



Lecture 15: Managing People

Organizational Structures

Building high Performance teams

Dealing with problems with team assignments



Starting point

You have a project

You have been given a team

a mixed set of skills

a mixed set of motivations

Problem:

How do you get everyone to work together?

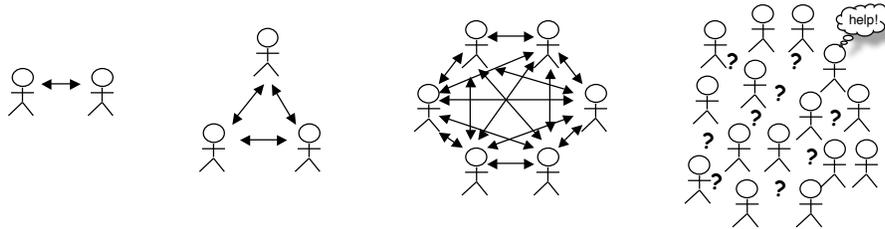
...and get the job done?





Scaling up...

Communication overhead is exponential



Exploit Modularity:

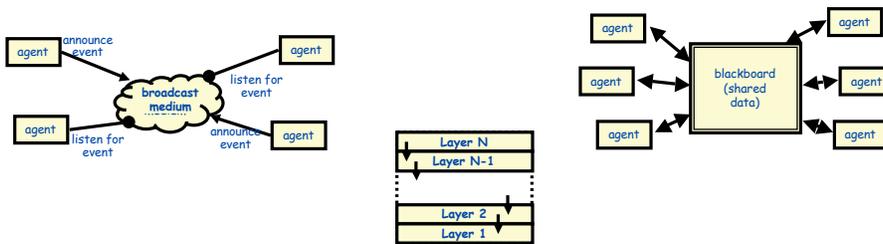


Team Organization



Conway's Law:

The structure of the software reflects the structure of the organisation that built it





Coordination Mechanisms

Direct supervision

simple structure - little formalization

Standardization of work processes

“machine bureaucracy” e.g. mass production and assembly

Standardization of work outputs

“divisionalized form” e.g. each division has performance targets

Standardization of worker skills

“professional bureaucracy” e.g. hospitals, law firms,...

Mutual adjustment

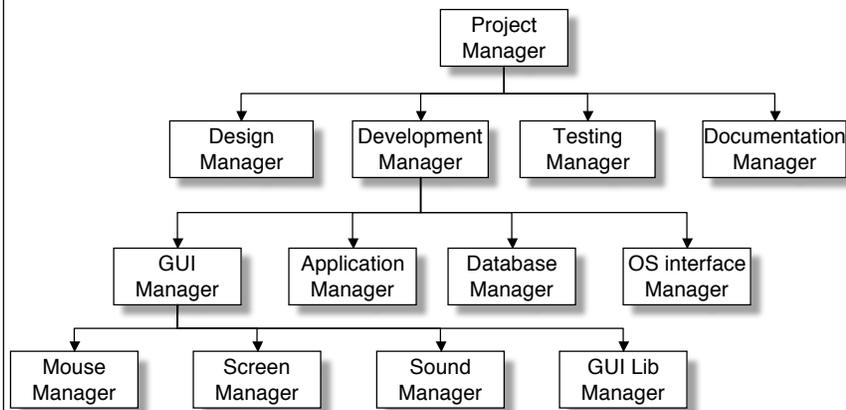
“adhocracy” e.g. skunkworks, high innovation



Hierarchical Teams

Team structure matches structure of the software

Disadvantage: vertical communication is ineffective

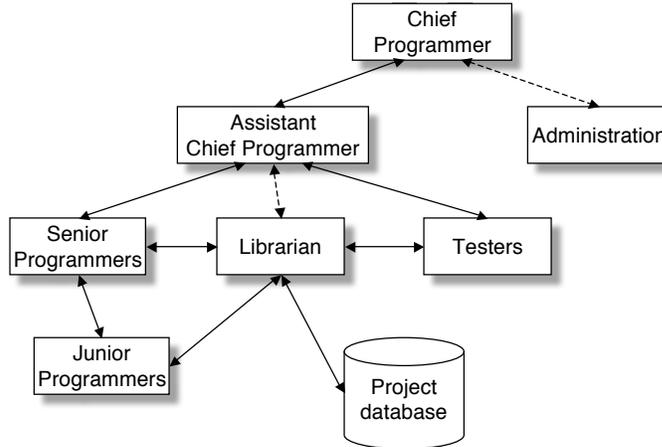




Chief Programmer Teams

Based on hospital surgical teams

Chief programmer is not a manager - concentrates on technical issues



Matrix Organization

Identify specific skill sets

Assign people to projects according to needed skills

People work on multiple projects

	real-time programming	graphics	data-bases	QA	Testing
Project 1	X			X	X
Project 2	X		X	X	X
Project 3		X	X	X	X



General Principles

Use fewer, better people

Performance of best programmers better by an order of magnitude!

Fit tasks to capabilities and motivations of people

Help people to get the most out of themselves

opportunity to accept new challenges and be rewarded

Balance the team

E.g. team players vs. star performers

Practice “egoless” programming

Remove people who do not fit the team



High Performance Teams

Nurture a team culture

A team is not a family

Team members help one another, but don't tolerate freeloaders

Instill the right values

Discuss examples

Reward people who uphold the team values

Build trust

all feedback is constructive

lively & healthy debate about issues and risks

Effective Communication

Use face-to-face whenever possible

Use phone or F2F to resolve email debates

Get everyone using IM

Encourage social events for the team

Physical layout of office space is important





Organizational Clarity

Stuff every team member should know:

- What is the mission of the team?
- What is the vision for the system to be delivered?
- How will you measure team success?
- Who are the project stakeholders?
- How will you measure project success?
- Who is responsible for what?
- What procedures should you follow to do the work?



Who can change the code?

Collective Ownership

- Anyone can change any code or model
- Works well for small teams
- Promotes shared responsibility
- (Needs good version management tools)

Change Control

- Each sub-team can only change their subsystem
- Reduces unexpected problems when code changed by others
- Promotes development of expertise
- More important on larger projects





Poisonous People: Anna

Anna knows more about every subject than everyone else on the team put together---at least, she thinks she does. No matter what you say, she'll correct you; no matter what you know, she knows better. Annas are pretty easy to spot: if you keep track in team meetings of how often people interrupt one another, her score is usually higher than everyone else's put together.



Poisonous People: Bao

Bao is a contrarian: no matter what anyone says, he'll take the opposite side. This is healthy in small doses, but when Bao does it, there's always another objection lurking behind the first half dozen.





Poisonous People: Caitlin

Caitlin has so little confidence in her own ability (despite her good grades) that she won't make any decision, no matter how small, until she has checked with someone else. Everything has to be spelled out in detail for her so that there's no possibility of her getting anything wrong.



Poisonous People: Frank

Frank believes that knowledge is power. He enjoys knowing things that other people don't---or to be more accurate, he enjoys it when people know he knows things they don't. Frank can actually make things work, but when asked how he did it, he'll grin and say, "Oh, I'm sure you can figure it out."





Poisonous People: Hadiyah

Hadiyah is quiet. Very quiet. She never speaks up in meetings, even when she knows that what other people are saying is wrong. She might contribute to the mailing list, but she's very sensitive to criticism, and will always back down rather than defending her point of view. Hadiyah isn't a troublemaker, but rather a lost opportunity.



Poisonous People: Kenny

Kenny is a hitchhiker. He has discovered that most people would rather shoulder some extra work than snitch, and he takes advantage of it at every turn. The frustrating thing is that he's so damn plausible when someone finally does confront him. "There have been mistakes on all sides," he says, or, "Well, I think you're nit-picking." The only way to deal with Kenny is to stand up to him: remember, if he's not doing his share, he's the bad guy, not you.





Poisonous People: Melissa

Melissa would easily have made the varsity procrastination team if she'd bothered to show up to tryouts. She means well---she really does feel bad about letting people down---but somehow something always comes up, and her tasks are never finished until the last possible moment. Of course, that means that everyone who is depending on her can't do their work until after the last possible moment...



Poisonous People: Petra

Petra's favorite phrase is "why don't we". Why don't we write a GUI to help people edit the program's configuration files? Hey, why don't we invent our own little language for designing GUIs? Her energy and enthusiasm are hard to argue with, but argue you must. Otherwise, for every step you move forward, the project's goalposts will recede by two. This is called feature creep, and has ruined many projects that might otherwise have delivered something small, but useful.





Poisonous People: Raj

Raj is rude. "It's just the way I talk," he says, "If you can't hack it, maybe you should find another team." His favorite phrase is, "That's stupid," and he uses obscenity as casually as minor characters in Tarantino films. His only redeeming grace is that he can't dissemble in front of the instructor as well as Kenny, so he's easier to get rid of.



Poisonous People: Sergei

Sergei is simply incompetent. He doesn't understand the problem, he hasn't bothered to master the tools and libraries he's supposed to be using, the code he checks in doesn't compile, and his thirty-second bug fixes introduce more problems than they solve. If he means well, try to re-partition the work so that he'll do less damage. If he doesn't, he should be treated like any other hitchhiker.

