1. WORKSHOP DESCRIPTION

The objective of this workshop is to bring together researchers from the software product line community and software architecture community to identify critical challenges and progress the state-of-the-art on variability in software product line architectures.

1.1 Motivation

A software product line is a collection of similar software systems that are constructed from a shared set of assets in a prescribed way. Software product lines are valued by industry as they increase productivity and enable strategic, planned reuse of assets among multiple products.

The product line architecture is key to the success of a software product line. In contrast to single system architectures, a product line architecture is designed to underpin multiple systems. A product line architecture reifies the commonalities between the various products and also clearly delineates the variability that is allowed between products. As such a product line architecture is paramount to predictably achieve the qualities of the various products in a software product line.
relation between variability and other qualities such as performance and scalability is a challenge that requires further investigation.

Discussion topics for VARI-ARCH include:

- Modeling variability using architectural views/viewpoints
- Mechanisms/tactics/patterns/styles for variability in product line architectures
- Evaluating variability in product line architectures
- Resolving architectural variability during product derivation
- Traceability of variability in product line architectures
- Conformance of products to the product line architecture
- Practical and industrial examples of variability in product line architectures
- Relation between variability and (other) quality attributes

The workshop will have a highly interactive program with focused presentations and break out sessions for discussion. The workshop will start with an invited talk by Tomi Männistö.

2. ACCEPTED PAPERS

The following papers were peer reviewed and accepted to the workshop, listed here with their abstract in alphabetical order according to the first author’s name.

Abbas, N., Andersson, J., Löwe, W., "Autonomic Software Product Lines"

This paper describes work in progress and proposes a novel variability mechanism that self-optimizes product-line instances. This will make the products resilient with respect to context changes and some aspects of product-line evolution. The variability mechanism is composed of a context guided dispatch and on-line training process. Together they form a simple, yet powerful variability mechanism that continuously learn through experimentation which variants to bind to a variation point given context information and optimization goals.

Galster, M., "Describing Variability in Service-oriented Software Product Lines"

Service-oriented architectures have gained popularity over the past few years and became a widely used concept in the software industry. Software product lines on the other hand allow one generic architecture to be configured and deployed in different clients. In this paper, we investigate what types of variability exist in service-oriented software architectures and suggest a way for representing variability. This is necessary in order to enrich service-oriented architectures with product line capabilities. To describe types of variability in service-oriented architectures, we propose formalizing the notion of variability. To allow different viewpoints on variability, we define a set of stakeholder roles that occurs in the context of service-oriented software architectures. By applying the proposed concepts, we hope to improve variability management on the software architecture level of service-oriented systems.

Galvao, I., van den Broek, P., Aksit, M., "A Model for Variability Design Rationale in SPL".

The management of variability in software product lines goes beyond the definition of variations, traceability and configurations. It involves a lot of assumptions about the variability and related models, which are made by the stakeholders all over the product line but almost never handled explicitly. In order to better manage the design with variability, we must consider the rationale behind its specification. In this paper, we present a model for the specification of variability design rationale and its application to the modelling of architectural variability in software product lines.

Geertsema, B., Jansen, S., "Increasing Software Product Reusability and Variability using Active Components: a Software Product Line Infrastructure"

Software Product Lines are typically used to support development of a software product family and not a software product population, which denotes a broader and more diverse range of software products. We present a Software Product Line Infrastructure (SPLI) that has been designed to increase reuse of software efforts by widening the software product line scope and supporting the reuse of application design via step-wise refinements. The SPLI takes a bottom-up approach by structuring product features in highly reusable software components called Active Components. A model-driven engineering approach is used in which application design is specified using domain-specific models and variability models. Variability within Active Components is bound during product derivation by executing model-to-artefact transformations. Components are active because they can control the derivation process. The SPLI enables step-wise refinements of application design by allowing model specialization and by merging models during product derivation. A prototype implementation of the SPLI has been developed that was used to develop and evaluate an experimental software product line. Furthermore, a questionnaire has been executed among five experts for further evaluation. The results are positive towards the feasibility and utility of the SPLI. It is concluded that within the context of our experimental software product line, the SPLI supports the development of diverse software products and enables step-wise refinements of application design.

Hilliard, R., "On representing variation"

Although primarily studied in the context of product lines, variability is a key fact of most systems and therefore of their architectures. Thus it is essential for the Architect to have suitable tools for representing, managing and reasoning about variation. This paper poses a simplified model of variation and then explores the consequences of that model for the representation of variation as a part of architecture description, using the conceptual foundation of ISO/IEC 42010 (the revision of IEEE 1471:2000).

Simidchieva, B., Osterweil, L., "Categorizing and Modeling Variation in Families of Systems: a Position Paper"

This paper presents an approach that considers variation in systems and system architectures according to the kind of relation among the variants in the software family. The approach highlights why it is beneficial to consider such dif-
ferent variation relations separately and gives examples of what these relations may be.

Two main categories of variation relations are presented, based on whether the system architecture remains constant (architecture-based variation), or whether the architecture itself is variable, i.e. the variants do not share a common architecture. The paper introduces several different kinds of variation families that seem to belong to these two categories, as well as yet other families comprising variants that do not neatly fit in either category, with only a subset of the variants sharing a common architecture. Each kind of variation relation is illustrated with an example software family from different domains, including operating systems (OS).

3. ORGANIZATION

Organizing Committee
- Alexander Helleboogh, DistriNet Labs, Katholieke Universiteit Leuven, Belgium
- Paris Avgeriou, University of Groningen, The Netherlands
- Nelis Boucke, DistriNet Labs, Katholieke Universiteit Leuven, Belgium
- Patrick Heymans, University of Namur, Belgium

Program Committee
- Pierre America, Philips Research, The Netherlands
- Sven Apel, University of Passau, Germany
- Felix Bachmann, Software Engineering Institute, USA
- David Benavides, University of Seville, Spain
- Jan Bosh, Intuit, USA
- Laurence Duchien, INRIA, France
- Rich Hilliard, USA
- Philippe Kruchten, University of British Columbia, Canada
- Patricia Lago, Department of Computer Science, VU University Amsterdam, the Netherlands
- Klaus Pohl, University of Duisburg, Essen, Germany
- Klaus Schmid, University of Hildesheim, Germany
- Michael Stal, Siemens AG, Germany
- Eddy Truyen, DistriNet Labs, Katholieke Universiteit Leuven, Belgium
- Andrzej Wasowski, IT University of Copenhagen, Denmark
- Uwe Zdun, Vienna University of Technology, Austria