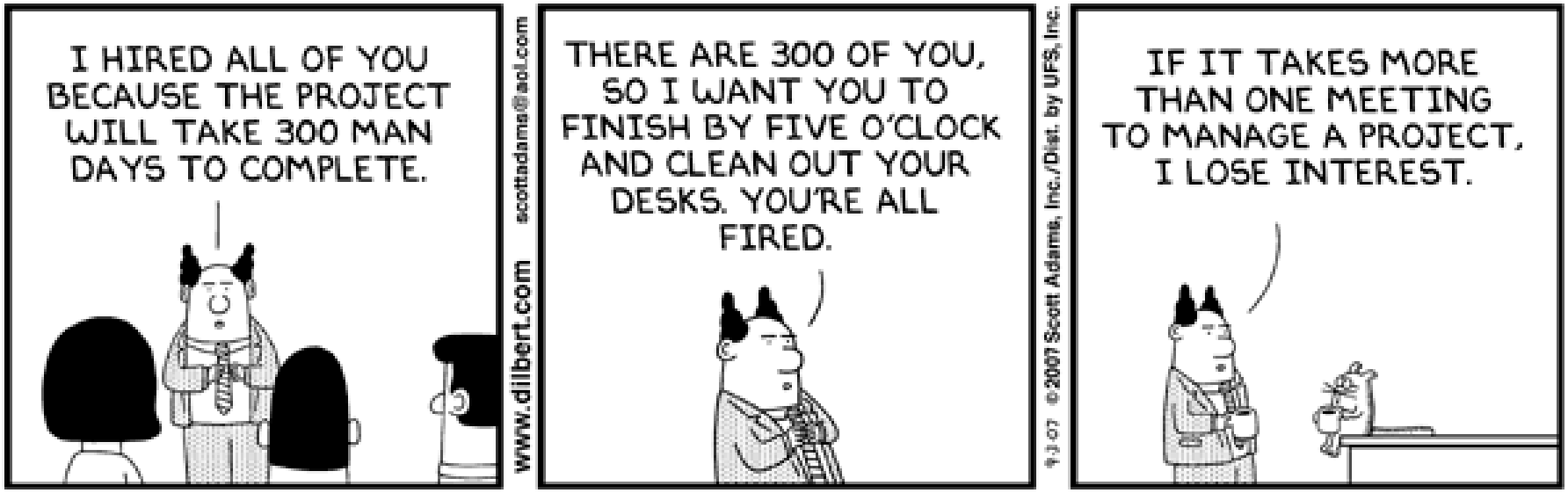
Different Views of the Process of Engineering SW Systems

Dewayne E Perry
Office: ACE 5.124 – Hours M/W 11-12:00
Phone: +1.512.471.2050
perry @ ece.utexas.edu
www.ece.utexas.edu/~perry/education/

Dilbert meets Fred Brooks ☺



SW versus HW Systems

- Hardware Systems
  - Deterioration with age
    - Wear and tear, corrosion, pollution etc
  - Improvement: major redesign, retooling and construction
- Software Systems
  - Good news
    - Software "ingrades" incrementally and continuously
    - Improve correctness — correct faults — corrective
    - Improve usefulness — add new features — adaptive
    - Improve characteristics — eg, performance — perfective
  - Bad news
    - Introduce new faults, more complexity
    - Context can change and reduce usefulness and expose faults
  - Called "soft" for good reasons
    - Malleable, with a low cost of change
    - BUT, cost equation has changed
      - ✓Past: hardware expensive, people cheap
      - ✓Now: people expensive and hardware cheap

50k View of Software Engineering

- Two critical concepts
  - Problem space – ie, the world
    - The world has lots of things in great variety
    - Information, objects, processes, etc
  - Solution space – ie, the machine (the entire system)
    - Solution languages, structures, representations
    - Eg, GUIs, DBs, middleware, protocols, components, etc
- We define a problem we want to solve in the world
  - Problems are often ill-defined, ill-understood
  - Begin by focusing on various artifacts and processes
    - Select – choose some, ignore others
    - Abstract – generalize across similarities
- From selection/abstraction process, we begin to create a theory (we call them requirements)
  - Iterate and improve our understanding of the problem
  - Improve our understanding of the elements in the world
  - May create multiple sub-theories – consistency a problem
  - May formally describe, analyze, and reason about our theory

50k View of Software Engineering

- Then reify the theory into an executable model
  - Ie, create a solution to the problem in the solution space
- We create the model (the software system) in stages
  - Architecture – basic concepts, structures and critical constraints
  - Design – abstractions, data structures and algorithms
  - Code – representations and detailed logical steps
  - Automatic generation to executable model
    - Compilation, linking, etc into an executable system
- Then we introduce it into the world
  - Often significantly disturbing the world
  - Certainly changing the world
- Flaws in the ointment
  - The world changes: uses, technologies, desires, facts, etc
  - Things left out often become irritants
  - Initial theory insufficient or inadequate
  - Model may not be good enough of a variety of reasons

Summary: Theories and Models

- Requirements are the theory for the system we create from the real world
  - There may additional domain theory that provides supplementary information to clarify the requirements
- The software system is the model of that theory
- The specification of the "model" is derived from the "theory" and is reified into an operational system.
- We build that model in stages
  - Architecture, Design, Code, Construction & Deployment
- There are, further, theories of how to proceed from a theory for the system to its model
- The models for those theories are sets of processes (some of which we will discuss in this course)

Another Useful View of our Models

- A critical distinction to keep in mind about reasoning about correctness etc
- 2 views
  - Programs as calculations
    - Small neat problems
    - Eg, scientific systems
    - Can be well-founded theory for reasoning
    - Mathematical
    - Calculi for constructing such programs
    - Calculi for reasoning about such programs
  - Programs as behaviors
    - Large messy problems
    - Eg, editors, word processors, internet sales, etc
    - Does not have neat mathematical basis
    - Have to reason differently
    - Logic and domain specific characteristics are critical
    - Patterns, guidelines, hints, etc for constructing such programs