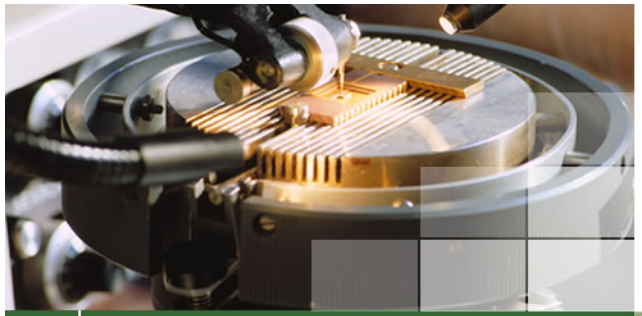


POLITECNICO DI MILANO

Dipartimento di
Elettronica e Informazione

Planning and Managing Software Projects 2010-11

Session 8 – 1st part

Development Management

Emanuele Della Valle

http://emanueledellavalle.org

Credits

- This slides are largely based on Prof. John Musser class notes on “Principles of Software Project Management”
- Original slides are available at http://www.projectreference.com/
- Reuse and republish permission was granted

Today

- People dimension
- Capability Maturity Model
- Requirements (most critical activity)
- Other notes

Session 7 Review

- Risk Management
- Feature Set Control
- Change Control

Risk Management

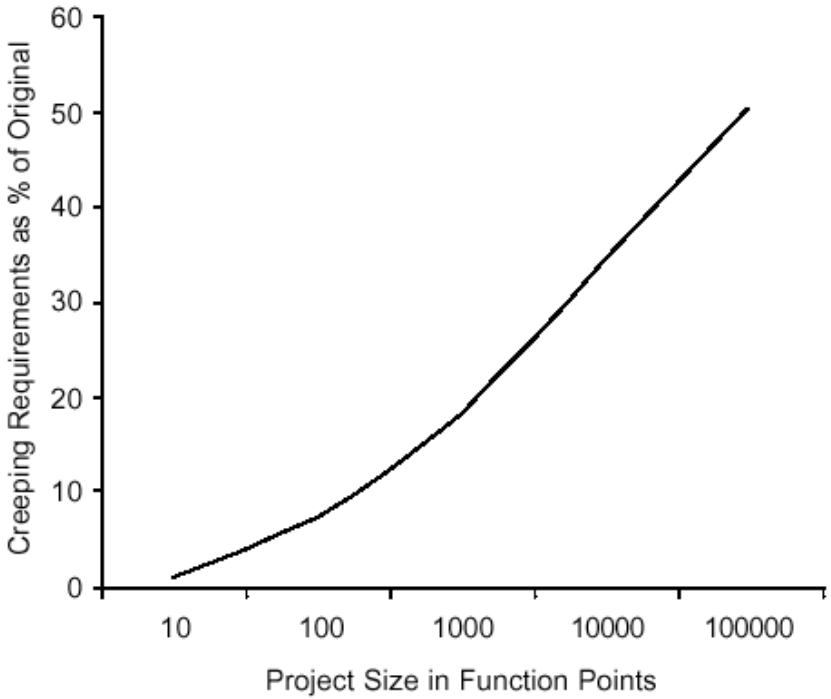
- Risk Management
  - Types of risk: schedule, cost, requirements
- Risk Identification
- Risk Analysis
  - Risk Exposure (RE = Prob. * Size)
- Risk Prioritization
- Risk Control
- Risk Resolution
  - Avoidance, assumption, transfer, gain knowledge
- Top 10 Risk List

Feature Set Control

- Early Stages
  1. Minimal Specification
  2. Requirements Scrubbing
  3. Versioned Development
- Mid-Project
  - Effective change control
- Late-Project
  - Feature cuts

Change Control

- Average project has 25% requirements change
- Sources of change
- Change control is a process
- Overly detailed specs. or prolonged requirements phase are not the answer
- Change Control Board (CCB)
  - Structure
  - Process
  - **Triage !!!**



Configuration Control

- Items: code, documents
- Change & Version control
- SCM
- Configuration Management Plan

Project Roles

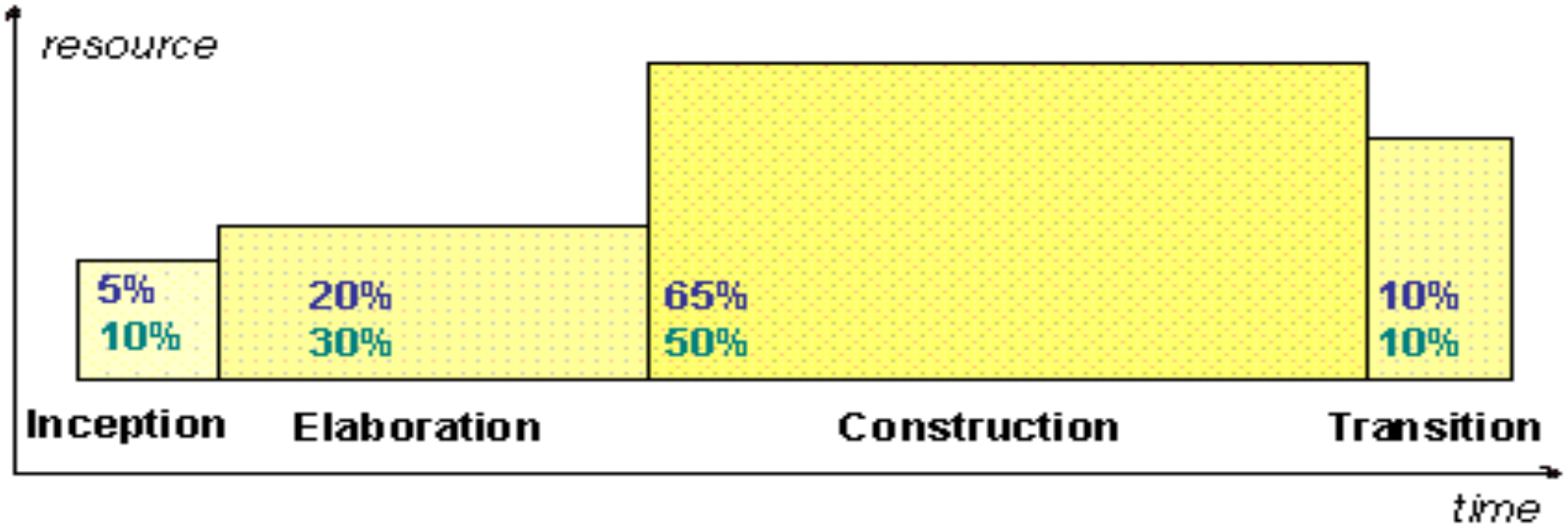
- Programmers (system engineers)
  - Technical lead, architect, programmer, Sr. programmer
- Quality Assurance (QA) engineers (testers)
  - QA Manager, QA Lead, QA staff
- DBAs
  - DB Administrator, DB Programmer, DB Modeler
- CM engineers (build engineers)
- Network engineers, System Administrators
- Analysts (business analysts)
- UI Designers
- Information Architects
- Documentation writers (editors, documentation specialist)
- Project manager
- Other
  - Security specialist, consultants, trainer

Project Roles & Homework 4

- You need to decide which of these are necessary for your class project
- Depends on what you're building
  - How big is it?
  - Is it UI intensive? Data intensive?
  - Are you installing/managing hardware?
  - Do you need to run an operations center?
  - Is it in-house, contract, Commercial off-the-shelf (COTS), etc?
- Depends on your budget
- More about it in the 2nd part of next lesson in the computer laboratory

Staffing Profile

- Projects do not typically have a ‘static team size’
- Who and how many varies as needed



Legend:
Actual Effort (% of project total)
Schedule (% of project total)

Roll-on & Roll-off

- PM must have a plan as to how & when
- Roll-on
  - Hiring or ‘reserving’ resources
  - Ramp-up time
    - Learning project or company
- Roll-off
  - Knowledge transfer
  - Documentation
  - Cleanup

Staffing Management Plan

- Part of Software Development Plan
- Includes
  - What roles needed, how many, when, who
  - Resource assignments
  - Timing: start/stop dates
  - Cost/salary targets (if hiring)
- Project Directory
  - Simply a list of those involved with contact info.
- Team size: often dictated by budget as often as any other factor

Team Structure

- 1st: What's the team's objective?
  - Problem resolution
    - Complex, poorly-defined problem
    - Focuses on 1-3 specific issues
      - Ex: fixing a showstopper defect
    - Sense of urgency
  - Creativity
    - New product development
  - Tactical execution
    - Carrying-out well-defined plan
    - Focused tasks and clear roles

No single team structure is best for all projects

Team Models

- Two early philosophies
  - Decentralized/democratic
  - Centralized/autocratic
- Variation
  - Controlled Decentralized

Business Team

- Most common model
- Technical lead + team (rest team at equal status)
- Hierarchical with one principal contact
- Adaptable and general
- Variation: Democratic Team
  - All decisions made by whole team
  - See Weinberg's "egoless programming" model [1]

[1] Gerald M. Weinberg, "Egoless Programming," IEEE Software, vol. 16, no. 1, pp. 118-120, Jan./Feb. 1999, doi:10.1109/MS.1999.744582
http://www2.computer.org/portal/web/csdl/doi/10.1109/MS.1999.744582

Chief-Programmer Team

- From IBM in 70's
  - See Brooks and Mythical Man-Month
- a.k.a. 'surgical team'
- Puts a superstar at the top
  - Others then specialize around him/her
    - Backup Programmer
    - Co-pilot or alter-ego
    - Administrator
    - Toolsmith
    - "Language lawyer"
- Issues
  - Difficult to achieve
  - Ego issues: superstar and/or team
- Can be appropriate for creative projects or tactical execution

"Skunkworks" Team [1]

- Put a bunch of talented, creative developers away from the mother ship
  - Off-site literally or figuratively
- Pro: Creates high ownership & buy-in
- Con: Little visibility into team progress
- Applicable: exploratory projects needing creativity
  - Not on well-defined or narrow problem

[1] http://searchcio.techtarget.com/sDefinition/0,,sid182_gci214112,00.html

SWAT Team [1]

- Highly skilled team
- Skills tightly match goal
- Members often work together
- Ex: security swat team
- The team model for tactical execution

[1] http://en.wikipedia.org/wiki/SWAT

Other (bad) kind of teams

Check out http://www.scottberkun.com/blog/2007/asshole-driven-development/ for

- Asshole Driven development
- Cognitive Dissonance development
- Cover Your Ass Engineering
- Development By Denial
- Get Me Promoted Methodology

Wrapping up

Team Model	Problem Resolution	Creativity	Tactical Execution
Business Team	***	*	**
Chief-Programmer Team		***	**
"Skunkworks" Team		***	
SWAT Team			***

LEGEND

*** Best suited

* Can be used

Large teams

- Communication increases multiplicatively
  - Square of the number of people
  - 50 programmers = 1200 possible paths
  - Communication must be formalized
- Always use a hierarchy
- Reduce units to optimal team sizes
  - Always less than 10

Team Size

- What is the optimal team size?
  - 4-6 developers
    - Tech lead + developers
  - Small projects inspire stronger identification
  - Increases cohesiveness
  - QA, operations, and design on top of this

Hiring

- "Hire for Attitude, Train for Skill"
- Look for: "Smart, Gets Things Done"
- For programmers, see joelonsoftare's "Guerilla Guide to Interviewing"
  - http://www.joelonsoftware.com/articles/fog0000000073.html
  - http://www.joelonsoftware.com/articles/GuerrillaInterviewing3.html
- Balance the team

Responsibility Assignment Matrix

- A resource planning tool
- Who does What
- Can be for both planning and tracking
- Identify authority, accountability, responsibility
- Who: can be individual, team or department
- Can have totals/summary at end of row or column (ex: total Contributors on a task)

Simple RAM

Item	WBS	Description		Sponsor	Developer	Developer	QA	Customer
1	1	Initiate Project		A				
2	1.1	PMP Signoff		A				R
3	1.2	Initial UI			L	C		R
4	1.3	DB Model			C	L		
5	1.4	Start Test					L	
	Legend							
	A	Approval						
	L	Lead						
	S	Secondary						
	C	Contributor						
	R	Reviewer						

Sample RAM With Stakeholders

Item		Development	Customer A	Customer B	Mgmt	QA	
Unit Test		A	S	S	R	A	
Systems Test		P	R	R	R	R	
Beta Test		P	R	R	P	R	
User Acceptance Test		A	S	S	S	S	
Accountable	A						
Participant	P						
Reviewer	R						
Sign-off Required	S						

Skills Matrix

- Another resource planning tool
- Resources on one axis, skills on other
- Skills can high level or very specific
- Cells can be X's or numeric (ex: level, # yrs.)

	Analyst	Developer (Java)	Developer (HTML)	QA Tester	Database Design
Dilbert	7	2			
Larry			8		4
Sarah	4	4			
Boss				4	
Fred					5

Capability Maturity Model: CMM

- A software process framework
- “Process determines capability”
- 5 ‘maturity’ levels
  - ‘Evolutionary plateaus’ to a mature software process
- Each level has its own goals
- Organizations can be ‘certified’
  - Later to be used as a marketing or validation tool
  - http://www.itil-officialsite.com/home/home.asp
- Links:
  - SEI - http://www.sei.cmu.edu/
  - Diagram - http://www.sei.cmu.edu/cmm/

Levels 1/2

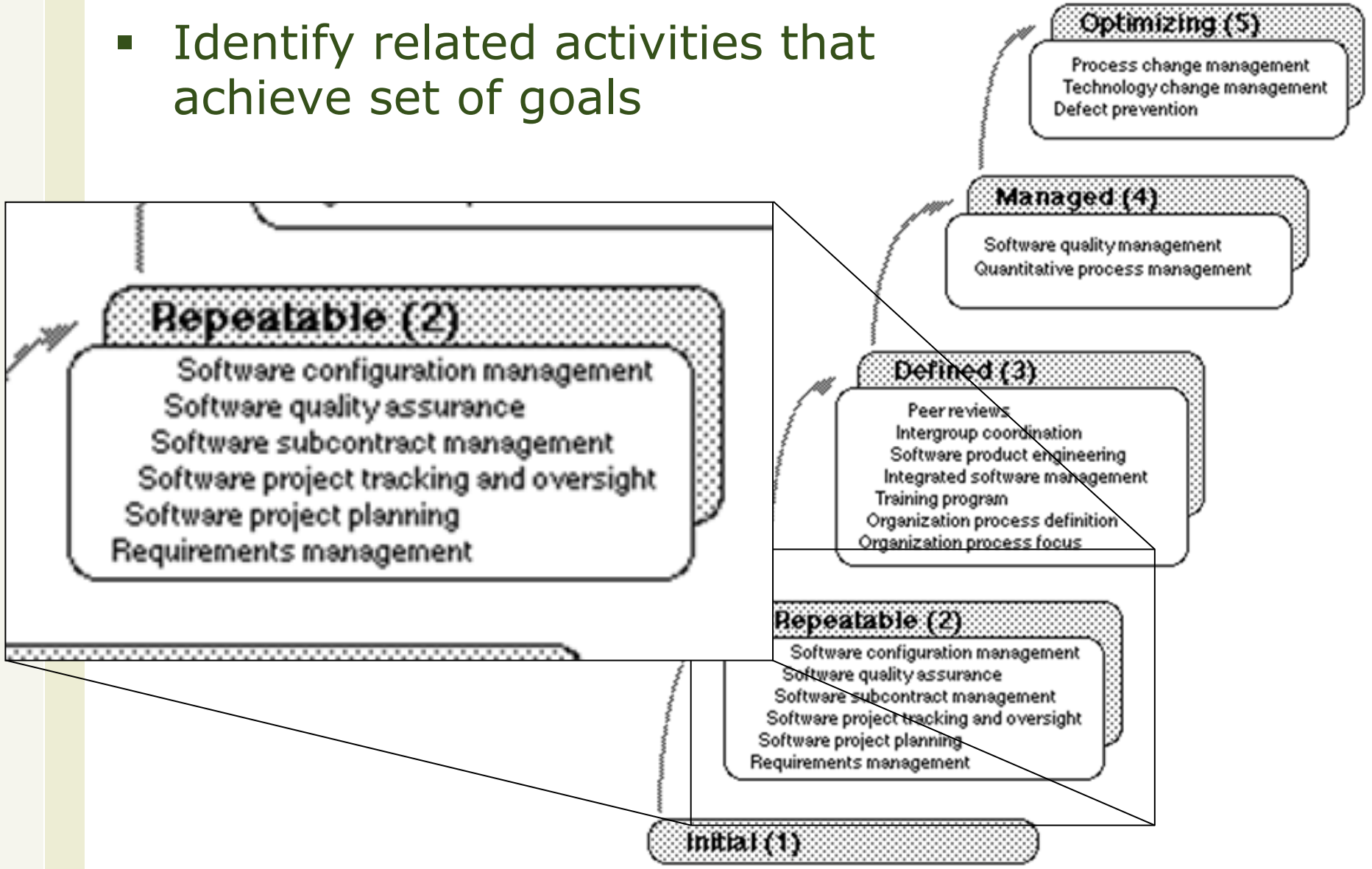
1. Initial
   - ‘Ad hoc’ process, chaotic even
   - Few defined processes
   - Heroics often required here
2. Repeatable
   - Basic PM processes
     - For cost, schedule, functionality
   - Earlier successes can be repeated
3. Defined
   - Software & Mgmt. process documented
   - All projects use a version of org. standard

Levels 2/2

4. Managed
   - Detailed metrics of process & quality
   - Quantitative control
5. Optimizing
   - Continuous process improvement
   - Using quantitative feedback

Key Process Areas

- Identify related activities that achieve set of goals



Who needs a CMM certification?

- India has more CMM level 4 & 5 companies than any other country
  - Why is that?

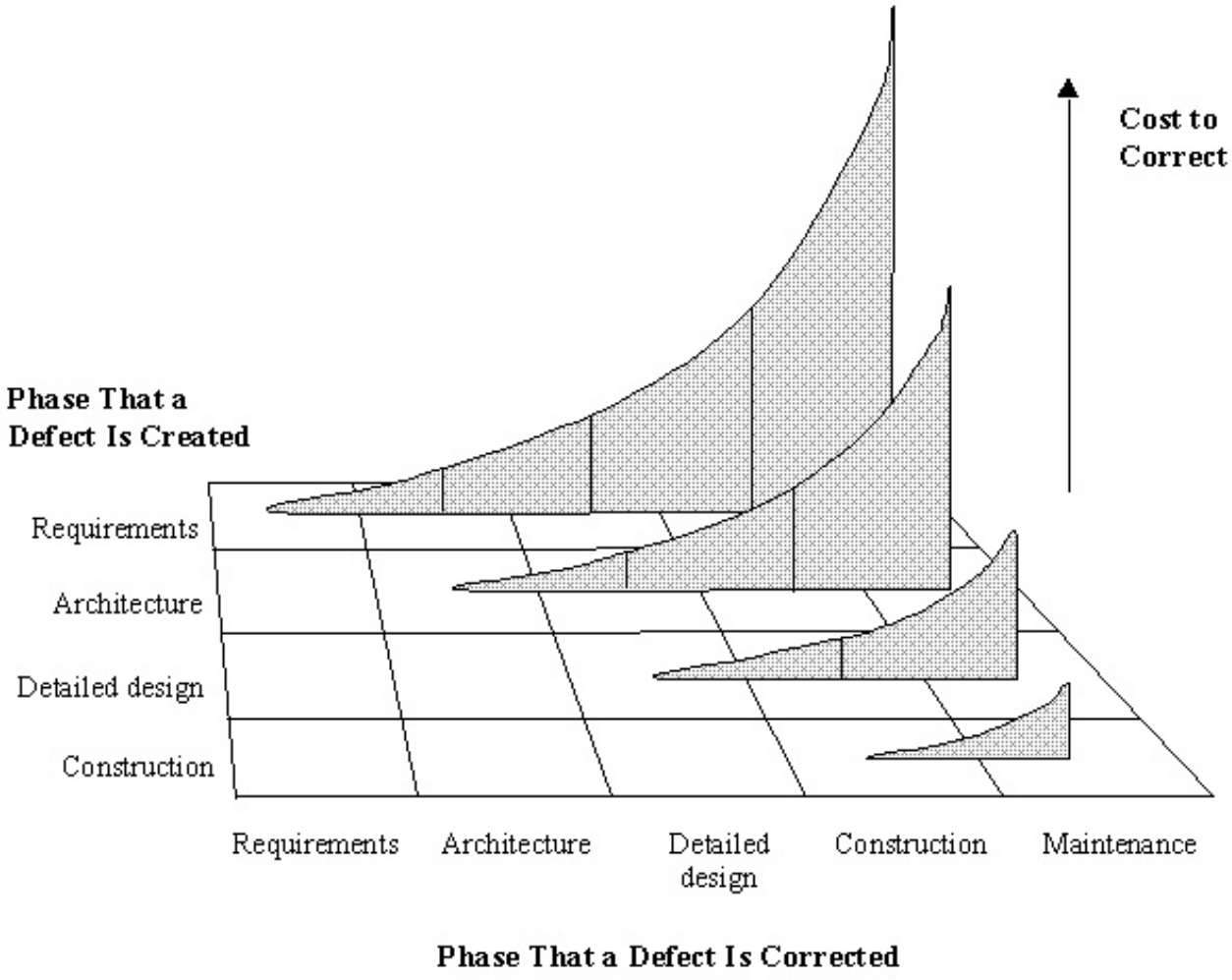
Requirements

- Requirements are capabilities and condition to which the system – more broadly, the project – must conform

Requirements

- Perhaps the most important & difficult phase to control
- Shortchanging it is a ‘classic mistake’
- Can begin with a Project Kickoff Meeting
- Can end with a Software Requirements Review (SRR)
  - For Sponsor and/or customer(s) approval

Why are Requirements so Important? (see lesson 3)



Characteristics & Issues

- Conflict of interest: developer vs. customer
- Potential tug-of-war:
  - Disagreement on Features & Estimates
  - Especially in fixed-price contracts
- Frequent requirements changes
- Achieving sign-off
- Project planning occurs in parallel

2 Types of Requirements (see lesson 3)

- Functional (behavioral)
  - Features and capabilities
- Non-functional (a.k.a. “technical”) (everything else)
  - Usability
    - Human factors, help, documentation
  - Reliability
    - Failure rates, recoverability, availability
  - Performance
    - Response times, throughput, resource usage
  - Supportability
    - Maintainability, internationalization
  - Operations: systems management, installation
  - Interface: integration with other systems
  - Other: legal, packaging, hardware

The “must” to remember

- Must be prioritized
  - Must-have
  - Should-have
  - Could-have (Nice-to-have: NTH)
- Must be approved

Used by many people for many purposes

- Management: Yes, that's what I funded
- Users: Yeah, that's what I need
- Developers: Yes, that's what I will build

Input and Outputs (see session 3)

- The “What” phase
- Inputs: SOW, Proposal
- Outputs:
  - Requirements Document (RD)
    - a.k.a.Requirements Specification Document (RSD)
    - Software Requirements Specification (SRS)
  - 1st Project Baseline
  - Software Project Management Plan (SPMP)
  - Requirements Approval & Sign-Off
    - Your most difficult task in this phase

Quotes to Think About

- “There’s no sense being exact about something if you don’t even know what you’re talking about”

  -- John von Neumann

- “When the map and the territory don’t agree, always believe the territory”

  -- taught to all Swedish army recruits

Requirements Gathering Techniques

- Interviews
- Document Analysis
- Brainstorming
- Requirements Workshops
- Prototyping
- Use Cases
- Storyboards
- There are more…

Interview Techniques

- Best practice: use ‘context free’ questions
  - A question that does not suggest a response
  - High-level, early questions to obtain ‘global’ properties of the problem and solution
  - Applicable to any project/product
  - Get the ball rolling

Context-free Questions

- Process, product and meta questions
- Process
  - “Who is the client for project X”?
  - “What is a very successful solution really worth to the client”?
  - “What is the reason for this project”?
- Product
  - “ What problems does this system solve”?
  - “What problems could this system create”?
- Meta-questions
  - “Are these questions relevant”?
  - “Is there anyone else who can give useful answers”?
  - “Is there anything you want to ask me”?

Document Analysis

- Review of existing documents
  - Business plans
  - Market studies
  - Contracts
  - Requests for proposals (RFP)
  - Statements of work (SOW)
  - Existing guidelines
  - Analyses of existing systems and procedures

Brainstorming

- Idea generation & idea reduction
- Typically via group meetings
- Generation best practices
  - Minimize criticism and debate
    - Editing occurs at end of meeting or later
  - Aim for quantity
  - Mutate or combine ideas
- Reduction best practices
  - Voting with a threshold (N votes/person)
  - Blending ideas
  - Applying criteria
  - Scoring with a weighting formula

Requirements Workshops

- Typically 1-5 days
- Who? Varies. Users & stakeholders
- Pros
  - Good for consensus building
  - Builds participant commitment
  - Can cost less than numerous interviews
  - Provide structure to capture and analysis process
  - Can involve users across organizational boundaries
  - Can help identify priorities and contentious issues
- Joint Application Design (JAD)
  - A type of requirements workshop using a facilitator
  - See http://www.carolla.com/wp-jad.htm

Prototyping

- Quick and rough implementation
- Good communications mechanism
- Can be combined with other techniques such as JAD
- Issues: creating the mis-appearance that development is more complete than it actually is
  - See joelonsoftware's "The Iceberg Secret Revealed"
  - http://www.joelonsoftware.com/articles/fog0000000356.html

The Iceberg Secret

- You know how an iceberg is 90% underwater? Well, most software is like that too
  - 10% of the work goes into a *pretty* UI
  - 90% of the programming work is under the covers
- That's not the secret. The secret is that *People Who Aren't Programmers Do Not Understand This*.
- Corollaries:
  1. Bad interface → bad program
  2. Beautiful interface → program is complete
  3. When demanding nontechnical managers or customers "sign off" on a project, give them several versions of the graphic design to choose from.
  4. When you're showing off, the only thing that matters is the screen shot. Make it 100% beautiful.

[Source: See joelonsoftware's "The Iceberg Secret Revealed" http://www.joelonsoftware.com/articles/fog0000000356.html]

Use Cases

- Picture plus description
- Often misunderstood: the text is more important
- Text structure
  - Use case name and number (ID)
  - Goal
  - Pre-conditions
  - Primary & secondary actors
  - Trigger
  - Description
  - Business rules
  - Open issues
- See also http://alistair.cockburn.us/Use+cases

Storyboards

- Set of drawings depicting user activities
- Paper prototyping
- Drawing screens or processes

Who?

- Customers and users (note: often not the same)
  - Customers can be users, but rarely opposite
  - Sometimes user constituencies need to be ‘found’
- Subject matter experts
- Other stakeholders
  - Marketing, sales
  - Product managers

Other Tips

- Meetings
  - Treat them like a tool: design them
  - Boy scout motto: “Be Prepared”
  - As small as possible – but no smaller
  - Make it safe not to attend
    - Publish an agenda before
    - Publish a summary after
  - Generate a ‘related issues’ list
  - Can formal/informal, scheduled/ad-hoc

Other Tips

- Manage expectations
  - Don't forget to ask people theirs
  - Listen
  - Make explicit otherwise implicit decisions
    - PDA: Possibility, Deferred, Absolutely impossible
- Technical reviews
  - Requirements can be wrong by: inadequate or inaccurate
    - Quantity & quality
  - Reviews help with the latter

Tools

- Borland Caliber Analyst
  - Collaborative Software Requirements Engineering
  - http://www.borland.com/us/products/caliber/index.html
- IBM Telelogic DOORS
  - Requirements Management for Complex Systems and Software Development
  - http://www.telelogic.com/Products/doors/doors/index.cfm
- Both offer
  - Databases of requirements
  - Displayable in various formats
  - Requirements control metrics:
    - Requirements Stability Index
      - See http://sunset.usc.edu/classes/cs577b_2001/metricsguide/metrics.html#p36
    - To help manage feature creep and 'vibration'

There's a Tool for Every Need

- Requirements Tools
- Design Tools
- Construction Tools
- Test Tools
- Maintenance Tools
- CM Tools

Insights

- Tools could save 10-25% on some projects
  - But that's optimistic at best
- Choose tools to meet your needs
- No can guarantee you anything
  - They **may** help
  - Tools don't control people, especially customers

Programming Languages

- Your projects: do you choose a language?
- Typically not the PM's choice, but does effect you
  - Staffing requirements
  - Methodology
  - Tools and infrastructure

Optional Readings

- Schwalbe: 7 “Project Quality Management”
- URLs
  - “Introduction to Software Testing”
    - http://www.iplbath.com/pdf/p0820.pdf

Questions?