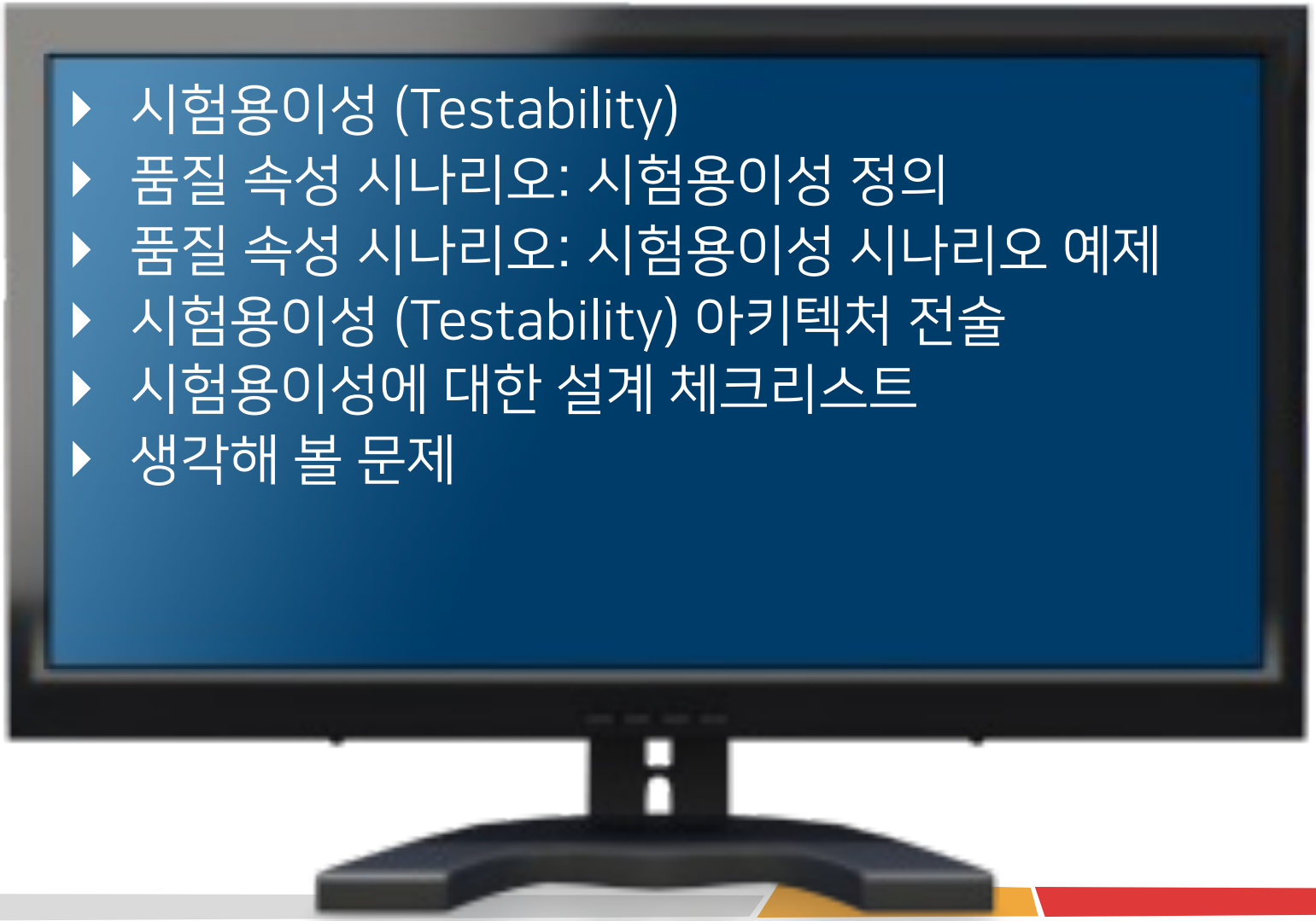
Spring 2023

Testability Tactics

Seonah Lee

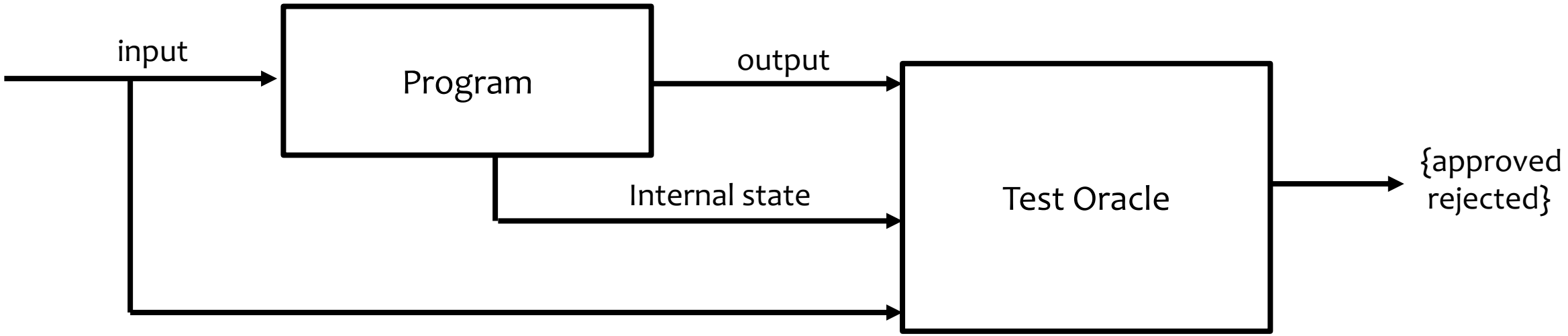
Gyeongsang National University

시험용이성 (Testability) 아키텍처 전술



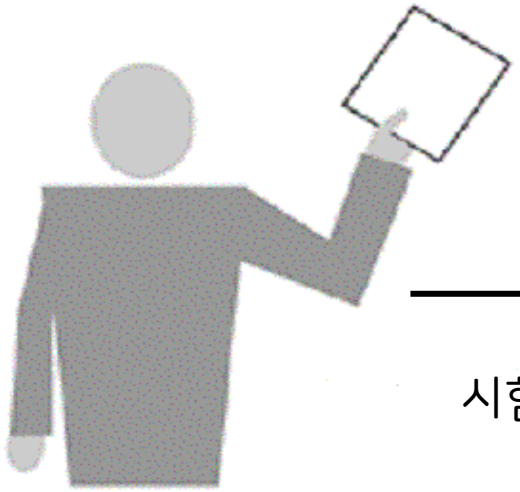
Testability

- 시험용이성 **(Testability)**
  - 잘 구성된 시스템의 **30~50%**의 비용이 테스팅에 소모
  - 테스팅을 통해 시스템이 얼마나 오류를 쉽게 찾아낼 수 있는지에 대한 시스템 능력



Quality Attribute Scenario for Testability

- 시험용이성: 테스팅을 통해 시스템이 얼마나 오류를 쉽게 찾아낼 수 있는지에 대한 시스템 능력



자극의 근원:
시험자

자극:
시험 단위 완료

대상(Artifact):
시험 대상

환경:
시험 수행 시점

응답:
시험 결과

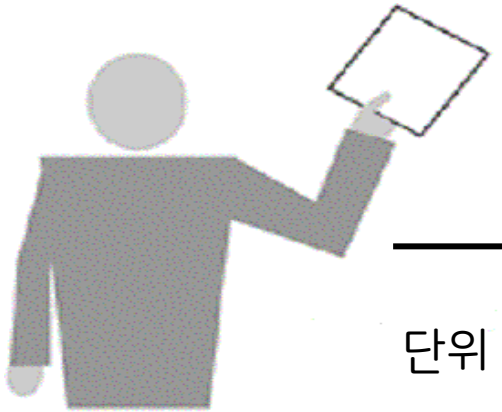
응답 측정:
시험 결과 척도

Quality Attribute Scenario for Testability

Component	Description
자극의 근원 (Source)	시험을 수행하는 사람(혹은 시스템) • 단위시험, 통합시험, 시스템시험, 클라이언트
자극 (Stimulus)	개발 프로세스의 마일스톤 • 분석이나 설계의 완료 • 코딩의 완료, 서브시스템 통합, 전체 시스템의 완료
환경 (Environment)	시험이 수행되는 시점 • 설계 시점, 개발 시점, 컴파일 시점, 배치 시점
대상 (Artifact)	시험의 대상 • 설계, 코드, 전체 시스템 등
응답 (Response)	시험을 수행하기 위해 시스템을 제어하고 시험을 관찰 가능할 수 있게 함
응답 측정 (Response Measure)	• 시험 가능한 명령문의 비율 • 가장 긴 시험 체인의 길이 • 추가적인 결함을 발견하게 될 확률

Quality Attribute Scenario Example for Testability

- 시험용이성 **(Testability)**의 시나리오 예
  - 단위 테스터는 컴포넌트에 대한 단위 시험을 수행한다. 이 때 전체 경로의 **85%**는 **3**시간 내로 시험할 수 있다.



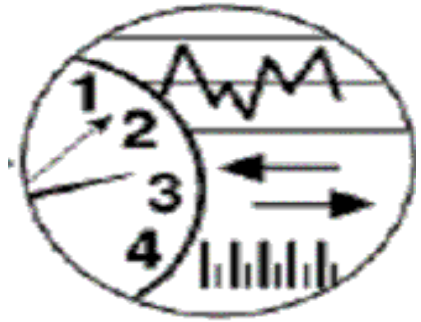
자극의 근원 :
단위 테스터

자극:
단위 테스트 수행

대상(Artifact) :
시스템의
컴포넌트

환경:
개발 중
컴포넌트 완료

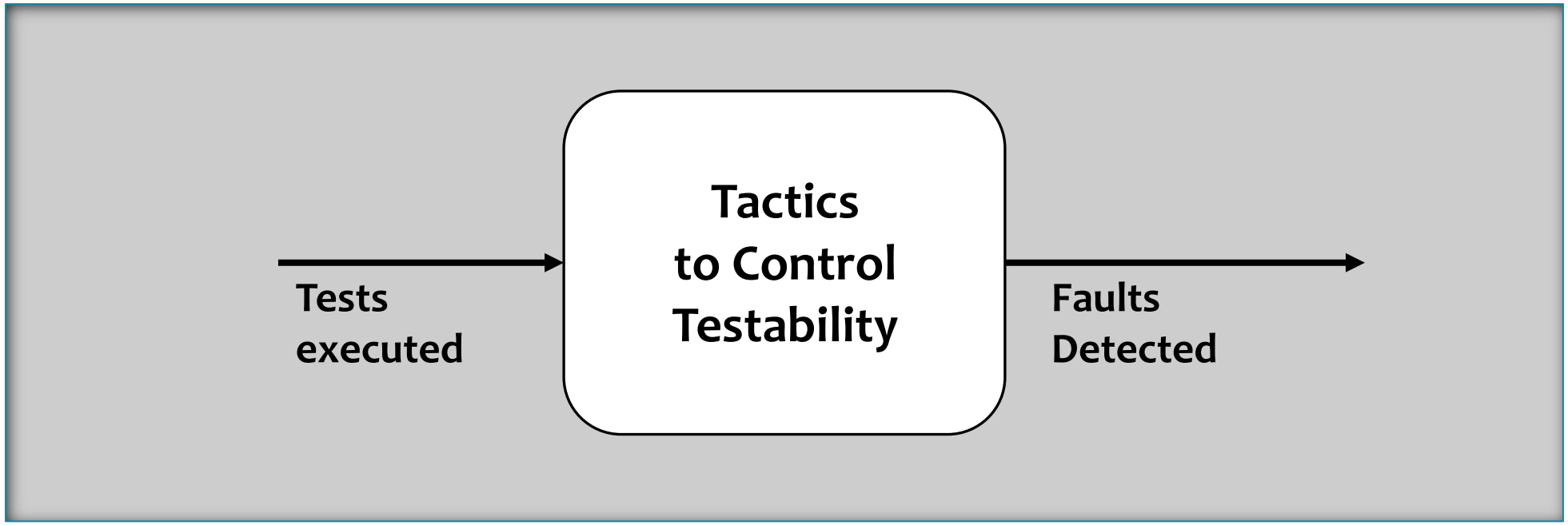
응답:
컴포넌트가 행위를
통제하기 위한
인터페이스를
가지고 컴포넌트의
출력은 관찰됨



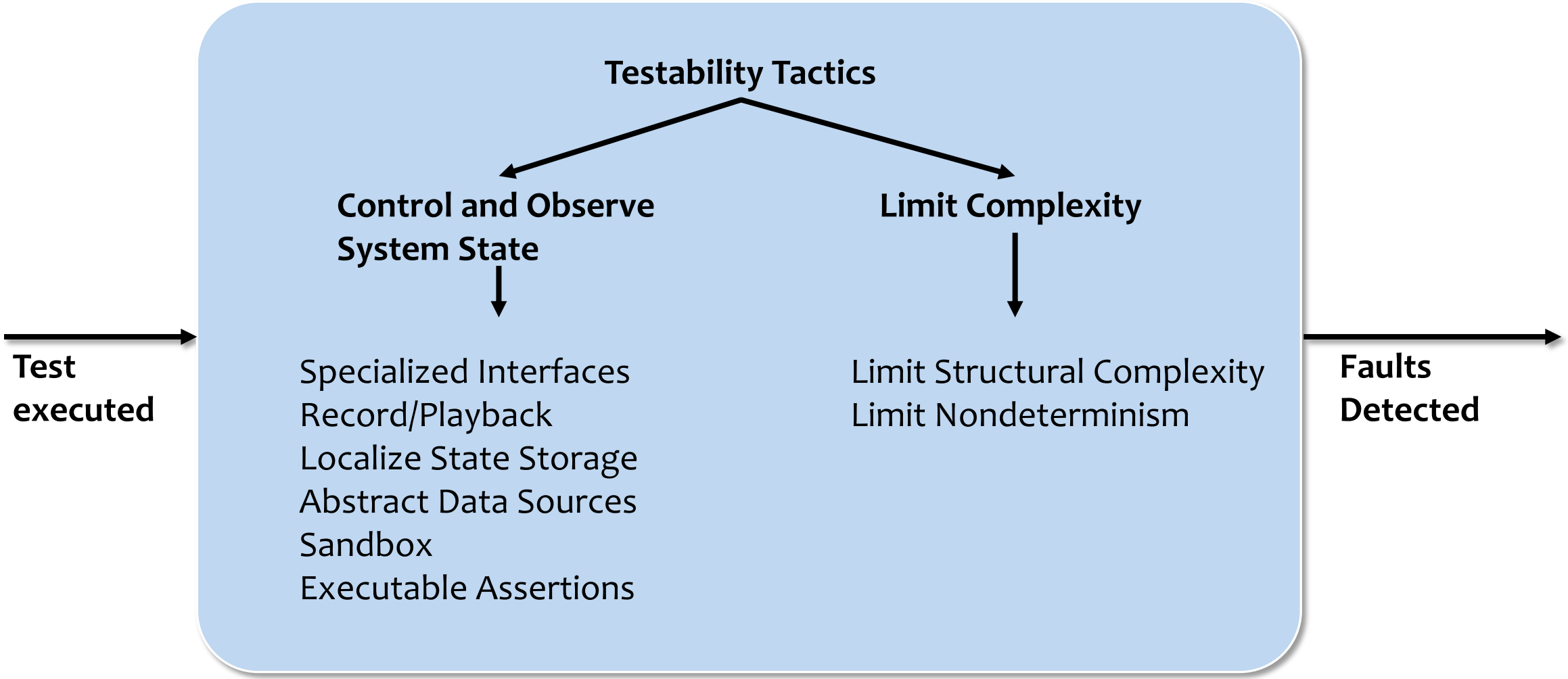
응답 측정:
전체 경로의 85%의 시험이
3시간 이내 달성

Testability Tactics

- 시스템 컴포넌트들을 개발하여 통합하는 과정에서 시험을 용이하게 하기 위한 전술



Testability Tactics



Testability Tactics

- 시스템 상태 제어 및 관찰의 이유
  - 소프트웨어 컴포넌트에 입력을 넣고 출력을 관찰
  - 소프트웨어 컴포넌트의 상태에 대해 시험자가 값을 할당하고, 필요 시 값을 가져와 볼 수 있도록 함
- 시스템 상태 제어 및 관찰 **(Control and Observe System State)**
  - **특화된 인터페이스 (Specialized Interfaces)**
    - 컴포넌트를 위한 변수 값을 제어하거나 볼 수 있도록 하는 특화된 테스팅 인터페이스 제공
      - ***report*** **method:** 객체의 전체 상태를 리턴
      - ***reset*** **method:** 객체의 모든 내부 상태를 명시한 상태로 설정
      - 기타, 이벤트 기록, 성능을 위한 **instrumentation,** 리소스 모니터링 등을 위한 메소드

Testability Tactics

- 시스템 상태 제어 및 관찰 (**Control and Observe System State**)
  - **기록/재생 (Record/Playback)**
    - 오류로 발생시킨 상태를 재생성하기 힘듦
    - 인터페이스를 옮길 때마다 상태를 기록
      - 시스템 행위를 재생하는 것이 가능, 오류 재생 가능
    - 기록: 인터페이스를 통과할 때 정보 기록
    - 재생: 향후 테스팅을 위해 기록된 정보를 입력으로 사용
  - **상태 저장소의 지역화 (Localize state storage)**
    - 시스템, 서브시스템, 모듈을 시험을 위해 임의의 상태에서 시작할 때, 상태를 하나의 장소에 저장한다면 편리함
    - 저장 장소를 표면화, 명시화하는 것은 현재의 상태를 추적하고 기록하기 위한 상태 머신(**State Machine**)을 활용하는 것임

Testability Tactics

- 시스템 상태 제어 및 관찰 (Control and Observe System State)
  - 데이타 소스의 추상화 (Abstract data sources)
    - 입력 데이터를 쉽게 제어하는 것도 시험 용이성을 도움.
    - 인터페이스의 추상화는 테스트 데이터 대체를 도움
      - 예: 고객 트랜잭션의 데이터베이스를 다른 테스트 데이터베이스로 대체
  - 샌드 박스 (Sandbox)
    - 시스템의 인스턴스를 실제 환경에서 고립시켜 실험 가능
    - 시험은 영구적인 결과를 내지 않고 시스템을 운영할 수 있어야 함
  - 실행 가능한 결정문 (Executable Assertions)
    - 프로그램이 오류 상태가 될 수도 있는 위치에 **Assertion** 문장 삽입하여 확인

Testability Tactics

- 복잡성 제약 (Limit Complexity)
  - **구조적 복잡성 제약 (Limit structural complexity)**
    - 컴포넌트 간의 순환적인 의존을 피하거나 해결하기 위함
    - 외부 환경의 의존성에 대해 고립하거나 캡슐화하고자 함
    - 객체지향 시스템에서 상속 계층을 단순화
      - 자식 클래스의 개수를 제약
      - 상속의 높이를 제약
      - 다형성과 동적 호출 제약
  - **비결정성 제약 (Limit nondeterminism)**
    - 비결정성의 소스를 찾음 (제약되지 않은 병행성을 찾아 제거함)

Questions

Q1.테스트할 수 있는 시스템은 쉽게 결함을 찾는 것이다.

- 반대로 결함 감내는 결함이 있더라도 동작해야 한다.
- 즉, 시스템이 결함을 드러내기 어렵도록 만든다.
- 높은 테스트 용이성과 높은 결함 감내를 동시에 모두 갖는 시스템을 설계하는 것이 가능한가? 또는 이들 두 설계 목표가 본질적으로 호환될 수 없는 것인가?

Q2.테스트 용이성과 가장 충돌한다고 생각하는 다른 품질 속성은 무엇인가?

- 테스트 가능성과 가장 호환된다고 생각하는 다른 품질 속성은 무엇인가?

Question?

Seonah Lee

saleese@gmail.com