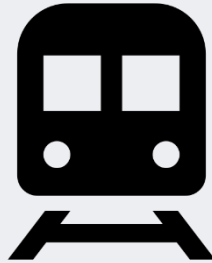


A Practical Technology for Safe World in Suresofttech

지능형테스트자동화실 소개

SURESOFIT

Software for safe world



SURESFT

Software for safe world

SW검증 자동화
원천기술 개발

SW 검증 자동화로
SW 품질 및 생산성 혁신 달성

SW 3자
검증·개발 서비스

SW 검증·개발 전문 인력 및
20년 업력 노하우 보유

SW
컨설팅·교육

SW 검증·개발인력 양성 및
선진 기술 전파

SURESFT

Software for safe world

SW검증 자동화
원천기술 개발

SW 검증 자동화로
SW 품질 및 생산성 혁신 달성

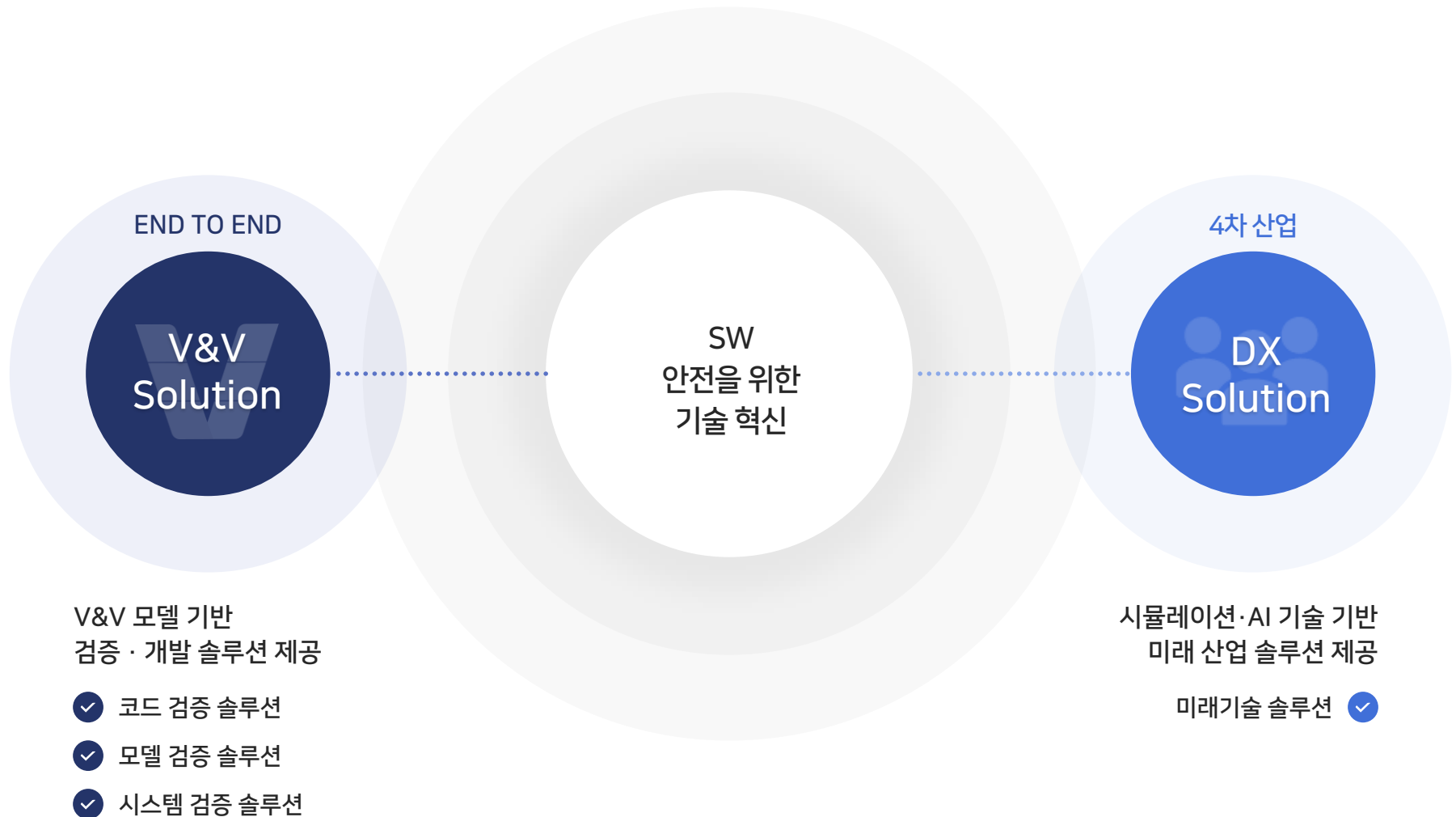
SW 3자
검증·개발 서비스

SW 검증·개발 전문 인력 및
20년 업력 노하우 보유

SW
컨설팅·교육

SW 검증·개발인력 양성 및
선진 기술 전파

디지털 시대 SW 안전을 위한 **ONE STOP 솔루션**을 제공합니다.



V&V Solution

V 모델 처음부터

끝까지!

요구사항 분석
및 설계

시스템 시험

아키텍처 설계

통합 시험

단위 설계
및 구현

단위 시험

코드 검증 자동화 도구

STATIC

코드 정적 검증 도구

CONTROLLER TESTER

코드 동적 검증 도구
(DISCOVERY 테스트 케이스 자동 생성 플러그인)

COVER

테스트 커버리지 측정 도구
(Standalone / Cloud 버전)

VPES

프로세스 리포팅 도구

V-SPICE

ASPACE 인증 지원 도구



모델 검증 자동화 도구

MODEL INSPECTOR

모델 정적 검증 도구

MODEL VERIFIER

모델 동적 검증 도구



시스템 검증 자동화 도구

FIT

결함 주입 시험 도구

PROV

타이밍/자원량 측정 도구

AUTORACT

통신 검증 도구

AESOP

룸 데이터 관리 도구



Tool

자동화 기술 고도화로 디지털 시대 SW 품질 및 생산성 혁신을 실현합니다

가상화 자동화 도구

SIMVA

제어기 가상화 검증 도구

AUTOSIM

시뮬레이션 검증 도구



자율주행 자동화 도구

ADAS INSPECTOR

데이터 기반 결함 검출 도구

VISTA

커넥티비티 시험 도구



사이버보안 자동화 도구

FUZZIT

퍼징 도구



STATIC

- 실제 실행 전 코드 작성 단계에서의 검증
- Runtime Error, Coding rule violation 검출

Controller Tester

- Unit/Integration Test를 위한 Test Driver/Test Case 자동 생성
- TC 자동 수행 및 Coverage 측정

COVER

- 편리한 커버리지 측정
- Cloud Service를 통한 협업의 생산성 증대

VPES/V-SPICE

- SW 개발 Process 검증 도구
- Process에 생소한 개발자도 보다 쉽게 접근 가능

정적분석기술팀

- White box testing 기술의 상용화 및 고도화

동적분석기술팀

- Network black box fuzzing 기술 개발

AI기술팀

- LLM을 활용한 분석 기능 확장 및 AI 모델 검증

- STATIC 도구 내 의미 분석 엔진의 기반 기술
- 소프트웨어 실행 단계 이전에 사용자가 작성한 소스코드를 분석
- C/C++ 소스코드 상에서 발생하는 런타임 에러(Runtime Error) 탐지

- Buffer Overrun/Underrun
- Type Overrun/Underrun
- Divide by zero
- Memory Leak
- NULL Pointer Dereference
- Use After Free
- Unused Variable
- Unreachable Code
- Etc...

요약해석 기반 엔진(Abstract Interpretation)

복잡한 코드 구조에 최적화된 고정점 계산 알고리즘 탑재

깊은 메모리 모양 분석(Shape Analysis)

다양한 관계형 변수 데이터 추적(Relational Abstract Domain)


보안 결함 탐지를 위한 오염분석 엔진(Taint Analysis)

다양한 사전분석 엔진 탑재(Pre Analysis)

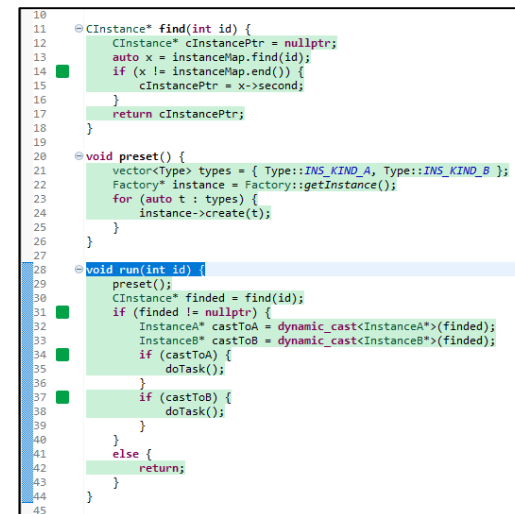
런타임 호출 관계 등을 고려한 링킹 단위 분석



- Controller Tester의 Test Case 자동 생성 Plugin
- Binary 대상 분석으로서 언어 및 실제 실행 환경에 의존하지 않는 분석 기술
- 커버리지 뿐 아니라 버그 탐지에 유용한 Test Data 생성 기대
- Test Driver 고도화를 통한 효용성 증대



```
1 #include "Factory.hpp"
2 #include <vector>
3
4 using namespace std;
5 using namespace global;
6
7 void doTask() {
8 }
9
10 CInstance* find(int id) {
11     CInstance* cInstancePtr = nullptr;
12     auto x = instanceMap.find(id);
13     if (x != instanceMap.end()) {
14         cInstancePtr = x->second;
15     }
16     return cInstancePtr;
17 }
18
19 void preset() {
20     vector<Type> types = { Type::INS_KIND_A, Type::INS_KIND_B };
21     Factory* instance = Factory::getInstance();
22     for (auto t : types) {
23         instance->create(t);
24     }
25 }
26
27 void run(int id) {
28     preset();
29     CInstance* finded = find(id);
30     if (finded != nullptr) {
31         InstanceA* castToA = dynamic_cast<InstanceA*>(finded);
32         InstanceB* castToB = dynamic_cast<InstanceB*>(finded);
33         if (castToA) {
34             doTask();
35         }
36         if (castToB) {
37             doTask();
38         }
39     }
40     else {
41         return;
42     }
43 }
44
45 }
```



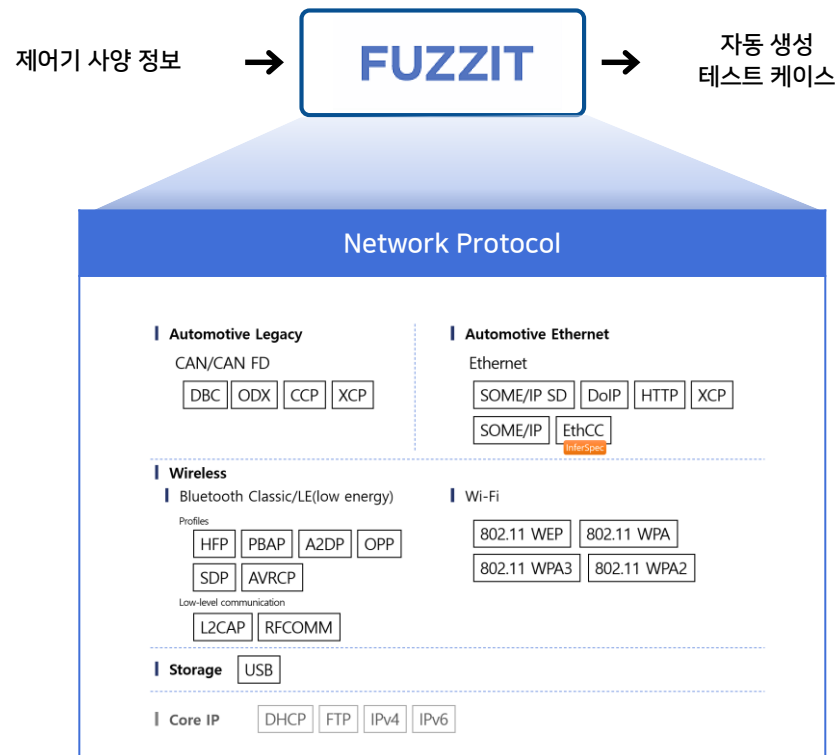
```
10 CInstance* find(int id) {
11     CInstance* cInstancePtr = nullptr;
12     auto x = instanceMap.find(id);
13     if (x != instanceMap.end()) {
14         cInstancePtr = x->second;
15     }
16     return cInstancePtr;
17 }
18
19 void preset() {
20     vector<Type> types = { Type::INS_KIND_A, Type::INS_KIND_B };
21     Factory* instance = Factory::getInstance();
22     for (auto t : types) {
23         instance->create(t);
24     }
25 }
26
27 void run(int id) {
28     preset();
29     CInstance* finded = find(id);
30     if (finded != nullptr) {
31         InstanceA* castToA = dynamic_cast<InstanceA*>(finded);
32         InstanceB* castToB = dynamic_cast<InstanceB*>(finded);
33         if (castToA) {
34             doTask();
35         }
36         if (castToB) {
37             doTask();
38         }
39     }
40     else {
41         return;
42     }
43 }
44
45 }
```

- Universal Parser
 - C/C++ 코드를 읽어내는 파싱 기술
 - 약 200여 종 이상의 컴파일러를 설정하고 파싱할 수 있는 분석 엔진
- 코드 삽입
 - 원본을 훼손하지 않는 코드 삽입 기술

- Visual Studio
- GCC
- Clang
- IAR
- NXP
- Intel
- Etc...



- 자동차 제어기의 잠재적 보안 취약점 검출을 위한 네트워크 퍼징 검증 자동화 도구 개발
- 통신 인터페이스 사양을 기반으로 테스트 케이스를 자동 생성하여 퍼징을 수행
- 다양한 프로토콜을 지원 및 더 나은 사용 경험을 제공하기 위해 개발 진행 중



■ LLM 활용 정적 분석 기능 개선 및 확장

- 정적 결함 수정
 - ChatGPT를 활용한 코드 수정 제안
 - AutoGPT를 활용한 자동 수정 Workflow
 - 정적 결함 수정에 적합한 Prompt Engineering
- OpenLLM 튜닝
 - 성능 평가 및 자체 튜닝 모델로 대체
- LLM 경량화
 - 자사 제품 내장을 통한 사용성 향상 기대



- 적대적 데이터 생성 기술
 - 딥러닝 모델의 오동작 유발 데이터 생성
 - 모델의 강건성 검증
- Neuron Coverage 향상을 위한 Dataset 생성
 - 모델의 테스트가 충분히 이루어졌는지 측정
- 생성 데이터의 자연성 측정
 - 인위적으로 생성된 데이터는 맥락에 맞지 않는 데이터일 수 있음
 - 현실적인 데이터셋 생성 기술 개발

INVESTOR RELATIONS 2023

SURESOFT

SOFTWARE FOR SAFE WORLD



감사합니다