{isantana,mperez,ocorcho}@fi.upm.es
‡ University of Southern California, Information Sciences Institute, Marina Del Rey, CA, USA
{rafsilva,rynge,deelman}@isi.edu

I. Introduction

Reproducibility of published results is a cornerstone in scientific publishing and progress. Therefore, the scientific community has been encouraging authors and editors to publish their contributions in a verifiable and understandable way. Efforts such as the Reproducibility Initiative [1], or the Reproducibility Projects on Biology [2] and Psychology [3] domains, have been defining standards and patterns to assess whether an experimental result is reproducible.

In computational science, or *in-silico* science, reproducibility often requires that researchers make code and data available to others so that the data can be analyzed in a similar manner as in the original publication. Code must be available to be distributed, data must be accessible in a readable format [4]. However, applications have been growing in complexity and volume. Therefore, many scientists now formulate their computational problems as scientific workflows running on hosted computing infrastructures such as campus clusters, clouds, and grids [5]. Scientific workflows are an enabler of complex scientific analyses, composed of heterogeneous components, potentially designed by multiple scientists.

Workflows are a useful representation for managing the execution of large-scale computations. This representation not only facilitates the creation and management of the computation but also builds a foundation upon which results can be validated and shared. Since workflows formally describe the sequence of computational and data management tasks, it is easy to trace the origin of data produced. Many workflow systems capture provenance at runtime, which provides the lineage of data products and as such underpins the whole of scientific data reuse by providing the basis on which trust and understanding are built. A scientist would be able to look at the workflow and provenance data, retrace the steps and arrive at the same data products. However, this information is not sufficient for reproducibility.

The need for data and code sharing in computational science has been widely discussed [6]. Currently, most of the approaches in computational science conservation, in particular for scientific workflow executions, have been focused on data, code, and the workflow description, but not on the underlying infrastructure—which is composed of a set of computational resources (e.g. execution nodes, storage devices, networking) and software components. We identify two approaches for conserving the environment of an experiment, depending on how relevant this environment is, and the difficulty in obtaining an equivalent one: *physical conservation*, where the real object is conserved due to its relevance and the difficulty in obtaining a counterpart; and *logical conservation*, where objects are described in a way that an equivalent one can be obtained in a future experiment.

The computational environment (e.g. supercomputers, clusters, or grids) is often conserved by using the physical approach, where computational resources are made available for long time period to scientists who want to perform experiments. As a result, scientists are able to reproduce their experiments in the same environment. However, such infrastructures demand a huge maintenance efforts, and there is no guarantee that it will not change or suffer from a natural decay process [7]. Furthermore, the infrastructure may be subjected to organization policies, which restricts its access to a selective group of scientists, thus limiting the scalability of the reproducibility. On the other hand, data, code, and workflow description can be conserved by using a logical approach that is not subjected to natural decay processes.

Accordingly, we propose a logical-oriented approach to conserve computational environments, where the capabilities of the resources (virtual machines (VM)) are semantically described. From this description, any scientist, insterested in reproducing an experiment, will be able to reconstruct the former infrastructure (or an equivalent one) in any Cloud computing infrastructure (either private or public). One may argue that would be easier to keep and share VM images with the community research through a common repository, however the high storage demand of VM images remains a challenging problem [8], [9].

Our approach uses semantic-annotated workflow descriptions to generate lightweight scripts for an experiment management API that can reconstruct the required infrastructure. We propose to describe the resources involved in the execution of the experiment, using a set of semantic vocabularies, and use those descriptions to define the infrastructure specification. This specification can then be used to derive the set of instructions that can be executed to obtain a new equivalent infrastructure.

In this paper, we discuss how this approach could address some of the reproducibility issues identified in the computational science field [6], and expose, by a proof-of-concept experiment, how it has been applied to real scientific workflow executions.

II. Semantic Modeling

As pointed out by Gary King in 1995 [10], the replication standard *"only requires sufficient information to be provided in the article or book or in some other publicly accessible form so that the results could in principle be replicated"*. Almost 20 years later, we share this view and claim that this principle should be applied to execution environments in computational science.

Many efforts have been carried out to provide information about the scientific procedure of an experiment, by means of workflows and scientific communities for sharing them, and about the experimental data, both input data and results. We argue that for a complete and sufficient description of an experiment, information about the computational resources involved should be also provided. Providing these descriptions will allow the targeted audience, usually another scientist in the same domain, to understand the underlying components involved in the execution.

In this work, we argue that semantic technologies fit in this view as an standard, flexible, and integrable way for describing and disseminating information. They allow both, machines and humans, to read and understand what the resources are, and how they depend on each other. Based on this information, it is possible to define the set of steps to be executed for obtaining a new infrastructure, capable of re-executing a former experiment.

We propose the definition of semantic models for describing the main domains of a computational infrastructure, defining the taxonomy of concepts and the relationships between them. These models describe the software components, hardware specifications, and the computational resources available (in the form of VMs or cluster nodes). The models also describe the workflow, and how it is related to the different resources. As a result, this process facilitates experiment's reusability since new experiments, which may reuse parts of the workflow previously modeled, will benefit from the infrastructure dependencies already described.

We have identified four main domains of interest for documenting computational infrastructures. We have developed a set of models, one for each domain, describing their main concepts:

- *Hardware domain*: identifies the most common hardware information, including CPU, Storage, and RAM memory, and their capacities.
- *Software domain*: defines the software components involved on the execution. It includes the pieces of executable software (e.g. scripts, binaries, and libraries) used in the experiment. In addition, dependencies between those components and configuration information are also defined, as well as the required steps for deploying them.
- *Workflow domain*: describes and relates workflow fragments (a.k.a transformations) to their dependencies. Therefore, scientists can understand what are the relevant components for each part of the workflow.
- *Computing resources domain*: expresses the information about the available computing resources. In this domain, only virtualized resources are currently considered (i.e. VMs). It includes the description of the VM image, its provider, and specifications.

These models have been implemented as ontologies, conforming the WICUS Ontology Network [11], available online[1]. Figure 1 shows the top level ontology of the network, defining the inter-domain relations between the four aforementioned models.

Semantic techniques enable scientists to easily integrate and extend novel models, identified by new domains of interest, in a systematic way. In particular, the WICUS models are an ongoing effort as we are continuously evaluating its expressiveness, and adding new capabilities based on real experiment examples.

III. Addressing Reproducible Research

In response to the Yale 2009 Declaration [6], our semantic modeling approach addresses the following

[1] http://purl.org/net/wicus

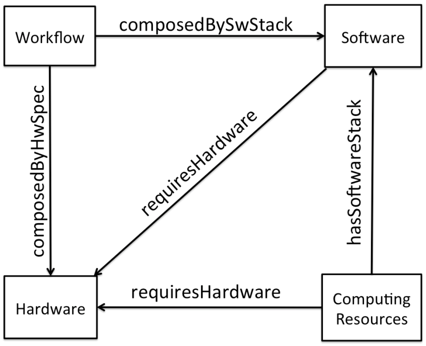

Figure 1. WICUS Ontology Network overview.

challenges on reproducible research in computational science:

- *"The bulk of the actual information required to reproduce results is not obvious from an article's text"*;
- *"...the complete software environment and the data generated those results- to be published along with the findings"*;
- *"Recommendation 3: Include a statement describing the computing environment and software version used in the publication"*.

Our semantic models capture the knowledge about the experiments, expressing the information about the workflow descriptions, and the relations and dependencies of the software components and the underlying infrastructure. As a result, these models enable the automation of the process of generating an equivalent execution environment.

- *"A VM Image with compiled code, sources, and data that can reproduce published tables and figures would let others explore the parameters around the publication point, examine the algorithm, and build on that work their own research"*.

The high storage needs of VM images (on the order of gigabytes) remains a challenging problem [8], [9]. Instead, we propose to describe the experiment resources using a set of semantic vocabularies, which can then be used to derive a set of instructions that can be executed to obtain a new equivalent infrastructure.

- *"Unfortunately, archived code can become unusable—sometimes quickly—due to changes in software and platform dependencies"*.
- *"Goal 9: ...Without maintenance, changes beyond individual's control (computer hardware, operating system, libraries, programming languages, and so on) will break reproducibility"*.

Semantical descriptions of software and infrastructure relations and dependencies empowers scientists to rapidly identify components that would be affected by changes on the infrastructure or software.

- *"Goal 6: require authors to describe their data using standardized terminology and ontologies. This will greatly streamline the running of various codes on data sets and a uniform interpretation of results"*.

In this work, we do not only propose to describe the data used in an experiment, but we also claim that by describing the software and infrastructure relations and dependencies, the reusability of an experiment would be significantly improved.

IV. Bringing the Semantic Vision to Fruition

Figure 2 shows an overview of the use case scenario in which we have applied the principles of our work. In this experiment, we performed a run of the Montage workflow [12] with the Pegasus Workflow Management System (WMS) [13] on FutureGrid [14]. From the execution logs, we generated annotations about the software components (binaries) and their dependencies, as well as the workflow by using the WICUS ontologies. We then removed all the binaries from the VM image and generated its corresponding annotations. This data represents an infrastructure specification, which is then translated into a PRECIP [15] script to build an equivalent infrastructure. The script describes the process to create a new virtual machine, to deploy the required binaries, and to re-execute the experiment.

One of the key points of our approach is to allow scientists to make the knowledge about their tools explicit and available. This knowledge is built upon the information about the available resources in an assisted manner. In this use case scenario, semantic annotations are generated from the resources descriptions provided by the Pegasus WMS. We summarize hereafter the main steps and components (tools and files) of the process used for the Montage workflow as shown in Figure 2:

1. DAX Annotator. This tool parses a DAX XML (Pegasus' workflow description) and generates a set of annotations, using the terms of the WICUS vocabulary, representing the steps of the workflow and its related infrastructure requirements.

2. Workflow annotations. An RDF file containing the description of the workflow and its infrastructure requirements.

3. Transformation Catalog Annotator. This tool queries the Pegasus Transformation Catalog (which describes the binaries for the Montage distribution and

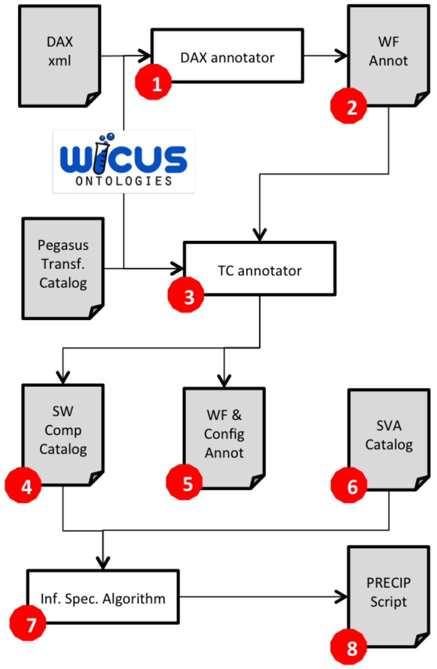

Figure 2. Overview of the reprodubility process for Montage.

their locations), to generate two set of annotations: the Software Components Catalog and the Workflow & Configuration Annotation file.

4. Software Components Catalog. An RDF file containing the set of annotations about the binaries, dependencies, deployment plans, and configuration information about the software from Montage. In this experiment, we identified 59 components from the Montage software toolkit. Only 11 out of the 59 components take part on the execution of the Montage workflow.

5. Workflow & Configuration Annotation File. An RDF file containing the same information as in 2, but enriched with the configuration information for each step, as specified in the transformation catalog.

6. Scientific Virtual Appliances Catalog. An RDF file describing available VM images. So far, we have only documented one virtual appliance, which is the same image as the one used in the original experiment (but without the Montage binaries).

7. Infrastructure Specification Algorithm. This process reads files 4, 5, and 6 and generates a configuration file (e.g. a PRECIP script), which describes VMs and software components to be created and deployed.

8. PRECIP script. This script creates a PRECIP experiment, which runs a VM, copies the required binaries, and executes deployment scripts to set the environment for the workflow execution. It also contains the PRECIP commands from the original experiment in order to re-execute it.

We have been able to reproduce the same output results from an execution where the Montage software was already in place, on a VM built from our semantic models. This first experimental result gives an insight on how semantic modeling can improve reproducibility in scientific workflows. Scripts and documentation used in this experiment are available online[2]

Acknowledgements

This material is based upon work supported in part by the National Science Foundation under Grant No. 0910812 to Indiana University for “FutureGrid: An Experimental, High-Performance Grid Test-bed” and FPU (“Formacion de Profesorado Universitario”) grant from the Spanish Science and Innovation Ministry (MICINN).

References

[1] Reproducibility initiative. http://reproducibilityinitiative.org.

[2] Reproducibility project: Cancer biology. http://osf.io/e81xl.

[3] Reproducibility project: Psychology. http://osf.io/ezcuj.

[4] V. Stodden, F. Leisch, and R. D. Peng, Eds., *Implementing Reproducible Research*. Chapman & Hall, 2014.

[5] I. Taylor, E. Deelman, D. Gannon, and M. Shields, *Workflows for e-Science*. Springer, 2007.

[6] (2009) Reproducible research: Addressing the need for data and code sharing in computational science. http://www.stanford.edu/~vcs/Conferences/RoundtableNov212009/RoundtableOutputDeclaration.pdf.

[7] M. Gavish and D. Donoho, “A universal identifier for computational results,” *Procedia Computer Science*, vol. 4, pp. 637 – 647, 2011, proceedings of the ICCS’11.

[8] X. Zhao, Y. Zhang, Y. Wu, K. Chen, J. Jiang, and K. Li, “Liquid: A scalable deduplication file system for virtual machine images,” *Paral. and Distr. Syst., IEEE Trans. on*, vol. PP, no. 99, 2013.

[9] B. Mao, H. Jiang, S. Wu, Y. Fu, and L. Tian, “Read-performance optimization for deduplication-based storage systems in the cloud,” *Trans. Storage*, vol. 10, no. 2, pp. 6:1–6:22, 2014.

[10] G. King, “Replication, replication,” *PS: Political Science and Politics*, vol. 28, no. 3, p. 443–499, September 1995.

[11] I. Santana-Pérez and M. S. Pérez-Hernández, “Towards reproducibility in scientific workflows: An infrastructure-based approach,” *IEEE Computing in Science & Engineering*, p. submitted, 2014.

[12] G. B. Berriman, E. Deelman, J. C. Good, J. C. Jacob, D. S. Katz, C. Kesselman, A. C. Laity, T. A. Prince, G. Singh, and M. Su, “Montage: a grid-enabled engine for delivering custom science-grade mosaics on demand,” in *SPIE Conference on Astronomical Telescopes and Instrumentation*, vol. 5493, 2004, pp. 221–232.

[13] E. Deelman, G. Singh, M. Su, J. Blythe, Y. Gil, C. Kesselman, G. Mehta, K. Vahi, and G. B. Berriman et al., “Pegasus: A framework for mapping complex scientific workflows onto distributed systems,” *Scientific Programming*, vol. 13, no. 3, 2005.

[14] Futuregrid. http://portal.futuregrid.org.

[15] S. Azarnoosh, M. Rynge, G. Juve, E. Deelman, M. Niec, M. Malawski, and R. Ferreira da Silva, “Introducing precip: an api for managing repeatable experiments in the cloud,” in *Workshop on Cloud Computing for Research Collab.*, 2013.

[2] http://pegasus.isi.edu/publications/xsede-reproducibility