Lecture 3: Project Management

Project Management
- Planning Tools
- PERT charts, Gantt Charts, etc.
- Meetings

Risk Management
- Risk Assessment
- Risk Control

Measurement
- choosing software metrics
- some example metrics

Project Management Basics

Thankless job:
- success is not noticeable
  - little evidence the manager did anything
  - project looks simple in hindsight
- failure is very obvious
  - the manager will get blamed when things go wrong

Difficult Job
- Problems to solve include:
  - Do we have the resources (funding, people, time) for the task?
  - Are we certain the task is stated correctly?
  - How can we use limited resources most effectively?
  - How does recent (lack of) progress affect the plan?
  - What lessons can be learnt for future tasks?

Source: Adapted from Blum, 1992, 426-7
see also: van Vliet Chapter 2

Principles of Management

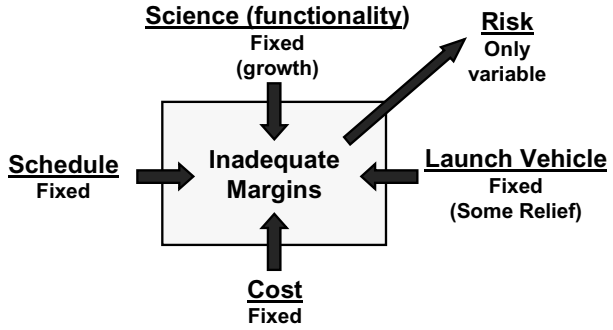
A manager can control 4 things:
- Resources (can get more dollars, facilities, personnel)
- Time (can increase schedule, delay milestones, etc.)
- Product (can reduce functionality - e.g. scrub requirements)
- Risk (can decide which risks are acceptable)

Approach (applies to any management)
- Understand the goals and objectives - quantify them where possible
- Understand the constraints - with uncertainty, use probability estimates
- Plan to meet the objectives within the constraints
- Monitor and adjust the plan
- Preserve a calm, productive, positive work environment

Note: You cannot control what you cannot measure!

Critique of Mars'98 Program

Source: Adapted from MPIAT 2000, p6



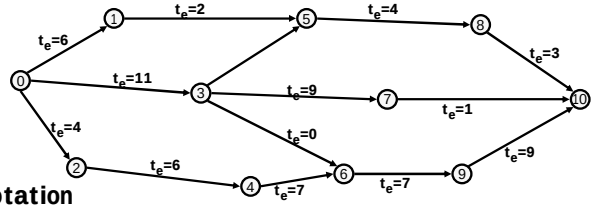
Tool 1: Work Breakdown Structure

Source: Adapted from Blum, 1992, p438
see also: van Vliet pp192-3

- 1.1 Software Systems Engineering
  - 1.1.1 Support to Systems Engineering
  - 1.1.2 Support to Hardware Engineering
  - 1.1.3 Software Engineering Trade Studies
  - 1.1.4 System Requirements Analysis
  - 1.1.5 Software Requirements Analysis
  - 1.1.6 Interface Analysis
  - 1.1.7 Support to Systems Test
- 1.2 Software Development
  - 1.2.1 Deliverable Software
  - 1.2.1.1 Requirements Analysis
  - 1.2.1.2 Architectural Design
  - 1.2.1.3 Procedural Design
  - 1.2.1.4 Code
  - 1.2.1.5 Unit Test
  - 1.2.1.6 Software Integration Test
  - 1.2.1.7 Technical Reviews
  - 1.2.1.8 Technical Training
  - 1.2.2 Non-deliverable Software
  - 1.2.3 Purchased Software
  - 1.2.3.1 Package Evaluation
  - 1.2.4 Development facilities and tools
- 1.3 Software Test and Evaluation
  - 1.3.1 Software Dev. Test & Evaluation
  - 1.3.2 End-Product Acceptance Test
  - 1.3.3 Test Bed & Tool Support
  - 1.3.4 Test Data Management
- 1.4 Management
  - 1.4.1 Project Management
  - 1.4.2 Administrative Support
  - 1.4.3 Management Tools
  - 1.4.4 Management Reviews
  - 1.4.5 Management Training
- 1.5 Product Assurance
  - 1.5.1 Configuration Management
  - 1.5.2 Library Operations
  - 1.5.3 Interface Control
  - 1.5.4 Data Management
  - 1.5.5 Quality Assurance
  - 1.5.6 Quality Control
- 1.6 Operations and Support
  - ...

Tool 2: PERT charts

Source: Adapted from Blum, 1992, p439
see also: van Vliet pp193-6



Notation

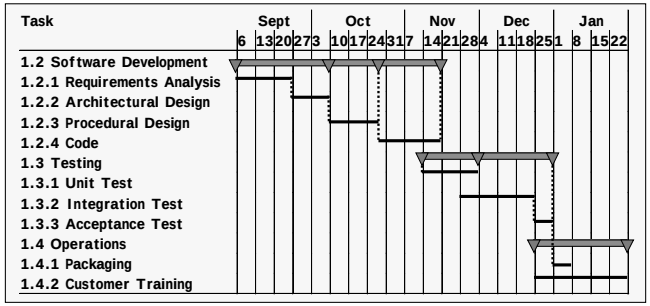
- Nodes indicate milestones
- Edges indicate dependencies
- Edges are labelled with time to complete

Shows Critical Path

- Longest path from start to finish
- any slippage on the critical path will cause project delay

see also: van Vliet pp195-6

Tool 3: Gantt Charts



Notation

- Bars show duration of tasks
- Triangles show milestones
- Vertical dashed lines show dependencies

Shows high level view of whole project

Tool 4: Meetings

Source: Adapted from Pfleeger, 1998, 92

Meetings are expensive

- E.g. 8 people on $40k. Meeting costs $320 per hour

Meetings are necessary

- Can save money by averting misunderstandings and coordination errors

Time wasters:

- Purpose of meeting unclear
- Attendees unprepared
- Essential people missing
- Discussion gets sidetracked
- Dominance by one or two people
- argumentative
- Decisions not followed up on

Meetings advice:

- Announce details in advance
  - who should attend
  - start and end times
  - goals of meeting
- Written agenda, distributed in advance
- Identify a chairperson who:
  - keeps the discussion on track
  - resolves arguments
- Identify a secretary who:
  - keeps track of decisions taken
  - records action items
  - ensures action items are carried out
- Associate a responsible person with each action item

Risk Management

Two Parts:

- Risk Assessment
- Risk Control

Source: Adapted from Blum, 1992, p441-447
see also: van Vliet pp189-191

Definitions

- Risk Exposure (RE) = p(unsat. outcome) X loss(unsat. outcome)
- Risk Reduction Leverage (RRL) = $(RE_{before} - RE_{after})$ / cost of intervention

Principles

- If you don't actively attack risks, they will attack you
- Risk prevention is cheaper than risk detection
- Degree and Cause of Risk must never be hidden from decision makers

"The real professional … knows the risks, their degree, their causes, and the action necessary to counter them, and shares this knowledge with [her] colleagues and clients" (Tom Gilb)

Top Ten Risks (with Countermeasures)

Source: Adapted from Boehm, 1989 see also: van Vliet p192

- **Personnel Shortfalls**
  - use top talent
  - team building
  - training
- **Unrealistic schedules and budgets**
  - multisource estimation
  - designing to cost
  - requirements scrubbing
- **Developing the wrong Software functions**
  - better requirements analysis
  - organizational/operational analysis
- **Developing the wrong User Interface**
  - prototypes, scenarios, task analysis
- **Gold Plating**
  - requirements scrubbing
  - cost benefit analysis
  - designing to cost
- **Continuing stream of requirements changes**
  - high change threshold
  - information hiding
  - incremental development
- **Shortfalls in externally furnished components**
  - early benchmarking
  - inspections, compatibility analysis
- **Shortfalls in externally performed tasks**
  - pre-award audits
  - competitive designs
- **Real-time performance shortfalls**
  - targeted analysis
  - simulations, benchmarks, models
- **Straining computer science capabilities**
  - technical analysis
  - checking scientific literature

Principles of Measurement

Source: Adapted from Blum, 1992, p457-458
see also: van Vliet pp104-9

Types of Metric

- algorithmic vs. subjective
- process vs. product

Good metrics are:

- simple (to collect and interpret)
- valid (measure what they purport to measure)
- robust (insensitive to manipulation)
- prescriptive
- analyzable

5 types of scale

- nominal (=, ≠ make sense; discrete categories)
- ordinal (<, >, =, make sense; e.g. oven temps: cool, warm, hot, very hot)
- interval (+, -, <, >, = make sense; e.g. temperature in centigrade)
- ratio (x, ÷, +, -, <, >, = make sense; e.g. temperature in Kelvin)
- absolute (a natural number count)

"You Cannot Control What You Cannot Measure"

Suggested metrics

- Plot planned and actual staffing levels over time
- Record number & type of code and test errors
- Plot number of resolved & unresolved problem reports over time
- Plot planned & actual number of units whose V&V is completed over time:
  - a) design reviews completed
  - b) unit tests completed
  - c) integration tests completed
- Plot software build size over time
- Plot average complexity for the 10% most complex units over time
  - (using some suitable measure of complexity)
- Plot new, modified and reused SLOCs for each CSCI over time
  - SLOC = Source Lines Of Code (decide how to count this!)
- Plot estimated schedule to completion based on deliveries achieved
  - (needs a detailed WBS and PERT or GANTT chart)

Source: Adapted from Nusenoff & Bunde, 1993

Summary

Project management is difficult

First Plan the project

- Requires Work Breakdown Structure
- Requires cost and effort data

Then identify risks

- Identify risk mitigation strategies
- Try for risk prevention

Keep Measuring Progress

- Choose metrics that help track progress towards goals
- Choose metrics that give early warning about risks

References

van Vliet, H. "Software Engineering: Principles and Practice (2nd Edition)" Wiley, 1999.

- van Vliet organizes this material differently from the way it is presented here, and provides a lot more detail on some aspects (especially people management and cost estimation). Chapter 2 provides a brief but excellent intro. Chapters 5, 6 and 8 are definitely worth reading at this stage in the course.

Blum, B. "Software Engineering: A Holistic View". Oxford University Press, 1992.

Pfleeger, S. "Software Engineering: Theory and Practice". Prentice Hall, 1997.

Nusenoff, R. and Bunde, D. "A Guidebook and a Spreadsheet Tool for a Corporate Metrics Program". *Journal of Systems and Software*, Vol 23, pp245-255, 1993.

Boehm, B. "Software Risk Management". IEEE Computer Society Press. 1989.

MPIAT - Mars Program Independent Assessment Team Summary Report, NASA JPL, March 14, 2000.
(available at http://www.nasa.gov/newsinfo/marsreports.html)