An Approach to Support Data Integrity for Web Services Using Semantic RESTful Interfaces

Hermano Albuquerque Lira
SERPRO
Av. Pontes Vieira, 832,
CEP 60.130-240
Fortaleza, CE, Brasil
hermano.lira@serpro.gov.br

José Renato Villela Dantas
SERPRO
Av. Pontes Vieira, 832,
CEP 60.130-240
Fortaleza, CE, Brasil
jose.dantas@serpro.gov.br

Bruno de Azevedo Muniz
University of Fortaleza
Av. Washington Soares, 1321
J-30, CEP 60.811-341
Fortaleza, CE, Brasil
brunoamuniz@gmail.com

Tadeu Matos Nunes
University of Fortaleza
Av. Washington Soares, 1321
J-30, CEP 60.811-341
Fortaleza, CE, Brasil
tadeu.matos@gmail.com

Pedro Porfírio Muniz Farias
University of Fortaleza
Av. Washington Soares, 1321
J-30, CEP 60.811-341
Fortaleza, CE, Brasil
porfirio@unifor.br

ABSTRACT

In the Web, linked data is growing rapidly due to its potential to facilitate data retrieval and data integration. At the same time, relational database systems store a vast amount of data published on the Web. Current linked data in the Web is mainly read only. It allows for integration, navigation, and consultations in large structured datasets, but it still lacks a general concept for reading and writing. This paper proposes a specification, entitled Semantic Data Services (SDS), for RESTful Web services that provide data access. To provide linked data read-write capacity, SDS proposes a mechanism for integrity checking that is analogous to that used in relational databases. SDS implements the Semantic Restful Interface (SERIN) specification. SERIN uses annotations on classes in an ontology, describing the semantic web services that are available to manipulate data. This work extends SERIN specification adding annotations to allow the adoption of data access integrity constraints.

Categories and Subject Descriptors

D.2.12 [**SOFTWARE ENGINEERING**]: Interoperability—*Interface definition languages*

Keywords

semantic web; linked data; RESTful semantic web service; semantic data service; SERIN

1. INTRODUCTION

In the Web, linked data is growing rapidly due to its potential to facilitate the data retrieval and the data integration. At the moment, linked data is mostly read-only in the Web[3]. While linked data allows integration, navigation, and large structured dataset query, it still lacks a general concept for data reading and writing. Garrote[6] states that the lack of support for data writing to the linked data repositories imposes barriers to the adoption of this technology.

At the same time, relational database systems store a vast amount of structured relational data. To make this huge amount of relational data available to the Web of data, it is necessary to develop mechanisms to facilitate the exchange between relational databases (RDB) and the Web of Data.

In this scenario, several works provides APIs or systems to map linked data formats to relational model formats. Mainly these APIs provides read access to data, stored in relational database, in order to publish them on the Web. Few works describe read-write implementations with ability to update information from Web in relational databases.

Part of the difficulty in integrating the Web of data with relational databases is due to the use of differing assumptions. The Web of data languages, like OWL and RDF, adopts the Open World Assumption (OWA) and the "not unique" designators. OWA makes the Web of data more open with less integrity constraints. On the other hand, relational databases adopts the Closed World Assumption (CWA) and the unique designators. These assumptions make relational database subject to strict rules of data integrity constraints.

In order to motivate our work, we imagine a scenario where several software agents, acting as clients, desire to insert or update data in a server database. These agents are external to the server. All of them, clients and server, exist on the Web. Lots of systems may fit this scenario as, for example:

- On a e-commerce scenario where a client agent have the task to buy a product with minimal human intervention. The agent must be able to identify which servers have the desired product, and to insert an order in the chosen server database;

- On a government acquisition scenario, suppliers interested in selling their product automatically insert their proposals to the government base;
- A project manager needs to organize his team agendas. For this, he wants a software agent with ability to identify available dates for all team components and insert/update some schedules.

On all these insert/update scenarios, it is desirable that the clients sends the data according to the data model adopted by the server. Moreover it is important that the data attends to some integrity constraints considering that the database follows a closed world assumption.

Considering the difficulties to mix both models, Web of data and relational databases, this paper proposes a specification, entitled Semantic Data Services (SDS), for building RESTful Web services to provide data access. SDS adopts the local closed world assumption[10, 11] to overcome the problem about differences between OWA and CWA. To provide a capacity for reading and for writing linked data, SDS proposes a mechanism for checking integrity constraints. This mechanism is analogous to those ones used in relational databases.

Semantic data services are concrete implementations that follow the abstract SERIN semantic interfaces specification. SERIN (Semantic Restful Interfaces)[14] is an annotated ontology whose classes and properties describe resources available in the Web. SERIN formally describes semantic Web services that provide such resources. It makes use of annotations in ontology concepts (classes and properties). The annotations specify the semantic web services that are available to manipulate any data resource in a data provider.

The present work proposes the inclusion of new annotations in SERIN specification to enable the adoption of integrity constraints when accessing such data. These annotations formally describe which constraints an agent must deal with when it sends a request to manipulate data. In turn, a data provider that follows SERIN specification may implements constraint checks in the services they provide.

This article is structured as follow: Section 2 presents our motivation and the background for this work. Section 3 presents the proposal of semantic data services (SDS). Section 4 outlines the constraint mechanism of data integrity services developed for semantic data. Section 5 presents few example scenarios where it is possible to apply the integrity constraints. At least, section 6 gather the conclusions and proposals for future studies.

2. BACKGROUND AND MOTIVATION

The main motivation for our work is the necessity to provide resources for a read-write linked data on the Web. Our focus is to define integrity constraints treatment for linked data access. A motivation scenario considers the existence of a client software agent that is responsible to execute some job. Some tasks, to get the job done, must insert or update data in a external host database on the Web. To ensure that the information is inserted in a useful and reliable way, the host must check some constraints.

One difficulty in integrating Web of data with relational databases is due to the use of differing assumptions. The Web of Data utilizes the open-world assumption (OWA) while relational databases uses the closed-world assumption (CWA) subordinated to strict rules of integrity constraints.

Next subsections, we discuss this two aspects, i.e., read-write linked data and the differences between OWA and CWA.

2.1 Read-write linked data

Linked Data [7] refers a set of interrelated datasets on the Web that is available in a standard format, managed by semantic tools. Four basic principles summarize these practices [2]:

1. Use URIs (Universal Resource Identification) as names for resources;
2. Use HTTP URIs so anyone can look up those names;
3. Provide useful information, using standards, when someone look up a URI;
4. Using links to other URIs so it is possible to find more resources.

The adoption of linked data has advantages for data integration and data retrieval on the Web. A dataset can integrate data from different hosts using RDF and semantic mechanisms. In this sense, one can imagine the Web of data as a large database [7].

Current linked data on the Web is mainly a read-only data. There is still the need for a general concept for reading and writing on the Web of Data. The lack of support for writing to the linked data repositories imposes barriers to the adoption of this technology.

Some proposals implement mechanisms to allow linked data reading and writing. Few good examples are Linked Data Basic Profile, Linked Data Platform, and oData. Nelly et al. [16] proposed Linked Data Basic Profile 1.0. This specification adopts the four principles for linked data. It extends them to be compatible with the REST architectural style. For its turn, W3C created Linked Data Platform[18] as a standard to define a RESTful way to read and write linked data. At last, Kirchhoff and Geihs proposes oData [9] as a data access protocol based on REST principles.

Despite all these services works fine in supply read-write access to RDF and linked data, they still lack in provide integrity constraints verification. In fact oData provides integrity constraints checking but it does not necessarily adopts an ontology to describe the data schema.

2.2 Local Closed World Assumption

In the closed world assumption, typically found in relational database systems, we consider that one single host contains all data, even if it adopts a distributed scheme. A Data Definition Language (DDL) syntactically describes the data schema. As DBAs and programmers are human agents and non-software agent it is unnecessary to use a formal semantic description of the data. System documentation semantically describes the data schema for human readers.

In this context, an unique designator identifies each record. Different designators are semantically mapped into different objects in the real world. This architecture adopts the close world assumption. This assumption considers that all objects in the domain are known. It allows users to explicitly represent only positive facts. Information is considered false if it is not available in the database and if the facts available can not prove it [4].

Open world assumption states that the datasets that are not present in a knowledge base and that can not be deducted as a logical consequence of the present data are not considered false, but unknown. This assumption matches the semantics traditionally used in first-order logic and the open architecture typically used in the Web Linked Data Web [7].

Ontologies normally do not work as database schema because ontologies intend to be consistent but not complete. They specify classes, possibly incomplete, but may also contain instances. Thus, there is no clear distinction between the data schema, or metadata, and the data itself. The main focus of ontologies is to maintain logical consistency that allows inferences.

These characteristics lead to significant challenges for data integration existing in more than one host. Some systems try to bypass the data integration problem with a mapping among different ontologies used by different hosts. These mappings are usually partial and inaccurate.

To overcome the differences between OWA and CWA, our work adopts the Local Closed World Assumption (LCWA) [11]. It meets our desirable open data scenario with several hosts that operate their data in a closed world. Each host has its own particular dataset that can be queried to form a broader set of information.

In this scenario, we consider a composite network where several hosts offer RESTful web services for data access and a set of clients that can manipulate these resources. The same semantic interface describes these features. Figure 1 illustrates this scenario. It shows three servers, each one with its own data following the same semantic classes defined in the interface.

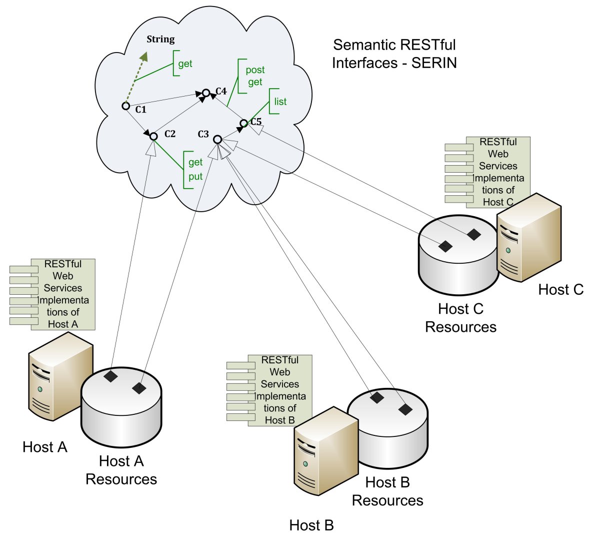
Semantic RESTful Interfaces - SERIN
String
get
post
get
list
C1
C2
C3
C4
C5
get
put
RESTful Web Services Implementations of Host C
Host C
Host C Resources
RESTful Web Services Implementations of Host A
Host A
Host A Resources
RESTful Web Services Implementations of Host B
Host B Resources
Host B

Figure 1: Semantic Interfaces approach

We assume that each host has a complete knowledge about its own resources. Locally, for each host, it is a closed world assumption, i.e., the resources that are not present for that host are allegedly false. On the other hand, for clients who access the resources, the world is considered open because there will always exist hosts that are not known for a particular client.

3. SEMANTIC DATA SERVICES

The present work defines the term “Data Services” as access points to data sources through Web services. The main characteristic of data services is the data manipulation through CRUD (Create, Read, Update and Delete) operations. Besides these, the data services does not perform any other data computing operation. These services act as data access layers, allowing abstracting data sources maintained by servers. Thus, an external agent can call these services to perform operations over their data.

In this work, we propose the adoption of Semantic Data Services (SDS). SDS are RESTful [5] web services endpoints to access linked data. They are described by SERIN[15] semantic interfaces.

A Semantic RESTful Interface (SERIN) is an annotated ontology, written in OWL[1]. Its classes and properties semantically describe resources and services available on a host. Four annotation types can be added to the ontology: GET, POST, PUT, and DELETE. These annotations indicate the RESTful Web services that a host must implement for each resource. A resource from a class with four annotations must thus implement a Web service to perform an operation for each annotation.

SERIN interfaces have four essentials characteristics which are the basis to SDS:

- Use of RDF (Resource Description Framework) [12] instances - SDS and its clients exchange these instances through a valid RDF document containing one or more instances.
- Use of annotations for data manipulation - A set of annotations, applied to the classes defined in SERIN interface, defines the operations for insertion, update and removal of instances on a semantic data service. There are four types of annotations: GET, POST, PUT, and DELETE, mapped in the four corresponding HTTP verbs.
- Addressing convention - URIs follow a pattern that can represent three types of resources: a instance, a set of instances, or a property value.
- Separation between shared knowledge (abstract semantic interface) and particular instances within each host - The SERIN interfaces defines classes and instances. They are global constants, whose semantics is shared with hosts and clients. Though each host has its own particular resource base, which are instances of the interface classes. Any client that adopts SERIN interfaces may change these resources.

SERIN addressing convention defines a pattern to Web services URLs. This pattern divides each URL into two parts. The first part identifies the host. The second part identifies the resource that the semantic interface describes. The basic format for a SERIN address is

```
http://{host URL}/{interface URI}/
       {class name}/{resource URI}
```

where:

- {host URL} - host address.
- {interface URI} - ontology URI, that represents the semantic interface.
- {class name} - ontology class name.
- {resource URI} - resource to be consulted.

RDF and OWL specifications are quite promising for the representation and integration of data. They are adhering to the open world assumption technologies. They have a strong emphasis on the inference of new information and not on checking data integrity. This feature may not be appropriate for many services that need to ensure the integrity for the data that they enter into the host. Semantic integrity ensures that data entered into a host accurately reflect the allowed values, in terms of structure and content.

4. SERIN INTEGRITY CONSTRAINTS

Mechanisms for verifying data integrity constraints are fundamental to the success of any CWA data centered system. These mechanisms are necessary to guarantee that CWA axioms remains true whenever new data is inserted or updated. In contrast, in OWA these mechanisms traditionally are not present.

In languages such as OWL, adherent to OWA, despite they have expressiveness that allows to represent efficiently a data model, they can not easily be used as a schema language. By adopting the OWA, the OWL axioms are mainly to make inferences. They are not proper to check data constraints. Motik [13] argues about the advantages of extending the semantics of OWL language features with the closed world. These extensions help to use either OWL inference as to data validation.

SDS extends the annotations provided in the SERIN specification. We propose the addition of six new annotations: NotNull, Unique, Id, ForeignURI, Embedded, and Internal. The first five apply to ontology properties while the latter one applies to ontology classes. The semantics of most of these annotations is similar to integrity constraints found in relational databases. With these new annotations, we intend provide a formal mechanism for integrity data validation in a SERIN interface, along the lines of LCWA.

The present work propose the adoption of six annotations:

1. NotNull - not null restriction [17] prohibits inserting a null value for a given attribute. The annotation NotNull requires an instance to have an association between a nonzero value with a annotated property. The annotation NotNull contributes to support LCWA in semantic data services. It brings data as complete as a data provider desire.

2. Unique - It checks if the values associated with a given property are unique within a given host. This annotation does not obligates the values to be explicitly defined for the property. It only checks whether they are single or not, if these values are set. The UNIQUE specification defines a set of attributes for a resource, such that no two resources have the same values for this set of attributes [17]. This annotation helps to support the unique name assumption in semantic data services. With this annotation, it is possible to infer that two instances are distinct one from another. The OWL language, in its current version, does not allow this assumption [8]. To do so, it is necessary to create an explicit statement stating that two instances relate (or not) to the same individual.

3. Id - The annotation Id has a relation with the relational database primary key specification. The primary key denotes one or more attributes as the principal means of identifying tuples within a table [17]. No two tuples in a table may have simultaneously the same value in the primary key attributes.

   In RDF model, every instance is a set of triplets associated with the same subject. This model provides URIs to identify subjects and predicates. Inside a single host, the subject URI represents an instance identifier. In this context, SDS server is responsible to control the creation and the use of these URIs. However, in scenarios where multiple hosts are considered, the RDF model does not allow to infer whether two instances URIs maintained in different hosts may represent the same entity. The annotation Id allows to infer that different instances in different hosts represent the same individual. To do so, it is necessary that both hosts share the same SERIN interface.

4. ForeignURI - It requires the annotated property value to be an instance of a URI maintained either inside the host database or inside the implemented SERIN interface. URIs outside the host and URIs that not follow the interface can not be associated with this type of property. An attempt to insert data which violates this rule imposes the SDS service launch a referential integrity failure.

   This annotation follows the concept of relational database foreign keys. Foreign keys are mechanisms for referential integrity constraint that guarantees that the values that appear in a relationship for a given table attributes set must also appear in another relationship for a set of equivalent attributes from another table[17].

5. Internal - Its purpose is to restrict the creation of new instances in a host database, except when they are members of a parent instance. The Internal annotation applies to classes rather than properties. As it is in the entity-relationship model, instances of Internal classes are weak entities, i.e., can only exist in the database if they participate in a relationship with a strong entity (the parent instance).

   SDS does not allow an an HTTP POST request to direct persist an Internal instance. The addition of a new Internal instance should occur along with its parent instance, via a HTTP PUT request. It is also possible to add a new Internal member for a parent instance already present in the base. A HTTP DELETE request can not exclude the parent instance unless it excludes all related Internal instances. This is necessary to maintain such consistency.

6. Embedded - The purpose of this annotation is that when a GET operation requests information about an object, it responses information about this object with

other objects that have relation with the requested object. The annotation Embedded over a object property defines this relation. This annotation therefore allows to relate a set of instances that should always be returned in a single block.

The advantage in using the annotation Embedded, is that with just an HTTP GET request is possible to obtain all necessary instances to generate the registration document. Without this annotation, on the other hand, several HTTP GET requests must be performed, one for each instance necessary to make the registration document. The annotation Embedded allows to combine instances in order to obtain a finer granularity representation.

The main characteristic for the presented annotations is to validate incoming data in a semantic data service. This means that whenever a data writing request (HTTP methods POST, PUT, or DELETE) is executed, the service must check the integrity constraints.

5. INTEGRITY CHECKING TEST CASE

To clearly illustrate the relevance of semantic data services (SDS), it is necessary to contextualize a scenario where there is a strong demand for linked data, and at the same time, an implicit need for these data are as complete and consistent as possible.

As an example scenario, let us suppose the following scenario: a software agent needs to automatically apply for a government acquisition. It needs to insert its proposal in the government server database. Some basic information must be checked before the server saves the data.

At first, both server and client must agree about what data may be inserted. The acquisition.owl defines the data model. This ontology defines classes and properties to describe the concepts Company and Sell Proposal, among others not shown here. The classes and properties described bellow are just part of the full model, not present here due to space restrictions.

Listing 1: Example of acquisiton.owl ontology classes with SERIN annotations.

```
<owl:Class rdf:about="#Company">
  <serin:get/>
  <serin:put/>
  <serin:post/>
</owl:Class>

<owl:Class rdf:about="#SellProposal">
  <serin:get/>
  <serin:put/>
  <serin:post/>
</owl:Class>
```

The acquisition ontology classes shown in listing 1 has received SERIN annotations that indicates possible operations for classes. The classes **Company** and **SellProposal** have annotations indicating that is possible to send only HTTP GET, PUT, and POST requests. Note that is is not possible to send DELETE requests since this annotation is not present on these classes definition.

The ontology section shown in listing 2 presents some integrity constraint definitions for some properties. The class **Company** is a domain for the properties **registryNumber** and **name**, each one with it own constraint annotations. Also property **hasCompany** indicates a relationship, with a constraint, between **Company** and **SellProposal**.

Listing 2: Example of properties with integrity constraints annotations.

```
<owl:DatatypeProperty
        rdf:about="#registryNumber">
<serin:id/>
<rdfs:domain rdf:resource="#Company"/>
<rdfs:range rdf:resource="&xsd;string"/>
</owl:DatatypeProperty>

<owl:DatatypeProperty rdf:about="#name">
<serin:notNull/>
<rdfs:domain rdf:resource="#Company"/>
<rdfs:range rdf:resource="&xsd;integer"/>
</owl:DatatypeProperty>

<owl:ObjectProperty
           rdf:about="#hasCompany">
<serin:foreignURI/>
<rdfs:domain
        rdf:resource="#SellProposal"/>
<rdfs:range rdf:resource="#Company"/>
</owl:ObjectProperty>
```

In this definition, a company registration number must be unique so it identifies the company. In this model, no other company can have the same id. The id may have a close and direct relationship with a company URI.

The property **name**, also related to the class **Company**, has a **notNull** annotation. This indicates that all Company instances must have a value for the property name. This value does not need to be unique but it must exist.

The property **hasCompany** indicates the existence of a relationship between Company and SellProposal. The addition of a SERIN annotation **foreignURI** states that these relationship must be explicit defined in the database. So every SellProposal instance must have an association to a Company instance, otherwise it violates this constraint.

As a last example, listing 3 shows an ontology section with an example of annotation **Embedded**. The existence of an annotation **Embedded** in the relationship between **SellProposal** and **Product** indicates that once a client request SellProposal instance data, it receives together the Product instances associated to the SellProposal instance. The advantage about this operation is that a client does not need to send requests to receive information about each Product associated to a SellProposal. It is possible to receive SellProposal and its products with only one request.

Based on this ontology, a host can implements the integrity constraint verification for every request it receives from each client agent. In this work, the OWL language was not expanded to support integrity constraints checking. This study is subject of future work. For now, hosts that implements the semantic interface can build algorithms for integrity constraints checking inside the web services methods. The algorithms should be developed also considering non-functional requirements.

Listing 3: Example of a property with annotation Embedded.

```
<owl:ObjectProperty
                rdf:about="#hasProduct">
  <serin:embedded/>
  <rdfs:domain
          rdf:resource="#SellProposal"/>
  <rdfs:range  rdf:resource="#Product"/>
</owl:ObjectProperty>
```

For its turn, when it needs to insert/update data in a host, the client agent can send the request with correct information because the data schema is already known. This can contribute to improve performance requirements.

6. CONCLUSIONS

This article describes an approach to provide integrity constraints to semantic data services. This approach adheres to the local closed world assumption.

This proposal is a step towards building a web of interconnected data that is available for writing. It allows clients to persist data in hosts databases without violating the established integrity constraints.

We have introduced a set of new annotations reaching those used in the semantic interfaces that describe the semantic data services. These annotations provide an integrity constraint analogous to the constraints found in relational data models. This mechanism allows SDS to validate the data that they process.

Our implementation for this proposal [1] uses Java language and the Jena framework. This implementation was evaluated against other data service proposals. Some related works use the closed world assumption without adopting semantic Web standards. Other proposals have adhered to the Semantic Web but they still presents some challenges regarding the data write permission to the web services clients. SDS fills some gaps observed in these studies.

As future work, we intend to include a mechanism for access permission for data writing. Other possible studies are: the construction of data access concurrency control; transaction control; and the definition of URIs to run parameterized queries, e.g. paging and selection filters.

7. ACKNOWLEDGMENTS

The authors would like to thank SERPRO, Federal Data Processing Service, for supporting this research.

8. REFERENCES

[1] S. Bechhofer, F. van Harmelen, J. Hendler, I. Horrocks, D. L. McGuinness, P. F. Patel-Schneider, and L. A. Stein. OWL Web Ontology Language Reference, 2004.

[2] T. Berners-Lee. Linked Data - Design Issues, 2006.

[3] T. Berners-lee, R. Cyganiak, M. Hausenblas, J. Presbrey, O. Seneviratne, and O.-e. Ureche. Realising A Read-Write Web of Data. Technical report, 2009.

[4] K. L. Clark. Negation as Failure. In H. Gallaire and J. Minker, editors, *Logic and Data Bases*, pages 292–322, New York, 1978. Plenum Press.

[5] R. T. Fielding. *Architectural styles and the design of network-based software architectures*. PhD thesis, University of California, Irvine, 2000.

[6] A. Garrote and M. N. M. García. RESTful writable APIs for the web of Linked Data using relational storage solutions. In *In WWW2011 Workshop: Linked Data on the Web (LDOW2011*, 2011.

[7] T. Heath and C. Bizer. *Linked Data: Evolving the Web into a Global Data Space*. Synthesis Lectures on the Semantic Web. Morgan {&} Claypool Publishers, 2011.

[8] P. Hitzler, M. Krötzsch, B. Parsia, P. F. Patel-Schneider, and S. Rudolph. *OWL 2 Web Ontology Language: Primer*. W3C Recommendation, 2012.

[9] M. Kirchhoff and K. Geihs. Semantic description of OData services. In *Proceedings of the Fifth Workshop on Semantic Web Information Management*, SWIM '13, pages 2:1—-2:8, New York, NY, USA, 2013. ACM.

[10] M. Knorr, J. J. Alferes, and P. Hitzler. Local Closed World Reasoning with Description Logics Under the Well-founded Semantics. *Artif. Intell.*, 175(9-10):1528–1554, 2011.

[11] A. Krisnadhi, K. Sengupta, and P. Hitzler. Local Closed World Semantics : Keep it simple , stupid! *Proceedings of the 24th International Workshop on Description Logics (DL)*, 2011.

[12] F. Manola and E. Miller. RDF Primer, 2004.

[13] B. Motik, I. Horrocks, and U. Sattler. Bridging the gap between OWL and relational databases. *Proceedings of the 16th international conference on World Wide Web (WWW'07)*, pages 807–816, 2007.

[14] B. d. A. Muniz, L. M. Chaves, H. A. Lira, J. R. V. Dantas, and P. P. M. Farias. SERIN - AN APROACH TO SPECIFY SEMANTIC ABSTRACT INTERFACES IN THE CONTEXT OF RESTFUL WEB SERVICES. *Proceedings of the IADIS International Conference on WWW/Internet*, pages 187–194, 2013.

[15] B. D. A. Muniz, L. M. Chaves, J. C. C. Neto, J. R. V. Dantas, and P. P. M. Farias. SERIN - SEMANTIC RESTFUL INTERFACES. *Proceedings of the IADIS International Conference on WWW/Internet*, pages 463–467, 2011.

[16] M. Nally, S. Speicher, J. Arwe, and A. L. Hors. *Linked Data Basic Profile 1.0*. W3C Member Submission. World Wide Web Consortium, 2012.

[17] A. Silberschatz, H. F. Korth, and S. Sudarshan. *Database System Concepts*. McGraw-Hill Science/Engineering/Math, 5 edition, 2005.

[18] S. Speicher, J. Arwe, and A. Malhotra. *Linked Data Platform 1.0*. W3C Recommendation. World Wide Web Consortium, 2014.

[1]An working example may be seen at `http://opendataserin-unifor.rhcloud.com/`.