Coldcode Si Labs Library WORKING PAPER 2026-03

# Unwitnessed Autonomy

*Externalities, borrowed identities and duplicate clocks in an autonomous growth pipeline*

**Courage Garikai** · Funded by Coldcode Si Labs

Coldcode Si Labs Library Working Paper 2026-03 · First edition · 2026

**Disclosure.** The case venture, Tapstall, belongs to the same owner who commissions the work of Coldcode Labs Think Tank. The audit and rebuild analysed here were carried out by the owner working with AI coding agents. The same analytic process produced the audit, the coding of its findings and this paper, so the case is not an independent evaluation. At the time of writing Tapstall had no paying customers, and nothing here is evidence of commercial results. This paper does not say how the first version of the venture was produced; where the evidence bears on that question, it is treated as a hypothesis. AI tools assisted in drafting this paper, and the author is responsible for its content. Individuals and businesses who received messages from the system are not named.

---

## Abstract

Small ventures can now run growth pipelines that act without a person in the loop. A scheduled job finds prospects, another writes to them, a third posts to social accounts, and a payment webhook switches accounts on. Each component can be correct by its own tests while the pipeline as a whole does things that nobody sees. This paper names that condition *unwitnessed autonomy*: an automated system that takes consequential actions on parties outside its boundary when no party is placed to observe those effects and feed them back to the operator.

The paper reports a design-science case study of Tapstall, a shop-link service for informal and social-commerce merchants in Zimbabwe and South Africa. Its growth pipeline ran on free cloud tiers with no paid staff. On the day of the audit, the venture's own test suite passed 420 of 420 tests against the live code. In that same code:

- Of the 17 distinct addresses the outreach runner had already mailed, 14 were not prospects at all. They included a magazine reviewer, a platform's support desk, a government weather service and a literal placeholder address.
- These messages went out from the owner's personal mailbox, which was also the support inbox of a separate venture, and the runner had been marking that venture's support mail as read.
- The press module had posted three times a day into the other venture's social account.
- An old copy of the service on the same account was answering writes against production data with a superseded secret. Two more copies on a second account ran their own scheduled jobs.
- Every merchant email the product promised had been a silent no-op since launch.
- The public facts printed on a storefront were enough to take over that store.

# UNWITNESSED AUTONOMY

**Working Paper 2026-03**

Copyright © 2026 Courage Garikai.

Published by Coldcode Si Labs Library, Harare, Zimbabwe.

First edition, 2026.

All rights reserved. No part of this publication may be reproduced, distributed, or transmitted in any form or by any means withou
written permission of the publisher, except in the case of brief quotations in a review or certain other noncommercial uses permit
copyright law. The frameworks, worksheets and checklists in this publication are meant to be used, copied and adapted by the rea

This is a work of analysis and practical guidance. It is not financial, legal, investment or professional advice, and no outcome is pr
results depend on the reader's own market, effort and judgement. Platform behaviour and figures were checked at the time of wri
change.

COLDCODE SI LABS LIBRARY · Harare, Zimbabwe

The audit and the rebuild logged 24 defects in total. Coding them shows that 16 (67%) were *witness gaps* of four kinds:

- the effect of an action was never observed;
- the recipient's position was never observed;
- the identity the system acted under was not its own;
- the number of clocks and copies running was unknown.

None of the 24 was found by the existing test suite. Four of the 13 defects found during the rebuild were introduced by the fixes themselves, and each was caught by a live dry run or by inspecting real output, not by a unit test.

The paper contributes:

- a definition of unwitnessed autonomy that separates it from ordinary software defects, technical debt and spam;
- a taxonomy of nine defect classes, of which five are witness gaps;
- a *witness map*, an audit instrument that asks five questions of every autonomous action;
- four design rules for small autonomous pipelines: one clock, one voice, one identity, one surface;
- seven propositions for testing.

The case is a single pre-revenue venture analysed by an interested party, and the propositions are offered for testing.

**Keywords:** unwitnessed autonomy; automation; externalities; growth pipelines; outreach; email; identity coupling; distributed clocks; silent failure; informal economy; design science

---

## 1. Introduction

A founder with no staff and no budget can now run a working company's growth machinery: prospect lists, first-touch emails, follow-ups, social posting, payment activation, renewal reminders. Free cloud tiers supply the compute, scheduled CI runners supply the minutes, and a social-media scheduler or a mailbox app password supplies distribution. The marginal cost of another automated action is close to zero. What the founder gains in reach, the founder loses in sight. A person who sends twenty emails by hand reads twenty addresses. A pipeline that sends twenty emails a day reads none of them, unless someone has built it to.

This paper is about what happens in that gap. Its starting observation is unremarkable to anyone who has run production systems: a system can pass its own tests and still behave badly. The specific concern is narrower. It is the class of failures where an automated system *acts on someone outside itself* (a stranger's inbox, another brand's social account, a payment, a merchant's livelihood), and the consequences of the action never return to anyone who could correct them. I call this condition *unwitnessed autonomy*. Its defining feature is not that the system is wrong but that nobody is placed to notice that it is wrong. The harm lands on parties who have no channel back to the operator, while the operator's own surfaces (dashboards, logs, tests) report success.

The literature on automation has studied the human operator's loss of awareness for four decades, from Bainbridge's "ironies of automation" [1] to Parasuraman and colleagues' work on misuse and levels of automation [2], [3]. Safety research has studied how organisations come to accept anomalies as normal [4] and how missing feedback loops make control structures unsafe [5]. Economists have long understood that costs borne by third parties are underpriced by those who impose them [6], [7], and email has been a canonical example since the economics of spam were measured [8]. Security research named the "confused deputy", a program that uses authority it holds for one purpose on behalf of another [9]. Distributed-systems research gave us the vocabulary of clocks, ordering and duplicate execution [10], [11]. Machine-learning engineering added "undeclared consumers" and hidden feedback loops as forms of debt [12]. What these literatures have not done is bring the strands together for the setting that cheap automation has now made common: the micro-venture whose growth pipeline is fully autonomous, runs on shared free infrastructure and is operated by one person who reads none of its output.

The paper asks three questions:

1. What forms does unwitnessed autonomy take in a small autonomous growth pipeline?
2. Why does the operator's ordinary evidence (a green test suite, a dashboard that says "active", a run marked successful) fail to reveal them?
3. What design rules and audit instruments make the effects of autonomous action visible again, at no cash cost?

It answers them through a design-science case study [13] of Tapstall, a service that gives informal and social-commerce merchants in Zimbabwe and South Africa a shop link: a menu page from which buyers order over WhatsApp and pay the merchant directly. Tapstall's growth pipeline was built to run with no person in the loop. A scheduled job hunted for prospective merchants, a server judged them and allocated each to one commissioned seller, an outreach runner emailed and followed up, and a press module posted to social rails three times a day. On 28 October 2026 its owner commissioned a read-only audit of the live system (version 1.9.1), followed the same day by a rebuild to version 2.0.0. The audit, the rebuild and their verification produced a dataset of 24 defects, together with the live records that exposed each of them: lead rows, sent-mail records, run logs, the credential inventory of two cloud accounts, and dry-run outputs.

The findings are uncomfortable in a specific way. The pipeline was not broken in the sense that matters to its own instruments. Its tests passed. Its runs completed. Its dashboard reported rails as configured. What it did to the outside world was a different matter, and the record of what it had done was scattered across a mailbox, a queue of rows, an error string stamped "done" each night, and a social account that belonged to somebody else. The dominant pattern was not a wrong computation. It was the absence of anyone positioned to see the computation's effect.

The paper makes five contributions.

**A definition.** Unwitnessed autonomy is distinguished from three neighbouring ideas. It is not just a software defect, because the defining harm falls outside the system's boundary and outside its tests. It is not technical debt, because debt is a cost the owner pays later, whereas unwitnessed autonomy imposes costs on others now. It is not spam, though it can produce spam, because no intent to send unwanted mail is required and the same structure shows up in payments, social posting and account security.

**A taxonomy.** The 24 defects sort into nine classes. Five of them are *witness gaps*, failures of observation rather than of computation:

- externality;
- borrowed identity;
- multiplicity of clocks and copies;
- silent failure;
- boundary interpretation.

The remaining four are more familiar: authority leakage, surface sprawl, dependency capacity and build defects. The witness gaps account for two-thirds of all defects and six of the nine critical- and high-severity audit findings. The other three sit in the authority and surface classes, and every critical or high finding concerns an effect beyond the function that caused it.

**An instrument.** The *witness map* asks five questions of every autonomous action:

- Under whose identity does it act?
- On which clock does it run, and how many copies of that clock exist?
- What evidence returns that it had its intended effect?
- What evidence returns about the recipient's position?
- What stops it?

Applied retrospectively, the questions would have surfaced every witness gap in the case.

**Design rules.** Four rules for small autonomous pipelines follow from the case:

- **One clock.** A single scheduler fires every job, and a failed fire is retried, not stamped.
- **One voice.** A recipient hears from one sender identity, and every message carries a working exit.
- **One identity.** The system acts only under credentials that belong to it.
- **One surface.** There is one place where actions on prospects are taken, so there is one door to guard.

**Propositions.** Seven propositions state how unwitnessed autonomy arises, why ordinary evidence misses it and what governs it. They are framed for testing in other ventures.

The paper also documents something that the case makes unusually visible: four of the 13 defects found during the rebuild were introduced by the fixes. A gap filled in one place opened another, as when removing a spreadsheet surface left no surface at all for working prospects. The corrected prospect hunter's first live run filed branches of supermarket chains and their corporate addresses. Each of these was caught by observing real output, not by tests. The implication is not that fixes are dangerous. It is that witnessing has to be a standing practice rather than a one-off audit, because the act of repair produces new unwitnessed actions.

The rest of the paper proceeds as follows. Section 2 reviews the relevant literatures and locates the gap. Section 3 describes the method and the case. Section 4 reports the findings: what the pipeline had done, the anatomy of its defects, how they were found, the rebuild, and the defects the rebuild introduced. Section 5 develops the definition, taxonomy, witness map, design rules and propositions. Section 6 discusses implications, limitations and directions for research, and Section 7 concludes.