Master-Builders Have Rich Conceptual Models of Software Design

George Fairbanks

22 May 2012

Rhino Research
Software Architecture Consulting and Training
Letter to your 20-year-old self

• You would like to be a master-builder, fast

• What would you tell the younger you?
Roman engineers

- We are no smarter today
- ... but kids can do better than the best 2000 years ago
- Partly: better materials; mostly: better concepts
- Today, we teach an improved conceptual model
Our knowledge is like a wall

- Easy to describe the bricks
- **Conceptual model** is the cement
What is a conceptual model?

- **What is a conceptual model?**
  - A conceptual model is a set of concepts that can be imposed on raw events to provide meaning and structure.

- **It organizes chaos**
  - Enables intellectual understanding
  - Fits big problems into our finite minds

- **Synonyms:**
  - Conceptual framework
  - Mental model
Without conceptual models

- **Traveling to new countries**
  - Everything is harder
  - You didn’t become stupid
  - ... but your conceptual model doesn’t work

- **Fighting with bureaucracy**
  - No understanding of the system or how to navigate it
  - Not clear if it has principles or if you are ignorant

- **The program crashes because it gets tired**

- **It won't rain**
  - Perhaps doing a dance will make it rain
  - If not, try throwing virgins into volcanoes

- **With conceptual model**
  - Dancing, rain, virgins, and volcanoes are not causally related
Example: Sports

- Things you cannot see or touch are important
- Conceptual models bring their own vocabulary

Plays, strategies, assignments
Example: Dandelions
Examples of conceptual models

Sports: Plays, strategies, assignments

Physics: Free Bodies

Energy cycle

Econ: Supply & demand

Accounting: Debits & credits
With a conceptual model
Conceptual model of software architecture

- Model relationships
  - Views & viewtypes
  - Designation
  - Refinement

- Canonical model structure
  - Domain model
  - Design model
    - Recursively
  - Code model

- Quality attributes

- Design decisions

- Tradeoffs

- Responsibilities

- Constraints (guide rails)

- Viewtypes
  - Module
  - Runtime
  - Allocation

- Module viewtype
  - Modules
  - Dependencies
  - Nesting

- Runtime viewtype
  - Components
  - Connectors
  - Ports

- Allocation viewtype
  - Environmental element
  - Communication channels
Where to learn it?
Kent Beck’s model

• Hierarchy: Values ‥ Principles ‥ Patterns

• Values:
  ▶ Communication, simplicity, flexibility

• Principles:
  ▶ Local consequences, minimize repetition, minimize repetition, logic and data together, symmetry, declarative expression, rate of change

• Note: no named abstractions
  ▶ E.g., modules, dependencies, ...

Copyright 2007-2012 George Fairbanks, Rhino Research
Interrogation by David Garlan

- Yearly software architecture class at Carnegie Mellon
- David interrogates students at end of semester

- Invariably:
  - David catches mistakes in reasoning (connections)
  - David finds holes (absence)

- Lessons
  1. Conceptual model is teachable
     - Even though David knew it better
  2. A conceptual model is helpful
     - Students know the “bricks” better than David
  3. Knowing the abstractions is not enough
     - They are themselves bricks (recursive!)
Master builder of buildings

- Master builder: My brother William
- Works across projects
- Diagnoses & solves problems
- Example: bamboo floor
Conclusion

- Invisible, colorless, odorless
  - Conceptual models are easy to miss

- Conceptual models
  - Key difference between novice and expert

- Become a master builder
  - Study the software architecture field

- What would you write in a letter to a younger you?
  - Hard question: how do we express that knowledge?
About me (George Fairbanks)

• PhD Software Engineering, Carnegie Mellon University

• Thesis on frameworks and static analysis (Garlan & Scherlis advisors)

• **Program chair: SATURN 2012**;
  Program committee member: WICSA 2009, ECSA 2010, ICSM 2009; CompArch 2011 local chair

• Architecture and design work at big financial companies, Nortel, Time Warner, others

• Teacher of software architecture, design, OO analysis / design

• **Author: *Just Enough Software Architecture***