Software Project Management

The Core View

The Mythical Man Power

Can one person build the Eiffel tower?

- Facts:

☞ Construction started 7/1/ 1887

☞ Completed 22 months later

☞ 18000 pieces used to build the tower

☞ Between 150-300 workers were on the building site daily.

The Mythical Man Power

Can one person build the Eiffel tower?

- Answers:
  - In theory: yes
  - In Practice: No
- Why?
  - Let us do the math … calculate the person-hours required to complete the project:
    - *= 22 months*20 business days * ((150+300)/2 HC)*8hours*
    - *= 792000 working hours*
    - *= 4950 months*
    - *=412.5 years*
  - The one person has a capacity that can't be exceeded; only 2 hands and 2 legs.

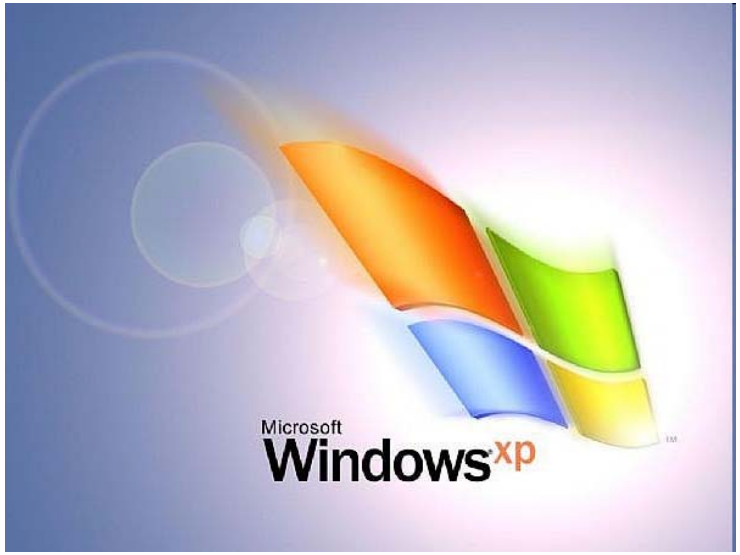
The Software Project vs. Eiffel Tower

Is the software project similar to Eiffel tower?

- Yes it is, you need:
  - Physical Resources
  - Human Resources:
    - ✓ Architect
    - ✓ Developers
    - ✓ Manager
    - ✓ Etc.
  - Budget
  - Plan
  - Quality

Software Product Size

- Windows Vista is said to have over 50 million lines of code, whereas XP was said to have around 40 million



Software Product Size

- Red Hat Linux version 7.1 (released April 2001) contained over 30 million SLOC

Software Product Size

- Mac OS X 10.4 has about 86 million SLOC

Software Product Size

- Debian 5.0 /GNU Linux has about 324 million SLOC

Software Product Size

- The Android operating system consists of 12 million lines of code SLOC

Software Product Size

- The Navigation system in the current S-class Mercedes-Benz requires over 20 million lines of code

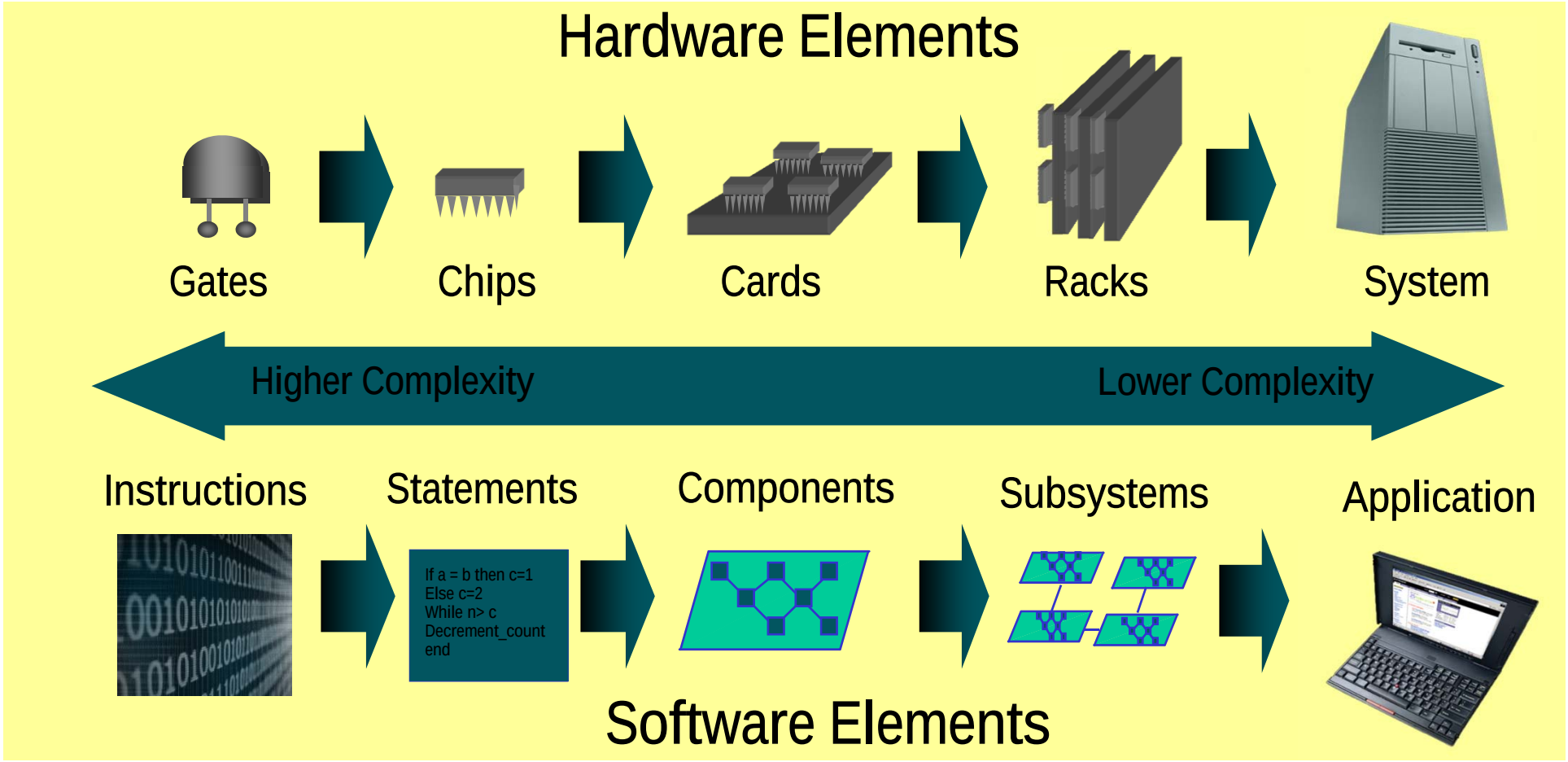
Software Product Size

- The avionics system in the F-22 Raptor, the current U.S. Air Force frontline jet fighter, consists of about 1.7 million lines of software code.

- The F-35 Joint Strike Fighter requires about 5.7 million lines of code to operate its onboard systems.

- Boeing's new 787 Dreamliner requires about 6.5 million lines of software code to operate its avionics and onboard support systems.

Software vs. Hardware



Software Project

What is a Software Project?

A sequence of connected and related activities ***(requirement engineering, system engineering, coding, testing, documentation, controlling, …)*** that must be completed by a specific time, within budget, and according to specification.

Software Crisis

Many software-related failures: auto-pilot systems, air traffic control systems, banking systems, IRS.

- On January 15, 1990, the AT&T long-distance telephone network broke down, interrupting long-distance telephone services in US for over 8 hours.
- On June 4, 1996, the maiden flight of the new and improved Ariane 5 rocket exploded 37 seconds after lift-off.
- On June 8, 2001, a software problem caused the NYSE to shut down the entire trading floor for over an hour.

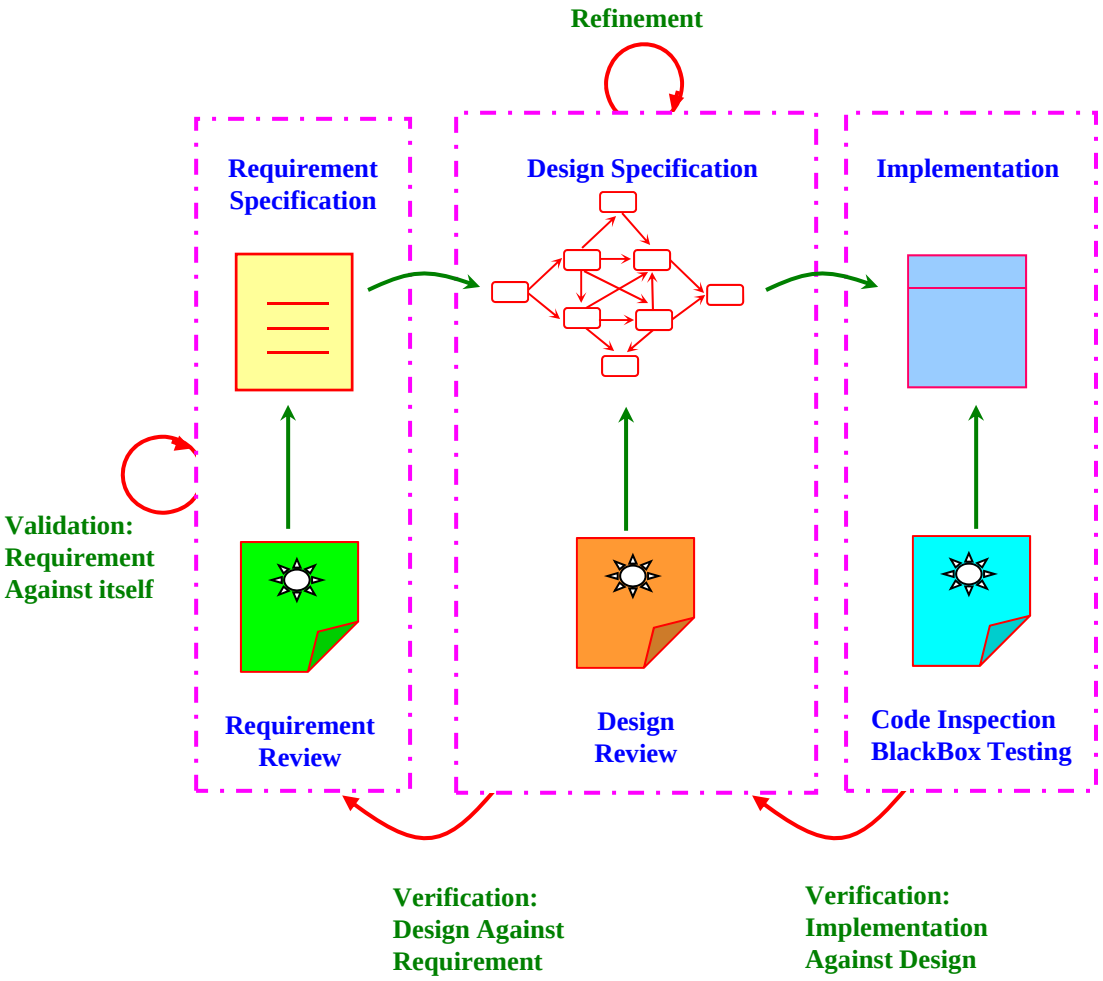
Software Project

What is the goal?

Balance the main three (other 2 constraints scope and resource) … in order to:

- Stay within the budget
- Deliver on time to gain market share
- Exceed customer satisfaction

Verification and Validation



Software Project Expenditures

Activity	Cost
Management	5%
Requirements	5%
Design	10%
Code and Unit Testing	30%
Integration and Acceptance testing	40%
Deployment	5%
Environment	5%
Total	100%

The 80% Benchmark

80%	Of the engineering is consumed by 20% of the requirements
80%	Of the software cost is consumed by 20% of the components
80%	Of the errors are caused by 20% of the components
80%	Of software scrap and rework is caused by 20% of the errors
80%	Of the resources are consumed by 20% of the components
80%	Of the engineering is accomplished by 20% of the tools
80%	Of the progress is made by 20% of the people

Conventional Software Management Performance

Barry Boehm's "Industrial Software Metrics Top 10 List":

1. Finding and fixing a software problem after delivery costs 100 times more than finding and fixing the problem in early design phases
2. You can compress software development schedules 25% of nominal, but no more
3. Fore every $1 you spend on development, you will spend $2 on maintenance
4. Software development and maintenance costs are primarily a function of source lines of code.
5. Variations among people account for the biggest difference in software productivity; hire good people to succeed.

Conventional Software Management Performance

6. The overall ratio of software to hardware costs is still growing.
7. Only about 15%of software development effort is devoted to programming
8. Software systems and products typically cost 3 times as much per SLOC as individual software programs. Software system products (system of systems) costs 9 times as much
9. Walkthroughs catch 60% of the errors
10. 80% of the contributions comes from 20% of the contributors.

Improving Software Economics

Cost Model Parameters	Trends
Size Abstraction and component-based development technologies	•Higher Order Languages •Object-Oriented OAD/OOP Reuse •Commercial Components
Process Methods and techniques	•Iterative development •Process maturity models •Architecture-first development
Personnel People factors	•Training and Personnel skill development •Teamwork •Win-win cultures
Environment Automation technologies and tools	•Integrated tool (visual modeling, compiler, editor, debugger ..) •Automation of coding, documents, testing
Quality Performance, reliability, accuracy	•Hardware platform performance •Statistical Quality Control

Software Project

How to achieve this goal?

Follow the documented processes when executing the following phases of the software project:

1. Define - Scope the project
2. Plan - Develop the project plan
3. Execute - Launch the plan
4. Monitor - Monitor/ control project progress
5. Close - Close out the project

Software Project

What is the software development process?

A process **is a set of documented** procedures, methods, practices, and tools used to produce a software product.

The process will answer the following:

- What to do? Tasks/activities
- How to do it? Procedure/practice
- When to do it? Sequence of activities
- What are the artifacts? (input/output)

Software Project

What is the difference between a process and Procedure/Methodology?

- The Programmer knows the procedure to create a Java program, based on the language syntax, compile the program, and then run it
- The Designer applies some OOD methodology when writing the detailed design document

Software Project

- But both the programmer and designer may produce some artifacts that are
  - ❑ Not compliant with the requirements
  - ❑ Hard to maintain
  - ❑ Hard to comprehend, not following consistent writing style
  - ❑ Not with acceptable quality
  - ❑ Not within acceptable milestones

The problem: we didn't know
the micro-steps of what/how/when/where it happened !

Software Project

If the programmer and designer follow the **process**, then the artifacts they produce will be

- ✓ Predictable
- ✓ Based on the requirements
- ✓ Easy to maintain and control
- ✓ consistent with the writing style
- ✓ Of acceptable quality
- ✓ within acceptable milestones

By following the process, we will be able to know precisely what/how/when/where it happened !

Software Project

The **process is**

- ✓ Not for Heroes
- ✓ For average technical staff to use
- ✓ For technical staff with software skills but limited capacity
- ✓ Not to increase productivity; in fact it may decrease it

The internet browser didn’t require a hero to produce, it required a skilled team with a plan

Software Processes

Software Process is an overloaded term

- **Metaprocess**: an organization's policies, procedures, and practices for pursuing a software-intensive line of business; the focus is on organizational economics, an dlong-term strategies.
- **Macroproces**: the project's policies, procedures, and practices for producing a complete software product within certain cost, schedule, and quality constraints.
- **Microprocess**: a project team's policies, procedures, and practices for achieving an artifact of the software process.

Attributes of Software Processes

Attributes	Metaprocess	Macroprocess	Microprocess
Subject	•Line of business	•Project	•Iteration
Objectives	•Business profitability •Competitiveness	•Project profitability •Risk Management •Project budget, schedule, quality	•Resource management •Risk resolution •Milestone budget, schedule, quality
Audience	•Customers •Organizational management	•Software project managers •Software engineers	•Subproject managers •Software engineers
Metrics	•Project predictability •Revenue, market share	•On budget, on schedule •Major milestone success •Project scrap and rework	•On budget, on schedule •Major milestone progress •Release/iteration scrap and rework
Concerns	•Bureaucracy vs. Standardization	•Quality vs. Financial Performance	•Content vs. schedule
Time Scales	6 to 12 months	1 to many years	1 to 6 months

Software Project

Quality Engineering Principle:

The quality of the software system is controlled by the quality of the process used to produce that software.

Quality Management Principle:

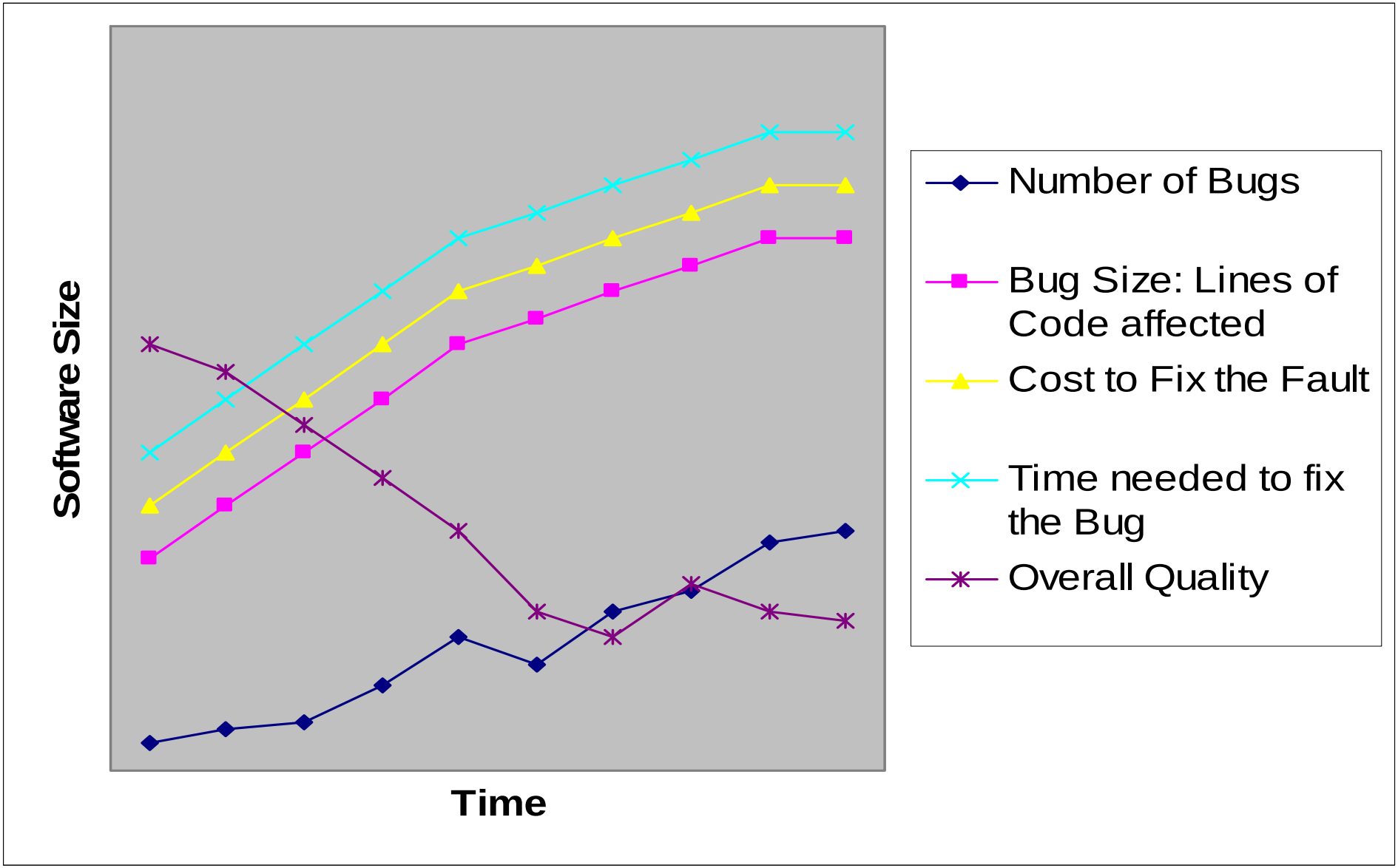
- Document the process
- Measure the process
- Improve the process based on the measurement

Hardware vs. Software

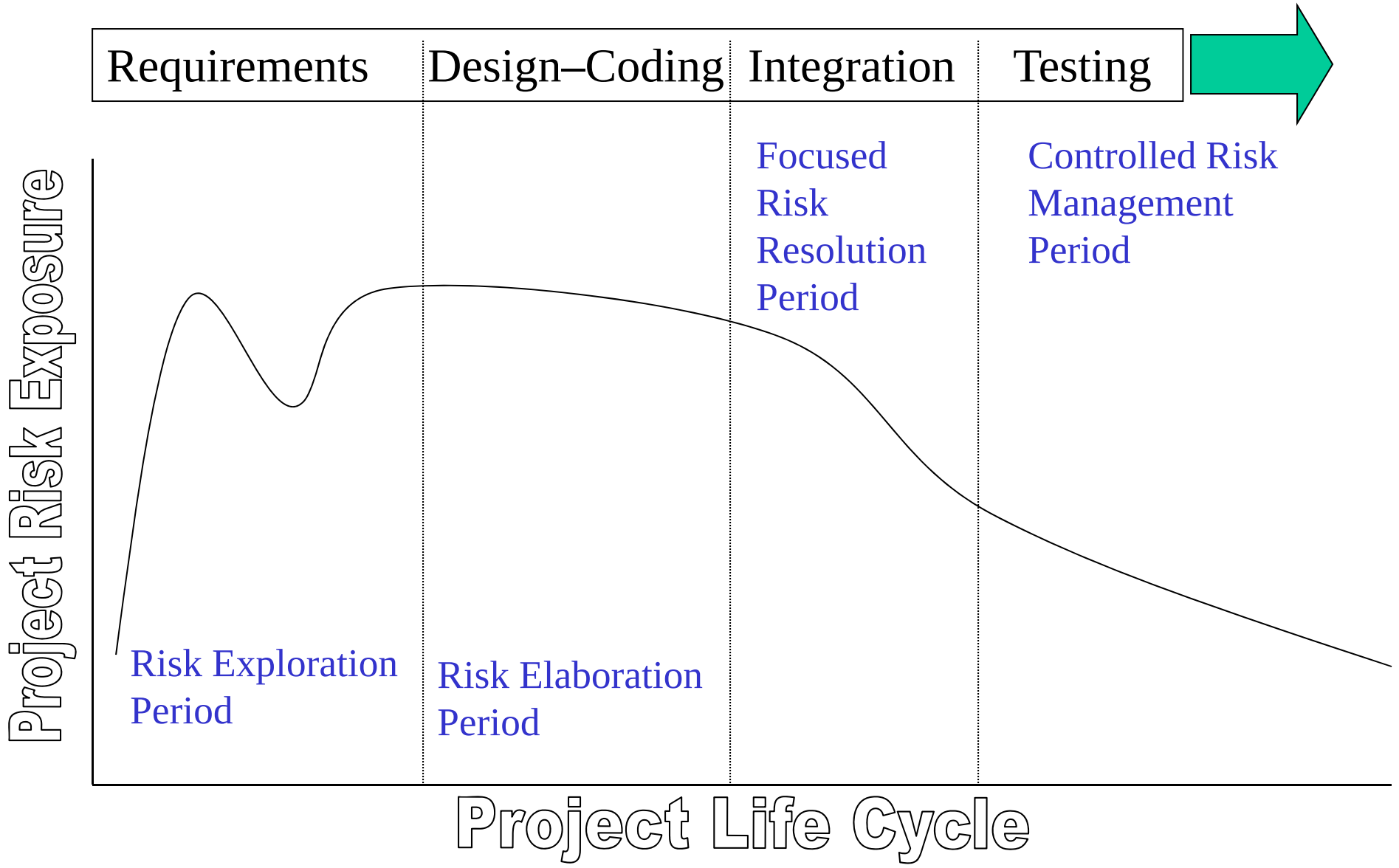
- Both Hardware and Software are becoming component-oriented; assemble system from basic components through plug-and-play
- Hardware: Getting smaller, cheaper, and faster

- Software: Getting larger
  - Is it really getting more expensive?
    - Customers demand more features
    - Regression testing cost becomes a real burden on the budget
    - The Overall ratio of software to hardware costs is still growing.
  - Is it really getting slower?
    - More feature interactions may cause a slow-down
    - Processor speed doubles every 18 months

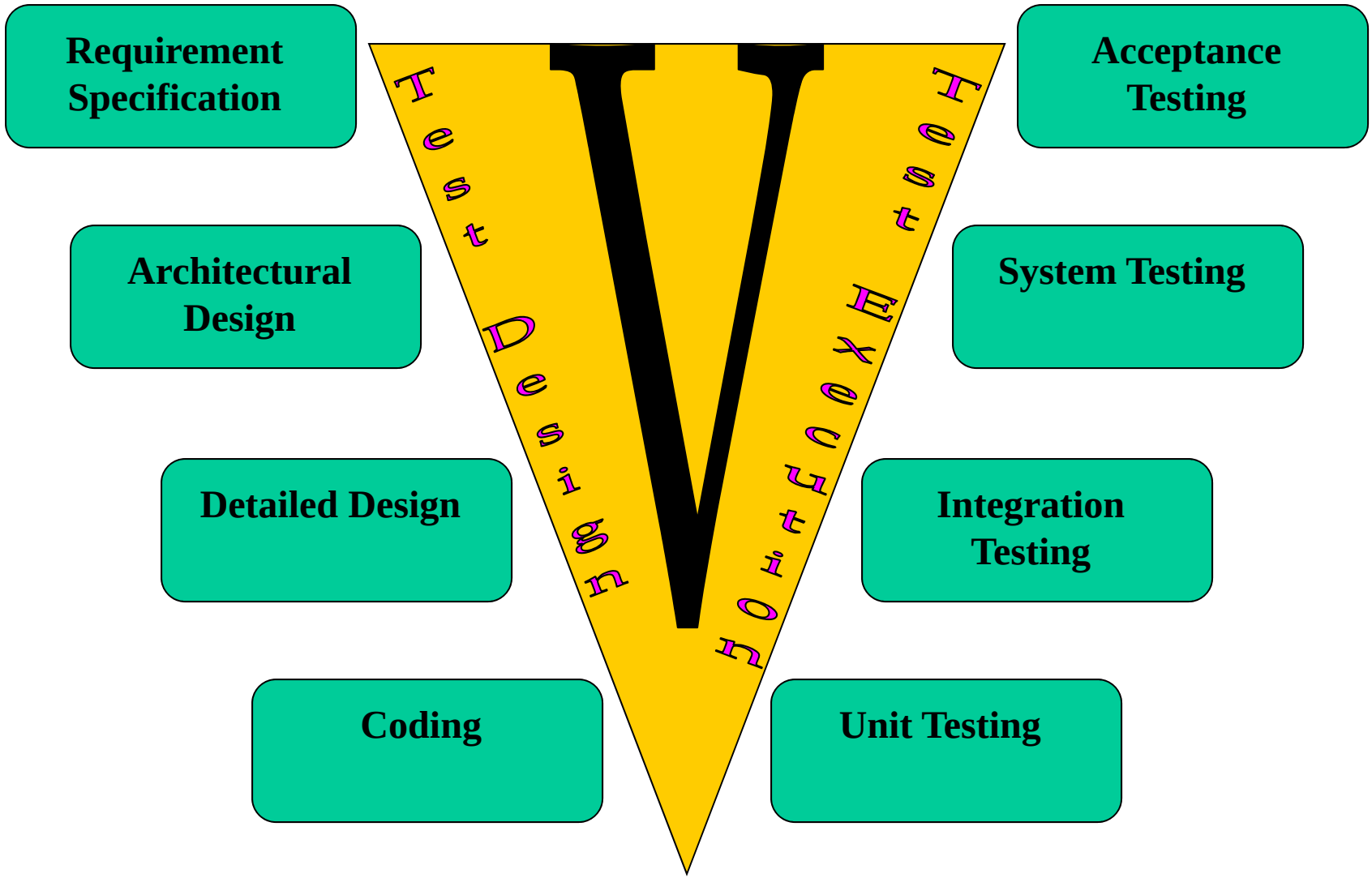
Software Trends



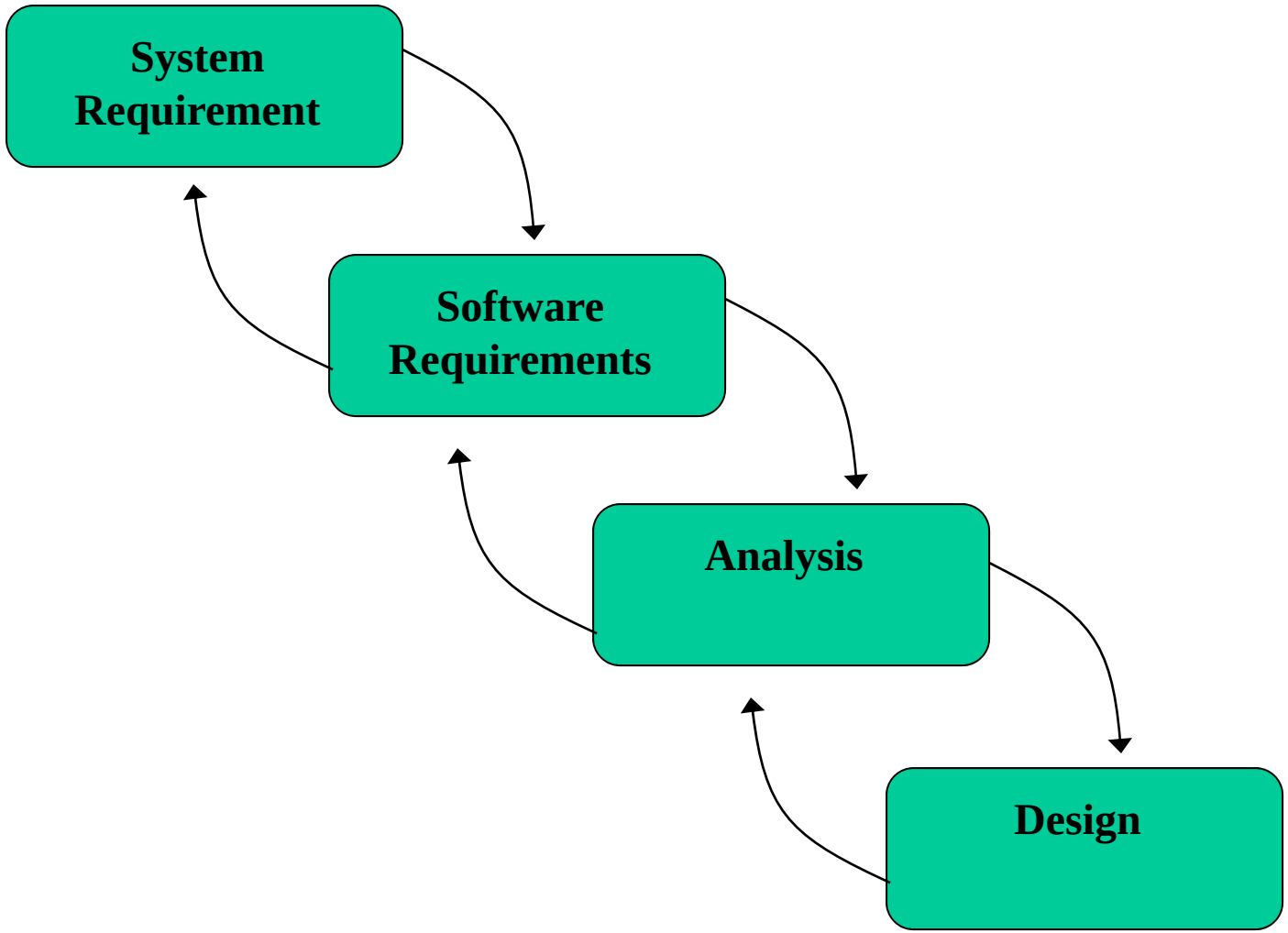
Risks of the Software Projects



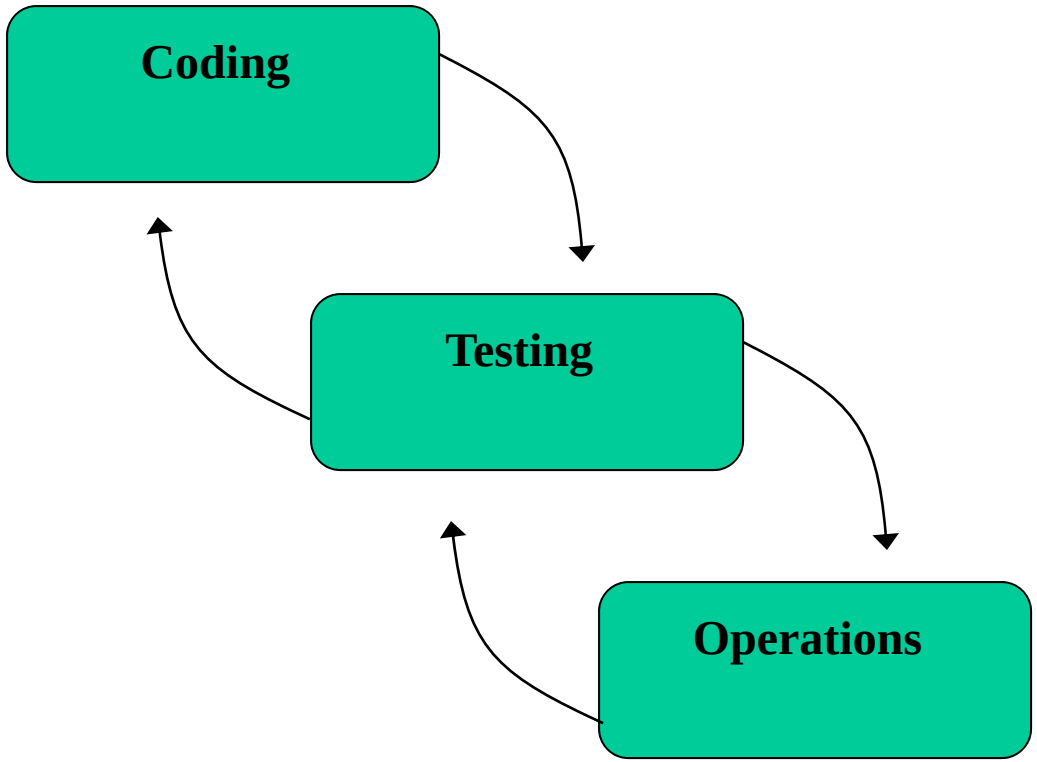
The V-Model of the Software Development



Waterfall Model of Software Development



Waterfall Model of Software Development



Software System Stakeholders

Customer	Manager	Architect	Developer	Tester
Requirements Cost Schedule Performance Reliability Security	Cost Schedule Requirements Process Resources	Maintainability Portability Feasibility Reusability Extensibility Flexibility The ***ilities***	Components Connectors Class/Pattern Data flow Reuse Flexibility Extensibility	Functionality Requirements Regression Tools Simulators

Requirement Driven Approach

The ATM machine's Requirement & Specification Document, RSD, may have the following requirements:

<Req 1> The system shall provide a text field that will allow the user to enter the pin number

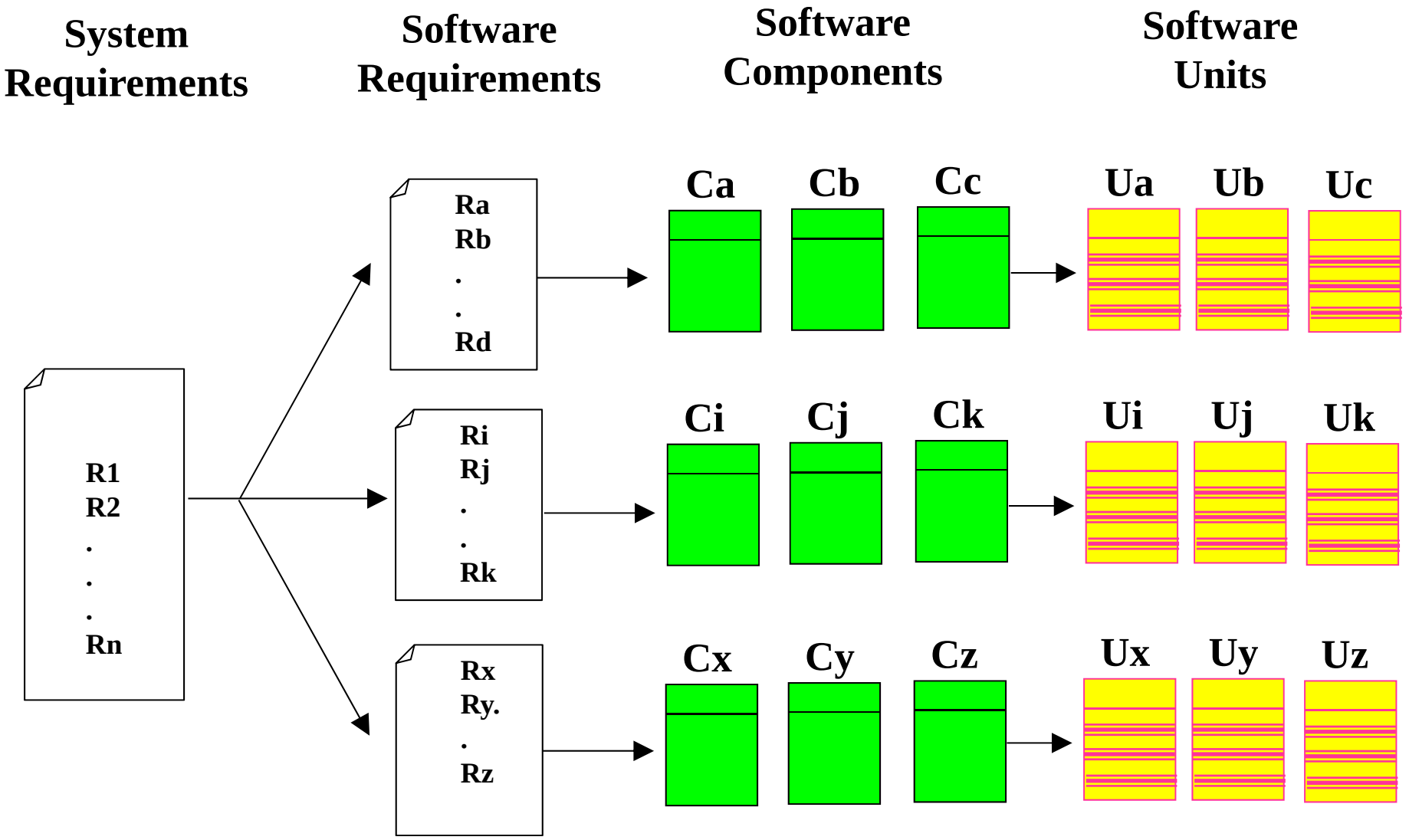
<Req 2> If the user entered a valid pin, the system shall provide a menu for the user in order to select a transaction

<Req 3> ….

<Req 4> …

<Req 5> …

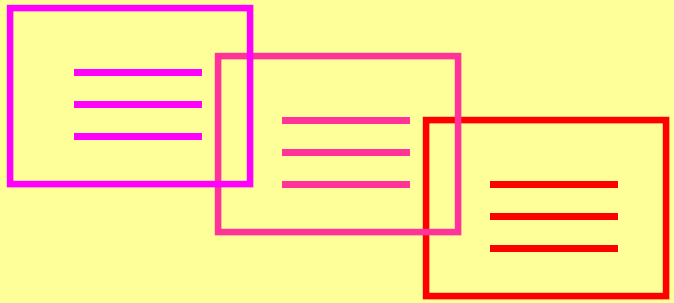
Requirement Driven Approach



Software Requirements and Specifications

Software Requirements and specification shall have the following properties:

- Abstract
- Concise
- Complete
- Unambiguous
- Maintainable
- Comprehensible
- Cost-Effective



Software Requirements and Specifications

- **Can we automate the process of mapping informal specs into formal specs?**
- **Can we get rid off the prose requirements?**
- **How about a semi-automated mapping process?**

The focus here is to find a semi-automated mapping process from the prose requirements to the formal mathematical model

Software Requirements and Specifications

- Mathematical models are used to prove correctness and precision of requirements, while software engineering principles are used to produce the actual products

Kathryn L. Heninger (worked for DoD + D. Pranas) listed 3 Principles for requirement document design:

1. State questions before trying to answer them
2. Separate concerns
3. Be as formal as possible

Software Requirements and Specifications

- Kathryn L. Heninger developed the following objectives for the existing flight software for the Navy's A-7 requirement document.

1. **Specify external behavior only**
2. **Specify constraints on the implementation**
   (Be alarmed to this one, at some point we said We want to have implementation-free specifications)
3. **Be easy to change**
4. **Serve as a reference tool**
5. **Record forethought about the life cycle of the system**
6. **Characterize acceptable responses to undesired events**

Science vs. Engineering

- Science focuses on ***facts+theories+formulas+proofs*** about the world
- Engineering emphasizes on how to apply the ***facts+theories+formulas*** ***in the*** design of the products
- When we talk about software, recognize the:
  - Impact of engineering on software development
  - Impact of people on software development

Science vs. Engineering

As a professional software engineer, prepare an answer for this question:
Do you write the specification of the car based on the road that you will use?

Science vs. Engineering

- Computer Science:
  - Is NOT about teaching tools; you should learn tools on your own
  - Is NOT ONLY how to use a programming language: syntax+semantic. Many programming languages come and go without us being aware that they even existed. What happened to Algol68, Pascal, …? However y=mx+b is still here and will continue to be there.
  - Is about automata theory, language theory
  - Discrete mathematics, Algorithms
  - Is about formal specifications and proof of correctness; programs and their mathematical descriptions are two different things.

Software Engineering vs. Software Project Management

- Software Engineering applies the sound scientific theories and principles to produce **software products**
- Software Project Management is the art to define, plan, execute, and monitor the activities that will bring **software products** to existence.

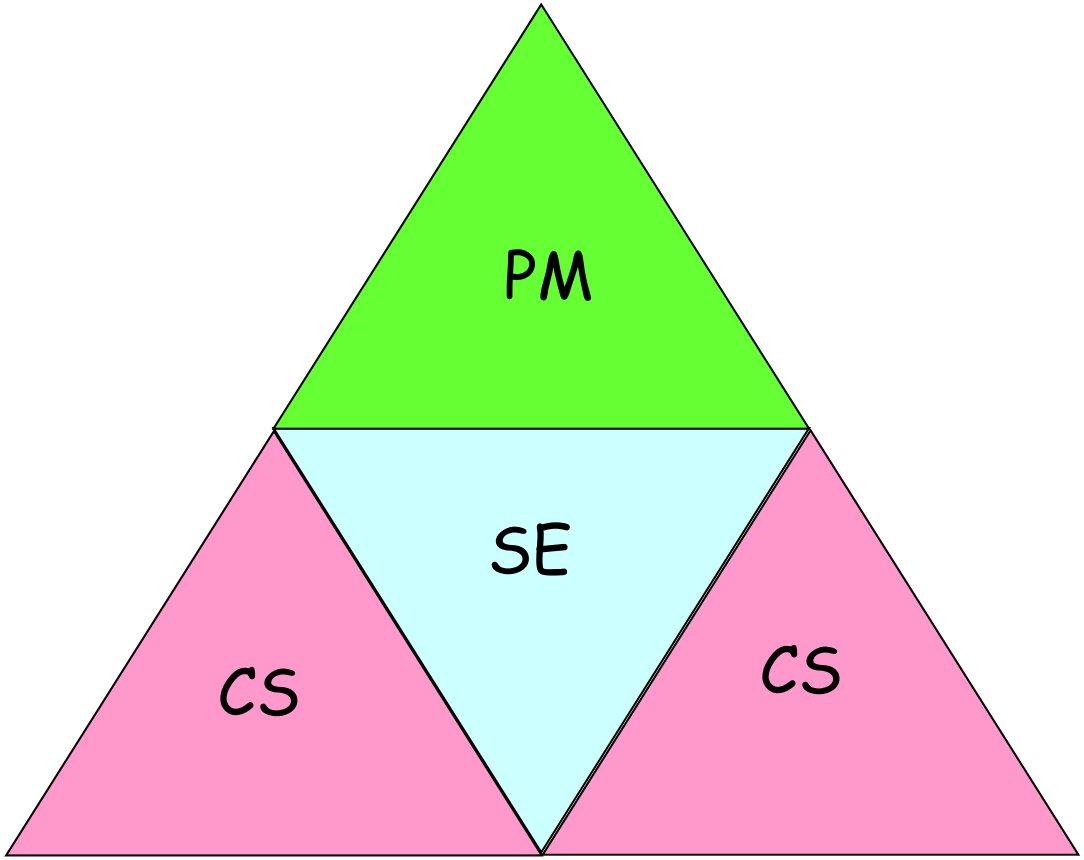
Programming as Engineering

- David Parnas in his famous commentary paper "Teaching Programming as Engineering" listed the following as the Mathematics needed for professional programming:

1. *Finite State Machines*
2. *Set, Functions, Relations, Composition*
3. *Mathematical Logic Based on Finite Sets*
4. *Programs as "Initial states"*
5. *Programs as Descriptions of State Sequences*
6. *Programs Described by functions from starting states to stopping state*
7. *Tabular Descriptions of Functions and Relations*

Project Managers

First, know where you are!



Project Managers

- Growing demand for software project managers
  - Organizations have become customer-driven.
  - Organizations have evolved from function to process structures.
  - Organizations are using task forces more frequently.
  - Organizations have become more project-oriented.

- From the organization perspective, project managers are needed to:
  - Gain market share
  - Be first to market
  - Stay profitable
  - Maintain Quality

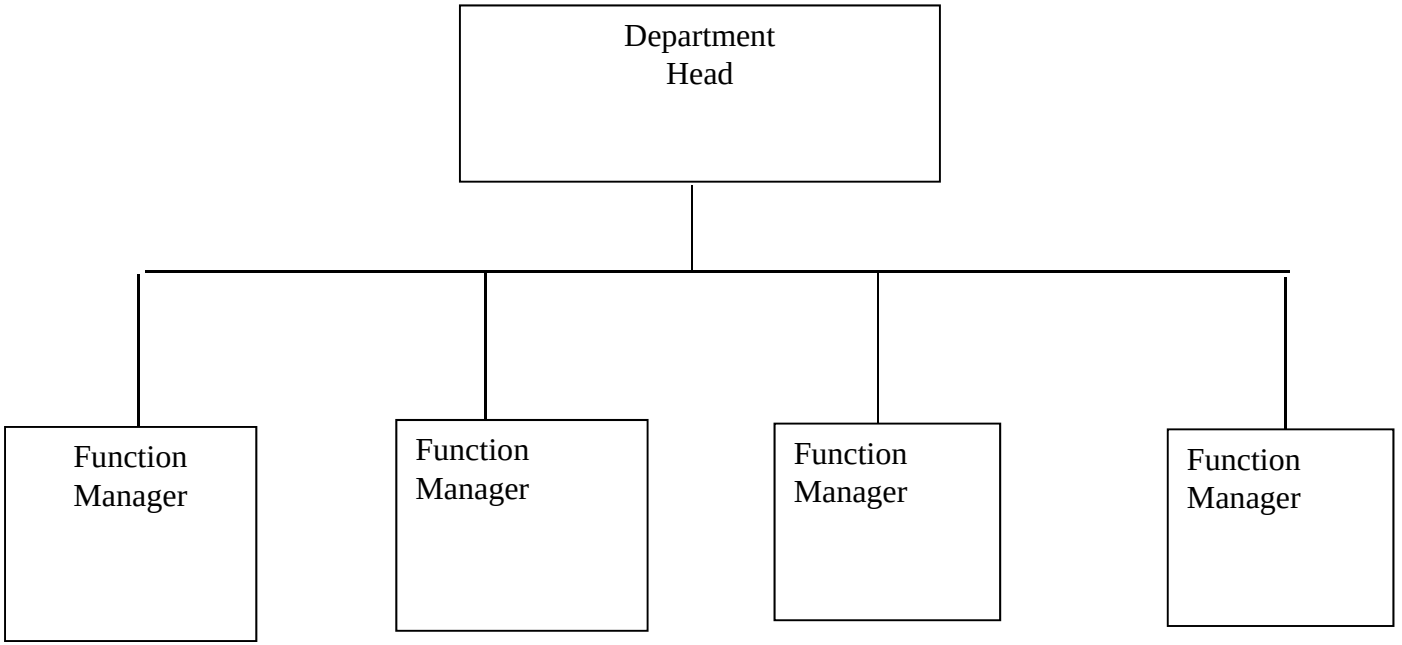
Project Managers

- Project Managers are mainly responsible to all issues related to the software project; issues may vary depending on the project scale, some of the common issues are:
  - Schedule
  - Budget
  - Quality
  - Delivery of products
  - Locking resources
- Bottom line, as a project manager you will notice that most of your time is consumed chasing and collecting the status of project tasks.

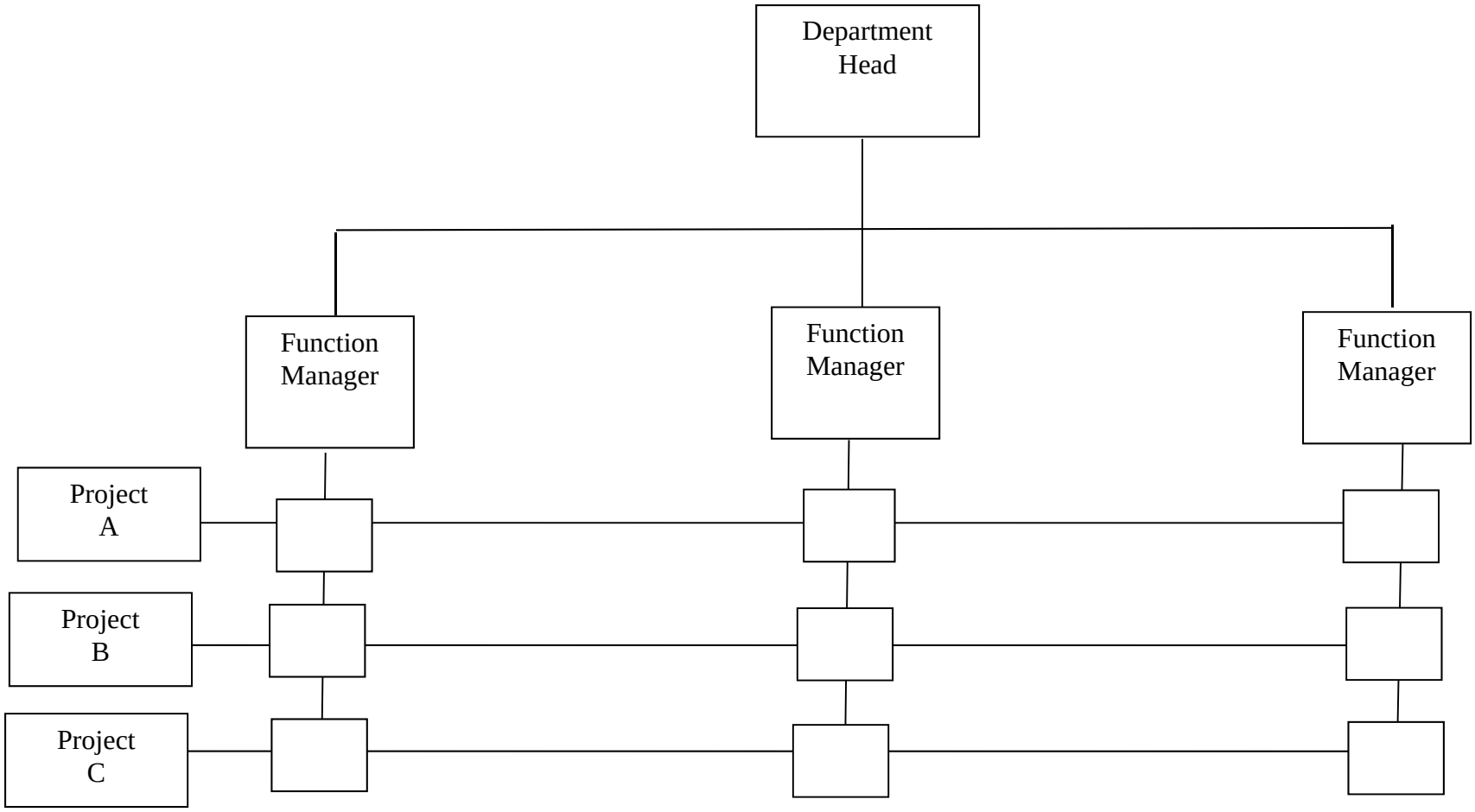
Organizational Environments

- The functional structure
- The matrix structure

The Functional Structure



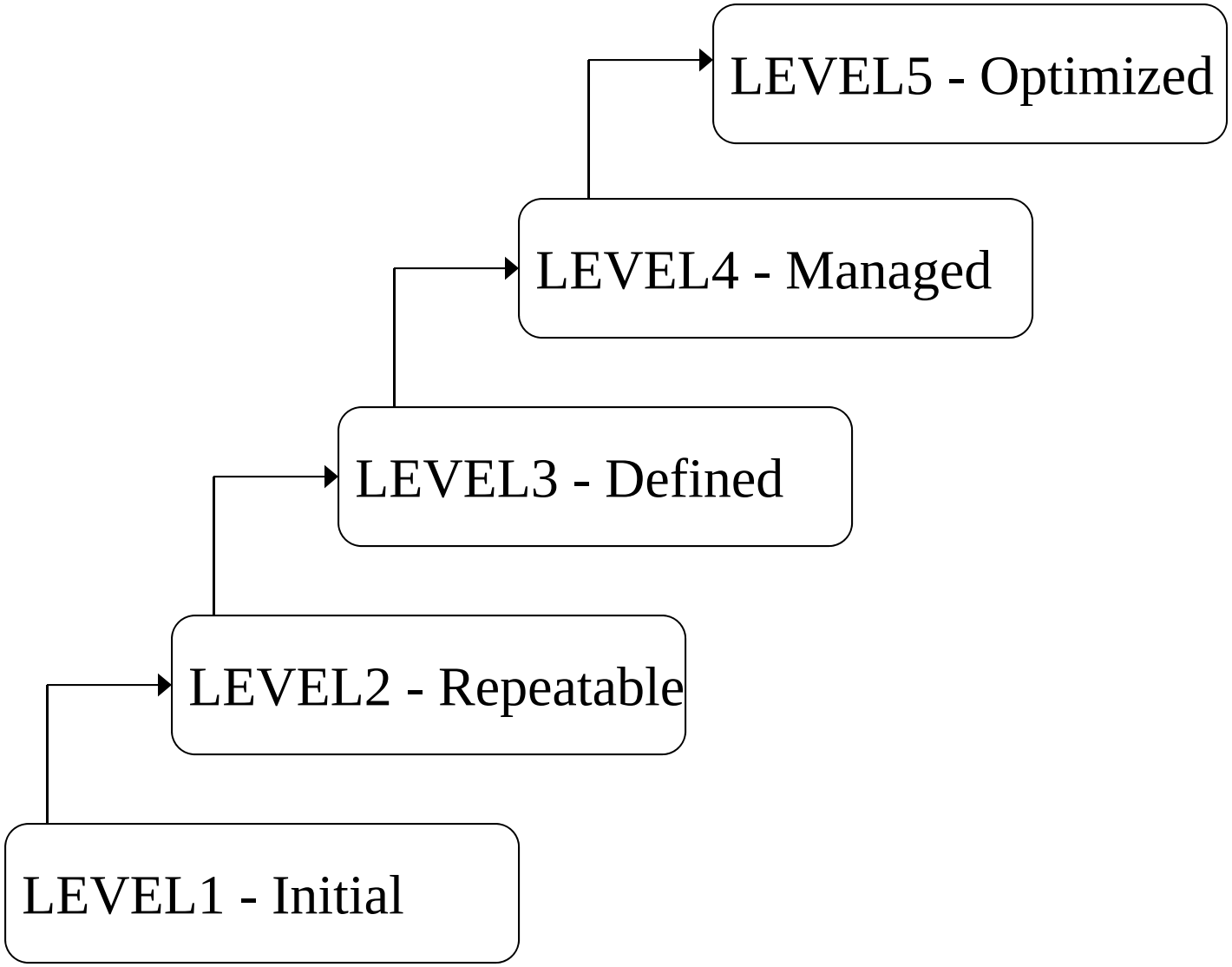
The Matrix Structure



The CMM : SEI's Capability Maturity Model

- The CMM for Software is a framework that was developed by the Software Engineering Institute, often referred to as SEI, at Carnegie Mellon University by observing the best practices in software and other organizations.

The CMM : SEI's Capability Maturity Model



CMM's KPA : Key Process Area

- Each level has a Key Process Area , KPA, except Level 1
- Level 2 – Repeatable
  - Requirements Management
  - Software Project Planning
  - Software Project Tracking & Oversight
  - Software Subcontract Management
  - Software Quality Assurance
  - Software Configuration Management

CMM's KPA : Key Process Area

- Level 3 – Defined
  - Organization Process Focus
  - Organization Process Definition
  - Training Program
  - Integrated Software Management
  - Software Product Engineering
  - Intergroup Coordination
  - Peer Reviews

CMM's KPA : Key Process Area

- Level 4 - Managed
  - Software Quality Management
  - Quantitative Process Management

CMM's KPA : Key Process Area

- Level 5 - Optimizing
  - Process Change Management
  - Technology Change Management
  - Defect Prevention

The Project Manager Responsibilities

1. Project planning
2. Managing the project
3. Lead project team
4. Building client partnerships
5. Targeting to the business

Classifying Project Managers

Managers:

- **Who is who?**
- **Who manages what?**

- Four classifications for project managers:
  - **Team leader**
  - **Project manager**
  - **Senior project manager**
  - **Program manager**

Classifying Project Managers

- **Team leader.** Team leaders have responsibility for part of the project. They are generally assigned responsibility for an activity and can have a small number of staff assigned to the activity whose work they will manage.

- **Project manager.** This individual will have management responsibility for projects that are classified as simpler, less complex, lower risk, or not mission-critical. They are the more junior of the two classes of project manager

Classifying Project Managers

- **Senior project manager.** These are more experienced of the two project manager classes. They are qualified to manage projects that are more complex, higher risk, or mission-critical.

- **Program manager.** This classification is reserved for those individuals who have achieved the highest level of professionalism and experience in project management. They will often manage project managers on very complex or multiproject undertakings.

Assessing the Project Managers

- Two levels of characteristics determine success or failure as a project manager:
  - *Skills : can be measured and a person can acquire them through training*
  - *Competencies: . We can see them in practice, but we cannot measure them, can't be acquired through training; observing the non-verbal behavior is an example.*

Phases For
Software Project Management

Few Rules Before We Embark in SPM

- Horse power needed

Few Rules Before We Embark in SPM

- Man power needed

Few Rules Before We Embark in SPM

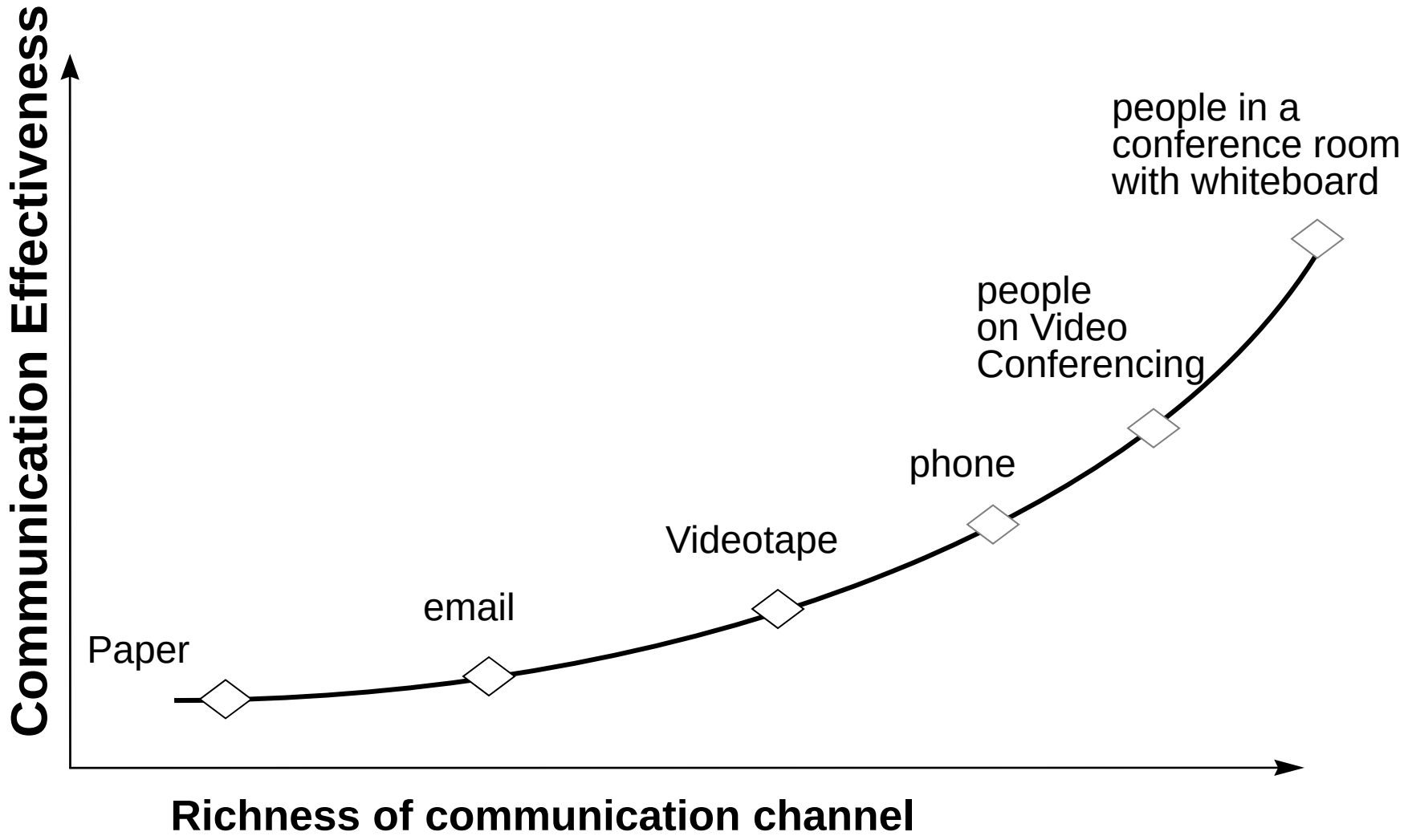
- Like the Ball-Game where the coach puts a plan to win the game based on **defense/offense strategy**, similarly in SPM you have to have a plan where :
    - Defense: skills, knowledge, experience
    - Offense: deliver on time, within budget, and with quality

Few Rules Before We Embark in SPM

- It is a TEAM work …

Few Rules Before We Embark in SPM

- And finally, communicate, communicate, and communicate!



Definition of a Project

- A *project* is a sequence of unique, complex, and connected activities having one goal or purpose and that must be completed by a specific time, within budget, and according to specification.

What is a Program?

- A *program* is a collection of projects.
- The projects must be completed in a specific order for the program to be considered complete. Because they compromise multiple projects, they are larger in scope than a single project.
- For example, the Untied States government has a space program that includes several projects such as the Challenger project. A construction company contracts a program to build an industrial technology park with several separate projects.

Project Parameters

- Five constraints operate on every project:
  1. Scope
  2. Quality
  3. Cost
  4. Time
  5. Resources
- A change in one of these constraints can cause a change in another constraint to restore the equilibrium of the project
- Let’s discuss each one of these in detail …

Scope

- *Scope* is a statement that defines the boundaries of the project. It tells not only what will be done but also what will not be done.
- In the information systems industry, scope is often referred to as a *functional specification*.
- In the engineering profession, it is generally called a *statement of work*.
- Scope may also be referred to as a document of understanding, a scooping statement, a project initiation document, and a project request form

Quality

- Two types of quality are part of every project:
  - The first is *product quality*. This refers to the quality of the deliverable form the project.
  - The second type of quality is *process quality*, which is the quality of the project management quality itself. The focus is on how well the project management process works and how can it be improved. Continuous quality improvement and process quality management are the tools used to measure process quality.

Cost

- The X-amount of dollars that it will cost to do the project is another variable that defines the project; the budget that has been established for the project.
- This is an important factor for projects that create deliverables that are sold to external customers

Time

- The customer specifies a timeframe within which the project must be completed.
- Cost and time are inversely related to one another. The time a project takes to be completed can be reduced, but cost increases as a result.

Resources

- Resources are assets, such as people, equipment, physical facilities, or inventory, that have limited availabilities, can be scheduled, or can leased from an outside party. Some are fixed, others are variable only in the long term. In any case, they are central to the scheduling of project activities and the orderly completion of the project .

Project Classification

Class	Duration	Risk	Complexity	Technology	Problems
A	> 18 months	High	High	Breakthrough	Certain
B	9-18 months	Medium	Medium	Current	Likely
C	3-9 months	Low	Low	Best of breed	Some
D	< 3 months	Very Low	Very Low	Practical	None

Phases of the Project Management

- There are five phases of the project management life cycle:
  1. Define - Scope the project
  2. Plan - Develop the project plan
  3. Execute - Launch the plan
  4. Monitor - Monitor/ control project progress
  5. Close - Close out the project

Scope the project

- State the problem/ opportunity.
- Establish the project plan.
- Define the project objectives.
- Identify the success criteria.
- List assumptions, risks, obstacles

Develop the project plan

- Identify the project activities.
- Estimate the activity duration.
- Determine resource requirements.
- Construct/ analyze the project network.
- Prepare the project proposal.

Launch the plan

- Recognize and organize the project team.
- Establish team operating rules.
- Level project resources.
- Schedule work packages.
- Document work packages.

Monitor/control project progress

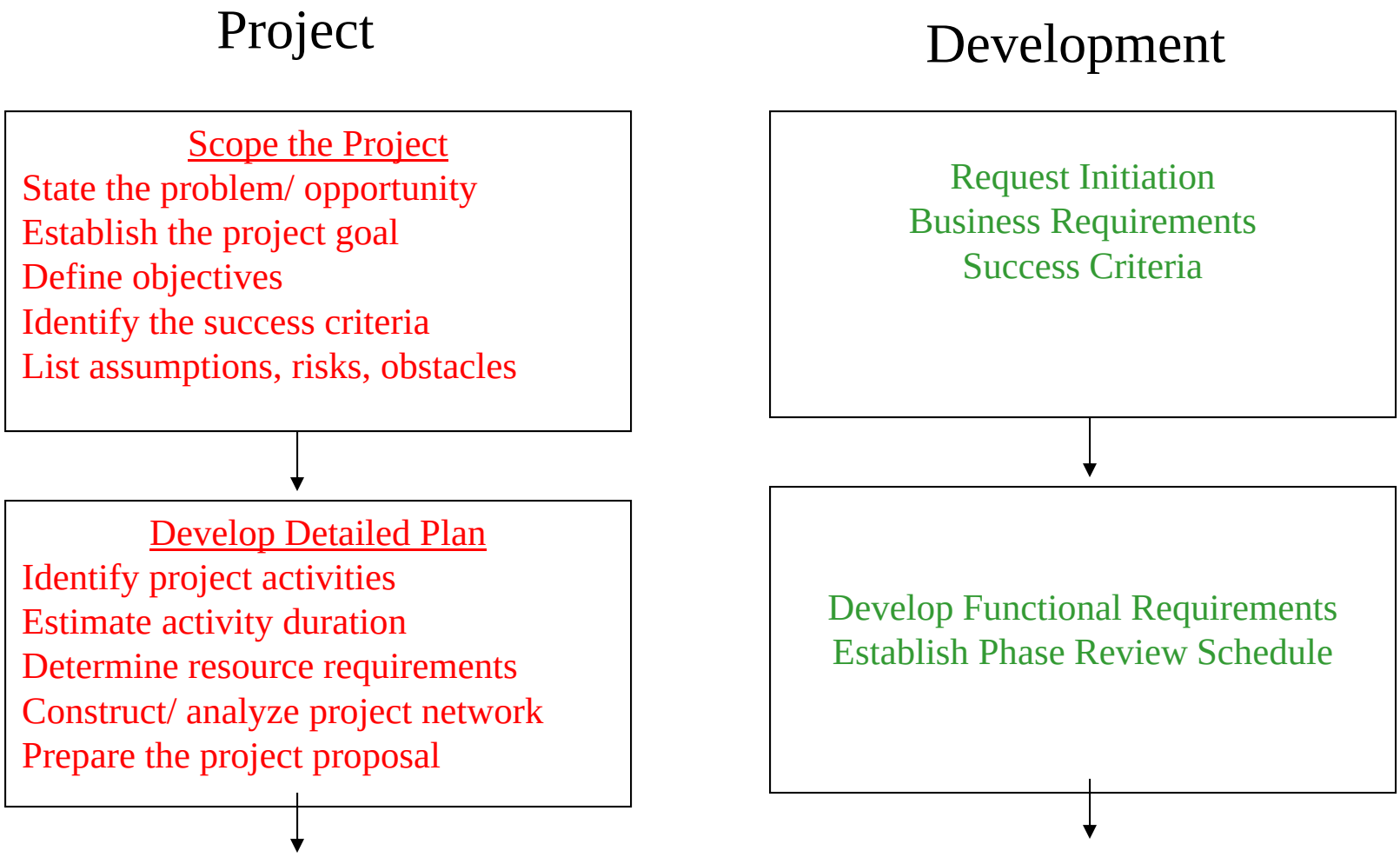
- Establish progress reporting systems.
- Install change control tools/process.
- Define problem-escalation process.
- Monitor project progress versus plan.
- Revise project plans.

Close out the project

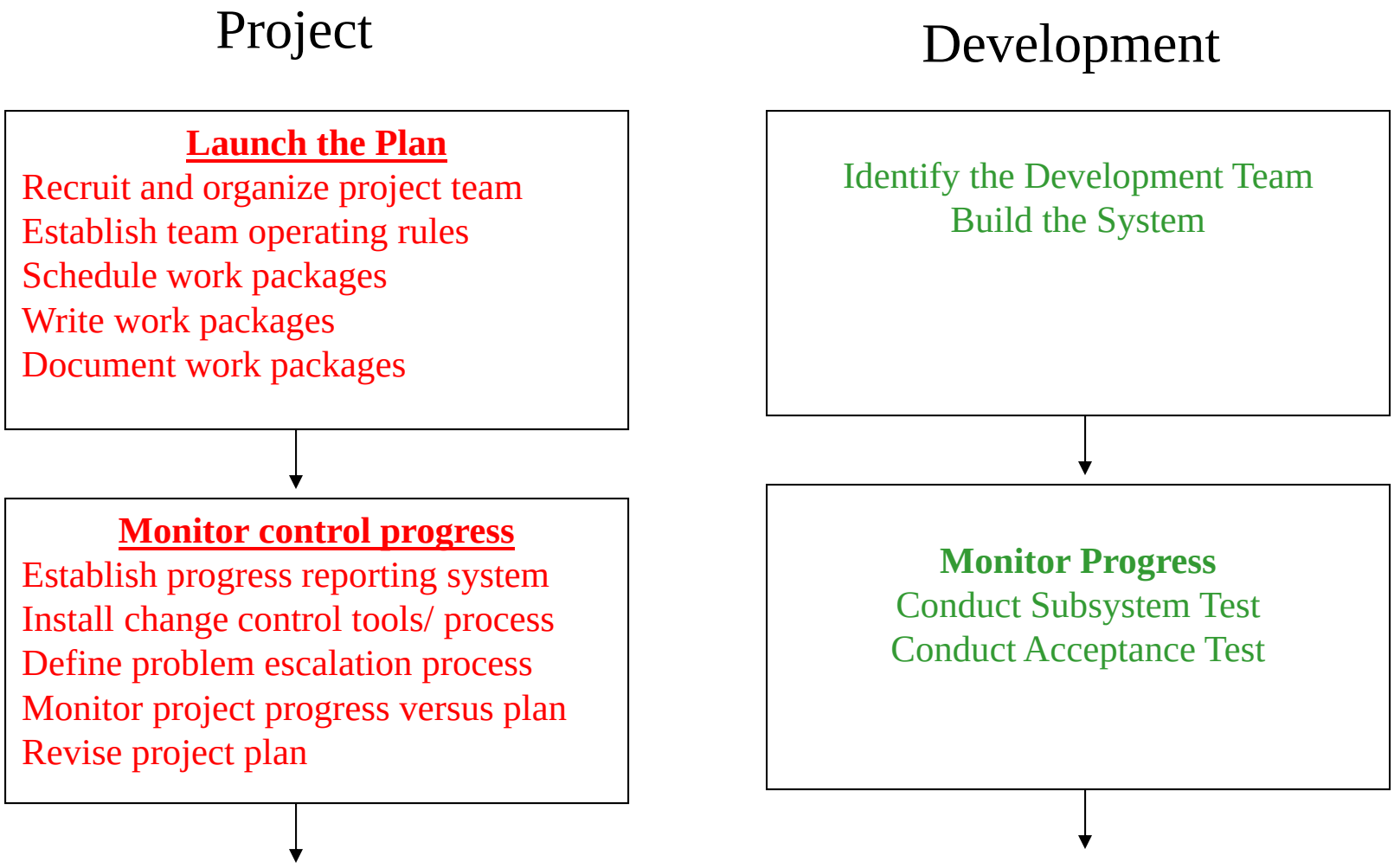
- Obtain client acceptance.
- Install project deliverables.
- Complete project documentation.
- Complete post-implementation audit.
- Issues final project report.

Project Management and Software Development life cycles

- There are similarities between the software development life cycle and the project development life cycle.



Project Management and Software Development life cycles



Project Management and Software Development life cycles

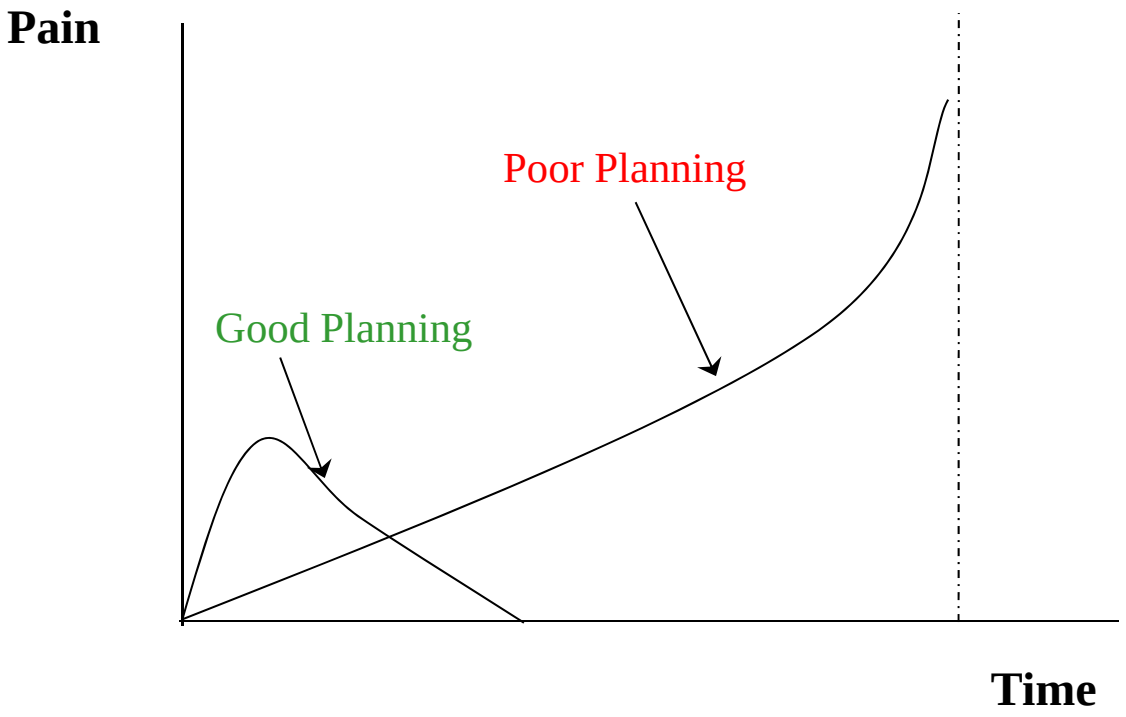
Project

Close out the project
Obtain client acceptance
Install project deliverables
Complete project documentation
Complete post implementation audit
Issue final project report

Development

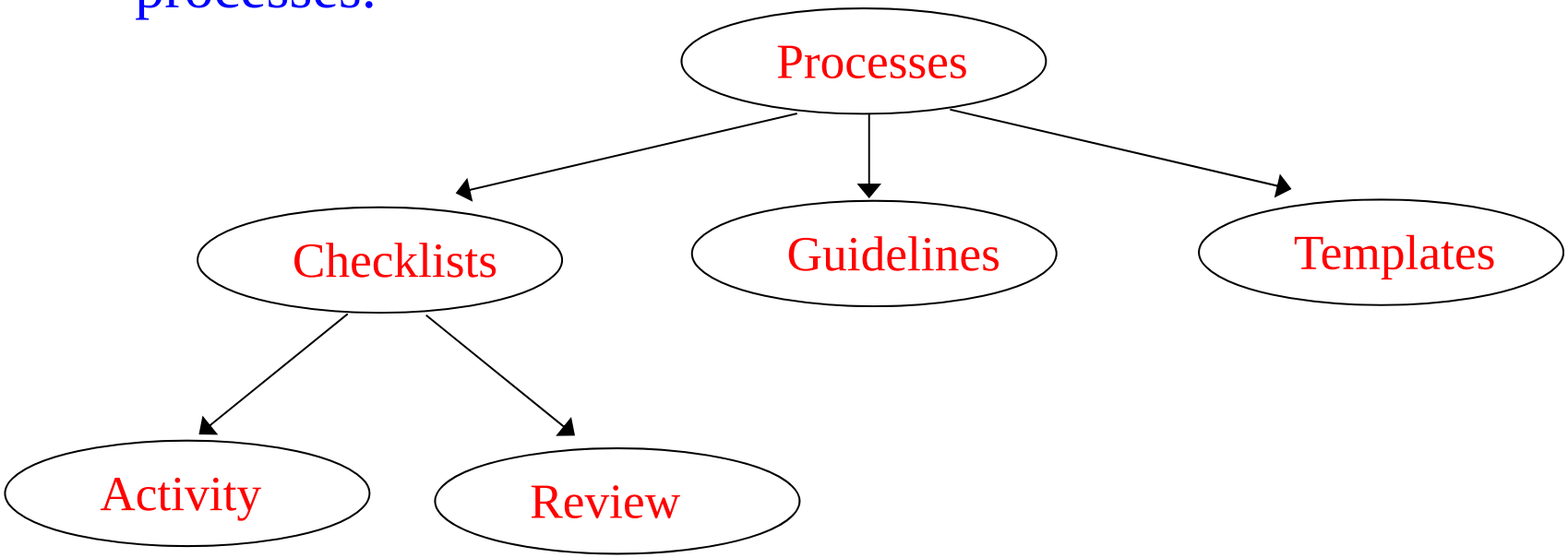
Evaluate System Performance
Conduct Post-Project Review

The Pain Curve for Project Planning



Process Assets and the Body of Knowledge System

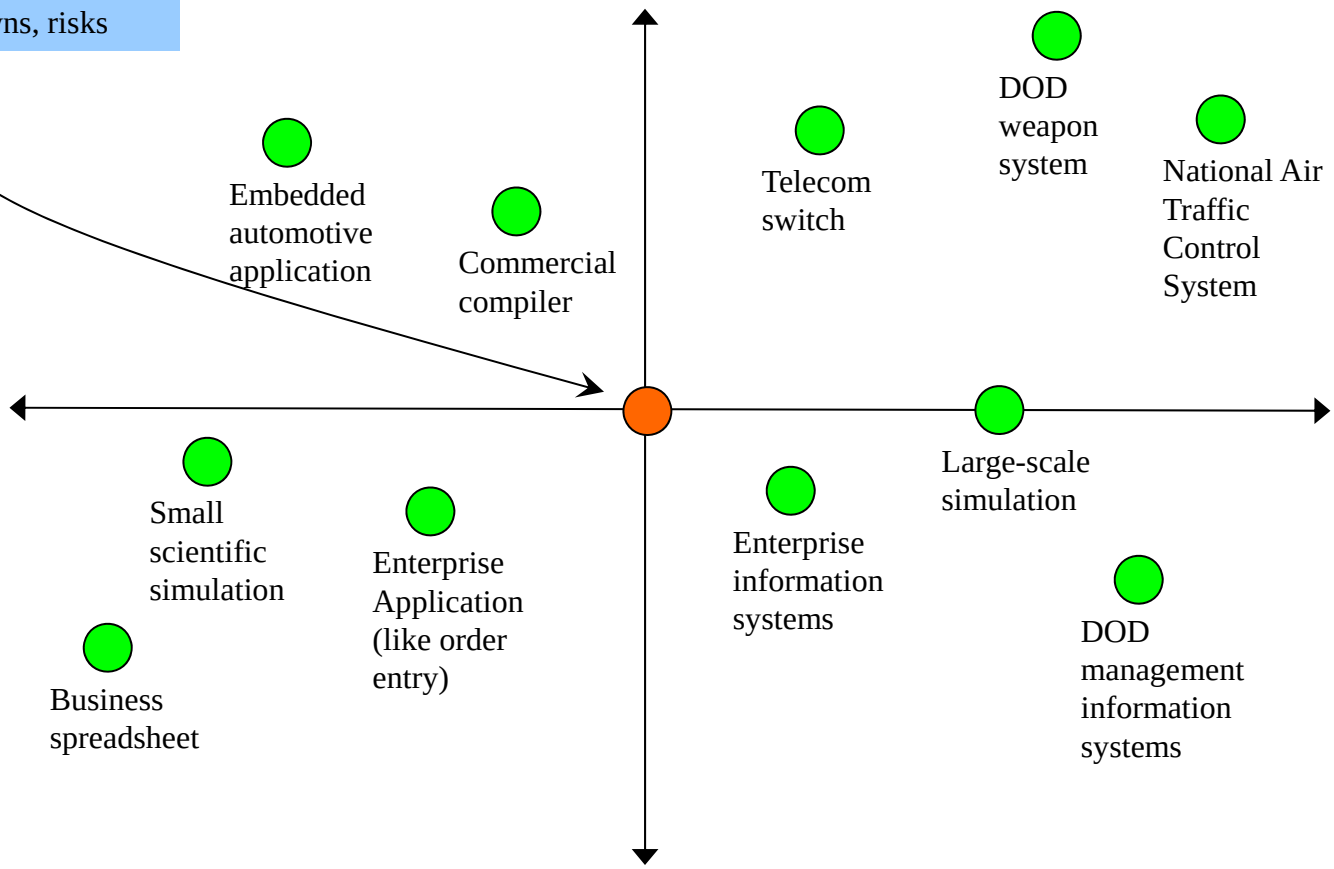
- A process encapsulates an organization's experience in form of successful recipes.
- Process descriptions,generally, contain the sequence of steps to be executed, who executes them, the entry/exit criteria for major steps, etc.
- Guidelines, checklists, and templates provide support to use the processes.



Process Variability

Average Software Project
- 5-10 people
- 10-12 months
- 3-5 external interfaces
- Some unknowns, risks

Higher Technical Complexity
- Embedded, real-time, distributed, fault-tolerant
- High-Performance, portable
- Unprecedented, architecture re-engineering



Lower Management Complexity
- Smaller scale
- Informal
- Few stakeholders
- Products

Higher Management Complexity
- Large scale
- Contractual
- Many stakeholders
- Project

Lower Technical Complexity
- Straightforward Automation, single thread
- Interactive performance, single platform
- Many precedent systems, application reengineering

State of the Practice in Software Management

- Factors that may influence the success or failure of the software projects could be:
    1. Social Factors
    2. Technology

State of the Practice in Software Management

Technologies on Unsuccessful Projects

- No historical software measurement data
- Failure to use automated estimating tool
- Failure to use automated planning tool
- Failure to monitor progress or milestones
- Failure to use effective architecture
- Failure to use effective development methods
- Failure to use design reviews
- Failure to use code inspections
- Failure to include formal risk management
- Informal, inadequate testing
- Manual design and specification
- More than 30% creep in user requirements

Technologies on Successful Projects

- Accurate software measurement
- Early use of estimating tools
- Continuous use of planning tool
- Formal progress reporting
- Formal architecture planning
- Formal development methods
- Formal design reviews
- Formal code inspections
- Formal risk management
- Formal testing methods
- Automated design and specification
- Automated configuration control
- Less than 10% creep in requirements

State of the Practice in Software Management

Social Factors on Unsuccessful Projects

- Excessive schedule pressure
- Executive rejection of estimates
- Severe friction with clients
- Divisive corporate politics
- Poor team communications
- Naïve senior executives
- Project management malpractice
- Unqualified technical staff
- Generalists used for critical tasks: quality assurance, testing, planning, estimating

Social factors on Successful Projects

- Realistic schedule pressure
- Executive understanding of estimates
- Cooperation with clients
- Congruent management goals
- Excellent team communications
- Experienced senior executives
- Capable Project management
- Capable technical staff
- Specialists used for critical tasks: quality assurance, testing, planning, estimating

Chaos of Software Projects

- Standish Group published a report on 1995 reaches the following conclusions:
  - U.S. companies would spend $81 billion on cancelled software projects in 1995
  - 31% of software projects studied were canceled before they were completed
  - 53% of software projects overran by more than 50%
  - Only 9% of software projects for large companies were delivered on time and within budget. For medium-sized companies the number was 16%, and for small-sized companies the number was 28%

Define the Project

- There is a need for clear understanding of exactly what was to be done. Project definition starts with the ***Conditions of Satisfaction*** document based on conversation with the customer.
- ***Project Overview Statement*** is generated from the Conditions of Satisfaction document .
- The Project Overview Statement clearly states what is to be done.
- Once the ***Project Overview Statement*** is approved, the scooping phase is complete.

The Project Overview Statement

- The Conditions of Satisfaction statement provides the input we need to generate the POS.
- The POS is a short document that concisely states what is to be done in the project, why it is to be done, and what business value it will provide to the organization when completed.
- The main purpose of the POS is to secure senior management approval and the resources needed to develop a detailed project plan.
- It will be reviewed by the managers who are responsible for setting priorities and deciding what projects to support. It is also a general statement, it is not detailed technical statement.

Parts of the POS

- The POS has five component parts:
  1. Problem/ opportunity
  2. Project goal
  3. Project objectives
  4. Success criteria
  5. Assumptions, risks, obstacles

S.M.A.R.T. characteristics for Goal

- **Doran's S.M.A.R.T. characteristics provide the criteria for a goal statement:**
  - **Specific: Be specific in targeting and objective.**
  - **Measurable: Establish measurable indicator(s) of progress.**
  - **Assignable: Make the object assignable to one person for completion.**
  - **Realistic: State what can realistically be done with available resources.**
  - **Time-related: State when the objective can be achieved; that is, duration**

POS Form – For Customer and PM

PROJECT OVERVIEW STATEMENT	Project Name	Project No.	Project Manager
Problem/ Opportunity			
Goal			
Objectives			
Success Criteria			
Assumptions, Risks, Obstacles			
Prepared by	Date	Approved By	Date

The Project Definition Statement : PDS

- Just as the customer and the project manager benefit from the POS, the project manager and project team can benefit from a closely related document, which we call the *Project Definition Statement (PDS).*
- The PDS uses the same form as the POS but incorporates ***considerably more detail***. The detailed information provided in the PDS is for the use of the project manager and the project team:

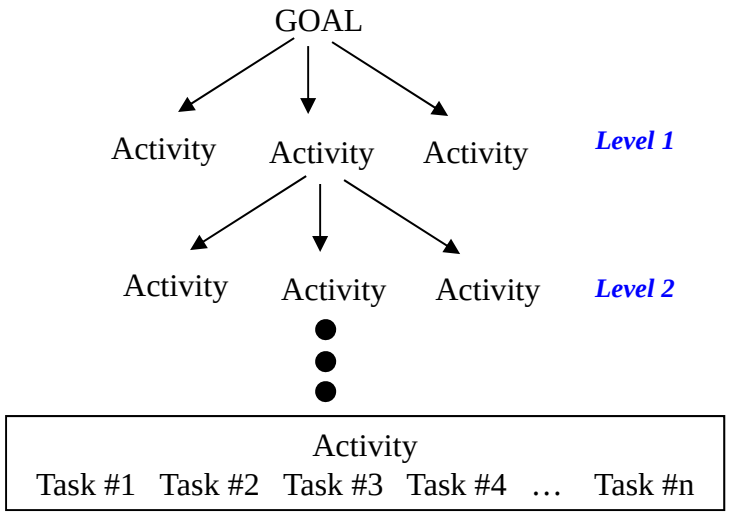
PDS Form

PROJECT DESCRIPTION STATEMENT	Project Name	Project No.	Project Manager
Problem/ Opportunity			
Goal			
Objectives			
Success Criteria			
Assumptions, Risks, Obstacles			
Prepared by	Date	Approved By	Date

The Work Breakdown Structure

- The Work Breakdown Structure (WBS) is a hierarchical description of the work that must be done to complete the project as defined in the Project Overview Statement (POS) .
- The WBS terms
  - *Activity* : An activity is simply a chunk of work.
  - *Task* : A task is a smaller chunk of work.

The Work Breakdown Structure



Uses for the WBS

- **Thought process tool:**
  - the WBS is a design and planning tool.
  - It helps the project manager and the project team visualize exactly how the work of the project can be defined and managed effectively.
- **Architectural design tool:**
  - the WBS is a picture of the work of the project and how the items of work are related to one another.

Uses for the WBS

- **Planning tool:**
  - In the planning phase, the WBS gives the project team a detailed representation of the project as a collection of activities that must be completed in order for the project to be completed.
  - It is at the lowest activity level of WBS that we will estimate effort, elapsed time, and resource requirements; build a schedule of when the work will be completed; and estimate deliverable dates and project completion .

Uses for the WBS

- **Project status reporting tool.**
  - The WBS is used as structure for reporting project status.
  - The project activities are consolidated from the bottom as lower-level activities are completed.
  - Completion of lower-level activities cause higher-level activities to be partially complete.
  - Therefore, WBS defines milestone events that can be reported to senior management and to the customer .

Generating the WBS

- The WBS is generated during the Joint Project Planning (JPP) session.
- Two different approaches to building the WBS:
  - Top-Down Approach
  - Bottom-Up Approach

Top-Down Approach

- The top-down approach begins at the goal level and successively partitions work down to lower levels of definition until the participants are satisfied that the work has been sufficiently defined.
- Once the project activities have been defined, they will allow you to estimate time, cost, and resource requirements first at the activity level and then aggregate to the project level.
- Two variations of this approach:
  - ***Team Approach***
  - ***Subteam Approach***

Top-Down Approach

- ***Team Approach***
  - The team approach requires more time to complete than the subteam approach even though it is the preferred approach.
  - In this approach the entire team works on all parts of the WBS. For each Level 1 activity, appoint the most knowledgeable member of the planning team to facilitate the further decomposition of that part of the WBS. Continue with similar appointments until the WBS is complete.
  - This approach allows all members of the planning team to pay particular attention to the WBS as it is developed, noting discrepancies and commenting on them in real time.

Top-Down Approach

- ***Subteam Approach***
  - The first step is to divide the planning team into as many subteams as there are activities at Level 1 of the WBS. Then follow these steps:
  - The planning team agrees on the approach to building the first level of the Work Breakdown Structure.
  - The planning team creates the Level 1 activities.
  - A subject matter expert leads the team in further decomposition of the WBS for his or her area of expertise.
  - The team suggests decomposition ideas for the expert until each activity within the Level 1 activities meets the WBS completion criteria.

Bottom-Up Approach

- This approach is more like a brainstorming session than an organized approach to building the WBS.
- The bottom-up procedure:
  - The planning team agrees to the first-level breakdown.
  - The planning team is then divided into as many groups as there are first-level activities.
  - Each group then makes a list of the activities that must be completed in order to complete the first-level activity. Someone in the group identifies an activity and tells it to the group. The process repeats itself until no new ideas are forthcoming. The group then sorts activities that are related to one another.
  - Each group then reports to the entire planning team the results of its work final critiques are given, missing activities added, redundant activities removed.

Six Criteria to Test for Completeness in the WBS

- The WBS is developed as part of JPP session. But how do you know that you've done this right? Each activity must possess six characteristics to be considered complete-that is, completely decomposed. The six characteristics are
  - Status/ completion is measurable
  - Start/ end events are clearly defined
  - Activity has a deliverable
  - Time/ cost is easily estimated
  - Activity duration is within acceptable limits
  - Work assignments are independent

Let us review each one in detail …

Six Criteria to Test for Completeness in the WBS

- **Measurable Status :** The project manager can ask for the status of an activity at any point in time during the project. If the activity has been defined properly, that question is answered easily.
  - Example: a system's documentation is estimated to be about 300 pages long and requires approximately four months of full-time work to write, here are some possible reports that project manager can provide regard the status:
    - I've written 150 pages, so I guess I am 50 percent complete.

Six Criteria to Test for Completeness in the WBS

- **Bounded :**
  - Each activity should have a clearly defined start and end event.
  - Once the start event has occurred, work can begin on the activity.
  - The deliverable is most likely the end event that signals work is closed on the activity.
  - For example, using the systems documentation example, the start event might be notification to the team member who will manage the creation of the systems documentation that the final acceptance tests of the systems are complete. The end event would be notification to the project manager that the customer has approved the systems documentation.

Six Criteria to Test for Completeness in the WBS

- **Deliverable**
  - The result of completing the work that makes up the activity is the production of a deliverable.
  - The deliverable is a visible sign that the activity is complete.
  - This could be an approving manager's signature, a physical product or document.

Six Criteria to Test for Completeness in the WBS

- **Cost/Time Estimate**
  - Each activity should have an estimated time and cost of completion.
  - Being able to do this at the lowest level of decomposition in the WBS allows you to aggregate to higher levels and the total project cost and the completion date.

Six Criteria to Test for Completeness in the WBS

- **Activity Independence**
  - It is more important that each activity be independent. Once work has begun on the activity, it can continue reasonably without interruption and without the need of additional input or information until the activity is complete.
  - Though it is possible that an activity could be scheduled during different times based on resource availability.

Approaches to Building the WBS

- There are many ways to build the WBS. There is no one correct way to create the WBS. Hypothetically, if we put each member of the JPP session in a different room and asked that person to develop the project WBS, they might all come with different answers.
- There are three general approaches to building the WBS:

Approaches to Building the WBS

- **Noun-type approaches**. Noun-type approaches define the deliverable of the project work in terms of the components (***physical*** or ***functional***) that make up the deliverable.
- **Verb-type approaches.** Verb-type approaches define the deliverable of the project work in terms of the actions that must be done to produce the deliverable. These include the ***design-build-test-implement*** and project ***objectives*** approaches.
- **Organizational approaches**. Organizational approaches define the deliverable of the project work in terms of the organizational units that will work on the project. This type of approach includes the department, process, and geographic location approaches.

Representing the WBS

- There are two ways to represent the WBS:
  - Outline Format
  - Hierarchical Format

- Both convey the same information

WBS for House: Outline Format

1 SITE PREPARATION
- 1.1 Layout
- 1.2 Grading
- 1.3 Excavation

2 FOUNDATION
- 2.1. Erect Forms
- 2.2. Pour Concrete
- 2.3. Remove Forms

3 FRAMING
- 3.1.Floor Joists
  - 3.1.1. Install first-floor joists
  - 3.1.2. Install second-floor joists
- 3.2. Subflooring
  - 3.2.1. Install first-floor subflooring
  - 3.2.2. Install second-floor subflooring
- 3.3. Stud Walls
  - 3.3.1. Erect first-floor stud walls
  - 3.3.2. Erect second-floor studwalls
- 3.4. Frame the roof

4 UTILITIES
- 4.1. Electrical
  - 4.1.1. Do rough-in Work
  - 4.1.2. Get Building Inspection
  - 4.1.3. Do Finish Work
- 4.2. Gas
  - 4.2.1. Do rough-in Work
  - 4.2.2. Get Building Inspection
  - 4.2.3. Do Finish Work
- 4.3. Water
  - 4.3.1. Do rough-in Work
  - 4.3.2. Get Building Inspection
  - 4.3.3. Do Finish Work

5 WALLS
- 5.1. Hang Sheetrock
- 5.2. Tape and Bed

6 ROOFING
- 6.1. Install Sheathing
- 6.2. Lay Shingles

7 FINISH WORK
- 7.1. Install Cabinets
- 7.2. Install Appliances
- 7.3. Install Furnace
- 7.4. Lay Carpet
- 7.5. Paint Walls and Molding
- 7.6. Hang Wallpaper
- 7.7. Lay Tile

8 LANDSCAPING

WBS for Waterfall systems Development Methodology