OMSE 532 - Week 1
Introduction to Software Architecture

Dr. Stuart Faulk
Computer and Information Science
University of Oregon

1

Overview

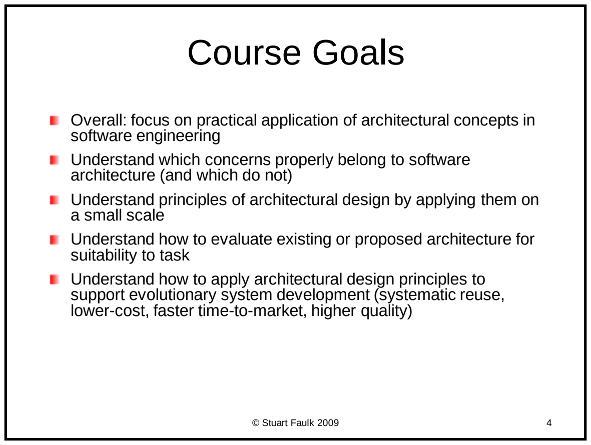
- Oregon Master of Software Engineering: the course in context
- Course Goals
- What is architecture? (discussion)
- The Architecture Business Cycle

2

OMSE Curriculum Themes

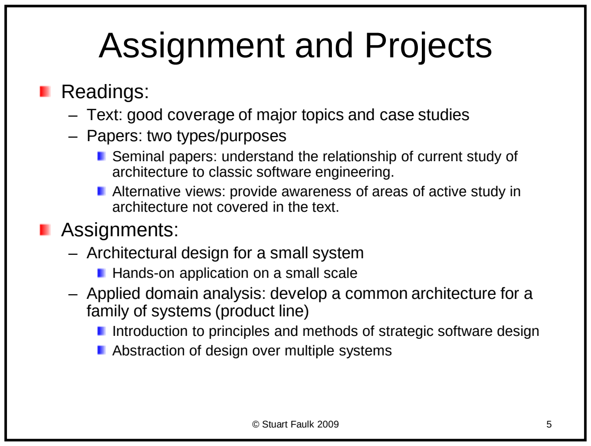
- Software Engineering in Context
  - Recognizes that real software products are always developed in a larger context (organizational, business, regulatory, etc.)
  - Effective technical decisions cannot be made without understanding the non-technical issues
  - No where is this more true than in decisions about software architecture
- Strategic Software Engineering
  - Looks beyond development of single systems
  - Recognizes that we typically build many, often similar systems over time.
  - Focus on overall process improvement, planned reuse, and other strategic development issues
  - Understand role of software as a strategic business asset
- Themes intersect most strongly in software architecture

3



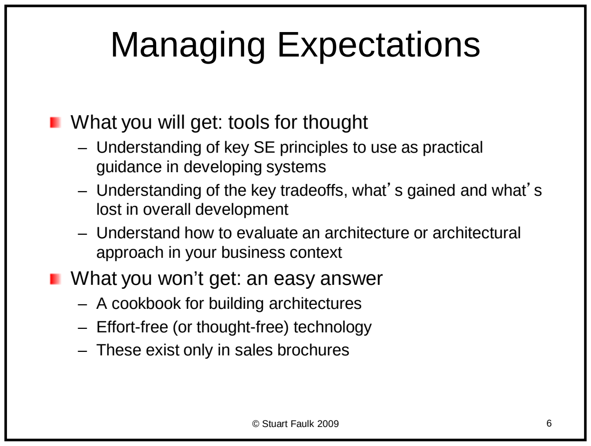
Course Goals

- Overall: focus on practical application of architectural concepts in software engineering
- Understand which concerns properly belong to software architecture (and which do not)
- Understand principles of architectural design by applying them on a small scale
- Understand how to evaluate existing or proposed architecture for suitability to task
- Understand how to apply architectural design principles to support evolutionary system development (systematic reuse, lower-cost, faster time-to-market, higher quality)



Assignment and Projects

- Readings:
  - Text: good coverage of major topics and case studies
  - Papers: two types/purposes
    - Seminal papers: understand the relationship of current study of architecture to classic software engineering.
    - Alternative views: provide awareness of areas of active study in architecture not covered in the text.
- Assignments:
  - Architectural design for a small system
    - Hands-on application on a small scale
  - Applied domain analysis: develop a common architecture for a family of systems (product line)
    - Introduction to principles and methods of strategic software design
    - Abstraction of design over multiple systems



Managing Expectations

- What you will get: tools for thought
  - Understanding of key SE principles to use as practical guidance in developing systems
  - Understanding of the key tradeoffs, what's gained and what's lost in overall development
  - Understand how to evaluate an architecture or architectural approach in your business context
- What you won't get: an easy answer
  - A cookbook for building architectures
  - Effort-free (or thought-free) technology
  - These exist only in sales brochures

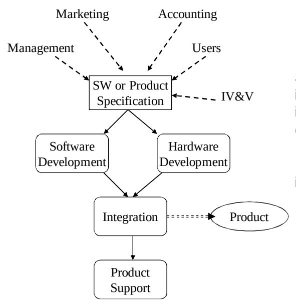
The Architectural Business Cycle

What is Architecture? Working Definition

"The software architecture of a program or computing system is the structure or structures of the system, which comprise software components, the externally visible properties of those components, and the relationships among them."

From *Software Architecture in Practice,* Bass, Clements, Kazman

SW Life-Cycle in Product Context



SW development inevitably occurs in a larger context (e.g., business, govt.)
…
Implies evolving external influences and objectives

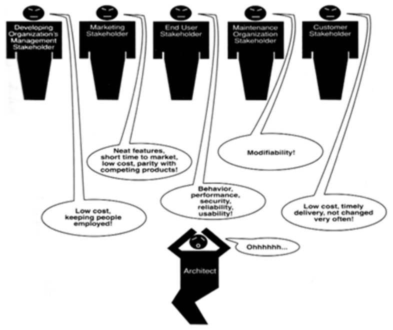
Effects of Architectural Decisions (What?)

- What kinds of system and development properties are affected by the system structure(s)?
- System run-time properties
  - Performance, Security, Availability, Usability
- System static properties
  - Modifiability, Portability, Reusability, Testability
- Production properties? (effects on project)
  - Work Breakdown Structure, Concurrency, Project cost, time to market
- Business/Organizational properties?
  - Lifespan, Versioning, Interoperability, Target market

Affects of Arch. Decisions (Who)

- What kinds of people or organizations have a stake in what choices are made between those properties?
- Why is it important for architects to understand what is wanted, who wants it, and how important each issue is?

Influences on the Architect

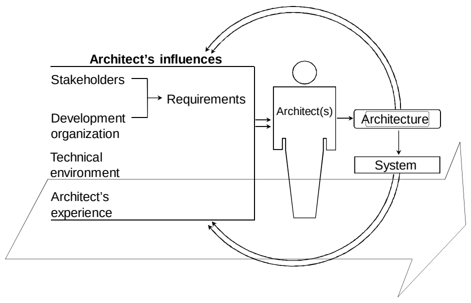

Developing Organization's Management Stakeholder
Marketing Stakeholder
End User Stakeholder
Maintenance Organization Stakeholder
Customer Stakeholder
Neat features, short time to market, low cost, parity with competing products!
Modifiability!
Behavior, performance, security, reliability, usability!
Low cost, keeping people employed!
Low cost, timely delivery, not changed very often!
Ohhhhhh...
Architect

Classes of Influences (per SAP)

- Customer and end user
  - Development cost, maintenance cost
  - Performance, reliability, availability, platform independence, interoperability
- Developing Organization
  - Immediate business: context of development such as reuse of legacy code, development cost, schedule
  - Long term business: strategic concerns such as building a system infrastructure
  - Organizational structure
- Technical Environment
- Architect's Background and Experience

Making successful tradeoffs requires understanding the *nature, source* and *priority* of these constraints.

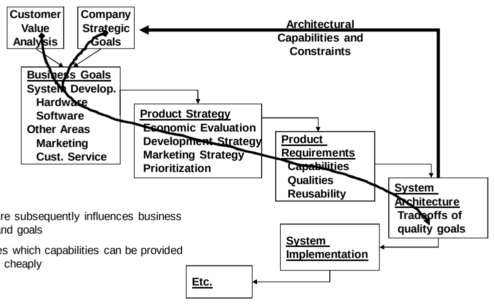
Architecture Business Cycle (ABC)



Architectural Feedback

- Architecture influences the things that influence it
- Architecture influences organization
  - Influences organizational structure by work breakdown
  - As an organizational resource, may affect business goals
- May influence customer requirements since it affects the cost and speed of subsequent development
- Act of construction influence the architect's experience (for good or ill)
- Occasionally influences the technical environment (e.g., Android or the Web)

The Business Cycle



• Architecture subsequently influences business decisions and goals

• Determines which capabilities can be provided quickly and cheaply

Where do architectures come from?

- Created by “software architects”
- Created in response to demands for particular qualities of the software product
- A product of tradeoffs between conflicting demands.
- Implication: *ideally a balance of technical, business and social influences*

Why Study Architecture in SE?

Definition: The *purpose of software engineering* is to gain and maintain intellectual control over the products and processes of software development*

- “Intellectual control” - means that we can make rational choices based on an understanding of the (downstream) effects of those choices.
- “managerial control” is related but focuses on control of software development *resources* (money, time, personnel).

*Memorize this defn!

Meaning of *Intellectual Control*

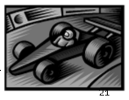
- Software development progresses though a sequence of decisions
  - Decisions about requirements (e.g., tradeoffs, priority)
  - Design decisions (e.g., first decomposition)
- Earlier decisions affect the difficulty of later decisions
  - Ensuring that we end up with the desired properties requires making the right decisions in the right order
  - E.g., cannot add properties like security late in the game, Windows demonstrates this vs. OSX (Unix private address space)
- Being *in control* means we can:
  - Decide in advance the functional and non-functional requirements the software should satisfy
  - Proceed systematically though the steps of software development to produce a system meeting those requirements

Meaning of *Managerial Control*

- Managerial control means we are able to make rational choices about *development resources*
- Real projects have finite set of resources: time, people, money
- Must choose where, when, and how much resources are allocated
- Being in control means we can:
  - Decide in advance the level of recourses needed to deliver software meeting requirements
  - Deliver that software on time and within budget

Analogy to Driving a Car

- Driving a car
  - In control: decide in advance where we want to go, how long, it should take, how much fuel it will take
  - Out of control: end up at a different destination, takes twice as long as expected, uses twice as much fuel, etc.
- Managing a software project
  - In control: decide in advance what capabilities and properties the software will have, how long it will take, how much effort
  - Out of control: software is delivered with less or wrong capabilities, delivered late, over budget
- Many software projects are out of control in this sense
  - Perfect control is not possible
  - Requires constant feedback and correction



SE for Architecture

- Need for engineering of software architecture
  - Profoundly affects system and business qualities
  - Requires making tradeoffs
  - Control implies achieving system qualities by choice not chance
    - Understanding what the tradeoffs are
    - Understanding the consequences of each choice
    - Making appropriate choices at appropriate times
- This is the central topic of this course! Varies in…
  - What we are trying to control
  - How we control it
  - When we make decisions
  - How we make tradeoffs
  - How we measure success

Course View of "Architecture"

- What is it?
- What is it for?
- What does this imply about design?

Working Definition

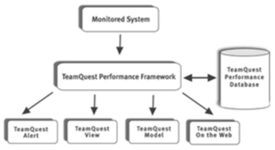
"The software architecture of a program or computing system is the structure or structures of the system, which comprise software components, the externally visible properties of those components, and the relationships among them."

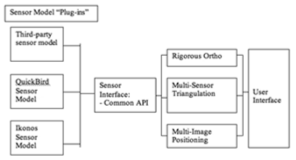
From *Software Architecture in Practice,* Bass, Clements, Kazman

Key Points of Definition

- An architecture = components + relations + interfaces
- Deals only with *externally visible properties* of components and relations
  - Externally visible component behavior is part of an architecture
  - "Behavior" and "component" have a very broad meaning here
- Is an *abstraction* in that much information is omitted
- Systems typically comprise *more than one* architecture
- *It exists* whether any thought goes into it or not
- Most things called "architecture" aren't

Is it architecture?



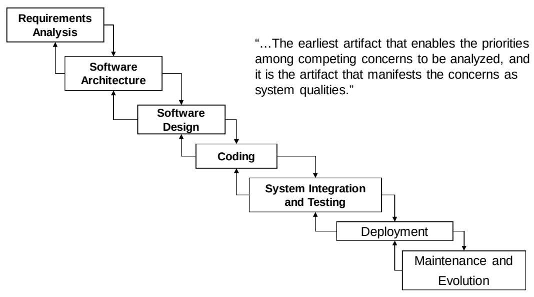


Purpose of Architecture

- Capture early design decisions
  - Focus on decisions affecting important system properties
  - What is important about the early decisions? (What happens if we get it wrong?)
- Communicate decisions among stakeholders
  - Why is *accurate* communication needed?
  - What does this imply about representing architectures and architectural design choices?

=> Implication: must provide *transferable abstractions*
  - i.e. a representation of architectural design decisions that can be preserved and conveyed

Fit in the Development Cycle



"…The earliest artifact that enables the priorities among competing concerns to be analyzed, and it is the artifact that manifests the concerns as system qualities."

© Stuart Faulk 2009 28

Implications for "Architectural Design"

View design as the process of building in the required system properties

- We design to a purpose
- We need to choose appropriate components and relations for the purpose
  - What are the issues in "appropriate?"
- That we need an effective design approach
  - Goal: systematically achieve desired properties
  - Principles to guide making appropriate choices
  - Representation to make it visible
- That we need some metric of "goodness" relative to purpose

© Stuart Faulk 2009 29

Architectural Design Process

Methodology independent view implied by *purpose* and *context*

1. Creating the business case for the system
2. Understanding the requirements
3. Creating or selecting the architecture
4. Representing and communicating the architecture
5. Analyzing or evaluating the architecture
6. Implementing the system based on the architecture
7. Ensuring the implementation conforms to the architecture

© Stuart Faulk 2009 30

Overview of Syllabus

Syllabus Overview

- Architectural Design (Weeks 1-6)
  - What it means to design at an architectural Level
  - Characterizing and achieving architectural quality attributes
  - Architectural design and documentation
  - Architectural evaluation
- Advanced topics (Weeks 7-10)
  - Other views of architecture
  - Domain analysis and domain-specific software architectures

Where We are Going

- “Tactical” view: architectural design to achieve system objectives
  - Performance, Security, Availability, Usability
  - Modifiability, Portability, Testability
  - Key issue: how to design architecture, make trade-off’s to achieve specific goals
- “Strategic” view: architectural design to achieve business goals
  - Focus on reducing cost, errors, and time-to-market
    - Developing architecture as a corporate asset
    - Springboard for systematic reuse
  - Moves away from single-instance, hand crafted code toward product families, large-scale reuse and code-generation
  - Key issue: how to design architecture so it can by systematically reused over multiple developments
    - Learn prerequisites, principles, process, and products
    - Construct a family architecture for small system

End

© Stuart Faulk 2009 34