e-BioFlow: Improving Practical Use of Workflow Systems in Bioinformatics

Ingo Wassink[1,6], Matthijs Ooms[1,6], Pieter Neerincx[2,6], Gerrit van der Veer[1,3,6], Han Rauwerda[4,6], Jack A.M. Leunissen[5,6], Timo M. Breit[4,6], Anton Nijholt[1,6], and Paul van der Vet[1,6]

[1] Human Media Interaction Group, University of Twente, the Netherlands
[2] Biomolecular Mass Spectrometry and Proteomics Group, Utrecht University, the Netherlands
[3] Open University, Heerlen, the Netherlands
[4] Microarray Department and Integrative Bioinformatics Unit, University of Amsterdam, the Netherlands
[5] Laboratory of Bioinformatics, Wageningen University and Research, the Netherlands
[6] Netherlands Bioinformatics Centre, Nijmegen, the Netherlands

Abstract. Workflow management systems (WfMSs) are useful tools for bioinformaticians. As experiences with using WfMSs accumulate, shortcomings of current systems become apparent. In this paper, we focus on practical issues that hinder WfMS users and that arise in the design and execution of workflows, and in access of web services. We present e-BioFlow, a workflow engine that demonstrates in which way a number of these problems can be solved. e-BioFlow offers an improved user interface, can deal with large data volumes, stores all provenance, and has a powerful provenance browser. e-BioFlow also offers the possibility to design and run workflows step by step, allowing its users an explorative research style.

1 Introduction

Today, workflow management systems (WfMSs) are recognised as useful tools for chaining computational tasks [1, 2] and in particular for orchestrating web services [3, 4]. Open-source WfMSs for scientific computation (e.g., Kepler [5] and Triana [6]) and specifically for bioinformatics (e.g., Taverna [7]) enjoy worldwide use. Several companies also sell proprietary WfMSs, but the bioinformatics community tends to standardise on open-source systems.

In the course of developing several bioinformatics workflows, including a workflow for demonstrating the use of R in Taverna [8] and OligoRAP (to be discussed below), we have run against a number of issues that stand in the way of smooth use of WfMSs in daily practice. Our experiences complement those of [2] and [4]. To demonstrate that many of the problems we found can be solved, we designed and built our own workflow system called *e-BioFlow* [9]. e-BioFlow pays particular attention to user interaction and provenance. e-BioFlow moreover enables the user to build and run workflows step by step. Bioinformaticians

can experiment with their workflows until they are satisfied and only then store them. The combination of the ability to handle large data volumes, improved user interaction, and the feature of designing a workflow in an ad-hoc fashion, inspecting all data at all times, is, we believe, unique. e-BioFlow is based on an open-source workflow engine that has proven its worth in practice, YAWL [10]. e-BioFlow is itself an open-source experimentation platform.

Summarising our experiences, we found user interaction provided by existing tools cumbersome, in design, in execution, and afterwards when provenance has to be accessed. We also found it necessary to take extra measures to handle gigabytes of data flowing through the system. Running times can be long for bioinformatics workflows; most of the workflows we ran take hours to complete. This makes working with most current workflow tools laborious. One has to re-run the entire workflow when one wants to change parts of the workflow or when an unexpected termination occurs, for example the system crashes or an essential external resource proves to be unavailable. We realised that we can solve the latter problems by building an elaborate mechanism for storing and retrieving provenance, which in itself is a necessary component of any workflow system for bioinformatics. We avoided many problems associated with web services by using only BioMoby [11] services on databases that were mirrored in-house. Nevertheless, we know that web services pose problems of their own and we therefore address such problems in this paper as well. Finally, we wanted to profit from the wealth of experience with workflows in the business administration community.

A typical example of the kind of bioinformatics workflows we address in this paper is OligoRAP, used for automatic re-annotation of transcriptomics probe libraries [12, 13]. A high-quality oligonucleotide probe library is an essential component of genome-wide microarray-based gene-expression experiments. In order to maintain the quality of the information associated with the probes, this information has to be updated when new sequence or annotation data is released. An OligoRAP client orchestrates BioMoby web services to automatically check the target specificity of the oligonucleotide probes and update their annotations. The client calls nucleotide alignment search tools such as BLAT [14] and BLAST [15], fetches annotations provided by the Ensembl [16] project and performs a quality assessment. An OligoRAP run is not wholly linear; for example, the BLAT service is iteratively polled for output and the BLAST service is only called under certain conditions. The result of an OligoRAP run consists of XML files that provide detailed information per probe and a quality assessment of the whole library.

2 Definitions

We will first lay down our terminology. A *workflow* is a representation of a coherent collection of several *tasks* to execute a procedure. At the lowest level, tasks (Taverna says "processors") are executed in a predefined order and with a preset data exchange. This may not be obvious at a higher level, as when, for

example, tasks are executed in a parallel interleaved routing scheme. In such a scheme, it would appear that a task t_i is executed repeatedly with a different set of data, while the next task t_{i+1} starts for each set of data output by t_i. Under the hood, however, t_i is not a single task: it is copied as many times as needed to process all data sets and every copy is executed in a predefined order.

A *workflow management system* (WfMS) is a software environment to design and run workflows. When the workflow is run or (as jargon has it) *enacted*, a concrete instance of the workflow is created and executed. Design of a workflow and enactment can in principle be done by different WfMSs as long as both systems understand the language used to represent the workflow. In practice, design and enactment are almost always done within the same WfMS. If we want to interleave design and execution, using the same WfMS for both is of course the only practical solution. A *data pipeline* is a type of workflow in which the flow of data is emphasised, and besides the control imposed by the data flow, there is little or no other control. *Workflow patterns* [17] are generalizations of often-occurring combinations of tasks and how they are executed in relation to each other in a workflow.

Each task in a workflow can be *composite* or *atomic*. If a task is composite, it is a workflow itself. By using composite tasks, the workflow designer can ease the design and interpretation of a workflow. Such workflows are also known as *hierarchic workflows*. Tasks can invoke *resources*, which are by definition extraneous to the workflow. Examples of such resources are: scripts, web services, and programs that govern user interaction.

A workflow can be offered to others for use as such or with some modification. In this way, the bioinformatics community can share and mutually assess their workflows. This concept is promising, as the success of the myExperiment (`www.myexperiment.org`, [18]) shows.

3 Workflow Use in Other Domains

The use of WfMSs in business administration [19] predates that in bioinformatics by at least a decade and there is a wealth of practical experience with such systems. Business administration WfMSs are used to (partly) automate fixed administrative procedures like that of approval of an application for a bank account. The emphasis is on control. In this tradition, the need for formal models underlying WfMSs has been stressed [20] and a repertoire of workflow patterns has been compiled [17] to assess the expressivity of different workflow languages, to understand the differences between workflow engines, and to serve as a set of benchmarks for comparing the performance of different WfMSs. These patterns mainly concern synchronisation. For example, a task can only start when all previous tasks have finished, or when at least one previous task has finished, or when precisely one previous task has finished. Similar patterns can be defined for when a task finishes: will it start all tasks downstream, at least one task, or precisely one task? We may want to impose conditions on the synchronisation links. More complicated patterns involving, for example, iteration, cancellation,

and interleaved parallel execution are defined, too. By way of good practice, any WfMS should be able to support and formally validate at least each of the relevant patterns for the domain it addresses.

It is tempting to profit from the experience gained with business administration WfMSs, but in bioinformatics the prevailing trend is to implement from scratch. Nonetheless, commercial business administration WfMSs such as Microsoft's BizTalk have successfully been used for designing and running bioinformatics workflows [21].

A second strand is the use of workflows for Grid computing, i.e., massive, data-intensive and collaborative computations that are parallelised over clusters of computers. Taverna has been developed in the course of the UK myGrid-project and Kepler and Triana have originally been designed for use on grids for heavy scientific (such as astronomical, meteorological, or geological) computations. Kepler and Triana can handle massive workloads, which are several orders of magnitude larger than what is now common in bioinformatics [2]. Although originating from a Grid project, Taverna has focussed on applications in the bioinformatics domain.

A third strand of workflow use is in Laboratory Information Management Systems (LIMSs). The data model of a LIMS is determined by the nature and sequence of wet-lab processes for which it has to capture the data and metadata. A WfMS is a suitable tool for data collection, in particular now automatic registration devices make their way into the laboratory. Therefore, each piece of laboratory equipment can communicate with a computer system to hold the (meta)data and produce an automatic lab journal. Such LIMSs can then be integrated with the WfMSs used by bioinformaticians for their *in-silico* experiments. One step further, but still based on workflows, is the idea to automate the wet-lab experimentation process wholly or partially [22]. In these WfMSs, the data aspect of Grid computing and the control aspect of business applications are both prominent.

4 Workflow Topics

Returning to the themes identified in the introduction, we have encountered the following topics over the past years of practical experience with workflows and WfMSs in bioinformatics. We group them into three categories:

- *Design*
  - (a) User interface for bioinformaticians.
  - (b) Verification tools for checking properties of the workflow.
  - (c) Enactment of parts and individual tasks of a workflow.
- *Provenance*
  - (d) Handling large data volumes.
  - (e) Storing provenance.
  - (f) Accessing provenance after the workflow has been enacted.
- *Web services*
  - (g) Finding the appropriate web service for the job at hand.

- (h) Quality of service of a web service.
- (i) Interoperability of web services.
- (j) Quality of content of a web service.

e-BioFlow pays particular attention to *Design* and to *Provenance*. In the remainder of this survey, we will discuss these topics. This list reflects our own experiences and is by no means exhaustive. Moreover, as bioinformaticians continue to design and enact workflows, other experiences will come in.

5 Design

Workflows are normally designed using a graphical user interface, in which the workflow is represented as a graph. The interface hides the representation of the technical details of the workflow. In the graphs, the tasks are the nodes of the workflow, while their relations (data flow, synchronisation) are the edges. Most design interfaces allow the user to drag and drop tasks onto the workflow pane and to connect them by means of a few mouse clicks. Still, designing a good workflow can be hard; see [23] for a more theoretical overview of design issues. A typical bioinformatician will primarily pay attention to the flow of data, but at the same time control aspects play a role [24]. Besides, the choice of resources invoked by tasks has to be addressed. The result is often information overload.

In the development of e-BioFlow, we have paid a lot of attention to the interaction with the system. We have improved the user interface, we have enabled formal verification of the workflow and, perhaps most importantly, e-BioFlow can be run as an ad-hoc WfMS, allowing the user to explore all kinds of tasks, task orderings and parameter settings. This makes e-BioFlow suited even for once-only WfMS experiments, a feature that is unique.

(a) The user interface for design

The user interface for design should take the prevention of information overload into account by not trying to squeeze all information onto a single screen. e-BioFlow has three design views, one for each aspect: control (synchronisation and iteration), data flow, and choice of resources [25]. The user can easily switch between views. Each view shows a pane that can be changed by the user. The panes are related because the same tasks are present in each of them. To help orientation, tasks are always in the same place on each of the three panes.

The three panes are related in another way as well. When we draw a data link between two tasks, this normally implies a control link, too. The data link stipulates that the task downstream needs the data output by the task upstream and therefore has to wait for the upstream task to finish. The converse is not true. A control link can be just that: a signal that a task has to wait for another task to finish even though no data are passed. As a result, every edge drawn in the data flow pane should automatically insert an edge in the control pane while an edge drawn in the control pane does not introduce an edge in the data pane. A data-oriented bioinformatician can start in the data flow pane, to switch to the control pane when the data relations are in place.

(b) Verification tools

Because a workflow is designed to run to completion when enacted, it is important to guard for deadlock and livelock (infinite loops). Another problem is tasks that will not be executed no matter how the workflow is enacted. Such tasks are called dead tasks (not to be confused with dead web services). The presence of a dead task in a workflow can indicate a workflow design error. Checking a workflow for deadlock, livelock and dead tasks can be done manually, but when the workflow gets large this becomes time-consuming, difficult and error-prone. It is far better to follow the tradition of business administration workflows and have an automatic check for such irregularities. This can be realised only when the workflow is represented in an underlying process algebra. Through the work of van der Aalst and co-workers [20], the Petri net formalism enjoys widespread popularity but there are other process algebras that will do the job. Kepler, for example, is based on the dataflow process network formalism [5, 26]. e-BioFlow is based on YAWL [10] which itself is a formally correct extension of Petri Nets.

(c) Enactment of parts and individual tasks

A large problem facing designers is that in the current generation of WfMSs, the workflow has to be designed fully from start to finish before it can be enacted. A workflow engine such as Taverna does allow individual tasks to be tested, but it is not possible to incrementally add further tasks, testing underway all the time. Design will normally proceed through several design-enact-debug cycles. For a once-only workflow, the investment of designing a workflow will not pay off. Even for workflows that are to be used several times, the investment may be too large because it is impossible to test parts of the workflow. Finally, researchers very often only know what their workflow should do when they have had the opportunity to look at the data and to play with parameter settings, different algorithms, and different ways to visualise results. For them, the obligation to design a complete workflow before it can be enacted makes a WfMS the wrong tool for the job.

Such problems can be solved by means of a so-called *ad-hoc workflow management system*, also known as an adaptive WfMS [27]. In e-BioFlow, which can be run as an ad-hoc WfMS, design and enactment are interwoven. Each task and each part of the workflow can be run individually. Input data can be provided if upstream tasks have been run or if we have kept data from a previous attempt. Based on the data produced by the previous task, a decision can be made for the next task. Thus, an in-silico experiment can be run on a trial-and-error basis, which means to a bioinformatician the possibility to explore the data. This is useful for once-only experiments, but it is also a good way to design workflows that have to be run repeatedly.

In a teach-back survey [28], most of the 50 bioinformaticians with workflow experience we interviewed found the idea of an ad-hoc WfMSs appealing [29].

6 Provenance

The second cluster of problems we ran into was related to the huge volumes of data that the WfMS has to be able to handle in real-life bioinformatics cases. An OligoRAP workflow that handles a realistic amount of oligonucleotide probes, say, 20,000 probes, has to cope with approximately 3 GB of data. Business administration WfMSs, typically dealing with few data, not only pass the data as such, but also copy data between tasks. If OligoRAP were implemented that way, we would have needed a special machine to run it. However, we wanted to run OligoRAP on a commodity PC. With such amounts of data, moreover, storing and in particular accessing provenance is a challenge as well.

(d) Handling large data volumes

e-BioFlow solves the data problem by storing data in a database as soon as they are produced. When a task needs data, it fetches its data from the database by means of an SQL query. The workflow only passes pointers to the data. This is known as *pass by reference* [13]. Apart from freeing the computer system from handling massive amounts of data, pass by reference also ensures that the data are kept in a safe place in case the computer system unexpectedly breaks down. It may be argued that storage and retrieval of data makes the WfMS slower. We found this effect to be marginal or non-existent. When no provisions are made for large volumes, the system gets slow because it has to update memory and perform swaps, if the workflow can be run to end at all.

The way of working of e-BioFlow carries additional advantages. Provenance is automatically recorded. Ad-hoc design needs data at every step. The database makes ad-hoc design possible. Finally, the database can be used as cache. We will discuss these issues now.

(e) Storing provenance

The importance of provenance has been stressed by various authors [30–32]. In the context of WfMSs, provenance is a complete record of a single enactment of a workflow and comprises the registration of all data passed between tasks and all relevant metadata, resulting in a complete trace of the workflow enactment. This relates to the importance of verification and reproducibility for bioinformatics experimentation [33–35]. As we said above, e-BioFlow stores all data in a database by default. For provenance purposes, the data are decorated with metadata that provide information about the task that produced them and further details (date/time, information about the resource accessed, such as version number and URL, and similar). In this way, provenance collection is an intended side-effect of the design of e-BioFlow. The design also guarantees that provenance is complete.

Ad-hoc design is impossible without storage of results of previous steps. Moreover, these results should be immediately accessible at all times in the ad-hoc process. The database takes care of both requirements.

The database can additionally be used as cache. This idea was first proposed in the context of the Kepler system under the name of smart reruns [36]. Results of computationally demanding tasks can be stored in the cache and reused. Reuse is advantageous in case of an error downstream in the workflow or when we run a workflow repeatedly and know that certain tasks will produce output that is not changed with respect to the last time the workflow was run. e-BioFlow offers the option to treat certain data as cachable, so that they can be reused without having to re-run the task(s) that produced them.

(f) Accessing provenance

An important requirement for provenance is a data model, to extract the appropriate information from a provenance file with relative ease. The Open Provenance Model (OPM) [37] has been developed in a series of workshops with representatives of, among others, the most popular WfMSs. OPM translates the workflow and the data produced into a graph structure. When data is stored in a SQL database, the requirements of OPM can be met by installing a front-end that can translate OPM queries into SQL queries and that translates the data returned by the query into OPM format. These translations are not yet standardised. We need a so-called *OPM Profile* to lay down how tasks, edges, and actors are translated into OPM concepts. An OPM Profile has been published as [38]. To our knowledge, this is the first published OPM profile. For e-BioFlow, we designed our own profile which maps OPM onto the database schema of our provenance database and *vice versa*.

e-BioFlow is equipped with a special graphical browser for accessing provenance data after the workflow has run. It uses symbols that are de facto standards in the OPM community. The graphical browser faces challenges due the the large number of nodes and edges in the provenance graph. In OligoRAP and in similar, real-life size cases, the OPM graph has several hundred thousands of elements and relations. Such a graph can be navigated provided the workflow designer has carefully chosen the hierarchy of the workflow, so that the branching factor is low and the tree is rather deep. e-BioFlow thus uses the hierarchy of the workflow itself to facilitate access to provenance data. Biton and co-authors propose to automatically generate user views on provenance graphs [39].

7 Web Services

For many bioinformaticians, orchestrating web services has been and still is the primary reason to use workflows. Web services are invoked by means of one of several invocation methods. SOAP/WSDL, SoapLab, BioMart and BioMOBY are the most popular in bioinformatics; the latter three are even designed specifically for bioinformatics purposes [3]. To evaluate the practical use of web services in workflows, we have built a Taverna workflow that collects all Taverna workflows offered at the myExperiment website at a given time, expands all composite

tasks, and groups tasks into various categories [40]. We did such a survey in December, 2008, counting the number of web service invocations per invocation method. Remarkably, the majority of all tasks in this set of workflows did not invoke a web service at all. Web service invocation apparently assumes a host of helper tasks. SOAP/WSDL was by far the most preferred method, being used in 66% of the cases. SoapLab, developed by EBI specifically for bioinformaticians, scored just 24%. BioMOBY, which as we experienced ourselves is easier to share than your toothbrush [11], and BioMart scored even less. Pure REST invocation [41, 42] is very rare. Yet the original version of YAWL relies on REST [43]. The advantages of REST over SOAP/WSDL are said to include speed, scalability, security, and tool support [43, p. 149].

(g) Finding appropriate web services

Finding the appropriate web service not only involves finding a URL for a WSDL file or similar, but also requires being convinced that the web resource maintainers ensure a constant, high quality of service. The issue was earlier addressed by initiatives such as the discontinued UDDI. One of the reasons for the popularity of Taverna is that it is shipped with close to 4,000 references to web resources. BioMOBY proposes to solve the issue by means of a central registry, a BioMOBY Central. This registry can be searched using BioMOBY's service ontology. Unfortunately, it relies for its maintenance on the discipline of its users, who have widely different interests. The result is a quite flat ontology tree that contains obvious duplicates, rendering the tree unfit for searching. Recent initiatives to fill this gap are BioCatalogue (`www.biocatalogue.org`, [44]), that aims to grow into a kind of revived UDDI for bioinformatics, and Magallanes [45], that is also able to construct workflows. SADI [46], to be discussed below, proposes a solution that makes use of Semantic Web technology.

(h) Quality of service of a web service

A web service may be dead, under construction, moved to another URL and/or port, or overloaded. A web service is maintained by another party that may have other priorities and may be unable to guarantee the bandwidth necessary to satisfy hundreds of bioinformaticians worldwide who want to access the same web service. The most radical solution is to mirror all important web services in-house. For OligoRAP, this is what we did. Still, mirroring is not always good practice and it runs counter to the spirit of the Web. Mirroring is costly in terms of both money and manpower. Also, it is impossible to cover special-interest services this way. A workflow can deal with slow web-services by incorporating an iterative structure that polls the external resource to find out whether results are already available. If they are not, a wait is introduced and the polling task is iteratively called again. e-BioFlow offers predefined task templates for polling.

A related problem is that of remotely assessing what went well and what went wrong in the web service itself. Most providers do not offer extensive log files.

In the absence of log files, the best one can do is to capture as much metadata as one can. This is handled by the provenance system, see below.

We have built a Taverna workflow that each month of the first half of 2009 inspected all Taverna workflows deposited at myExperiment to estimate the half-life of bioinformatics web services. Our results show that of the approximately 200 workflows that call web services, over 10% was marred by dead services. Worse, the number of broken workflows rose from 25 to 38 in just six months, an increase of over 50%.

From the perspective of the workflow designer, late binding results in workflows that are better sharable because tasks do not define resources explicitly but only roles. A role stipulates what a resource should be able to do in order to meet the requirements of the task. The resource view of the design interface of e-BioFlow supports late binding by default. Binding roles to resources is done at enactment time.

(i) Interoperability

Interoperability of web services is an important aspect in a workflow approach, but data format incompatibility is a major obstacle. In the Taverna experiment aimed at classification of tasks in myExperiment workflows we mentioned earlier [40], we tried to estimate which proportion of all tasks is devoted to data transformation. Our results indicate 30% as a lower bound, suggesting that data incompatibility still is a huge problem. Using only BioMoby services eliminates the data incompatibility problems, as we found for the OligoRAP case. The data format ontology of BioMoby however suffers from the same drawback as BioMoby's service ontology. It is evident that when users are free to define and use their own data formats, they will.

The problem is strictly speaking not technical but social. Purely technical means thus cannot solve it, but they may help. A number of projects have attempted to combat interoperability problems, among them BioMOBY, TAMBIS [47] and caBIO [48]. A newcomer is SADI (`sadiframework.org`, [46]), a framework based on the Semantic Web standards RDF and OWL to enable use of OWL reasoners and SPARQL. SADI assumes that web services are stateless and atomic, and that they consume and produce data in RDF format. SADI proposes to focus on data properties rather than data types. It is no longer required that the community agrees on a unifying ontology of data types; instead, it is sufficient that the community agrees on the meanings of relationships which can then be used by OWL reasoners to find compatible services. The initiative is very young with just a handful of services available. Time will tell whether SADI will be widely adopted and whether it facilitates the handling of interoperability problems.

(j) Quality of content

The quality of the content a web service offers is important because the results of a workflow can never be better than those of the worst web service. Although

it is evident that the databases used by bioinformaticians are not perfect, one by and large has to guess the frequency and distribution of errors. Recently, more systematic efforts have been undertaken to assess the quality of content of databases and to improve where needed and possible [49–51].

8 Discussion

In the course of designing and running bioinformatics workflows, we found that existing WfMSs suitable for bioinformatics fall short of our expectations. We eventually decided to build our own WfMS, e-BioFlow. We decided to rely exclusively on open-source software, and to make e-BioFlow an open-source project as well (`ewi.utwente.nl/~biorange/ebioflow`). To profit from experience with business administration WfMSs, we did not build e-BioFlow from scratch but instead relied on the well-proven WfMS YAWL (`www.yawlfoundation.org`). The core of YAWL is largely unchanged, except that in e-BioFlow data are passed by reference while YAWL copies data between tasks. We implemented a user interface with the help of the JGraph package (`www.jgraph.com`). Finally, the provenance database is implemented in PostgreSQL (`www.postgresql.org`). Java code ties everything together. The result was satisfying. The problems we identified in particular for *Design* and *Provenance* have been solved. e-BioFlow's performance is adequate for most bioinformatics jobs. The OligoRAP use case for an oligo library for the mouse comprising 21,766 unique oligo's [13, ch. 4], for example, has been run in e-BioFlow on a commodity PC in just three hours where the original Perl version needs twice that time. Runs sometimes ran against unavailable web services. The cache function of the database saved a lot of time in those cases. When the workflow had been run to end, the provenance archive could be browsed with relative ease.

To summarise, e-BioFlow has the following features:

- The design interface allows two modes of design: an entire workflow, or ad-hoc design. When designing an entire workflow, information is divided over three related panes to prevent information overload.
- Ad-hoc design allows a bioinformatician to play with tasks and (partial) workflows. At all times, the bioinformatician can inspect all data produced so far. This exploratory way of working is close to the way a research bioinformatician normally works.
- Late binding: resources (web services, scripts, user interaction) are selected only at runtime. At design time, the user specifies a role. There is a database that relates roles to resources. e-BioFlow supports scripting, offering syntax highlighting for Perl, R, JavaBeans, and Sleep. It supports BioMoby services and any service that can be specified in a WSDL file.
- e-BioFlow can import workflow specifications in two languages, YAWL [10] and Scufl, the language of Taverna [7].It can export control-flow related information in the Open XML Process Definition Language, XPDL (see [52] and `www.wfmc.org/xpdl.html`).

– Data are stored in a database as soon as they are produced. In the workflow, data are passed by reference. The database can serve as cache. It also contains the complete provenance once the workflow has run to end.
– There is a provenance browser based on the de facto standard OPM that enables a user to browse a provenance archive of hundreds of thousands of nodes. In the OligoRAP case, the provenance archive contains over 3 GB of data, 2,900 composite tasks and 200,000 elements in the OPM graph. Based on our experiences, even larger archives pose no problems.

e-BioFlow was solely developed to provide a proof-of-principle that a WfMS for a commodity PC can fulfil the requirements formulated in this article. As such, we hope that these notions will find their way in the already established bioinformatics WfMSs, forming the next step in the evolution of WfMSs.

Acknowledgement

This work was part of the BioRange programme of the Netherlands Bioinformatics Centre (NBIC), which is supported by a BSIK grant through the Netherlands Genomics Initiative (NGI).

References

1. Stevens, R., Goble, C., Baker, P., Brass, A.: A classification of tasks in bioinformatics. Bioinformatics **17** (2001) 180–188
2. Deelman, E., Gannon, D., Shields, M., Taylor, I.: Workflows and e-Science: An overview of workflow system features and capabilities. Future Generation Computer Systems **25** (2009) 528–540
3. Neerincx, P., Leunisse, J.: Evolution of web services in bioinformatics. Briefings in Bioinformatics **6** (2005) 178–188
4. Romano, P.: Automation of in-silico data analysis processes through workflow management systems. Briefings in Bioinformatics **9** (2007) 57–68
5. Ludäscher, B., Altintas, I., Berkley, C., Higgins, D., Jaeger, E., Jones, M., Lee, E., Tao, J., Zhao, Y.: Scientific workflow management and the Kepler system. Concurrency and Computation: Practice and Experience **18** (2006) 1039–1065
6. Taylor, I., Shields, M., Wang, I., Rana, O.: Triana applications within grid computing and peer to peer environments. Journal of Grid Computing **1** (2003) 199–217
7. Oinn, T., Addis, M., Ferris, J., Marvin, D., Senger, M., Greenwood, M., Carver, T., Glover, K., Pocock, M., Wipat, A., Li, P.: Taverna: a tool for the composition and enactment of bioinformatics workflows. Bioinformatics **20** (2004) 3045–3054
8. Wassink, I., Rauwerda, H., Neerickx, P., van der Vet, P., Breit, T., Leunissen, J., Nijholt, A.: Using R in Taverna: RShell v1.2. BMC Research Notes **2** (2009) nr. 138
9. Wassink, I.: Work flows in life science. PhD thesis, University of Twente, Enschede, the Netherlands (2010)
10. van der Aalst, W.M.P., ter Hofstede, A.: YAWL: Yet another workflow language. Information systems **30** (2005) 245–275
11. BioMoby Consortium: Interoperability with Moby 1.0 – it's better than sharing your toothbrush! Briefings in Bioinformatics **9** (2008) 220–231

12. Neerincx, P., Rauwerda, H., Nie, H., Groenen, M., Breit, T., Leunissen, J.: OligoRAP - an oligo re-annotation pipeline to improve annotation and estimate target specificity. BMC Proceedings **3** (2009) S4
13. Neerincx, P.: Web services for transcriptomics. PhD thesis, Wageningen University and Research, Wageningen, the Netherlands (2009)
14. Kent, W.: BLAT – the BLAST-like alignment tool. Genome Research **12** (2002) 656–664
15. Altschul, S., Gish, W., Miller, W., Myers, E., Lipman, D.: Basic local alignment search tool. Journal of Molecular Biology **215** (1990) 403–410
16. Flicek, P., Aken, B., Ballester, B., Beal, K., Bragin, E., Brent, S., Chen, Y., Clapham, P., Coates, G., Fairley, S., Fitzgerald, S., Fernandez-Banet, J., Gordon, L., Graf, S., Haider, S., Hammond, M., Howe, K., Jenkinson, A., Johnson, N., Kahari, A., Keefe, D., Keenan, S., Kinsella, R., Kokocinski, F., Koscielny, G., Kulesha, E., Lawson, D., Longden, I., Massingham, T., McLaren, W., Megy, K., Overduin, B., Pritchard, B., Rios, D., Ruffier, M., Schuster, M., Slater, G., Smedley, D., Spudich, G., Tang, Y., Trevanion, S., Vilella, A., Vogel, J., White, S., Wilder, S., Zadissa, A., Birney, E., Cunningham, F., Dunham, I., Durbin, R., Fernandez-Suarez, X., Herrero, J., Hubbard, T.P., Parker, A., Proctor, G., Smith, J., Searle, S.J.: Ensembl's 10th year. Nucleic Acids Research **38**(Database issue) (2010) D557–D562
17. van der Aalst, W., ter Hofstede, A., Kiepuszewski, B., Barros, A.: Workflow patterns. Distributed and Parallel Databases **14** (2003) 5–51
18. De Roure, D., Goble, C., Stevens, R.: The design and realisation of the virtual research environment for social sharing of workflows. Future Generation Computer Systems **25** (2009) 561–567
19. van der Aalst, W., van Hee, K.: Workflow management: models, methods, and systems. MIT Press, Cambridge MA (2002)
20. van der Aalst, W.: The application of Petri Nets to workflow management. Journal of Circuits, Systems and Computers **8** (1998) 21–66
21. Rygg, A., Mann, S., Roe, P., On, W.: Bio-workflows with BizTalk: using a commercial workflow engine for eScience. In Stockinger, H., Buyya, R., Perrott, R., eds.: e-Science'05. Proceedings of the First International Conference on e-Science and Grid Computing, Los Alamitos, IEEE Computer Press (2005) 116–123
22. King, R., Rowland, J., Oliver, S., Young, M., Aubrey, W., Byrne, E., Liakata, M., Markham, M., Pir, P., Soldatova, L., Sparkes, A., Whelan, K., Clare, A.: The automation of science. Science **324** (2009) 85–89
23. McPhillips, T., Bowers, S., Zinn, D., Ludäscher, B.: Scientific workflow design for mere mortals. Future Generation Computer Systems **25** (2009) 541–551
24. Shields, M.: Control- versus data-driven workflows. In Taylor, I., Deelman, E., Gannon, D., Shields, M., eds.: Workflows for e-science, Berlin, Springer (2007) 167–173
25. Wassink, I., Rauwerda, H., van der Vet, P., Breit, T., Nijholt, A.: e-BioFlow: different perspectives on scientific workflows. In Elloumi, M., Küng, J., Linial, M., Murphy, R., Schneider, K., Toma, C., eds.: Bioinformatics Research and Development, BIRD 2008, Vienna, Austria, Berlin, Springer
26. Lee, E., Parks, T.: Dataflow process networks. Proceedings of the IEEE **83** (1995) 773–801
27. Santos, I., Göbel, M., Raposo, A., Gattass, M.: A multimedia workflow-based collaborative engineering environment for oil & gas industry. In: Proceedings VRCAI 2004 - ACM SIGGRAPH International Conference on Virtual Reality Continuum and its Applications in Industry, Singapore, ACM (2004) 112–119

28. Puerta-Melguizo, M., Chisalita, C., van der Veer, G.: Assessing users' mental models in designing complex systems. In Borne, P., ed.: Proceedings of the IEEE International Conference on Systems, Man and Cybernetics 2002, IEEE Computer Press (2002) 420–425
29. Wassink, I., van der Vet, P., van der Veer, G., Roos, M., van Dijk, E.: New interactions with workflow systems. In Norros, L., Koskinen, H., Salo, L., Savioja, P., eds.: ECCE 2009 - European Conference on Cognitive Ergonomics: Designing beyond the Product-Understanding Activity and User Experience in Ubiquitous Environments, Helsinki, Finland, Helsinki, Finland, VTT (2009) 349–352
30. Greenwood, M., Goble, C., Stevens, R., Zhao, J., Addis, M., Marvin, D., Moreau, L., Oinn, T.: Provenance of e-science experiments - experience from bioinformatics. In Cox, S., ed.: Proceedings of UK e-Science All Hands Meeting 2003, Swindon, United Kingdom, EPSRC (2003)
31. Stevens, R., McEntire, R., Goble, C., Greenwood, M., Zhao, J., Wipat, A., Li, P.: myGrid and the drug discovery process. Drug Discovery Today: Biosilico **2** (2004) 140–148
32. Groth, P., Miles, S., Moreau, L.: A model of process documentation to determine provenance in mash-ups. ACM Transactions on Internet Technology **9** (2009) 1–31
33. Miles, S., Wong, S., Fang, W., Groth, P., Zauner, K.P., Moreau, L.: Provenance-based validation of e-science experiments. Web Semantics **5** (2007) 28–38
34. Bose, R., Frew, J.: Lineage retrieval for scientific data processing: a survey. ACM Computing Surveys **37** (2005) 1–28
35. Cohen-Boulakia, S., Davidson, S., Froideveaux, C.: A user-centric framework for accessing biological resources and tools. In Ludäscher, B., Raschid, L., eds.: Data Integration in the Life Sciences. Second International Workshop, DILS 2005, Berlin, Springer (2005) 3–18
36. Altintas, I., Barney, O., Jaeger-Frank, E.: Provenance collection support in the Kepler scientific workflow system. In Moreau, L., Foster, I., eds.: Provenance and Annotation of Data, Berlin, Springer (2006) 118–132
37. Moreau, L., Freire, J., Futrelle, J., McGrath, R., Myers, J., Paulson, P.: The Open Provenance Model: an overview. In Freire, J., Koop, D., Moreau, L., eds.: Provenance and Annotation of Data and Processes (IPAW 2008), Berlin, Springer (2008) 323–326
38. Kwasnikowska, N., Bussche, J.: Mapping the NRC dataflow model to the Open Provenance Model. In Freire, J., Koop, D., Moreau, L., eds.: Second International Provenance and Annotation Workshop (IPAW2008), Berlin, Springer (2008) 3–16
39. Biton, O., Cohen-Boulakia, S., Davidson, S., Hara, C.S.: Querying and managing provenance through user views in scientific workflows. In Alonso, G., Blakeley, J., Chen, A., eds.: IEEE 24th International Conference on Data Engineering, IEEE Computer Press (2008) 1072–1081
40. Wassink, I., van der Vet, P., Wolstencroft, K., Neerincx, P., Roos, M., Rauwerda, H., Breit, T.: Analysing scientific workflows: why workflows not only connect web services. In Zhang, L.J., ed.: IEEE Congress on Services 2009, IEEE Computer Press (2009) 314–321
41. Fielding, R., Taylor, R.: Principled design of the modern Web architecture. In Ghezzi, C., Jazayeri, M., Wolf, A., eds.: Proceedings of the 22nd International Conference on Software Engineering (ICSE'00), New York, ACM Press (2000) 407–416
42. Fielding, R.: Architectural styles and the design of network-based software architectures. PhD thesis, Information and Computer Science, University of California

43. van der Aalst, W., Aldred, L., Dumas, M., ter Hofstede, A.: Design and implementation of the YAWL system. In Person, A., Stirna, J., eds.: Proceedings of the 16th International Conference on Advanced Information Systems Engineering (CAiSE 2004), Berlin, Springer (2004) 142–159
44. Goble, C., Stevens, R., Hull, D., Wolstencroft, K., Lopez, R.: Data curation + process curation = data integration + science. Briefings in Bioinformatics **9** (2008) 506–517
45. Rios, J., Karlsson, J., Trelles, O.: Magallanes: a web services discovery and automatic workflow composition tool. BMC Bioinformatics **10** (2009) nr. 334
46. Wilkinson, M., Vandervalk, B., McCarthy, L.: SADI Semantic Web Services - 'cause you can't always GET what you want! In Ngoh, L., Teo, H., eds.: IEEE Asia-Pacific Services Computing Conference 2009 (APSCC 2009), IEEE (2009) 13 –18
47. Stevens, R., Baker, P., Bechhofer, S., Ng, G., Jacoby, A., Paton, N., Goble, C., Brass, A.: TAMBIS: transparent access to multiple bioinformatics information sources. Bioinformatics **16**(2) (2000) 184–186
48. Covitz, P., Hartel, F., Schaefer, C., De Coronado, S., Fragoso, G., Sahni, H., Gustafson, S., Buetow, K.: caCORE: A common infrastructure for cancer informatics. Bioinformatics **19**(18) (2003) 2404–2412
49. Joosten, R., Vriend, G.: PDB improvement starts with data deposition. Science **317** (2007) 195–196
50. Ott, M., Vriend, G.: Correcting ligands, metabolites, and pathways. BMC Bioinformatics **7** (2006) nr. 517
51. Spronk, C., Nabuurs, S., Krieger, E., Vriend, G., Vuister, G.: Validation of protein structures derived by NMR spectroscopy. Progress in Nuclear Magnetic Resonance Spectroscopy **45** (2004) 315–337
52. van der Aalst, W.: Business process management demystified: a tutorial on models, systems and standards for workflow management. In Desel, J., Reisig, W., Rozenberg, G., eds.: Lectures on concurrency and Petri Nets, Berlin, Springer (2004) 21–85