An organizational environment for in silico experiments in molecular biology

Yuan Lin[1], Marie-Angélique Laporte[1,3], Lucile Soler[4], Isabelle Mougenot[1,2], and Thérèse Libourel[1,2]

[1] LIRMM, UMR5506 CNRS-UM2, 161, rue Ada, 34095 Montpellier, Cedex 5, France
firstname.lastname@lirmm.fr
[2] UMR ESPACE DEV IRD-UM2, 500 rue J.F. Breton, 34093 Montpellier, Cedex 5, France
firstname.lastname@univ-montp2.fr
[3] Centre d'Ecologie Fonctionnelle et Evolutive, UMR5175 CNRS, 1919, route de Mende, 34293 Montpellier, Cedex 5, France
firstname.lastname@cefe.cnrs.fr
[4] CIRAD-PERSYST, Campus International de Baillarguet, 34398 Montpellier cedex 5, France
firstname.lastname@cirad.fr

Abstract. Molecular biologists, just like geneticists, make use of various experimental mechanisms and devices to conduct research and to validate or invalidate their theories or initial hypotheses. Mechanisms powered by information technology, called in silico, put data and analysis tools at the centre of the experiments, and are thus different from in vivo, ex vivo and in vitro mechanisms.
Multiple resources (data sources as well as analysis tools) are widely available and, very often, allow various modes of operation, requiring certain expertise for their optimal use. This is especially true when drawing up complex analysis scenarios based on the sequential use of appropriate processing tools. To facilitate the construction of these experimentation mechanisms, we propose a scientific workflow infrastructure which uses an organizational environment to allow abstract planning of the experimentation, followed by its concretization. The concretization phase includes a verification of the conformity of the planned process chains composition to avoid any error during execution.

Keywords: Scientific workflow, analysis pipeline, specification language, validation aspects of service composition.

1 Introduction

Life sciences often rely on the chaining of data and application resources to express the experimentation process. Valuable resources for biology, while available in ever-increasing quantities, remain, for the most part, cost-expensive and time-consuming to acquire and thus their reuse becomes almost a necessity.

To design these complex experiments, scientists often need to locate suitable resources and then to organize or reorganize them. In addition, each experiment deserves to be saved so that it can be re-executed several times, either in various different configurations or with diverse test data. In such a context, the use of a scientific workflow proves to be an invaluable help. Several dedicated software applications for this purpose now exist, most notably in the financial sector, and research in the field is relatively advanced. A first study [7] presented our approach based on the concept of the scientific workflow environment. Its objective is to help the user to:

- design experimentation process chains (in as abstract a manner as possible),
- better organize resources (data and processes) which will be elements in the concretization of these process chains,
- capitalize on the existing by constructing new processes from previously devised experimentation plans.

This article develops our research advances in terms of resource organization and semi-automatic verification of validity of workflows designed within a prototype.

This article is structured as follows: section 2 presents a brief state of the art, section 3 proposes an architecture for implementing a scientific workflow and section 4 provides a glimpse of the organization brought about. Section 5 covers the proposed verification of conformity, section 6 illustrates with an example the validation of conformity of a concrete process chain, and section 7 presents perspectives in progress.

2 State of the art

A study was conducted based on characteristics we deemed relevant [8]:

- The existence of a meta level for describing and creating process chains. In fact, the generic aspect conferred by meta-modelling appears to be fundamental for all of us.
- Taking the experimental aspect into account. The unique characteristics of scientific data and processes should show through at the formalism level.

We present here only two representative projects, Kepler[1] and Taverna[6], which gain a certain amount of popularity among workflow scientists.

2.1 KEPLER

KEPLER [5] is a complete scientific workflow environment based on the Ptolemy II platform of the University of Berkeley. As far as process chains are concerned,

[5] http://kepler-project.org/

KEPLER adopts a human organization metaphor. It is Actor-Based and considers all components of a process chain as actors. Actors (services) are accessed via a structure corresponding to the business ontology of the concerned domain.

The workflow is represented using a graphical language in the form of a graph linking *ports* (input/output parameters) of *actors* via *channels*. One or more actors in charge, *Directors*, plan tasks for other actors of the organization; they do so based on the available ontology. The execution plan of a process chain (or a portion of a process chain) is therefore created by a *Director* of the system. Any necessary adaptations are achieved by intermediary *sender* and *receiver* programs, which ensure the compatibility of data transferred over a channel. The process chain is saved in the form of MoML (Modelling Markup Language) files. (MoML is an XML-based language.) At the environment-interface level, a specific zoom feature is associated with the concept of an *opaque actor* (cf. figure 1). An *opaque actor* appearing in a process chain can be opened, thus revealing its constituent details.

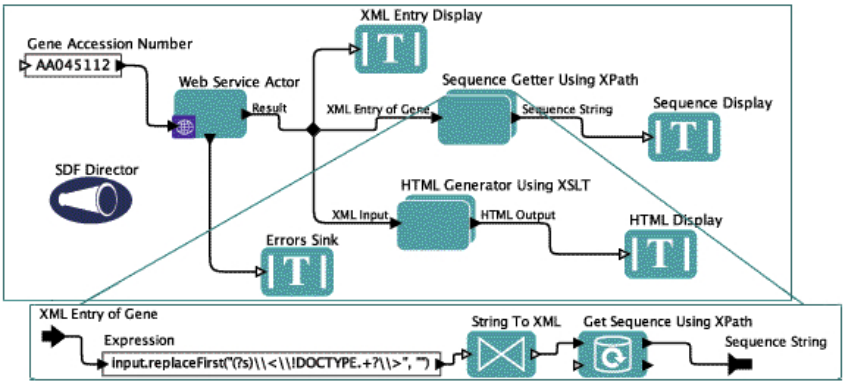

Fig. 1. Overview of a process chain in the KEPLER environment

2.2 Taverna

Taverna is a workflow project created by the ^{my}Grid team in England and used mainly in the life sciences. A workflow in Taverna is considered as a process graph in which processes are connected by data links or control links. Processes used are essentially web services (which can be supplemented by local libraries, manuscript scripts, etc.). During process composition, the user manually couples input/output parameters of web services or invokes *shim services*, specific adaptors existing from couplings constructed and tested for experiments. In addition, the process chain is saved in the form of a SCUFL (Simple Conceptual Unified Flow Language) file. (SCUFL is an XML-based language.)

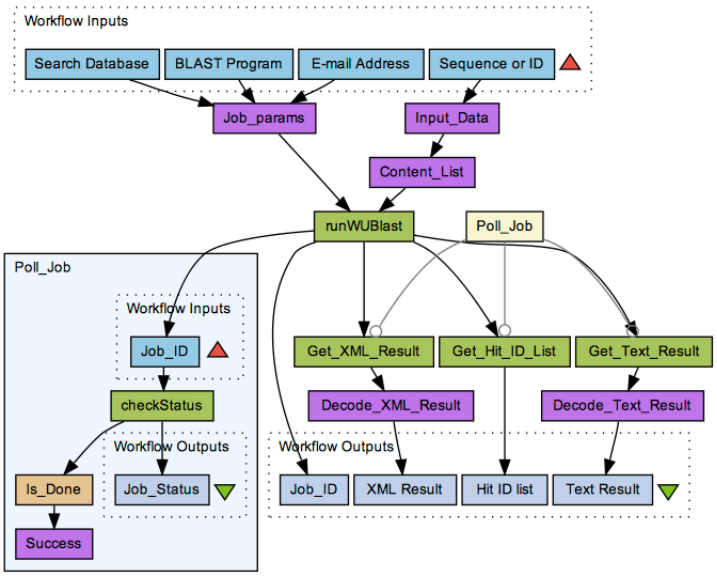

Fig. 2. A concrete workflow in Taverna (taken from the *myExperiment* Taverna sharing site)

2.3 Other related works of interest

The Taverna and Kepler projects both provide generic models for instantiation and composition of services. Additionally, some other approaches are also highly relevant to scientific workflow management:

- The project BioMoby [17], as a first attempt to assist process chaining by using scientific resources, which are described and classified in the *MOBY Central*.
- PISE ans its revised system Mobyle [18] that provides a web environment (a Web Portal) to define and execute bioinformatics analyses. Registered analysis programs are pre-classified in a hierarchy, as well as some frequently-used workflows. Experts can easily find them by using the search function panel that is integrated in the web site.
- The project ProtocolDB [19] proposed to model scientific workflows at two different layers (design protocol/ implementation protocol). An implementation protocol for a given design protocol is realized by mapping design tasks to different implementation tasks (scientific resources like database queries/ tools), and by connecting them together.
- In [20–22], scientific workflow modeling is supported by resource discovery approaches.

In this manuscript we focus mainly on scientific workflows and the way they are modeled and implemented. Our proposal introduces an additional level of

abstraction, whose purpose is to describe the business domain prior to creating the process chains. This additional modelling level is predicted to facilitate the construction of process chains by allowing biologists to use their expertise of their domain, but without requiring them to have expert and often precise knowledge of the underlying resources and their locations. It also plays the role of a prescription model, to which instantiation and service composition models have to conform.

3 Workflow architecture

Our efforts have been guided by the business point of view, that of the experimenters. Designing an experimental protocol corresponds to general model with three stages: 1) *Definition*: abstract definition of a process chain corresponding to an experimentation sequence (planning the experiments), 2) *Instantiation*: a more specific definition after identifying the various elements of the chain (data/processes), 3) *Execution*: customized execution (according to strategies corresponding to the requirements).

Based on this experimental life cycle, and inspired by the architectural styles proposed by OMG [11], we propose the following 3-level architectural vision (cf. figure 3):

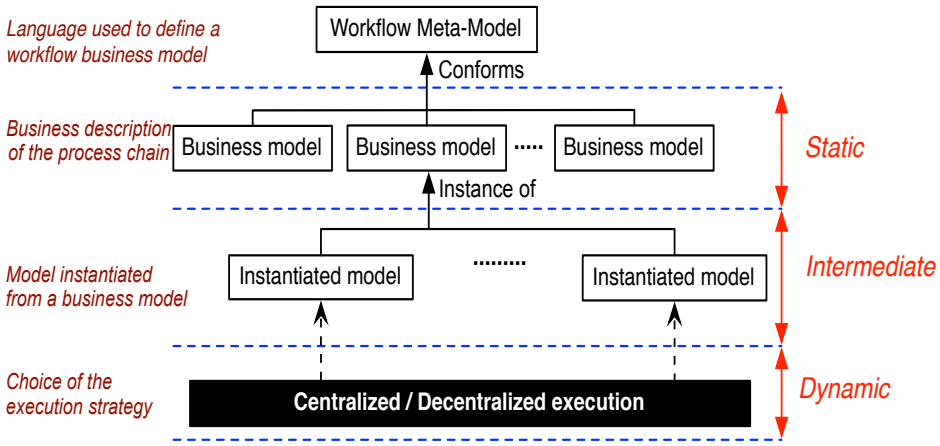

Fig. 3. 3-level architecture of a workflow component

The *static* level concerns the design phase. It is a matter of constructing (abstract) business-process models using a simple language. The *intermediate* level represents an instantiation and pre-verification phase. Using the business

process model, the user constructs the real process chain by selecting and locating the processes and data most appropriate to the planned experimentation. The pre-verification is semi-automatized (cf. section 4). The *dynamic* level concerns the actual execution phase. It takes place based on the various strategies defined by both the user and the operational configurations.

The *static* level has been studied in some detail in our [7, 8]. We have analyzed various language standards such as UML (activity diagram) [9] and SPEM [10], as also various existing projects such as BioSide [5], Meta-model WDO-It! [12] and CIMFlow [4]. Following this study, we proposed a simple but complete language. It is based on a language defined by a meta-model whose abstract elements, *tasks* or *processes*, are connected by unidirectional links and by the intermediary of *ports*. To facilitate the manipulation of abstract process chains, a corresponding graphical language was created within a prototype (cf. the top part of the figure 4). By using this workflow definition language, a simple example is modelled and shown in the lower part of the figure 4 [6].

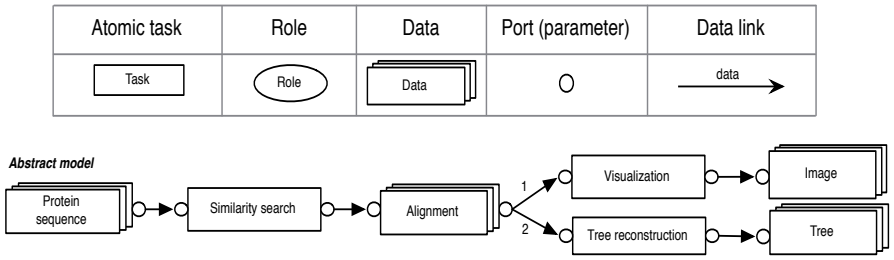

Fig. 4. Some essential elements of our graphical language and a simple example

We currently focus on the *intermediate* level, which consists of two essential stages:

- instantiation of the abstract model with existing resources (data/processes);
- validation of the concrete model instantiated from the organizational environment.

4 Organizational environment

To carry out the experimental protocols, the abstract model *instantiation* stage consists of finding and reusing existing resources. To facilitate this search, we base ourselves on the concept of *organizational environment*. This environment relies on the description of resources (data and processes) in the form of metadata

[6] This example is also used in the later sections, we will explain it in detail during the following sections.

(expressed in XML schema format). The resource descriptions are hierarchized in resource categories and in concrete resources. As shown in figure 5, it consists of:

- an organization relating to processes. It manages the hierarchy of descriptions of process categories and of concrete processes. The concept of *Converter* corresponds to the concept of a specific process responsible for adapting data between different formats of the same data category.
- an organization relating to data. It manages a hierarchy of descriptions of data categories, of concrete data and of the various associated data formats[7].

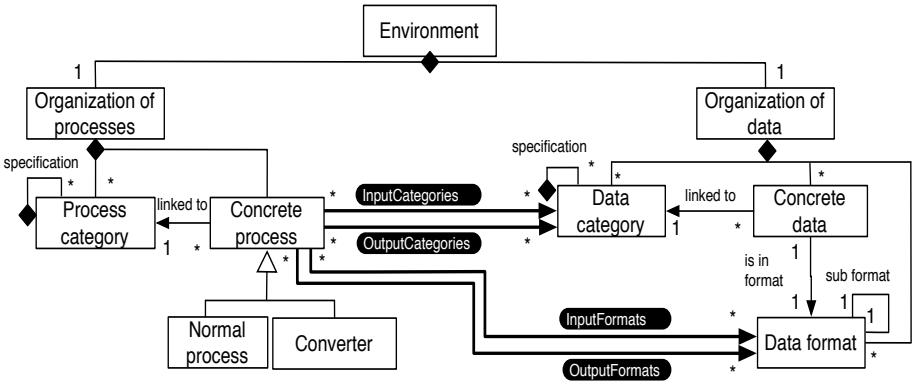

Fig. 5. Organizational environment

To illustrate this concept of the environment, we take an example from the world of molecular biology (cf. figure 6). The upper part of each hierarchy (processes and data) represent a set of categories (shown as ovals) sorted according to the generalization/specialization relationship. The descriptions of concrete resources (data or processes) are then associated to their category.

The description of a concrete data describes its format, whereas that of a concrete process corresponds to its *signature*, which we formalize thus:

Definition 1. *Formalized signature of a concrete process*

Name (Input parameter list) : (Output parameter list), where each parameter is described by the doublet (Data category : data format).

A set of data formats (Fasta, xml, MultiFasta, Clustal, Newick, Jpeg) is also presented. Figure 6 is therefore complemented by the description of signatures of some example concrete processes:

[7] Remark: It should be noted that several data categories can share the same format.

Blastp(ProteinSeq:Fasta) : (SeqPairs:xml)
ClustalW(ProteinDataBank:MultiFasta) : (MultipleAlignment:Clustal)
InteractiveSelection(SeqPairs:xml) : (ProteinDataBank:MultiFasta)
Logo(MultipleAlignment:Clustal) : (Image:jpeg)
PhyML(MultipleAlignment:Clustal) : (PhylogeneticTree:Newick)

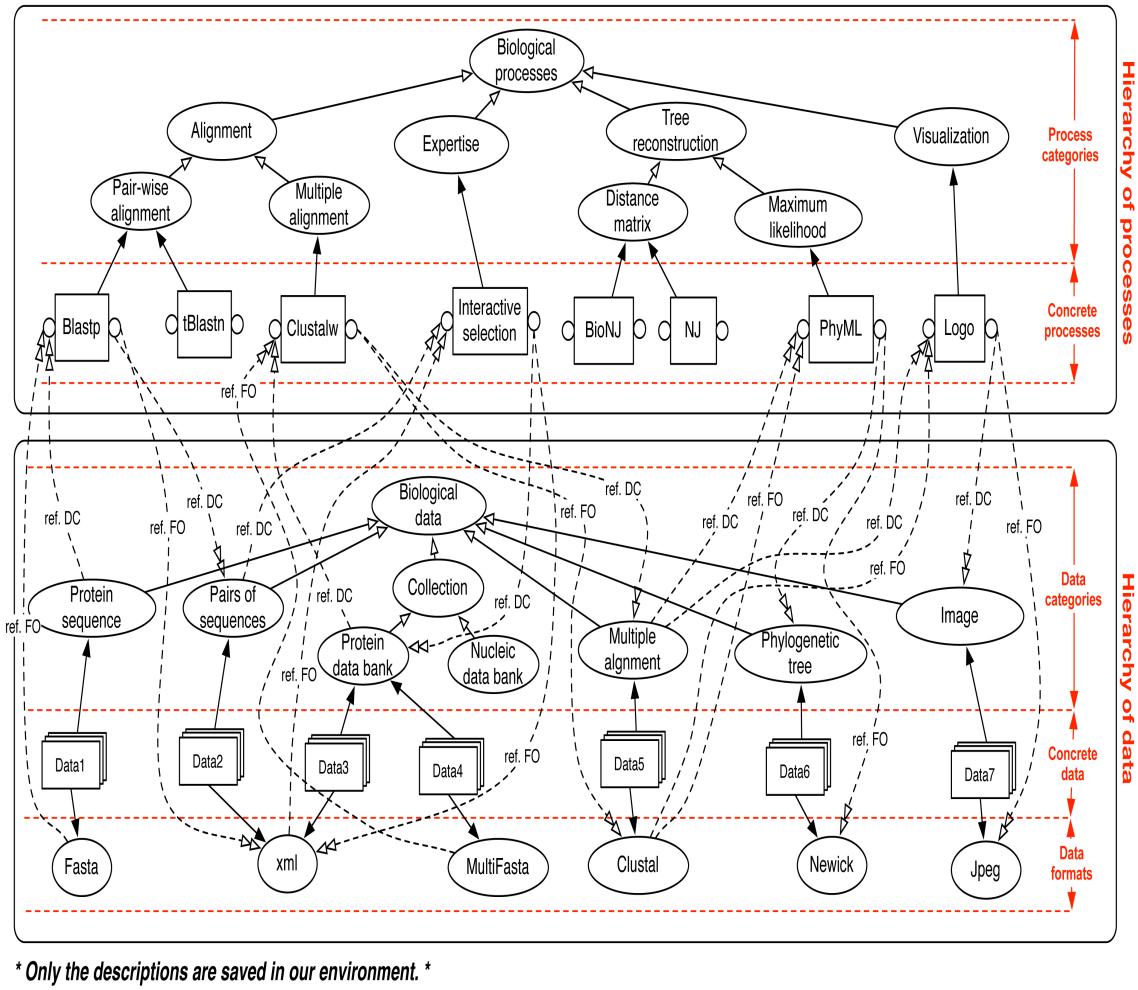

Fig. 6. Illustration of an organizational environment in a biological context

5 Conformities

5.1 The problem

As already mentioned, the second important stage of the *intermediate* level consists of validating the concrete model instantiated from the abstract model.

Let us take an example described by using the workflow language, corresponding to an abstract process chain model that a biologist designs with the intention of characterizing a protein sequence which interests him in the context of his putative functional domains.

At the concrete level, the idea is to begin by using the *Blast* similarity-search tool to compare the protein sequence under consideration with a data bank of protein sequences and to thus identify segments with high similarity shared both by the protein sequence under consideration and by various sequences in the sequence data bank. These similar segments indicate the possible presence of functional domains. The biologist then continues his study by reusing the results output from the *Blast* tool [2], either to construct a phylogenetic tree and retrace the evolutionary history of the sequence via the *PhyML* tool [3] or to display the preserved positions common to all the similar segments via the *Logo* tool [13]. This simplified example of a process chain in molecular biology allows us to highlight the difficulties encountered by the biologist in using the results output by one tool as input to another tool. The difficulties relate, at the same time, to the nature of the data (here characterized as data category), to the format of this data, and, finally, to the biologists expertise. In the example, we make willing use of the discrepancy which arises between the Blast tool, which outputs a collection of simple alignments, and the PhyML and Logo tools, which require multiple alignments to run. In fact, Blast leads to multiple discrepancies two-by-two, involving the sequence under consideration and one of the sequences from the sequence data bank which is similar to it; whereas PhyML and Logo use the shared similarity by a set of sequences which includes the sequence under consideration. This example highlights what we will subsequently term *semantic incompatibility*.

In its upper part, the figure 7 shows the abstract process chain and in the lower the concrete chain obtained after locating data descriptions *S1* and adapted processes *Blastp* and *PhyML*. The problem which we designate as one of *validation of the instantiated (concrete) model* consists of verifying the *compatibility* of each *composition*. A *composition* corresponds to the link between an output parameter p1 of a process T and an input parameter p2 of the process following T; we denote it *(p1 → p2)*.

5.2 Identifying situations of compatibility

Verification is undertaken by analyzing the signatures of linked processes. To do so, we have to take two important aspects into account:

– The *syntactic* aspect : relating to the data formats used by the parameters.

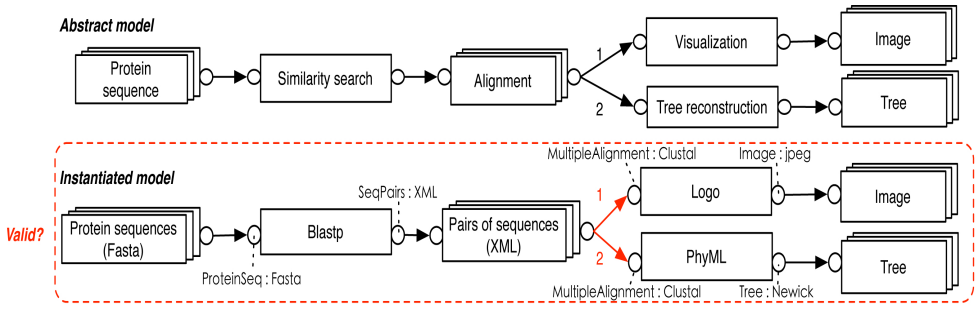

Fig. 7. Problem at hand

– The *semantic* aspect : relating to the processs functionality. It not only depends on the processs name but also on the signification of the input/output parameters.

For two processes *T1(dc1:fo1) : (dc2:fo2, dc3:fo3)* and *T2(dc4:fo4) : (dc5:fo5)*, let us suppose that there exists a composition, denoted *p1→p2*, between the *p1 (dc3:fo3)* output parameter of process *T1* and the *p2 (dc4:fo4)* input parameter of process *T2*.

Syntactic and semantic compatibilities are defined as follows:

Definition 2. *Syntactic compatibility*

$p1 \rightarrow p2$ is syntactically compatible if $(fo3 = fo4) \vee (fo3$ *is a sub-format of* $fo4)$, denoted $p1 \overset{Syn}{\rightarrow} p2$. Two parameters are syntactically compatible if they use the same data format or if they use an output format which is a sub-format of the input format. Else $p1 \overset{Syn}{\nrightarrow} p2$.

Definition 3. *Semantic compatibility*

$p1 \rightarrow p2$ is semantically compatible if $(dc3 = dc4) \vee (dc3$ *is a sub-category of* $dc4)$, denoted $p1 \overset{Sem}{\rightarrow} p2$. Two parameters are semantically compatible if they use the same category, or if they use an output category which is a sub-category of the input category. Else $p1 \overset{Sem}{\nrightarrow} p2$.

The verification of a compositions compatibility is thus done at two levels: syntactic and semantic. Three types of situations can arise:

– Situation 1 $(p1 \overset{Sem}{\rightarrow} p2) \wedge (p1 \overset{Syn}{\rightarrow} p2)$: *p1* and *p2* are compatible at the semantic and syntactic levels. This is the ideal situation in our context; we designate it as valid.
– Situation 2 $(p1 \overset{Sem}{\rightarrow} p2) \wedge (p1 \overset{Syn}{\nrightarrow} p2)$: *p1* and *p2* are compatible at the semantic level but not at the syntactic level. The composition is syntactically adaptable. An adaptation between the two data formats will be necessary (cf. converters).

– Situation 3 $p1 \overset{Sem}{\nrightarrow} p2$: The two parameters are not semantically compatible. In such a case, it is pointless to proceed to verify their syntactic compatibility (in fact, for us, two parameters with different significations cannot be paired). The composition is semantically adaptable.

From these definitions, we develop our proposed approach for resolving the incompatibilities.

6 Validation of the experimental chain

Of the three compatibility situations identified, the latter two require an adaptation stage before going on to the execution phase. It is a matter of finding one or more intermediate processes which can overcome the compositions incompatibility. For situations 2 and 3, two types of adaptations are proposed:

– *semantic* adaptation (for situation 3). The incompatibility of situation 3 represents the case where the two parameters of a composition use incompatible data categories. The adaptation here consists of finding a possible intermediate process chain between these two categories.
– *syntactic* adaptation (for situation 2). In situation 2, where the composition is already semantically compatible, the problem can be expressed as a divergence between the data formats used by the two connected parameters. All that is required is to find *converters* to convert one data format into the other.

These adaptations are based on the organizational environment. The search for intermediate processes can be equated to a search for itineraries between two incompatible data categories or formats. We will illustrate this using the example and the organizational environment constructed earlier (cf. figure 6).

Let us consider again the previous example. The verification conducted on the instantiation of the abstract model detects a semantic incompatibility in the composition between Blastp and Logo or between Blastp and PhyML due to difference in categories *Pairs of sequences* and *Multiple Alignment* (*Incompatibility situation 3*). The (semantic) adaptation will be applied; it consists of finding in what we call the (semantic) resource graph the path allowing the conversion of categories.

The construction of the (semantic) resource graph consists of extracting, from the organizational environment, the descriptions of processes and of data categories referenced by their parameters. Such a (semantic) resource graph generated from the environment described in the figure 6 is shown in the figure 8.

A graph traversal algorithm is used to find all the possible paths between the two concerned data categories (*Pairs of sequences* and *Multiple Alignment*). A single path is found in the graph: *Pairs of sequences* → *InteractiveSelection* → *ProteinDataBank* → *ClustalW* → *Multiple Alignment*. The two processes, *InteractiveSelection* and *ClustalW*, will therefore be added to the incompatible chain (cf. figure 9).

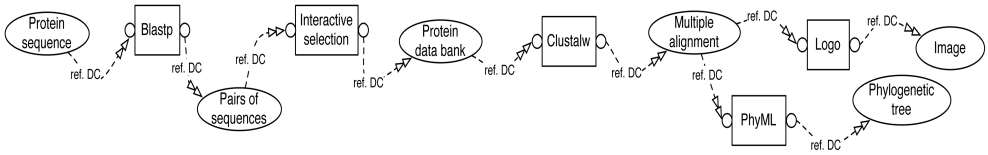

Fig. 8. (Semantic) resource graph generated from the organizational environment of the figure 6

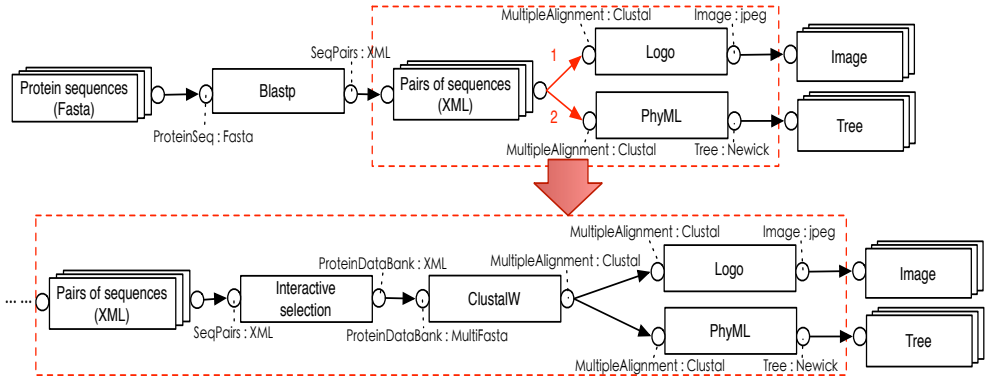

Fig. 9. Semantic adaptation

Once this adaptation is done, there still remains the existing syntactic incompatibility of the composition between the *InteractiveSelection* and *ClustalW* processes because even though *InteractiveSelection* outputs the same data category that is accepted for input by *ClustalW*, their data formats are different (*xml* and *MultiFasta*). Syntactic adaptation consists of finding specific *converters*, or compositions of *converters*, necessary for these conversions. We will not cover this stage in detail; it is simply enough to understand that converters (or their composition) can be added to obtain the required validity.

7 Conclusion and perspectives

A prototype (http://www.lirmm.fr/ lin/project/) illustrating the key aspects of our approach for designing and validating scientific process chains is currently being developed. This prototype serves as a basis for an inductive experimental approach using data of BAC and EST nucleic sequences as well as physical and genetic maps for identifying and characterizing genetic markers relating to sex of the Nile tilapia (Oreochromis niloticus). Over a longer term, we intend to integrate the current prototype into a platform with a search engine based on resource descriptions to be able to undertake the execution using real re-

sources, after requisite validation of experimentation chain. It will eventually also use open-source controlled vocabularies such as PFO (Protein Feature Ontology)[14], SO (Sequence Ontology)[15], and GO (Gene Ontology)[16] to enrich data categories by additional representations and thus extend the descriptive capacities of the organizational environment.

References

1. I. Altintas, B. Ludäscher, S. Klasky, and M. A. Vouk. S04 - *introduction to scientific workflow management and the kepler system*. In SC, page 205, 2006.
2. S. Altschul, W. Gish, W. Miller, E. Myers and D. Lipman. *Basic local alignment search tool*. In Journal of Molecular Biology, vol 215, pages 403-410, 1990.
3. S. Guindon and O. Gascuel. *A simple, fast, and accurate algorithm to estimate large phylogenies by maximum likelihood*, in Systematic Biology, vol 52, pages 696-704, 2003.
4. L. Haibin, F. Yushun, *CIMFlow: A Workflow Management System Based on Integration Platform Environment*. In Proceedings of 7th IEEE International Conference on Emerging Technologies and Factory Automation. Barcelona : ETFA, 1999: 187-193.
5. M. Hallard & al. *Bioside : faciliter l'acceès des biologistes aux ressources bioinformatiques*, JOBIM, Montreéal 2004, p 64.
6. D. Hull, K. Wolstencroft, R. Stevens, C. A. Goble, M. R. Pocock, P. Li, and T. Oinn. *Taverna: a tool for building and running workflows of services*. Nucleic Acids Research, 34(Web-Server-Issue):729732, 2006.
7. T. Libourel, Y. Lin, I. Mougenot, C. Pierkot, JC. Desconnets, *A Platform Dedicated to Share and Mutualize Environmental Applications*. Proceedings of 12th International Conference on Enterprise Information Systems, Madere, 2010.
8. Y. Lin, T. Libourel, I. Mougenot, *A Workflow Language for the Experimental Sciences*, Proceedings of 11th International Conference on Enterprise Information Systems, Milan, 2009.
9. Object Management Group (OMG), *OMG Unified Modeling LanguageTM (OMG UML), Infrastructure Version 2.3*. OMG Document Number: formal/2010-05-03.
10. Object Management Group (OMG), *SPEM - Software & Systems Process Engineering Meta-Model Specification, Version 2.0*. OMG Document Number: formal/2008-04-01.
11. Object Management Group (OMG), *Meta Object Facility (MOF) Core Specification OMG Available Specification Version 2.0*, OMG Document Number: formal/06-01-01.
12. P. Pinheiro da Silva, L. Salayandia, A.Q. Gates, *WDO-It! A Tool for Building Scientific Workflows from Ontologies* (2007). Departmental Technical Reports (CS). Paper 201.
13. T. D. Schneider and R. M. Stephens, *Sequence Logos: A New Way to Display Consensus Sequences*. In Nucleic Acids Res., vol 18, pages 6097-6100, 1990.
14. G.A. Reeves, K.Eilbeck, M.Magrane, C.O'Donovan, L.Montecchi-Palazzi, M.A. Harris, S.E. Orchard, R.C. Jimenez, A.Prlic, T. J. P. Hubbard, H.Hermjakob, J.M. Thornton. *The Protein Feature Ontology: a tool for the unification of protein feature annotations*. In Bioinformatics, vol 24, pages 2767-2772, 2008.
15. K.Eilbeck, S.E Lewis, C.J Mungall, M.Yandell, L.Stein, R.Durbin, M.Ashburner. *The Sequence Ontology: a tool for the unification of genome annotations*. In Genome Biology, vol 6, pages R44, 2005.

16. M.Ashburner, C.A. Ball, J.A. Blake, D.Botstein, H.Butler, J. Michael Cherry, A.P. Davis, K.Dolinski, S.S. Dwight, J.T. Eppig, M.A. Harris, D.P. Hill, L.Issel-Tarver, A.Kasarskis, S.Lewis, J.C. Matese, J. E. Richardson, M.Ringwald, G.M. Rubin, G.Sherlock, *Gene ontology: tool for the unification of biology. The Gene Ontology Consortium.* In Nature Genetics, vol 25, pages 25-29, 2000.
17. Michael DiBernardo, Rachel Pottinger, Mark Wilkinson: *Semi-automatic web service composition for the life sciences using the BioMoby semantic web framework.* Journal of Biomedical Informatics 41(5): 837-847 (2008).
18. Bertrand Néron, Hervé Ménager, Corinne Maufrais, Nicolas Joly, Julien Maupetit, Sébastien Letort, Sébastien Carrère, Pierre Tufféry, Catherine Letondal: *Mobyle: a new full web bioinformatics framework.* Bioinformatics 25(22): 3005-3011 (2009).
19. Michel Kinsy, Zoé Lacroix, Christophe Legendre, Piotr Wlodarczyk, Nadia Yacoubi Ayadi: *ProtocolDB: Storing Scientific Protocols with a Domain Ontology.* WISE Workshops 2007: 17-28
20. Zoé Lacroix: Resource Discovery, Second International Workshop, RED 2009, Lyon, France, August 28, 2009. Revised Papers Springer 2010.
21. Zoé Lacroix, Cartik R. Kothari, Peter Mork, Rami Rifaieh, Mark Wilkinson, Juliana Freire, Sarah Cohen Boulakia: *Biological Resource Discovery.* Encyclopedia of Database Systems 2009: 220-223.
22. Nadia Yacoubi Ayadi, Zoé Lacroix, Maria-Esther Vidal: *A Deductive Approach for Resource Interoperability and Well-Defined Workflows.* OTM Workshops 2008: 998-1009.