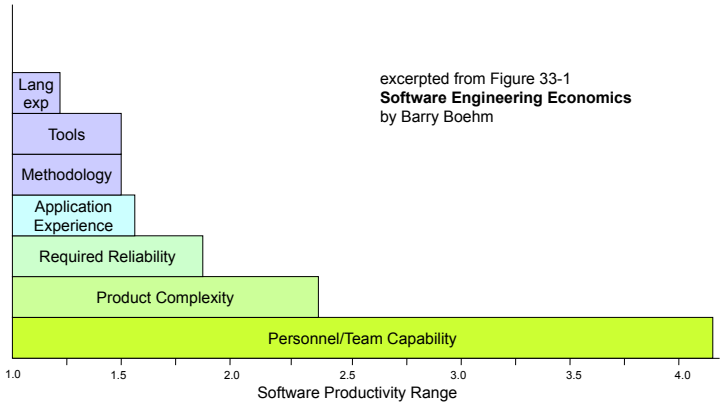
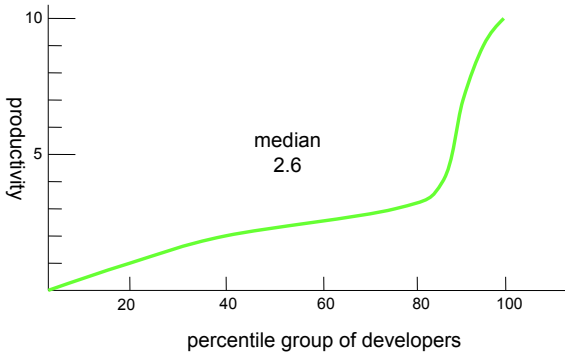
Collaboration and Productivity

- Productivity
  - group and individual productivity
  - contributing factors,
  - developing individual productivity
- Collaboration Models
  - eXtreme Programming
    - philosophy and practices
  - Pair Programming
  - team productivity
  - team leadership

Contributors to Productivity



individual productivity



Individual 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 this may be offset by rising quality

Improving Productivity

- work in many different domains
  - learn different design approaches/issues
  - learn different testing techniques
  - learn different debugging techniques
- work with many different people
  - help other people with their problems
  - learn perspectives & tricks from other people
- learn from our mistakes
  - “what warning signs did I miss?”
  - “what should I have done differently?”

eXtreme Programming

- an agile development philosophy
  - embracing change, short feedback-driven cycles, high stake-holder involvement, communication intensive
- a set of techno-moralistic principles
  - personal courage, mutual respect, responsible behavior, sustainable pace, minimalism, incrementality, continuous integration
- a collection of development practices
  - requirements gathering, design, coding, testing

eXtreme Programming Philosophy

- small, frequent releases
  - lots of micro projects
  - with regular customer feedback
- process minimalism
  - working features are the only deliverables
- team empowerment
  - self organizing teams
  - maximum face-to-face communication
  - collective code ownership

eXtreme Programming Practices

- minimalism in design and implementation
  - prototype to find the best solutions
  - strive for the simplest solution
  - just-in-time s/w development
  - regular refactoring
- pair programming
- standards based w/reusable technology
- test driven development
- continuous integration

Good Collaboration

- prevents mistakes
  - duplicated or unclear code
  - unwarranted assumptions
- improves productivity
  - give us options when we become blocked
  - more hands to help with big jobs
- improves our skills and knowledge
  - disseminate understanding of system parts
  - 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
  - **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 defined complementary roles
    - design, challenge, suggest, code, review, 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 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/hard enough problems

Share your experiences

- Have you done “Pair Programming”?
  - where … class, internship, personal project
- What roles did you assume?
  - how … assigned, evolved, found
- What did you learn?
  - about pair programming
  - design, coding, testing, debugging skills
  - about yourself

Team Productivity

- the team is of paramount importance
  - team does the design and estimation
  - team accepts and commits to sprint tasks
  - team decides how to divide up the work
  - team identifies and solves problems
- key XP team productivity enablers
  - continuous communication
  - collective code ownership
  - team is empowered to solve problems

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 reach consensus
- anarchy … probably doesn’t work
  - and doesn’t achieve benefits of collaboration

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 27, 28.3
  - good chapter on implications of program size
  - brief advice on project estimation
- Kampe: S/W Estimation Principles
  - 3 pages of basic principles
- Wikipedia: COnstructive COst MOdel
  - introduction to constructive estimation
- Peters: S/W Project Estimation
  - general discussion of project estimation
- Wiegers: Risk Assessment
  - good overview of risk assessment and management
- SCRUM Backlog Grooming

Supplementary Slides

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

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