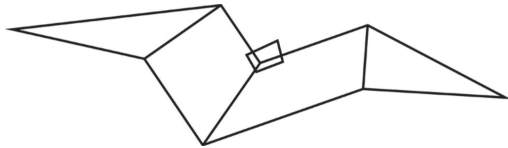


Rethinking Insurance Software Delivery With AI



By
David Kuhn

Table of Contents

Executive Summary 1

Why Delivery Slows Before Implementation 2

The Software Knowledge Supply Chain 3

Four Myths Standing Between Carriers and Faster Delivery 5

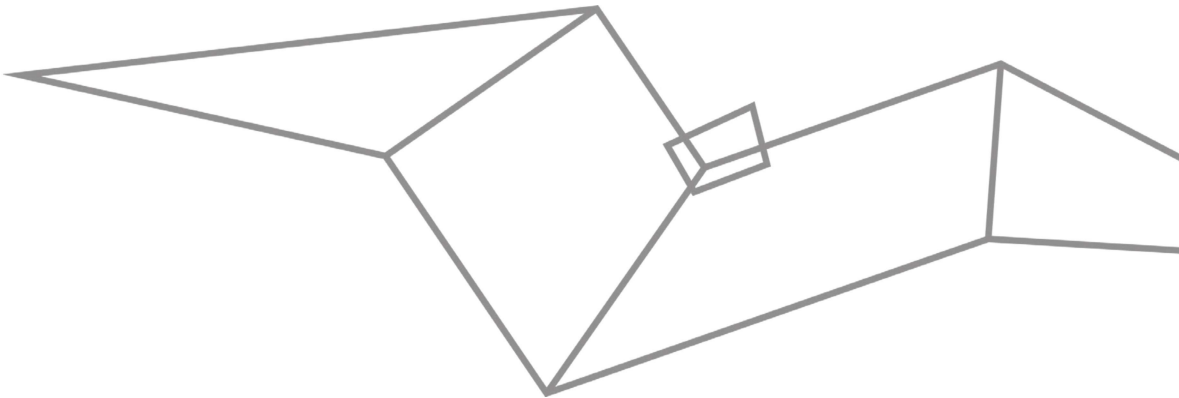
Five Considerations for AI-Assisted Delivery 7

The Assisted Delivery Life Cycle 9

What Changes When Understanding Is Maintained 12

Rethinking Delivery With a Strategic Partner 13

About ValueMomentum 13



Executive Summary

Carriers have spent the past decade investing in delivery speed. Agile reorganizations, DevOps tooling, cloud migration, and now artificial intelligence (AI) coding assistants have all been introduced with the same objective: shorten the time between a business decision and a working capability. Yet a modest change to the processes behind policy issuance or claims adjudication still takes far longer than the available tech stack would suggest.

The issue isn't a carrier's IT abilities. Instead, before a carrier can change something, it has to reconstruct how that capability operates today, including which systems participate in it and what breaks downstream when any part of it changes. Insurance software encodes decades of rating rules and adjudication logic that were rarely documented, written largely by people who have since retired. The knowledge still exists somewhere in the environment, but reassembling it now often costs more than the change it enables. Boston Consulting Group (BCG) puts the [failure rate for large-scale core transformations at 74%](#), and developers [spend between 58% and 70% of their time understanding existing code](#) rather than writing it.

This “understanding gap,” the distance between how a process works and what an insurance organization knows about the why and how behind the process, costs insurers time, money, and business agility. When it comes to software delivery, many insurers forget to address foundational questions: Why does a process exist? What steps are involved in that process? What workarounds have become accepted as part of the process?

In this whitepaper, you will learn:

- Why delivery timelines stall before implementation begins
- What the understanding gap costs carriers
- How to rethink the delivery life cycle with AI
- How to incorporate AI into each stage of the delivery life cycle for measurable business results

Why Delivery Slows Before Implementation

Carriers are not underinvesting in modernization. North American insurers committed roughly [\\$10.5 billion to core IT modernization](#) between 2024 and 2026, and AI adoption inside delivery organizations has accelerated alongside that spend. But the results have been inconsistent; the failure rate for large-scale core transformations [remains at 74%](#), a figure that has held steady even as tooling has improved.

The pattern behind that number is visible in the programs that failed. The same study documented a central European group that [abandoned a cross-country platforming project](#) after eight years and wrote off more than \$500 million. A carrier in southern Europe completed its claims platforming program but finished 500% over budget. Another central European insurer attempted to build a new platform internally to support direct business and never brought it to production.

That gap between spending and outcome has a specific cause. AI investment inside delivery has concentrated almost entirely on code generation and developer copilots, which accelerate the stage of the life cycle that was already the least expensive. Writing code never took up the bulk of delivery timelines, which is why the productivity gains appear clean in engineering metrics without moving the dates the business sees. That is a difficult outcome to defend to a board that has already funded several rounds of delivery transformation.

These projects increasingly require time to be spent reconstructing documentation before teams can even begin to work on new capabilities. A carrier's core systems hold decades of accumulated decisions, including state-specific rating rules, bureau deviations, underwriting exceptions, and adjudication logic built up over years of filings and product changes. Most of it was implemented correctly and documented sparsely, and the people who could explain why a particular exception exists have, in many cases, left the organization.

Before a delivery team can change a capability, the organization has to rebuild an accurate picture of how that function works today. Issuing a policy, for example, may involve separate systems for policy administration, billing, document generation, and more, each of which has its own set of integrations, rules, and workarounds. Teams assemble this holistic view through stakeholder interviews, source code analysis, document review, and dependency tracing. But typically, when a particular initiative closes, the understanding behind the system also disappears,

