Human Aspects of Software Engineering

Overview

• Traits of successful software engineers
• Attributes of effective software teams
• Team organization
Traits of Successful Software Engineers

• Sense of individual responsibility
  – Do what needs to be done in an overriding effort to achieve a successful outcome

• Aware of the needs of stakeholders
  – Observe the environment in which people work and adapt his/her behavior

• Brutally honest about design flaws and offers constructive criticism
  – Be realistic and truthful

Traits of Successful Software Engineers

• Resilient under pressure
  – Manage the pressure/chaos which comes in many forms: changing requirements, demanding stakeholders, unrealistic manager

• Heightened sense of fairness
  – Share the credit with colleagues, don’t sabotage the work of others
Traits of Successful Software Engineers

• Attention to details
  – Consider the technical decisions against broader criteria

• Pragmatic
  – SE is a discipline to be adapted based on circumstances

Attributes of Effective Software Team

• Sense of purpose
  – Everybody agrees on the goal

• Sense of involvement
  – Everybody feels that his/her skill set and contributions are valued

• Sense of trust
  – Everybody should trust the skills and competence of their peers and their managers
Attributes of Effective Software Team

• Sense of improvement
  – Periodically reflect on the approach to think about ways for improvement

• Diversity of team member skill sets
  – Highly skilled technologists are complemented by members with less technical background but are more empathetic to needs of stakeholders

Why Do We Need a Team?

• SW too big or too complex to be constructed by a single person
“Not every group is a team, and not every team is effective”
--Glenn Parker

- The effectiveness is affected by:
  - Strengths and weaknesses of team members
  - Team Organization

Communication is the key

- IBM 1978 programmer study: 50% time interacting, 30% working alone; 20% non-productive activity
  - Often informal consultation (e.g. at the coffee machine)
- Diminishing returns of adding team members: team of 3 can do twice the work of 1 (because of communication needed)
Mold a Team out of Disparate Individuals

• Identify project goals early and get consensus from all members
• Encourage effective communication
  – Regular meetings to assess progress
  – Shared resources (e.g., CVS repository, tests)
• Recognize expertise of individuals and plan assignments accordingly

Egoless Programming

• Definition
  – Style of group working where designs, programs, and other documents are considered to be the common property of the group, rather than the individual who developed them [SOM]
Key Points

• Programmers do not regard programs as an extension of themselves
  – Otherwise, finding errors will be impossible
• Group members would like to accept criticism and work together to improve software
• Shared responsibility leads to cohesiveness
  – Make sure group loyalty does not prevent criticism
• Encourage uninhibited discussion and more co-operation between group members

How to achieve egoless pgmg?

• Hold regular code walk-throughs by peers for each team members
  – Programmer has to explain code to other people, such as designer, tester, to detect errors, without correcting them
  – Proven bug-finding technique in industry; takes time but finds bugs!
  – Insures that >1 person understands the code
  – Increases sense of joint ownership of the code
Traits of a Successful Team Leader

• Have the ability to influence people
  – Deal with problems effectively
  – Make decisions when consensus fails
  – Assess team members’ progress
• Two Types
  – Task specialist - allocates, co-ordinates work (Technical Lead)
  – Maintenance specialist - Irons out conflicts between people (Project Manager)

Team Models

• Authoritarian
  – More experienced, better influencer of others, better organized, a better programmer or designer
• Democratic
  – Leadership moves about the group, depending on current needs; most knowledgeable person acts as leader
Possible Team Crises

• Examples
  – Team member leaves
  – Team member laziness (or incompetence)
  – Team member is anti-social
• Make sure you are going to ‘stick it out in CS3704’ before constructing a team!
• Crisis provokers:
  – Machine problems, unyielding bugs, schedule or requirements changes, difficulties in integration test, ...

Avoid Team “Toxicity"

• A frenzied work atmosphere
  – Have access to all information required to do the job
  – Major goals and objectives, once defined, should not be modified unless absolutely necessary
Avoid Team “Toxicity”

- High frustration that causes friction among team members
  - A team is given as much responsibility for decision making as possible

Avoid Team “Toxicity”

- Fragmented or poorly coordinated procedures
  - Avoid the inappropriate process by understanding the product to be built, the people doing the work, and by allowing the team to select the process model
Avoid Team “Toxicity”

• Unclear definition of roles
  – Technical reviews and define a series of corrective approaches when a member fails to perform

• Continuous and repeated exposure to failure
  – Establish team-based techniques for feedback and problem solving