Team Structure and Processes

- Productivity
  - group and individual productivity
  - contributing factors, development
- Collaboration
  - practices, Pair Programming
- Teams & Projects
  - training & mentoring
  - team structure & leadership
  - distributed development issues

Contributors to Productivity

- more than a 10x difference among programmers with similar tenure
- there are many contributing factors
  - familiarity with domain and tools
  - range of experience (not years)
  - innate programming and debugging ability
  - motivation, initiative and enthusiasm
- raw speed often falls with experience
  - but quality rises, improving productivity

Individual Productivity

Was uns nicht umbringt
macht uns nur stärker.

Friedrich Wilhelm Nietzsche
Mensch und Übermensch
Good Collaboration

• prevents mistakes
  – a second set of eyes finds many problems
  • duplicated or unclear code
  • unwarranted assumptions
• helps us out of problems
  – give us options when we become blocked
• improves project understanding
  – learn how other parts of project work
• improves skill dissemination
  – we can learn others’ skills and techniques

many ways to share the work

• collaborative design
  – working together at the white-board
  – A challenges, B defends
  – A draws/types, B suggests, enumerates
• cooperative coding
  – code different parts of a collaborative design
  – A codes functionality, B codes test cases
• Pair Programming
  – A types, B reviews, challenges, suggests

Pair Programming

• not a review, but a development practice
  – does not eliminate need for reviews
• difficult design/coding is done in pairs
  – two heads to solve difficult problems
  – two sets of eyes to see mistakes
  – serving complementary roles
  • design, code, review, how to test
• in many cases it works very well
  – improving productivity, reducing errors

getting Pair Programming right

• must be able to work very well together
  – approach, pace, personality, style
• each partner must carry his own weight
  – if one is doing most of the work, it is a waste
  – Pair Programming is not newby training
• don’t re-use same teams every day
  – different people have different strengths
  – we learn new things from new people
• only use it on big enough problems

Training

• usually for new team members
  – familiarize them with domain and process
  – develop skill with tools and techniques
• may be formal, practical, or combination
  – reading, seminars, workshops
  – assignments chosen for skill development
  – internship rotations
• may be standardized or ad-hoc
• may involve a designated trainer

Mentoring

• usually for high potential individuals
  – long term, one-on-one relationship
  – to help them make the next step in growth
• career coaching
  – general discussion and counsel
  – seldom involves formal instruction
  – mentor may have little relationship to mentee
• protégé relationships
  – training and assessment for a new position
  – usually starts out as an assistantship role
Team Structure

- strict hierarchy works ...
  - if leader has required skills and experience
  - if team trusts the their leader
- flat collaboration works ...
  - if members have required skills & experience
  - if people step-up to all responsibilities
  - if people can manage their own productivity
  - if the team can always reach consensus
- anarchy ... probably doesn’t work

Leadership

- trust and respect
  - ability and fairness must be beyond reproach
- ability to inspire and motivate
  - you can only lead if others will follow you
- communication skills
  - must be able to communicate up and down
- organizational and domain knowledge
  - must understand processes we are part of
  - must understand problem and solution

Distributed S/W Development

- Advantages
  - larger pool of developers
  - closer proximity to customers
  - potential cost savings
- Challenges
  - communications (language, distance, time)
  - coordination (aggravated by communication)
  - social distance (cultural, organizational)
- Effectiveness
  - depends on problem, organization, people

For Next Lecture

- McConnell, chapter 8
  - the practices of “Defensive Programming”
- Kampe: Availability Taxonomy
  - key terms and concepts
- Kampe: What every programmer needs to know about S/W High Availability
  - introduction to basic principles and methodology

Poisonous Activities

- in-fighting
  - trying to make other people/groups look bad
- sabotage
  - undermining a process you disagree with
- finger pointing
  - blame avoidance rather than problem solving
- empire building
  - advancing yourself at the expense of others
- dishonesty
  - telling people what you think they want to hear
- putting form ahead of substance
  - following rules, while avoiding doing the right thing