Teams

Chapter 4

Part #1

1

4.1 Team Organization

A product must be completed within 3 months, but 1 person-year of programming is still needed

- Solution:
  - If one programmer can code the product in 1 year, four programmers can do it in 3 months
- Nonsense!
  - Four programmers will probably take nearly a year
  - The quality of the product is usually lower

2

Task Sharing

- If one farm hand can pick a strawberry field in 10 days, ten farm hands can pick the same strawberry field in 1 day

3

Programming Team Organization

Example:

- Sheila and Harry code two modules, m1 and m2, say

What can go wrong

- Both Sheila and Harry may code m1, and ignore m2
- Sheila may code m1, Harry may code m2. When m1 calls m2 it passes 4 parameters; but m2 requires 5 parameters
- Or, the order of parameters in m1 and m2 may be different
- Or, the order may be same, but the data types may be slightly different

4

Communications Problems

- Example
  - There are three channels of communication between the three programmers working on a project. The deadline is rapidly approaching but the code is not nearly complete
- "Obvious" solution:
  - Add a fourth programmer to the team

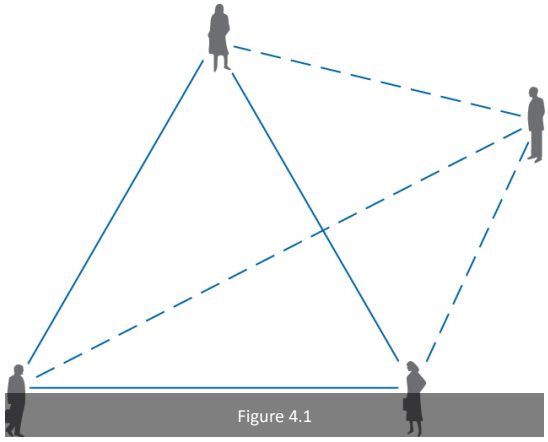


Figure 4.1

5

Communications Problems

- But other three have to explain in detail
  - What has been accomplished
  - What is still incomplete
- Brooks's Law
  - Adding additional programming personnel to a team when a product is late has the effect of making the product even later

6

Team Organization

- Teams are used throughout the software production process
  - But especially during implementation
  - Here, the discussion is presented within the context of programming teams
- Two extreme approaches to team organization
  - Democratic teams (Weinberg, 1971)
  - Chief programmer teams (Brooks, 1971; Baker, 1972)

7

4.2 Democratic Team Approach

- Basic underlying concept — *egoless programming*
- Programmers can be highly attached to their code
  - They even name their modules after themselves
  - They see their modules as extension of themselves

8

Democratic Team Approach

Proposed solution

Egoless programming

- Restructure the social environment
- Restructure programmers' values
- Encourage team members to find faults in code
- A fault must be considered a normal and accepted event
- The team as whole will develop an ethos, a group identity
- Modules will "belong" to the team as whole
- A group of up to 10 egoless programmers constitutes a *democratic team*

9

Difficulties with Democratic Team Approach

- Management may have difficulties
  - Democratic teams are hard to introduce into an undemocratic environment
- May not fit personality types

10

Strengths of Democratic Team Approach

- Democratic teams can be enormously productive
- They work best when the problem is difficult
- They function well in a research environment
- Problem:
  - Democratic teams have to spring up spontaneously

11

4.3 Classical Chief Programmer Team Approach

- Consider a 6-person team
  - Fifteen 2-person communication channels
  - The total number of 2-, 3-, 4-, 5-, and 6-person groups is 57
  - This team cannot do 6 person-months of work in 1 month

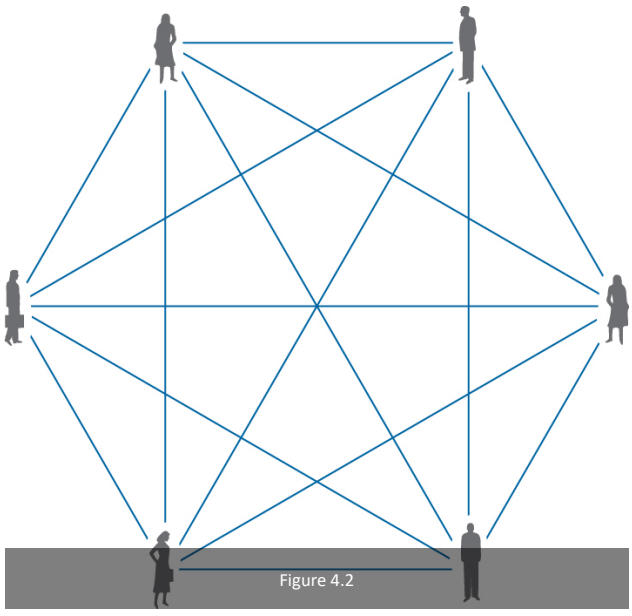


Figure 4.2

12

Classical Chief Programmer Team

Six programmers, but now only 5 lines of communication

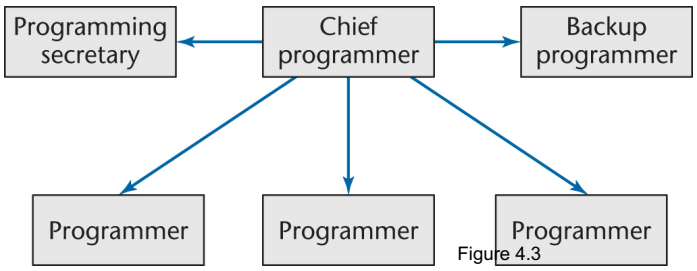


Figure 4.3

13

Classical Chief Programmer Team

- Chief programmer
  - Successful manager *and* highly skilled programmer
  - Does the architectural design
  - Allocates coding among the team members
  - Writes the critical (or complex) sections of the code
  - Handles all the interfacing issues
  - Reviews the work of the other team members
  - Is personally responsible for every line of code

14

Classical Chief Programmer Team

- Programming secretary
  - A highly skilled, well paid, central member of the chief programmer team
  - Responsible for maintaining the program production library (documentation of the project), including:
    - Source code listings
    - JCL
    - Test data
  - Programmers hand their source code to the secretary who is responsible for
    - Conversion to machine-readable form
    - Compilation, linking, loading, execution, and running test cases (this was 1971, remember!)

15

Classical Chief Programmer Team

- Programmers
  - Do nothing but program
  - All other aspects are handled by the programming secretary

16

The *New York Times* Project

- Chief programmer team concept
  - First used in 1971
  - By IBM
  - To automate the clippings data bank ("morgue") of the *New York Times*
- Chief programmer — F. Terry Baker

17

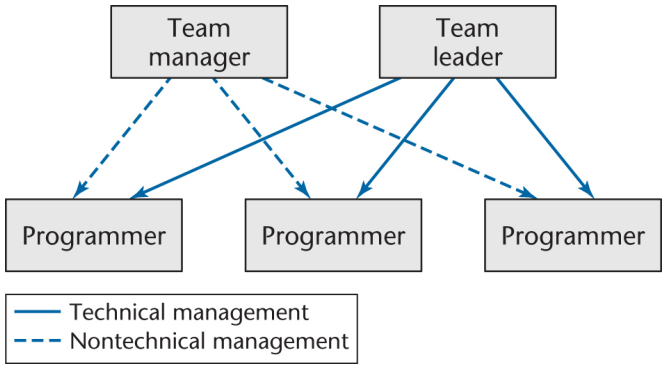
4.4 Beyond CP and Democratic Teams

- We need ways to organize teams that
  - Make use of the strengths of democratic teams and chief programmer teams, and
  - Can handle teams of 20 (or 120) programmers
- A strength of democratic teams
  - A positive attitude to finding faults

18

Beyond CP and Democratic Teams

- Solution
  - Reduce the managerial role of the chief programmer



19

Beyond CP and Democratic Teams

- It is easier to find a team leader than a chief programmer
- Each employee is responsible to exactly one manager — lines of responsibility are clearly delineated
- The team leader is responsible for only technical management

20

Beyond CP and Democratic Teams

- Budgetary and legal issues, and performance appraisal are not handled by the team leader
- The team leader participates in reviews — the team manager is not permitted to do so
- The team manager participates in regular team meetings to appraise the technical skills of the team members

21

Larger Projects

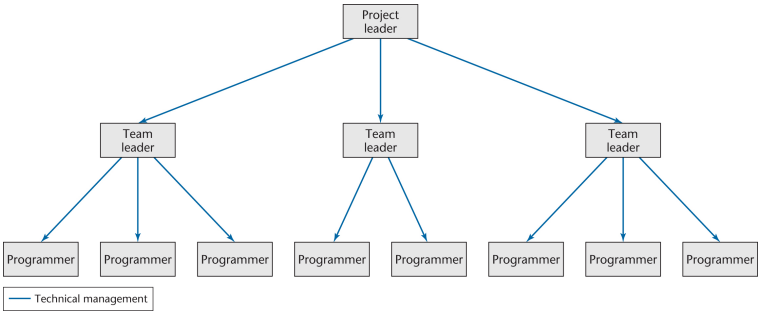


Figure 4.5

- The nontechnical side is similar
  - For even larger products, add additional layers

22

Beyond CP and Democratic Teams

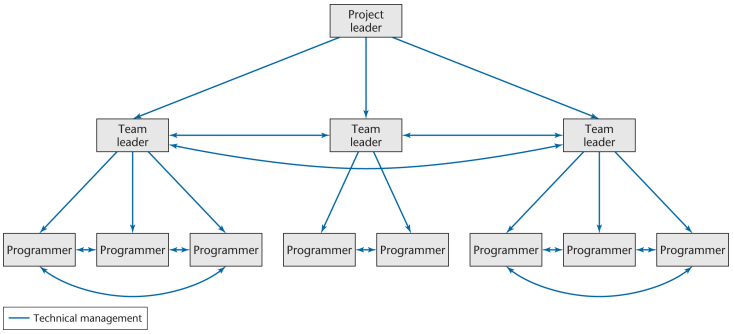


Figure 4.6

- Decentralize the decision-making process, where appropriate
  - Useful where the democratic team is good

23

4.5 Synchronize-and-Stabilize Teams

- Used by Microsoft
- Products consist of 3 or 4 sequential builds

- Small parallel teams
  - 3 to 8 developers
  - 3 to 8 testers (work one-to-one with developers)
  - The team is given the overall task specification
  - They may design the task as they wish

24

4.6 Teams For Agile Processes

- Feature of agile processes
  - All code is written by two programmers sharing a computer
  - "Pair programming"
- (This is no longer true)

25

Open-Source Programming Teams

- Individuals volunteer to take part in an open-source project for two main reasons
- Reason 1: For the sheer enjoyment of accomplishing a worthwhile task
  - In order to attract and keep volunteers, they have to view the project as "worthwhile" at all times
- Reason 2: For the learning experience

26

People Capability Maturity Model

- P–CMM is a framework for improving an organization's processes for managing and developing its workforce
- No one specific approach to team organization is put forward

27

The End!

28