Introduction to Software Product Lines: Engineering, Service, and Management Minitrack

Timo Käkölä
University of Jyväskylä, Finland
timokk@jyu.fi

Andrea leitner
Graz University of Technology, Austria
andrea.leitner@tugraz.at

Software has become the key asset for competitive products and services in all industries. Thus, competitiveness in software development, maintenance, and related services has become a concern for organizations. Competitiveness can be increased through (1) internal strategies such as the strategic creation and reuse of software assets and (2) external strategies such as outsourcing software development, maintenance, and/or services from third party service providers and acquiring off-the-shelf components from providers and open source communities. A viable third strategy is to enact both strategies in parallel. This minitrack focuses on the first and third strategy.

Software product line engineering (SPL) is an industrially validated methodology for developing software-intensive systems and services faster, at lower costs, and with better quality and higher end-user satisfaction. It differs from single system development:

1. It needs two development processes to work optimally: domain engineering and application engineering. Domain engineering defines and realizes the commonality and variability of the product line by establishing a common software platform. Application engineering derives applications by exploiting the commonality and binding variability built into the platform.
2. It needs to explicitly define and manage variability. During domain engineering, variability is introduced in all assets such as requirements, architectural models, components, and test cases. It is exploited during application engineering to mass-customize applications to the needs of customers.

Software product line research has mostly focused on the modeling and management of variability in the context of embedded systems (e.g., cellular phones). Most software product line experiences have been obtained from large government and private organizations. This minitrack welcomes contributions from the mainstream product line research. It also acknowledges that the extant body of knowledge in the field is fragmented. More holistic and integrative research approaches are needed to help practitioners leverage the research results in establishing and improving software product lines. Indeed, experienced practitioners have sometimes established innovative product lines and enabling practices and systems with limited awareness of the software product line body of knowledge.

This minitrack accepted five papers this year. One of them appears in another minitrack and another paper was cancelled. The minitrack consists of three papers.

Groher and Weinreich note that variability management and architecture design and implementation are mostly separate activities in the extant literature. Existing variability management approaches support architecture design only to the extent it is necessary for product derivation. Existing architecture design and implementation tools lack support for variability tracing and modeling. Groher and Weinreich present an approach for integrated variability management during software architecture design and implementation.

The paper of Hou, Makarov, Samaan, and Etingov focuses on a standardized forecast error analysis and prediction tool that can be implemented into a software package that predicts the uncertainty range for generation resources involved in the power grid balancing service. The tool has deliberately been designed in order to be reusable in various current and future applications. Different sources of uncertainty and variability are to be supported, while providing an efficient and accurate prediction tool.

Fant, Gomaa, and Pettit describe a pattern-based approach to specify product line variability at a high level of granularity through architectural design patterns. This approach requires less modeling and design during domain engineering but necessitates additional modeling and design during application engineering. It is argued to work well in situations where all the variability is not known during domain engineering or when each application needs unique variant components. In such situations application engineering plays a significant role anyway.

1530-1605/12 $26.00 © 2012 IEEE
DOI 10.1109/HICSS.2013.327