Semantic Web-driven Development of Services-oriented Systems – Exploiting Linked Data for Services Annotation and Discovery

Stefan Dietze[1], Dong Liu[2], Hong Qing Yu[2], Carlos Pedrinaci[2]

[1] L3S Research Center, Hanover, Germany
dietze@l3s.de

[2] Knowledge Media Institute, The Open University, Milton Keynes, MK76AA, UK
{d.liu, h.q.yu, c.pedrinaci}@open.ac.uk

Abstract. Within a service-oriented architecture (SOA), software components are accessible via well-defined interfaces. To this end, the discovery and integration of Web services and APIs is becoming an increasingly important task in present-day software engineering. Despite considerable research dedicated to Semantic Web Services (SWS), structured semantics are still not used significantly to facilitate services and API discovery. This is due to the complexity of comprehensive SWS models and has led to the emergence of a new approach dubbed *Linked Service* which adopt Linked Data principles to produce simplified, RDF-based service descriptions that are easier to create and interpret. However, current Linked Services tools assume the existence of services documentation (HTML, WSDL) and do not sufficiently support non-functional properties (NfP). Therefore, we introduce SmartLink, a Web-based editor and search environment which allows both humans as well as machines to produce light-weight service descriptions from scratch by addressing both, functional and non-functional service properties.

Keywords: Software Engineering, Semantic Web, Linked Data, SmartLink.

1 Introduction

An essential part of Software Engineering nowadays is concerned with the discovery of reusable *software components* which satisfy one or more requirements of the overall system to be implemented. The past decade has seen the emergence and large-scale success of another fundamental paradigm: *service-orientation*. Within a service-oriented architecture (SOA), components are accessible via well-defined interfaces and usually exchange messages via remote-procedure calls (RPC), HTTP or SOAP. Particularly the emergence of REST-ful services has led to the widespread availability of public and reusable Web APIs, such as the wide range of APIs offered by Google[1]. To this end, the discovery and integration of Web services and APIs is becoming an increasingly important task in present-day software engineering.

[1] https://code.google.com/

Research efforts in the area of *Semantic Web Services (SWS)* were mainly aiming at the automation of Web service-related tasks such as discovery, orchestration or mediation. Several conceptual models, such as OWL-S [6], WSMO [3], and standards like SAWSDL [7] have been proposed, usually covering aspects such as service capabilities and interfaces. However, SWS research has for the most part targeted WSDL or SOAP-based Web services, which are not prevalent on the Web. Also, due to the inherent complexity required to fully capture computational functionality, creating SWS descriptions has represented an important knowledge acquisition bottleneck and required the use of rich knowledge representation languages and complex reasoners. Hence, so far there has been little take up of SWS technology within non-academic environments. That is particularly concerning since Web services – nowadays including a range of often more light-weight technologies beyond the WSDL/SOAP approach, such as RESTful services or XML-feeds – are in widespread use throughout the Web. That has led to the emergence of more simplified SWS approaches such as WSMO-Lite [9] SA-REST [7] and Micro-WSMO/hRESTs [4] which benefit from simpler models expressed in RDF(S).

While the Semantic Web has successfully redefined itself as a Web of *Linked (Open) Data (LOD)* [1], the emerging *Linked Services* approach [7] exploits the established LOD principles for service description and publication. By supporting annotation of a variety of services, such as WSDL services as well as REST APIs, the Linked Services registry and discovery engine iServe[2] enables publishing of service annotations as linked data expressed in terms of a simple conceptual model: Minimal Service Model (MSM), a simple RDF(S) ontology able to capture (part of) the semantics of both Web services and Web APIs.

However, while Linked Services appears to be a promising stream of research, we observe two major issues which hinder a large-scale take-up of the Linked Services approach:

(i1) Lack of consideration of non-functional service properties and less formal metadata
(i2) Lack of appropriate editors and annotation environments

With respect to (i1), previous efforts have largely focused on formalizing the actual functionalities of a service (capabilities, interfaces). However, in order to allow assessment about suitability of individual services or APIs for a particular service consumer, *non-functional properties (NfP)* are of crucial importance. These include, for instance, basic metadata about the development status or the licensing model as well as information about the quality of service (QoS). In addition, less formal services annotations turned out to be very useful since one of the yet most established mode of using Linked Services aims at rather semi-automated service discovery where developers browse or navigate through Linked Services libraries based on filtering mechanisms, as opposed to fully automated services discovery and orchestration. While the latter is fundamentally dependent on complex and formal specifications of services capabilities and interfaces (i.e. functional properties) the former can be supported based on rather light-weight and often non-functional service

[2] http://iserve.kmi.open.ac.uk

metadata, such as classifications, tags or development status information. However these are not sufficiently supported within current schemas such as MSM and WSMO-Lite.

With regard to (i2), editors had been developed which support developers in creating semantic annotations for services: SWEET [5] (SemanticWeb sErvices Editing Tool) and SOWER (SWEET is nOt a Wsdl EditoR). However, SWEET and SOWER build on the assumption that either HTML documentation of services/APIs (SWEET) or WSDL files (SOWER) are available as starting point for annotation. While that holds for a certain set of services, a growing number of services on the Web neither provide a WSDL nor an HTML documentation and hence, current Linked Services editors cannot be deployed in a range of cases. In this regard, we particularly would like to promote an approach where services documentation relies exclusively on structured RDF(S) while additional human-readable documentation is not provided manually but automatically generated to avoid redundancies.

Therefore, we introduce *SmartLink*[3] ("SeMantic Annotation enviRonmenT for Linked services"), which addresses (i1) and (i2) by contributing:

(a) an RDF schema and data store for service NfP
(b) an integrated editing and browsing environment for Linked Services on the Web (taking into account both functional and non-functional data)

In the following Section we provide some background information on Linked Services, while Section 3 introduces the SmartLink NfP schema. Section 4 describes overall architecture of SmartLink. We finally discuss our results in Section 6.

2 Non-functional properties for Linked Services

Previous work dealing with the exploitation of SWS and Linked Services technologies in NoTube[4] and mEducator[5], as described in [2][10], has shown that one of the most established and accepted use cases for Linked Services annotations aims at browsing and searching services in a meaningful way as opposed to automated services discovery and execution. To this end, Linked Services seem of particular use when aiding developers in finding APIs for a given software engineering task.

In this regard, formal specifications turned out to be less important while light-weight service annotation with tags/keywords and classifications plaid a vital role. Particularly when supporting collaborative annotation of entities – services like any documents, content or data – by a multiplicity of service consumers and developers, formal correctness of the generated data can hardly be enforced and means are required to provide descriptions in a more loose and flexible way. For instance, in many cases, Linked Data resources can be roughly associated with a service – for instance, by tagging it with a service category or keyword which might not provide formal enough semantics to facilitate automation of discovery-based execution, but might still be useful to facilitate users in finding appropriate services. For instance, an

[3] http://smartlink.open.ac.uk & http://kmi.open.ac.uk/technologies/name/smartlink
[4] http://notube.tv
[5] http://www.meducator.net

API exposing *metadata* of resources could be associated with a keyword "metadata" or a reference to http://dbpedia.org/resource/Metadata. However, the current scope of SWS and Linked Services does not provide appropriate facilities to represent such rather lose relationships in an appropriate way but focuses on formal representations of service elements, such as message parts or operations. In that respect, a need for less formal services annotations was observed, to facilitate developers and service consumers to collaboratively annotate services based on Linked Data principles without constraining them by insisting on complete coherence of the provided annotations. Instead of enforcing non-contradictory data, collaborative annotation schemas need to embrace diversity even if that reduces the opportunities for reasoning-based automation.

On a similar note, current service description schemas (e.g., MSM, OWL-S, WSMO-Lite) seem to be fundamentally focused on functional properties while not providing sufficient support for NfPs, which would, for instance, allow users to specify licensing schemes, quality of service information or development status descriptions. While some schemas already allow the association of additional service information with particular service instances, the use of dedicated Linked Data vocabularies to further specify NfPs is still underdeveloped.

SmartLink NfP schema

To this end, we have developed a dedicated schema that addresses the aforementioned issues by (a) focusing in particular on NfPs and (b) facilitating collaborative, naturally diverse and less formally coherent annotation. To ensure the widespread applicability and reusability of the NfP schema, we reuse existing ontologies and vocabularies rather than constructing new ontologies from scratch. As shown in Fig. 1, the schema captures four main aspects of the non-functional properties of Web services, i.e. *social*, *technical*, *licensing* and *QoS*. *Social* attributes include human factors such as developer, contact person, organisation, project. The FOAF[6] vocabulary is adopted to describe those personal and social factors. Furthermore, tags attached to Web services are also regarded as an important social attribute, which helps in service classification and organization. Thus, the CommonTag[7] vocabulary is adopted to support the tagging by ensuring interoperability of provided service tags. The *technical* NfPs refer to information about how to interact with the services and cover, for instance, the communication protocol (e.g. HTTP and SOAP), data (exchange) format (e.g. XML, RDF and JSON), status (e.g. testing, final, work-in-progress), authentication model (e.g. HTTP Basic, API Key, OAuth). It is worth noting that technical NfPs do not describe the behaviours of services, but clarify the prerequisites for consumers to invoke those Web services.

The *licensing* properties indicate the terms and conditions with respect to the usage of individual Web services. As shown in Fig. 1, we currently define four concepts for the licensing properties, i.e. service license, data license, usage limits and fees. A service license authorizes and constrains invocation of the service, whereas a data license is for the reuse or repurpose of data generated or provided by the service. Usage limits cover the amount of times of service invocation within a certain time period, or the minimum interval between two times of invocation. Obviously, fees are

[6] http://www.foaf-project.org/
[7] http://commontag.org/

applicable to non-free services only and refer to the price a consumer needs to pay for consuming a service.

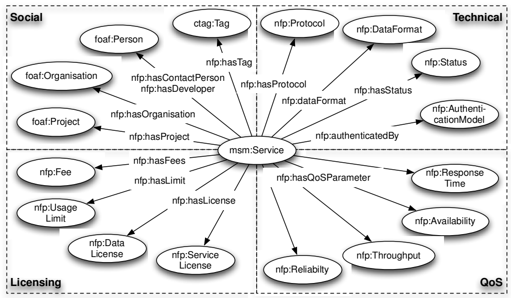

Fig. 1. A partial view of the SmartLink NfP Schema.

The *licensing* properties indicate the terms and conditions with respect to the usage of individual Web services. As shown in Fig. 1, we currently define four concepts for the licensing properties, i.e. service license, data license, usage limits and fees. A service license authorizes and constrains invocation of the service, whereas a data license is for the reuse or repurpose of data generated or provided by the service. Usage limits cover the amount of times of service invocation within a certain time period, or the minimum interval between two times of invocation. Obviously, fees are applicable to non-free services only and refer to the price a consumer needs to pay for consuming a service.

With respect to the quality of Web services, we adopt the model from [12], where the QoS parameters are divided into two classes: *objective parameters* and *subjective parameters*. The former are quantitative measures like availability, reliability, throughput and response time, whereas the latter are qualitative measures like user ratings. Here, we only focus on the objective QoS parameters, because theses have been published on the Web[8,9].

Schema mapping and alignment

We reuse existing vocabularies to represent the NfPs of Web services. It allows interoperability between individual service description repositories and facilitates the import of publicly available service NfP metadata into SmartLink. Here, we take *ProgrammableWeb*[10] as an example to demonstrate schema mapping and alignment. Parts of the mappings between our schema and the one of ProgrammableWeb are shown in the table below. In addition, *API Status*[8] provides the statistics of the availability and response time of public APIs. Similarly, *Mashery*[9] monitors on the availability and response time of a set of services. The metadata these repositories exploit can be completely mapped to SmartLink schema. Moreover, the data can also be imported to SmartLink.

[8] http://api-status.com/
[9] http://developer.mashery.com/status
[10] http://www.programmableweb.com/

Table. 1. NfP schema mapping between SmartLink and ProgrammableWeb.

SmartLink NfP Schema	ProgrammableWeb's Schema
ServiceLicense	Commercial Licensing
ServiceLicense	Non-Commercial Licensing
Fee	Usage Fees
Usage Limit	Usage Limits
Authentication Model	Authentication Model
foaf:Organization	Provider
foaf:Company	Company
foaf:weblog	API Blog

3 SmartLink: a Linked Services editor and browser

In order to provide a Linked Services editor which allows (a) the annotation of RESTful services without any pre-existing documentation and (b) annotation of services according to multiple schemas, in particular SmartLink NfP, we have developed the SmartLink editor. SmartLink provides editing and browsing facilities to interact with multiple RDF stores and data sets. It allows annotation of services from scratch, that is, without any pre-existing services documentation such as WSDL or HTML files, as assumed by existing annotation tools (Section 1).

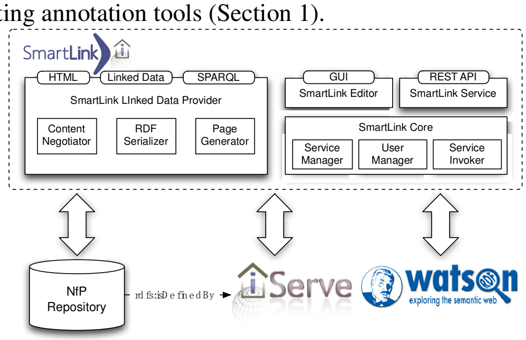

Fig. 2. SmartLink – overall architecture.

As shown in Fig. 2, SmartLink operates on top of Linked Data stores that exploit the MSM and the SmartLink NfP schemas and are interlinked with other Linked Data sets. MSM-schema properties are directly stored in iServe, while additional properties are captured in our SmartLink NfP repository[11]. The repository provides a SPARQL endpoint[12]. Following *rdfs:isDefinedBy* links from SmartLink to iServe, more information about the functionalities and behaviours of the services can be retrieved. Being an LOD-compliant environment, one of the core features of SmartLink is the capability to associate service descriptions with so-called model references which refer to RDF descriptions in external vocabularies defining the semantics of the

[11] http://ckan.net/package/smartlink
[12] http://smartlink.open.ac.uk/smartlink/sparql

service or its parts. However, while this feature is useful and even necessary in order to provide meaningful service models, finding appropriate model references across the entire Web of data is a challenging task. Therefore, SmartLink uses established Linked Data APIs – currently the WATSON[13] API - to identify and recommend suitable model references to the user.

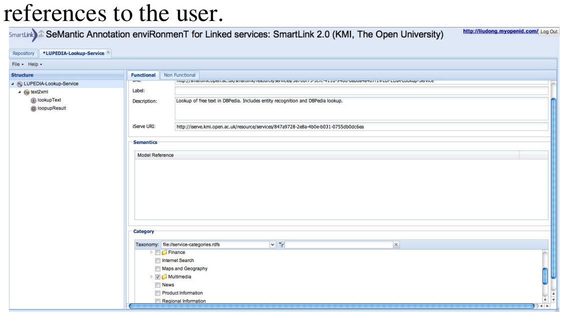

Fig. 3. SmartLink – Service editor.

After loading RDF triples from both iServe and SmartLink, the editor visualizes the description of a service as shown in Fig. 3. The left-hand side of the editor is the tree-based overview of the service, which represents a hierarchy composed of a service, its operations and input/output messages. The right hand side displays more details about the selected element in a form, which essentially include the semantics, categories, and literal descriptions. To persistently store changes made to a service description, SmartLink publishes the descriptions as Linked Data by invoking the RESTful APIs provided by iServe and the SmartLink NfP repository. SmartLink currently provides mechanisms that enable the export of particular service instances as RDF or human-readable HTML. In order to facilitate service model transformation between MSM and other SWS formalisms, current research deals with the establishment of an export mechanism of MSM/SmartLink NfP services. In addition, SmartLink also offers a simple UI for filtering services by NfPs. That way, developers can easily construct queries without having to formulate SPARQL queries to create specific views on the services data.

4 Discussion and conclusion

In this paper, we have proposed SmartLink which provides (a) an RDF schema which allows to describe non-functional properties of Web APIs and services (SmartLink NfP) and (b) a public environment which enables developers to annotate services and store descriptions in a public Linked Data-compliant store, to interlink them with other service descriptions such as the ones offered by iServe, and to search for available services and APIs by exploiting the structured semantics of the SmartLink NfP repository. To this end, SmartLink facilitates software engineering processes, particularly in the context of the prevailing SOA paradigm, by supporting developers in annotation and discovery of software components, i.e., services and APIs, across the Web.

[13] http://watson.kmi.open.ac.uk/WatsonWUI/

Currently ongoing work deals with the exploitation of SmartLink in the context of two European projects, NoTube and mEducator (see [10]). While NoTube exploits the SmartLink approach merely as a means to aid software developers in documenting and searching software/services, in mEducator SmartLink also supports the execution and alignment of heterogeneous services. However, while the currently implemented execution approach is tailored to a specific kind of services – educational metadata harvesting services – no general-purpose execution approach had been developed yet.

From our initial use cases, a few observations have been made which will shape our future efforts. Current research and development deals with the extension of the MSM/SmartLink NfP schemas by taking into account execution and composition oriented aspects. These extensions will be supported by the development of additional APIs, which allow the discovery, execution and semi-automated composition of Linked Services in a general-purpose fashion.

Acknowledgements. This work is partly carried out within the research projects NoTube and mEducator, kindly funded by the European Commission. The authors would like to thank the European Commission for their support.

5 References

[1] Bizer, C., T. Heath, et al. (2009). "Linked data - The Story So Far." Special Issue on Linked data, International Journal on Semantic Web and Information Systems (IJSWIS).

[2] Dietze, S., Benn, N., Yu, H., Pedrinaci, C., Makni, B., Liu, D., Lambert, D., and Domingue, J. (2010) Comprehensive service semantics and light-weight Linked Services: towards an integrated approach, Workshop: Fourth International Workshop SMR2 2010 on Service Matchmaking and Resource Retrieval in the Semantic Web at 9th International Semantic Web Conference (ISWC), Shanghai, China

[3] 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.

[4] Kopecky, J.; Vitvar, T.; and Gomadam, K. (2008). MicroWSMO. Deliverable, Conceptual Models for Services Working Group, URL: http://cms-wg.sti2.org/TR/d12/v0.1/20090310/d12v01_20090310.pdf.

[5] 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.

[6] Martin, D., Burstein, M., Hobbs, J., Lassila, O., McDermott, D., McIlraith, S., Narayanan, S., Paolucci, M., Parsia, B., Payne, T., Sirin, E., Srinivasan, N., and Sycara, K. (2004). OWL-S: Semantic Markup for Web Services. W3C Member Submission November 2004.

[7] Pedrinaci, C., and Domingue, J. Toward the Next Wave of Services: Linked Services for the Web of Data. Journal of Universal Computer Science, 16(13):1694–1719, 2010.

[8] Sheth, A. P., Gomadam, K., and Ranabahu, A. (2008). Semantics enhanced services: Meteor-s, SAWSDL and SA-REST. IEEE Data Eng. Bul l., 31(3):8–12.

[9] 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 SemanticWeb Conference, LNCS. Berlin, Heidelberg: Springer Verlag.

[10] Yu, H. Q., Dietze, S., Li, N., Pedrinaci, C., Taibi, D., Dovrolis, N., Stefanut, T., Kaldoudi, E., Domingue, J., A Linked Data-driven & Service-oriented Architecture for Sharing Educational Resources, in Proceedings of Linked Learning 2011: the 1st International Workshop on eLearning Approaches for the Linked Data Age, CEUR-Vol 717, 2011.