Chapter 02

SW 테스팅 개요

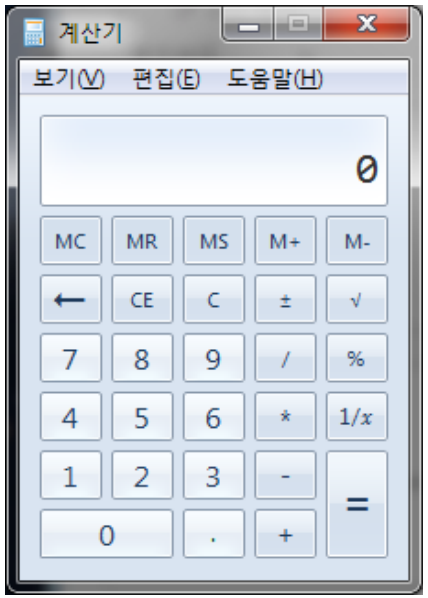
01. SW 테스팅 이란?

소프트웨어 테스팅

소프트웨어 테스팅이란 소프트웨어에 있는 결함을 찾아내는 활동

소프트웨어 테스팅의 수행

- 소프트웨어 테스팅의 수행은 소요되는 비용을 항상 고려해야함
- 테스팅을 대충 수행해서도 무작정 많이 수행해서도 안됨



계산기를 테스팅 하기 위해 각 버튼들을 조합하여 경우의 수를 생각해보면?

정수 계산 기능 테스팅
실수 계산 기능 테스팅

자릿수, 사칙연산, 루트, 그 이외의 기능 etc.....

02. SW 결함의 원인

소프트웨어의 결함 원인

- 요구사항 오류
  - 요구사항 정의 미흡
  - 요구사항 관리 실패
  - 개발자와의 소통 미흡

- 설계 오류
  - 요구사항이 제대로 반영되지 않은 설계

- 개발자 오류
  - 개발자의 실수(언어에 대한 이해 미흡, 개발환경에 대한 이해 미흡 등)

- 기타 오류
  - 통신 오류
  - 소프트웨어 구동 환경 오류
  - 다른 시스템과의 상호연동

03. SW 결함

결함(Fault)

- 소스코드에 있는 오류
- 개발자의 실수로 잘못 작성된 소스코드

에러(Error)

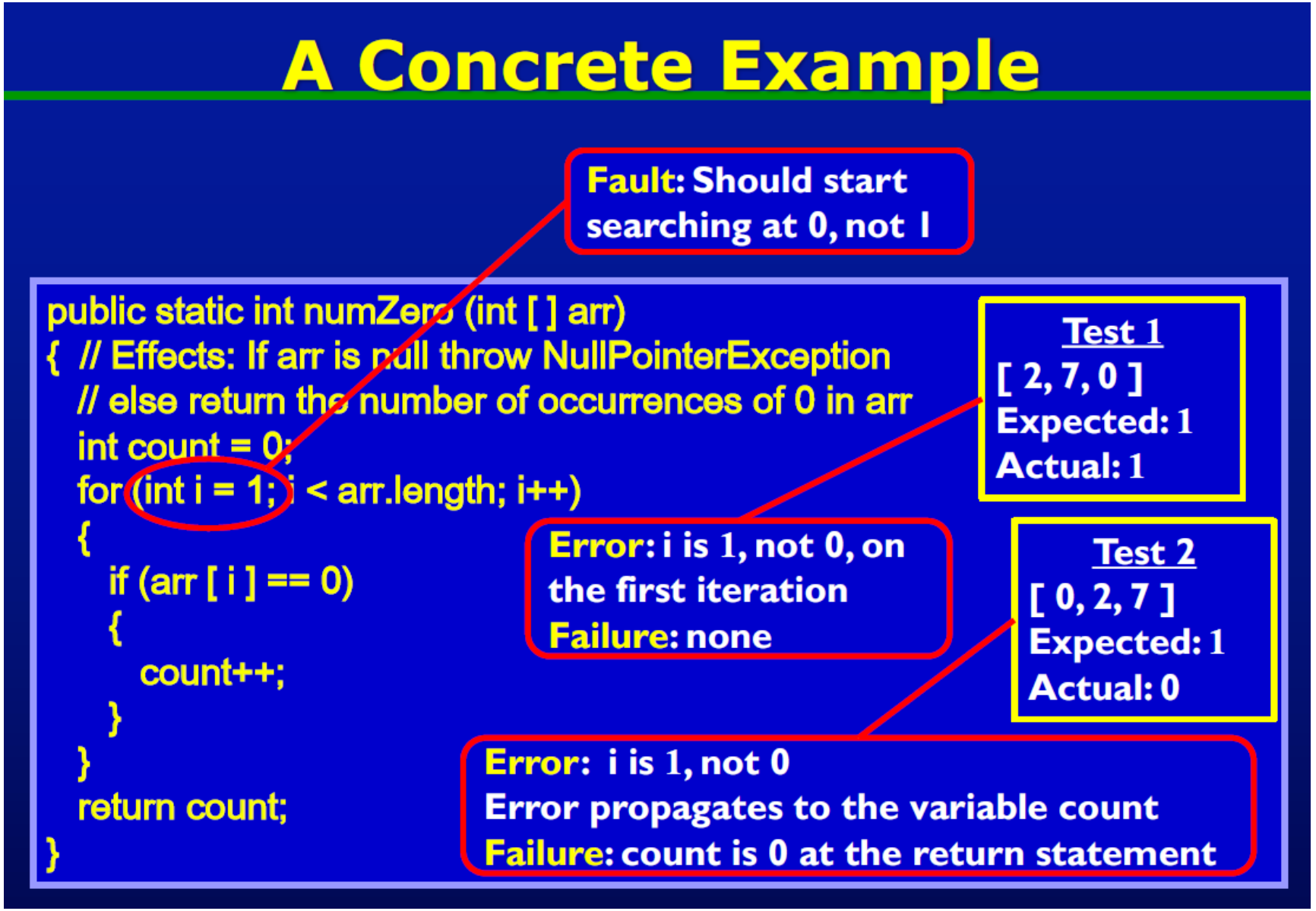
- 결함으로 인해 내부적으로 잘못 동작되고 있는 상태
- 결함으로 인해 프로그램이 잘못 동작하고 있는 상태

실패(Failure)

- 에러로 출력되는 결과

- 결함이 있다고 무조건 실패로 이어지진 않음
- 에러가 있다고 무조건 실패로 이어지진 않음

03. SW 결함예시


A Concrete Example
Fault: Should start searching at 0, not 1
public static int numZero (int [ ] arr)
{ // Effects: If arr is null throw NullPointerException
// else return the number of occurrences of 0 in arr
int count = 0;
for (int i = 1; i < arr.length; i++)
{
if (arr [i] == 0)
{
count++;
}
}
return count;
}
Test 1
[2, 7, 0]
Expected: 1
Actual: 1
Error: i is 1, not 0, on the first iteration
Failure: none
Test 2
[0, 2, 7]
Expected: 1
Actual: 0
Error: i is 1, not 0
Error propagates to the variable count
Failure: count is 0 at the return statement

04. 테스팅 기본 용어

테스팅 기본 용어

- 테스트 데이터(Test Data)
  - 시스템을 테스트하기 위한 입력

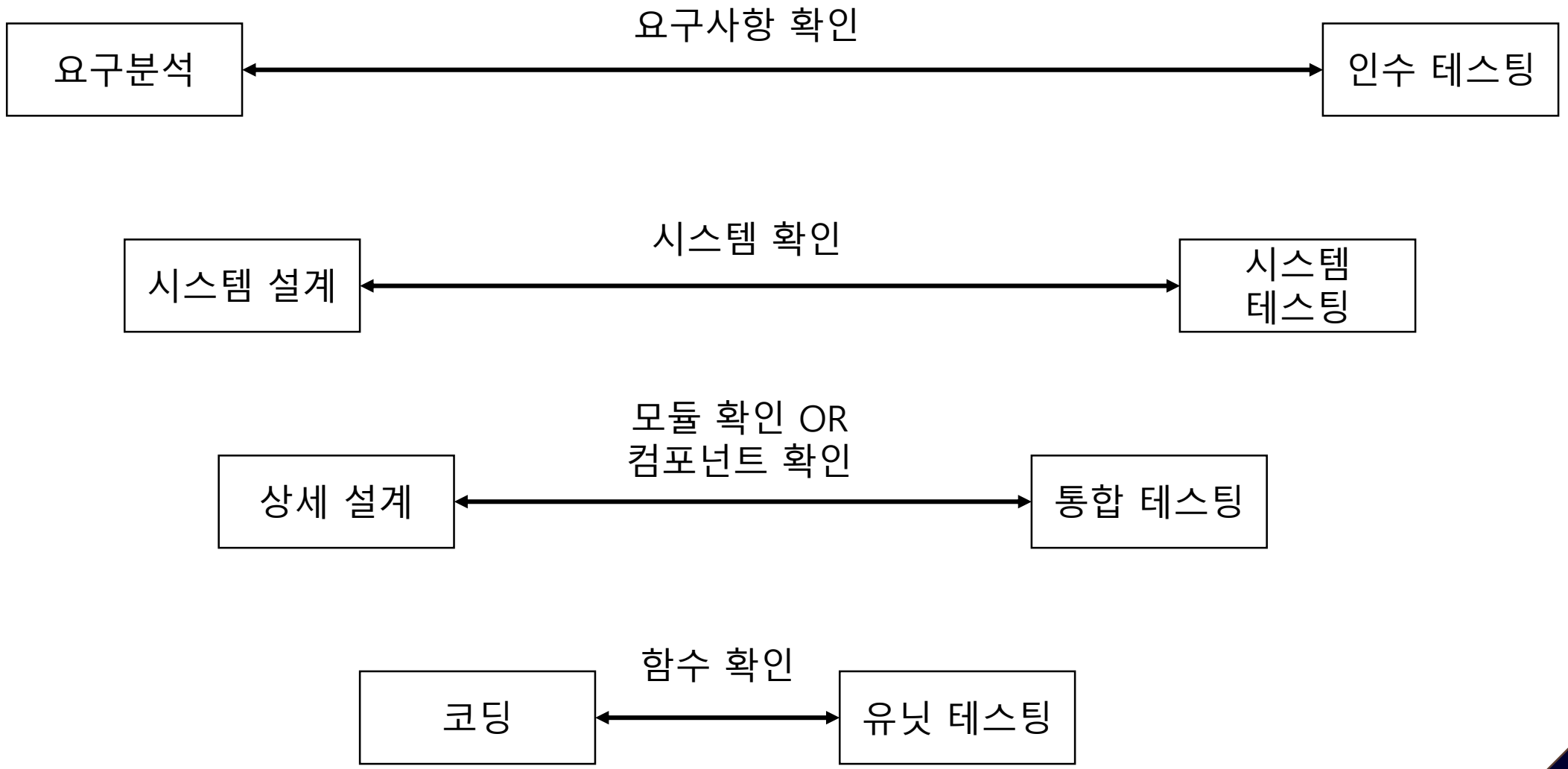
- 테스트 케이스(Test Case)
  - 시스템을 테스트하기 위해 Spec에 따라 작성된 입력과 예상 출력 결과

- 테스트 스위트(Test Suite)
  - 시스템을 테스트하는데 사용되는 테스트 케이스의 모음

- 테스트 오라클(Test Oracle)
  - 테스트가 통과했는지 실패했는지 판단하기 위해 테스터가 사용하는 메커니즘
  - 시스템의 스펙, 관련 문서 등

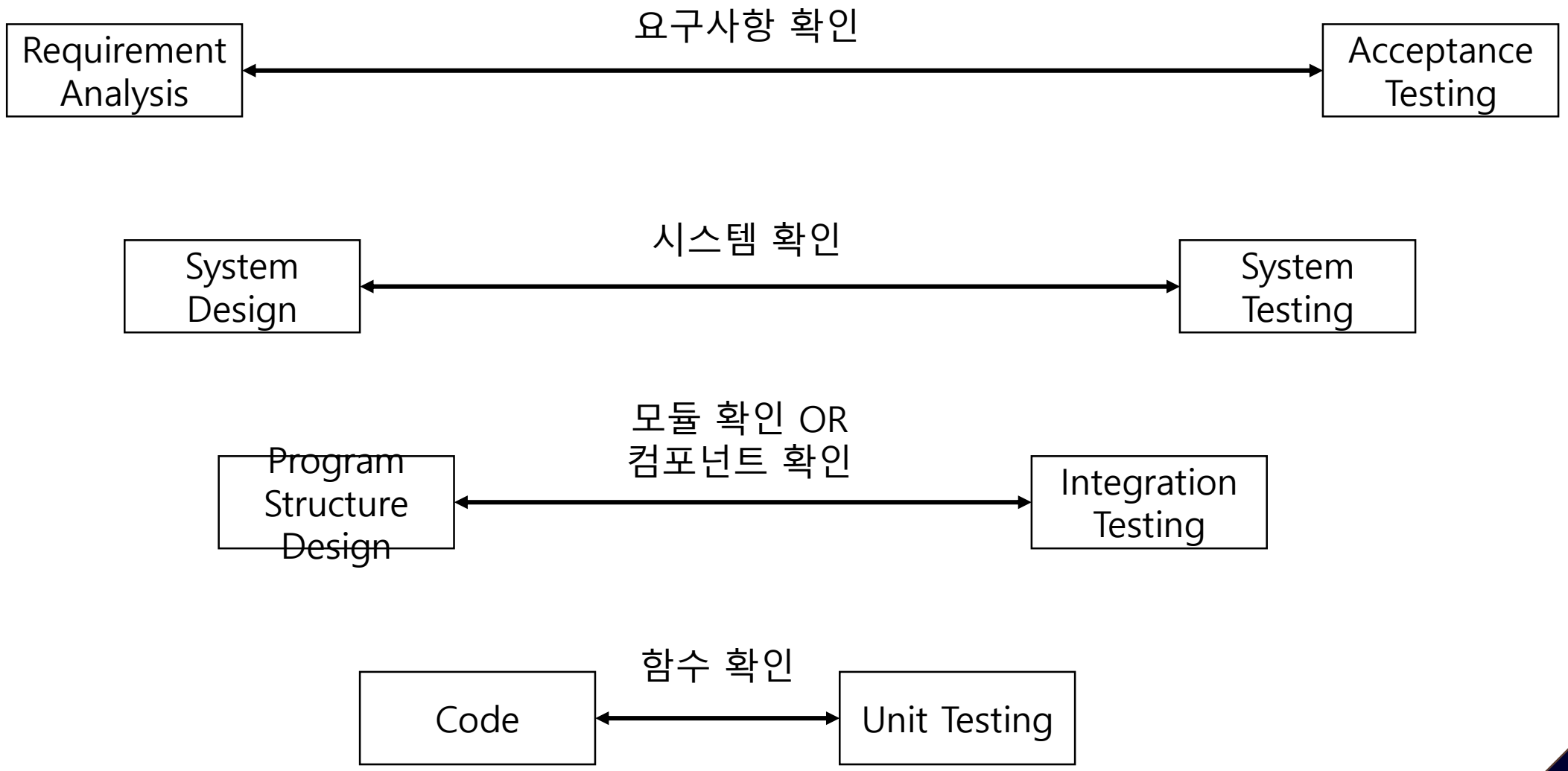
05. V모델

V모델


요구분석
요구사항 확인
인수 테스팅
시스템 설계
시스템 확인
시스템
테스팅
상세 설계
모듈 확인 OR
컴포넌트 확인
통합 테스팅
코딩
함수 확인
유닛 테스팅

05. V-Model

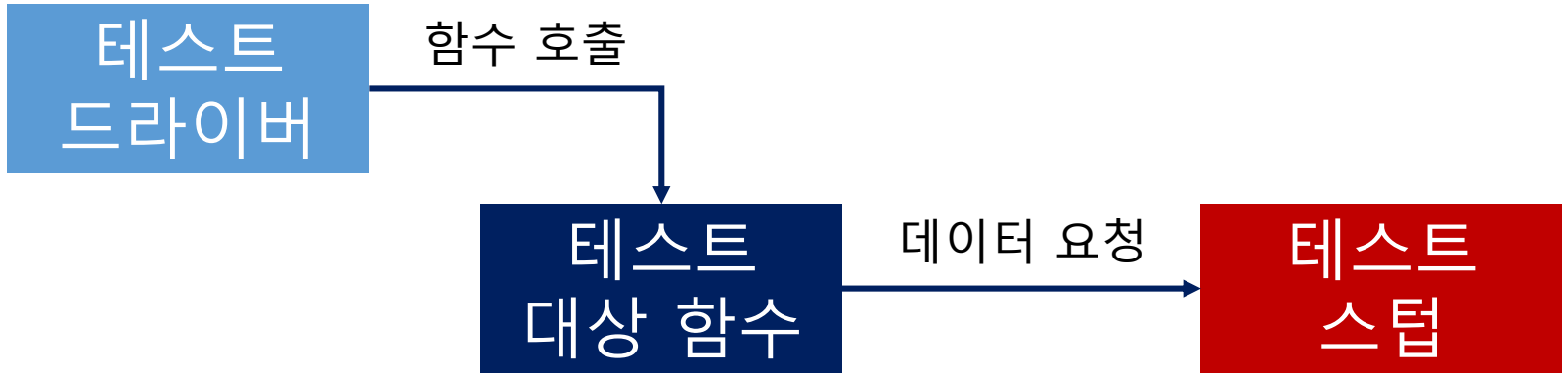
V-Model



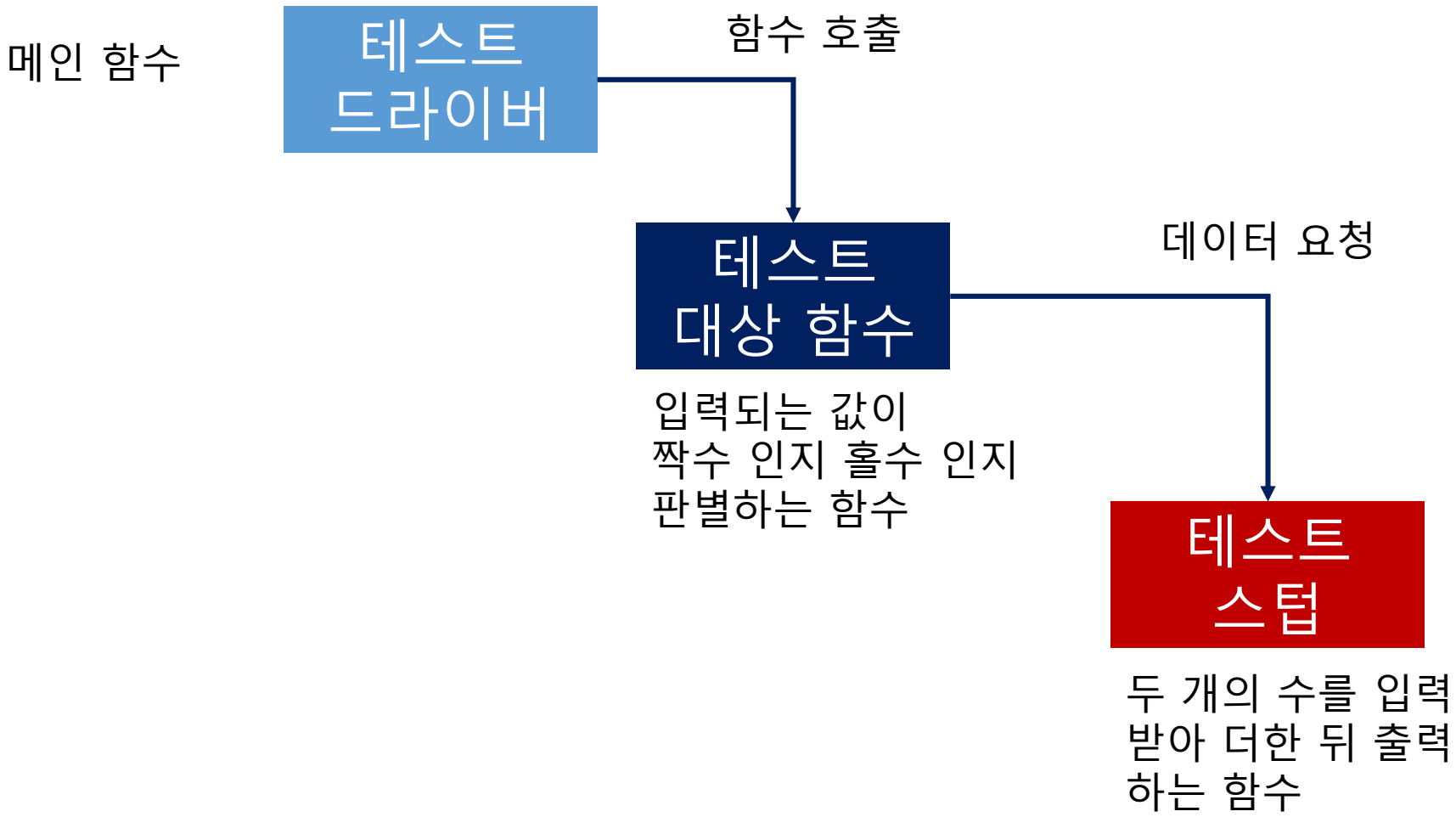
06. 테스팅 종류

유닛 테스트

- 테스트의 최소 단위(일반적으로 함수)를 테스트하는 기법
- 구현 단계에서 개발자가 작성한 함수를 테스트하기 위해 사용
- 유닛 테스트를 수행하기 위해 테스트 드라이버와 테스트 스텁이 필요함
  - 테스트 드라이버(Test Driver) : 테스트 대상 함수를 호출하는 개체
  - 테스트 스텁(Test Stub) : 호출되는 함수가 개발이 완료되지 않았을 경우, 테스트를 위해 작성하는 더미 함수



06. 테스팅 종류


메인 함수
테스트 드라이버
함수 호출
테스트 대상 함수
입력되는 값이 짝수 인지 홀수 인지 판별하는 함수
데이터 요청
테스트 스텁
두 개의 수를 입력 받아 더한 뒤 출력 하는 함수

06. 테스팅 종류

통합 테스트

- 개발된 함수나 모듈들을 통합하여 수행하는 테스트
- 통합되는 범위에 따라 다양한 테스트가 수행됨
- 각 모듈이나 컴포넌트들 사이의 상호작용을 테스트함
- 소스코드 단위에서 수행될 수도 있고 프로그램 단위로 수행될 수도 있음

Example)
스마트 폰의 카메라 모듈이 완성하여 테스트를 수행
카메라 모듈을 스마트 폰에 결합하여 카메라가 동작하는지 테스트

06. 테스팅 종류

통합 방법

	백본 (Backbone)	빅뱅 (Big bang)	상향식 (Bottom up)	하향식 (Top down)
수행 방법	가장 중요하고 리스크가 높은 모듈로 초기 통합 형성	모드 테스트 모듈을 동시에 통합	가장 하부의 모듈부터 통합해 가면서	가장 상부의 모듈부터 통합해 가면서
드라이버/스텁	드라이버/스텁을 필요에 따라 만들어 사용	드라이버/스텁 없이 실제 모듈로 테스트	테스트 드라이버가 필요하며 점차 개발되고 테스트된 상부 모듈로 대치	테스트 스텁이 필요하며 점차 개발되고 테스트된 하부 모듈로 대치
장점	결함 격리 쉬움 리스크가 높은 결함을 초기에 발견	단시간 테스트	결함 격리 쉬움 하위 모듈을 충분히 테스트	결함 격리 쉬움 설계상의 결함을 빨리 발견
단점	테스트 시간이 오래 걸릴 수 있음	결함 격리 어려움	수정이 어려운 중요한 결함을 상부구조에서 발견 가능 비즈니스 로직 반영 어려움	수정이 어려운 중요 결함을 하부에서 발견 가능 EX)디자인 결함을 가진 DB

06. 테스팅 종류

시스템 테스트

- 시스템 테스트는 전체 시스템 또는 제품의 동작에 대해 수행하는 테스트
- 가능한 실제 사용 환경과 유사한 환경에서 수행해야 함
- 요구사항에 있는 기능적, 비기능적 사항들이 모두 반영되었는지 확인

06. 테스팅 종류

회귀 테스트

- 소스코드에 변경이 발생했을 때 이전에 수행했던 모든 테스트를 다시 수행하는 테스트 기법
- 소스코드를 변경하면서 새로운 오류가 소스코드에 포함되었는지 확인
- 테스트 케이스가 중복해서 작성되지 않도록 하는 효과도 있음
- 테스트 비용이 많이 요구됨
- 테스트 비용이 많이 요구됨에도 이 테스트를 수행하는 이유는 소스코드의 변경에 대한 분석 비용이 매우 많이 들기 때문

06. 테스팅 종류

스모크 테스트

- 본격적인 테스트의 수행에 앞서, 대상이 테스트를 수행할만한 상태인지 확인
- 쉽게 말하면 테스트 대상이 구동이 되는지 확인하는 테스트로 테스트 케이스가 필요없음

Q&A