On the Integration of Services with the Web of Data*

Carlos Pedrinaci and **John Domingue**
Knowledge Media Institute,
The Open University, Milton Keynes, UK

Reto Krummenacher
Semantic Technology Institute,
University of Innsbruck, Austria

Abstract

Research on Semantic Web Services has pursued the automation of tasks on the Web by enriching Web services technologies with semantics. Thus far, however, Semantic Web Services have failed to gain a significant uptake due to a big extent to the complexity of the solutions proposed and the limited amount of high quality data and ontologies that were available until recently. In this report we explore the relationship between Semantic Web Services and the Web of Data. We identify the potential benefits that could be obtained by adequately integrating these so far disconnected worlds. We present a vision outlining how this integration could take place by using simpler vocabularies for describing services, through the adoption of linked data principles for publishing services on the Web, and by reusing principles originating research on Knowledge Based Systems and Knowledge Engineering such as the Blackboard model and Problem-Solving Methods. The vision presented herein represents at the same time the outline of a research roadmap we are pursuing and we shall, where appropriate, illustrate some of these ideas through concrete examples and prototypes we have already developed.

Introduction

Over the past decade there has been considerable research activity in the area of Semantic Web Services (SWS). A number of ontology-based frameworks for describing services – including a W3C standard – have been created (Martin et al. 2004; Fensel et al. 2007; Farrell and Lausen 2007). Up until now the impact of SWS on the Semantic Web has been minimal. In the Web context semantics are used to mark up a wide variety of data-centric resources but are not used to annotate online functionality in any form in significant numbers. The reasons for this are two-fold. Firstly, SWS research has for the most part targeted WSDL/SOAP-based Web services (Erl 2007) which are not prevalent on the Web (Davies et al. 2009). Secondly, due to the inherent complexity required to fully capture computational functionality, creating SWS descriptions has represented an important knowledge acquisition bottleneck and has required the use of rich semantic languages and complex reasoners.

In parallel, much research on the Semantic Web has lately been devoted to creating what is referred to as the Web of Data, "a Web of things in the world, described by data on the Web" (Bizer, Heath, and Berners-Lee 2009). The Web of Data is based upon a set of linked data principles and provides publicly large amounts of interconnected data across a wide range of topics described in terms of lightweight ontologies (Bizer, Heath, and Berners-Lee 2009). Despite the outstanding evolution so far, most linked data applications solely gather data from different sources and display it alongside each other (Bizer, Heath, and Berners-Lee 2009).

In this paper we explore the relationship between services and the Web of Data. We identify the potential benefits that could be obtained by adequately integrating these so far rather disconnected worlds. We outline how this integration could take place by using simpler vocabularies for describing services and through the adoption of linked data principles for publishing services on the Web. Finally, we propose a minimal extension to existing linked data infrastructure in order to enable the development of intelligent systems over the Web of Data borrowing a widely successful conceptual architecture originating from AI research, namely the blackboard architecture.

The remainder of this report is organised as follows. First, we present the relevant technological around services and the Web of Data. We then present how the use of lightweight semantics can allow us to bring services into the Web of Data thus enabling their discovery through state of the art linked data technologies. Next, we focus on how services can contribute to the Web of Data both generating new data and processing existing one. Final, we present some architectural considerations and outline how, through two minimal extensions, the Web of Data could enable the creation of complex knowledge-based systems for the Web.

Background

Web services and the Service-Oriented Architecture are commonly lauded as a silver bullet for Enterprise Application Integration, implementation of inter-organizational business processes, and as a general solution for the development of all complex distributed applications over the Web (Erl 2007). Web services are software systems offered

*This work is supported by the EU project SOA4All (FP7-215219)

over the Internet via platform and programming-language independent interfaces defined on the basis of a set of open standards such as WSDL, SOAP and further WS-* specifications (Erl 2007). Constructing distributed systems out of Web services becomes a matter of identifying suitable Web services and orchestrating them in a way such that they achieve the goal pursued. Although highly appealing from an engineering perspective this way of developing systems requires a good deal of manual effort for locating, interpreting and integrating existing services and has therefore mostly been relegated to controlled environments like enterprises rather than the Web (Davies et al. 2009).

Recently, the world around services on the Web, thus far limited to "classical" Web services based on SOAP and WSDL, has significantly evolved with the proliferation of Web APIs, also called RESTful services (Richardson and Ruby 2007) when they conform to the REST architectural style (Fielding 2000). This newer kind of services is characterized by their relative simplicity and their natural suitability for the Web, which is indeed closely related to the growing popularity and use of Web 2.0 technologies. Unfortunately, most often these services do not have machine-readable interface definitions like WSDL, and are instead described in HTML pages that need to be interpreted by humans. As a consequence, the difficulties that previously affected the discovery and interpretation of Web services are even greater in the case of RESTful services.

Research on Semantic Web Services (SWS) has been devoted to reducing the manual effort required for using Web services and more recently also for RESTful services. The main idea behind this research is to combine services and Semantic Web technologies so that tasks such as the discovery, negotiation, composition and invocation of services can have a higher level of automation. Over the years significant results have been obtained in terms of ontologies and languages for describing Web services semantically (Martin et al. 2004; Fensel et al. 2007) as well as engines and platforms able to support the automation of some of the previously mentioned tasks covering the life-cycle of services and service-based applications (Domingue et al. 2008; Sheth, Gomadam, and Ranabahu 2008; Fensel, Kerrigan, and Zaremba 2008). Their uptake has however been limited due to their complexity both computationally and in terms of knowledge acquisition mostly due to the lack of semantic data publicly available up until the advent of the Web of Data.

The Web of Data is based upon a set of linked data principles and the use of URIs, HTTP and RDF (Bizer, Heath, and Berners-Lee 2009). Since the linked data principles were outlined in 2006, there has been a large uptake most notably by the Linking Open Data project through DBpedia (Auer et al. 2008) and ulterior additions of data about reviews (Heath and Motta 2008), scientific information, geographical information and governmental data, to name a few. Currently there is a growing body of data expressed in terms of a variety of vocabularies such as FOAF and Dublin Core. Consequently substantial efforts are underway trying to exploit this growing Web of Data. Among the main applications produced so far there are a number of data browsers that help people navigate through the linked data like Disco and Tabulator (Berners-Lee et al. 2007). There are search engines that crawl and index the Web of data like Sindice (Oren et al. 2008) and Watson (d'Aquin et al. 2008). And finally, there are a few domain-specific applications such as Revyu.com (Heath and Motta 2008) or DBPedia Mobile (Becker and Bizer 2008) that provide domain-specific functionality by gathering and mashing up data. Although useful these applications hardly go beyond presenting together data gathered from different sources leaving the great potential of this massive data space unexploited. It is therefore paramount to identify means by which systems that carry out advanced processing over this data could systematically be developed.

The current technological landscape is therefore characterised by a number of highly complementary technologies that have so far remained disconnected. In the remainder of this paper we shall describe how we envision that these technologies could be integrated, how combined they could mitigate the drawbacks of each other and enable the creation of advanced applications on top of the Web of Data.

Services on the Web of Data

Arguably, the main limitation of previous approaches to integrating services in the Semantic Web, is their complexity from a computational, architectural and knowledge-acquisition perspective. Semantic Web Services have suffered from a number of complications that were not foreseen and have significantly hampered their evolution and adoption. In particular, the demanding requirements that Semantic Web Services approaches like OWL-S and WSMO have with respect to reasoning, storage as well as quantity and quality of semantic descriptions contrasted with the state of the art technologies when research work started.

For instance, a fundamental tenet of Service-Oriented Architectures is the notion of service repositories for programmatic access and discovery of suitable services. Enhancing repositories with semantics has been one of the key issues of SWS research (Sycara, Paolucci, and Srinivasan 2003). And yet the largest public SWS repository is probably OPOS-Sum, a test collection with less than 3000 service annotations which provides programmatic access to its content solely through direct access to the database management system (Küster and König-Ries 2008).

Before any significant uptake of SWS technology can happen, proper mechanisms for publishing SWS must be in place. In this respect, the evolution of the Web of Data shows that i) lightweight ontologies together with the possibility to provide custom extensions prevail against more complex models; ii) linked data principles are an appropriate means for publishing large amounts of semantic data, both for human and machine consumption; iii) links between publicly available datasets are essential for the scalability and the value of the data exposed. In the remainder of this section we shall cover how where we believe the technologies for publishing SWS should be heading and we shall illustrate it with the models and technologies we are developing.

Lightweight Modeling of Services on the Web

In order to publish services on the Web of Data it is necessary to provide a common vocabulary based on existing Web standards able to describe services in a way that allows machines to automatically locate and filter services according to their functionality or the data they handle, and to appropriately support their automatic invocation. Additionally, as opposed to most SWS research to date, it is of utmost importance to support the annotation of both "classical" Web services described in WSDL, as well as the increasing number of Web APIs and RESTful services which appear to be preferred on the Web (Davies et al. 2009). To this end, we build upon WSMO-Lite (Vitvar et al. 2008), a minimal extension to SAWSDL, MicroWSMO a microformat-like language for annotating Web APIs, and the Minimal Service Model a simple RDF(S) model able to capture the semantics for both Web services and Web APIs in a common model that can easily be published in the Web of Data.

WSMO-Lite W3C produced in 2007 the Semantic Annotations for WSDL and XML Schema specification, a minimal bottom-up approach to annotating services semantically which has gained further uptake than more ambitious solutions like OWL-S and WSMO. SAWSDL provides simple hooks for pointing to semantic descriptions from WSDL and XML elements. In particular, it supports three kinds of annotations, namely modelReference, liftingSchemaMapping and loweringSchemaMapping which allow pointing to semantic elements described elsewhere on the Web, or to specifications of data transformations from a syntactic representation to the semantic counterpart and back respectively. SAWSDL does not advocate for a particular representation language for these documents nor does it provide any specific vocabulary that users should adopt. This characteristic is a means to support extensibility but also forces users to choose their own ontologies for describing services semantically.

WSMO-Lite continues this incremental construction of a stack of technologies for Semantic Web Services by addressing precisely this lack (Vitvar et al. 2008). WSMO-Lite takes the main types of semantic annotations for services, provides a simple vocabulary for them and defines annotation mechanisms based on this simple ontology and SAWSDL in order to link WSDL services to semantic models. The WSMO-Lite ontology includes means for specifying service taxonomies through the notion of FunctionalClassificationRoot and it provides hooks for more advanced definition of non-functional properties as well as conditions and effects. The ontology is entirely expressed in RDF(S) and where the expressivity of RDFS is not sufficient (notably for expressing conditions and effects) other languages such as WSML, SWRL or those produced by the RIF Working Group can be used. The reader is referred to (Fischer, Kopecký, and Vitvar 2009) for the latest details.

MicroWSMO As we previously introduced, Web APIs and RESTful services are increasingly used on the Web. Therefore any approach to using services on the Web that would disregard them would be unnecessarily limiting. Annotating this kind of service does, however, bring additional complexities given that in most of the cases services are solely described through Web pages aimed at humans. Microformats offer means for annotating human-oriented Web pages in order to make key information machine-readable and is therefore the solution we adopt in this case (Maleshkova, Kopecký, and Pedrinaci 2009; Maleshkova, Pedrinaci, and Domingue 2009). In particular, hRESTS enables the creation of machine-processable Web API descriptions based on available HTML documentation (Kopecky, Gomadam, and Vitvar 2008). hRESTS is complemented by MicroWSMO, which supports including semantic annotations in a SAWSDL-like manner (Kopecky, Vitvar, and Gomadam 2008). MicroWSMO introduces additional HTML classes to enable the specification model references as well as lifting and lowering schema mappings.

The Minimal Service Model The minimal service model, described in other previous publications as being part of WSMO-Lite, provides a common model for capturing services in RDF(S) based on the principle of minimal ontological commitment. The minimal service model builds upon a number of modules, namely the SAWSDL RDF mapping (Kopecký 2007), WSMO-Lite as a minimal extension to SAWSDL, and hRESTS in order to support also Web APIs. The minimal service model defines services as having a number of operations each of which have input and output messages and faults. Additionally, the model encompasses preconditions and effects, non-functional parameters, lifting and lowering mechanisms, and for the purposes of supporting Web APIs, the address as a URI template and the HTTP method to be used.

Thanks to its simplicity, the minimal service model captures the essence of services in a way that can support service matchmaking and invocation and still remains largely compatible with the RDF mapping of WSDL, with WSMO-based descriptions of Web services , with OWL-S services and with services annotated according to WSMO-Lite and MicroWSMO, therefore providing a suitable model for the publication of a variety of service annotations in the Web. Although providing a formal mapping for each of these languages is out of the scope of this paper, we note that the elements captured in the minimal service model are common to existing models (with the exception of the hRESTS extensions). Indeed, the mapping is not loss-less but an appropriate use of rdfs:isDefinedBy as explained next, can help circumvent this limitation and still provide a common ground for publishing Linked Services with the Web of Data in a way that is amenable to automated processing and where more expressive definitions can also be used if needed.

Publishing Services as Linked Data

Alongside the WSMO-Lite and MicroWSMO specifications transformations have been defined so that WSDL files containing SAWSDL annotations as well as Web pages including MicroWSMO annotations can be automatically processed and directly transformed into RDF expressed in terms of the Minimal Service Model (see (Maleshkova, Pedrinaci, and Domingue 2009) for details on the MicroWSMO transformation). In this way, it is possible to store, index, and process services at the semantic level by using state of the art technologies from the Semantic Web. Services can in this way be offered on the Web of Data through SPARQL endpoints so that agents (human or software) can discover suitable services and eventually use them. Additionally, and thanks to using a common vocabulary for expressing the annotations of WSDL and RESTful services, we provide a common umbrella under which both kinds of services can be discovered, manipulated, and utilised in an homogeneous way, thus merging the best of both worlds.

Alongside the Minimal Service Model, WSMO-Lite and MicroWSMO we are currently developing iServe[1], a platform for publishing SWS as linked data, no matter their original format. Currently iServe supports the publication of WSMO-Lite, MicroWSMO by transformation into RDF and transformations for other models (e.g., OWL-S and WSMO) are being developed. iServe adopts linked data principles to enable humans and machines to discover and use services via a Web API and a SPARQL endpoint.

Since knowledge-acquisition has been a significant bottleneck for SWS technologies, we are devoting much effort to creating tools that support users in the annotation of services by leveraging the Web of Data as a source of background knowledge. One such application is the Web API annotation tool SWEET (Maleshkova, Pedrinaci, and Domingue 2009) which assists users in annotating HTML descriptions of Web APIs[2]. The tool, through Watson (d'Aquin et al. 2008), support users in browsing the Semantic Web while annotating services so that they can identify suitable vocabularies such as eCl@ass, Good Relations and FOAF, and the use them for the annotation. The acquisition of service annotations is in this way simplified while at the same time links between service annotations and the Web of Data are established, paving the way for linked data application developers to locate interesting services in a simple manner (e.g., based on the input and output types of services).

Services for the Web of Data

The notion of services as well-defined, independent, invokable and distributed pieces of functionality is indeed a very powerful architectural notion for developing distributed systems. Providing functionality in this way independently from the underlying technology provides the capacity for maintaining a loose coupling between integrated components which, when it comes to an environment like the Web, appears as a highly beneficial (if not necessary) feature. Therefore, it appears that services, may they be traditional Web services or RESTful services, provide a suitable architectural abstraction for the integration of processing capabilities over the Web of Data in a loosely coupled manner. In the remainder of this section we shall cover what services can provide to the Web of Data both as a means to providing further data as well as for processing existing assertions.

[1] See iserve.kmi.open.ac.uk

[2] See sweet.kmi.open.ac.uk.

Integrating Legacy Systems

Currently a good part of the Web of Data is generated from existing databases by using tools such as D2R (Bizer, Heath, and Berners-Lee 2009). Indeed, this allows exposing huge amounts of data which would otherwise be private or in the best case offered through channels which are not convenient for automated processing. In other cases data is already stored in RDF and can be exposed easily. There is however a large body of information behind RESTful services, or offered by sensors that still remains within controlled silos and expressed in formats that do not follow linked data principles. By creating annotations of existing Web APIs and WSDL services we enable a new means for unleashing valuable data from their silos on demand. In this way, data from legacy systems, state of the art Web 2.0 sites, or sensors, which do not embrace linked data principles could be made available as linked data. Indeed proper care should be taken in order to ensure that linked data principles are followed in these cases. We anticipate, however, that at least for services strictly adhering to REST principles this should be relatively straight-forward since they should already offer means for exploring their resources.

Processing Linked Data

Integration and fusion of disparate data coming from the Web of Data hardly takes place nowadays and therefore applications do not perform any ulterior processing of this data other than for presenting it to the user. Generating new data based on what has been found or the provisioning of services that exploit this data thus remains a pending issue. For instance, something as simple and useful as a unit transformation service is still to be provided for the Web of Data other than through proprietary extensions. To a certain extent this is natural since the Web of Data is precisely about this, data; and storing an RDF triple per possible transformation result would simply be absurd since there are infinite possibilities. There is, however, a clear need for enabling the processing of linked data in ways such that application developers could conveniently apply them over data gathered at runtime to carry out computations as simple as unit transformations, more complex as deriving similarities between things based on the reviews published by different users on Revyu.com, or even more advanced as envisioned for the Semantic Web (Berners-Lee, Hendler, and Lassila 2001).

The Web of Data provides large amounts of machine-processable data ready to be exploited and, as we saw, services provide a suitable abstraction for encapsulating functionality as platform and language independent reusable software. It therefore seems natural to approach the development of systems that process linked data by combining services. These services should be able to consume RDF data (either directly or via lowering mechanisms), carry out the concrete activity they are responsible for (e.g., unit conversion), and return the result if necessary in RDF as well. The invoking system could then store the result obtained or continue with the activity it is carrying out using these newly obtained RDF triples. In a sense this is quite similar to RDF mashups (Phuoc et al. 2009) with the important difference that services may range from RDF-specific manipulation up to highly complex processing beyond data fusion. The use of services as the core abstraction for constructing linked data applications is therefore more generally applicable than that of current data integration oriented mashup solutions.

It is worth noting in this respect the benefit brought by having services annotations available on the cloud as we saw earlier. When developing applications that process linked data, discovering presumably useful services would be a matter of sending SPARQL queries to known service repositories or directly querying indexing systems like Sindice (Oren et al. 2008). And as opposed to traditional Web services repositories like UDDI-based ones, developers would benefit from the existence of semantic annotations in order to filter them based on the inputs, outputs, their classification with respect to well-known taxonomies such as eCl@ss (Hepp 2006), etc. In this way, linked data application developers would have access to an ever growing body of reusable components ready to be combined and exploited.

The Services Ecosystem

Integrating services with the Web of Data would in this way give birth to a services ecosystem on top of linked data, whereby people would be able to collaboratively construct more and more complex systems by reusing the results of others, gradually taking us closer to the ambitious vision initially presented for the Semantic Web. In this process, we anticipate that two main families of services will emerge depending on whether they are domain-independent or not.

On the one hand, task-specif yet domain-independent services will allow developers to perform some of the typical tasks involved when processing linked data ranging from relatively basic activities such as transforming data between different schemas to more complex actions such as determining how trust-worthy a piece of data is or even, eventually, to carry out knowledge intensive tasks such as Parametric Design or Diagnosis (Schreiber et al. 1999). These domain-independent services which are already starting to appear[3] can in fact be seen from a Knowledge Engineering perspective as a new generation of Problem-Solving Methods adapted to the Web as some researchers already start considering (van Harmelen, ten Teije, and Wache 2009).

The approach for developing applications on top of linked data envisioned herein will however exhibit a greater sensitivity with respect to data quality and correctness than it does for current applications mostly focussed on presenting data from different sources to the user. In fact, since data will be directly processed by machines, it will necessarily have to be validated, fused and cleaned prior to any execution since this would otherwise yield execution errors or incorrect results. Hence, a good deal of domain-independent services will precisely be devoted to performing these tasks. This new generation of Problem-Solving Methods will necessarily have to include data pre-processing activities among the first steps. As a side effect, though, it is likely that data quality in the Web of Data will increase as software matures, and especially as it starts been processed by applications which would indirectly detect inconsistencies and incorrect data.

On the other hand, we refer as domain-dependent services to those abstracted away from the technicalities and specificities of linked data and generic tasks. This kind of services will be for example those directly providing access to traditional systems in order to obtain some data and carry out actions like sending an SMS or booking a hotel. These services will only be relevant for a particular domain, e.g., hotel services, and will mostly be populated by services directly addressing end-users and therefore better showcasing the potential of the Semantic Web. It is worth noting, however, that a wide proliferation of advanced domain-specific solutions for end-users will only occur when a sufficient set of stable domain-independent services able to solve complex tasks will be available. For instance, a system able to book rooms from a wide-range of hotels will most probably require good support for transforming data between different schemas.

Architectural Considerations

The Web of Data builds upon four simple principles which are to a big extent the reason for its fast evolution. In a nutshell the principles dictate that things should be given URIs; that these URIs should be HTTP ones so that they can be looked up; that information on these URIs should be offered using standards such as RDF and SPARQL; and that data should link to other URIs. By means of these very simple principles the Web of Data has reached more than 4 billion triples. And yet, the Web of Data is essentially static.

Changes on the Web of Data happen uniquely because of users interactions. And these changes are in many cases deferred until some batch processing takes place. Researchers have already identified this issue and have been devoting efforts towards what they refer to as the writable Semantic Web (Berners-Lee et al. 2007). So far the approaches suggested typically aim at extending SPARQL with support for updating. Indeed, doing so opens up a wide range of possibilities which are of particular relevance when it comes to integrating services with the Web of Data, which will often require updating the assertions hosted in the Web servers.

However, supporting updates is not all there needs to be to enable the development of complex systems such as those initially envisioned for the Semantic Web. Supporting SPARQL updates is certainly a step forward but the impact of an update would unfortunately be limited to the silo where it took place. Despite the fact that data is linked, when it comes to data updates, the Web of Data remains essentially disconnected which limits to a big extent its dynamicity as well as the kinds of applications that can be developed. Developing systems that can process large amounts of widely distributed sensor data in order to predict geographic disasters, systems that monitor market data for avoiding economic crashes, or simply systems that provide automated notifications when relevant news are published cannot be developed or in the best case require unnecessarily expensive pulling-based solutions.

Taking the Web of Data to the next level where software can truly benefit from the wealth of machine processable

[3] See http://alignapi.gforge.inria.fr/

data available and where the Web of Data itself can benefit from advanced services processing this data, requires providing suitable mechanisms allowing Web servers to notify other systems about changes occurred in their assertions. Our previous experience shows that the implementation of Publish/Subscribe mechanisms by allowing systems to register templates of triples for which they wish to be notified is a simple and suitable approach (Krummenacher et al. 2009). Web-scale experiments should, however, be carried out to assess the scalability of these approaches, although research on Complex Event Processing has already shown how highly efficient systems can be developed (Luckham 2001).

The Web of Data as a Blackboard

The Blackboard Model was proposed, developed, and applied in several applications as a way to surmount the inconveniences of the classic Knowledge Based Systems approach (Engelmore and Morgan 1988). The Blackboard Model is often presented using the analogy of a group of persons in front of a blackboard trying to put together a jigsaw puzzle. Each of them looks at his or her pieces and sees if any of them fit, in which case they update the solutions. New updates cause other pieces to fall into place and the whole puzzle can be solved in complete silence. The solution is built incrementally and opportunistically as opposed to, say, starting systematically from the top left corner and trying each piece.

The fundamental philosophy of this problem-solving model establishes that the agents collaborating do not communicate with each other directly, instead all the interactions strictly happen through modifications on a shared workspace, i.e., the blackboard. Experts of particular aspects of the problem contribute to the overall problem-solving activity in an incremental and opportunistic way. The Blackboard Model as described by the metaphor, is a conceptual definition of a reasoning behaviour and does not prescribe any particular implementation detail. It is therefore important not to take the Blackboard Model as a computational specification, but rather as a conceptual guideline about how to perform problem-solving reasoning (Engelmore and Morgan 1988).

The Blackboard Model has been applied to develop a wide-range of applications including voice recognition, military situations monitoring and assessment, signal processing, drug design support, military planning, process scheduling systems, etc (Engelmore and Morgan 1988). The Blackboard Model is a general and versatile reasoning model, particularly well suited for supporting reasoning processes over the Web (Pedrinaci, Smithers, and Bernaras 2007). It provides an outstanding support for reasoning in highly dynamic environments. And it supports adapting the reasoning process to the very typical and diverse events of the Web, such as remote execution exceptions, a continuous data flow and connectivity problems.

Extending the Web of Data with these two minimal extensions, i.e., support for updates and a publish/subscribe mechanism, would pave the way for treating the Web of Data as a blackboard. Doing so would, without introducing new architectural elements, open up a wide-range of possibilities for the construction of advanced systems exploiting the vast amount of linked data for performing complex tasks as initially envisioned for the Semantic Web.

Conclusions and Outlook

Since the linked data principles were first outlined, the amount of RDF data available on the Web has increased exponentially. The state of the art of the applications processing linked data is however not that outstanding. Most applications limit themselves to gathering data and presenting it to the user. Still, the main reason for creating a Web of Data is to enable the creation of applications that can exploit this vast amount of interlinked machine-processable data.

We have explored the relationship between services and the Web of Data. In particular we have highlighted how Web services and RESTful services can be brought into the Web of Data by means of a simple RDF vocabulary and supporting tools. We have outlined how the presence of services in the Web of Data could better support developers in creating applications that process linked data. We have discussed how the evolution towards more complex linked data applications could be supported and we have identified the need for making publicly available domain-independent services that carry out common tasks such data cleaning or mapping.

Finally, we have revisited architectural concerns trying to identify the extensions that would be required for supporting the development of complex knowledge-based linked data applications. We propose two minimal extensions, i.e., updates support and publish/subscribe mechanisms, which *a priori* do not require any additional architectural elements.

The overall vision outlined herein basically represents the roadmap for the research we are currently carrying out trying to expand the capabilities of the linked data applications as well as trying to simplify the use of services on the Web through lightweight Semantic Web Services technologies.

References

Auer, S.; Bizer, C.; Kobilarov, G.; Lehmann, J.; Cyganiak, R.; and Ives, Z. 2008. Dbpedia: A nucleus for a web of open data. In *Proceedings of 6th International Semantic Web Conference, 2nd Asian Semantic Web Conference (ISWC+ASWC 2007).* 722–735.

Becker, C., and Bizer, C. 2008. Dbpedia mobile: A location-enabled linked data browser. In *Linked Data on the Web (LDOW2008).*

Berners-Lee, T.; Hollenbach, J.; Lu, K.; Presbrey, J.; d'ommeaux, E. P.; and m.c. schraefel. 2007. Tabulator redux: Writing into the semantic web.

Berners-Lee, T.; Hendler, J.; and Lassila, O. 2001. The Semantic Web. *Scientific American* (5):34–43.

Bizer, C.; Heath, T.; and Berners-Lee, T. 2009. Linked data - the story so far. *International Journal on Semantic Web and Information Systems (IJSWIS).*

d'Aquin, M.; Motta, E.; Sabou, M.; Angeletou, S.; Gridinoc, L.; Lopez, V.; and Guidi, D. 2008. Toward a new generation of semantic web applications. *IEEE Intelligent Systems* 23(3):20–28.

Davies, J.; Domingue, J.; Pedrinaci, C.; Fensel, D.; Gonzalez-Cabero, R.; Potter, M.; Richardson, M.; and Stincic, S. 2009. Towards the open service web. *BT Technology Journal* 26(2).

Domingue, J.; Cabral, L.; Galizia, S.; Tanasescu, V.; Gugliotta, A.; Norton, B.; and Pedrinaci, C. 2008. IRS-III: A broker-based approach to semantic Web services. *Web Semantics: Science, Services and Agents on the World Wide Web* 6(2):109–132.

Engelmore, R. S., and Morgan, A. J. 1988. *Blackboard Systems*. The Insight Series in Aritificial Intelligence. Addison-Wesley. ISBN: 0-201-17431-6.

Erl, T. 2007. *SOA Principles of Service Design*. The Prentice Hall Service-Oriented Computing Series. Prentice Hall.

Farrell, J., and Lausen, H. 2007. Semantic Annotations for WSDL and XML Schema. http://www.w3.org/TR/sawsdl/. W3C Candidate Recommendation 26 January 2007.

Fensel, D.; Lausen, H.; Polleres, A.; de Bruijn, J.; Stollberg, M.; Roman, D.; and Domingue, J. 2007. *Enabling Semantic Web Services: The Web Service Modeling Ontology*. Springer.

Fensel, D.; Kerrigan, M.; and Zaremba, M., eds. 2008. *Implementing Semantic Web Services: The SESA Framework*. Springer.

Fielding, R. T. 2000. *Architectural Styles and the Design of Network-based Software Architectures*. Ph.D. Dissertation, University of California, Irvine.

Fischer, F.; Kopecký, J.; and Vitvar, T. 2009. Wsmo-lite. http://cms-wg.sti2.org/TR/d11/v0.3/20090616/. Last visited: October 2009.

Heath, T., and Motta, E. 2008. Revyu: Linking reviews and ratings into the web of data. *Web Semant.* 6(4):266–273.

Hepp, M. 2006. Products and services ontologies: A methodology for deriving owl ontologies from industrial categorization standards. *International Journal on Semantic Web and Information Systems (IJSWIS)* 2(1):72–99.

Kopecky, J.; Gomadam, K.; and Vitvar, T. 2008. hrests: an html microformat for describing restful web services. In *The 2008 IEEE/WIC/ACM International Conference on Web Intelligence (WI2008)*. Sydney, Australia: IEEE CS Press.

Kopecky, J.; Vitvar, T.; and Gomadam, K. 2008. MicroWSMO. Deliverable, Conceptual Models for Services Working Group.

Kopecký, J. 2007. Web services description language (wsdl) version 2.0: Rdf mapping. Working group note, W3C.

Krummenacher, R.; Simperl, E.; Cerizza, D.; Valle, E. D.; Nixon, L. J. B.; and Foxvog, D. 2009. Enabling the european patient summary through triplespaces. *Computer Methods and Programs in Biomedicine* 95(2):33–43.

Küster, U., and König-Ries, B. 2008. Towards standard test collections for the empirical evaluation of semantic web service approaches. *Int. J. Semantic Computing* 2(3):381–402.

Luckham, D. C. 2001. *The Power of Events: An Introduction to Complex Event Processing in Distributed Enterprise Systems*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc.

Maleshkova, M.; Kopecký, J.; and Pedrinaci, C. 2009. Adapting sawsdl for semantic annotations of restful services. In *Workshop: Beyond SAWSDL at OnTheMove Federated Conferences & Workshops*.

Maleshkova, M.; Pedrinaci, C.; and Domingue, J. 2009. Supporting the creation of semantic restful service descriptions. In *Workshop: Service Matchmaking and Resource Retrieval in the Semantic Web (SMR2) at 8th International Semantic Web Conference*.

Martin, D.; Burstein, M.; J., H.; Lassila, O.; McDermott, D.; McIlraith, S.; Paolucci, M.; Parsia, B.; Payne, T.; Sirin, E.; Srinivasan, N.; and Sycara, K. 2004. OWL-S: Semantic Markup for Web Services. http://www.daml.org/services/owl-s/1.0/owl-s.pdf.

Oren, E.; Delbru, R.; Catasta, M.; Cyganiak, R.; Stenzhorn, H.; and Tummarello, G. 2008. Sindice.com: a document-oriented lookup index for open linked data. *IJMSO* 3(1):37–52.

Pedrinaci, C.; Smithers, T.; and Bernaras, A. 2007. Opportunistic reasoning for the semantic web: Adapting reasoning to the environment. *Proceedings of the First International Workshop Workshop New Forms of Reasoning for the Semantic Web: Scalable, Tolerant and Dynamic*.

Phuoc, D. L.; Polleres, A.; Hauswirth, M.; Tummarello, G.; and Morbidoni, C. 2009. Rapid prototyping of semantic mash-ups through semantic web pipes. In Quemada, J.; León, G.; Maarek, Y. S.; and Nejdl, W., eds., *WWW*, 581–590. ACM.

Richardson, L., and Ruby, S. 2007. *RESTful Web Services*. O'Reilly Media, Inc.

Schreiber, G.; Akkermans, H.; Anjewierden, A.; de Hoog, R.; Shadbolt, N.; de Velde, W. V.; and Wielinga, B. 1999. *Knowledge Engineering and Management: The CommonKADS Methodology*. MIT Press.

Sheth, A. P.; Gomadam, K.; and Ranabahu, A. 2008. Semantics enhanced services: Meteor-s, sawsdl and sa-rest. *IEEE Data Eng. Bull.* 31(3):8–12.

Sycara, K.; Paolucci, M.; and Srinivasan, A. A. N. 2003. Automated discovery, interaction and composition of Semantic Web services. *Web Semantics* 1(1):27–46.

van Harmelen, F.; ten Teije, A.; and Wache, H. 2009. Knowledge engineering rediscovered: towards reasoning patterns for the semantic web. In *K-CAP '09: Proceedings of the fifth international conference on Knowledge capture*, 81–88. New York, NY, USA: ACM.

Vitvar, T.; Kopecky, J.; Viskova, J.; and Fensel, D. 2008. Wsmo-lite annotations for web services. In Hauswirth, M.; Koubarakis, M.; and Bechhofer, S., eds., *Proceedings of the 5th European Semantic Web Conference*, LNCS. Berlin, Heidelberg: Springer Verlag.