Guidance for Domain Specific Modeling in Small and Medium Enterprises

Henning Agt
Technische Universität Berlin
Einsteinufer 17
10587 Berlin, Germany
henning.agt@tu-berlin.de

Ralf-D. Kutsche
Technische Universität Berlin
& Fraunhofer Institut für Rechnerarchitektur und Softwaretechnik
Einsteinufer 17
10587 Berlin, Germany
ralf-detlef.kutsche@tu-berlin.de

Timo Wegeler
Fraunhofer Institut für Rechnerarchitektur und Softwaretechnik
Kekuléstr. 7
12489 Berlin, Germany
timo.wegeler@first.fraunhofer.de

ABSTRACT

We observe that small and medium enterprises who wish to adopt domain specific modeling techniques do so under different preconditions and with different expectations. In our report, we categorize our observations made in 7 different industrial branches. Further, we present the current state of our solution to provide guidance to both ends of stakeholders involved in a DSM development lifecycle, domain experts and DSL designers. By supporting a DSL development process with semantic knowledge bases and metrics, our goal is to make a DSL development feasible and beneficial through the guidance provided by EXAMINE and within the DIESEL-Framework.

Keywords

Domain Specific Modeling, DSL Metric, DSL Engineering, DSL Guidance, Semantic Knowledge Base, Ontology

1. INTRODUCTION

From the year 2007 to 2010, the R&D project BIZYCLE[1], jointly initiated by TU Berlin and a group of small and medium enterprises (SMEs) in Berlin, was run in order to establish a standardized methodology for model-based integration [20] of heterogeneous distributed software components. The integration methodology was based on multi-level modeling abstractions, automated conflict analysis [3, 2] and connector code generation. One of the biggest achievements of the BIZYCLE project was a solid meta-modeling foundation [1], and a comprehensive toolkit to support the development process.

Learning from these experiences, BIZWARE[2], the follow-up project, was started in 2010, in order to develop a systematic and standardized process of model-based software construction and operation, under the paradigm of domain specific modeling (DSM). Participative modeling between software professionals and domain experts is enabled by dedicated (graphical and textual) domain languages in the given domains.

[1]BIZYCLE was partially funded by BMBF under grant number (Förderkennzeichen) 03WKBB1B.
[2]BIZWARE is partially funded by BMBF under grant number (Förderkennzeichen) 03WKBU01A and 03WKBU02B (TU Berlin) and 03WKBU01B (Fraunhofer FIRST).

BIZWARE research consequently addresses the development of domain specific languages (DSLs) for the given domains, and of a toolkit, based on meta-DSLs, the so-called "BIZWARE model and software factory". Outputs of the factory are software components with a built-in "plug-and-play" mechanism for easier integration, as well as software generating tools.

The development process of such DSLs remains – under industrial restrictions – a complex task. The goal of the research presented in this paper is to investigate and to evaluate potential support of the DSL development process by querying semantic knowledge bases. We introduce a strategy of providing guidance to all participating groups: stakeholders involved in a DSM development lifecycle, domain experts and DSL designers.

2. RELATED WORK

Khalaoui et al. have examined the success factors for domain specific modeling and compiled a list of qualitative criteria with positive and negative impacts [16]. Rodriguez at al. suggest an evaluation management process [23]. They introduce checklists for the syntactic, semantic and pragmatic quality of an artifact. An empirical study measured understandibility time to compare different solutions of a modeling example in UML [10]. White et al. suggest constraint checking before committing changes to a model [29] as a guidance measure for modelers. Lahtinen et al. propose a wizard that guides the creation of a valid model using a task list [18] to help new users unaware of the underlying metamodel. Genero at al. [11] argue that even a small set of simple structural metrics can indicate a model's understandability. Wu et al. measured [30] effort for creating software applications using DSML's and proposes metrics. A language workbench is presented by Kats et al. [15], which is integrated into the Eclipse IDE. Gailly and Poels examined the domain-specific quality of profiled UML diagram variants and demonstrated that domain ontologies help the evaluation process [9].

In a report on technology transfer from academia to industry [4], the evolution of a domain specific language and its interface is analyzed. The practical use of a visual DSL is reported by [14]. In particular, the evolutionary history is described and a supervised evolution process is considered a requirement for successful language development. The iterative character of a DSL development is presented by vanAmstel et al. [27]. Strembeck and Zdun examined [24] activities in a DSL engineering process, which were derived by analyzing industry and research DSL projects.

The use of ontologies in model-driven engineering has been investigated from different viewpoints. Tairas et al. [26] show how manual ontology creation improves the domain analysis phase of DSL development. In our approach, we propose automated querying of semantic knowledge bases to provide modeling suggestions. These knowledge bases are lexical databases (e. g., WordNet [8], FrameNet), ontologies, such as SUMO [21] and Cyc, and automatically constructed knowledge bases (e. g., YAGO [25], DBpedia [5]). We build on the fact that formalized knowledge has significantly increased in the recent years, supported by the field of knowledge harvesting [28] in which large scale knowledge bases are created by automatically extracting facts from semi-structured and unstructured natural language text. Our research work aims to contribute to the connection between model-driven approaches and ontology-driven as it has been analyzed by Guizzardi [12] and Henderson-Sellers [13].

3. BENEFITS OF DOMAIN SPECIFIC MODELING

During the first year of our collaboration, we have analyzed the motivation and perceived benefits of a DSL development among our consortium of 2 academic and 8 industrial partners for the different domains as well as the demands for capabilities of the BIZWARE software factory and DSLs in general. We have surveyed the motivation of our partners for investing into the development of DSLs for specific purposes. All of them develop software themselves and address different industrial domains: health care, finance, publishing, facility management, industrial production, web application development, and system integration.

The first observation made was that the main focus was either on *technical DSLs* addressing so-called horizontal abstractions, or on *business (or: functional) DSLs* providing the so-called vertical abstractions, being intended to be used by domain experts. An example of a technical DSL is the high-level definition of a GUI, an example of a business DSL is the definition of insurance contracts. However, the need to abstract from implementation details is a common motivation and the ultimate goal is to ease the software development process.

As a first benefit was acknowledged that better documentation is achieved even in early phases of the DSL development, where relevant terms are being identified and the discussions contribute to a better understanding. This way, a DSL development can be beneficial even when the DSL it not yet completed, but still in the process of being sketched. The systematic involvement of end users in the development process during the design, implementation and testing phase is considered a necessity. Other strong motivational issues are:

- to improve documentation
- to use design artifacts as first class development artifacts, without interrupting refinement processes
- to be able to better involve customers and end users
- to ease the software configuration
- to ease deployment on different target platforms

Demanded benefits of a DSL development are characterized as:

- effort reduction
- higher levels of abstraction
- improvement of development processes and their organization
- better internal and external communication
- leverage existing documents
- integrate into existing development landscape
- overcome business/technical view mismatches

As the design effort however is difficult to quantify, a standard DSL engineering process is required in order to be able to measure it along its execution. Benefits of better communication are also difficult to measure. Furthermore, methodological support is required in the DSL development processes. In the context of BIZWARE it is provided by TU Berlin and FIRST, adapting to the need of its partners, evaluating existing approaches and DSL workbenches and offering decision support, e.g. whether a DSL development makes sense according to qualitative criteria. We are also investigating how metrics can help to support the design process, especially in finding a suitable solution for a concrete DSL design.

4. WORK IN PROGRESS

DSL development requires both expertise in language engineering as well as detailed knowledge of the application domain [19]. While DSL engineers usually are capable of the language engineering tasks, such as creating metamodels, finding a suitable level of abstraction and modularity, they are constantly confronted with new problem domains. One of the most important early-stage activities of language creation is the definition of an abstract syntax model [17]. In the analysis phase of a DSL development, it is essential to identify existing concepts and relations of the domain and to choose the relevant ones for the language. Additionally, using proper terms for classes, attributes and associations in the metamodel of a DSL is important for DSL users.

However, a domain expert may be unaware of common metamodeling techniques or lack interest or understanding of using abstract symbols for modeling. To explore a suitable representation, which may be of textual or graphical nature, it is required to assess the domain expert's preferences and known concepts. This may sound trivial, but we identified that a systematical approach to identify appropriate representations is lacking.

4.1 Modeling Guidance for DSL Design

In this section, we propose an approach for guidance of DSL engineers during DSL analysis and design phase using automated knowledge acquisition on semantic knowledge bases, such as ontologies and lexical databases. This method is part of our DIESEL-Framework and is being implemented in the EXAMINE-System (**Ex**tr**a**cting Infor**m**ation for Software Language Eng**ine**ering). The EXAMINE-System incorporates the *Extractor* component and the *Model Advisor* service (shown in Figure 1) that provide modeling suggestions and domain exploration as novel features for language workbenches. In the following we describe the suggested method in more detail.

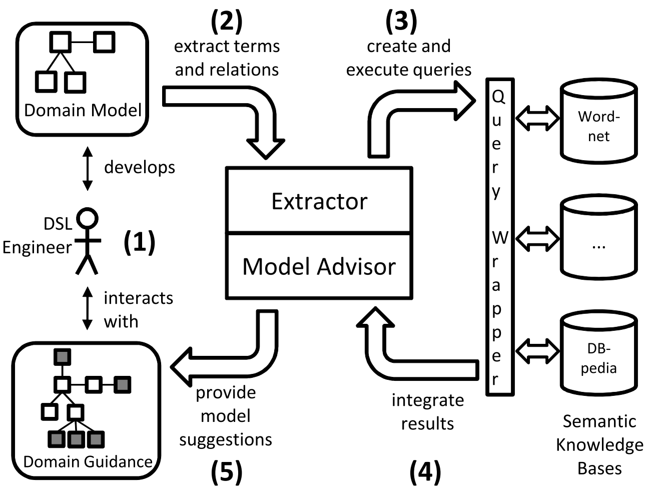

Figure 1: Using Semantic Knowledge Bases to Provide Modeling Assistance for Domain Specific Modeling

(1) The DSL engineer develops the domain model of a new language. For example, he creates an abstract syntax model starting with a few classes using a MOF-based metamodeling environment such as Eclipse Modeling Framework. Our goal is to support this activity with suggestions of semantically related concepts that might be included in the abstract syntax. At the moment, we concentrate on taxonomic relations (e. g., represented by generalizations/specializations) and part-whole relations (e. g., represented by aggregations) between concepts.

(2) The Extractor determines terms of the class names contained in the domain model. We also plan to process all other kinds of terms, such as attribute and association names. In order to start gathering domain knowledge a few initial terms are sufficient.

(3) There exists a variety of semantic knowledge bases containing formalized domain and commonsense knowledge that can be exploited to support a domain specific modeling process. Based on the extracted terms we create and execute queries formulated in SPARQL [22] language against lexical databases (e. g., WordNet [8], FrameNet) manually created ontologies (e. g., SUMO [21], Cyc) and automatically constructed knowledge bases (e. g., YAGO [25], DBpedia [5]). The queries ask for semantically related concepts (e. g., hypernyms, meronyms, synonyms) of the terms of the model.
Although most of the knowledge bases use standard knowledge representation languages (e. g., RDF, OWL), they incorporate different schemas or data models. The query wrapper takes care of translating the queries into the respective destination format with appropriate namespaces and descriptors.

(4) SPARQL queries on these knowledge bases retrieve a set of subject-predicate-object triples (e. g., *Scientist is-hyponym-of Person*) that form a result graph. Queries on the same knowledge base can be easily merged. We intend to use standard ontology matching techniques [7] to integrate the query results of different knowledge bases. Furthermore, a lot of research related to knowledge base integration is carried out in the area of Linked Open Data [6].

(5) In the last step of our approach a comprehensible domain visualization is created. The result graph is transformed into a MOF-based language to be able to build graphical representations similar to UML class diagrams. The DSL engineer then can take modeling decisions based on the presented visualization. We also plan to provide an explorative interface in which the engineer can browse through the acquired knowledge.

4.2 Use Case in Healthcare Domain

In this section we demonstrate the EXAMINE-System with a use case in healthcare domain. Two of our industrial partners develop enterprise application integration solutions in the area of hospitals and healthcare providers. In order to adopt domain specific modeling techniques for their needs, it is important for them to receive support in DSL development. We show how an initial domain model can be complemented and improved by querying WordNet [8] and how the results are used to generate a domain excerpt for guidance.

WordNet[3] is a popular lexical database for the English language developed at Princeton University. It consists of so called *synsets* that group *words* sharing the same sense. Thus, each synset expresses a semantic concept for nouns, verbs, adjectives and adverbs, respectively. Ambiguous words that have several meanings belong to multiple synsets. A *gloss* describes each synset by an additional comment. WordNet models several semantic relations between synsets, such as hyponymy/hypernymy (is-a relations, e. g., *an engineer is a person*), different types of holonymy/meronymy (part-whole relations, e. g.,*a nose is part of a face*), synonyms, instance knowledge and others. We use WordNet 3.0 that contains 155.287 words organized in 117.659 synsets of which 82.115 are noun synsets. Currently, we concentrate on nouns and their relations for modeling guidance but we also plan to exploit verb synsets for associations.

In our use case, the task of the industrial partner was to create a classification of medical examinations and hospital types to support the discussion with his customer on which procedures and underlying legacy systems should be integrated with his hospital information system. From his own knowledge and first discussion with the customer, the industrial partner created an initial domain model containing

[3]http://wordnetweb.princeton.edu/perl/webwn

three types of medical examinations as shown in Figure 2. Although the model is far from complete and still can be refined with his knowledge, the EXAMINE-System is already able to provide modeling guidance by retrieving additional domain knowledge for the terms of the initial model.

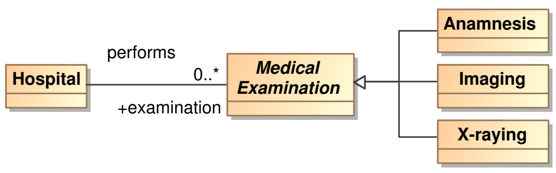

Figure 2: Initial Domain Model of Medical Examinations

For each of the terms the WordNet word sense is determined in case of ambiguity (polysemy). We employ the following heuristics: Firstly, we do a key word search with the remaining terms of the domain model on the descriptions of all possible word senses. Secondly, we query connected terms of each possible word sense and compare these terms with the other terms of the model. As a fallback solution we use the synset that has the highest frequency score (provided by WordNet).

After having determined the correct word senses, for each of the synsets queries are generated to retrieve directly related hyponyms and hypernyms. Currently, we use the RDF version[4] and SPARQL endpoint of WordNet 3.0 provided by VU University Amsterdam to execute the queries. Listing 1 shows the queries for determining sub-concepts of *Hospital* (lines 6-8), sub-concepts of *Imaging* (lines 9-11), and super-concepts of *Hospital* (lines 12-14). The PREFIX command defines shortcuts for the WordNet schema and synset identifier namespaces.

```
1  PREFIX wnschema: <http://www.w3.org/2006/03/
2                          wn/wn20/schema/>
3  PREFIX wn30: <http://purl.org/vocabularies/
4                    princeton/wn30/>
5
6  SELECT * WHERE { ?HospitalHyponyms
7                   wnschema:hyponymOf
8                   wn30:synset-hospital-noun-1.}
9  SELECT * WHERE { ?ImagingHyponyms
10                  wnschema:hyponymOf
11                  wn30:synset-imaging-noun-2.}
12 SELECT * WHERE { wn30:synset-hospital-noun-1
13                  wnschema:hyponymOf
14                  ?HospitalHypernyms.}
```

Listing 1: SPARQL queries to retrieve is-a relations from WordNet for terms of the domain model (excerpt)

Each query of Listing 1 retrieves a partial graph of the domain knowledge. Listing 2 shows the corresponding results in ASCII representation. The variable name of the previously executed query is followed by a list of found nodes. The retrieved domain knowledge includes the information that a *Hospital* is a *Medical Building* (lines 3-4) and that there exist six sub-concepts of *Hospital* (lines 6-12) and five sub-concepts of *Imaging* (lines 14-19) in WordNet.

[4]http://semanticweb.cs.vu.nl/lod/wn30/

```
1  PREFIX wn30: <http://purl.org/vocabularies/
       princeton/wn30/>
2
3  HospitalHypernyms
4  <wn30:synset-medical_building-noun-1>
5
6  HospitalHyponyms
7  <wn30:synset-creche-noun-1>
8  <wn30:synset-lazaretto-noun-1>
9  <wn30:synset-maternity_hospital-noun-1>
10 <wn30:synset-mental_hospital-noun-1>
11 <wn30:synset-military_hospital-noun-1>
12 <wn30:synset-sanatorium-noun-1>
13
14 ImagingHyponyms
15 <wn30:synset-X-raying-noun-1>
16 <wn30:synset-positron_emission_tomography-noun-
17 <wn30:synset-magnetic_resonance_imaging-noun-1>
18 <wn30:synset-sonography-noun-1>
19 <wn30:synset-radioscopy-noun-1>
```

Listing 2: Resulting hyponyms and hypernyms of the executed SPARQL queries

In the next step, the Model Advisor creates a merged graph of all results. Due to the nature of knowledge representation in RDF (URI-based, subject-predicate-object statements) a graphical visualization for domain specific modeling purposes is not rational without any further processing of the result graph. We transform the query results into an Ecore-based model by mapping nodes to classes, node labels to class names, hypernym/hyponym relations to generalizations/specializations and holonym/meronym relations to aggregations. The visualization of the model is created with the Ecore diagram editor of the Eclipse Modeling Framework as shown in Figure 3. For space reasons we have not included all sub-classes of hospital in the figure.

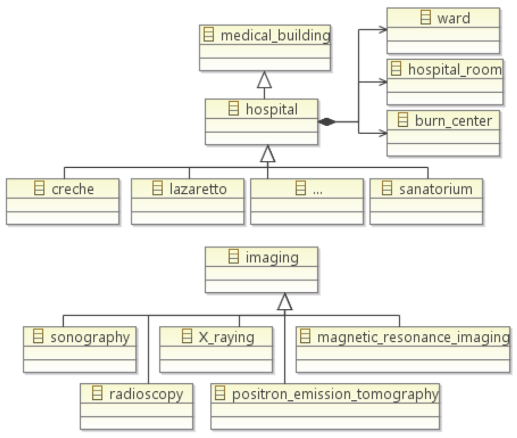

Figure 3: Generated Guidance Model

Finally, the guidance model provided by the EXAMINE-System helps the industrial partner with modeling decisions to complement and improve his domain model. He receives important assistance to overcome two principal problems: (1) He may be unaware of a certain concept or he is unsure whether a certain term should be included in the model or not. Showing semantically related concepts of the domain model improves the *completeness* of the domain model. (2) In the original model (Figure 2) ”X-raying” was a subtype of ”Medical Examination”. From the domain guidance the developer receives the information that ”X-raying” is a special type of ”Imaging”. Thus, the *semantic correctness* of the domain model is improved as well. Using the information provided by the guidance model, the industrial partner can then improve his domain model accordingly. Figure 4 depicts the adjusted domain model.

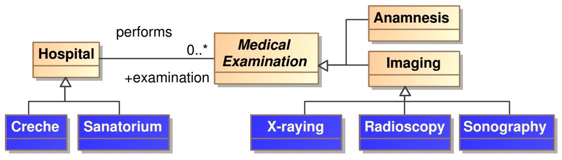

Figure 4: Improved Domain Model

4.3 The DIESEL-Framework

In order to support the process of matching a domain model to the customer’s requirements, the EXAMINE system is embedded into a framework for Domain-Impelled Engineering and Supervised Evolution of Languages (DIESEL- framework). It constitutes the core of the software factory and is used for a continuous engineering of software by supporting a complete development process of DSL’s for both DSL designer and domain expert. An overview of the framework’s architecture is presented in Figure 5.

By systematically assessing the domain expert’s preferences and by providing feedback for DSL prototypes using a metric, a DSL prototype is evaluated. This way, guidance is provided for the whole engineering process of a DSL, including the concrete syntax definition. DIESEL interfaces with the semantic repository and includes the EXAMINE system component together with the Model Advisor service. Using the framework, the domain model can be systematically verified by executing the BIZWARE reference process, which so far is sketched as follows:

During a DSL development’s decision phase, informal documents are gathered cooperatively in talks between the domain expert and the DSL designer. By starting with the creation of a glossary, a disambiguation of terms is formed and saved into the semantic repository. The domain analysis follows, supported by the EXAMINE system and by manually modeling the domain model or ontology. The domain expert may use the assessment feature to work through an interactive questionnaire, to determine his knowledge of modeling concepts and other preferences. The results are evaluated and measured using a metric and presented to the DSL designer, which pre-selects possible DSL patterns for the subsequent steps. Iteratively looping through the design, implementation and deployment phase, the DSL designer can implement alternative solutions for the DSL and present them to the domain expert. The domain expert’s language use is monitored and measured with a cognitive load metric.

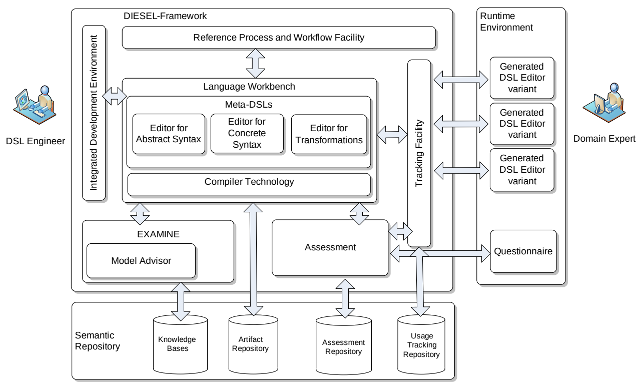
DIESEL-Framework
Runtime Environment
Reference Process and Workflow Facility
Integrated Development Environment
Language Workbench
Meta-DSLs
Editor for Abstract Syntax
Editor for Concrete Syntax
Editor for Transformations
Compiler Technology
Tracking Facility
Generated DSL Editor variant
Generated DSL Editor variant
Generated DSL Editor variant
DSL Engineer
Domain Expert
EXAMINE
Model Advisor
Assessment
Questionnaire
Semantic Repository
Knowledge Bases
Artifact Repository
Assessment Repository
Usage Tracking Repository

Figure 5: Architecture Overview

The purpose of the usage tracking is to help the DSL designer to systematically improve the DSL. The domain expert is continuously involved in the modeling process, not only of the domain models, but also the DSL’s syntax and semantics, and can even try out language variants in the absence of the DSL designer. The process can be extended to support several domain experts to find an adequate alternative for all of them. So far we have identified requirements for the DIESEL-framework’s components: Assessment, Tracking, Repository and Model Advisor. We have also sketched parts of a DSL engineering reference process and identified a pattern for the decision phase. The results were presented and verified within the BIZWARE consortium. We have identified qualitative criteria for the feasibility check of a DSL development and are currently working on a first version of the metric to measure cognitive load in a textual DSL environment.

5. CONCLUSIONS

In this paper, we provided an illustrative example about DSL development motivations in SMEs and presented novel features required for the successful application of domain specific modeling. We believe that DSL engineering processes benefit from a systematical approach involving the use of knowledge bases and metrics to supply guidance to both DSL engineer and domain expert. Supported by preliminary results, we believe that using such techniques within a framework, the application of DSM in different industrial domains proves to be beneficial and the entry barrier can be lowered for SMEs. However, we have to conduct further research based on our findings and continue to develop the framework’s prototype, eventually evaluating the practical benefits in industrial application contexts given by the project BIZWARE.

6. REFERENCES

[1] H. Agt, G. Bauhoff, M. Cartsburg, D. Kumpe, R. Kutsche, and N. Milanovic. Metamodeling Foundation for Software and Data Integration. In *Proc. ISTA*, 2009.

[2] H. Agt, G. Bauhoff, R.-D. Kutsche, and N. Milanovic. Modeling and Analyzing Non-functional Properties to Support Software Integration. In *CAiSE 2011 Workshops*, Berlin, Heidelberg, 2011. Springer-Verlag.

[3] H. Agt, G. Bauhoff, R.-D. Kutsche, N. Milanovic, and J. Widiker. Semantic Annotation and Conflict

Analysis for Information System Integration. In *Proceedings of the MDTPI at ECMFA*, 2010.

[4] T. Aschauer, G. Dauenhauer, and W. Pree. A modeling language's evolution driven by tight interaction between academia and industry. In *Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering - Volume 2*, ICSE '10, pages 49–58, New York, NY, USA, 2010. ACM.

[5] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and Z. Ives. DBpedia: a nucleus for a web of open data. In *Proceedings of the 6th international The semantic web and 2nd Asian conference on Asian semantic web conference*, ISWC'07/ASWC'07, Berlin, Heidelberg, 2007. Springer-Verlag.

[6] C. Bizer, T. Heath, T. Berners-Lee, and M. Hausenblas. 4th linked data on the web workshop (LDOW2011). In *WWW (Companion Volume)*, pages 303–304, 2011.

[7] J. Euzenat and P. Shvaiko. *Ontology Matching*. Springer, Berlin, 2007.

[8] C. Fellbaum. *WordNet : An Electronic Lexical Database*. The MIT Press, Cambridge, MA, 1998.

[9] F. Gailly. Conceptual modeling using domain ontologies : Improving the domain-specific quality of conceptual schemas. In *Domain-specific modelling workshop (DSM-10) - SPLASH/OOPSLA workshop*, number 2009/573. University Ghent, 2010.

[10] M. Genero, M. Piattini, S. Abrahao, E. Insfran, J. A. Carsi, and I. Ramos. A controlled experiment for selecting transformations based on quality attributes in the context of mda. *Empirical Software Engineering and Measurement, International Symposium on*, 0:498, 2007.

[11] M. Genero, G. Poels, and M. Piattini. Defining and validating metrics for assessing the understandability of entity-relationship diagrams. *Data Knowl. Eng.*, 64:534–557, March 2008.

[12] G. Guizzardi. On Ontology, ontologies, Conceptualizations, Modeling Languages, and (Meta)Models. In *Proceeding of the 2007 conference on Databases and Information Systems IV: Selected Papers from the Seventh International Baltic Conference DB&IS'2006*, pages 18–39, Amsterdam, The Netherlands, 2007. IOS Press.

[13] B. Henderson-Sellers. Bridging metamodels and ontologies in software engineering. *J. Syst. Softw.*, 84:301–313, February 2011.

[14] M. Karaila. Evolution of a domain specific language and its engineering environment - lehmanŠs laws revisited. In *Workshop on DomainSpecific Modeling at OOPSLA*, 2009.

[15] L. C. L. Kats and E. Visser. The Spoofax language workbench: rules for declarative specification of languages and IDEs. In W. R. Cook, S. Clarke, and M. C. Rinard, editors, *Proceedings of the 25th Annual ACM SIGPLAN Conference on Object-Oriented Programming, Systems, Languages, and Applications, OOPSLA 2010*, pages 444–463, Reno/Tahoe, Nevada, 2010. ACM.

[16] A. Khalaoui, A. Abran, and E. Lefebvre. Dsml success factors and their assessment criteria. *METRICS News*, Vol.13(No.1):p.p 43–51, February. 2008. Research Notes: 63.

[17] A. Kleppe. *Software Language Engineering: Creating Domain-Specific Languages Using Metamodels*. Addison-Wesley Longman Publishing Co., Inc., Boston, 2009.

[18] S. Lahtinen, J. Peltonen, I. Hammouda, and K. Koskimies. Guided model creation: A task-driven approach. In *Proceedings of the Visual Languages and Human-Centric Computing*, pages 89–94, Washington, DC, USA, 2006. IEEE Computer Society.

[19] M. Mernik, J. Heering, and A. M. Sloane. When and how to develop domain-specific languages. *ACM Comput. Surv.*, 37:316–344, December 2005.

[20] N. Milanovic, M. Cartsburg, R. Kutsche, J. Widiker, and F. Kschonsak. Model-based Interoperability of Heterogeneous Information Systems: An Industrial Case Study. In *Proceedings of the ECMDA*, 2009.

[21] I. Niles and A. Pease. Towards a standard upper ontology. In *Proceedings of the international conference on Formal Ontology in Information Systems - Volume 2001*, FOIS '01, New York, NY, USA, 2001. ACM.

[22] E. Prud'hommeaux and A. Seaborne. SPARQL Query Language for RDF. W3C Recommendation, 2008.

[23] M. Rodriguez, M. Genero, D. Torre, B. Blasco, and M. Piattini. A methodology for continuos quality assessment of software artefacts. In *Quality Software (QSIC), 2010 10th International Conference on*, pages 254 –261, july 2010.

[24] M. Strembeck and U. Zdun. An approach for the systematic development of domain-specific languages. *Softw. Pract. Exper.*, 39:1253–1292, October 2009.

[25] F. M. Suchanek, G. Kasneci, and G. Weikum. Yago: A Core of Semantic Knowledge. In *16th international World Wide Web conference (WWW 2007)*, New York, NY, USA, 2007. ACM Press.

[26] R. Tairas, M. Mernik, and J. Gray. Using Ontologies in the Domain Analysis of Domain-Specific Languages. In *MoDELS Workshops*, pages 332–342, Berlin, Heidelberg, 2008. Springer-Verlag.

[27] M. van Amstel, M. van den Brand, and L. Engelen. An exercise in iterative domain-specific language design. In *Proceedings of the Joint ERCIM Workshop on Software Evolution (EVOL) and International Workshop on Principles of Software Evolution (IWPSE)*, IWPSE-EVOL '10, pages 48–57, New York, NY, USA, 2010. ACM.

[28] G. Weikum and M. Theobald. From information to knowledge: harvesting entities and relationships from web sources. In *Proceedings of the twenty-ninth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems of data*, PODS '10, pages 65–76, New York, NY, USA, 2010. ACM.

[29] J. White, D. C. Schmidt, A. Nechypurenko, and E. Wuchner. Model intelligence: an approach to modeling guidance. *CEPIS UPGRADE*, IX, issue No. 2:22–28, 2008.

[30] Y. Wu, F. Hernandez, F. Ortega, P. J. Clarke, and R. France. Measuring the effort for creating and using domain-specific models. In *Proceedings of 10th Domain Specific Modeling Workshop*, 2010.