기술

저희 시뮬레이터를 밤새 15번 스스로 고치게 했습니다

무엇이 되돌릴 수 있게 했나

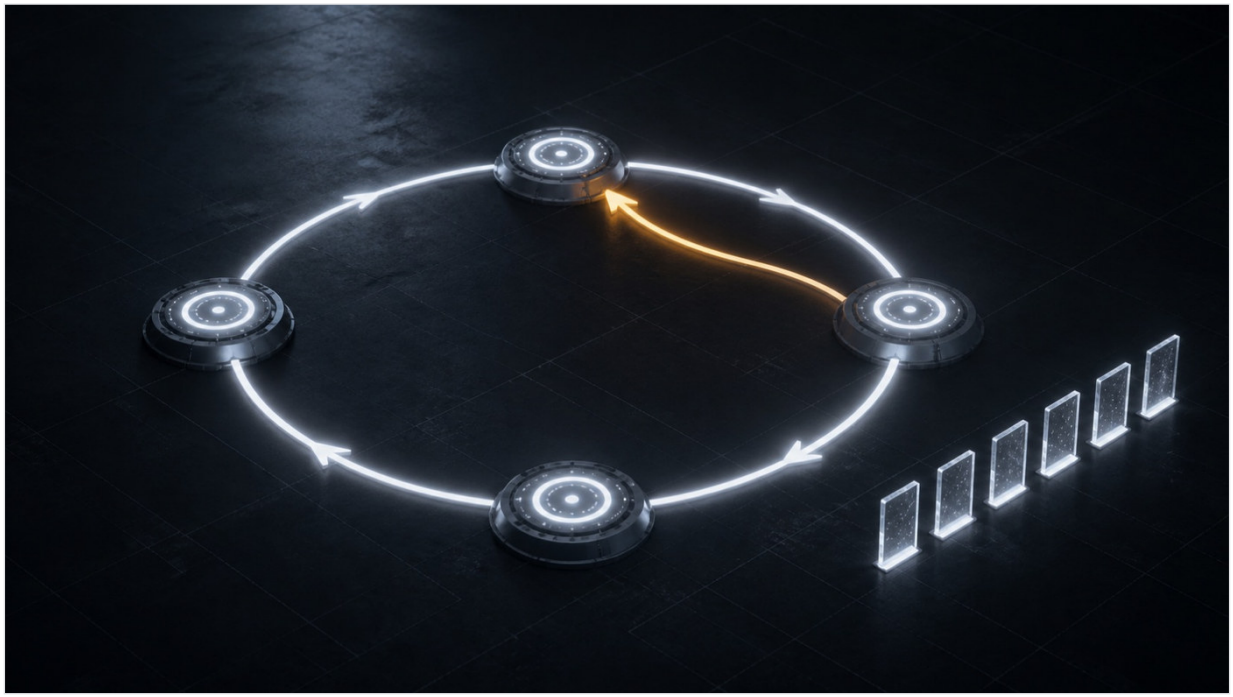
방동걸 (주)웨이스 대표이사
2026년 8월 31일

설비 수십 대가 맞물려 돌아가는 생산 라인을 3D로 재현하는 시뮬레이터를, 밤새 사람 없이 **15라운드** 돌렸습니다. 빌드 → 검증 → 수정 → 릴리스를 한 묶음으로 두고 그 묶음을 반복하게 한 것입니다. 아침에 결과를 열어 봤습니다. 웨이스가 VEXPLOR의 가상공장 시뮬레이터를 만들면서 직접 실행한 결과를 그대로 적었습니다.

가장 값이 있었던 것은 고쳐 낸 14번이 아니었습니다. **14번째 라운드에서 루프가 개선안 하나를 넣었다가, 검증에서 떨어뜨리고 스스로 철회한 것**이었습니다. 그다음 라운드는 같은 문제를 다른 방법으로 풀었습니다.

자동으로 고치는 반복을 돌려 보신 분이라면 이 대목이 왜 중요한지 아실 것입니다. 앞으로만 가는 루프는 몇 라운드만 지나면 손댈 수 없는 상태가 됩니다. 이 글은 그 밤에 무엇이 되돌림을 가능하게 했고, 무엇은 끝까지 자동으로 잡히지 않았는지를 정리한 것입니다.

밤에 돌리는 이유는 단순합니다. 한 라운드를 도는 데 걸리는 시간보다 **사람이 결과를 보고 다음 지시를 주기까지 걸리는 시간이 훨씬 길기 때문**입니다. 사람이 붙어 있으면 그 왕복이 전체 속도를 정합니다. 왕복을 없애면 라운드 수가 시간에 비례해 늘어납니다. 다만 그 순간부터 **아무도 안 보는 동안 무슨 일이 일어나는가**가 전부 설계 문제가 됩니다.



밤새 사람 없이 돌아가는 빌드·검증·수정·릴리스 반복 — 아침에 남는 것은 최종본 하나가 아니라 라운드별 기록입니다

01

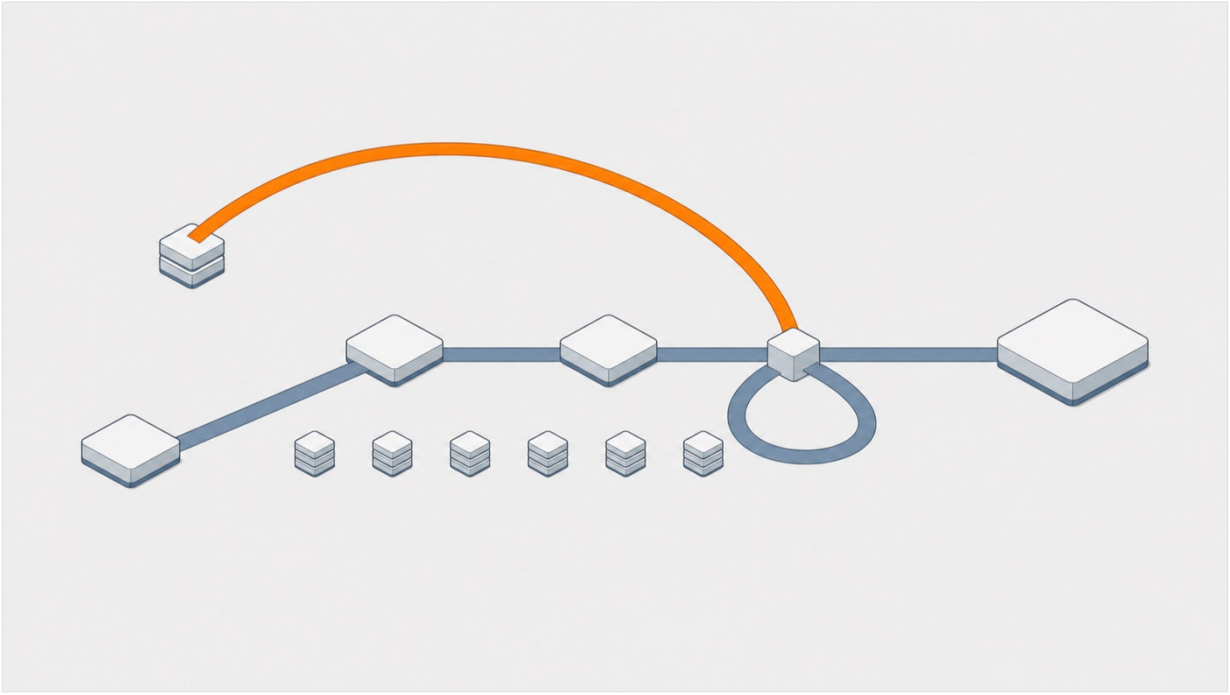
자동으로 고치는 반복은 대개 앞으로만 갑니다

루프를 처음 짜면 경로가 보통 둘입니다. 검증을 통과하면 다음으로 가고, 실패하면 다시 고칩니다. 여기까지는 누구나 넣습니다.

문제는 셋째 경우입니다. **고쳤는데 다른 것이 나빠진 경우**입니다.

이때 루프에 되돌림 경로가 없으면, 나빠진 상태 위에 다시 고치기를 쌓습니다. 그 위에 또 쌓습니다. 세 라운드쯤 지나면 지금 남아 있는 결함이 몇 번째 수정에서 들어온 것인지 아무도 모릅니다. 아침에 사람이 보게 되는 것은 "밤새 12번 고쳤습니다"라는 요약과, 무엇이 왜 이렇게 됐는지 설명할 수 없는 결과물 하나입니다.

사람이 하는 개발에서는 되돌리기가 워낙 당연해서 따로 설계하지 않습니다. 이상하면 직전 상태로 돌아가고, 그게 안 되면 그 전으로 갑니다. **자동 루프에는 그 습관이 없습니다.** 명시적으로 짜 넣지 않으면 그 경로 자체가 존재하지 않습니다.



한 지점에서 갈라지는 세 경로 — 앞으로 계속 가기, 제자리에서 다시 고치기, 그리고 앞서 저장해 둔 지점으로 물러서기

02

되돌릴 수 있으려면 3가지가 있어야 합니다

밤을 넘긴 뒤 되짚어 보니, 철회가 실제로 작동하려면 아래 셋이 전부 있어야 했습니다. 하나라도 빠지면 되돌림은 이름만 있고 작동하지 않습니다.

있어야 할 것	없으면 생기는 일	확인 방법
라운드별 산출물 보존	되돌릴 대상이 없다	결과물 수 = 라운드 수
판정 기준이 루프 밖	기준을 낮춰 통과한다	기준 파일 쓰기 차단
철회가 무인으로 완료	아침까지 멈춰 있다	야간 중단 0회

첫째, 매 라운드의 결과물이 통째로 남아야 합니다. 최종본 하나만 덮어쓰는 구조면 되돌릴 자리가 없습니다. 저장 공간이 아깝게 느껴지지만, 라운드 수만큼 결과물이 남아 있는지 세어 보는 것이 되돌림 준비 여부를 확인하는 가장 빠른 방법입니다.

둘째, 무엇을 통과로 볼 것인지가 루프 바깥에 있어야 합니다. 이게 핵심입니다. 판정 기준을 루프가 고칠 수 있으면, 개선이 막힐 때 루프는 기준을 낮추는 쪽으로 갑니다. 악의가 아니라 그것이 목표에 더 빨리 도달하는 길이기 때문입니다. 그 상태에서는 되돌림이 발동할 일이 없습니다 — 모든 라운드가 통과하니까요.

셋째, 철회가 사람 없이 끝나야 합니다. 되돌림 경로가 사람 확인을 요구하면, 새벽 2시에 발동한 철회는 아침까지 그 자리에 서 있습니다. 밤새 돌린 의미가 거기서 없어집니다.

03

루프가 고칠 수 없는 것을 먼저 정합니다

둘째 항목은 저희가 전에 원칙으로 적어 둔 것이기도 합니다. 자기 작업 방식을 고치는 시스템에서 **채점표만은 그 시스템이 못 건드리게 둔다**는 이야기였습니다. 그때는 원칙이었고, 이번은 그 원칙이 실제로 무엇을 만들어 냈는지를 본 경우입니다.

이번 루프에서 바깥에 둔 것은 판정 항목들이었습니다. 목표 지점 도달 오차, 정적인 겹침 여부, 작업 영역 경계 이탈 여부. 루프는 이 항목들의 값을 계산할 수는 있어도 항목 자체나 통과선을 바꿀 수는 없었습니다.

그래서 14번째 라운드에서 일어난 일이 "개선안이 통과하지 못했다"에서 그치지 않았습니다. **루프가 통과하지 못했다는 사실을 스스로 알 수 있었습니다.** 기준이 안쪽에 있었다면 그 개선안은 통과했을 것이고, 아침에 저희는 나빠진 결과물을 좋아진 것으로 받았을 것입니다.

되돌림 경로를 잘 짜는 것보다 **되돌릴지 말지를 판정하는 쪽을 손대지 못하게 하는 것**이 먼저입니다. 순서를 바꾸면 나머지가 전부 헛돕니다.

04

밤새 돌려 보니 잘 고치는 것과 못 고치는 것이 갈렸습니다

15라운드를 다 보고 나서 정리한 구분입니다. 자동 반복이 잘하는 일과 사람이 봐야 하는 일의 경계가 생각보다 뚜렷했습니다.

대상	자동 반복	사람
값 조정·여유 폭	잘한다	확인만
동작 순서·경유 지점	잘한다	확인만
무엇을 검사할 것인가	못 한다	정해 준다
언제 시작할 것인가	못 한다	정해 준다

위 두 줄은 후보를 만들어 시험하고 더 나은 쪽을 고르는 일입니다. 루프가 밤새 사람보다 훨씬 많이 시도했습니다.

아래 두 줄은 성격이 다릅니다. **판정 항목에 없는 것은 나빠져도 나빠진 줄 모릅니다.** 검사 목록을 늘리는 것으로 메울 수 있을 것 같지만, 그렇지 않은 경우가 그 밤에 실제로 나왔습니다.

한 가지 덧붙이면, 항목이 있어도 **재는 단위가 결과를 만듭니다.** 이번에도 팔이 서로 얼마나 가까운지를 재는 방식을 성긴 단위에서 촘촘한 단위로 바꾸자 같은 항목의 값이 크게 달라졌습니다. 성긴 쪽은 비스듬히 뻗은 팔을 실제보다 크게 잡습니다. 판정 항목을 정하는 것과 그 항목을 무엇으로 재는지를 정하는 것은 별개의 결정이고, 둘 다 루프 밖에 있어야 합니다.

05

가장 크게 틀렸던 것은 물리가 아니라 시작 조건이었습니다

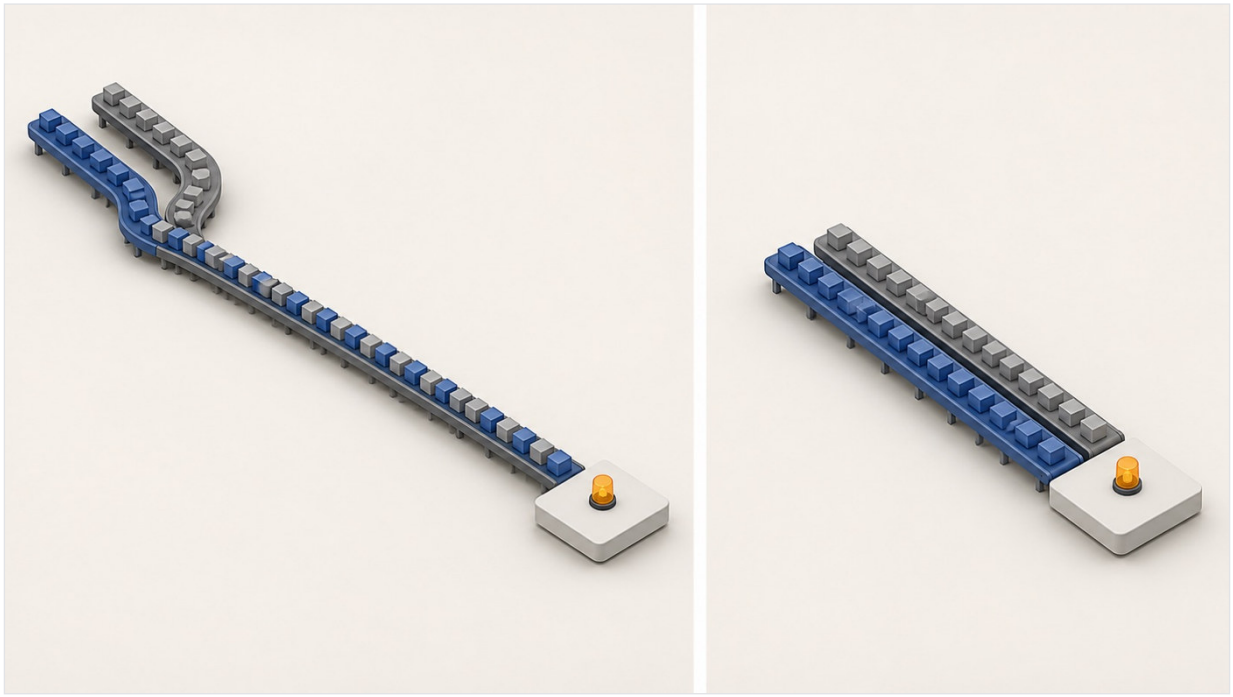
라인 안에 두 흐름이 한 제품에서 만나는 지점이 있었습니다. 각각 따로 만들어져 온 것이 한자리에서 합쳐집니다.

뒤쪽 흐름을 **언제 시작할지**를 "앞 공정이 끝난 시점"으로 잡아 뒀습니다. 읽어 보면 자연스럽습니다. 앞이 끝나야 뒤가 의미가 있으니까요.

그런데 그렇게 두면 나란히 돌아갈 수 있는 두 흐름이 한 줄로 늘어섭니다. 뒤쪽이 앞쪽을 끝까지 기다린 다음에야 움직이기 때문입니다. 그리고 이 상태에서 **검증은 전부 통과합니다.** 부딪히지 않고, 경계를 넘지 않고, 도달 오차도 정상이고, 순서도 틀리지 않았습니다. 화면으로 봐도 이상한 데가 없습니다.

드러나는 곳은 한 군데뿐이었습니다. 라인 전체를 도는 데 걸리는 시간이 눈에 띄게 늘어난 것입니다. 시작 조건을 "앞 공정의 결과물이 만나는 자리에 도착한 시점"으로 바꾸자 두 흐름이 다시 겹쳐 돌아갔고, 대당 시간이 제자리로 왔습니다.

이건 자동 루프가 잡을 수 없었습니다. 검증 항목에 "직렬로 늘어서지 않았는가"가 없었고, 넣으려 해도 넣기 어렵습니다. 그건 옳고 그름의 판정이 아니라 **설계 의도**이기 때문입니다. 직렬이 맞는 라인도 있습니다. 무엇이 정상인지가 대상마다 다른 항목은 판정 기준으로 옮겨 놓기 어렵습니다.



시작 조건을 앞 공정 완료 시점으로 두면 두 흐름이 한 줄로 늘어서고 도착 시점으로 두면 겹쳐 돕니다 — 검증은 두 경우 모두 통과합니다

시뮬레이션에서 결과를 크게 흔드는 것은 대개 물리 계산이 아니라 이런 정의입니다. 무엇을 시작 신호로 볼 것인가, 무엇을 하나의 단위로 셀 것인가. 계산은 틀리면 티가 나는데 정의는 틀려도 조용합니다.

06

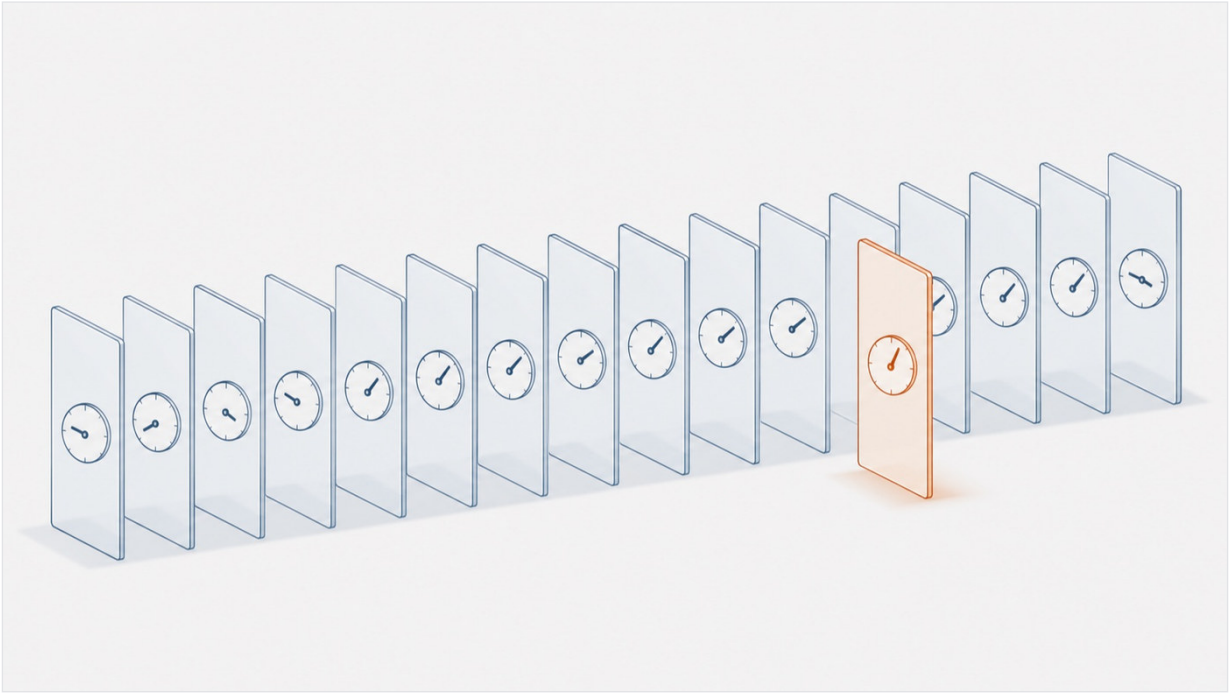
마지막 라운드가 가장 좋은 판은 아니었습니다

아침에 최종본을 고를 때, 마지막 라운드를 그대로 쓰지 않았습니다.

라운드마다 좋아진 항목과 나빠진 항목이 섞여 있었기 때문입니다. 마지막 판은 "가장 나중"일 뿐이지 "가장 좋음"이 아닙니다. 후반 라운드가 앞선 라운드의 장점을 되돌려 놓는 경우도 있습니다.

그래서 라운드별 판정값을 같은 형식으로 나란히 놓고 골랐습니다. 되돌리려고 산출물을 전부 남겨 뒀던 것이 여기서 두 번째 값을 했습니다 — **밤새 돌린 결과가 최종본 하나가 아니라 고를 수 있는 후보 15개**가 됐기 때문입니다.

밤새 돌린 반복의 산출물을 이렇게 보시면 판이 달라집니다. 목적이 "좋은 것 하나를 만들어 두기"에서 "고를 만한 것을 여러 개 만들어 두기"로 바뀌고, 그러면 라운드마다 무엇을 기록해 둬야 하는지가 저절로 정해집니다.



라운드별 판정값을 같은 형식으로 남겨 두면 마지막 판이 아니라 가장 좋은 판을 고를 수 있습니다

07

사람 없이 돌릴 반복을 짜실 때 물어보실 것

같은 구조를 준비하고 계시다면 아래 다섯을 먼저 정하시면 시행착오가 줄어듭니다.

- **실패했을 때 말고, 고쳤는데 다른 것이 나빠졌을 때** 루프는 무엇을 합니까
- **판정 기준을 루프가 고칠 수 있습니까** — 루프가 쓰기 권한을 갖는 경로 목록을 뽑아, 기준 파일이 그 안에 들어 있는지 눈으로 확인하시면 됩니다. 들어 있으면 되돌림은 영영 발동하지 않습니다
- 라운드마다 **되돌릴 수 있는 산출물**이 남습니까
- **판정 항목에 없는 것이 나빠졌을 때** 무엇이 그것을 알려 줍니까
- 아침에 사람이 보는 것은 **최종본 하나입니까, 라운드별 기록입니까**

마지막 항목이 의외로 갈립니다. 최종본만 보면 밤새 무슨 일이 있었는지 알 수 없고, 되돌린 라운드가 있었다는 사실 자체가 보이지 않습니다. 저희가 그 밤에 가장 많이 배운 것도 최종 결과물이 아니라 중간에 한 번 물러섰던 기록에서 나왔습니다.

밤새 사람 없이 돌아가는 반복을 이미 쓰고 계신다면, 지금까지 **되돌린 라운드가 몇 번이었는지** 한번 세어 보실 만합니다. 0이라면 루프가 완벽한 것일 수도 있지만, 되돌림 경로가 아예 없거나 판정 기준이 안쪽에 있는 것일 수도 있습니다. 두 경우는 결과 화면이 똑같이 생겼습니다.

가능성이 하나 더 있습니다. **루프가 되돌릴 일이 없을 만큼 안전한 것만 시도하고 있는 경우**입니다. 실패가 비싸면 시스템이든 사람이든 확실한 수정만 고르게 되고, 그러면 밤새 돌려도 몇 걸음밖에 못 나갑니다. 되돌림을 싸게 만들어 두는 진짜 이유가 여기에 있습니다 — 물러설 수 있어야 크게 시도해 볼 수 있습니다.

여러분의 루프는 나빠졌을 때 무엇을 합니까?

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

원문 blog.vexplor.com/post/what-makes-an-autonomous-loop-reversible
발행 2026년 8월 31일 · 방동걸 (주)웨이스 대표이사

회사 주식회사 웨이스 (WACE Inc.) · 대표이사 방동걸 · 사업자등록번호 439-87-02852
주소 (18462) 경기 화성시 동탄구 동탄영천로 150, 에이동(A동) 1702-1708호 (영천동, 실리콘앨리동탄)
연락처 전화 070-4730-0629 · contact@wace.me
홈페이지 wace.me · linkedin.com/company/wace-inc

원문 홈페이지 링크드인