Treating "shimantic web" syndrome with ontologies

Duncan Hull, Robert Stevens, Phillip Lord, Chris Wroe, and Carole Goble

School of Computer Science, University of Manchester, Oxford Rd, Manchester, UK.

Abstract. This paper describes shimantic web syndrome, the use of "shims" to align or mediate mismatching third party Web Services that have closely related, but incompatible, inputs and outputs. The syndrome is illustrated using services from ^{my}Grid bioinformatics analyses. An ontology driven treatment for managing this syndrome through semi-automated service discovery and invocation is outlined. This treatment is likely to be applicable to other domains.

1 Introduction

In the vision of Semantic Web Services, machine understandable descriptions of data and Web Services facilitate their automated use. Yet, as ever, the devil is in the detail. We observe the need to use *shims* to align Web Services to make them useful or even to work at all. Shims align or mediate data that is syntactically or semantically closely related, but not directly compatible. As an example, we use Services from ^{my}Grid workflows that perform bioinformatics analyses [1]. A significant proportion of the Web Services in these workflows are shims, not directly relevant to the biological purpose of the workflow but required to mediate between outputs and inputs of consecutive services.

Shims are an important part of this workflow because bioinformatics data and services are autonomously created by different groups around the world. Consequently, there is no universally accepted data model for describing the data inputs and outputs that services operate upon. Standards like XML Schema, if used at all, rarely describe more than primitive types like `xsd:string` [2]. These are of limited use when mediating between services that operate on complex structured types like GenBank[1] and UniProt records[2] or BLAST reports[3].

Superficially, services that produce and consume `xsd:string` all match and are compatible with each other. However, because `xsd:string` hides many different complex data types, the degree of matching actually falls into three categories

1. Exact match: output and input are equivalent and compatible
2. Close match: some kind of shim required to align services and bridge gap
3. No match: either syntactic or semantic, services are incompatible

[1] See `http://www.ncbi.nlm.nih.gov/Sitemap/samplerecord.html`
[2] The UNIversal PROTein resource `http://www.uniprot.org`
[3] see `http://www.ncbi.nlm.nih.gov/Education/BLASTinfo/information3.html`

The second scenario, close matching, is common in bioinformatics and exemplifies shimantic web syndrome, where many services are *nearly* compatible. For example, a service producing a UniProt record (which contains a protein sequence) and a BLAST service consuming protein sequences are nearly compatible and require shimming. An `accessor` shim (see Table 1) is required to access the protein sequence from the UniProt record so that it can be passed to BLAST for further analysis. Given the open nature of the Web and the autonomy of the data and service providers [3] this syndrome is unlikely to be cured in the foreseeable future. What distinguishes our work from related projects integrating biological data on the semantic web is, firstly we aim to accommodate autonomous Web Services in an *open* world. Secondly, some of these services do not, and may never, use XML for their data, but pass around strings encapsulating many legacy proprietary file formats. Finally, our motivating examples are taken from real live scenarios using complex data where automation can be dangerous or impossible.

In this paper, we characterise the shimming problem and introduce the potential role of semantic description of services that aim to plug the gap left by the absence of effective, domain specific type systems in aligning third party services.

2 The shimming problem

Shims are symptomatic of integrating implicitly typed bioinformatics data. They are analogous to the shims in the physical world – thin strips of metal used to align pipes or rails. In bioinformatics, shims align data by performing some of the operations normally facilitated by a type system. Some example shims are shown in Table 1. Our shims are subclasses of WSMO[4] mediators, probably `wwMediators` [4]. However, an important difference is that the deployment and invocation of some bioinformatics shims may be difficult to fully automate, because their safe use can be context dependent. The `semantic translator` is one example of this.

We feel the shim metaphor is a useful one for describing and understanding the software components currently required to integrate and understand bioinformatics data. Our definition for a shim is software that transforms between closely related data (either syntactically or semantically) in order to join outputs and inputs of two components, in this case Web Services. As shims are hard to precisely define, we think of shims as a set of symptoms of integrating weakly or implicitly typed (both semantically and syntactically) data. Characterising shims is the first step in working towards novel treatments for "shimantic web" syndrome; which is not peculiar to bioinformatics. In an open world such as the Web, it is likely that the majority of services will need some kind of shim in order to align them with the user's needs.

[4] `http://www.wsmo.org`

Shim type	Input	Operation	Output	Description
Dereferencer	Identifier or Pointer	Dereference	The dereferenced resource	GenBank ID replaced with GenBank record
Syntax translator	Data represented in concrete representation, x	Translate	Data represented in an alternate concrete representation, y	SeqRet translates between representations of sequence data.
Semantic translator	DNA sequence	Translate	Protein sequence	Translate DNA into protein
Mapper	Unique identifier	Map	Unique Identifier	Maps between IDs. E.g. GenBank to EMBL
Parser	Record	Parse	Abstract syntax tree	Parse BLAST report.
Iterator	A set	Iterate	A member of a set	Iterate over members of a given set
Comparer	Two or more sets	Diff	Set of differences	Comparing BLAST reports notifies of new sequences
Accessor	Record	Access	Subset of record	Access a subset

Table 1. Examples of shims taken from the ^{my}Grid project. This partial classification is based on inputs, outputs and the task or operation performed.

3 Semantic approaches to type systems

The capabilities of a type system would alleviate some problems that shims currently address. Being able to coerce, access, infer, reflect and cast data would all be easier if bioinformatics data were more explicitly typed. However, creating a type system for a distributed, dynamic and complex domain like bioinformatics is a non-trivial task fraught with many technical and social pitfalls. Registries of Web Services such as BioMOBY [5] have succeeded because they have allowed data providers to register services quickly with only minimal typing. We cannot expect all bioinformatics data to be typed both semantically and syntactically in order to allow smoother mediation. Some systems [6] impose an internal type system on top of third party services via XML. While providing a working solution, we feel that such an approach is less scalable and restricts the number of services available. In the ^{my}Grid project, we have adopted an approach of using third party services in their native form.

In the ^{my}Grid project we have created an ontology to describe bioinformatics data and services [7] and we are currently extending this model to specifically tackle this syndrome by describing the shim services, the data they process and the main bioinformatics services that the shim services mediate between. The ontology describes the existence of data types without the detailed description of structure, and can map to XML schemas when and where they exist. It describes services semantically so that we know that the output "UniProt record" and input "protein sequence" are closely related and can be mapped between using an `accessor` shim, (see Table 1) to access the protein sequence contained in the UniProt record. This process currently requires human intervention; however,

with the help of an ontology, these sorts of transformations can be supported and the user guided to make biologically sensible workflows.

We intend to use the ontology when constructing workflows to recognise service mismatches and identify what kind of shim is required. With this information, a suitable shim or shims could be automatically retrieved from a library. Our goal is to make shim management more transparent to the user. By describing the transformations shims perform, the user composing the workflow can be abstracted away from the details of mediating the underlying services.

4 Conclusion

We have used bioinformatics as our example of shimantic web syndrome, but we suggest this problem will be found throughout the development of applications using autonomous Web Services. There is a high probability that third party services will not *exactly* match the user's needs but will be closely related; this is when a shim will be needed. We propose to treat the syndrome as follows

1. Describe and classify shims using an ontological model
2. Create a library or factory of shim services
3. Use model to identify closely related, but mismatching services
4. Semi-automate discovery and invocation of shims, using the above.

Whichever semantic web architecture is used, successful adoption by the bioinformatics community will require mechanisms for describing, discovering and composing shim services that currently make mediation possible.

References

1. Robert D. Stevens, Hannah J. Tipney, Chris Wroe, Tom Oinn, Martin Senger, Phillip W Lord, Carole A. Goble, Andy Brass, and May Tassabehji. Exploring Williams-Beuren Syndrome Using myGrid. In *Intelligent Systems for Molecular Biology, Glasgow, UK.*, volume 20, 2004. ISSN 1367-4803.
2. Phillip Lord, Sean Bechhofer, Mark D. Wilkinson, Gary Schiltz, Damian Gessler, Duncan Hull, Carole Goble, and Lincoln Stein. Applying semantic web services to bioinformatics: Experiences gained, lessons learnt. 2004. Proceedings of the 3rd International Semantic Web Conference, Hiroshima, Japan.
3. Lincoln Stein. Creating a bioinformatics nation. *Nature*, (417):119–120, May 2002.
4. Massimo Paolucci, Naveen Srinivasan, and Katia Sycara. Expressing WSMO Mediators in OWL-S. *International Semantic Web Conference 2004*, W6:120–134, 2004. Workshop notes VI: Semantic Web Services.
5. M Wilkinson and M Links. BioMOBY: An Open Source Biological Web Services proposal. *Briefings in Bioinformatics*, 3(4):331–341, 2002.
6. Shawn Bowers and Bertram Ludäscher. An ontology-driven framework for data transformation in scientific workflows, 2004. Intl. Workshop on Data Integration in the Life Sciences (DILS'04), March 25-26, 2004 Leipzig, Germany, LNCS 2994.
7. Chris Wroe, Robert Stevens, Carole Goble, Angus Roberts, and Mark Greenwood. A Suite of DAML+OIL Ontologies to Describe Bioinformatics Web Services and Data. *International Journal of Cooperative Information Systems*, 12(4):197–224, June 2003.