StrongDM Software Factory Case Study

ASDM Level 6: Autonomous Coding & Review

ProductBuildersHQ

May 2026

StrongDM's Software Factory is one of the clearest public examples of ASDM Level 6: Autonomous Coding & Review. The system removes humans from the normal coding and code-review loop. Humans define intent, scenarios, and constraints; agents generate code, run validation harnesses, and iterate until the system converges.

The case is important because it shows a concrete operating model beyond AI-assisted coding. It also shows the boundary of the evidence: StrongDM demonstrates autonomous coding and review, but the public material does not establish full autonomous production operations. Under ASDM, that places StrongDM at Level 6, not Level 7.

The Core Shift

Most AI coding adoption starts with a familiar pattern: humans write prompts, agents produce code, and humans review the result. The workflow gets faster, but the accountability model stays the same. A person still reads the code before it is trusted.

StrongDM's Software Factory changes that model. Its published rule set is explicit:

- Code must not be written by humans.
- Code must not be reviewed by humans.

That does not mean humans disappear. It means humans move upstream. Their job becomes defining the desired behavior, writing scenarios, setting constraints, and maintaining the validation environment. The agent's job is to produce code and keep iterating until the system satisfies those scenarios.

This is the Level 5 to Level 6 transition in ASDM: human review stops being the quality gate. Scenario satisfaction becomes the quality gate.

How the Factory Works

The factory starts with specifications and scenarios. Humans describe what the software should accomplish, not how the code should be structured. Those scenarios are stored outside the implementation path, which matters because the implementation agents should not be able to rewrite the tests that judge them.

Agents then generate code, run validation harnesses, observe failures, and revise. The goal is not a perfect first pass. The goal is convergence: repeated implementation attempts that move the system toward scenario satisfaction.

The most important infrastructure is StrongDM's Digital Twin Universe. StrongDM builds behavioral clones of third-party services such as Okta, Jira, Slack, Google Docs, Google Drive, and Google Sheets. These twins allow agents to run high-volume validation without hitting production API limits, creating real customer-side effects, or paying external API costs for every test run.

In ASDM terms, this is not merely "AI writes code." It is a validation architecture designed to make human code review unnecessary.

What StrongDM Changed

From Code Review to Scenario Satisfaction

Traditional software delivery relies on humans reading code before merge. StrongDM replaces that step with scenario-based validation. Scenarios are end-to-end user stories stored outside the implementation path.

This matters because AI-generated tests can be reward-hacked. If implementation agents can see or rewrite the tests, they can produce code that satisfies narrow checks without satisfying user intent. StrongDM's scenario model creates an external evaluation layer.

From Production APIs to Digital Twins

StrongDM's Software Factory depends on systems that interact with external SaaS services. Testing against real Okta, Jira, Slack, or Google Workspace environments creates rate limits, cost constraints, fragility, and limited failure-mode coverage.

The Digital Twin Universe addresses this by creating behavioral clones of those services. The DTU allows agents and simulated users to exercise scenarios at volumes that would be impractical or unsafe against production APIs.

From Human Iteration to Agentic Convergence

The factory model is iterative. Agents generate code, run harnesses, observe failures, revise, and repeat. StrongDM links this possibility to a late-2024 model inflection, when long-horizon coding workflows started compounding correctness rather than compounding error.

Metrics and Signals

The strongest public metrics are structural rather than conventional productivity measures:

- 0% human-written code target. StrongDM's published rule says code must not be written by humans.
- 0% human code-review target. StrongDM's published rule says code must not be reviewed by humans.
- At least $1,000/day token-spend benchmark per human engineer. StrongDM frames token spend as a practical sign that the factory is using enough agent runtime.
- Thousands of scenarios per hour. StrongDM reports that the DTU can run validation at volumes unavailable against production APIs.
- Six named Digital Twins. StrongDM publicly names Okta, Jira, Slack, Google Docs, Google Drive, and Google Sheets.
- Three-person AI team signal. Simon Willison describes visiting StrongDM's three-person AI team in October 2025.

- Public codebase-size signal. Simon Willison reports StrongDM's cxdb repository as roughly 16,000 Rust LOC, 9,500 Go LOC, and 6,700 TypeScript LOC.

These numbers do not prove lower defect rates or faster cycle time by themselves. They prove something different: StrongDM has designed a system where the normal human coding and review activities are no longer part of the delivery loop.

Case Study Classification

Dimension	Assessment
ASDM level	Level 6: Autonomous Coding & Review
Primary evidence	StrongDM Software Factory site, StrongDM blog, Simon Willison analysis
Human role	Specification owner and scenario designer
AI role	Code generation, validation-loop execution, convergence
Validation model	Scenario-based validation, satisfaction metrics, Digital Twin Universe
Operational status	Autonomous coding/review demonstrated; autonomous production operations not established publicly
Level 7 evidence?	Insufficient

ASDM Practice Mapping

ASDM practice	StrongDM evidence	Maturity
Specification-first development	Humans define intent and constraints before agents execute.	Strong
Scenario-based validation	End-to-end user stories are stored outside the codebase and used as holdout validation.	Strong
Probabilistic satisfaction metrics	StrongDM uses "satisfaction" to measure what fraction of observed scenario trajectories likely satisfy the user.	Strong
Digital Twin testing	DTU clones third-party services and allows high-volume validation.	Strong
Zero human code review	StrongDM's rule explicitly prohibits human code review.	Strong
Governance-by-policy	StrongDM defines rules and constraints, but public material is lighter on production policy governance.	Partial
Full provenance	cxdb suggests investment in conversation/tool-output history, but public material does not fully describe audit controls.	Partial

ASDM practice	StrongDM evidence	Maturity
Autonomous operations	Public evidence does not establish autonomous production deployment, remediation, or incident response.	Not established

The practice mapping shows why StrongDM is such a strong Level 6 case. It has strong evidence for the practices that remove human coding and code review, but weaker public evidence for the governance and production operations practices needed for Level 7.

Level Boundary Analysis

Why This Qualifies for Level 6

StrongDM's public material is strong evidence for Level 6:

- Humans do not write code.
- Humans do not review code.
- Validation is scenario-based.
- Digital Twins enable high-volume behavioral validation.
- Agents run end-to-end once work is specified.
- Satisfaction metrics replace binary pass/fail testing as the primary validation lens.

Why This Does Not Yet Qualify for Level 7

The public material is not enough to classify StrongDM as Level 7:

- It does not establish autonomous deployment into production customer environments.
- It does not establish autonomous rollback, remediation, or incident response.
- It does not establish that enterprise offerings such as StrongDM's PAM product are operated without routine human gates.
- It does not describe production SLO governance, policy-based operational authority, or compliance evidence for autonomous operations.

Under ASDM, Level 7 requires autonomous operations, not just autonomous coding and review.

Transferable Lessons

Validation must become independent from implementation. If agents can edit the tests that judge them, the system can converge on test-passing behavior rather than user-satisfying behavior.

Digital Twins turn validation into infrastructure. StrongDM's DTU is not a side project. It is the enabling substrate for removing human code review.

The human role moves upstream. Humans still matter, but their work shifts from implementation and review to intent definition, scenario design, constraints, and factory governance.

Level 6 is expensive before it is cheap. StrongDM's $1,000/day per engineer token benchmark is a useful warning. Autonomous coding at this level consumes significant model runtime, simulation, and validation capacity.

Level 6 does not imply Level 7. A team can remove human coding and code review while still relying on humans for production exposure, operations, and incident response.

Open Questions

- What percentage of StrongDM's internal and customer-facing software is built with this factory model?
- Are the strongest examples open-source, internal tooling, test software, or production product code?
- What defect rates, rollback rates, or escaped issue rates does the factory produce?
- How are scenario libraries reviewed, versioned, and audited?
- How does StrongDM prevent architectural drift when humans do not read code?
- What operational gates remain before software reaches enterprise customers?

Sources

- StrongDM AI. "Software Factories And The Agentic Moment." February 6, 2026. https://factory.strongdm.ai/
- StrongDM. "The StrongDM Software Factory: Building Software with AI." February 19, 2026. https://www.strongdm.com/blog/the-strongdm-software-factory-building-software-with-ai
- Willison, Simon. "How StrongDM's AI team build serious software without even looking at the code." February 7, 2026. https://simonwillison.net/2026/Feb/7/software-factory/