The human lives in a safety software by Thinktools

THE HUMAN LIVES IN A SAFETY SOFTWARE BY THINKTOOLS

열정과 의지만으로 결과가 나오지 않았다면
이제는 도구를 바꿔보세요.

ThinkforBL

WHO WE ARE

산업 현장 중심의 소프트웨어 실용 공학 전문가 입니다.

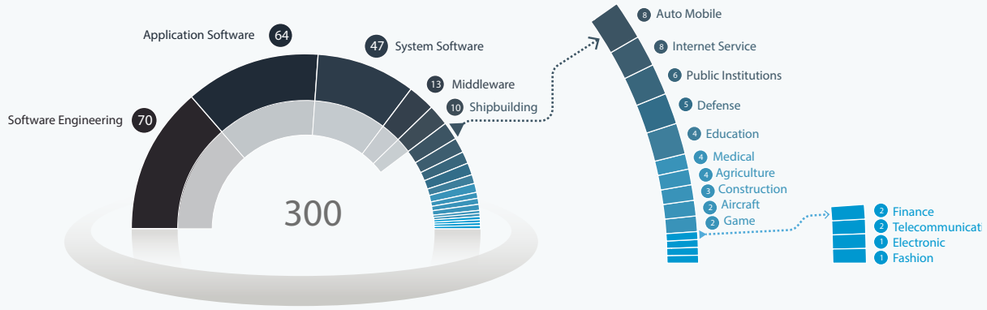
—

씽크포비엘은 소프트웨어 산업 현장의, 실질적 문제들에 대하여 과학적, 기술적 관점에서 최적화된 해법을 도출하는 실용 공학 전문기업입니다. 문제의 현상이 아니라 원인과 본질을 조명하며 필요한 각종 도구를 개발, 소프트웨어 공학의 새로운 글로벌 표준을 제시하고 있습니다.

OVER 300 REFERENCE IN 10 YEARS

—

10여년 간 도메인 별 300여 소프트웨어 R&D 기업의 컨설팅 경험으로 축적된, 우리만의 노하우와 방법론, 기술을 자부합니다.



〈 References by Industrial Area 〉

Thinktools IS

장인은 도구를 가리지 않는다고요?
도구를 잘 골라 사용하는 사람이 장인입니다.

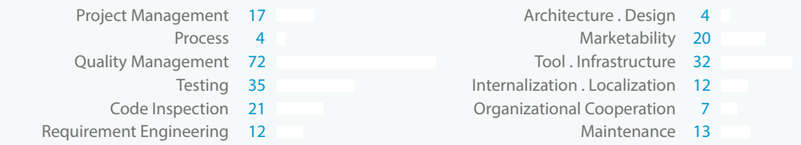
소프트웨어 산업 특성에 따라, 높은 수준에서의 공학 활동이 필요한 영역과 다양한 지원 도구들이 있습니다.
ThinkTools는 특히 Safety Software, Mission Critical Area, CPS (Cyber-Physical Systems),
대형 프로젝트의 거버넌스 등에 특화된 도구입니다.

CETA
Cause and Effect Test Auto Analyzer
테스트 설계를 지원하는 반 자동화 도구

DRONACE
Drone Advanced Certification Environment
드론 내 소프트웨어의 성능을 가상환경에서 자동 검증하는 도구

SMAC
Safety Mechanism Analysis in Code
전장 소프트웨어의 Safety Mechanism 코드를 제안하는 도구

kiuwan
당신만의 코드 관리 컨설턴트



〈 References by Software Engineering Area 〉

OUR COVERAGE AREA based on SWEBOK 3.0

—

T-Biz Engineering: ORGANIZATIONAL COOPERATION 1 · INTERNALIZATION .LOCALIZATION 12 · MARKETABILITY 20 · HUMAN RESOURCE DESIGN 7

SWEBOK 2.0: SOFTWARE REQUIREMENTS 12 · SOFTWARE DESIGN 4 · SOFTWARE CONSTRUCTION 21 · SOFTWARE TESTING 35 · SOFTWARE MAINTENANCE 13 · SOFTWARE CONFIGURATION MANAGEMENT · SOFTWARE ENGINEERING MANAGEMENT 17 · SOFTWARE ENGINEERING PROCESS 4 · SOFTWARE ENGINEERING TOOLS AND METHODS 32 · SOFTWARE QUALITY 72 · SOFTWARE ENGINEERING PROFESSIONAL PRACTICE

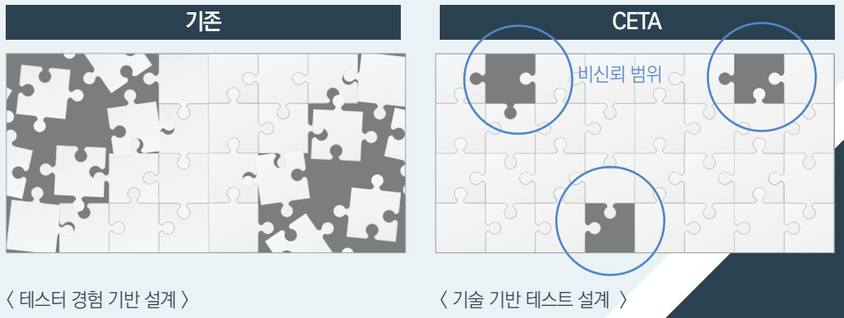
SWEBOK 3.0: SOFTWARE ENGINEERING ECONOMICS · COMPUTER FUNDATIONS · MATHEMATICAL FOUNDATIONS · ENGINEERING FOUNDATIONS

		ORGANIZATIONAL COOPERATION	INTERNALIZATION .LOCALIZATION	MARKETABILITY	HUMAN RESOURCE DESIGN	SOFTWARE REQUIREMENTS	SOFTWARE DESIGN	SOFTWARE CONSTRUCTION	SOFTWARE TESTING	SOFTWARE MAINTENANCE	SOFTWARE CONFIGURATION MANAGEMENT	SOFTWARE ENGINEERING MANAGEMENT	SOFTWARE ENGINEERING PROCESS	SOFTWARE ENGINEERING TOOLS AND METHODS	SOFTWARE QUALITY	SOFTWARE ENGINEERING PROFESSIONAL PRACTICE	SOFTWARE ENGINEERING ECONOMICS	COMPUTER FUNDATIONS	MATHEMATICAL FOUNDATIONS	ENGINEERING FOUNDATIONS
		1	12	20	7	12	4	21	35	13		17	4	32	72					
Universal software	System		•		K	• (C)		• (K)	• (C)						• (K/C)		•	•		
	Middleware		•		K	• (C)		• (K)	• (C)						• (K/C)		•	•		
	Application			•	K	• (C)										• (K)	•				
	Construction	•	•		K	• (C)										• (K)	•			•	•
	Agriculture	•		•	K	• (C)								•			•				•
	Public Institutions	•			K	• (C)				•		•				• (K)					
	Education	•	•	•	K	• (C)											•				
	Logistics			•	K	• (C)											•				•
	Internet Service	•	•	•	K	• (C)											•				
	Telecommunication				K		•	• (K)	• (C)	•	•				• (K/C)		•				
Industrial software	Finance				K	• (C)	•	• (K)	• (C)			•				• (K)	•	•			
	Auto mobile	•	•	•	K	• (C)	•	• (S/K)	• (C)		•	•	•	•	• (S/K/C)	• (K)	•	•	•	•	•
	Aircraft	•	•		K		•	• (K)	• (D/C)		•	•	•	•	• (D/K/C)	• (K)		•	•	•	•
	Shipbuilding	•			K	• (C)	•	• (K)	• (C)	•	•	•	•	•	• (K/C)	• (K)	•	•	•	•	•
	Medical	•			K	• (C)	•	• (K)	• (C)						• (K/C)	• (K)	•	•	•	•	•
	Defense	•	•	•	K		•	• (K)	• (C)	•	•	•			• (K/C)	• (K)		•	•	•	•
	Energy chemistry	•			K			• (K)	• (C)				•	•	• (K/C)	• (K)		•	•	•	•
	Nuclear Power	•			K		•	• (K)	• (C)	•	•		•	•	• (K/C)	• (K)		•	•	•	•
	Railway	•			K		•	• (K)	• (C)	•	•		•	•	• (K/C)	• (K)	•	•	•	•	•

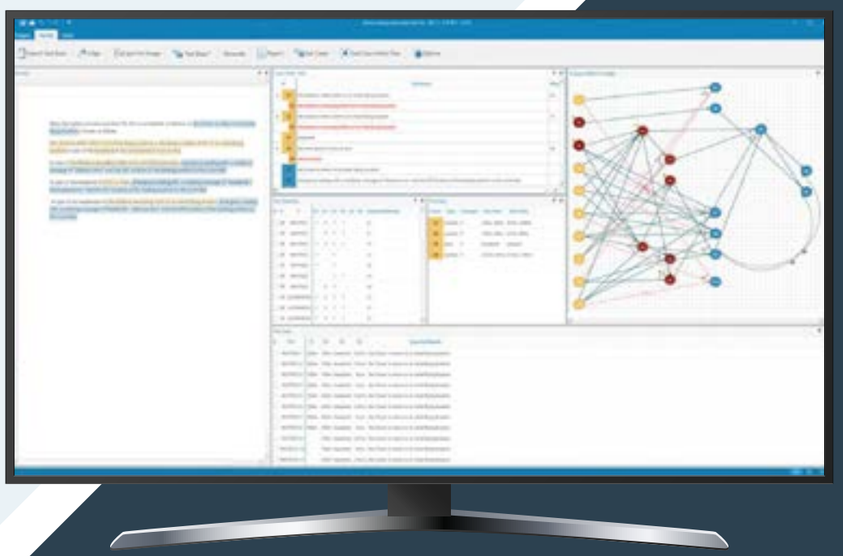
THINKTOOLS ONE. CETA
WHAT IS CETA?

드디어, 모든 경우의 예외상황까지 테스트하는 도구가 나왔습니다.

지금까지는 모든 경우를 테스트 하는 것은 불가능하다고 생각한 잘못된 방식을 받아 들였습니다.
CETA는 인간이 실수할 수 있는 모든 조건을 (반)자동으로 도출하는 기능 레벨의 기술적 설계 도구입니다. 이제는 '생각나는 만큼' 이 아닌, '필요한 만큼' 하는 테스트를 진행할 수 있습니다.
안전과 생명을 지키는 일이라면 '99%'는 부족합니다.



〈 테스터 경험 기반 설계 〉 〈 기술 기반 테스트 설계 〉

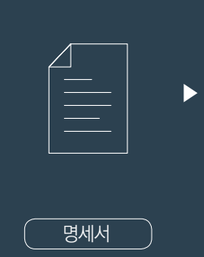


WHY
SHOULD YOU USE CETA?

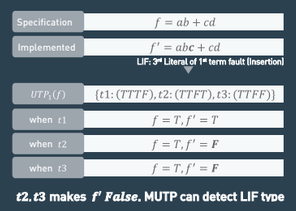
테스트를 했음에도 결함 및 사고가 발생해 온 이유는,
'제대로 된' 테스트가 아니었기 때문입니다.

CETA 는 테스터 경험에 의존, 주관적으로 진행하던 테스트 설계를 과학적, 기술적, 객관적으로 수행하도록 합니다. CETA 는 해당 상황에서 인간이 발생시킬 수 있는 모든 경우의 실수를 자동으로 도출합니다. CETA는 효율적 테스트를 위하여, 자동 도출된 수만 개의 테스트케이스 중 기능 특성에 적합한 테스트 우선순위를 제안합니다.

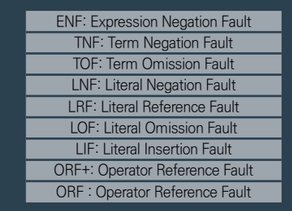
인간의 실수를 고려한 테스트 케이스 설계 방법



명세서



논리식 분석 방법 (MUTP 예시)



DNF 기반 실수 유형

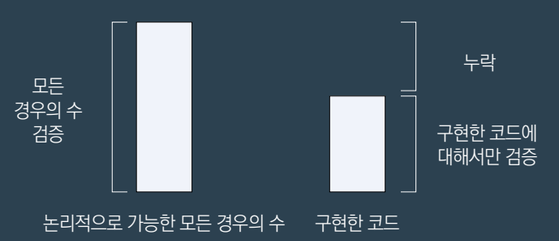
산업에 미치는 SW의 물리적 영향이 커지는 만큼, 테스트 완성도도 매우 중요합니다.

이전의 산업 현장에서 테스트는 귀찮고 비효율적인 과정이라 여겨지기도 했습니다.
하지만 이제는 SW가 모든 산업을 통제하는 시대, SW의 오류가 산업을 마비시키고 인명피해를 가져오는 시대입니다.
CETA는 무엇과도 바꿀 수 없는 동료와 가족의 안전을 보다 확실히 보호하고자, 가장 현실적이고 체계적인 지원을 합니다.

기존의 테스트 커버리지에 속지 마십시오.

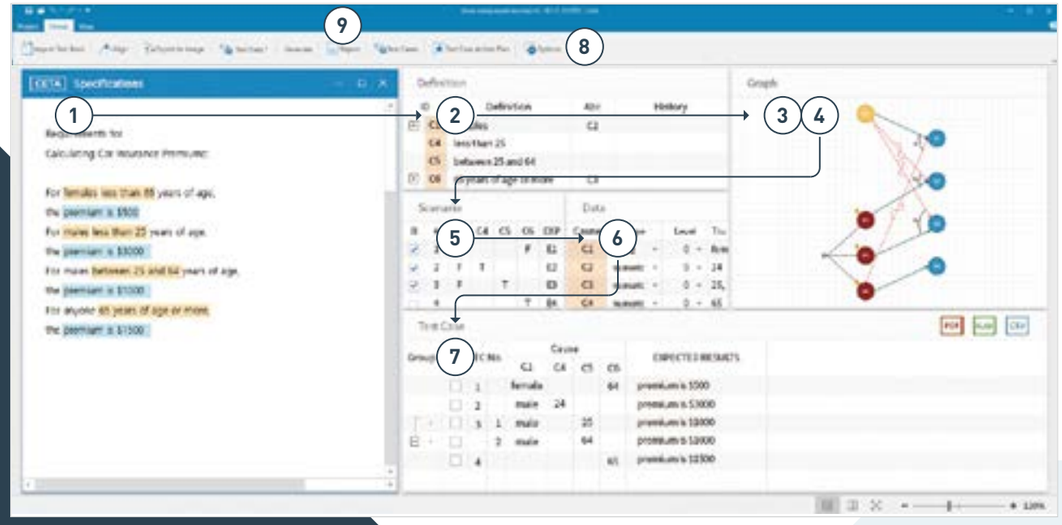
기존의 코드 기반 테스트 커버리지는 품질 커버리지를 의미하지 않습니다.
구현된 코드가 제대로 동작하는 지를 의미할 뿐, 구현되어야 할 안전 메커니즘이 구현 되어 있는지를 검사하지 않습니다.
특정 기능이 논리적으로 발생 가능한 모든 경우의 수 대비 얼마만큼 테스트 되었는지에 대해 정보가 없을 경우, 현장에서의 의사 결정에 차질이 생기게 됩니다.

기존과 CETA에서의 커버리지 차이



THINKTOOLS ONE. CETA
DETAILS FROM CETA

Test Design Process using CETA



1. Cause & Effect 도출
2. Cause & Effect 축약
3. Cause & Effect 그래프 작성
4. Cause & Effect 그래프 인스펙션
5. 테스트 시나리오 확인 및 베이스 시나리오 선택
6. 테스트 데이터 입력
7. 테스트 케이스 출력
8. 테스트 케이스 우선순위화
9. 테스트 케이스 설계 보고서 생성

01 IEEE 후원 테스트 컨퍼런스(ICST - International Conference Software Testing) 에서 논문 게재 및 발표
2017.03.17 IEEE InSTA 2017 in Tokyo
2019.04.22 IEEE InSTA 2019 in Xian

02 아시아 SW품질관리 컨퍼런스 (ASQN - Asia Software Quality Network) 발표
2018.06.29 ASQN 2018 in Beijing
2019.09.09 ASQN 2019 in Tokyo

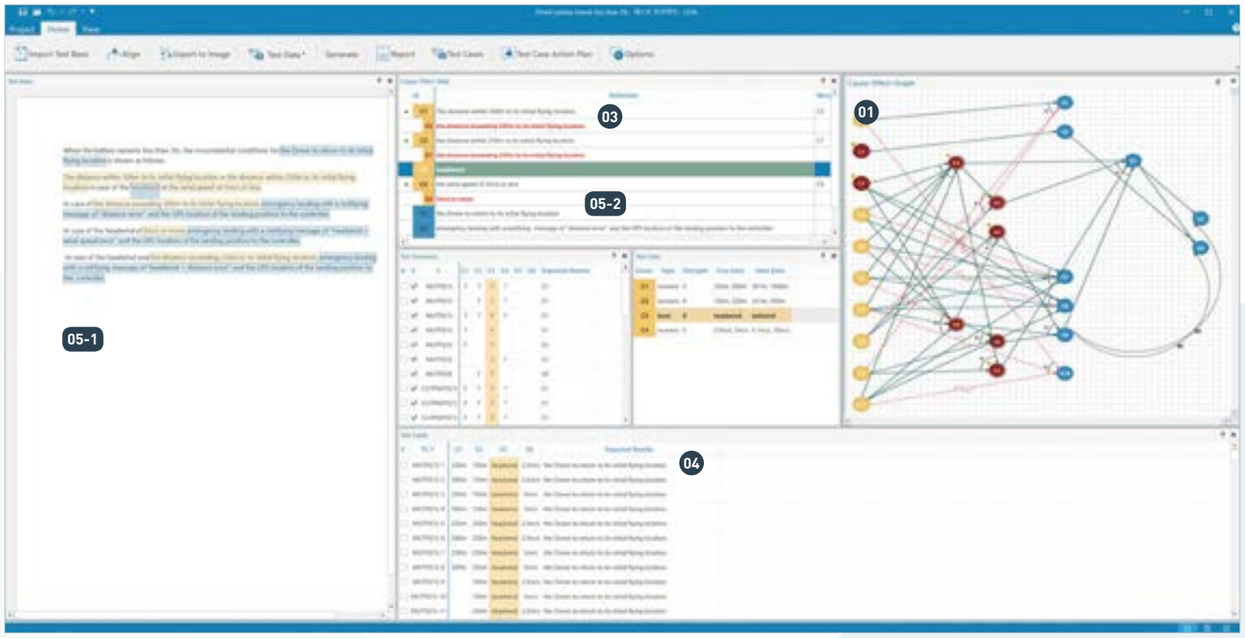
03 한국정보통신기술협회(TTA) 의 기능 안전성 기반 소프트웨어 검증 표준 제정
소프트웨어 기능 안전성 검증을 위한 명세 기반 테스트 설계 방법(TTAK.KO-11.0251)
소프트웨어 기능 안전성 검증을 위한 테스트 커버리지 측정 방법(TTAK.KO-11.0250)

현재 내 상황에서는 어떤 커버리지를 보아야 하나요?

〈 CHARACTERISTIC OF FUNCTION 〉	〈 TEST COVERAGE 〉	
결과 논리식의 원인 조합 결함을 확인할 경우	UTPC (Unique True Point Coverage)	Viewpoint : CODE
결과 논리식의 원인 반영 결함을 확인할 경우	CUTPNFP (Unique True Point And Near Faise Point Pair Coverage)	
기능의 기본적인 동작만 확인이 필요할 경우	Base Scenario Coverage	Viewpoint :SPEC
예외 처리 구현 여부를 확인하고자 할 경우	Invailid Scenario Coverage	
입력 데이터가 많고 다양한 논리적 영향을 줄 경우	Cause Coverage	
복잡하고 다양한 결과를 만들어 낼 경우	Effect Coverage	
데이터 유효성 검증이 중요할 경우	Cause Test Data Coverage	
원인 요소들 간 복잡한 논리적 관계를 가질 경우	Valid Scenario Coverage	
	Complex Logical Relation Coverage	

THINKTOOLS ONE. CETA
DETAILS FROM CETA

Functions & Effects



01
VISUALIZATION OF LOGICAL SENTENCE BASED ON SPECIFICATIONS (Requirements, Functions)

인간의 추상 언어를 기반으로 소프트웨어 기능이 어떻게 동작하는지 기술하는 것은 큰 도전과 같습니다. 소프트웨어는 언제나 논리적 조건과 논리적 상태를 의미하기 때문에, 논리적 표현이 용이한 방법으로 기술한다면, 빠른 이해와 상호 간의 커뮤니케이션 오류를 줄일 수 있습니다.

02
ERROR SUGGESTION REGARDING SUSPECT SPECIFICATIONS

사람이 작성한 요구사항에는 누락, 중복, 모호함, 충돌과 같은 많은 오류들이 있습니다. 하지만, 작성된 문장에서 논리적 오류를 발견하는 것은 쉽지 않습니다. CETA 는 논리적 오류로 의심되는 명세서의 내용을 자동으로 식별하여 제시합니다.

03
SUGGESTION REGARDING DUPLICATED ITEMS IN THE SPECIFICATION

사실상 같은 내용이지만, 요구사항 명세서에 있는 반복적인 항목들을 자동으로 추출하여, 실수 예방을 위해 중복 제거할 수 있도록 제안합니다.

04
AUTOMATIC SUGGESTION FOR ALL POSSIBLE CASES

인간의 논리적 실수 유형에 따라, 발생 가능한 모든 경우의 수를 자동으로 도출하여 테스트케이스로 생성합니다.

05
EXTREMELY CONVENIENT UI BASED ON MOUSE CLICK AND DRAG/DROP

입력에 대해 부담 갖지 않으셔도 됩니다. 명세서 불러오기를 시작으로, 몇 번의 마우스 클릭과 드래그 만으로 모든 설계를 마칠 수 있을 만큼 편리성을 강화하였습니다.

05-1 드래그를 통한 Cause/Effect 입력

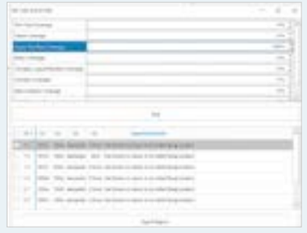
05-2 드래그를 통한 축약

06
INSTANT VISUALIZATION OF ALL ITEMS BY TRACEABILITY

그물같은 복잡한 화면에 두려워할 필요 없습니다. 아무리 복잡한 명세서의 수십여 개의 항목이라도, 선택한 항목만 바로 추적하여 볼 수 있도록 하이라이트 가시화 합니다.

07
AUTOMATIC SUGGESTION OF EFFICIENT TEST PRIORITIZATION TO ACHIEVE THE BEST TEST COVERAGE

방대한 테스트케이스를 모두 수행하는 것은 현실적으로 가능하지도 않으며, 낭비입니다. 기능의 중요도와 유형을 9가지 관점으로 분류하여, 목표 커버리지 달성을 위한 테스트를 자동으로 추출하여 우선순위를 제시합니다. 시간을 추가로 확보하는 것보다, 확보한 시간을 잘 사용하는 것이 보다 의미 있는 활동입니다.



08
VISUALIZATION FOR DIFFERENT CHANGES (Next Version)

변경이 발생될 때마다, 처음부터 다시 할 필요가 없습니다. 변경 사항이 비교되고, 그에 따라 영향 받는 항목만 표시함으로써, 효율적 변경 관리와, 변경에 대한 논리적 영향 분석이 가능합니다.

CETA 는 테스트 설계 를 지원하는 세계 유일의 반 자동화 도구입니다.

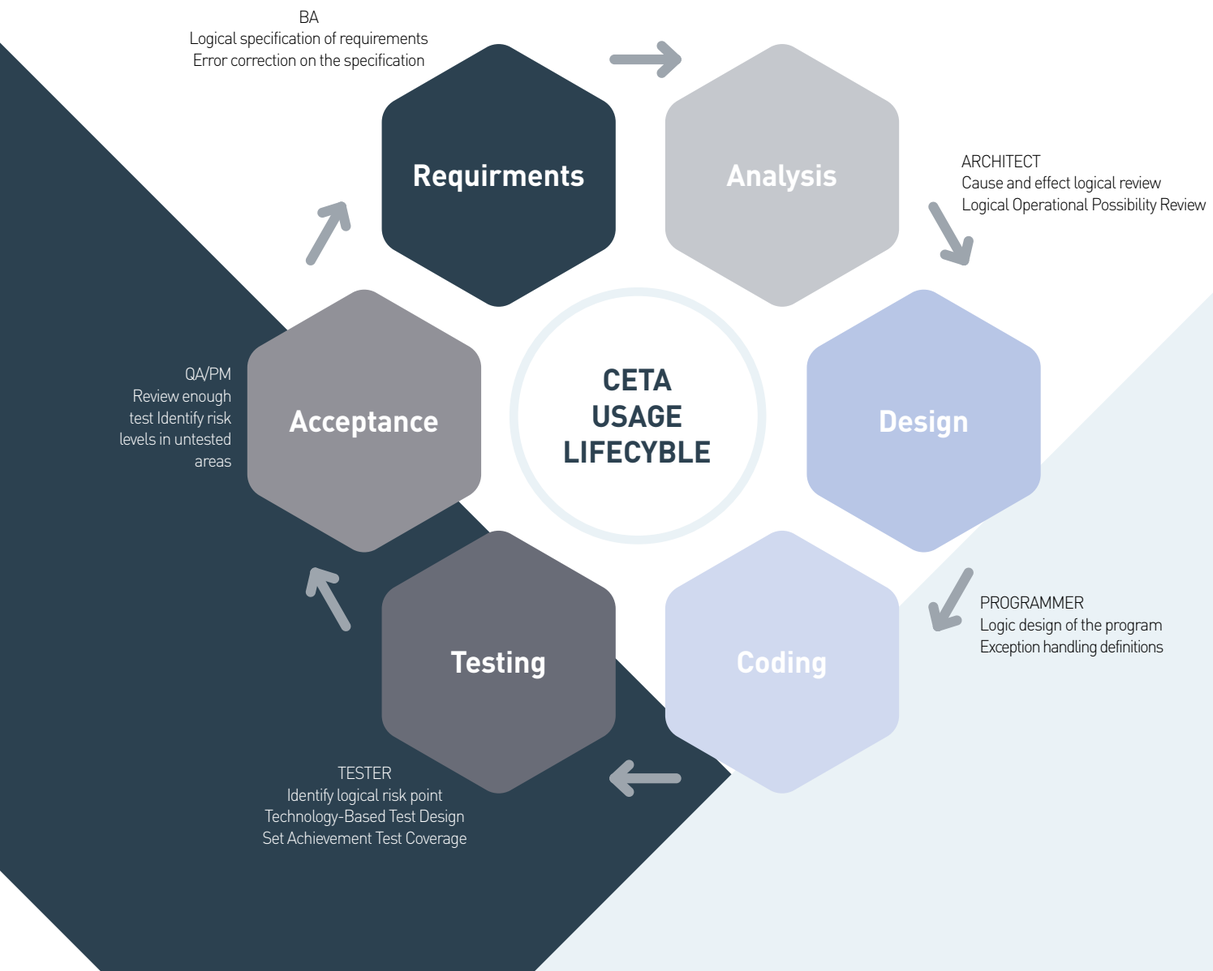
등록
- 소프트웨어 테스트 케이스 생성 자동화 방법 및 장치 (10-1554424)
- 테스트 케이스 생성 방법, 장치 및 컴퓨터 판독가능 기록매체(10-1826628)
- 화상 디자인이 표시된 디스플레이 패널 (30-0905834)
- 전장용 소프트웨어 코드의 안전성 검증을 위한 테스트 기반 정량화 측정 방법 및 장치(10-1899778)
- 소프트웨어 분석 방법 및 장치(10-1706098)
- 소프트웨어 위험분석 방법 및 장치(10-1734418)
- 테스트케이스 설계 정보의 추적 분석을 위한 시각화 방법(10-1826618)
- 소프트웨어 현지화를 위한 테스트 케이스 생성 장치 및 방법(10-1588027)

PCT
- 테스트케이스 설계 정보의 추적 분석을 위한 시각화 방법, 테스트케이스 생성 장치 및 컴퓨터 판독가능 기록매체 (PCT/KR2017/013459)

THINKTOOLS ONE. CETA
DETAILS FROM CETA

How to use CETA

- 시스템에서 중요하거나 복잡한 기능에 대해, 논리적 동적 흐름에 대한 가시적 표현이 가능하며, 명세 상의 오류를 쉽게 발견합니다.
- 예외 처리 필요 영역과, 논리적 위험 영역을 쉽게 발견하여 테스트 계획을 수립할 수 있습니다.
- 기능의 중요도, 구현 난이도 등을 고려하여 목표 검증 수준을 설정하고, 납품, 인수, 출시 시 객관적 기준으로 사용할 수 있습니다.
- 변경이 발생할 경우, 전체 재 테스트가 아닌, 논리적으로 영향을 분석한 선택적 테스트의 타당성을 제시합니다.
- 소프트웨어 개발 및 변경 과정의 SDLC 상에서, 각 이해관계자 간 높은 수준의 논리 표현 커뮤니케이션을 이루게 합니다.



▶ CETA는 테스트 (수행) 자동화 도구인가요?

테스트 수행이 아니라 설계 (반)자동화 도구이므로 수행은 매뉴얼로 해야 합니다. 말하자면 2차 방정식에 계수까지만 넣으면 푸는 건 계산기가 해주는 것처럼, CETA역시 논리적인 부분을 사용자가 작성하면 나머지를 자동으로 설계해 줍니다. 사용자가 '무엇을 만들고 싶은지'의 논리적인 부분은 직접 정해야 합니다. 그러면 나머지는 CETA가 진행합니다.

▶ CETA는 누가 사용해야 하나요? 프로그램 까막눈인 제가 논리 기호를 그릴 수 있나요?

누구든 가능합니다. 가능하다면 테스터가 하는 게 좋습니다. 기획자는 SW가 어떻게 동작하는지, 그리고 어떤 가치를 창출할 수 있는지에 집중하고, 테스터는 어떤 경우에 잘 못하는지를 고민한다면, 더 효율적일 것입니다. 누구든 사용할 수 있지만 기능의 부정적 가능성을 고민하는 이의 손에서 더 잘 활용될 수 있을 것입니다.

▶ 명세서를 처음부터 CETA로 만들면 더 좋을까요?

감기에 걸렸다고 곧장 MRI를 하진 않겠지요? 모든 기능에 다 적용하는 것은 가능하지만 효율적이지 않습니다. 중요하거나 복잡하거나 위험한 부분에 선택적으로 사용하기를 권장합니다.

▶ 시간이 없는데 언제 저것을 다 하나요?

시간이 없기 때문에 효율적 테스트가 필요합니다. 시간이 없다고 시험공부를 안 할 수는 없겠죠? CETA는 짧은 시간 내에 가장 중요한 요소를 점검하고 해법을 제시하는 족집게 선생님입니다. 안전성을 포기할 게 아니라면, CETA를 통해 효율적으로 대비하세요.

▶ 기존에 코드 커버리지를 사용했는데, 뭐가 다른가요?

비유하자면 코드 커버리지는 내가 구현한 코드를 시험 범위로 모의고사를 치르는 셈입니다. 반면에 CETA는 코드의 기능이 발생시킬 수 있는 결과에 대해 가능한 모든 경우의 수를 시험 범위로 합니다. 내가 공부한 범위에서만 모의고사를 칠 경우 당장 점수는 좋게 나오겠지만, 실제 시험이나 현실에 대한 대비는 되지 않습니다. CETA는 내가 공부하지 않은 부분까지 체크해서 확인시켜 줍니다. 당장 커버리지 100%를 만드는 게 중요한 게 아니라, 제품이 실제 출시된 후 문제가 생기지 않게끔 점검하는 것이 테스트의 진짜 목적입니다.

▶ 이 방법은 믿을만 한가요?

기존의 테스트는 방정식을 공식 없이 감각에 의지해서 풀이하는 방식이었습니다. 결과적으로 일부 천재들은 정답을 내놓을 수 있었겠지만 보편적 신뢰성을 얻을 수 없었습니다. CETA는 근의 공식입니다. 테스터에게 탁월한 감각이나 경험이 없이, 있는 공식을 차분히 적용하는 훈련만 되어 있더라도 객관적인 결과물을, 검증 가능하게 내놓는 신뢰성 높은 방식입니다.

▶ 그렇게 훌륭한 기술을 유명하지 않은 기업에서 개발했다는 것이 믿기지 않거든요.

저희도 왜 다른 기업들이 이런 시도와 연구를 안 하는지 믿기지 않습니다. 너무 많은 회사들이 아키텍쳐나 프로그래밍에 조예가 없는 테스터들에게, 오로지 그들의 경험과 감각만 믿은 채 중요한 테스트를 맡기고 있다는 게 믿기 어려운 현실입니다. CETA는 테스트 분야의 거장 제임스 마이어가 1974년에 만든 방법이며, 한동안 발전이 정체되어 있다가 최근 DNF와 연계한 방안이 해외에서 연구되고 있습니다. 저희는 그것을 조금 간소화해서 현장에서 쓸 수 있도록 개선하는 과정에서, 완벽을 추구하기 보다는 효율을 추구하기 위한 정량적 전략을 고안했습니다. 그리고 그것을 자동화된 도구로 개발한 것이 CETA입니다.결국 당장 편하기 위해 당면한 문제를 외면해 왔는지, 아니면 어떻게든 기술적으로 방법을 찾아냈는 지의 차이라고 생각합니다.

▶ CETA의 앞으로의 발전 방향은 무엇인가요?

그 부분은 더 지켜봐야 할 것 같습니다. CETA는 전체 면적을 하나 하나 도구로 측량하기 어려운 이들을 위해 만들어진 적분 계산기입니다. 기업들이 적분 공부가 하기 싫다고 면적을 계산하지 않은 채 지금처럼 눈대중으로 찍어 진행하는 방식을 고수한다면 큰 문제가 발생할 수 있을 것입니다. 가능하면 더 많은 기업들이 테스트 문제로 인한 큰 어려움을 겪기 전에 CETA와 같은 방식을 사용해서 더 안정적이고 효율적인 업계 환경을 만들어주기를 기대하고 있습니다.

최근에는 학습이 빠르고 상대적 비용이 저렴한 베트남 인력들을 대상으로 CETA 방법론을 교육하고 있습니다. 그들과의 협업 속에서 조만간 또 다른 성과가 가시화되길 기대해도 좋을 듯합니다.

THINKTOOLS TWO.DRONACE
WHAT IS DRONACE?

당신의 드론에 탑재된 소프트웨어는 어떤 상황에서도 안전하게 구동될까요?

사람이 직접 원격 조종하던 기존의 드론과 달리, 이제부터 개발되는 산업용 드론은 인공지능에 의해 움직이게 됩니다.
예를 들어 해안 2km 에서 발생된 사고 현장에 구명 튜브를 투하하는 일을 원격 조종으로 처리할 경우, 영상 수신의 어려움이나 전파 장애, 원거리 송수신 딜레이 등의 문제 때문에 오히려 2차 사고를 불러올 수 있기 때문입니다.
산업현장에는 이제 스스로 판단하고 임무를 수행하는 자율임무수행 드론이 등장하고 있습니다.
이렇게 복잡한 임무를 수행해야 하는 드론의 SW에 대해, 센서와 HW 고장률을 진단하는 수준의 기존 시험 방식으로는 그 안전성을 확실하게 검증하기 어렵습니다.

DROANCE는 물리 레벨의 시험 환경을 가상으로 구성하여,
드론에 탑재된 SW의 안전성을 체크하는 유일한 도구입니다.

01 **IEEE 후원 테스트 컨퍼런스**
(ICST - International Conference Software Testing) 논문 발표
2017.03.17 IEEE InSTA 2017 in Tokyo
2019.04.22 IEEE InSTA 2019 in Xian

02 **ASQN - Asia Software Quality Network 기술 발표**
2018.06.29 ASQN 2018 in Beijing
2019.09.09 ASQN 2019 in Tokyo

03 **2019.05.24 The Korean Association of SMART Policing 참여**

등록
- 무인 항공기 비행 기록 저장 방법 및 장치 (10-1703236)
- 무인 항공기 자율비행 제어 검증 방법 및 장치 (10-1769218)

Flight Controller

Sensor (GPS, Temp etc.)

Software

Actuator (Motor etc.)

Virtual Sensor

Value (Serial)

Drone flight Command (Serial)

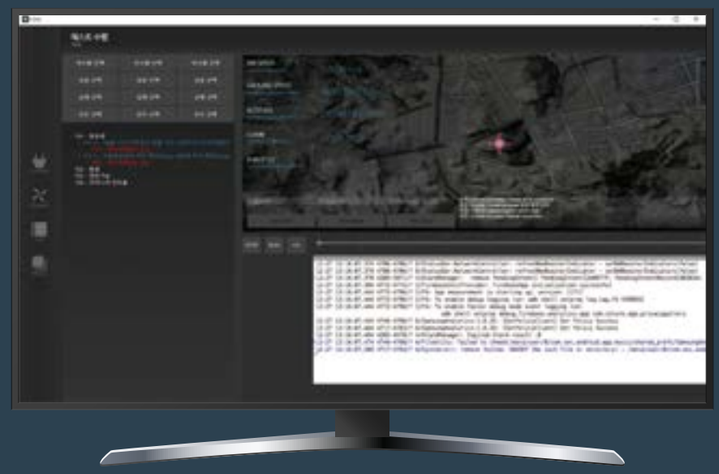
WHY SHOULD YOU USE DRONACE?

시험 환경은 인위적으로 만들 수도,
마냥 기다릴 수도 없습니다.

드론에 탑재된 자율임무수행 SW는 모든 종류의 악천후, 장애 환경 등 돌발상황에 대처해야 합니다. 그런데 그 대처가 안전한지 점검하기 위해서 실제 기후 상황 등 물리 환경을 만들어내는 것은 불가능합니다. 그렇다고 강풍이 불 때까지 기다리거나, 해상바람이나 난기류 등을 재현하는 것 또한 시간적으로 감당할 수 없습니다.
드론이 국경접경지대를 안전하게 회피하는 지를 실제 비행으로 검증한다면, 그것은 군사적 문제 발생의 또 다른 위험이 될 수도 있습니다.

- Flight Controller 의 Stability (비행 제어의 안정성) :
  강풍에도 안정적으로 자세를 제어하는가
- Autonomous Flight 의 Accuracy
  (자율 비행의 정확성) :
  난기류에서도 정확하게 목적지에 도착하는가
- Safety Law 의 Compliance
  (안전 법규의 준수성) :
  비행 금지 구역에 진입할 경우 회피할 수 있는가
- Mission Execution 의 Reliability
  (임무 수행의 신뢰성) :
  주어진 미션을 결과적으로 완전하게 수행하는가

TEST TARGET



드론 탑재 SW 테스트 화면

THINKTOOLS TWO.DRONACE
DETAILS FROM DRONACE

FUNCTIONS & EFFECTS

CPS 소프트웨어 모듈의 자동 테스트를 위해 정의된 7가지 요구 사항입니다.

01 Hooking
- 시험하고자 하는 외부환경의 센서 데이터 (GPS, Acceleration , Angular speed, Magnetic field, Battery) 를 소프트웨어와 센서 사이에 주입
- 드론이 실내에 위치하더라도, 센서에 주입된 데이터를 통해 시험장소를 외부 환경으로 인식하도록 가상화

02 Sniffing
- 탑재된 소프트웨어가 자율비행이나 임무 수행을 위해 액츄에이터(모터)로 보내는 신호를 갈취
- 실제 모터에는 아무런 신호가 전달되지 않아 움직임이 없으나, 드론 SW는 정상적으로 움직인다고 판단하게 함

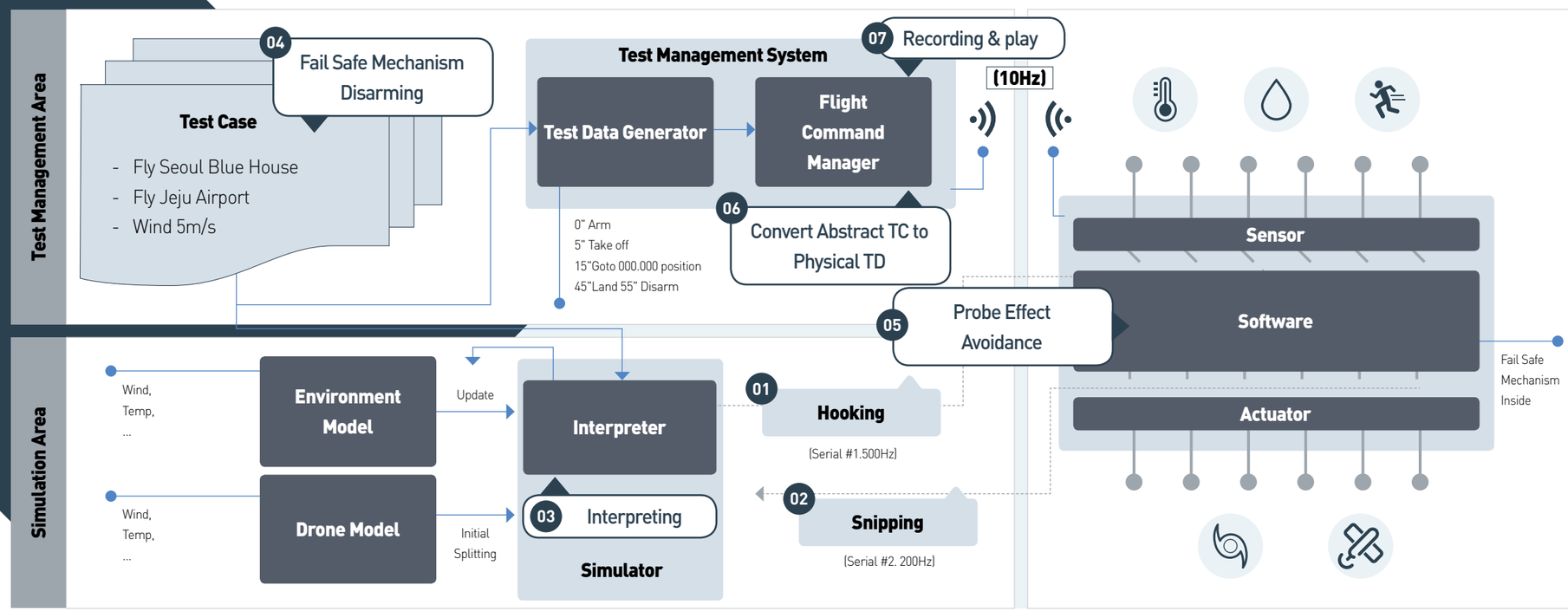
03 Interpret of actuator control raw value
- 기체 유형과 모터 수에 따라 엑츄에이터(모터) 에 보내진 신호를 분석하여 어떠한 동작을 지시하였는지를 해석
- 실제로 드론이 움직이고 있다고 판단할 수 있도록, 해석된 움직임에 대응하는 센서 데이터를 재 생산하여 주입

04 Disarm fail safe mechanism
- 드론에 인위적으로 주입한 센서 데이터를 내재된 오류 감시 기능이 탐지하는 것을 무력화
- 일반적으로 강제 재부팅 ,또는 이전 테스트 시험 환경에서 정상적으로 다음 테스트 환경까지 전개되는 모든 가상 데이터를 주입

05 Avoid Probe Effect
- 성능에 민감한 시험 환경 특성을 고려하여, 가상 테스트 환경이 실제 기체 성능에 미치는 영향을 회피
- 가상으로 주입한 센서 정보와 모니터링 정보의 송수신 과정을 분리하여 별도로 구성

06 Convert the abstract test scenario to physical level virtual data
- 테스트 시나리오의 추상적 언어를 실제 물리 수준의 가상 센서 데이터로 변환하는 모델링 알고리즘
- 해당 테스트 시나리오를 수행하기 위한 시간 단위의 모든 보간 데이터를 생성 (500Hz)

07 Support Test Recording and Replay
- 테스트 결과 확인 및 디버깅을 위한 보조적 기능으로 레코딩 플레이 지원
- 마치 비행 기의 블랙박스와 같이, 당시 비행 상황을 재현할 수 있도록 구성



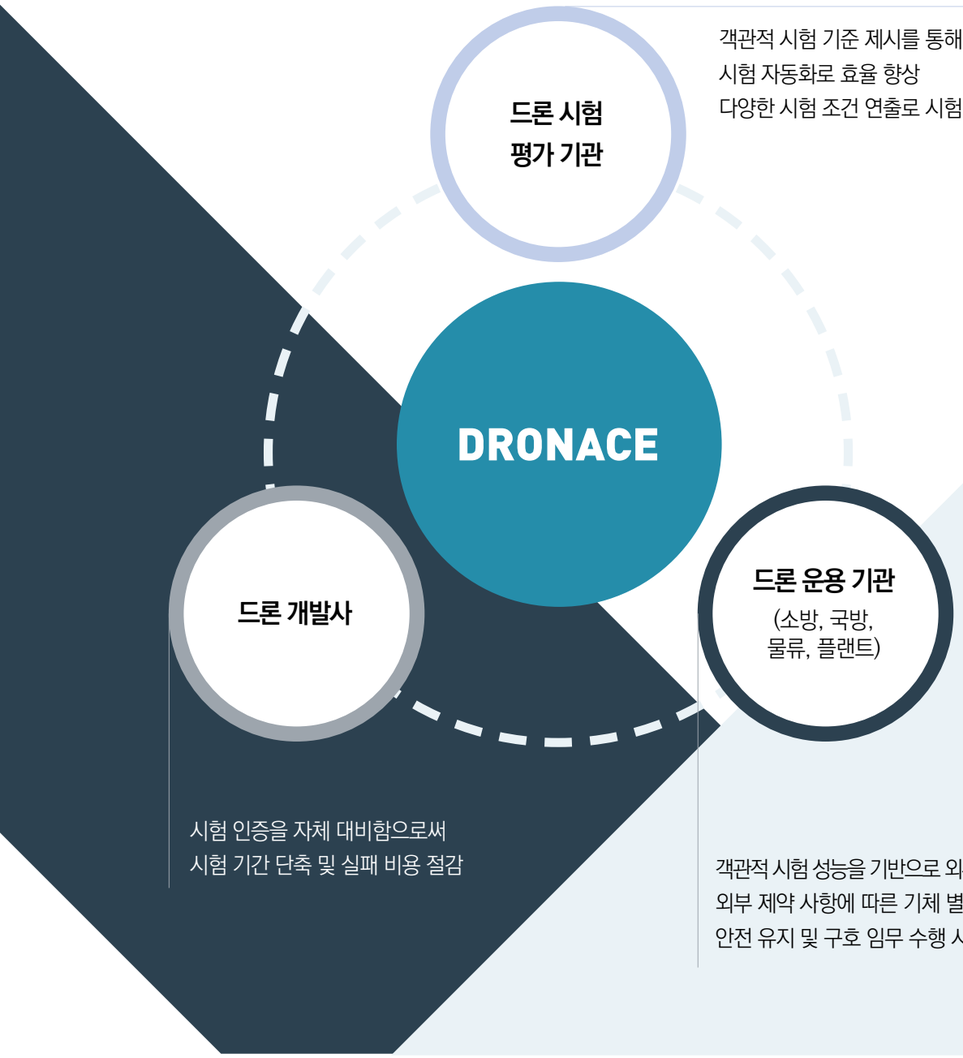
Defined 7 requirements for automated testing of the CPS SW module

"Suggestion of Testing Method for Industrial Level Cyber-Physical System in Complex Environment." (2019), IEEE International Conference on Software Testing, Verification and Validation Workshop (ICST), 2019, pp 148-152

THINKTOOLS TWO.DRONACE
DETAILS FROM DRONACE

How to use DRONACE

더 적은 노력과 시간을 들이면서도 보다 안전한 드론 세상을 만듭니다.



객관적 시험 기준 제시를 통해 신뢰도 및 공신력 제고
시험 자동화로 효율 향상
다양한 시험 조건 연출로 시험 수준 향상

시험 인증을 자체 대비함으로써
시험 기간 단축 및 실패 비용 절감

객관적 시험 성능을 기반으로 외부 제약에 따른 운영 가능 여부 결정
외부 제약 사항에 따른 기체 별 선택적 운영 가능
안전 유지 및 구호 임무 수행 시 검증된 신뢰 수준 확보

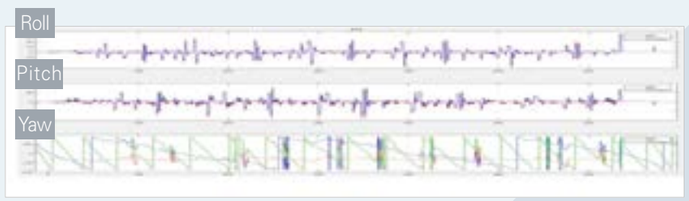
다양한 산업 도메인 별 특화된 드론 모델의 시험을 위한 시뮬레이션 기술 구현

- 신규 개발되는 드론의 산업 도메인 특징별 특화된 스펙을 분석, 테스트 대상 드론을 보다 정밀하게 재 모델링 후 비행 시뮬레이션
- 드론의 길이, 무게, 착용 모터의 속성 등 다수의 가변 요인을 조정하여, 테스트 대상 드론에 대해 정확도 98% 까지의 비행 시뮬레이션 달성

Safety Mechanism Structure – Safe ISO 26262 Deliverable D3.6b

Drone Flight Simulator reflect Weather

시뮬레이션과 실제 비행의 정확도 비교 그래프



THINKTOOLS THREE. SMAC
WHAT IS SMAC?

인공지능이 당신의 전장 코드를 세이프티한 코드로 만들어 줍니다.

세이프티한 코드란 세이프티 매커니즘이 반영된 코드로, 다양한 위험 상황을 식별/회피/회복할 수 있는 코드를 의미합니다.

- Identification of risk
- Avoidance of risk
- Recovery in case of risk

세이프티 전문가들이 손수 진행해야 했던 코드 안전성 확보 컨설팅 대신, 인공지능 기술이 여러분의 코드를 안전하게 완성시켜 줍니다.
스맥의 인공지능은 개인의 주관적 컨설팅보다 더 객관적이고, 전문적이며, 안전합니다.

등록
- 소프트웨어 안전성 분석 방법 및 장치(10-1734872)
- 전장용 소프트웨어 안전성 분석 방법 및 장치-2(10-1834247)

PCT
- 전장용 소프트웨어 안전성 분석 방법 및 장치(PCT/KR2017/006036)

코드 분석 결과 화면

액션 플랜 화면

WHY
SHOULD YOU USE SMAC?

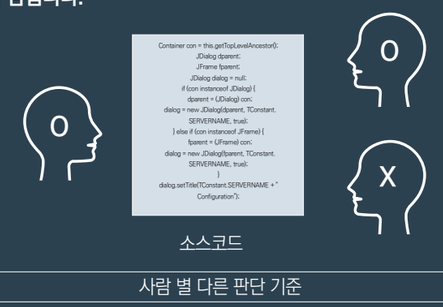
Mission Critical Area에서 실수라는 말은 허용되지 않습니다.

기존의 코드 검사 도구들로는 안전을 보장받을 수 없습니다. 해당 도구들은 이미 구현된 코드만 확인할 뿐 반드시 필요한 안전 메커니즘 코드의 구현 여부를 검증하지 않기 때문입니다. 이를 해결하기 위해 전문가의 컨설팅을 진행하더라도, 개별 컨설턴트의 주관적인 경험과 역량에 의존하게 됩니다. SMAC은 전장 분야의 국제 표준인 Safe ISO 26262 Deliverable D3.6b 의 Safety Mechanism Structure 의 내용을 학습한 인공지능이, 개발된 코드를 분석하여 세이프티 매커니즘이 필요한 위치를 발견하고, 가이드 합니다.

필요한 Safety Mechanism을 찾아주는 SMAC

납품과 검수 단계에서 객관적 평가 기준의 부재는 불편한 마찰을 남깁니다.

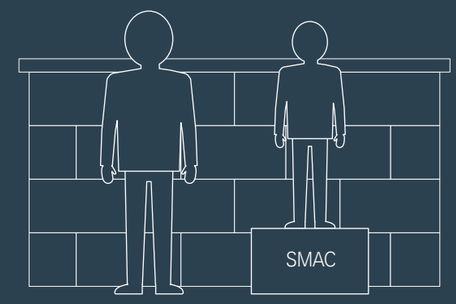
시험 결과가 채점자에 따라 다르다면 어떻게 신뢰할 수 있을까요?
좋은 코드에 대한 정의가 사람마다 다르다면 어떻게 평가할 수 있을까요?
SMAC은 전장분야에서 좋은 코드로 평가된 패턴을 인식하도록 학습하여, 개발된 코드를 평가합니다. 절대적이지는 않더라도 보다 보편적인 기준을 제시하는 데 도움을 줍니다.



사람 별 다른 판단 기준

나오지 않는 고급 엔지니어를 한없이 기다릴 수 없습니다.

아무리 급여를 높이더라도 실력을 향상시키는 데는 상당한 시간이 소요됩니다.
특히나 전장 분야에서는 실수라는 말이 허용될 수 없습니다. 수억 원 대의 인건비를 투입하여 전문 인력을 양성하더라도, 최종적인 결과가 보장되는 것은 아닙니다. SMAC을 사용한 초급 엔지니어라면, 전문가 수준의 안전성을 확보할 수 있습니다.

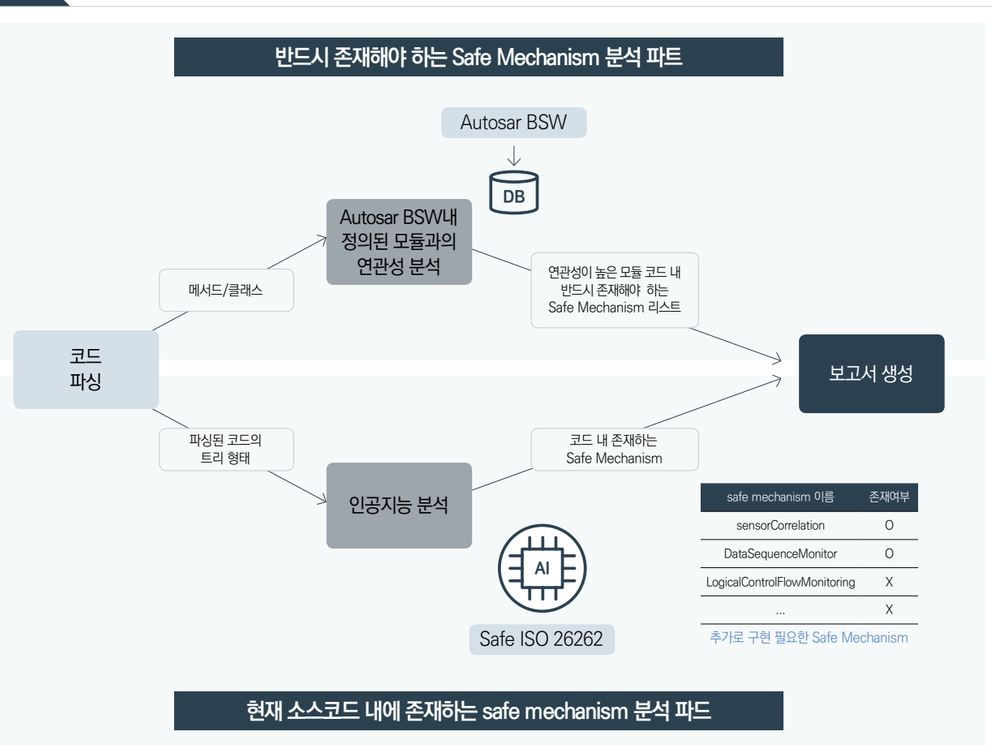


THINKTOOLS THREE. SMAC
DETAILS FROM SMAC

FUNCTIONS & EFFECTS

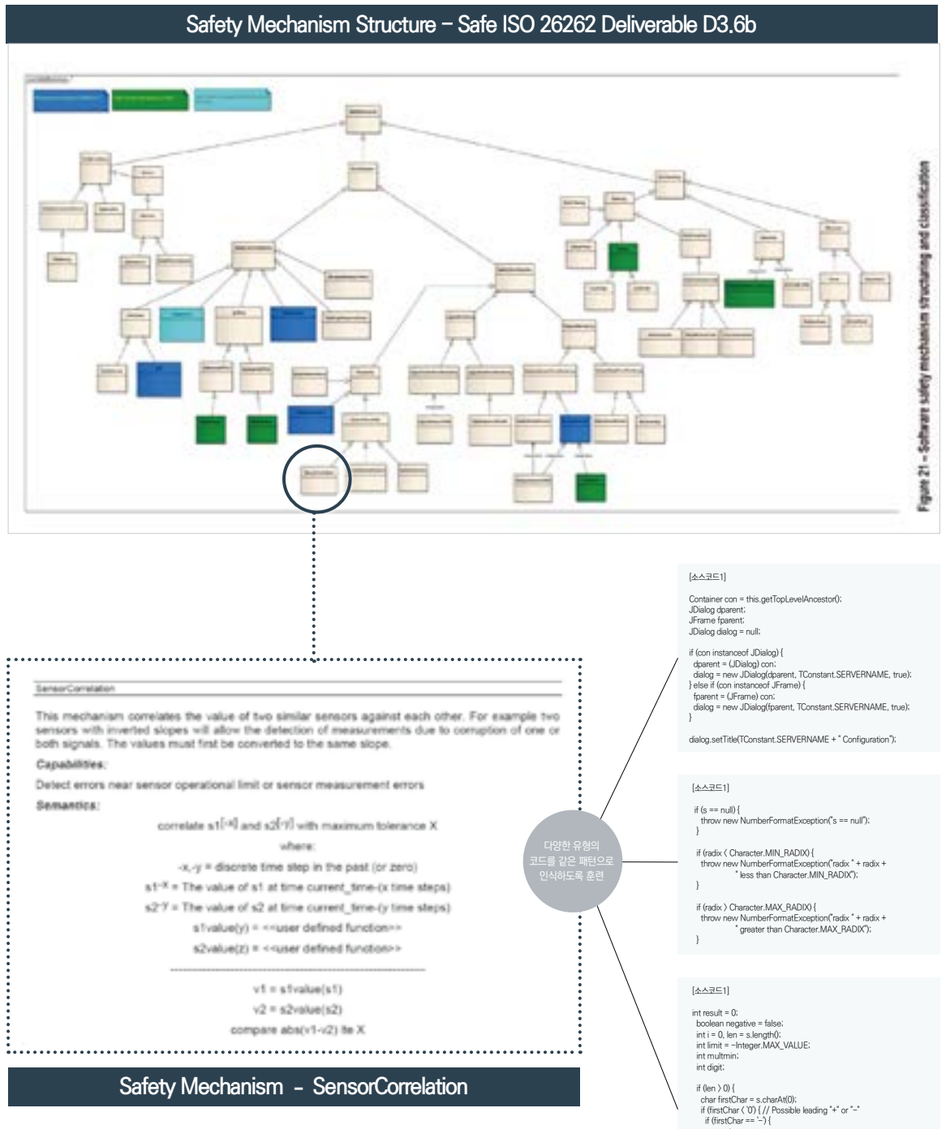
SMAC은 세계 최초의 인공지능 세이프티 메커니즘 분석 도구입니다.

- 전장 SW 내 코드를 분석하여, 세이프티 코드가 되기 위해 현 상황에서 어느 위치에 세이프티 매커니즘이 필요한지를 분석하여 알려줍니다.
- 스맥은 선 학습된 인공지능에 기반하여 소스코드로부터 메서드, 컴포넌트에 필요한 Safety Mechanism의 추가 필요성 여부를 분석합니다.
- 소스코드의 분할된 코드 문자열, 메트릭 변수 정보 등을 모두 입력 받은 후, 해당 클래스 또는 메서드 위치에 어떤 Safety Mechanism이 필요한지를 분석하여 알려줍니다.



AI

여러 유형의 코드를 특정 Safety Mechanism으로 인식할 수 있도록 AI 머신을 훈련하는 과정



Safety Mechanism - SensorCorrelation

THINKTOOLS ANOTHER. KIUWAN
WHAT IS KIUWAN?

500
COMPANIES

25+
COUNTRIES

당신을 돕기 위한 도구가, 때로는 당신의 사업을 방해합니다.
기존의 도구들은 코드의 문제점만 자동으로 발견해 주었습니다.
하지만 현실적이고 실용적인 개선 을 위해서는, 코드의 문제점으로 인한 사업적 영향과 그 의미까지 파악하여 개선방안에 우선순위를 정하는 수정 전략이 반드시 필요합니다.
이러한 전략적 조언 없는 전문가의 컨설팅은 무의미한 수정 비용과 시간 낭비, 출시 지연을 가져와 제품 출시와 사업 진행에 방해가 되는 일이 많았습니다.

KIUWAN 은 검사 도구가 아닌 컨설턴트입니다.
KIUWAN 은 전세계 최고 전문가들의 지식을 기반으로, 코드 수정 전략을 제시하는 컨설팅 도구입니다.

코드 개선 활동	기존 정적 분석도구	키우완
코드분석	도구 활용	도구 활용
문제점 도출	도구 활용	도구 활용
개선방안 수립	전문가 투입 필요	도구 활용
코드 개선	개발자 활용	개발자 활용

〈 코드 개선 활동 과정의 도구와 전문가의 역할 〉

매니지먼트 관점
- 거버넌스
- 라이프사이클

오픈소스 관점
- 라이선스
- 빌드 의존성

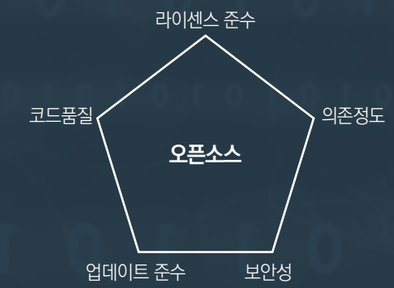
코드분석 관점
- 코드메트릭
- 코드오딧
- 코드구조

보안 관점
- 보안 취약점
- 시큐어코딩

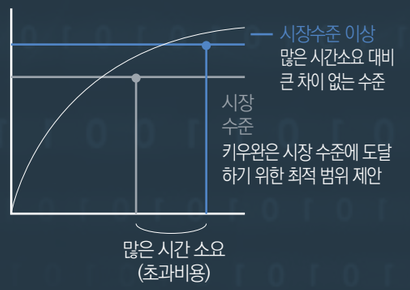
WHY SHUOLD YOU USE KIUWAN?

누구에게 맡겨야 제대로 개발할 수 있을까요?
여러 하청업체들, 업체 내 다수의 개발 부서, 부서에 소속된 수많은 개발자들 중에서 이 기능을 제대로 개발할 수 있는 주체는 도대체 어디일까요? KIUWAN 은 기존 코드의 생산성과 품질을 분석하여, 도메인별, 기능별, 구현 언어별 최적의 개발담당자를 제시합니다.

오픈소스: 라이선스가 전부가 아닙니다. 의존성 때문에 올라가는 비용은 어떻습니까?
개발비용 절감을 위해 사용되는 오픈소스가, 유지보수로 인해 얼만큼의 추가 비용을 요구하는지 분석되고 있을까요?
사용하고 있는 오픈소스가 품질이나 보안에 문제가 없는지 검증이 되었을까요?
오픈소스의 라이선스를 준수하는 것만으로는 개발 과정에서의 문제들을 해결할 수 없습니다.
KIUWAN은 내가 사용하는 오픈소스로 인해 발생할 문제들을 사전에 분석하여 제시합니다.
라이선스, 업데이트 및 노후화 수준, 의존성, 코드 견고함과 보안 수준 등 모든 면이 파악되어야만 '오픈소스가 개발 효율화에 도움이 된다 ' 라고 말할 수 있습니다.



수정할 시간이 없다고요? 주어진 시간을 잘 활용하는 것도 중요한 전략입니다.
제품은 잘 만들기 위해 만드는 것이 아니라, 잘 팔기 위해 만드는 것입니다. 내가 수정한 코드가 제품을 잘 파는 데에 얼마나 영향을 미치는 지가 비용대비 효과로 입증되어야 합니다.
무조건 '더 잘 만들기' 위한 무한 품질 비용은 오히려 사업에 방해와 위협이 될 수 있습니다.
KIUWAN은 사업 성공을 목적으로 한 품질 목표를 5가지 관점에서 분석하여, 최대 효율을 시간 단위로 계산 후 우선순위를 세우고, 그에 따른 수정 전략과 계획을 제시합니다.



품질에 대한 자기만족을 채우기 위해 골든 타임을 놓칠 수는 없습니다. 품질을 위해 시간과 비용을 쓰는 것이 아니라, 시장 상황에 맞는 제품을 내놓기 위하여 품질을 맞추는 것이 중요합니다.

다 같은 시큐어 코딩이 아닙니다. KIUWAN은 OWASP 최고 점수를 얻은 도구입니다.
KIUWAN은 현존하는 도구들 중 OWASP Benchmark 에서 55% 이상의 탐지율을 달성한 유일한 보안 검사 도구입니다. (평균 26%)

THINKTOOLS ANOTHER. KIUWAN
DETAILS FROM KIUWAN

CODE ANALYSIS (QA)

GAIN UNPARALLELED SCOPE WITH EXTRAORDINARY EASE

Application VULNERABILITY ASSESSMENT

Place automatic audits on application changes to enforce security

ACTION PLANNING BASED ON QUALITY CHRACTERISTICS

- 5개의 품질 특성 (ISO 9126) 에 코드를 연계하여 품질 상태 지표화
- 사용자 품질 목표 달성을 위한 최적의 전략 제시
- 수정 위치 및 해결 방법의 가이드

IMPLEMENT ASSESSMENT OF THE SECURITY RISKS IN YOUR APPLICATIONS

- 개발된 코드 및 외부 제공 모듈의 취약점 발견
- 완화 및 개선 전략 수립을 제시하는 유일한 자동화 도구

SAME TOOL FOR EVERY STAKEHOLDER

- 보안 전문가, 설계자, 개발자, 테스터 모두에게 유용한 동일 사용 플랫폼 지원
- IDE 와의 통합으로 빠른 위치 추적 및 해결 방법 제시

RISK IMPACT & RETEST REPORT

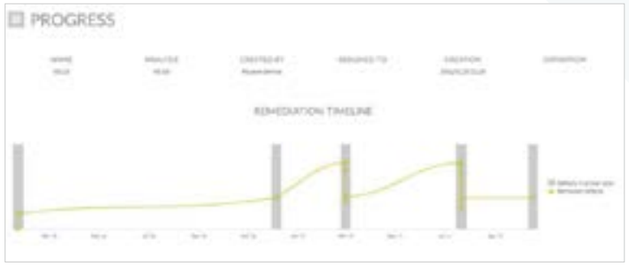
- 취약점 해결 위치 파악, 해결 과정에서 주변 기능에 미치는 영향 및 의존성 분석
- 그에 따른 기능 보장을 위한 테스트 계획 수립 지원

Beyond code analysis

- 발견 된 주요 결함 수정에 필요한 노력 관리
- Visual configurator 로 분석 규칙과 속성 선택
- 자동 또는 수동의 수정 계획 수립, 자동화 모니터링
- 지속적 분석을 위한 Jenkins 등의 CI 와 통합

- 분석 내역에서 각 버전 별 결함의 발생 및 수정을 확인하는 비교 보고서
- 다른 도구와의 통합 (PMD, Findbugs, Chekstyle..)
- 편리한 분석 범위 설정을 위한 애플리케이션 포트폴리오 그룹화 또는 필터링
- 당신만의 코딩 규칙 개발을 위한 Kiuwan SDK 제공
- 외부 도구의 분석 결과 (결함 메트릭)를 반영, Kiuwan 의 분석 결과에 통합 (결함 제거, 실행 계획, 규칙 설정 등)
- 경영 관점의 의사 결정을 위한 보고서
- JiRA 에 수정 계획 연동 (또는 PDF, CSV)

Defects Remediation Timeline



Summary

Governance Dashboard

Defects distribution + filtering

For all major languages

20+
INCLUDING:

IDE
- Visual Studio
- eclipse
- ...

BUILD SYSTEMS
- Jenkins
- Bamboo
- Visual Studio TEAM FOUNDATION SERVER
- ...

BUG TRACKING
- JIRA
- Visual Studio TEAM FOUNDATION SERVER
- ...

REPOSITORIES
- Bitbucket
- GitHub
- GitLab
- ...

THINKTOOLS ANOTHER. KIUWAN
DETAILS FROM KIUWAN

CODE SECURITY (SAST)

SHIELD YOUR APPLICATIONS, ELIMINATE SECURITY VULNERABILITIES

APPLICATION SECURITY ARCHITECTURE

ARE YOU AWARE OF THE IMPACT OF SECURITY IN YOUR BUSINESS?

OWASP BENCHMARK NO.1
- 모든 국제 표준 기반의 보안 요구사항 충족
- OWASP 벤츠마크에서 55% 탐지 결과 (상용 제품 평균 26%)

BUSINESS RISK EVALUATION
- 비즈니스 위험 평가를 위한 보안 취약점 발견 지원

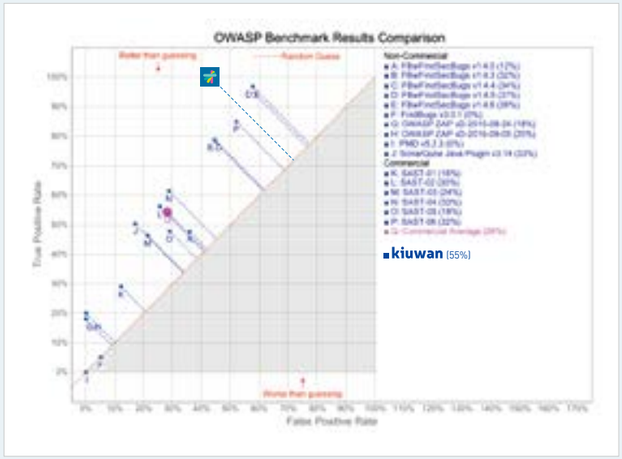
IMPACT OF VULNERABILITIES ON PROJECTS
- 취약점 해결 과정이 발생시키는 또 다른 영향을 분석
- 변경에 따른 기능 보장을 위한 테스트 계획 수립 지원

APP SECURITY ARCHITECTURE MANAGEMENT
- 프로젝트 종속성 구조 및 계층의 이해, 이에 따른 보안 아키텍처 유지 지원

INTEGRATED IDE FOR CONTINUOUS DEVELOPMENT
- 개발과정에서 지속적으로 삽입되는 취약점의 용이한 탐지를 위한 개발 프로세스와의 통합

Kiuwan,
모든 OWASP Top 10
취약점 100 % 탐지

OWASP BANCHMARK NO.1



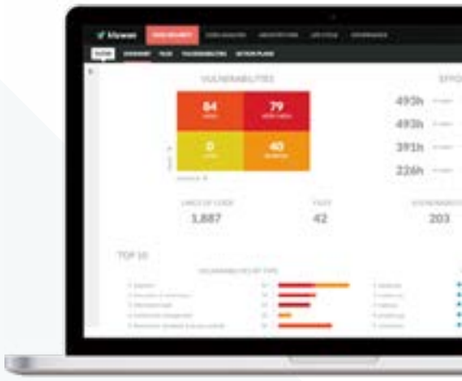
Some detected vulnerabilities

- Uninitialized Variables
- Application Misconfiguration
- Credential/Session Prediction
- Directory Indexing
- Insufficient Authorization/ Authentication
- Automatic Reference Counting
- Cross Site Request Forgery
- Information Leakage
- Insufficient Binary Protection
- Insufficient Transport Layer Protection
- Cross Site Scripting
- Injection Attacks
- Interprocess Communication
- OS Commanding
- Insecure Cryptography
- SQL injection
- Cryptographic Related Attacks
- Buffer Overrun
- Free Non-Heap Variable
- Use After-Free
- Double Free/Close
- Format String Vulnerability
- Return Pointer To Local

. . .

For all major languages

OWASP · MISRA · PCI · CWE COMPATIBLE · CERT · NIST



THINKTOOLS ANOTHER. KIUWAN
DETAILS FROM KIUWAN

INSIGHTS (SCA)

MANAGE YOUR COMPONENTS & LIBRARIES

Dependencies

- 오픈 소스 프로젝트에는 종종 버전에 따라 배포 환경의 업데이트로 인한 종속성의 영향을 받게 됩니다.
- 대다수의 개발자가 방대한 오픈 소스를 활용하는 과정에서 종속성을 격리하는데 어려워 합니다.
- Kiuwan Insights는 프로젝트의 아키텍처를 주적하고, 코드 품질을 모니터링하여 개발자가 코드 기반의 종속성을 낮출 수 있도록 가이드 합니다.
- 완전히 종속성을 제거할 수는 없지만, 개발자가 코드 이해 수준을 높인다면, 보다 효과적으로 해결됩니다.

License Compliance

- 개발자는 오픈 소스를 가용하여 소프트웨어를 복제, 수정, 배포 및 판매 할 수 있습니다.
- 하지만, 법적 요소를 준수하는 사용 지침을 이해하는 데 있어서 종종 어려움을 느낍니다.
- Kiuwan Insight 는 오픈 소스를 활용하여 개발한 프로젝트에서 사용된 라이선스를 준수하는 가이드를 제시합니다.

In a nutshell

▷ **Components inventory**
빌드 환경을 분석, 사용된 모든 오픈소스와 타사 컴포넌트의 완전하고 정확한 인벤토리 생성

▷ **Detect security threats**
오픈 소스 구성 요소와 관련된 보안 위험 조사 및 탐지

▷ **Avoid obsolescence**
업데이트, 버전 및 보안 문제가 있는 라이브러리의 폐기 관리. 더 이상 사용되지 않는 요소의 알림

▷ **Eliminate time consuming**
새로운 보안 취약점 발견이 미치는 영향이나 라이선스 문제를 확인하기 위한 과정의 인벤토리의 수동 컴파일 소요 시간 단축 및 오류 제거

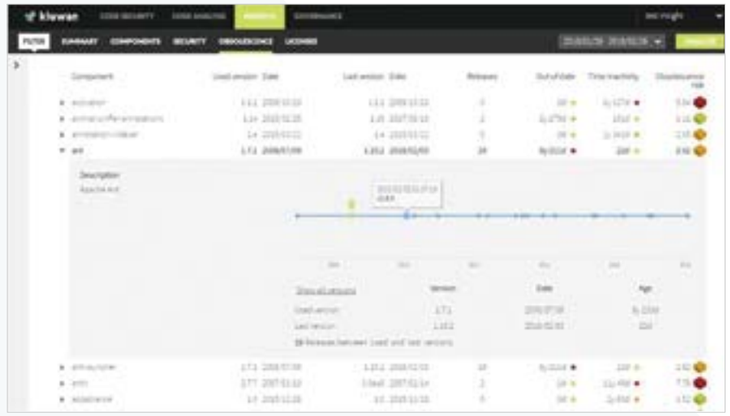
▷ **Unveil security risks**
오픈 소스 구성 요소와 관련된 보안 위험 탐지 후, 프로젝트가 영향을 미치는 시점에 해결 시도

▷ **Isolate dependencies**
사용하지 않는 오픈 소스가 일으키는 배포 이슈를 해결하기 위하여 사용하지 않는 코드를 식별 후 제거. 종속성 및 의존성 문제 위험 해소

Security indicators & alerts ▷

Obsolescence indicators & alerts ▷

Release Timeline ▷



THINKTOOLS ANOTHER. KIUWAN
DETAILS FROM KIUWAN

GOVERNANCE

EXECUTIVE OVERVIEW ALL THE WAY TO THE DEEPEST INSIGHTS

APPLICATION RISK GOVERNANCE

Know the security risks in your Enterprise PROJECTS

PERFORMANCE VISIBILITY OF YOUR PROJECT

- 측정된 데이터 기반 외부 공급 업체와 사내 개발팀의 생산성과 위험 지표의 가시화 및 깊이 있는 통찰
- 팀, 조직, 외부 공급 업체 중 가장 안정적이고 빠른 개발 적임자 선정 가이드

BREAKDOWN BY BUSINESS CRITICALITY

- 비즈니스의 중요도, 타사 제공, 자체 개발 등을 고려하여 프로젝트 정책을 설정하여 위험 수준 파악

EXECUTIVE OVERVIEW TO MAKE DATA-DRIVEN DECISIONS

- 출시 시점, 위험 수준, 심지어 보안 요소를 위반한 공급 업체에게 요청할 계약 항목 도출 등을 관리하기 위한 프로젝트 인벤토리 생성

GOVERNANCE DASHBOARDS

▷ **Decision quadrants**
위험한 프로젝트 탐지

▷ **Evolution**
프로젝트의 개선 예측

▷ **Activity**
개발 팀, 외부 업체의 모든 활동 기록

Beyond code analysis

- 프로젝트 별 정보 필터링 및 그룹화
- 중요도 높은 비즈니스 위험 분석
- 외부 공급 업체, 사업 분야, 기술 분야 별 제품의 중요 정보 비교
- 의사 결정 사분면의 비교 기반 위험성 탐지
- 비즈니스 위험, 생산성 위험, 유지보수 위험, 보안의 잠재적 취약 프로젝트 발견
- 프로젝트 개선 분석으로, 문제 소지의 사전 예측
- 개발 및 유지보수 프로젝트의 변경 요청 활동 기록
- 외부 공급 업체로부터의 반려 수 비교
- 외부 공급 업체의 산업 표준 준수 편차 비교
- 특정 시점의 상황을 정확하게 파악하기 위한 전체 내역 기록화
- 사용자 별 권한 및 역할 지정
- 기업 거버넌스 리포트 (PDF)
- 서비스 수준 계약 (SLA)에 포함될 준수 항목 정의 및 여부 확인
- 기간 별 각 팀 구성원의 생산성 수준 측정 및 비교

We love technology,
but not as much as being human.

모든 SW의 코드에 명석한 뇌를 심고
모든 SW의 기능에 건강한 심장을 뛰게 하여
드론의 날개가 믿음으로 날게 하고
자동차의 눈과 귀가 사람을 지키게 합니다.
당신의 더 나은 삶을 위해서

Think for a better life

ThinkforBL

SEOUL

LS BLDG., 6F, 15,
Nonhyeon-ro 87-gil,
Gangnam-gu, Seoul,
Korea

TEL
+82-2-562-6545
FAX
+82-2-562-6549

JEONJU

Deoksoo BLDG 5F,
20, Hongsan 1-gil,
Wansan-gu,
Jeonju-si,
Jeollabuk-do,
Korea

TEL
+82-63-246-6545
FAX
+82-63-246-6546

SINGAPORE

8 Shenton #04-01
AXA Tower,
Singapore
068811

VIETNAM

Y5 Office,
Hai Ba Trung Street,
Ben Nghe ward,
District 1,
Ho Chi Minh City,
Vietnam 700000

www.thinkforbl.com
E-mail contact@thinkforbl.com