

Outline of Research Plans

Harold Ossher

IBM Thomas J. Watson Research Center

My current research is on the *Concern Manipulation Environment* (CME). It is an integrated environment for *aspect-oriented software development* (AOSD), built on Eclipse and available as an Eclipse open-source project. It is intended to support both software developers who want to use AOSD, and AOSD tool-providers and researchers.

A *concern* is any area of interest in a software system, such as a component, a feature, a variant, a quality of services issue or a systemic concern (such as logging). Some concerns align with the modular structure of the software, whereas others cut across it. For example, a feature might consist of a number of individual methods and fields in a number of classes in an object-oriented system. Concerns occur in different artifacts and lifecycle phases, and many concerns span artifacts and phases. For example, a concern such as “the call-waiting feature” might be identified in a requirement, but then will impact architecture, design, code and testing, and some portions of the software in the artifacts produced during these phases will address it. Support for crosscutting concerns is the hallmark of AOSD. The CME aims to support uniform manipulation of all kinds of concerns, whether crosscutting or not and whatever artifacts are involved.

There are several key software engineering activities supported by the CME, including:

- Identifying concerns, whether early on during requirements analysis, as the system evolves, or in retrospect in an existing system.
- Modeling and visualizing the concerns, and allowing developers to use concerns to navigate and manipulate the software.
- Extracting the software that pertains to specific concerns into separate modules.
- Composing concerns, thereby combining their capabilities. Since concerns include features, variants and the like, such composition is suitable for configuring product lines.

Extraction and composition involve interesting forms of Program Generation. A concern might consist of fragments of software modules; extraction must produce a new set of modules containing just the required fragments, with hanging references dealt with in an acceptable manner. Composition is more flexible than traditional component composition, in that it involves identifying *join points* (such as method definitions or calls) at which the concerns are to be combined, and then combining the join points appropriately in the generated composed software (such as generating compound methods or calls to multiple methods from a single call site). There are many possibilities, and details of composition is an active research area.

Another relationship to Program Generation is that whenever software is generated from some sort of specification, there are concerns that span the specification and the generated software. Modeling these concerns and their relationships is related to traceability, and can be a significant help in working with such software.

Current work on the CME is focused on delivering an initial set of components and tools. The initial tools support code artifacts, but much of the underlying system is artifact-independent, and we are eager to move on to others. We hope to engage the community in expanding the set of tools and enhancing the underlying components along with us.