Cost-effective evolution of research prototypes into end-user tools: The MACH case study

Much of Software Engineering research needs to provide an implementation as proof-of-concept. Often such implementations are created as exploratory prototypes without polished user interfaces, making it difficult to (1) run user studies to validate the tool's contribution, (2) validate the author's claim by fellow scientists, and (3) demonstrate the utility and value of the research contribution to any interested parties. However, turning an exploratory prototype into a "proper" tool for end-users often entails great effort. Heavyweight mainstream frameworks such as Eclipse do not address this issue; their steep learning curves constitute substantial entry barriers to such ecosystems.

In this paper, we present the Model Analyzer/Checker (MACH), a stand-alone tool with a command-line interpreter. MACH integrates a set of research prototypes for analyzing UML models. By choosing a simple command line interpreter rather than (costly) graphical user interface, we achieved the core goal of quickly deploying research results to a broader audience while keeping the required effort to an absolute minimum. We analyze MACH as a case study of how requirements and constraints in an academic environment influence design decisions in software tool development. We argue that our approach while perhaps unconventional, serves its purpose with a remarkable cost-benefit ratio.

Context: Previously, we have defined the notion of diagram size and studied its impact on the understanding of UML diagrams. Subsequently, questions have been raised regarding the reliability and generality of our findings. Also, new questions arose regarding how the quality of diagrams could be defined, and how it interacts with diagram size. Goal: We pursue three goals. First, we want to increase the validity of our research by analyzing a substantially larger data set than before. Second, we broaden the generalizability of our results by including two more diagram types. Our main contribution,
though, is our third goal of extending our analysis aspects of diagram quality. Method: We improve our definition of
diagram size and add a (provisional) definition of diagram quality as the number of topographic layout flaws. We apply
these metrics on 60 diagrams of the five most commonly used types of UML diagram. We carefully analyze the structure
of our diagram samples to ensure representativeness. We correlate diagram size and layout quality with modeler
performance data obtained in previous experiments. The data set is the largest of its kind (n=156). Results: We replicate
earlier findings, and extend them to two new diagram types. We provide an improved definition of diagram size, and
provide a definition of topographic layout quality, which is one more step towards a comprehensive definition of diagram
quality as such. Both metrics are shown to be objectively applicable. We quantify the impact of diagram size and quality on
diagram understanding. Conclusions: The overall results of previous studies are confirmed, while our previous
recommendations for creating better diagrams are revised and refined.
Model Manipulation for End-User Modelers

End-user modelers are domain experts who create and use models as part of their work. They are typically not Software Engineers, and have little or no programming and meta-modeling experience. However, using model manipulation languages developed in the context of Model-Driven Engineering often requires such experience. These languages are therefore only used by a small subset of the modelers that could, in theory, benefit from them.

The goals of this thesis are to substantiate this observation, introduce the concepts and tools required to overcome it, and provide empirical evidence in support of these proposals. To achieve its first goal, the thesis presents the findings of a Systematic Mapping Study showing that human factors topics are scarcely and relatively poorly addressed in model transformation research. Motivated by these findings, the thesis explores the requirements of end-user modelers, and proposes the VM* family of model manipulation languages addressing them. This family consists of the Visual Model Query Language (VMQL), the Visual Model Constraint Language (VMCL), and the Visual Model Transformation Language (VMTL). They allow modelers to specify and execute queries, constraints, and transformations using their modeling notation and editor of choice.

The VM* languages are implemented via a single execution engine, the VM* Runtime, built on top of the Henshin graph-based transformation engine. This approach combines the benefits of flexibility, maturity, and formality. To simplify model editor integration, the VM* Runtime is deployed as a collection of lightweight Web Services. The claim that VM* languages offer end-user modelers superior learnability compared to existing model manipulation languages is verified empirically via user experiments complemented by qualitative evidence.

General information

State: Published
Organisations: Department of Applied Mathematics and Computer Science, Software Engineering
Authors: Acretoaie, V. (Intern), Störrle, H. (Intern)
Number of pages: 254
Publication date: 2016

Publication information
Place of publication: Kgs. Lyngby
Publisher: Technical University of Denmark (DTU)
Original language: English

Series: DTU Compute PHD-2016
Number: 405
ISSN: 0909-3192
Main Research Area: Technical/natural sciences
Electronic versions: phd405_Acretoaie_V.pdf

Relations
Projects:
Model Manipulation for End-User Modelers
Publication: Research › Ph.D. thesis – Annual report year: 2016

On the impact of size to the understanding of UML diagrams

Background: Practical experience suggests that usage and understanding of UML diagrams is greatly affected by the quality of their layout. While existing research failed to provide conclusive and comprehensive evidence in support of this hypothesis, our own previous work provided substantial evidence to this effect, also suggesting diagram size as a relevant factor, for a range of diagram types and layouts.

Aims: Since there is no generally accepted precise notion of “diagram size,” we first need to operationalize this concept, analyze its impact on diagram understanding, and derive practical advice from our findings.

Method: We define three alternative, plausible metrics. Since they are all highly correlated on a large sample of UML
diagrams, we opt for the simplest one. We use it to re-analyze existing experimental data on diagram understanding.

Results: We find a strong negative correlation between diagram size and modeler performance. Our results are statistically highly significant and exhibit a very large degree of validity. We utilize these results to derive a recommendation on diagram sizes that are, on average, optimal for model understanding. These recommendations are implemented in a plug-in to a widely used modeling tool, providing continuous feedback about diagram size to modelers.

Conclusions: The effect sizes are varying, but generally suggest that the impact of size matches or exceeds that of other factors in diagram understanding. With the guideline and tool, modelers are steered toward avoiding too large diagrams.

General information
State: Published
Organisations: Department of Applied Mathematics and Computer Science , Software Engineering
Authors: Störrle, H. (Intern)
Number of pages: 20
Publication date: 2016
Main Research Area: Technical/natural sciences

Publication information
Journal: Software and Systems Modeling
ISSN (Print): 1619-1366
Ratings:
BFI (2018): BFI-level 2
BFI (2017): BFI-level 2
Web of Science (2017): Indexed yes
BFI (2016): BFI-level 2
Scopus rating (2016): CiteScore 1.65 SJR 0.461 SNIP 1.706
Web of Science (2016): Indexed yes
BFI (2015): BFI-level 2
Scopus rating (2015): SJR 0.445 SNIP 1.846 CiteScore 1.53
Web of Science (2015): Indexed yes
BFI (2014): BFI-level 2
Scopus rating (2014): SJR 0.454 SNIP 1.822 CiteScore 1.41
Web of Science (2014): Indexed yes
BFI (2013): BFI-level 2
Scopus rating (2013): SJR 0.778 SNIP 3.148 CiteScore 2.25
ISI indexed (2013): ISI indexed yes
Web of Science (2013): Indexed yes
BFI (2012): BFI-level 2
Scopus rating (2012): SJR 0.935 SNIP 2.503 CiteScore 2.66
ISI indexed (2012): ISI indexed yes
BFI (2011): BFI-level 2
Scopus rating (2011): SJR 0.829 SNIP 2.178 CiteScore 2.28
ISI indexed (2011): ISI indexed no
BFI (2010): BFI-level 2
Scopus rating (2010): SJR 0.946 SNIP 2.311
BFI (2009): BFI-level 2
Scopus rating (2009): SJR 1.051 SNIP 2.414
BFI (2008): BFI-level 2
Scopus rating (2008): SJR 1.501 SNIP 3.344
Scopus rating (2007): SJR 0.928 SNIP 4.07
Scopus rating (2006): SJR 1.455 SNIP 6.799
Original language: English
Diagram understanding, Diagram size metrics, Cognitive load, Experiment, Gestalt principles
Electronic versions:
VL_4d2_Size.pdf. Embargo ended: 21/05/2017
DOIs:
10.1007/s10270-016-0529-x
VMTL: a language for end-user model transformation

Model transformation is a key enabling technology of Model-Driven Engineering (MDE). Existing model transformation languages are shaped by and for MDE practitioners—a user group with needs and capabilities which are not necessarily characteristic of modelers in general. Consequently, these languages are largely ill-equipped for adoption by end-user modelers in areas such as requirements engineering, business process management, or enterprise architecture. We aim to introduce a model transformation language addressing the skills and requirements of end-user modelers. With this contribution, we hope to broaden the application scope of model transformation and MDE technology in general. We discuss the profile of end-user modelers and propose a set of design guidelines for model transformation languages addressing them. We then introduce Visual Model Transformation Language (VMTL) following these guidelines. VMTL draws on our previous work on the usability-oriented Visual Model Query Language. We implement VMTL using the Henshin model transformation engine, and empirically investigate its learnability via two user experiments and a think-aloud protocol analysis. Our experiments, although conducted on computer science students exhibiting only some of the characteristics of end-user modelers, show that VMTL compares favorably in terms of learnability with two state-of-the-art model transformation languages: Epsilon and Henshin. Our think-aloud protocol analysis confirms many of the design decisions adopted for VMTL, while also indicating possible improvements.

General information
State: Published
Organisations: Department of Applied Mathematics and Computer Science, Software Engineering, Philipps-Universität Marburg
Authors: Acretoaie, V. (Intern), Störrle, H. (Intern), Strüber, D. (Ekstern)
Number of pages: 29
Publication date: 2016
Main Research Area: Technical/natural sciences

Publication information
Journal: Software and Systems Modeling
ISSN (Print): 1619-1366
Ratings:
BFI (2018): BFI-level 2
BFI (2017): BFI-level 2
Web of Science (2017): Indexed yes
BFI (2016): BFI-level 2
Scopus rating (2016): CiteScore 1.65 SJR 0.461 SNIP 1.706 Web of Science (2016): Indexed yes
BFI (2015): BFI-level 2
Scopus rating (2015): SJR 0.445 SNIP 1.846 CiteScore 1.53 Web of Science (2015): Indexed yes
BFI (2014): BFI-level 2
Scopus rating (2014): SJR 0.454 SNIP 1.822 CiteScore 1.41 Web of Science (2014): Indexed yes
BFI (2013): BFI-level 2
BFI (2012): BFI-level 2
Effective and efficient model clone detection

Code clones are a major source of software defects. Thus, it is likely that model clones (i.e., duplicate fragments of models) have a significant negative impact on model quality, and thus, on any software created based on those models, irrespective of whether the software is generated fully automatically ("MDD-style") or hand-crafted following the blueprint defined by the model ("MBSD-style"). Unfortunately, however, model clones are much less well studied than code clones. In this paper, we present a clone detection algorithm for UML domain models. Our approach covers a much greater variety of model types than existing approaches while providing high clone detection rates at high speed.

General information
State: Published
Organisations: Department of Informatics and Mathematical Modeling, Department of Applied Mathematics and Computer Science, Software Engineering
Authors: Störrle, H. (Intern)
Pages: 440-457
Publication date: 2015

Host publication information
Title of host publication: Software, Services, and Systems: Essays Dedicated to Martin Wirsing on the Occasion of His Retirement from the Chair of Programming and Software Engineering
Publisher: Springer
Editors: De Nicola, R., Hennicker, R.
ISBN (Print): 978-3-319-15544-9
ISBN (Electronic): 978-3-319-15545-6

Series: Lecture Notes in Computer Science
Volume: 8950
ISSN: 0302-9743
Main Research Area: Technical/natural sciences
Electronic versions:
Effective_and_Efficient_Model_Clone_Detection_MC_1d_CloneDetection.pdf. Embargo ended: 01/04/2016
DOIs:
10.1007/978-3-319-15545-6_25
Source: FindIt
Source-ID: 274275949
Publication: Research - peer-review › Book chapter – Annual report year: 2015

Efficient Model Querying with VMQL

Context: Despite model querying being an important practical problem, existing solutions lack either usability, expressiveness, or generality. The Visual Model Query Language (VMQL) is a query by example solution created to satisfy these requirements simultaneously. Objective: In the present paper we study whether VMQL queries can be executed in an efficient way, such that VMQL is suitable for ad-hoc model querying in practical settings involving large models. Method: We study VMQL query execution performance on sets of models ranging over a broad spectrum of sizes and degrees of complexity. The models are based on large and realistic case studies. Results: We observe that our approach exhibits competitive performance, while providing superior usability and generality.

General information
State: Published
Organisations: Department of Applied Mathematics and Computer Science, Software Engineering
First International Workshop on Human Factors in Modeling (HuFaMo 2015): Preface

Modeling is a human-intensive enterprise. As such, many research questions related to modeling can only be answered by empirical studies employing human factors. The International Workshop Series on Human Factors in Modeling (HuFaMo) is dedicated to the discussion of empirical research involving human factors in modeling. Our goal is to improve the state of the science and professionalism in empirical research in the Model Based Engineering community. Typical examples of research questions might consider the usability of a certain approach, such as a method or language, or the emotional states or personal judgements of modelers.

While concerned with foundations and framework support for modeling, the community has been somehow neglecting the issue of human factors in this context. There is a growing need from the community concerned with quality factors to understand the best practices and systematic approaches to assert usability in modeling and confirm the claims of productivity. This workshop creates a space for discussion being a get together of both MDE, Usability, Human Interfaces and the Experimental Software engineering community.

HuFaMo expressly focuses on human factors, in order to raise the awareness for these topics and the associated research methods and questions in the modeling community, providing an outlet for research of this type, guaranteeing high quality reviews by people that apply these research methods themselves. Along with fully complete empirical evaluations, the workshop organizers explicitly encouraged researchers new to empirical methods to discuss study designs before conducting their empirical evaluations. The rationale was to create a constructive environment where the HuFaMo participants could contribute to improving the proposed study designs so that stronger (and more easily replicable) empirical designs and results can be obtained. Ultimately, we aim to congregate a community of researchers and practitioners that promotes (possibly independently replicated) empirical assessments on claims related to human factors in modeling.
Transparent Model Transformation: Turning Your Favourite Model Editor into a Transformation Tool

Current model transformation languages are supported by dedicated editors, often closely coupled to a single execution engine. We introduce Transparent Model Transformation, a paradigm enabling modelers to specify transformations using a familiar tool: their model editor. We also present VMTL, the first transformation language implementing the principles of Transparent Model Transformation: syntax, environment, and execution transparency. VMTL works by weaving a transformation aspect into its host modeling language. We show how our implementation of VMTL turns any model editor into a flexible model transformation tool sharing the model editor’s benefits, transparently.

General information
Efficient Representation of Timed UML 2 Interactions

UML 2 interactions describe system behavior over time in a declarative way. The standard approach to defining their formal semantics enumerates traces of events; other representation formats, like Büchi automata or prime event structures, have been suggested, too. We describe another, more succinct format, interaction structures, which is based on asymmetric event structures. It simplifies the integration of real time, and complex operators like alt and break, and leads to an efficient semantic representation of interactions. We provide the formalism, and a prototypical implementation highlighting the benefits of our approach.
Pen and paper is still the best tool for sketching GUIs. However, sketches cannot be executed, at best we have facilitated or animated scenarios. The Advanced User Interaction Environment facilitates turning hand-drawn sketches into executable prototypes.
Hypersonic: Model Analysis and Checking in the Cloud

Context: Modeling tools are traditionally delivered as monolithic desktop applications, optionally extended by plug-ins or special purpose central servers. This delivery model suffers from several drawbacks, ranging from poor scalability to difficult maintenance and the proliferation of ‘shelfware’. Objective: In this paper we investigate the conceptual and technical feasibility of a new software architecture for modeling tools, where certain advanced features are factored out of the client and moved towards the Cloud. With this approach we plan to address the above mentioned drawbacks of existing modeling tools.

Method: We base our approach on RESTful Web services. Using features implemented in the existing Model Analysis and Checking (MACH) tool, we create a RESTful Web service API offering model analysis facilities. We refer to it as the Hypersonic API. We provide a proof of concept implementation for the Hypersonic API using model clone detection as our example case. We also implement a sample Web application as a client for these Web services.

Results: Our initial experiments with Hypersonic demonstrate the viability of our approach. By applying standards such as REST and JSON in combination with Prolog as an implementation language, we are able to transform MACH from a command line tool into the first Web-based model clone detection service with remarkably little effort.

General information
State: Published
Organisations: Department of Applied Mathematics and Computer Science, Software Engineering
Authors: Acretoaie, V. (Intern), Störrle, H. (Intern)
Number of pages: 8
Pages: 3-13
Publication date: 2014

Host publication information
Title of host publication: Proceedings of the 2nd Workshop on Scalability in Model Driven Engineering (BigMDE 2014)
Editors: Kolovos, D., Di Ruscio, D., Matragkas, N., De Lara, J., Ráth, I., Tisi, M.
Series: CEUR Workshop Proceedings
Number: 1206
ISSN: 1613-0073
Main Research Area: Technical/natural sciences
Conference: 2nd Workshop on Scalability in Model Driven Engineering 2014, York, United Kingdom, 24/07/2014
Hypersonic, MACH, Models in the Cloud, Clone detection, Web services, Prolog
Electronic versions:
MT_1a_MACH_in_the_Cloud.pdf
bigmde_2014_hypersonic.pdf

Links:
http://ceur-ws.org/Vol-1206/

Source: PublicationPreSubmission
Source-ID: 98696879
Publication: Research - peer-review » Article in proceedings – Annual report year: 2014
Hypersonic - Model Analysis as a Service
Hypersonic is a Cloud-based tool that proposes a new approach to the deployment of model analysis facilities. It is implemented as a RESTful Web service API offering analysis features such as model clone detection. This approach allows the migration of resource intensive analysis algorithms from monolithic desktop modeling tools to a wide range of mobile and Web-based clients. As a technology demonstrator, a Web application acting as a client for the Hypersonic API has been implemented and made publicly available.

General information
State: Published
Organisations: Department of Applied Mathematics and Computer Science, Software Engineering
Authors: Acretoaie, V. (Intern), Störrle, H. (Intern)
Number of pages: 5
Pages: 1-5
Publication date: 2014

Modeling Requirements with RED
General information
State: Published
Organisations: Department of Applied Mathematics and Computer Science, Software Engineering
Authors: Störrle, H. (Intern)
Pages: 26-30
Publication date: 2014

Host publication information
Title of host publication: Poster Session and ACM SRC of MODELS 2014: Joint Proceedings of MODELS 2014 Poster Session and ACM Student Research Competition
Editors: Sauer, S., Wimmer, M., Genero, M., Qadeer, S.
Series: CEUR Workshop Proceedings
Volume: 1258
ISSN: 1613-0073
BFI conference series: Model Driven Engineering Languages and Systems (5000533)
Main Research Area: Technical/natural sciences
Electronic versions:
TSE_3B_RED.pdf
Links:
http://ceur-ws.org/Vol-1258/
Source: PublicationPreSubmission
Source-ID: 100853133
Publication: Research - peer-review › Article in proceedings – Annual report year: 2014
On the Impact of Diagram Layout: How Are Models Actually Read?
This poster presents the latest results from a very large eye tracking study (n=29) that explores how modelers read UML diagrams. We find that various factors like layout quality, modeler experience, and diagram type lead to significant differences in diagram reading strategies. We derive elements of a theory of diagram reading behavior from our findings. This paper presents only late breaking results: all findings presented, theories constructed, and conclusions drawn are of a preliminary nature. This paper does not present the amount and degree of evidence that would allow us to consider the contents as being scientifically validated, yet.

General information
State: Published
Organisations: Department of Applied Mathematics and Computer Science, Software Engineering, Department of Management Engineering, Production and Service Management, Engineering Systems Group, Technical University of Denmark
Authors: Störrle, H. (Intern), Baltsen, N. (Ekstern), Christoffersen, H. (Ekstern), Maier, A. (Intern)
Pages: 31-35
Publication date: 2014

Host publication information
Title of host publication: Poster Session and ACM SRC of MODELS 2014: Joint Proceedings of MODELS 2014 Poster Session and ACM Student Research Competition
Editors: Sauer, S., Wimmer, M., Genero, M., Qadeer, S.
Series: CEUR Workshop Proceedings
Volume: 1258
ISSN: 1613-0073
BFI conference series: Model Driven Engineering Languages and Systems (5000533)
Main Research Area: Technical/natural sciences
Electronic versions:
VL_4e_Behavior_Poster.pdf
Links:
http://ceur-ws.org/Vol-1258/
Publication: Research - peer-review › Article in proceedings – Annual report year: 2014

On the Impact of Layout Quality to Understanding UML Diagrams: Size Matters
Practical experience suggests that usage and understanding of UML diagrams is greatly affected by the quality of their layout. While existing research failed to provide conclusive evidence in support of this hypothesis, our own previous work provided substantial evidence to this effect. When studying different factors like diagram type and expertise level, it became apparent that diagram size plays an important role, too. Since we lack an adequate understanding of this notion, in this paper, we define diagram size metrics and study their impact to modeler performance. We find that there is a strong negative correlation between diagram size and modeler performance. Our results are highly significant. We utilize these results to derive a recommendation on diagram sizes that are optimal for model understanding.

General information
State: Published
Organisations: Department of Applied Mathematics and Computer Science, Software Engineering
Authors: Störrle, H. (Intern)
Number of pages: 17
Pages: 518-534
Publication date: 2014

Host publication information
Title of host publication: Proceedings of 17th International Conference on Model Driven Engineering Languages and Systems
Publisher: Springer
Editors: Dingel, J., Schulte, W., Ramos, I., Abrahao, S., Insfran, E.
ISBN (Print): 978-3-319-11652-5
ISBN (Electronic): 978-3-319-11653-2
Series: Lecture Notes in Computer Science
Volume: 8767
ISSN: 0302-9743
BFI conference series: Model Driven Engineering Languages and Systems (5000533)
Main Research Area: Technical/natural sciences
Towards Diagram Understanding: A Pilot Study Measuring Cognitive Workload Through Eye-Tracking

We investigate model understanding, in particular, how the quality of the UML diagram layout impacts cognitive load. We hypothesize that this will have a significant impact on the structure and effectiveness of engineers' communication. In previous work, we have studied task performance measurements and subjective assessments; here, we also investigate behavioral indicators such as fixation and pupillary dilation. We use such indicators to explore diagram understanding- and reading strategies and how such strategies are impacted, e.g. by diagram type and expertise level. In the pilot eye-


General information
State: Published
Organisations: Department of Applied Mathematics and Computer Science, Software Engineering
Authors: Carré, B. (ed.) (Ekstern), Sahroui, H. (ed.) (Ekstern), Störrle, H. (ed.) (Intern)
Number of pages: 52
Publication date: 2014

Publication information
Publisher: Technical University of Denmark (DTU)
ISBN (Print): 978-87-643-1188-4
Original language: English
Number: 01
ISSN: 1601-2321
Main Research Area: Technical/natural sciences
Electronic versions:
tr14_01_Storrle_H.pdf
Publication: Research › Report – Annual report year: 2014

The Requirements Editor RED

General information
State: Published
Organisations: Department of Applied Mathematics and Computer Science, Software Engineering, Technical University of Denmark
Authors: Störrle, H. (Intern), Kucharek, M. (Ekstern)
Pages: 33-35
Publication date: 2014

Host publication information
Title of host publication: Proceedings of the joint track "Tools", "Demos", and "Posters" of ECOOP, ECSA, and ECMFA, 2013
Publisher: Technical University of Denmark (DTU)
ISBN (Print): 978-87-643-1188-4
Number: 01
ISSN: 1601-2321
Main Research Area: Technical/natural sciences
Electronic versions:
TSE_3a_RED.pdf
Links:
http://orbit.dtu.dk/files/83225486/tr14_01_Storrle_H.pdf
Publication: Research › peer-review › Article in proceedings – Annual report year: 2014

Towards Diagram Understanding: A Pilot Study Measuring Cognitive Workload Through Eye-Tracking

We investigate model understanding, in particular, how the quality of the UML diagram layout impacts cognitive load. We hypothesize that this will have a significant impact on the structure and effectiveness of engineers’ communication. In previous work, we have studied task performance measurements and subjective assessments; here, we also investigate behavioral indicators such as fixation and pupillary dilation. We use such indicators to explore diagram understanding- and reading strategies and how such strategies are impacted, e.g. by diagram type and expertise level. In the pilot eye-

Towards Diagram Understanding: A Pilot Study Measuring Cognitive Workload Through Eye-Tracking

We investigate model understanding, in particular, how the quality of the UML diagram layout impacts cognitive load. We hypothesize that this will have a significant impact on the structure and effectiveness of engineers’ communication. In previous work, we have studied task performance measurements and subjective assessments; here, we also investigate behavioral indicators such as fixation and pupillary dilation. We use such indicators to explore diagram understanding- and reading strategies and how such strategies are impacted, e.g. by diagram type and expertise level. In the pilot eye-
tracking experiment run so far, we have only examined a small number of participants (n=4), so our results are preliminary in nature and do not afford far reaching conclusions. They do, however, corroborate findings from earlier experiments, for example, showing that layout quality indeed matters and improves understanding. Our results also give rise to a number of new hypotheses about diagram understanding strategies that we are investigating in an ongoing data acquisition campaign.

General information
State: Published
Organisations: Department of Management Engineering, Production and Service Management, Engineering Systems Group, Department of Applied Mathematics and Computer Science, Software Engineering, Technical University of Denmark
Authors: Maier, A. (Intern), Baltsen, N. (Ekstern), Christoffersen, H. (Ekstern), Störrle, H. (Intern)
Number of pages: 6
Publication date: 2014

Host publication information
Title of host publication: Proceedings of International Conference on Human Behaviour in Design 2014
Article number: HBID2014/114
Main Research Area: Technical/natural sciences
Conference: International Conference on Human Behaviour in Design 2014, Ascona, Switzerland, 14/10/2014 - 14/10/2014
Design cognition, Design communication, Diagram understanding, Eye-tracking, Layout quality, Model comprehension, Model-based systems engineering, UML, Visual attention
Electronic versions:
2014_MaierBaltsenChristoffersenStoerrle_Towards_Diagram_Understanding_HBiD_Ascona_Paper_No.pdf

Relations
Activities:
International Conference on Human Behaviour in Design 2014
Source: PublicationPreSubmission
Source-ID: 93363464
Publication: Research - peer-review › Article in proceedings – Annual report year: 2014

Improving modeling with layered UML diagrams
Layered diagrams are diagrams whose elements are organized into sets of layers. Layered diagrams are routinely used in many branches of engineering, except Software Engineering. In this paper, we propose to add layered diagrams to UML modeling tools, and elaborate the concept by exploring usage scenarios. We validate the concept by implementation, lab assessments, and field testing. We conclude that layers enhance and complement conventional diagrams and model structuring techniques, are easy to add to existing modeling infrastructure, and are easy to apply by modelers.

General information
State: Published
Organisations: Department of Applied Mathematics and Computer Science, Software Engineering
Authors: Störrle, H. (Intern)
Pages: 206-209
Publication date: 2013

Host publication information
Title of host publication: MODELSWARD 2013 - Proceedings of the 1st International Conference on Model-Driven Engineering and Software Development
Publisher: SciTePress
Main Research Area: Technical/natural sciences
Conference: 1st International Conference on Model-Driven Engineering and Software Development (MODELSWARD 2013), Barcelona, Spain, 19/02/2013 - 19/02/2013
Source: dtu
Source-ID: n:oai:DTIC-ART:compendex/387417477::28879
Publication: Research - peer-review › Article in proceedings – Annual report year: 2013

Improving the Usability of OCL as an Ad-hoc Model Querying Language
The OCL is often perceived as difficult to learn and use. In previous research, we have dened experimental query languages exhibiting higher levels of usability than OCL. However, none of these alternatives can rival OCL in terms of adoption and support. In an attempt to leverage the lessons learned from our research and make it accessible to the OCL community, we propose the OCL Query API (OQAPI), a library of query-predicates to improve the user-friendliness of OCL for ad-hoc querying. The usability of OQAPI is studied using controlled experiments. We nd considerable evidence to support our claim that OQAPI facilitates user querying using OCL.
Making sense to modelers: Presenting UML class model differences in prose

Understanding the difference between two models, such as different versions of a design, can be difficult. It is a commonly held belief in the model differencing community that the best way of presenting a model difference is by using graph or tree-based visualizations. We disagree and present an alternative approach where sets of low-level model differences are abstracted into high-level model differences that lend themselves to being presented textually. This format is informed by an explorative survey to elicit the change descriptions modelers use themselves. Our approach is validated by a controlled experiment that tests three alternatives to presenting model differences. Our findings support our claim that the approach presented here is superior to EMF Compare.

MOCQL: A Declarative Language for Ad-Hoc Model Querying

This paper starts from the observation that existing model query facilities are not easy to use, and are thus not suitable for users without substantial IT/Computer Science background. In an attempt to highlight this issue and explore alternatives, we have created the Model Constraint and Query Language (MOCQL), an experimental declarative textual language to express queries (and constraints) on models. We introduce MOCQL by examples and its grammar, evaluate its usability by means of controlled experiments, and find that modelers perform better and experience less cognitive load when working with MOCQL than when working with OCL. While MOCQL is currently only implemented and validated for the different notations defined by UML, its concepts should be universally applicable.
Querying Business Process Models with VMQL

The Visual Model Query Language (VMQL) has been invented with the objectives (1) to make it easier for modelers to query models effectively, and (2) to be universally applicable to all modeling languages. In previous work, we have applied VMQL to UML, and validated the first of these two claims. In this paper, we apply VMQL to the Business Process Modeling Notation (BPMN) to evaluate the second claim. We explore the adaptations required, and re-evaluate the usability of VMQL in this context. We find similar results to earlier work, thus both supporting our claims and establishing the usability of VMQL beyond the realm of UML.

General information
State: Published
Organisations: Department of Applied Mathematics and Computer Science, Software Engineering
Authors: Störrle, H. (Intern), Acretoaie, V. (Intern)
Number of pages: 10
Publication date: 2013

Towards an Operationalization of the “Physics of Notations” for the Analysis of Visual Languages

We attempt to validate the conceptual framework “Physics of Notation” (PoN) as a means for analysing visual languages by applying it to UML Use Case Diagrams. We discover that the PoN, in its current form, is neither precise nor comprehensive enough to be applied in an objective way to analyse practical visual software engineering notations. We propose an operationalization of a part of the PoN, highlight conceptual shortcomings of the PoN, and explore ways to address them.

General information
State: Published
Organisations: Department of Applied Mathematics and Computer Science, Software Engineering, University of Brighton
Authors: Störrle, H. (Intern), Fish, A. (Forskerdatabase)
Pages: 104-120
Publication date: 2013
Towards Clone Detection in UML Domain Models

Code clones (i.e., duplicate fragments of code) have been studied for long, and there is strong evidence that they are a major source of software faults. Anecdotal evidence suggests that this phenomenon occurs similarly in models, suggesting that model clones are as detrimental to model quality as they are to code quality. However, programming language code and visual models have significant differences that make it difficult to directly transfer notions and algorithms developed in the code clone arena to model clones. In this article, we develop and propose a definition of the notion of “model clone” based on the thorough analysis of practical scenarios. We propose a formal definition of model clones, specify a clone detection algorithm for UML domain models, and implement it prototypically. We investigate different similarity heuristics to be used in the algorithm, and report the performance of our approach. While we believe that our approach advances the state of the art significantly, it is restricted to UML models, its results leave room for improvements, and there is no validation by field studies.

General information
State: Published
Organisations: Software Engineering, Department of Informatics and Mathematical Modeling
Authors: Störrie, H. (Intern)
Pages: 307-329
Publication date: 2013
Main Research Area: Technical/natural sciences

Publication information
Journal: Software and Systems Modeling
Volume: 12
Issue number: 2
ISSN (Print): 1619-1366
Ratings:
BFI (2018): BFI-level 2
BFI (2017): BFI-level 2
Web of Science (2017): Indexed yes
BFI (2016): BFI-level 2
Scopus rating (2016): CiteScore 1.65 SJR 0.461 SNIP 1.706
Web of Science (2016): Indexed yes
BFI (2015): BFI-level 2
Scopus rating (2015): SJR 0.445 SNIP 1.846 CiteScore 1.53
Web of Science (2015): Indexed yes
BFI (2014): BFI-level 2
Scopus rating (2014): SJR 0.454 SNIP 1.822 CiteScore 1.41
Web of Science (2014): Indexed yes
BFI (2013): BFI-level 2
Scopus rating (2013): SJR 0.778 SNIP 3.148 CiteScore 2.25
ISI indexed (2013): ISI indexed yes
Web of Science (2013): Indexed yes
Model maintenance, Model clones, Model similarity, Model evolution, Model management

Electronic versions:
MC-1b-CloneDetection.pdf

DOIs:
10.1007/s10270-011-0217-9

Bibliographical note
The original publication is available at www.springerlink.com

Source: orbit
Source-ID: 313345
Publication: Research - peer-review › Journal article – Annual report year: 2012

Modelling Foundations and Applications: 8th European Conference, ECMFA 2012; Kgs. Lyngby, Denmark, July 2-5, 2012; Proceedings
The 20 revised full foundations track papers and 10 revised full applications track papers presented were carefully reviewed and selected from 81 submissions. Papers on all aspects of MDE were received, including topics such as architectural modelling and product lines, code generation, domain-specific modeling, metamodeling, model analysis and verification, model management, model transformation and simulation. The breadth of topics, as well as the high quality of the results presented in these accepted papers, demonstrate the maturity and vibrancy of the field.

General information
State: Published
Organisations: Department of Informatics and Mathematical Modeling, Computer Science and Engineering, Software Engineering, University of York, Universidad de Malaga, MetaCase
Number of pages: 446
Publication date: 2012

Publication information
Publisher: Springer
ISBN (Print): 978-3-642-31490-2
ISBN (Electronic): 978-3-642-31491-9
Original language: English

Series: Lecture Notes in Computer Science
Volume: 7349
ISSN: 0302-9743
Main Research Area: Technical/natural sciences
DOIs:
10.1007/978-3-642-31491-9
Publication: Research › Book – Annual report year: 2012
MQ-2 A Tool for Prolog-based Model Querying

MQ-2 integrates a Prolog console into the MagicDraw building modeling environment and equips this console with features targeted specifically to the task of querying models. The vision of MQ-2 is to make Prolog-based model querying accessible to both student and expert modelers by offering powerful query features and a tight integration with the host modeling environment.

On the Impact of Layout Quality to Understanding UML Diagrams: Diagram Type and Expertise

Practical experience suggests that the use and understanding of UML diagrams is greatly affected by the quality of their layout. In previous work, we have presented evidence supporting this intuition. This contrasts with earlier experiments that yielded weak or inconclusive evidence only. In the current paper, we expand on our earlier experiments by varying both diagram types and populations studied. We find no difference in the beneficial evidence of good layout wrt. diagram types. We also find support for the hypothesis that experts benefit less than novices. While still lacking independent replication of our earlier results, these results add further evidence in support of our hypothesis.

Expressing Model Constraints Visually with VMQL

OCL is the de facto standard language for expressing constraints and queries on UML models. However, OCL expressions are very difficult to create, understand, and maintain, even with the sophisticated tool support now available. In this paper, we propose to use the Visual Model Query Language (VMQL) for specifying constraints on UML models. We examine VMQL’s usability by controlled experiments and its expressiveness by a representative sample. We conclude that VMQL is less expressive than OCL, although expressive enough for most of the constraints in the sample. In terms of usability, however, VMQL is superior to OCL, although the experimental evidence we present here is not as compelling as
the one we presented when comparing VMQL and OCL on model querying.

**On the impact of layout quality to understanding UML diagrams**

Practical experience suggests that use and understanding of UML diagrams is greatly affected by the quality of their layout. However, existing experimental evidence for this effect is been weak and inconclusive. In this paper, we explore two explanations. Firstly, we observe that the visual qualities of diagrams are more prominent in earlier life cycle phases so that the impact of layout quality should be more apparent in models and diagram types used there, an aspect not studied in previous research. Secondly, in practice, good layouts use many different heuristics simultaneously whereas previous research considered them in isolation only. In this paper, we report the results of a series of controlled experiments using compound layouts on requirements analysis models. With very high significance, we find a notable impact of the layout quality measured by different aspects of cognitive load.
VMQL: A Generic Visual Model Query Language

Shifting the focus from code to models in software development brings into view model-related tasks such as querying which are not very well supported by current CASE tools. Existing textual query languages like OCL are often not acceptable for domain modelers. Also, most query languages suffer from a mismatch between models, queries, and results. The visual model query language (VMQL) tries to overcome this by using a modeling language also as the query language and result presentation language.

VMQL: A Visual Language for Ad-Hoc Model Querying

In large scale model based development, analysis level models are more like knowledge bases than engineering artifacts. Their effectiveness depends, to a large degree, on the ability of domain experts to retrieve information from them ad hoc. For large scale models, however, existing query facilities are inadequate. The Visual Model Query Language (VMQL) is a novel approach that uses the respective modeling language of the source model as the query language, too. The semantics of VMQL is defined formally based on graphs, so that query execution can be defined as graph matching. VMQL has been applied to several visual modeling languages, implemented, and validated in small case studies, and several controlled experiments.
What are the Characteristics of Engineering Design Processes?

This paper studies the characteristic properties of Engineering Design (ED) processes from a process modelling perspective. In a first step, we extracted nine characteristics of engineering design processes from the literature and in a second step validated the findings using results from our survey among academic and industrial ED process modelling experts. In a third step, we added a further nine characteristics from personal experiences in the Language Engineering Domain to capture the pragmatic perspective. We arrive at a comprehensive set of 18 characteristics grouped into 6 challenges for process modelling in the engineering design domain. The challenges process modelers need to address when using and developing process modelling approaches and tools are: Development, Collaboration, Products &
We then compare the importance of elicited and suggested challenges and characteristics within engineering design with software engineering and business process modelling and discuss similarities and differences.

Model driven development of user interface prototypes: an integrated approach
Many approaches to interface development apply only to isolated aspects of the development of user interfaces (UIs), e.g., exploration during the early phases, design of visual appearance, or implementation in some technology. In this paper we explore an integrated approach to incorporate the whole UI development life cycle, connect all stakeholders involved, and support a wide range of levels of granularity and abstraction. This is achieved by using Window/Event-Diagrams (WEDs), a UI specification notation based on UML 2 state machines. It affords closer collaboration between different user groups like graphic designers and software developers by integrating traditional pen-and-paper based methods with contemporary MDA-based CASE tools. We have implemented our approach in the Advanced Interaction Design Environment (AIDE), an application to support WEDs.

Structuring very large domain models: Experiences from industrial MDSD projects
View/Viewpoint approaches like IEEE 1471-2000, or Kruchten’s 4+1-view model are used to structure software architectures at a high level of granularity. While research has focused on architectural languages and with consistency between multiple views, practical questions such as the structuring at a lower level of detail have not been dealt with. This paper aims at filling this gap by reporting personal experiences from a very large scale industrial domain modeling project. There, structuring the logical view turned out to be a critical success factor. We explain the project and its setting, analyze the role and repercussions of model structuring, and examine the implications model structuring decisions have on other parts of the project. We then explain the model structure abstracted from a very large scale industrial modeling project. Finally, we discuss lessons learned.
Towards Clone Detection in UML Domain Models

Code clones - that is, duplicate fragments of code - have been studied for a long time. There is strong evidence that code clones are a major source of software faults. Anecdotal evidence suggests that this phenomenon is not restricted to code, but occurs in models in a very similar way. So it is likely that model clones are as detrimental to model quality as they are to code quality. However, programming language code and visual models also have significant differences so that notions and algorithms developed in the code clone arena cannot be transferred directly to model clones. In this article, we discuss how model clones arise by analyzing several practical scenarios. We propose a formal definition of models and clones, that allows us to specify a generic clone detection algorithm. Through a thorough analysis of the detail structure of sample UML domain models, recommendations for clone detection algorithms are derived. We investigate different algorithms and heuristics to detect clones, some of which we have implemented in the MQ_lone tool (pronounced "m clone").

A Logical Model Query Interface

General information
State: Published
Organisations: Software Engineering, Department of Informatics and Mathematical Modeling
Authors: Störrle, H. (Intern)
Pages: 49-54
Publication date: 2010

Host publication information
Title of host publication: Proc. 4th European Conference on Software Architecture (ECSA’10)
Publisher: ACM
Main Research Area: Technical/natural sciences
Conference: 4th European Conference on Software Architecture, Copenhagen, Denmark, 23/08/2010 - 23/08/2010
Source: orbit
Source-ID: 270983
Publication: Research - peer-review › Article in proceedings – Annual report year: 2010

DOIs:
10.1145/1842752.1842805

General information
State: Published
Organisations: Software Engineering, Department of Informatics and Mathematical Modeling
Authors: Störrle, H. (Intern)
Pages: 285-293
Publication date: 2010

Host publication information
Title of host publication: Proc. 4th European Conference on Software Architecture (ECSA’10) : Proc. 8th Nordic Workshop on Model Driven Engineering (NW-MODE’10)
Volume: Companion Volume
Publisher: ACM
ISBN (Print): 9781450301794
Main Research Area: Technical/natural sciences
Conference: 8th Nordic Workshop on Model Driven Software Engineering, Copenhagen, Denmark, 23/08/2010 - 23/08/2010
DOIs:
10.1145/1842752.1842805

General information
State: Published
Organisations: Software Engineering, Department of Informatics and Mathematical Modeling
Authors: Störrle, H. (Intern)
Pages: 18-36
Publication date: 2009

Host publication information
Title of host publication: Intl. Ws. Visual Languages and Logic (VLL’09) : CEUR Proceedings
Volume: 510
Main Research Area: Technical/natural sciences
Conference: Intl. Ws. Visual Languages and Logic, 01/01/2009
Links:
Many researchers create tools that could be very valuable to a broader audience. Using them, however, is often impeded by the required expertise and/or effort. This project aims at providing services to all of DTU Compute to help them make their tools and available to the world.

Our goal is to reach out to science and industry to promote the use of the tools and datasets we have created, and thus advance scientific progress at large and its economic dissemination. On a smaller scale, the individual stakeholders each will benefit in turn:

Compute will benefit from increased visibility and newly established contacts and created collaboration opportunities,

The sections will benefit by having more insight into the activities of other sections,

Individual researchers can benefit by more citations and higher visibility.

Department of Applied Mathematics and Computer Science

Software Engineering

Statistics and Data Analysis

Period: 01/01/2015 → 31/12/2015

Number of participants: 4

Acronym: CSG

Project participant:

Störrle, Harald (Intern)

Madsen, Jan (Intern)

Ersbøll, Bjarne Kjær (Intern)

Kristensen, Kristian (Intern)

Communication with Models

Department of Applied Mathematics and Computer Science

Software Engineering

QualiWare ApS

Period: 01/10/2014 → 30/09/2017

Number of participants: 2

Project participant:

Störrle, Harald (Intern)

Phd Student:

Davidsen, Jóhan (Intern)

Communication with models: Integrated Product and Process Views

Department of Applied Mathematics and Computer Science

Period: 01/10/2014 → 28/08/2015

Number of participants: 3

Phd Student:

Davidsen, Jóhan (Intern)

Supervisor:

Gøtze, John (Intern)

Main Supervisor:

Störrle, Harald (Intern)
**Delivering the Next Generation of Model Transformation Languages and Tools**

Department of Applied Mathematics and Computer Science  
Period: 15/01/2013 → 22/06/2016  
Number of participants: 5  
Phd Student:  
Acretoaie, Vlad (Intern)  
Main Supervisor:  
Störrle, Harald (Intern)  
Examiner:  
Kindler, Ekkart (Intern)  
Chaudron, Michel R. V. (Ekstern)  
Taentzer, Gabriele (Ekstern)

**Integrating Design Decision Management with Model-based Software Development**

Department of Informatics and Mathematical Modeling  
Period: 01/02/2008 → 01/06/2011  
Number of participants: 6  
Phd Student:  
Könemann, Patrick (Intern)  
Supervisor:  
Baumeister, Hubert (Intern)  
Main Supervisor:  
Kindler, Ekkart (Intern)  
Examiner:  
Störrle, Harald (Intern)  
Babar, Muhammad Ali (Ekstern)  
Paige, Richard F. (Ekstern)

**Financing sources**
Source: Internal funding (public)  
Name of research programme: Institut stipendie (DTU)  
Project: PhD

**Relations**
Publications:  
Model Manipulation for End-User Modelers  
Project: PhD

**Financing sources**
Source: Internal funding (public)  
Name of research programme: Institut stipendie (DTU)  
Project: PhD