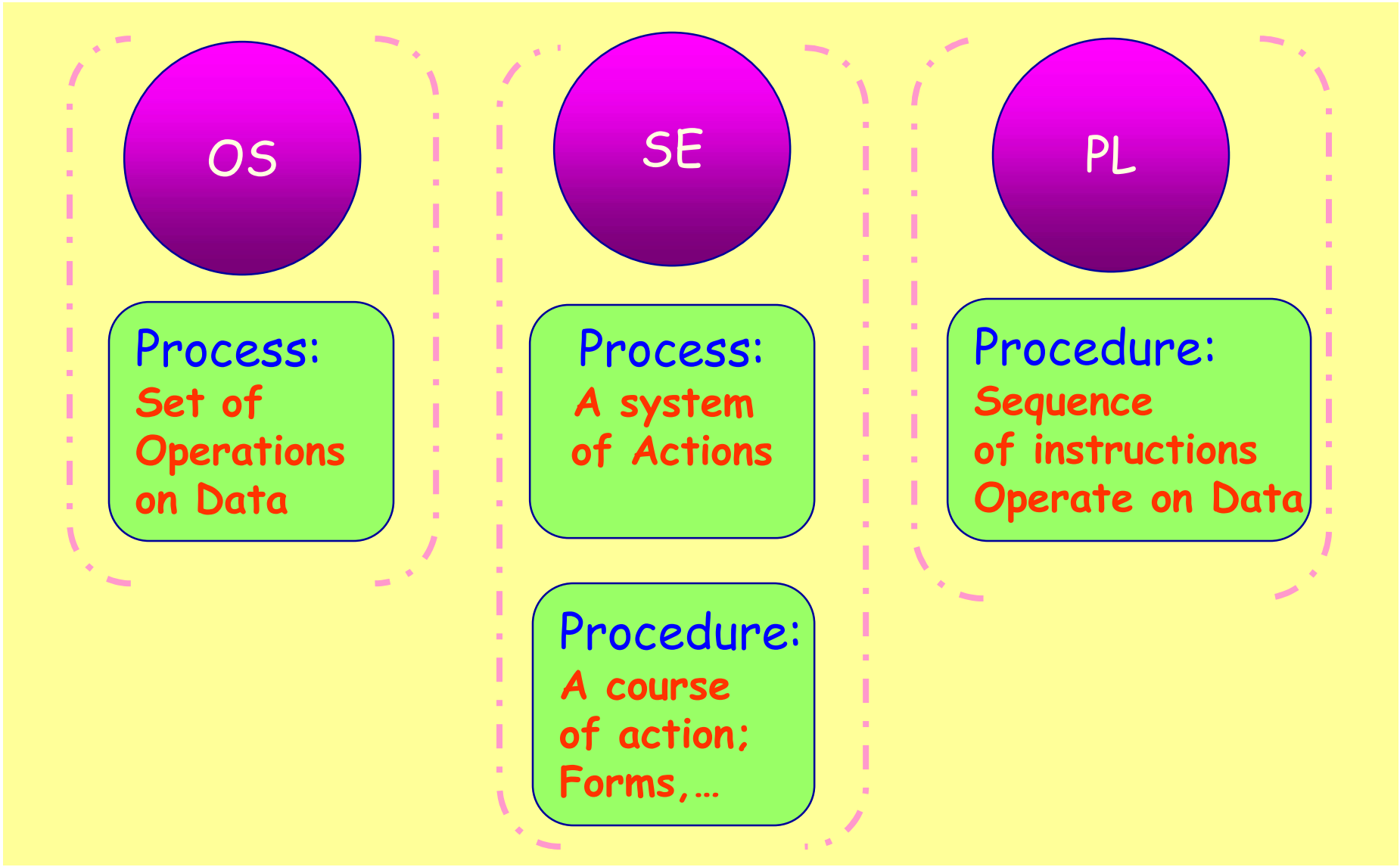
Mapping Process Activities into Project Tasks

How to represent the Software Development Process Activities
By Tasks in the Project Plan?

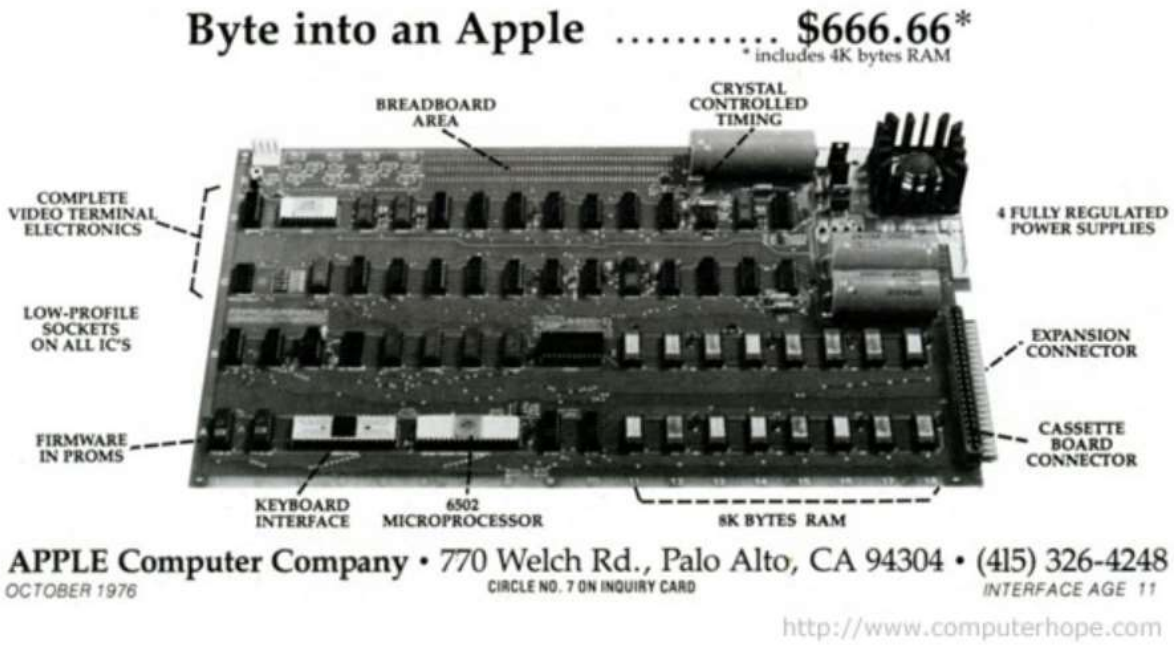
Process vs. Procedure



Software/Hardware Evolutionary Process

The **Apple I** (**Apple 1**) was the first Apple computer that originally sold for $666.66. The computer kit was developed by Steve Wozniak in 1976 and contained a 6502 8-bit processor and 4 kb of memory, which was expandable to 8 or 48 kb using expansion cards. Although the Apple I had a fully assembled circuit board the [illegible] power supply, display, keyboard, and case to be operational. Below is an Apple I from an advertisement by Apple.

4 kb of memory



http://www.computerhope.com

Software/Hardware Evolutionary Process

MacBook

from $1299

- 12-inch (diagonal) LED-backlit Retina display
- 1.1GHz, 1.2GHz, or 1.3GHz dual-core Intel Core M processor
  Turbo Boost up to 2.9GHz
- Up to 9 hours battery life[1]
- Up to 512GB flash storage[2]
- 2.03 pounds[3]
- Available in gold, silver, and space gray

MacBook Air 11-inch

from $899

- 11.6-inch (diagonal) LED-backlit display
- 1.6GHz dual-core Intel Core i5 or 2.2GHz dual-core Intel Core i7 processor
  Turbo Boost up to 3.2GHz
- Up to 9 hours battery life[1]
- Up to 512GB flash storage[2]
- 2.38 pounds[3]

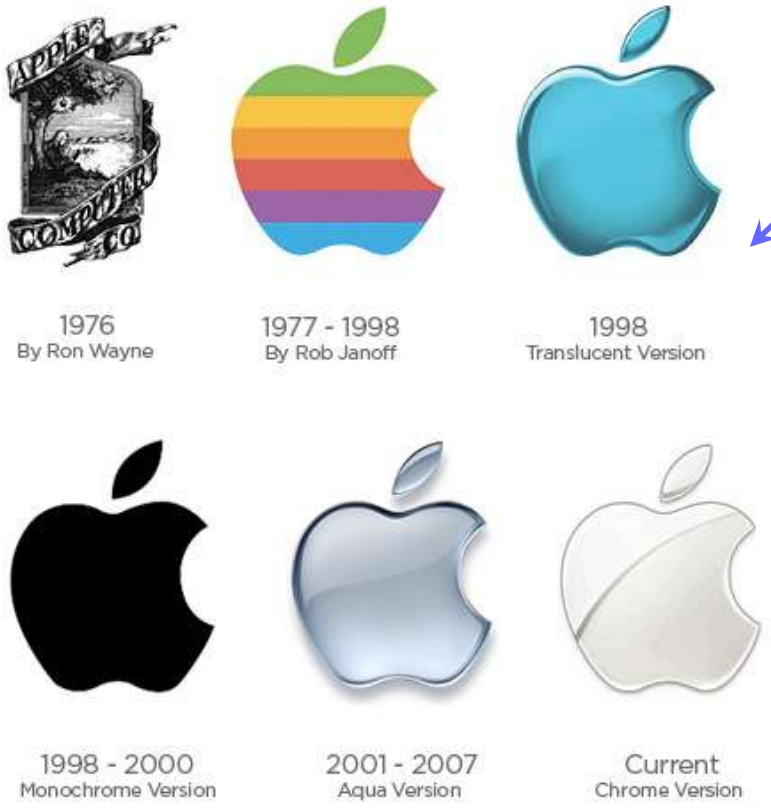
MacBook Air 13-inch

from $999

- 13.3-in displa
- 1.6GH 2.2GH proce
  Turbo Boost up to 3.2GHz
- Up to 12 hours battery life[1]
- Up to 512GB flash storage[2]
- 2.96 pounds[3]

512 GB of memory

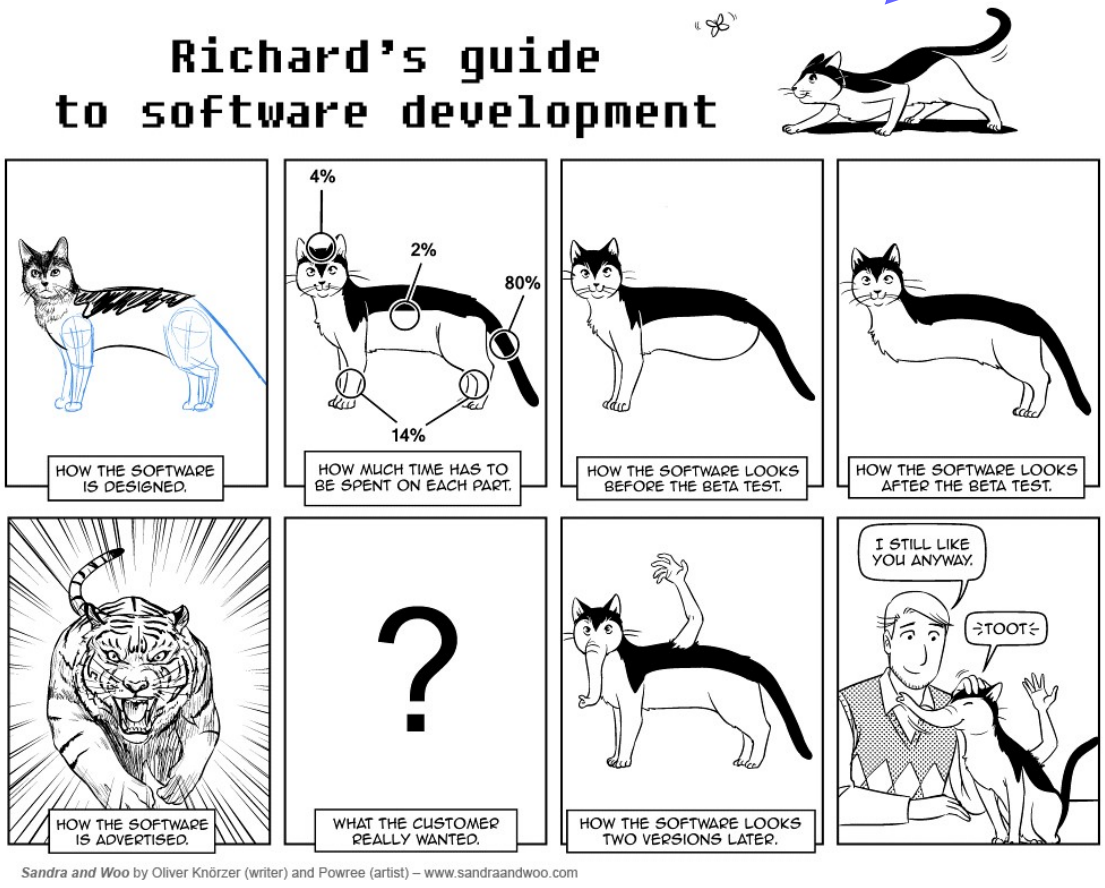
Software/Hardware Evolutionary Process



Even the logo of the company has its own evolutionary process

Software/Hardware Evolutionary Process

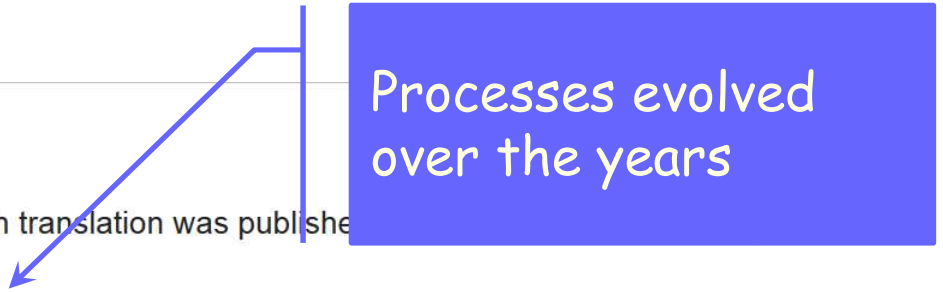
Software Process aligns perceptions with expectations



Software/Hardware Evolutionary Process

Project evolves over multiple phases

Software/Hardware Evolutionary Process



1970s

- Structured programming since 1969
- Cap Gemini SDM, originally from PANDATA, the first English translation was publishe
  Development Methodology

1980s

- Structured systems analysis and design method (SSADM) from 1980 onwards
- Information Requirement Analysis/Soft systems methodology

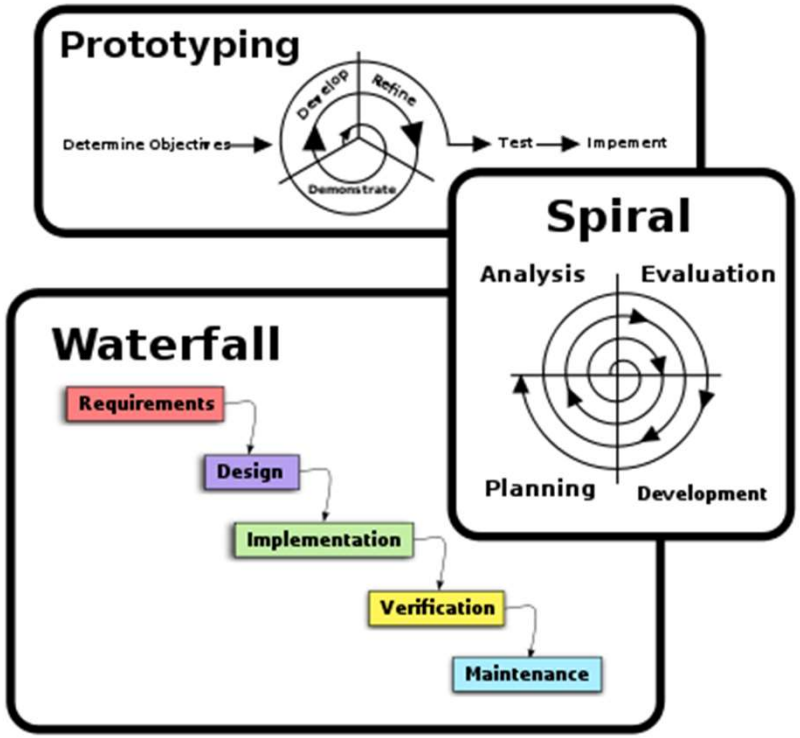
1990s

- Object-oriented programming (OOP) developed in the early 1960s, and became a dominant programming approach during the mid-1990s
- Rapid application development (RAD), since 1991
- Dynamic systems development method (DSDM), since 1994
- Scrum, since 1995
- Team software process, since 1998
- Rational Unified Process (RUP), maintained by IBM since 1998
- Extreme programming, since 1999

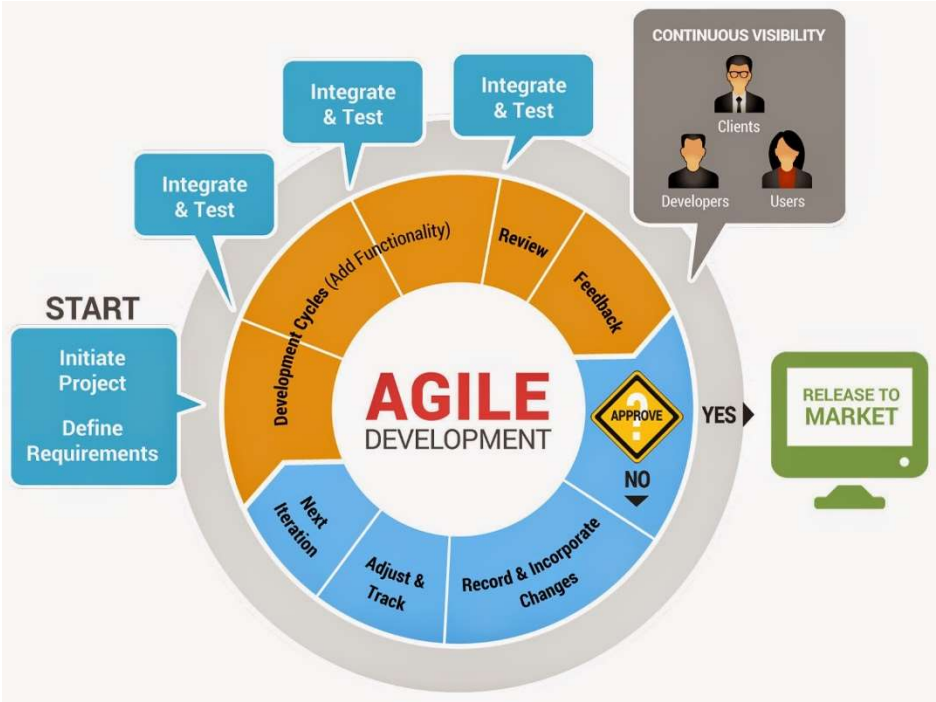
2000s

- Agile Unified Process (AUP) maintained since 2005 by Scott Ambler
- Disciplined agile delivery (DAD) Superseded of AUP

Software/Hardware Evolutionary Process



Processes evolved over the years

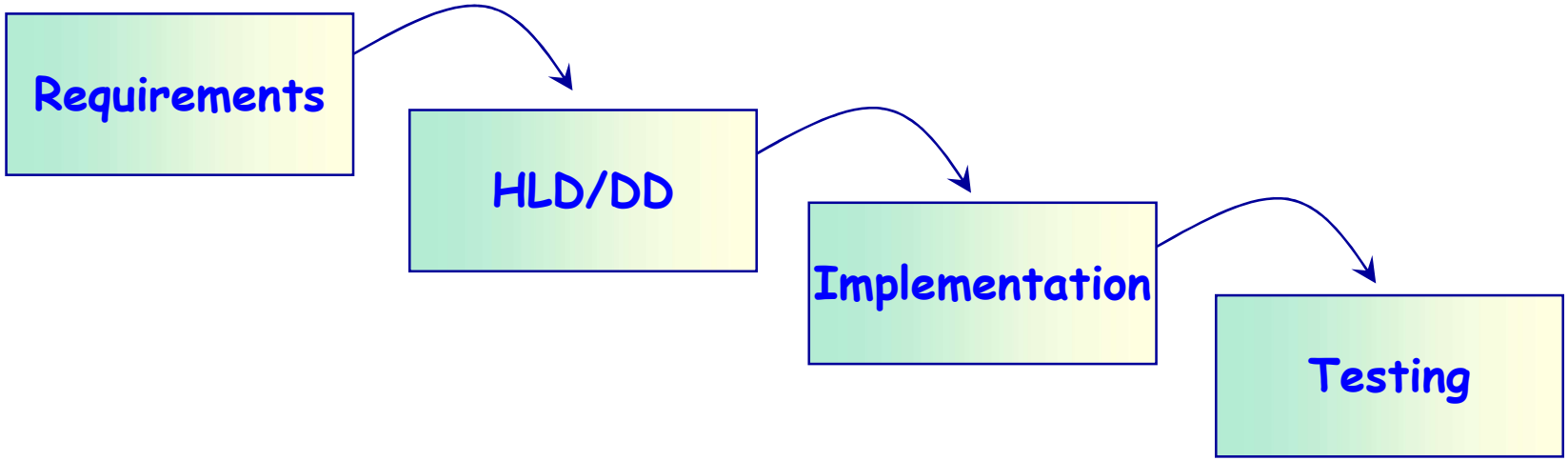


The Software Process

- A documentation for a sequence of steps that shall be followed when :
  - Starting/Closing the project
  - Writing/review requirement document
  - Writing/review High-level/detailed design document
  - Implementation and Code inspection
  - Writing test plan and Execute test cases

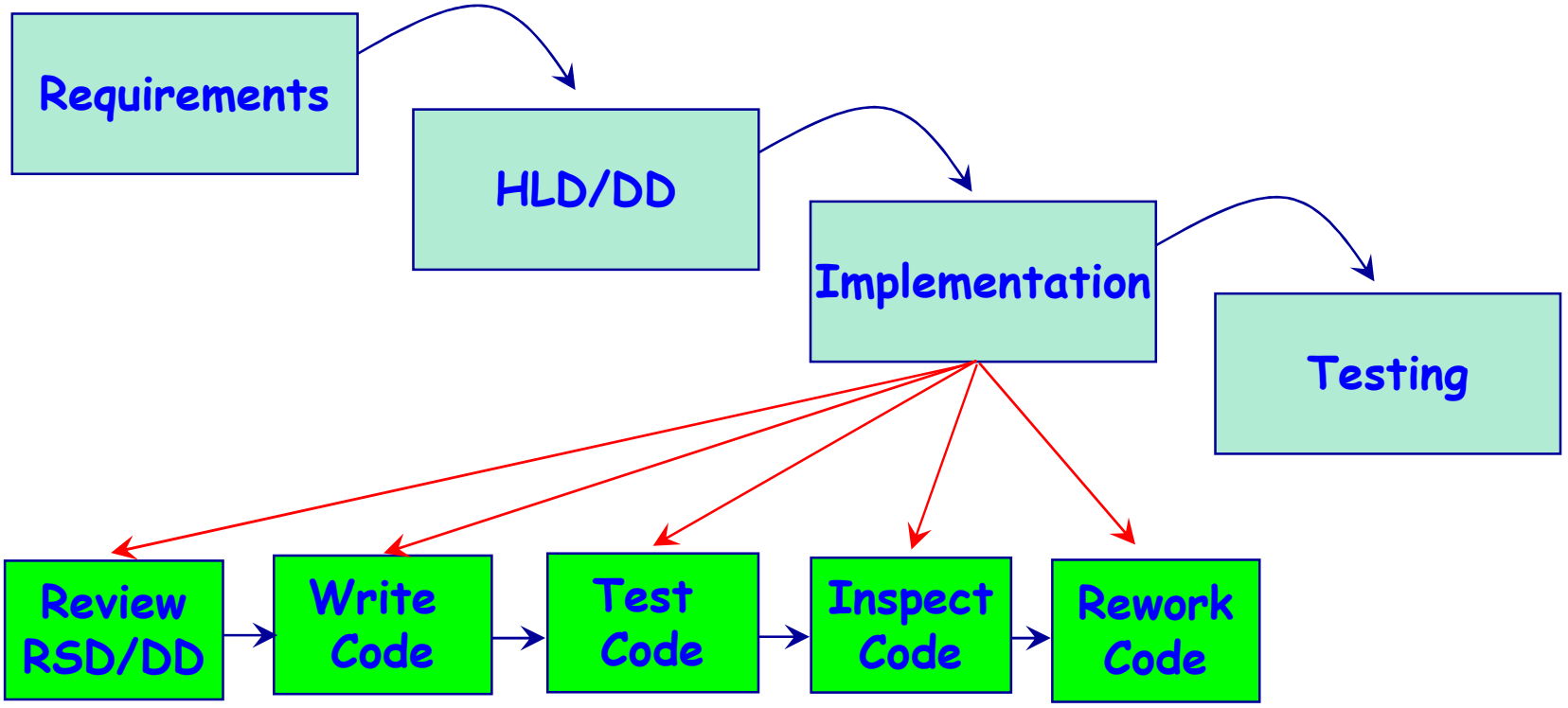
The Detailed Process

- For each phase in the software development, we need to have a detailed process that has been tailored mainly to that phase



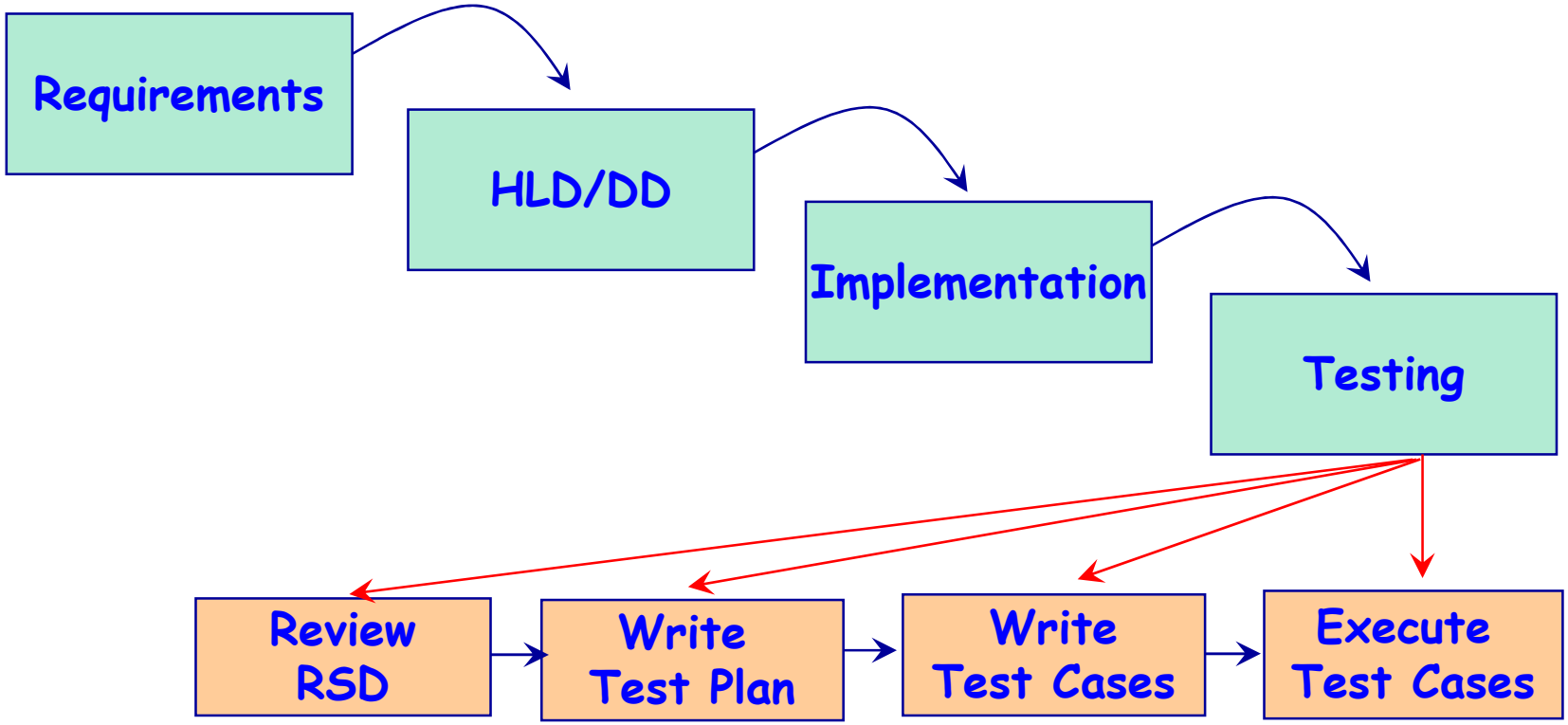
The Implementation Phase Process

- Here is the tailored process for the implementation phase:



The Testing Phase Process

- Here is the tailored process for the testing phase:



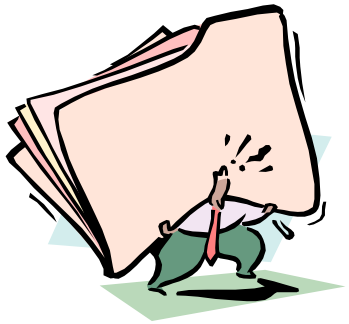
Tailoring the Process

How to tailor a process?

- First, each process shall have an owner.
- Adopt a standard process or borrow an existing one
- Add/Delete tasks or artifacts as you see appropriate but remember there are always risks for deleting an artifact and there is an overhead for adding one.
  - For example, you may not have formal requirement review or a test plan document

The Risk to Tailor a Process

- A process shall be documented, however to tailor a process may entail some risks; therefore the software organization shall provide general guidelines for process changes:



  - What are the artifacts or tasks
  - Who relies on these artifacts
  - What are the risks of deleting or modifying a task or an artifact
  - Guidelines for tailoring the process to certain domains
  - Document the rational behind the changes

Software Development Process Artifacts

- What is the artifact?

- The software artifact is an item produced as a result of the process execution
- The artifact could be a document, code, test results, …
- Examples:
  - Source Code and executable image
  - Test Results
  - RSD
  - HLD, class-diagram, sequence-diagram
  - Test Plan

Documentation Plan

- **Documentation plan is an essential part of the SW Development Plan and it provides answers to the following questions:**

- What are the artifacts to be produced?
- Who produces these artifacts?
- Who uses these artifacts?
- What is the format of the artifact?
- Who is the owner of the artifact?

Documentation Release

- Like software, documentation shall be kept under configuration management.

- Ideally, the documentation and the software releases shall be kept synchronized

Artifact dependencies and Ownership

Artifact	Produced By	Used By	Owner
Requirement Document	Customer	System Engineer Developer Tester	Customer
HLD/DD Document	System Engineer	Developer	System Engineer Process owner
Source Code	Developer	Developer Tester	Developer Process owner
Testing Plan	Tester	Tester	Tester Process Owner
⋮	⋮	⋮	⋮

Common Artifacts in the SW Development

Artifact	Description
Statement of Work	•The requirements for the project and the process •What is the outcome of the project
Requirement Specifications	•Technical specifications (functions, performance, standards met, etc.) of the final product
High-Level and Detailed Design Specification	•Modules and components of the system and how they interact •What are the software development files that will be implemented
Implementation	•What are the files that will be produced •Procedure for the build •End User Documentation •Maintenance Documentation •Installation Guide
Testing	•Test Plan •Test Cases •Test Results

How to honor the Software Process?

- **The process has to be practiced in order to ensure the quality control and quality assurance for the software system.**

- Reviews and Audits are the means by which we ensure the integrity and health of the process.
- Reviews are conducted by peers and managers
- Audits are conducted by independent organization

Reviews and Audits

Why we need reviews?

- **We can check a program for quality by testing, …**

- But:
  - How do we check that the test plan being used for testing has the right test cases?
  - How we check the requirement specification documents for defects?
  - How we check the design document for design errors?

Reviews

- **Reviews are the most effective method to improve quality by identifying defects.**

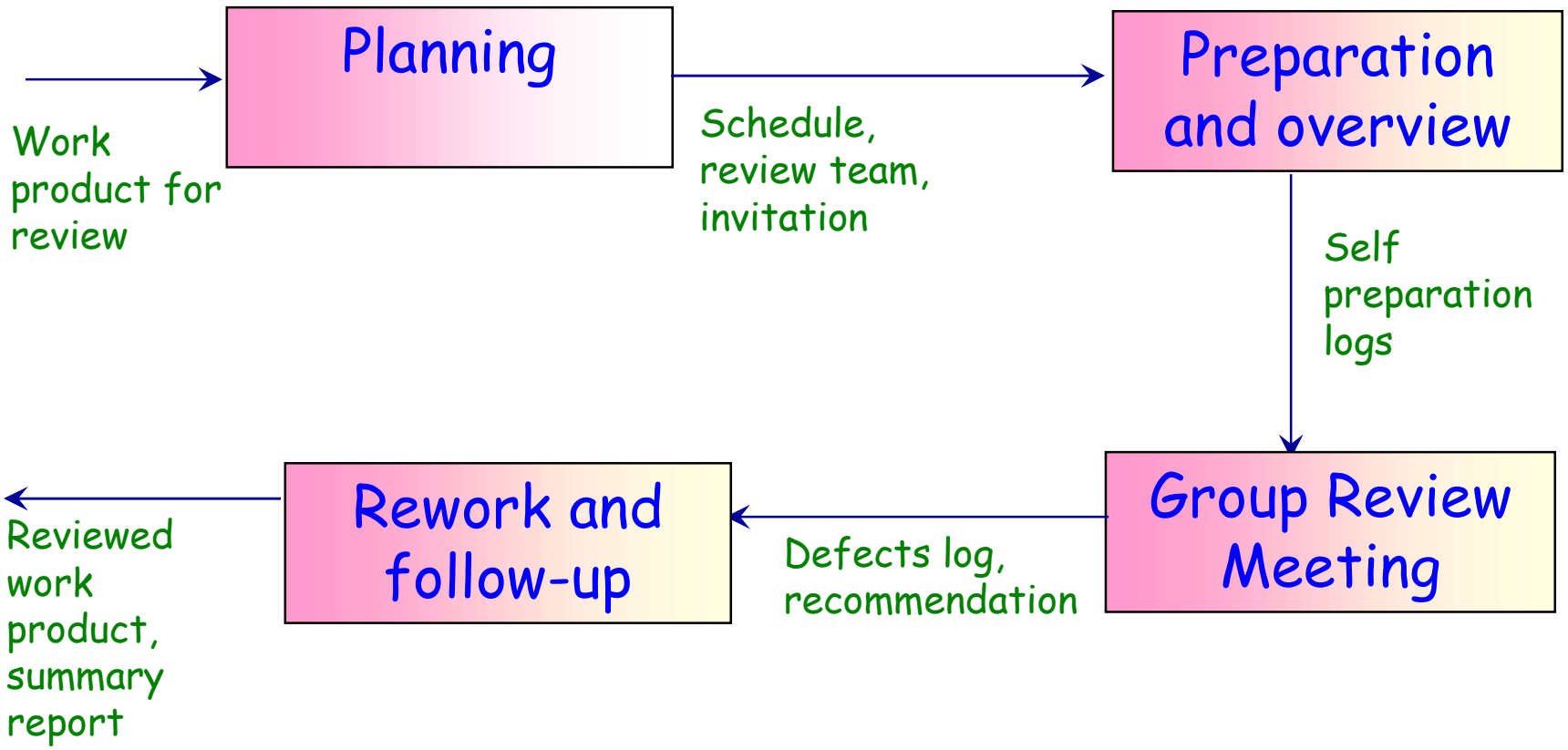
- Applied in
  - Design document
  - Test plan
  - Code "inspection"

- Can be used to
  - Track progress
  - Prevent defects discovered by Customer
  - Improve productivity by finding defects in effective ways

Reviews

- **Reviews can come in different forms:**
  - Formal Group Review or inspection
  - Desk Review, one person involved in the review
- **Reviews are conducted by**
  - Technical people for technical people
- **Reviews intent**
  - To identify problems NOT to resolve them

The Review Process



The Planning Phase of the Review Process

- **Objective:**
  - Prepare for the review by selecting the group review team and schedule the review
- **Participants:**
  - Author
  - Project Manager
  - Moderator
  - Reviewers

The Overview and Preparation Phase

- **Objective:**
  - Deliver and explain the work package to the reviewers
- **Work package includes:**
  - **Work product**
  - **Specifications**
  - **Checklists**
  - **Standards**
- **Outcome:**
  - The self-review forms are completed; defects, actual time spent are recorded

The Group Review Meeting Phase

- **Objective:**
  - Come up with a final defect list that agreed upon by the whole group
- **Logistics**
  - **Moderator is controlling the Review Report**
  - **Moderator logs actual time spent for preparation**
  - **Start/End time of the session**
  - **Concerns/issues from team members**
- **Two Roles have to be filled from the reviewers:**
  - Scribe
  - Reader

The Rework and Follow-Up Phase

- **Objective:**
  - The author performs the rework to fix the defects raised during group review meeting
- **Moderator**
  - Has to decide on fix reworks
  - Has to decide whether a re-review meeting is required to go over the fixes.
  - Has to ensure that all data and review results are recorded and must be submitted along with group review summary to the project leader

Guidelines for Reviews in Projects

- Not all products shall go under a formal group review because :

  - Reviews are expensive
  - Reviews are overhead activities

Review Types

- **What are the types of the reviews?**

1. Technical
   - Peer Reviews
2. Managerial
   - Status Reviews
   - Quality Gates Reviews

Guidelines for RSD Reviews

Work Product	*Focus*	*Entry Criteria*	*Participants*
Requirement Specification Document	• Requirements meet customer needs • Requirements are implementable • Omissions, inconsistencies, and ambiguities in the requirements	• The Document conforms to the standards	• Customer • Designers • Testers • Installation team members User documentation author

Guidelines for High-Level Design Reviews

Work Product	*Focus*	*Entry Criteria*	*Participants*
High-Level Design	• High-level design implements the requirements • The design is implementable • Omissions and other defects in the design	• The document conforms to standards • The requirements have been reviewed and finalized	• Requirements Author • Detailed Design author • Developer

Guidelines for Code Reviews

Work Product	*Focus*	*Entry Criteria*	*Participants*
Code	• Code implements the design • Code is complete and correct • Defects in code	• The code compiles and passes style and other norms	• Designer • Tester • Developer

Guidelines for Test Cases Reviews

Work Product	*Focus*	*Entry Criteria*	*Participants*
System Test Cases	• The set of test cases checks all conditions in the requirements • System test cases are correct • Test cases are executable	• Requirements have been base lined • System test plans is consistent with the standards	• Requirements author • Tester • Project Leader

Guidelines for High-Level Design Reviews

Work Product	*Focus*	*Entry Criteria*	*Participants*
Project Management Plan	**PProject management plan meets project management and control needs** **• Completeness** **• Project management plan is implementable** **• Omissions and ambiguities**	**• The project management plan follows the standard template**	**• Project leader** **• Another Project Leader**

Data Collection During Reviews

- **Reviews are mainly human processes, therefore:**
  - We need to record the data
- **What to record ?**
  - Effort Data
  - Defect Data
- **Why to record ?**
  - Analyze the effectiveness of the reviews
  - Construct the Review Capability Baseline

Data Collection During Reviews

- **Reviews are mainly human processes, therefore:**
  - We need to record the data
- **What to record ?**
  - Effort Data
  - Defect Data
- **Why to record ?**
  - Analyze the effectiveness of the reviews
  - Construct the Review Capability Baseline

Data Collection Forms in Technical Reviews

- Forms used for data collection during technical reviews:

  - Self-Preparation log
  - Group Review Meeting Log
  - Group Review Summary report

Self-Preparation Log

- **Objective :**
  - **Record all defects**
  - **Record Effort Spent**
- **The Form:**

Project Code:
Work Product ID:
Reviewer Name
Effort Spent for Preparations (hours):
Issue list:

SI	Location	Description	Criticality

Group Review Meeting Log

- **Objective :**
  - **Record defects agreed upon by the team**
  - **Record Effort Spent by the team**
- **The Form:**

Project Code:
Moderator:
Author:
Date:
Effort Spent on review meeting (hours):
Defects to be closed by (date):
Defect List:

Meeting Type:
Scribe:
Reviewers:
Observers:
Work Product ID:

SI	Location	Description	Reviewer	Criticality

Group Review Summary Report

- Objective :
  - To analyze the effectiveness of the review
- The Form:

Group Review Summary Report

- The Form:

Project Work Product Type Size of Product Moderator Reviewers Author	
Effort (Person-Hours) Overview meeting Preparation Group review meeting	
Defects Number of critical defects Number of major defects Number of minor defects Number of defects found during preparation Number of defects found during group review meeting	
Result	***Moderator reexamination***
Recommendations for next phase	
Comments (Moderator)	
Prepared by:	***Date:***

The Review Capability Baseline

- The effectiveness of the review process depends on how well the process has been executed
- How does the project manager or the moderator evaluate whether a review has been effective?

- The statistical process control (SPC) can be used to monitor and control the reviews

The Review Capability Baseline

- How can the SPC be applied to monitor the reviews?
  - Project managers must
    1. Identify Performance parameters
    2. Control limits for performance parameters
    3. Monitor actual performance
    4. Determine effectiveness through
       a) Control charts (plot performance parameters)
       b) Control limits (see whether parameters within the ranges); EASY to APPLY

The Review Capability Baseline

- Examples of Performance Parameters
  - Coverage rate during preparation
  - Coverage rate during group review meeting
  - Defect density for minor defects
  - Defect density for major defects
- Control limits are determined from past data and group review capability baseline

The Review Capability Baseline

Review Item	Preparation Coverage Rate	Group Review Coverage Rate	Minor Defect Density	Major Defect Density
Requirements	5-7 pages/hour	0.5-1.5 defects/page	0.1-0.3 defects/page	
High-Level design	4-5 pages/hour (200-250 specification statements/hour)	0.5-1.5 defects/page	0.2-0.6 defects/page	
Detailed design	3-4 pages/hour (70-100 specification statements/hour)	0.5-1.5 defects/page	0.2-0.6 defects/page	
Code	160-200 LOC/hour	110-150 LOC/hour	0.01-0.06 defects/LOC	0.01-0.06 defects/hour
Integration test plan	5-7 pages/hour	0.5-1.5 defects/page	0.1-0.3 defects/page	
Integration test cases	3-4 pages/hour			
System test plan	5-7 pages/hour	0.5-1.5 defects/page	0.1-0.3 defects/page	
System test cases	3-4 pages/hours			
Project management and configuration management plan	4-6 pages/hour	2-4 pages/hour	0.6-1.8 defects/page	0.1-0.3 defects/page

Charting the Reviews

- Monitoring the reviews using the SPC Charting tool:
  - The SPC tool is a spreadsheet that has the review capability baseline built into it
  - If process changes, then we need to change the control limits that affect the control charts
  - Change the control limits based on the past 10-15 performance data points

Analysis Guidelines for the Reviews

- If the number of the defects found during the review is within the range given in the baseline, the review is considered effective, and the exit criteria is satisfied, ELSE :
  - The moderator or the project leader needs to determine the cause and take preventive and corrective actions
  - There are two sets of guidelines, one if the defect density below the limit and another one when defect density above the limit

Analysis Guidelines for the Reviews

- **If defects found are less than norms:**

Possible Reason	Actions to Consider
Work product was very simple	•Convert group review of similar work product to one person review (desk review) •Combine reviews
Reviews may not be thorough	Check coverage rate; if too low, reschedule a review, perhaps with a different team
Reviewers do not have sufficient training on group reviews or experience with the reviewed material	•Schedule or conduct group review training •Re-review with a different team
Work product of very good quality	•Confirm this fact by coverage rate, experience of the author, reviewers, and so on; see if this quality can be duplicated in other parts of the project. •Revise defect prediction in downstream activities; see if there are general process improvement lessons

Analysis Guidelines for the Reviews

- **If defects found are more than norms:**

Possible Reason	Actions to Consider
Work product is of low quality	•Examine training needs for author •Have the work product redone •Consider reassigning future tasks (easier tasks to the author)
Work product is very complex	•Ensure good review or testing downstream •Increase estimates for system testing •Break the work product into smaller components
There are too many minor defects and two few major defects	•Identify causes of minor defects; correct in the future by suitably enhancing checklists and making authors aware of the common causes •Reviewer may have insufficient understanding of the work product. If so, hold an overview meeting or have another review with different reviewers
Reference document against which review was done is not precise and clear	•Get the reference document reviewed and approved
Reviewed modules are the first ones in the project	•Analyze the defects, update the review checklist, and inform developers. Schedule trainging.

The NIH and NAH Software Syndromes

- In the old days, the rule for software tool usage
  - NIH : Not Invented Here
- In the current days, the rule for software process and review practices
  - NAH : Not Applicable Here

Status Reviews

- This review focuses on the project progress
- Planned vs. actual progress
- Status on issues raised in previous status review meeting
- Identify and address concerns that may have negative impact on the project
    1. Schedule
    2. Cost
    3. Quality

Conducting the Audit

- What to audit and why?
  - The auditors focus on whether the defined process is being followed in the project
  - It discovers problems but doesn't solve them
- Who execute it?
  - EXPENSIVE Independent Auditors

Conducting the Audit

- **When to audit?**
  - Regularly
  - Audits are sign of healthy projects and organizations
  - Audits shall be thought of as preventive measures rather than reactive measures

Conducting the Audit

- The audit methodology:
  - Verify stakeholders are practicing their processes
  - Look how an activity is done
  - Verify the output (artifacts) of the activity
  - Number of questions are asked from the audit checklist

The Audit Checklist

1. Is the project plan documented in the standard project plan template?
2. Has the project plan been group reviewed?
3. Has the project plan been approved and baselined, and is it under configuration management?
4. Is there a signed contract?
5. Have commitments to customer or other group been approved?
6. Is there an estimated effort for the project that is based on historical data?
7. Have the effort estimates and the schedule been reviewed?

The Audit Checklist

8. Is the quality plan complete, and has it been reviewed?
9. Is the life cycle in the project identified and documented?
10. Are personnel identified and the responsibility for each work element defined and tracked?
11. Are deliverables to the customer, including the documentation, clearly identified?
12. Are risks and risks mitigation plans identified and properly documented?
13. Are reviews, progress reporting, tracking, and approval mechanisms identified?

The Audit Follow-Up

- The Audit intent is to Audit the applied process rather auditing the people
- If evidence suggests that the approved processes are not followed, a noncompliance report(NCR) will be issued
- People involved with this process shall not be punished because of the NCR report, rather it shall be an opportunity to take corrective actions.
- Identifying the noncompliance is the goal of these audits