*3*
*The Surgical Team*

## *The Surgical Team*

- The Problem
- The Mill's Proposal
- How It Works

## *Staffing for large systems*

- Most managers prefer small, sharp teams
- Not all projects fit a small team's capacity
- How do we manage large projects?

## *Economics*

- Wide productivity gap among programmers, e.g.,
  - 10:1 ratio on best-to-worst productivity
  - 5:1 ratio on program efficiency
- For example, in the extreme, could have $20k/a programmer versus $10k/a programmer
  - 10x more productive
  - 5x more efficient program

## *Small-team approach*

- 200-person project
  - 25 managers
  - Who are also experienced programmers
  - Therefore, fire the other 175 programmers
- But,
  - Still exceeds “small” team, i.e., 10 persons
  - Will require two levels of management
  - Will require support staff

## *Small-teams on large projects*

- E.g., OS/360
  - Peaked at 1,000 persons, including
  - Programmers, secretaries, operators, managers
  - 5,000 person-years from 1963 to 1966
- Assume men and months traded evenly
  - 200-person team would take 25 years!
- Assume 7x productivity and 7x communication
  - 10-person team would take 10 years!

## *Mill's proposal*

- Specialize the project roles
- Ensure conceptual integrity
  - i.e., 1 or 2 persons do the high-level design
- Maximize throughput at the bottleneck
  - i.e., other roles support the designers
- Surgical team

## *Surgeon*

- A.k.a., chief programmer
- Personally defines the specifications
- Creates the code
  - Designs, writes, tests, and documents
- Has direct access to host and target machines
- Possesses great talent, e.g.,
  - 10+ years experience
  - Extensive technical and application knowledge

## *Co-pilot*

- Surgeon's alter ego
  - i.e., same skills but less experienced
- Shares in thinking, discussing and evaluating
- Represents team in place of surgeon
- Knows all code intimately
- Researches design alternatives
- Provides insurance against loss of surgeon

## *Administrator*

- Manages all non-technical resources
  - e.g., personnel, space, money
- Manages all non-technical constraints
  - e.g., legal, contractual, regulatory and financial
- Subordinate in authority to the surgeon

## *Editor*

- Transforms the surgeon's technical information into useable documentation
  - e.g., organizes with sections, cross-references, bibliography and indices
- Refines multiple successive versions
- Oversees physical production and distribution

## *Program clerk*

- Organizes all files
- Relieves programmers of clerical chores
- In Brooks' day
  - Key-entered data sets; Indexed source listings
- Today
  - Manages the source code and file repositories
  - May also manage the build process and trouble-reporting system

## *Toolsmith*

- Maintains standard software development tools
  - e.g., machines, operating systems, user accounts, editors, compilers, libraries, IDE's
- Acquires or creates specialised tools
  - e.g., performance profilers, configuration editors, legacy language translators
- Today, toolsmith responsibilities may split between a system administrator and a toolsmith

## *Tester*

- Devises, implements and runs all tests, including
  - System-level tests for validation, and
  - Component-level tests for unit verification
- Organizes test cases and files test run results
- Prepares and maintains test scaffolding
- May manage daily smoke test

## *Language lawyer*

- Consults on programming language issues
- Devises tricks to exploit language features
- Today, the language lawyer role is made obsolete through instant Internet access to
  - Standard language specifications
  - Public forums with expert advisors

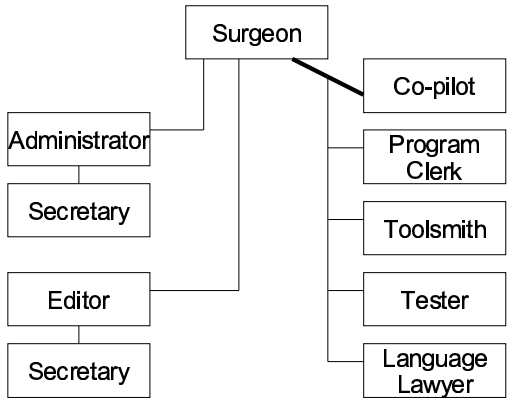
## *Secretaries*

- Assist the Administrator and the Editor
- Handle all correspondence and non-product files
- Teams may expand this role to manage meeting minutes, especially technical meeting minutes involving the surgeon

## *How it works*

- Multiple contributors, e.g., 10 people
- But with a single mind, i.e., surgeon
- Consequently
  - No division of technical task
  - Assurance of conceptual integrity
- Contrasting with
  - Negotiation among technical "peers"
- E.g., C versus PL/1, Java versus Ada

## *Communication pattern*




## *Scaling up*

- Super-surgeon designs overall architecture
- Co-ordinate the subteam surgeons
- E.g., 200-person project =
  - 20 surgical teams X
  - 10 persons per team
- Requires a sharp distinction between architecture and implementation