ProtocolDB: Storing Scientific Protocols with a Domain Ontology

Michel Kinsy, Zoé Lacroix, Christophe Legendre,
Piotr Wlodarczyk, and Nadia Yacoubi

Scientific Data Management Laboratory
Arizona State University
Tempe AZ 85287-5706, USA

Abstract. This paper addresses a systemic problem in science: although datasets collected through scientific protocols may be properly stored, the protocol itself is often only recorded on paper or stored electronically as the script developed to implement the protocol. Once the scientist who has implemented the protocol leaves the laboratory, this record may be lost. Collected datasets without a description of the process used to produce them become meaningless; furthermore, the experiment designed to produce the data is not reproducible. In this paper we present the ProtocolDB system that aims at assisting scientists in the process of (1) designing and implementing scientific protocols, (2) storing, querying, and transforming scientific protocols, and (3) reasoning about collected experimental data (data provenance).

1 Introduction

Public biological resources form a complex maze of heterogeneous data sources, interconnected by navigational capabilities and applications. Although this wide and valuable network offers scientists multiple options to execute their scientific protocols, selecting the resources suitable to obtain and exploit their data of interest is a tedious task. When designing a scientific protocol, they struggle to consolidate the best information about the scientific objects being studied and to implement it in terms of queries against biological resources. These protocols, although expressed at a conceptual level, are typically implemented using the resources the scientist is most familiar with, instead of the resources that may best meet the protocol's needs. This implementation-driven approach to express scientific protocols may significantly affect the outcome of a scientific experiment.

The biological semantic Web is diverse and offers multiple orthogonal viewpoints on scientific data. Each viewpoint is expressed by the way the data are organized (e.g., GenBank is sequence-centric when GeneCards is gene-centric), the access capabilities offered to scientists to retrieve data (e.g., to access gene descriptions in GeneCards, one can use a full-text search engine or provide a HUGO symbol), the applications, annotations, and links and indices to other relevant resources. In addition to this structural diversity, biological resources offer a rich semantic diversity characterized by data coverage (entries present

M. Weske, M.-S. Hacid, C. Godart (Eds.): WISE 2007 Workshops, LNCS 4832, pp. 17–28, 2007.

in the data source), identity, characterization and annotations (set of attributes pertaining to each entry), links and indices between entries, the domain, image, and cardinality of those links, quality, consistency, reliability, etc. All these semantic characteristics are metrics that may be used to predict the outcome of the execution of a scientific protocol on selected resources. Syntactic, semantic and cost characteristics all participate in the outcome of the execution of a scientific protocol. Indeed, the selection of a resource may dramatically affect the dataset collected at execution time [7].

To assist adequately scientists in the process of expressing scientific protocols, it is necessary to understand what scientific protocols are, how they are structured, how scientists express them, and how they are implemented for execution. While they are critical components of the scientific process that leads to discovery, scientific protocols have been poorly studied. A scientific protocol is the process that describes the experimental component of scientific reasoning. Scientific reasoning follows a hypothetico-deductive pattern, i.e., the successive expression of a causal question, a hypothesis, the predicted results, the design of an experiment, the actual results of the experiment, the comparison of the predicted results and the actual results, and the conclusion, which may or may not be supportive of the hypothesis [8]. Scientific protocols (or equivalently pipelines, workflows, or dataflows) are complex procedural processes composed of a succession of scientific tasks that express the way the experiment is conducted. Although there is no commonly used definition of what a scientific protocol really is, in January 2003 a brainstorming session devoted to scientific protocols[1] identified the following characteristic: a succession of steps (recipe) that describes a process that can be reproduced. A scientific protocol thus describes how the experiment is conducted and records all information necessary to reproduce the same experiment. In the context of digital scientific protocols each step of the protocol is a bioinformatics task [15,1] that records how biological data are produced from measurements, extracted from a data source or resulting from an application, etc.

In this paper we present the ProtocolDB system that aims at assisting scientists in the process of (1) designing and implementing scientific protocols, (2) storing, querying, and transforming scientific protocols, and (3) reasoning about collected experimental data (data provenance). A motivation example is presented in Section 2. Section 3 is devoted to conceptual protocols whereas the selection of resources and their integration are discussed in Section 4. We discuss related work in Section 5 and conclude in Section 6.

2 Motivating Example

Alternative Splicing (AS) is the splicing process of a pre-mRNA sequence transcribed from one gene that leads to different mature mRNA molecules thus to

[1] The session took place during the Dagstuhl Seminar 03051 devoted to Information and Process Integration: A Life Science Perspective. The material presented at the seminar is available at http://www.dagstuhl.de/03051/.

different functional proteins. Alternative splicing events are produced by different arrangements of the exons of a given gene. The Alternative Splicing Protocol (ASP) described as follows is currently supporting the BioInformatics Pipeline Alternative Splicing Services BIPASS [6].

> *The Alternative Splicing Protocol (ASP) takes a set of transcripts as input and returns clusters of transcripts aligned to a gene. The process of alignment consists of an alignment of each transcript sequence against each genomic sequence of a whole genome of one or more organisms. This step is executed with all known transcripts extracted from different public databases. A clustering step immediately follows the alignment step. That step allows delimiting the transcript region of a gene excluding its regulation region. A cluster normally represents or may be representative of all intermediate transcripts (from the Pre-messenger-RNA(s) to the mature messenger-RNA(s)) required to obtain one or several functional translated proteins from the same gene.*

Such a protocol description expresses the *design protocol*, i.e., its scientific aim. It specifies two scientific tasks with a conceptual description of their inputs and outputs illustrated in Figure 1.

1. Task 1 performs an *alignment* of transcripts against genomic sequences. The results (output) of this task is an alignment of the transcripts with respect to the genomic sequence.
2. Task 2 performs a *clustering* of the aligned transcripts. The result (output) of this task is a list of clusters of transcripts.

In general the description of a scientific protocol is a textual document that combines the scientific aim with the resources used to implement it. For example, ASP could be described as follows.

> *ASP performs a first alignment with BLAT, selects the 10% first ranked alignments, extracts the aligned transcripts and the aligned genomic sequences. Then it completes the extracted genomic sequences with 50,000 bases in upstream and downstream. It re-aligns the transcripts against the resulting genomic sequences with SIM4 and clusters the results.*

Such a protocol description is a poor record of the process.

- A textual document is difficult to parse to extract the exact tasks involved, their ordering, and all needed semantic and structural information exploited when retrieving, querying, and reasoning about scientific protocols.

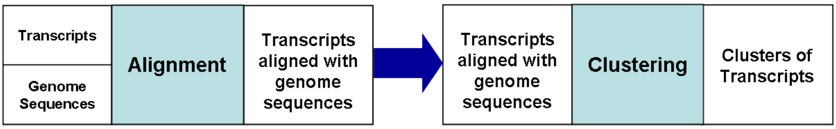

Transcripts
Genome Sequences
Alignment
Transcripts aligned with genome sequences
Transcripts aligned with genome sequences
Clustering
Clusters of Transcripts

Fig. 1. Alternative splicing design protocol

- It mixes the scientific aim and the resource selection (i.e., BLAT and SIM4). This makes it difficult to revise the protocol with new resources and compare their results. It also affects the ability to integrate data collected from different protocol implementations with similar scientific aim.
- It does not record the reasons why a simple scientific task such as a sequence alignment needs to be split into a sub-protocol involving five tasks: alignment, filter, extraction, collection, and alignment.
- It does not specify how the resources were integrated (schema mapping, variable binding).

In ProtocolDB, a scientific protocol is composed of a *design protocol* that captures its scientific aim expressed with respect to a domain ontology, and one or more *implementations* that specify the resources selected to implement each task and the dataflow expressed as ontology-driven schema mappings. The *design protocol* is mapped to *implementation protocols*, themselves mapped to experimental data collected after their execution as illustrated in Figure 2.

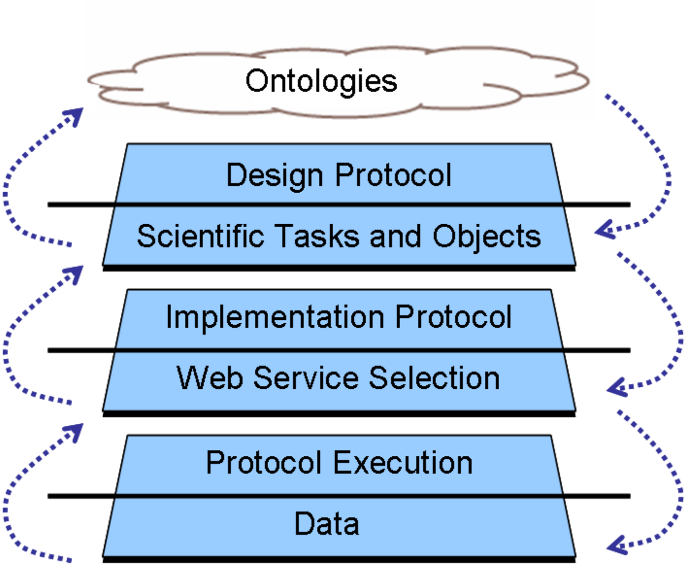


Fig. 2. Life Cycle of a scientific protocol

3 Design Protocol

A *design protocol* (or conceptual protocol) is a graph composed of connected scientific tasks whose inputs and outputs are collections of conceptual variables. Each scientific design task is a design protocol. Complex design protocols are composed of scientific design tasks connected with two binary operators $\bullet$ and $\otimes$, respectively denoting the *successor operator* and the *parallel composition* [4,3]. I (resp. O) denotes the input (resp. output) of the design task or protocol.

Definition 1. *The set of design protocols D is the closure of the set of design tasks T under two binary operators $\bullet$ and $\otimes$.*

- $T \subset D$
- $\forall P_1, P_2 \in D$ *if* $I_{P_2} = O_{P_1}$ *then* $P_1 \bullet P_2 \in D$, $I_{P_1 \bullet P_2} = I_{P_1}$ *and* $O_{P_1 \bullet P_2} = O_{P_2}$.
- $\forall P_1, P_2 \in D$ *then* $P_1 \otimes P_2 \in D$, $I_{P_1 \otimes P_2} = [I_{P_1}, I_{P_2}]$ *and* $O_{P_1 \otimes P_2} = [O_{P_1}, O_{P_2}]$.

A design protocol expresses the scientific aim of the protocol in terms of a domain ontology such as described in Figure 3. The design protocol is a conceptual protocol where each task expresses a conceptual scientific task or relationship. The dataflow captured by a design protocol is expressed in terms of collections of variables typed with respect to classes in an ontology. For example, the ASP design task *alignment* takes as input a record $[s_1, \{s_2\}]$, where s_1 and s_2 are two variables of type `Sequence`. Two design tasks (or protocols) may be composed sequentially if the input of the latter is the same as the output of the former.

3.1 Protocol Entry in ProtocolDB

The ProtocolDB entry interface allows scientists to edit and store scientific protocols. To enter a new protocol, a scientist registers in the system and records the context of the protocol (institution, author, etc.). Documents may be uploaded to the system. Once the overall scientific protocol is described, the user starts constructing the design protocol (conceptual workflow) from the interface shown in Figure 4. The initial step consists of a protocol blackbox (root) that represents the overall protocol.[2]

The user may rename the protocol (1), specify inputs (2) and outputs (3), and enter a description (4). Then the user may select one of two operators: *split sequential* or *split parallel* shown in the lower right square. By selecting the *split sequential* operator, the system splits the selected task box (root) into two successive tasks. The operator *split sequential* inserts a new task right after the selected task in the protocol branch. By default, the input (resp. output) of the new inserted task is a tuple composed of the input and output (resp. output) of the selected task. In contrast *split parallel* splits the protocol branch into two branches. One of the branches corresponds to the selected task whereas the second branch inserts a new task box with the same input and output than the selected task. The default specifications may be changed when documenting the new task box (input variables may be removed and new output may be produced).

Editing functions (currently under development) allow the user to remove tasks and upload existing protocols to instantiate a task box. Once the design protocol is entered in ProtocolDB, the user may map it to one or several implementations.

[2] Our approach follows a top-down approach splitting step-by-step the protocol into connected tasks.

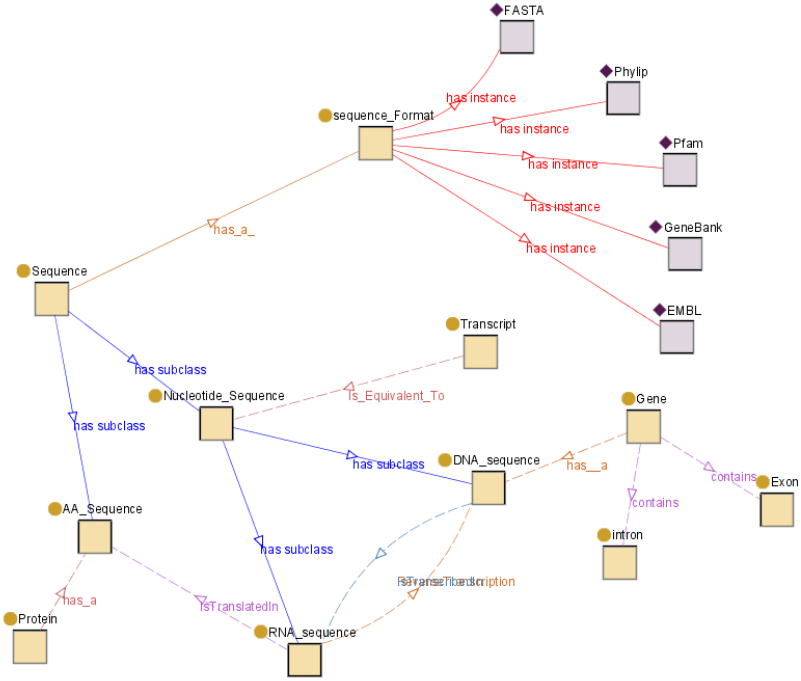


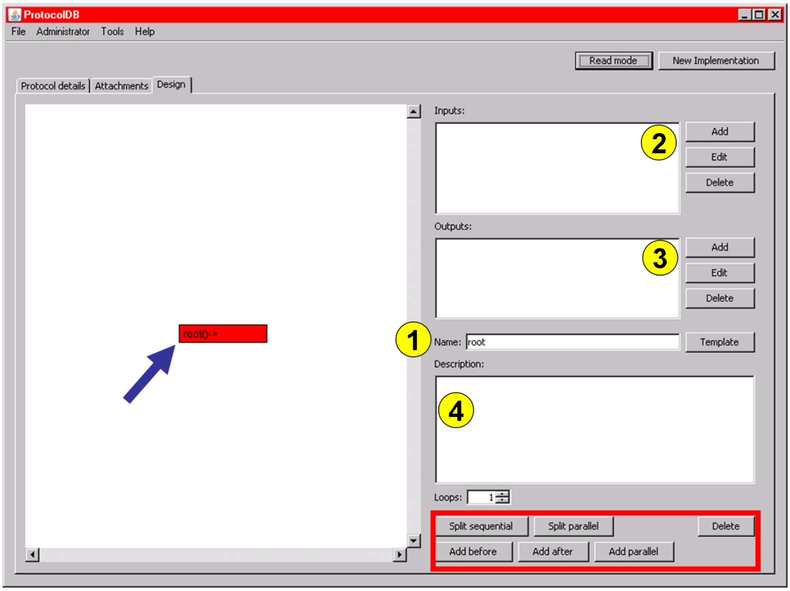
Fig. 3. Portion of a domain ontology

4 Implementation Protocol

An *implementation protocol* is a graph composed of connected scientific resources (database queries or tools) whose inputs and outputs are data types. A single bioinformatics service is an implementation protocol. Complex implementation protocols are composed of scientific resources connected with the same two binary operators • and ⊗ used to express design protocols. Each design task box of the design protocol is mapped to an implementation protocol. A single design task may be mapped to a complex implementation protocol invoking several bioinformatics resources, queries, and/or connectors to translate data formats (see Section 4.2). We first describe how services are selected to implement a design protocol in Section 4.1. We address the problem of service composition in Section 4.2.

4.1 Selecting Services

To enter an *implementation protocol* for a given design protocol, the user clicks on the `New Implementation` button that displays the tab `implementation`. Clicking on the implementation tab activates a new window where the user documents the implementation version to be entered. Then the implementation protocol



Name: root

Loops: 1

Fig. 4. ProtocolDB entry interface

may be entered with the window shown in Figure 5 opened by clicking on the `Implementation Diagram` tab.

Each selected design task (A) is first mapped to an implementation task box (B) that can be expanded with the two operators to create the corresponding implementation sub-protocol (red dark rectangle). Each implementation task may be instantiated with a bioinformatics resource by choosing a tool within a of resources (C).

4.2 Mapping Services

The selection of resources to implement a scientific protocol may lead to multiple successive attempts often failing because of syntactic, semantic, and efficiency reasons [5]. After discovering services that are relevant to the implementation of a protocol, the next step is to identify whether these services are *compatible*. Two services are semantically compatible when their respective input and output types refer to the same ontological class. However, semantic compatibility does not always correspond to syntactic interoperability. ProtocolDB relies on the use of domain ontologies to reconciliate conflicts occurring when a given scientific object is represented with different syntactical structures by bioinformatics resources. For instance, a scientist may select BLAT and SIM4 to implement the alignment design task identified in Section 2. The two alignment tools are semantically similar: their inputs and outputs are constructs of the same conceptual classes (Figure 3).

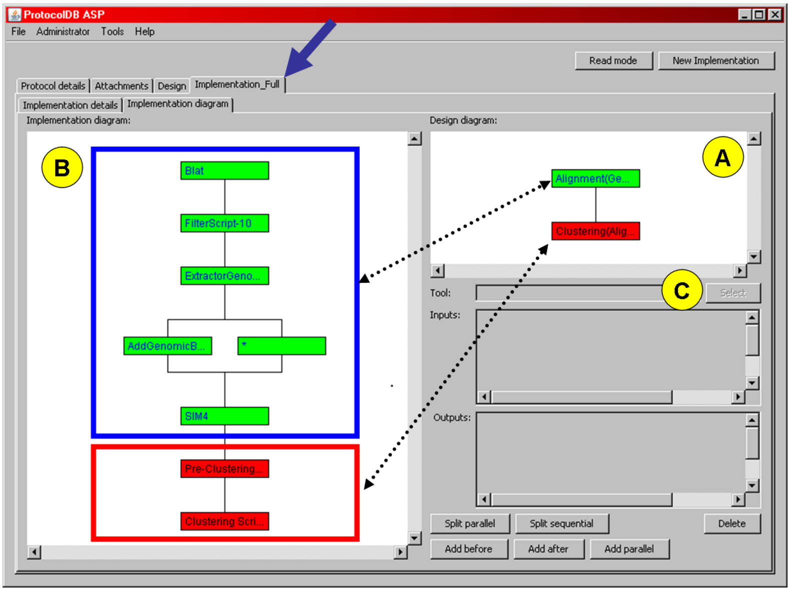
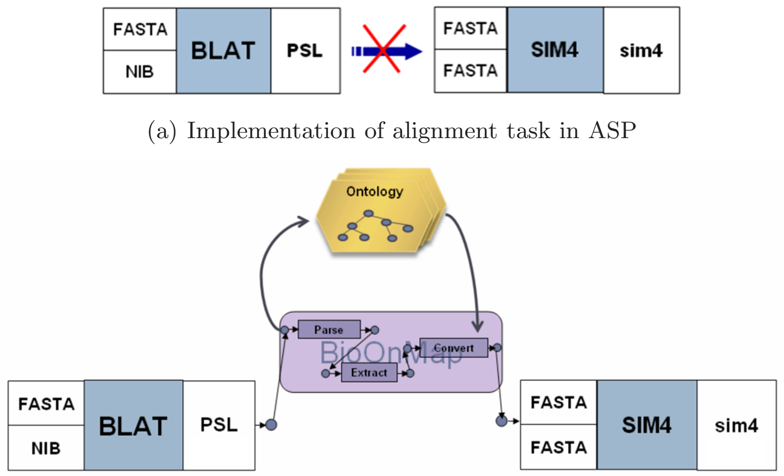


Fig. 5. ProtocolDB implementation protocol entry interface

The outputs and inputs of BLAT and SIM4 are closely related but cannot be composed (Figure 6(a)). An executable implementation would require a *connector* to compose the two services (Figure 6(b)). ProtocolDB relies on BioOnMap to generate mappings for data transformation and conversion [17,18] by exploiting domain knowledge expressed in an ontology. Because the description of the implementation of scientific protocols in ProtocolDB is mapped to a design protocol characterized by an ontology, the task of mapping bioinformatics services may exploit the domain knowledge of the ontology.

To generate a mapping, BioOnMap considers the *semantic type* of each input and output of services to identify whether the output of the former is compatible with the input of the latter. For instance, BLAT produces an output of semantic type aligned `Transcript` while SIM4 accepts as inputs a set of `DNA_sequence` and a set of `Sequence`. Exploiting the domain ontology depicted in Figure 3, BioOnMap may infer that a `Transcript` is semantically equivalent to a `Sequence` and a `DNA_sequence` is a sub-class of the same class, i.e., `Sequence`, which means that the two concepts `Transcript` and `DNA_sequence` are semantically equivalent. Consequently, BioOnMap infers that the two services are semantically compatible. On the syntactic side, the format expected by SIM4 is `FASTA` which is an instance of the `sequence_Format` class, the output format of BLAT is `PSL` which is also a `sequence_Format`. The mapping between the design and the implementation levels is given by the ontological relationship "a `sequence` has-a `sequence_Format`". `FASTA` and `PSL` are instances of the same conceptual class, they refer to different representations of the same scientific object, i.e., `Sequence`. In that case, the BioOnMap generates a connector service



(a) Implementation of alignment task in ASP

(b) Semantic mapping

Fig. 6. BLAT takes a set of transcripts in `FASTA` format and genomic sequences in `NIB` format and returns an alignment file in `PSL` format. SIM4 takes both inputs in `FASTA` format and produces an alignment in a format specific to SIM4.

that syntactically translates a `PSL` file into a `FASTA` file. Finally, each connector is invoked automatically in the implementation protocol. Once the implementation is completed, the protocol can be executed with a workflow system such as Taverna [14], Kepler [11], or Mobyle [12]. After execution, the data collected at each step of the protocol execution are stored in ProtocolDB and provide the fourth layer of our approach (bottom of Figure 2). The mapping of collected datasets to their corresponding implementation and design tasks provides the framework needed to reason about data provenance [2].

5 Discussion

The distinctive features of ProtocolDB presented in this paper include: 1) a two-layer model for the representation of protocols and 2) a light-weight semantic support by the use of domain ontologies that enhances significantly the composition and enactment of Web services.

ProtocolDB aims at providing support for designing, storing, and reasoning on scientific protocols. The two-layer representation of protocols with a *design protocol* mapped to one or several *implementation protocols* offers valuable functionalities to the user. The design protocol expresses the scientific aim in terms of classes and relationships of a domain ontology. An implementation protocol describes the way the scientific aim will be achieved. Because technology changes over time, it is a way to record within a laboratory the various ways a particular experiment is conducted identifying the machines, robots, and other technology involved in the process. Another benefit of the approach is to let the scientist explore and compare the performance of different implementations. ProtocolDB is not developed to execute scientific protocols but it is a system that offers the ability to reason about scientific protocols. BioOnMap allows support for composing services and generates protocols that can be executed. Future functionalities of the system include support to selection of resources suitable with the user's needs, prediction of the outcome of an execution (performance and quality of results), protocol re-use (query protocols, find similar protocols).

In contrast, workflow systems such as Taverna [13], Kepler [11], or Mobyle [12] enable the construction and execution of workflows over distributed Web services. These platforms are implementation-driven and they do not capture the scientific aim of the protocol. They do not provide a query language to query, compare, re-use scientific protocols stored in a repository. ProtocolDB aims at generating implementation protocols in a format compatible with these platforms so that once they have been entered, they can be easily uploaded and executed by these systems.

Expressing complex executable workflows remains a difficult, time-consuming, and expensive task. One of the reasons for this difficulty is the large number of bioinformatics resources. The paradigm of semantic Web services offers the possibility of highly flexible Web services architectures where new services can be quickly discovered, orchestrated, and composed into workflows [10]. Taverna relies on the Feta system to search semantically candidate services [9]. In the future we will integrate the Semantic Map [16] approach to ProtocolDB to guide scientists who wish to explore the maze of available resources to implement design tasks.

In the BioOnMap system [18], we present a formal framework of semantic descriptions for stateless services. From a syntactic point of view, our framework enhances resource description provided by OWL-S Service Profile (i.e., input and output description). Web services are being independently created by many parties worldwide, using different terminologies (ontologies) and datatypes, hindering their integration and reusability [19]. Taverna proposes a list of "*shims*", i.e., services that resolve basic syntactical mismatches in order to reconciliate closely related inputs and outputs. However, a new shim needs to be manually created for each pair of services that need to interoperate which make this manual approach not scalable. [11] describes a scalable framework that uses mappings to one or more ontologies for reconciling two services. Instead, the BioOnMap approach provides a uniform approach to workflow and data integration [17].

6 Conclusion

Recording scientific protocols together with experimental data is critical to scientific discovery. ProtocolDB presented in this paper is a system that assists scientists in the expression of protocols and provides a framework for protocol reuse and analysis, sharing, archival, and reasoning on provenance of experimental data. Our approach exploits a domain ontology to index semantically each protocol task and collected dataset. Scientific protocols are expressed as a pair of a design protocol that captures the scientific aim and one or more implementations where services are selected and composed into an executable workflow using BioOnMap. ProtocolDB provides the framework needed to record scientific protocols so that they can be reproduced, thus validating experimental results and to query, reuse, compare scientific protocols and their corresponding collected datasets. In the future we will include Semantic Map [16], a system designed to assist scientists in the selection of the bioinformatics resources to implement their protocols. ProtocolDB is available at http://bioinformatics.eas.asu.edu/siteProtocolDB/indexProtocolDB.htm.

Acknowledgments. This research was partially supported by the National Science Foundation[3] (grants IIS 0223042, IIS 0431174, IIS 0551444, IIS 0612273, and IIS 0738906). The authors would like to thank Louiqua Raschid, Maria Esther Vidal, Natalia Kwasnikowska, Jan Van den Bussche, Susan Davidson, and Sarah Cohen-Boulakia for their valuable input and exciting discussions.

References

1. Bartlett, J.C., Toms, E.G.: Developing a Protocol for Bioinformatics Analysis: An Integrated Information Behaviour and Task Analysis Approach. Journal of the American Society for Information Science and Technology 56(5), 469–482 (2005)
2. Cohen-Boulakia, S., Biton, O., Davidson, S.: Querying Biologically Relevant Provenance information in Scientific Workflow Systems with Zoom*UserViews. In: Proc. 33^{rd} International Conference on Very Large Data Bases (demonstration paper), Vienna, Austria, pp. 1366–1369 (2007)
3. Kinsy, M., Lacroix, Z.: Storing Efficiently Bioinformatics Workflows. In: IEEE 7th International Symposium on Bionformatics and Bioengineering, Boston, MA (2007)
4. Kwasnikowska, N., Lacroix, Z., Chen, Y.: Modeling and Storing Scientific Protocols. In: Meersman, R., Tari, Z., Herrero, P. (eds.) On the Move to Meaningful Internet Systems 2006: OTM 2006 Workshops. LNCS, vol. 4277, pp. 730–739. Springer, Heidelberg (2006)
5. Lacroix, Z., Legendre, C.: Analysis of a Scientific Protocol: Selecting Suitable Resources. In: Proc. First IEEE International Workshop on Service Oriented Technologies for Biological Databases and Tools, Salt Lake City, UT, July 9-13, pp. 130–137 (2007)
6. Lacroix, Z., Legendre, C., Raschid, L., Snyder, B.: BIPASS: BioInformatics Pipelines Alternative Splicing Services. Nucleic Acids Research, Volume Web Services Issue 35, 292–296 (2007)
7. Lacroix, Z., Raschid, L., Eckman, B.: Techniques for Optimization of Queries on Integrated Biological Resources. Journal of Bioinformatics and Computational Biology 2(2), 375–411 (2004)

[3] Any opinion, finding, and conclusion or recommendation expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

8. Lawson, A.E.: The nature and development of hypothetico-predictive argumentation with implications for science teaching. International Journal of Science Education 25(11), 1387–1408 (2003)
9. Lord, P.W., Alper, P., Wroe, C., Goble, C.A.: Feta: A light-weight architecture for user oriented semantic service discovery. In: Gómez-Pérez, A., Euzenat, J. (eds.) ESWC 2005. LNCS, vol. 3532, pp. 17–31. Springer, Heidelberg (2005)
10. McIlraith, S.A., Son, T.C., Zeng, H.: Semantic web services. IEEE intelligent systems 16(2), 46–53 (2001)
11. McPhillips, T.M., Bowers, S., Ludäscher, B.: Collection-oriented scientific workflows for integrating and analyzing biological data. In: Leser, U., Naumann, F., Eckman, B. (eds.) DILS 2006. LNCS (LNBI), vol. 4075, pp. 248–263. Springer, Heidelberg (2006)
12. Néron, B., Tufféry, P., Letondal, C.: Mobyle: a Web portal framework for bioinformatics analyses. In: Network Tools and Applications in Biology (poster), Naples, Italy (2005)
13. Oinn, T., Greenwood, M., Addis, M., Alpdemir, M.N., Ferris, J., Glover, K., Goble, C., Goderis, A., Hull, D., Marvin, D., Li, P., Lord, P., Pocock, M.R., Senger, M., Stevens, R., Wipat, A., Wroe, C.: Taverna: lessons in creating a workflow environment for the life sciences. Concurrency and Computation: Practice and Experience 18(10), 1067–1100 (2006)
14. Oinn, T.M., Addis, M., Ferris, J., Marvin, D., Senger, M., Greenwood, R.M., Carver, T., Glover, K., Pocock, M.R., Wipat, A., Li, P.: Taverna: a tool for the composition and enactment of bioinformatics workflows. Bioinformatics 20(17), 3045–3054 (2004)
15. Stevens, R., Goble, C., Baker, P., Brass, A.: A Classification of Tasks in Bioinformatics. Bioinformatics 17(2), 180–188 (2001)
16. Tufféry, P., Lacroix, Z., Ménager, H.: Semantic Map of Services for Structural Bioinformatics. In: Tufféry, P., Lacroix, Z. (eds.) Proc. of the 18th International Conference on Scientific and Statistical Database Management, Washington DC, pp. 217–224 (2006)
17. Yacoubi, N., Lacroix, Z.: Resolving Scientific Service Interoperability With Schema Mapping. In: Proc. IEEE 7th International Symposium on Bionformatics and Bioengineering, Boston, MA, pp. 14–17 (2007)
18. Yacoubi, N., Lacroix, Z., Vidal, M.-E., Ruckhaus, E.: Deductive Web Services: an Ontology-driven Approach to Service Composition and Data Integration. In: Proc. International on Semantic Web and Web Semantics, Vilamoura, Portugal, pp. 29–30 (2007)
19. Zamboulis, L., Martin, N., Poulovassilis, A.: Bioinformatics Service Reconciliation By Heterogeneous Schema Transformation. In: Data Integration in the Life Sciences. LNCS, vol. 4544, pp. 89–104. Springer, Heidelberg (2007)