Planning and executing a software project

PSU CS 300 Lecture 3-1

Bart Massey
Assoc Prof Computer Science
Portland State University
<bart@cs.pdx.edu>

Focus: Personal SW dev

- **Oddly, mil-spec giant project methods don't work**
- **Goal: create a personal success *process***
- **Lots more than just coding**

Professionalism and responsibility

- **You're working on your own, *but* you still owe it to folks to do it right**
  - **yourself**
  - **your “customer”**
  - **society**
- **The biggest disasters start with basic neglect**

Project management

- **Activities like**
  - **set up**
  - **estimate & plan**
  - **measure & control**
  - **evaluate**
  - **deliver**
- **You must *balance* these against actual dev work**

Work product

- **A *work product* is anything you produce as part of your dev work**
  - **code, external docs, internal docs, SCMS repo, etc**
- **Many tiny work products rather than one big one**

Setup: Get basics in place

- Environment(s)
  - place, tools
- Available and needed time
- Resources
- Infrastructure
  - SCMS, backups, build
  - web + email + ... comm

What you need to estimate & plan

- Need requirements
  - but not necessarily complete
- Need architecture
  - but not necessarily correct
- Need design constraints
- Need capabilities and resources

How to plan

- Build a *work breakdown structure* (WBS)
- WBS is hierarchical decomposition of work
- Should decompose to fine grain: preferably 1-2 hour chunks
- WBS maintained ongoing

A WBS is tasks

- ID
- Parent / children
- Resources needed
- Constraints
  - schedule / precedence
- Risks

Why a WBS helps

- Can produce a more accurate schedule
- Can trace against project to prevent missed tasks
- Can identify trouble spots early

Project scheduling

- Build PERT or Gantt chart
  - identify slack time
  - level resource profile
  - identify critical path
- For an individual project
  - helps stay “on task”
  - helps limit reqs creep

Milestones

- Generate milestones from schedule and track 'em
- Not too many, nor too few
- Milestones before and after critical / high-risk steps
- Contingency plans

Don't get stuck!

- Milestones help you identify when you are not making expected progress
- Remediations:
  - Fix something broken
  - Replan or redesign
  - Get help
  - *Not* “just work harder”

SCMS's measure & control

- Measure productivity and change in productivity
- Measure quality and change in quality
- Locus for V&V

V&V

- Verification is an ongoing process
- “V model”: plan system validation initially, system verification after reqs, integration test after arch, unit test after detailed design

Traceability

- Verification is enabled by knowing relation between each work product and
  - work products before and after
  - project data

Deployment

- You will have a build infrastructure during development
- Need to be able to build a deployable product
  - installer?
  - docs

Maintenance

- Workflow described here sets you up for maintenance
  - backups, scms, build env
  - good internal docs
  - V&V tools for regression
  - comm infrastructure