

 POLITECNICO DI MILANO

Dipartimento di
Elettronica e Informazione

Planning and Managing Software Projects 2010-11
Session 10 – 2nd part

Final Stages

Emanuele Della Valle
<http://emanueledellavalle.org>

- 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/>
- Reuse and republish permission was granted

- Migration and Roll-out
- Maintenance Phase (often and after project activity)
- Project Recovery
- Post Project Reviews (Post-mortems)
- Success tips
- Final exam review

- Planning
- Measuring
- Evaluating
- Acting

- BCWS
- BCWP
 - Earned value
- ACWP
- Variances
 - CV, SV
- Ratios
 - SPI, CPI, CR
- Benefits
 - Consistency, forecasting, early warning

Final Steps

6

- Migration
- Roll-Out
- Training
- Documentation
- Shipping Details
- Installation

Migration

7

- Moving users from existing system to your new one

- Includes
 - Description of environment (computers, DBs, interfaces)
 - Description of existing data needed
 - Description of operational constraints
 - when can we move to the new system?
 - Weekends only?
 - Last week of month only?
 - List of affected organizations and contacts
 - Plan of steps to be taken

- Does it require a service interruption?
 - If so, when does this happen? A weekend?
- Training?
- Is there a helpdesk?
 - If so, do they have “scripts” or new material?

- Communication with customers is crucial
 - What is happening, when, and why
 - “Why” should remind them of the benefits
 - Not too much detail or too little
 - Where do customers go for more information?
- Minimize intrusiveness
- Find-out about customer’s key dates
 - When does the system absolutely need to be stable?
 - Know about their important deadline dates
 - They must buy-into the approach!

1. Flash-Cut

- Straight-move from old system to new
 - A) Immediate Replacement
 - Fastest approach
 - Still want a back-out plan
 - Requires strong planning and testing
 - B) Parallel Operation
 - Mitigates risk
 - Parallel to either existing manual or system process
 - Cut occurs once new system “burned-in”

2. Staged

- Replace one part of existing system at a time

- Level of business disruption
- Degree of freedom in “production” date
- How much internal opposition to system is there?
 - If higher, perhaps a longer ‘adjustment’ period
- Your comfort level of system quality
 - If questionable, may want to mitigate risk

- Criteria: What conditions must be met prior?
- Responsibility: Who decides?
- Operations: Who 'owns' it once it's live?
- Rehearsals: Sometimes used.

Migration Flash-Cut

14

- Immediate Replacement
 - Ex: new corporate-wide calendaring system
- Requires very careful planning & testing
- Still try to get some users to “try” it first if possible
- Develop a back-out plan

- Especially important for “conversions”
 - Customers already have expectations and needs as defined by their existing system
 - Must be able to restore customer’s service ASAP
- May mean running both simultaneously “just in case”
- Leave it in place for awhile (more than a day!)
- When to back out?
 - Management: sooner
 - Technicians: one-more-fix
 - Set a time limit (ex: 3 hours of start)

- Multiple variations of this method
- An “adoption” period
 - See telephone industry with new area codes
 - Both work for a period of time
- Avoid flash-cuts if possible
 - Start with test subjects

- Quote:
 - If you add a cup of champagne to a barrel of sewage, you'll have a barrel of sewage
 - If you add a cup of sewage to a barrel of champagne, you'll have a barrel of sewage
- Most systems need this step
- Most PMs forget this
- Impacts both completely new and replacement systems
- The “data” often more valuable than the “system”

- Data Sources:
 - Where does it come from?
 - Do you need to modify data on the way in?
 - Is it accurate?
- Process Controls:
 - Does it happen all at once?
 - How do you guarantee it's been done correctly?
- Completion:
 - How do you handle any 'exceptions' ?
 - Do you make backups? Can you restart?

Other Final Steps Beyond Migration

19

- Roll-Out
- Training
- Documentation
- Shipping Details
- Installation

- Create a “Release Checklist”
 - Avoid activities falling through the cracks
 - Example
 - <http://www.construx.com/Page.aspx?hid=1216>
 - Cached
http://emanueledellavalle.org/slides/P&MSP2011_10b_Sample-Release-Checklist-Construx.pdf
 - Activities by Group:
 - Engineering, QA, Documentation, Operations
 - Possibly sign-off signatures
- Roll-out: Must have a plan for the process
 - Often on a given day (ex: a Sat.)

- Often more than just end-users
 - Users
 - Sales & Marketing staff
 - System operators
 - Maintenance engineers (possibly)
 - Sales engineers (possibly)

- Must be ready by ship-date
- Final user documentation
- Updates to other
 - Operations documentation
 - Development documentation
 - Sales and marketing material
 - Web site
 - Test reports

- Packaging (if commercial product)
- Marketing collateral
- Security mechanisms (if commercial product)
- Licensing

- Scripts
- Uninstall (if not Web-based)
- If you need to install your software (as on PCs):
 - Don't underestimate:
 - Time this takes to develop
 - Importance of a “first impression”
- Or, if “custom” software you're reselling
 - Installation at site is often a “mini-project”

- Project management not always carried over
- The “No respect” phase
- Less “glamorous”
 - Lack of enthusiasm
- Pressure to make fixes quickly
 - For “production” problems
- Software can become “hacked” “patchwork” over time
- Finding a support & test platform can be difficult
 - Often the forgotten child until fixes are needed

- Compare to hardware maintenance
 - Not to keep state same; but changes to state
 - Fixes and enhancements
- Configuration control is very important
 - Fixing the “right” version; tracking branches
- Smaller team
 - Often not a ‘dedicated team’
 - Drawn from developer with other main tasks

- Contracts, remember those?
 - Always consider the maintenance phase here
 - Often via a “labor hours” contract
 - Time & materials in a “direct” scenario
 - Otherwise via “maintenance contract”
 - Percentage of software license fee
 - Ex: 20% of original cost per year
- Corp. budget if internal projects
 - Often annual/monthly “maintenance” allocations

- How to save a “drowning project”
- 3 Approaches
 1. Cut the size of the software
 2. Increase process productivity
 3. Slip the schedule, proceed with damage control
- Opportunity for decisive leadership action
- Not a time to ‘just cut corners’
 - Be realistic (not foolish)
- Timing: politically important
 - Not too early, not “too” late

- Assess situation
 - Is there a hard deadline, what's negotiable, etc.
- Don't do what's been done already
- Ask team what needs to be done

- Restore morale
 - Sacrifice a sacred cow
 - Dress code, off-site, catered meals, etc
 - Cleanup personnel problems
- Focus people's time
 - Remove non-essential work

- Fix classic mistakes
 - Inadequate design, shortchanged activities, etc?
- Create “Miniature Milestones”
 - Small (in day(s)), binary, exhaustive
 - Boosts morale: getting things done!
- Track progress meticulously
- Recalibrate after a short time
- Manage risk painstakingly

- Stabilize the requirements
- Raise the bar on change requests
- Trim the feature set
 - Determine priorities, cut the low ones
- “Take out the garbage”
 - Find error-prone modules; re-design
- Get to a known, stable state & build from there

- a.k.a.
 - Lessons Learned Review
 - Postmortem
 - Post Project Analysis (PPA)
 - Post Performance Analysis
- Focused on: Process not People!
 - Potentially a finger-pointing, blame-game exercise

- Email team to schedule meeting
- Use a Survey Form to gather initial feedback
 - http://www.havering.gov.uk/media/doc/7/m/LBH_Post_Project_Review_Template_TMP.doc
 - Cached
http://emanueledellavalle.org/slides/P&MSP2011_10c_LBH_Post_Project_Review_Template_TMP.doc
- Ask them to collect all potentially relevant data
 - Dimensional project data work products: size, qty, etc
 - Change requests
 - Time and effort data
- Conduct meeting
 - Collect data and feedback, discuss
- Summarize in a PPR report

1. On schedule
 - Requires good: plan; estimation; control
2. Within budget
 - Again: planning, estimation & control
3. According to requirements
 - Importance of good requirements
 - Perception & negotiation critical

- Learn to say “No”
 - Be polite but firm
- The Value of Versions
 - “We will put that in phase 2”
- An Ounce of Prevention

- Keep requirements tight & focused
- One milestone at a time
- Smaller, incremental chunks
- As simple as possible but no simpler

- Analysis Paralysis
 - Over-process
 - Nothing gets finished
 - 65% of software professionals have experienced this
- Paralysis Paranoia
 - Fear of over-process = process avoidance

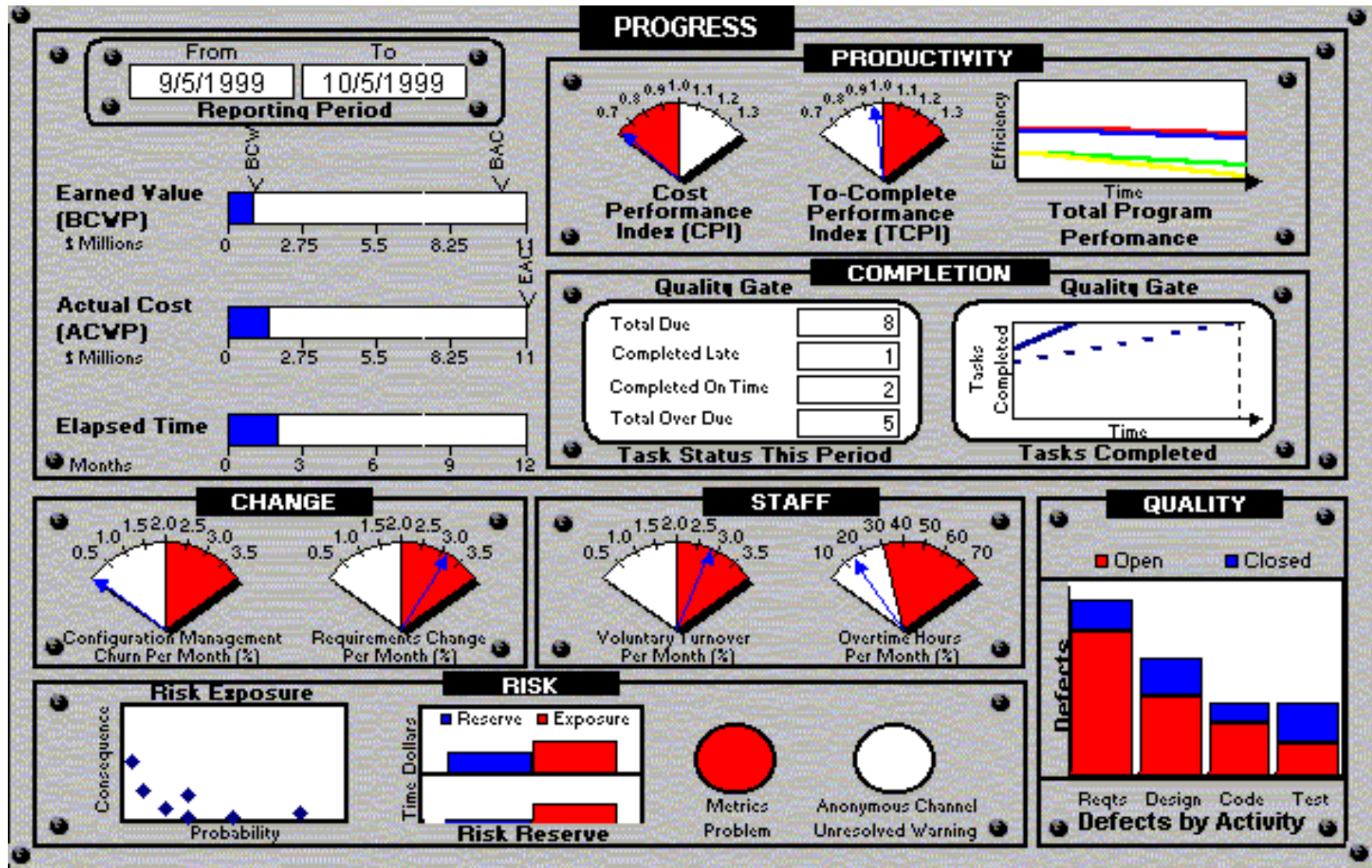
Management by Walk About

- Shows your actually involved day-to-day
- Recognizes individuals may say more 1-on-1
- Allows spontaneity
- Finds personnel problems sooner

- Don't be a "Control Freak"
- You need to be the "hub" but not everything

- Herbsleb, 1994, “Benefits of CMM-Based Software Process Improvement”

Measure Category of SPI	Range	Median	Number of Data Points
Years engaged in SPI	1 - 9	3.5	24
Yearly cost of SPI per software engineer	\$490 - \$2004	\$1375	5
Productivity gain per year	9% - 67%	35%	4
Early defect detection gain per year	6% - 25%	22%	3
Yearly reduction in time to market	15% - 23%	19%	2
Yearly reduction in post-release defect reports	10% - 94%	39%	5
Business value (savings/cost of SPI)	4.0 - 8.8	5.0	5



[Source http://www.iceincusa.com/Content/controlpanel_new.gif]

- By Industry
 - Best: Retail
 - Tight cost controls in general
 - Worst: Government
 - Least cost controls
- By Size
 - Smaller is better: cost, duration, team
- Stats
 - <http://www.ambyssoft.com/surveys/success2007.html>
 - <http://www.ambyssoft.com/surveys/success2008.html>
 - <http://www.ambyssoft.com/surveys/success2010.html>

Questions?

45