RE-IN-ESSENCE

WHITE PAPER · PRODUCT AND ENGINEERING

Edge to Core

Turning AI speed into product margin without breaking the Core.

An operating playbook for founder-led and sponsor-backed software companies with 30 to 500 staff.

By Colin Whitfield · Founder & Operating Partner, Re-In-Essence

Version 10 · November 2026 · www.reinessence.tech

A note on the figures

Industry studies are listed in the Sources section at the end. The financial projections come from the 25-engineer cost model in Section 10 and the worked example in Section 2, both modelled. Treat gate times, tripwire thresholds and targets as practical starting points to calibrate to your own estate.

The companion Operating Pack and Jira ticket templates are at the end of this paper, free with one email.

Executive Summary

AI has made building a working version of software cheap, but it has not made maintaining it cheap. Today, a product manager or engineer can put a working feature in front of a customer in 48 hours.

The risk sits in what happens next.

This matters most at exit. Leaving custom code in the Core turns subscription revenue into support revenue, and buyers price support like IT services. As we model in Section 2, separating the two in a EUR 20M vertical software business can move enterprise value from roughly EUR 36M to EUR 54M on the exact same top-line revenue.

In most software teams, fast prototypes reach production through **silent promotion**. A proof of concept works for one client, sales promises it to the next prospect, and prototype code quietly slides into the Core R&D budget without an architectural review, a multi-tenant test, or a commercial contract.

Our rule is simple: **never say no to the customer, just say no to funding their custom work out of R&D.**

You do not need to walk away from services revenue to protect the product. You just need a hard line between what you build for every customer and what you build for one. Building on the product architecture Andy Hooper introduced at Fabric, that boundary sits across three layers:

- **The Core** is the multi-tenant product every customer runs, holding the data model, business rules, and versioned APIs. It is built once, sold across the customer base, and owned and supported entirely by Core R&D.
- **The Enablers** are the standard SDKs, APIs, event hooks, and configuration tools that connect to the Core. They let customers tailor workflows within the Core's data structure and rules, and are built and supported as part of the platform.
- **The Edge** holds everything built for a single test or a single customer. That covers two types of work: **experiments** (time-boxed prototypes in an isolated sandbox with no SLA) and **customer-specific work** (the custom integrations, workflows, and adapted configurations one client needs). The customer owns and supports its customer-specific work, or pays Professional Services (PS) to support it under a separate statement of work (SoW). Core R&D funds and supports neither.

Every experiment leaves the sandbox at one decision point, where it is either promoted into the Core, kept at the Edge as customer-specific work, or retired.

A prototype that misses the bar for Core does not have to be shut down. It can stay at the Edge as customer-specific work, either run by the customer on the Enablers or supported by PS under a separate statement of work.

How the system works

Early gates stay light, so builders actually use them. Registering an experiment takes minutes and checking that it is safely fenced takes a day or two, so a customer still sees working software in the same week.

When a pilot gets serious, tripwires agreed in advance, not argued case by case, force one decision: where it belongs and who pays for it. Product, engineering and commercial make that call together. The Tech Lead alone then decides how the code enters the Core.

> ### The 3-Year P&L Effect (25-Engineer Baseline)
>
> Left ungoverned, 538 person-weeks (EUR 968,400) of Core R&D capacity leak into emergency rework, shelfware, and unfunded custom support over three years.
>
> Installing this system eliminates EUR 822,600 (457 person-weeks) of that waste. After roughly 1 FTE per year of governance overhead, you recover **EUR 559,800 (311 person-weeks) in net Core R&D capacity**.
>
> Separately, converting just six bespoke prototypes into customer-specific Edge contracts (at EUR 75k to EUR 200k per year) brings in **EUR 450,000 to EUR 1.2M+ per year in Professional Services revenue**. We keep that services revenue completely separate from the R&D recovery above.
>
> *Modelled on a representative 25-engineer software business, counting new work only. Custom code already in the Core is outside the model (Section 10). Calibrate the baseline to your own P&L.*

How to Read This Playbook

Reader	Priority Sections	Core Takeaway
CEOs, Operating Partners & PE Sponsors (8 mins)	Executive Summary, Sections 1 to 3, and Section 12 (First 90 Days)	How to protect SaaS exit multiples while monetising bespoke enterprise workflows through a ring-fenced PS model.
CFOs & Investment Committees (10 mins)	Section 2 (the multiple), Sections 9 to 11 (P&L Model & Examples), and Appendix A	What custom code in the Core does to the multiple, recovering 311 net person-weeks of R&D drag, and moving custom maintenance into profitable PS contracts.
CTOs, CPOs & VPs of Engineering (15 mins)	Sections 4 to 8, and Section 14 (The Horizon)	AI-era lifecycle speeds, the Core, Enabler and Edge architecture, the five gates, and keeping custom code off Core R&D budgets.

Part I · The Strategic & P&L Problem

1. Silent Promotion vs. Transparent P&L Boundaries

When building a prototype took six weeks, companies had to filter ideas before writing a single line of code. Now that AI tools can spin up a working prototype in two days, filtering up front wastes the very speed AI gives you.

The trap is silent promotion.

A weekend prototype solves a real problem for a flagship client, and sales immediately promises it to three prospects in the pipeline. Before anyone checks the plumbing, Core engineering inherits a fragile script that breaks on the next database migration. Worse still, your R&D budget ends up subsidising single-customer custom work for free.

Most of that cost never reaches a roadmap or an invoice. It is the **invisible maintenance** custom work leaves behind: the bug fixes, the single-tenant set-ups, the special branches, and the code that grows heavier with every exception.

The fix is not banning prototypes. It is making the cost of running software transparent:

- Let small pods build fast at the **Edge**.
- Put a hard commercial and technical gate between the Edge and the Core.
- Promote what many customers need into the **Core**. Give every customer the **Enablers** to configure the rest within the Core's rules.
- When a customer needs a workflow that does not belong in the Core, keep it at the Edge as customer-specific work. The customer owns it, PS only supports it under a separate SoW, and R&D never funds it.

Figure 1 · What changes

	Today, typically	With Edge to Core
Where ideas start	A spec waiting in a backlog, or a prototype already used by one customer and promised to others	A working prototype in a fenced Edge sandbox, registered on day one
What engineering receives	Words and screens to interpret, or someone else's live code	A Promotion Pack: proven behaviour in a live prototype
Who decides when it gets serious	Whoever argues hardest, case by case	Seven tripwires fixed in advance, then Gate 2
What happens to prototype code	Thrown away by habit, or shipped by silent promotion	A signed verdict: *Adopt, Refactor or Rebuild*
Where customer-specific work goes	Into the Core, maintained free by R&D	At the Edge, owned by the customer; PS support only under a separate SoW
How AI work is reviewed	By whoever is free	Three risk tiers; a named human owner for every merge

2. The Valuation Math: Protecting Both SaaS Multiples and Enterprise Revenue

What custom code does to the multiple

Take a vertical software company with EUR 20M in revenue and 10 to 20 years of custom code sitting in its Core. Most of what customers pay each year is billed as support and maintenance: paying for access, updates, and fixes to a codebase bent around each individual account. In the legacy vertical software estates we have reviewed, most recurring revenue is billed as support and maintenance, and true multi-tenant SaaS is a small fraction of the book.

In due diligence, a buyer prices that revenue mix as IT services. Services-heavy software trades at 1.0x to 2.6x revenue, whereas multi-tenant vertical SaaS trades at 3.0x to 3.8x (Aventis). When you separate the codebase and the contracts, the valuation math changes completely on the exact same EUR 20M top line:

	Revenue	Multiple	Enterprise value
As billed today, read as services	EUR 20M	1.0–2.6x	EUR 20–52M (midpoint 36M)
Separated: subscription on a multi-tenant Core	EUR 14M	3.0–3.8x	EUR 42–53M
Separated: services and customer-specific maintenance under PS	EUR 6M	1.0x	EUR 6M
Separated, total	**EUR 20M**		**EUR 48–59M (midpoint 54M)**

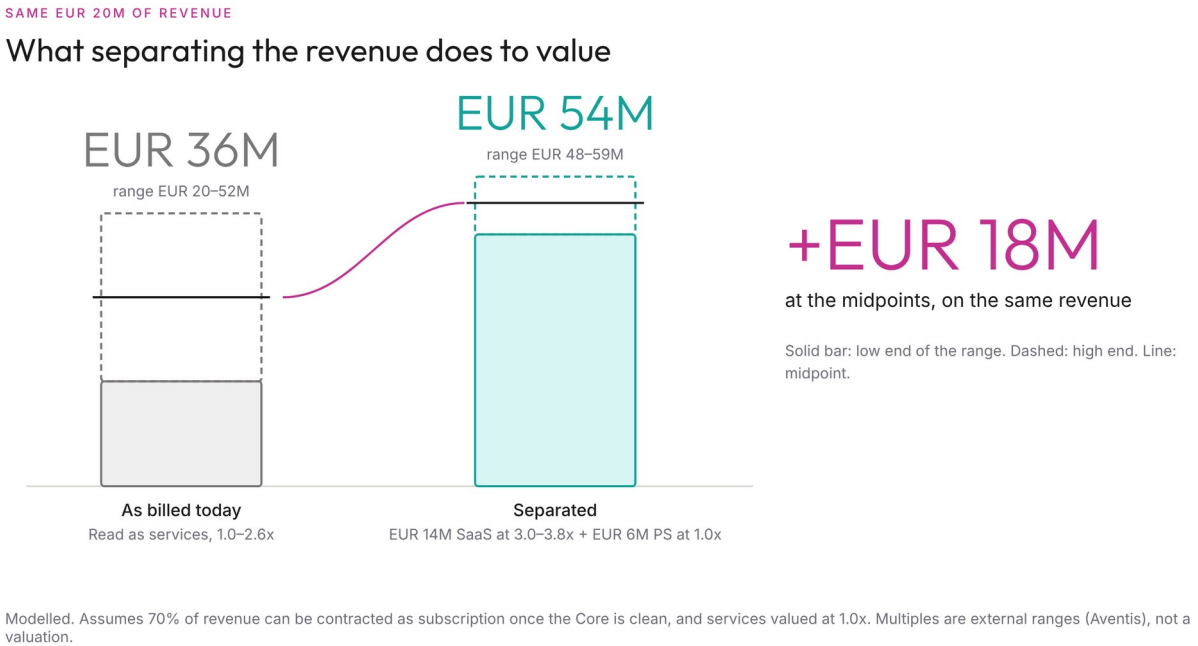


Figure 2. Same revenue, separated. Modelled; multiples are external ranges.

The result is roughly EUR 18M in additional enterprise value at the midpoint on the exact same revenue. *(This baseline model assumes 70% of revenue moves to multi-tenant subscription once the Core is clean, with the remaining 30% valued at 1.0x services multiples based on external market ranges.)*

Of course, you cannot fix this by simply relabelling invoices in the data room. Technical diligence inspects whether those subscriptions genuinely run on a single multi-tenant Core or on disguised customer forks. That is why the five gates matter for every new prototype today, and why legacy custom code in the Core eventually requires a structured rebuild (see Section 14).

What the evidence says about AI speed

- **Task speed does not equal company throughput:** Developers finish isolated coding tasks up to 55% faster in lab trials and complete 26% more tasks in field studies (GitHub/Microsoft). Yet DORA's 2024 and 2025 reports show that overall platform stability drops when review and deployment gates are not built to handle the extra volume.
- **Code churn and security drift:** Longitudinal data from GitClear shows short-term code churn and copy-pasted blocks rising sharply. Veracode found security flaws in 45% of AI-generated tasks, while Stanford researchers showed developers using AI assistants wrote less secure code while feeling more confident about it.
- **The Commercial Trap to Avoid:** Companies handle custom demand one of two ways. Some walk away from it, turning down EUR 1M in enterprise services to protect a EUR 300k licence. Others take the revenue and carry the maintenance inside R&D and support. The revenue is booked, but the custom code stays in the Core, and a buyer reads the recurring revenue as support rather than software, at the lower multiple. The answer is to take the revenue and price the boundary: services billed as services, maintenance on its own contract, and the Core kept multi-tenant. The full valuation argument is in the report listed at the end of this paper.

3. Resolving the Three-Way Executive Standoff

Every software company between 30 and 500 staff hits the same three-way standoff:

1. **The Founder or CPO** wants AI speed and feels Core engineering has become slow and defensive.
2. **The CTO and Principal Engineers** know 48-hour prototypes skip multi-tenancy and security, and they refuse to be paged at 2 AM for unvetted code.
3. **Commercial, Sales, and PS leaders** know that once an enterprise client loves a working prototype, taking it away kills the account.

A policy memo will not fix this.

You need a clear operational bargain across all three leaders. Product gets to build and test with live customers in days on the Edge. Engineering holds an unbreakable veto over how code enters Core, and Commercial gets a transparent price list to sell customer-specific Edge work whenever a feature does not fit the standard roadmap.

Part II · The Edge-to-Core Operating System

4. Three Layers: Core, Enablers and Edge

Building on the product architecture Andy Hooper introduced at Fabric, the estate is divided by who builds, who can change, and who supports each layer:

	Core	Enablers	Edge: experiments	Edge: customer-specific work
Holds	Multi-tenant product, data model, business rules, versioned APIs	SDKs, APIs, event hooks, configuration tools	Time-boxed prototypes	Integrations, workflows, adapted configurations
Builds	Core engineering	Core engineering	Edge pod	Customer, partner or PS
Customer can change	No	Configure, within Core rules	No	Yes
Supports	Vendor (under standard licence)	Vendor (under standard licence)	None (no SLA)	Customer (or PS under a separate SoW)
Paid from	Core R&D	Core R&D	Discovery budget	Customer or PS contract

Where configuration ends: Configuring standard Enabler options is part of the platform and covered by standard support. Anything custom-built or adapted on top of those Enablers for a single customer sits at the Edge.

Two tests follow from this model. First, put into the Core only what you can sell to many customers; most legacy estates have put far too much there. Second, support triage must tell a product fault from a customer's own workflow: an automated report that stops is a Core ticket if the API failed, and the customer's if their own flow did.

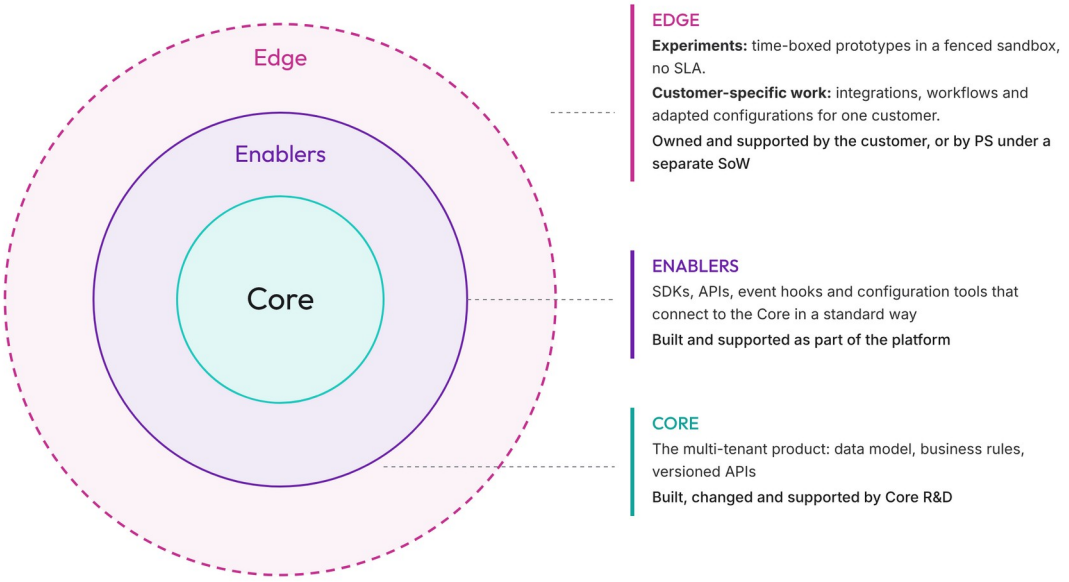


Figure 3. Three layers: Core, Enablers and Edge.

5. The 8-Stage Lifecycle (Stress-Tested for AI Speeds)

AI coding tools write code in hours, but human decisions and customer adoption still take weeks.

The stage timings below separate actual AI build speed from human and customer waiting time:

Stage	Owner	AI-Era Duration	Deliverable & Reality Check	Exit Gate / Commercial Routing
1. Signal	PM / Domain Lead	**2 hours to 2 days**	Short Edge Card defining the customer problem and target metric.	**Gate 0:** Logged in EDGE with a 30-day review clock and 90-day hard cap.
2. Edge Build	Edge Builder / Pod	**1 to 5 days** *(AI coding speed)*	Working prototype in sandbox using Enabler APIs and synthetic data.	**Gate 1 (Containment):** Steward checks data and tenant isolation (1 to 2 days).
3. Edge Pilot	Builder + PM	**1 to 4 weeks** *(Customer usage time)*	1 to 2 customers test live workflows. Time is user adoption, not coding.	Hits a **Tripwire** or day-30 check and triggers **Gate 2**.
4. Verdict	Triad (G2) & Tech Lead (AR)	**3 days (G2) + 5 days (AR)**	Commercial routing decision (G2) plus technical Architecture Verdict (AR).	**G2 routes to:** (A) Core Roadmap, (B) Customer-specific work at the Edge, or (C) Retire.
5. Core Prep	PM + Tech Lead	**2 to 5 days**	Working prototype serves as the spec; defines what goes into the Core, what into Enablers, and the non-functional requirements.	Scheduled into Core sprint capacity or handed to the PS delivery pod.

Stage	Owner	AI-Era Duration	Deliverable & Reality Check	Exit Gate / Commercial Routing
6. Core Build	Core Squad	**3 days (Adopt/Refactor) to 4 weeks (Tier 3 Rebuild)**	Hardened multi-tenant code, Enabler hooks, automated tests, and security checks.	**Gate 3 (Readiness):** Risk Tier review (Section 8) and named human merge sign-off.
7. Rollout & Cutover	PM + CS / PS	**1 to 3 weeks** *(Customer change mgmt)*	Standard SaaS users move to Core and Enablers. Customers who want to retain a bespoke workflow keep it at the Edge, either running it themselves or paying for a PS SoW.	**Gate 4:** Free pilot sandbox closed; converted to Core SLA or a customer-specific SoW.
8. Run & Optimise	Core or PS Owner	**Ongoing** *(Annual review)*	Core runs multi-tenant; the customer or PS runs customer-specific work at the Edge, with an annual cost review.	Annual check: keep the customer-specific SoW or move the customer to newer Core features.

Three Operational Traps & How the Commercial Lens Solves Them

Trap 1: The "Unfunded Sandbox" at Stage 3. A prototype misses the multi-tenant bar for Core, but an enterprise customer demands to keep it. *The Fix:* Never say no to the customer, just stop funding it from R&D. At Gate 2, it stays at the Edge as customer-specific work, run by the customer on the Enablers or supported by PS under a separate SoW.

Trap 2: The "Spec Translation Stall" at Stage 5. Product managers waste weeks writing a 20-page PRD to describe a prototype that already works. *The Fix:* The working prototype is the functional spec. Core Prep only documents what goes into the Core versus the Enablers, and the non-functional requirements.

Trap 3: The "Double-Running Tax" at Stage 7. Core ships the standardised feature, but the pilot client refuses to give up their bespoke tweaks. *The Fix:* Give the customer a straightforward choice: adopt the Core and Enabler release included in their standard SaaS tier, or sign a PS maintenance schedule priced at the true cost of running their separate workflow. Once the price is transparent, the customer makes the call.

6. Hard Tripwires, The Five Gates, and Commercial Routing

Hard Tripwires (When a free pilot stops being a free pilot)

Customer-specific work can live on the Edge long-term only if it is commercially contracted and ring-fenced.

An uncontracted discovery pilot must enter Gate 2 within 3 working days the moment it crosses any one of these seven tripwires:

1. A second customer asks for it.
2. It requires SSO, audit logs, or a formal security review.
3. It reads or writes live Core platform data outside an approved Enabler API.
4. It touches personal data (PII) or contract-sensitive commercial data.
5. Usage crosses 50 users or 3 customer departments.
6. A customer asks to include it in an SLA, renewal, or order form.
7. It hits the time limit or support drag threshold: reaching day 90 since Gate 0 (following a day-30 check), or pulling in Core engineers outside the Edge/PS pod to fix it more than twice.

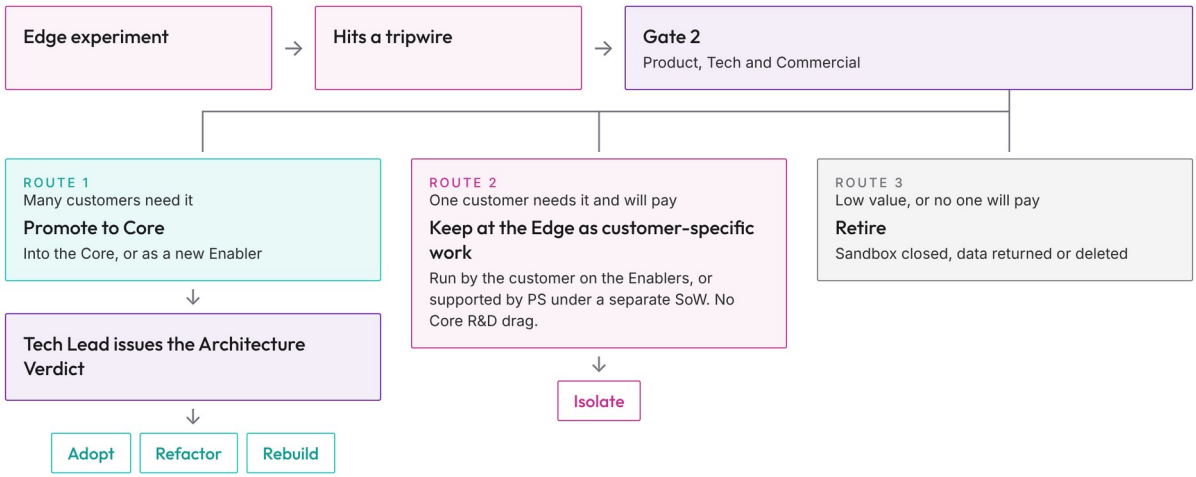


Figure 4. Gate 2 and the Architecture Review.

Gate	Decision Question	Decision Owner	Turnaround SLA	Allowed Dispositions
G0: Register	Who owns this and when is the day-30 check?	Edge Builder (Self-serve)	**15 mins**	Logged in EDGE with a 30-day review clock (90-day cap).
G1: Containment	Is it safely isolated to the sandbox and Enabler APIs?	Edge Steward	**1 to 2 working days**	Cleared for Pilot, or returned to fix the boundary.
G2: Commercial Routing	Promote to Core, keep at the Edge as customer-specific work, or retire?	Product + Tech + Commercial/PS Triad	**3 working days**	**1. Promote to Core · 2. Customer-specific work at the Edge · 3. Retire**
AR: Architecture Verdict	*How* does it enter the Core or Enablers (or isolate as customer-specific work)?	**Core Tech Lead alone**	**5 working days**	**Adopt**, **Refactor**, **Rebuild** (or **Isolate** on Enabler APIs as customer-specific work).

Gate	Decision Question	Decision Owner	Turnaround SLA	Allowed Dispositions
G3: Release	Does it pass its risk-tier checks (Tier 1 UI and reports, Tier 2 logic and integrations, Tier 3 critical; Section 8) and have a named owner?	Core Lead (or PS Lead for customer-specific work)	**Sprint cadence**	Shipped to Core general availability, or handed over under the customer-specific SoW.
G4: Cutover	Is the free pilot closed and billing or migration active?	PM + CS / PS Lead	**30 days post-G3**	Customer live on Core licence, or active under a customer-specific SoW.

Separation of Powers at Gate 2 and the Architecture Review

Gate 2 is a joint commercial and product call on *whether* a feature moves forward and *who pays for it*. The Architecture Verdict is a technical call made by the Core Tech Lead alone on *how* the code is built. Product and Commercial can retire or hold a feature, but they can never overrule a *Rebuild* verdict into *Adopt* to save time on a quarterly roadmap.

RFPs and Contracts

Answer each requirement as Core, Enabler or customer-specific work. Map customer-specific commitments into the SoW. Exclude long-term support for customer-specific deliverables unless it is quoted separately from product support. Where an RFP allows only “compliant”, keep an internal tracker of what must go into the SoW at contract.

7. Roles & Team Structure (~1 FTE Governance Overhead)

Running this model in a 25-engineer company takes roughly 1 FTE of combined governance time across existing senior staff. You do not need to hire process managers.

- **Edge Builders (PMs, Domain Leads, Forward-Deployed Engineers):** Build fast discovery prototypes on top of Enabler APIs.
- **Edge Steward (~0.5 FTE part-time Staff Engineer or Delivery Lead):** Clears Gate 1 containment in 1 to 2 days, tracks pilot clocks, and flags tripwires. The Steward checks sandbox isolation and data boundaries, not internal code style.
- **Core Tech Lead (~0.3 FTE allocation):** Owns the Architecture Verdict, oversees the Enabler API design, and ensures every Core merge has a named human owner.
- **Professional Services team (funded by customer contracts):** Builds and supports customer-specific Edge work under separate, client-funded SoWs, never out of the Core R&D budget.

8. Risk Tiers & Cutover Rules

Risk Tier	Scope & Examples	Typical Verdict	Verification & Sign-Off Required
Tier 1 (Low)	UI views, read-only reports, Enabler field and naming configs.	**Adopt** or light **Refactor**	Automated CI checks + 1 peer engineer review + named human owner.
Tier 2 (Medium)	Multi-tenant business logic, Enabler webhooks, external integrations.	**Refactor** or **Rebuild**	Multi-tenant isolation tests + Senior Engineer review + staged feature flag.
Tier 3 (Critical)	Auth/RBAC, billing ledgers, core schema migrations, PII pipelines.	**Mandatory Rebuild** in Core	Principal Architect + Security sign-off, load test, and rollback drill.

The Stage 7 Cutover Rule: Once a promoted feature ships in Core, pilot customers get 30 days to switch over to the standard release included in their SaaS tier.

If a customer prefers to keep their bespoke workflow, it stays at the Edge as customer-specific work: they sign a PS maintenance schedule or take over maintenance on top of your Enabler APIs by Day 31.

Part III · Whole-Company P&L Model (R&D + Professional Services)

9. Four Board Metrics That Matter

1. **Unfunded Silent Promotions (Target: 0):** Prototypes running in production without either a Core Gate 3 sign-off or a customer-specific SoW.
2. **Edge Conversion Split (proposed target: ~25% Core, ~10% customer-specific work at the Edge, ~65% retired):** Proves Gate 2 is filtering out weak ideas while monetising bespoke enterprise requests. This split is not externally validated. The retired share fits experiment evidence: at Microsoft, about two-thirds of tested ideas did not improve the metric they targeted (Kohavi et al., 2009). Calibrate it on your first two quarters.
3. **Gate SLA Compliance (Target: >90% within the gate times):** Gate 1 Containment within 2 working days, Gate 2 Routing within 3, and the Architecture Review within 5. Keeps governance fast enough that commercial and product teams never bypass it.
4. **PS Custom Margin vs. Core Gross Margin (Target: >20% PS margin / >80% SaaS GM):** Confirms customer-specific work is priced at full cost and never subsidised by Core R&D. Public SaaS companies with positive services margins typically run 15 to 25 points (Tunguz, 2015).

10. The 3-Year Cost & Revenue Model (25-Engineer Baseline)

To make the P&L trade-offs concrete, we model a 25-engineer software business over three years (3,450 person-weeks at a blended European loaded cost of EUR 1,800 per person-week; scale by 1.6x to 2.0x for UK, DACH, or US payrolls), producing 20 prototypes a year (60 over three years). The model counts new work only: the 60 prototypes started over those three years. Appendix A shows how each line is built.

Beyond recovering Core R&D capacity, routing roughly 10% of those prototypes (six enterprise workflows over three years) into full-cost customer-specific contracts turns an unmonitored R&D leak into a recurring profit stream:

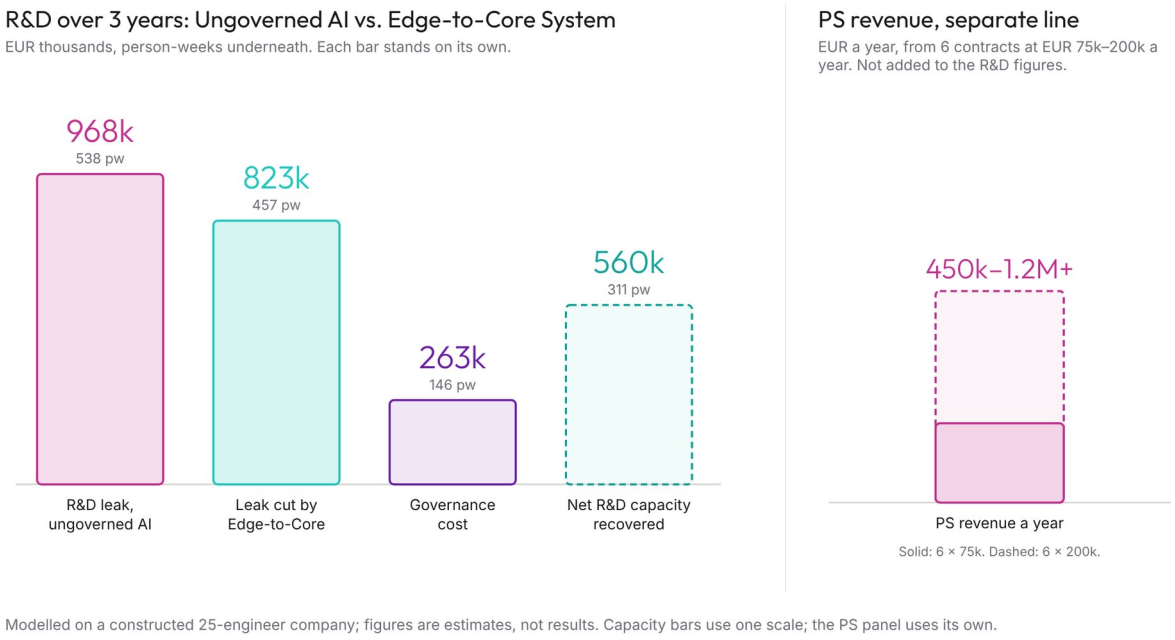


Figure 5. 3-Year P&L: R&D recovery and PS revenue, shown as separate lines. Modelled.

3-Year P&L Bucket	Ungoverned AI ("Silent Promotion")	With Edge-to-Core System	3-Year Net P&L Impact
1. Core R&D Emergency Rework *(Fixing leaked prototypes in production)*	-252 person-weeks (-EUR 453,600)	0 unplanned leak weeks *(Planned rebuilds sit on the roadmap)*	**+252 pw saved** (+EUR 453,600)
2. Unvalidated Core Shelfware *(Full Core builds for low-adoption features)*	-192 person-weeks (-EUR 345,600)	-60 person-weeks (-EUR 108,000, filtered at G2)	**+132 pw saved** (+EUR 237,600)
3. Support Drag on Core* *(invisible maintenance: bug fixes, single-tenant set-ups, code bloat)*	-94 person-weeks (-EUR 169,200)	-21 person-weeks (-EUR 37,800; rest moved to PS under customer-specific SoWs)	**+73 pw saved** (+EUR 131,400)
Gross Core R&D Leak / Recovery	**-538 person-weeks (-EUR 968,400)**	**-81 person-weeks (-EUR 145,800)**	**+457 pw gross saved (+EUR 822,600)**

3-Year P&L Bucket	Ungoverned AI ("Silent Promotion")	With Edge-to-Core System	3-Year Net P&L Impact
4. Less: Governance & Enabler Investment *(~1 FTE/yr Steward, AR & sandbox setup)*	0 formal *(Hidden in rework chaos)*	-146 person-weeks (-EUR 262,800)	**-146 pw investment (-EUR 262,800)**
Net Core R&D Capacity Recovered	**-538 pw lost (~3.9 FTE/yr)**	**Protected Core Roadmap**	**+311 net pw recovered (+EUR 559,800)**
5. Separate PS Revenue Stream *(6 customer-specific Edge contracts at EUR 75k to 200k/yr, kept separate from R&D savings above)*	Given away free inside R&D, or walked away from the deal	Billed on a dedicated PS retainer or customer self-serve contract	**+EUR 450,000 to EUR 1.2M+ per year in PS revenue**

* New work only. Custom code already in the Core is a larger drag; Section 2 shows its effect on value.

11. Two Modelled Examples

- **Case A: 75-Person Founder-Led Vertical SaaS (18 Engineers).** The technical co-founder built six custom routing prototypes in Cursor for key accounts. Maintaining them for free carried invisible maintenance: bug fixes, single-tenant set-ups, and code that grew with every exception. Together with older custom code, it took 35% of every Core engineering sprint: 12 points from the six new prototypes and 23 points from custom code built up over earlier years. *The Fix:* Installed Gate 2 and the Enabler layer in 60 days. Two prototypes were rebuilt cleanly into Core, two were absorbed into standard Enabler configurations, and two stayed at the Edge as EUR 85k a year customer-specific PS contracts. New drag from prototypes fell from 12 points to under 2 within two quarters. The 23 points from older custom code stay until the Core is rebuilt, which is a separate programme with its own business case. With AI, that rebuild is now within reach; request the operating model and the case study to see its effect on margin and multiple.
- **Case B: 280-Person PE-Backed Software Platform (65 Engineers).** Pre-exit technical diligence flagged undocumented bespoke code paths that were dragging gross margin down by 6 percentage points. Walking away from custom work meant losing flagship accounts, including one client paying EUR 1M a year in services on top of EUR 300k in SaaS licences. *The Fix:* Moved all custom extensions onto strict Enabler APIs as customer-specific work under a ring-fenced PS business unit, while locking the Core platform behind Gate 3 Risk Tiers. The company kept 100% of its enterprise services revenue and presented buyers with a clean, multi-tenant Core SaaS P&L.

Both cases are modelled examples, not client results.

Part IV · Execution & The Horizon

12. The First 90 Days: Installing the System Without Stopping Delivery

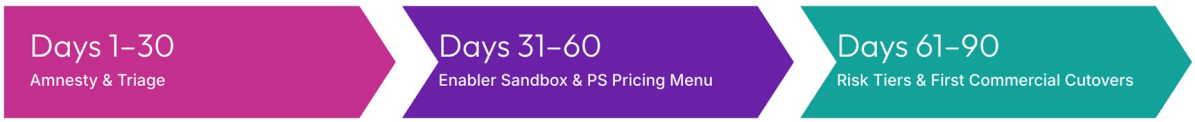


Figure 6. 30-60-90 day rollout.

Window	Key Actions	Concrete Exit Milestone
Days 1 to 30: Amnesty & Triage	Run a two-week no-blame Prototype Amnesty; log every live script or POC in EDGE; appoint the part-time Edge Steward; publish the 7 Tripwires.	100% visibility of active prototypes on one board; zero unlogged scripts touching production databases.
Days 31 to 60: Enablers & PS Pricing	Stand up the Edge sandbox and top 3 Enabler APIs; publish the standard PS rate card for customer-specific Edge work; run the first Gate 2 and Architecture Review verdicts.	First 3 pilots routed inside the gate times (Gate 1 Containment within 2 working days, Gate 2 Routing within 3, Architecture Review within 5); Sales and PS armed with standard customer-specific SoW terms.
Days 61 to 90: Cutover & P&L Split	Enforce the risk tiers on all Core pull requests: Tier 1 (UI and reports), Tier 2 (business logic and integrations), Tier 3 (auth, billing, schema, personal data). Run the first Stage 7 Rollout & Cutover moves to the standard Core release or a customer-specific SoW; retire the first failed pilot. Put new customers on the model from first contract, and existing customers at renewal.	Zero unfunded custom code in Core R&D; first monthly 4-metric scorecard delivered to the board.

13. Four Questions Every Founder & CEO Will Ask

- **"Won't five gates slow down our AI speed?"** No. Gate 0 takes 15 minutes and Gate 1 takes 1 to 2 working days, so teams still put working software in front of a customer in the same week.
- **"What if a flagship enterprise customer refuses the standard Core feature and demands their custom workflow?"** It is a hard conversation, and it should not be the first one. Before any customer hears a price, do the cost-benefit work on each strategic account: revenue, margin, the real cost of their custom code, and their weight on the roadmap. Bring the largest accounts into the transformation plan early. Re-confirm what they need now, not what was written down years ago, and show them their own workflow running on the new Core. Then offer two or three priced options: the standard release, a configuration on the Enablers, or customer-specific work at the Edge under a PS contract. Customers accept change when they can see what they gain: faster releases, fewer outages, a product that keeps up. Some strategic accounts will justify an exception. Make it a decision with a price, not a default.
- **"Why rebuild a working prototype at Stage 6?"** Because an Edge prototype optimises for 48-hour discovery, while Core optimises for 5-year multi-tenant margin and security. Even when the Tech Lead rules Rebuild, the prototype saved weeks of written specs and proved commercial demand first.
- **"Can the founder still code prototypes on weekends?"** Yes. Founder prototypes are often the highest-signal experiments in the business. They just enter Gate 0 on Monday morning and follow the same commercial and architectural routing as everyone else.

14. The 12-to-24 Month Horizon: Why a Clean Core Matters More Than Ever (And What AI Cannot Do)

Be realistic about what AI can and cannot rebuild.

Re-architecting the underlying data schema and core platform of a mature EUR 9M+ software product still takes 12 to 18 months of hard graft, and that is before customer migrations. Typing the code was never the slow part. Untangling legacy customer data, reconciling edge-case requirements across hundreds of live accounts, and managing human change cannot be prompted away.

Where AI changes the game over the next 24 months is **feature velocity on top of a clean Core**:

- **Spaghetti in, faster spaghetti out:** If your Core is undocumented and polluted by silent promotions, coding agents will only generate technical debt faster.
- **Fast assembly on clean boundaries:** When your Core has documented schemas, strict tenant isolation, and a structured Enabler layer, AI coding agents can already build modular features that pass *Adopt* or light *Refactor* verdicts in days rather than weeks.
- **What remains stubbornly human:** Even as data migration scripts get faster to generate, **customer migration is still the most expensive bottleneck in enterprise software**. Getting an operations director to change their team's daily workflow, sign off on a cutover, and choose between standard Core or paid customer-specific work is a commercial and human governance problem.

Companies that master the Edge-to-Core boundary today will be the ones that turn AI speed into compounding enterprise value tomorrow.

Appendix A · How the Cost Model Is Built

Every line below is modelled. The external studies in the last column are cross-checks on the order of magnitude, not proof of the inputs.

Line	Calculation	Result	External cross-check
Capacity	25 engineers × 46 working weeks × 3 years	3,450 pw	—
Rework, ungoverned	21 of 60 prototypes leak into production (35%) × 12 pw to stabilise each (75% of a 16 pw full Core build)	252 pw	None. A stated assumption.
Shelfware, ungoverned	24 prototypes built to full Core × 16 pw × 50% low adoption	192 pw	Pendo (2019, 615 products): 80% of features rarely or never used. Our 50% is lower.
Shelfware, with the system	15 promoted (25%) × 16 pw × 25% low adoption	60 pw	—
Support drag	6 customer-specific workflows × about 5.2 pw a year × 3 years; about 7 pw a year left in Core with the system	94 pw → 21 pw	Stripe (2018): developers spend 17.3 hours a week on technical debt, bad code and maintenance.
Governance	~1 FTE a year (46 pw) × 3 + 8 pw sandbox and Enabler set-up	146 pw	—
Gross leak	252 + 192 + 94	538 pw (15.6% of capacity)	McKinsey (2020): CIOs say 10–20% of new-product budget is diverted to technical debt.
Net recovery	538 − 81 − 146	311 pw (EUR 559,800)	—
PS revenue (separate)	6 customer-specific contracts × EUR 75k–200k a year	EUR 450k–1.2M+ a year	Never added to the R&D lines.

Sources & Benchmarks

1. **DORA / Google Cloud (2024 & 2025):** *Accelerate State of DevOps Reports* (AI task acceleration versus delivery stability). https://dora.dev/research/2024/dora-report/ · https://dora.dev/research/2025/dora-report/

2. **GitHub / Microsoft Research (Peng et al., 2023; Cui et al., 2024):** Controlled and field trials showing up to 55.8% faster task completion and 26% higher task volume with AI coding assistants. https://arxiv.org/abs/2302.06590 · https://papers.ssrn.com/sol3/papers.cfm?abstract_id=4945566

3. **GitClear (2024 & 2025):** Longitudinal studies on rising short-term code churn and copy-pasted blocks. https://www.gitclear.com/ai_assistant_code_quality_2025_research

4. **Veracode (2025) & Stanford University (Perry et al., 2023):** Security flaws in 45% of AI-generated code tasks, and higher developer overconfidence. https://arxiv.org/abs/2211.03622

5. **Aventis Advisors (2024–2026):** EV/Revenue multiples for multi-tenant vertical SaaS (3.0x to 3.8x) versus services-heavy software (1.0x to 2.6x). https://aventis-advisors.com/software-valuation-multiples/

6. **McKinsey (2020):** *Tech debt: Reclaiming tech equity*, survey of 50 CIOs. https://www.mckinsey.com/capabilities/mckinsey-digital/our-insights/tech-debt-reclaiming-tech-equity

7. **Pendo (2019):** *Feature Adoption Report*, 615 software products. https://www.pendo.io/resources/the-2019-feature-adoption-report/

8. **Stripe (2018):** *The Developer Coefficient*. https://stripe.com/files/reports/the-developer-coefficient.pdf

9. **Kohavi et al. (2009):** *Online Controlled Experiments*, Microsoft ExP. https://ai.stanford.edu/~ronnyk/2009-06-28KDDTutorialT4part1.pdf

10. **Tunguz (2015):** Professional services margins at 13 public SaaS companies. https://www.tomtunguz.com/professional_services_margins/

11. **Hooper, A. (Fabric):** The *Core, Deployment Enablers and customer-specific work* product architecture this paper builds on.

About the Author

Colin Whitfield is Founder and Operating Partner of Re-In-Essence. He works with founder-led and private-equity-backed B2B software companies on scaling, operational turnarounds, target operating models (R&D and Professional Services), and AI delivery governance.

WHERE THIS PAPER SITS

How the full playbook fits together

For product, engineering and AI leaders, this paper is your starting point. For founders and sponsors, start with Report 01 below.

01

The Thesis

Closing the Delivery Discount. Why the discount exists, and how to detect it

Read the report →

02

The Scaling Ladder

Where it forms, stage by stage

Read the report →

03

The Operating Model

Portfolio OS, Scale OS or Founder OS

Request the operating model →

04

The Modelled Case

The arithmetic, entry to exit

Request the modelled case →

Companion: the Edge to Core Operating Pack and Jira templates. **Download →**

RE-IN-ESSENCE

WORK WITH US

Want an independent read on your own software estate?

We map where custom code is sitting in your Core, what it is costing your engineering team every sprint, and how buyers will price it at exit. From there, we set out the gate structure, the Enabler boundary, and the commercial case for a clean Core. Most engagements start with our **Two-Week Read**: two weeks, no fee.

Book a call →

www.reinessence.tech · hello@reinessence.tech