

Class 4 CS540

Gregg Vesonder
Stevens Institute of Technology
Copyright 2005 Gregg Vesonder

Class 4

1

Roadmap- Class 4

- Clarifications from last class
- Journal entry
- Brooks - Mythical Man Month
- Scheduling
- MULTICS Case Study
- Questionnaire Design & multi class survey
- Software Project Reviews + Arch Reviews
- Review of 1st 4 lectures for test
- Readings next week, chapter 2, pp 39-48 in BY, chapters 3-5 in Brooks
- Reading for next class: Brooks Chapter 6, [BY 393-398](#)

Class 4

2

Class 4 Clarifications

- none

Class 4

3

Calendar -Key Dates

- **October 3rd - first test**
- Oct 11th - class on Tuesday not Monday!
- November 7th - second test
- November 21st - log books due
- December 12th - final exam

Class 4

4

Log Book

- The Schedule/Estimation Dance
- Your Entry

Class 4

5

Brooks, Chapter 3

- "small is beautiful, big is necessary" - Bob Factor
- Programmer productivity can vary as much as an order of magnitude and the space and speed of the code can be as much as 5 times better
- Systems should be built by as few people as possible, a small sharp team of < 10
- However OS 360 (1963-'66) required 5,000 staff years. Even with order of magnitude improvement, team of < 10 would take over 50 years!

Class 4

6

Chapter 3 (cont'd)

- Harlan Mills: surgical team rather than hog butchers! One does the cutting and the rest do support
- The roles:
 - Surgeon = chief programmer/boss lots of experience (10+ years) and application knowledge
 - Copilot = shares in the design as thinker, discussant and evaluator. Knows all the code. (My buddy system)
 - Administrator- money, people, space, machines, bureaucracy
 - Editor- surgeon generates documentation, ed does rest
 - Secretaries
 - Program clerk- maintains all the records in a programming project
 - Toolsmith - tools/libraries that need to be built or adapted
 - Tester
 - Language lawyer
- 10 people, 1 mind, 200 people only 20 minds have to be coordinated

Class 4

7

Brooks, Chapter 4

- Conceptual integrity - one set of design ideas
- Ease of use (and understanding) = conceptual integrity
- Good features and ideas that do not integrate with a system's concepts should be omitted, BUT if this happens too frequently - redesign
- Architect does not steal all the fun, cost/performance is implementers task
- Constraints on the architect are good
- Brooks multimillion dollar mistake - the appeal of putting everyone to work!

Class 4

8

Brooks, Chapter 5

- First system spare and clean with deferred ideas
- Second system dangerous, tendency to overdesign (true for me)
- Leap year example -- 26 bytes permanently resident vs 1 manual op every 4 years
- Overlays versus compilers
- Beware of your second system!

Class 4

9

More on Schedules (appreciate it more)

- Before Project
 - Identify tasks
 - Estimate duration
 - Allocate resources
 - Schedule when task starts
- During project monitor and adjust, assistance of project manager - must know task
- Gantt chart is more of a milestone chart, can be used to track progress against time spent

Class 4

10

PERT Charts

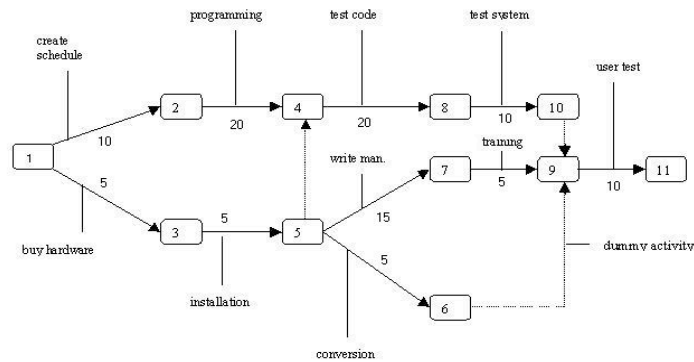


Fig. 1:
PERT Chart

- * Numbered rectangles are nodes and represent events or milestones.
- * Directional arrows represent dependent tasks that must be completed sequentially.
- * Diverging arrow directions (e.g. 1-2 & 1-3) indicate possibly concurrent tasks
- * Dotted lines indicate dependent tasks that do not require resources.

From <http://searchsystemsmanagement.techtarget.com>

Class 4

11

Multics

- *Multiplexed Information and Computing Service*
- Joint project of BTL, GE & MIT in Fall 1964 (NJ, MA, AZ)
- Effort to design a computer utility, a computer community
- Goals:
 - Convenient remote terminal use
 - Continuous operation analogous to power and telephones
 - Configurable capacity that could be dynamically rearranged
 - Internal file system with high, trusted reliability - only place users need store programs and data
 - Ability for users to selectively share information
 - Ability to store information hierarchically
 - Ability to support a wide range of apps from numerical to word processing
 - Multiple programming environments and interfaces w/in same system
 - Ability to evolve

Class 4

12

Multics Phases

- Phase 1, Fall 1964
 - Writing of detailed module spec, 3K pages
 - PL/I was the language
- Phase 2, early 1967
 - Module implementation and unit checkout
 - Huge discrepancies between predicted and actual performance
 - Not caused by poor programming
 - Specs of less important features were introducing complexity
 - Choice of modularity and interfacing were awkward
 - Rediscovered simplicity was the most important property of algorithms
 - Devised experimental, iterative view
 - Spec and design review
 - Iterative design and implementation: build, evaluate, redesign, evolve

Class 4

13

Multics Phase (cont'd)

- Phase 3 late 1968
 - System programmers used system while developing it -- "eat your own dog food"
- Phase 4 October 1969
 - Available for general use, 500 subs
 - 22 hrs/day, 7 days a week
 - 55 simultaneous users 1-5 sec response time
 - 300K code, 1500 modules
 - Through 1969 150 staff years, 167 LOC/month
- 50 more staff years in the next 2 years for maintenance

Class 4

14

Comparison to Microsoft/UNIX (from Lucovsky)

- NT
 - 2/89 coding begins (slated for 18 months)
 - 7/93 NT 3.1 ships
 - 9/94 NT 3.5 ships
 - 5/95 NT 3.51 ships
 - 7/96 NT 4.0 ships
 - 12/99 Win 2000
- UNIX
 - '69 coding begins
 - '71 1st edition
 - '73 4th edition (in C)
 - '75 5th edition - basis for BSD
 - '79 7th edition
 - '82 System III
 - '84 4.2 BSD

Class 4

15

Multics

- Used a systems programming language -- only 250 of the 1500 modules were written in assembler
- Arpanet, user graffiti board, memorandum formatting
- "... it was not properly appreciated then how many things could go wrong -- or in effect -- how much of the state of the art was being pushed"
- PL/I is a strength but a simpler language should have been chosen or development of a new language, e.g., C
- Iterative nature of design process was not appreciated - breadboarding

Class 4

16

Issues (apparent later)

- Different organizational goals/ 3 organization cooperation
 - R&D in CS
 - Commercial product development
 - Useful computational service
- > 2 year project
 - Turnover of developers, 6-9 mos to bring a person on board
 - Management turnover and education in 3 companies
 - Remedies included careful design, enforcement of programming standards, adherence to software engineering practices

Class 4

17

Issues cont'd

- Misestimated schedules - got better with time
- Imbalanced resource - add more hardware even if it reduces staff (still not followed)
- Necessity for 3 company consensus resulted in unnecessary generality

Class 4

18

Management Tools

- High level systems programming language gained factor of 5 to 10
- Structured programming = good engineering practice
- Design review - "design leadership exposed rather than imposed" but unique environment
 - Process: position paper, design doc, review by peers, consensus
 - Code should be read by one peer (to debug or not to debug)

Class 4

19

Management Tools cont'd

- Test environments are good
- Production techniques
 - Eat your own dog food
 - Make changes in minutes - obstacle updating doc
 - New systems once or twice a week
- Management and performance tools - Pert mixed results, the journey was the value

Class 4

20

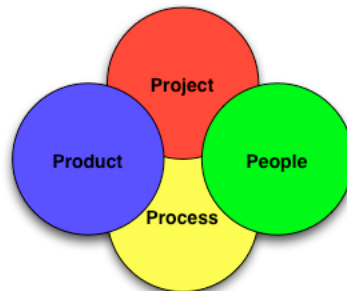
General Observations

- Size is a liability - communication, filtering by staff and biasing by management
- Inhomogeneity of technical understanding .. Managers should code and programmers should understand big picture
- Decision costs - contingency planning and insufficient effort
- Psychology
 - Must see progress - use frequent milestones that are unambiguous and significant
 - Immaturity of programmers
- Value of early performance warnings

Class 4

21

The Big Four



Class 4

22

On Gurus

- Twenty times more productive than average developer
- Only 1% of all developers are gurus, but 50% think they are
- How to recognize a guru:
 - Broad competency in hdw, sfw, operating systems and applications (system admin)
 - Profound depth in 1 or more areas
 - Peer recognized
 - "Go to" folks - teaching, info advice
 - Opinionated, often go beyond their skills
 - Overall picture of system but can go to highly detailed level
 - Focused on problem solving
 - Broader interests from anime to zoology

Class 4

23

Gurus - 2

- Impatient with fools, non technical managers and folks who do not read documentation
 - No acceptance of status quo (+/-)
- Guru attraction:
 - High quality software organization - publicity
 - Opportunity for independent work - Google 1 day/week
 - Pinball - best & newest projects, best tools

Class 4

24

Questionnaire Design

- Much of the information derived from "A brief guide to questionnaire development" by Robert Frary
- Questionnaire preparation:
 - Focus on the information desired, write as few questions as possible, avoid "nice to know"
 - Have an analysis plan - arrange for a manageable number of ordinally scaled variables
 - Prototype it, including post questionnaire critique
 - Field trial of mailed questionnaire - response rate, question applicability (if trial shows same response for everyone, may be redundant)
 - Question performance

Class 4

25

Q2

- Avoid open ended questions - try blank completion
 - If all else fails, place open ended questions at the end, with a predetermined scoring strategy to max inter-rater reliability
- Objective questions
 - Avoid "other" - exception is if categories are clear-cut, few in number and some responders might feel uncomfortable
 - Avoid category proliferation - get the information you need. If you are interested in Windows XP users only, ask Win XP or other.
 - Ordering of categories - go from lower level to higher level, e.g., never, seldom, occasionally and frequently

Class 4

26

Q3

- Scale points
 - Avoid scale point proliferation ... 4 to 5 points is usually sufficient and never go for more than 6 or 7. - JNDs, just noticeable differences. This is true even if only the endpoints are labeled.
 - The use of a scale midpoint (odd number) be careful. May indicate: ignorance, lack of cooperation, reading difficulty, reluctance to answer, inapplicability. In most instances there will be a high number of neutral (midpoint) responders.
 - Without a midpoint (neutral) response, responders may avoid responding, try:
 - Encourage skipping if no response
 - Word responses so that a firm stand may be avoided, but a direction indicated, "tend to disagree"
 - Include options clarifying reluctance - not applicable, prefer not to answer
 - However, sometimes a midpoint is justified, "the amount of homework for this course was: too little, reasonable, too much"

Class 4

27

Q4

- Response category language:
 - "strongly agree" is redundant, use agree
 - Agree/disagree vs agree, tend to agree, ...
 - Read your questions carefully and test for hidden assumptions
- Avoid ranking if possible, and at most rank 6 or less things
- Apple pie phenomenon - rating everything at one end of the scale - have them rate both positive and negative statements

Class 4

28

Q-1

Demographics:

*Education:

- Undergraduate major(s)
- Graduate major(s) (note if in progress)

*Age:

- 30 or younger
- 31 to 40
- 41 to 50
- 51+

Class 4

29

Q-2

• Development Experience:

• 1) My first computer language was _____

• 2) My first Object Oriented language was _____

• 3) During the past year I developed:

- -0 LOC
- -100 LOC
- -1000 LOC
- -10,000 LOC
- -more than 10000 LOC

• 4) During my career I have developed:

- -0 LOC
- -100 LOC
- -1000 LOC
- -10,000 LOC
- -more than 10000 LOC

• 5) I have been involved in (check all that apply):

- -requirements
- -architecture
- -design
- -coding
- -testing
- -build management
- -system administration

Class 4

30

Q-3

- Project Experience:
- 1) My current project is predominantly software, hardware, other or no current project.
- 2) The current size of the staff on my current project is:
 - -1
 - -5
 - -10
 - -20
 - -50
 - -100
 - -more than 100
- 3) The current staff that I manage is:
 - -0
 - -1
 - -5
 - -10
 - -20
 - -50
 - -100
 - -more than 100

Class 4

31

Q-4

- 4) I have been involved in an Object Oriented project(y/n)
- 5) I have been involved in an Open Source project(y/n)
- 6) My current work project uses Open Source(y/n)
- 7) My current personal project uses open source(y/n)
- 8) My guess of the Capability Maturity Level of my team is (1-5)
- 9) I use Agile methods (y/n)
- 10) I write code for fun (y/n)

Class 4

32

Architecture Reviews

- Adapted from Starr & Zimmerman(2002) and personal experience
- These reviews have been around for at least 11 years/ 500+ reviews at AT&T and Lucent (SARB, Systems Architecture review Board)
- Stresses architecture and evaluating it early in the project
- Architecture in this context is viewed as a solution to a problem for a client and should cover a broad range from cost to client needs
- After you establish an architecture:
 - Decide what and when to test
 - Establish success criteria
 - Make decisions based on your findings

Class 4

33

How and what do you address?

- Code inspections examine source code, Design reviews examine models, modules and interfaces
- However a single document or artifact rarely captures architecture
- Architecture is reviewed as an interactive oral presentation which varies from a chalk talk to a few overheads
- Review should be done before contracts are signed or announcements made

Class 4

34

The Architecture Problem Statement

- 1-4 pages in length
- It is criteria to test the architecture and includes:
 - What project must do (functional aspects)
 - What it must be (constraints)
 - Economic constraints (time, schedule, money)
 - How system relates to past (backward compatibility), present and future (evolvability)
 - Unique aspects of the problem (e.g. risks)
- NOT requirements document

Class 4

35

Preparing for the review

- Problem statement should be approved by architect, client, development manager and other stakeholders (customer representatives, system engineers/analysts). Also should iterate with review coordinator.
- Review team selected (review team << project team):
 - Review leader - technical person trained in SARB and facilitation and responsible for writing review report
 - Angel - manager not associated with project, interface between review team and project management
 - Reviewers - several internal tech experts or outside consultants with relevant expertise, including: design, technology, management, domain knowledge, wild card reviewer

Class 4

36

Review process

- Pre-review meeting a week before review
- Review meeting (2-3 days):
 - Begins with problem statement - what success means
 - Overview of proposed solution
 - Presentation on "ilities" reliability, maintainability, ...
 - Reviewers ask questions and record issues and strengths on snow cards "out in the open" "reconfigurable"
 - Review team has caucus for several hours
 - Snow cards (50-200) arranged by functional categories (requirements, management, interfaces, ...) and severity
 - Process provides overall assessment of chances for success and key messages

Class 4

37

Snow Cards



38

Review process (cont'd)

- Readout:
 - Encourage project team to invite clients and stakeholders
 - Provides overall assessment, strengths and issues
 - Elicits feedback for Project team
- Followup:
 - Within 2 days snow cards are numbered, recorded and sent to project team
 - If there are immediate issues angel and leader compose letter to client
 - Within 2 weeks detailed writeup of key issues and strengths
 - Within 2 months review leader provides report to SARB, assessment, key issues and review critique
- SARB creates annual report of key issues and trends

Class 4

39

SARB Heuristics

- Requires cultural change:
 - Cost to project, reviewers
 - Project team safety
 - Ubiquity
- Key steps:
 - Combine executive and grass roots support
 - Establish and maintain self reinforcing support for experts
 - Recognize contributors
 - Maintain safety for team and reviewers
 - Start small

Class 4

40

On Team Safety - Key Element

- Protect individuals from blame
- Keep no secrets from project team
- Treat all findings as confidential:
 - Specific review findings only reported to team and board
 - Written report is property of project team
 - Only general/anonymous findings are part of annual report
- Review team are partners and colleagues do not have role as policeman

Class 4

41

Arch Review Payoff

- Average review pays back 12 times its cost
- Reduced development effort and interval - find defects early
- Higher product quality
- Lower product cost
- Faster less costly product evolution (planned)
- Company wide learning - annual report
- Yes, projects were canceled after reviews and the attitude of the project team was often surprising

Class 4

42

More General Review Heuristics

- When do you review -- early and often, note arch review is very early
- Roles in the review:
 - Author - responsible for updating entity afterwards (e.g., actual architect, developer)
 - Moderator - enforces roles and responsibilities -- manages meeting and is "neutral"
 - Scribe - accurately documents review points (active role, sometimes shared role with monitor), distributes drafts and with moderator and author publishes final review report
 - Reviewers - find issues, try not to solve at that point (can volunteer to help) constructive
- ALL IN THE TONE

Class 4

43

Active Reviews

- Parnas suggestion to get folks engaged:
 - Ordinary assumption, smaller number of comments during design review indicates higher quality work - NOT!
 - Parnas suggests it indicates a superficial review, besides effective review is not reading a document end to end but using it to
 - Evaluate correctness of document
 - Usefulness of it for future tasks
 - Give reviewers questions on the Design Document so they have to use information - Active Review
 - And a rigorous active design review process should include questionnaires written by all groups needing information, not just authors

Class 4

44

Project Reviews

- From Pinto, J.K. & Kharbanda, O.P. "How to fail in project management (without really trying)."
- (an affirmation)
- Project management techniques are becoming universal - should examine failures
 - All corporations make mistakes
 - Potential for learning
- A quick guide to ruining your project.

Class 4

45

Project Reviews-2

- **Ignore the project environment, including the stakeholders**
 - Receptivity of the organizations internal environment to the proposed project
- **Push a new technology to market too quickly**
 - New and unknown risks
 - Strong likelihood of inadequate testing (is there ever enough?)
 - Performs in positive expected ways?
 - Is the market/environment ready?
 - Genichi Taguchi - Quality as avoiding "the loss the product causes society after it is shipped."

Class 4

46

Project Reviews-3

- **Don't bother to build in fallback options**
 - Test of project managers flexibility and capacity to respond rationally, looking for opportunities
 - One key is "what-ifs" searching out likely problem areas and doing it upfront
 - Experience
 - My worry beads
 - Risk analysis
- **When problems occur, shoot the most visible**
 - A public sacrifice/execution helps productivity (not!)
 - Brooks showed personnel changes in mid stream can be disastrous - learning curve/context
 - Japanese, go hard on the problem, soft on the people, "fix the problem, not the blame"
 - Paranoia kills flexibility

Class 4

47

Project Reviews - 4

- **Let new ideas starve to death through inertia**
 - XEROX PARC, alto personal computer, 1972 -> Steve Jobs in 1979
- **Don't bother conducting feasibility studies, prototypes**
 - Doing upfront homework - sufficient time for risk analysis, cost analysis, stake holder analysis, ...
 - Ready, fire, aim leads to incredible waste

Class 4

48

Project Reviews - 5

- **Never admit a project is a failure**
 - Recognizing when it is no longer sensible to continue
 - There's a pony somewhere!
 - Very difficult - lots of emotions, sunk costs, psychic energy
 - Throw more money at it - buy success
 - Company is sometimes unwilling, even when top management knows it is in trouble, "image, pride"
 - Escalation of commitment to a bad decision - managers are loath to admit to a bad decision
 - Adding resources can help, but first step is conducting a realistic analysis as early as possible (arch reviews)
 - Bernstein corollary - what is worse is a slightly successful project! You continue to pay

Class 4

49

Project Reviews - 6

- **Over manage projects and their teams**
 - Size of organization and management works against you
 - Bureaucracy, "staff infection" vs. lean and mean
- **Never, never conduct project failure reviews**
 - Swept under rug, written off as flukes
 - Mistakes are a natural side effect of new ventures
 - Effective dept heads can help project managers understand what happened
 - Ignore it and consider the problem as new/unique each time
 - Learning from mistakes is not a luxury it is a duty - make time

Class 4

50

Project Reviews -7

- **Never bother to understand project tradeoffs**
 - Make decisions based on rational insight
- **Allow political expediency and infighting to dictate crucial product decisions**
 - Emphasize parochial needs - not
- **Make sure your project is run by a weak leader**
 - Projects left to their own devices run toward entropy
 - Weak leaders are not a null effect, they are negative

Class 4

51

Project Reviews - Final Thoughts

- Failure is often a byproduct of risky ventures
- Past failure should not discourage us from future attempts
 - Do not become a victim either through failure to learn or failure to try again

Class 4

52

Thought Problems

- You are creating a new web browser and you would like to get user feedback. What is your plan for getting it?
- Your Vice President/General has read the Starr & Zimmerman paper and wants to initiate architecture reviews. She has assigned the task to you. Where do you begin?

Class 4

53

So Far

- Software Process Models
- Software Project Planning (woosh!)
- Requirements
- Estimation
- Risk Analysis
- Multics case study
- Architecture Reviews
- Questionnaire Design
- Next Time: Software Quality Assurance and the test

Class 4

54

Quiz review

- Short answers, fill in blanks, acronym expansion, 2-4 sentence essays, definitions, true/false and identification (multiple choice)
- 5 Chapters in Brooks
- All the class notes
- BY:
 - Chapters 1, 3 & 6, pp39-48

Class 4

55

Lecture Resources

- Pinto, J.K. & Kharbanda, O.P. "How to fail in project management (without really trying)." In D.J. Reifer (Ed.), *Software Management*, 6th Edition, IEEE Computer Society, 2002. ISBN: 0-7695-1100-7
- Starr, D. & Zimmerman, G. "A blueprint for success: Implementing an architectural review system." www.stqemagazine.com, 2002.
- <http://www.multicians.org>

Class 4

56