소프트웨어공학 활용

소프트웨어 정적분석과 동적분석

한국기술교육대학교
온라인평생교육원

학습내용

- 소프트웨어 정적분석
- 소프트웨어 동적분석

학습목표

- 소프트웨어 정적분석에 대한 개념을 이해하고 적용할 수 있다.
- 소프트웨어 동적분석에 대한 개념을 이해하고 적용할 수 있다.

소프트웨어 정적분석

1 정적 프로그램 분석

1 정적 프로그램 분석이란

1 개념

정적 프로그램 분석

- 정적 프로그램 분석(Static Program Analysis)은 프로그램의 실행 없이 분석
- 코드의 의미를 분석해 버그를 찾는 방법
- 명세서나 코드만 보고 테스트 수행

2 종류

정적 화이트박스 테스트 (Static WhiteBox Test)	정적 블랙박스 테스트 (Static BlackBox Test)
소스코드만 보고 실행 없이 테스트하는 방법	소스코드와 실행 없이 테스트하는 방법

2 정적 화이트박스 테스트(Static WhiteBox Test)

1 개념

정적 화이트박스 테스트란?

프로그램이 실행되지 않은 상태에서도 대상 소프트웨어의 설계, 아키텍처, 코드 등에서 상세하게 버그를 찾을 수 있는 방법

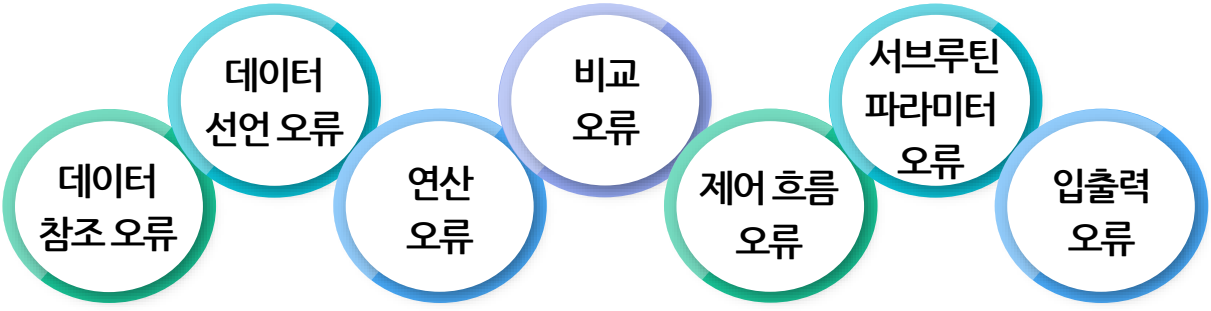
소프트웨어 정적분석

1 정적 프로그램 분석

2 정적 화이트박스 테스트(Static WhiteBox Test)

1 개념

테스트를 위하여 코드 검토 체크리스트 작성



2 동료 검토(Peer Reviews)

동료에게 원시 코드, 여러 가지 산출물의 검토를 의뢰하여 오류를 찾는 방법

정해진 형식도 없고, 격식을 차린 회의를 수행할 필요가 없는 비공식적 검토

3 워크스루(Walk Throughs)

개발자가 소집한 전문가들에 의해 개발자의 작업을 검토

3~5명 정도의 전문가들이 절차에 따라 평가

설계문서가 고객 요구사항에 대한 정확한 명시 여부, 작업 진척 상황 등 확인

소프트웨어 정적분석

1 정적 프로그램 분석

2 정적 화이트박스 테스트(Static WhiteBox Test)

4 검사(Inspections)

검토회의	문제점을 찾는데 초점, 검토회의 후 개발자가 문제 수정
소프트웨어 검사	문제점 수정지침을 제시하고 수정을 잘하고 있는지 추후에 조사

원시 코드, 각 단계 산출물의 문서 등을 포함하여 분석하고 품질을 평가

소프트웨어 품질보증기법으로 유용하며 공식 검토에 속함

3 정적 블랙박스 테스트(Static BlackBox Test)

정적 블랙박스 테스트란?

- 소스코드를 파악할 수 없고 실행시키지도 않는 검사
- 명세서 테스트가 이에 해당함
- 제품 개발의 방향을 잡아주는 개발 명세서를 활용하여 판단

→ 완결성, 정확성, 정밀성, 일관성, 연관성, 실행 가능성, 코드와의 무관성, 테스트 가능성

소프트웨어 정적분석

1 정적 프로그램 분석

4 정적분석 도구

1 도구에 의한 정적분석

도구에 의한 정적분석의 목적은 소프트웨어의 소스 코드와 모델에서 결함을 발견

대상 소프트웨어를 실제로 실행하지 않는 상태에서 도구의 지원으로 수행

2 특징

1 장애보다 결함을 발견

2 프로그램 코드를 분석 및 HTML, XML과 같이 생성된 결과물도 분석

개발자

단위(컴포넌트) 테스팅과 통합 테스팅 동안에 주로 개발자에 의해 사용

설계자

소프트웨어 모델링을 하는 동안에는 설계자에 의해 사용

소프트웨어 정적분석

1 정적 프로그램 분석

4 정적분석 도구

4 정적분석 툴로 발견하는 일반적인 결함

정의되지 않은 값으로 변수 참조	모듈과 컴포넌트 사이의 일관성 없는 인터페이스	사용하지 않거나 잘못 선언된 변수
누락되거나 오류가 있는 로직	지나치게 복잡한 구조	프로그래밍 표준 위반
보안 취약점	코드나 소프트웨어 모델의 구문 위반	

소프트웨어 정적분석

2 정적 프로그램 분석 도구

1 Findbugs

1 라이선스와 목적

Findbugs 라이선스

- Lesser GNU Public License 사용
- 이클립스 플러그인으로 툴 제공

목적

잠재적 버그 찾기(소스 파일이 아닌 바이트 코드를 이용)

2 장점·단점

장점

- 실제 결함을 찾음
- 찾은 결함이 엉뚱한 결함일 확률이 낮음 (정확성이 높음)
- 바이트 코드를 읽으므로 속도가 빠름

단점

- 컴파일된 클래스 파일에서 바이트 코드를 읽어서 사용해야 하므로 빌드과정이 필수

소프트웨어 정적분석

2 정적 프로그램 분석 도구

1 Findbugs

3 규칙 수와 규칙 카테고리

규칙 수	• 408
규칙 카테고리	• Correctness, Bad Practice, Dodgy Code, Multithreaded Correctness, Performance Malicious, Code Vulnerability, Security Experimental, Internationalization

2 PMD

1 라이선스와 목적

PMD 라이선스	• BSC-style License • 이클립스 플러그인으로 툴 제공
목적	잠재적인 문제들, 버그 가능성이 있는 부분들, 사용되지 않았거나 최적화되지 않은 코드를 검색

소프트웨어 정적분석

2 정적 프로그램 분석 도구

2 PMD

2 장점·단점

장점

- 실제 결함을 찾아줌
- Find Bad Practices

단점

- 복제된 코드를 찾는 속도가 느림

3 규칙 수와 규칙 카테고리

규칙 수	• 234
규칙 카테고리	• JSP : Basic JSF, Basic JSP • XSL : XPath in XSL • Ecmascript : Basic Ecmascript, Unnecessay, Braces • XML : Basic XML • Java : Design, Coupling, Jakarta Commons Logging, Basic, Strict Exceptions, Security Code Guidelines, Java Logging, Android-Controversial, Comments, Type Resolution, Empty Code, String and StringBuffer, Code Size, Braces, Unused Code, Unnecessary, J2EE, JavaBeans, Migrations, Import Statements, JUnit, Naming, Finalizer, Optimization, Clone Implementation

소프트웨어 정적분석

2 정적 프로그램 분석 도구

3 Checkstyle

1 라이선스와 목적

Checkstyle 라이선스

- Lesser GNU Public License 사용
- 이클립스 플러그인으로 툴 제공

목적

자바 소스 파일을 읽어서 소스코드 표준에 위반되는 것들을 검색

2 장점·단점

장점

- 정해준 코딩 규약에 위반되는 것들을 검사
- 직접 코딩 규약을 만들어 사용 가능

단점

- 실제 버그를 찾을 수 없음

소프트웨어 정적분석

2 정적 프로그램 분석 도구

3 Checkstyle

3 규칙 카테고리

규칙 수	• 132
규칙 카테고리	• Annotations, Block Checks, Class Design, Coding, Duplicate Code, Headers, Imports, Javadoc Comments, Metrics, Miscellaneous, Modifiers, Naming Conventions, Regexp, Size Violations, Whitespace

4 Multi-language 지원 툴

1 RATS-Rough Auditing Tool for Secuity

1. 보안 소프트웨어 엔지니어들이 만든 툴
2. C, C++, Perl, PHP, Python Source Code를 모두 스캐닝
3. 문제의 소지가 있는 부분과 문제점, 해결책 분석

➡ 분석결과에 따른 에러의 심각도와 우선순위를 책정해서 보여줌

소프트웨어 정적분석

2 정적 프로그램 분석 도구

4 Multi-language 지원 툴

2 Yasica

1 프로그래밍 언어와 같은 형태의 파일을 분석하는 플러그인 형태의 프레임워크

- C, C++, Java, JavaScript, ASP,PHP, HTML/CSS, ColdFusion, COBOL

2 Command-line 툴로 코드에 오류가 발견되면 HTML 형태의 파일로 알려줌

5 C Language 지원 툴 종류

- Sparse
- Splint, Uno
- BLAST
- Cppcheck (C++ 지원 툴)

6 Java Language 지원 툴 종류

- Checkstyle
- Findbugs
- PMD
- Hammurapi

소프트웨어 동적분석

1 동적 프로그램 분석

1 동적 프로그램 분석이란

1 개념

동적 프로그램 분석

- 동적 프로그램 분석(Dynamic Program Analysis)은 프로그램을 실행하여 분석
- 디버깅이 바로 동적 프로그램 분석
- 유닛 테스트도 일종의 동적 프로그램 분석에 속함
- 스크립트 언어의 인기를 기반으로 한 테스트기반개발법(Test Driven Development), 행위기반개발법(Behavior Driven Development)의 성장

2 종류

동적 화이트박스 테스트 (Dynamic WhiteBox Test)

- 소스코드를 가지고 실행하며 테스트하는 방법
- 디버깅, 유닛 테스트 및 스크립트를 통한 자동실행 테스트 등이 속함

동적 블랙박스 테스트 (Dynamic BlackBox Test)

- 소스코드 없이 실행만으로 테스트하는 방법
- 테스트케이스를 만들고 기대하는 결과 값을 산출한 후 해당 테스트케이스에 대해 기대하는 값이 계산되는지 확인

소프트웨어 동적분석

1 동적 프로그램 분석

2 동등분할(Equivalence Partitioning)

1 개요

동등분할 이란?

- 입력, 출력 값 영역을 유한개의 상호 독립적인 집합으로 나누어 수학적인 등가 집합을 만든 후 각 등가 집합의 원소 중 대표 값 하나를 선택해 테스트 케이스를 선정
- 동등분할 클래스는 유효한 입력 데이터와 유효하지 않은 입력 데이터를 포함할 수 있음

같은 특성을 가지면서 같은 방식으로 처리된다고 판단하는 모든 등가 집합에서 대표하는 입력 값들을 적어도 한 개씩은 사용하여 작성

2 동등분할 적용 범위

1. 출력 값
2. 내부 값
3. 통합 개발 환경, 신속 응용 프로그램 개발
4. 통합 테스팅에서 다루는 모듈 간 인터페이스 파라미터 (Interface Parameters)

소프트웨어 동적분석

1 동적 프로그램 분석

3 경계 값 분석(Boundary Value Analysis)

1 개요

경계 값 분석이란?

- 동등분할의 경계 부분에 해당하는 입력 값에서 결함이 발견될 확률이 경험적으로 높기 때문에 결함을 방지하기 위해 경계 값까지 포함하여 테스트
- 경계 값은 해당 분할영역의 최대값과 최소값
- 경계 값을 고려하여 테스트 케이스 설계
- 결함 발견율이 높고, 적용하기 쉬운 장점이 있어 가장 많이 사용되는 테스트 기법

2 한계점

1. 일련의 동작에 대한 조합을 테스트하기에는 적합하지 않음
2. 입력 값 조합의 수가 테스트 가능한 수를 넘어서는 경우가 많음

➡ 입력범위를 동등분할하여 제한하더라도 해당함

3. 입력 조합이 상호 간에 독립적이라는 가정에서만 적합한 기법
4. 입력조건을 동등분할하는 것이 매우 어려울 수 있음

➡ 출력이 입력조건이나 변수들 사이의 관계에 따라 달라지는 경우에도 해당함

소프트웨어 동적분석

1 동적 프로그램 분석

4 결정 테이블 테스팅(Decision Table Testing)

1 개요

결정 테이블 테스팅이란?

- 논리적인 조건이나 상황을 구현하는 시스템 요구사항을 도출하거나 내부 시스템 디자인을 문서화하는 데 매우 유용함
- 시스템이 구현해야 하는 복잡한 비즈니스 규칙을 문서화하는 데 사용
- 입력 조건과 동작은 참과 거짓으로 주로 표현
- 결정 테이블은 동작을 유발하는 조건 또는 상황, 각 해당하는 조합에 대한 예상 결과까지 포함

2 장점

장점

- 논리적·의존적인 모든 조건의 조합을 생성
- 테스트 베이시스의 문제점을 드러내게 하는 효과적인 테스트 케이스 생성이 가능하고, 불완전성과 모호함 지적이 가능
- 테스트 케이스를 만들면서 결함을 발견하는 것이 가능

단점

- 작성에 큰 노력과 시간이 소요될 수 있음
- 복잡한 시스템을 표현하기 어려울 수 있으며, 작성 시 논리적 실수와 소지가 있음

소프트웨어 동적분석

1 동적 프로그램 분석

5 유스 케이스 테스팅(Use Case Testing)

1 개요

유스 케이스 테스팅이란?

- 유스 케이스나 비즈니스 시나리오를 기반으로 테스트를 명세화
- Mainstream scenario, Basic or Main Flows와 Alternative branches or Alternative flows로 구성
- 각각의 유스 케이스는 자세하게 표현하기 위해 유스 케이스 상세를 가짐

유스 케이스	유스 케이스 테스팅
시스템이 실제 사용되는 방식에 기반하여 프로세스 흐름을 기술	통합테스트 단계에서 서로 다른 컴포넌트 사이의 상호 작용과 활동을 테스팅 하는 방법을 생각할 수 있음

2 테스트 케이스 도출 방법

유스 케이스 각각의 테스팅 → 유스 케이스 상세에서 흐름과 시나리오만을 고려하여 테스트 케이스를 도출

유스 케이스 상세를 문장별로 분석하여 **테스트 케이스를 도출**하는 방법

소프트웨어 동적분석

1 동적 프로그램 분석

6 조합 테스팅(Pariwise Testing)

1 개요

커버해야 할 기능적 범위보다 상대적으로
적은 양의 테스트 세트를 구성하여 소프트웨어의 결함을 찾고
테스트에 대한 자신감을 얻을 수 있는 방법

대부분 결함이 2개 요소의 상호작용에 기인한다는 것에 착안,
2개 요소의 모든 조합을 다룸

자원과 시간이 제한된 상황에서 테스트 대상 소프트웨어의
모든 설정, 기능, 이벤트 등의 조합을 테스트하는 것은
일반적으로 불가능

테스트를 하지 않거나, 일부 조합을 의도적으로
누락시키는 것은 그만큼의 리스크를 동반하게 되므로
조합 테스팅은 매우 중요한 의미를 가짐

소프트웨어 동적분석

1 동적 프로그램 분석

7 상태 전이 테스팅 (State Transition Testing)

현재 상황과 이전의 상태를 반영,
그 변화에 따라 동작을 상태 를 전이 다이어그램으로 표현

소프트웨어 상태 사이의 관계(상태 간의 전이, 상태를 변화시키는 이벤트와 입력 값, 상태의 변화로 유발되는 동작)를 파악

상태-이벤트 테이블 구성	주어진 상태 다이어그램의 모든 상태와 이벤트를 테이블로 구성하여 다시 표현
전이 트리 구성	상태가 전이되는 경로를 파악하기 위하여 테이블을 트리 형태로 전환
반응 테스트 케이스와 스크립트 구성	상태가 이벤트에 의하여 변경되는 트리만 케이스로 작성
무반응 테스트 케이스와 스크립트 구성	상태가 이벤트에 의하여 변경되지 않는 트리만 케이스로 작성
가드 또는 조건 테스트 케이스와 스크립트 구성	상태가 변경이 조건에 따라 변경되는 트리만 케이스로 작성

소프트웨어 동적분석

2 동적 프로그램 분석 도구

1 Avalanche

1 라이선스와 설명

라이선스	• GNU GPL v2
설명	• 중요한 소프트웨어 오류를 자동으로 찾음 • 감지된 각 오류에 대해 Input of Death를 각각 생성 • 프로그램에서 Tainted 데이터를 추적 • 커버리지를 높이고 새로운 오류를 찾기 위해 입력 시퀀스를 반복적으로 생성 • Valgrind 프레임 워크와 STP(Simple Theorem Prover) 기반으로 동적분석 구현

2 Valgrind

1 라이선스와 설명

라이선스	• GNU GPL v2
설명	• C,C++ 기반 프로그램에 대한 메모리 및 쓰레드 문제를 동적으로 분석할 수 있음 • 실행 중인 프로그램에서 메모리 누출을 찾아냄

소프트웨어 동적분석

2 동적 프로그램 분석 도구

2 Valgrind

2 오류 발견

- 초기화되지 않은 메모리를 사용하는 경우
- Free된 메모리에 읽기, 쓰기를 시도하는 경우
- Malloc된 메모리 블록 외에 읽기, 쓰기를 시도하는 경우
- Stack의 부적절한 지역에 읽기, 쓰기를 시도하는 경우
- 초기화되지 않거나 주소를 알 수 없는 메모리가 시스템 호출로 넘겨지는 경우 등

소프트웨어 동적분석

2 동적 프로그램 분석 도구

3 기타

BoundsChecker	윈도우 기반 애플리케이션의 메모리 오류 탐지
Cenzic	보안 취약점을 위한 웹 애플리케이션 스키닝을 하는 동적 애플리케이션 보안 툴
ClearSQL	PL/SQL을 위한 리뷰, 품질 관리, 코드 일러스트레이션
Daikon	프로그램을 실행하고 프로그램이 계산한 결과를 관찰하며, 관찰한 실행 동안 참인 값, 즉 전체 실행 기간 동안 참일 것 같은 값을 찾음
VB Watch	비주얼베이직 프로그램에 동적분석 코드를 삽입해서 그것의 성능, 콜 스택, 실행 추적, 초기화 객체들, 변수들 그리고 코드 커버리지를 모니터 함

학습정리

1. 소프트웨어 정적분석

- 프로그램의 실행 없이 분석하고 코드의 의미를 분석해 버그를 찾으며 명세서나 코드만 보고 테스트를 수행하는 방법
- 정적 화이트박스 테스트는 소스코드만 보고 실행 없이 테스트하는 방법이며, 정적 블랙박스 테스트는 소스코드와 실행 없이 테스트하는 방법
- 동료 검토는 동료에게 원시 코드나 여러 가지 산출물에 대한 검토를 의뢰하여 오류를 찾는 방법으로 비공식적인 검토 방법
- 워크스루 방법은 개발자가 소집한 전문가들에 의해 개발자의 작업을 검토하며 설계문서가 고객의 요구사항을 정확히 명시하고 있는지, 작업 진척 상황 등을 확인
- 정적분석 도구는 소프트웨어의 장애보다 결함을 발견하는 것이며 실제로 실행하지 않은 상태에서 도구의 지원으로 분석을 수행함
- 정적분석 도구로 일반적인 결함을 찾아냄

학습정리

2. 소프트웨어 동적분석

- 프로그램을 실행하여 오류를 찾고 프로그램을 분석하는 기법이며 화이트박스 테스트 방법은 소스코드를 가지고 테스트하는 방법
- 블랙박스 테스트는 소스코드 없이 실행만으로 테스트하는 방법
- 동등분할(Equivalence Partitioning)은 프로그램의 출력 값이 같은 입력을 그룹으로 묶어 각 그룹의 대표 입력 값에 대해 테스트하는 방법
- 경계 값 분석(Boundary Value Analysis)은 동등분할의 경계 값에서 오류가 발생할 가능성이 높기 때문에 이 부분을 대상으로 테스트하는 방법
- 결정 테이블 테스팅(Decision Table Testing)은 프로그램의 명세를 결정 테이블로 나타내 모든 조합에 대한 테스트 케이스를 생성하는 방법
- 상태 전이 테스팅(State Transition Testing)은 프로그램의 각 상태를 다이어그램으로 나타내 상태 간의 전이에 필요한 입력과 그 입력에 대해 실제로 상태 전이가 올바르게 일어나는지를 테스트하는 방법
- 유스 케이스 테스팅(Use Case Testing)은 프로그램이 실제 사용되는 과정을 모델링 한 유스 케이스를 바탕으로 테스트를 생성
- 조합 테스팅(Pairwise Testing)은 프로그램의 입력 값을 몇 가지 조합으로 묶어서 테스팅하는 방법