Linked Data Platform as a novel approach for Enterprise Application Integration

Nandana Mihindukulasooriya, Raúl García-Castro, and Miguel Esteban-Gutiérrez*

Center for Open Middleware, Ontology Engineering Group
Universidad Politécnica de Madrid, Spain
{nmihindu, rgarcia, mesteban}@fi.upm.es

Abstract. Enterprises are increasingly using a wide range of heterogeneous information systems for executing and governing their business activities. Even if the adoption of service orientation has improved loose coupling and reusability, applications are still isolated data silos requiring complex transformation and mediation for integrating them. The W3C Linked Data Platform (LDP) Working Group aims to standardize a RESTful way to read and write Linked Data. This opens the door for industry to benefit more effectively from Linked Data by building interoperable LDP applications and implementing new approaches for Enterprise Application Integration (EAI). For a wide industrial adoption of LDP-based EAI, the advantages of LDP have to be clearly illustrated and LDP needs to be assessed for enterprise readiness. This paper analyses LDP as an EAI approach by considering both its advantages over existing approaches and identifying the gaps and challenging enterprise EAI requirements it has to satisfy. The paper also presents lessons learned from a project that uses LDP for integrating open source Application Lifecycle Management tools.

Key words: Linked Data, Linked Data Platform, Enterprise Application Integration, Industry Adoption

1 Introduction

Enterprise information systems play a key role in providing a technology platform for executing the business processes of organizations. However, these systems have to be integrated in order to have a consistent unified view of the whole enterprise. Integrating heterogeneous applications, each with its own data model and business restrictions, is a complex problem and Enterprise Application Integration (EAI) has evolved over time taking different approaches to solve this issue, such as following a Service-Oriented approach, which is currently the most widely used approach in the industry [1].

* This research is supported by the ALM iStack project of the Center for Open Middleware.

The Linked Data[1] principles promote publishing data in a machine-readable manner using Web standards and interlinking them. On top of it, the Linked Data Platform[2] (LDP) specification defines a RESTful protocol for accessing read/write Linked Data. This protocol brings new opportunities to application integration at the data level which were not easily possible in previous EAI approaches. In order to ensure a wide adoption of LDP-based EAI, it is important to advocate the concrete benefits of Linked Data over existing approaches. Furthermore, it is also important to identify the gaps that need to be filled in order to make Linked Data-based approaches more promising to the industry and to provide an assessment of any potential risks.

As seen in previous approaches for EAI, there is a set of highly-demanding enterprise quality requirements that an EAI technology needs to satisfy to be widely accepted in the industry, such as security, transactions, performance, and interoperability. For example, starting from a few basic specifications, the Web Service standards stack gradually grew in numbers in order to satisfy these industrial requirements. Similarly, we can expect the Linked Data Platform specification to evolve and LDP middleware and tool support to become more available.

For building an industrial Linked Data-based EAI ecosystem, we need to answer a few concrete questions that early adopters raise:

- What are the concrete benefits that Linked Data-based EAI brings when compared to the current approaches followed in industry?
- Is this approach mature and enterprise ready? If not, which are the gaps?
- What are the lessons learned from the current projects using LDP for EAI?

This paper discusses these questions based on experiences derived from the ALM iStack project[3], which integrates Application Lifecycle Management tools by using the Linked Data Platform protocol and a common ontology. The paper is organized as follows. After an overview of EAI and existing approaches in Section 2, Section 3 introduces the Linked Data Platform protocol. Section 4 discusses the benefits of using LDP for EAI and the foreseen challenges. Section 5 presents a real world case study of using LDP for EAI and the lessons learned. Finally, Section 6 draws some conclusions.

2 Enterprise Application Integration

Business processes comprise a set of activities performed by several stakeholders of an organization in coordination with the organizational and technical environment to realize a business goal [2]. Business applications help organizations to carry out these processes more efficiently and generally consist of (1) business logic or rules of a domain, (2) data stores managing the application persistent state, and (3) a set of interfaces to interact with the applications (e.g., GUIs,

[1] http://www.w3.org/DesignIssues/LinkedData.html
[2] http://www.w3.org/TR/ldp/
[3] https://sites.google.com/a/centeropenmiddleware.com/alm-istack/

services, APIs). Since business processes spread across several applications, there is a high demand for integrating applications without making significant changes to those applications or their underlying data models [3].

Enterprise Application Integration (EAI), defined as "*the unrestricted sharing of data and processing among any connected applications and data sources in the enterprise*" [4], aims to solve this problem by combining the technologies and processes that enable business applications to exchange business level information in formats and contexts that each understand [5]. However, often these applications are backed by relational databases (or recently NoSQL databases) and act as isolated data silos. This forces duplication of data and hinders interlinking and the possibility of traversing through different applications that manage the different aspects of a business process.

EAI can be viewed in three main levels: (i) *physical integration* (interconnection of devices via computer networks), (ii) *application integration* (integration of software applications and database systems), and (iii) *business integration* (coordination of functions that manage, control and monitor business processes) [6]. EAI can be designed using several topologies including point-to-point, message broker, or service bus and can be implemented through different approaches: shared database, file transfer, remote procedure calls, or message bus.

Service-Oriented Architecture (SOA), which is currently the most popular approach for EAI in the industry [1, 7], is an architectural style for building enterprise applications using independent business-aligned services that can be combined into agile and flexible business processes [8]. Services are well-defined, self-describing, and platform-agnostic computational elements that provide discrete units of business functionality through a service contract and support a rapid and low-cost composition of distributed applications [8, 9]. The standard and discoverable interfaces provided by services help overcoming the application connectivity challenge of EAI; however, the information integration challenges require handling complex transformations between the different domain models used by the applications. Furthermore, applications still act as data silos that are not linked but duplicated in many applications. Despite Service-Oriented systems foster application communication and interoperability, they do not address the semantic interoperability problem [6].

As an attempt to solve this, Semantic Web Services (SWS) try to combine Web Services with Semantic Web technologies by annotating services with semantic markup. Nevertheless, prominent SWS approaches like OWL-S [10], WSMO [11], and SA-WSDL[4] are still grounded on heavyweight XML-based Web Service standards like SOAP, WSDL and XML Schema.

Representational State Transfer (REST) is an architectural style that is becoming popular in the industry and that defines a set of constraints for designing a hypermedia system: (i) *resource identification*, (ii) *uniform interfaces*, (iii) *self-descriptive messages*, (iv) *hypermedia as the engine of application state* (HATEOAS), and (v) *stateless interactions* [12]. These principles allow building scalable applications that can discover links to other resources at runtime us-

[4] `http://www.w3.org/TR/sawsdl/`

ing identifiers within a resource representation and interact with these resources through uniform interfaces without having previous knowledge [13]. Properly using the Linked Data and REST principles by design solves the data silos problem by making data global and allowing the development of interlinked applications.

3 Linked Data Platform

RDF provides a simple and flexible data model that is well-suited for data integration and the conceptualization of domain models can be expressed in terms of RDF Schema and OWL ontologies. The Linked Data principles help creating a global data space [14] with typed links between data from different sources [15], hence breaking isolated data silos. Machine-readable structured data with explicit formal semantics that are expressed using standards makes merging, integrating, processing, and analyzing data possible without needing out-of-band knowledge or proprietary tools. Links to related entities in data make it possible to start from a piece of data and traverse through different sources with a follow-your-nose approach[5] in order to discover more entities and get context information.

The LDP protocol brings benefits from both the REST and Linked Data worlds to application integration. The LDP specification provides a set of best practices and a simple approach for a read-write Linked Data architecture, based on HTTP access to web resources that describe their state using the RDF data model. LDP introduces two main building blocks for building Linked Data applications: *LDP Resources (LDPR)* and *LDP Containers (LDPC)*. LDPRs are HTTP resources whose state is represented in RDF and that can be retrieved, updated, and deleted using HTTP methods (adhering to the constraints enforced by the LDP specification). LDPCs are specialization of LDPR which helps organizing other resources (i.e., LDPRs) as its members. LDPC serves two main purposes: enumeration of its member resources and creation of new member resources. In addition, the LDP specification provides a standard way for paginating large RDF resources and ordering triples inside LDP containers.

Whether RESTful applications are enterprise ready, i.e., they can fulfill the quality requirements of EAI such as advanced security scenarios and handling business transactions [13], is one of the concerns of LDP early adopters. As most enterprise applications operate on a controlled environment where closed world assumptions apply, whether Semantic Web languages such as OWL and RDF Schema (which operate under an open world assumption) are still suitable for data validation and ensuring data quality is another concern.

Furthermore, regarding Linked Data there is a major misconception about the fact that data should always be publicly available under an open license. Though this is true for Linked Open Data, it is possible to keep the data private as Linked Closed Data [16] or Linked Enterprise Data [17] with limited access within intranets protected by firewalls similar to most Enterprise Information Systems in use today in industry.

[5] `http://patterns.dataincubator.org/book/follow-your-nose.html`

4 Integrating Enterprise Applications using LDP

LDP enables a novel approach for integrating applications. Applications that support the LDP protocol can expose all or part of their data using one or more vocabularies and can consume Linked Data from other applications. In contrast to traditional applications, the data that LDP applications expose and consume can have links to data in other applications. Thus, these applications are capable of crawling Linked Data, traversing through data according to their business needs, and interacting with applications that expose those data using the LDP protocol.

The first logical step towards adopting the LDP protocol in an industrial ecosystem is to provide LDP interfaces to existing applications. There are two approaches, shown in Figure 1, for achieving this: (a) to provide native support for LDP by modifying the application or (b) to provide LDP support via an adapter or a wrapper. On the one hand, providing native support will be more preferable to take full advantage of the Linked Data principles by designing applications that can benefit of the follow-your-nose approach of Linked Data. On the other hand, providing an adapter would be a more feasible adoption path in the beginning as it does not require any changes to existing applications. In this case, however, protocol conversion might introduce some overhead and the application business logic may not be aware of the links in the data.

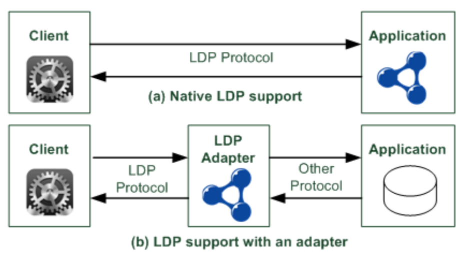

Fig. 1. (a) Native LDP support vs (b) LDP adapters.

4.1 LDP Applications in the Context of EAI

EAI involves integration at different levels: *physical integration*, *application integration*, and *business integration* [6].

Physical integration of LDP applications uses the de-facto Internet protocol suite infrastructure (TCP/IP) similar to traditional Web Service-based EAI.

Application integration, where most benefits from LDP are visible, can again be divided into two different levels: *application connectivity (interfaces)* and *data integration*. Even if both Web Services and LDP use the HTTP protocol to connect applications, their use of it is different: while SOAP Web Services use

HTTP as a tunneling protocol with protocol layering [18], LDP uses HTTP as an application protocol by using dereferenceable URIs for identifying resources and HTTP headers to convey the interaction semantics, enabling applications to discover affordances on resources only by using the HTTP protocol without requiring any additional heavyweight protocol.

In data integration, there are three heterogeneity problems to be solved: *syntactic heterogeneity*, *structural heterogeneity*, and *semantic heterogeneity* [19]. The usage of standardized data exchange formats solves the syntactic heterogeneity problem for both traditional and LDP-based EAI. The graph-based flexible RDF data model obsoletes the structural heterogeneity problem and makes integration from multiple data sources possible even if their schemas differ or are unknown. Traditional EAI based on hierarchal XML data types with strict structural schemas requires complex schema transformations. Ontology-based data integration approaches can be used to overcome the semantic heterogeneity problem either by using a global ontology, multiple ontologies, or a hybrid approach [20]. By following the links created according to the Linked Data principles, new ontologies can be discovered which can be mapped to the ones that the application is aware of with the help of ontology alignment techniques.

Approaches for business integration at the process level depend on how much each application is aware of the whole business process and on who is driving the process. *Orchestration* and *choreography* are two ways in which business processes can be described and implemented [21]. In orchestration, the business process is controlled and driven by an orchestrator in a centralized manner and in choreography interactions are implemented as a distributed collaboration between applications where each application is aware of its part of the whole process. LDP applications can support both approaches. Further, LDP applications can use the fourth Linked Data rule (i.e., include links to other dereferenceable URIs so that they can discover more things) and the HATEOAS REST principle to discover new applications without any previous knowledge about them and to communicate with them using the LDP protocol, thus enabling agile business processes. This is a major advantage over the existing approaches.

4.2 Challenging EAI Requirements

In addition to being able to integrate heterogeneous applications, EAI imposes several requirements that have to be fulfilled for an approach to be adopted and used in production. This section looks at some challenging requirements for LDP-based EAI and their current state of the practice in RESTful and Linked Data applications in the industry.

Data validation is a vital step for ensuring the quality of data in applications and expressive schema languages and related tools are essential for effective data validation. Both relational databases and XML have expressive schema languages for defining the structure and the constraints on data. In RDF, which is built upon the Open World Assumption and the Non-unique Name Assumption, data validation becomes a challenge as the languages currently used to describe these constraints (i.e., RDF Schema and OWL) are more suited for inferring than for data validation and using them for validation could lead to unexpected results [22]. Most applications require some validation to be done under a (local) closed world assumption; the work done for tackling this problem includes adding integrity constraints to OWL [23], closed world reasoning [24], SPARQL and SPIN-based solutions [25], and Resource Shapes [22]. There is not a standard for RDF data validation, though we can see a recent movement[6] in that direction.

There are several security requirements including authentication, authorization, integrity, confidentiality, and non-repudiation. A few authentication protocols are widely used by RESTful web applications: HTTP Basic Authentication and Digest Access Authentication[7], TLS[8], and SSL[9]. Recently, a set of user-centric decentralized URI-based identity systems became popular, including OpenID[10], BrowserID[11], and WebID [26], and fit well with Linked Data approaches. Regarding authorization, OAuth 2.0[12] is a widely used authorization framework for web applications and Web Access Control[13] (WAC) is a decentralized system for authorizing users and groups where users are identified by WebIDs and groups are identified by HTTP URIs. The S4AC [27], PPO [28], RelBAC [29], and AMO [30] ontologies define fine-grained models for defining and implementing access control for Linked Data. Though there is no standard security stack for LDP applications, like in the case of the WS-Sec* stack [31], we can see emerging technologies like WebID and WAC that have the potential of becoming W3C recommendations. However, there is work done on other security requirements, such as digitally signing and encrypting RDF data [32].

Usually, a business process is composed of several business transactions [33], i.e., consistent changes from one state of the business to another state that are driven by a well-defined business function. These transactions need to support ACID (Atomicity, Consistency, Isolation, Durability) to ensure that a system is always at a consistent state. With REST gaining traction in the industry, several RESTful transaction models have been proposed in the last few years [34] including the Try-Cancel/Confirm pattern [35], action resources that expose workflow-related operations on the parent resource [36], lockable resources [34, 37, 38], or Optimistic Concurrency Control [39]. In order to have interoperability at industrial level, one of these has to become a standard.

However, the usage of a strong consistency model introduces other problems. As the CAP (or Brewer's) theorem states, a distributed application cannot provide the following three guarantees simultaneously: (1) *consistency*, all nodes see the same data at the same time; (2) *availability*, every request receives a response about whether it was successful or failed; and (3) *partition tolerance*,

[6] `http://www.w3.org/2012/12/rdf-val/`
[7] `http://tools.ietf.org/html/rfc2617`
[8] `http://tools.ietf.org/html/rfc5246`
[9] `http://tools.ietf.org/html/rfc6101`
[10] `http://openid.net/specs/openid-authentication-2_0.html`
[11] `http://persona.org/`
[12] `http://tools.ietf.org/html/rfc6749`
[13] `http://www.w3.org/wiki/WebAccessControl`

the system continues to operate despite arbitrary message loss or failure of part of the system [40, 41].

There exists a trend for using weaker consistency models in distributed applications so that high-availability and partition tolerance can be guaranteed. A popular model is that of *eventual consistency*, which ensures that if no new updates are made to a given data item, eventually all read accesses to that item will return the last updated value (the system *converges*) [42]. Distributed systems based on this model provide BASE guarantees (Basically Available, Soft-state with Eventual-consistency) in contrast to the ACID guarantees provided by traditional transaction-based applications [43].

In addition to these requirements, there are several others that have to be supported including performance requirements (throughput or response time), discovery requirements (finding vocabularies and data model restrictions), reliability requirements (reliably do operations on an unreliable infrastructure including network failures, unavailability of the destination system and other possible error conditions), and contract requirements (expressing quality agreements such as SLA contracts [44]). Further, unlike in the Web, enterprises may want to ensure data consistency and link maintenance [15], i.e., there are no dead links or dangling pointers when resources are deleted or moved and links always point to the correct resources, which is a challenge which may require notifications or eventing.

5 Application Lifecycle Management with Linked Data

This section briefly describes a case study of using the LDP protocol for building a proof-of-concept that integrates the Application Lifecycle Management (ALM) tools used by a software development organization.

The software development process involves a set of activities including project planning, requirements gathering, software design, software development, testing and quality assurance, deployment, support, etc. Each of these activities produce and consume different sets of artifacts like project plans, software requirement specifications, architecture documents, source code, or test cases and involves possibly different teams carrying out these activities using different tools. Collective management, coordination and governance of all these activities and artifacts is called ALM [45].

Successful ALM requires the horizontal integration of heterogeneous information systems that track the different activities of the process in order to have an overall view of such process. This becomes a complex integration problem because it involves a set of heterogeneous tools coming from different vendors and open source communities and using different technologies. As a concrete example, a software development team using open source tools may use Eclipse for software development, Bugzilla for issue tracking, TestLink for test case management, and so on.

The ALM iStack project is developing LDP middleware and a proof-of-concept application to demonstrate how open source ALM tools can be seam-

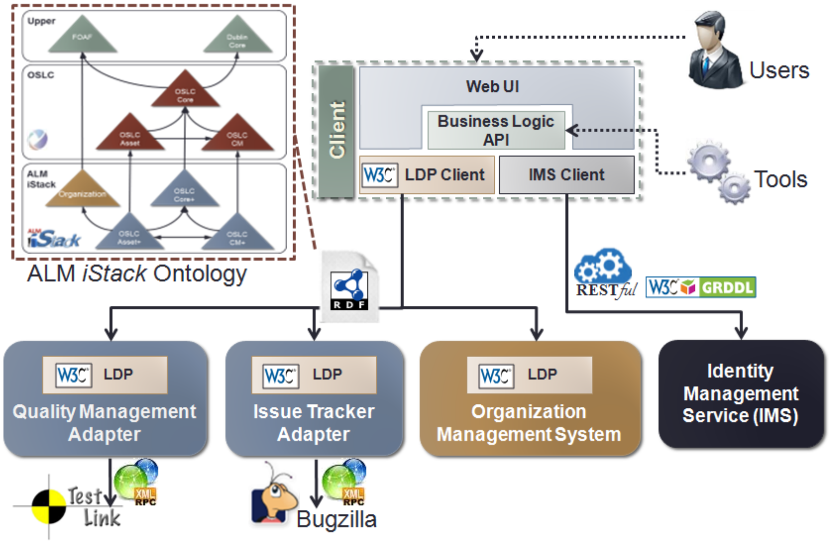

Fig. 2. Architecture of ALM iStack Proof-of-Concept

lessly integrated using the Linked Data Platform protocol and a common vocabulary. The project uses a hybrid integration approach [20] with an enhanced version of the vocabularies defined by Open Services for Lifecycle Collaboration[14] (OSLC), an open community that has the goal of building specifications for the integration of ALM software following the Linked Data principles. The strategy followed is to develop LDP adapters for each ALM tool and to integrate them using an LDP client that acts as an orchestrator of the process (Figure 2).

Adapters expose the existing application data as Linked Data by minting dereferenceable HTTP URIs for those entities. There has been much work done in exposing legacy data as Linked Data and specifications like W3C R2RML[15] provide guidance for this. To increase performance and reduce HTTP traffic and application response times, the adapters are designed to expose LDP resources with proper granularity and to use composite resources when appropriate.

The developed adapters expose data using a shared ALM ontology. The mismatch between the application data model and this ontology is handled by the adapters by persisting the extra information they require to handle the mismatch. In addition, adapters use a SPARQL-based approach to verify integrity constraints and validate data.

The security of the adapter is decoupled from the security mechanism used by the application using the trusted subsystem security pattern[16]; thus, adapters are free to use its own security approach.

[14] http://open-services.net/
[15] http://www.w3.org/TR/r2rml/
[16] http://msdn.microsoft.com/en-us/library/aa905320.aspx

For managing coreferences of entities residing in different applications, ALM iStack includes a coreference service (i.e., Identity Management Service) [46]. For the time being, no support for transactions or reliable messaging is provided.

6 Conclusions

Linked Data-based EAI has several advantages over current EAI approaches when it comes to application and data integration. To summarize, (a) Linked Data allows having global identifiers for data that can be accessed using the Web infrastructure and typed links between data possibly from different applications (b) the graph-based RDF data model allows consuming and merging data without having to do complex structural transformations, and (c) application-specific domain conceptualizations expressed in terms of RDF Schema or OWL ontologies can be aligned and mapped to other applications using knowledge representation techniques much easier than in traditional approaches.

The Linked Data Platform protocol provides a standard uniform interface for managing these data. Having a standard for building interoperable Linked Data applications is a big step towards the industrial adoption of Linked Data as an application integration approach. LDP allows breaking data silos in applications and helps building integrated applications that can link to data in related applications using standards, enabling clients to discover and interact with those applications in a scalable manner.

However, there is still some work to be done on fulfilling the quality requirements imposed by the industry to make the approach enterprise ready and exploit the full potential of LDP such as supporting advanced security scenarios, transactions, data consistency, link maintenance, notifications, eventing, data validation, and discovery of vocabularies and data model restrictions. At the moment, some of these requirements are fulfilled by LDP applications using their own proprietary mechanisms but having interoperable standard mechanisms will help LDP to be more widely adapted as a novel approach for EAI.

References

1. Linthicum, D.S.: Next Generation Application Integration: From Simple Information to Web Services. Addison-Wesley Professional (2004)
2. Weske, M.: Business Process Management: Concepts, Languages, Architectures. Springer (2012)
3. Sharif, A.M., Elliman, T., Love, P.E., Badii, A.: Integrating the IS with the enterprise: key EAI research challenges. Journal of Enterprise Information Management **17**(2) (2004) 164–170
4. Linthicum, D.S.: Enterprise Application Integration. Addison-Wesley Professional (2000)
5. Ring, K., Ward-Dutton, N.: Enterprise Application Integration. Making the Right Connections. Ovum Ltd, London, UK (1999)
6. Chen, D., Doumeingts, G., Vernadat, F.: Architectures for enterprise integration and interoperability: Past, present and future. Computers in industry **59**(7) (2008) 647–659

7. Vernadat, F.B.: Enterprise Modelling and Integration. Springer (2003)
8. Rosen, M., Lublinsky, B., Smith, K.T., Balcer, M.J.: Applied SOA: Service-Oriented Architecture and Design Strategies. Wiley (2008)
9. Papazoglou, M.P., Van Den Heuvel, W.J.: Service oriented architectures: approaches, technologies and research issues. The International Journal on Very Large Data Bases **16**(3) (2007) 389–415
10. Martin, D., Burstein, M., Hobbs, J., Lassila, O., McDermott, D., McIlraith, S., Narayanan, S., Paolucci, M., Parsia, B., Payne, T., et al.: OWL-S: Semantic Markup for Web Services. W3C member submission **22** (2004) 2007–04
11. Fensel, D., Lausen, H., Bruijn, J., Polleres, A.: Enabling Semantic Web Services. Springer (2007)
12. Fielding, R.T.: Architectural Styles and the Design of Network-based Software Architectures. PhD thesis, University of California (2000)
13. Wilde, E., Pautasso, C.: REST: From Research to Practice. Springer Verlag (2011)
14. Heath, T., Bizer, C.: Linked Data: Evolving the Web into a Global Data Space. Synthesis lectures on the semantic web: theory and technology **1**(1) (2011) 1–136
15. Bizer, C., Heath, T., Berners-Lee, T.: Linked Data - The Story So Far. International Journal on Semantic Web and Information Systems (IJSWIS) **5**(3) (2009) 1–22
16. Cobden, M., Black, J., Gibbins, N., Carr, L., Shadbolt, N.: A Research Agenda for Linked Closed Data. In: Proceedings of Second International Workshop on Consuming Linked Data (COLD2011). (2011)
17. Wood, D.: Linking Enterprise Data. Springer (2010)
18. Pautasso, C., Zimmermann, O., Leymann, F.: RESTful Web Services vs. "Big" Web Services: Making the Right Architectural Decision. In: Proceedings of the 17th international conference on World Wide Web, ACM (2008) 805–814
19. Gagnon, M.: Ontology-based integration of data sources. In: Proceedings of 10th International Conference on 10th International Conference on Information Fusion (FUSION2007), IEEE (2007) 1–8
20. Wache, H., Voegele, T., Visser, U., Stuckenschmidt, H., Schuster, G., Neumann, H., Hübner, S.: Ontology-Based Integration of Information - A Survey of Existing Approaches. In: Proceedings of Workshop on Ontologies and Information Sharing (IJCAI2001). (2001) 108–117
21. Pahl, C., Zhu, Y.: A Semantical Framework for the Orchestration and Choreography of Web Services. Electronic Notes in Theoretical Computer Science **151**(2) (2006) 3–18
22. Ryman, A.G., Le Hors, A.J., Speicher, S.: OSLC Resource Shape: A language for defining constraints on Linked Data. In: Proceedings of the 6th Workshop on Linked Data on the Web (LDOW2013). (2013)
23. Tao, J., Sirin, E., Bao, J., McGuinness, D.L.: Integrity Constraints in OWL. In: Proceedings of the 24th AAAI Conference on Artificial Intelligence (AAAI-10). (2010)
24. Knorr, M., Alferes, J.J., Hitzler, P.: Local Closed World Reasoning with Description Logics under the Well-Founded Semantics. Artificial Intelligence **175**(9) (2011) 1528–1554
25. Fürber, C., Hepp, M.: Using SPARQL and SPIN for Data Quality Management on the Semantic Web. In: Business Information Systems, Springer (2010) 35–46
26. Story, H., Harbulot, B., Jacobi, I., Jones, M.: FOAF+SSL: RESTful Authentication for the Social Web. In: Proceedings of the First Workshop on Trust and Privacy on the Social and Semantic Web (SPOT2009). (2009)

27. Villata, S., Delaforge, N., Gandon, F., Gyrard, A.: An Access Control Model for Linked Data. In: On the Move to Meaningful Internet Systems: OTM 2011 Workshops, Springer (2011) 454–463
28. Sacco, O., Passant, A.: A Privacy Preference Ontology (PPO) for Linked Data. In: Proceedings of the Linked Data on the Web Workshop (LDOW2011). (2011)
29. Giunchiglia, F., Zhang, R., Crispo, B.: Ontology Driven Community Access Control. In: Proceedings of the 1st Workshop on Trust and Privacy on the Social and Semantic Web (SPOT2009). (2009)
30. Buffa, M., Faron-Zucker, C.: Ontology-Based Access Rights Management. In: Advances in Knowledge Discovery and Management. Springer (2012) 49–61
31. Adamczyk, P., Smith, P.H., Johnson, R.E., Hafiz, M.: REST and Web Services: In Theory and in Practice. In: REST: From Research to Practice. Springer New York (2011) 35–57
32. Giereth, M.: On Partial Encryption of RDF-Graphs. In Gil, Y., Motta, E., Benjamins, V.R., Musen, M.A., eds.: The Semantic Web - ISWC 2005. Volume 3729 of Lecture Notes in Computer Science. Springer Berlin Heidelberg (2005) 308– 322
33. Papazoglou, M.P.: Web Services and Business Transactions. World Wide Web: Internet and Web Information Systems **6**(1) (2003) 49–91
34. Razavi, A., Marinos, A., Moschoyiannis, S., Krause, P.: RESTful Transactions Supported by the Isolation Theorems. In: Web Engineering. Springer (2009) 394–409
35. Pardon, G., Pautasso, C.: Towards Distributed Atomic Transactions over RESTful Services. In: REST: From Research to Practice. Springer New York (2011) 507–524
36. Hadley, M., Pericas-Geertsen, S., Sandoz, P.: Exploring hypermedia support in Jersey. In: Proceedings of the First International Workshop on RESTful Design (WS-REST 2010), ACM, ACM (2010) 10–14
37. Richardson, L., Ruby, S.: RESTful Web Services. O'Reilly (2008)
38. Khare, R., Taylor, R.N.: Extending the Representational State Transfer (REST) Architectural Style for Decentralized Systems. In: Proceedings of the 26th International Conference on Software Engineering (ICSE), IEEE (2004) 428–437
39. da Silva Maciel, L.A.H., Hirata, C.M.: An optimistic technique for transactions control using REST architectural style. In: Proceedings of the 2009 ACM symposium on Applied Computing, ACM (2009) 664–669
40. Brewer, E.A.: Towards robust distributed systems. In: Proceedings of the nineteenth annual ACM symposium on principles of distributed computing. PODC2000, New York, NY, USA, ACM (2000) 7
41. Gilbert, S., Lynch, N.: Brewer's conjecture and the feasibility of consistent, available, partition-tolerant web services. SIGACT News **33**(2) (2002) 51–59
42. Vogels, W.: Eventually consistent. Commun. ACM **52**(1) (2009) 40–44
43. Pritchett, D.: BASE: An Acid Alternative. Queue **6**(3) (2008) 48–55
44. Kübert, R., Katsaros, G., Wang, T.: A RESTful implementation of the WS-Agreement specification. In: Proceedings of the Second International Workshop on RESTful Design. WS-REST '11, ACM, ACM (2011) 67–72
45. Kääriäinen, J., Välimäki, A.: Impact of Application Lifecycle Management - A Case Study. In: Enterprise Interoperability III. Springer (2008) 55–67
46. Esteban-Gutiérrez, M., García-Castro, R., Mihindukulasooriya, N.: A Coreference Service for Enterprise Application Integration using Linked Data. In: 7th International Workshop on Applications of Semantic Technologies (AST 2013), Koblenz, Germany (2013)