

Forge Modernized a Fortune 1000 Retailer's Legacy Application **in Hours, Not Weeks**

Executive Summary

A global beauty and personal-care retailer needed to modernize a legacy application documented only by ambiguous XML examples. That kind of gap normally costs eight weeks of discovery, drafting, and rework. Using Forge, Opsera ingested the source material directly, resolved the interpretation gaps in a single guided session, and generated a full, aligned set of delivery artifacts down to execution-ready work orders. A project the team projected at eight weeks was delivery-ready in 2 hours and 21 minutes, with roughly \$250K in projected savings and no artifacts skipped along the way.

2 hrs 21 min**~50**

work orders generated

40%

lower token spend

~\$250K

in projected savings

Background

The company is a global beauty and personal-care manufacturer and retailer, selling across dozens of markets through a large direct-sales network. Its legacy application had no documentation to speak of. The only reference material was a set of XML examples left behind by the original build.

Opsera brought this engagement to Forge to see whether a full discovery-to-implementation cycle could be compressed without cutting corners on rigor.

The Challenge

The XML examples weren't broken, but they were interpretable in more than one way. That ambiguity is exactly what a traditional delivery flow is slow to resolve. It gets worked out through discovery workshops, tribal knowledge, back-and-forth with stakeholders, BRD drafts that get redrafted, architecture reviews that drift, and a long tail of work-order rewrites once developers hit the first surprise.

The real cost isn't just calendar time. Each handoff between requirements, architecture, and engineering is a place where intent gets restated, reinterpreted, and partially lost. By the time a team is writing code, the spec, the design, and the implementation are often quietly answering different questions. That drift is where technical debt gets built in from day one.

The Approach

1 Ingest the source material directly

Forge ingested the XML documentation as-is and worked through the full delivery chain in a single guided session. No separate discovery phase was required.

2 Generate the full artifact chain in lockstep

The system produced the assessment, ForgeScore, a new intent brief with modernization analysis, the PRD, BRD, and architecture proposal. It then expanded the architecture into roughly 50 discrete, execution-sized work orders, all traceable to the same source.

3 Surface and resolve ambiguity with the team

Rather than papering over the gaps in the original XML, Forge flagged them explicitly, proposed resolutions, and refined the artifacts through an iterative loop with the engagement team. Each pass propagated changes through every dependent document automatically, so the PRD, architecture, and work orders stayed in lockstep instead of drifting apart. The clearer intent behind each work order is also what drove the 40% reduction in token spend during development.

4 Hand off to automated development

Once the work orders were stable, the Forge-Cursor integration picked them up for automated development, turning structured units of scope directly into working code without manual translation.

Key Outcomes

Key Outcome	Current	With Forge
Time to delivery-ready package	~8 weeks	2 hrs 21 min
Artifact set	Drafted sequentially, realigned manually	Assessment, ForgeScore, PRD, BRD, architecture, and ~50 work orders, generated with full intent and context
Requirement refinement	Workshops, email threads, change requests	Guided, iterative, in-session, spec-based development
Spec-based development	Manual translation of specs into tickets	Forge-to-Cursor native integration, with clear test cases and pre-commit hooks that check for architecture, security, and compliance drift
Token optimization	N/A	Work orders built with clear intent and test cases cut token spend by 40%

Business Impact

Velocity

The compression isn't from skipping steps. Every artifact an eight-week engagement would produce was still produced here. The gain comes from collapsing the gaps between steps, not skipping them.

Reduced Technical Debt

The ForgeScore, assessment, PRD, architecture, and work orders were all generated from the same resolved interpretation of the source. The usual drift between layers never accumulated.

Significant Cost Savings

The labor footprint of an eight-week engagement versus a single guided session is the headline number. Fewer change orders and less rework compound the savings to roughly \$250K.

Where This Applies

Teams working from dense, ambiguous source material, such as XML schemas, regulatory specs, legacy system descriptions, or vendor APIs, can expect the same shape of gains.

ABOUT FORGE

Forge is the first Intent & Context-aware Software Factory, not a faster IDE. It operationalizes intent through living specifications, persistent context, and human-auditable Work Orders, so every output is aligned, governed, and production-ready by design. Zero drift. No rework.

softwareforge.ai

© 2026 Opsera, Inc.