INTRODUCTION TO SOFTWARE TESTING

QA, QC AND TESTING

QA – QUALITY ASSURANCE

QC – QUALITY CONTROL

TESTING

QA – QUALITY ASSURANCE

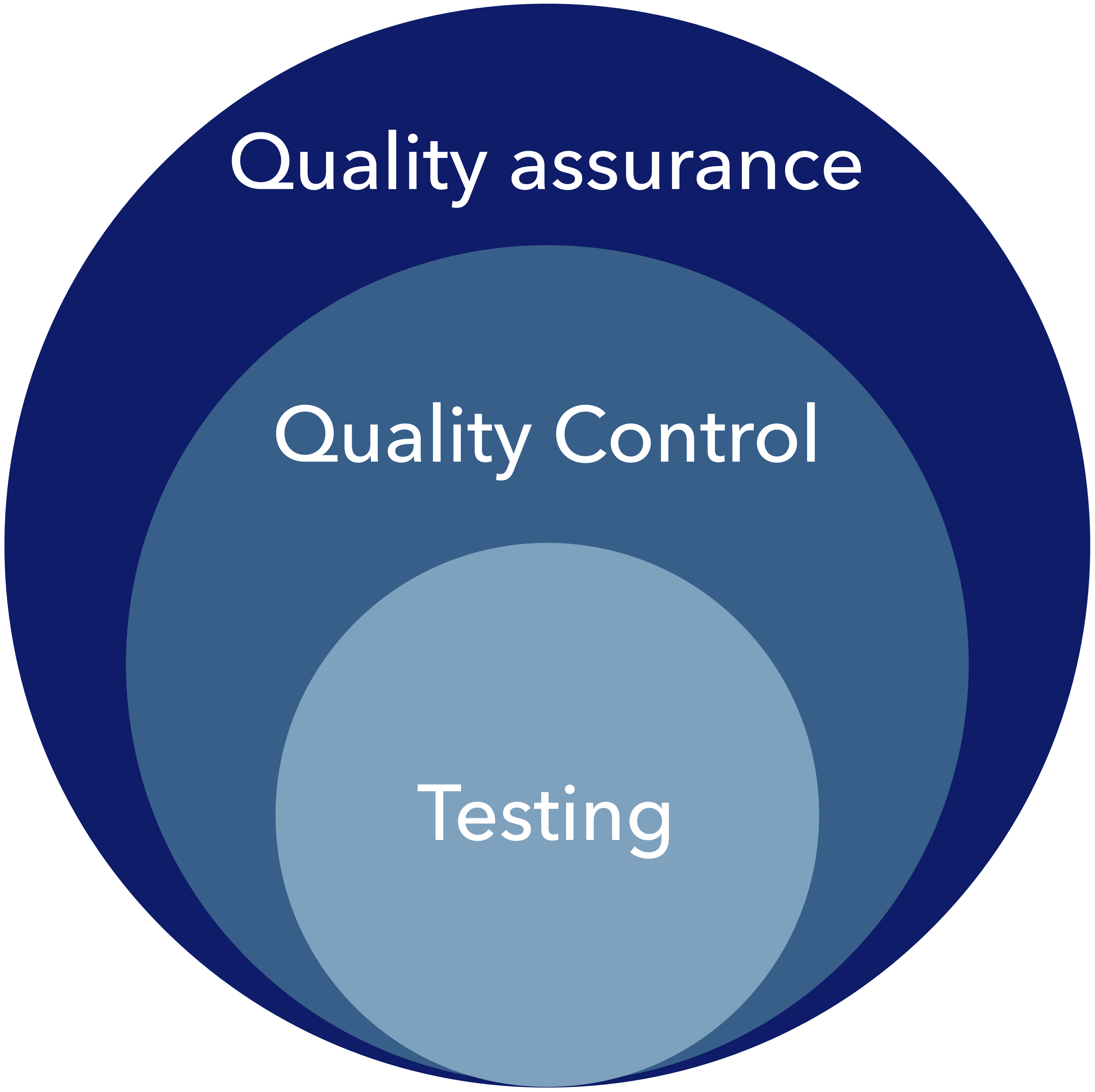
- Setting up adequate processes to ensure the quality of the product.
- Prevent possible bugs instead of detect bugs.
- Process oriented activities.

QC- QUALITY CONTROL

- Making sure that the product correspond to the requirements.
- Requirements verification.
- Product oriented activities.

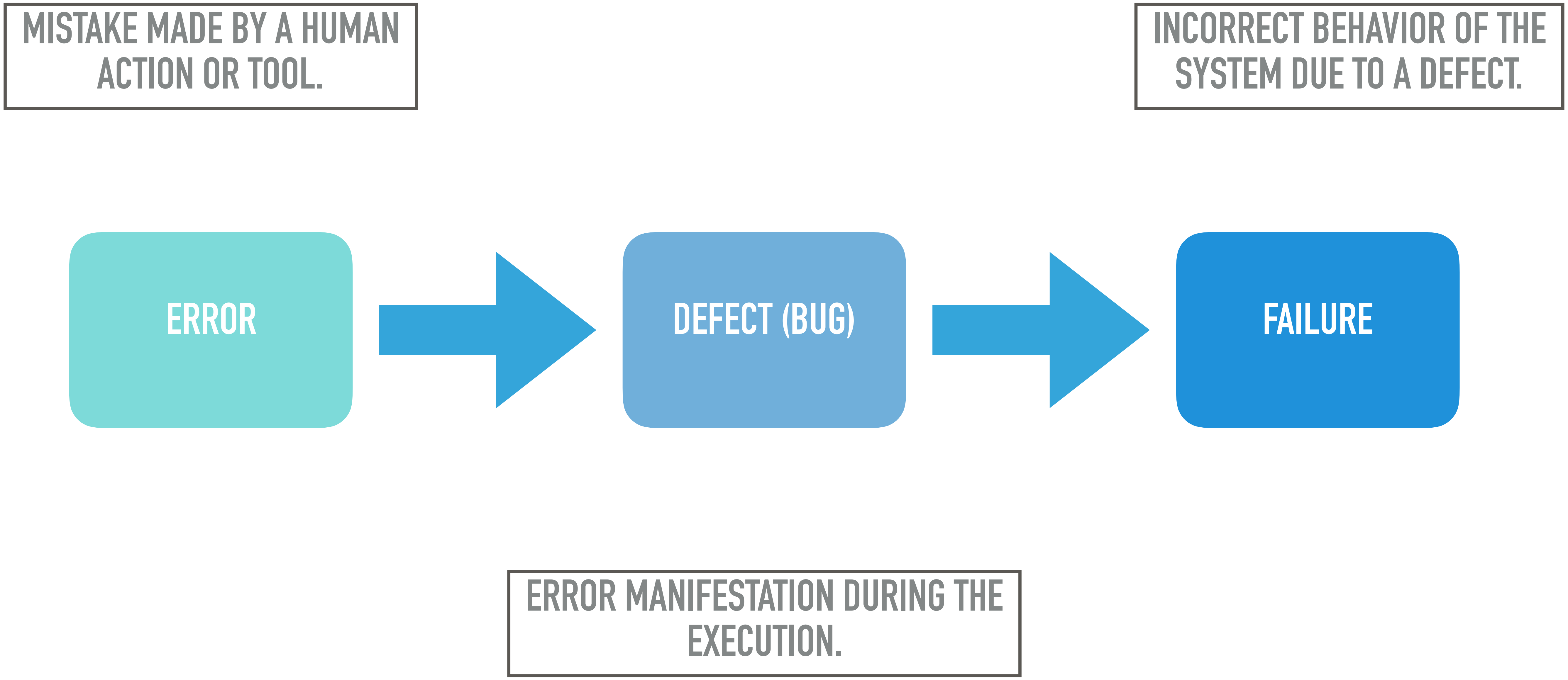
TESTING

- Exploring to system to check how it works.
- Detect possible bugs (different techniques).
- Product oriented.


Quality assurance
Quality Control
Testing

ERROR, DEFECT, FAILURE

ARE THEY THE SAME?


MISTAKE MADE BY A HUMAN ACTION OR TOOL.
INCORRECT BEHAVIOR OF THE SYSTEM DUE TO A DEFECT.
ERROR
DEFECT (BUG)
FAILURE
ERROR MANIFESTATION DURING THE EXECUTION.

According ISTQB (International Software Testing Qualifications Board)

SEVEN TESTING PRINCIPLES

1. TESTING SHOWS THE PRESENCE OF DEFECTS, NOT THEIR ABSENCE

2. EXHAUSTIVE TESTING IS IMPOSSIBLE

3. EARLY TESTING SAVES TIME AND MONEY

4. DEFECTS CLUSTER TOGETHER

5. BEWARE OF THE PESTICIDE PARADOX

6. TESTING IS CONTEXT DEPENDENT

7. ABSENCE-OF-ERRORS IS A FALLACY

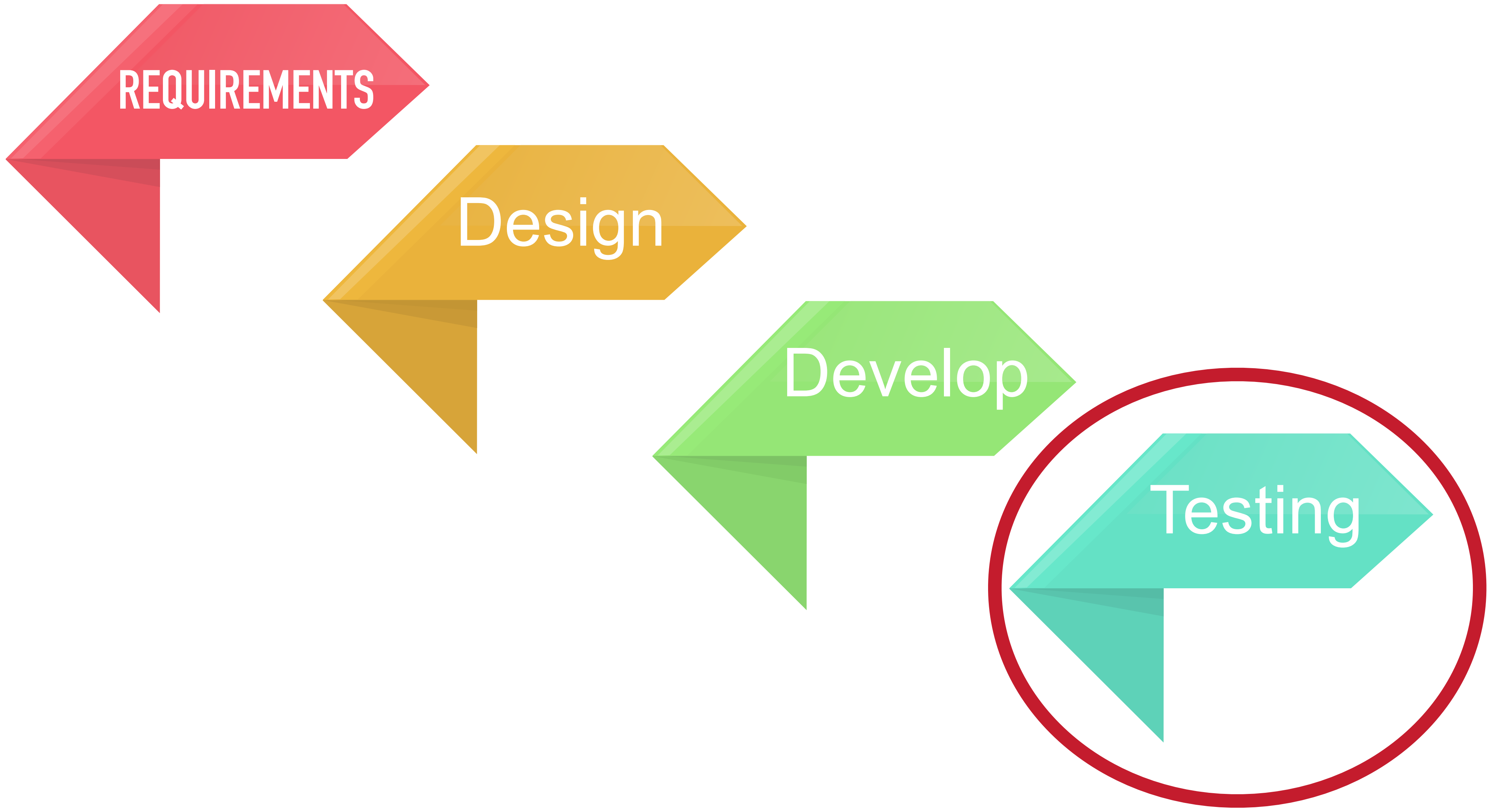
TESTING THROUGHOUT THE SOFTWARE DEVELOPMENT LIFECYCLE

WHAT TYPE OF SDLC DO YOU KNOW?

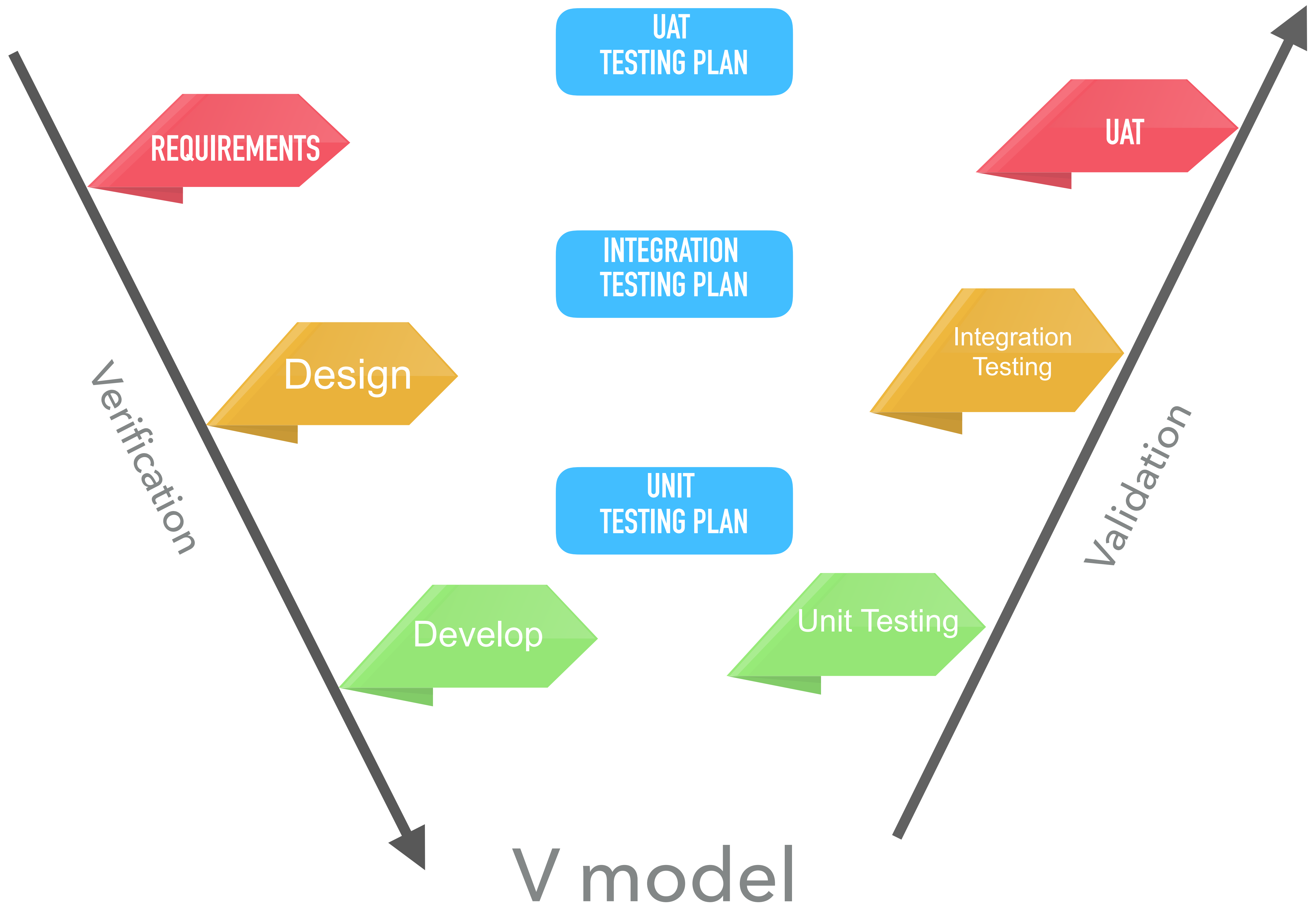
- Sequential development models
- Iterative and incremental development models

- Sequential development models
  - Examples:
    - Waterfall
    - V Model

- WaterFall



TESTING THROUGHOUT THE SOFTWARE DEVELOPMENT LIFECYCLE



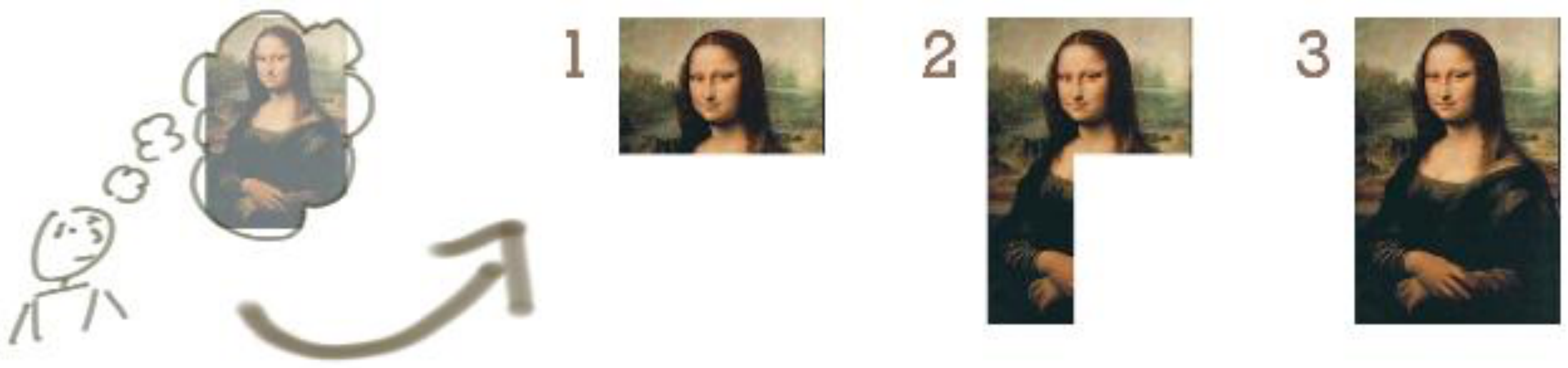
- Iterative and incremental development models

The Mona Lisa Analogy By Jeff Paton

- Incremental

Incrementing calls for a fully formed idea
"incrementing" builds a bit at a time.

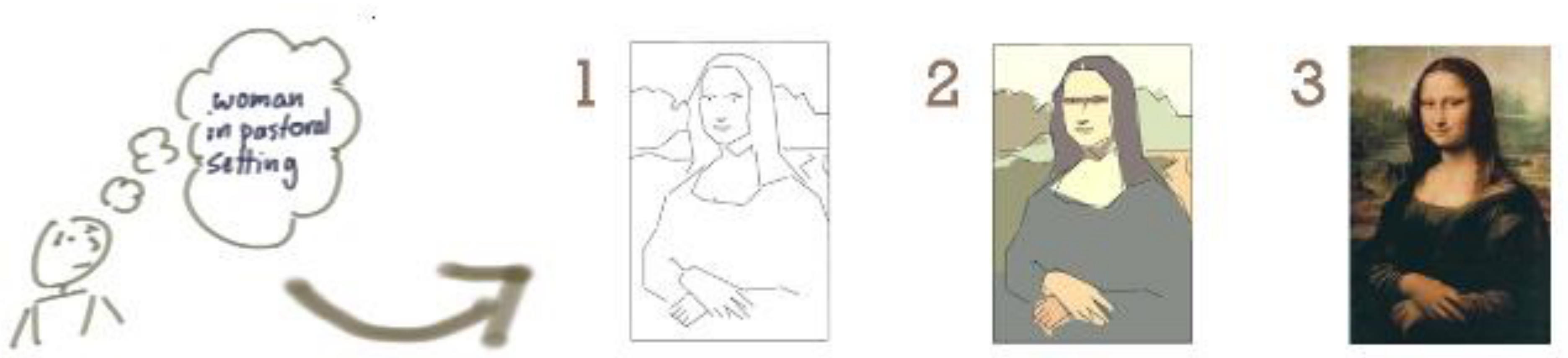
Jeff Paton



- Iterative

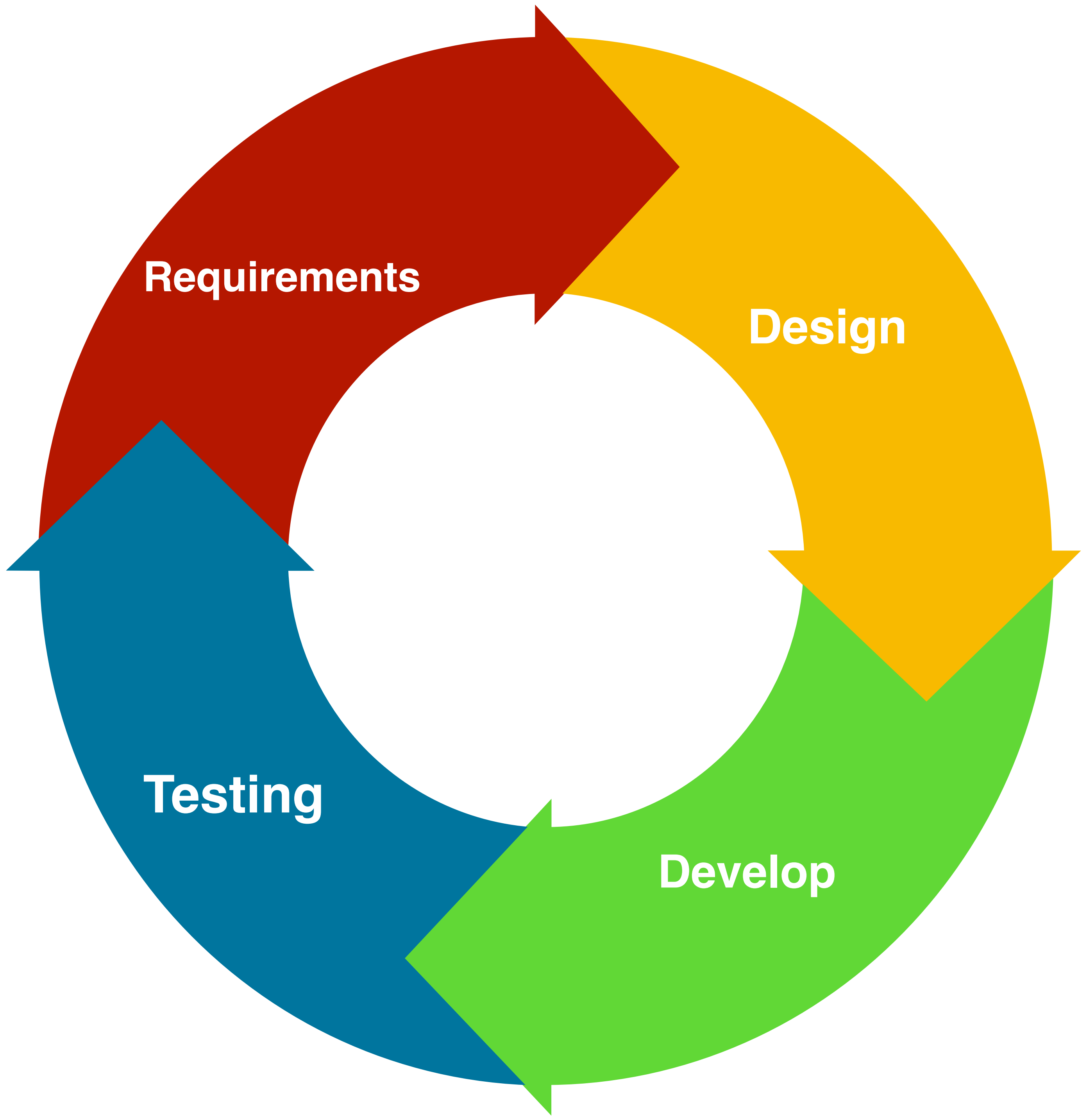
Iterating allows you to move from vague idea to realization "iterating" builds a rough version, validates it, then slowly builds up quality

Jeff Paton

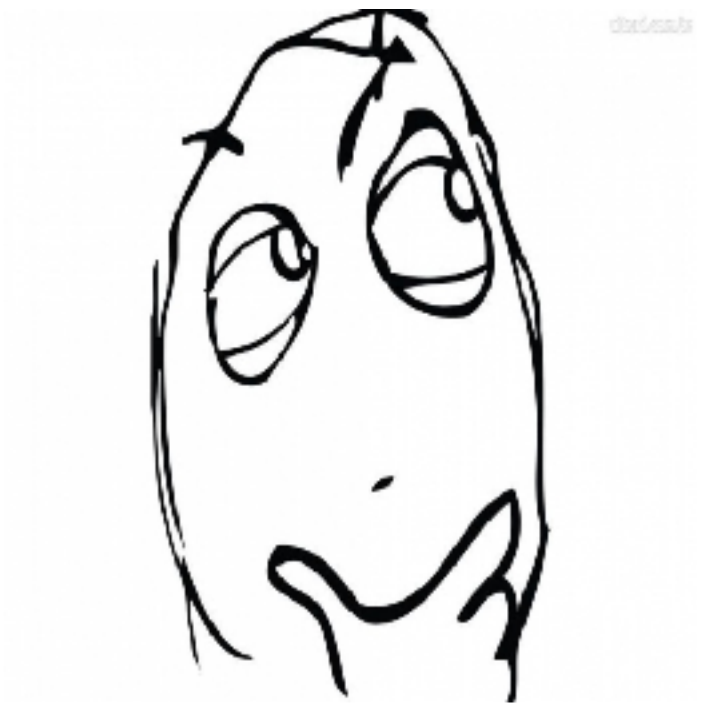


TESTING THROUGHOUT THE SOFTWARE DEVELOPMENT LIFECYCLE

- Agile software development



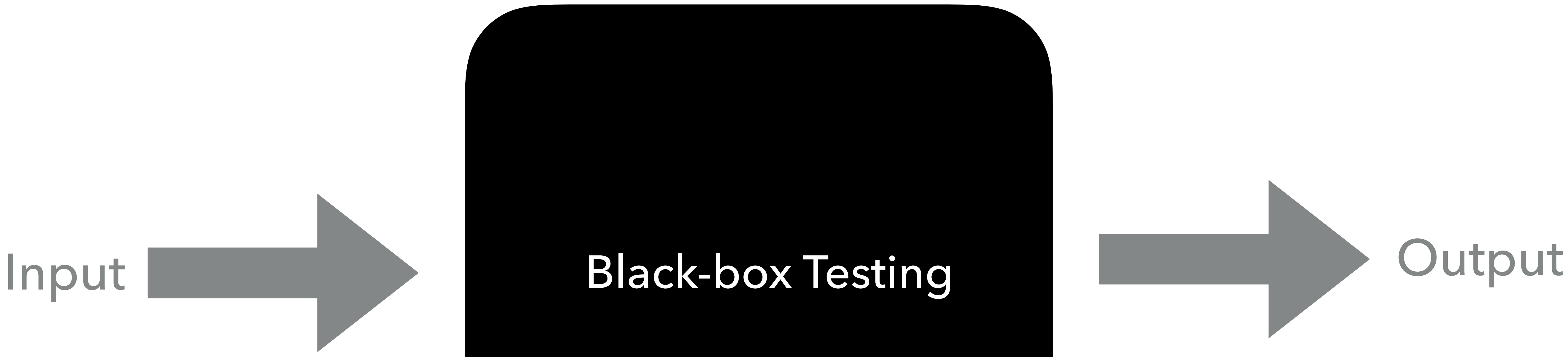
IS THIS A CLASS OF SOFTWARE TESTING OR SDLC?



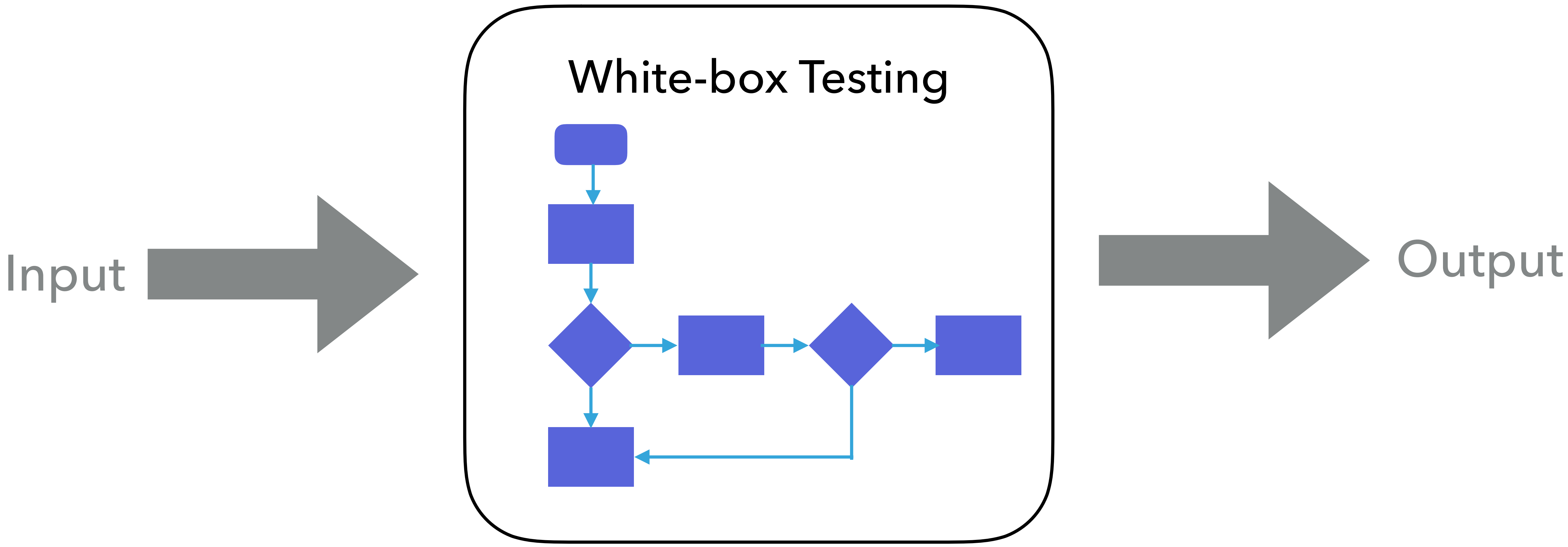
- No matter which software development lifecycle model is chosen.
- Test activities.
    - Early testing principle.

TEST TECHNIQUES

TEST TECHNIQUES

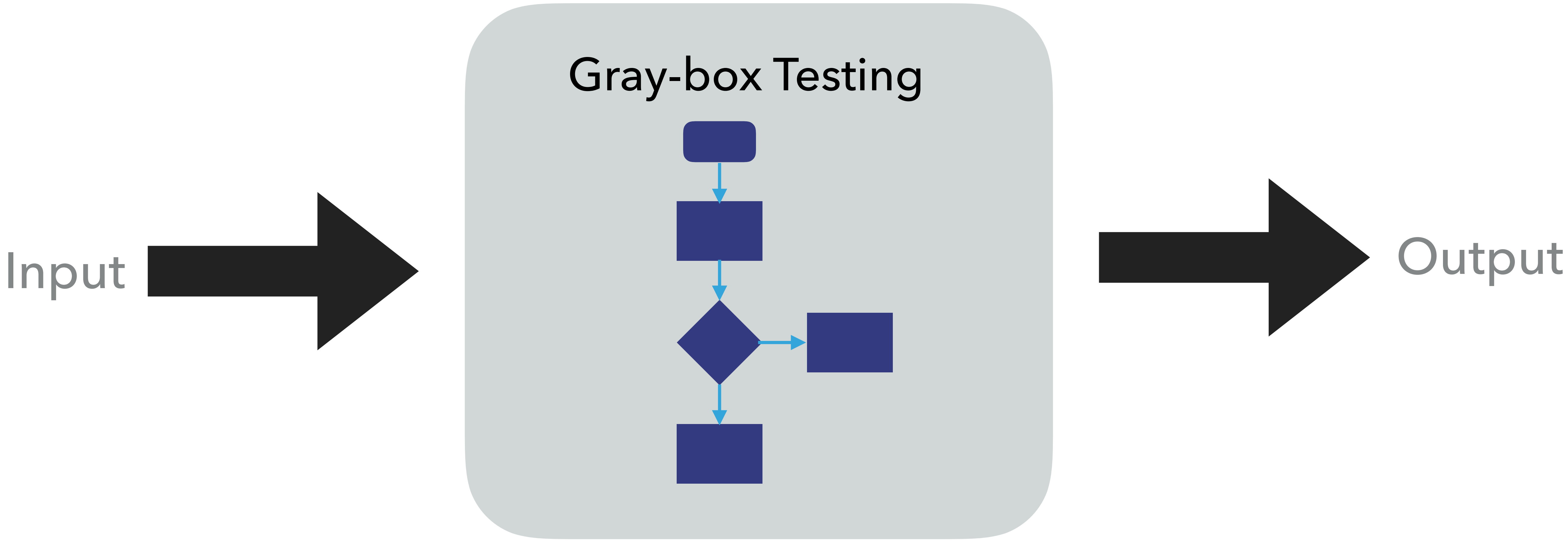


- Equivalence Partitioning
- Boundary Value Analysis
- Decision Table Testing
- State Transition Testing
- Use Case Testing



- Statement Testing
- Decision Testing

TEST TECHNIQUES



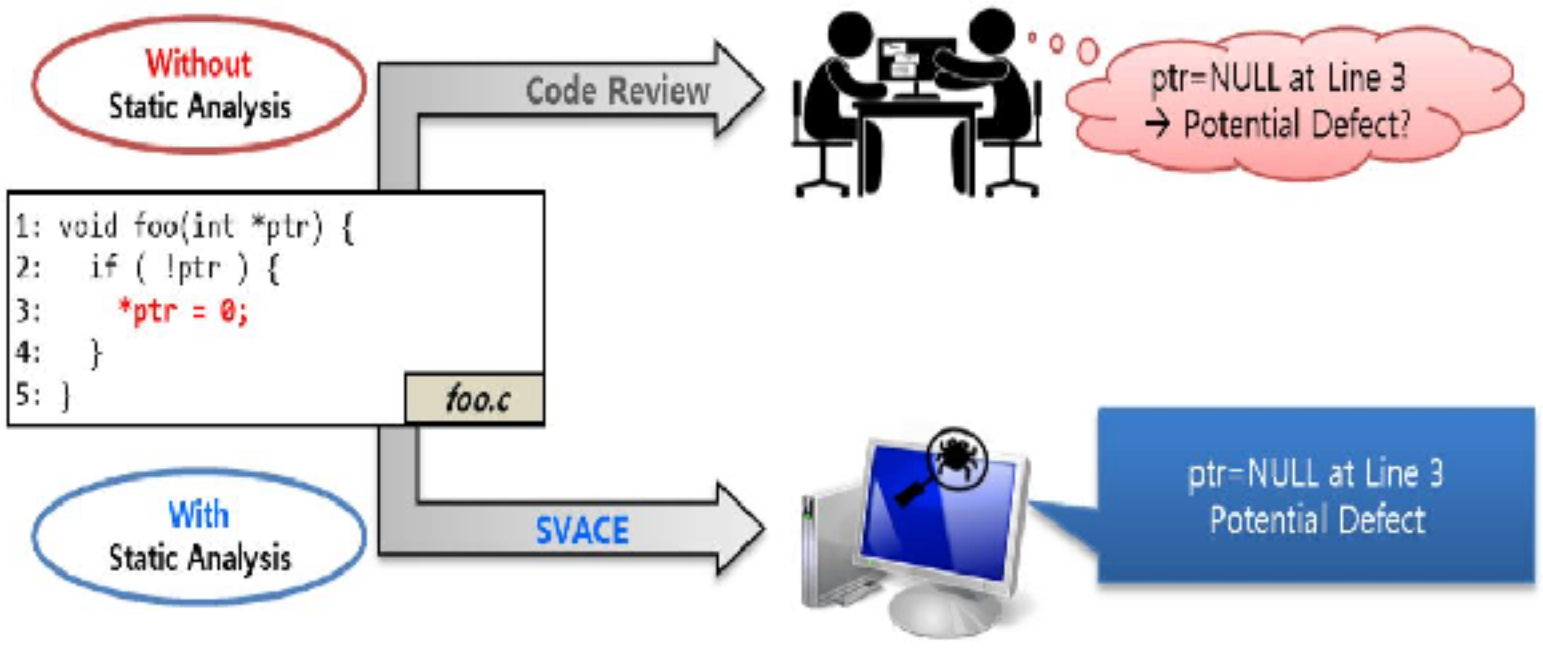
EXPERIENCE-BASED TESTING

- Error Guessing
- Exploratory Testing
- Checklist-based Testing

STATIC & DYNAMIC TESTING

STATIC TESTING

- Software is tested without execution.
- Two Parts:
  - Reviews
  - Static analysis



```
1: void foo(int *ptr) {
2:   if ( !ptr ) {
3:     *ptr = 0;
4:   }
5: }
```

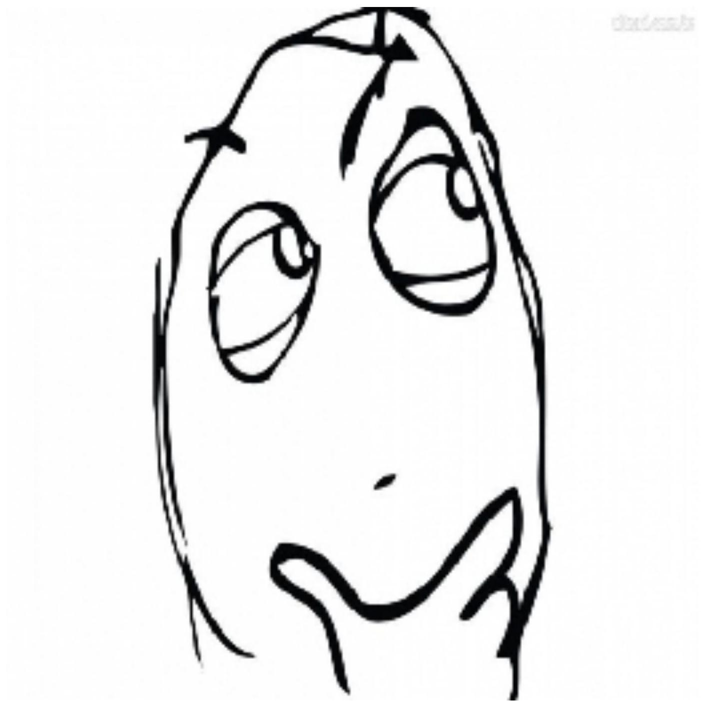
DYNAMIC TESTING

- Software is tested with execution.
- Examples:
  - Unit Testing
  - Integration Testing
  - E2E Testing

WHAT'S THE MAIN DIFFERENCE BETWEEN BOTH?

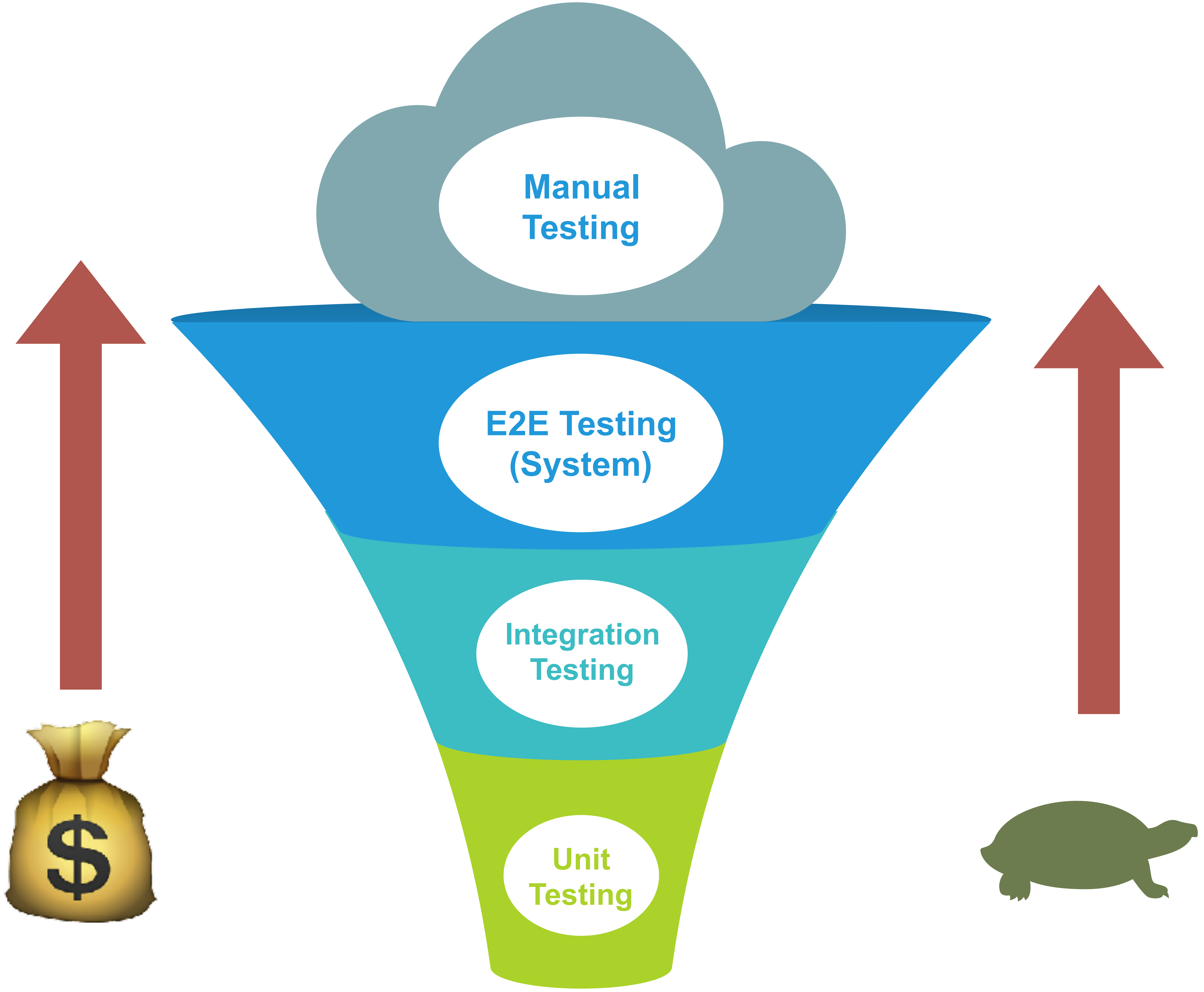
- Static Testing
  - Find defect not failures.
- Dynamic Testing
  - Find failures not defects.

$$\begin{tikzcd}\text{ERROR} \arrow[r] & \text{DEFECT} \arrow[r] & \text{FAILURE}\end{tikzcd}$$



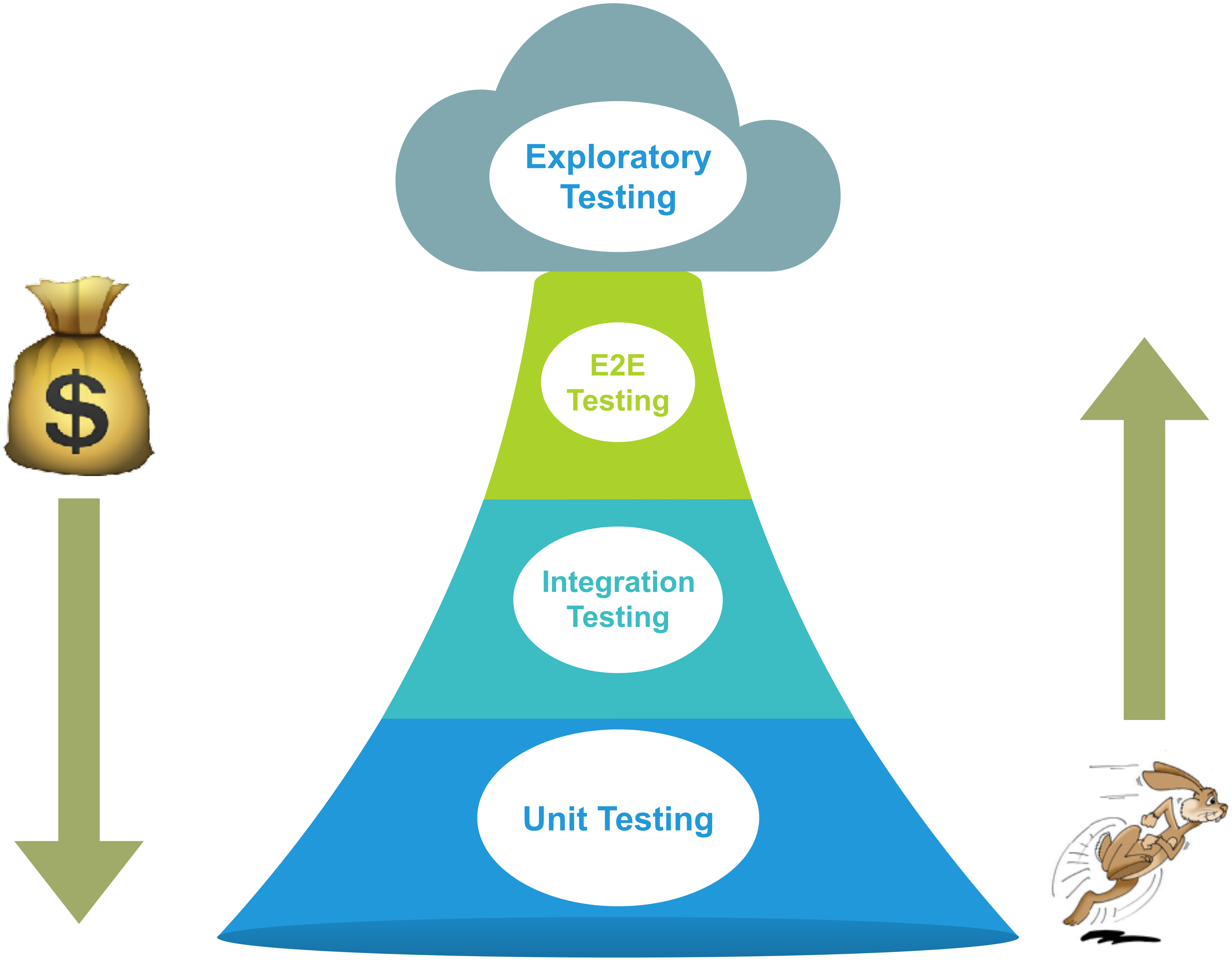
TEST PYRAMID

TRADITIONAL SOFTWARE DEVELOPMENT


Manual Testing
E2E Testing (System)
Integration Testing
Unit Testing
$

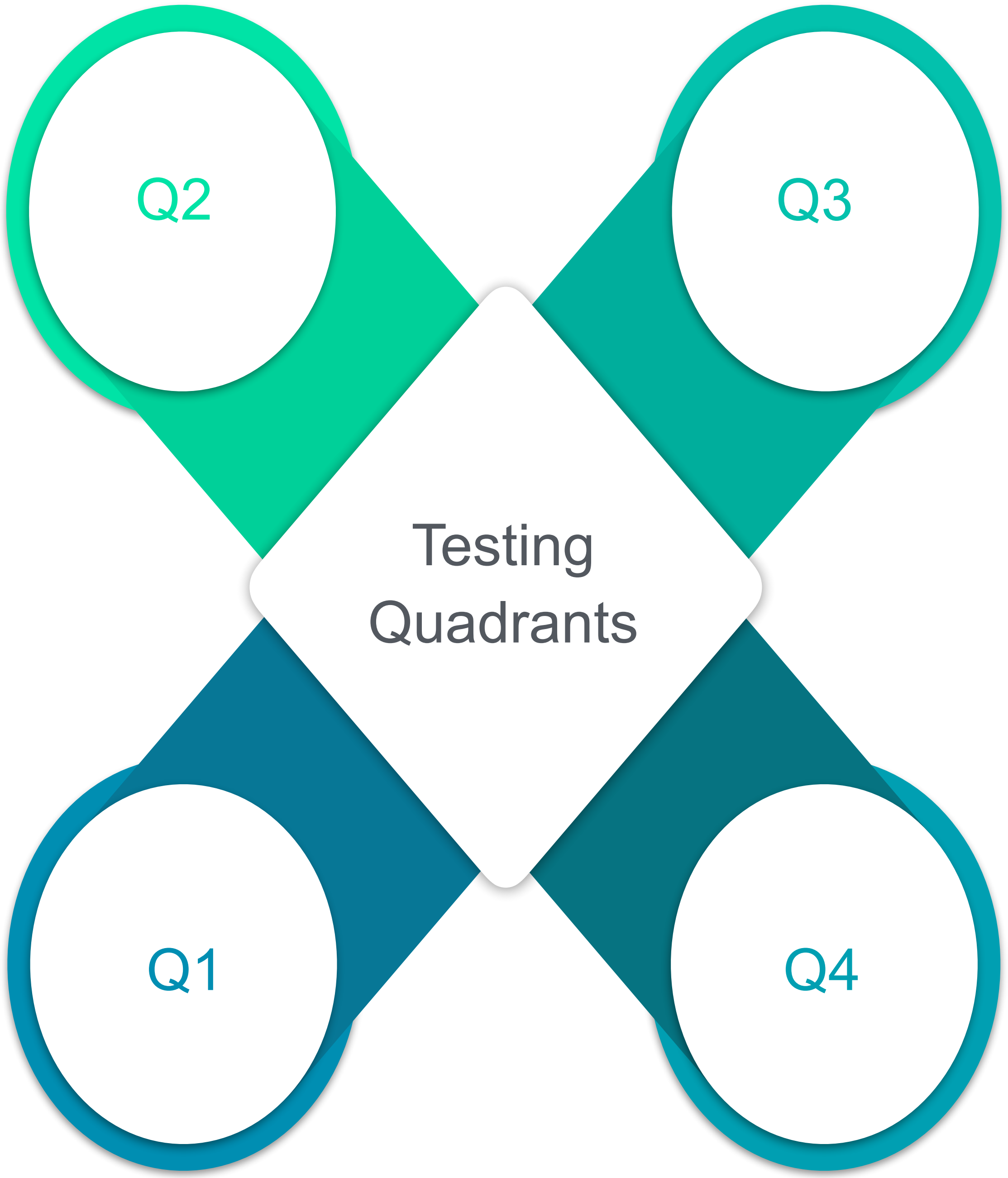
"ICE SCREAM TESTING MAKES EVERYONE WANT TO SCREAM!"

AGILE SOFTWARE DEVELOPMENT


Exploratory Testing
E2E Testing
Integration Testing
Unit Testing

TESTING QUADRANTS

- Defined by Brian Marick.
- Helps to ensure that all test types and test levels are included in the development cicle.
- Can be business or technology facing.
- Can be manually (supported by tools), automated or combination.



- System level.
- Business facing.
- Functional tests.
- Manually or Automated.
- Regression test suites.

- System or user acceptance level.
- Business facing.
- Critique tests for the product.
- Exploratory testing.
- Often Manually.

- Unit level.
- Technology Facing.
- Automated.
- Included in the CI process.

- System or operational acceptance level.
- Technology facing.
- Performance, load, stress, data migration.
- Monitoring.
- Automated.

WHY TESTING IS NECESSARY?

WHY TESTING IS NECESSARY?

- Humans make mistakes ALL the time.
- Reduce risks of failures.
    - Failures could be expensive or dangerous.
- Quality of the system.
- Meet the customer requirements.

THANK YOU.