What?
Architecture Definition

Ruth Malan
Bredemeyer Consulting
Tel: (812) 335-1653
Fax: (812) 335-1652
Email: ruth_malan@bredemeyer.com
Web: http://www.bredemeyer.com

Architecture Impetus

...the dog houses have been built… You can't build a sky rise the way you build a dog house…

Booth, SD'99

Architecture 501

- What is architecture?
  - the set of decisions that an architect makes
- What decisions does the architect make?
  - the architecturally significant ones
- What is architecturally significant?
  - the architect decides

What: System Architecture
Eb Rechtin's Definition

- System
  - "A system is defined ... as a set of different elements so connected or related as to perform a unique function not performable by the elements alone." p7
- Architecture


  - "The term 'architecture' is widely understood and used for what it is--a top-down description of the structure of the system."
    - *Systems Architecting: Creating and building complex systems*, Eberhardt Rechtin, Prentice-Hall, 1991

What: Software Architecture Formal Definition

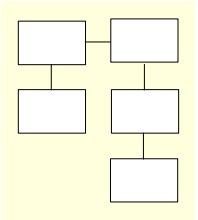
- "architecture is the structure of the system, comprised of
  - components or building blocks
  - the externally visible properties of those components, and
  - the relationships among them"
    - adapted from Bass, Clements, and Kazman. *Software Architecture in Practice*, Addison-Wesley 1997

Is the Jar Full?

Is the Jar Full Now?

Architecture Essentials
Large Rocks First

- Key idea: *Put the "large rocks" in place first*
- What are the "large rocks"
  - large-grained chunks of the system



  - important properties of the system

Architecture: More than Decomposition—Do the pieces *fit*?

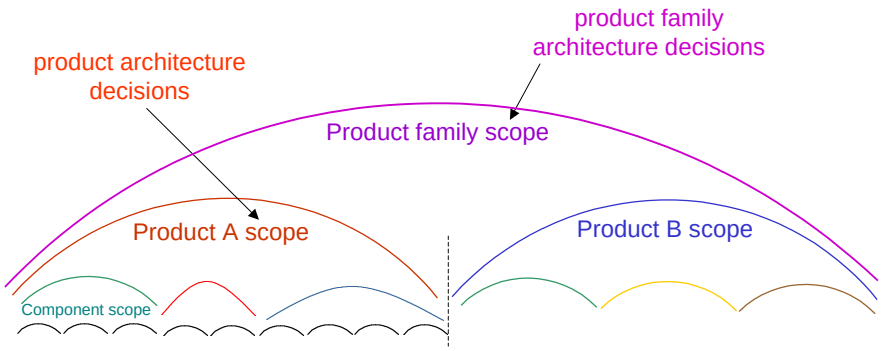
- Assign world's best engineers to pick best
  - engine
  - transmission
  - suspension
  - etc
- Can they build the world's best car?

adapted from Russ Ackoff

Architectural Perspective

- *System integrity can't be achieved bottom-up*
  - if you optimize the parts, you *will* compromise the whole
- You need a system-wide perspective to
  - address cross-cutting concerns
    - design architectural mechanisms to address the system properties
  - make the tradeoffs necessary to ensure that the important system properties are met
- Architectural decisions optimize the whole
  - making compromises for some of the parts to achieve the overall good of the whole

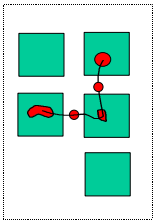
Architectural Decisions
A matter of scope



Architecture is the set of decisions that cannot be delegated without compromising overall system objectives.

Software Architecture
Key Concerns

- System decomposition
  - how do we break the system up into pieces?
  - do we have all the necessary pieces?
  - do the pieces *fit* together?
- \+ Cross-cutting concerns
  - broad-scoped qualities or properties of the system
  - tradeoffs among the qualities
- \+ System integrity



Software Architecture
Key Concerns

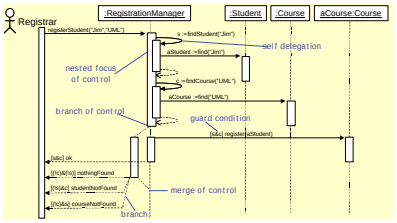
- System decomposition
- Cross-cutting concerns
- System integrity

and

- Alignment with business
  - with business strategy
  - with business environment
    - legacy and existing investments
    - organizational capabilities and culture
  - with customers and channel
- System evolution
  - Architectures are long-lived!
  - they must provide the blueprint for implementing today's strategy, *and*
  - they must to be able to evolve, because the business strategy *will* change (with increasing frequency)!

Architecture Representation

- Architecture models
  - thinking tools
    - explore alternatives and ideas (more cheaply than prototyping or trial by building the system)
    - e.g., find interface operations by exploring component collaborations
  - document the architecture
    - descriptive or prescriptive
  - communicate the architecture
    - help visualize the system



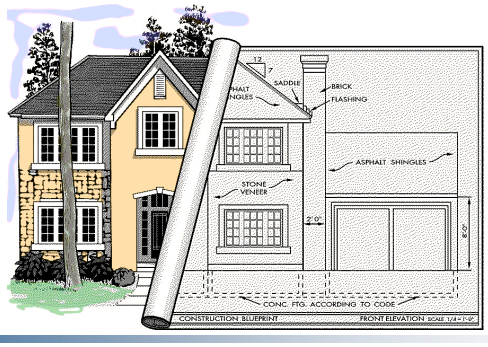
- Architecture documentation
  - architecture models

  + rationale, assumptions, explanations, implications

Architecture Views

- Different audiences have different information needs

customer



builder

What Do You Get From This? What Does Your Manager Get?

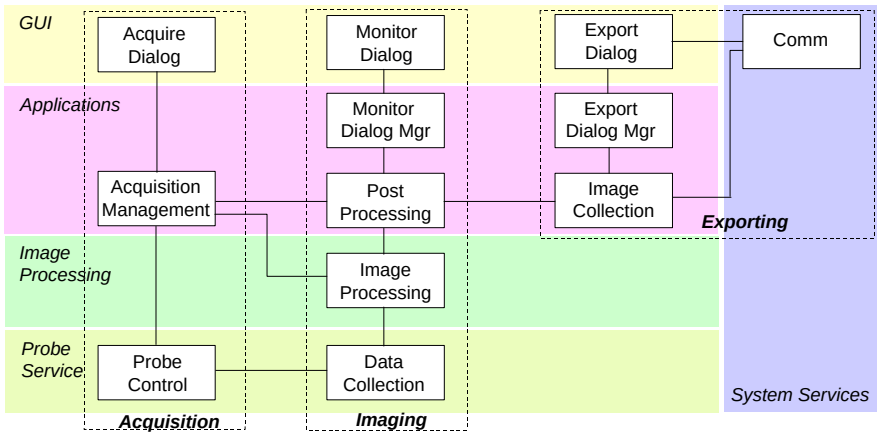
interface ContextManager {

exception UnknownParticipant { long unknownParticipant; }
exception TransactionInProgress { string instigatorName; }
exception InvalidTransaction { string reason; }
exception InvalidContextCoupon { }
exception ChangesNoteEnded { }
exception AcceptNotPossible { }
...
StartContextChanges (in long participantCoupon, out long contextCoupon) raises (UnknownParticipant, TransactionInProgress, InvalidTransaction)

EndContextChanges (in long contextCoupon, out boolean noContinue, out string[ ] responses) raises (InvalidContextCoupon, NotInTransaction, InvalidTransaction)

PublishChangesDecision (in long contextCoupon, in string decision) raises (NotInTransaction, InvalidContextCoupon, ChangesNotEnded, AcceptNotPossible)
… }

What Do You Get From This? What Does Your Manager Get?



Software Architecture Views

Overall system view

Blueprint for developers
- **unambiguous**
- **precise**
- **actionable**

Conceptual Architecture

- Architecture Diagram, CRC-R cards
- *Focus: identification of components and allocation of responsibilities to components*

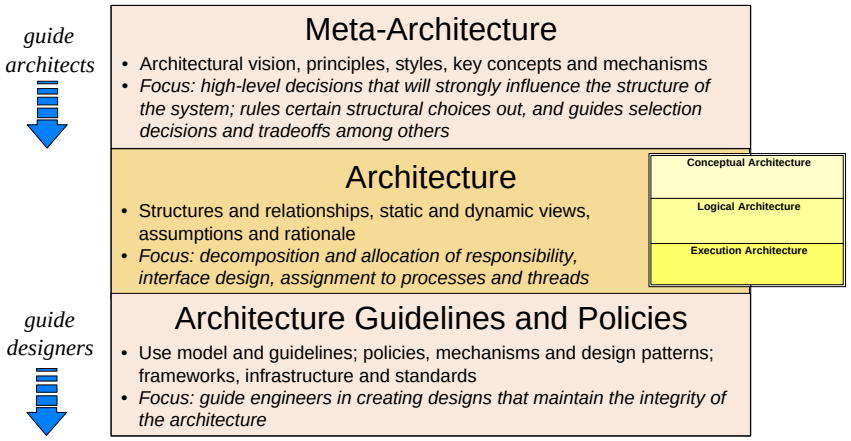
Logical Architecture

- Updated Architecture Diagram (showing interfaces), Interface specifications, Component specifications and usage guides
- *Focus: design of component interactions, connection mechanisms and protocols; interface design and specification; providing contextual information for component users*

Execution Architecture

- Process View (shown on Collaboration Diagrams)
- *Focus: assignment of the runtime component instances to processes, threads and address spaces; how they communicate and coordinate; how physical resources are allocated to them*

Architecture Decision Framework



Architecture Decisions
Not Simply an Matter of Abstraction

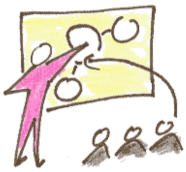
- Some Software Architecture decisions will be very high level, and some may be quite detailed and "low level"
  - Some architectural objectives can be achieved by Meta-Architecture (e.g., an Architectural Principle) alone
  - Some architectural objectives must be solved by working together at the product family level on quite detailed aspects of the system, e.g.,
    - components and interfaces at the interface between interoperating applications (e.g., CCOW for context management)
    - standards to allow interoperability, information sharing, and convergence of the infrastructure to support these

Minimalist Architecture

- Minimalist Architecture Principle: Keep your architecture decision set as small as it possibly can be, while still meeting your architectural objectives

See Ruth Malan and Dana Bredemeyer, "Less is More With Minimalist Architecture," *IT Professional*, IEEE, September 2002.
http://www.bredemeyer.com/pdf_files/MinimalistArchitecture.PDF

Review



- We have covered
  - *What* architecture is
    - Building blocks of the system, their externally visible properties and relationships to each other and the environment
  - Our Architecture Decision Framework
    - layered decision model, consisting of Meta-Architecture, Architecture and Architectural Guidelines and Policies
    - Architecture is represented through views
      - Conceptual, Logical, Execution Architecture
      - other views as appropriate to cross-cutting concerns, e.g., security view

Architecture Book

- *Software Architecture Action Guide*
  by Malan, Ruth and Dana Bredemeyer, see *draft chapters at*
  http://www.bredemeyer.com/ArchitectingProcess/SWAActionGuideTOC.htm

Part I: Software Architecture and the Visual Architecting Process

1. Software Architecture: Central Concerns, Key Decisions
2. The Visual Architecting Process: Good, Right and Successful
3. Initiate and Gain Commitment: Getting Started
4. Meta-Architecture: Getting Strategic
5. Conceptual Architecture: Getting the Big Chunks Right
6. Logical Architecture: Getting Precise, Making Actionable
7. Execution Architecture: Getting Physical
8. Architecture Guideline and Policies: Getting Specific
9. Architecture Deployment: Getting Real

Resources

- Resources for Software Architects web site
  - http://www.bredemeyer.com
- Training from Bredemeyer Consulting
  - Role of the Architect Workshop, Bloomington, IN, May 26-28, 2005
  - Software Architecture Workshop, Indianapolis, IN, September 26-29, 2005