Agile Development with Domain Specific Languages

Scaling up Agile – is Domain-Specific Modeling the key?

Alan Cameron Wills[1] and Steven Kelly[2]

[1] Microsoft, http://blogs.msdn.com/alan_cameron_wills/, +44 122 347 9719
awills@microsoft.com
[2] MetaCase, http://www.metacase.com/blogs/stevek/, +358 14 4451 401
stevek@metacase.com

This workshop will investigate the application of Domain Specific Languages within Agile development. A Domain Specific Language (DSL) is designed to express the requirements and solutions of a particular business or architectural domain. SQL, GUI designers, workflow languages and regular expressions are familiar examples. In recent years, Domain-Specific Modeling has yielded spectacular productivity improvements in domains such as telephony and embedded systems. By creating graphical or textual languages specific to the needs of an individual project or product line within one company, DSM offers maximum agility. With current tools, creating a language and related tool support is fast enough to make DSM a realistic possibility for projects of all sizes.

1 Introduction

A Domain Specific Language (DSL) is designed to express the requirements and solutions of a particular business or architectural domain. SQL, GUI designers, workflow languages and regular expressions are familiar examples in 'horizontal' domains. Each allows its user to concentrate on expressing what is required in terms directly related to the domain, leaving the platform to apply the most appropriate implementation patterns – a feat made possible by its restricted scope. Problems that were very substantial projects before the advent of these languages and their implementing engines, are now the work of an afternoon.

Can individual projects or product lines within one company, make similar gains in agility and productivity by creating graphical or textual languages specific to their own domain?

Domain Specific Modeling is the creation and use of DSLs, often graphical DSLs with domain-specific generators that create full production code directly from models [1–3]. In recent years, DSM has yielded spectacular productivity improvements, particularly in vertical domains where many similar variants of a generic system are to be developed, e.g. telephony and embedded systems [4–9].

A particular concern in agile methodology is how to scale the approach to large projects [15]. DSM is a particularly effective tool to help decouple top-level requirements from implementation layers, allowing a large project to be separated into several well-decoupled smaller projects. With current tools, creating a language and re-

lated tool support is fast enough to make DSM a realistic possibility for projects of all sizes [11, 12]. By creating languages and generators specific to the needs of an individual project or product line within one company, DSM offers maximum agility [10].

This workshop will investigate the application of Domain Specific Modeling within Agile development.

2 Workshop topics

Topics to be tackled include:

- Process and Roles surrounding DSM
  - DSL users and DSL designers – are they separate?
  - Does DSM affect the development process?
  - How effective is DSM for high-level design of big projects? Or is it better at focused aspects inside a design?
- How do you design a DSL?
  - Identifying variable aspects of the domain – do you make a model or do you look at existing code?
  - Gradual evolution or big upfront design? – optimizing investment
  - Maintaining compatibility as the language evolves
  - Creating the execution platform from existing systems
- Ecology of DSLs
  - Designing a DSL from fragments of others
  - DSL repositories and markets
- Return on Investment
  - When to use DSM
  - Are DSLs syntactic sugar on a framework API?
- The development cycle
  - Testing in terms of the DSL and its abstractions
  - Debugging via a DSL
- Choosing a type of DSL
  - Graphical, textual, interactive
  - Uses of the DSL in an agile process; user roles
- Defining DSLs
  - Grammars and editing tools
  - Constraints and validation
- Generating code and artifacts
  - Generating or mapping code and other artifacts
  - DSLs used for validation and testing
  - Composing aspects of multiple DSLs

3 Audience and benefits of attending

The intended audience consists of developers and technical managers interested in finding out more about DSM. Experience of product lines, building product frameworks or creating DSLs is a bonus, but by no means necessary. Benefits of participation include a better understanding of:

- When to use DSM, and what for
- How to adjust the development process to use DSM
- How to create DSLs and use DSM

4 Submissions

Position papers are invited from each prospective participant. Each paper should:

- Have one author, and state his/her experience or interest in the area
- Explore a single topic related to the field outlined above
- Be no longer than two pages
- Clearly distinguish solid experience from wild ideas (both of which are welcome)
- Pose interesting questions for the workshop to consider
- Include web references to relevant material

Accepted position papers will be circulated to participants on the workshop website. In addition, the organizers will circulate short example DSLs and applications of them. These will be used as a basis for discussion at the workshop.

5 Workshop format and schedule

- Short presentation from workshop leaders.
  - Overview of positions and topics of interest
  - Setting vocabulary
- General goldfish-bowl discussion: “Can DSM scale up Agile?”
- “Cocktail party” of topics
  - This is like a poster session: some people put up headings on flipcharts, and then people wander between the discussions
- Separation into Working Groups based on popularity of topics
- Working Group discussions
- Reports of Working Groups
- General discussion

6 Workshop leaders

Alan Cameron Wills (Microsoft) works on DSL tools for the Visual Studio IDE. Before joining Microsoft, he was a consultant in development methods, and co-developed the Catalysis approach to software development. He has run successful workshops and tutorials in related topics at SPA2005 and OT97-04.

Steven Kelly is CTO at MetaCase, and has been the lead on the MetaEdit+ DSM tool since 1996. He is also co-founder of the DSM Forum, has served on the committee of the OOPSLA workshops on DSM since 2001 [13–14], and has been giving metamodeling and DSM tutorials around the world since 1993.

References

1. Kelly, S., Tolvanen, J-P, "Visual domain-specific modeling: Benefits and experiences of using metaCASE tools", *Proceedings of International workshop on Model Engineering, ECOOP 2000, (ed. J. Bezivin, J. Ernst), 2000.*
2. Pohjonen, R., Kelly, S., "Domain-Specific Modeling", *Dr. Dobb's Journal*, August 2002.
3. Greenfield, J., Short, S, Cook, S., Kent, S., *Software Factories: Assembling Applications with Patterns, Models, Frameworks, and Tools*, Wiley, 2004.
4. Kieburtz, R. et al., "A Software Engineering Experiment in Software Component Generation," *Proceedings of 18th International Conference on Software Engineering*, Berlin, IEEE Computer Society Press, March, 1996.
5. Long, E., Misra, A., and Sztipanovits, J., "Increasing Productivity at Saturn," *IEEE Computer*, August 1998, pp. 35-43.
6. Sztipanovits, J., Karsai, G., and Bapty, T., "Self-Adaptive Software for Signal Processing," *Communications of the ACM*, May 1998, pp. 66-73.
7. MetaCase, MetaEdit+ Revolutionized the Way Nokia Develops Mobile Phones, http://www.metacase.com/papers/MetaEdit_in_Nokia.pdf, 1999.
8. Weiss, D., Lai, C. T. R., *Software Product-line Engineering*, Addison Wesley Longman, 1999.
9. Moore, M., Monemi, S., Wang, J., Marble, J., and Jones, S., "Diagnostics and Integration in Electrical Utilities," *IEEE Rural Electric Power Conference*, Orlando, FL, May 2000.
10. Cook, S., "Domain-Specific Modeling and Model Driven Architecture", MDA Journal, http://www.bptrends.com, January 2004
11. Nordstrom, G., Sztipanovits, J., Karsai, G., and Ledeczi, A., "Metamodeling - Rapid Design and Evolution of Domain-Specific Modeling Environments," *IEEE ECBS Conference*, April 1999.
12. Kelly, S., "Tools for Domain-Specific Modeling", *Dr. Dobb's Journal*, September 2004.
13. Tolvanen, J-P., Kelly, S. Gray, J., Lyytinen, K., (eds.) Proceedings of OOPSLA workshop on Domain-Specific Visual Languages, Tampa Bay, Florida, USA, University of Jyväskylä, Technical Reports, TR-26, Finland, 2001.
14. Gray, J., Rossi, M., Tolvanen, J-P., (eds.) Domain-Specific Modeling with Visual Languages, Special issue of Journal of Visual Languages and Computing, Vol. 15 (3-4), Elsevier, Jun-Aug, 2004.
15. Eckstein, J. *Agile Software Development in the Large*, Dorset House, 2004.
16. Czarnecki, K., Eisenecker, U. *Generative Programming*, Addison Wesley, 2000.