Incorporating Software Architecture in the Computer Science Curriculum

Martin L. Barrett  
East Tennessee State University  
barrettm@etsu.edu

Ayse Bener  
Ryerson University  
ayse.bener@ryerson.ca

Steve Chenoweth  
Rose-Hulman Institute of Technology  
chenowet@rose-hulman.edu

Abstract

This workshop introduces the concepts of software architecture and how to incorporate these concepts into the computer science and software engineering curriculum. Participants will learn techniques used in industry to specify quality attributes critical to system performance, modifiability, and availability, and to use those attributes to drive the system structure using well-known architectural styles. Exercises will be used to demonstrate the techniques and to practice effective methods for students to learn the techniques in CS and SE classes. Participants will become part of the community of educators sharing educational resources in software architecture.

1. Intended audience

Undergraduate and graduate computer science and software engineering instructors who want to understand the importance of software architecture in the creation of software systems and wish to learn how to incorporate software architecture concepts into the curriculum. Software architecture education prepares students to work on large software systems. Software architecture represents the earliest design decisions that are both the most difficult to get right and the hardest to change downstream. System qualities, such as performance, security, availability, interoperability, or modifiability, are largely supported or precluded once the software architecture has been established.

2. Agenda

1. Overview of software architecture. 15 minutes.
2. Quality attributes and their role in architecture. 30 minutes.
   - overview of quality attribute categories
   - scenario examples
   - brainstorming by participants of additional examples and uses (in small groups)
3. Eliciting quality attributes. 30 minutes.
   - exercise in elicitation and ranking quality attributes (in small groups)
4. Architectural style guide example. 30 minutes.
   - overview of architectural viewtypes and styles
   - component-and-connector example
   - module example
   - allocation example
   - exercise in using multiple views for architecture analysis (in small groups)

5. Software architecture in the curriculum. 60 minutes.
   - undergraduate software engineering courses
   - dedicated software architecture courses
   - examples of successful learning methods in software architecture
   - brainstorming of teaching ideas and strategies (in small groups).
   Possibilities include:
     - useful class, homework, and project ideas
     - methods for inclusion of architecture in undergraduate courses
     - how to sustain an ongoing community of interested educators

6. Workshop wrap up. 15 minutes.