Today’s Lecture

• Discuss additional issues from The Mythical Man-Month
  – The Second-System Effect
  – Passing the Word
  – Why Did the Tower of Babel Fail?
  – Calling the Shot

The Second-System Effect

• An engineer is careful in designing a system the first time
  – He or she realizes that they are working in uncharted territory
  – Extraneous features get delayed until…

• The Second System!
  – Now, you’ve got some experience and you want to throw everything into the design!

Symptoms of Second-System Effect

• Functional Embellishment
  – to an unnecessary degree

• Optimizations to obsolete functionality
  – OS/360 linker had sophisticated program overlay functionality
  – The problem: the application architecture no longer depended on overlays!

  • Side-effect: linker is slower than it otherwise would have been; slower than the compilers whose usage it was meant to reduce!
How to avoid it?

• Must employ extra self-discipline
  – avoid functional ornamentation
  – be aware of changes in assumptions
  – strive for conceptual integrity
• How do manager’s avoid it?
  – Insist on a senior architect with more than two systems under his or her belt

Passing the Word

• Communicating Design Decisions
  – Written Specifications
    • “The Manual”
      – Answers questions
      – Conceptual Integrity
      – Demands high precision
  – Formal Definitions
    • Natural language is not precise
    • Formal notations have been developed to help

Formal Definitions

• Notations help express precise semantics
  – However, natural language is often needed to “explain” the meaning to the uninitiated
• What about using an implementation?
  – Benefits: Precise specification
  – Disadvantages: Over-prescription, potential for inelegance, may be modified!

Communicating Design, continued

• Meetings
  – Weekly half-day meetings
    • Problems and change proposals distributed beforehand
    • Chief architect has final say
  – Annual “Supreme Court” sessions
    • Typically lasts two weeks
    • Agenda typically had 200 items!
Communicating Design, continued

- **Multiple Implementations**
  - Inconsistencies between implementations can identify problems in the specs;
  - With only one implementation, it’s easier to change the manual!
- **The Telephone Log**
  - Or, be sure to capture all decisions made by the chief architect!
- **Product Test**
  - An external test group keeps the implementation honest

Why Did The Tower of Babel Fail?

- **Communication, (the lack of it)**
  - This made it impossible to coordinate
- **How do you communicate in large project teams?**
  - Informally (telephone, e-mail), meetings, workbook
- **Workbook**
  - It is a structure placed on a project’s documents
  - Why is it important? Technical prose lives a long time; best to get it structured formally from the beginning; it also helps with the distribution of information

More on the Workbook

- **OS/360**
  - Each programmer should see *all* the material
  - Each book was updated quickly (one-day)
  - Problem
    - The workbook grew to 5 feet thick!
    - They switched to microfiche; we need to take advantage of on-line artifacts, information management techniques like open hypermedia, information retrieval, and the WWW

Reducing communication paths

- **Communication needs are reduced by**
  - division of labor
  - specialization of function
- **A tree structure often results from applying this principle**
  - However this serves power structures better than communication (since communication between siblings is often needed)
  - So communication structure is often a network
Organizational Structure

• Brooks outlines
  – mission, producer, director, schedule, division of labor, and interfaces between the parts
• The new items are the producer and the director
  – producer: manages project and obtains resources
  – director: manages technical details
• Microsoft’s program and product manager
  – former is director, latter does more marketing than Brooks specifies for producer but has some overlap

Calling the Shot

• Discusses Estimates and data from real projects
  – Estimates for Programming in the small don’t scale
    • You need to add planning, documentation, testing, system integration, and training in large projects
  – Nanus and Farr at System Development Corporation
    • Found that effort vs. program size increases exponentially (1.5)
  – Portman’s Data
    • Tasks taking twice as long as estimated; number of technical work-hours was being vastly overestimated
    • Downtime, higher priority tasks reduced amount of productive work hours towards the project’s goal