Research and Practice in Software Architecture and Dynamic Software Systems

Judy Stafford and Richard Hall
COMP 250SA – Meeting 1
January 16, 2008

Agenda

- Administrative Stuff
- Course overview
- Introduction to Software Architecture
- Introduction to Dynamic Software Systems
- Expectations
  - Paper Readings and Presentations
  - Individual Software/Paper Research Projects
- Subtopic Overviews
- Discussion
Administrative Stuff

◆ Introductions
  – Instructors
  – Students
◆ Course Structure
  – Overview
    » Motivation for offering course
    » Expectations for students
  – Syllabus
◆ Course Meeting Time?
◆ Reschedule Jan 30 to Jan 23

Architecture

◆ Put some pictures of buildings here – talk about conceptual integrity
  – From different perspectives
    » Code and documentation
    » User experience
    ◆ Different parts of system
    ◆ Different versions of system
Software Architecture (current thinking)

◆ What is it?
  – Something that is done?
    » Who?
    » When?
    » How?
  – Something that is?
    » Description of something?
◆ Why do we need it?
  – Systems are too big for one person to build
  – Need to figure out if the plan is likely to work
  – Reference throughout system life
◆ What does an Architect do?
  – Figures out how to get the stakeholders what they want
  – Planning and strategizing

Categories, Styles, and Views

◆ Three Basic Categories
  – Component&Connector
    » Runtime
    » Generally good for analyzing system’s ability to meet behavioral goals
  – Module
    » Code related --- stuff that will be created
    » Generally good for analyzing maintainability
  – Allocation
    » How Modules and/or Components & Connectors relate to non-software stuff
    » For instance, work assignments (think global development…)
◆ Different categories allow us to focus our attention and avoid mixing concerns
Software Architecture

- Different Categories support communication with stakeholders:
  - Project sponsors
  - New developers
  - Maintainers
- Different stakeholders have different concerns with regard to:
  - Types of information
  - Parts of the system

Architecture-Based Analysis

- Dependence Analysis
  - If I make a change, what else might be affected?
  - The system doesn’t work correctly. What could be causing it?
- Performance Analysis
  - What’s the best decision to ensure adequate response time?
- Maintainability
  - How hard is it to replace a part?
- Evolvability
  - Can new features added easily?
Abstraction

◆ Abstraction supports reasoning about something specific
  – Maps…
    » How long will it take?
    » How do I get there?
    » What’s the best driving route?
    » What’s the easiest walking route?
    » Where can I park?
    » Where is West Hall?

Software Architecture Research

◆ What exactly should be captured?
  – Maybe capture what the Architect knows
    » Decisions
    » Rationale
  – But, how to organize this for use by stakeholders?
    » Knowledge Management
    » Concept Maps
  – Check wicsa program…
BREAK

- Coming Attractions
  - Dynamic Software Systems

Dynamic Software System Demo

- Rather than telling you about it, let me show you...
- Demo “Hello World” dynamic software system
  - Very simple paint program
  - Supports dynamic shape extensions
Dynamic Software Systems

◆ What is a dynamic software system?
  – A software system whose structure can change at run time
    » i.e., constituent pieces and the connections among them
  – e.g., operating systems

What is the Range of Dynamism?

◆ System knows how and who
  – e.g., installable options
◆ System knows how, but not who
  – e.g., plugins
◆ System does not how nor who
  – e.g., operating systems, application servers
  – In truth, system generally has some idea how...
Which Types of Dynamism to Manage?

- **Implementation Dynamism**
  - e.g., upgrades, reconfigurations
  - Under control of the system or its operator

Which Types of Dynamism to Manage?

- **Environment Dynamism**
  - e.g., external failures, QoS changes, priority changes
  - Not under control of the system or its operator
When to Manage Dynamism?

- **Compile time**
  - i.e., #ifdef
- **Link time**
  - i.e., static linker
- **Load time**
  - i.e., dynamic linker
- **Run time**
  - i.e., continuously during execution
  - Most interesting

Where to Manage Dynamism?

- **In system code itself**
  - Less interesting
- **In middleware**
  - More interesting
- **In architecture**
  - Most interesting
How is Dynamism Managed?

◆ Restart/reboot
  – e.g., Windows 95
◆ Visible error handling
  – e.g., try-catch blocks
  – i.e., distributed systems
◆ Managed/transparent
  – e.g., proxied access

Why Create Dynamic Systems?

◆ Rate of change is too great for static approaches
◆ Too important to shutdown...
  – ...even for a little while
  – e.g., home health care, security, financial, energy
◆ Innovative application domains
  – e.g., ubiquitous/context-aware computing, grid computing
Challenges of Dynamic Software Systems

◆ Related to many other software engineering disciplines
  – Dynamism is generally not a first-class concept
    » Most SE concepts target static software systems, must be extended
◆ Very difficult to manage transparently
  – Both during design and at run time
◆ Dynamism will eventually require
  – Semantic, contextual, and behavioral description
  – Artificial intelligence

What’s Next

◆ Topic(s)
  – Sub-topic overviews
  – Tracking things down
    » Using the library (Inspec, Citeseer, google…)
    » Knowing good sources from bad
◆ Your todo list
  – Reading
  – Submit paper preferences if you want
    » Send preferences to Judy and/or Rick
  – Project proposal due Feb 3 copy it into:
    » /g/250SA/public_html/projprops