Framework support for usability evaluation of domain-specific languages

Ankica Barisic

Framework Support for Usability Evaluation of Domain-Specific Languages*

Ankica Barišić
NOVA-LINCS, FCT/UNL
Caparica, Portugal
a.barisic@campus.fct.unl.pt

Abstract

In this paper we propose a conceptual framework that supports the iterative development process of DSLs concerning the issue of their Usability evaluation. A multiple-case studies were conducted in order validate the proposed method.

CCS Concepts • **Software and its engineering → Software usability**; **Domain specific languages**;

Keywords Domain-Specific language, Usability evaluation

ACM Reference format:
Ankica Barišić. 2017. Framework Support for Usability Evaluation of Domain-Specific Languages. In *Proceedings of SPLASH '17 Companion, Vancouver, BC, Canada, October 23–27, 2017,* 3 pages.
https://doi.org/10.1145/3135932.3135953

1 Motivation and Related work

The adoption of Domain-Specific Languages (DSLs) is regarded as an approach to reduce the accidental complexity of software systems development. The availability of sophisticated language workbenches facilitates the development of DSLs making them increasingly more popular. This comes at the risk that a badly designed DSL can bring more harm and decrease productivity when compared to an alternative General-Purpose Language (GPL).

In particular, a poorly designed DSL can be too hard to adapt to its domain users. As such, Usability is one of the key characteristics to mitigate this risk as it has an important impact on the achieved productivity of DSL users. The current state of practice in Software Language Engineering (SLE) neglects the Usability of DSLs. The results of the preliminary systematic literature review of Gabriel et al. [13] and more recently by the systematic mapping study of Kosar et al. [19], share a particular concern regarded to the clear lack of DSL evaluation research, in particular, controlled experiments. There are also documented evidence that the above problems are real industry problems [16].

Some authors did tackle above mentioned problems. Kolovos et al. [18] list the core quality requirements for a DSL. Hermans et al. [15] identify success factors for designing DSLs by performing an empirical study. Wu et al. [24] present an approach to determine the effort while using DSLs during application development. Kelly and Pohjonen [17] discuss worst practices for creating DSLs which developers should avoid. Nishino [21] solve the usability problems of DSLs caused by a bad design by analyzing a set of cognitive dimensions proposed by Green et al. [14]. There are also approaches which enable collaborative design of the DSL with domain experts [11, 12, 20, 22, 23, 25]. Presented approaches support the testing of usability problems with a domain experts during a DSL design and they doesn't include the end-users into evaluation or support of controlled experiments.

2 Research Objective and Approach

A pertinent research question in SLE is **'How to systematically engineer Usability into DSLs?'**. The objective of this research is *to promote quality in use of DSLs by building up a conceptual framework that supports their development process by leveraging usability as a first class concern.* This involves the integration and adaptation of the current evaluation methodologies, their concepts, tools, and methods.

A timely systematic approach based on user interface experimental evaluation techniques should be used to assess the impact of DSLs during their development process, while the cost of fixing the usability problems is relatively low when compared to fixing them at the end of the development process. We proposed a set of patterns [10] to promote usability concerns since an early stage of development, as an iterative process which integrates well with the DSL development phases and supports the experimental design [8].

To address the problem of the absence of the systematic approach for the DSL usability evaluation we proposed the conceptual modelling framework called The Usability Software Engineering Modelling Environment (USE-ME) [2].

*The authors would like to thank FCT/MEC NOVA LINCS; PEst UID/CEC/04516/ 2013 and DSML4MA TUBITAK/0008/2014 Projects, and to COST Action IC1404 MPM4CPS for the support.

SPLASH '17 Companion, October 23–27, 2017, Vancouver, BC, Canada

ACM ISBN 978-1-4503-5514-8/17/10…$15.00
https://doi.org/10.1145/3135932.3135953

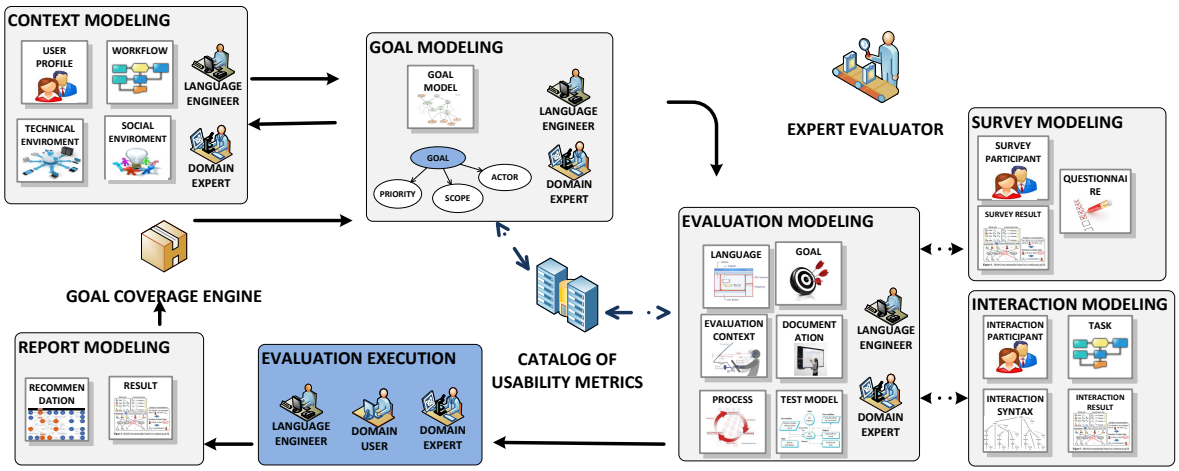

Figure 1. USE-ME life-cycle

This comprehensive methodology supports empirical studies and controlled experiments with end users. It provides a set of practices that should be introduced to provide a complete solution to a complex problem of placing intended users as a focal point of the DSLs design and conception.

The USE-ME process (Figure 1) start with context modelling in which DSL stakeholders define 'who will use the DSL?', 'where the DSL will be used?' and 'how the DSL is expected to be used?'. The usability goals define 'why DSL should be used?' and are stored in a Goal Model, where context dependent requirements and metrics are specified. During evaluation modelling, the evaluation goals and their corresponding evaluation steps are set. Test model can be specified by survey and/or interaction modelling. The participants, representing end-users, are selected to match a user profile from context model. Next, evaluator proceeds with evaluation execution, after which he analyses the results of the test models. Evaluator creates the report model with recommendations and calculates a success coverage of the evaluated usability goal. Finally, it is up to all DSL stakeholders to decide to continue to new evaluation cycle or to finalise the assessment period. Ideally, this decision will eventually indicate the end of the development cycle.

3 Evaluation methodology

We conducted a multiple-case study in order validate the proposed method; two academic case studies served to specify and confirm the evaluation model and experiment designs[1, 7]; two industrial case studies were used to apply the approach during iterative real case development [5, 6]. The researcher was not included in DSL development but was taking a role of expert evaluator. Early evaluations showed to be beneficial and integrable with an agile development process of FlowSL [6]. The controlled experiments on several releases of a Visualino [5] showed reuse of the evaluation model instances and significant improvements of usability.

Despite the importance of patterns and comprehensive methodology, they may not be sufficient to enable the language engineers to learn and use the approach in practice. To evaluate the *feasibility* of the USE-ME methodology, we developed tool support integrable into DSL development infrastructure [3] and illustrated an instantiation of prototype models on Visualino case study [2]. The prototype was validated in the context of the DSL course projects [4], showing that enables novice language engineers to prepare evaluations. Finally, we combined approach with an existing requirement engineering approach for DSLs to show it's *integrability* with existing DSL development support [9].

4 Conclusions and future work

The USE-ME framework identifies all the mandatory concepts and activities and aggregates them into a formal metamodel. It highlights the complexity of the information that should be traced to streamline and automate the user-centred development process. The framework helps the language engineers to explicitly model the evaluation process, which contributes to monitoring the impact of language evolution to the efficiency and effectiveness of practitioners using the language. Additionally, it supports specifying experimental assessments and tracing the impact of usability improvements since an early stage of development of DSLs.

As part of the future work, it is necessary to improve the usability of the tool support to guide the users through the modelling process and validate model instances. Another direction is a creation of domain-related usability catalogue which will support a specification of the usability assessment methods. We will also conduct the survey with DSL experts to systematically obtain a feedback about the feasibility and usefulness of the proposed methodology.

References

[1] Ankica Barišić. 2016. STSM Report: Evaluating the efficiency in use of search-based automated model merge technique. In *Multi-Paradigm Modelling for Cyber-Physical Systems (MPM4CPS)*. European cooperation in science and technology (COST), European cooperation in science and technology (COST). https://doi.org/10.5281/ZENODO.237420

[2] Ankica Barišić, Vasco Amaral, and Miguel Goulão. 2017. Usability Driven DSL development with USE-ME. *Computer Languages, Systems and Structures (ComLan)* (2017). https://doi.org/10.1016/j.cl.2017.06.005

[3] Ankica Barišić, Vasco Amaral, and Miguel Goulão. 2017. Usability Software Engineering - Modeling Environment (USE-ME 1.1). *Faculdade de Ciências e Tecnologia, Universidade Nova da Lisboa* (2017). https://doi.org/10.5281/ZENODO.345941

[4] Ankica Barišić, Vasco Amaral, and Miguel Goulão. 2017. *USE-ME Empirical evaluation pilot study*. Faculdade de Ciencias e Technologia, Universidade Nova da Lisboa. https://doi.org/10.5281/ZENODO.398830

[5] Ankica Barišić, Vasco Amaral, and Miguel Goulão. 2017. Visualino Companion Site. (2017). https://sites.google.com/view/vl-empiricalstudy/home

[6] Ankica Barišić, Vasco Amaral, Miguel Goulão, and Ademar Aguiar. 2014. Introducing Usability Concerns Early in the DSL Development Cycle : FlowSL Experience Report. *MD2P2 2014 Model-Driven Development Processes and Practices Workshop Proceedings, CEUR-WP* VOL-1249 (2014). https://doi.org/10.1.1.665.6015

[7] Ankica Barišić, Vasco Amaral, Miguel Goulão, and Bruno Barroca. 2011. Quality in use of domain-specific languages: a case study. *Proceedings of the 3rd ACM SIGPLAN workshop on Evaluation and usability of programming languages and tools (PLATEAU) at SPLASH* (2011), 65–72. https://doi.org/10.1145/2089155.2089170

[8] Ankica Barišić, Vasco Amaral, Miguel Goulão, and Bruno Barroca. 2012. Evaluating the Usability of Domain-Specific Languages. *Formal and Practical Aspects of Domain-Specific Languages: Recent Developments* (2012), 386–407. https://doi.org/10.4018/978-1-4666-2092-6

[9] Ankica Barišić, Dominique Blouin, Vasco Amaral, and Miguel Goulão. 2017. A Requirements Engineering Approach for Usability-Driven DSL Development. In *10th International Conference on Software Language Engineering (SLE)*. ACM, Vancouver, British Columbia, Canada.

[10] Ankica Barišić, Pedro Monteiro, Vasco Amaral, Miguel Goulão, and M P Monteiro. 2012. Patterns for Evaluating Usability of Domain-Specific Languages. *Proceedings of the 19th Conference on Pattern Languages of Programs (PLoP), SPLASH 2012* (10 2012). https://doi.org/ISBN:978-1-4503-2786-2

[11] Javier Luis Cánovas Izquierdo and Jordi Cabot. 2016. Collaboro: a collaborative (meta) modeling tool. *PeerJ Computer Science* 2 (10 2016), e84. https://doi.org/10.7717/peerj-cs.84

[12] Hyun Cho, Jeff Gray, and Eugene Syriani. 2012. Creating visual domain-specific modeling languages from end-user demonstration. In *Proceedings of the 4th International Workshop on Modeling in Software Engineering*. IEEE Press, 22–28.

[13] Pedro Gabriel, Miguel Goulão, and Vasco Amaral. 2010. Do Software Languages Engineers Evaluate their Languages?. In *XIII Congreso Iberoamericano en "Software Engineering"(CIbSE'2010), ISBN: 978-9978-325-10-0*, Xavier Franch, Itana Gimenes, and Juan-Pablo Carvallo (Eds.). Universidad del Azuay, Cuenca, Ecuador, 149–162. https://www.dropbox.com/s/uhwmwxbbya5nqcs/DoSoftwareLanguagesEngineersEvaluatetheirlanguages.pdf?dl=0

[14] Thomas R G Green and Marian Petre. 1996. Usability analysis of visual programming environments: a cognitive dimensions framework. *Journal of Visual Languages & Computing* 7, 2 (1996), 131–174.

[15] Felienne Hermans, Martin Pinzger, and Arie Van Deursen. 2009. Domain-Specific Languages in Practice: A User Study on the Success Factors. In *12th International Conference on Model Driven Engineering Languages and Systems*, Vol. 5795/2009. Lecture Notes in Computer Science, Denver, Colorado, USA, 423–437. http://link.springer.com/chapter/10.1007/978-3-642-04425-0_33

[16] Juha Kärnä, Juha-Pekka Tolvanen, and Steven Kelly. 2009. Evaluating the use of domain-specific modeling in practice. In *Proceedings of the 9th OOPSLA workshop on Domain-Specific Modeling*.

[17] Steven Kelly and Risto Pohjonen. 2009. Worst Practices for Domain-Specific Modeling. *IEEE Software* 26, 4 (2009), 22–29.

[18] Dimitrios S Kolovos, Richard F Paige, Tim Kelly, and Fiona A C Polack. 2006. Requirements for domain-specific languages. In *Proc. of ECOOP Workshop on Domain-Specific Program Development (DSPD)*, Vol. 2006.

[19] Tomaz Kosar, Sudev Bohra, and Marjan Mernik. 2016. Domain-Specific Languages: A Systematic Mapping Study. *Information and Software Technology* 71 (2016), 77–91.

[20] Marco Kuhrmann, Georg Kalus, and Alexander Knapp. 2013. Rapid Prototyping for Domain-specific Languages-From Stakeholder Analyses to Modelling Tools. *Enterprise Modelling and Information Systems Architectures* 8, 1 (2013), 62–74. https://doi.org/10.18417/emisa.8.1.4

[21] Hiroki Nishino. 2011. Misfits in abstractions: towards user-centered design in domain-specific languages for end-user programming. In *Proceedings of the ACM international conference companion on Object oriented programming systems languages and applications companion*. ACM, 215–216.

[22] Jesus Sánchez-Cuadrado, Juan De Lara, and Esther Guerra. 2012. Bottom-up meta-modelling: An interactive approach. In *MODELS (Lecture Notes in Computer Science)*, RobertB. France, JÃijrgen Kazmeier, Ruth Breu, and Colin Atkinson (Eds.), Vol. 7590 LNCS. Springer Berlin Heidelberg, 3–19. https://doi.org/10.1007/978-3-642-33666-9{_}2

[23] Maria Jose Villanueva, Francisco Valverde, and Oscar Pastor. 2014. Involving end-users in the design of a domain-specific language for the genetic domain. In *Information System Development*. Springer, 99–110.

[24] Y Wu, F Hernandez, F Ortega, P J Clarke, and R France. 2010. Measuring the Effort for Creating and Using Domain-Specific Models. In *Proceedings of the 10th Workshop on Domain-Specific Modeling*. ACM, 14.

[25] Dustin Wuest, Norbert Seyff, and Martin Glinz. 2013. Semi-automatic generation of metamodels from model sketches. In *Automated Software Engineering (ASE), 2013 IEEE/ACM 28th International Conference on*. IEEE, 664–669.