Risk and Change Management

Emanuele Della Valle

http://emanueledellavalle.org
Credits

- This slides are largely based on Prof. John Musser class notes on “Principles of Software Project Management”
  - Original slides are available at [http://www.projectreference.com/](http://www.projectreference.com/)

- SVN slides are based on “Version Control With Subversion” by Samnang Chhun
  - Original slides are available at [http://www.slideshare.net/samnang.chhun/version-control-with-subversion-1871040](http://www.slideshare.net/samnang.chhun/version-control-with-subversion-1871040)

- Reuse and republish permission was granted
Today

- Risk Management
- Feature Set Control
- Change Control
Risk Management

- Problems that haven’t happened yet
- Why is it hard?
- Some are wary of bearing bad news
  - No one wants to be the messenger
  - Or seen as “a worrier”
- You need to define a strategy early in your project
Risk Management

- Identification, Analysis, Control
- Goal: avoid a crisis
- Thayer: Risk Mgmt. vs. Project Mgt.
  - For a specific vs. all projects
  - Proactive vs. reactive
Risk Management
Definitions

- **Project Risk**
  - Characterized by:
    - Uncertainty \(0 < \text{probability} < 1\)
    - **NOTE:** If the probability is high, you may have planned the project in a wrong way.
    - An associated loss (money, life, reputation, etc)
    - Manageable – some action can control it

- **Risk Exposure**
  - Product of probability and potential loss

- **Problem**
  - A risk that has materialized
Risk Management

Types of Risks

- Schedule Risks
  - Schedule compression (customer, marketing, etc.)

- Cost Risks
  - Unreasonable budgets

- Requirements Risks
  - Incorrect
  - Incomplete
  - Unclear or inconsistent
  - Volatile

- Quality Risks

- Operational Risks

- Most of the “Classic Mistakes”
  - Classic mistakes are made more often
Risk Management

Types of Unknowns

- Known Unknowns
  - Information you know someone else has

- Unknown Unknowns
  - Information that does not yet exist
Risk Management

Processes

Risk Management

Risk Assessment
- Risk Identification
- Risk Analysis
- Risk Prioritization
- Risk Management Planning

Risk Control
- Risk Resolution
- Risk Monitoring

[Source: “Software Risk Management”, Boehm, 1989]
Risk Management – Risk Assessment

Risk Identification

- Get your team involved in this process
  - Don’t go it alone

- Produces a list of risks with potential to disrupt your project’s schedule (but also budget, quality, ...)

- Use a checklist or similar source to brainstorm possible risks
  - Cached version available
Risk Management Processes

- Risk Identification
- Risk Analysis
- Risk Prioritization
- Risk Management Planning
- Risk Resolution
- Risk Monitoring

[Source: “Software Risk Management”, Boehm, 1989]
Risk Management – Risk Assessment

Risk Analysis 1/2

- Determine impact of each risk
- Risk Exposure (RE)
  - RE = Probability of loss * size of loss

- Examples
  - risk is “Facilities not ready on time”
    - Probability is 25%, size is 4 weeks, RE is 1 week
  - risk is “Inadequate design – redesign required”
    - Probability is 15%, size is 10 weeks, RE is 1.5 weeks

- Statistically are “expected values”
- Sum all RE’s to get expected overrun
  - Which is pre risk management
Risk Management – Risk Assessment

Risk Analysis 2/2

- Estimating size of loss
  - Loss is easier to see than probability
    - You can break this down into “chunks” (like WBS)

- Estimating probability of loss
  - Use team member estimates and have a risk-estimate review
  - Use Delphi or group-consensus techniques
  - Use gambling analogy “how much would you bet”
  - Use “adjective calibration”:
    - highly likely
    - probably
    - improbable
    - unlikely
    - highly unlikely
Risk Management Processes

- Risk Assessment
  - Risk Identification
  - Risk Analysis
  - Risk Prioritization
  - Risk Management Planning
- Risk Control
  - Risk Resolution
  - Risk Monitoring

[Source: “Software Risk Management”, Boehm, 1989]
Risk Management – Risk Assessment

Risk Prioritization

- Remember the 80-20 rule
- Often want larger-loss risks higher
  - Or higher probability items
- Possibly group ‘related risks’
- Helps identify which risks to ignore
  - Those at the bottom
Risk Management Processes

Risk Management

- Risk Assessment
- Risk Analysis
- Risk Prioritization
- Risk Management Planning
- Risk Control
- Risk Resolution
- Risk Monitoring

[Source: “Software Risk Management”, Boehm, 1989]
Risk Management Planning

- Can be 1 paragraph per risk
  - For an example see Service-Finder’s “Risk Management and contingency plan”

- McConnell’s example
  - Cached version available
Risk Management Processes

- Risk Identification
- Risk Analysis
- Risk Prioritization
- Risk Management Planning
- Risk Resolution
- Risk Monitoring

[Source: “Software Risk Management”, Boehm, 1989]
Risk Management – Risk Control

Risk Resolution 1/2

- **Risk Avoidance**
  - Don’t do it
  - Scrub from system

- **Risk Assumption**
  - Don’t do anything about it
  - Accept that it might occur
  - But still watch for it

- **Problem control**
  - Develop contingency plans
  - E.g., allocate extra test resources

- **Risk Transfer**
  - To another part of the project (or team)
  - Move off the critical path at least
Knowledge Acquisition

- Investigate
  - Ex: do a prototype
- Buy information or expertise about it
- Do research
Risk Management Processes

- Risk Identification
- Risk Analysis
- Risk Prioritization
- Risk Management Planning
- Risk Resolution
- Risk Monitoring

[Source: “Software Risk Management”, Boehm, 1989]
Risk Management – Risk Control

Risk Monitoring

- Top 10 Risk List
  - Rank
  - Previous Rank
  - Weeks on List
  - Risk Name
  - Risk Resolution Status

- A low-overhead best practice

- Interim project post-mortems
  - After various major milestones

- McConnell’s example
  - Cached version available
Risk Management

Processes

Risk Management

- Risk Identification
- Risk Analysis
- Risk Prioritization
- Risk Management Planning
- Risk Resolution
- Risk Monitoring

Risk Assessment

Risk Control

[Source: “Software Risk Management”, Boehm, 1989]
Risk Management

Risk Communication

- Don’t be afraid to convey the risks
- Use your judgment to balance
  - Sky-is-falling whiner vs. information distribution
Risk Management
Miniature Milestones 1/3

- A risk-reduction technique
- Use of small goals within project schedule
  - One of McConnell’s Best Practices (Ch. 27)
- Fine-grained approach to plan & track
- Reduces risk of undetected project slippage

Pros
- Enhances status visibility
- Good for project recovery

Cons
- Increase project tracking effort
Risk Management

Miniature Milestones 2/3

- Can be used throughout the development cycle
- Works with hard-to-manage project activities or methods
  - Such as with evolutionary prototyping
- Reduces unpleasant surprises
- Success factors
  - Overcoming resistance from those managed
  - Staying true to ‘miniature’ nature
- Can improve motivation through achievements
Risk Management

Miniature Milestones 3/3

- Requires a detailed schedule
- Have early milestones
- McConnell says 1-2 days
  - Longer is still good (1-2 weeks)
- Encourages iterative development
- Use binary milestones
  - Done or not done (100%)
Feature-Set Control

- It is a class mistake avoidance technique

- Early Stages
  1. Minimal Specification
  2. Requirements Scrubbing
  3. Versioned Development

- Mid-Project
  - Effective change control

- Late-Project
  - Feature cuts
Feature-Set Control - Introduction

Traditional Specs

- Drive for “traditional” specs
  - Necessity
  - Downstream cost avoidance
  - Full control over all aspects

- As McConnell notes:
  - “But the goal is not to build exactly what you said you would at the beginning. It is to build the best possible software within the available time.”
  - Idealistic but worth remembering
Feature-Set Control - Early Stages

Minimal Specification 1/2

- Tradition spec. issues
  - Wasted effort
    - Too much detail
  - Obsolescence
  - Lack of efficacy -- details do not guarantee success
  - Overly constrained design
  - User assumption that costs are equal (UI ex.)
Feature-Set Control - Early Stages

Minimal Specification 2/2

- **Benefits**
  - Improved morale and motivation
  - Opportunistic efficiency
  - Shorter requirements phase

- **Costs and Risks**
  - Omission of key requirements
  - Unclear or impossible goals
  - Gold plating
  - Used for the wrong reasons
    - Lazy substitute for doing good requirements

- **Success Factors**
  - Used only when requirements are flexible
  - Capture most important items
  - Involve key users
This is not XP (extreme programming)

In XP Requirements are
- expressed as automated acceptance tests rather than specification documents
- defined incrementally, rather than trying to get them all in advance

XP has broader goals
- An attempt to reconcile humanity and productivity
- A mechanism for social change
- A path to improvement
- A style of development
- A software development discipline
Feature-Set Control - Early Stages
Requirements Scrubbing

- Removing a feature from the product
  - Eliminates all effort: spec., design, dev., test, doc.
  - The earlier the better
  - Typically done during or right after Requirements

- Less risky than minimal specification

- Aims
  - Eliminate all but absolutely necessary requirements
  - Simplify all complicated requirements
  - Substitute cheaper items
Feature-Set Control - Early Stages

Versioned Development

- Eliminate them from the current version
- “Let’s put it in release 1.1”
  - You’re still saying “Yes”, not “No”
- By next rev. the list has changed anyway
- My favorite ;-)
Change Management
The Feature-Creep Phenomenon 1/2

- Avg. project has 25% change in requirements during development

- Sources of change
  - Marketing: want to meet customer’s check-list
  - Developers: want to perfect r1 deficiencies
  - Users: want more functionality or now ‘know’ what they want

- They will all try to ‘insert’ these during dev.
The devil is in the details

McConnell’s example: “trivial” feature can have +/- weeks of impact

Developers can insert things when you’re not looking

No specification can cover all details. You must specify!

Programmer ideal: flip switch

Up to 10-1 differences in program size with the same specifications
Change Management

Change Control Board (CCB) 1/2

- McConnell “best practice” (see Ch. 17)
- Structure: representatives from each stakeholder party
  - Dev., QA, Marketing, Mgmt., Customer support
- Perform “change analysis”
  - Importance, priority, cost, benefit
Change Management

Change Control Board (CCB) 2/2

- CCB are similar to a triage
  - Allocating scarce resources
  - Some will not receive treatment
  - Life-critical to the project

- CCB will say “No” more than “Yes”

- Watch for bureaucracy
Change Management

Change Control

- A set of fully fledge methods and tools

Change Control System

Configuration Management System

Configuration Management Tool

Configuration Control Items

- Good Reference
  - “Quality Software Project Management”, Futrell, Shafer, Shafer
    - Preview available on Google Books Search
      http://books.google.com/books?id=8GqC7xHTwGsC
Change Management

Change Control – The Problem 1/2

Without Configuration Management

- Which version works?
- Which versions have bug/feature X?
- What’s the different between certain versions?
Without Configuration Management

- How to combine these two version into one working program?
- Who is responsible on keeping the latest version?
Change Management

Change Control – Why?

- Backup & Restore
- Synchronization
- Short-Term Undo
- Long-Term Undo
- Track Changes
- Track Owner
- Branching & Merging
Change Management

Terminology

- Configuration Control Item (CCI)
  - Anything suitable for configuration control
  - Source code, documents, diagrams, etc.

- Configuration Control Tool
  - Any support for managing CCIs (e.g., see slide 44)

- Configuration Control
  - process of evaluating, approving and disapproving, and managing changes to CCIs.

- Change Control
  - process of controlling changes
  - typical steps (see slide 50)
    - Proposal, evaluation, approval, scheduling, implementation, tracking

- Version Control: controlling software version releases
  - Recording and saving releases
  - Documenting release differences
Change Management

Configuration Management Tools

- CSV
- Subversion (svn)
  - My preferred one :-)
- Microsoft Visual Source
- SVK
- Safe
- Barzzar
- Mercurial (hg)
- Git
- ...

...
Change Management

Configuration Management Tools - SVN

- Cross Platform / Open Source / Free
- Central repository
- Atomic commit
- Availability of free client software / Plugin for most known IDEs
  - e.g. Eclipse
- Most of Open source hosting sites support it
  - e.g. sourceforge, codeplex, google code
- Free SVN hosting available
  - e.g. XP-Dev
The Working Cycle
Terminology

- **Repository (repo)**: the database storing the files
- **Working Copy**: Your local directory of files, where you make changes.
- **Revision**: What version a file is on (v1, v2, v3, etc.)
- **Check out**: Download a file from the repo
- **Check in**: Upload a file to the repository (if it has changed). The file gets a new revision number, and people can “check out” the latest one.
- **Update**: Synchronize your files with the latest from the repository. This lets you grab the latest revisions of all files.
- **Head**: The latest revision in the repo
- **Changelog/History**: A list of changes made to a file since it was created
- **Revert**: Throw away your local changes and reload the latest version from the repository
Change Management

Configuration Control

- A management support function

- Includes
  - Program code changes
  - Requirements and design changes
  - Version release changes

- Essential for developed items
  - Code, documentation, etc.

- Example
  - The case of the code that used to work
    - But didn’t in time for the demo
Change Management

Configuration Control Needs

- Establish clearly defined management authority
- Setup control standards, procedures and guidelines
  - All team members must be aware of these
- Requires appropriate tools and infrastructure
- Configuration Management Plan must be produced during planning phase
Change Management

Configuration Management Plan

- Must be produced during planning phase
- Often part of Software Development Plan
- Example of Change Control Procedure

- Cached version available
Change Management

Software Configuration Management

- Formal engineering discipline
- Methods and tools to identify & manage software throughout its use
- For basic information consult http://en.wikipedia.org/wiki/Software_configuration_management
3rd Homework assignment

- Top 10 Risk List for your project
  - Fine if you use a check-list (see slide 10) ...
  - **BUT** think about *your* project

- Format and Submission
  - Use the template available at http://www.emanueledellavalle.org/slides/P&MSP2011_07f_template-homework-3.doc
  - Add homework-3 to the zip file you created for homework-1 and homework-2 and resubmit the entire package through the easychair Web site
Optional Readings

- McConnell: 11 “Motivation”, 13 “Team Structure”
- Schwalbe: 8 “Project Human Resource Management”
Questions?