Webology, Volume 10, Number 1, June, 2013

Linked-OWL:
A new approach for dynamic linked data service workflow composition

Hussien Ahmad
Faculty of Information Technology, Damascus University, Baramkeh, Damascus, Syria. E-mail: hussien824 (at) gmail.com

Salah Dowaji
Associated Professor, Faculty of Information Technology, Damascus University, Baramkeh, Damascus, Syria. E-mail: sdowaji (at) tarassul.sy

Received October 5, 2012; Accepted February 25, 2013

Abstract

The shift from Web of Document into Web of Data based on Linked Data principles defined by Tim Berners-Lee posed a big challenge to build and develop applications to work in Web of Data environment. There are several attempts to build service and application models for Linked Data Cloud. In this paper, we propose a new service model for linked data "Linked-OWL" which is based on RESTful services and OWL-S and copes with linked data principles. This new model shifts the service concept from functions into linked data things and opens the road for Linked Oriented Architecture (LOA) and Web of Services as part and on top of Web of Data. This model also provides high level of dynamic service composition capabilities for more accurate dynamic composition and execution of complex business processes in Web of Data environment.

Keywords

Linked data; Linked Data Services; Dynamic Workflow; OWL-S; RDF; SPARQL

Introduction

The need of high level accurate interaction in large scale environments such as the Web has formed until now a big challenge of the different models of services and service composition. Semantic web services (Cardoso, 2006) tried to add semantics at the different levels of the Web service stack to enable an accurate and common understanding between the different parties of the service composition and consuming. Meanwhile, the linked data principles (Berners-Lee, 2009) led to a new form of web "Web of Data" in which knowledge is represented as links between things where a thing can be anything one can imagine such as document, word, resource, or whatever. On top of this new form of web is the Linked Open Service approach (Norton et al., 2010a) that was emerged to provide services and service composition in Web of Data environment using the RDF (Klyne & Carroll, 2004), SPARQL (Prud'hommeaux & Seaborne, 2008) and RESTfull services (Richardson & Ruby, 2007) over HTTP technologies.

Linked Open Services (LOS) provides a SPARQL-driven approach for service composition, which makes RDF the interface of all interaction and uses SPARQL in the service definition. In LOS, all services are shown as RDF prosumers and it may wrap any type of service into the LOS architecture (Norton et al., 2010b).

LOS service takes an RDF graph as input and produces another one as output and the composition mechanism uses the SPARQL queries and constructs to define the processes (Norton et al., 2010b). Since SPARQL is mainly a query language for RDF, it lacks to the workflow composition capabilities while OWL-S has a strong composition constructs.

LIDS (Speiser & Harth, 2011) is another linked data service approach that focuses on data-providing service which returns specific data according to specific user input using HTTP methods and RESTfull service model and SPARQL.

The challenge is still not to go far from human-oriented web in the process of looking for accuracy and machine-oriented web (Hausenblas, 2009). This challenge brings the issues of human-machine interaction to the surface so that any proposed model should respect both sides of the process, humans and machines.

In this research, we provide a new approach to compose services in the Web of Data environment by redesigning OWL-S into what we call "Linked-OWL" to cope with the principles of Linked Data and RESTful services and to keep the composition capabilities of OWL-S (Martin et al., 2004) in addition to extending it to fit with the dynamic execution requirements such as substitution management and change management.

Our approach continues to use this new service architecture in a four stages composition mechanism to provide high level of dynamicity in service composition. Using this approach, the services will be linked to each other either by composition or by the underlying linked data sets used to describe the services. This linkage forms service sets which will together forms a Web of Services over and along with the Web of Data.

Motivations and requirements for dynamic service composition

The need of dynamic web-based workflows composition and execution in Web of Data environment leads to three types of requirements:

- Linked data requirements
- Workflow patterns requirements
- Dynamicity requirements

Linked data requirements

Linked data principles defined by Tim Berners-Lee (2009) are:

- Use URIs as name for things
- Use HTTP URIs so that people can look up those names
- When someone looks up a URI, provide useful information using the standards (RDF*, SPARQL)
- Include links to other URIs so that they can discover more things

It is clear that linked data relays on three main technologies which are Uniform Resource Identifier (URI), HyperText Transfer Protocol (HTTP) and Resource Description Framework (RDF).

The question is: Can a service be a part of Linked Data space? Since a service is an entity "*a functional entity*" and can have an address and a description then it is a potential entity in the linked data space, but how? Starting from the linked data principles the service as a web entity should:

- [REQ 01] Have a single URI to identify it.
- [REQ 02] The URI should be an HTTP URI so that the service input/output could be transferred over HTTP protocol and not using other technologies.
- [REQ 03] Have a description that could be accessed via its URI.
- [REQ 04] The service description should be linked data typed so it is browsable via SPARQL in RDF format and leads to other information helping in understanding and composition of the service.
- [REQ 05] Services should be interlinked with each other to form service sets and linked with the data sets from Web of Data.

Workflow patterns requirements

This is to provide all kinds of expected composition capabilities to enable very complex business processes starting from primitive services. In previous works (van der Aalst et al., 2003; Wohed et al., 2002) these requirements were analyzed and defined quite well.

The patterns should be available in any workflow enabling environment are (Wohed et al., 2002): sequence, parallel split, synchronization, exclusive choice, simple merge, multi choice, synchronized merge, multi merge, cycles and looping, deferred choice, interleaved parallel routing, implicit termination and cancel activity and case.

The workflow composition patterns enable high level of linking services with each other in a structured manner in order to form complex business processes and also to form linked service sets.

- [REQ 06] Service model should enable service composition using workflow composition patterns.

Dynamicity requirements

Dynamicity enabling in workflows is a key feature that should be available. This feature should exist on two levels of the workflow lifecycle:

Workflow design time: To enable a dynamic composition of the business process starting from a single user request we should satisfy the following requirements:

- [REQ 07] The composition process depends on ontology semantic matching and knowledge based analysis to transform the user request into a valid and correct business process (Gabula & Hoheisel, 2007).
- [REQ 08] At any level of request refinement and process composition, the user should be able to provide a feedback and redirect the composition in different directions.

Workflow execution time: To enable a controlled execution of the process providing the following capabilities, we should satisfy the following requirements:

- [REQ 09] Transactional control: it must provide the transaction grouping actions in order to have a high level and dynamic error management.
- [REQ 10] Substitution management: it must provide the capability of defining multi-level binding for the same task (Beco et al., 2006).
- [REQ 11] Process change during execution: it must provide the ability of process change in terms of definition and binding (Caeiro-Rodriguez et al., 2008).

Our Approach: Linked-OWL

Linked-OWL is a linked data service and not a service for linked data space. Linked-OWL uses a customized type of RESTful services that respects all principles of Linked Data and at the same time keeps the strong composition mechanism of OWL-S present in order to facilitate the dynamic composition of Linked-OWL services into complex business processes.

Linked-OWL puts the keystone of Linked Oriented Architecture (LOA) in which service are: (i) linked data services, (ii) deployed in linked data repositories, and (iii) consumed in linked data space.
Figure 1 shows the components of Linked Oriented Architecture and their links with each other.

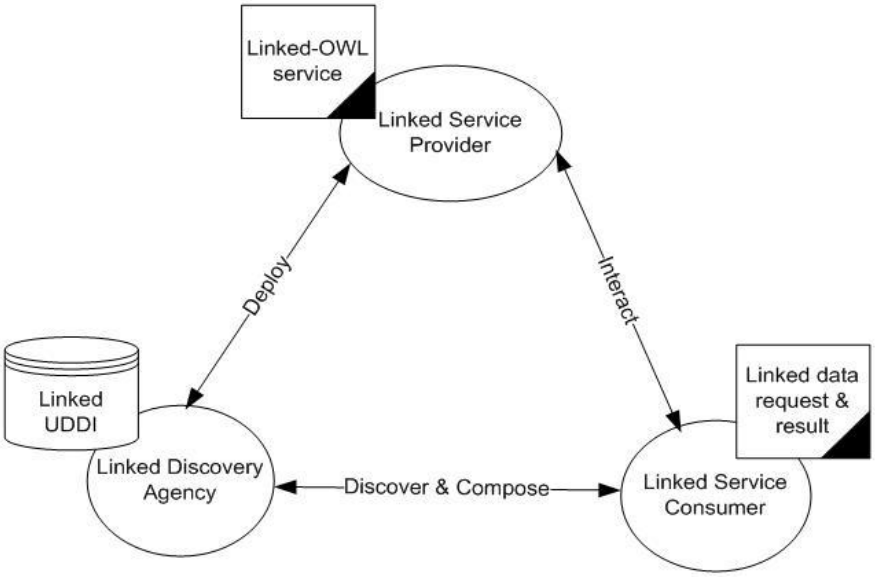

Figure 1. Linked Oriented Architecture

In LOA, (i) a service provider composes services using Linked-OWL service architecture described in the next section and (ii) deploys them in the Linked-UDDI repository using APIs provided by the Linked-UDDI itself to enable Linked-OWL service deployment and then (iii) a service consumer asks for an existing service or compose complex ones starting from those deployed in Linked-UDDI using the discover and compose APIs.

From provider point of view, a service may be simple with its functionality included inside it and provided by the provider himself or complex which is composed from other services deployed in one or more Linked-UDDI from different service providers.

From consumer point of view, a service is already deployed in Linked-UDDI and then it could be consumed directly or the consumer has to compose the service starting from his request using the dynamic composition mechanism described in section entitled "*Dynamic Linked-OWL Services' Composition*". In case of composed service, the consumer has the choice to share his experience with one Linked-UDDI repository to enrich the existing service sets by adding new valid functional and descriptive links.

The Linked-UDDI is a UDDI repository provided with APIs to serve in the linked data space. Each Linked-UDDI node forms a service set which is linked to other nodes either functional using the service composition links or descriptively using the links from the underlying linked data sets.

LOA architecture forms a basis for a Web of Services which lies over and along with Web of Data. From users' point of view, the Web of Services will be part of the Web of Data since it uses the same concepts and technologies to create, deploy and access services, but from internal point of view the Web of Services will be over the Web of Data and uses its linked data sets to provide its functionality.

Linked-OWL Architecture

Linked-OWL is a specific type of RESTful services that reuse some parts of the OWL-S and extends its syntax to enable substitution, transaction and change management. Linked-OWL has profile, model and process parts.

The profile part reuses the concept of service profile from OWL-S to describe "what the service does" using only RDF graphs to describe all the service elements to enable service deployment and consumption.

The model part reuses the concept of service model from OWL-S to describe how to access the service by telling what are the service input/output, service steps, and constraints using only RDF graphs.

The process part reuses the composition constructs from OWL-S to build a complex business processes starting from other Linked-OWL simple or complex services in addition to the group, substitution and transaction constructs to enable high level substitution, change and error management.

Figure 2 shows simple Linked-OWL service architecture. Linked-OWL does not have the grounding part of OWL-S because the service functionality is included in simple Linked-OWL service and service URI is used for both service composition and access.

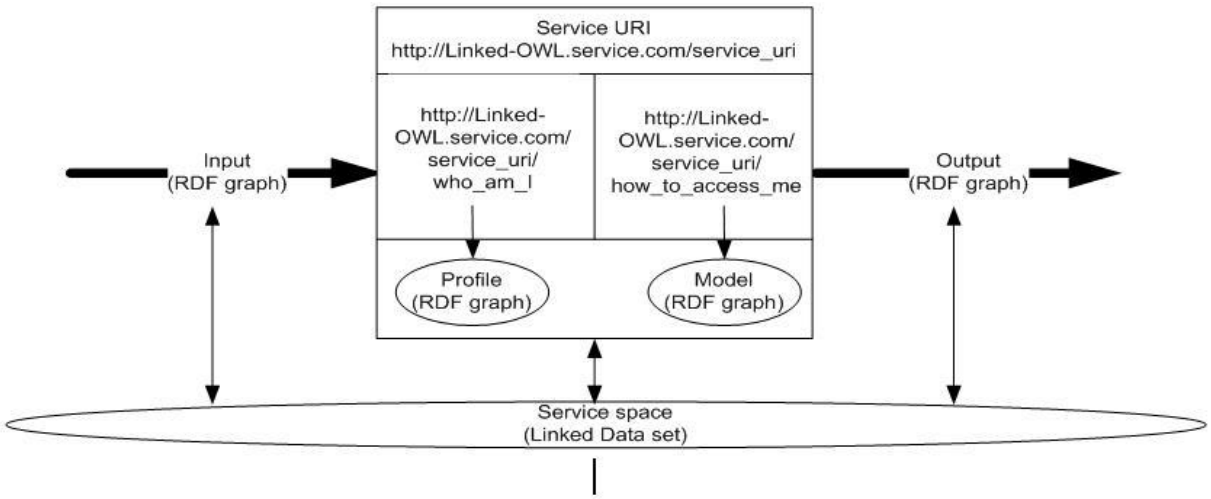

Figure 2. Simple Linked-OWL Service

Linked-OWL does not wrap any other type of services in comparison to LOS which provides service wrapper and makes the input/output lowering and shifting on activity level from outside the service (Norton et al., 2010a). LOS tries to bring the different models of web services from ordinary web to Web of Data space while Linked-OWL define a new service model for Web of Data to build Web of Data APIs and to form the Web of Services on the top of Web of Data.

Linked-OWL overcomes the parameter passing and grounding issues by using only RDF graphs for Input/ Output parameters and URIs to define grounding to other Linked-OWL services since it is a RESTful service type and then it is accessible via single URI.

The use of RDF graphs for parameter passing will ease the understanding of the parameters semantics and make the relation between input and output semantics clear since they are based on the same linked data set for each service. This is clear for a simple service that uses the same data set to represent its input, output, service profile and service model, but a new requirement appears with complex services because it is not necessary to use the same data set for all simple services that participate in its composition.

The structure of linked open data guarantees the common understanding of the different data sets if there were links between the same concepts in each data set with those in the other data sets (Berners-Lee, 2009). Figure 3 shows the concept links between different data sets in a complex Linked-OWL service where the output RDF graph of one activity from a specific linked data set is mapped into another RDF graph input of another activity. The mapping is done on two levels: graph structure level and values level.

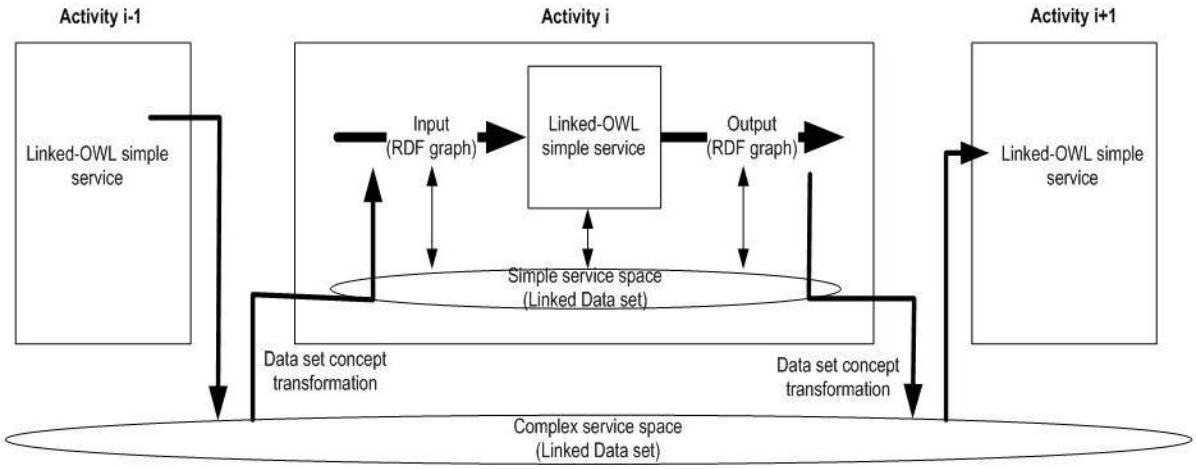

Figure 3. Complex Linked-OWL Service

Dynamic Linked-OWL Services' Composition

Dynamic service composition is enabled through a four stages approach (Paik et al., 2011): Request definition, candidate discovery, candidate selection, and process execution (see Figure 4).

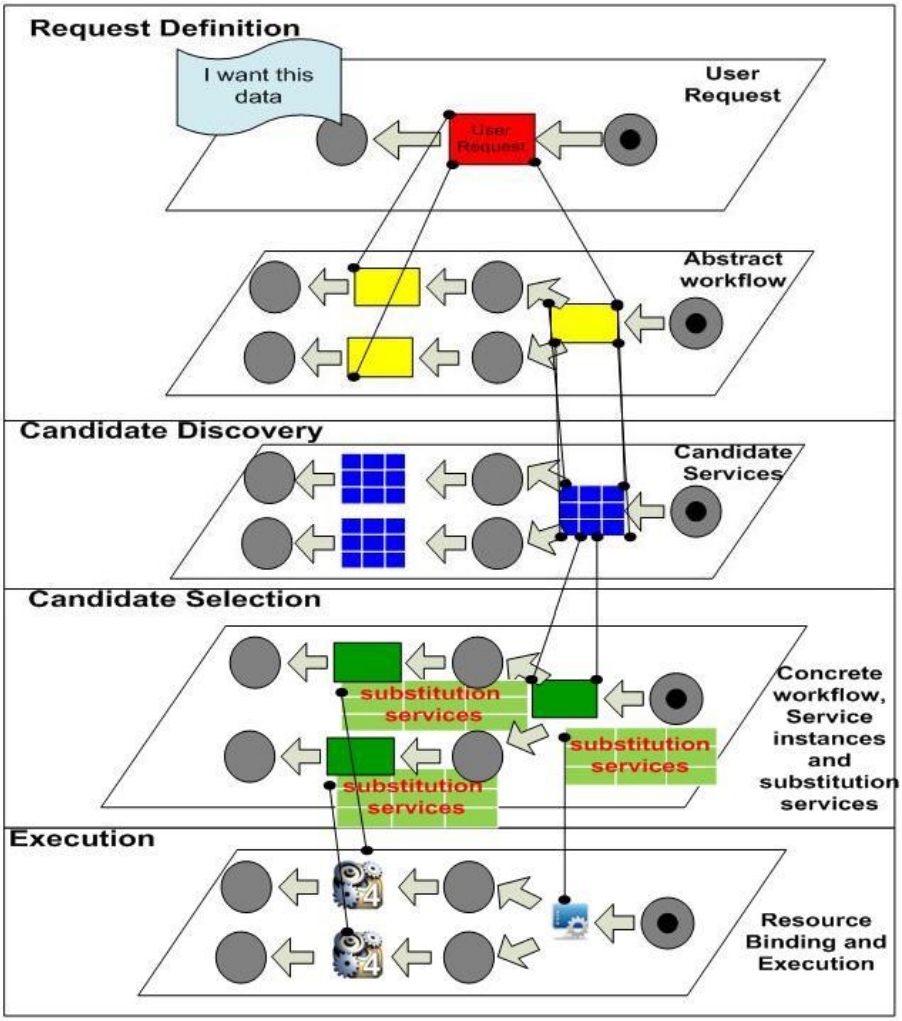

Figure 4. Dynamic service composition stages (Adapted from: Gabula & Hoheisel, 2007)

Linked-OWL implements an updated version of the four stages approach defined in (Paik et al., 2011) to conform to linked data adoption and to enable new composition constructs defined in Linked-OWL.

1. **Request definition**: The user composes his request in RDF or abstract Linked-OWL format or even in any language he wants.
2. **Candidate service discovery**: This stage is the main stage in workflow composition life cycle. It consists of two sub-stages: (i) candidate discovery and (ii) request simplification. The discovery engine searches for potential matches in the Linked UDDI where the concrete services are deployed using SPARQL query match against the service profile and model; if the requested services exist then it moves to the next step "Candidate Selection"; otherwise it makes an advanced search in the deployed services' profiles and models and targets linked data set using SPARQL queries and the input RDF description of the requested service to find potential simplification of the user request to form a simplified request and then after user feedback and approval the simplified request enters the search engine again to find potential matches.

The request simplification can be done on a part of the entire request if the services of other parts exist and deployed in the Linked-UDDI.

The natural language-formatted user request will pass in the simplification engine to build an RDF graph for the request to be able to find service matches in the candidate discovery stage. This process called request refinement may be repeated several times with user feedback and approval in order to have the most accurate RDF graph and then the correct service process.

3. **Candidate service selection**: Once the service matches are found the selection engine will select the primary services and the substitution ones based on the selection criteria defined by the user in the request and the user feedback.
4. **Execution**: Once the selection stage is done the execution starts with the ability to change any selected candidate or even to change an entire portion of the built workflow. Once an error occurs, the next substitution in the queue will be executed unless the retry threshold is not reached yet. The execution of any substitution will give the control back to main path of the process and the process execution continues as planned first.

Figure 5 shows the architecture of the four stages approach described above.

The updated composition approach adds the simplification engine to the architecture instead of inner ASC (Automatic Service Composition) in original approach and uses the linked data sets as knowledge base to fetch the semantics of the user request to compose complex services and to define the rules of parameter passing and relations between input and output of each atomic service. On the other hand, the architecture of Linked-UDDI provides hierarchical categories in form of nested composition patterns, i.e., a category of complex service will have the categories of its atomic services related in the same form of the workflow in the complex service. This architecture will decrease the simplification cycles needed to find appropriate services for a user request.

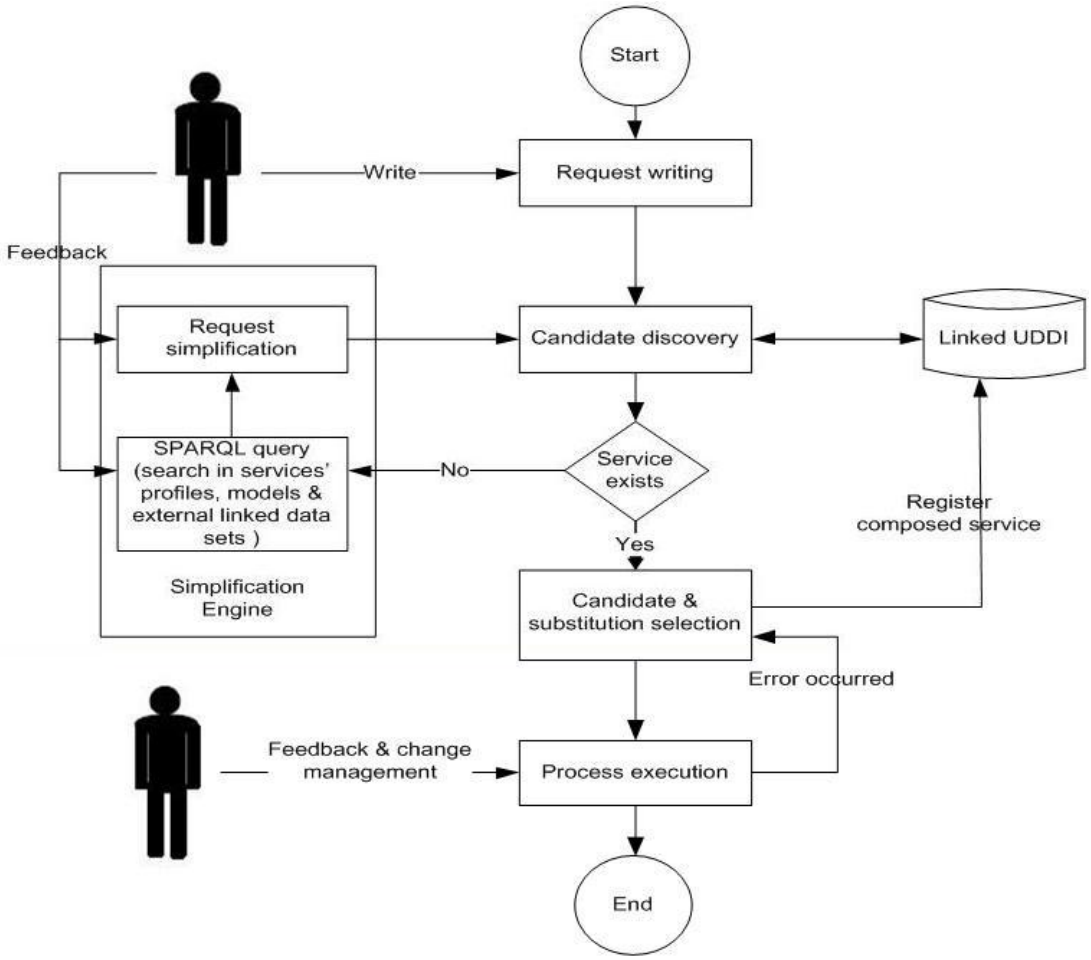

Figure 5. Dynamic Service Composition Architecture

Discussion

The proposed service architecture Linked-OWL meets all the motivation requirements defined in the section entitled "*Motivations and requirements for dynamic service composition*". The following sections make an architectural analysis on each type of requirements against the proposed Linked-OWL.

Linked data requirements

Linked-OWL service architecture shifts the service concept from a function into a thing where things can be represented easily in the linked data paradigm. The Linked-OWL service has:

- A single name to identify it, which is its HTTP URI http://Linked-OWL.service.com/service_uri. This satisfies the requirements [REQ 01] and [REQ 02].
- Linked-OWL is RESTful service so it uses only HTTP protocol to provide its functionality. This satisfies the requirement [REQ 02].
- Two description properties which are accessible via service name:
  - Profile "Who_am_I" property: an RDF graph to describe what the service does: "http://Linked-OWL.service.com/service_uri/who_am_I".

- Model "How_to_access_me" property: an RDF graph to describe how to access the service: "http://Linked-OWL.service.com/service_uri/how_to_access_me".

- Input and Output parameters which are RDF graphs.

The previous two points satisfies the requirements [REQ 03] and [REQ 04].

Linked-OWL service keeps the functionality of the service but transforms it into a thing which a machine talks to via its name and looks at it from different angles (input parameters RDF graph values) and then has different pictures of it (output parameters RDF graph values). This new understanding of the service puts it in the linked data paradigm without any additional effort and then it will be part of the linked data space as all things on the Web are.

The complex Linked-OWL services are functionally linked using structured links "Workflow Composition Patterns" and are descriptively linked using the underlying linked data sets. This linkage leads to service sets deployed in Linked-UDDI repositories forming what we call "Web of Services" on top of Web of Data. This two level linkage provides a solution for [REQ 05].

It is clear now that Linked-OWL service does not only meet the linked data requirements defined above, but also it tries to put the service concept in the same class of the thing concept defined in the original linked data principles.

Workflow patterns

OWL-S supports most of the workflow composition patterns including sequence, parallel split, synchronization, exclusive choice, simple merge, multi choice, synchronized merge, multi merge, cycles and looping, implicit termination but it does not support deferred choice, interleaved parallel routing and cancel activity and case.

The adoption of these composition constructs from OWL-S in Linked-OWL means that Linked-OWL meets most of the workflow composition patterns requirements. This adoption satisfies partially the requirement [REQ 06] because Linked-OWL does not enable all composition constructs.

Transactional patterns

The "Group" construct proposed by Linked-OWL enables the representation of groups of activities inside the full complex Linked-OWL service with decision direction for each group. The transaction commands "Commit" and "Rollback" are enabled inside each group decision direction in addition to more advanced error management. By default, each activity forms a separate group with a rollback decision on error occurrence. This satisfies the requirement [REQ 09].

Substitution management

The "Substitution" construct of Linked-OWL enables the definition of an activity or group of activities' substitutions in order to direct process execution on error occurrence or on user feedback during process execution.

Linked-OWL defines a queue of substitutions for each group of activities to enable multi-level substitution management which is mandatory in highly changed environments such as web. Linked-OWL assumes that an alternative path for each group of activities will come back to the original one whenever it is possible; this means that the substitution activities will not be executed unless their original activity faced a problem. This satisfies the requirement [REQ 10].

Linked-OWL dynamic composition

The four stages service composition approach provides high level of dynamicity through the multi-cycle candidate discovery strategy and the simplification process which is guided by user feedback. The multi-cycle process composition provides an advanced solution for [REQ 07] while the user feedback processing in the simplification engine provides a solution for [REQ 08].

The level of dynamicity is steadily increased with the volume of the semantic content published in the linked data sets and the interconnectivity between things in those data sets over the Web of Data.

Also this approach allows the user to monitor and control the process execution. The user can change any unexecuted part of the process; this change may include the substitution queue priorities change and the change of the entire path. This satisfies the requirement [REQ 11].

Conclusions

In this study, we analyzed the need of high dynamic workflow composition to serve the business process management in the Web environment. New service architecture "Linked-OWL" has been presented in the Linked Data paradigm, which enables high dynamic and accurate complex service composition taking into consideration most of the requirements related to linked data principles, workflows and dynamicity enabling in the Web environment. An adapted four stages approach for service composition has been adopted to serve along with the Linked-OWL services in the dynamic workflow enablement paradigm.

Linked-OWL forms a basis for Web of Services on top of Web of Data where services are linked to each other either functionally using the composition capabilities or descriptively using the underlying linked data sets. The Web of Services is formed of multiple service sets deployed and accessible through Linked-UDDI repositories. Linked-OWL also shifts the concepts from SOA and ROA into LOA (Linked Oriented Architecture) in which everything can be imagined are things in terms of linked data vocabulary. This linked understanding opens the road of linked data applications and services keeping high level of human-machine interaction.

In future, we plan to analyze the specification and principles of Web of Services and service sets to enable wide range of linked data applications, also to develop the architecture of Linked-UDDI to serve at the same level with Linked-OWL and to enable process composition patterns. On the other hand, we plan to develop and study algorithms that automatically analyze natural language texts to build RDF graphs and links with existing RDF data sets in order to provide more efficient user request simplification and to provide more accurate business process composition starting from natural language requests.

References

- Beco, S., Cantalupo, B., Matskanis, N., & Surridge, M. (2006). *Putting semantics in grid workflow management: The OWL-WS approach*. DATAMAT S.P.A, University of Southampton IT Innovation Centre. Retrieved September 15, 2012, from http://www.nextgrid.org/publications.html
- Berners-Lee, T. (2009). *Linked data - Design issues*. Retrieved September 15, 2012, from http://www.w3.org/DesignIssues/LinkedData.html
- Bizer, C., Heath, T., & Berners-Lee, T. (2009). Linked data - The story so far. *International Journal on Semantic Web and Information Systems*, 5(3), 1-22.
- Caeiro-Rodriguez, M., Priol, T., & Nemeth, Z. (2008). *Dynamicity in scientific workflows*. CoreGrid Technical Report TR-0162.
- Cardoso, J. (2006). *Semantic web services: Theory, tools, and applications, information science reference*. Hershey, New York.
- Gubala, T., & Hoheisel, A. (2007). Highly dynamic workflow orchestration for scientific applications. In *CoreGRID Integration Workshop 2006 (CIW06)*, pp. 309-320.
- Hausenblas, M. (2009). *Linked data application*. DERI Technical Report 2009-07-26.
- Klyne, G., & Carroll, J. (2004). *Resource Description Framework (RDF) Concepts and Abstract Syntax*. Retrieved September 15, 2012, from http://www.w3.org/TR/rdf-concepts/
- Martin, D. et al. (2004). *OWL-S: Semantic markup for web services*. Retrieved September 15, 2012, from http://www.w3.org/Submission/OWL-S/
- Norton, B., Krummenacher, R., & Marte, A. (2010a). Towards linked open services and processes. *Proceedings of the Third Future Internet Conference on Future Internet*, pp. 68-77, September 20-22, 2010, Berlin, Germany.
- Norton, B., Krummenacher, R., Marte, A., & Fensel, D. (2010b). Dynamic linked data via linked open services. In*Workshop on Linked Data in the Future Internet at the Future Internet Assembly*, pp. 1-10.
- Paik, I., Chen, W., & Komiya, R. (2011, July). A functional - scalable architecture for automatic service composition. In *Proceedings of the 7th IEEE World Congress on Services 2011(SERVICES2011)*, Washington, D.C., USA, 2011, (pp. 311-318).
- Prud'hommeaux, E., & Seaborne, A. (2008). *SPARQL query language for RDF*. Retrieved September 15, 2012, from http://www.w3.org/TR/rdf-sparql-query/
- Richardson, L., & Ruby, S. (2007). *RESTful web services*. O'Reilly Media Inc. United States of America.
- Speiser, S., & Harth, A. (2010). Towards linked data services. *The Semantic Web - Posters and Demonstrations (ISWC)*, Shanghai, China, 2010.
- van der Aalst, W.M.P., ter Hofstede, A. H.M., Kiepuszewski, B., & Barros, A. P. (2003). Workflow patterns. *Distributed and Parallel Databases*, 14(1), 5-51.
- Wohed, P., van der Aalst, W. M.P., Dumas, M., & ter Hofstede, A. H.M. (2002). *Pattern based analysis of BPEL4WS*. QUT Technical report, FIT-TR-2002-04, Queensland University of Technology, Brisbane. Retrieved September 15, 2012, from http://workflowpatterns.com/documentation/documents/qut_bpel_rep.pdf

Bibliographic information of this paper for citing:

Ahmad, Hussien, & Dowaji, Salah (2013). "Linked-OWL: A new approach for dynamic linked data service workflow composition." *Webology*, 10(1), Article 105. Available at: http://www.webology.org/2013/v10n1/a105.html