How are Conceptual Models used in Industrial Software Development? A Descriptive Survey

Harald Störrle[1]

Original publication: Proc. 21st. Intl. Conf. Evaluation and Assessment in Sw. Eng. (EASE), 2017, pp. 518–534, E. Mendes, K. Petersen, S. Counsell (eds.), ACM, 2017.

There has been considerable controversy regarding the extent to which industrial software engineering benefits from conceptual modeling. Those in favor maintain that *"Model-based approaches [...] hold out the promise of significantly improving the productivity of software developers and the quality of the products they generate"*. They particularly promote using the Unified Modeling Language (UML) as the *"lingua franca of software engineering"* in the context of the Model Driven Engineering (MDE), and claim increased efficiency of software development, and better product quality. However, Mohagheghi and Dehlen famously asked *"Where is the proof?"* in their 2008 landmark paper. Even MDA-supporters had to admit that *"adoption of this approach has been surprisingly slow"*. Some scientists have outright denied that UML is used to any degree in industry [Pe13]. Where UML is used, Petre claims, it is used only informally, and *"if models end up merely as documentation, they are of limited value [...]."*.

In order to clarify this issue, we conducted a survey (see `http://tinyurl.com/MU-survey-2014` for the live survey, and `https://figshare.com/s/86d797cc3b816e972f1b` for anonymized results). We asked for specific usages of models and offered 16 predefined items with a five-point Likert scale. We also asked participants to describe further scenarios in prose, resulting in only one additional usage category ("Testing and Verification"). The results Fig. 1 indicate that the most popular usage scenarios by far (scenarios 1-3) center around communicative and cognitive processes: between 70 and 79% of all participants use models for such activities "often" or "always", while only between 4 and 8% do this "rarely" or "never". In contrast, about half the population in our survey used models "rarely" or "never" for generating code or a DSL, while only a quarter to a third of the population did this "often" or "always". Also, using models for domain- and requirements-oriented tasks (scenarios 4, 6, 9) is apparently more common than using models for technology-specific tasks (scenarios 5, 7, 10, 11). This is in line with the findings of [Pe13] who concludes that UML is mostly used as a "thought tool" and to facilitate communication with stakeholders. Usage scenario 2 ("Visualize an idea or concept") is the one that is used at least "sometimes" by most participants (96%).

[1] QAware GmbH, Aschauer Str. 32, 81549 München, Germany, HStorrle@acm.org

doi:10.18420/se2019-26

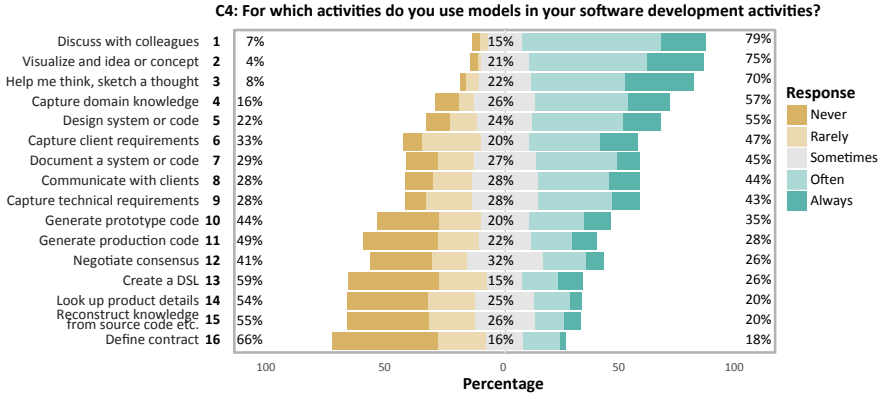

Fig. 1: Frequency of model usage scenarios sorted by decreasing frequency.

Based on our findings and prior experience, we distinguish three model usage scenarios: **Informal models** support communication and cognition, utilizing rich information implicit in the situational context; **Semi-formal models** support design and documentation activities; and **Fully formal models** are to be taken literal and binding, so as to allow the analysis of system properties, simulation, and generation of code and test cases. Observe that this grouping aligns with Fowler's distinction of models as sketches, blueprints, and programs [Fo03]. From this and other results of our study, we conclude that

- conceptual modeling languages like UML or BPMN are indeed used in industry, at least in some regions of the world.
- Models are used for a great variety of purposes by diverse stakeholders in the context software development, though Software Architects seem to benefit most from modeling.
- There are three distinct modes of modeling: for cognition and communication, for planning and documentation, and for code generation and contracts.
- Informal modeling clearly dominates in terms of frequency, followed by semi-formal modeling, but even formal modeling does occur in relevant quantities.

Our findings align with much of the previous research, but contrast with Petre's findings [Pe13]. It is remarkable, though, that the populations sampled by Petre and us have different regional distribution (Petre is unfortunately a little bit vague on this and many other methodological details of her study). This could indicate regional differences in software development practices, which are likely the result of more general cultural differences, and will possibly also impact other aspects of software development practice.

References

[Fo03] Fowler, Martin: UML Distilled. Addison-Wesley, 3rd edition, 2003.

[Pe13] Petre, Marian: UML in Practice. In: Proc. Intl. Conf. Software Engineering (ICSE). IEEE Press, pp. 722–731, 2013.