SOFTWARE

Executing SADI services in Galaxy

Mikel Egaña Aranguren*, Alejandro Rodríguez González and Mark D. Wilkinson

*Correspondence: mikel.egana.aranguren@gmail.com
Biological Informatics, Centre for Plant Biotechnology and Genomics (CBGP), Technical University of Madrid (UPM), Campus of Montegancedo, 28223 Pozuelo de Alarcón, Spain
Full list of author information is available at the end of the article

Abstract

Background: In recent years Galaxy has become a popular workflow management system in bioinformatics, due to its ease of installation, use and extension. The availability of Semantic Web-oriented tools in Galaxy, however, is limited. This is also the case for Semantic Web Services such as those provided by the SADI project, *i.e.* services that consume and produce RDF. Here we present SADI-Galaxy, a tool generator that deploys selected SADI Services as typical Galaxy tools.

Results: SADI-Galaxy is a Galaxy tool generator: through SADI-Galaxy, any SADI-compliant service becomes a Galaxy tool that can participate in other out-standing features of Galaxy such as data storage, history, workflow creation, and publication. Galaxy can also be used to execute and combine SADI services as it does with other Galaxy tools. Finally, we have semi-automated the packing and unpacking of data into RDF such that other Galaxy tools can easily be combined with SADI services, plugging the rich SADI Semantic Web Service environment into the popular Galaxy ecosystem.

Conclusions: SADI-Galaxy bridges the gap between Galaxy, an easy to use but "static" workflow system with a wide user-base, and SADI, a sophisticated, semantic, discovery-based framework for Web Services, thus benefiting both user communities.

Keywords: Galaxy; Web Services; SADI; RDF; SPARQL; OWL

Background

Introduction

In silico research involves multiple steps of data integration and processing, generally making-use of a wide range of online databases and tools in addition to local applications. There are various approaches to undertaking these analyses, ranging from stepwise manual and error-prone copy/paste processes, *ad hoc* scripted pipelines that automate the flow of data through a series of analytical steps, through to formal pipelines created in a workflow environment such as Taverna [1], Kepler [2], or Galaxy [3] where complex pipelines can be constructed with little or no programming knowledge. With recent calls from organizations such as the U.S. Institute of Medicine insisting that high throughput biomedical studies be conducted with more rigor using less error-prone tools [4], it seems likely that there will be increased pressure on biologists to make use of these more formal workflow environments.

In parallel with the move to formal workflows, there is also a growing global movement towards representation of bioinformatics data and knowledge using contemporary syntaxes and semantic languages approved by the World Wide Web Consortium (W3C). Resource Description Framework (RDF) [5] is a syntax for data representation that is tightly wedded with the core functionality of the Web, while Web Ontology Language (OWL) [6] is the Knowledge Representation language recommended for encoding machine-readable semantics that aid in data discovery, annotation, and interpretation. Major bioinformatics resources making their data available using these formats include UniProt [7], EBI [8], and soon, the DNA Databank of Japan [9]. Beyond these core providers, there are also large integrated warehouses of bioinformatics data in RDF format including, most significantly, Bio2RDF [10], which integrates critical bioinformatics resources such as Affymetrix, dbSNP, OMIM, and KEGG.

At the intersection of these two trends is the inevitable requirement for tools that support the flow of native RDF data through a formal bioinformatics analysis pipeline. The Semantic Automated Discovery and Integration (SADI) project has established design-patterns for bioinformatics resources that wish to natively consume and produce RDF data [11], and there are SADI plug-ins to several popular data workflow and exploration environments, including Taverna and the IO Informatics Sentient Knowledge Explorer [12]. While Taverna is a rich and full-featured environment for constructing and editing complex workflows, Galaxy is showing itself to be a favorite of bench-biologists due to its relative simplicity compared to Taverna, and the “familiarity” it brings biologists by exposing the tools they commonly use in a manner that they can quickly interpret and work with. As such, it was desirable to bring support for SADI-based, RDF-native data and analysis tools into the Galaxy environment.

Here we describe SADI-Galaxy - a set of scripts that retrieve and “wrap” SADI Semantic Web Services in a manner that allows them to be included in Galaxy workflows.

Galaxy

Galaxy is a Web server, written in Python, that offers a very usable interface for the typical bioinformatics computational analyses (Figure 1). A user can store data, and analyse it using a variety of tools, sending the output of one tool as input to the next. This process is stored in a chronological history from which workflows can be extracted, providing an easy-to-reproduce abstraction of common steps. Data, histories and workflows are Web-shareable and can be imported and exported, building an ecosystem for more reproducible computational science [13], particularly when “Galaxy pages” are used (*e.g.* [14]). The usual practice for users of Galaxy is to create an account in a public Galaxy server, such as “main” [15], or install a local Galaxy server if specific tools are needed and/or sensitive data is analysed.

A Galaxy tool (the menu items in the left column of Figure 1) is, typically, a wrapper for a terminal executable program. Since such wrappers are defined by an XML file describing the inputs and outputs of the tool as well as its Web interface [16], creating Galaxy tools from pre-existing executables is not technically demanding.

SADI

SADI is a standards-compliant set of lightweight design patterns for publishing bioinformatics data and analysis services on the Semantic Web. It uses Semantic Web technologies at every level of the Web services “stack”. In particular, a SADI service describes its interface in OWL, and then both consumes and produces RDF data that match that OWL logical description. Finally, SADI requires that the output data is semantically connected to the input data by a meaningful relationship. As such, workflows of SADI services output unbroken chains of RDF Linked Data [17]. SADI services are catalogued in a publicly-accessible database (registry), and queries against that database will be used in this study to find, and retrieve the interface definitions for, SADI services of interest to any given Galaxy user; the retrieved service definitions will become templates for the Galaxy "wrapper", and thus the services can be accessed through these "wrappers".

Implementation

SADI-Galaxy consists of two parts (Figure 2): the "Core", and the Tool-generator. SADI-Galaxy Core includes two normal Galaxy tools that are installed in the same manner as any other Galaxy tool [18]. These two tools are:

- SADI generic client. This tool is able to execute any SADI service, given the service's URI and an RDF input file that is compatible with the service. The RDF output of the service is stored as any Galaxy output. Automated reasoning is used to check if the RDF input is compliant with the SADI Service's input class; *i.e.* whether the RDF instance is inferred to be a member of the input OWL Class.
- RDF Syntax Converter. This tool is able to convert an RDF file into a variety of formats, most importantly including tab-delimited data (three columns for subject, predicate and object), so that non-RDF-based Galaxy tools can consume SADI's output.

SADI-Galaxy then offers an additional, more advanced functionality through the Tool-generator (Figure 3). The Tool-generator adds the ability to query a given SADI Service registry, using arbitrary query parameters, to retrieve a set of matching SADI services. These Services will then be deployed in Galaxy together with the Core tools described above. The Tool-generator, at the code-level, is simply a command line executable that reads one or more SPARQL [19] queries and executes them against a SADI registry. The matching service URIs retrieved by the query are then used to generate, for each service, a Galaxy compliant wrapper and install it as a new Tool.

The SPARQL queries used by the Tool-generator can be tuned by the user in order to install a concrete set of services. Query examples are provided in the bundle, as well as a query generator that is able to produce parameterised queries from a base query (*e.g.* for different publishers). The default query can be seen in Figure 4.

The Galaxy tools created by the Tool-generator utilize the Core's generic SADI client, but pre-configure many of the parameters of that generic tool; thus rather than requiring users to know, for example, the address and data-types of a service, each is provided as an independent Galaxy tool, enabling simple invoking and storing of desired SADI tools as "bookmarks", and for using Galaxy's tool-search function to explore SADI Service descriptions. In this way, the same SADI Service can be invoked either through the generic client (if the URI is known by the user) or through the corresponding Galaxy tool (if the URI is unknown and/or the service is used frequently).

Results and discussion

The use case depicted in Figures 5 and 6 shows how SADI services can be exploited as part of Galaxy[a]: it demonstrates how data can be transformed to RDF, in order for SADI services to consume it, and how different SADI services can be combined as regular Galaxy tools. The aim of the workflow is to obtain the UniProt entry associated with a PDB entry [20], that is, to obtain information of the protein whose 3D structure is described in PDB. The workflow starts from a simple tab delimited file (as it is customary in Galaxy) and the `TAB2RDF` tool[b], part of the SPARQL tools official tool suite [21], is used to transform it to the RDF/XML syntax that a SADI service can consume. The file is submitted to the SADI service `pdb2uniprot` to obtain the UniProt ID, which is returned as an RDF file. The RDF file is submitted to another SADI service, `uniprotInfo`, to obtain all the information of the entry, also as an RDF file. The final RDF file can be converted to a tab delimited file with RDF2TAB, or queried with SPARQL-Galaxy to obtain concrete information about the protein [22].

SADI-Galaxy is inspired by the Galaxy Web Services Extensions tool [23, 24], which is able to dynamically load SAWSDL/WSDL web services as Galaxy tools. SADI-Galaxy focuses on SADI's Semantic Web-compliant services, rather than SAWSDL/WSDL services, and therefore expands the target audience for Galaxy in general. Although SADI-Galaxy is unable to dynamically load Tools into Galaxy's left column, SADI tools will become available after a restart of Galaxy. Moreover, any Service can be executed dynamically by simply providing the service's URI to the generic tool, which is equivalent to dynamically loading it. More importantly, dynamic tool loading is not yet part of the standard Galaxy distribution and requires the modification of Galaxy's core Python code, thus SADI-Galaxy does not go outside of the normal functioning of Galaxy itself. Finally, SADI-Galaxy offers the possibility of querying a SADI registry to automatically generate a large number of desirable Galaxy tools, a powerful feature not available in the WS extensions tool.

Another system to some extent comparable to SADI-Galaxy is Tavaxy [25], which is a standalone server (not part of Galaxy nor a plugin for Taverna) that offers the possibility of mixing Taverna and Galaxy workflows. However, unlike Tavaxy, SADI-Galaxy's aim is to provide a light extension to the default Galaxy distribution (SADI-Galaxy is just a regular Galaxy tool, in the case of the generic caller) in order to use SADI services. One can, of course, use Taverna to build a workflow that exploits SADI services, import it to Tavaxy, and then mix it with a Galaxy workflow, but: 1) The process is rather complex and it does not happen in a regular Galaxy server (we want users to use SADI services as seamlessly as possible, within Galaxy, without having to use another system in-between); 2) Querying a SADI registry to obtain the services and generate the tools is not possible.

We suggest SADI-Galaxy as the "minimum" infrastructure that marks the point of intersection between SADI Services and Galaxy, which can now act as the core codebase upon which more powerful functionality is constructed. In particular, we expect two major developments in the near future: discovery, where given an RDF input, Galaxy is able to infer and automatically select the appropriate SADI Tool; and adding tools dynamically as WS Extensions already does, once a consensus has been reached by Galaxy developers on how to implement a standard function, included in Galaxy's core API, for dynamic tool loading.

Conclusions

The simplicity and predictability of the SADI Service design patterns - effectively, consume and produce raw RDF over HTTP POST - has allowed us to create a highly generic service invocation infrastructure; *i.e.*, unlike in other Web Service frameworks, a client need only declare *what* it wants, not *how* to obtain it. Building a framework that focuses on semantics, rather than syntax - not only for the data itself, but also for the messaging infrastructure - means that the client can be largely agnostic so as to how to invoke any service, making the necessary decisions on an *ad hoc* basis at invocation-time. For example, in SADI-Galaxy, when combining different SADI services, the RDF of the intermediate steps can just as easily be consumed by any other RDF-based tool.

In combination with Galaxy's easy-to-use platform for storing data, programs for analysing data, and the resulting workflows, we find this a near-ideal "ecosystem" to provide SADI services to our target end-users in a very familiar and straightforward manner. We hope that, by providing SADI services as part of the widely-used Galaxy platform, we can encourage the more rapid adoption of these powerful new Semantic Web technologies.

Availability and requirements

- Project name: SADI-Galaxy.
- Project home page: http://github.com/mikel-egana-aranguren/SADI-Galaxy[c].
- Operating system(s): UNIX-based (GNU/Linux, Mac OS X, *BSD, *etc.*).
- Programming language: Java, Python, and Shell Script.
- Other requirements: a working Galaxy installation is needed (http://galaxy.psu.edu/).
- License: General Public License (http://www.gnu.org/copyleft/gpl.html) version 3.

List of abbreviations used

OWL: Web Ontology Language; RDF: Resource Description Framework; SPARQL: SPARQL Protocol and RDF Query Language; URI: Uniform Resource Identifier; W3C: World Wide Web Consortium; SADI: Semantic Automated Discovery and Integration; XML: eXtensible Markup Language.

Competing interests
The authors declare that they have no competing interests.

Author's contributions
Mikel Egaña Aranguren and Alejandro Rodríguez González developed SADI-Galaxy. Mark D. Wilkinson contributed with test-cases, wrote and revised portions of the manuscript, and supervises the Biological Informatics laboratory.

Acknowledgements
Mikel Egaña Aranguren is funded by the Marie Curie Cofund programme (FP7) of the European Union. Alejandro Rodríguez González and Mark D Wilkinson are funded by the Isaac Peral Programme of the CBGP-UPM.

Endnotes

[a]The use case can be explored and executed in the following Galaxy page (Also available in the same Galaxy server through "shared data" and then "published pages"): http://biordf.org:8983/u/mikel-egana-aranguren/p/sadi-galaxy-jbms-use-cases. In order to reproduce it, a user must be created in the Galaxy server and the history and the workflow imported, so that the first item of the history can be used as input for the workflow.

[b]A locally modified version of `SPARQL tools` was used, adding the possibility of rendering user defined namespaces. A patch has been submitted to the original author; the modified version is included in SADI-Galaxy and can be also obtained at http://github.com/mikel-egana-aranguren/SPARQL_tools_tab2rdf.

[c]See also `TAB2RDF` (http://github.com/mikel-egana-aranguren/SPARQL_tools_tab2rdf), `SADI generic client` (http://toolshed.g2.bx.psu.edu/repos/mikel-egana-aranguren/sadi_generic/) and `SPARQL Galaxy` (http://toolshed.g2.bx.psu.edu/repos/mikel-egana-aranguren/sparql_galaxy/).

References

1. Wolstencroft, K., Haines, R., Fellows, D., Williams, A., Withers, D., Owen, S., Soiland-Reyes, S., Dunlop, I., Nenadic, A., Fisher, P., Bhagat, J., Belhajjame, K., Bacall, F., Hardisty, A., de la Hidalga, A.N., Balcazar Vargas, M.P., Sufi, S., Goble, C.: The taverna workflow suite: designing and executing workflows of web services on the desktop, web or in the cloud. Nucleic Acids Research **41**(W1), 557–561 (2013)
2. Ludäscher, B., Altintas, I., Berkley, C., Higgins, D., Jaeger, E., Jones, M., Lee, E.A., Tao, J., Zhao, Y.: Scientific workflow management and the kepler system. Concurrency Computat.: Pract. Exper. **18**(10), 1039–1065 (2006)
3. Goecks, J., Nekrutenko, A., Taylor, J., Galaxy Team: Galaxy: a comprehensive approach for supporting accessible, reproducible, and transparent computational research in the life sciences. Genome biology **11**(8), 86 (2010)
4. The Institute of Medicine of the National Academies: Evolution of Translational Omics Lessons Learned and the Path Forward. http://www.iom.edu/Reports/2012/Evolution-of-Translational-Omics/Report-Brief.aspx. Online; accessed 26-November-2013
5. W3C: RDF current status. http://www.w3.org/standards/techs/rdf. Online; accessed 28-March-2012
6. W3C: OWL Web Ontology Language current status. http://www.w3.org/standards/techs/owl. Online; accessed 28-March-2012
7. Jain, E., Bairoch, A., Duvaud, S., Phan, I., Redaschi, N., Suzek, B., Martin, M., McGarvey, P., Gasteiger, E.: Infrastructure for the life sciences: design and implementation of the UniProt website. BMC Bioinformatics **10**(1), 136 (2009)
8. EBI: EBI RDF Platform. http://www.ebi.ac.uk/rdf/. Online; accessed 26-November-2013
9. Kosuge, T., Mashima, J., Kodama, Y., Fujisawa, T., Kaminuma, E., Ogasawara, O., Okubo, K., Takagi, T., Nakamura, Y.: DDBJ progress report: a new submission system for leading to a correct annotation. Nucleic Acids Research (2013)
10. Belleau, F., Nolin, M., Tourigny, N., Rigault, P., Morissette, J.: Bio2RDF: Towards a mashup to build bioinformatics knowledge systems. Journal of Biomedical Informatics **41**(5), 706–716 (2008)
11. Wilkinson, M., Vandervalk, B., McCarthy, L.: The semantic automated discovery and integration (SADI) web service Design-Pattern, API and reference implementation. Journal of Biomedical Semantics **2**(1), 8 (2011)
12. IO Informatics: IO Informatics Sentient Knowledge Explorer. http://www.io-informatics.com/products/sentient-KE.html. Online; accessed 26-November-2013
13. Sandve, G.K., Nekrutenko, A., Taylor, J., Hovig, E.: Ten simple rules for reproducible computational research. PLoS Comput Biol **9**(10), 1003285 (2013)
14. Goto, H., Dickins, B., Afgan, E., Paul, I.M., Taylor, J., Makova, K.D., Nekrutenko, A.: Dynamics of mitochondrial heteroplasmy in three families investigated via a repeatable re-sequencing study. http://wiki.g2.bx.psu.edu/Admin/Tools/ToolConfigSyntax. Online; accessed 26-November-2013
15. Galaxy project: Galaxy main server. https://usegalaxy.org/. Online; accessed 26-November-2013
16. Galaxy project: Galaxy Tool XML File. http://wiki.g2.bx.psu.edu/Admin/Tools/ToolConfigSyntax. Online; accessed 28-March-2012
17. Bizer, C., Heath, T., Berners-Lee, T.: Linked Data - The Story So Far. International Journal on Semantic Web and Information Systems (IJSWIS) **5**(3), 1–22 (2009)
18. Mikel Egaña Aranguren: SADI generic. http://toolshed.g2.bx.psu.edu/repos/mikel-egana-aranguren/sadi_generic/. Online; accessed 26-November-2013
19. W3C: SPARQL current status. http://www.w3.org/standards/techs/sparql. Online; accessed 28-March-2012
20. Berman, H.M., Westbrook, J., Feng, Z., Gilliland, G., Bhat, T.N., Weissig, H., Shindyalov, I.N., Bourne, P.E.: The protein data bank. Nucleic Acids Research **28**(1), 235–242 (2000)
21. Sem4j: SPARQL tools. http://toolshed.g2.bx.psu.edu/repos/sem4j/sparql_tools. Online; accessed 26-November-2013
22. Mikel Egaña Aranguren and Alejandro Rodríguez González: SPARQL Galaxy. http://toolshed.g2.bx.psu.edu/repos/mikel-egana-aranguren/sparql_galaxy. Online; accessed 26-November-2013
23. Dhamanaskar, A., Cotterell, M.E., Zheng, J.Z., Miller, J.A., Kissinger, J.C., Stoeckert, C.J.: Suggestions in galaxy workflow design based on ontologically annotated services. In: Proceedings of the 7th International Conference on Formal Ontology in Information Systems (FOIS'12), pp. 29–42 (2012)
24. Wang, R., Brewer, D., Shastri, S., Swayampakula, S., Miller, J.A., Kraemer, E.T., Kissinger, J.C.: Adapting the galaxy bioinformatics tool to support semantic web service composition. In: Proceedings of the 2009 Congress on Services - Part I, pp. 283–290 (2009)

25. Abouelhoda, M., Issa, S., Ghanem, M.: Tavaxy: Integrating taverna and galaxy workflows with cloud computing support. BMC Bioinformatics **13**(1), 77 (2012)

Figures

Figure 1 Galaxy interface. The Galaxy main interface consists of three columns: available tools (left), current tool (center), and history of executed tools and obtained/uploaded data (right).

Figure 2 SADI-Galaxy tools. Galaxy interface (only left column) resulting from the installation of the Core (SADI generic client and RDF Syntax Converter, under "SADI COMMON UTILITIES" on the picture) and a number of specific SADI services, retrieved by a SPARQL query in the Tool-generator (Under "SADI SERVICES"; only a few shown).

Figure 3 SADI-Galaxy Tool-generator. The Tool-generator is executed with a Shell script that reads a SPARQL query and generates XML files (an XML tool file for each SADI service) that are copied to the appropriate location on a Galaxy server.

Figure 4 Tool-generator default query. This query returns around 250 active SADI services from the default registry. Other queries can be defined, as long as the `?s` variable is used for service URIs. For example, the `?org dc:publisher "wilkinsonlab.info"` triple can be defined to retrieve SADI services provided by `wilkinsonlab.info`. Examples are included in the SADI-Galaxy bundle.

Figure 5 Galaxy workflow. The workflow starts with a tab delimited file containing the information that will be sent as input to the SADI service `pdb2uniprot`. The tab file is processed to convert it to a column format Galaxy can recognise and then converted to RDF with the `tab2rdf` tool from `SPARQL tools`. The RDF file is submitted to `pdb2uniprot` using the SADI generic client and the output RDF is sent to the second SADI service, namely `uniprotInfo`, also with the SADI generic client. The output RDF from `uniprotInfo` can be converted to a tab delimited file with `RDF2TAB` or queried with `SPARQL-Galaxy` to obtain concrete information (Also in tab-delimited format).

Figure 6 Galaxy workflow, simplified. The workflow from Figure 5 is shown in a simplified version, with details for files. Not all the triples of the RDF files are shown. The workflow starts from a tab file (dotted square at the top left) and it proceeds to the first Galaxy tool (orange square). The first RDF file is produced (graph within dotted square) and then submitted to a SADI service (another Galaxy tool). The output is submitted to another SADI service and the final result is then converted to a tab delimited file and queried with SPARQL Galaxy (Not shown).