Verification and Validation

Objectives

- To introduce software verification and validation and to discuss the distinction between them
- To describe the program inspection process and its role in V & V
- To explain static analysis as a verification technique

Topics covered

- Verification and validation planning
- Software inspections
- Automated static analysis

Verification vs validation

- Verification:
  "Are we building the product right".
  - The software should conform to its specification.
- Validation:
  "Are we building the right product".
  - The software should do what the user really wants.

The V & V process

- Is a whole life-cycle process - V & V must be applied at each stage in the software process.
- Has two principal objectives
  - The discovery of defects in a system;
  - The assessment of whether or not the system is useful and useable in an operational situation.

V & V goals

- Verification and validation should establish confidence that the software is fit for purpose.
- This does NOT mean completely free of defects.
- Rather, it must be good enough for its intended use and the type of use will determine the degree of confidence that is needed.

V & V confidence

- Depends on system's purpose, user expectations and marketing environment
  - Software function
    - The level of confidence depends on how critical the software is to an organisation.
  - User expectations
    - Users may have low expectations of certain kinds of software.
  - Marketing environment
    - Getting a product to market early may be more important than finding defects in the program.

IV & V: Independent Validation and Verification

- Can be done by another internal team or external (other company)

developer

Understands the system
but, will test "gently"
and, is driven by "delivery"

independent tester

Must learn about the system,
but, will attempt to break it
and, is driven by quality

Static and dynamic verification

- Software inspections. Concerned with analysis of the static system representation to discover problems (static verification)
  - May be supplement by tool-based document and code analysis
- Software testing. Concerned with exercising and observing product behaviour (dynamic verification)
  - The system is executed with test data and its operational behaviour is observed

Program testing

- Can reveal the presence of errors NOT their absence.
- The only validation technique for non-functional requirements is the software has to be executed to see how it behaves.
- Should be used in conjunction with static verification to provide full V&V coverage.

Types of testing

- Defect testing
  - Tests designed to discover system defects.
  - A successful defect test is one which reveals the presence of defects in a system.
  - Covered in next lecture
- Validation testing
  - Intended to show that the software meets its requirements.
  - A successful test is one that shows that a requirements has been properly implemented.

Testing and debugging

- Defect testing and debugging are distinct processes.
- Verification and validation is concerned with establishing the existence of defects in a program.
- Debugging is concerned with locating and repairing these errors.
- Debugging involves formulating a hypothesis about program behaviour then testing these hypotheses to find the system error.

The debugging process

Debugging Techniques

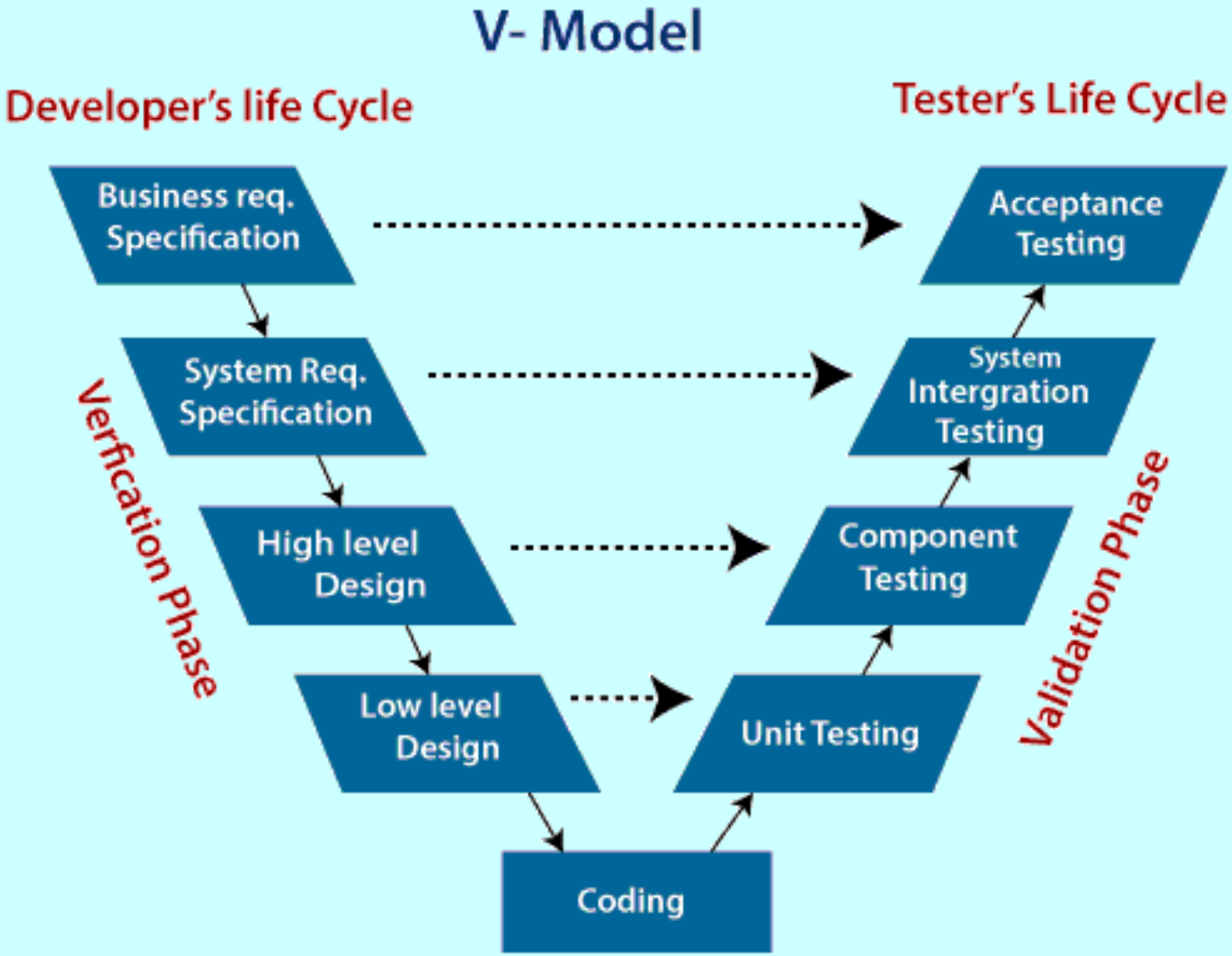
- brute force
- backtracking
- Cause elimination

When all else fails, ask for help!

V & V planning

- Careful planning is required to get the most out of testing and inspection processes.
- Planning should start early in the development process.
- The plan should identify the balance between static verification and testing.
- Test planning is about defining standards for the testing process rather than describing product tests.

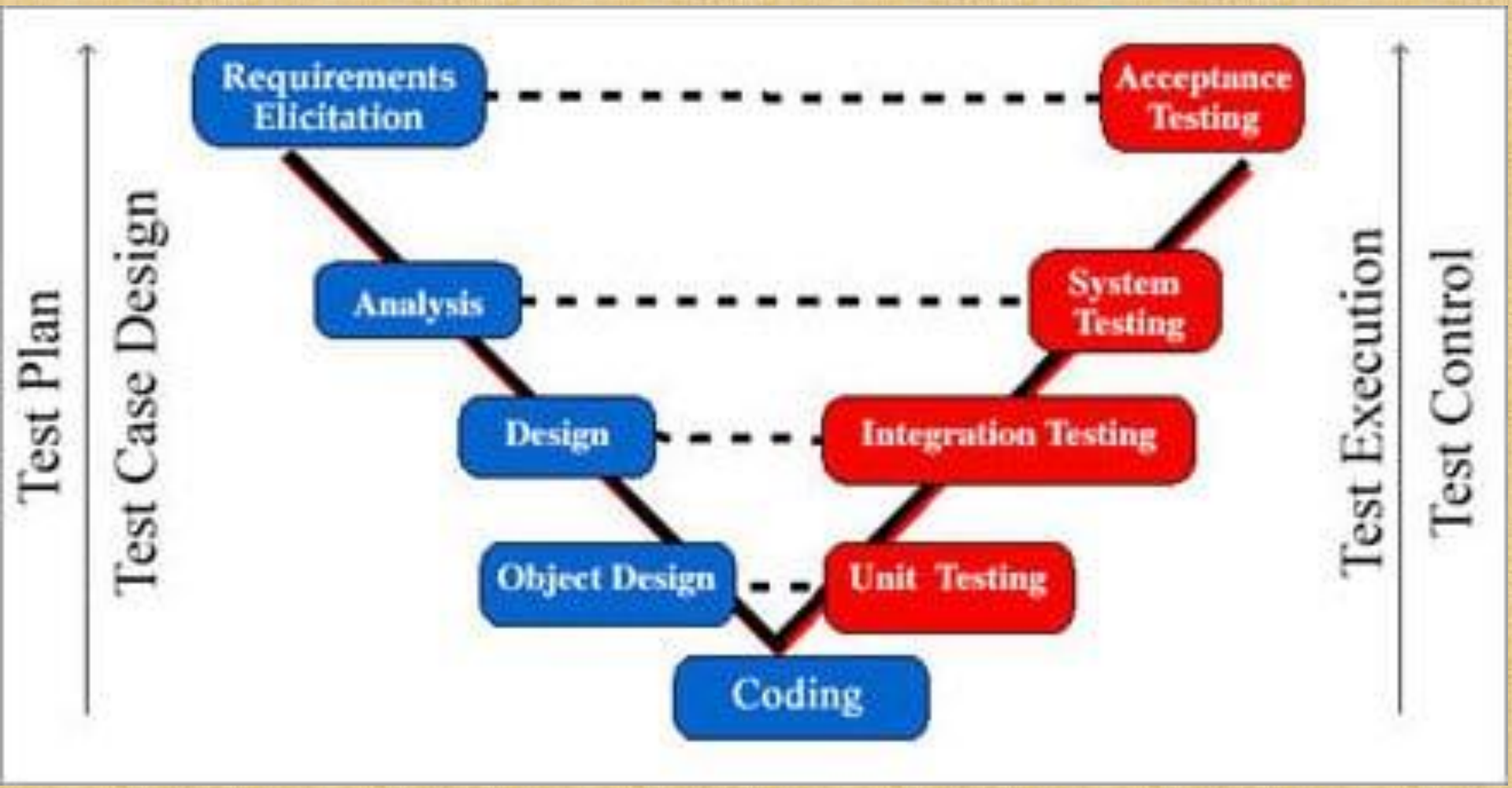
The V-model of development



The structure of a software test plan

- The testing process.
- Requirements traceability.
- Tested items.
- Testing schedule.
- Test recording procedures.
- Hardware and software requirements.
- Constraints.

The software test plan



Software inspections

- These involve people examining the source representation with the aim of discovering anomalies and defects.
- Inspections do not require execution of a system so may be used before implementation.
- They may be applied to any representation of the system (requirements, design,configuration data, test data, etc.).
- They have been shown to be an effective technique for discovering program errors.