Scientific workflows

From molecular biology and chemistry to astronomy, earth sciences and particle physics, modern experimental science increasingly relies on the acquisition, manipulation, and processing of large amounts of data, and the systematic orchestration of computationally intensive simulations and analyses. Much of the analysis is expensive, in terms of the data processing and storage resources required, sometimes running for weeks and needing careful monitoring. It is also laboriously repetitive as scientists explore different settings for their algorithms, data sets are continually updated from instruments, and new information prompts the whole "in silico" experiment to be rerun and cross-checked.

Over the past six years, workflow technology has been increasingly adopted by scientists in response to the need to specify and repeatedly execute these pipelines. A scientific workflow is the description of a process, often completely automated, that specifies the co-ordinated execution of multiple tasks. The tasks are software programs, run locally or remotely and increasingly published as Web services. Thus workflows are a particular form of scripted distributed computing over service oriented architectures.

As an example, Fig. 1 shows a workflow designed to be run by the Taverna Workflow Management System from a workflow library held at myExperment.org. This workflow is used by bioinformaticians. It obtains genes from a public database (Ensembl), annotates them with Entrez and UniProt identifiers, and uses another public database (KEGG) to find the biological pathways for each gene. In this case, the tasks, also known as *steps*, *nodes*, *activities*, *processors* or *components* (depicted as rectangles), represent either the invocation of a remote Web service (the databases), or the execution of a local program. Data flows along data links from the outputs of a task to the inputs of another, according to a pre-defined graph topology. The workflow defines how the output produced by one task is to be consumed by a subsequent task, a feature referred to as *orchestration* of a flow of data.

A workflow is an accurate description of the scientific plan. The benefits of workflows to scientists are the rapid repeatability of complex operations through automation, an explicit and precisely accurate record of a scientific procedure that leads to more transparent and reproducible science, a way of reusing others' applications, resources and workflows, and a means to avoid the details of executing a third party application.

Workflows are executed by a Scientific Workflow Management System (SWfMS). There is no SWfMS adopted by all, and likewise there is no one workflow language for describing workflows. Over 50 different systems are routinely used, either open source software such as Taverna, Triana, Kepler and Knime or as commercial products such as Pipeline Pilot and InforSense KDE.

Workflow models

The workflow model describes the exact behaviour of the workflow when it is executed. The features of different models are dictated by the types of application domains that they are designed to support, as well as by the target user community. The models vary in several aspects:

- *Computation*: Dataflow pipelines dictate that each processor be executed as soon as its data inputs are available, and processors that have no data dependencies amongst each other can be executed concurrently. They are used for integrating data from different sources, data capture, preparation and analysis pipelines, and populating scientific models or data warehouses. Control flows directly dictate the flow of process execution, using loops, decision points etc. These are used for controlling multiple sweeps of different parameter settings for simulations. Kepler supports several different models of computation. Others layer control flow on top of data flow or are mono-model.

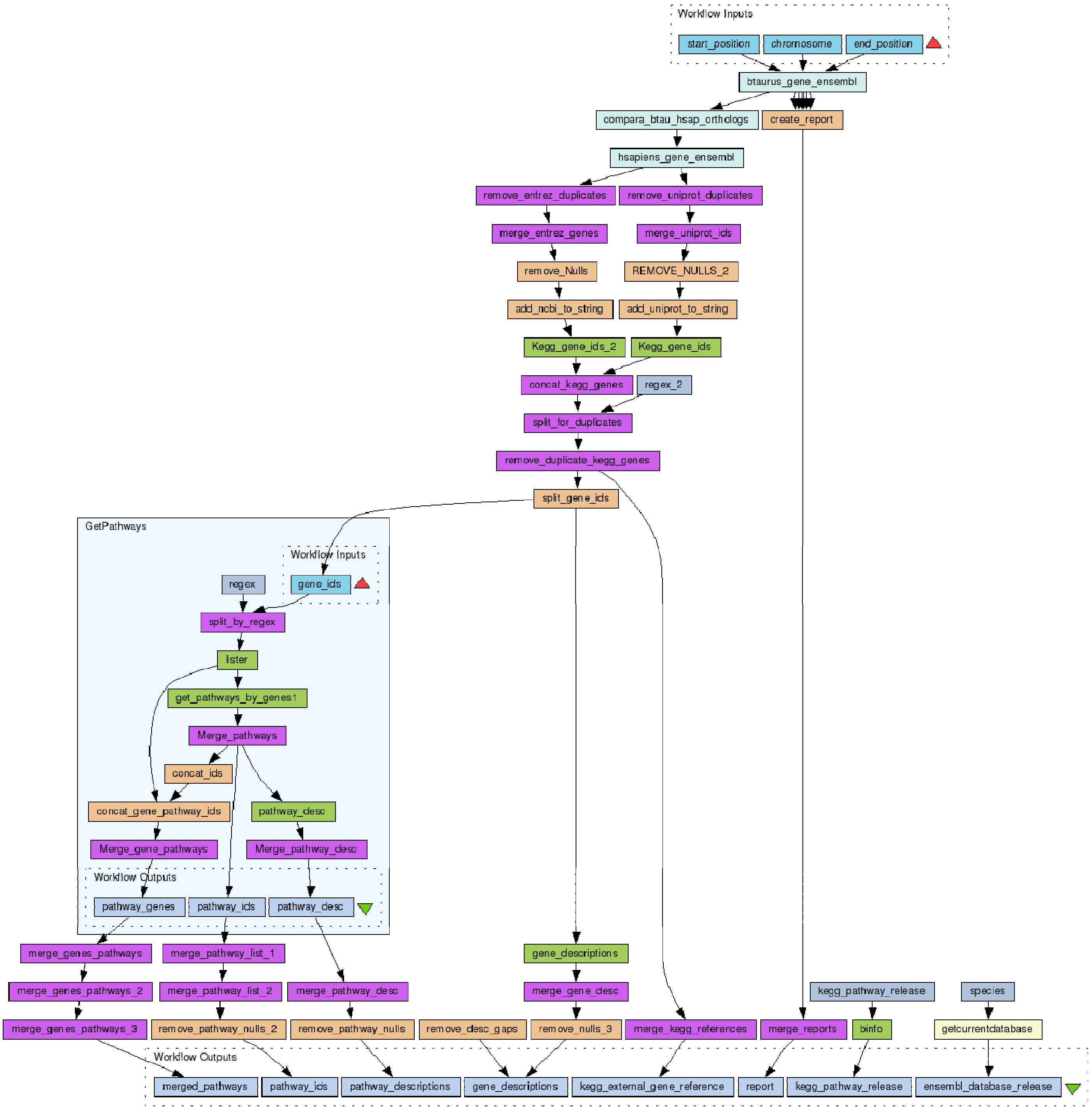

Fig 1: A Taverna workflow that chains together several searches over different publicly available databases in the Life Sciences. From P. Fisher et al., 2007.

- *Resource type:* The tasks can be high level application programs familiar to the scientist, as in the Taverna system or low-level scheduling and monitoring of jobs on a Grid or compute cluster such as the Pegasus system. Given a workflow description and a set of resource requirements (such as memory, processing power, operating system, and system libraries), Pegasus generates an appropriate efficient, executable workflow. The logic design of the workflow is decoupled from its execution environment, so as the pool of available resources changes, re-assignment of the resources is hidden from the user.
- *Compatibility:* Tasks may have been designed a priori to be compatible or designed independently and be incompatible without transformation sub-steps (called shims).

- *Interactivity*: Workflow execution could be wholly automatic or interactively steered by the scientist.
- *Adaptivity*: The workflow design or instantiation can be dynamically adapted “in flight” by the scientist or by automatically reacting to changed environmental circumstances.
- *Abstraction*: Most workflow models allow for some form of modular composition of workflows, for example, through hierarchical nesting.

Two scenarios illustrate the range of operational contexts for scientific workflows. The Pegasus system was used to manage computationally greedy earthquake simulations. A few developers in this planned collaborative project produced the necessary workflow designs then used by many scientists. The tasks are from a constrained set of codes incorporated with rigorous preparation procedures, mapped to run over a Grid. The workflows, once run, were not repeated. Another project used the Taverna system to investigate parasite resistance in cattle by linking metabolic pathway and gene expression public datasets. The workflow designs were developed directly by a single scientist in an exploratory, iterative way. The components were unprepared third party publicly accessible databases executed at their host site. Workflows were constantly rerun as the underlying datasets were updated.

All SWfMS aim to make their model intuitive for user scientists to understand, and to provide them with a language and facilities for workflow design without requiring a specific competence in computer programming. Striking a balance between simplicity and expressivity remains a challenge. Part of the unpopularity in Science of the BPEL (Business Process Execution Language), despite its status as an industry standard for orchestrating web services, is its complexity and the lack of available good free software for designing its workflows.

SWfMS Components supporting the Workflow Life Cycle

SWfMS are software environments that provide users with a number of functionalities to manage the complete life cycle of a workflow (Fig 2). Legacy code or new applications may need to be adapted in preparation for use as components in a workflow.

- *Workflow design* is typically supported through a graphical user interface which allows users to compose workflows, either from scratch by wiring together individual components, by using workflow templates, or by reusing and composing existing workflows;
- *Workflow planning* includes the validation of workflow correctness, i.e., by static type checking, allocation of resources to tasks (for instance, processing nodes on a Grid cluster), task scheduling, various types of optimisations, and staging of input data prior to its processing on the nodes, as well as delivery of output to the user;
- A workflow is executed using an *enactment engine*, which is a runtime environment similar to those that are normally associated with traditional programming languages. The engine controls the order of invocation of each of the tasks, monitors their execution, reacts in case of failures, manages the data and control flow and logs operational metrics such as processor execution time and data transfer time;
- *Post mortem analysis* of the workflow execution designed to help users understand and interpret the results produced by the workflow, as well as the details of its execution. Minimally, the SWfMS offers a debugging facility and gathers performance metrics. At a higher level, a *provenance subsystem* keeps an accurate, historical record of the execution of the workflow. Provenance metadata traces the lineage between input data, intermediate data and final results, and keeps an auditable record of the processes used, their configuration and their order.
- *Storage, collaboration and sharing* of components, workflows, patterns and templates through *libraries*, *warehouses* or *registries*. An example of a cross-system, public workflow repository is myExperiment.org, which was inspired by social networking sites. Discovering components and prior workflow designs needs semantically-rich metadata.

A SWfMS is typically run over middleware that provides infrastructure for accessing the applications or resources consumed by the workflow, and facilities like security and access control.

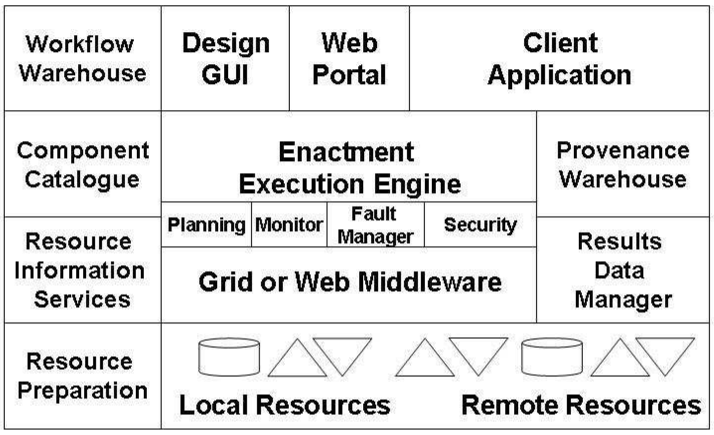

Fig 2: The high level components of a Scientific Workflow Management System.

Challenges

A number of aspects of a workflow's life cycle pose challenging problems.

Simplifying design: designing (reusable) workflows is difficult and takes skill and time. Languages need the salient features of the intended scientific analysis, whilst removing as much of the complexity as possible. One approach introduces multiple process description languages specialised according to different application domains, expressed at a level of abstraction that users are comfortable with, and supported by dedicated visual editing environments. These specifications are then translated into actual executable workflows.

Dynamic adaptation: dynamic changes to the workflow are necessary to deal with intermittently available resources, failures (especially for workflows that rely on external services for their execution), and user-based decisions that are based on observed intermediate results. Users should be able to re-execute portions of the workflow while modifying the data sources, the execution systems, and the workflow structure itself and exploit opportunities to reuse the results of past task executions rather than running the computation again.

Sharing and publishing: As workflows are rich, scientific protocols that represent expert know-how, they need to be pooled, published and curated just like scientific data, and accompany the articles that arise from their use for truly reproducible science. Specific problems are managing workflow versions, and making it possible for users to repeat the execution of a workflow, and compare results across large time spans (i.e. years). Social issues include managing intellectual property and credit, and peer review of workflows for accuracy and credibility.

Outlook. The co-existence of multiple workflow management systems is expected to be the norm. Rather than devising a single language or model, the community is working towards operational and provenance interoperability. This raises numerous challenges.

Carole Goble; Paolo Missier; David De Roure

Key Words: enactment; grid; process; provenance; science; service; workflow.

Bibliography

Ewa Deelman, Yolanda Gil. Managing Large-Scale Scientific Workflows in Distributed Environments: Experiences and Challenges, Workflows in e-Science, e-Science 2006, Amsterdam, December 4-6, 2006. DOI 10.1109/E-SCIENCE.2007.29

David De Roure, Carole Goble and Robert Stevens, Designing the myExperiment Virtual Research Environment for the Social Sharing of Workflows. e-Science 2007 – Third IEEE International Conference on e-Science and Grid Computing, 2007. Bangalore, India, 10-13 December 2007. Pages 603-610. DOI 10.1109/E-SCIENCE.2007.29

Paul Fisher, Cornelia Hedeler, Katherine Wolstencroft, Helen Hulme, Harry Noyes, Stephen Kemp, Robert Stevens and Andrew Brass. A Systematic Strategy for Large-Scale Analysis of Genotype-Phenotype Correlations: Identification of candidate genes involved in African Trypanosomiasis. Nucleic Acids Research 2007 35(16):5625-5633; DOI 10.1093/nar/gkm623

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

Taylor, I.J., Deelman, E., Gannon, D.B., Shields, M. (Eds.) Workflows for e-Science. Springer 2007.

Additional Reading

Concurrency and Computation: Practice and Experience 18(10) pp: 1067-1100, (Aug 2006) Special Issue: Workflow in Grid Systems. Issue edited by Geoffrey C. Fox, Dennis Gannon.

Goble CA and Ludascher B, *Special Issue on Scientific Workflows* ACM SIGMOD Record, 34(3) Sept 2005.

Chris Wroe, Carole Goble, Antoon Goderis, Phillip Lord, Simon Miles, Juri Papay, Pinar Alper, Luc Moreau Recycling workflows and services through discovery and reuse. Concurrency and Computation: Practice and Experience 19(2) pp: 181 – 194, February 2007.

Antoon Goderis, Christopher Brooks, Ilkay Altintas, Edward A. Lee and Carole Goble. Composing Different Models of Computation in Kepler and Ptolemy II Proc. of the 2nd Int. Workshop on Workflow Systems in e-Science (WSES 07) in conjunction with the Int. Conference on Computational Science (ICCS) 2007, Beijing, China, May 27-30, 2007.

Concurrency and Computation: Practice and Experience Volume 20 Issue 5, Pages 409 - 597 (10 April 2008) Special Issue: The First Provenance Challenge. Issue Edited by Luc Moreau, Bertram Ludäscher.