Linked Open Data: Are we Drowning in Information and Starving for Know-How?

Matthias Thimm, Thomas Gottron, Gerd Gröner, and Ansgar Scherp

Institute for Web Science and Technologies
University of Koblenz-Landau, Germany
{thimm, gottron, groener, scherp}@uni-koblenz.de

Abstract. Today's most popular means for publishing semantic information on the web is the paradigm of Linked Open Data (LOD) and the technologies behind the Resource Description Framework (RDF). The amount of openly available RDF data is drastically increasing. However, even when considering ontological information such as RDF Schema, LOD is only little more than a collection of pieces of information. In this paper, we critically review the current situation of LOD and have a bold look into the possible future of *Linked Open Know-How*. While today's LOD is mostly on representing factual information (which can also be called *know-that*) we also consider the possibility of representing procedural information (*know-how*) in the future version of LOD. In our vision, by considering procedural information in the same way as factual information, automatic knowledge acquisition and learning how and to what purpose to use this knowledge can be tightly integrated.

1 Introduction

In the last years, the semantic web has gained momentum thanks to the Linked Open Data (LOD) movement. As for the web itself, the distributed and semantically light approach of LOD has spawned a wider adoption of semantic technologies and the publishing and interlinking of semantic data. This data is freely available and due to its semantic nature can easily be integrated with private data in order to extend the factual knowledge of a semantic web client. While this works well for factual knowledge, publishing and exchange of procedural information is—in general—not possible to the same extent. Current research aims at extending the LOD principle to semantic web services [2, 10]. For example, Linked Services [4, 8] can be embedded seamlessly into linked data. By representing interface descriptions via the LOD principles services can be discovered and executed by client applications. This might allow for the integration of LOD and open services within the near future. There is, however, one drawback in this approach: Using web services requires the client to send its data to the web service for processing. By doing so, the client has to disclose potentially private data.

There are various scenarios, in which semantic data cannot be incorporated into the public web or processed by public web services. One example is health record data, which is sensitive and typically highly protected by legal obligations. Therefore, it would be impossible to provide a public service that processes this kind of data in order to, e. g., diagnose cancer from a blood exam. Another example are confidential financial data. A company might be reluctant to reveal all of its assets to the public, but might be interested or even legally obliged to publish aggregates and statistics over their financial situation. Also here, sending detailed data to a public service for computing these values is not an option. Also, it might be infeasible to send data over the web if its size is disproportional huge compared to the complexity of the process to be conducted.

If instead, it would be possible to integrate the *know-how* of how to do a cancer diagnosis or a statistical analysis into the semantic web client, there is no need to disclose sensitive data. In a certain way, open source software already provides procedural knowledge in the form of software libraries and applications. It is possible to integrate these libraries with own applications, analyse and adapt the contained procedure to the own needs. The increased perception of semantic technologies will affect the open source movement and the techniques will be adopted in the world of describing software packages and libraries. At the same time, the semantic web clients will learn to incorporate first publicly available (linked) web services and then extend their functionality by publicly available procedural knowledge. As a result we envision that *Linked Open Know-How* will become reality in the next years. In practice such a fusion would allow for shifting process knowledge to the location of the data rather then transferring the data over the web to the processes.

The rest of this paper is organized as follows. In Sec. 2 we discuss some foundations for knowledge representation as used for today's and the future semantic web. In Sec. 3 we continue with a critical review of today's LOD. Afterwards, we have a look into a possible future of the semantic web by discussing how the representation of procedural information can enrich LOD in Sec. 4, before we conclude the paper.

2 Data, Information, Know-that, and Know-how

In philosophy and knowledge representation, one traditionally distinguishes between *data*, *information*, and *knowledge* [1]. Briefly said, data can be regarded as a syntactical description of something, like a string "12° C". Data becomes information when it is extended by context and semantics as in "the temperature outside is 12 degree Celcius" (given that concepts such as "temperature" and "Celsius" are well-defined). We speak of knowledge if some information is known by an individual. Furthermore, the common usage of these terms refers to pieces of information we have of the observable part of the world such as in the statements "it is raining" or "cancer is a disease". They are also used to describe relations between entities of the world such as "a disease is an abnormal condition of an organism". Singh [13] introduces the concept of *know-how*—which has been used in philosophy for quite some time [11]—into computer science, artificial intelligence, and knowledge representation. In contrast to factual knowledge (also called *know-that*) know-how refers to knowledge on action and procedures such as in the statements "if it is raining take an umbrella" or "in order to cure disease X you have to administer the antidote Y". Although this two types of knowledge are very similar and often hard to distinguish, research in rational agency often explicitly separates these two. For example, in many formal models for intelligent agents one usually strictly distinguishes between *beliefs* (the factual part of the knowledge of an agent) and *intentions* (which usually corresponds in many implementations to the procedural part) [15], see also most approaches to automatic planning [5]. Singh [13] argues that representing know-how explicitly as part of an agent's knowledge allows for a unified treatment of both types of knowledge. As a result, new procedures can be learned in the same way as new information is obtained and reasoning can be performed in the same way on both types allowing, e. g., to verify whether newly acquired knowledge on procedures can help in solving a task and to what certainty [9].

3 What is there today?

Today's web of data features a similar strong distinction between know-that and know-how as do models for rational agents. On the one hand, we have LOD as a collection of interlinked pieces of information. This is already a huge advantage to the situation ten years ago as structured information is made available using the generally agreed upon standard of RDF. The enhancement of LOD in general is an active research field pursued by many contributors. For example, LOD2[1] and LATC[2] are two EU projects providing tools and methods for processing and managing today's LOD. Moreover, we have a series of procedural approaches such as stand-alone applications, web-based agents, and web services. For semantic web service descriptions, there are also some generally agreed upon standards such as OWL-S, WSMO and WSDL. But more popular are practical community-driven approaches like exchanging information on web services via the *programmable web*[3]. Linked services [4, 8] are the first initiative towards a specification of services in RDF and SPARQL. While linked services are able to generate linked data from linked data, they suffer from the limitations that the usage of these services requires the submission of all input data to the services (cf. Sec. 1). Further, while research on web services in general is already quite mature and has led to a lot of desirable properties we also expect from know-how—such as automatic service discovery [2] and dynamic composition of services [10]—they reside on a different conceptual level than the data itself, i. e. web services *use* Linked Open Data but they *are not* Linked Open Data themselves. What is missing so far is a semantical representation of the procedures behind a service, e. g. in terms of source code, that describes *how* particular knowledge is obtained from factual knowledge.

Several ontologies formalize processes and procedural aspects. As part of the Gene Ontology project, the biological process ontology[4] is a collection of biological processes in RDF. WS-BPEL[5] is an XML-based language for process description. It was originally designed to specify Web service orchestration. An OWL representation for scientific workflows is presented in [6]. The *strukt* ontology [12] facilitates the representation and integration of structured workflows (i. e., business processes) and weakly structured workflows (i. e. workflows that require a constant acquisition and sharing of knowledge and are highly dynamic in their execution). It explicitly supports the integration of existing web services such as Web 2-0 applications like Doodle[6] but also Linked

[1] http://lod2.eu/
[2] http://latc-project.eu/
[3] http://www.programmableweb.com/
[4] http://www.geneontology.org/GO.process.guidelines.shtml
[5] http://docs.oasis-open.org/wsbpel/2.0/OS/wsbpel-v2.0-OS.html
[6] http://www.doodle.com/

Data Services [14] as part of the workflow execution. These approaches show the potential of using semantic technologies to formalize procedural knowledge on the Semantic Web. However, a take up of these approaches for describing and sharing *Linked Open Know-How*, i.e., procedural knowledge in Linked Open Data has yet not happened.

4 What will the future bring?

We envision the future semantic web to grow from a data-oriented web to both a web of know-that and a web of know-how. In particular, due to the current success of open source software exchange of procedural information—such as source code—will be enhanced by semantics that allows services and agents to extend their own capabilities.

Example 1. Consider the new knowledge graph feature of Google[7]. This service automatically collects semantic information for a given search query and augments Google's search results with a concise representation of the available information. Imagine you are not searching for a specific piece of (factual) information but for some procedural information such as "how can I use the Facebook API in Python for doing task X?". What you get in the envisioned future is not a recipe on how to solve this question but the actual programming code. By traversing through LOD, the service obtains both the necessary code for performing basic tasks (such as a single method call to the Facebook API) as well as the information on how to combine those basic tasks. Let us think even further and imagine we do not ask this question in a search engine but just tell our program code in a declarative manner what it should do (use the Facebook API for doing task X) and point it to some direction where it can acquire the necessary information.

Example 2. Imagine a website for a travel agency that provides a web interface for booking flights. When a new flight airline appears, the travel agency usually has to extend their website by incorporating this new airline as well by developing new program code to access the new airline's API. If the new airline offers some completely new features—as for example private tours in a helicopter—the website has to be adapted even further. If the airline would disclose the knowledge on what information is provided and how it can be processed, the travel agency could easily just add the new airline to its program and let the program figure out how to access the necessary information.

But what makes the above examples different from standard scenarios for web service discovery [7], planning [10], and composition [3]? The problem in the latter approach is the (more or less) strict technological separation on how knowledge is accumulated in LOD and web services are composed. In today's LOD we, already have the technologies and infrastructure for traversing information and finding new relationships on factual information. The principles of LOD imply one procedure that all web-based agents can follow: dereferencing URIs. This procedure enables all agents to perform the atomic action of browsing and, thereby, discovering new information. So, currently we are in a situation where we built web agents which are capable of extending their factual knowledge by discovering new snippets of information. The actions they can take based on this factual-knowledge are limited to whatever abilities we gave them at design time

[7] `http://www.google.com/insidesearch/features/search/knowledge.html`

and the ability to collect more information. Why not use the same technologies for procedural information? Publishing LOD has become fairly easy, even for non-researchers and non-developers, while the infrastructure for web service composition is still at a very early state. Placing procedural knowledge on the LOD cloud will allow the agents to extend their abilities and perform new actions to fulfill their task.

Example 3. We continue Ex. 2 and consider the perspective of the new airline, which wants their flight booking service to be available for travel agencies. What they currently have to do is to provide a series of web services, an API documentation, and somehow disseminate this information to travel agencies. These three steps could all be merged into a single step by providing the necessary know-how in form of RDF. By doing this, the web service is implicitly reflected in those procedural RDF statements. The documentation can be derived directly using RDF semantics and the tight integration of both procedural and factual information and dissemination can be achieved by standard means of dissemination in LOD, i. e. by linking the data set to other data sets.

In Sec. 1, we already discussed another drawback of today's web of services: the need to disclose data. In order to conduct, e. g. some sophisticated analysis on patient data such as stress X-rays or MR imaging, a medical facility has to send this data to an external service. Although confidentiality of the client is usually preserved in these settings, retrieving the procedural information on how to conduct the analysis would make the patient and the medical personnel feel more comfortable. Another issue with disclosing one's data to a web service is the need for transferring the data over the network. Although the infrastructure of today's web and the capacities of modern web servers are (more or less) sufficient to serve the current need, there are many possible applications where it is more appropriate to let the client do the computation itself.

Example 4. Consider online file conversion tools[8] that allow you to upload a file in some format, convert it to another format, and send it back to you. Offering such a service has high demands on both computational power of the server and on network bandwidth, given that the service is used by many users. For the service provider itself, this is not so much of a drawback as he collects a lot of data with his service that he can exploit otherwise (depending on the business model of the provider). For the user of the service, the uploading and processing time might be annoying (in particular if the service is used often and with big data) and the disclosure of the data might violate confidentiality issues. However, the procedural information of how to convert a file are often publicly available. Instead of sending the file to the conversion server, the know-how on how to convert the file is send to the client who then performs the conversion itself. Usually, the computational power of the client will also suffice for the task.

The scenario above could also be realized by e. g. a Java applet that performs the computation locally on the client. However, this applet also has to be extended manually if novel file formats have to be added and does not adhere to the principles of LOD as we envision them for procedural information.

[8] see e. g. `http://www.online-convert.com/`

5 Conclusion

In this paper, we took a look at a possible future for the semantic web by incorporating procedural information into Linked Open Data. In this incorporation we see, on the one hand, a reasonable projection of the developments of the LOD community and the open source community and, on the other hand, a desirable boost to take both communities a step further. Representing methods, services, and programming code in the context of LOD helps web services and web agents in automatically extending their functionalities without the need of manual integration. As the presentation of our vision has been been done on an abstract level here, we intend to concretize our ideas and develop potential implementations in a more formal context in the future.

Acknowledgments: The research reported here was partially supported by the FP7 projects SocialSensor (EC contract no. 287975) and ROBUST (EC contract no. 257859).

References

1. M. Alavi and D. E. Leidner. Review: Knowledge management and knowledge management systems: Conceptual foundations and research issues. *MIS Quarterly*, 25(1):107–136, 2001.
2. A. Averbakh, D. Krause, and D. Skoutas. Exploiting User Feedback to Improve Semantic Web Service Discovery. In *Proceedings of the 8th International Semantic Web Conference (ISWC'09)*, pages 33–48. Springer-Verlag, 2009.
3. Z. Azmeh, M. Driss, F. Hamoui, M. Huchard, N. Moha, and C. Tibermacine. Selection of Composable Web Services Driven by User Requirements. In *Web Services (ICWS), 2011 IEEE International Conference on*, pages 395–402. IEEE, 2011.
4. J. Domingue, C. Pedrinaci, M. Maleshkova, B. Norton, and R. Krummenacher. Fostering a Relationship between Linked Data and the Internet of Services. In *Future Internet Assembly*, pages 351–366, 2011.
5. M. Ghallab, D. Nau, and P. Traverso. *Automatic Planning: Theory and Practice*. Morgan Kaufmann, 2004.
6. Y. Gil, V. Ratnakar, E. Deelman, G. Mehta, and J. Kim. Wings for Pegasus: Creating Large-Scale Scientific Applications Using Semantic Representations of Computational Workflows. In *AAAI*, pages 1767–1774, 2007.
7. M. Klusch, B. Fries, and K. P. Sycara. Automated semantic web service discovery with OWLS-MX. In *AAMAS*, pages 915–922, 2006.
8. R. Krummenacher, B. Norton, and A. Marte. Towards Linked Open Services and Processes. In *Future Internet Symposium (FIS)*, pages 68–77, 2010.
9. P. Krümpelmann and M. Thimm. A logic programming framework for reasoning about know-how. In *Proc. of the 14th Int. Workshop on Non-Monotonic Reasoning*, 2010.
10. S. A. McIlraith and T. C. Son. Adapting golog for composition of semantic web services. In *Int. Conference on Knowledge Representation, KR*, pages 482–496, 2002.
11. G. Ryle. *The Concept of Mind*. Chicago, 1949.
12. A. Scherp, D. Eißing, and S. Staab. strukt - A Pattern System for Integrating Individual and Organizational Knowledge Work. In *Internat. Semantic Web Conf.*, pages 569–584, 2011.
13. M. P. Singh. Know-how. In A. S. Rao and M. J. Wooldridge, editors, *Foundations of Rational Agency, Applied Logic Series*, pages 105–132. Kluwer, 1999.
14. S. Speiser and A. Harth. Integrating Linked Data and Services with Linked Data Services. In *8th Extended Semantic Web Conference (ESWC)*, pages 170–184, 2011.
15. G. Weiss, editor. *Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence*. MIT Press, Cambridge, MA, USA, 1999.