2009 Ninth International Conference on Intelligent Systems Design and Applications

A Semantic Infrastructure for the Integration of Bioinformatics Services

Giorgos Zacharioudakis Lefteris Koumakis Stelios Sfakianakis Manolis Tsiknakis

Foundation for Research and Technology-Hellas,
Institute of Computer Science (FORTH-ICS),
Heraklion, Crete, Greece
E-mail: {gzaxar,koumakis,ssfak,tsiknaki}@ics.forth.gr

Abstract

Web services integration has been a vigorous research area for the last years. With the introduction of the Semantic Web the publication of expressive metadata in a shared knowledge framework enables the deployment of services that can intelligently use web resources. Syntactic and semantic interoperability of services is crucial for services integration and complex scientific workflows creation. In this paper we propose a semantic based infrastructure for bioinformatics services integration that is designed and implemented in the context of the ACGT European project. This infrastructure features the integration of many different service registries in unified "meta-repository" and provides a knowledge based querying facility.

1. Introduction

Recent advances in many areas of genomics research resulted in the production of huge amounts of data that need to be managed, analyzed, processed, interpreted, and reason about. The volume of these data in combination with their inherent heterogeneity put a lot of stress on the Knowledge Discovery and Data Mining tools. To this end, a cooperative environment which enables the sharing of data, resources or tools for comparing results and experiments, and a uniform platform supporting the seamless integration and analysis of disease-related data at all levels would greatly facilitate the biomedical research. The ACGT European integrated project aims to build such an infrastructure [19].

In the ACGT computational and data processing environments much of the added value originates from the composition of the different tools. In order for such compositions to be fruitful, certain interoperability constraints must be satisfied and a generic framework must be in place to facilitate the involved tasks. It is therefore the purpose of this paper to describe the integration of the ACGT tools and services under a unified semantic framework. In section 2 we introduce the ACGT architecture and the different aspects of service integration while in the next section we provide details about the syntactic interoperability requirements. In section 4 we describe the ACGT semantic framework that we are proposing and finally section 5 concludes.

2. Service Integration

The ACGT platform aims to facilitate the seamless and secure access and analysis of multi-level clinico-genomic data using high-performing knowledge discovery operations and services. In order to achieve this goal, a well defined data analysis and processing environment needs to be in place, which would make possible the integration and interoperability of the different ACGT components. The goal of the integration process is to make disparate and heterogeneous applications work together so as to produce a unified set of functionality, possibly by complementing each other. Whereas integration is concerned with the building of a unified system that incorporates the functionality of its constituent parts, interoperability is more a virtue of a single software entity so that it can be easily deployed in an unanticipated environment. Therefore defining interoperability guidelines is a prerequisite for building the ACGT integrated environment

In ACGT two notions of interoperability have been specified:

- Syntactic; and
- Semantic interoperability.

Syntactic interoperability of software may be defined as the ability for multiple software components to interact regardless of their implementation programming language or hardware platform. Syntactic interoperability in ACGT requires standardization of data formats and data structures for the representation of, access to and exchange between biomedical informatics resources.

978-0-7695-3872-3/09 $26.00 © 2009 IEEE
DOI 10.1109/ISDA.2009.221

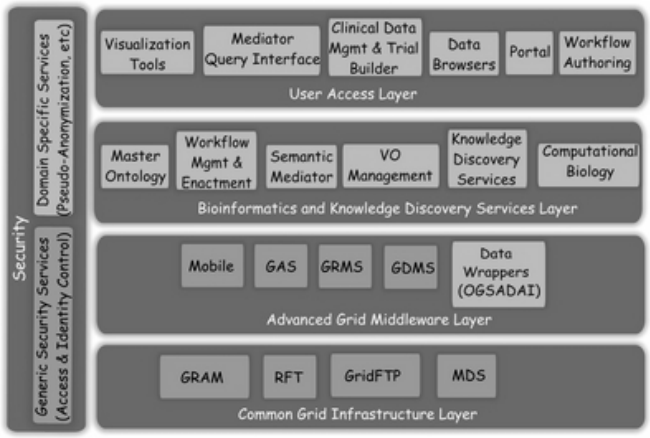

Figure 1. The ACGT Architecture

On the other hand, semantic interoperability is related to the "meaning" of the exchanged information and it is the ability of two or more interacting computer systems to have the meaning of that information accurately and automatically interpreted and "understood". To achieve syntactic interoperability programming and messaging interfaces must conform to standards that specify consistent syntax and format across all systems in the ACGT environment. Furthermore, in order to support the semantic interoperability, all data must be annotated with metadata by means of terminology and ontology identifiers and codes that support aggregation, comparison, summarization, mining, etc. of information that resides in separate resources.

The complexity and the diversity of user requirements have a strong impact on the design of the ACGT architecture. The adopted architecture for ACGT is shown in Figure 1. A layered approach has been followed for providing different levels of abstraction and a classification of functionality into groups of homologous software entities. In this approach we consider the security services and components to be pervasive throughout ACGT so as to provide both for the user management, access rights management and enforcement, and trust bindings that are facilitated by the grid and domain specific security requirements like pseudonymization. Apart from the security requirements, the grid infrastructure and other services are located in the first (lowest) two layers: the Common Grid Layer and the Advanced Grid Middleware Layer. The upper layer is where the user access services, such as the portal and the visualization tools, reside. Finally, the Bioinformatics and Knowledge Discovery Services are the "workhorse" of ACGT and the corresponding layer is where the majority of ACGT specific services lie.

For the realization of this architecture a multidisciplinary and multi paradigm approach has been followed. The ACGT platform is designed according to the following technologies and standards: Service Oriented Architecture (Web Services [7]), the Grid [8], and the Semantic Web [16]. In particular, Grid and Web Services technologies are the basis for defining the syntactic interoperability:

- The machine to machine communication is performed via XML programmatic interfaces over web transport protocols (SOAP), which are specified using the Web Service Definition Language (WSDL). These common data representation and service specification formats, when properly deployed, make the syntactic integration of the ACGT components a lot easier
- The Grid defines the general security framework, the virtual organization (VO) abstraction, the user management mechanisms, authorization definition and enforcement etc. It also provides the computational and data storage infrastructure that is required for the management and processing of large clinical and genomic data sets.

On the other hand, the Semantic Web provides the infrastructure for the semantic interoperability: it adds the knowledge representation mechanisms by the means of RDF Schemas and OWL ontologies, the unique identification of concepts and resources through the URIs, the implementation-neutral query facilities with the SPARQL "universal" query language and the associated query interfaces, etc. These enabling technologies are used for the specification of the service related metadata, such as the semantic description of input and output parameters, the service functionality and intent annotations, the quality of services, etc. These semantic annotations can be used in a multitude of ways: service discovery, selection, and "matchmaking" scenarios, quality control and monitoring, etc.

Interoperability at the syntactic level, although not trivial in some cases, is generally a lot easier than the semantic interoperability. In the next section we delve more into the semantics based description of services to provide this higher level integration.

3. Service Semantics

Semantic interoperability requires the introduction of semantics based annotation and description of the Semantics provide "meaning" for "understanding" the entities and the processes in a domain of discourse. It is nevertheless true that defining the meaning of things as the main task of ontology engineering never ends. There is usually a multitude of views, abstraction layers, uses and goals to provide "meaning" to a certain artifact. Therefore we need to clearly define the role of semantics descriptions of services and to prioritize the different use cases of them in order to provide some

useful and practical solution. For these reasons we have selected the service discovery, selection, and "matchmaking" (composition) as the primary use cases where semantics descriptions for services fit in. All of these are advanced features of a modern problem solving environment such as the Workflow Editor and Enactment environment that the ACGT aims to deliver [15].

In the prototypical Web Service use case scenario a "Service Requester" locates the available services by searching in a "Service Repository" (or Registry) where the services have been advertised by storing there their descriptions.

In order for such scenarios to take place, services should be annotated and described in the most appropriate way so that they are easily discovered and used. The Semantic Service Stack adopts the following general types of service contracts [6]:

- Information Model defines the data model for the input, output and fault messages of the services.
- Functional Descriptions define service functionality and its capabilities, i.e. what a service provide to its callers.
- Non-Functional Descriptions define additional aspects of the service implementation and environment such as "quality of service" (performance, throughput, accuracy, etc) or policies, e.g. security.
- Behavioral Descriptions define the external and internal behavior of the service. The externally visible behavior ("choreography") is for example the "protocol" the client has to follow when contacting the service, e.g. the sequence of operation invocations. On the other hand the internal behavior is related to the way the service is implemented by the composition and orchestration of other services.
- Technical Descriptions define messaging details, such as message serializations, communication protocols, and physical service access points.

For the ACGT purposes the functional and informational descriptions are of particular importance. The Functional descriptions give semantics descriptions about the service capabilities and therefore are important for the discovery of services based on what they can do for the user. Also at the semantic level the Informational descriptions support the discovery, integration, and composition scenarios for web services since they provide information about the input and output messages of the services. On the other hand the Technical information is strictly at the syntactic level, specifying transport and communication specific features or requirements of the services. Finally Behavioral descriptions are an interesting case where especially the externally visible behavior of the service can be used for automatically constructing parts of a workflow or "workflow templates".

Another aspect related to the technologies we have briefly described above is the level of the ontologies employed. We need to distinguish two types of ontologies:

- Foundational (upper-level) ontologies, such as the ones provided by OWL-S [13] and WSMO. These are domain agnostic ontologies that aim to provide the general framework used for service annotation and discovery.
- Domain specific ontologies, such as the BioMOBY's data types [20]. These ontologies provide some classification of domain specific terms and concepts and therefore are orthogonal to the upper-level ontologies. These ontologies can be used to support service discovery and also composition of services based on the annotation of inputs and outputs.

As an example an upper ontology for services can express the proposition that "a service offers some functionality" but a more ground, domain specific ontology, like the BioMOBY Service ontology, is needed in order to specify what the possible "functionalities" are. In ACGT we have selected the OWL-S upper ontology and the BioMOBY data type ontology for annotating the ACGT services and tools.

4. The ACGT Semantic Integration Framework

Today a lot of different standards, service registries, integration APIs, etc. exist. An open world approach seems to be the most pragmatic and future proof approach for supporting the integration of services. For these reasons we are building a Semantic Web infrastructure to ease the composition of the different service providing entities. The conceptual design of this solution is based on an ontology-mediated integration of a number of possibly heterogeneous services registries. The goal is to map (or to describe) the schema of each of these service registries in a common foundational ontology and then query the resulted high level view of the mapped data. The general architectural view of this framework is shown in Figure 2.

The major technology of this design is the Semantic Web related standards and tools. RDF [12] is used as the universal data representation model and SPARQL [14] as the generic query language thereof. OWL [2] and RDF Schema [4] provide the description logic based definition of the underlying data. An ontology based integration is therefore pursued and supported by a "reasoner" component that makes the proper inferences to sustain the semantics based

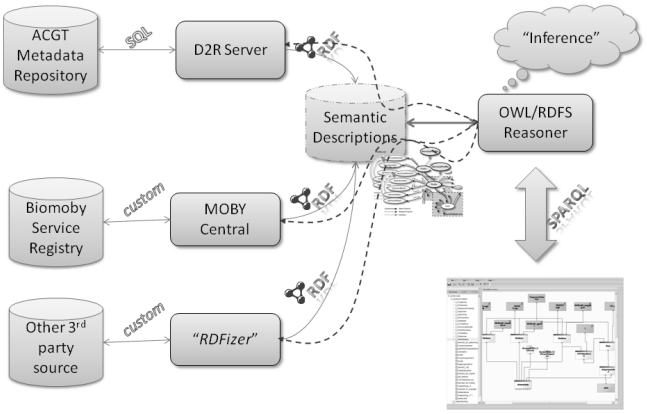

Figure 2. The Semantic Integration Framework

matching and selection of services. This "reasoner" component is a central entity the system responsible for answering the service discovery queries that are submitted in the higher level (foundational) ontology.

In terms of the architectural components this framework basically defines the following components:

- The service registries and repositories. In ACGT this is the Metadata Repository but additional third party registries exist, such as the BioMOBY ones. These are the primary sources of service descriptions and need not be implemented with the same technologies or contacted and searched with the same protocols. We assume though that conceptually they are compliant with the minimal upper level service ontology we have selected, which is the OWL-S Service Profile [13].

- The "RDFizers" (i.e. converters to RDF) are components (either "in-house" developments or "off the shelf") for exporting the service registries information in the RDF format and in the schema defined by the foundational ontology.

- The "Reasoner" (e.g. [18, 17, 9]) is the component that performs the actual tasks of service discovery or matching by employing certain inference rules on the RDF data exported by the RDFizers. These inference rules are of course in accordance with the foundational and domain specific ontologies and define how the data exported by the registries are mapped in the upper ontology.

- The interested user level tools, like the ACGT workflow editor [15], or other services contact the Reasoner in order to make the proper entailments and inferences and answer their queries.

In the implementation of this system a number of key considerations need to be made. First of all the definition of the mapping from the local schemas/ontologies to the upper ontology is of paramount importance for achieving integration. The role of the "RDFizers" is to provide an RDF-compliant interface to the registries so that the subsequent aggregation and alignment of the exported data is feasible. Nevertheless it is the inference rules of the Reasoner that provide the mapping and transformation logic. An example of such a mapping rule could be that the BioMOBY Service concept is a (RDF Schema) subclass of the general OWL-S Service. Using this statement a query requesting OWL-S services can return BioMOBY service instances according to the "entailments" supported by the RDF semantics [10].

Secondly, the type of inference needed has a strong impact on the performance and scalability of the architecture. For example there are three variants of OWL (Lite, DL, and Full) and also RDF Schema, each having different strengths and weaknesses in terms of their reasoning facilities. Furthermore, for the reasoner to make the proper decisions and respond to the submitted queries access to the actual data is required. A straightforward solution is to use a "warehouse" approach where periodically the underlying service registries export their data (through the "RDFizers") and update the central repository that is then subject to queries. This approach compared to the more "lazy"and "real time" access to the registries at query time offers simplicity and performance advantages at the expense of timeliness and the additional operational costs introduced by the "extract, transform, and load" process (especially the "load" part).

Nevertheless it is evident that the aforementioned design provides the following benefits:

- Integration of many different service registries in a unified meta-repository

- A uniform data model for service descriptions that is also extensible to accommodate future service metadata schemas

- A theoretical sound (description logic based [1]) reasoning and querying facility that performs the supplied queries onto the original and the generated, through the possible entailments, service information.

5. Conclusions

The interoperability of the ACGT components is tested by the developers but it's also continually exercised by the users themselves. The ACGT Workflow Editor (Figure 3) is the end user application for the designing and execution

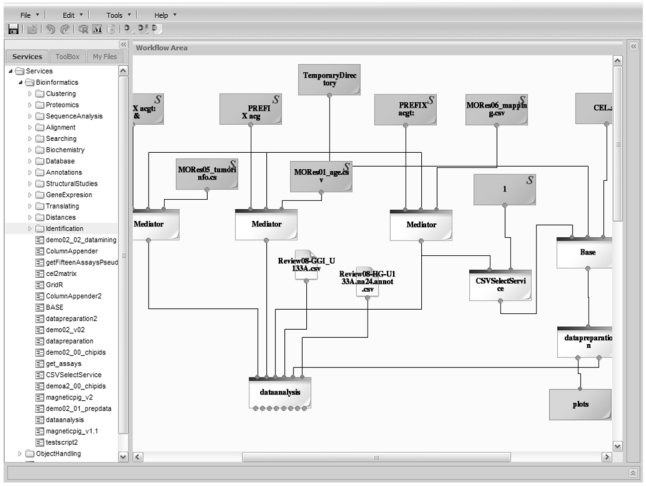

Figure 3. The ACGT workflow editor

of high level scientific workflows. In this web based application the users are facilitated to graphically combine the data retrieval and discovery services and the knowledge extraction and data analysis tools. The definition of the syntactic representation of the data and most importantly the annotation of the services with semantic metadata descriptions gives a lot of flexibility in the workflow editor for supporting user friendliness and intelligence. If properly annotated, incompatible services cannot be directly connected because the data types of their inputs and outputs do not conform to each other, either in the syntactic or the semantic level, while service recommendation and intelligent workflow composition can be also supported.

In conclusion the integrated ACGT environment is built through the adoption of common industry and open standards and well known software engineering practices. The semantic annotation of data and services is of utmost importance and in ACGT the necessary infrastructure (service and data type ontologies, service and metadata registries, etc.) has been designed and implemented. An integration framework has been designed to provide higher level service selection and matching functionality by abstracting the underlying service registries on the ontology level. The Semantic Web methodologies and standards are the primary foundation with respect to the design and the implementation.

A prototype implementation of this framework is currently available for the integration of the ACGT metadata repository with BioMOBY service registries. This implementation is based on open source tools like the D2R Server [3], SwiftOWLIM [11], and Sesame [5].

Acknowledgments The authors wish to thank the ACGT consortium for their contributions and various ideas on which the ACGT project was developed. The ACGT project is funded by the European Commission (FP6/2004/IST-026996).

References

[1] F. Baader, D. Calvanese, D. McGuinness, P. Patel-Schneider, and D. Nardi. *The description logic handbook: theory, implementation, and applications*. Cambridge University Press, March 2003.

[2] S. Bechhofer, F. Van Harmelen, J. Hendler, I. Horrocks, D. McGuinness, P. Patel-Schneider, L. Stein, et al. OWL web ontology language reference. *W3C recommendation*, 10:2006–01, 2004.

[3] C. Bizer and R. Cyganiak. D2R server–publishing relational databases on the semantic web. In *5th International Semantic Web Conference*, 2006.

[4] D. Brickley and R. Guha. RDF vocabulary description language 1.0: RDF schema. *W3C recommendation*, 2004.

[5] J. Broekstra, A. Kampman, and F. Van Harmelen. Sesame: A generic architecture for storing and querying rdf and rdf schema. *Lecture Notes in Computer Science*, pages 54–68, 2002.

[6] J. Cardoso and A. Sheth. Semantic Web processes: semantics enabled annotation, discovery, composition and orchestration of Web scale processes. In *Proceedings of the 4 th Int. Conf. on Web Information Systems Engineering, Rome, Italy*, pages 10–12, 2003.

[7] F. Curbera, M. Duftler, R. Khalaf, W. Nagy, N. Mukhi, S. Weerawarana, I. Center, and Y. Heights. Unraveling the Web services web: an introduction to SOAP, WSDL, and UDDI. *IEEE Internet computing*, 6(2):86–93, 2002.

[8] I. Foster. The grid: computing without bounds. *Scientific American*, 288(4):78–85, 2003.

[9] V. Haarslev and R. Möller. Racer: A core inference engine for the semantic web. In *Proceedings of the 2nd International Workshop on Evaluation of Ontology-based Tools*, pages 27–36, 2003.

[10] P. Hayes and B. McBride. RDF semantics. *W3C recommendation*, February 2004.

[11] A. Kiryakov, D. Ognyanov, and D. Manov. Owlim-a pragmatic semantic repository for owl. *Lecture Notes in Computer Science*, 3807:182, 2005.

[12] G. Klyne, J. Carroll, and B. McBride. Resource description framework (RDF): Concepts and abstract syntax. *W3C recommendation*, 2004.

[13] D. Martin, M. Paolucci, S. McIlraith, M. Burstein, D. McDermott, D. McGuinness, B. Parsia, T. Payne, M. Sabou, M. Solanki, et al. Bringing semantics to web services: The OWL-S approach. *Lecture Notes in Computer Science*, 3387:26–42, 2005.

[14] E. Prud'hommeaux and A. Seaborne. SPARQL query language for RDF, January 2008.

[15] S. Sfakianakis, L. Koumakis, G. Zacharioudakis, and M. Tsiknakis. Web-based authoring and secure enactment of bioinformatics workflows. *Grid and Pervasive Computing Conference, Workshops at the*, 0:88–95, 2009.

[16] N. Shadbolt, W. Hall, and T. Berners-Lee. The semantic web revisited. *IEEE Intelligent Systems*, 21(3):96–101, 2006.

[17] E. Sirin, B. Parsia, B. Grau, A. Kalyanpur, and Y. Katz. Pellet: A practical OWL-DL reasoner. *Web Semantics: science, services and agents on the World Wide Web*, 5(2):51–53, 2007.

[18] D. Tsarkov and I. Horrocks. FaCT++ description logic reasoner: System description. *Lecture Notes in Computer Science*, 4130:292, 2006.

[19] M. Tsiknakis, M. Brochhausen, J. Nabrzyski, J. Pucacki, S. Sfakianakis, G. Potamias, C. Desmedt, and D. Kafetzopoulos. A Semantic Grid Infrastructure Enabling Integrated Access and Analysis of Multilevel Biomedical Data in Support of Postgenomic Clinical Trials on Cancer. *IEEE transactions on information technology in biomedicine*, 12(2):205–217, 2008.

[20] M. Wilkinson and M. Links. BioMOBY: an open source biological web services proposal. *Briefings in bioinformatics*, 3(4):331–341, 2002.