Requirements Engineering is *SO* Twentieth Century...

Richard N. Taylor
Institute for Software Research
University of California, Irvine
http://www.ics.uci.edu/~taylor

The Voices of the Sages

- Pappus, a.d. 300: "In analysis we start from what is required, we take it for granted, and we draw consequences from it,till we reach a point that we can use as [a] starting point in synthesis."
- Polya "How to Solve it":
  - First, you have to understand the problem.
  - Second, find the connection between the data and the unknown. You should obtain eventually a plan of the solution.
  - Third, carry out your plan.
  - Fourth, examine the solution obtained.

Problem Frames

- "The central part of this paper sketches an approach to problem analysis and structuring that aims to avoid the magnetic attraction of solution-orientation. The approach is based on the idea of a problem frame [1-3]. A problem frame characterises a class of elementary simplified problems that commonly occur as subproblems of larger, realistic, problems. The intention is to analyse realistic problems by decomposing them into constituent subproblems that correspond to known problem frames. This analysis guides the decomposition, gives warning of the concerns and difficulties that are likely to arise, and provides a context in which previously captured experience can be effectively exploited." — Michael Jackson, August 2000

Others

- "… systematic derivation of architectures from requirements" — vaguely after van Lamsweerde
- Various OO approaches: essentially building applications as a simulation of the world
- The title of this workshop: "From Software Requirements to Architectures"

Why the Primacy of Requirements?

- Parnas's Washing Machines
  - Remember the automated rock basher example?
    - Avoid implementation bias
    - Allow innovation
    - Ease analysis: complete, consistent

Why Are Requirements Done After the Fact, If at All, in So Many Applications?

- Standard answers:
  - Bad engineering
  - Bad discipline
  - Lack of a good mathematical training
  - Lack of time

Maybe the Reason Is Different

- Maybe it is because it hasn't proven useful
- Maybe it is because you can't do a good job with requirements until the architecture is in hand
- Maybe it is a matter of size and complexity?
  - Abstract requirements may work on small(er) problems: Why? you can think of many solutions quickly, or there may only be a small number of solutions AND the total time involved in finding one solution is not so great that anyone will much notice
- Maybe Petroski is right: failure is the driver of engineering and the basis for innovation

Twin Peaks Model (Nusibeh)

- Simultaneously develop requirements and architectures
- *Incrementally* elaborate detail
- Keeps both communities happy and doesn't prescribe an approach too different from common practice
- — but I'll let Bashar present and defend his approach later today…

What Happens If We Follow the Money?

- Do new companies start with a good list of requirements?
- Do VC's (that are still in business) buy a nice list of requirements?
- What do marketeers focus on?
- How do you build market niche?
- How do you minimize your time to market?

The Starting Point, IMHO, is thus:

- Characterizing what you
  - Have
  - Know what would be better with improvement
  - Can/can't change
- Thus, let the architecture(s) drive the requirements

Architectures in the Lead

- Think of requirements as incremental improvements needed to existing architectures, or as compositions of architectures
- Architectures provide a frame of reference
  - a vocabulary
  - a basis for describing properties
  - a basis for analysis
- Create new architectures based upon experience with and improvement to pre-existing architectures
- Beware analogies; beware conceptual predecessors

Conceptual predecessors v. Physical predecessors

- Physical predecessors: architecture-based evolution
- Conceptual predecessors provide a vision (high risk/high innovation)
  - Ascension: steps, ladders, elevators, airplanes, rockets
  - Parnas's washing machine example: makes an amusing anecdote , but doesn't reflect how we actually got washing machines
  - … but requirements should be stated in terms of architectural compositions or improvements if at all possible
- Moral: greenfields are minefields in seductive disguise
  - (Quickly start looking for physical predecessors when faced with a new vision)

Are All Architectures up to the Task of Being "Improved" in a Cost-effective Way?

- Adaptable architectures; composable elements
- Not all architectural styles are created equal
  - And are not necessarily up to the task posed by my position

So, Do We Need Requirements at all?

- You do have to know your objective before you start new work.
- You do need a contract with the customer
  - (but when you are building to a market?)
- But let architectures:
  - Provide the vocabulary
  - Provide the basis for discussion
  - … as well as *being* the solution basis
- Thus: new objectives and solutions, from old problems and old solutions

So the Workshop Is Mis-Directed

- It should be:

  "From Architectures to New Architectures, With a Requirements Sidebar"