Comparing XML Files with a DOGMA Ontology to Generate *Ω*-RIDL Annotations

Nadejda Alkhaldi and Christophe Debruyne

Semantics Technology and Applications Research Lab,
Vrije Universiteit Brussel, Brussels, Belgium
{nadejda.alkhaldi,christophe.debruyne}@vub.ac.be

Abstract. To facilitate the process of annotating data in the DOGMA ontology-engineering framework, we present a method and tool for semi-automatic annotation of XML data using an ontology. XML elements are compared against concepts and their interrelations in the ontology using various metrics at different levels (lexical level, semantic level, structural level, etc.). The result of these metrics are then used to propose the user a series of annotations from XML elements to concepts in the ontology, which are then validated by that user. Those annotations - expressed in *Ω*-RIDL - are then used to transform data from one format into another format. In this paper, we demonstrate our approach on XML data containing vendor offers in the tourism domain, more precisely holiday packages.

Key words: Annotation, Ontology/XML Matching

1 Introduction

In the Community Driven Request for Proposals (COMDRIVE) [4] project, a platform was developed in which a potential customer's purchase intent is matched against existing vendor offers. Those offers - described in XML - came from autonomously developed information systems and were therefore different in structure and the kind of content they provided. In order for such a platform to be successful, a common from understanding of the domain from both the vendor and customer's side needs to be established and captured in an ontology, a [formal,] explicit specification of a [shared] conceptualization [7].

In the DOGMA [13] ontology engineering framework a special controlled natural language called *Ω*-RIDL [16, 15] - both which will be explained later on in Section 2 - was developed to annotate data sources with concepts and relations from the ontology. Those annotations can then be used to, for instance, transform data from one format into another format or consult the data through the ontology. In the COMDRIVE RFP project, annotation was done manually. However, this process requires the person annotating the data to examine the structure and content of the data and compare it to the ontology. In this paper, we present a method and tool, presented in Sections 3, to facilitate the annotation process by having an agent examine the data and suggest the user annotations.

The user will thus have a hint to what the meaning of his data *could* be, after which he can decide to keep, edit or extend the generated annotation. The tool will thus be able to aid the user in *deciding* on a particular annotation (or commitment to) to the ontology. The annotations will be the result of *matching* the structure and content of the XML files against the concepts and relations in the ontology. We demonstrate our method and tool using the data used by the project in Section 4 before concluding and presenting our future directions in Section 5.

2 Background

In this section we first describe the DOGMA ontology engineering framework and commitments described in *Ω*-RIDL. We then go over to a brief state-of-the-art on various matching techniques relevant for this paper.

2.1 Developing Ontology Guided Methods and Applications

DOGMA [13] is an ontology approach having some characteristics that make it different from traditional ontology approaches such as its groundings in the linguistic representations of knowledge and the methodological separation of the domain- and application-conceptualization [12]. The knowledge building blocks - called *lexons* [9] - only need to express "plausible" facts (as perceived by the community of stakeholders) in order to be entered into the *Lexon Base*, a repository containing large sets of such lexons. A lexon is formally described as a 5-tuple $(\gamma, t_1, r_1, r_2, t_2)$, where γ is an abstract *context identifier* pointing to a resource such as a document on the Web or the community it originated from. The term t_1 plays the role of r_1 on term t_2 and t_2 plays the role of r_2 on t_1. The context identifier is assumed to identify unambiguously (to human users at least) the concepts denoted by the term and role labels.

The *Commitment Layer* contains ontological commitments made by users that use a selection of lexons to annotate data sources and specify constraints defining the use of the concepts in the ontology. DOGMA distinguishes two types of ontological commitments: *community commitments* and *application commitments*. The first denotes a meaningful selection of lexons and constraints that capture well the intended semantics of a domain. A community commitment corresponds with the ontology shared across stakeholders. In our approach, we use the ontology developed with a collaborative ontology engineering approach built around DOGMA, called DOGMA-MESS [3]. Details on the construction of this ontology have been reported elsewhere [4]. The latter extends the community commitments with mappings describing how one individual application commits to the ontology and are made with a special controlled natural language that we will explain in the following Section.

DOGMA is a fact-oriented modeling approach that is communication oriented, which means that the community comes to a shared understanding of the domain by communicating facts (lexons) rather then concepts, attributes and relations. This differs from other frame-oriented formalisms such as OWL in which one can choose to model whether a particular relation in the world becomes an attribute or entity (denoting the relation) at design time. When a DOGMA ontology is transformed into OWL, these decisions are based on the constraints on those facts. This reduced to complexity of creating ontologies for the users.

2.2 Ω-RIDL

Ω-RIDL [16, 15] allows users to describe how instances of concepts and their interrelations inside an information system commit to an ontology by: (i) selecting lexons from the Lexon Base, (ii) constraining over those lexons to capture well the intended meaning of that system and (iii) providing mappings between application symbols (e.g., XPaths[1] or fields in a table) and terms in the selected lexons. Examples of such constraints are frequency and totality constraints. The taxonomic relation can also be specified in a separate section (the default relation for taxonomy is "is a/subsumes"). It allows subtypes of concepts to inherit the relations of their supertype. Ω-RIDL provides a controlled natural language by allowing constraints and mappings to be expressed in terms of the natural language sentences created with lexons. An example of such a commitment can be found in Fig. 1 below. The application symbols mapped in this example are XPaths expressions and come from one of the XML files provided by vendors.

```
BEGIN SELECTION
''Product Community'' Product with / of Name
''Product Community'' Product with / of Description
''Tour Operator Community'' Holiday Package is a / subsumes Product
...
END SELECTION
BEGIN CONSTRAINTS
Holiday Package is identified by Name.
Holiday Package has exactly 1 Name.
Holiday Package has at most 1 Description.
...
END CONSTRAINTS
BEGIN MAPPINGS
map ''/items/item'' on Holiday Package.
map ''/items/item/title'' on Name of Holiday Package.
map ''/items/item/description'' on Description of Holiday Package.
...
END MAPPINGS
```

Fig. 1. Example of a commitment for a particular application showing pieces of the three parts: selection, constraints and annotations (or application symbol mappings).

[1] http://www.w3.org/TR/xpath/

2.3 Related Work on Matching Techniques

In this section we will present a brief survey on schema and ontology matching techniques. We have looked at several existing techniques for schema matching [8, 19], database schema matching [5], matching XML schema to OWL [1], aligning and matching OWL ontologies [14, 2], a method that detects similarities between two versions of an XML file [17], and a general framework for computing semantic similarity [11]. Some of those techniques use a linguistic match [8, 19, 2, 11] using a thesaurus [8, 2], a lightweight domain ontology [19]. [11] uses WordNet [6]. [8, 19, 1, 14, 2, 17] also take into account the structures of the documents to be compared. Machine learning techniques are used by [19, 5, 14]. [8, 5] also compare the data types and [19, 5, 14] even look at the similarity of values. [8, 1, 17] look at the keys (and referential constraints) of the XML schemas, [5] looks at the keys and integrity constraints in the base. [1] is discovering semantic isomorphic tree in an ontology based on the structure and also mentioned to look at the cardinality constraints of the schema and ontology.

3 Method

In our particular case, we start from an XML file without an XML schema and a DOGMA ontology. Since we have no XML schema, we cannot rely on the constraints, structure and data types typically provided by such a schema. Because of the above-mentioned strict separation of schema and instances in the DOGMA Ontology Engineering framework, we can only map elements of the XML file to facts in the ontology. Moreover, the goal is to annotate the instances in the XML *with* the DOGMA ontology. Based on these constraints, we decide to use string matching techniques to overcome spelling errors or slight variations in words (e.g., UK English vs. US English, concatenation of words, etc.), linguistic matching to look at the linguistic features of concepts in both schemas (e.g. to discover synonyms) and a structural match to detect the relation between two nodes that are not necessarily directly related to each other. We will now briefly describe how these matches can be used to generate Ω-RIDL mappings. We furthermore need a way to describe the type of a XML element based on the contents of one its attributes or contents of tags. For instance, `<facility type=‘‘bar’’>` should be mapped onto Bar, but not onto `Facility`, of which `Bar` is a subtype.

Fig. 2 gives the general idea of our Ω-RIDL mapping generator. It takes as input the lexons of the DOGMA ontology and an XML file. The XML file *must be related* to the ontology's application domain in order to obtain meaningful results. The user also defines a series of weights and thresholds to select to most successful mappings. The external resources are thesauri, reference ontologies or WordNet. In essence, the framework is extensible enough to plug in additional matching strategies. In this paper, we only took into account WordNet.

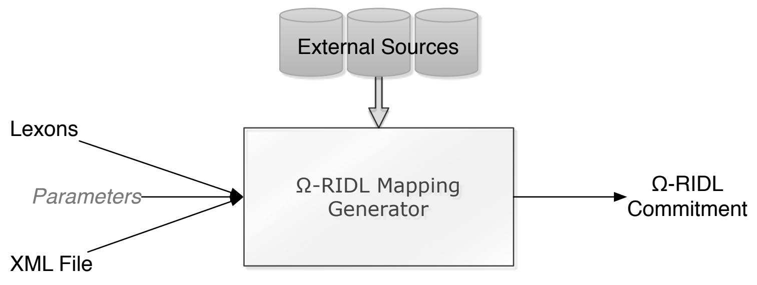

Fig. 2. The method. Users give an XML file and a DOGMA ontology as well as a series thresholds and weights for the particular heuristics.

3.1 Heuristics

In this section, we describe the different heuristics applied by Ω-RIDL mapping generator.

1. **Element Match** This heuristic gives a similarity score based on a string metric between two labels, one of the XML (tag or attribute) and a concept in the ontology. We chose Jaro-Winkler [18] for its tolerance for spelling mistakes and variations over different dialects. As the interval of the scores lies between 0 and 1, the scores need not to be normalized.
2. **Linguistic Match** uses WordNet to give similarity score for XML and ontology concepts' names exploiting their linguistic features. This heuristic returns a score of 1.0 if both names are synonyms. 0.8 is returned if name of an ontology concept is a hyponym (narrower term) or a hypernym (broader term) of XML concept's name. Those scores are based on [2].
3. **Content Match** looks at the text inside XML tags and attributes (the previous two heuristics look at the label of XML tags and attributes) and matches those to the terms in the ontology using a string match.

When all those matches are applied, Ω-RIDL mapping generator computes their weighted mean based on a score that every match gives and its assigned weight. After that the fourth heuristic, structural match, is applied.

4. **Structural Match** adjusts the previously computed weighted means by looking to the structure of both the ontology graph and XML-tree. If two nodes from the XML and ontology graph are similar and have similar parents, then those nodes obtain a score of 1.0 for this heuristic. If the nodes are similar and do not have similar ancestors, we go up in the XML-tree's hierarchy and we look at the subtypes of the parent in the ontology graph. The ancestor of the XML node is compared to the subtypes and if a match between those two is found, the taxonomic relation is kept as a reference. A maximum number of steps over the XML-tree is specified by the user, and with every step the similarity score decreases by 0.1. If a match were not directly found, but after two steps, for example, the score would be 0.8. Fig. 3 depicts this process graphically.

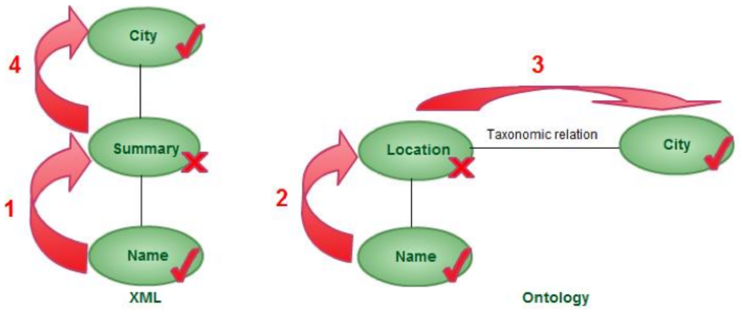

Fig. 3. The structural match traverses both graphs upwards to find common ancestors. The XML contained the name of a city inside a summary tag, whereas locations have names and city is a subtype of location in the ontology. After retrieving the parents of "Name" in both graphs (1, 2), we see they do not correspond. We then look for the parent of "Summary" in the XML graph (3) and compare it to the subtypes of "Location" in the ontology graph.

To summarize: using an XML and a DOGMA ontology, a series of mapping scores are calculated based on element, linguistic and content match. Those scores are then refined using the structural match. The refined scores are then compared against a threshold to produce the *Ω*-RIDL mappings.

4 Experiment

We tested our method and tool using the data obtained by the COMDRIVE RFP project. The data contains information about holiday packages in the winter sports domain. Those ontologies were developed in a modular way: a "general purpose" product ontology was extended with concepts suited for (winter sports) holiday packages [4]. Fig. 4 show a couple of such lexons. The structure of the XML file we are going to use is shown in Fig. 5. In this figure, the leaf nodes represent the text nodes in the XML.

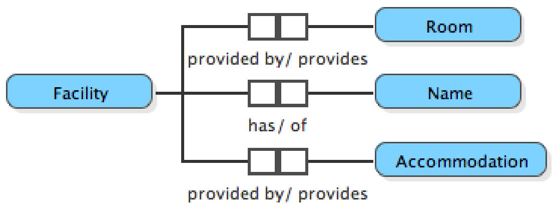

Fig. 4. Some lexons describing domain knowledge in the tourism domain. These lexons describe that both rooms and accommodations can have facilities, which in turn have a name.

When a user runs the *Ω*-RIDL mapping generator, the tool asks the user to provide a DOGMA ontology and XML file. The user then selects the heuristics to be used and assigns a weight to every heuristic. The user furthermore chooses

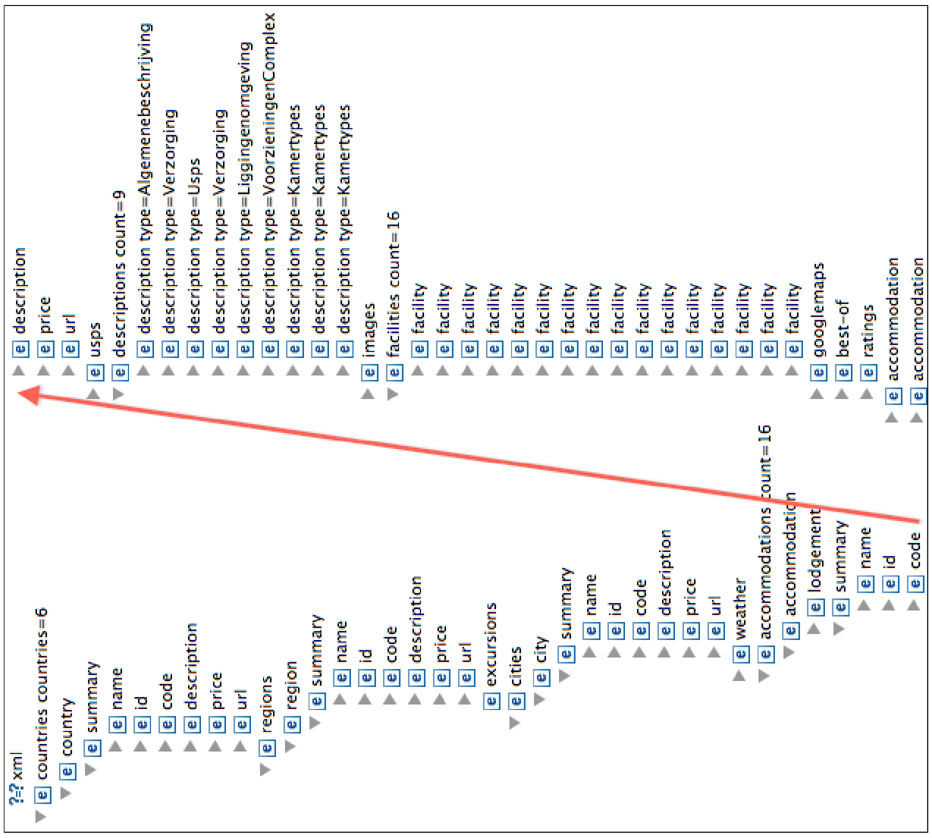

Fig. 5. Structure of the XML file used in our demonstration. Leaf nodes in this tree represent text nodes in the XML.

a depth for the structural match, and specifies thresh. The system takes those parameters and starts the mapping process. A series of mappings is then generated and used to output a selection and mappings (as described in section 2.2). The user can then use the generated mappings to *get an idea* how his application can commit to the ontology. In the example below, you can see that are two proposals for mapping the "region" tag: one to the ski area (destination of a holiday package) and on to the region. In the ontology, ski area is a subtype of region. As the agent cannot decide, the user should choose the most appropriate mapping (the last one, as it contains the ski area of all accommodations of all cities in that region). The result of content matching is a predicate to select elements in an XML that fulfill a certain condition, e.g., tags containing the word "Bar" can denote instances of Bar (a subtype of facility).

```
map ''/countries/country/sumary/code'' on
    Code identifies / identified by Commodity.
map ''/countries/country/regions/region'' on Region.
map ‘‘/countries/country/regions/region’’ on
    Ski Area destination of / with destination Holiday Package.
map ‘‘/countries/country/regions/region/cities/city’’ City.
map ‘‘/countries/country/regions/region/cities/city/accommodations/
    accommodation/facilities/facility[text()=‘Bar’]’’ on Bar.
map ‘‘/countries/country/regions/region/cities/city/accommodations/
    accommodation/facilities/facility[text()=‘Sauna’]’’ on Sauna.
map ‘‘/countries/country/regions/region/cities/city/accommodations/
    accommodation/googlemaps/latitude’’ on Latitude of / with Location.
map ‘‘/countries/country/regions/region/cities/city/accommodations/
    accommodation/googlemaps/longitude’’ on Longitude of/with Location.
```

When a user chooses an inappropriate combination of weights and threshold, however, the system proposes quite a few mappings that are making no sense. Examples of such mappings are shown below. Future work will consist of looking at appropriate combinations of weights and threshold for which more experiments and data will be needed. In the following mappings, we see how a description of a region is wrongly accepted as a description of an RFP and the name of a city to the name of a facility. The third mapping, however, is more interesting. The ontology contains the fact that commodities have prices. The price of a region in the XML however, contains the lowest price for accommodation in that region. The structural match thus also needs to be revisited to cope with such cases.

```
map ‘‘/countries/country/regions/region/summary/description’’ on
    Description of / has RFP.
map ‘‘/countries/country/regions/region/cities/city/name’’ on
    Name of/has Facility.
map ‘‘/countries/country/regions/region/summary/price’’ on
    Price of / has Commodity.
```

In this experiment, the four heuristics were able to tackle the problems described in Section 3 (variation is spelling, synonyms, identifying the type based on the textual content and choosing an appropriate type based on the structure). The weights and threshold, however, needed to be appropriately chosen in order to come up with good results. For this dataset, using 0.5 or 0.6 for the overall threshold value seemed to do well. If the threshold is higher, some important mappings are lost (especially the ones constructed by the content match because they most likely get a score of 0.0 from the other matches). For that reason we also advise to assign the content match - if used - quite an important weight. Structural match does not have a weight because it is used to adjust the final score given to every candidate match. There is also a special threshold for the element match. We recommend using at least 0.7 because element match can give a relatively high score for words that are very different (e.g., 0.55 when comparing “Accommodation” with “Military airport” and 0.6 for “Accommodation” with “Date and Time)”. But we did not notice any case where it gave 0.7 or more for irrelevant words.

5 Conclusions

In this paper, we presented a method and tool for semi-automatically annotating XML XPaths to concepts in a DOGMA ontology using Ω-RIDL. For this, we defined four heuristics, each with a different purpose. Element match compares labels of XML tags and attributes against labels of ontology concepts. Linguistic match can map words linguistically related to each other (synonyms, hypernym, hyponym, etc.). Content match looks to the text of the tags and attributes in XML and helps to produce more accurate mappings. Structural match looks at the surrounding concepts or a particular concept (n deep) to determine the correct relation in the ontology to be mapped with.

Even though the tool generates quite a few false positives, which depends on the weights and thresholds chosen, the tool does give the user an idea about the kind of data in the XML file. The tool can thus be used to guide the user in the manual process of annotating the XML file's structure by means of, for example, an autocompletion user interface element in the editor.

Future work is threefold. Firstly, we need extra experiments to determine an appropriate combination of weights and thresholds. We furthermore only took into account the mappings of a commitment. Secondly, even though we do not rely on an XML schema from which some constraints such as mandatoriness can be deduced, we can look at the nodes and the contents of those nodes to infer similar constraints. If a particular tag has exactly one other tag every time it occurs, a mandatory and uniqueness constraint can be proposed to the user.

And thirdly, we mainly focused at the labels of both XML and ontology concepts. Whenever two relations between two concepts occurs in the ontology and two XML concepts match relatively well to those concepts, both relations are proposed and it's up to the user to decide what to keep. Sometimes, however, the labels of the XML tags contain enough information to choose the appropriate relation, e.g., the XML tag `<startDate>` can be matched with the concept `Date` playing the role `start of` on another concept in the DOGMA ontology.

References

1. An, Y., Borgida, A., Mylopoulos, J.: Constructing complex semantic mappings between xml data and ontologies. In: International Semantic Web Conference: Gil, Y., Motta, E., Benjamins, V.R., Musen, M.A. (eds.). Vol. 3729 of Lecture Notes in Computer Science., Springer (2005) pp. 6–20
2. Castano, S., Ferrara, A., Montanelli, S.: Matching ontologies in open networked systems: Techniques and applications. In: J. Data Semantics V: Spaccapietra, S., Atzeni, P., Chu, W.W., Catarci, T., Sycara, K.P. (eds.). Lecture Notes in Computer Science, Springer (2006) pp. 25–63
3. de Moor, A., De Leenheer, P., Meersman, R.: DOGMA-MESS: A meaning evolution support system for interorganizational ontology engineering. In: Proc. of the 14th Int. Conf. on Conceptual Structures (ICCS 2006). Vol. 4068 of LNCS., Springer (2006) pp. 189–203

4. Debruyne, C., Meersman, D., Baert, M., Hansenne, R.: Community driven request for proposals: Applying semantics to match customer purchase intents to vendor offers. In: Proc. of 7th International Conference on Web Information Systems and Technologies (WEBIST 2011), SciTePress (2011)
5. Dhamankar, R., Lee, Y., Doan, A., Halevy, A.Y., Domingos, P.: iMAP: Discovering complex mappings between database schemas. In: SIGMOD Conference: Weikum, G., König, A.C., Deßloch, S. (eds.), ACM (2004) pp. 383–394
6. Fellbaum, C.: WordNet: An Electronic Lexical Database. MIT Press (1998)
7. Gruber, T.R.: Toward principles of the design of ontologies used for knowledge sharing. International Journal of Human and Computer Studies **43** (1995) pp. 907–928
8. Madhavan, J., Bernstein, P.A., Rahm, E.: Generic schema matching with cupid. In: VLDB: Apers, P.M.G., Atzeni, P., Ceri, S., Paraboschi, S., Ramamohanarao, K., Snodgrass, R.T. (eds.), Morgan Kaufmann (2001) pp. 49–58
9. Meersman, R.: Semantic ontology tools in IS design. In: ISMIS: Ras, Z.W., Skowron, A. (eds.). Vol. 1609 of LNCS., Springer (1999) pp. 30–45
10. Meersman, R., Dillon, T., Herrero, P. (eds.): On the Move to Meaningful Internet Systems, OTM 2010 - Confederated International Conferences: CoopIS, IS, DOA and ODBASE, Hersonissos, Crete, Greece, October 25-29, 2010, Proceedings, Part II. In: OTM Conferences (2): Meersman, R., Dillon, T., Herrero, P. (eds.). Vol. 6427 of LNCS., Springer (2010)
11. Pirrò, G., Euzenat, J.: A semantic similarity framework exploiting multiple parts-of speech. [10] pp. 1118–1125
12. Spyns, P., Meersman, R., Jarrar, M.: Data modelling versus ontology engineering. SIGMOD Record Special Issue **31 (4)** (2002) pp. 12–17
13. Spyns, P., Tang, Y., Meersman, R.: An ontology engineering methodology for DOGMA. Applied Ontology **3**(1-2) (2008) pp. 13–39
14. Straccia, U., Troncy, R.: oMAP: Combining classifiers for aligning automatically owl ontologies. In: Web Information Systems Engineering: Ngu, A.H.H., Kitsuregawa, M., Neuhold, E.J., Chung, J.Y., Sheng, Q.Z. (eds.). Vol. 3806 of Lecture Notes in Computer Science., Springer (2005) pp. 133–147
15. Trog, D., Tang, Y., Meersman, R.: Towards ontological commitments with Ω-RIDL markup language. In: RuleML: Paschke, A., Biletskiy, Y. (eds.). Vol. 4824 of LNCS., Springer (2007) pp. 92–106
16. Verheyden, P., De Bo, J., Meersman, R.: Semantically unlocking database content through ontology-based mediation. In: Semantic Web and Databases: Bussler, C., Tannen, V., Fundulaki, I. (eds.). Vol. 3372. (2004) pp. 109–126
17. Viyanon, W., Madria, S.K.: Xml-sim-change: Structure and content semantic similarity detection among xml document versions. [10] pp. 1061–1078
18. Winkler, W.E.: The state of record linkage and cur- rent research problems. Statistics of Income Division, Internal Revenue Service Publication R99/04. (1999)
19. Xu, L., Embley, D.W.: Discovering direct and indirect matches for schema elements. In: Database Systems for Advanced Applications, IEEE Computer Society (2003) pp. 39–46