SW 개발·테스트 중심
DevOps 환경구축 및 활용

강사 정보

시네틱스 대표 한동준
handongjoon@gmail.com
dongjoon.han@synetics.kr

7 JDepend

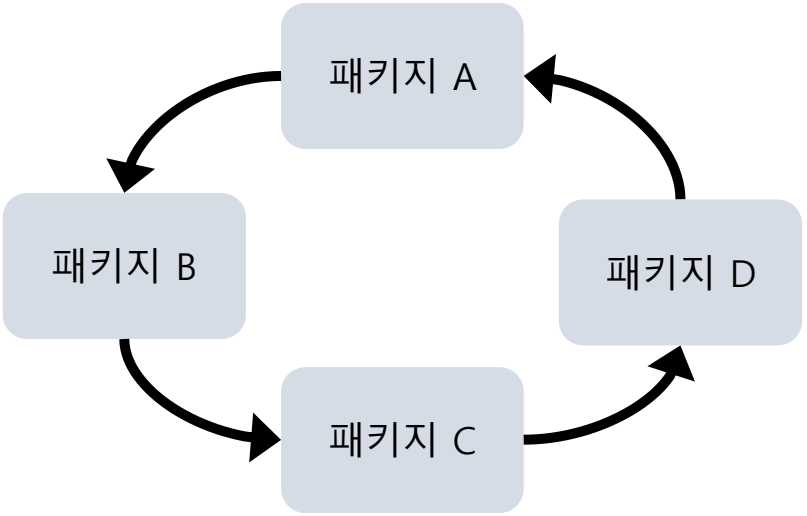
㈜시네틱스 한동준 대표

JDepend

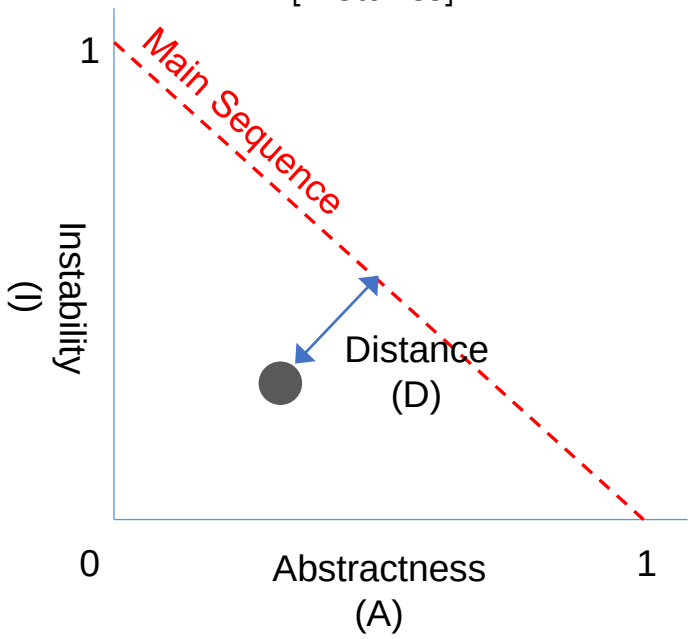
❑ 도구 개요

- 패키지 간 의존성 분석
  - 나를 호출한 패키지
  - 내가 호출한 패키지
- 특히, 원형 의존성 존재 여부 확인 가능
- Distance 분석

[원형 의존성]



[Distance]



의존성 분석

❑ 개요

- 함수, 변수의 **호출관계를 분석**하는 도구
- 도구가 추구하는 방향에 따라 패키지/클래스(파일)/함수 단위로 표현

❑ 목적

- 아키텍처 구조에서 서브 시스템 간의 의존이 적절한지 확인
  - 서브 시스템 레이어에서 각 레이어 간 호출 관계 (예. Autosar)
  - 서브 시스템 레벨에서 각 레벨 간 호출 관계 (자식과 부모 패키지)
  - 원형 의존성(상호 참조) 관계
- 객체지향에 특화되어, 추상화와 구체화의 정도를 확인 (Part 2 내용)

❑ 대표 도구

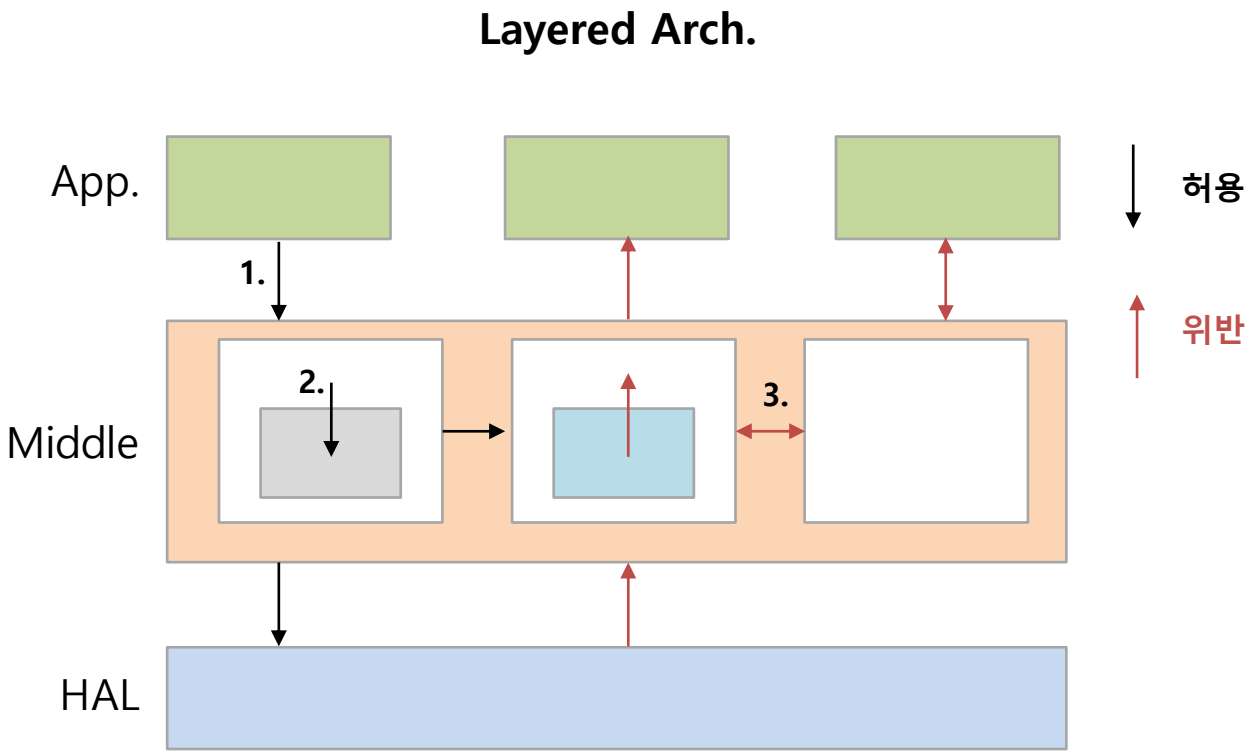
언어	외산		오픈소스
C/C++	Lattix Imagix4D	CppDepend	Doxygen
Java		JArchtect	JDepend Doxygen

❑ 의존성 분석 표현 종류

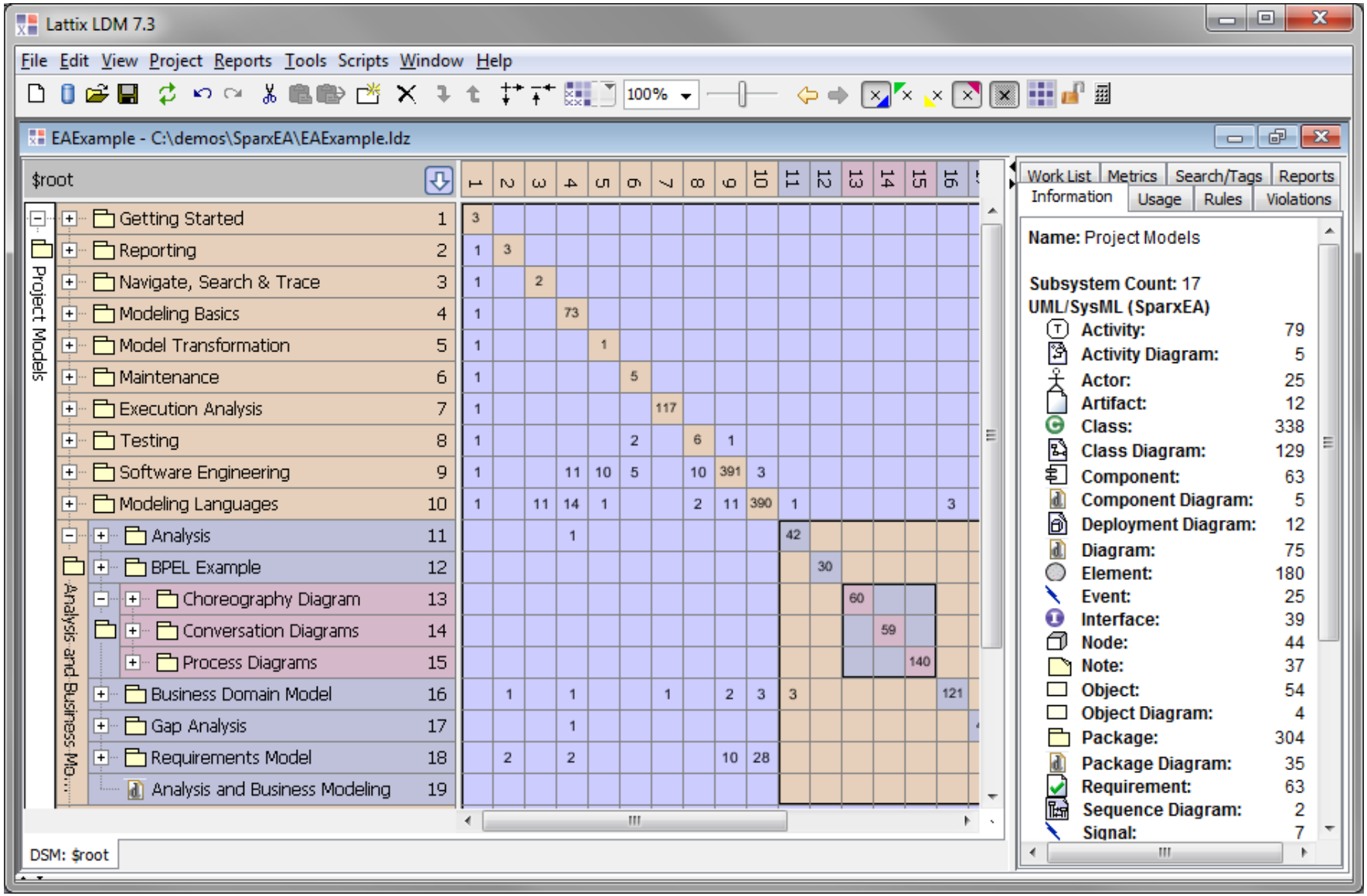
- DSM: Dependency Structure Matrix로, X/Y축의 형태로 서브 시스템 간의 의존성 확인에 유리
- 지표: 호출 대상을 지표로 표현
  - 특히, 객체지향 관련 도구는 **Robert C. Martin**의 논문을 근거로, 의존성 관련 지표를 정의하고 숫자로 표현
- 다이어그램: 호출 관계를 다이어그램으로 표현

Layered Architecture에서 서브 시스템간 의존성

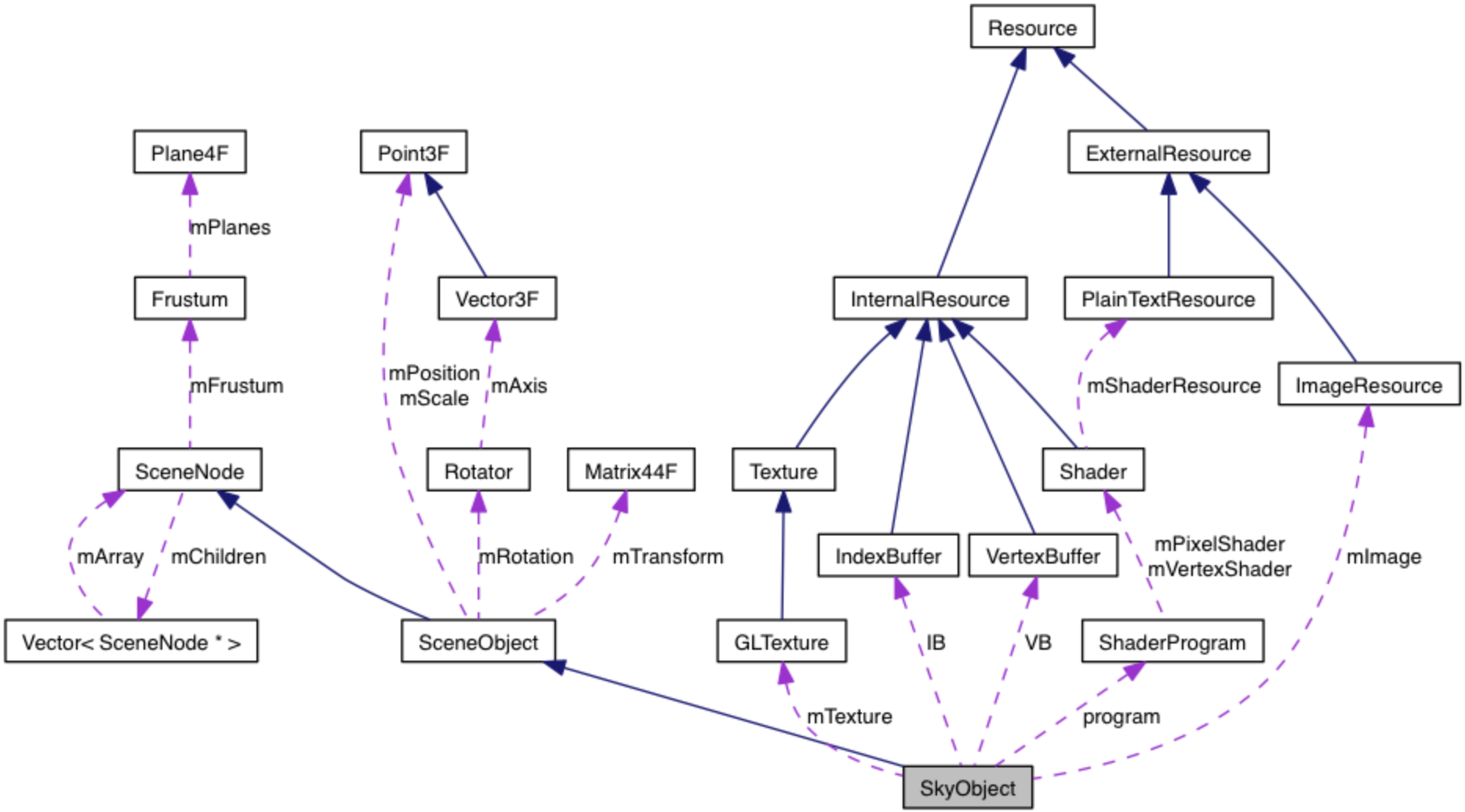
1. 각 레이어 간 호출 관계 (예. Autosar)
2. 동일 레벨 간 호출 관계 (자식과 부모 패키지)
3. 원형 의존성(상호 참조) 관계



예제) DSM - Lattix

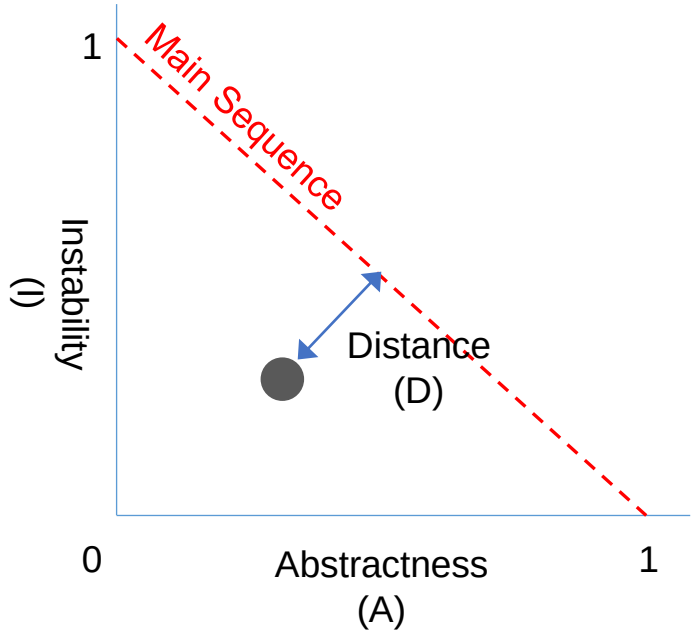


예제) 다이어그램 - Doxygen

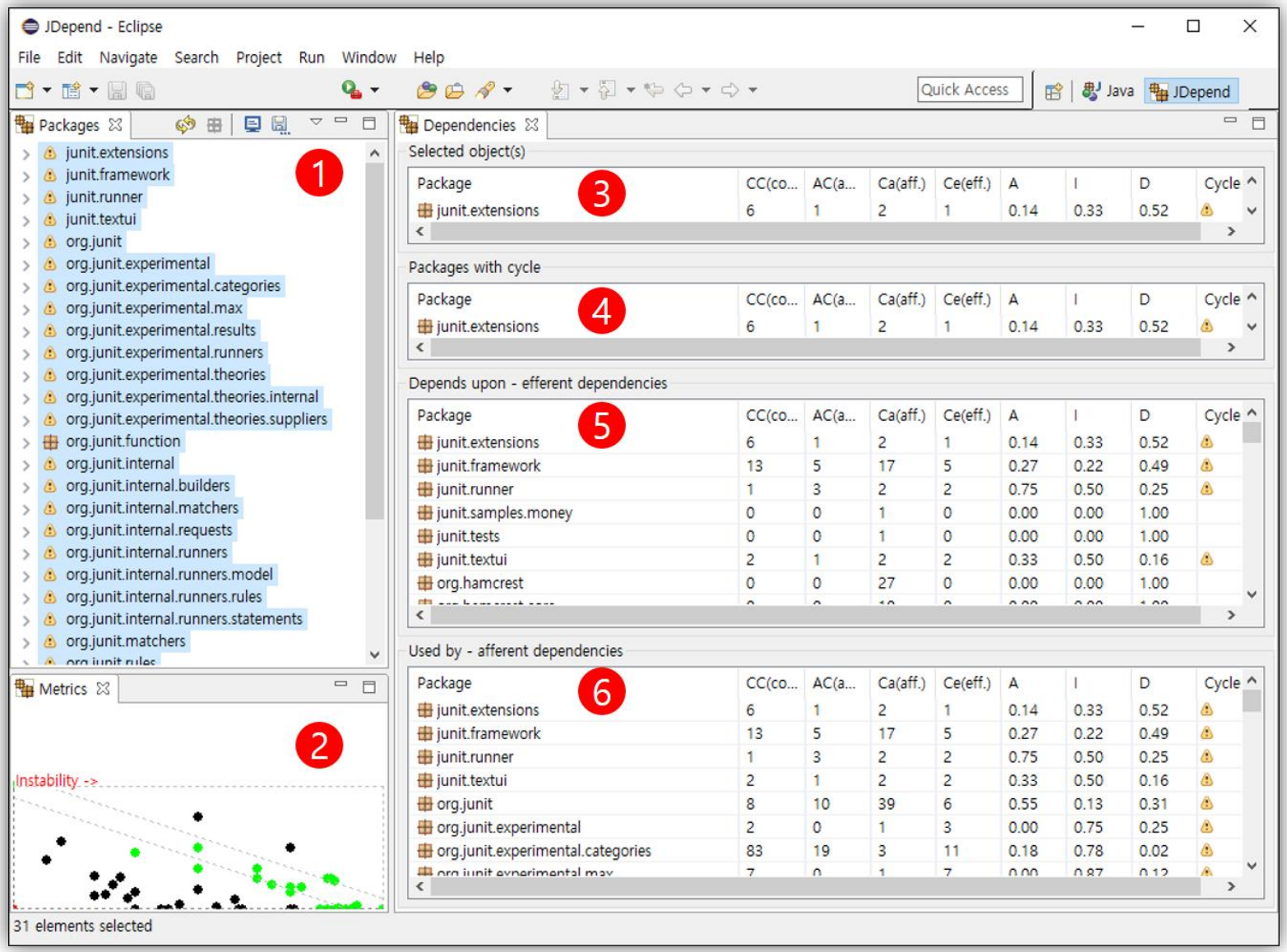


Robert C. Martin – OO Metrics

- **기준은 패키지 단위**
- **CC(Concrete Class)**
  - 인터페이스나 추상 클래스가 아닌 구체 클래스의 수
- **AC(Abstract Class)**
  - 추상 클래스나 인터페이스의 수를 나타내며 확장성의 척도
- **Ca(Afferent Couplings)**
  - 나에게 의존하는 패키지 수를 나타내며 책임의 척도
- **Ce(Efferent Couplings)**
  - 내가 호출하는 패키지의 수를 나타내며 독립성의 척도
- **A(Abstractness)**
  - A = AC / (CC+AC) **추상화 정도**를 나타내며, 0 은 구체적인 패키지이며, 1 은 추상적인 패키지
- **I(Instability)**
  - I = Ce / (Ce+Ca) **변화에 대한 안정성**을 나타내며 0 부터 1 사이의 값
  - 0 은 외부 변화에 영향 없는 패키지이며, 1 은 작은 변화에도 영향 받는 패키지
- **D(Distance to Main Sequence)**
  - Main Sequence 로부터의 거리
  - Main Sequence란 이상적인 패키지로 완전 추상적이면서 안정적이거나 완전 구체적이면서 불안정한 패키지
- **Cycle(Package dependency cycles)**
  - 패키지들 상호 간에 의존성을 가지고 있을 때 발생



JDepend 모습


JDepend - Eclipse
File Edit Navigate Search Project Run Window Help
Quick Access
Java
JDepend
Packages
1
junit.extensions
junit.framework
junit.runner
junit.textui
org.junit
org.junit.experimental
org.junit.experimental.categories
org.junit.experimental.max
org.junit.experimental.results
org.junit.experimental.runners
org.junit.experimental.theories
org.junit.experimental.theories.internal
org.junit.experimental.theories.suppliers
org.junit.function
org.junit.internal
org.junit.internal.builders
org.junit.internal.matchers
org.junit.internal.requests
org.junit.internal.runners
org.junit.internal.runners.model
org.junit.internal.runners.rules
org.junit.internal.runners.statements
org.junit.matchers
Metrics
2
Instability ->
Dependencies
Selected object(s)
3
Package CC(co... AC(a... Ca(aff.) Ce(eff.) A I D Cycle
junit.extensions 6 1 2 1 0.14 0.33 0.52
Packages with cycle
4
Package CC(co... AC(a... Ca(aff.) Ce(eff.) A I D Cycle
junit.extensions 6 1 2 1 0.14 0.33 0.52
Depends upon - efferent dependencies
5
Package CC(co... AC(a... Ca(aff.) Ce(eff.) A I D Cycle
junit.extensions 6 1 2 1 0.14 0.33 0.52
junit.framework 13 5 17 5 0.27 0.22 0.49
junit.runner 1 3 2 2 0.75 0.50 0.25
junit.samples.money 0 0 1 0 0.00 0.00 1.00
junit.tests 0 0 1 0 0.00 0.00 1.00
junit.textui 2 1 2 2 0.33 0.50 0.16
org.hamcrest 0 0 27 0 0.00 0.00 1.00
Used by - afferent dependencies
6
Package CC(co... AC(a... Ca(aff.) Ce(eff.) A I D Cycle
junit.extensions 6 1 2 1 0.14 0.33 0.52
junit.framework 13 5 17 5 0.27 0.22 0.49
junit.runner 1 3 2 2 0.75 0.50 0.25
junit.textui 2 1 2 2 0.33 0.50 0.16
org.junit 8 10 39 6 0.55 0.13 0.31
org.junit.experimental 2 0 1 3 0.00 0.75 0.25
org.junit.experimental.categories 83 19 3 11 0.18 0.78 0.02
31 elements selected

JDepend 모습

Metric Results

[summary] [packages] [cycles] [explanations]

The following document contains the results of a JDepend metric analysis. The various metrics are defined at the bottom of this document.

Summary

[summary] [packages] [cycles] [explanations]

Package	TC	CC	AC	Ca	Ce	A	I	D	V
junit.extensions	6	6	0	1	2	0.0%	67.0%	33.0%	1
junit.framework	17	13	4	7	9	24.0%	56.0%	20.0%	1
junit.runner	3	1	2	2	6	67.0%	75.0%	42.0%	1
junit.textui	2	2	0	0	6	0.0%	100.0%	0.0%	1
org.junit	18	8	10	11	6	56.0%	35.0%	9.0%	1
org.junit.experimental	2	2	0	0	5	0.0%	100.0%	0.0%	1
org.junit.experimental.categories	11	7	4	0	10	36.0%	100.0%	36.0%	1
org.junit.experimental.max	8	8	0	0	11	0.0%	100.0%	0.0%	1
org.junit.experimental.results	6	6	0	0	7	0.0%	100.0%	0.0%	1
org.junit.experimental.runners	1	1	0	0	5	0.0%	100.0%	0.0%	1
org.junit.experimental.theories	15	8	7	2	9	47.0%	82.0%	28.0%	1
org.junit.experimental.theories.internal	8	8	0	1	6	0.0%	86.0%	14.0%	1
org.junit.experimental.theories.suppliers	2	1	1	0	4	50.0%	100.0%	50.0%	1
org.junit.internal	13	11	2	13	10	15.000001%	43.0%	41.0%	1
org.junit.internal.builders	8	8	0	3	10	0.0%	77.0%	23.0%	1
org.junit.internal.matchers	4	3	1	2	5	25.0%	71.0%	4.0%	1
org.junit.internal.requests	3	3	0	2	6	0.0%	75.0%	25.0%	1

[Maven] Goal 및 Usage

❑ Goal

Goal	설명
jdepend:generate	JDepend 의존성 분석 보고서를 생성한다.

❑ Usage

```
<project>
  ...
  <reporting>
    <plugins>
      ...
      <plugin>
        <groupId>org.codehaus.mojo</groupId>
        <artifactId>jdepend-maven-plugin</artifactId>
        <version>2.0</version>
      </plugin>
      ...
    </plugins>
  </reporting>
  ...
</project>
```

[Maven] 보고서 화면

Metric Results

[summary] [packages] [cycles] [explanations]

The following document contains the results of a JDepend metric analysis. The various metrics are defined at the bottom of this document.

Summary

[summary] [packages] [cycles] [explanations]

Package	TC	CC	AC	Ca	Ce	A	I	D	V
junit.extensions	6	6	0	1	2	0.0%	67.0%	33.0%	1
junit.framework	17	13	4	7	9	24.0%	56.0%	20.0%	1
junit.runner	3	1	2	2	6	67.0%	75.0%	42.0%	1
junit.textui	2	2	0	0	6	0.0%	100.0%	0.0%	1
org.junit	18	8	10	11	6	56.0%	35.0%	9.0%	1
org.junit.experimental	2	2	0	0	5	0.0%	100.0%	0.0%	1
org.junit.experimental.categories	11	7	4	0	10	36.0%	100.0%	36.0%	1
org.junit.experimental.max	8	8	0	0	11	0.0%	100.0%	0.0%	1
org.junit.experimental.results	6	6	0	0	7	0.0%	100.0%	0.0%	1
org.junit.experimental.runners	1	1	0	0	5	0.0%	100.0%	0.0%	1
org.junit.experimental.theories	15	8	7	2	9	47.0%	82.0%	28.0%	1
org.junit.experimental.theories.internal	8	8	0	1	6	0.0%	86.0%	14.0%	1
org.junit.experimental.theories.suppliers	2	1	1	0	4	50.0%	100.0%	50.0%	1
org.junit.internal	13	11	2	13	10	15.000001%	43.0%	41.0%	1
org.junit.internal.builders	8	8	0	3	10	0.0%	77.0%	23.0%	1

[Maven] 보고서 화면

Cycles

[summary] [packages] [cycles] [explanations]

Package	Package Dependencies
junit.extensions	junit.framework org.junit.runner.manipulation org.junit.runner org.junit.runners.model org.junit.internal.runners.model org.junit.internal org.junit org.junit.internal
junit.framework	org.junit.runner.manipulation org.junit.runner org.junit.runners.model org.junit.internal.runners.model org.junit.internal org.junit org.junit.internal
junit.runner	junit.framework org.junit.runner.manipulation org.junit.runner org.junit.runners.model org.junit.internal.runners.model org.junit.internal org.junit org.junit.internal

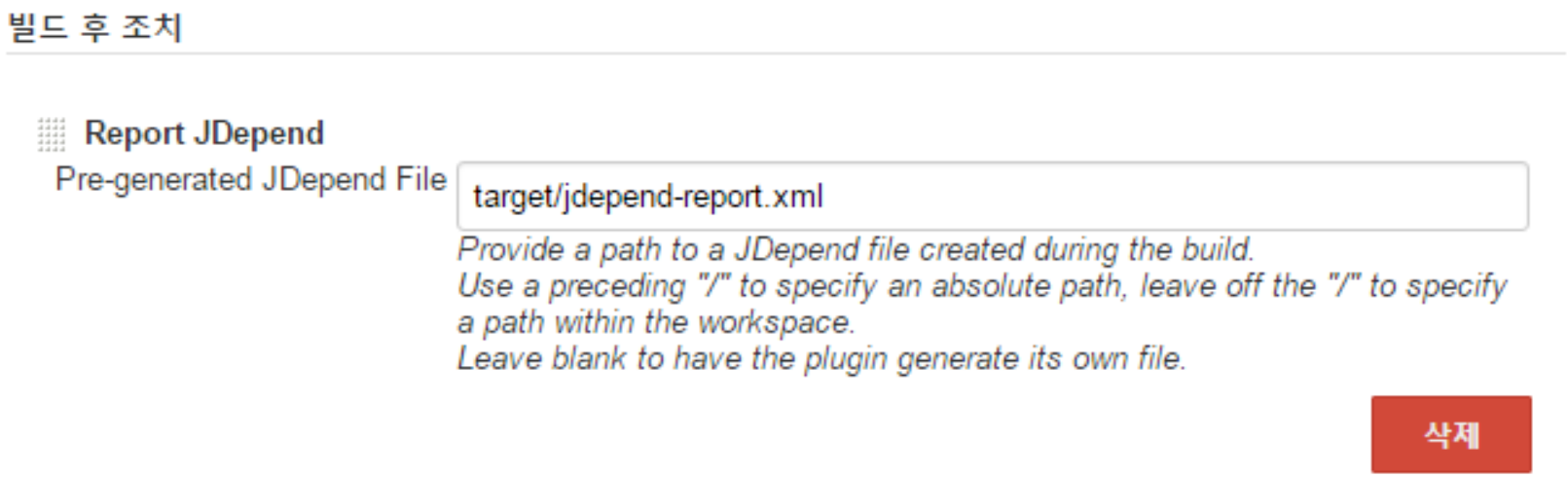
[Jenkins] 플러그인 설치

❑ 플러그인 관리에서 idepend 검색

[Jenkins] Job 설정

❑ Goal 설정

❑ 보고서 생성 설정



[Jenkins] 보고서 생성

Metric Results

[summary] [packages] [cycles] [explanations]

The following document contains the results of a JDepend metric analysis. The various metrics are defined at the bottom of this document.

Summary

[summary] [packages] [cycles] [explanations]

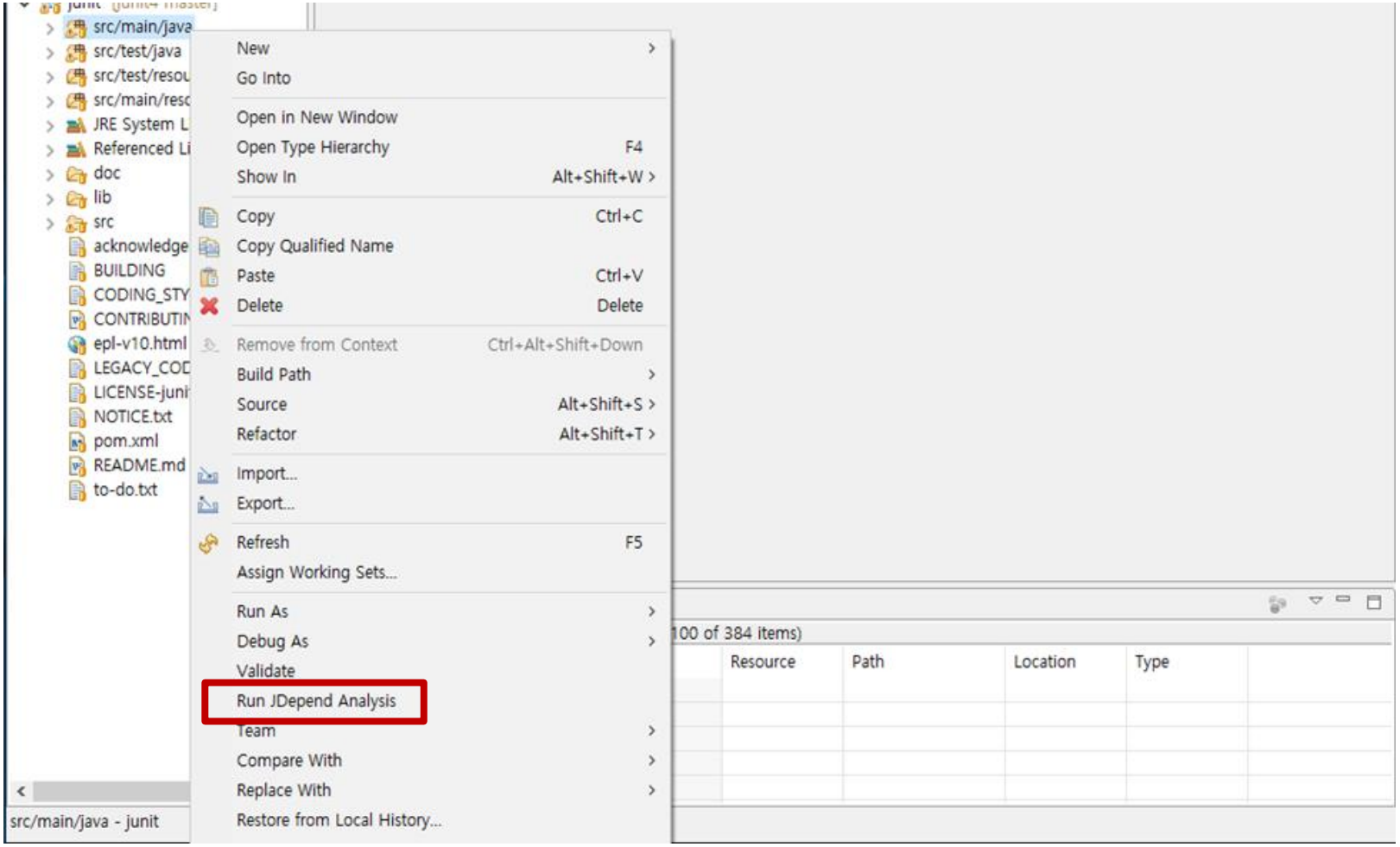
Package	TC	CC	AC	Ca	Ce	A	I	D	V
junit.extensions	6	6	0	1	2	0.0%	67.0%	33.0%	1
junit.framework	17	13	4	7	9	24.0%	56.0%	20.0%	1
junit.runner	3	1	2	2	6	67.0%	75.0%	42.0%	1
junit.textui	2	2	0	0	6	0.0%	100.0%	0.0%	1
org.junit	18	8	10	11	6	56.0%	35.0%	9.0%	1
org.junit.experimental	2	2	0	0	5	0.0%	100.0%	0.0%	1
org.junit.experimental.categories	11	7	4	0	10	36.0%	100.0%	36.0%	1
org.junit.experimental.max	8	8	0	0	11	0.0%	100.0%	0.0%	1
org.junit.experimental.results	6	6	0	0	7	0.0%	100.0%	0.0%	1
org.junit.experimental.runners	1	1	0	0	5	0.0%	100.0%	0.0%	1
org.junit.experimental.theories	15	8	7	2	9	47.0%	82.0%	28.0%	1
org.junit.experimental.theories.internal	8	8	0	1	6	0.0%	86.0%	14.0%	1
org.junit.experimental.theories.suppliers	2	1	1	0	4	50.0%	100.0%	50.0%	1
org.junit.internal	13	11	2	13	10	15.000001%	43.0%	41.0%	1
org.junit.internal.builders	8	8	0	3	10	0.0%	77.0%	23.0%	1
org.junit.internal.matchers	4	3	1	2	5	25.0%	71.0%	4.0%	1
org.junit.internal.requests	3	3	0	2	6	0.0%	75.0%	25.0%	1

[Eclipse] 플러그인 설치

1. Eclipse 상단의 Help -> Eclipse Marketplace 선택
2. Find에 'JDepend' 입력 후 검색
3. 'JDepend4Eclipse'의 'Install' 클릭
4. 설치 확인 화면에서 'Confirm' 클릭
5. 라이선스에 동의하면 설치 진행
6. 설치 완료 후 Eclipse 재시작

[Eclipse] 분석 실행

- ❑ 프로젝트의 분석 대상 소스코드 패키지를 선택하고 오른쪽 클릭
- ❑ 'Run JDepend Analysis'를 선택



[Eclipse] 분석 결과 확인