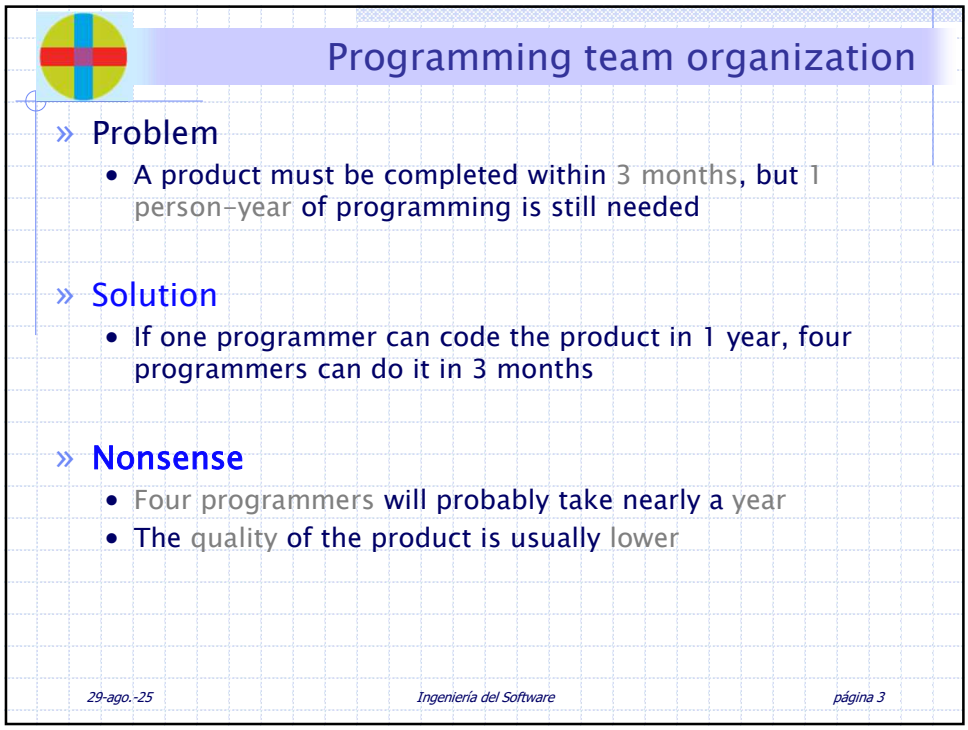

Organización de equipos de desarrollo de software
Ingeniería del Software
Curso 2025/2026
Universidad San Pablo-CEU
Escuela Politécnica Superior
Campus de Montepríncipe

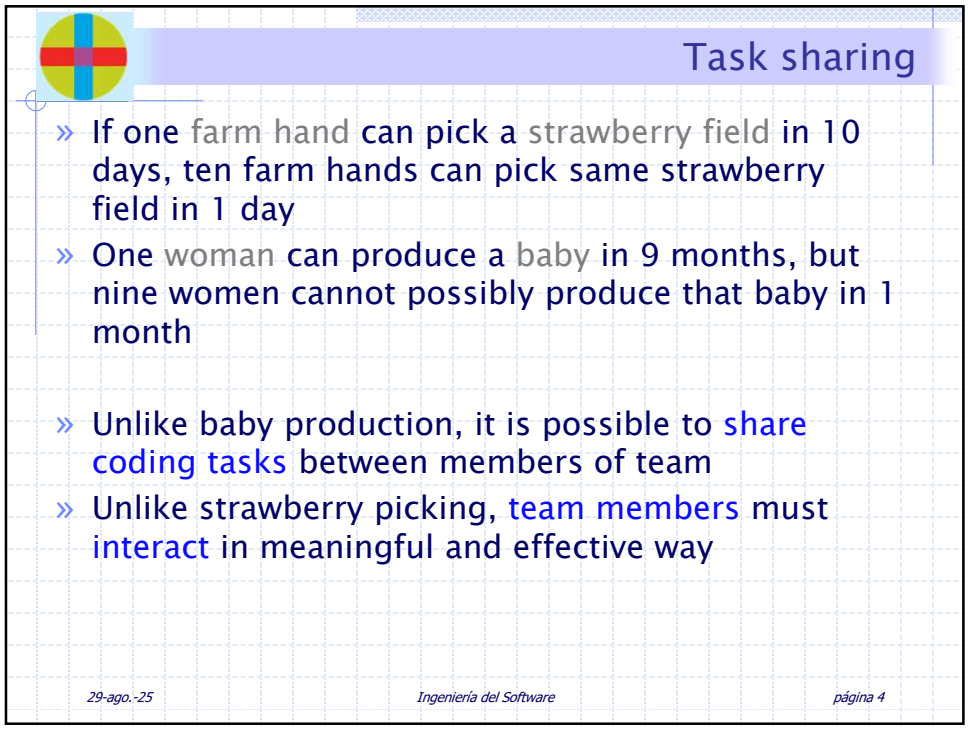
1


Overview
1. Team organization
2. Democratic team approach
3. Classical chief programmer team approach
4. Beyond chief programmer and democratic teams
5. Synchronize-and-stabilize teams
6. Extreme programming teams
29-ago.-25
Ingeniería del Software
página 2

2


Programming team organization
» Problem
• 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
29-ago.-25 Ingeniería del Software página 3

3


Task sharing
» If one farm hand can pick a strawberry field in 10 days, ten farm hands can pick same strawberry field in 1 day
» One woman can produce a baby in 9 months, but nine women cannot possibly produce that baby in 1 month
» Unlike baby production, it is possible to share coding tasks between members of team
» Unlike strawberry picking, team members must interact in meaningful and effective way
29-ago.-25 Ingeniería del Software página 4

4

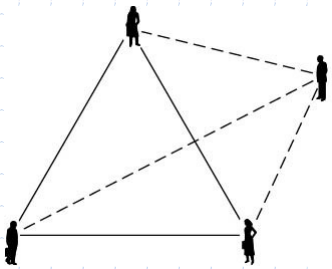
Programming team organization

- Example:
  - Freda and Joe code two modules, `mA` and `mB`, say
- What can go wrong?
  - Both Freda and Joe may code `mA`, and ignore `mB`
  - Freda may code `mA`, Joe may code `mB`. When `mA` calls `mB` it passes 4 parameters; but `mB` requires 5 parameters
  - Or, the order of parameters in `mA` and `mB` may be different
  - Or, the order may be same, but the data types may be slightly different
- This has nothing whatsoever to do with technical competency
- Team organization is a managerial issue

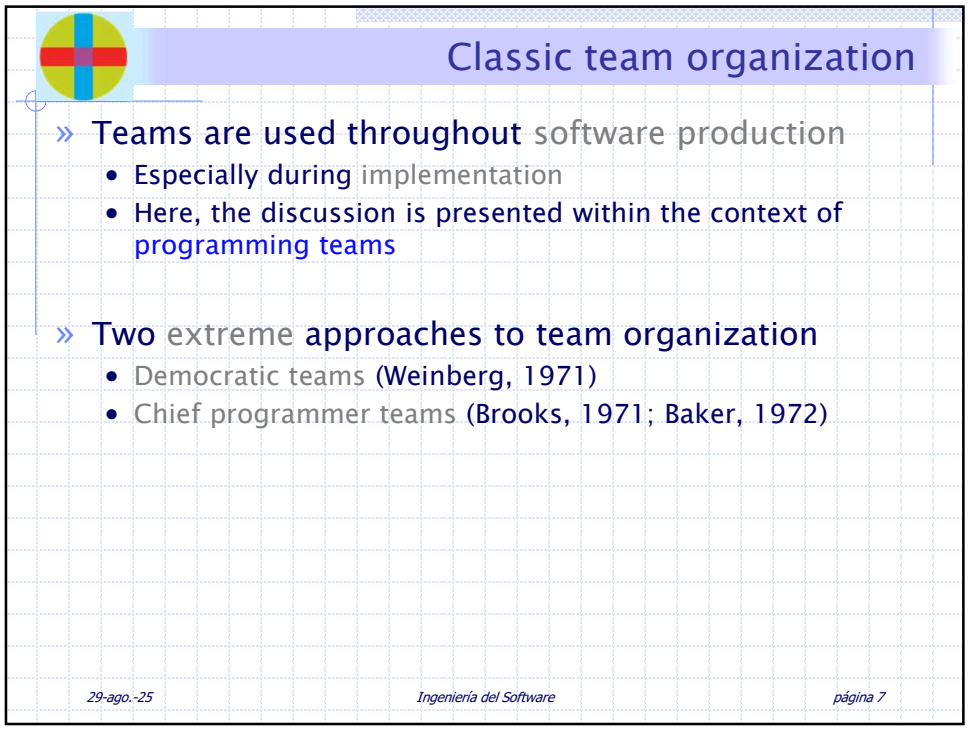
5

Communications problems

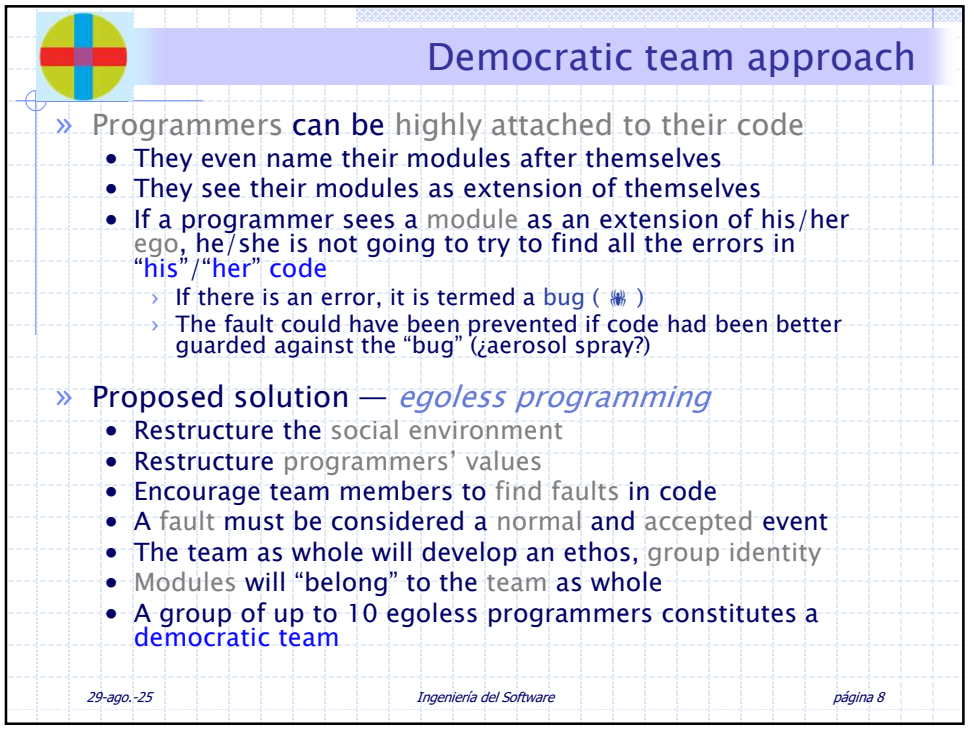
- Example
  - There are three channels of communication between 3 programmers working on project. The deadline is rapidly approaching but the code is not nearly complete
- "Obvious" solution:
  - Add a fourth programmer to the team
- But other three have to explain in detail
  - What has been accomplished
  - What is still incomplete
- Brooks's Law
  - Adding additional programming personnel to a team when product is late makes the product even **later**



6


Classic team organization
» Teams are used throughout software production
• 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)
29-ago.-25
Ingeniería del Software
página 7

7


Democratic team approach
» Programmers can be highly attached to their code
• They even name their modules after themselves
• They see their modules as extension of themselves
• If a programmer sees a module as an extension of his/her ego, he/she is not going to try to find all the errors in "his"/"her" code
› If there is an error, it is termed a bug ( )
› The fault could have been prevented if code had been better guarded against the "bug" (¿aerosol spray?)
» 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, group identity
• Modules will "belong" to the team as whole
• A group of up to 10 egoless programmers constitutes a democratic team
29-ago.-25
Ingeniería del Software
página 8

8

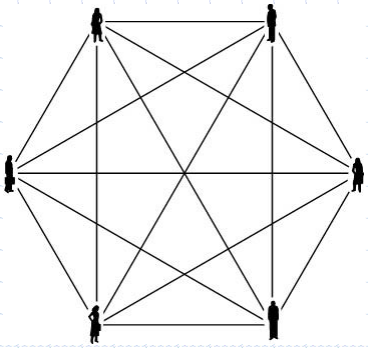
Strengths and difficulties of democratic teams

- Strengths:
  - Democratic teams are 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
    - Leadership!
- Difficulties:
  - Management may have difficulty
  - Difficult to introduce into an undemocratic environment

9

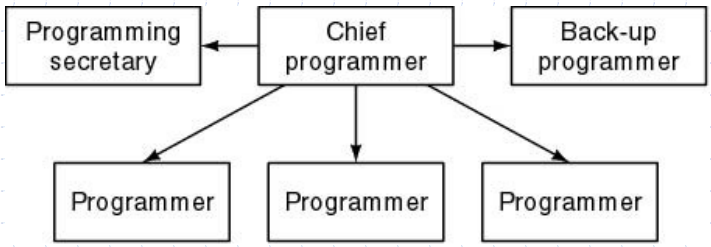
Chief programmer teams

- 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



10

Chief programmer teams (cont)



» Six programmers, but now only 5 lines of communication

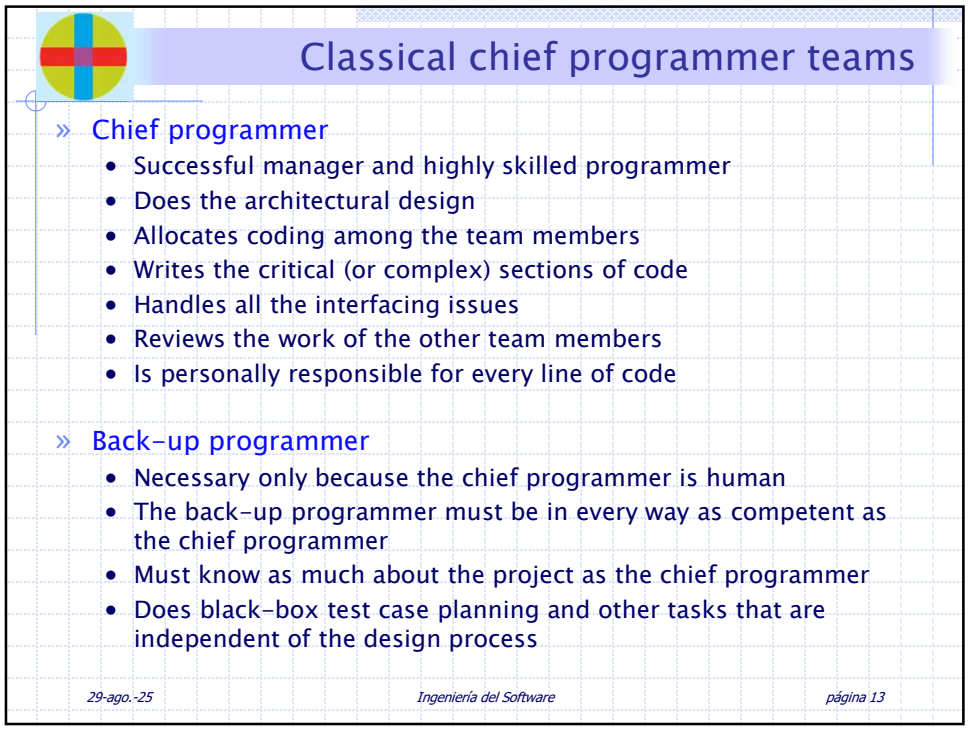
11

Classical chief programmer teams

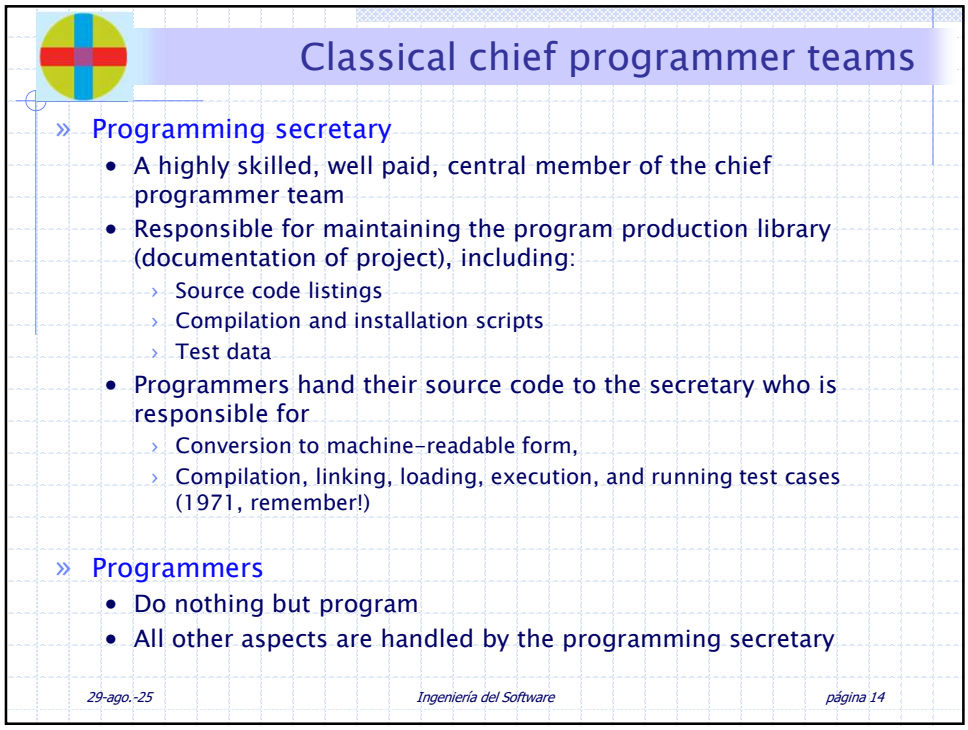
» Basic idea behind the concept
- Analogy: chief surgeon directing operation, assisted by
  - Other surgeons
  - Anesthesiologists
  - Nurses
  - Other experts, such as cardiologists, nephrologists

» Two key aspects
- Specialization
- Hierarchy

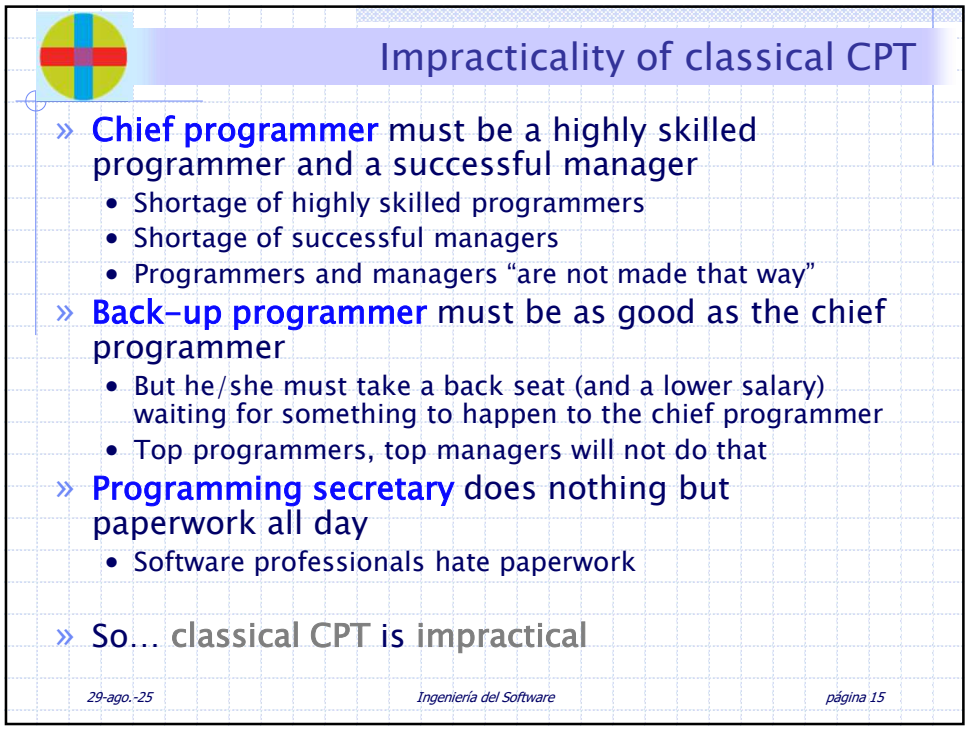
12


Classical chief programmer teams
» 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 code
• Handles all the interfacing issues
• Reviews the work of the other team members
• Is personally responsible for every line of code
» Back-up programmer
• Necessary only because the chief programmer is human
• The back-up programmer must be in every way as competent as the chief programmer
• Must know as much about the project as the chief programmer
• Does black-box test case planning and other tasks that are independent of the design process
29-ago.-25 Ingeniería del Software página 13

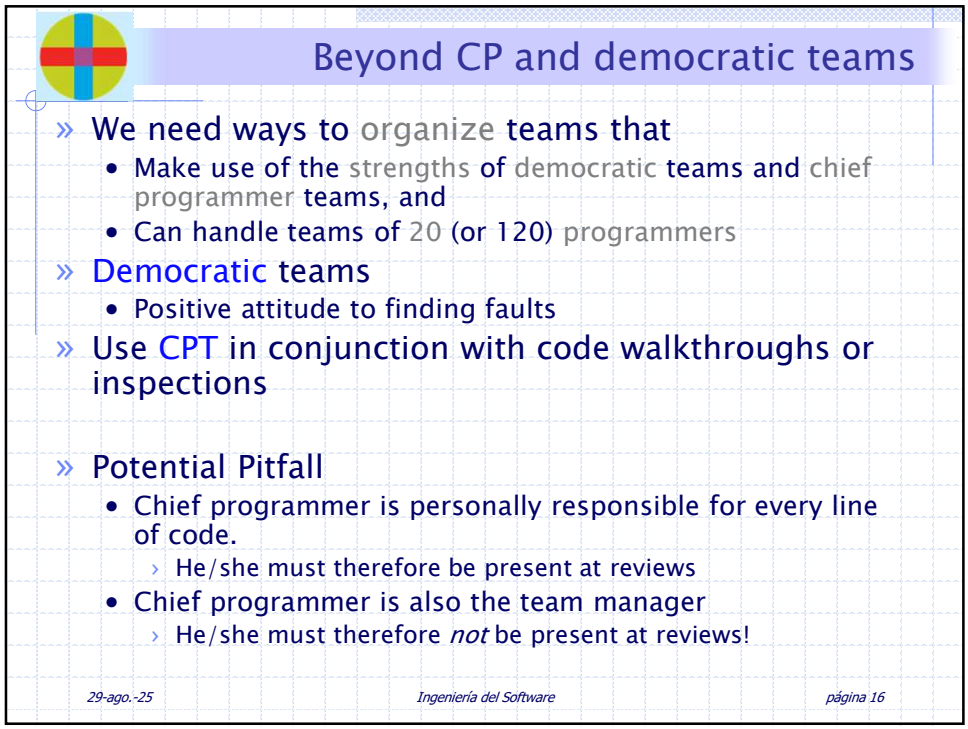
13


Classical chief programmer teams
» Programming secretary
• A highly skilled, well paid, central member of the chief programmer team
• Responsible for maintaining the program production library (documentation of project), including:
› Source code listings
› Compilation and installation scripts
› 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 (1971, remember!)
» Programmers
• Do nothing but program
• All other aspects are handled by the programming secretary
29-ago.-25 Ingeniería del Software página 14

14

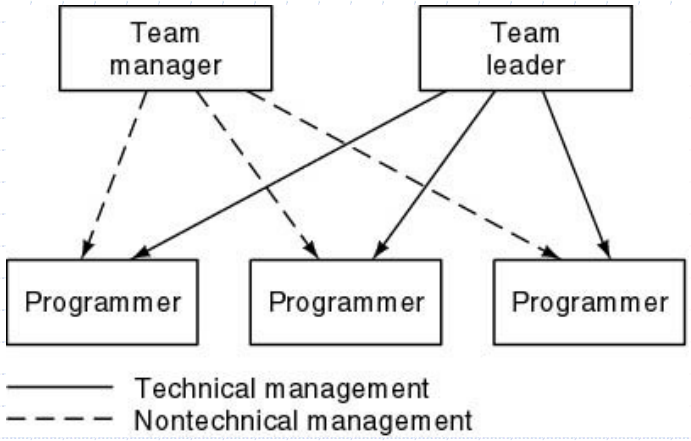

Impracticality of classical CPT
» Chief programmer must be a highly skilled programmer and a successful manager
• Shortage of highly skilled programmers
• Shortage of successful managers
• Programmers and managers "are not made that way"
» Back-up programmer must be as good as the chief programmer
• But he/she must take a back seat (and a lower salary) waiting for something to happen to the chief programmer
• Top programmers, top managers will not do that
» Programming secretary does nothing but paperwork all day
• Software professionals hate paperwork
» So… classical CPT is impractical
29-ago.-25 Ingeniería del Software página 15

15


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
» Democratic teams
• Positive attitude to finding faults
» Use CPT in conjunction with code walkthroughs or inspections
» Potential Pitfall
• Chief programmer is personally responsible for every line of code.
› He/she must therefore be present at reviews
• Chief programmer is also the team manager
› He/she must therefore *not* be present at reviews!
29-ago.-25 Ingeniería del Software página 16

16

Beyond CP and democratic teams



- Solution
  - Reduce the managerial role of the chief programmer

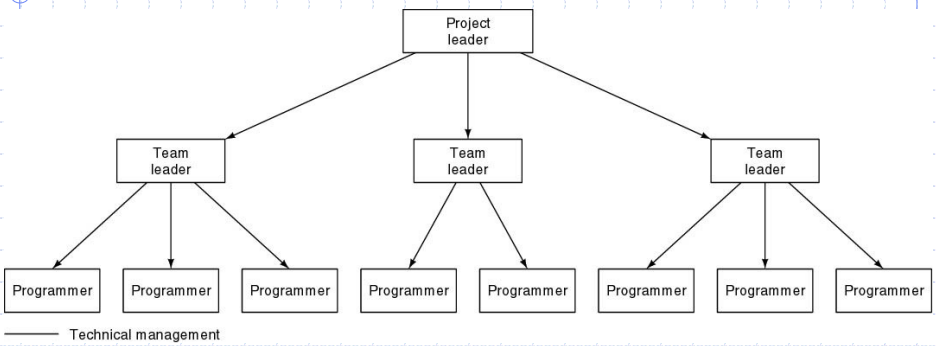
17

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

18

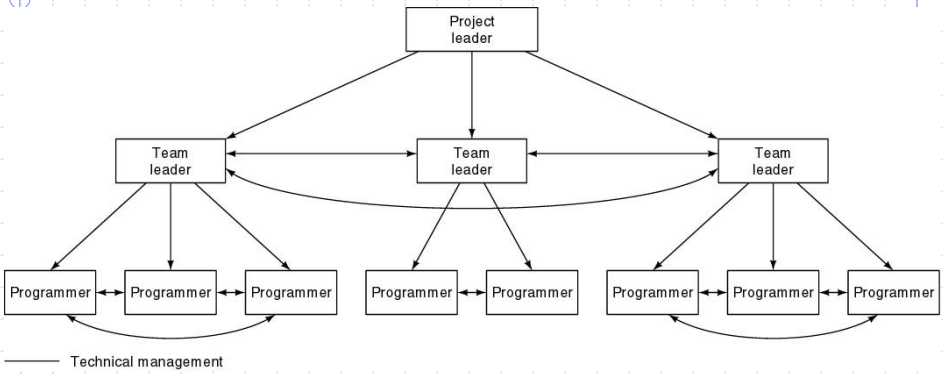
Larger projects



- Non-technical side is similar
- For even larger products, add additional layers

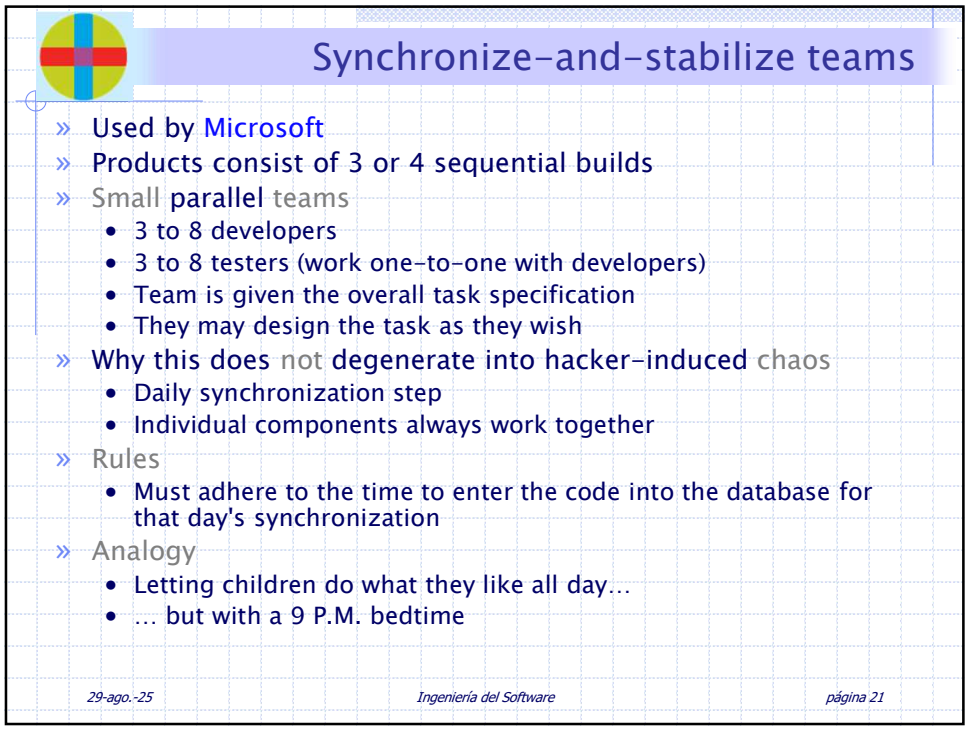
19

Larger projects

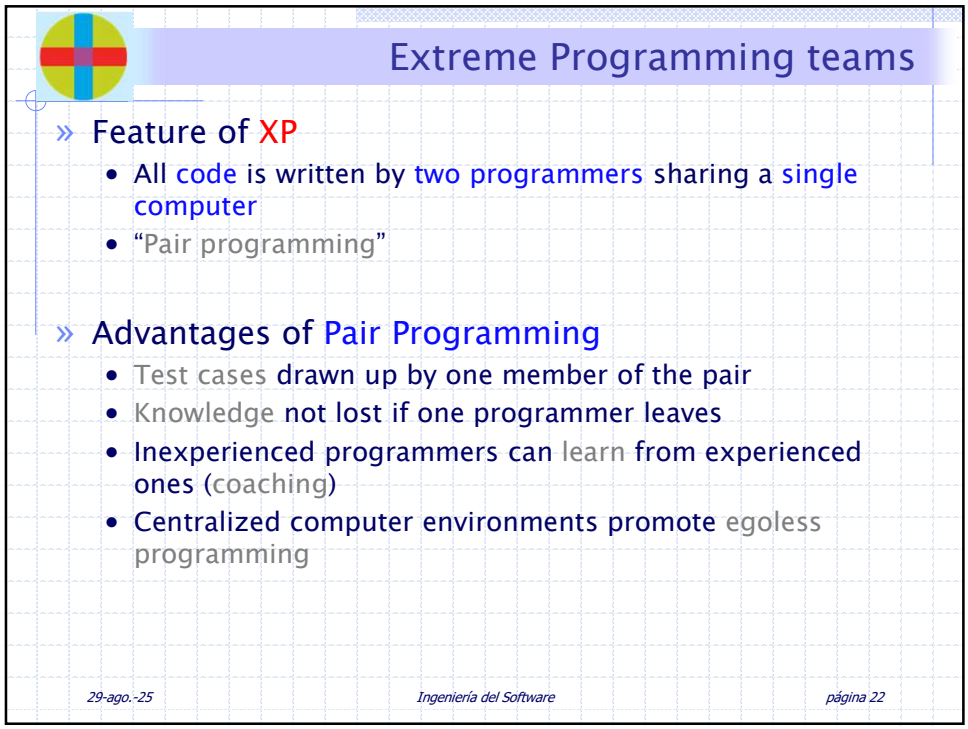


- Decentralize the decision-making process where appropriate
- Useful where the democratic team is good

20


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)
• Team is given the overall task specification
• They may design the task as they wish
» Why this does not degenerate into hacker-induced chaos
• Daily synchronization step
• Individual components always work together
» Rules
• Must adhere to the time to enter the code into the database for that day's synchronization
» Analogy
• Letting children do what they like all day…
• … but with a 9 P.M. bedtime
29-ago.-25 Ingeniería del Software página 21

21


Extreme Programming teams
» Feature of XP
• All code is written by two programmers sharing a single computer
• "Pair programming"
» Advantages of Pair Programming
• Test cases drawn up by one member of the pair
• Knowledge not lost if one programmer leaves
• Inexperienced programmers can learn from experienced ones (coaching)
• Centralized computer environments promote egoless programming
29-ago.-25 Ingeniería del Software página 22

22

Final remarks

- There is no one solution to the problem of team organization
- The "correct" way depends on
  - The product
  - The outlook of the leaders of the organization
  - Previous experience with various team structures
- Very little research has been done on software team organization
  - Instead, team organization has been based on research on group dynamics in general
- Without *relevant* experimental results, it is hard to determine optimal team organization for a specific product

¿Preguntas?