Review

Key topics

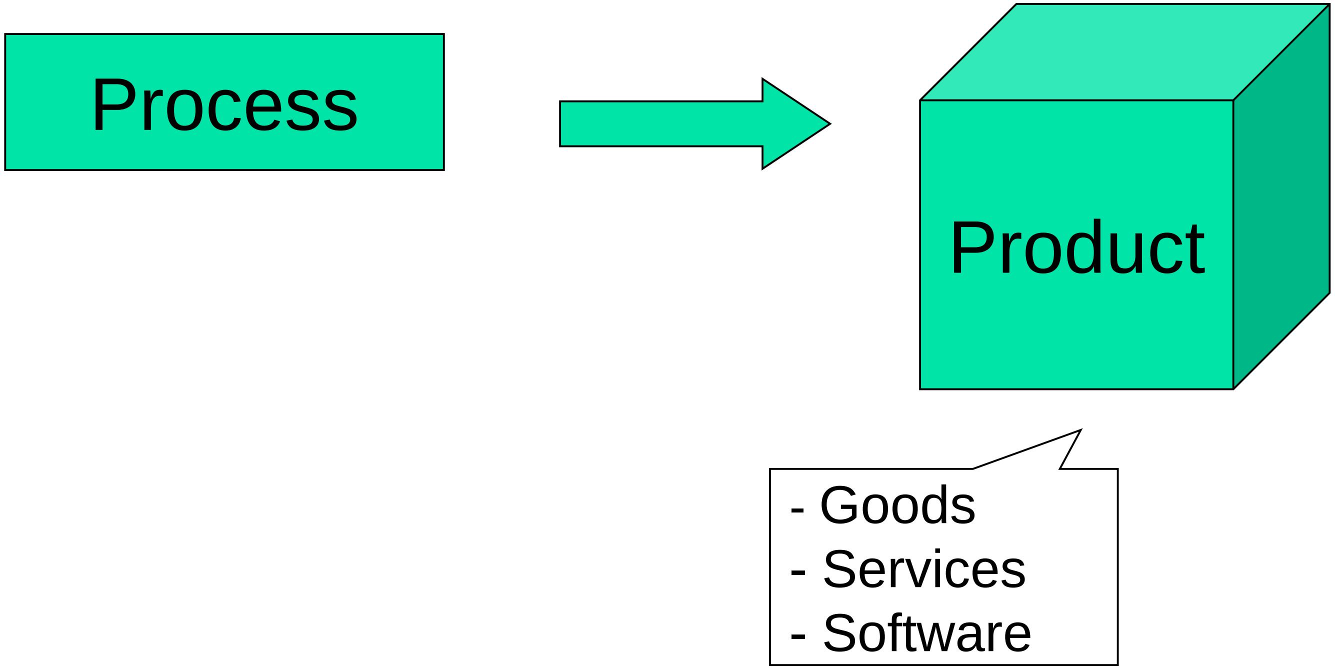
Quality helps bring success

- **Goal**: survival of the software vendor
  - *Revenue: customer satisfaction and loyalty*
  - *Cost: effective and efficient development*
- **QA Message**
  - Good processes yield good products

- What makes development successful?
  - Product delivered on time
  - Development costs within budget
  - Product meets requirements and fit for use

Quality and Process

- Quality is about doing things better, with less cost and more enjoyment
- It provides business opportunities and is essential to developer competition and survival



Quality assurance and control

- Quality Assurance → **PROCESS**
  - Discipline to maximize chances software products conform to requirements, as perceived by the users
    - Defining and documenting procedures and standards
    - Ensuring they are followed
    - Monitoring success and seeking to improve
- Quality Control → **PRODUCT**
  - Build quality into the software throughout the software development life cycle
    - Reviewing documents and testing code
    - Managing defects
    - Gathering metrics

Assessing maturity with CMMI

- CMMI is a process improvement approach to:
  - Guide process improvement across a project, a division, or an entire organization
  - Help integrate traditionally separate organizational functions
  - Set process improvement goals and priorities
  - Provide guidance for quality processes
  - Provide a point of reference for appraising current processes
- Must to be implemented at corporate level
  - Management initiative, from the top down

ISO 9001 certification

- Seven steps to implement and maintain a quality management system (QMS)
  1. Engage top management
     - Quality policy and organizational objectives come from the top
  2. Identify key processes and interactions to meet quality objectives
  3. Use process management techniques to implement QMS
  4. Perform gap analysis between QMS and ISO 9001 requirements and then add required activities, procedures and controls
  5. Implement the system, train staff, verify effective operation
  6. Manage the QMS
     - Focus on customer satisfaction
     - Monitor operation and strive for continual improvement
  7. Seek third party certification/registration or issue self-declaration of conformity

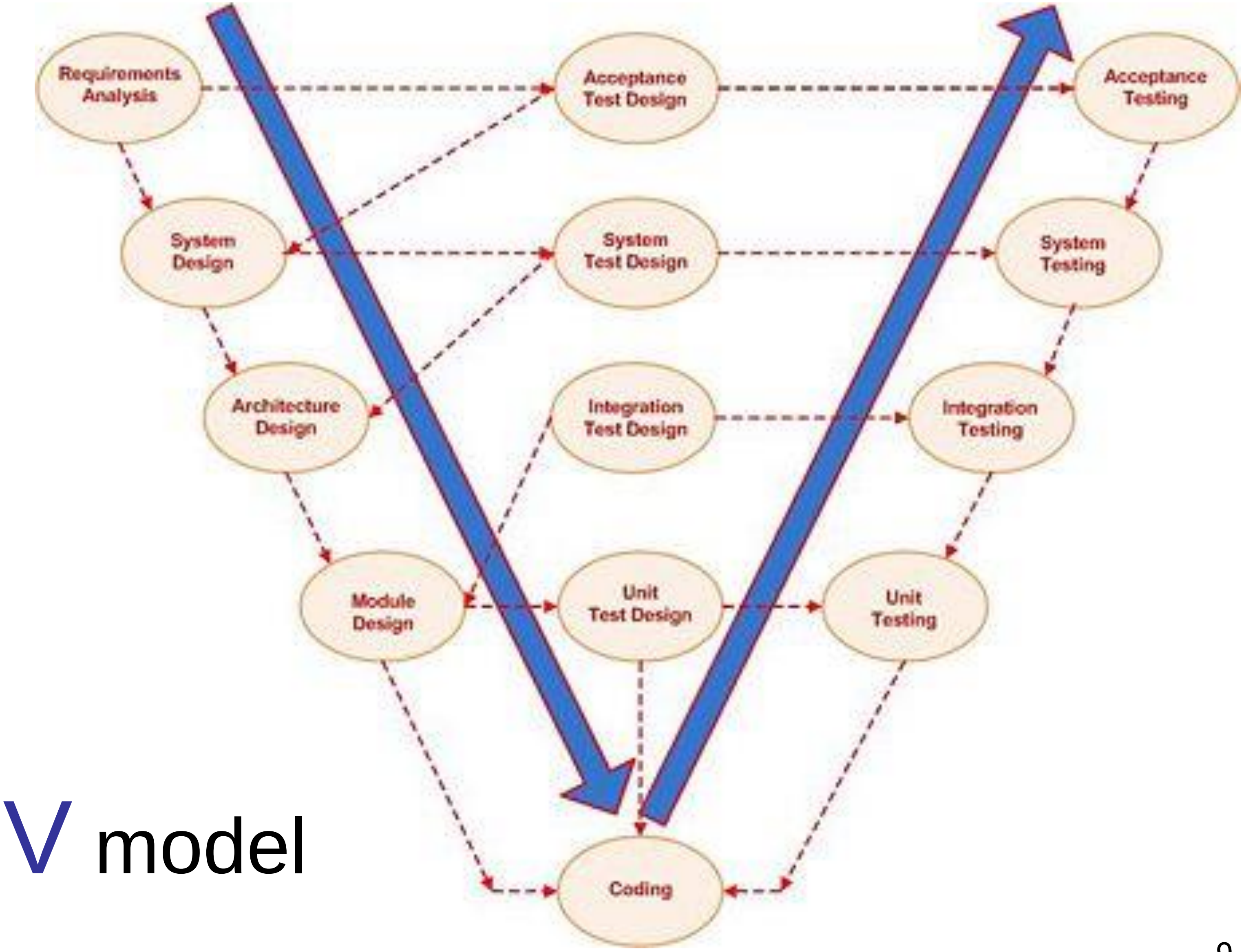
Software Quality Characteristics ISO/IEC 9126:2001

- Functionality
  - Suitability, Accuracy, Interoperability, Compliance, Security
- Reliability
  - Maturity, Recoverability, Fault tolerance
- Usability
  - Learnability, Understandability, Operability
- Efficiency
  - Resource efficiency, time efficiency
- Maintainability
  - Stability, analysability, Changeability, Testability
- Portability
  - Installability, Replaceabiltiy, Adaptability, Conformance

Recall the general testing principles

1. Testing shows the presence of defects
   - *The primary purpose of testing is to find defects*
2. Exhaustive testing is impossible
   - *Except in trivial cases 100% coverage is impossible*
3. Early testing
   - *Start early and focus on defined objectives*
4. Defect clustering
   - *Focus effort where early testing show weakness / manage risk*
5. Pesticide paradox
   - Repeating the same tests tends to finding zero defects
   - *Constantly vary test data, update scripts …*
6. Testing is context dependent
   - *Load test Web, try to break security, test extreme conditions …*
7. Absence of errors fallacy
   - *No defects does not guarantee product is **fit for use***

Introducing early Test



The V model

V-model concepts of testing

- Validation
  - The process of evaluating a system or component during or at the end of the development process to determine whether it satisfies specified requirements

Does the product do the right things?
Fitness for use
What the users want and need

- Verification
  - The process of evaluating a system or component to determine whether the products of a given development phase satisfy the conditions imposed at the start of that phase

Did developers build the product right?
Correctness
Built to design, assuming design fulfils requirements

Recalling the 4 levels of test

- The V-model introduced levels:
  1. Unit / Component
     - Class, package, Web asset, module
  2. Integration
     - Package, module, collections of assets, module
     - Vertical or horizontal slices through application
  3. System
     - Verification on mirror of production environment
  4. Acceptance
     - Validation

ALM

- Modern development replaces **waterfall** with modern methodologies:
  - **Increments** add functionality
    - New use cases
    - Additional capability
    - More slices of the application
  - **Iterations** refine existing code
    - Improve on code from previous or current increment
    - Refactoring
- Each increment may include several iterations

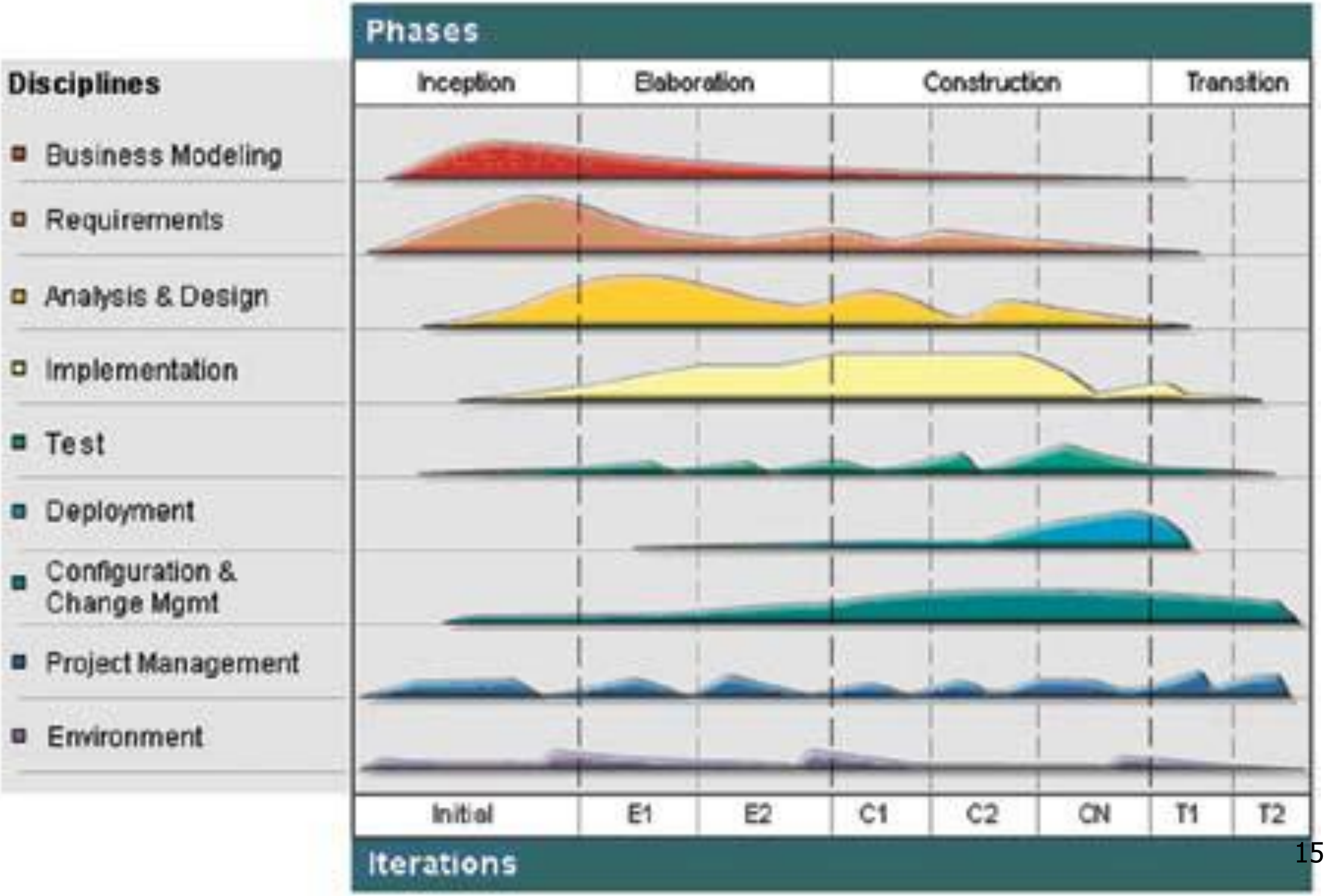
Development methodologies

- Each model has different impact on
  - Project management practices
  - Roles and responsibilities of team members
  - Programming culture
  - Work patterns
  - Amount and formality of documentation
  - Interaction with test process
  - Gathering metrics and reacting to feedback

Three proven Agile systems

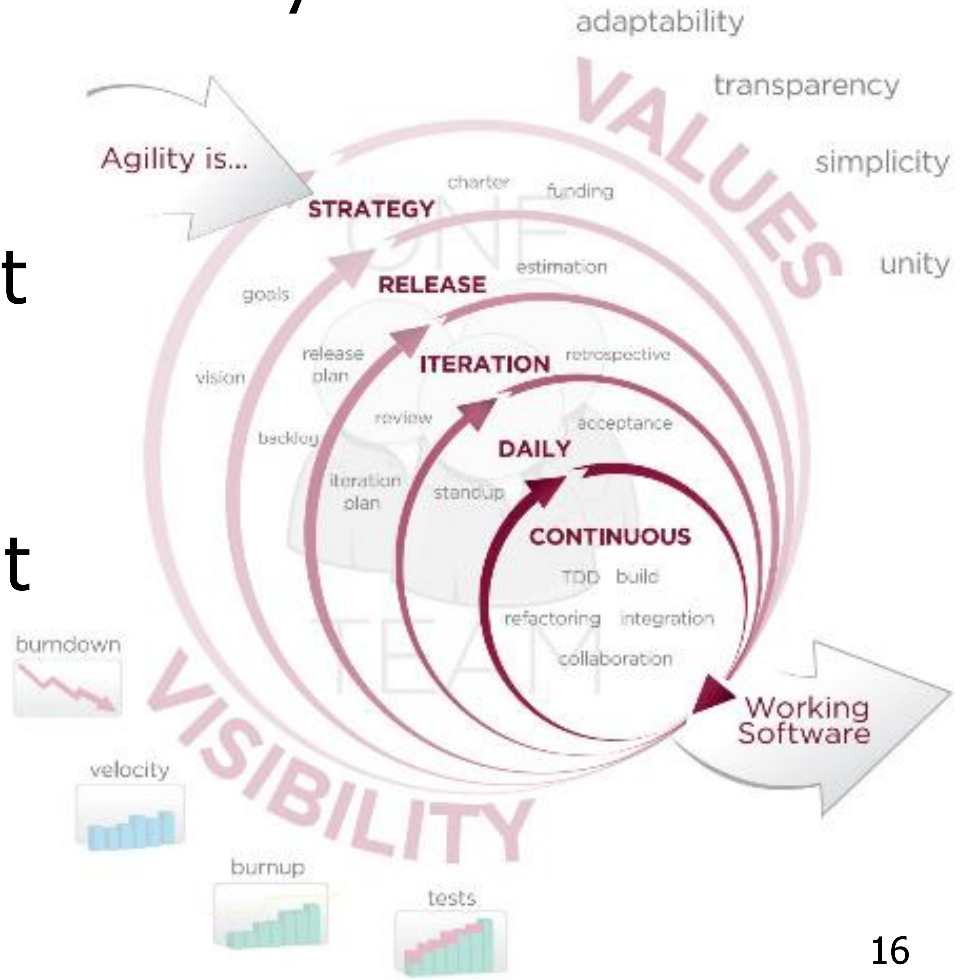
Dynamic Systems Development Method (DSDM)	**Extreme Programming (XP)**	**SCRUM**
An outgrowth of, and extension to, rapid application development (RAD) Boasts best-supported training/ documentation of any Agile method. DSDM’s principles include: • Active user involvement frequent delivery • Team decision making • Integrated testing throughout SDLC • Reversible changes in development.	Values of community, simplicity, feedback, and courage. Emphasis on cost of change and technical excellence through refactoring and test-first development. A proven system of dynamic practices, with integrity as a holistic unit: • Pair programmming • Daily cycles • Direct involvement of customer	Provides a project management framework Cycles are 2-4 week Sprints drvien by specified Backlog features. Team lead is Scrum master. Daily 15-minute team meetings for coordination and integration. Scrum has been in use for nearly ten years may now be most widely usuccessfully used Agile practice.

RUP is an Agile process



Agile methodologies involve

- Feature-driven development: business stories – scenarios – to define functionality
- Involving users all the time
- Continuous development with shared code ownership
- Test-driven development writing tests first
- Frequent integration



Reviewing the test process

- Testing consists of the following main activities
  1. Planning and control

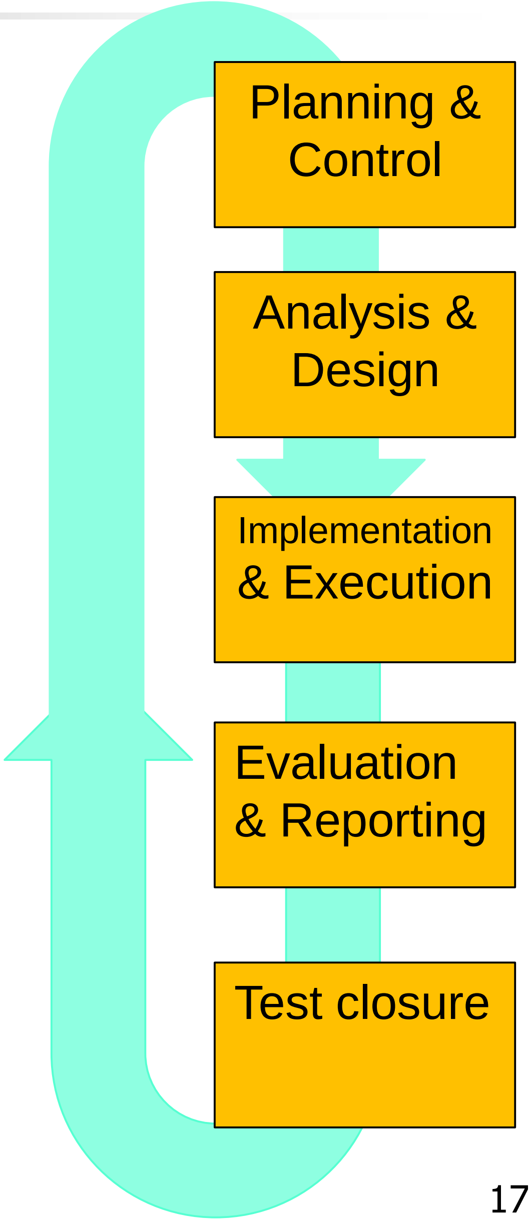
     When goals of test are defined
  2. Analysis and Design

     When test cases are specified
  3. Implementation and execution

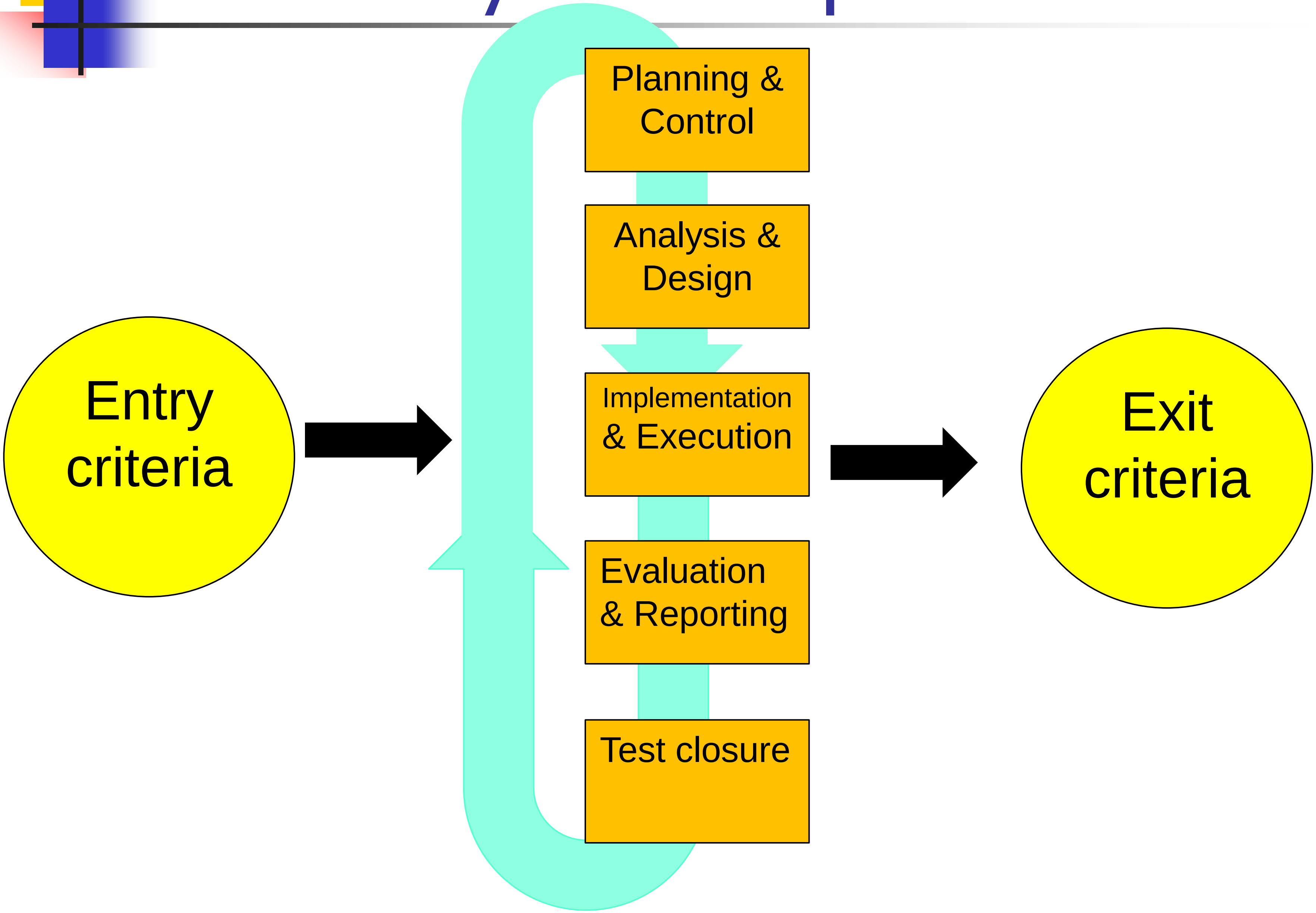
     When test suites are run
  4. Evaluating exit criteria and reporting

     When test results are analyzed
  5. Test closure activities

     When test artifacts are archived



Summary of test process



Entry criteria

Planning & Control

Analysis & Design

Implementation & Execution

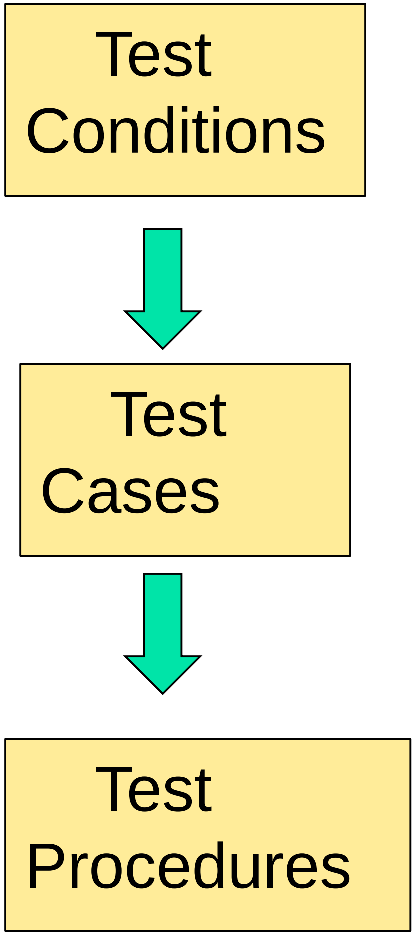
Evaluation & Reporting

Test closure

Exit criteria

Test Design Activities

- Start with test obectives
- Define:
  - Test conditions
    Does the software pass?
  - Test Cases
    Includes test data
  - Test procedures
    Combine sequence of test into repeatable, automatable scripts



Test Conditions

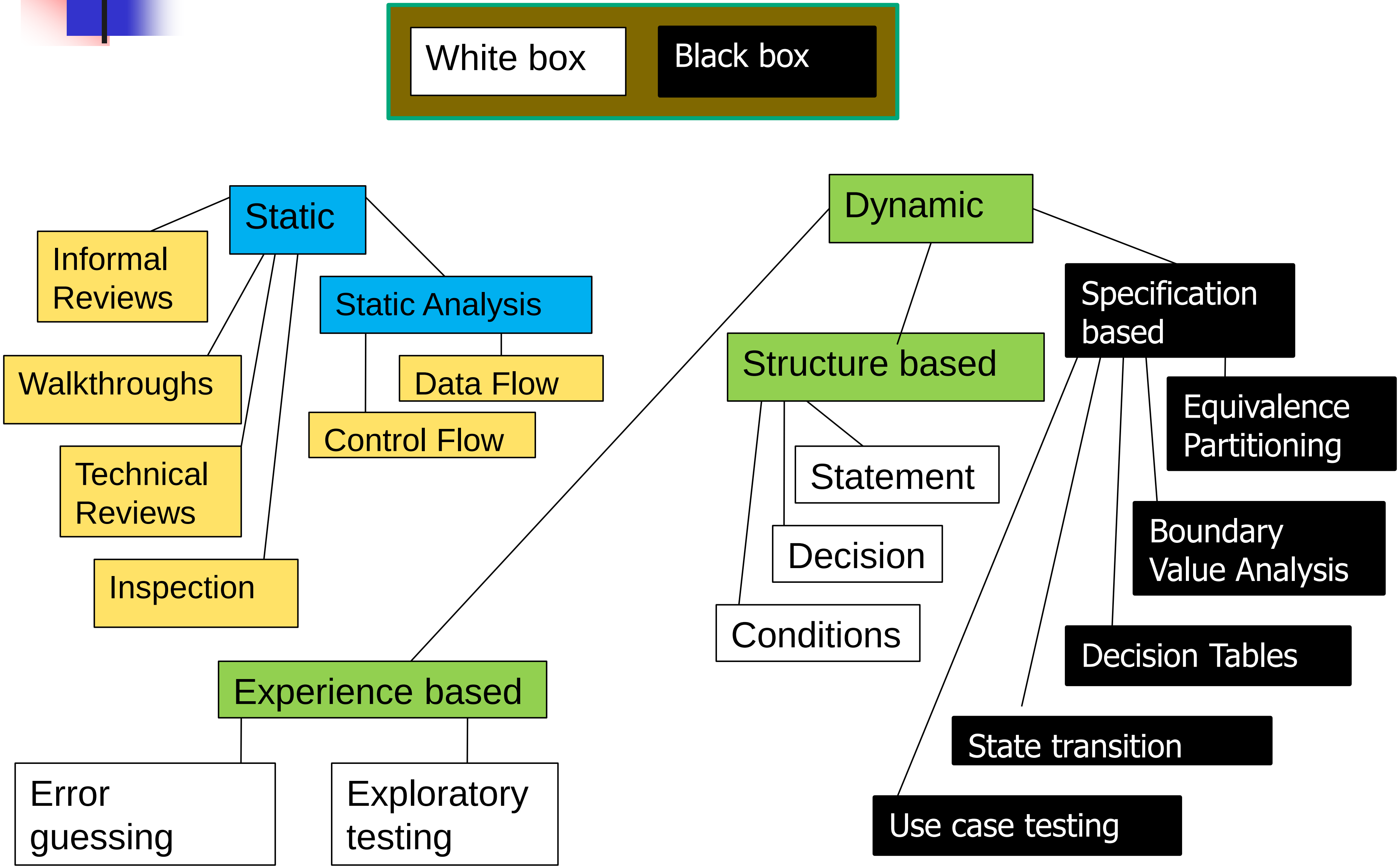
Test Cases

Test Procedures

Review: Goals of the test process

- **Validate software product**
  - Assess fitness for use
- **Verify software artifacts**
  - Evaluate correctness of code and documents
- **Gather metrics to measure quality**
  - Is product ready for release?
  - Is quality assurance process working effectively?
- **Identify and manage defects.**
  - What is a defect?
    - A deviation from required/expected
    - Software bug/fault
    - Design flaw
    - Requirements – omission or misconception

Overview of testing techniques



Tests used throughout SDLC

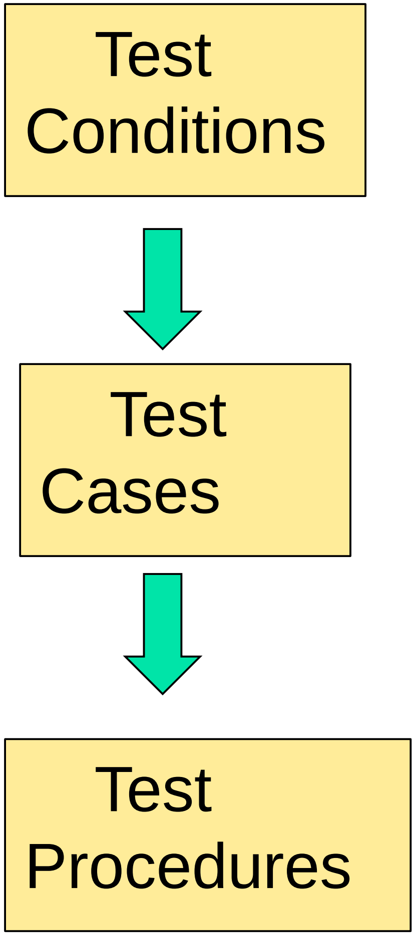
- Functional quality characteristics
  - Functional testing includes
    - Suitability, Interoperability, Security, Accuracy, Compliance
- Non-functional quality characteristics
  - Non functional testing includes
    - Performance, load, stress testing

ISO/IEC 9126

- Structure or architecture
  - Measuring thoroughness – code coverage
- Testing related to changes
  - Confirmation testing to verify success of corrective actions
  - Regression testing to ensure changes introduce no new defects

Test Design Activities

- For each goal/objective from planning stage, define:
  - Test conditions
    - Refine generic goals or objectives
  - Test Cases
    - Add specific data
      precondition + action → postcondition
  - Test suites / scripts
    - Combine sets of tests for automatic execution
- Test procedures that testers follow
  - Should ensure that high level goals are tested



Test design artifacts

- **Test objectives**
  - Based on requirements, architecture, design, …
  - Evaluate testability of test basis
- **Test conditions/ criteria**
  - Define what success means
  - Identify and prioritize conditions
- **Test cases**
  - Identifying test data to support cases and conditions
  - Specify test environment, required infrastructure and tools