Effective Reduction of Avoidable Complexity in Embedded Systems

Complexity Management in Software Models

Safety-Critical Systems are becoming extremely software-reliant. Software complexity can increase the total acquisition costs as much as 16%. The ERACES project aims to identify and remove complexity in software models, such as SCADE.

Why detect complexity in models?

Safety-critical systems development is shifting from traditional programming (e.g., Ada, C, assembly) to modeling languages (e.g., Simulink, SCADE). Model-Based Engineering (MBE) provides an accurate semantics for system analysis, validation, and automatic code production —reducing development and testing efforts. Parts of the Airbus A380 and A400M planes have been designed using models. Current costs savings estimates show that using models can help save as much as 57% on the development of an avionics system at the highest criticality level (DAL A).

Why complexity in models matters?

Software complexity increases not only development but the overall acquisition costs. As maintenance activity accounts for 70% of the lifecycle costs, reducing complexity of models is of primary importance. As MBE is a new development paradigm, we need new methods and metrics to identify complexity.

What has been done by the SEI?

During the ERACES project, the SEI team focused on these areas:

- Develop complexity metrics in models
- Understand the use of modeling tools
- Estimate the costs of software complexity

Complexity Metrics in Behavior Models

The SEI has been working on applying existing complexity metrics in software models. We selected complexity metrics that have a different focus from:

McCabe: focus on state space

Halstead: focus on operators and operands

Zage: focus on components connections

These metrics have been tested and implemented within the SCADE tool.

The SEI worked on new, model-specific complexity metrics that rely on the specific data-flow semantics of SCADE. This new complexity metrics reports for each flow the related number of operators, operands and outputs. Model designers use this information to update their design and reduce the system complexity with different strategies (e.g., refactoring components, change connections, change interfaces definition). By reducing the number of connections, designers decrease the number of tests (c.f. DO-178C) required to certify the software. These metrics have been implemented in the SCADE tool.

All ERACES plugins and tools are available on the SEI github forge under the BSD license.

Software Architecture Complexity

We identified software architecture patterns that incur unnecessary complexity. Software configuration and deployment policy (e.g., execution rate, communication queues dimensions) impacts system behavior and might have a significant impact (e.g., early/late values, missing values). We developed a method that identified inappropriate software architecture patterns using AADL that might incur complexity and suggest workaround and fixes. Our approach has been tested on a customer project and successfully detected an error related to missing values.

Understanding Complexity

The SEI started an experiment to understand the current vision of complexity in software models by practitioners. We also asked professionals and students to design a model from textual specifications. We found that many users, even experienced ones, have issues using models. When transitioning to a full MBE approach, training becomes the key to success.

Making an impact

The SEI initiated a collaboration with ANSYS, the developer of SCADE, to use the SEI complexity metrics tools and understand the impact of complexity in software models. ANSYS is currently working on integrating these metrics into their products to ultimately help system designers detect potential design issues and improve their models. The SEI has been invited to present the ERACES research project results at the SCADE User Conference to be held in Paris on October 2015.

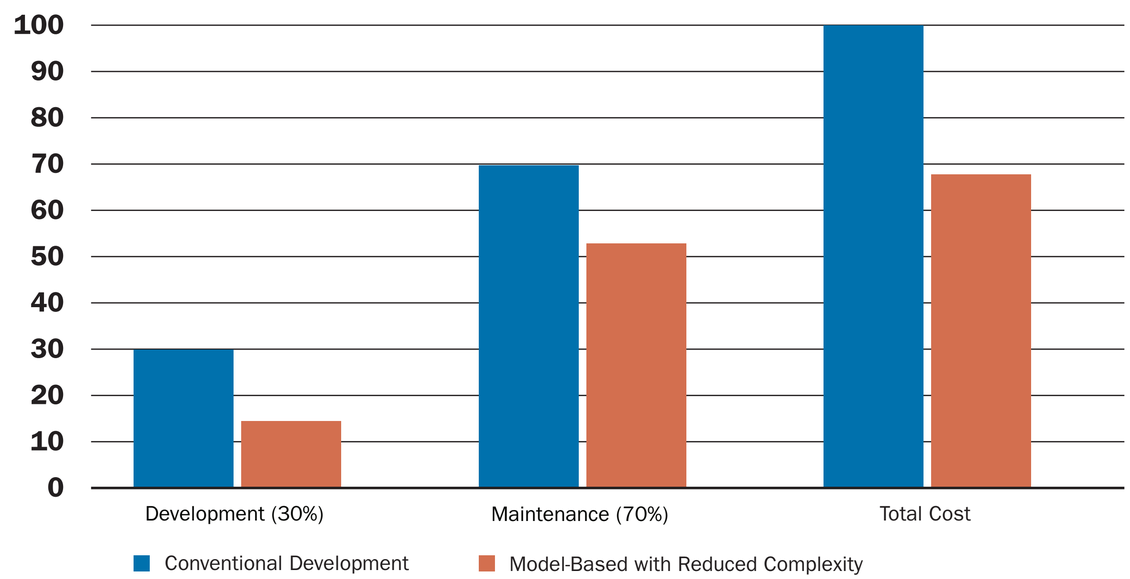
Costs of complexity

The SEI evaluates the cost of complexity when using models. MBE development approaches reduce development and testing efforts, especially for critical systems that require intense testing efforts. Our estimates shows a reduction of cost development of 50% for safety-critical systems when using a MBE development approach that includes a certified code generator.

Reducing software complexity can reduce acquisition costs by more than 30%.

However, inappropriate design decisions can put these savings at risk and spread throughout the software lifecycle. Complexity can increase maintenance costs by 25%. As maintenance costs of safety-critical systems accounts for more than 70% of the total acquisition costs, complexity by itself can increase them by more than 30%.

Model-Based Engineering: Costs Savings Estimates



Model Complexity has a large impact on maintenance activities, which represents at least 70% of the TCO

Contact: Julien Delange jdelange@sei.cmu.edu

This material is based upon work funded and supported by the Department of Defense under Contract No. FA8721-05-C-0003 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.

DM-0002945