Project Management and Embedded Systems

Christopher Brooks, PMP

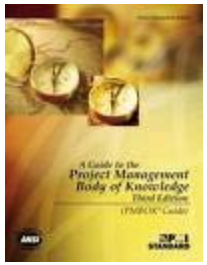
Center for Hybrid and Embedded Software Systems (CHESS) Executive Director

Feb. 7, 2012, EECS 149: Introduction to Embedded Systems

Christopher Brooks

- I’m a release engineer, training electrical engineers in the art of software engineering.
- I’ve worked with Professor Edward A. Lee since 1992, first on Ptolemy Classic (C++) and now on Ptolemy II (Java).
- I’ve taken undergrad CS classes at Berkeley
- In 2009, I became a Project Management Professional (PMP) via UC Extension
  - 11 classes over 4.5 years, 330 hours of lecture

PMI: Project Management Institute

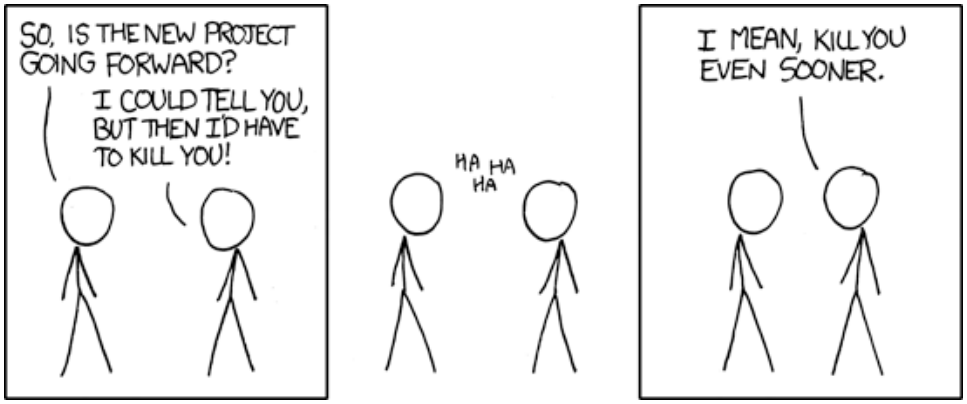


- "*A Guide to the Project Management Body of Knowledge* " (aka PMBOK)

- PMI Certification
  - *Certified Associate in Project Management (CAPM)*
  - *Project Management Professional (PMP)*
  - *Program Management Professional (PgMP)*

Why does Project Management Matter?

- Many efforts are project based
  - Movies, electronic hardware, construction, weddings
- You will meet project managers and it is good to know when you being project managed.
- So you know when Bad Things are about to happen



Credit: xkcd.com

What is a project?

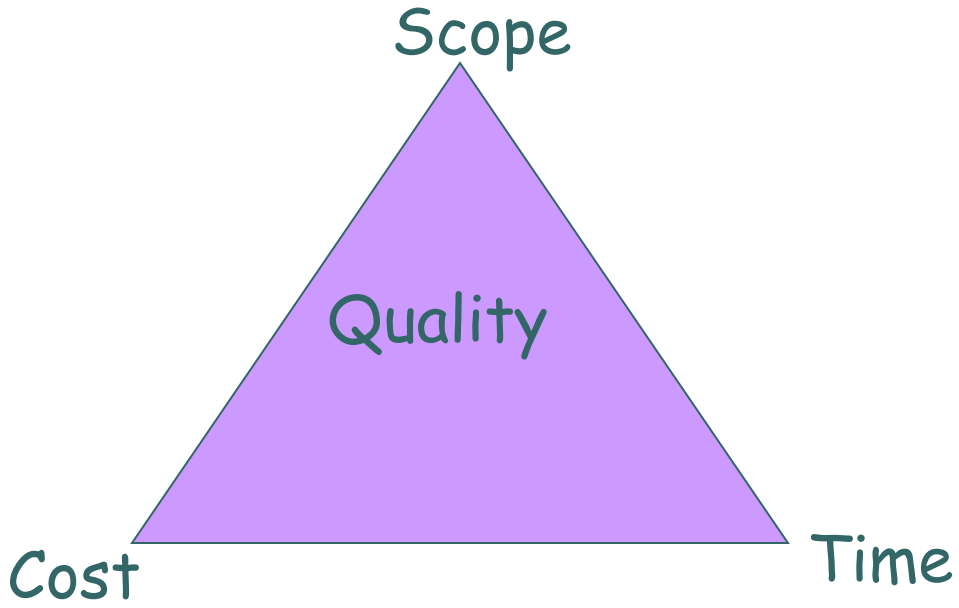
Q: What is a project?

A: "A temporary endeavor undertaken to create a unique *product*, *service* or *result*" -source PMBoK, p368

Note that a project is a **temporary** endeavor with a start and a finish. Avoid confusing continuous processes with projects.

Project Management

- Project management is?
- The art of managing projects to a successful completion.
- The Project Management Triple Constraint:



Time, Cost, Scope and Quality in the definition of a Project

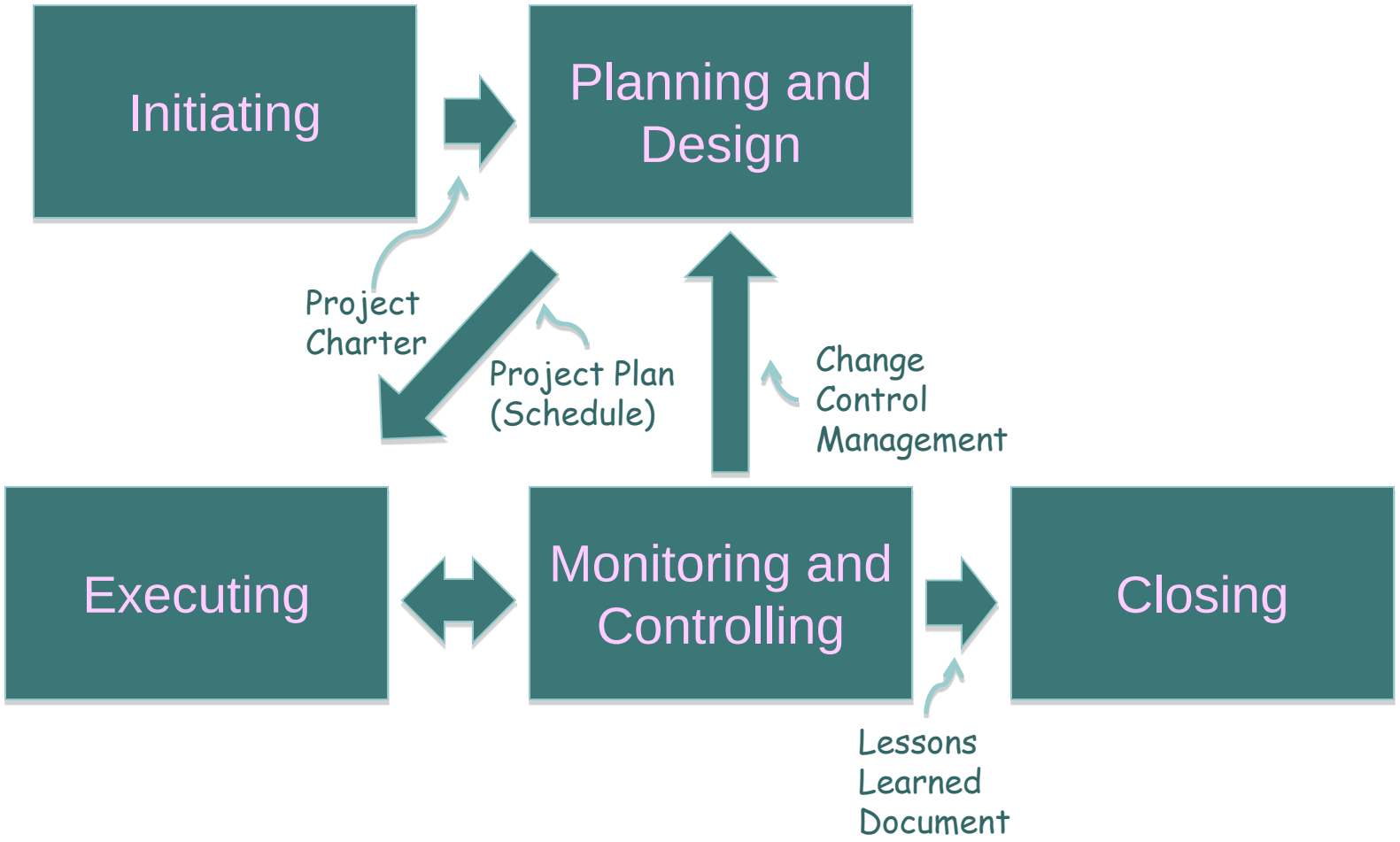
Q: What is a project?

A: "A temporary endeavor undertaken to create a unique *product*, *service* or *result*"

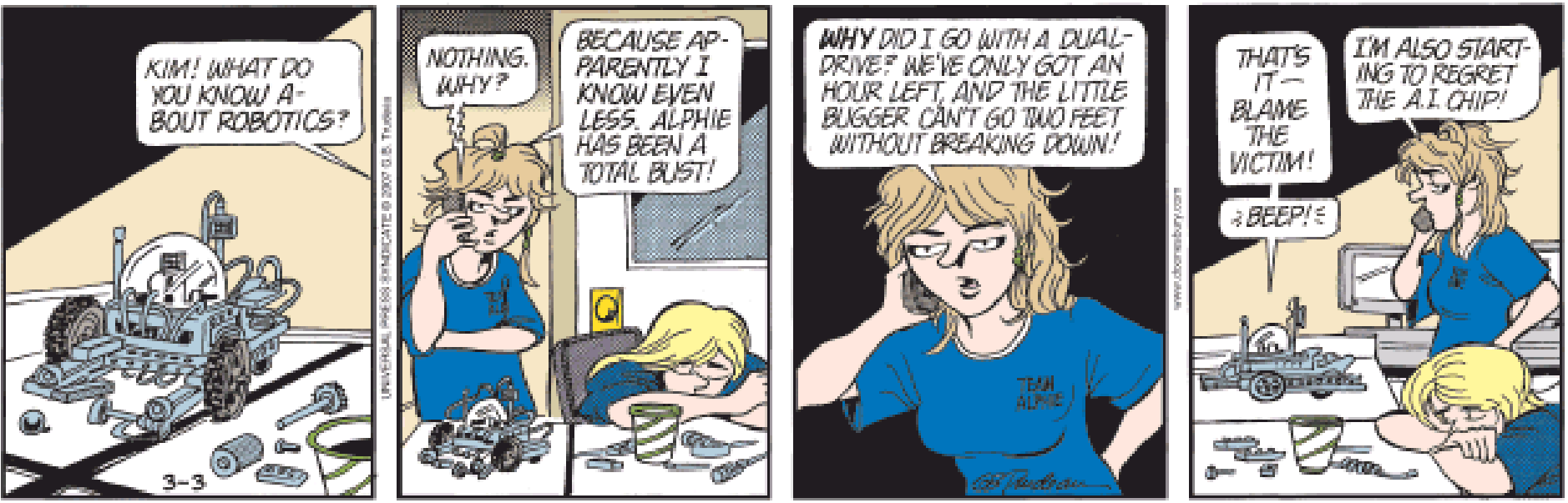
Where are time, cost, scope and quality?

Quality is missing from the definition!

Project Management Phases



How to have an unsuccessful project

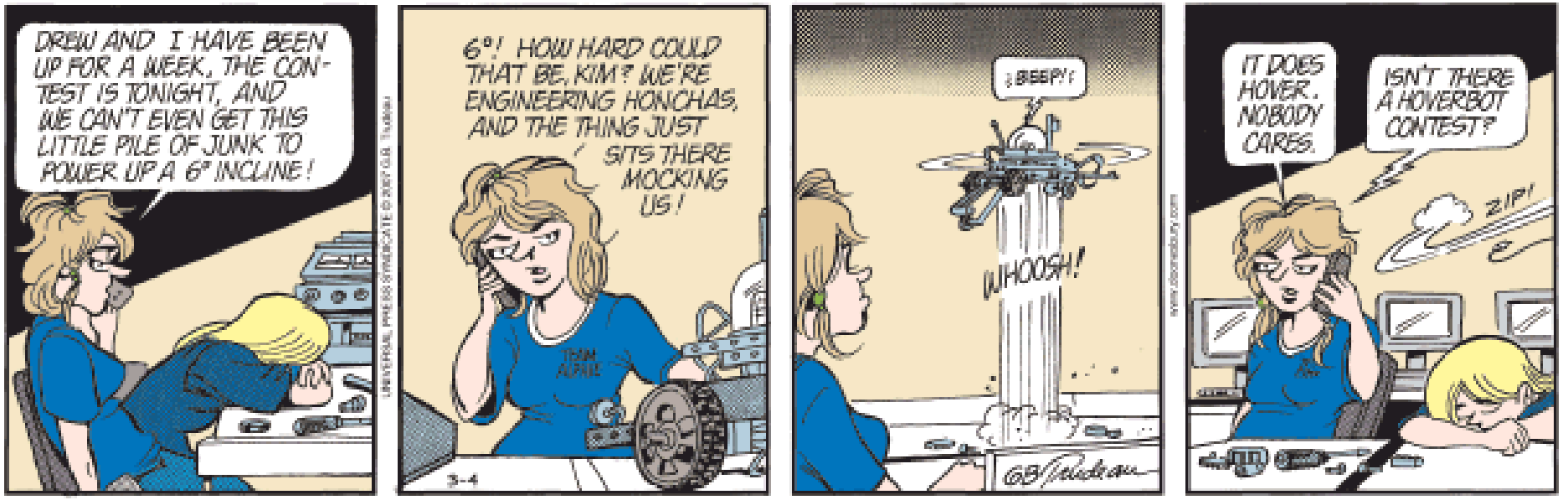


Copyright © Garry Trudeau

- What happened?
- Team dynamics: team mate is crashed out
- Scheduling: only an hour left?
- Overambitious: dual drive? AI Chip?

Solving Alex's Problems in terms of Scope, Time, Cost and Quality

- Team dynamics: team mate is crashed out
  - Time & Scope Problem, Quality suffers
- Scheduling: only an hour left?
  - Time & Scheduling Problem
- Overambitious: dual drive? AI Chip?
  - Scope: Vague requirements
  - Goldplating



Copyright © Garry Trudeau

- What's the problem?
  - Scope: Gold plating
  - Solution: Deliver the good stuff first

How I've messed up

- LED T-Shirt Project: 8x10 LEDs
- Tansy sewed the 80 surface mount LEDs
- I worked on the Ptolemy Software
- Problems
  - Missed deadline – bad planning
  - Data size too large for chip!
- Could have avoided with
  - Better planning: avoid surprises

Credit: Jason Madara/Craft Magazine

Project Management/Embedded Systems Problems

- Hardware and software designs affect each other
- But you don't know about problems until too late.
- Solutions include?
  - Software Simulation
  - Hardware Prototyping

How to successfully manage a project

- Develop a **one** page project charter
  - Get your sponsor to sign off
- Develop a time line with milestones
  - Work backwards
  - Describe deliverables
  - Break up deliverables into smaller deliverables
  - Assign deliverables to specific people
  - Associate Milestones with Deliverables
- Monitor progress
  - If you don’t look at the plan, then why bother?
  - Status reports can be email messages or meetings

EECS 149 Lab Project Management Requirements

Q: Where do you get your list of requirements for the Project Management Lab??

A: From the customer!

Q: Who is the customer?

A: In some ways, the faculty is the customer, but in the context of a class, the customer is the student. The faculty is the sponsor.

Q: Where do we find requirements?

A: Customer documents such as the website! The EECS149 website says:

Lab 7 (March 5, 2012): **Project management**: Students will finalize project teams and project definitions, construct a plan for the project with specific milestones and assign responsibilities to project participants.

Lab 7 Guide (includes prelab and checkout)

How to do the Project Management Lab?

2. Plan your Project: Follow Project Charter Instructions [5] to create a project charter. Ensure that you are reasonable in your objectives and deliverables, especially considering the experience of your group. i.e. Do not make a circuit board a major deliverable if no group member has any circuit design experience. On the same token, the scope of the project should match the size of the group and the goal be to produce an interesting and impressive product.

Project Management Lab (Continued)

Your charter should include:
(a) Overview
(b) Approach
(c) Objectives
(d) Major Deliverables
(e) Constraints
(f) Risk and Feasibility
Your charter should be one (1) page. The point is succinctly summarize your project for a busy manager.
Each group should submit one charter.

- The project charter
  - Will help define the project
  - Will help with the second part, the schedule of milestones and responsibilities

Project Charter

- Formalize the project with the sponsor
- Sections:
  - Project Overview
  - Project Approach
  - Project Objectives
  - Major Deliverables
  - Constraints
  - Risks and Feasibility
- In one page
  - Most people, especially busy people, will not read more. In fact, it has been shown that most people don't even read (hi mom!) slides.

Project Charter Example: Web Site

One page!

www-cad Project Overview

This project is to create a new web site for the CAD group faculty. The current website at www-cad.eecs has a very old look, it needs an update so that we can attract new students.

Project Approach

The project is a fairly small website based partly on a preexisting site, so we will use a classic waterfall approach with milestones. The project team will consist of the following people. I've estimated the maximum amount of time we can get from each person over the life of the project.
Kurt Keutzer (2 hrs week for 6 weeks) Ken Lutz (2 hrs/week for 6 weeks)
Brad Krebs (10 hrs/week for 6 weeks) Christopher Brooks (10 hrs/week for 6 weeks)
Allen Hopkins (5 hrs/week for 6 weeks) Carol Sitea (1 hr/week for 6 weeks)
The project sponsor is Professor Keutzer. Professor Keutzer is on sabbatical this semester, but we hope to get feedback from him on a continuing basis.

Project Objectives

- Update the look and feel of the website to a modern standard
- Provide access to student and faculty pages
- Provide access to active projects
- Provide access to summaries, downloads and key papers of inactive projects. The old pages of inactive projects should be archived.
- Provide a simple static listing of seminars. A more complex calendar and a search engine are deferred due to schedule constraints.

Major Deliverables

- A schedule along with time estimates.
- A prioritized list of features.
- An example of the main page so we can review look and feel.
- An archive of the old website
- The final website.

Constraints

Professor Keutzer would like to see the web site completed by mid-March: that is when students start looking at graduate schools. Developers might not have much time to work on this project. The project requires timely feedback from the faculty.

Risk and Feasibility

The primary risk is that the project takes too long to complete and we miss the mid-March opportunity. Another risk is that we complete the project too quickly and quality suffers. A third risk is that there are only so many resources available. By fast tracking, we can handle some of the tasks in parallel and avoid these risks. The project is definitely feasible if we roll out the website in stages.

Project Charter: Overview

- In a sentence, describe the project:

This project is to create a new web site for the CAD group faculty.

- Possibly include the business reason:

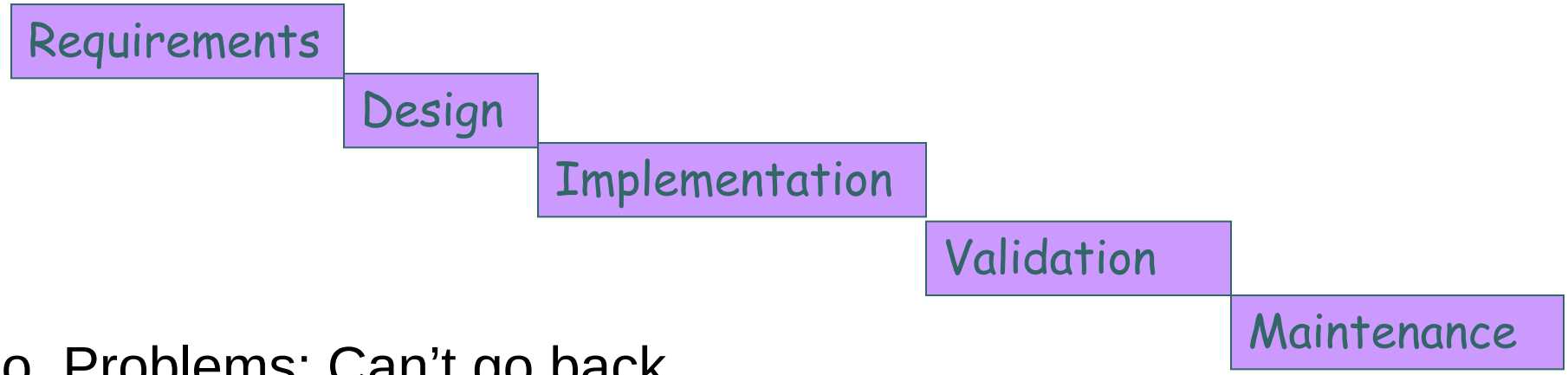
The current website at www-cad.eecs has a very old look, it needs an update so that we can attract new students.

Project Charter: Approach

- Describe your software development life cycle (SDLC):
- What's that?
  - Methodologies: Waterfall, V-Model, Iterative, Spiral, **Agile**, Lean, XP, Cleanroom, Rapid Application Development (RAD), Rational Unified Process (RUP), Scrum, Test Driven Development (TDD)
- List the team
- List the time commitments

Waterfall Method

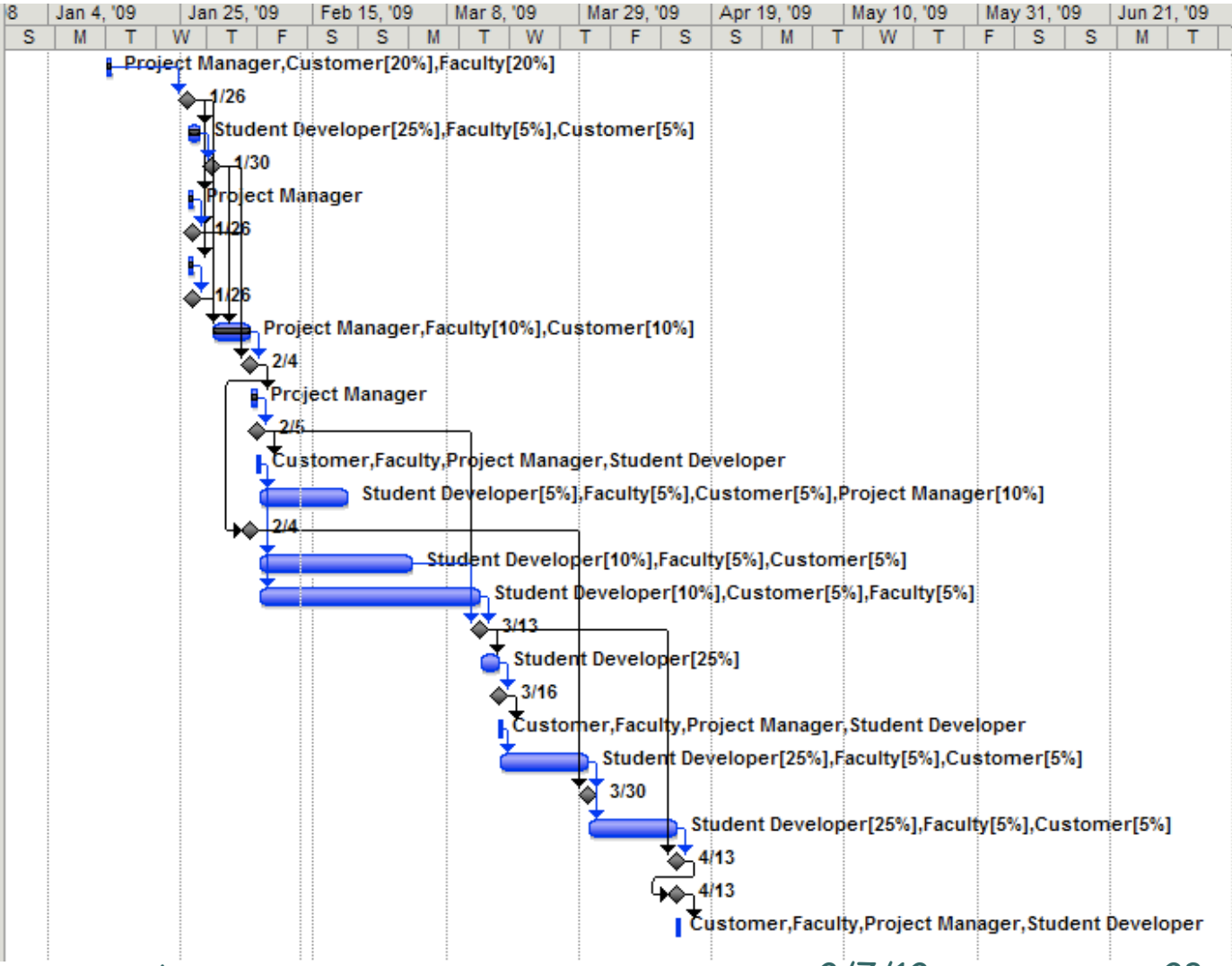
- Complete one phase before going on to the next [Royce, 1970]



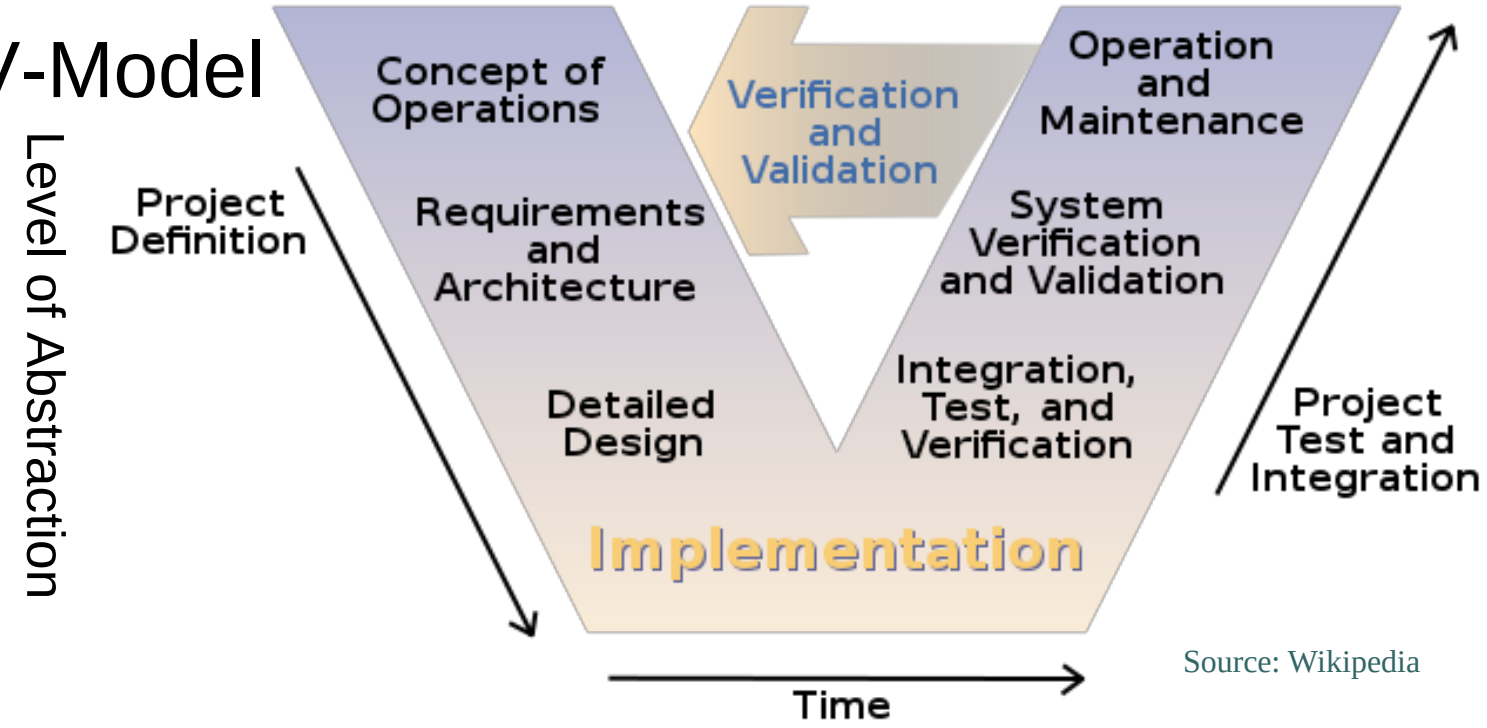
- Problems: Can't go back
- Good for extending known solutions

Waterfall Example: Microsoft Project Gantt Chart

	Task Name	Duration
1	Develop Project Charter	1 day
2	Project Charter	0 days
3	Develop User Interface Design	2 days
4	User Interface Design Doc.	0 days
5	Develop Risk Mgmt Plan	1 day
6	Risk Management Plan	0 days
7	Develop Change Mgmt Plan	1 day
8	Change Management Plan	0 days
9	Develop Project Management Plan	4 days
10	Project Management Plan	0 days
11	Develop Project Schedule	1 day
12	Project Schedule	0 days
13	Kickoff Meeting	1 hr
14	Develop Marketing Plan	10 days
15	Marketing Plan	0 days
16	Develop new features	16.67 days
17	Bug fixes	25 days
18	Beta Software Release	0 days
19	Develop Status Report	1 day
20	Status Report	0 days
21	Status Meeting	1 hr
22	Develop website & poster	10 days
23	Marketing Website & Poster	0 days
24	Bug Fixes	10 days
25	Final Software & Doc Release	0 days
26	Lessons Learned Doc.	0 days
27	Closing Meeting	1 hr



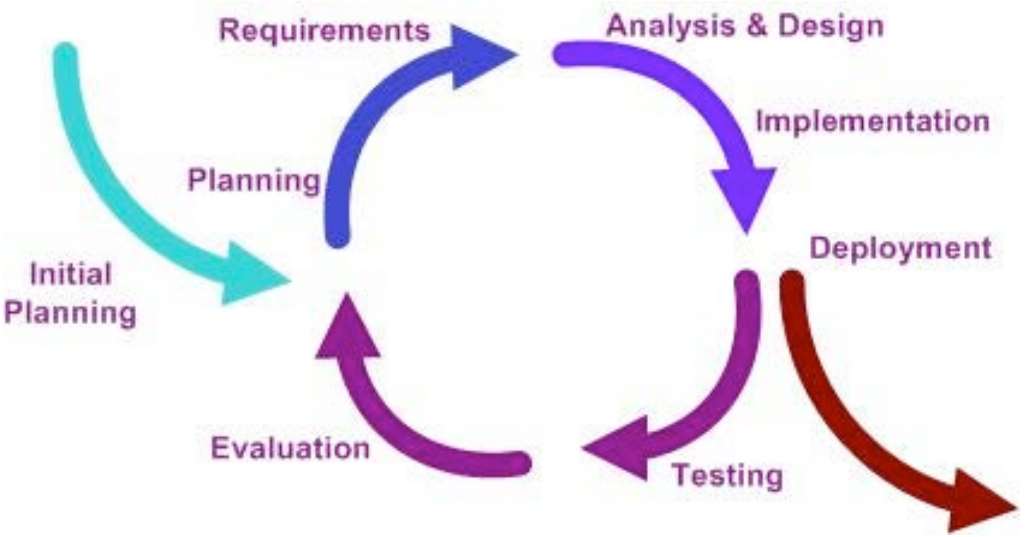
V-Model



Source: Wikipedia

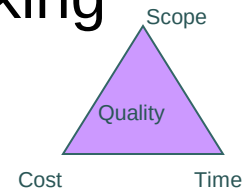
- Based on Waterfall
- Validation: “Are you building the right thing?” (User)
- Verification: “Are you building it right?”
- Shows relationship between each phase of development and its testing phase

Iterative and Incremental Development



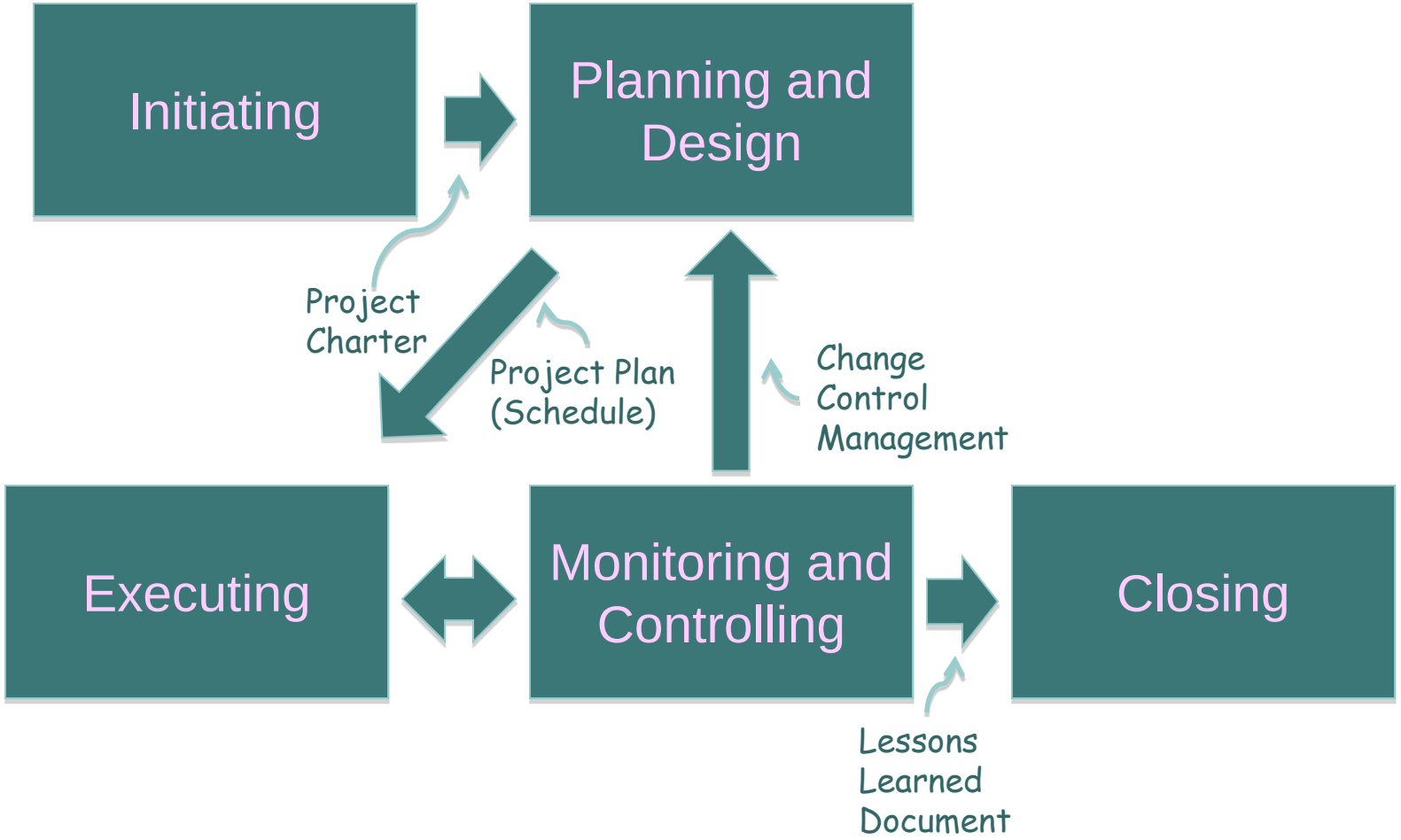
- Project Mercury (1960)
- Three steps
  - Initialization
  - Iteration
  - Project Control List: tasks and new features
- Advantage over Waterfall: Iterative allows backtracking
- Iterative is time boxed, not feature boxed
- Spiral, Agile and RUP are based on Iterative

Source: Wikipedia



Project Management Phases

Recall the PMBoK phases and the similarity to Iteration

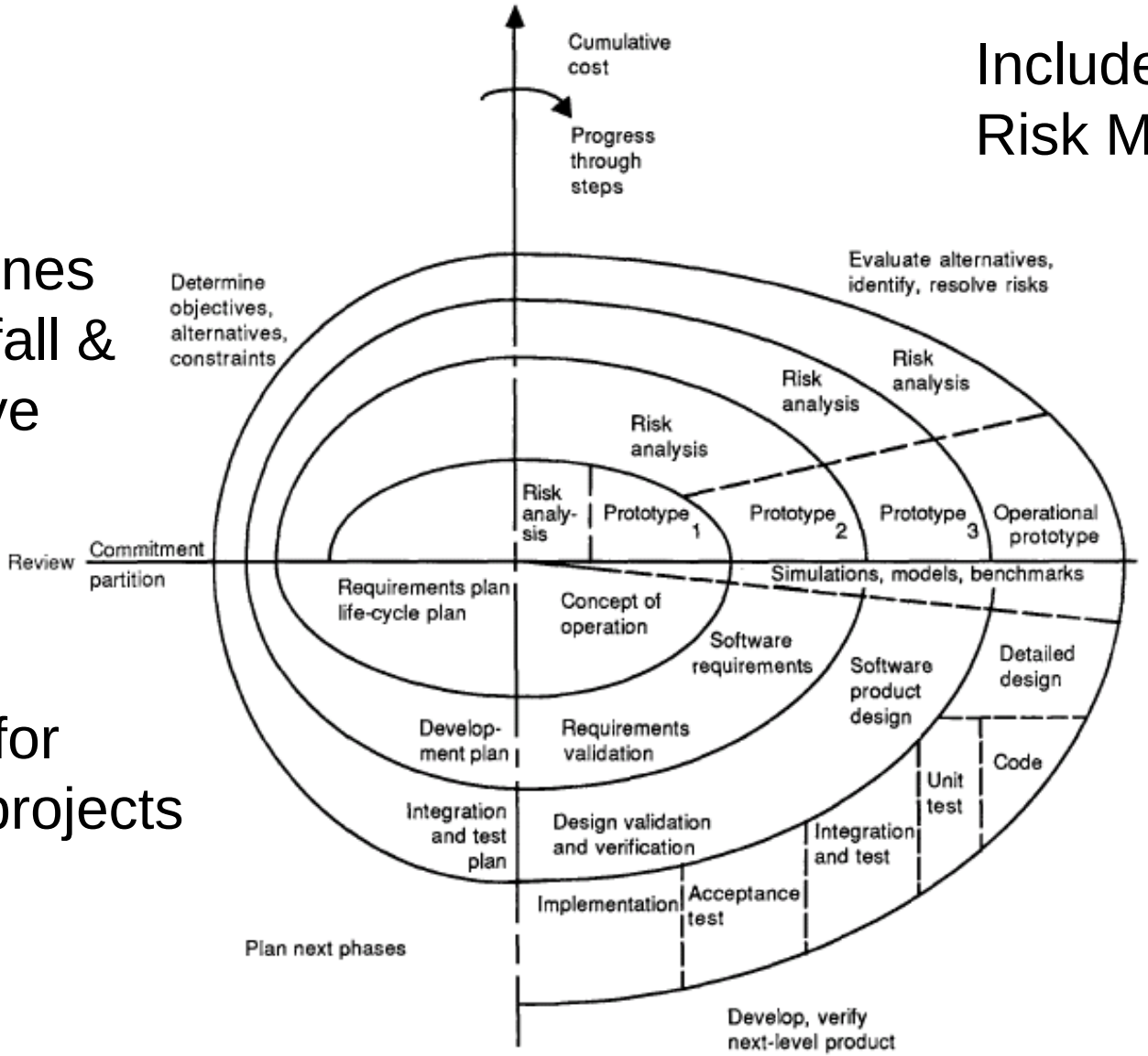


Spiral Model [Boehm, ’88]

Includes
Risk Management

Combines
Waterfall &
Iterative

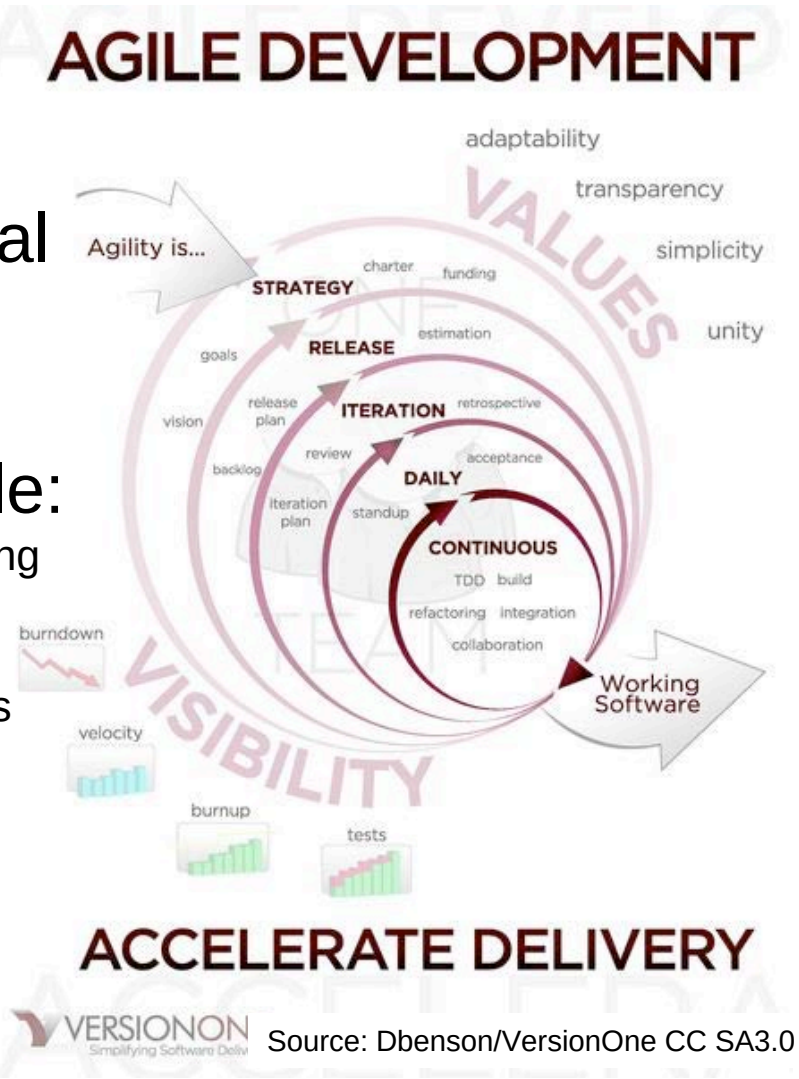
Good for
large projects



- Source: B.W Boehm, “A spiral model of software development and enhancement,”

Agile Development

- A group of methodologies
- Based on Iterative/Incremental
- Break up tasks
  small timeboxes (1-4 weeks)
- Each iteration: Full devel cycle:
  Planning, Requirements, Design, Coding, Testing
- Customer Representative:
  Appointed by stakeholders. Answers Questions
- Adaptive, not Plan-driven



Source: Dbenson/VersionOne CC SA3.0

Manifesto for Agile Software Development

We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:

Individuals and interactions over processes and tools
Working software over comprehensive documentation
Customer collaboration over contract negotiation
Responding to change over following a plan

That is, while there is value in the items on the right, we value the items on the left more.

Source: http://agilemanifesto.org/ (Beck et. al, 2001)

Principles behind the Agile Manifesto

We follow these principles

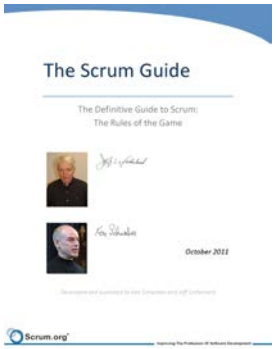
- Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
- Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.
- Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
- Business people and developers must work together daily throughout the project.
- Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.
- The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.
- Working software is the primary measure of progress.
- Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.
- Continuous attention to technical excellence and good design enhances agility.
- Simplicity--the art of maximizing the amount of work not done--is essential.
- The best architectures, requirements, and designs emerge from self-organizing teams.
- At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

Source: http://www.agilemanifesto.org/principles.html (Beck et. al, 2001)

Lean Software Development

- Based on a the Toyota Production System (1948-1975), originally known as Just In Time (JIT) production
- The Toyota Way: Continuous Improvement and **Respect** for People
- Lean has 7 Principles:
  - Eliminate Waste: Remove unnecessary code, requirements
  - Amplify Learning: Short cycles. Feedback from customer
  - Decide as late as possible: Max. Flexibility. Wait for customer changes
  - Deliver as fast as possible: Frequent Releases. Parallel Teams
  - Empower the team: Developer access to customer…
  - Build integrity in: Unit/Integration/System/Acceptance Tests. Refactoring.
  - See the whole: "Think big, act small, fail fast; learn rapidly"

Scrum



- “Scrum: A framework within which people can address complex adaptive problems, while productively and creatively delivering products of the highest possible value. Scrum is:
  - Lightweight
  - Simple to understand
  - Extremely difficult to master”

Source: The Scrum Guide, Schwaber & Sutherland (2011)

Scrum Roles

- Scrum Master: ensures process is followed, removes impediments
- Product Owner: Represents stakeholders
- Development Team: The people doing the work

Scrum Artifacts: Product Backlog

- Product Backlog
  - Everything needed in the product
  - Product Owner is responsible for content
  - An ordered list of items.
    - Each item description, order, and estimate
    - Ordered by value, risk, priority, and necessity
  - Product Backlog grooming: add detail, update estimates, change order. Usually no more than 10% of development team's capacity
  - Monitoring: Total work remaining is visible.

Scrum Artifacts: Sprint Backlog & Increment &"Done"

- Sprint Backlog: the set of Product Backlog items for the current split and a plan to complete the sprint goal.
  - Level of detail: Enough to show changes in progress during the Daily Scrum.
  - Monitoring: Total work remaining is visible.
- Increment: Sum of all Product Backlog items completed during the current sprint an previous sprints
- Definition of "Done": A shared understanding of what has been completed

Scrum Events

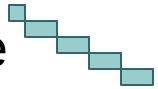
- Sprint: 1-5 weeks
- Daily Scrum: Fast stand up meeting: What was done yesterday? What will be done today? Am I blocked or will I be blocked?
- Backlog grooming: estimate existing backlog
- Scrum of Scrums: Each group sends a representative
- Sprint Planning: select the work, prepare the sprint backlog
- Sprint Review: (end of sprint). Present to stakeholders
- Sprint Retrospective: (end of sprint). What worked?

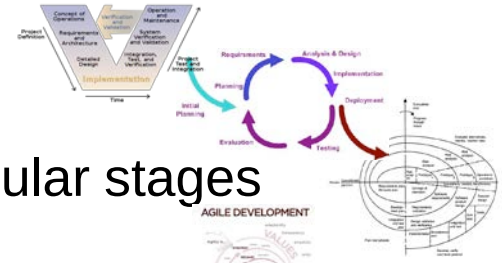
Extreme Programming (XP)

- Planning
  - Release planning
  - Frequent Releases
- Coding
  - Standards
  - Unit tests first
  - Pair Programming
  - Late Optimization
- Designing
  - Simplicity
  - System Metaphor
  - Refactoring
- Testing
  - Unit Tests
  - Bugs need Tests
  - Acceptance Tests

Based on http://www.extremeprogramming.org/rules.html

Software Development Life Cycles (SDLCs)





- Waterfall - Complete a phase before the next phase
- V-Model – Development vs. testing
- Iterative – Cyclic phases – Basis for many SDLCs
- Spiral – Waterfall & Iterative, Risk Management at regular stages
- Agile
  - Lean: Based on Just In Time. 7 principles
  - Scrum: Fast standup meeting: What was done yesterday? What will be done today? Am I blocked or will I be blocked?
  - Test-Driven Development (TDD): Write a Unit Test before coding
  - Extreme Programming (XP): **Fine scale feedback** (pair programming, TDD), **Continuous Process** (small releases), **Shared Understanding** (Coding standards), **Programmer welfare** (sustainable pace)
- Cleanroom: Formal Methods, Statistical Quality Control, Statistically Sound Testing
- Rapid Application Development (RAD): Construct Prototypes
- Rational Unified Process (RUP): Adaptable Process Framework

Suitability of different SDLC's

Agile Home Ground ***Adaptive methods*** **(Lean, Scrum, XP …)**	**Plan-driven Home Ground** ***Predictive Methods*** **(Waterfall, Iterative, Spiral …)**	**Formal Methods (Cleanroom …)**
Low Criticality	High Criticality	Extreme Criticality
Senior Developers	Junior Developers	Senior Developers
Requirements change often	Requirements do not change often	Limited requirements, Limited Features (Wirth's Law[1])
Small Number of Developers	Large Number of Developers	Requirements that can be modeled
Culture that thrives on chaos	Culture that demands order	Extreme quality

Based on: Wikipedia Agile Article, From Boehm and Turner (2004)

[1]Wirth's Law: "Software is getting slower more rapidly than hardware becomes faster" (1995)

Project Charter: Approach (again)

- Describe your software development life cycle (SDLC):
- What's that?
  - Methodologies: Waterfall, V-Model, Iterative, Spiral, **Agile**, Lean, XP, Cleanroom, Rapid Application Development (RAD), Rational Unified Process (RUP)
  - Techniques: Scrum, Test Driven Development (TDD)
- List the team
- List the time commitments

Project Charter: Approach Example

The project is a fairly small website based partly on a preexisting site, so we will use a classic waterfall approach with milestones. The project team will consist of the following people. I've estimated the maximum amount of time we can get from each person over the life of the project.

Kurt Keutzer (2 hrs week for 6 weeks)

Ken Lutz (2 hrs/week for 6 weeks)

Brad Krebs (10 hrs/week for 6 weeks)

Christopher Brooks (10 hrs/week for 6 weeks)

Allen Hopkins (5 hrs/week for 6 weeks)

Carol Sitea (1 hr/week for 6 weeks)

The project sponsor is Professor Keutzer. Professor Keutzer is on sabbatical this semester, but we hope to get feedback from him on a continuing basis.

Project Charter: Objectives

- List your objectives
- Objectives are one way to measure success
- Don't get too specific, there is time for that later
- Check your objectives with customer provided documentation
  - Are you solving the problem presented by the customer?

Project Charter: Objectives Example

- Update the look and feel of the website to a modern standard
- Provide access to student and faculty pages
- Provide access to active projects
- Provide access to summaries, downloads and key papers of inactive projects. The old pages of inactive projects should be archived.
- Provide a simple static listing of seminars. A more complex calendar and a search engine are deferred due to schedule constraints.

Project Charter: Deliverables

- What's a deliverable?
- Physical artifacts which describe your progress and include your product
- Deliverables should include:
  - A description (Scope)
  - A date (Time)
  - A person or persons who are responsible (Cost)

Project Charter: Deliverables Example

- A schedule along with time estimates.
- A prioritized list of features.
- An example of the main page so we can review look and feel.
- An archive of the old website
- The final website.

Project Charter: Constraints

- Schedule, Budget and Resource problems
  - What's a common constraint for eecs149?
  - Hardware availability
- Example:

Professor Keutzer would like to see the web site completed by mid-March: that is when students start looking at graduate schools. Developers might not have much time to work on this project.
The project requires timely feedback from the faculty.

Project Charter: Risks

- List things that could go wrong and how you will avoid them
- Don't skip the risks.
- Example:

> The primary risk is that the project takes too long to complete and we miss the mid-March opportunity. Another risk is that we complete the project too quickly and quality suffers. A third risk is that there are only so many resources available. By fast tracking, we can handle some of the tasks in parallel and avoid these risks. The project is definitely feasible if we roll out the website in stages.

EECS 149 Deliverables (Cont'd)

From the Project Management lab:

3. Set Milestones and a Schedule: Working backwards from the completion due date, determine when tasks need to be done to complete the project on time. *Ensure you are allowing adequate time for debugging – integrating disjoint components from several people can take a week or more.* Be sure to consider events such as Spring Break, Finals Week, etc.. Milestones should indicate which team member is responsible and be no more than one week apart.
Be sure to include non-technical tasks such as practicing the presentation or writing the report. Also include a plan to meet with your mentor, preferably once a week (you may want to organize your milestones to land on these dates). Be sure to include vacations like Spring Break.
Do not hesitate to be painfully specific. Through the duration of this project, you should submit milestone reports to track your progress. Certainly, the milestones could change during the course of the project as you learn and debug things.

Why Milestones

- How will you measure success?
- How will you divide up the work?
- How will you make sure everyone participates?

- Milestones

What is a milestone?

- A milestone is a checkpoint in the project
  - A milestone has
    - A description of the point
    - A date
    - A person or persons who are responsible
- A milestone may or may not have a deliverable associated with it.
  - A deliverable has
    - A description of a deliverable (scope)
    - A date (time)
    - A person or persons who are responsible (cost)

Q: Where to get Milestones?

A: From the Customer and Sponor!

On 2/4/12, the EECS149 Website said:

Date	Deliverable
3/6	5 minute project presentations due
3/13	*Monday labs have completed charter, WBS & milestones on 3/12*
3/20	First 5-min Project Mini-Updates, 1st Milestone Update Report due
4/5	5-min Project Mini-Updates, New Milestone Report due
4/12	5-min Project Mini-Updates, New Milestone Report due
4/21	Demo your projects!
4/24	5-min Project Mini-Updates, New Milestone Prediction (including goals for the final presentation)
5/3	Final project milestone reports due
5/9	Project Presentations
5/11	Project reports due at 12 noon (Pacific)

Keep checking the website.

Schedules change.

Milestones

Date	Deliverable
3/6	5 minute project presentations due
3/13	*Monday labs have completed charter, WBS & milestones on 3/12*
3/20	First 5-min Project Mini-Updates, 1st Milestone Update Report due
4/5	5-min Project Mini-Updates, New Milestone Report due
4/12	5-min Project Mini-Updates, New Milestone Report due
4/21	Demo your projects!
4/24	5-min Project Mini-Updates, New Milestone Prediction (including goals for the final presentation)
5/3	Final project milestone reports due
5/9	Project Presentations
5/11	Project reports due at 12 noon (Pacific)

- However, the above is not detailed enough for success.
- That's why the Lab includes "Set Milestones and a Schedule".

Work Breakdown Structure

From the Project Management Lab:

4. Set a Work Breakdown Structure: Generate a Work Breakdown Structure (WBS) for your project, being careful to satisfy the 100% and mutual exclusion rules.

The diagram should show dependencies between milestones and which tasks can be performed in parallel.

For simpler projects, this could be very linear, but try to divide the tasks so as much as possible can be done in parallel so work can be split between people.

Ensure that the WBS answers when a milestone will be finished and who is responsible for it finishing.

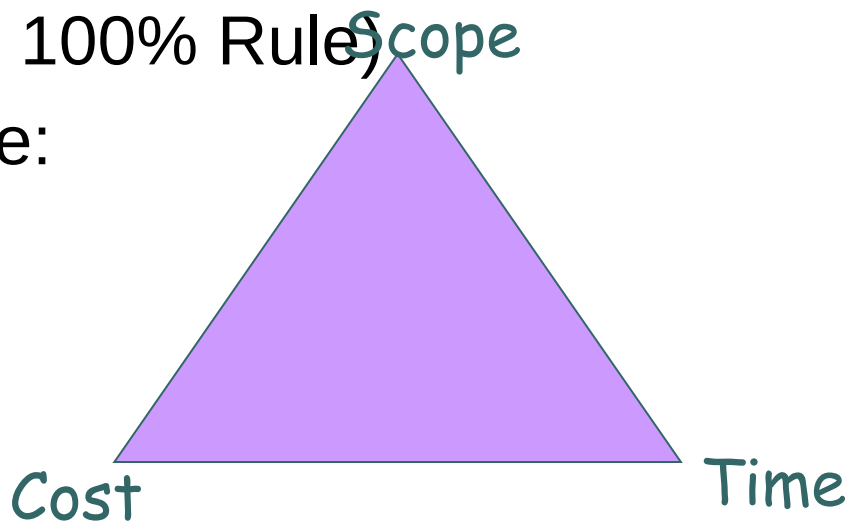
Work Breakdown Structure (WBS)

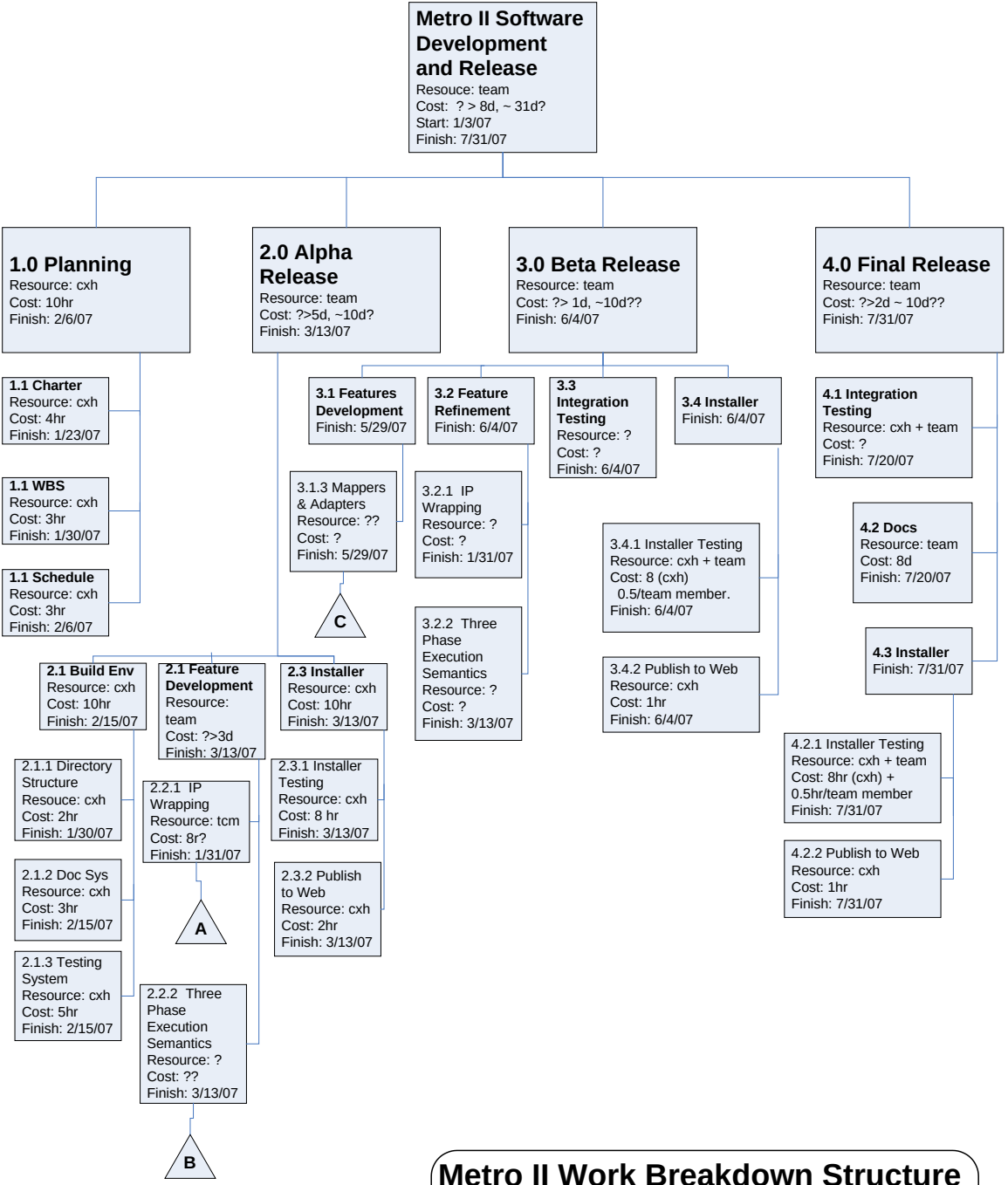
- How to do a schedule: Use a Work Breakdown Structure
- A work breakdown structure is an outline that describes the deliverables.
- Each level of the outline describes 100% of the work below it (the 100% Rule)
- A WBS should include:

  What (Scope)

  When (Time)

  Who (Cost)





Metro II Work Breakdown Structure

Page 1 of 2	3/6/2008	Christopher Brooks

Partial Informal WBS for the LED T-Shirt

- LED T-Shirt (Tansy and Christopher) due on 5/10
  - Hardware (Tansy)
    - Obtain Electronics (Christopher) (3/15)
    - Obtain Fabric Materials (Tansy) (3/15)
    - Design Schematic (Christopher) (3/10)
    - Solder LEDs (Christopher) (3/20)
    - Sew Schematic (Tansy) (4/15)
    - Test Circuit (Christopher) (4/20)
  - Software (Christopher)
    - Build Code Generation Environment (Christopher) (3/15)
    - Develop algorithm in simulation (Christopher) (3/15)
    - Breadboard Circuit (Christopher) (3/20)
    - Provide Software requirements for Schematic (Christopher) (4/10)
    - Download Software onto Chip (Christopher) (4/20)
  - Integration (Tansy and Christopher) (4/25)
  - Testing (Tansy and Christopher) (5/1)
- Problem: schematic design due early!

Textual WBS for a small project

WBS	Name	Duration	Effort	Resources	Cost	Finish
spccdt1	Develop Project Charter	1 day	11.2 hrs	PM,Cust,Fac	$1,168	1/13/09
spccdt2	Project Charter	0 days	0 hrs		$0	1/26/09
spccdt3	Develop User Interface Design	2 days	5.6 hrs	Dev,Fac,Cust	$304	1/27/09
spccdt4	User Interface Design Doc.	0 days	0 hrs		$0	1/30/09
spccdt5	Develop Risk Mgmt Plan	1 day	8 hrs	PM	$800	1/26/09
spccdt6	Risk Management Plan	0 days	0 hrs		$0	1/26/09
spccdt7	Develop Change Mgmt Plan	1 day	0 hrs		$0	1/26/09
spccdt8	Change Management Plan	0 days	0 hrs		$0	1/26/09
spccdt9	Develop Project Mgmt Plan	4 days	38.4 hrs	PM,Fac,Cust	$3,936	2/4/09
spccdt10	Project Management Plan	0 days	0 hrs		$0	2/4/09
spccdt11	Develop Project Schedule	1 day	8 hrs	PM	$800	2/5/09
spccdt12	Project Schedule	0 days	0 hrs		$0	2/5/09
spccdt13	Kickoff Meeting	1 hr	4 hrs	Cust,Fac,PM,Dev	$360	2/6/09
spccdt14	Develop Marketing Plan	10 days	10.22 hrs	Dev,Fac,Cust,PM	$789	2/20/09
spccdt15	Marketing Plan	0 days	0 hrs		$0	2/4/09
spccdt16	Develop new features	16.67 days	21.33 hrs	Dev,Fac,Cust	$1,320	3/2/09
spccdt17	Bug fixes	25 days	28 hrs	Dev,Cust,Fac	$1,520	3/13/09
spccdt18	Beta Software Release	0 days	0 hrs		$0	3/13/09
spccdt19	Develop Status Report	1 day	2 hrs	Dev	$60	3/16/09
spccdt20	Status Report	0 days	0 hrs		$0	3/16/09
spccdt21	Status Meeting	1 hr	4 hrs	Cust,Fac,PM,Dev	$360	3/16/09
spccdt22	Develop website & poster	10 days	28 hrs	Dev,Fac,Cust	$1,520	3/30/09
spccdt23	Marketing Website & Poster	0 days	0 hrs		$0	3/30/09
spccdt24	Bug Fixes	10 days	28 hrs	Dev,Fac,Cust	$1,520	4/13/09
spccdt25	Final Software & Doc Release	0 days	0 hrs		$0	4/13/09
spccdt26	Lessons Learned Doc.	0 days	0 hrs		$0	4/13/09
spccdt27	Closing Meeting	1 hr	4 hrs	Cust,Fac,PM,Dev	$360	4/13/09
Totals					$14,817	

Note: Milestones have a duration of 0

WBS Columns Explained

WBS	Name	Duration	Effort	Resources	Cost	Finish
spccdt1	Develop Project Charter	1 day	11.2 hrs	PM,Cust,Fac	$1,168	1/13/09
spccdt2	Project Charter	0 days	0 hrs		$0	1/26/09

- WBS – A unique identifier for the package
- Name
- Duration
  - "The total number of work periods . . . required to complete a schedule activity or work breakdown component" (PMBoK)
- Effort
  - "The number of labor units required to complete a schedule activity or work breakdown structure component" (PMBoK)
- Resources
- Cost
- Finish

Q: How many hours per week of effort per person?

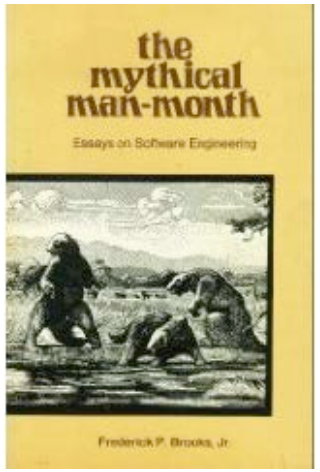
A: Not more than ~32.5 hrs/week

EECS149: WBS and Cost are probably not necessary

The Critical Path

- What's the Critical Path?
- The longest path through the project, which determines the shortest time to completion.
- Solutions during Planning:
  - Do things in parallel
  - Add more resources
  - Prune deliverables
- Solutions during Operation:
  - Replan
  - Panic

Brooks's Law



- "adding manpower to a late software project makes it later" Fred Brooks, 1975
- Why?
  - Ramp Up
  - Communication Overhead
- But what about Open Source? (Cathedral and the Bazaar)
  - Cheap Communication
  - Many Programmers

Schedule Problems to Avoid

- Milestones are too coarse grained
  - Have at least one milestone per week
  - Why? How will you know if you are falling behind?
- No parallelism in the schedule
  - Why? Get more work done in a shorter term, though integration/synchronization has costs
- Schedule is not updated
  - Why? How will you know if you are falling behind?
- Poor Estimates
  - Why? Humans are optimistic, when was the last time you finished early? One rule of thumb is to multiply an estimate by pi.
    Also, consider that each teammate has other classes.

Common EECS149 Project Failures

- Overly ambitious a project that lacks intermediate results
  - Solution: Fine grained milestones
  - Solution: Replan
- One or more team members don't do the work.
  - Solution: Fine grained milestones so that you see the problem, then replan
  - Pay attention to Team Dynamics: Forming, Storming, Norming, Performing, Mourning
- Equipment constraints
  - Determine equipment needs ASAP

11 Tips to successfully completing a project

1. Create a one page charter
2. Separation of concerns: keep algorithms separate from drivers
3. Start writing tests early, use a code coverage tool if you can
4. Use version control and a nightly build
5. Meet often: Update each other of progress and problems
6. Use tools: memory leaks, warnings, spelling errors, performance problems, other compilers, other operating systems.
7. Document your code. Writing documentation first can prevent hours of wasted time.
8. Don't debug for more than an hour by yourself – get help.
9. Design Review and Code Review (or at least desk check)
10. Expect the unexpected: wacky user input, wacky user interaction, problems with threads.
11. Don't be afraid to throw away code and start over.

Quality	Time	Cost	Scope
+	+	+	+
+	+		
+	+		
+	+		
+	+		
+	+		

EECS149 Project Management Conclusions

- Your Deliverables for the Project Management Lab:
  - One page project charter (example on website)
  - Set Milestones and a Schedule
    - Milestones (what, when, who)
    - Work backwards: what can you realistically accomplish? Deliver the good stuff first.
  - Set a Work Breakdown Structure
- Avoid common mistakes
  - Plan
  - Allow for testing and for mistakes
  - Order parts now: risk for project charter?
  - Integrate early: simulate, prototype
  - Partition the work: work in teams. Updates about progress/problems
  - Keep track of your status: update the plan
  - Give credit to your sources

Questions?