Cost Estimation for Ontology Development

Elena Paslaru Bontas Simperl, Malgorzata Mochol
Free University Berlin
[paslaru|mochol]@inf.fu-berlin.de

Abstract

Techniques for reliably estimating development efforts are a fundamental requirement for a wide-scale dissemination of ontologies in business contexts. In this paper we account for the similarities and differences between software and ontology engineering in order to establish the appropriateness of applying software cost models to ontologies. We propose a parametric approach to cost estimation for ontology development and analyze various cost factors implied in the ontology engineering process.

1. Introduction

Ontologies are targeted at providing means to formally specify a commonly agreed understanding of a domain of interest in terms of concepts, relationships, and axioms [Gruber 1993]. With applications in fields such as Knowledge Management, Information Retrieval, Natural Language Processing, eCommerce, Information Integration or the emerging Semantic Web, ontologies are part of a new approach to building intelligent information systems [Fensel 2001]: they are intended to provide knowledge engineers with *reusable pieces of declarative knowledge*, which can be – together with *problem-solving methods* and *reasoning services* – easily assembled to high-quality and cost-effective systems [Neches 1991]

Though ontologies and associated ontology management tools have become increasingly popular in the last decades – in particular in conjunction with the emergence of Semantic Web technologies – the dissemination of ontology-driven technologies in real-world business contexts is inconceivable without fine-grained methodologies which are able to deal with both *technical* and *economical* challenges of ontology engineering. Besides feasible tool support, a major requirement for ontologies to be built and deployed at large scale is the availability of methods which address the *business-oriented aspects* of ontology development and deployment.

A wide range of ontology engineering methodologies have been elaborated in the Semantic Web community (see e.g. [Goméz-Pérez 2004] for a description of and a comparison among the most relevant ones). According to these methodologies, engineering ontologies is defined as an iterative process, which shows major similarities with established models from the neighbored research field of Software Engineering. However, unlike adjacent engineering disciplines,

current Semantic Web research ignores several aspects of engineering processes, which are fundamental in real-world business contexts. Issues like costs estimation (using pre-defined cost models), quality assurance procedures (w.r.t. both end product i.e. ontologies and associated engineering process) or means to monitor the business value or the impact of these technologies in organizational context have yet been marginally exploited by the Ontology Engineering research community.

In this paper we aim at contributing to the alleviation of this situation by proposing **ONTOCOM** (**ONTO**logy **CO**st **M**odel), a parametric cost model for the estimation of the efforts involved in building, reusing and maintaining ontologies in information systems.

The remaining of this paper is organized as follows: After introducing a high-level process model for engineering ontologies in Section 2, we analyze general-purpose cost estimation methodologies in order to assess their appropriateness for the area of Ontology Engineering (Section 3). These investigations result in the cost estimation model ONTOCOM, which is presented in Section 4. Section 5 gives an overview of the cost drivers supported by the parametric cost model while Section 6 presents a brief example on the usage of this model and its cost drivers. The evaluation of our approach and planed future work are addressed in Section 7 and 8 respectively.

2. Ontology vs. Software Engineering

The overall process of developing a cost model for ontology development shares commonalities with existing approaches in adjacent engineering disciplines. In particular, the similarities between ontology engineering (OE) and software engineering (SE) processes, as well as between ontology and software modeling make software cost estimation methods a reasonable starting point for the development of a cost estimation methodology for ontologies. In the following we account for the similarities and differences between these disciplines in order to establish the appropriateness of applying software cost models to OE.

SE and OE belong to the general area of *engineering*, a field which is usually defined by the usage of scientific principles and methods to construct artifacts. SE, one of the core disciplines in Computer Science, provides systematic methods to handle the complexity of designing, implementing and maintaining software systems. One of the basic actions of the SE is modeling. The process of building a model – a concrete or mental image of an existing thing or a paradigm of an original which can be created – supplies [Glinz 2005]:

- an inventory of fundamental languages and methods to describe and analyze specific classes of problems
- fundamental knowledge of problem solving methods
- methodologies for the building of software artifacts, as well as
- a deep understanding of the nature of these artifacts

Several decades of research and development in SE showed that software projects can not be reduced to software implementation: they should equally

focus on user requirements, design, testing, documentation, deployment and maintenance. Modelling is applied in many of these activities for various purposes:

- requirements as *models of the problem definition*,
- architectures as *models of the solution*,
- test regulations as *models of correct functioning of the code*, and
- software itself as *model of cutout of a real world* [Glinz 2004].

Likewise Software Engineering, Ontology Engineering includes scientific methods to create, deploy and maintain ontologies (see Table 1). Though OE and SE share the same final goal – the generation of high-quality artifacts – and a similar development process [Staab 2000], OE methodologies available so far do not address every phase of the underlying process at the same level of detail and do not provide a fully-fledged tool environment to aid the engineering team in their attempt to create, maintain or use ontologies.

Table 1. SE vs. OE

Models	Software Engineering	Ontology Engineering
Model of the problem	System analysis (requirements specification, reuse)	Domain analysis (requirements specification, knowledge acquisition)
Model of the solution	System design (architecture)	Conceptualization (conceptual model)
Model of the solution	Implementation (program code)	Implementation (specification of the conceptual model)
Model for tests	Test data generation	Ontology population
Model of correct functioning of the code	System testing (refinement)	Ontology evaluation (refinement)
Model for the evolution of the system	System maintenance	Ontology maintenance

The main difference between the two engineering processes focuses on their outcomes: software vs. ontologies. A *software system* is a collection of computer programs and associated data and computer equipment that performs a specific function or purpose. It consists of interrelated software components which exchange data by means of interfaces and produce a certain result or behavior given user-defined inputs. An *ontology* specifies a commonly agreed conceptualization of a given universe of discourse. It is a collection of ontological primitives, usually concepts or classes connected to each other by relations and constrained by axioms, which may belong to specific sub-ontologies. Consequently the way to measure the size of a software system, usually expressed in lines of code or function/object points, can not be directly applied to ontologies. Due to the fact that the implementation of an ontology is mostly realized using tools (i.e. ontology editors), the main size factor to express the complexity of an ontology is not given by the actual size of an implementation in a specific representation language, but by the number of

ontological primitives contained by conceptual model. Further on, modelling different types of ontological primitives is associated to significantly different amounts of efforts: the conceptualization of classes and their classification in a subclass-hierarchy is recognized to imply considerably less efforts than modelling relationships or axioms. In comparison to a software system, an ontology may be used in a twofold manner: it may be embedded in a software system for a specific purpose (e.g. to index domain-specific documents) or it may be used directly by domain experts without any mediator in form of some computer program (e.g. as a commonly agreed vocabulary). Nevertheless an ontology does not show any behavior – it does not produce an output for a given input as for a software system. This aspect has implications in the way software and ontologies are evaluated. While the evaluation of software (in the sense of software testing) is a mature discipline despite the difficulties in achieving a commonly agreed, general purpose software quality model, evaluating ontologies is still an open issue both from a technical and a user-centered point of view. Another important difference between SE and OE is related to engineering process themselves: the ontology population/instantiation, a particularity of OE vs. the related software field, is often associated with significant efforts depending on the characteristics of the data to be aligned to the ontology. The project team involved in the construction of an ontology is relatively small compared to the common team size in software development. As a consequence of the importance of the conceptualization phase, the role of the user/domain expert in the ontology construction and his experience in the domain to be modeled are crucial factors for the success of OE projects. Finally, as for the knowledge of the authors, given the present state of the art of the OE area, the duration of OE projects is significantly shorter than the typical duration of software projects.

Despite these differences a cost estimation model for ontologies should benefit from the similar properties of the two engineering fields. The efforts associated with the engineering process can still be assumed to depend on the complexity of the resulting artifact: the size of the ontology to be modeled and in its particularities, and in the size and functionality of the software product respectively [Boehm 1981]. A cost model for OE should take into account the results achieved by the SE community and customize and extend them in order to cover the particularities of the ontology development task. Consequently, we now turn to a survey of some of the most relevant methodologies for costs estimation in SE for the purpose of assessing their suitability for *current* ontology engineering processes.

3. Estimating Costs for Ontology Engineering

Due to its high relevance in real-world situations, cost estimation is investigated in a wide range of methods and methodologies [Boehm 1981] that are often used in conjunction in business context due to their limitations w.r.t.

certain classes of situations. In the following we give an overview of some of most important ones:

- The *analogy method* extrapolates the available data from similar projects to estimate the costs of the proposed project. This method is suitable in situations where empirical data from previous project is available and trustworthy and depends on the accuracy in establishing real differences between completed and current projects.
- The *bottom-up method* involves identifying and estimating costs of individual project components separately and subsequently combining the outcomes to produce an estimation for the overall project. It can not be applied early in the life cycle of the process because of the lack of necessary information related to the project components.
- The *top-down method* relies on overall project parameters. For this purpose, the project is partitioned into lower-level components and life cycle phases beginning at the highest level. This method is more applicable to early cost estimates when only global properties are known, but it can be less accurate due to the less focus on lower-level parameters and technical challenges – usually predictable later in the process life cycle, at most.
- The *expert judgment/Delphi method* is based on a structured process for collecting and distilling knowledge from a group of human experts by means of a series of questionnaires interspersed with controlled opinion feedback. The involvement of human experts using their past project experiences is a significant advantage of the Delphi approach, while the most extensive critique point is related to the subjectivity of the estimations and the difficulties to explicitly state the decision criteria used by the contributing experts.
- The *parametric/algorith*mic method involves the usage f mathematical equations based on research and historical data from previous projects. The method analyzes main cost drivers of a specific class of projects and their dependencies and uses statistical techniques to refine and customize the corresponding formulas. As in the case of the analogy method the generation of a roved and tested cost model using the parametric method is directly related to the availability of reliable and relevant data to be used in calibrating the initial core model.

The aforementioned approaches were evaluated against the requirements of a cost estimation method for Ontology Engineering. Given the lack of fine-grained engineering process models correlated with the low amount of cost-relevant data published so far, the *top-down, parametric and expert-based* methodologies can be currently used to partially develop a cost estimation process model for ontologies. Due to the expected incompleteness of the empirical data, a combination of the three is likely to overcome certain limitations of individual approaches. Independently of the availability of the historical project data – definitely required to calibrate a parametric model – the analysis of the main factors affecting the costs of an ontology engineering process can be performed

at present on the basis of existing case studies. The compilation of a set of cost drivers is an important step towards the realization of an accurate cost estimation tool for Ontology Engineering; while the initial cost model (the so-called a-priori model) is subject of continuous calibration, the definition of a fixed spectrum of cost factors is essential for a controlled collection of existing real-world project data. Further on, the limited amount of empirical data can be counterbalanced by taking into account the significant body of expert knowledge available in the Semantic Web community. Human expertise w.r.t. this topic can be used to complement the results of the parametric predictions.

4. ONTOCOM – ONTOlogy COst Model

For the development of the ONTOCOM cost model we adopted a combination of three aforementioned estimation methodologies, which are in our opinion applicable to OE according to the current state of the art in the field. We start with a top level approach, by identifying upper-level sub-tasks of the OE process and define the associated costs using a parametric method. We distinguish among three areas, whose costs are to be defined separately:

- *Ontology Building* includes domain analysis (except the knowledge acquisition, i.e. the re-usage of available knowledge sources), conceptualization, implementation, ontology population and evaluation.
- *Ontology Maintenance* involves costs related to getting familiar and updating the ontology.
- *Ontology Reuse* accounts for the efforts related to the re-usage of existing (source) ontologies for the generation of new (target) ontologies. Consequently ontology reuse involves costs related to finding, evaluating and adapting the former ones to the requirements of the latter.

This upper-level distribution is of course subject of future refinements in order to increase the usability of the estimation method in real-world engineering projects. In particular, the ontology development area should be elaborated in the same top-down manner in order to partition this tedious and complex process down to a level in which the associated efforts can be reliably predicted. In this case, the cost drivers relevant the overall ontology building process are to be aligned to the corresponding sub-phases and activities (see [Paslaru 2005c] for a detailed description of the process of adapting ONTOCOM to more elaborated ontology engineering methodologies).

Estimating the effort (in person months) related to OE is reduced to a sum of the costs arising in the building (with or without reuse) and maintaining ontologies:

$$PM = PM_B + PM_M + PM_R,$$

where PM_B, PM_M and PM_R represent the effort associated to building, maintaining and reusing ontologies, respectively. The partial costs are calculated as (ONTOCOM-formula):

$$PM_x = A * (Size_x)^B * \prod CD_{xi}$$

Each of the three development phases is associated with specific cost factors. Experiences in related engineering areas [Kemerer 1987, Boehm 1981] let us assume that the most significant one is the size of the ontology involved in the corresponding process. In the formula above the size parameter $Size_x$ is expressed as a weighted sum of (thousands of) ontological primitives i.e. concept, relations, axioms and instances:

- $Size_b$ (building) corresponds to the size of the newly built ontology, i.e. the number of primitives which are expected to result from the conceptualization phase.
- $Size_m$ (maintenance) depends on the expected number of modified items.
- $Size_r$ (reuse) is the size of the original source after being tailored to the present application setting. In particular this involves the parts of the source ontologies which have to be translated to the final representation language, the ones whose content has to be adapted to the target scope and the fragments directly integrated.

The possibility of a non-linear behavior of the model w.r.t. the size of the ontology is covered by parameter B while start-up costs, which are not proportional to the size of a project, are intended to be counterbalanced by the constant A. The core parts of the ONTOCOM-formula are the cost drivers CD_{xi}. They give a rating level (from *very low* to *very high*) that expresses their impact on the development effort. For the purpose of a quantitative analysis, each rating level of each cost driver is associated to a weight (*effort multiplier* – EM). The average EM assigned to a cost driver is 1.0 (*nominal weight*). If a rating level causes more development effort, its corresponding EM is above 1.0 otherwise it is less than the nominal value 1.0.

In the a-priori cost model a team of five ontology engineering experts assigned start values between 0.7 and 1.9 to the effort multipliers, depending on the perceived contribution of the corresponding cost driver to the overall development costs.[1] In the same manner, the start value of the A parameter was set to 2.3. These values are subject of further calibration on the basis of the statistical analysis of real-world project data.

5. Cost Drivers for Ontology Engineering

In the following we give an overview of the cost drivers involved in ontology building, maintenance and reuse. We differentiate among *product*, *process* and *personnel* cost drivers. The product category accounts for the influence of product properties on the overall costs. The process category states the dimensions of the engineering process which are relevant for the cost estimation, while the personnel one emphasizes the role of team experience, ability and continuity for the effort invested in the process.

---

[1] A list of the values is available in [Paslaru 2005c]

5.1. Product factors

5.1.1. Ontology building

- *Complexity of the Domain Analysis (DCPLX)* This driver states for the efforts additionally arisen in the engineering project by the particularities of the ontology domain and its analysis during ontology building. The decision which concepts will be included and in which form they will be represented in an ontology depends not only on the intrinsic domain to be modeled (e.g., tourism), but rather on the application domain.
- *Complexity of the Conceptualization (CCPLX)* The conceptualization complexity accounts for the impact of the structure of the conceptual ontology (taxonomy, conceptual graph etc.) and of help techniques such as modeling patterns on the overall engineering costs. It considers the number, the type and the form of the sources.
- *Complexity of the Implementation (ICPLX)* In some cases the implementation of the ontology requires a mapping between the knowledge level of the conceptualization and the paradigms beyond the used representation language. The costs arisen during this mapping are stated in the driver ICPX.
- *Instantiation (DATA)* The population of an ontology and the associated testing operations might be related to considerable costs [Buitelaar 2004, Dittenbach 2004]. The measure attempts to capture the effect instance data requirements have on the overall process. In particular the form of the instance data and the method required for its ontological formalization are significant factors for the costs of the engineering process.
- *Required Reusability (REUSE)* The REUSE attempts to capture the effort associated with the development of a reusable ontology. Usually (there is no commonly agreed understanding) reusability is mentioned in the context of application-independency, in that it is assumed that application-dependent ontologies are likely to imply significant customization costs if reused. Additionally several types of ontologies are often presumed to endue an increased reusability: core ontologies and upper-level ontologies describing general aspects of the world are often used in alignment tasks in order to ensure high-level ontological correctness.
- *Documentation match to lifecycle needs (DOCU)* The DOCU measure is intended to state the additional costs caused by detailed documentation requirements.
- *Ontology Evaluation (OE)* The cost drivers captures the effort invested in evaluating ontologies, be that testing, reviewing, usability or ontological evaluation.
- *Ontology Integration (OI)* This cost drivers measures the costs produced by integrating different ontologies to a common framework. The integration step is assumed to be performed on ontologies sharing the same representation language.

5.1.2. Ontology Reuse and Maintenance

For ONTOCOM we assume that relevant ontologies might be available to the engineering team for reuse purposes. According to the mentioned top-level approach and to current case studies in ontology reuse [Paslaru 2005a, Paslaru 2005b, Russ 1999, Uschold 1998] we examine the following two phases of the reuse process w.r.t. the corresponding cost drivers: ontology evaluation and ontology customization.

For the evaluation phase the engineering team is supposed to assess the relevance of a given ontology to particular application requirements. The success of the evaluation depends crucially on the extent to which the ontology is familiar to the assessment team. The customization phase implies the identification/extraction of sub-ontologies which are to be integrated in a direct, translated and modified form, respectively. In the first categories sub-ontologies are included directly to the target ontology. The re-usage of the second category is conditioned by the availability and the appropriate costs of knowledge representation translators, while the last category involves modifications of the original model in form of insertions, deletions or updates at the ontological primitives level.

In the following we describe the defined cost drivers for ontology reuse whereat two of them (OU, OM) are also used in the ontology maintenance phase:

- *Ontology Understandability (OU)* Reusing an ontology and the associated efforts depend significantly on the ability of the ontologists and domain experts to understand the ontology, which is influenced by two categories of factors, the complexity and the clarity of the conceptual model [Paslaru 2005c, Paslaru 2005d].
- *Ontology Modification (OM)* This measure reflects the complexity of the modifications required by the reuse process after the evaluation phase has been completed.
- *Ontology Evaluation (OE)*[2] This measure accounts for the real effort needed to evaluate the ontology for reuse purposes. The measure assumes a satisfactory ontology understanding level and is associated solely with the efforts needed in order to decide whether a given ontology satisfies a particular set of requirements and to integrate its description into the overall product description.
- *Ontology Translation (OT)*[3] Translating between knowledge representation languages is an essential part of a reuse process and depends on the compatibility of the source and target representation languages and on the availability and performance of the translating tools (amount of pre- and post-processing required) [Falkovych 2003].
- *Ontology Integration (OI)*[4] This measure accounts for the additional efforts required to integrate or merge multiple ontologies to a common target ontology.

[2,3,4] These drivers are relevant solely for ontology reuse.

5.2. Personnel Factors

- *Ontologist/Domain Expert Capability (OCAP/DECAP)* The development of ontologies requires the collaboration between a team of ontology engineers (ontologists), usually with an advanced technical background, and a team of domain experts that provide the necessary know-how in the field to be ontologically modeled. These cost drivers account the perceived ability and efficiency of the single actors involved in the process, as well as their teamwork capabilities.
- *Ontologist/Domain Expert's Experience (OEXP/DEEXP)* These measures take into account the experience of the engineering team consisting of both ontologists and domain experts w.r.t. the OE process. They are not related to the abilities of single team members, but relate directly to the experience in constructing ontologies and in conceptualizing a specific domain respectively.
- *Language and Tool Experience (LEXP/TEXP)* The aim of these cost drivers is to measure the level experience of the project team constructing the ontology w.r.t. the conceptualization language and the ontology management tools respectively. The conceptualization phase requires the usage of knowledge representation languages with appropriate expressivity (such as Description Logics or Prolog), while the concrete implementation is addicted to support tools such as editors, validators and reasoners.
- *Personnel Continuity (PCON)* As in other engineering disciplines frequent changes in the project team are a major obstacle for the success of an OE process within given budget and time constraints.

5.3. Project Factors

- *Support tools for Ontology Engineering (TOOL)* The usage of ontology management tools is an essential success factor in every OE process. Apart from the implementation phase, which is worst-case performed by manually feeding the conceptual model to some ontology editor, the evaluation of the ontology can not be imagined without the utilization of validation and reasoning tools.
- *Multi-site Development (SITE)* Constructing an ontology requires intensive communication between ontology engineers and domain experts on one hand and between domain experts for consensus achievement purposes on the other hand. This measure involves the assessment of the communication support tools.
- *Required Development Schedule (SCED)* SCED takes into account the particularities of the engineering process given certain schedule constraints. Accelerated schedules tend to produce more efforts in the refinement and evolution steps due to the lack of time required by an elaborated domain analysis and conceptualization. Stretch-out schedules generate more effort in the earlier phases of the process while the evolution and refinement tasks are best case negligible.

6. Using ONTOCOM – Example

Starting from a typical ontology building scenario, in which a domain ontology is created from *scratch* (without integration) by the engineering team, we simulate the cost estimation process according to the parametric method underlying ONTOCOM. Given the top-down nature of our approach this estimation can be realized in the early phases of a project, in particular after the domain analysis has been accomplished and an initial prediction of the size of the target ontology is available. The first step of the cost estimation is the specification of the *size of the ontology* to be build, expressed in thousands of ontological primitives (concepts, relations, axioms and instances): if we consider an ontology with 1000 concepts, 200 relations (including is-a) and 100 axioms, the size parameter of the estimation formula will be calculated as follows:

$$Size_b = \frac{1000 + 200 + 100}{1000} = 1{,}3$$

The next step is the specification of the cost driver ratings corresponding to the information available at this point (i.e. without reuse and maintenance factors). Assuming that the ratings of the cost drivers are those depicted in Table2 these ratings are replaced by numerical values. The value of the DCPLX cost driver was computed as an equally weighted, averaged sum of a high-valued rating for the domain complexity (1,3) , a nominal rating for the requirements complexity (1,0) and a high effort multiplier for the information sources complexity (1,3) (for details of other rating values see [Paslaru 2005e]):

$$DCPLX = \frac{1*1{,}3 + 1*1{,}0 + 1*1{,}3}{3} = 1{,}2$$

According to the ONTOCOM main formula the estimated effort in person months would be amount to ***4,10 PMs*** and be calculated as follows (the A parameter was set to 2.3 as described in Section 4):

$$PM_B = 2{,}3 * (1{,}3)^1 * (1{,}2 * 1^7 * 0{,}85^4 * 1{,}30^2 * 1{,}15 * 0{,}7 * 1{,}60)$$

Table 2. Relevant factors and their ratings

Group of factors	Factors	Ratings	Values
Product Factors (without reuse and maintenance)	*DCPLX*	High	1,20
	CCPLX	Nominal	1
	ICPLX	Low	0,85
	DATA	High	1,30
	REUSE	Nominal	1
	DOCU	Low	0,85
	OE	Nominal	1
Personnel Factors	*OCAP*	High	0,85
	DCAP	Low	1,15
	OEXP	High	0,85
	DEEXP	Very Low	1,30
	LEXP	Nominal	1
	TEXP	Nominal	1

	PCON	Very High	0,70
Project Factors	*TOOL*	Very Low	1,60
	SITE	Nominal	1
	SCED	Nominal	1

7. Evaluation

ONTOCOM is currently being validated towards an accurate method for estimating the costs of ontology engineering. The most important evaluation criterion is of course the reliability of its predictions, which depends on the amount and the quality of the historical project data used to calibrate the model (i.e. adjust the values of the modifiers and identify eventual correlations between cost drivers). However, a comprehensive evaluation of the model should go beyond the evaluation of its functionality (i.e. the accuracy of its estimations) and also address issues related to its usability in typical ontology engineering scenarios. For the evaluation of the model we rely on the quality framework for cost models by Boehm [Boehm 1981], which was adapted to the particularities of ONTOCOM and Ontology Engineering. Parts of this framework are used to assess the quality of the a-priori and the a-posteriori cost models, respectively [Paslaru 2005d]. According to this differentiation, the evaluation of the cost model is performed in two steps. First we evaluate the relevance of the model, in particular of the ONTOCOM cost drivers, for the purpose of predicting costs arisen in ontology engineering projects. The remaining aspects of the framework are directly related to the usage of the model in real-world situations and to the accuracy of its cost predictions. Therefore they are applied in a second step of the evaluation on the a-posteriori model resulting from the calibration of the preliminary one.

The evaluation of the a-priori model was performed by conducting interviews with two groups of experts in the area of Ontology Engineering, consisting of 4 and, respectively 8 participants. In each of the two phases of the evaluation, participants were given a one hour overview of the ONTOCOM approach, which was followed by individual interviews according to the aforementioned framework. During the interviews, the participants expressed their concerns w.r.t. a wide range of costs-related issues in ontology engineering projects and discussed about their experiences in building, reusing or deploying ontologies.

The first phase of the evaluation resulted in major changes w.r.t. the definition of the cost drivers and the introduction of missing ones. The second phase of the evaluation produced minimal adaptations of the revised model, in particular w.r.t. a more precise scope of the model, and helped us identifying possible directions for further research. The model presented in the previous sections is the result of this first phase. The second phase of the evaluation is being performed, in parallel to the recently started data collection initiative. Empirical data from previous ontology engineering projects is collected from various organizations in the Semantic Web field, contributing to the continuous

calibration of the model. Furthermore, the data collection procedure offers us valuable information about the usability of ONTOCOM and its cost drivers w.r.t. a wide range of ontology engineering scenarios (by now we collected data from 27 projects). The analysis of the empirical data collected so far indicates a well-balanced influence of the cost drivers on the model estimations. The reliability of the predictions is planed to be computed on a data set of at least 75 data points.

8. Conclusions and Future Work

Reliable methods for cost estimation are a fundamental requirement for a wide-scale dissemination of ontologies in business contexts. However, though the importance of cost issues is well-recognized in the community, no cost estimation model for ontology engineering is available so far. Starting from existing cost estimation methodologies applied across various engineering disciplines, we propose a parametric cost estimation model for ontologies by identifying relevant cost drivers having a direct impact on the effort invested in the main activities of the ontology life cycle. We evaluate the model a-priori and a-posterior.

In the future we will apply ONTOCOM to other ontology engineering methodologies. This can lead to the introduction of new cost drivers or the redefinition of exiting ones, or both. While definitely in an incipient phase, the a-posterior evaluation has already indicated a balanced influence of the cost factors. The collection of data is also a pre-requisite for a more accurate calibration of the model. Nevertheless our present experiences with the model are very promising: the evaluation of the a-priori model demonstrated the applicability of the model for ontology development processes, while the public data collection initiative was received favourably by the community and will continue in the future.

Acknowledgements: This work has been partially supported by the "KnowledgeWeb – Network of Excellence", by the project "Semantic Web for Pathology" funded by the DFG (German Research Foundation) and by the "Knowledge Nets"" project, which is part of the InterVal – Berlin Research Centre for the Internet Economy, funded by the German Ministry of Research BMBF. Further information about ONTOCOM can be found under: http://ontocom.ag-nbi.de. The tool supporting the ONTOCOM data collection initiative is available at:http://kompass.mi.fu-berlin.de/phpESP/public/survey.php?name=ontocom2final_260905.

References

[Boehm 1981] B. W. Boehm, Software Engineering Economics, Prentice-Hall, 1981.

[Buitelaar 2004] P. Buitelaar, D. Olejnik, M. Sintek, "A Protégé Plug-In for Ontology Extraction from Text Based on Linguistic Analysis", Proc. of the European Semantic Web Symposium ESWS04, 2004.

[Dittenbach 2004] M. Dittenbach, H. Berger, D. Merll, "Improving domain ontologies by mining semantics from text", Proc. of the 1st Asian-Pacific Conference on Conceptual Modelling, p. 91-100, 2004.

[Falkovych 2003] K. Falkovych, M. Sabou , H. Stuckenschmidt, "UML for the Semantic Web: Transformation-Based Approaches", Vol. Knowledge Transformation for the Semantic Web, IOS Press, 2003.

[Fensel 2001] D. Fensel. Ontologies: A Silver Bullet for Knowledge Management and Electronic Commerce. Springer Verlag, 2001.

[Glinz 2005] M. Glinz. Informatik IIa: Modellierung. Folienskript, Universität Zürich. http://www.ifi.unizh.ch/req/courses/inf_II/, 2005.

[Glinz 2004] M. Glinz, B. Rumpe. Informatik Forsch. und Entwick., 18: 103–104,http://www.sse.cs.tu-bs.de/publications/GMR_Editorial_IFE_04.pdf, 2004

[Goméz-Pérez 2004] A. Goméz-Pérez, M. Fernández-Lopéz, and O. Corcho. Ontological Engineering. Advanced Information and Knowledge Processing. Springer Verlag, 2004.

[Gruber 1993] T. R. Gruber, "A translation approach to portable ontologies. Knowledge Acquisition, 5(2), S. 199-220, 1993.

[Kemerer 1987] C. F. Kemerer, "An Empirical Validation of Software Cost Estimation Models", C-ACM Journal, Vol. 30, Nr. 5, 1987.

[Neches 1991] R. Neches et al. Enabling technology for knowledge sharing. AI Magazine, 12(3):35-56, 1991.

[Paslaru 2005a] E. Paslaru Bontas, M. Mochol, R. Tolksdorf, "Case Studies in Ontology Reuse", Proc. of the 5th International Conference on Knowledge Management IKNOW05, 2005.

[Paslaru 2005b] E. Paslaru Bontas, M. Mochol, "Towards a methodology for ontology reuse", Proc. of the International Conference on Terminology and Knowledge Engineering TKE05, 2005.

[Paslaru 2005c] E. Paslaru Bontas, M. Mochol, "A cost model for ontology engineering", Technical Report TR-B-05-03. FU Berlin. Apr'05.

[Paslaru 2005d] E. Paslaru Bontas, C. Tempich, "How much does it cost? Applying ONTOCOM to DILIGENT", Technical Report TR-B-05-20. FU Berlin. Oct'05.

[Paslaru 2005e] E. Paslaru Bontas, M. Mochol, "Ontology Engineering Cost Estimation with ONTOCOM", Technical Report TR-B-06-01. FU Berlin. Jan'06.

[Russ 1999] T. Russ, A. Valente, R. MacGregor, W. Swartout, "Practical Experiences in Trading Off Ontology Usability and Reusability", Proc. of the Knowledge Acquisition Workshop KAW99, 1999.

[Staab 2000] S. Staab, A. Maedche, "Ontology Engineering beyond the Modelling of Concepts and Relations", Proc. of the ECAI'2000 Workshop on Applications of Ontologies and Problem-Solving Methods, R.V. Benjamins, A. Gomez-Perez, N. Guarino, and M. Uschold, 2000, http://www.aifb.uni-karlsruhe.de/WBS/Publ/2000/ecai_sstama_2000.pdf.

[Uschold 1998] M. Uschold, P. Clark, M. Healy, K. Williamson, S. Woods, "An Experiment in Ontology Reuse", Proc. of the 11th Knowledge Acquisition Workshop KAW98, 1998.