Keep it Sweet & Simple

The Third KISS Workshop
25 October 2009 @ OOPSLA
Orlando FL, USA
Status Update

- www.industrialized-software.org/kiss-initiative
  - Reaching a strong consensus on fundamental values and principles for designing and using DSLs
  - Progress towards interoperability between tools

- KISS conference events to date
  - 14 April, Australian SW Engineering 09, Gold Coast, Australia
  - 16 June, Code Generation 09, Cambridge, UK
  - 25 October, OOPSLA 09, Orlando, Florida

- Upcoming KISS conference events
  - 16 or 17 November, Automated Software Engineering 09, Auckland, NZ

- For a powerful message on simplicity, visit www.spinellis.gr/blog/20090203
Building software intensive systems has taught us that domain specific languages represent the key to capturing, preserving, and exploiting knowledge in virtually all industries.

When designing and using domain specific languages we keep it sweet & simple (KISS). Most importantly, we are committed to the following values:

- We strive to **automate** software construction from domain models
- We work with **domain-specific assets**
- We support the emergence of **supply chains** for software services
- We see **Open standards, as well as Open Source** as driving interoperability
- We use methodologies that conform with the values of the **Agile Manifesto**

*Peter Bell, Jorn Bettin, Tony Clark, Keith Duddy, Scott Finnie, Matthew Fowler, Steven Kelly, Jack Kennedy, Frank Sauer, Laurence Tratt, Markus Voelter, Jos Warmer*
DSL User Perspective

1. The DSL must be meaningful to users
2. The DSL should be cognitively efficient
3. The DSL should have multiple notations where necessary
4. DSLs should offer mechanisms for modularizing and integrating models
5. The DSL should be supported by appropriate tooling

Goals

1. Optimize support for cognitive domain models
2. Minimize the number of errors in system specification artifacts
3. Simplify the system specification process
4. Disambiguate communication between domain experts
DSL Owner Perspective

1. There **must** be an economic imperative for the development of a DSL

Goals

1. Empower all stakeholders in system development
2. Improve quality of systems
3. Meet regulative/legal compliance requirements
4. Minimize the resources required to develop systems
5. Minimize the costs of exchanging solution technologies
6. Decouple systems specifications from particular solution technologies
7. Facilitate the migration of legacy systems and legacy data
DSL Developer Perspective

1. The DSL **must** not be polluted with implementation features
2. Model processing **must** always be based on a formal DSL definition
3. DSLs **should** be kept small through modularization and integration

Goals

1. Enable a modular and composition-based approach to formal DSL definition
2. Ensure that DSL syntaxes meet user needs
3. Provide formal DSL definitions in a form amenable for processing in multiple tools
4. Integrated support for defining DSL syntax; semantics; tooling; and serialization formats
Guidelines for DSL Developers

1. DSL development **must** assemble sufficient expertise in:
   - ★ Problem Domain
   - ★ Solution Domain
   - ★ Designing DSLs
   - ★ Language Implementation Technologies

2. DSL development **must** anticipate evolution of the DSL

3. A DSL based approach **must** provide adequate means for:
   - ★ Management and maintenance of artifacts
   - ★ Incremental transition from a traditional code based approach
   - ★ A well defined development, deployment and support process
Recommendations for DSL Developers

1. DSL modularization should take into account DSL user roles
2. DSL design and validation should use an iterative approach
3. DSL design should involve the construction of concrete examples
4. DSL notations should take into account established domain notations
5. A DSL based approach should provide adequate support for distributed multi-user projects
More examples

to underscore

• Fundamental Principles

• Guidelines

Counter-examples

that illustrate the limits of

• Fundamental Principles

• Guidelines
Publish the Foundations for DSSD [Domain Specific Software Development]

1. Confirm signatories for the official wording
2. Publish content on home page of industrialized-software.org
3. Promote KISS on signatory company web sites, refer to KISS principles and guidelines in our daily work with customers, and include appropriate references to industrialized-software.org in articles and case studies

Identification of open challenges

1. Positioning of domain analysis in relation to traditional requirements engineering
2. Avoiding “my model is right and yours is wrong”
3. Avoiding the design by committee effect
4. Concrete steps towards model based interoperability
Keep it sweet and simple

Thank you!

Jorn Bettin  
jbe @ sofismo.ch

Tony Clark  
tony.clark @ tvu.ac.uk

William Cook  
wcook @ cs.utexas.edu