The UNIVERSITY of VERMONT

Benjamin Fenster
CS-205 (Damon)
20 December 2004

Introduction & Project Lifecycle

I. Course Goals
   a. The point of the course is to learn and *do* significant software development
   b. Big difference from student project-type development
   c. Need to communicate ideas about software design to others and yourself

II. Introduction
   a. Definition of Software Engineering
      i. Goal: Good quality software – “timely and affordable development, deployment, maintenance”
      ii. More than just programming. Programming comprises a very small percentage of time spent on a project
      iii. Also Includes:
         1. Project Planning – what should it do?
         2. Design
         3. Testing
         4. Performance Testing
   b. Why?
      i. Prepare for a better job
      ii. Learn how software *is* developed and *should be* developed
   c. Focus
      i. Concepts!
      ii. Not specific tools; not typical job preparation
      iii. Tools / languages change much more quickly than concepts
      iv. The appropriate tools / languages are different from one project to the next
   d. Kinds of Development
      i. “Simple” applications
      ii. Business processing
      iii. Scientific Computing. Very different mindset – machines are still way too slow for what’s needed in intense scientific applications
      iv. Machine Control. Very small programs, usually no user interface. Real time! Very different time constraints.
      v. Games. People write games they want to play. Not like other development.
      vi. Web. Fundamentally developing user interface.
      vii. Database. Structure of data is more important than algorithms to process it.
      viii. Systems. Among the highest reliability constraints

III. Project Features
   a. Features that may apply to any type of project that change things substantially
      i. High Reliability
      ii. Life Critical – *cannot* be wrong
      iii. Business Critical – ATM network goes down, the administrator gets fired. Lose huge amounts of money when the system goes down.
      iv. Performance Critical.
      v. Easy to Learn – mall kiosk, ATM
      vi. Efficient to Use – Might *not* be easy to learn, but very efficient once you know it.
   b. Greenfield vs. Brownfield Development
      i. Greenfield: Creating a brand new program (almost everything you do in school)
      ii. Brownfield: Update existing software (almost everything you do in real life)

IV. Project Roles
   a. Project Manager: What should the project do?
   b. System Architect: How it looks technically
   c. Designer
   d. Programming
   e. Quality Assurance Engineer: Write / run tests
   f. Technical Writer

g. Release Engineer
h. Trainer: Teach people to use the product
i. Customer Support Engineer: Ranges from the know-nothing who answers the phone to people who solve big problems and may even fix the code
j. Porting Engineer: Move from one platform to another
k. Technical Sales Support: Go with salespeople to answer the technical questions

V. Tradeoffs
  a. Trading off Features, Quality, Cost (where Cost == Time)
  b. Features
    i. Define the product.
    ii. When you ask, "What's that product?" the answer is a list of features.
    iii. Solve customer problems. Means you have to *identify* customer problems. Means you have to know the customers.
    iv. Need to be better than the alternatives (so you need to know the competition)
  c. Quality
    i. Reliability, performance, usability
    ii. Expectations depend on version (beta?), company.
    iii. Not all bugs are the same. Does it render the product virtually unusable or is it just really annoying?
  d. Cost
    i. Revenue drops after a while since competitors enter the market
    ii. If you slip, say, a month, that's one month "earlier" that the competition gets to start eating into your revenue.

VI. Why Start a Project?
  a. Many different reasons.
  b. Non-Software Company
    i. Big Corporations have IT groups to build custom software
    ii. Havre to face "Build or Buy" decision every time
    iii. More expensive to build, but get more control
  c. Existing Software Companies
    i. Have existing customers they can study; develop anything they need
    ii. Have "Techies vs. Sales" clashes regularly
  d. New Software Companies
    i. Greenfield development
    ii. Someone sees a solution to an unsolved problem
  e. Marketing – May make the wrong decision. Often does.

VII. Software Economics
  a. A bizarre economy
  b. All costs are like development costs

VIII. Process
  a. Much like custom architecture
  b. Much effort goes into understanding customers' needs
  c. Subject to changes throughout development
  d. That generates delays / costs going up
  e. In architecture, no expectation of ongoing maintenance, or concept of "testing"

IX. Lifecycle
  a. Phases
    i. Requirements, Design, Implementation, Test, Release
    ii. Called Lifecycle
    iii. Then include Maintenance
  b. Waterfall
    i. Each step follows its predecessor, until you release.
    ii. Doesn't work very well. Need input from each step in the earlier steps
    iii. Cheaper to figure out problems along the way than to try to get everything right in advance. "Plan to build two!"
  c. Cyclic / Spiral Model

i. Effort is focused on different area on each cycle
ii. Note that in CS-205 we only have time for one cycle.