Program for after PC meeting event:

Ralf Laemmel
Title: Reflections on modular semantics

Abstract: In Semantics and type systems, a modest amount of generic concepts has been identified. A good showcase is action semantics --- the modularity of which allows us to compose somewhat arbitrary language definitions. Similarly, in the program analysis community, generic frameworks have been presented that allow for an instantiation for a range of languages. Again similarly, in the program transformation community, specifically in the context of refactoring and other re-engineering transformations, such language parameterisation has been identified as an issue. Finally, in the AOSD field, related attempts are underway to generalise the notion of join-point models so that one would obtain such models in a very systematic manner from a language definition at hand. This presentation argues that all these genericity efforts can be integrated by taking a certain semantics-biased point of view.

Shriram Krishnamurthy
Title: Verifying Product-Line Systems

Feature-oriented programming organizes programs around features rather than objects, thus supporting extensible, product-line architectures. Programming languages increasingly support this style of programming, but programmers get little support from verification tools.

Ideally, programmers should be able to verify features independently of each other and use automated compositional reasoning techniques to infer properties of a system from properties of its features. Unfortunately, most modular model checking techniques do not support feature-oriented modules; they betray their hardware roots by assuming that modules compose in parallel. In contrast, feature-oriented modules compose sequentially in the simplest case; most interesting feature-oriented designs are really quasi-sequential compositions of parallel compositions. These designs therefore demand and inspire new verification techniques.

This talk gives an overview of feature-oriented modules, our compositional model checking methodology for them, and its application to real software systems.