자유와 활력이 넘치는
파워풀 대구

FIX All on AI
Future Innovation tech eXpo Conference 2025

Enhancing SDV Development Efficiency through SILS – CI/CT Support Strategy

아이비스 윤승현 이사

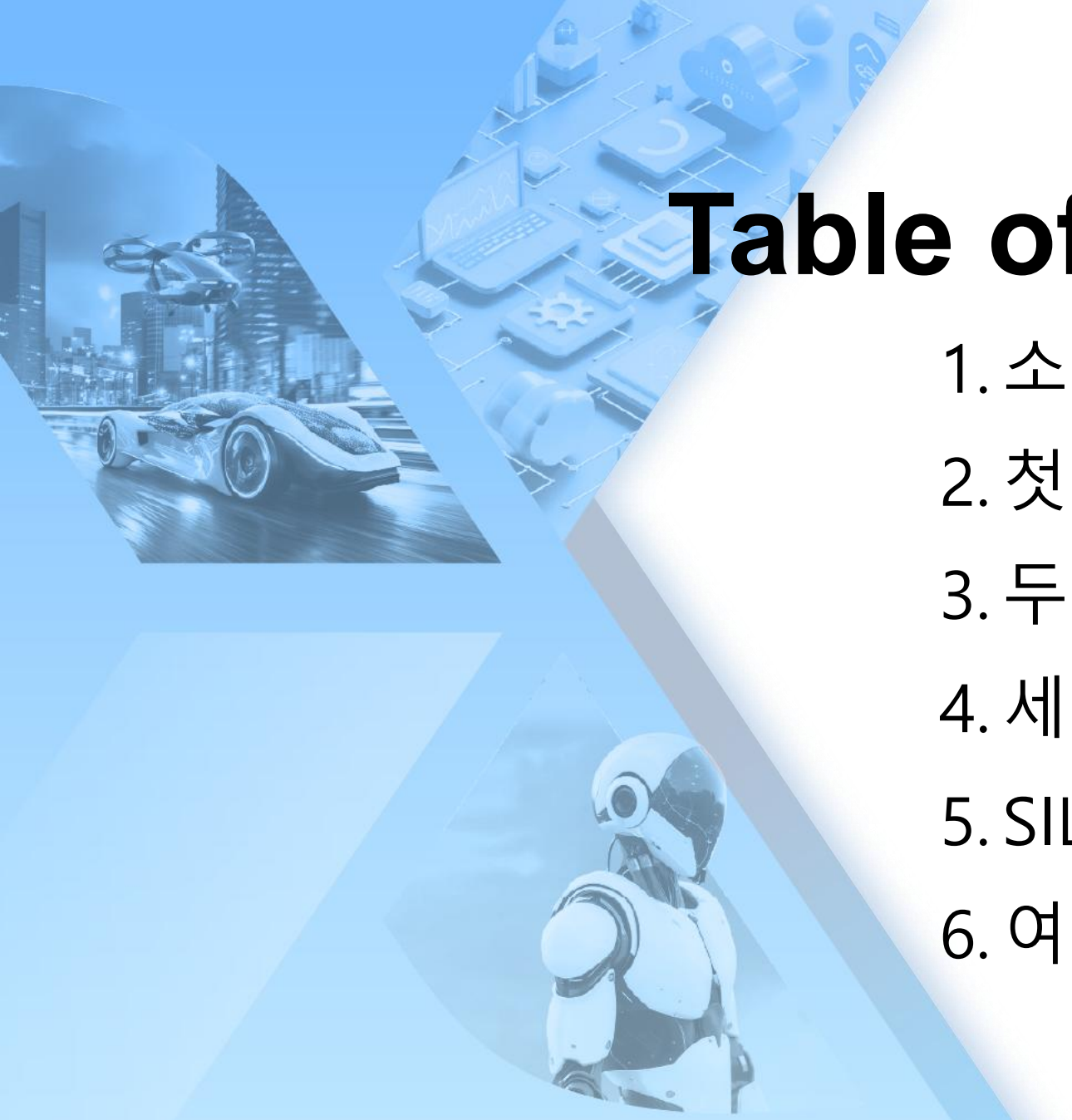


Table of Contents

About ivis

We define
Future mobility software in vehicles

Founded in 2010

2010년 창립 이후 15년 동안 꾸준히 성장, 차량용 소프트웨어 개발사로서 시장 선도

Automotive Embedded Software

차량용 소프트웨어의 차별화된 기술력 기반, 소프트웨어를 중심으로 진화하는 SDV실현.

Software Platform Design Expertise

소프트웨어 플랫폼 설계의 전문 기술 확보,
고객 요구에 맞춰 설계된 플랫폼을 통해 합리적이고 안정된 소프트웨어 공급

90% Inborn Engineers

소프트웨어 개발 인력을 지속적으로 발굴하고 개발 전문가로 양성,
개발자 출신의 다양한 인재들과 함께 미래 모빌리티를 위한 기술을 실현

인포테인먼트 시스템

디지털 클러스터 시스템

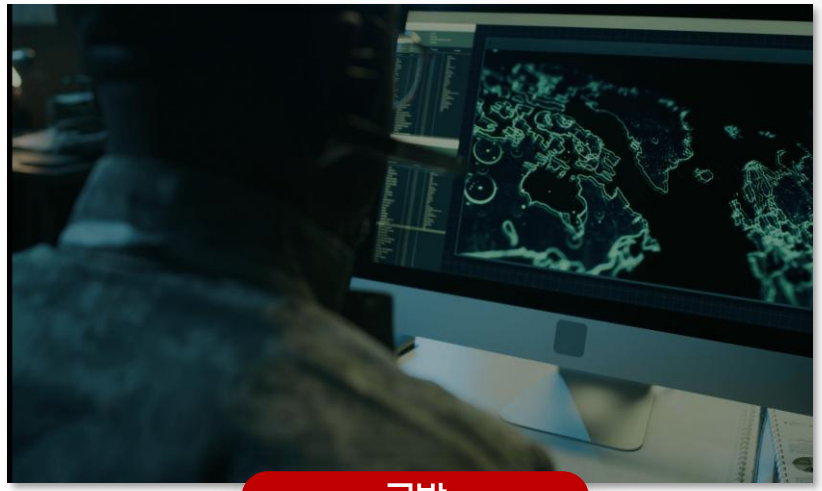
About ivis

미래 모빌리티 산업으로 확장

무인이동체

FMS

On-device AI



국방

디지털 클러스터


100
120
80
140
60
170
40
200
20
230
km/h
260
E
F
3
4
2
5
1
6
7
x1000rpm
8
C
H
0
10
20
30
545 km
-33°C
--.- km/L
1234 km

직면했던 문제들

첫 번째 문제점: CAN 시뮬레이션 장비 부족

직면한 문제

- 디지털 클러스터 = CAN 입력 → 화면 표시
- 개발에 CAN 시뮬레이션 장비 필수
- 전문 장비 비용: **수천만 원/대**

현실적 제약

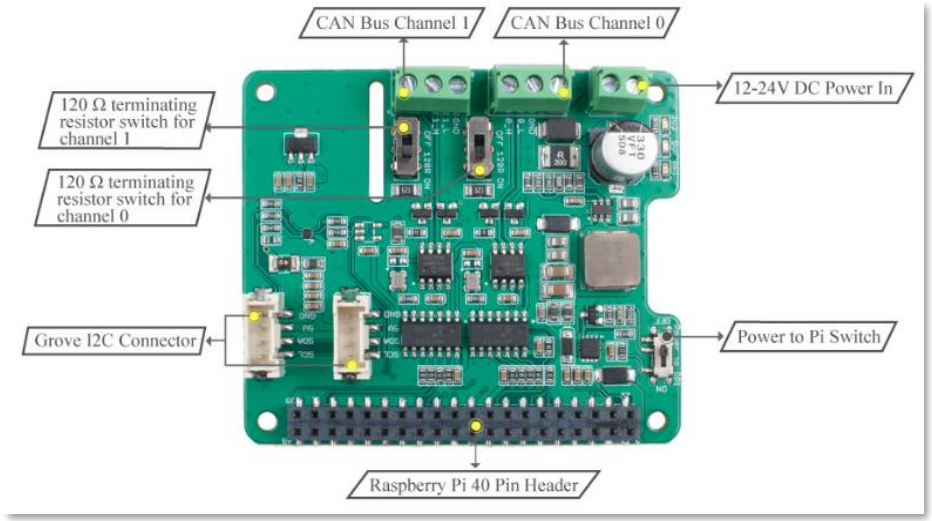
- 모든 개발자에게 1대씩 지급 불가능
- 장비 대기 시간 → 개발 효율성 저하

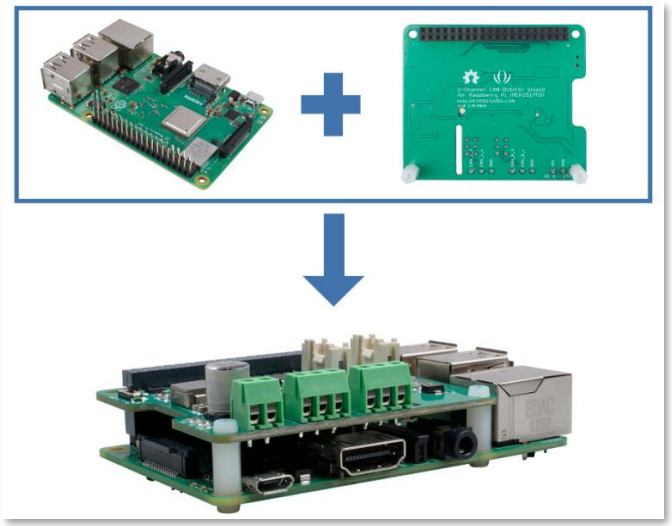
테스트 장비 부족은 업계의 공통적인 문제점

첫 번째 해결책 : 자체 제작

Raspberry Pi + CAN Bus Shield

- CAN 2Ch 지원
- CAN-FD 지원
- SPI 통신으로 socketcan 지원





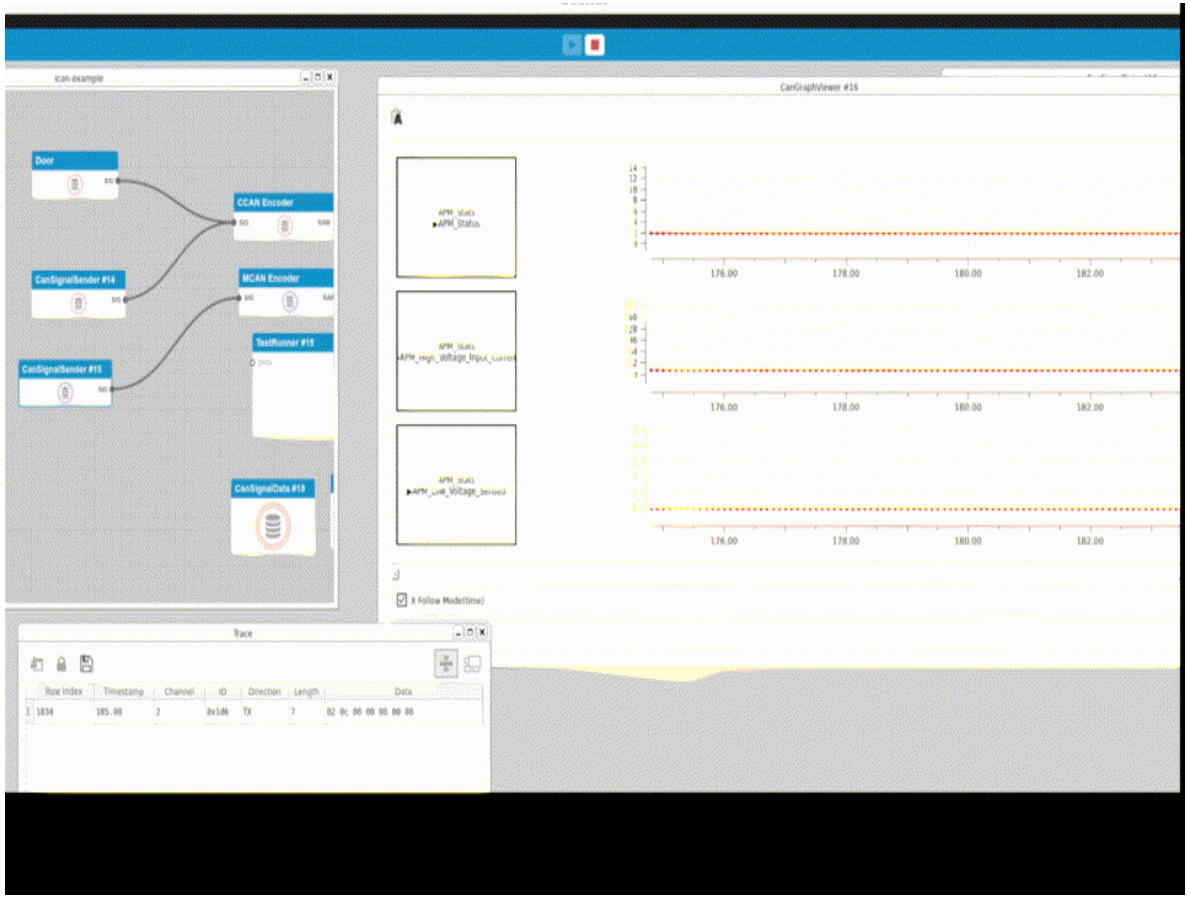
출처 : https://wiki.seeedstudio.com/2-Channel-CAN-BUS-FD-Shield-for-Raspberry-Pi/

첫 번째 해결책 : 자체 제작

시뮬레이션 PC Tool

- 오픈소스 활용
- CAN-FD 기능 추가
- CAN Log Replay 기능 추가(BLF, ASC)
- CAN Graph 기능 추가
- 성능 개선

➔ **PC와 RP4와의 통신은 이더넷을 활용**



첫 번째 해결책 : 자체 제작

RP4 → 마이컴 기반 전환 이유

- 느린 부팅 시간
- 실제 차량 CAN 로그 재생시 패킷 드롭
- CAN 송/수신 성능 부족

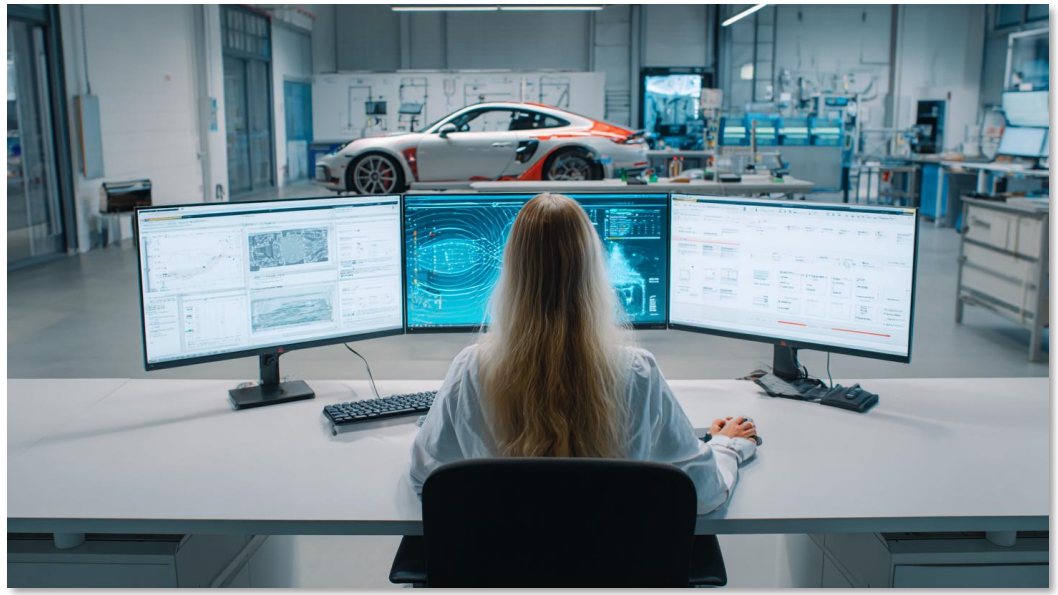
➔ **PC와의 통신은 이더넷을 유지**

모든 개발자가 CAN 시뮬레이션 장비를 보유하게 되어 문제 해결

두 번째 문제점: 타겟 보드 부재

직면한 상황

- 개발 시작 시점에 타겟 보드 미제공
- Working Sample 제공까지 장시간 소요
- 하지만 개발은 진행되어야 함



하드웨어와 소프트웨어의 서로 다른 개발 사이클은 업계 공통된 문제점

개발된 소프트웨어를 테스트 할 수 있는 환경 필요

두 번째 해결책 : 가상 환경

시도했던 해결책들

- PC 환경 : UI 확인용, 바이너리 호환성 없음(x86 vs arm)
- QEMU : 바이너리 호환성 개선, BSP 호환 불가

모두 제한적 활용만 가능

SDV와 디지털 트윈의 만남

계기와 시장 배경

- 컨퍼런스에서 디지털 트윈 트렌드 확인
- SOAFEE 활동을 통한 학습
- 전문 하드웨어 가상 플랫폼 존재 인지

실행 결과

- 접근성 높은 RP4 선정
- Yocto SD 카드 이미지를 가상 플랫폼에 직접 적용
- 결과 : 정상 부팅 및 동작 확인

발견 : 실제 하드웨어와 동일한 환경이 가능함을 확인

세 번째 문제점: 수동 테스트의 한계

SDV 시대의 요구사항

- 짧은 소프트웨어 개발 주기
- 애자일 개발 방법론 도입
- 데일리 빌드 및 통합

현실적 상황

- 첫 양산 이후 차종 전개 시 데일리 이미지 생성
- 고객사 CI/CD 파이프라인 완비
- 하지만 **테스트는 수동**

세 번째 문제점 : 시간과 품질의 딜레마

테스트 규모

- 디지털 클러스터 테스트 케이스 : 수만개
- 데일리 이미지 → 매일 테스트 필요
- 사람의 테스트 속도 한계

테스트 병목 현상의 문제점과 해결

- BAT도 수동 → 자동화 필요
- 데일리 이미지 대응 불가능 → 연속적 품질 보증 체계 구축 필요
- 품질 보증의 어려움 → 예측적 결함 감지 시스템 도입 필요

자동화 테스트의 필요성 확인

세 번째 해결책 : 자동화 테스트

SDV 시대의 요구사항

- 비용 효과 : 개발 초기 단계 버그 수정 시 시장 출시 후 대비 1000배 비용 절감
- 테스트 커버리지 : 자동화로 수백만 시나리오 동시 검증 가능
- 규제 준수 : ISO 26262, ISO/SAE 21434 등 안전 표준 충족

YAML 기반 자동화 테스트 구현

- YAML 형식으로 테스트 케이스 작성
- 자동 테스트 수행
- 테스트 수행 속도 대폭 향상
- 반복 가능한 테스트 환경 구축
- 사람의 실수 요인 제거

Software-in-the-Loop Simulation 테스트 환경

SILS 장점

- 비용 절감 : HILS 대비 70~80% 비용 절약
- 확장성 : 가상화로 무제한 병렬 테스트 가능
- 개발 속도 : 물리적 하드웨어 준비 시간 불필요

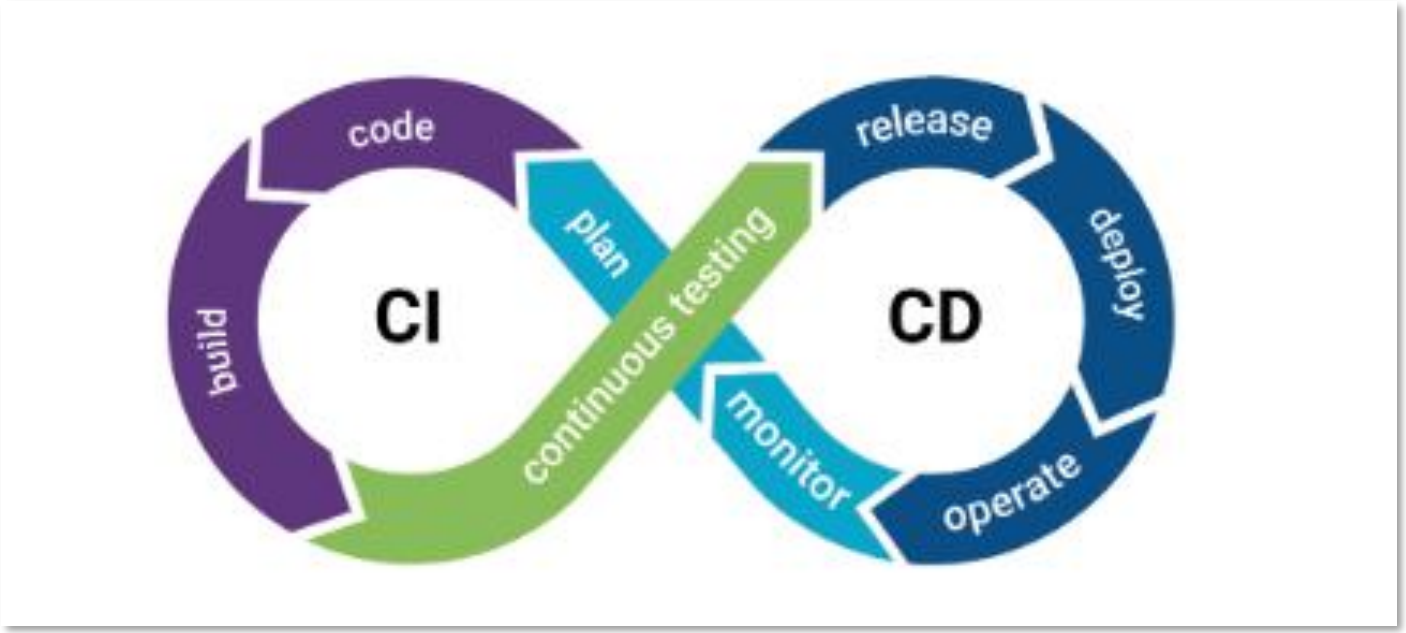
시장에서의 SILS 가치

- McKinsey 연구 : 개발 시간 20~50% 단축
- ROI : 테스트 자동화로 투자 대비 300~500% 수익
- 품질 향상 : 결함 감지율 90% 이상 달성

HILs vs SILs

구분	SILS	HILS
정확도	95% (가상환경 제약)	100% (실제 하드웨어)
비용	낮음 (소프트웨어만)	높음 (하드웨어 필요)
시간	빠름 (즉시 구축)	느림 (하드웨어 준비)
확장성	높음 (다중 인스턴스)	낮음 (하드웨어 1:1)
병렬성	무제한	제한적
유지보수	낮음	높음

CI/CT의 중요성



- CI/CD + CT(Continuous Testing) = 완전한 파이프라인
- 자동화된 테스트 → 지속적 품질 보증
- 가상 환경을 활용한 SILs 환경 구축 및 테스트 자동화가 필요

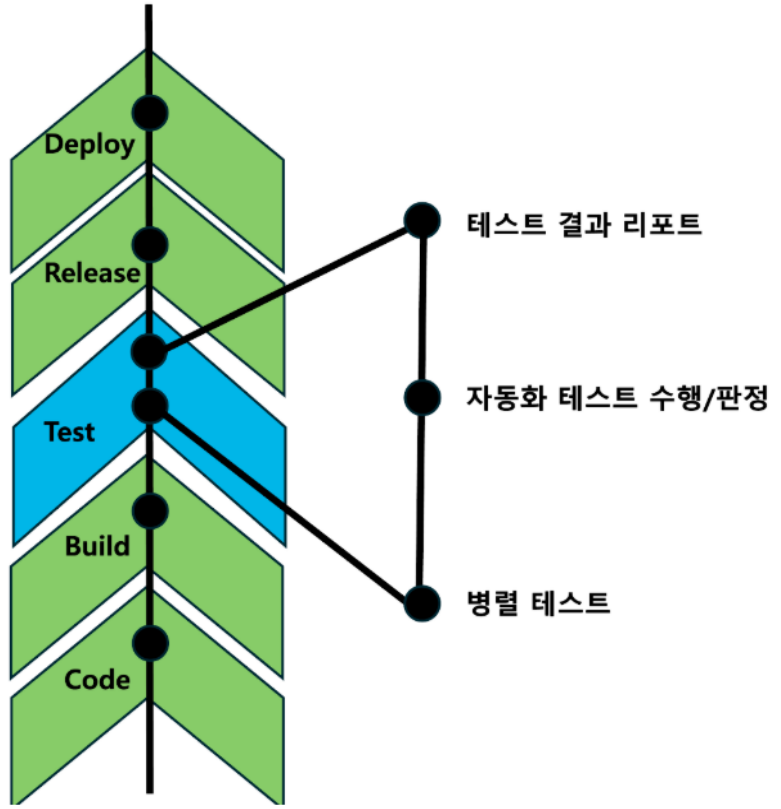
여정 요약

문제→ 해결 → SDV 진화

- CAN 장비 부족 → 자체 제작 → 개발 효율성 확보
- 타켓 보드 부족 → 디지털 트윈 →가상 검증 환경
- 수동 테스트 한계 → SILS 구축 → 자동화 품질 보증

시장과의 일치성

- 우리의 경험이 SDV 트렌드와 일치
- 비용 절감, 품질 향상, 개발 속도 증진 → 업계 요구 사항 충족
- 디지털 트윈, 가상화, 자동화 → 시장이 가는 방향과 동일




Innovate.
Connect.
Drive
with ivis
Visit our booth
SDV 소프트웨어 개발 고민
대화 나누며, 함께 SDV 생태계 만들어가요!
Booth #: E5-213

Thank You