

SE/16

Software Engineering

Teams
James Gain
(jgain@cs.uct.ac.za)
<http://people.cs.uct.ac.za/~jgain/courses/SoftEng/>

SE/16

Objectives

1. Point out Common Problems with Team Structures in Software Engineering
2. Describe a variety of team structures:
 - Democratic
 - Chief Programmer
 - Modified Chief Programmer
 - Synchronize and Stabilize
 - Extreme Programming
3. Present a categorisation of teams

SE/16

Teamwork Problems

- Most software is too large or complex to be developed by an individual → a team is born!
- The Nature of Tasks:
 1. Shared e.g. strawberry picking
 - Can be done in parallel. Four people on the job gets it done in 1/4 the time
 2. Individual, e.g. pregnancy
 - Can only be done in serial. Nine women cannot produce a baby in one month
 3. Combination, e.g. software implementation
 - Coding can be done separately but requires communication for integration

SE/16

Communication Explosion

- Assume everyone in a team of n people communicates with everyone else
- Number of communication paths = $0.5 \times n \times (n-1)$

SE/16

Democratic Teams

- Requires Egoless Programming:
 - Coders not too attached to their code
 - Needs an atmosphere of cooperation
 - Bug finding is given a positive spin
- 10 Egoless Programmers = A Democratic Teams
- Everyone is equal and the team is self-organizing
- There is no appointed leader → Goes against conventional management wisdom
- Group theory suggests it will work well for complex problems

SE/16

Chief Programmer Teams

- Modelled on Surgical Teams: A hierarchy is used to overcome communication explosion, specialization to improve productivity
- A chief programmer teams of 6 people reduces communication links from 15 to 5
- Roles:
 - Chief Programmer: designs the architecture; codes complex classes; overall manager
 - Back-Up Programmer: takes over if needed; designs tests
 - Librarian: maintains code base and documentation; runs tests

SEMS

Chief Programmer Track Record

- Results from Initial Test Project were extraordinary:
 - New York Times automated clipping file
 - 83,000 LOC in 22 months (=11 person-years)
 - Half the modules (200-400 LOC each) were correct on first compilation
 - File Maintenance component operated 20 months without a single fault
- But no comparable wild success stories since
- Because the Chief Programmer (Terry Baker) was one of a breed of super-programmers

SEMS

Problems with the Chief Programmer

- Good Chief Programmers are rare:
 - Intersection set of highly skilled programmers and successful managers
- Backup Programmers are even more rare:
 - As capable as the Chief Programmer but prepared to accept a lower salary and subordinate position
- Librarians are also difficult to find:
 - Who wants to do nothing but paperwork all day
- Also Doesn't Scale Well

SEMS

Modified Chief Programmer Teams

- Split Chief Programmer into:
 - Team Leader (technical issues)
 - Team Manager (non-technical management issues)
- Carefully separate responsibilities to avoid conflicts of interest
- But there may be some areas of overlap (e.g. annual leave)
- Can be scaled with layers of leaders

```

graph TD
    TM[Team Manager] -.-> P1[Programmer]
    TM -.-> P2[Programmer]
    TM -.-> P3[Programmer]
    TL[Team Leader] --> P1
    TL --> P2
    TL --> P3
  
```

SEMS

Synchronize and Stabilize Teams

- Microsoft model; successful for very large projects
 - More than 3000 .dev and .test worked on Windows 2000
- Small parallel teams of 3-8 developers and 3-8 testers
- Individuals allowed latitude to design and implement a spec but:
 - Code must be integrated on a daily basis
 - If your code prevents compilation then it must be fixed immediately
- But Microsoft is more than Synchronize and Stabilize. Their success is based on a strong corporate culture

SEMS

Extreme Programming Teams

- Based on Pair Programming:
 - Spreads knowledge
 - Brings less experienced programmers up to speed
- Evidence in Favour:
 - From Williams *et al.* "Strengthening the Case for Pair Programming" *IEEE Software*
 - Subject were students at the University of Utah doing an Advanced Programming course
 - By the end pairs took 60% of the time to do the same programming task as individuals BUT
 - Passed 94% of test cases rather than 78%
 - Pair programming improved job satisfaction and overall confidence

SEMS

Exercise: Choosing a Team Structure

- Problem: As a project manager you must choose a team structure for the following projects:
 1. A quantum computing project with 5 researchers. There is no strict deadline. If successful this team will remain intact for later projects
 2. A payroll system for a mining company. This has a work estimate of 30 person-years and must be delivered according to a very strict 10 month deadline
- Solution:
 1. Small size, high difficulty, long team lifetime = Democratic or Extreme Programming
 2. Medium size, low difficulty, strict delivery date = Modified Chief Programmer

SEMS **A Categorisation of Team Structures**

- **Democratic Decentralized (DD):**
 - No permanent leader; decisions are made by group consensus
 - Communication and control are horizontal
- **Controlled Decentralized (CD):**
 - A leader coordinates tasks; Problem solving remains a group activity
 - Communication is horizontal and control is vertical
- **Controlled Centralized (CC):**
 - A leader coordinates tasks and solves problems
 - Communication and control are vertical

SEMS **Exercise: Classifying Teams**

- **Problem:**
 - Categorise the Democratic, Classical and Modified Chief Programmer, Synchronize and Stabilize, and XP team structures as Democratic Decentralized (DD), Controlled Decentralized (CD) or Controlled Centralized (CC)
- **Solution:**
 - Democratic = DD
 - Classical and Modified Chief Programmer = CC
 - Synchronize and Stabilize = CD
 - XP = DD

SEMS **Jelling**

- In business groups are assigned to work together; often without team spirit
- An effective tightly knit group displays Jell or “Esprit de Corps”. *“Once a team begins to jell, the probability of success goes way up. The team can become unstoppable, a juggernaut for success”.*
- **Team Toxicity (factors that work against jelling):**
 1. A frenzied work atmosphere (which wastes energy and lacks focus)
 2. High frustration caused by personal, business, or technological factors that cause friction among team members
 3. Fragmented or poorly coordinated procedures
 4. Unclear definition of roles resulting in a lack of accountability
 5. Morale damaged by continuous and repeated failure

SEMS **A Comparison of Teams**

Organization	Strengths	Weaknesses
Democratic	Collective code ownership Handles hard problems	Cannot be externally imposed May not scale
Classical Chief Programmer	Miraculous success with New York Times	Impractical
Modified Chief Programmer	Scales well Handles strict delivery	No successes comparable to New York Times
Synchronize and Stabilize	Encourages creativity Scales very well	Is it effective outside Microsoft?
Extreme Programming	Many - sharing info, group ownership of code, improved quality	Not fully proven

This document was created with Win2PDF available at <http://www.daneprairie.com>.
The unregistered version of Win2PDF is for evaluation or non-commercial use only.