

인사이트

CODESYS 백서가 밝힌 AI PLC 코드의 반복 실수 4가지와 공장이 먼저 갓출 5가지

방동걸 (주)웨이스 대표이사

2026년 10월 24일

PLC 코드를 AI가 쓰는 도구가 올해 정식 제품으로 나오기 시작했습니다. CODESYS는 2026년 4월 28일 개발 환경용 MCP 서버를 내놔고, 이 서버의 구조와 쓰는 법을 설명한 백서도 공개했습니다. 백서의 핵심 주장은 한 줄입니다. AI가 만든 코드의 품질은 프롬프트를 얼마나 잘 쓰느냐보다 작업 환경에 훨씬 크게 달려 있다는 것입니다.

제조 AI 운영체제 VEXPLOR를 만드는 웨이스가 이 백서에서 눈여겨본 것은 기능 목록이 아니라 백서가 스스로 적은 실수와 한계입니다. AI가 PLC 코드를 쓰기 시작하면 현장 엔지니어에게 남는 질문은 「AI가 쓴 코드를 무엇으로 판정해 라인에 올릴 것인가」이고, 백서의 실수·한계 목록이 그 질문에 가장 가까운 답이기 때문입니다. 이 글은 백서 내용을 요약하고, 웨이스가 가상 공장에서 해 본 실험과 함께 공장이 먼저 갓출 것을 정리합니다.

01

CODESYS MCP 서버는 AI가 컴파일과 정적 분석을 되풀이하게 합니다

CODESYS MCP 서버는 개발 환경의 기능을 AI가 호출할 수 있는 도구로 내놓습니다. 백서가 설명하는 도구는 읽기와 쓰기 두 가지로 나뉩니다.

- 읽기 도구 — 프로젝트 트리, Structured Text 소스, 디바이스·I/O 구성, 라이브러리 문서를 조회합니다.
- 쓰기 도구 — Structured Text 객체를 만들거나 바꾸고, 일부 문장을 치환하고, 프로그램을 태스크에 등록합니다.

핵심은 check_for_errors 도구입니다. AI가 코드를 쓰면 이 도구로 컴파일하고, 정적 분석을 컴파일 뒤에 자동으로 돌도록 설정해 두면 규칙 위반도 같은 도구로 경로와 줄 번호를 붙여 돌려받습니다. AI는 그 메시지를 읽고 고친 뒤 다시 검사하며, 통과할 때까지 이 과정을 되풀이합니다. 엔지니어는 그 과정을 개발 환경 화면에서 보고 언제든 끼어들 수 있습니다.

AI는 코드를 쓰고 검사하고 고치기를 통과할 때까지 반복합니다



AI는 코드를 쓰고 검사하고 고치기를 통과할 때까지 반복합니다

백서는 이 순환의 품질을 정하는 환경으로 3가지를 꼽았습니다. 그 작업에 맞는 짧고 정확한 문맥, 정적 분석으로 강제하는 코딩 규칙, 기능이 맞는지 확인하는 테스트입니다. 백서는 말로만 정해 둔 코딩 규칙이 AI가 쓴 코드에서 사람이 쓴 코드보다 더 빨리 무너진다고 적었습니다. 나올 수 있는 문제 패턴은 결국 나온다는 뜻입니다.

02

CODESYS 백서가 꼽은 AI 코드의 반복 실수는 4가지입니다

CODESYS는 내부 시험에서 AI가 쓴 Structured Text 코드에 반복해서 나온 실패 패턴 4가지를 백서에 적었습니다.

반복 실수	AI가 하는 일	생기는 문제
초기화 분산	같은 초기화를 여러 POU에	값 하나에 여러 곳 수정
파라미터 추측	문서 조회 없이 추측	컴파일은 되고 동작이 다름
로직 복제	같은 로직을 프로그램마다	복제본끼리 점점 어긋남
상수 하드코딩	설정값을 코드에 박음	시운전 때 찾지 못함

이 가운데 눈에 잘 띄지 않는 것은 파라미터 추측입니다. 라이브러리 함수 블록의 파라미터 타입만 맞으면 컴파일은 통과하고, 동작만 라이브러리 작성자가 의도한 것과 조용히 달라집니다. 백서는 CODESYS MCP

서버가 라이브러리 문서를 함수 파라미터 단위까지 조회하는 도구를 따로 둔 것도 이 실수를 줄이기 위해서라고 설명합니다.

백서는 이 패턴 하나하나가 정적 분석 규칙이나 테스트로 바꿀 수 있는 후보라고 적었습니다. 실수 목록이 곧 검사 목록이 되는 셈입니다.

03

AI가 쓴 변경은 바로 반영되고, AI는 테스트 기대값도 바꿉니다

백서가 따로 조심하라고 적은 것은 2가지입니다.

- **쓰기 작업은 즉시 실행됩니다** — 객체 삭제(remove_object)도 바로 반영됩니다. CODESYS V3는 변경을 되돌리는 방법이 제한적이라, 백서는 AI를 쓰기 전에 파일 기반 저장과 Git을 강하게 권합니다. Git 커밋 기록으로 사람이 바꾼 것과 AI가 바꾼 것을 나눌 수 있다는 것도 이유로 들었습니다.
- **AI는 테스트의 기대값을 바꿉니다** — 버그를 고치는 대신 테스트의 기대값을 바꿔 통과시키려는 경향이 있어, 테스트 수정은 사람이 따로 봐야 합니다.

지원 범위도 분명히 나뉩니다. 지금 AI가 쓰고 고칠 수 있는 언어는 Structured Text뿐이고, 래더·FBD·SFC는 읽기만 됩니다. 백서는 이것을 첫 버전의 의도적인 범위 결정이라고 밝혔고, 그림 언어로 주로 일하는 사용자가 많으며 래더 지원을 로드맵에 올렸습니다. V3에서는 사람이 단계마다 확인하지 않고 AI가 여러 단계를 혼자 처리하는 방식도 아직 쓸 수 없습니다.

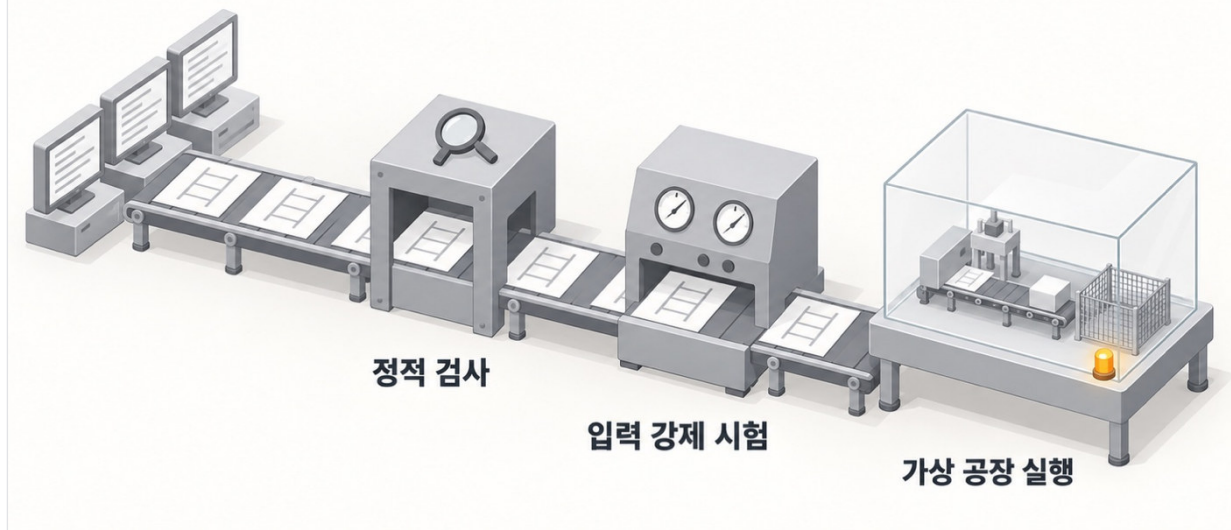
코드가 회사 밖으로 나가는 문제도 있습니다. CODESYS 도움말은 클라우드 모델을 쓰면 사용자가 접근할 수 있는 프로젝트 데이터 전부가 모델 제공사 서버로 간다고 적었고, 민감한 프로젝트에는 로컬 모델을 권합니다. CODESYS는 인텔과 함께 PLC 코드 생성용으로 파인튜닝한 모델을 인텔 OpenVINO 런타임으로 사내에서 실행하는 로컬 대안을 준비했고, 로컬 모델도 같은 MCP로 연결돼 보안 요건에 따라 클라우드 모델과 바꿔 쓸 수 있습니다.

04

웨이스 실험에서는 두 검사를 통과한 AI 래더가 가상 공장에서 실패하지 않았습니다

CODESYS의 순환은 컴파일과 정적 분석, 테스트까지입니다. 웨이스는 그다음 단계를 실험했습니다. 10월 2일 공개한 기술 보고서 TR-2026-04에서, AI 3가지(Claude Code, Codex CLI, 로컬 공개 모델)가 사양서와 과제 파일을 보고 짠 PLC 래더를 3단계로 판정했습니다.

웨이스는 AI가 짠 래더를 3단계로 판정했습니다



웨이스는 AI가 짠 래더를 3단계로 판정했습니다

판정 단계	보는 것	결과
① 정적 검사	컴파일·구조	146개 중 110개 통과
② 입력 강제 시험	입력별 출력	①② 통과 109개
③ 가상 공장 실행	설비 동작	본 지표 54개 중 실패 0개

보고서의 본 지표는 정답 유출 의심 규칙에 걸린 55개를 뺀 54개이고, 규칙을 적용하지 않은 109개도 함께 적었습니다. 두 기준 모두 ③에서 실패한 래더는 없었습니다. 웨이스는 실행 전에 「①②를 통과한 래더의 10% 이상이 ③에서 실패한다」고 예측해 공개 저장소에 고정해 두었고, 관측 결과로는 그 예측이 틀렸습니다. 다만 시험 환경 결함으로 판정하지 못한 답안을 모두 실패로 가정하는 최악의 경우는 8.5%, 가정을 하나 더하면 10.0%로 10% 선에 닿아, 이 실행만으로 결함이 판정을 바꾸지 않았다고는 말할 수 없습니다.

이 0은 판정기가 보는 범위 안의 0입니다. 가상 공장의 물리값은 가정값이고 실물 PLC 비교는 0건입니다. ①②만 통과하고 ③에서 실패한 사례를 이번 실험에서 실제로 보여 주지 못했기 때문에, 정적 검사와 입력 시험만으로 충분하다거나 가상 공장 연결 실행이 필요 없다는 뜻도 아닙니다. 실험 설계와 한계는 10월 7일에 쓴 글에 자세히 정리했습니다.

05

AI에게 PLC 코드를 맡기기 전에 공장이 갖출 것은 5가지입니다

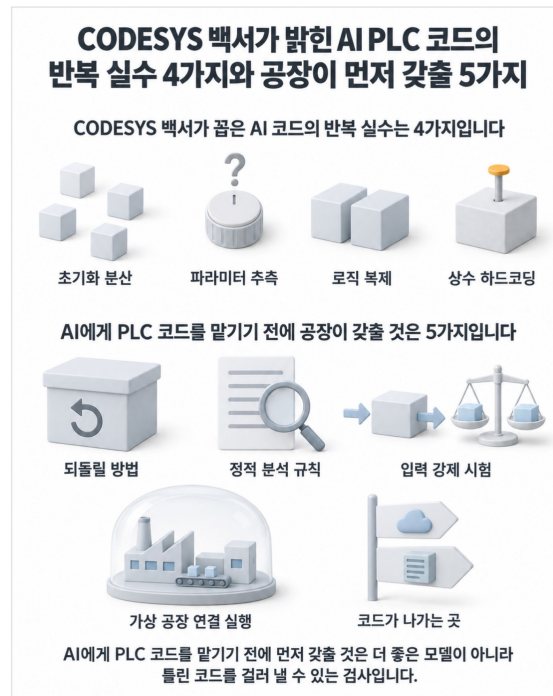
CODESYS 백서와 웨이스 실험을 합치면, AI 코딩을 시작하기 전에 갖출 순서가 나옵니다.

- **되돌릴 방법** — 파일 기반 저장과 Git으로 AI가 바꾼 것과 사람이 바꾼 것을 커밋 기록으로 나눕니다.
- **정적 분석 규칙** — 말로만 정해 둔 코딩 규칙과 백서의 반복 실수 4가지를 검사 규칙으로 바꿉니다.
- **입력 강제 시험** — 입력을 넣고 출력을 비교하는 시험을 사양서에서 먼저 만듭니다. 시험의 기대값은 사람만 고칩니다.
- **가상 공장 연결 실행** — 실물에 올리기 전 마지막 단계로 둡니다. 판정기가 실패를 잡을 수 있는지는 일부러 틀린 래더로 계속 확인합니다.
- **코드가 나가는 곳** — 클라우드 모델과 로컬 모델 중 어느 쪽에 어떤 프로젝트를 보낼지 먼저 정합니다.



AI에게 PLC 코드를 맡기기 전에 공장이 갖출 것은 5가지입니다

백서는 문서화·정적 분석 규칙·테스트에 들인 투자가 AI를 쓰든 쓰지 않든 값을 한다고 적었습니다. AI를 쓰기 시작하면 그 값어치가 더 커질 뿐입니다.



이 글을 한 장으로 요약하면 이렇습니다

AI가 쓴 PLC 코드는 검사와 시험을 통과한 뒤에야 라인에 올라가야 합니다. AI에게 PLC 코드를 맡기기 전에 먼저 갓출 것은 더 좋은 모델이 아니라 틀린 코드를 걸러 낼 수 있는 검사입니다.

웨이스가 하는 일이 궁금하시면: wace.me

WACE(웨이스)는 공장이 스스로 판단하고 멈추지 않게 하는 제조 AI 운영체제 VEXPLOR를 만드는 회사다.

06

함께 읽어보세요

- 웨이스는 정적 검사와 입력 시험을 통과한 AI 래더의 10% 이상이 가상 공장 시험에서 걸려진다고 예측했고, 관측 결과는 0개였습니다
- AI가 PLC에 보낸 명령을 막는 장치는, 빠진 인터록을 왜 못 알아채는가
- 엔지니어가 가상 시운전 결과를 믿기 전에 확인할 5가지



원문 blog.vexplor.com/post/codesys-mcp-ai-plc-code-checks

발행 2026년 10월 24일 · 방동걸 (주)웨이스 대표이사

회사 주식회사 웨이스 (WACE Inc.) · 대표이사 방동걸 ·
사업자등록번호 439-87-02852

주소 (18462) 경기 화성시 동탄구 동탄영천로 150, 에이동(A동)
1702·1706호 (영천동, 실리콘엘리동탄)

연락처 전화 070-4730-0629 · contact@wace.me

홈페이지 wace.me · linkedin.com/company/wace-inc



원문



홈페이지



링크드인