학습자료

임베디드 SW 엔지니어링

학습모듈

임베디드 시스템 테스팅
(분류번호 2001020309)

17. 임베디드 시스템 테스트

한국산업인력공단
HUMAN RESOURCES DEVELOPMENT SERVICE OF KOREA

17. 임베디드 시스템 테스트

NCS 기본 정보

- 세분류 : 시스템 엔지니어링
- 능력단위 : 임베디드 시스템 테스팅
- 학습모듈 : 임베디드 시스템 테스팅
- 분류번호 : 2001020309
- 능력단위요소 : 임베디드 시스템 단위 테스트하기
  임베디드 시스템 통합 테스트하기
  시스템 테스트 지원하기

학습목표

1. 바이너리 이미지를 활용하여 테스트 프로그램을 수행하고, 각 테스트 케이스가 정상적으로 동작하는지 확인하고 문서화 할 수 있다.
2. 제품의 요구사항에 따라 테스트 설계를 통한 요구사항의 모든 항목을 정상적으로 수행하는지 확인하고 검증 할 수 있다.
3. 제품의 요구사항에 따라 테스트 설계를 통한 요구사항의 모든 항목을 정상적으로 수행하는지 확인하고 검증 할 수 있다.

학습내용

- 임베디드 시스템 단위 테스트하기
- 임베디드 시스템 통합 테스트하기

1. 임베디드 시스템 단위 테스트하기

1 임베디드 시스템 단위 테스트 수행

1. 임베디드 시스템 단위 테스트

(1) 목적

- 임베디드 시스템의 단위 모듈들이 설계 명세서의 요건을 만족할 수 있도록 확인하는 것
- 각각의 단위 모듈이 정의되고 그 하나의 단위 모듈이 제품을 구동하기 위한 최소 기능을 수행하는 소스코드로써 정확성 및 적합성이 검증되어야 함

(2) 단위 테스트의 특징

- 일반적으로 개발자가 개발 환경에서 수행을 하게 되며 개별적인 단위 모듈 구동 시 문제가 없도록 결함을 제거하는 행위
- 단위 테스트의 대상인 단위의 기준은 산업 도메인 특성을 고려한 조직 및 대상 프로젝트에 따라 다를 수 있으며 일반적으로는 모듈(module), 함수(fuction), 프로그램(program), 컴포넌트(component), 오브젝트(object), 클래스(class) 등으로 항목을 정의할 수 있음
- 단위 테스트 대상 항목을 독립적으로 실행시키기 위하여 테스트 스텁(test stub) 및 테스트 드라이버(test driver)를 구성하여 모의 호출 관계를 검증함
- 프로젝트 테스트 계획서, 단위 테스트 계획서, 요구 사항 명세서, 테스트 대상 코드 단위 명세서, 상세 설계서, 프로그램 라이브러리 등을 이용하여 단위 테스트 케이스를 설계하여 개발 초기시점에 결함(defect을 많이 발견하여 개발 기간 중 디버깅 시간을 단축시킬 수 있도록 함
- 단위 테스트 수행 시 수작업(manual) 방식과 자동화(automatic) 방식이 존재
- 인베디드 소프트웨어 산업계의 품질 안정성에 대한 Code Coverage를 만족시키기 위해서는 소프트웨어 테스트 자동화 방식이 필요함

(3) 다중 V모델

- 개발 단계와 대응되는 단위 즉, 컴포넌트 테스트, 통합테스트, 시스템테스트, 인수 테스트를 진행하는 것

(4) 정적 테스트(static test)

- 정의
  - 소프트웨어 실행 없이 소스코드, 설계 문서, 요구 사항 정의서를 작업(manual) 및 자동화(automatic) Tool을 사용하여 Test하는 방법
- 목적
  - 소프트웨어 실행 전에 코딩 표준, 코드 메트릭을 통해 표준 준수 여부를 체크하고 문서 목록에 산출물이 제대로 작성되었는지 확인하여 잠재된 결함(defect)을 식별

- 기법

  (가) 리뷰(review)

  - 개발자가 직접 소스코드, 요구 사항, 설계 문서, 표준 프로세스 문서, 개발 산출물을 보면서 소스코드의 품질을 체크, 확인하는 기법이며 개발 비용대비 효과가 큰 방식
  - 리뷰의 종류에는 동료 기술 검토(peer review), 워크스루(walk-throuth), 코드 인스펙션(code inspection)등이 있으며 진행 형식과 목적에 따라 구분함

  (나) 정적 분석(static analysis)

  - 자동화 툴 기반의 소스코드 분석 기법이며 소스코드의 복잡도, 데드 코드(deadcode) 코딩 룰 에러(변수명 규칙, 함수명 규칙 등), 소스코드 구조(Data Flow, Control Flow) 및 의존 관계를 분석하고 검출하는 방식
  - 툴 기반의 분석 방식이므로 적용 Rule Setting과 요구 수준에 대한 기준이 명확해 야 한다. 오픈소스(find bug, PMD 등)와 상용 툴(McCabe IQ, QAC, Klockwork 등)이 존재

(5) 동적 테스트(dynamic test)

- 정의
  - 소프트웨어 프로그램이나 시스템 실행을 통하여 무엇이 어떻게 수행되고 동작되는지 Test하는 방법
- 목적
  - 개발 명세서를 기반으로 소프트웨어 실행 결과와 기능이 잘 작동하는지, 프로그램이나 시스템 내부구조의 동작상의 결함(defect을 명확히 식별
- 기법

  (가) 블랙박스 테스트(black box test)

  - 테스트 설계 기법 분류 중 명세 기반의 기법으로서 소스코드 실행만을 통해 테스트를 수행하는 방법

  (나) 화이트박스 테스트(white box test)

  - 테스트 설계 기법 분류 중, 구조 기반의 기법으로써 소스코드를 실행하여 모듈 내 의 동작이 어떻게 되는지 테스트를 수행하며, 기능 및 동작상의 결함을 식별하는 기법

1. 임베디드 시스템 단위 테스트하기

2 단위 테스트 수행하기

1. 재료·자료

(1) (단위) 테스트의 대상이 되는 시스템

(2) 요구 사항 명세서, 상위/상세 설계서

2. 기기(장비·공구)

(1) 화이트보드(칠판) 및 보드마커

(2) 전지(종이) 및 포스트 잇

(3) 컴퓨터, 프린터

3. 안전·유의 사항

(1) 실습 시 지도 교수의 지시를 따름

(2) 실습 중 불필요한 행동 금지

(3) 실습 후 컴퓨터 전원 끔

4. 수행 순서

(1) 테스트 프로세스의 이해

- 요구분석
- 테스트 계획
- 테스트 설계
- 테스트 수행
- 테스트 평가

(2) 테스트 프로세스 수행

- 테스트 계획 수립
- 테스트 설계
- 테스트 수행
- 테스트 평가

2. 임베디드 시스템 통합 테스트하기

1 임베디드 시스템 통합 테스트 수행

1. 시스템 통합 테스트 개요

(1) 통합 테스트(integration testing)

- 시스템 통합은 하드웨어와 소프트웨어를 모두 포함함
- 하드웨어와 하드웨어, 소프트웨어와 소프트웨어, 하드웨어와 소프트웨어 종속적 관계를 식별하고 통합 영역의 가용성, 시스템의 규모, 기존 시스템 및 신규 시스템 개발구조를 식별함
- 상향식 통합 방식, 하향식 통합 방식, 빅뱅 통합 방식에 따라 시스템 통합을 수행할 수 있음

  (가) 상향식 통합 방식

  - 드라이버를 사용하여 연관성이 적은 최하위 부분부터 점진적으로 모듈을 통합하여 테스트하는 방법이며, 서브 시스템을 병렬적으로 개발하거나 단계적으로 개발한 후 통합하는 방식을 사용할 경우 유용
  - 장점 : 개발 초기에 통합이 가능하여 테스트가 용이하며, 인터페이스 문제의 조기 발견이 가능하기 때문에 개발 완료 후 발견된 것보다 수정 비용이 적게 듬
  - 단점 : 드라이버 개발이 필요하며 반복적인 테스트가 필요 시에는 시간 소요가 많이 듬

  (나) 하향식 통합 방식

  - 전체적인 상위 레벨의 시스템 주요 제어 구조에서부터 순차적인 하향 방식으로 상위 논리 및 데이터 흐름을 개발 프로세스 초기에 테스트하여 통합을 시켜 나가는 방식으로 아직 개발이 되지 않은 모듈은 테스트가 가능하도록 스텁을 만들어서 대체해 나가는 방식
  - 장점 : 주요 기능에 대한 초기 테스트가 가능
  - 단점 : 통합 순서 및 영역 범위가 명확해야 하고 요구 사항이 변경될 때마다 하위 모듈에 영향을 끼칠 수 있기 때문에 테스트 재수행이 빈번하게 일어날 가능성이 높으며 시스템 개발 후반부에는 요구 변경에 따른 고려 사항이 매우 많아질 수 있음

  (다) 빅뱅식 통합 방식

  - 모듈 간 결함도가 높고 단계적 통합이 어려운 경우와 기존의 안정된 시스템에 새로운 모듈을 추가할 경우 모듈 간의 인터페이스를 고려하지 않은 상태에서 결합시켜 테스트하는 방식
  - 장점 : 소규모 시스템 및 매우 단순한 프로그램의 인터페이스와 같이 복잡하지 않은 경우 효과가 있으며 드라이버와 스텁이 필요하지 않음

- 단점 : 규모가 큰 시스템에서는 결함 원인을 처음부터 찾아 나아가야 하는 시간이 많이 필요하며 결함 원인 또한 발견이 쉽지 않다. 또 테스트에 필요한 모든 모듈이 준비되어야 가능한 통합 방식

2. 시스템 테스트

(1) 품질 특성 기반의 테스트 설계

- ISO/IEC 25010 품질 특성을 기반으로 체크리스트를 활용해 테스트 케이스를 도출하는 방법
- 사용성, 유지 보수성, 신뢰성, 효율성, 이식성, 상호 운용성, 보안성 등을 검증

(가) 사용성

- 명세된 조건하에서 사용자가 이해하고 배우기 쉽고 용이하게 사용 가능한 시스템 기능을 말하며 UI(User Interface), 기능 실행 절차, 메시지 또는 표시가 적절한지를 확인하는 품질 특성

(나) 유지 보수성

- 환경이나 요구 사항의 변화에 맞춰 시스템을 수정, 개선하는 능력을 말하는 품질 특성

(다) 신뢰성

- 사용자가 시스템을 사용하면서 특별한 상황에서 일정 이상의 성능을 유지하는 시스템의 능력의 품질 특성

(라) 효율성

- 사용자가 원하는 기대치를 시스템의 자원 낭비 없이 달성하는 능력

(마) 이식성

- 어떤 환경에서 다른 환경으로 변환하여 적용할 수 있는 능력

(바) 보안성

- 인증된 사용자만이 기능에 접속할 수 있게 하는 시스템의 능력

2 시스템 테스트 지원 환경 구성

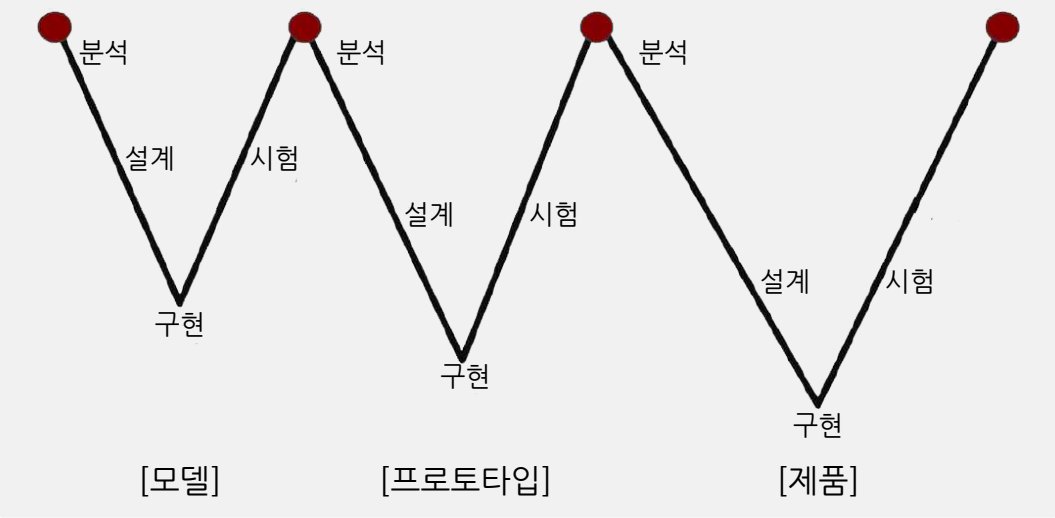
1. 임베디드 시스템 테스트 지원 환경

(1) 임베디드 시스템 테스트 프로세스 이해

- 임베디드 시스템 개발 단계에서 다중 V-모델의 3가지 단계(시뮬레이션 단계, 프로토타이핑 단계, 시제품 단계)를 통해 점진적, 구체화됨으로써 뮬레이션 시스템에서 제품 시스템으로 진화하는 과정의 테스트 프로세스를 말함

2. 임베디드 시스템 통합 테스트하기

- 테스트 인프라는 테스트 프로세스를 지원할 수 있도록 구성되어야 하는데 개발 프로세스상에서 이전 단계의 테스트 결과를 도출하고 검토, 분석하여 현재의 결과와 비교할 수 있는 형태가 만들어져야 함
- 모든 테스트 데이터를 데이터베이스에 저장하고 관리할 수 있도록 하며, 아래와 같은 내용이 포함 될 수 있도록 함
  - 테스트 날짜, 시간
  - 테스트 수행 횟수
  - 테이터 기록 횟수
  - 테스트 계획 변경 식별
  - 테스트 결과 보고서 식별



[그림] 멀티V모델

- 일반적으로 임베디드 테스트는 [그림 3-2]와 같이 테스팅 프로세스로 구분될 수 있으며, 위에서 언급한 개발 단계와 연계하여 표현할 수 있음

3 시스템 테스트 지원 환경 구성하기

1. 수행 순서

(1) 테스트 프로세스를 이해한다.

- 시뮬레이션 단계

  (가) 목적

  - 설계 개념을 증명
  - 최적화된 설계를 구성

(나) 구성 및 유형

- 단방향 시뮬레이션(one way simulation)
- 피드백 시뮬레이션(feedback simulation)
- 래피드 프로토타이핑

• 프로토타이핑 단계

(가) 목적

- 시뮬레이션 모델의 유효성을 검증
- 시스템이 요구 사항을 만족하는지 검증
- 생산 이전 단위 배포

(나) 구성 및 유형

- 임베디드 시스템 동작
- 프로세서
- 임베디드 시스템 나머지 부분
- 플랜트(임베디드 시스템의 환경)

• 시제품 단계

(가) 목적

- 최종 모든 요구 사항이 만족되었는지 검증
- 환경 표준, 산업 표준, 국제 표준(ISO), 정부 및 군용 표준과 규격 기준 준수
- 정해진 예산, 기간과 제품 개발 환경 내에서 임베디드 시스템이 구축되었는지 입증
- 임베디드 시스템이 실제 환경에서 유지되고 평균 수리 시간 내 요구 사항을 만족하는지 입증
- 고객(잠재)에게 제품을 입증
- 시제품 단계에서 활용할 수 있는 테스트 타입의 종류

정리하기

1. 임베디드 시스템 단위 테스트하기

Multiple V-Model

- 모델, 프로토타입, 최종제품 개발로 제품의 진화하는 단계마다 V 모델을 활동 전체로 수행하는 것

프로그램(program)

- 파일 형태로 저장되어 있으며, 소스파일과 실행파일
- 사용자가 컴퓨터에 작업을 시키기 위한 명령어의 집합(명령어 코드)을 담고 있음

2. 임베디드 시스템 통합 테스트하기

시스템 통합 테스트

- 개발된 단위 모듈을 결합하여 하나의 완성된 프로그램/시스템으로 구성하는 과정에서 수행되는 테스트

시스템 테스트

- 통합 테스트가 완료된 후, 완전한 시스템에 대해 시스템 명세서에 따라 개발되었는지 검증하기 위해 수행