

Project Management Course Book

T++ Technical Skills Training Program

CUNY Institute for Software Design & Development (CISDD)

New York Software Industry Association (NYSIA)

Aron Trauring

Class 10 November 4th, 2004

Class Agenda

1:30-2:00 Exercise 9 Break-out Presentations

2:00-3:00 Agile Project Management (Presentation)

3:00-3:15 Break

3:15-4:15 Tools and Techniques (Presentation)

4:15-4:30 Exercise 10 — Course Summary

4:30-4:45 Break-Out Presentations

4:45-5:00 Break and Dot Voting

5:00-5:30 Course Summary

Agile Project Management

Prediction is very difficult, especially if it's about the future. – Niels Bohr

Agile Project Time Planning

- Protect time by aggregate time buffer
- Aggregate components into groupings and look at dependencies of groupings
- Critical Path — longest path through real time required for a single chain of dependent tasks to complete in order
- 'Eventually everything hits the CP' – Goldratt
- Components not tasks
- PERT chart for parallel development
- Resource constraints — specialists need to be added to PERT
- Samples http://whatis.techtarget.com/definition/0,,sid9_gci331391,00.html

When to Start a Parallel Piece?

- Work-in-Process (WIP) inventory should be minimized
- Delay in delivery delays project
- Project buffer is for Critical Path
- Feeding Buffer — separate buffer for each parallel path, including resource constraints
- Delivery date when needed by CP
- Start date backed out from there

Agile Tracking

- Monitor all buffers
- Criticality of buffer usage varies by time in project
- Early in project — over 40% used — 'Watch' status
- 90% used — 'Danger' status
- Issue logs — indicates blocked inventory

Key Monitoring Metrics

- Number of open issues — including trend and age
- Number of units of blocked inventory — including trend and age of blockage
- Project and Feeding buffer usage

Development Manager

- Manages software organization
- Makes sure appropriate tools and environment available
- Motivates the team
- Ensures a learning organization
- Manages constraints including build business case for elevating constraints
- Measured on Production Rate (R) and Lead Time (LT)

Program Manager

- Responsible for control and co-ordination of a Software System Product
- A set of projects with mutual dependencies that are occurring in parallel
- Manages the 'ends' — input and output
- Feeds the inventory at rate it can be consumed
- Manages the release to customers — deployment, packaging etc
- Overall responsibility for Throughput (T) and Inventory in process (V)

Product Manager

- Determines product mix — set of inventory for production cycle
- Responsible for creation of requirements
- Overall responsibility for Net Profit (NP) and Return on Investment (ROI)
- Determines potential Throughput, selects delivery date and determines investment in inventory

Project Manager

- Shepherds inventory through the production system
- Responsible for project planning and monitoring
- Creates the PERT chart
- Controls internal release of inventory
- Negotiates project plan with Development manager
- Keeps other managers informed of events that impact CP
- Key job: keep the issue logs empty

Multi-team/-site Projects

- Start early iterations in one place, one common project room
- Team has representatives from subsequent teams/site
- Focus on discovering and building core architecture
- UP — inception and elaboration
- Once core built work in parallel, multiple teams
- Team representatives have good understanding of core
- Relationship builds good communication
- Try to work with common iteration time-lines so can do joint integration

Adaptive Planning

- No detailed task plan for all iterations
- No fixed plan on number and length of iterations, or what gets done
- Create PERT Chart with potential interim deadlines
- Product Manager may ask for interim deadlines — e.g. customer demo and final delivery
- Project Manager does protected time estimate
- Task plan in detail for upcoming iteration only
- Ensures you are taking steps with maximum information and least risk
- Allows for easy mid-course adjustments

Wide-band Delphi for additional time estimates

1. Kickoff meeting — At least three people meet to discuss project and components
2. Estimation — each person makes three estimates — most likely, optimistic, pessimistic
3. Meeting: give estimates to facilitators and discuss without revealing identities
4. Repeat steps 2 and 3 at least once
5. Take averages of final cycle estimates
6. Calculate final estimate: $(\text{Optimistic} + \text{Pessimistic} + (4 * \text{Most Likely})) / 6$
7. Calculate likely deviation: $(\text{Pessimistic} - \text{Optimistic}) / 6$

Task Generation

- Iteration planning meeting before iteration
- Team generates tasks
- Account for overhead tasks — meetings, demos

Task Assignment

- People choose their own tasks
- Self-directed teams
- In case of conflicts and tasks not getting done manager advises and guides, team decides
- Visible plan and issue tracker on project log

Criteria for Ranking Requests

1. Risk — all different types, technical, political, requirements
2. Coverage — how many parts of system effected
3. Criticality — business value
4. Skills development — training team in new areas

Dot Voting

- All requests listed on white board
- Every team member gets 20 dots and distributes them to requests
- Discuss and decide
- Another dot iteration optional
- Rank before every iteration to take into account new external and internal developments
- Numeric ranking is an alternative

Iteration Length

- Between 1-6 weeks (some methods are specific)
- Small team: 1-3 weeks; large team: 4-6 weeks
- Early discovery iterations are longer

Iteration Plan Steps

1. Decide the time-box
2. Decide worker availability keeping in mind availability buffers
3. Choose high priority request and create estimate if it doesn't have
4. Repeat step 3 until no people time left

Iteration Checklist Advice

- Make sure all technology etc. available before iteration
- Make first iterations longer
- Spread out new ideas and practices over several iterations
- Complete a request within an iteration — Use Cases may have to be broken up
- If running out of time, remove or simplify requests
- Get customer feedback on which requests are secondary\
- No new requests once iteration starts

Iteration Task Tracking

- Someone assigned to do tracking gathers information
- Do it daily
- Task log
- Physical cards

Continuous Integration

1. Check in code, e.g. to CVS
2. CI system wakes up every N minutes and queries CVS if new check in
3. CI extracts new code with email of developers
4. CI compiles entire application
5. If passes compilation, CI runs unit tests
6. If pass unit test run, CI runs integration tests
7. CI updates project log with information about what and who failed
8. CI emails developers and PM about test fails

Project Logs

- Wikis
- Collectors
- CMS

Moore-Style Vision Statements

- For — target customer
- Who — statement of need or opportunity
- The — product name — is a — product category
- That — key benefit, compelling reason to buy
- Unlike — primary competitive alternative
- Our product — statement of primary differentiation

Requirements Workshops

- First workshop might discover most major requirements
- Do detailed analysis of only one or two for first iteration
- Schedule additional workshops as needed

Various types of brainstorming techniques

- Traditional — don't comment or laugh
- Brain writing — put ideas on cards
- Affinity clustering — group clusters cards
- Rotation writing — write and pass

Test Driven Development

- Test everything
- Write tests before code
- All tests are pass / fail
- Tests always get written and more fun to write
- Writing test focuses on proper public behavior

Tools and Techniques

Tools and Techniques: <http://www.fourm.info/TPP/SPM2/links.html>

