Models to Code with no Mysterious Gaps

Name Leon Starr

Abstract text

There are many benefits to a model-oriented approach to software engineering. Complex problems can be simplified and better understood. The developer can work at a higher level of abstraction and thus do more with fewer building blocks. Implementation and platform-specific details can be factored out of fundamental application logic. Concepts, rules and policies can be made unambiguous and precise without binding them to particular design decisions. Communication and documentation of these concepts can be improved.
But the path from models to code is not always obvious. Model developers must fully understand the kinds and level of detail that must be addressed in the models vs. what must be addressed during implementation. The specific optimization capabilities in the code generation process must be absolutely clear. Most importantly, model-development teams and management must have confidence that there is a trustworthy process that will yield production code.

This is not a talk about theory, magic and wishful thinking. We will explore an existing open source tool chain that leads from complete Executable UML models to efficient C on an embedded microprocessor. The goal is not to extoll the benefits of using this or any particular tool chain. But specifics are key to illustrating the essential steps, components and processes required to transform executable models into efficient production code. Hopefully this example will inspire or assist the development of code generation systems targeting other languages and platforms.