SEMANTIC VALIDATION OF T&E XML DATA

Jakub Moskal, Mieczyslaw Kokar, Jon Morgan[†]

* VIStology, Inc., Framingham, MA 01701
† JT3, LLC, Edwards AFB, California

ABSTRACT

It is anticipated that XML will heavily dominate the next generation of telemetry systems. The syntax of XML-based languages can be constrained by a schema that describes the structure of valid documents. However, the schemas cannot express all dependencies between XML elements and attributes, both within a single document and across multiple documents. This prohibits the XML validation process from being fully automated with standard schema processors.

This paper presents an approach that is based on the W3C Semantic Web technologies and allows different vendors and system integrators to independently develop their own semantic validation rules. The rules are equipped with powerful semantics, which allows for specification and validation of complex types of constraints. The approach is not specific to a particular T&E standard and is entirely standards-based.

Keywords: XML semantic validation, semantic web, inference rules, TMATS, MDL

I INTRODUCTION

It is anticipated that XML will heavily dominate the next generation of telemetry systems [1]. Already today, the IRIG 106[2], a comprehensive telemetry standard maintained and developed by the Telemetry Group of the Range Commanders Council (RCC), incorporates three XML-based standards: Telemetry Attributes Standard (TMATS), Data Display Markup Language (DDML) and Instrumentation Hardware Abstraction Layer (IHAL)[3]. Additionally, the integrated Network Enhanced Telemetry (iNET) Metadata Description Language (MDL)[4] is anticipated to be incorporated into IRIG as well.

Since XML does not capture the semantics of particular domains, the applications that handle the XML documents must hard-code procedures for the interpretation of the meaning of XML documents. For instance, there is no difference between the semantics of the following two XML snippets: <name>John</name> and <aaa>John</aaa>. Obviously, the first version is easier to comprehend by humans, but it makes no difference to a computer. Thus, independent vendors must agree on the meaning of the XML tags and implement the interpretation procedures in order to effectively make use of the data exchanged between them.

The syntax of XML-based languages can be constrained by a schema that describes the structure of valid documents and allowable values of attributes using data typing facilities. However, even on the syntactical level, the schemas cannot express all dependencies between

XML elements and attributes, both within a single document and across multiple documents. This greatly complicates the validation process of XML data used in testing, because the process cannot be fully automated with standard schema processors. This applies to large multi-vendor systems in particular, where XML data might contain conflicts that cannot be detected with off-the-shelf validators.

We have investigated a solution that is based on Web Ontology Language (OWL) [5, 6], a language with formal, computer-processable semantics. Using a language with formal semantics guarantees that inference is sound and complete, and with OWL, tractable. XML data can be automatically *lifted* to OWL and then processed using standards-based inference engines, which can detect conflicts not detectable at the XML level. The inference capability of OWL can be augmented with rules that can be used to encode specific semantic rules for handling XML, but also for detecting conflicts. The rules are of declarative nature — they have no prescribed order of invocation. This allows vendors and system integrators to independently develop their own validation rules. Moreover, rules can invoke procedural code and interact with a GUI, showing the user information about the validation conflicts. Figure 1 shows the concept of the investigated approach. It must be noted that the approach relies on open standards only and can be realized using general-purpose tools — XSD validator and an OWL inference engine.

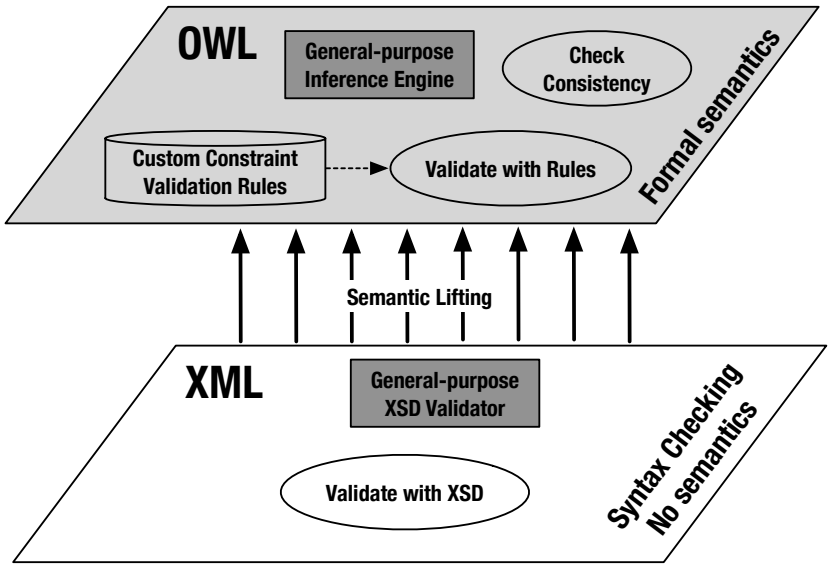

Figure 1: The approach to validation of semantic constraints described in this paper.

This paper describes the results of the investigation and is organized as follows: Section II provides a short introduction to the Semantic Web technologies, while Section III describes the process of semantic lifting and how it was applied to TMATS and MDL schemas and instance documents. Section IV describes how semantic rules can be applied to the semantic representation of T&E data in order to detect conflicts.

II SEMANTIC WEB TECHNOLOGY

The knowledge-based approach described in this paper stems from the production systems architecture where an *inference engine* component, also known as *reasoner*, plays the central role. This approach separates the knowledge about the particular domain from the flow of control (algorithm). The reasoner that is oblivious to the domain handles the control flow, i.e. it is general-purpose. The knowledge about the domain is formulated by domain experts and is encapsulated in the Knowledge Base (KB) in the form of ontology (facts) and rules. Ontology defines common terms and concepts used to describe and represent the domain knowledge. Rules, similarly as ontologies, have a declarative nature and express the logic of a domain-specific program without describing its control flow, which allows the reasoner to infer new facts given some initial facts. In practice, rules can be used to implement a specific application, but they can also be used to implement policies, e.g. security or regulatory.

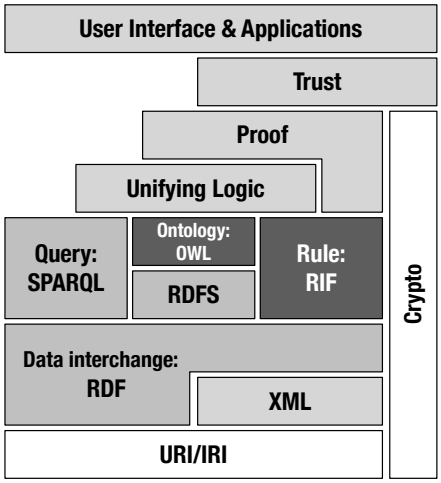

Figure 2: *Semantic Web Layer Cake* — the architecture of Semantic Web [7]

We investigated the use of OWL as the knowledge representation language. OWL is a standard that has been widely adopted in the Semantic Web community. Semantic Web (SW) builds on the traditional hypertext web technologies: IRI/URI, Unicode, XML and XML namespaces. Figure 2 shows the so called "Semantc Web Layer Cake", emphasizing the foundation of OWL in previous technologies. The elements of the cake, representing different technologies, have been subject to standardization since the dawn of SW. Currently, from the bottom up to OWL, all technologies are widely-used and accepted W3C standards (the remaining ones are subject of ongoing research):

- **Resource Description Framework (RDF)** [8] — a framework for conceptual modeling of information that is based on the idea of making statements about resources.
- **RDF Schema** [9] — provides means for user communities to define application-specific vocabularies (or ontologies).
- **SPARQL** [10] — a powerful query language for RDF with SQL-like syntax, which allows to pull, explore and transform semantic data using graph patterns.
- **Rule Interchange Format (RIF)** [11] — an extensible family of languages, called *dialects*, that enables exchange of rules among rule systems. RIF can be used to extend the power of OWL by adding implication in the form of `if-then` rules.
- **Web Ontology Language (OWL)** OWL, is the last part of the SW layer cake that has been standardized under the W3C umbrella. The semantics of OWL is based on Description Logics (DL) [12], a family of Knowledge Representation (KR) formalisms equipped with a formal, model-based semantics. DLs are decidable fragments of First Order Logic (FOL) and have been designed to optimize the trade-off between expressiveness and complexity of reasoning. The architecture of a system based on DL allows

for setting up a *knowledgebase* (KB) and for reasoning about its content. The KB consists of two components, *TBox* and *ABox*. In the context of OWL, the TBox introduces axioms defining the ontology — classes and properties; the ABox contains assertions about named individuals — instances of classes and properties defined in the TBox.

XML and SW technologies

There are many parallels between the technologies used in XML and in SW (see Table 1). [13] describes the comparison of the two worlds in great detail. While both worlds facilitate representation of data, metadata and querying, the SW world provides technologies that rely on formal semantics and the capability of automatic inference, which in turn can be utilized to enhance the system integration process.

Table 1: Comparison of XML and SW technologies

	XML Technologies	**SW technologies**
Data	XML	RDF, OWL ABox
Metadata	Schema	RDFS, OWL TBox
Query language	XQuery, XPath	SPARQL
Semantics	None	Model Theory-based
Inference capability	No	Yes

III LIFTING THE T&E XML DATA

The process of semantic lifting, is the process of converting non-semantic data (e.g. XML) to a language with formal semantics (e.g. OWL). It can be applied to any XML data, not just to that of T&E. This process can be largely automated, but with some caveats because not all XML schema constructs have a direct counterpart in OWL. A recent survey of approaches to applying Semantic Web tools to XML data [13] shows that there is a plethora of research efforts in this area. Unfortunately, virtually all of those efforts are limited to prototypes (at best) that are not publicly available either as free or as commercial products. In the end, the following three tools were evaluated against the TMATS and MDL XML schemas: XSD2OWL/XML2RDF [14], TopBraid Composer Maestro Edition [15] and Ontmalizer [16]. Unfortunately, none of the tools produced a complete ontology for TMATS or MDL that could be used for the purpose of validation. There are various reasons for which the translation did not work well. The following observations have been made:

- The TMATS schema makes extensive use of the more complex XSD Schema constructs, like xs:choice, xs:key, or xs:keyref. These constructs were typically omitted, or not translated correctly by the reviewed tools.
- The simple types were translated well by XSD2OWL and Ontmalizer.
- TMATS uses a type taxonomy, which when automatically translated to OWL, looks far from what one would design when building an ontology from scratch. This is important, since the complexity of the resulting ontology has a direct impact on the quality of rules that would be written for this ontology.

Given the lack of appropriate tools to lift T&E data to OWL, we decided to develop our own semantic lifter (Figure 3 shows its design). The TBoxBuilder takes an XSD schema and generates a corresponding OWL ontology. It also instantiates an ABoxBuilder for building OWL data from XML instance documents, based on the mapping produced by the TBoxBuilder. Both builders generate semantic data that closely reflects the structure of the input XML documents. In order to yield an ontology that better represents the domain, both builders can be tailored to a specific schema by implementing appropriate Customizer API in Java.

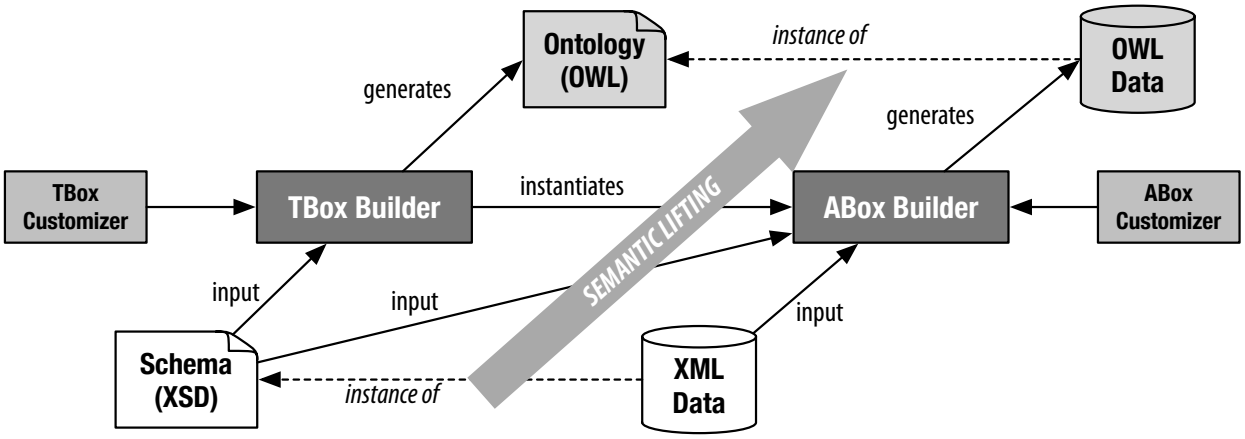

Figure 3: Semantic Lifter

The lifter is entirely generic, i.e. it can work with any XML schema and data, thus can be applied to any T&E XML-based standard. The implementation relies on JAXB and Java reflection. While JAXB is used to generate class definitions corresponding to complex types defined in the schema, reflection is used to dynamically inspect these definitions in runtime and produce corresponding OWL concepts. This approach made a robust and rapid construction of the tool possible, mainly because many of the decisions on how to interpret the XML constructs are already made by JAXB. In a sense, the object world of Java is semantically closer to OWL than XML.

The XML-based standards in T&E are still being defined/modified and slowly adopted by the users. For this reason, the development and standardization of complete and stable ontologies at this time is not possible since such ontologies would have to be modified in sync with the modifications of the XML-based standards. Our semantic lifter can take any XML-based standard and map it to an OWL ontology. In addition, we developed customizers for TMATS and MDL schemas, which together yield ontologies suitable for expressing semantic rules. This process is entirely automated and would have to be adjusted only if major schema changes occur.

IV DETECTING CONFLICTS WITH RULES

With this semantic foundation in place, constraint violation rules can be written to address the constraints that cannot be expressed in XML schema languages. The constraint validation rules were developed in the BVR language supported by the BaseVISor inference engine

that we used. In the future, we plan to develop rules in open standards, like RIF. BVR rules can be easily converted to RIF since on the semantic level, RIF can express everything that BVR can. The choice of BVR was dictated merely by convenience since that is the native rule format for BaseVISor, which was used in evaluation of the approach.

Figure 4 shows how semantic rules (BVR or RIF) are used to provide constraint validation. Each rule consists of two parts — "if" and "then". While the first part defines what constitutes a constraint violation, the latter defines what should be done if the violation occurs. Hence, each constraint must have an associated rule. These rules are taken as a runtime input and can be easily shared among different entities (users, vendors, etc.), or combined with other rule sets.

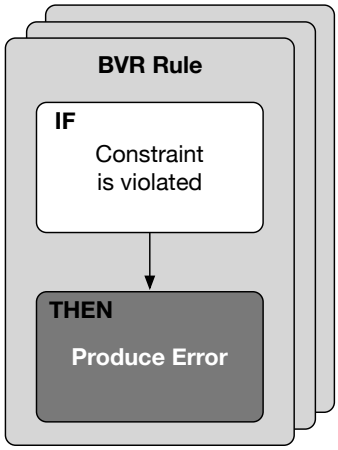

Figure 4: Structure of Semantic Validation Rules

The rules can be used to express basic constraints such as the following:

- <BasebandSignal/> element is not allowed when <SignalStructureType/> is "SCO's" or "Other"
- <PostProcessModificationType/> is allowed only when the value of <Type/> is not "N".
- <NumberOfTracksOrChannels> element is required when the value of <Type/> is not "N".

Listing 1 shows a rule, which checks whether the dirty bit of the MDL root element and the dirty bit of any of TmNSManageableApp elements defined in an MDL document are the same. If they are, then the inference engine triggers the <head> part, which in this case means producing an error message with corresponding details. The constraint is not really part of the standard, but it shows that the rules can access values from any part of the ontology (produced from XML) and process them. In this case, the values are compared, but they could also be processed further, calculated, compared with other values etc.

Listing 1: Example constraint validation rule.

```
<rule name="Rule1">
    <body>
      <Individual variable="Root" rdf:type="mdl:MDLRootType">
        <mdl:isDirtyBit variable="RootBit" />
          <mdl:hasNetworkDomains>
            <mdl:hasNetwork>
              <mdl:hasNetworkNode>
                <mdl:hasTmNSManageableApps>
                  <mdl:hasTmNSManageableApp variable="App">
                    <mdl:isDirtyBit variable="AppBit" />
                  </mdl:hasTmNSManageableApp>
                </mdl:hasTmNSManageableApps>
              </mdl:hasNetworkNode>
            </mdl:hasNetwork>
          </mdl:hasNetworkDomains>
        </Individual>
      <eq><RootBit /><AppBit /></eq>
```

```
    </body>
    <head><error>
        MDL (<Root/>) and TmNSManageableApp (<App/>) use the same dirty bit value.
      </error></head>
  </rule>
```

Another example are more sophisticated constraint checking rules that require algebraic operations as well as access to multiple parts of the XML documents. In particular, constraint validation rules were written to ensure that in a given MDL instance document the maximum power consumption of each module within the DAU network node is not higher than what it can source. Due to the length constraints on the paper, the syntax of these rules is not shown here.

Because the rules are applied to a semantic representation, they can leverage numerous *implicit* facts that can be produced based on the 75 generic inference rules that come with OWL and are processed by any standard inference engine. OWL inference is triggered by axioms that can be added to the ontology via tailoring, as described above.

Figure 5 shows the kind of OWL inference that can be done by the inference engine when provided with the following axioms in the MDL ontology:

- Add a new property, hasPart, and assert it as transitive.
- Add a new property, isPartOf, and assert it as an inverse property of hasPart.
- Define all properties that represent parthood (hasNetworkDomains, hasNetwork, etc.) as subproperties of hasPart.

Given these axioms, and the taxonomy of the classes shown on the left side of Figure 5, the inference engine infers numerous facts (the right side of the figure shows only a fraction of these new inferred facts, for clarity). Because hasPart is transitive, any instance of MDL class will be inferred as related by hasPart to everything below. Conversely, because isPartOf is the inverse of hasPart, everything below will be inferred as related by isPartOf with the MDL root.

Implicit facts *inferred* by the reasoner can be leveraged in order to produce significantly less verbose and more intuitive rules. For instance, consider the rule shown in Figure 2, which validates the following constraint: "Every part of the MDL configuration must be manufactured by the same vendor as the vendor of the network node on which it should be deployed.". Note that the pattern in the <body> of the rule does not have to explicitly list all children of the MDL root element and all possible combinations. The pattern leverages part-whole relations (hasPart, isPartOf) that were added to the ontology.

Listing 2: Example constraint validation rule.

```
<rule>
<body>
  <Individual variable="MDL" rdf:type="MDL">
    <hasPart variable="A">
      <rdf:type variable="PartType" />
      <hasManufacturer variable="Vendor1"/>
      <isPartOf variable="B" rdf:type="NetworkNode">
        <hasManufacturer variable="Vendor2" />
      </isPartOf>
    </hasPart>
```

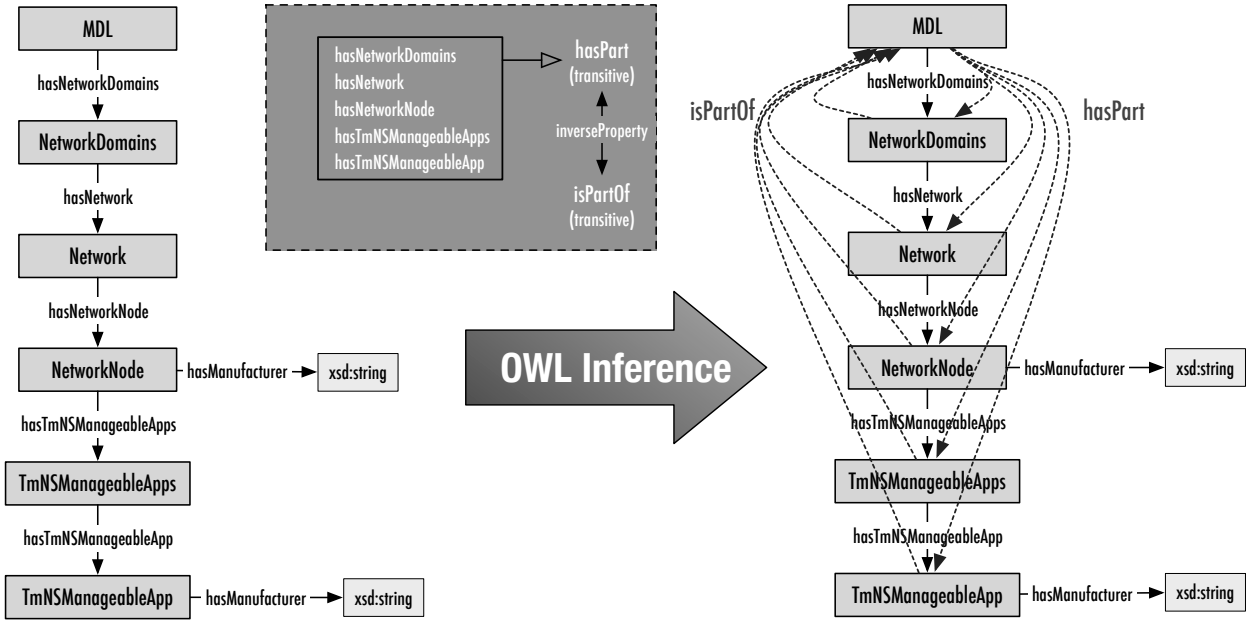

Figure 5: OWL axioms and related inference in the MDL ontology.

```
  </Individual>
  <notEqual><Vendor1/><Vendor2/></notEqual>
</body>
<head>
  <warning><A/> (<PartType/>) is manufactured by a different
  vendor than the network node (<B/>) on which it is deployed.
  </warning>
</head>
<rule>
```

In the follow up to the work we are describing in this paper, our plan is to promote these results withing the T&E community and develop rule sets for each T&E standard. These sets could be standardized themselves so that the entire validation process could be done automatically using off the shelf general-purpose tools.

V CONCLUSION

We investigated the use of Semantic Web technologies to automatically lift the XML data to OWL, a language with formal semantics, and apply the validation rules to that representation. Constraint validation rules expressed with Semantic Web technologies have high expressiveness and are suitable for expressing the types of semantic constraints that are needed for the T&E data. The approach was implemented in the form of a prototype, called xVISor, and successfully tested with MDL and TMATS data. xVISor was demonstrated to detect violations of semantic constraints that are not expressible in XML. Additionally, the use of ontologies within xVISor demonstrated the capability of developing semantic rules that take advantage of the inheritance and the generalization in the representation of the domain knowledge — the aspect that cannot be achieved with rules alone.

The effort described in this paper showed that the proposed approach is feasible and that it could eventually lead to a standardization of at least two types of artifacts:

- Rule structures — a language for defining constraints that is agnostic to any specific XML standard.
- Rule sets — sets of semantic rules that are specific to particular T&E XML standards.

If the standardization effort succeeds, T&E standards like TMATS and MDL should include standard validation rule sets. This would greatly improve the existing validation process, which currently partially relies on constraints that are meant for human consumption only and need to be interpreted in software. Declarative rules that are supported by this technology have formal semantics, yet they are not hardcoded and can be altered without making any changes to the validation software. This design promotes a loose coupling between the T&E standardization efforts and the development of the validation software.

ACKNOWLEDGEMENTS

This work was performed under Air Force contract FA9302-14-M-0002 "Rule-Based XML Validation for T&E (RuBX)". Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the Air Force.

We would like to thank Bruce Lipe from the Air Force Test Center at Edwards Air Force Base who provided us with relevant data and use cases. We would also like to thank William A. Malatesta from Naval Air Systems Command (NAVAIR) in Patuxent River, who provided us with a challenging problem related to a system-leve'l constraint in the MDL schema and gave very useful feedback on the technology described in this paper.

REFERENCES

[1] T. Darr, J. Hamilton, R. Fernandes, and C. H. Jones, "Design considerations for xml-based t&e standards," in *Proceedings of the 47th International Telemetering Conference*, Las Vegas, NV, 2011.

[2] T. Standards, "Irig standard 106," Range Commanders Council, White Sands Missile Range, NM, Tech. Rep., 2013.

[3] Hamilton, Fernandes, Koola, and E. Jones, "An instrumentation hardware abstraction language," in *Proceedings of the 42nd International Telemetering Conference*, San Diego, CA, 2006.

[4] M. Moore, J. Price, A. Cormier, and W. Malatesta, "Metadata description language: The iNET metadata standard language," in *Proceedings of the 45th International Telemetering Conference*, Las Vegas, NV, 2009.

[5] W3C, *Web Ontology Language (OWL)*. W3C Recommendation, February 10, 2004, available at http://www.w3.org/2004/OWL/.

[6] W3C, *OWL 2 Web Ontology Language Primer*. W3C Recommendation, December 11, 2012, available at http://www.w3.org/TR/owl2-primer/.

[7] T. Berners-Lee, "Semantic web - xml2000: Architecture," http://www.w3.org/2001/09/06-ecdl/slide17-0.html, 2000.

[8] R. W. Group, *Resource Description Format (RDF) Primer*. W3C Recommendation, February 10, 2004, available at http://www.w3.org/TR/rdf-primer/.

[9] D. Brickley and R. V. Guha, "RDF Vocabulary Description Language 1.0: RDF Schema," http://www.w3.org/TR/rdf-schema, Feb. 2004.

[10] S. Harris and A. Seaborne, "Sparql 1.1 query language. w3c recommendation," http://www.w3.org/TR/sparql11-query/, March 21 2013.

[11] M. Kifer and H. Boley, "RIF overview," 2010, retrieved from http://www.w3.org/TR/rif-overview/.

[12] F. Baader, D. Calvanese, D. McGuinness, D. Nardi, and P. Patel-Schneider, *The Description Logic Handbook: Theory, Implementation and Applications*. Cambridge University Press, 2003.

[13] N. Bikakis, C. Tsinaraki, N. Gioldasis, I. Stavrakantonakis, and S. Christodoulakis, "The xml and semantic web worlds: Technologies, interoperability and integration: A survey of the state of the art," in *Semantic Hyper/Multimedia Adaptation*, ser. Studies in Computational Intelligence, I. E. Anagnostopoulos, M. Bieliková, P. Mylonas, and N. Tsapatsoulis, Eds. Springer Berlin Heidelberg, 2013, vol. 418, pp. 319–360. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-28977-4_12

[14] R. García, "A semantic web approach to digital rights management," Ph.D. dissertation, Department of Technologies. Universitat Pompeu Fabra, Barcelona, 2006.

[15] TopQuadrant, "Topbraid composer maestro edition," http://www.topquadrant.com/tools/IDE-topbraid-composer-maestro-edition/.

[16] M. Yüksel, "A semantic interoperability framework for reinforcing post market safety studies," Ph.D. dissertation, Middle East Technical University, 2013.