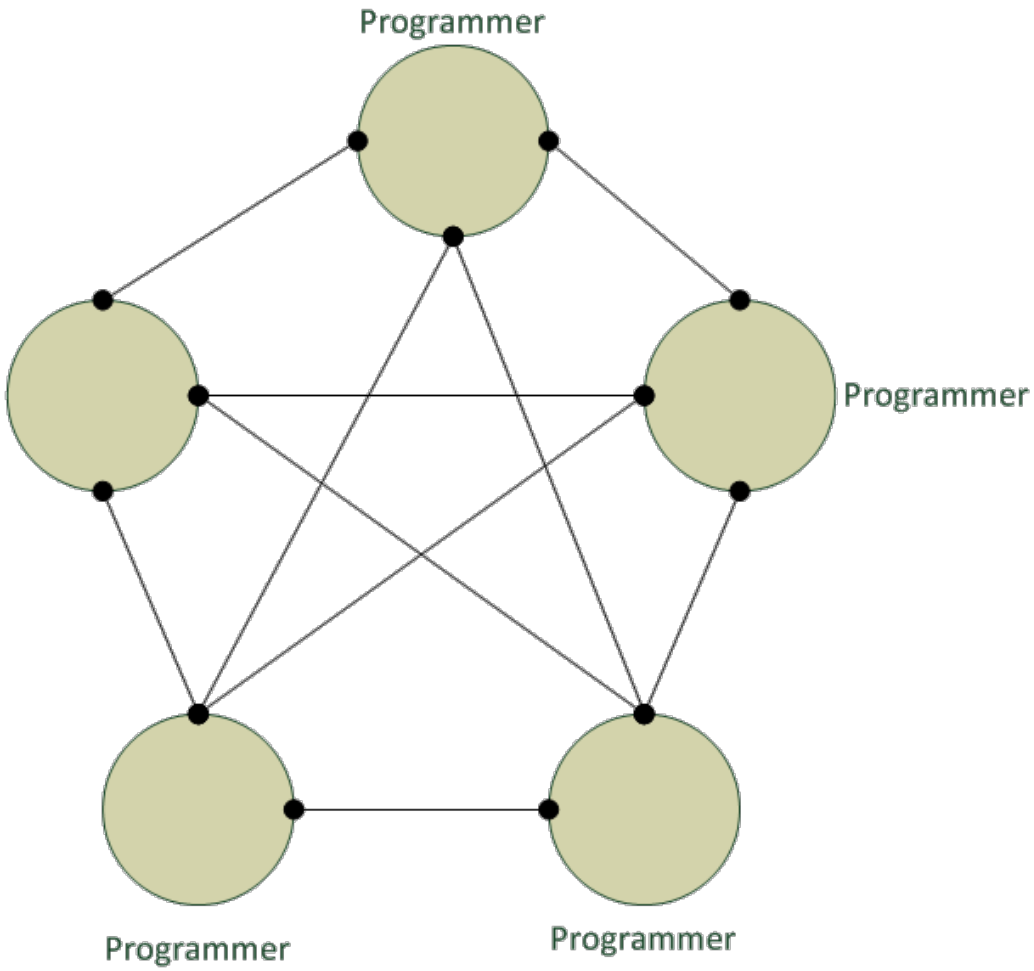
CS 315 - Lecture 4 - Aug 31, 2015

Chapter 4: Software Teams

- Understand what the Client needs
  - Example
    - A product must be completed within 3 months, but 1 person-year of programing 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
- Software Teams
  - 3 programmers, deadline is rapidly approaching and code is still incomplete
  - 3 channels of communication between 3 programmers
  - Solution: Add a fourth programmer!
  - But other 3 have to explain in detail:
    - What has been done?
    - What is left to do?
  - Now 6 channels of communication
    - `(n(n-1))/2`
  - Brooks' Law
    - "Adding manpower to a late software project makes it later"
    - The Mythical Man-Month (Fred Brooks, 1975)
- Democratic Team Approach

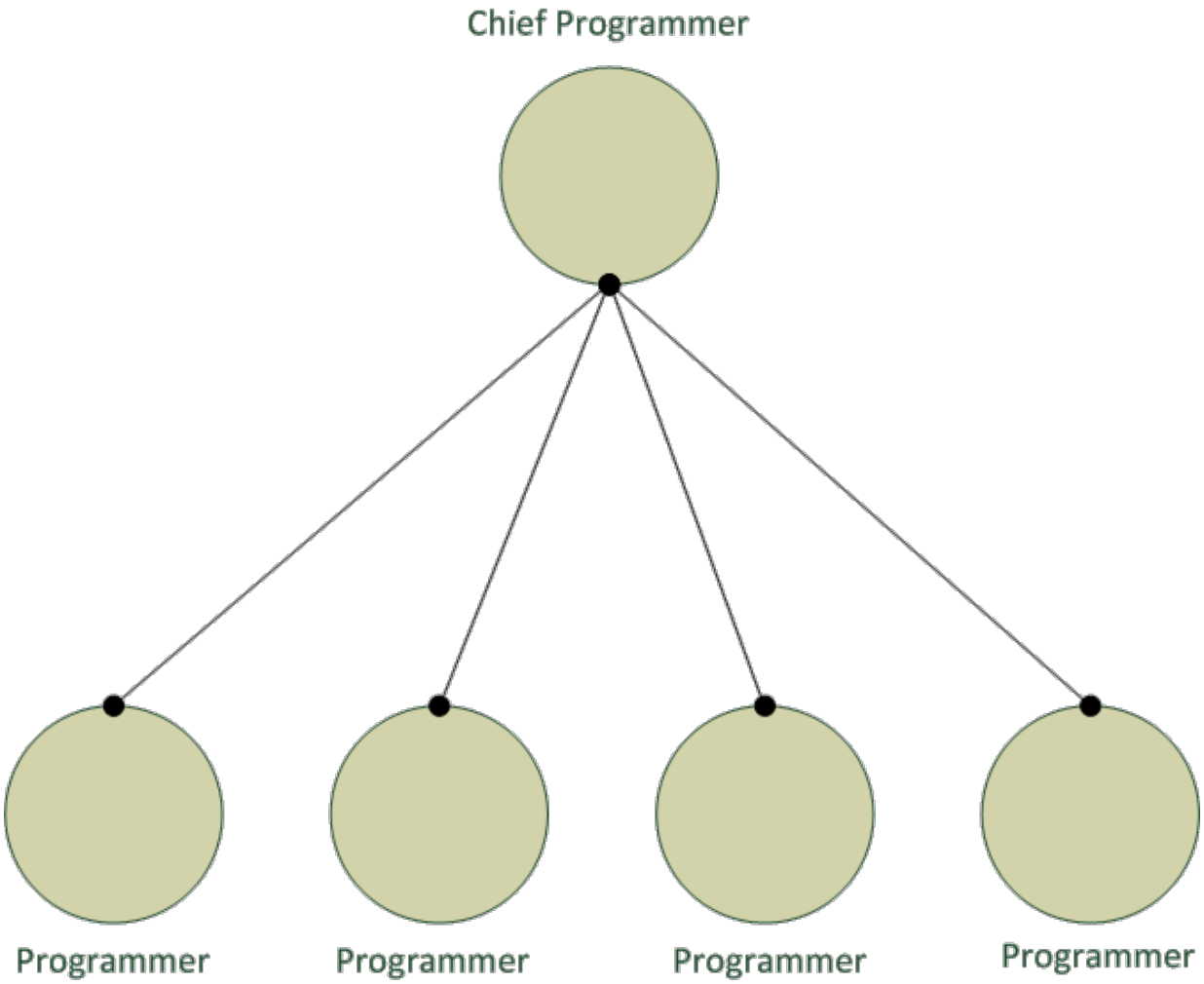
Ego-less team (aka democratic team)



- Basic underlying concept - egoless programming
- Programming can be highly attached to their code
  - They even name their modules after themselves
  - they see their modules as extension of themselves
  - Encourages team members to find faults in code
  - A fault must be considered a normal and accepted event
- Democratic teams are enormously productive
- They work best when the problem is difficult
- They function well in a function in a research environment
- Management may have difficulties
  - Democratic teams are hard to introduce into an undemocratic environment
  - Who's in charge?
  - Who keeps the team on schedule?

- Classical Chief Programmer Team

Chief-programmer team



- The basic idea behind the concept
  - Analogy: chief surgeon directing an operation, assisted by:
    - Other surgeons
    - Anesthesiologists
    - Nurses
    - Other experts, such as cardiologist, nephrologists
  - Two key aspects
    - Specialization
    - Hierarchy
- Impracticality of Classical CPT
  - The chief programmer must be a highly skilled programmer *and* a successful manager
  - There is a shortage of successful managers
  - The qualities needed to be a highly successful skilled programmer are unlikely to be

found in a successful manager, and vice versa
        - The *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 not
        - the *programming secretary* does nothing but paperwork all day
            - Software professionals hate paperwork
        - **Classical CPT is impractical**

- Beyond CP and Democratic Teams
    - Organization that combines democratic and chief programmer
    - Models
        - Positive attitude
        - Reduce managerial role of chief programmer
    - Large Projects
        - The nontechnical side is similar
            - For even larger projects, add additional layers
    - Decentralize the decision making process, where appropriate
        - Useful where the democratic team is good
    - Table of examples on page 120 of the book