ECA2LD: Generating Linked Data from Entity-Component-Attribute runtimes

Torsten Spieldenner*, René Schubotz, Michael Guldner
German Research Center for Artificial Intelligence (DFKI)
(Saarbrücken Graduate School of Computer Science)*
66123 Saarbrücken, Germany
Email: {firstname.lastname}@dfki.de

***Abstract*—Large-scale Internet of Things (IoT) applications, like Smart Cities, are ever changing pieces of software. The paradigm of Entity-Component-Attribute (ECA) based software design is well-suited to keep these applications changeable and maintainable. At the same time, W3C working groups propose the Web as IoT convergence platform. Semantic annotations on the data in RDF help tackle cross-domain interoperability issues. While the generation of RDF data from data storage layers has undergone thorough research, the automatic exposure of application run-time data is to this day incomplete. We for this formalize the Entity-Component-Attribute pattern and present an automated mapping to a structure compliant with the Linked Data Platform W3C standard. This structural mapping is then augmented by application domain specific semantics. The result lifts a software design pattern highly suitable for large-scale IoT applications to Linked Data.**

***Keywords*-Entity-Component-Attribute model; Linked Data; Resource Description Framework (RDF); Mappings;**

I. Introduction

The continual change that software undergoes during its lifetime is generally called evolution, and the degree to which it is easy or hard to change existing software is often called *changeability* [1], [2]. Especially in perdurable and large scale Internet of Things (IoT) platforms, software design with the intention to optimize changeability is imperative [3]. Towards this end, the architectural pattern of *Entity-Component-Attribute (ECA) based software design* is particularly well-suited [4]. Focusing on the principle of "composition over inheritance", the role of an entity is no longer determined by class inheritance or attribution hierarchy, but dynamically determined by the set of attached components. This significantly improves changeability of entities and reuse of components.

ECA patterns have been successfully applied in the design of changeable IoT platforms and applications [5], however, the enablement of *seamless cross-domain interoperability* between independently developed IoT applications and platforms, one of the central challenges facing IoT [6], is not directly addressed by this design paradigm.

In this respect, the W3C Web of Things Working Group[1] proposes to use the Web as an IoT convergence platform. The group develops initial standards for this *Web of Things* (WoT) [7] by defining a Web-based abstraction layer for IoT platforms, protocols, data models and communication patterns. To unleash its full potential, the emerging WoT is expected to evolve into a *Semantic Web of Things* (SWoT) [8].

The SWoT will heavily rely on Linked Data principles [9] to semantically describe IoT entities semantically in terms of their actions, properties, events and metadata [10] independent of the underlying IoT platform. For large-scale scenarios and environments, the development of IoT platforms and applications on the SWoT will require software tooling that enables

1) (semi-) automated mappings from application data layouts to Linked Data
2) declarative augmentation of Linked Data with application-specific semantics
3) exposition of dynamic IoT runtime data as Linked Data

While there are numerous works investigating (semi-) automated mappings from heterogeneous data structures and serializations to the RDF data model [11], [12], [13], [14], [15], little research has been conducted on the dynamic mapping of runtime environments [16], [17] to RDF. In particular, we are not aware of any works on how to leverage ECA-driven software applications on the Semantic WoT.

This paper makes the following contributions. We fomalize the notation of an ECA system, and detail on the automated *structural mapping* between ECA runtimes and the W3C Linked Data Platform[2]. Next, we explain how domain experts can declaratively augment these generated structural mappings with domain-specific semantics. Finally, Linked Data clients can retrieve data as provided by the ECA-based Web Application via a semantic Linked Data Web-interface.

II. Entity-Attribute Models

The understanding of Entity-Attribute models varies in literature. Common for all variations is the notion of an *entity* as an empty data container which is closer specified by a set of typed *attributes* that carry the actual values. The

[1]https://www.w3.org/WoT/WG/

[2]https://www.w3.org/TR/ldp/

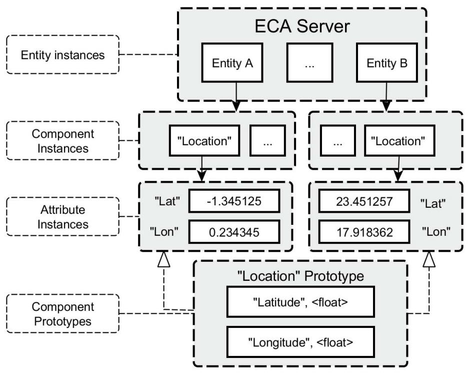


Figure 1. Entity-Component-Attribute model based on the example of a component that defines a "Location".

IoT Context Broker by Moltchanov et. al [5] keeps to the levels of entities and attributes.

The systems RealXtend [18], [19] and FiVES [20] include the notion of *components* (*Entity-Component-Attribute pattern, ECA*; see also Fig. 1). Components can be considered as prototypes of attribute sets that belong to the same concept. When a component is attached to an entity instance, a new instance of the component and the respective attribute set is created from this prototype. RealXtend and FiVES share this design with game engines like Unity3D[3] and Unreal Engine[4].

RealXtend equips components with application logic that is directly contained as code in the component implementation. Same holds for game engines. The work by Wiebusch as well as the FiVES server system consider the Entity-Component-Attribute model as data model only. Logic is implemented in independent *systems* (referred to as *plugins* in FiVES), with a careful design of how external logic accesses the data.

We adapt the understanding of components as by Wiebusch, and FiVES, with logic implemented separately to avoid the tight coupling between components and their specific implementation as in RealXtend or game engines. We derive from this the following formal definition of the ECA architectural pattern.

Let $\mathbf{e}$ denote an entity instance, and $\mathcal{P}_\mathbf{C}$ denote the set of all component prototypes. Then we define the following sets:

- $\mathbf{E}$ is the set of all entity instances. An entity instance is defined as $\mathbf{e} = (n_\mathbf{e}, \mathbf{C}_\mathbf{e})$, with $n_\mathbf{e} \in \Sigma^+$ being the unique identifier for $\mathbf{e}$ over alphabet Σ, and $\mathbf{C}_\mathbf{e}$ being the set of component instances attached to $\mathbf{e}$.
- $\mathbf{C}_\mathbf{e}$ is the set of all component instances attached to an entity instance $\mathbf{e}$. A component instance is defined as $\mathbf{c} = (n_\mathbf{c}, p_\mathbf{c}, \mathbf{A}_{\mathbf{c},\mathbf{e}})$, with $n_\mathbf{c} \in \Sigma^+$ being the unique identifier for $\mathbf{c}$, $p_\mathbf{c} \in \mathcal{P}_\mathbf{C}$ being the prototype that $\mathbf{c}$ is an instance of, and $\mathbf{A}_{\mathbf{c},\mathbf{e}}$ being the set of all attribute instances attached to $\mathbf{c}$.
- $\mathbf{A}_{\mathbf{c},\mathbf{e}}$ the set of all attribute instances attached to a component instance $\mathbf{c}$. An attribute instance is defined as $\mathbf{a} = (n_\mathbf{a}, v, t)$, with $n_\mathbf{a} \in \Sigma^+$ being the unique identifier for $\mathbf{a}$, v denoting the attribute instance's current *value*, and t denoting the ECA runtime type of $\mathbf{a}$.

[3]http://www.unity3d.com
[4]http://www.unrealengine.com

III. Linked Data Platform in very brief

The *Linked Data Platform* (LDP) specifies a simple architecture to expose Linked Data on the Web.

The basic element of LDP is a *LDP Resource*. Every LDP Resource must be an HTTP endpoint with at least HTTP/1.1 protocol compatibility, and accept at least HTTP GET requests, and others depending on the type of resource.

`ldp:RDFSource` exposes its data upon a GET request and `MUST` return a full RDF graph in `text/turtle` format (or `application/ld+json`, if requested).

`ldp:Container` is a `ldp:RDFSource` that manages a set of LDP Resources and provides information about access, modification, and filtering of the contained elements.

`ldp:BasicContainer` is a `ldp:Container` that specifies linked documents in the form of *Containment Triples* of the form (*container-uri*, `ldp:contains`, *document-uri*).

The LDP specification specifies more concepts. Our mapping, however, will only make use of above concepts.

IV. Structural Interoperability

Structural interoperability establishes a common format for data exchange between applications [21]. In the following, we detail on the automated structural mapping between ECA runtime environments and W3C LDP compliant Linked Data servers. By repeated application of a set of mapping rules (cf. ① to ④), structural interoperability between ECA runtimes and LDP servers is established.

We assume the existence of functions $\nu : \Sigma^+ \rightarrow \texttt{IRI}$ and $\rho : \mathcal{P}_\mathbf{C} \rightarrow \texttt{IRI}$ for minting fresh `IRI`s from identifiers and component prototypes. Although several guidelines exist for minting `IRI`s[5], we do not make assumptions on ν or ρ.

$$① \quad \frac{(n_\mathbf{e}, \mathbf{C}_\mathbf{e}) \in \mathbf{E} \quad \forall (n_\mathbf{c}, p_\mathbf{c}, \mathbf{A}_{\mathbf{c},\mathbf{e}}) \in \mathbf{C}_\mathbf{e}}{\begin{array}{l} \nu(n_\mathbf{e})\ \texttt{rdf:type}\ \texttt{ldp:BasicContainer}\ . \\ \nu(n_\mathbf{e})\ \texttt{dct:identifier}\ \text{“}n_\mathbf{e}\text{”\^{}\^{}}\texttt{xsd:String}\ . \\ \nu(n_\mathbf{e})\ \texttt{ldp:hasMemberRelation}\ \texttt{dct:hasPart}\ . \\ \nu(n_\mathbf{e})\ \texttt{dct:hasPart}\ \nu(n_\mathbf{c})\ . \end{array}}$$

[5]https://www.w3.org/TR/cooluris/

① Each entity instance $\mathbf{e} = (n_\mathbf{e}, \mathbf{C}_\mathbf{e})$ is mapped to a `ldp:BasicContainer` with `IRI` $\nu(n_\mathbf{e})$. This entity container maintains a membership triple $(\nu(n_\mathbf{e}), \texttt{dct:hasPart}, \nu(n_\mathbf{c}))$ for each component instance $(n_\mathbf{c}, p_\mathbf{c}, \mathbf{A}_{\mathbf{c},\mathbf{e}})$ attached to $\mathbf{e}$.

$$② \quad \frac{(n_\mathbf{c}, p_\mathbf{c}, \mathbf{A}_{\mathbf{c},\mathbf{e}}) \in \mathbf{C}_\mathbf{e} \quad \forall (n_\mathbf{a}, v, t) \in \mathbf{A}_{\mathbf{c},\mathbf{e}}}{\begin{array}{l} \nu(n_\mathbf{c})\ \texttt{rdf:type ldp:BasicContainer .} \\ \nu(n_\mathbf{c})\ \texttt{dct:identifier} \text{ “}n_\mathbf{c}\text{”}\texttt{^^xsd:String .} \\ \nu(n_\mathbf{c})\ \texttt{dct:isPartOf}\ \nu(n_\mathbf{e})\ \texttt{.} \\ \nu(n_\mathbf{c})\ \texttt{ldp:hasMemberRelation dct:hasPart .} \\ \nu(n_\mathbf{c})\ \texttt{dct:hasPart}\ \nu(n_\mathbf{a})\ \texttt{.} \\ \nu(n_\mathbf{c})\ \texttt{rdfs:isDefinedBy}\ \rho(p_\mathbf{c})\ \texttt{.} \end{array}}$$

② Each component instance $\mathbf{c} = (n_\mathbf{c}, p_\mathbf{c}, \mathbf{A}_{\mathbf{c},\mathbf{e}})$ is mapped to a `ldp:BasicContainer` with `IRI` $\nu(n_\mathbf{c})$. This component container uses `dct:isPartOf` to indicate its containing entity container $\nu(n_\mathbf{e})$ and maintains a membership triple $(\nu(n_\mathbf{c}), \texttt{dct:hasPart}, \nu(n_\mathbf{a}))$ for each attribute instance $(n_\mathbf{a}, v, t)$ attached to $\mathbf{c}$. In addition, we use `rdfs:isDefinedBy` to indicate an authoritative resource $\rho(p_\mathbf{c})$ semantically defining the component container $\nu(n_\mathbf{c})$. We detail on $\rho(p_\mathbf{c})$ in the next section.

$$③ \quad \frac{(n_\mathbf{a}, v, t) \in \mathbf{A}_{\mathbf{c},\mathbf{e}}}{\begin{array}{l} \nu(n_\mathbf{a})\ \texttt{rdf:type ldp:RDFResource .} \\ \nu(n_\mathbf{a})\ \texttt{dct:identifier} \text{ “}n_\mathbf{a}\text{”}\texttt{^^xsd:String .} \\ \nu(n_\mathbf{a})\ \texttt{dct:isPartOf}\ \nu(n_\mathbf{c})\ \texttt{.} \\ \nu(n_\mathbf{a})\ \texttt{rdf:value} \text{ “}\mu(v)\text{”}\texttt{^^}\nu(t)\ \texttt{.} \end{array}}$$

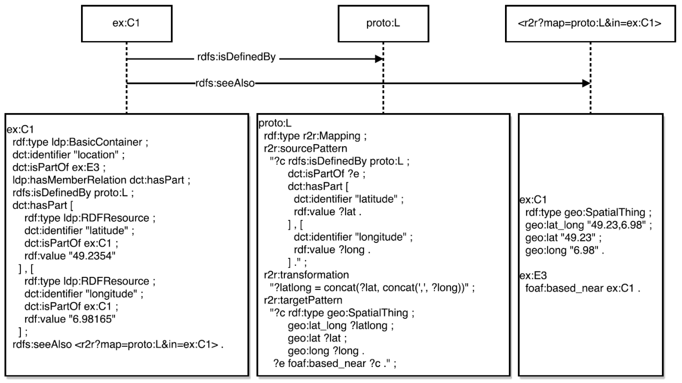
③ Each attribute instance $(n_\mathbf{a}, v, t) \in \mathbf{A}_{\mathbf{c},\mathbf{e}}$ is represented by a `ldp:RDFResource` with `IRI` $\nu(\mathbf{n_a})$. This attribute resource uses `dct:isPartOf` to indicate its containing component container $\nu(n_\mathbf{c})$. The triple $(\nu(n_\mathbf{a}), \texttt{rdf:value}, \text{“}\mu(v)\text{”}\texttt{^^}\nu(t))$ encodes the attribute's current value v and type t as a typed literal “v”^^$\nu(t)$ (cf. ④).

④ Since the RDF datatype abstraction is compatible with XML Schema, we rely on the data type support between an ECA runtime enviroment and XML Schema Types for datatype conversion. Given an attribute $(n_\mathbf{a}, v, t) \in \mathbf{A}_{\mathbf{c},\mathbf{e}}$, we denote by $\nu(t)$ the datatype `IRI` of the RDF-compatible XSD type corresponding to t. The lexical form $\mu(v)$ may be any lexical form, ie. a Unicode string in Normal Form C, from $\nu(t)$'s lexical space that represents the same value as v. Extensions that handle domain-specific or user-defined datatypes beyond the RDF-compatible XSD types are expected to behave as outlined here.

V. Augmenting Semantics

Rules ① to ④ provide a structural mapping from ECA runtime objects to Web resources described using the LDP vocabulary. Our structural mapping is generic and auto-generatable, but so far it only conveys little application-specific semantics.

A domain expert tasked with semantic augmentation thus requires support for specifying expressive RDF mappings



```
ex:C1
  rdf:type ldp:BasicContainer ;
  dct:identifier "location" ;
  dct:isPartOf ex:E3 ;
  ldp:hasMemberRelation dct:hasPart ;
  rdfs:isDefinedBy proto:L ;
  dct:hasPart [
     rdf:type ldp:RDFResource ;
     dct:identifier "latitude" ;
     dct:isPartOf ex:C1 ;
     rdf:value "49.2354"
   ] , [
     rdf:type ldp:RDFResource ;
     dct:identifier "longitude" ;
     dct:isPartOf ex:C1 ;
     rdf:value "6.98165"
   ] ;
  rdfs:seeAlso <r2r?map=proto:L&in=ex:C1> .
```

```
proto:L
  rdf:type r2r:Mapping ;
  r2r:sourcePattern
    "?c rdfs:isDefinedBy proto:L ;
        dct:isPartOf ?e ;
        dct:hasPart [
          dct:identifier "latitude" ;
          rdf:value ?lat .
        ] , [
          dct:identifier "longitude" ;
          rdf:value ?long .
        ] ." ;
  r2r:transformation
    "?latlong = concat(?lat, concat(',', ?long))" ;
  r2r:targetPattern
    "?c rdf:type geo:SpatialThing ;
        geo:lat_long ?latlong ;
        geo:lat ?lat ;
        geo:long ?long .
     ?e foaf:based_near ?c ." ;
```

```
ex:C1
  rdf:type geo:SpatialThing ;
  geo:lat_long "49.23,6.98" ;
  geo:lat "49.23" ;
  geo:long "6.98" .

ex:E3
  foaf:based_near ex:C1 .
```

Figure 2. Structural mapping and semantic augmentation of a “Location” component (cf. Figure 1).

that enable fine-grained term correspondences, literal transformations and structural graph transformations at dataset-level. Ideally, these RDF mappings should be dereferencable and executable, self-contained and interoperably represented as RDF triples. Natural candidates for expressing and executing such RDF mappings are SPIN SPARQL[6], RIF in RDF[7], the LDIF framework[8] or the R2R framework[9].

In the scope of this paper and without loss of generality, we describe and publish such RDF mappings using the R2R Mapping Language [22]. Similar to SPARQL CONSTRUCT queries, a `r2r:Mapping` (cf. Figure 2(b)) has a `r2r:sourcePattern`, `r2r:transformation` and a `r2r:targetPattern`. The source pattern is matched against data generated from rules ① to ④ (cf. Figure 2(a)) and produces a set of variable bindings. Transformations define how variable bindings are transformed before being inserted into the target pattern. The target pattern is used to produce the triples resulting from the `r2r:Mapping` (cf. Figure 2(c)).

Rule ② uses `rdfs:isDefinedBy` to indicate an authoritative resource $\rho(p_\mathbf{c})$ defining all instances of a component prototype $p_\mathbf{c} \in \mathcal{P}_\mathbf{C}$. Hence, a domain expert can publish her RDF mapping under $\rho(p_\mathbf{c})$ and make it discoverable for Linked Data clients. By retrieving a representation of $\rho(p_\mathbf{c})$, a Linked Data client will be instructed on how to locally render additional application-specific RDF triples. Note that execution of a RDF mapping may also be delegated to a suitable Linked Data Service (LIDS) [23]. We suggest `rdfs:seeAlso` (cf. Figure 2(a)) to indicate the respective LIDS invocation `IRI`.

VI. Conclusion

This paper presents a generic and auto-generatable structural mapping between Entity-Component-Attribute (ECA)

[6]https://www.w3.org/Submission/2011/SUBM-spin-sparql-20110222/
[7]https://www.w3.org/TR/rif-in-rdf/
[8]http://ldif.wbsg.de/
[9]http://wifo5-03.informatik.uni-mannheim.de/bizer/r2r/

runtimes and the W3C Linked Data Platform. Building upon this basic level of structural interoperability, we explain how domain experts may declaratively specify and publish expressive RDF mappings in order to convey the application-specific semantics of the respective ECA runtime objects. By executing the published RDF mappings, a Linked Data client is instructed on how to semantically interpret the dynamically exposed ECA runtime objects. A prototype implementation of the presented approach is available at https://github.com/tospie/eca2ld.

ACKNOWLEDGMENT

The work presented in this paper received funding from the European Union's project FI-NEXT under grant agreement no. 732851, and by the Federal Ministry of Education and Research of Germany in the project Hybr-iT under support code 01IS16026A.

REFERENCES

[1] P. Mohagheghi and R. Conradi, "An empirical study of software change: origin, acceptance rate, and functionality vs. quality attributes," in *Empirical Software Engineering, 2004. ISESE'04. Proceedings. 2004 International Symposium on*. IEEE, 2004, pp. 7–16.

[2] H. Mannaert, J. Verelst, and K. Ven, "Towards evolvable software architectures based on systems theoretic stability," *Software: Practice and Experience*, vol. 42, no. 1, pp. 89–116, 2012.

[3] A. Zanella, N. Bui, A. Castellani, L. Vangelista, and M. Zorzi, "Internet of things for smart cities," *IEEE Internet of Things journal*, vol. 1, no. 1, pp. 22–32, 2014.

[4] D. Wiebusch and M. E. Latoschik, "Decoupling the entity-component-system pattern using semantic traits for reusable realtime interactive systems," in *Software Engineering and Architectures for Realtime Interactive Systems (SEARIS), 2015 IEEE 8th Workshop on*. IEEE, 2015, pp. 25–32.

[5] B. Moltchanov and O. R. Rocha, "A context broker to enable future IoT applications and services," in *Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT), 2014 6th International Congress on*. IEEE, 2014, pp. 263–268.

[6] S. P. Mohanty, U. Choppali, and E. Kougianos, "Everything you wanted to know about smart cities: The internet of things is the backbone," *IEEE Consumer Electronics Magazine*, vol. 5, no. 3, pp. 60–70, 2016.

[7] D. ard and V. Trifa, "Towards the web of things: Web mashups for embedded devices," in *Workshop on Mashups, Enterprise Mashups and Lightweight Composition on the Web (MEM 2009), in proceedings of WWW (International World Wide Web Conferences), Madrid, Spain*, vol. 15, 2009.

[8] D. Pfisterer, K. Romer, D. Bimschas, O. Kleine, R. Mietz, C. Truong, H. Hasemann, A. Krller, M. Pagel, M. Hauswirth, M. Karnstedt, M. Leggieri, A. Passant, and R. Richardson, "Spitfire: toward a semantic web of things," *IEEE Communications Magazine*, vol. 49, no. 11, pp. 40–48, November 2011.

[9] T. Heath and C. Bizer, "Linked data: Evolving the web into a global data space," *Synthesis lectures on the semantic web: theory and technology*, vol. 1, no. 1, pp. 1–136, 2011.

[10] R. Schubotz, C. Vogelgesang, A. Antakli, D. Rubinstein, and T. Spieldenner, "Requirements and specifications for Robots, Linked Data and all the REST," in *Proceedings of 2nd Workshop on Linked Data in Robotics and Industry 4.0. (LIDARI-2017), located at Semantics 2017, Amsterdam, Netherlands*. CEUR, 2017.

[11] C. Bizer and A. Seaborne, "D2RQ-treating non-RDF databases as virtual RDF graphs," in *Proceedings of the 3rd international semantic web conference (ISWC2004)*, vol. 2004. Proceedings of ISWC2004, 2004.

[12] S. Auer, S. Dietzold, J. Lehmann, S. Hellmann, and D. Aumueller, "Triplify: light-weight linked data publication from relational databases," in *Proceedings of the 18th international conference on World wide web*. ACM, 2009, pp. 621–630.

[13] F. Michel, J. Montagnat, and C. Faron-Zucker, "A survey of RDB to RDF translation approaches and tools," Ph.D. dissertation, I3S, 2014.

[14] L. F. de Medeiros, F. Priyatna, and O. Corcho, "Mirror: Automatic R2RML mapping generation from relational databases," in *International Conference on Web Engineering*. Springer, 2015, pp. 326–343.

[15] F. Michel, L. Djimenou, C. Faron-Zucker, and J. Montagnat, "Translation of relational and non-relational databases into RDF with xR2RML," in *11th International Confenrence on Web Information Systems and Technologies (WEBIST'15)*, 2015, pp. 443–454.

[16] E. Oren, B. Heitmann, and S. Decker, "ActiveRDF: Embedding Semantic Web data into object-oriented languages," *Web Semantics: Science, Services and Agents on the World Wide Web*, vol. 6, no. 3, pp. 191–202, 2008.

[17] G. Hillairet, F. Bertrand, J. Y. Lafaye *et al.*, "Bridging EMF applications and RDF data sources," in *Proceedings of the 4th International Workshop on Semantic Web Enabled Software Engineering, SWESE*, 2008.

[18] T. Alatalo, "An entity-component model for extensible virtual worlds," *IEEE Internet Computing*, vol. 15, no. 5, pp. 30–37, 2011.

[19] T. Dahl, T. Koskela, S. Hickey, and J. Vatjus-Anttila, "A Virtual World Web Client utilizing an Entity-Component model." in *NGMAST*. IEEE, 2013, pp. 7–12.

[20] T. Spieldenner, M. Guldner, S. Byelozyorov, and P. Slusallek, "FiVES: An aspect-oriented Virtual Environment Server," in *Proceedings of the 2017 International Conference on Cyberworlds. International Conference on Cyberworlds (CyberWorlds-2017), September 20-22, Chester, United Kingdom*. IEEE Xplore, 2017.

[21] A. P. Sheth, "Changing focus on interoperability in information systems: from system, syntax, structure to semantics," in *Interoperating geographic information systems*. Springer, 1999, pp. 5–29.

[22] C. Bizer and A. Schultz, "The R2R framework: Publishing and discovering mappings on the Web." *COLD*, vol. 665, 2010.

[23] S. Speiser and A. Harth, "Integrating linked data and services with linked data services," in *Proceedings of the 8th Extended Semantic Web Conference on The Semantic Web: Research and Applications - Volume Part I*, ser. ESWC'11. Berlin, Heidelberg: Springer-Verlag, 2011, pp. 170–184.