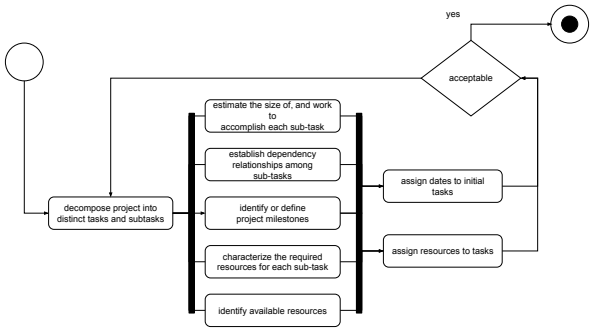
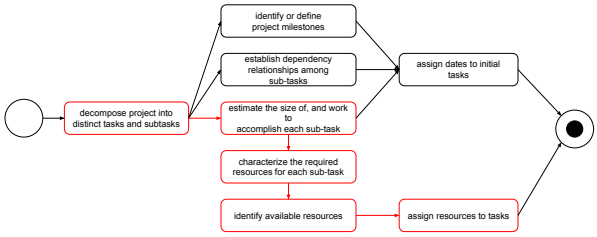
Scheduling and Monitoring

- Project Scheduling
  - task dependency (PERT) analysis/charts
  - staffing levels (PNR curves)
  - time lines (Gantt Charts)
- Project Status Tracking
  - Milestones
  - Earned Value Analysis
  - Scrum points and velocity
- Causes of Slippage and Failure

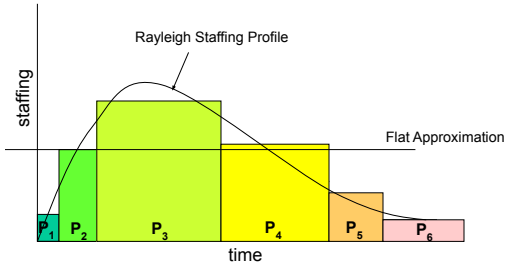
Project Scheduling



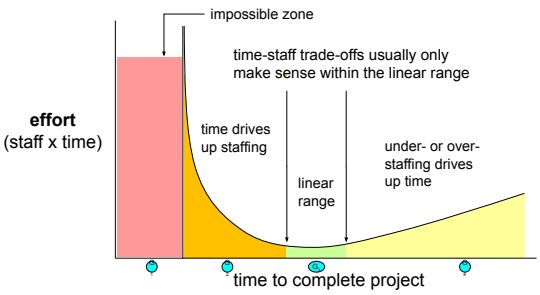
Dependencies (PERT charts)



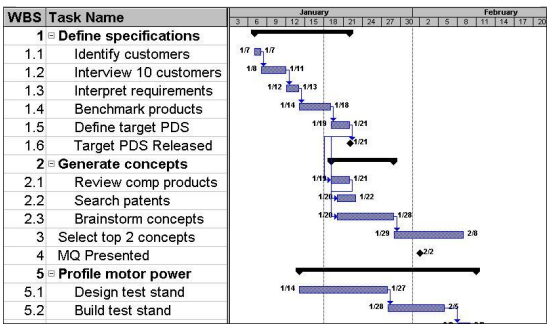
Rayleigh Staffing Profile Curve



PNR Effort/Time Curve
(there is an optimal size and time)

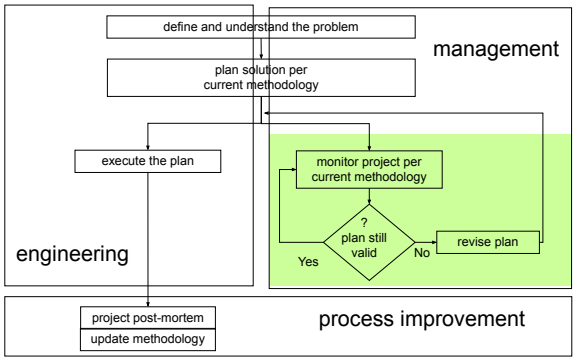


Timelines (Gantt charts)



WBS	Task Name
1	Define specifications
1.1	Identify customers
1.2	Interview 10 customers
1.3	Interpret requirements
1.4	Benchmark products
1.5	Define target PDS
1.6	Target PDS Released
2	Generate concepts
2.1	Review comp products
2.2	Search patents
2.3	Brainstorm concepts
3	Select top 2 concepts
4	MQ Presented
5	Profile motor power
5.1	Design test stand
5.2	Build test stand

Project Management 1A



Project Status Tracking

- [stand-up] what is each person doing
  - what tasks are they have recently finished
  - what tasks they expect to complete next
  - what obstacles they are encountering
- [management] how is project progressing
  - are resources allocated according to plan
  - is progress proceeding according to plan
  - does someone need a little help
  - does the plan need to be revised
    - changes in problem, resources, approach

Project Milestones

- Specific and Measurable
  - an objectively ascertainable moment
  - avoid subjective assessments
- Relevant measures of project progress
  - goals achieved, work completed
  - not merely hours of work done
- Timely (relatively closely spaced)
  - enable fine-grained progress tracking
  - accurate assessment of work state and rate

A Common Status Discussion

mgr: *Where are we on the app-server?*
engr: I'm about 80% done.
mgr: *You've been 80% done for six weeks!*
engr: The first ¾ was easy.
All the hard stuff is in the last ¼!

Q: How does a project get to be a year late?
A: One day at a time.

Fred Brooks, The Mythical Man-Month

Quantifying Progress

- task completions are obvious milestones
  - **specific**, **measurable**, **achievable**, **relevant**
- they may be poor measures of progress
  - not usually evenly spaced measures of work
  - may be too large for fine grained tracking
- we need a different kind of measure
  - to enable fine grained (e.g. daily) tracking
  - to enable meaningful schedule tracking
  - to enable meaningful budget tracking

Earned Value Analysis

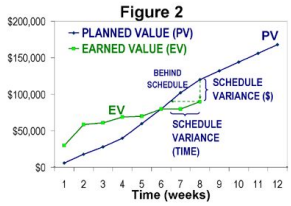
- construction size and effort estimates
  - yield an expected cost for each sub-task
  - this is the budgeted value of that sub-task
- the Earned Value of an effort
  - is the value of all the tasks **completed** so far
    - or … ¼ earned at start, ¾ earned at completion
    - or … partial value for progress (e.g. tests passed)
- Tracking Earned Value enables us to
  - assess project completion and speed
  - meaningfully assess cost-performance

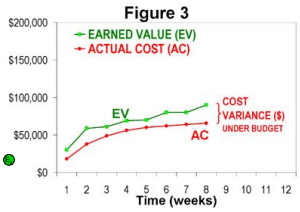
Tracking with Earned Value

Comparing Earned Value with Planned value enables us to determine …

(a) what fraction of the project is complete

(b) whether or not the work is on schedule.





Comparing Earned Value with Actual Cost enables us to

(a) determine whether or not we are within budget (on the work completed so-far).

(b) estimate the cost to completion, based on the performance so-far.

SCRUM Points

- relatively easy to estimate
  - developer-convenient unit: "best-case days"
  - a measure of work, <u>not a delivery date</u>
  - less misleading and arbitrary than "dollars"
  - small task estimation is easy and accurate
- excellent progress tracking
  - small tasks enable fine-grained tracking
  - a more linear measure of progress
- well correlated to product progress
  - only <u>accepted</u> features earn points

Sprint Velocity

- self-calibrating measure that includes:
  - productivity, overhead, bugs, competing priorities
- uncertainty is recognized and quantified
  - consistency of recent velocity measurements
  - convergence of backlog grooming/estimates
- enables better projections of completion
  - replaces optimistic promises with extrapolations
- enables better management
  - guides choice of what to accept in next sprint
  - highlights backlog, productivity, distractions
  - makes product owner a partner in development

Team Exercise

- Consider your P1-4 plans and results:
  - tasks not recognized or under-estimated
  - tasks that slipped, didn't meet expectations
- Consider your project management:
  - did you know what everyone was doing?
  - were problems promptly recognized?
  - how prompt and effective was response?
- In retrospect:
  - how well chosen were project milestones?
  - what else should have been called out?
  - what worked, what would you do differently?

Common Causes of Slippage

- poor or unstable requirements
- unrealistic schedules (poor estimates)
- "Scope Creep" (new input, lose focus)
- unanticipated construction problems
- unanticipated quality problems
- unanticipated integration problems
- external dependency issues
- unplanned distractions

Why Projects Fail

Rank	Challenged	Failure
1	Lack of user input	Incomplete requirements
2	Incomplete requirements/specs	Lack of user involvement
3	Changing requirements/specs	Lack of resources
4	Lack of executive support	Unrealistic expectations
5	Technological incompetence	Lack of executive support
6	Lack of resources	Changing requirements/specs
7	Unrealistic expectations	Lack of Planning
8	Unclear objectives	Didn't need it any longer
9	Unrealistic time frames	Lack of IT management
10	New technology	Technological illiteracy

Requirements 30% | **Planning 25%** | **Management 20%** | **Technological 7%**

Was uns nicht umbringt macht uns nur stärker.

Friedrich Wilhelm Nietzsche
Also Sprach Zarathustra

Final Exam

- When: Wed 5/12, 2-5PM
- Scope: learning objectives since mid-term
  (a few may spill over into first half)
- Length: 12 multi-part questions, +1 XC
- Format: same as midterm
- Difficulty: similar to midterm
  - mostly list, define, describe, justify, examples
  - a few "apply to this situation" questions

Feel free to email me with questions that arise while you are reviewing for final

Supplementary Slides

The Traditional Model

- generate a huge list of requirements
- development commits to a long schedule
- the requirements continue to change
- development hit with other emergencies
- work is not completed on schedule
- everybody is surprised
- destructive finger-pointing ensues

Work Breakdown

- hierarchical decomposition of work
  - independent sub-tasks that sum to the whole
  - can be based on either tasks or goals
  - both process- and problem-centric
- sub-tasks are fundamental unit of work
  - the granularity of work estimation
  - the granularity of assignment & scheduling
  - the granularity of task inter-dependencies
  - the basis for progress tracking
- granularity will evolve with the project

Task Dependencies

- input/output relationships between tasks
  - may be strict
    - e.g. system test starts after component integration
  - may allow overlap
    - e.g. design can start one month after architecture
- process-mandated pre-requisites
  - support training must precede beta shipment
- these dictate the order of scheduling
  - they also determine the (longest) <u>critical path</u>