Standards Initiatives for Software Product Line Engineering and Management within the International Organization for Standardization

Timo Käkölä
University of Jyväskylä
Finland
What is software product line engineering?

- Software product line engineering is a methodology for developing software-intensive systems and services faster, at lower costs, and with better quality and higher end-user satisfaction than is possible through the engineering of single systems.

- It needs two distinct development processes: domain engineering and application engineering.
  - Domain engineering defines and realizes the commonality and variability of the software product line and introduces variability in all software product line assets (e.g., domain requirements and domain test cases), thus establishing the common software platform for rapidly mass-customizing high-quality applications to the needs of different customers and markets.
  - Application engineering derives specific applications by strategically reusing the platform and by exploiting the variability built into the platform.
Challenges involved: The need for high levels of abstraction

- The platforms require substantial investments, have long life cycles, and have to provide product line architectures and features generally applicable to a wide range of products, services, and markets.
- Industrially validated modeling methods and commercially available modeling tools are critical to deal with the abstractions.
- In addition to traditional system modeling, variability modeling is required to document explicitly how the applications within the product line can vary.
Challenges involved: The need to cope with high levels of complexity

- *Coordinated* changes are needed in development methodologies, processes, tools, product architectures, organizational designs, business models, and capability levels of the stakeholders involved.
- Domain engineering typically requires development methodologies far different from application engineering => It may be impossible to choose a development methodology for organization-wide use.
- Domain engineering and application engineering are often carried out in separate organizational units due to their different concerns.
- The two engineering life cycles and enabling knowledge management systems need to be holistically designed because the cycles are knowledge intensive and require the management of complex dependencies between many types of artifacts.
- The business models need to be appropriate and clear from both external and internal viewpoints.
  - Internally, the business model must facilitate the appropriate financing and resourcing of domain engineering.
Challenges involved: The fragmented software product line body of knowledge

- An internationally accepted academic educational curriculum and readily available teaching materials are missing
- Few methods and tools presented in scientific papers are so widely used in the industry that they could be considered as de-facto standards for software product line engineering
- Industrial validation of most methods and tools is in early phases
- International standards do not cover product line engineering
- Individual academic papers cannot have both the holistic scope and the details required to help businesses take coordinated product line adoption actions
- There is only one book about software product line engineering for students in undergraduate and graduate level university courses
- => it is challenging for educators to teach and for students and practitioners to understand and apply software product line concepts
Challenges involved: The lack of tools

- Interoperable information systems are critical to support knowledge management throughout the domain and application engineering life-cycles and during variability modeling and resolution.
- Commercially available and industrially validated software tools to implement the systems are scarcely available partly because the markets do not know what to expect from such tools.
- There is no internationally standardized variability metamodel to capture variation points, that is, the items that vary; variants defining how the variable items can vary; and constraints between variants, between variants and variation points, and between variation points.
- Tool vendors have circumvented the problem by
  - focusing on features that support the engineering of single systems well
  - enabling extension mechanisms that let organizations tailor their own methods and tools for purposes such as variability modeling.
- Organization-specific extensions are inadequate when organizations work in global ecosystems and need to share models, methods, and tools.
How to deal with the challenges?

- The field must focus on consolidating its body of knowledge
- The next level of development requires coordinated actions beyond the research community
- Markets have a key role in consolidation efforts by determining which research deliverables and other stocks of knowledge produce most value in practice
- International standardization efforts are needed to determine those parts of the body of knowledge that
  - are stable and coherent enough for standardization purposes
  - can serve as the baseline on which new breakthroughs can be built
Background of software product line standardization efforts

- ISO/JTC1/SC7 decided to initiate requirements engineering standardization activities in its plenary meeting in Helsinki in May 2005
- I was involved with the largest European software engineering research project series (ESAPS, CAFÉ, and FAMILIES) ever conducted, focusing on software product line research with a budget of more than 100 million euros
- The project series had substantially advanced the tools and methods related to software product line requirements engineering
- Existing standards focused on requirements engineering of single systems
- Dan Lee and I proposed to SC7 that a standardization project for software product line requirements engineering should be initiated
- The ISO/IEC 29118 project was started in 2006 to deal with the issue
- During the next two years, Dan Lee and I created several drafts of 29118, describing a generic reference model and detailing software product line requirements engineering methods and tools
The status of software product line standardization efforts

- SC 7 has decided to create a set of interrelated software product line engineering standards
- The contents of the 29118 project have been divided into two new projects:
  - ISO/IEC 26520 will establish a reference model for software product line engineering, covering phases of the domain and application engineering life-cycles on a high level of abstraction
  - ISO/IEC 26521 will detail the domain and application requirements engineering processes
- During the next decade, three new projects will create three standards for the other life-cycle phases:
  - Product line architecting
  - Product line realization, and
  - Product line testing
- There will also be two new standards for technical management (ISO 26525) and organizational management practices
The preliminary reference model of the ISO/IEC 26520 for software product line engineering and management
The preliminary reference model of the ISO/IEC 26521 for software product line requirements engineering and management

**Domain Requirements Engineering**
- **Domain Requirements Elicitation** (Initial set of common and variable stakeholder needs)
- **Domain Requirements Analysis** (A set of common and variable requirements)
- **Domain Requirements Specification** (Documented set of common and variable requirements)
- **Domain Requirements Verification and Validation** (Clear, complete, and correct requirements)

**Application Requirements Engineering**
- **Application Requirements Elicitation** (Initial set of stakeholder needs for an application)
- **Application Requirements Analysis** (A list of differences between domain and application requirements)
- **Application Requirements Specification** (Documented set of application requirements)
- **Application Requirements Verification and Validation** (Clear, complete, and correct application requirements)

**Core Asset Management in Requirements**
- **Core Asset Repository Implementation**
- **Core Asset Identification**
- **Core Asset Configuration and Structure Definition**
- **Core Asset Annotation**
- **Core Asset Validation**
- **Core Asset Evolution**

**Application Requirements Management** (Traceable and managed baseline)

**Variability in Requirements**
- **Variability Modeling**
- **Traceability Management between a Variability Model and Domain Artifacts**
- **Variability Annotation**
- **Variability Validation**
- **Variability Evolution**

**Product Line Scoping**
- **Product Portfolio Scoping**
- **Domain Scoping**
- **Asset Scoping**
The preliminary reference model of the ISO/IEC 26525 for the technical management of software product line engineering

<table>
<thead>
<tr>
<th>Process Management</th>
<th>Variability Management</th>
</tr>
</thead>
<tbody>
<tr>
<td>Process-Enabling</td>
<td>Variability Modeling</td>
</tr>
<tr>
<td>Domain Process Definition</td>
<td>Variability Documentation</td>
</tr>
<tr>
<td>Application Process Definition</td>
<td>Variability Binding</td>
</tr>
<tr>
<td>Process Monitoring and Control</td>
<td>Variability Traceability</td>
</tr>
<tr>
<td>Process Improvement</td>
<td>Variability Control &amp; Evolution</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Support Management</th>
</tr>
</thead>
<tbody>
<tr>
<td>Technical Planning</td>
</tr>
<tr>
<td>Quality Assurance</td>
</tr>
<tr>
<td>Configuration Management</td>
</tr>
<tr>
<td>Technical Risk Management</td>
</tr>
</tbody>
</table>
The plan for the next three years

- All three standards are new and none of them have been balloted
- They will be further developed by the small editorial teams
- Timo Käkölä is an editor in all the projects
- Development will be facilitated by national teams at least in Finland, India, Japan, South Korea, and the USA
  - National teams are especially needed to collect data about the applicability, validity, and usefulness of the proposed life-cycles, phases, practices, methods, and tools
- 26520 and 26521 should be ready for first balloting in 2011 and 26525 in 2012
- The new organizational management standard will be planned but most resources will be used to develop and ballot the three standards
Benefits from applying the software product line standards

- The standards will detail the domain and application engineering life-cycles, all phases of the life-cycles, and the core asset management, organizational management, and technical management practices
- Users of the standards can
  - specify, verify, and validate engineering and management practices for both existing and envisioned product lines based on the good product line practices described in the standards
  - holistically understand, adopt, and enact the domain and application engineering life-cycles
  - evaluate and select relevant methods and tools for product line practices based on business and user-related criteria
- In addition, tool vendors can use the standards to
  - develop interoperable tool suites and features supporting domain and application engineering life-cycles and their phases
  - communicate about their tools to the markets