Software Engineering Institute | Carnegie Mellon PCS

Engineering Performance-Critical Systems:

meeting and exceeding performance, dependability, and interoperability goals

Does your process ensure adherence to specified dependability requirements?

Can you systematically catch errors through all stages of development?

Can you identify problems early enough to make necessary cost, schedule and functionality decisions?

Solutions...

Software engineers must frequently design systems to perform successfully under adverse circumstances, as part of complex, networked systems. The need for performance under heavy load or in the presence of subsystem failures is often critical.

The Performance Critical Systems (PCS) team at the Software Engineering Institute is dedicated to helping with these challenges. We can help you to implement available practices so that your organization can realize significant benefits:

- Potential performance and dependability problems are identified early enough to make acceptable cost, schedule, and functionality decisions.
- Performance and dependability anomalies are found infrequently during system integration and even less frequently in operational use. Any problems that do arise are readily corrected.
- System reliability and availability is increased through the use of more effective design. In particular, the impact of various dependability strategies on system performance can be modeled at a high level of system design.
- Delivered systems are more easily adjusted to respond to changes in system load and evolving requirements and technology.
- System tests reliably predict the behavior of heavily loaded systems.

...through Better Assurance Practices

When software-intensive systems experience significant performance problems, these problems usually arise because existing and effective state-of-the-art practices are not available or routinely used.

To address that shortcoming, we are investigating techniques for showing how software dependability claims can be supported by evidence derived from a combination of analysis and testing. The purpose of these techniques is to provide **analysis-based assurance**—the confidence, derived from means other than solely by testing, that a system or component of a system will perform as expected.

Analysis-based assurance can augment testing where thorough testing would be infeasible or too costly or reduce the number of tests that would be needed to assure the desired system dependability.

We develop tools and methods to document and predict the dependability of a system. We are currently focused on creating and documenting structured rationales (assurance cases) that show how evidence gathered during system design and test supports dependability and real-time performance claims for specific systems.

An assurance case is a structured argument showing how claims of dependability are being met for a particular system. We are developing case studies to examine this notation as an effective method for documenting why engineers should have confidence in the dependability of a system.

The concept of an assurance case has been derived from the safety case, a construct that has been used successfully for more than a decade to show that safety-critical systems meet their safety properties in areas such as flight control, railroad signaling, and nuclear reactor shutdown systems.

Engineering Performance-Critical Systems:

meeting and exceeding performance, dependability, and interoperability goals

...and Model Based Real-Time System Design and Analysis

To help engineers meet performance-critical requirements, PCS is addressing the challenges of assuring the behavior of real-time and embedded systems.

PCS has established a **model-based engineering practice** for embedded and real-time systems development. This work has applications in avionics, aerospace, automotive, and autonomous robotics, among other areas. We have developed case studies, tutorials, and use guides showing how to couple analysis tools and methods with the specification of embedded system designs.

The system specification method we are using is the Architecture Analysis and Design Language (AADL). This SAE standard has attracted wide interest from the automotive and aerospace communities.

An organization using the AADL standard can lower system development and maintenance costs. The AADL standard provides

- a precise syntax and semantics, so that documentation can be well defined
- the ability to model multicontractor architectures in a single architectural model that can be incrementally defined
- the capturing of an "architectural API" for evaluating the effect of change—such as the emergent effects of integration (e.g., safety, schedulability, end-to-end latency, and security)
- early and life-cycle tracking of modeling and analysis
- analysis of the system structure and runtime, rather than functional, behavior
- a great complement to reference architectures and component-based or product-line development

How PCS Can Help Your Organization

- a public two-day course on model-based engineering
- tailored tutorials on model-based engineering and the assurance case method that can be delivered at your organization's location
- consultation on how to
  - improve confidence in software and system architecture
  - uncover issues prior to system integration
  - plan for developing assurance cases
- assistance by
  - identifying solutions and validating them with AADL and associated tools
  - validating your system's dependability attributes
  - documenting assurance patterns used in your organization
- analysis-based assurance methods to provide confidence that a component or system will perform as expected

For Further Reading

Feiler, P.; Gluch, D.; & Hudak, J. *The Architecture Analysis & Design Language (AADL): An Introduction* (CMU/SEI-2006-TN-011). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2006. http://www.sei.cmu.edu/publications/documents/06.reports/06tn011.html.

Feiler, Peter H.; Gluch, David P.; Hudak, John J.; Lewis, Bruce A. *Embedded Systems Architecture Analysis Using SAE AADL* (CMU/SEI-2004-TN-005). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2004. http://www.sei.cmu.edu/publications/documents/04.reports/04tn005.html.

Goodenough, John; Lipson, Howard; & Weinstock, Chuck. *Arguing Security—Creating Security Cases.* https://buildsecurityin.us-cert.gov/daisy/bsi/articles/knowledge/assurance/643.html?branch=1&language=1 (2007).

Weinstock, Charles B. & Goodenough, John B. *Dependability Cases* (CMU/SEI-2004-TN-016). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2004. http://www.sei.cmu.edu/publications/documents/04.reports/04tn016.html.

Related Web Sites

www.sei.cmu.edu/pcs/
www.aadl.info

For Course Registration

www.sei.cmu/products/courses/p52.html

For More Information

To learn more, please contact
John Goodenough
Phone: 412-268-6391
jbg@sei.cmu.edu
Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213-2612

For General Information

For information about the SEI and its products and services, contact
Customer Relations
Phone: 412-268-5800
FAX: 412-268-6257
customer-relations@sei.cmu.edu
www.sei.cmu.edu

03/26/2007