RESEARCH REPORT

The Connective Tissue: Why Enterprise Integration Decides How Fast Your Business Can Move

Every process that matters crosses systems that were never designed to talk. How the roads between your systems get built, and why they just became the foundation for AI agents.

SDTC Digital · What We Think

September 2026 · Approx. 5,000 words

PART 1

Content Metadata

Content type	Research Report
Title	The Connective Tissue: Why Enterprise Integration Decides How Fast Your Business Can Move
Slug	enterprise-integration
Meta title	Enterprise Integration: A Research Report | SDTC Digital
Meta description	Every process that matters crosses systems that were never designed to talk. Our research report explains enterprise integration in plain language, and why it just became the foundation for AI agents.
Excerpt / card summary	Your business runs on dozens of systems bought at different times for different reasons, and every process that matters crosses several of them. This report explains how integration works, why point-to-point wiring becomes a trap, and why the connections you build now decide what AI can do for you next.
Hero image direction	An editorial aerial photograph of a city at dusk where distinct districts are visibly linked by illuminated roads and bridges, muted tones with warm light tracing the connections. It carries the report's metaphor: an enterprise as a city of buildings that must be deliberately connected. No circuit boards, no network diagrams.
Card image direction	A simple flat graphic of scattered squares joined first by a tangle of crossing lines, resolving on one side into a clean hub-and-spoke pattern, in SDTC's brand palette. Legible at thumbnail size beside the "Research Report" category label.
Primary audience	CTOs, CIOs, enterprise architects, SaaS founders and product leaders, operations executives, and decision-makers responsible for making business systems work together.
Search intent	Informational and evaluative: "enterprise integration," "what is an iPaaS," "API integration strategy," "point-to-point vs platform integration," "integration center of excellence," "event-driven architecture."
Suggested related articles	1) The Simplest Thing That Works (Research Report). 2) Fit for Purpose (Research Report). 3) Paved Roads (Research Report). 4) Legacy Is a Condition, Not an Age (Research Report). 5) A Guide to designing an API strategy.

PART 2 · REPORT BODY

The Connective Tissue: Why Enterprise Integration Decides How Fast Your Business Can Move

Executive Summary

Picture your company as a city. Finance is one building, put up fifteen years ago. Sales lives in a gleaming new tower it rented last year. The warehouse runs on something older than most of its staff. Each building works. But every process that actually matters, taking an order, paying a supplier, onboarding an employee, answering a customer, has to travel *between* buildings. And nobody, originally, built the roads.

Enterprise integration is the discipline of building those roads: the technologies, platforms, and practices that connect applications, data, and processes across an organisation's technology landscape so that work can flow end to end. It is among the least glamorous subjects in enterprise technology, and, as this report's research makes clear, among the most decisive. The goal has been stable for four decades, since the earliest days of computer-integrated manufacturing, and one academic definition in our sources states it perfectly: ensure that *the right people and the right processes have the right information and the right resources at the right time*. What has changed is the stakes.

Three findings organise the evidence. First, **integration failure is a business failure wearing a technical costume.** Where systems don't talk, the symptoms are commercial: inconsistent data, processes that break at handoffs, armies of people re-keying information between screens, and decisions made on stale numbers. The remedy for each is a road that was never built.

Second, **the way you connect matters more than whether you connect.** The trap the sources warn about unanimously is point-to-point integration: direct, custom wires between each pair of systems, quick to build, and brittle at scale, because ten systems can require dozens of separate wires, each breaking whenever anything changes. The mature pattern is platform thinking: shared, reusable connections, standard interfaces (APIs), event-driven messaging, governed flows, treated as long-term managed assets rather than one-off plumbing jobs, often stewarded by an integration centre of excellence.

> CENTRAL FINDING
>
> **Integration has just become the foundation of the AI era.** AI agents can only observe events, access data, and act across applications through standardised, governed connections. The forty-year-old goal, right information, right place, right time, now has a new class of recipient: software that acts. The roads you build this year decide what can drive on them next.

The Context

No enterprise ever chose fragmentation. It accumulated, one sensible decision at a time. The finance system was selected a decade ago for good reasons. The CRM arrived with a new sales leader. The warehouse system came with an acquisition. Each cloud service was adopted because it solved a real problem quickly, and the sources note that this is precisely the modern pattern: as organisations adopt more cloud services, digital tools, and specialised applications, their landscapes become increasingly complex, systems introduced at different times, for different needs, now expected to work together.

The expectation is the new part. For most of computing history, systems ran their own domains and people carried information between them, on paper, by phone, through re-keying. The integration ambition is old (the manufacturing world pursued it as "computer-integrated manufacturing" from the 1980s, and a US federal law, the Enterprise Integration Act of 2002, tasked NIST with advancing it), but for decades, partial integration was survivable, because processes moved at human speed and humans filled the gaps.

Three shifts ended that tolerance. **Speed**: digital business runs processes in minutes that once took days, and a process that stalls at a system boundary now stalls visibly, in front of customers. **Scale**: the number of systems exploded, a mid-sized company now runs dozens of applications, an enterprise hundreds, and the gaps between them multiplied faster than the systems themselves. **Automation**: once work is automated, there is no human in the corridor carrying the folder between buildings; automation, as the sources put it, depends on the ability of systems to work together reliably, which makes integration the precondition for every efficiency programme now underway.

The vocabulary of the field, translated once, plainly: an **API** (application programming interface) is a service counter with a defined menu, a standard way for one system to request data or action from another without knowing anything about its internals. **Messaging and events** are the bulletin-board alternative to phone calls: instead of one system calling another directly and waiting, a system posts an announcement ("order placed") and any interested system reacts in its own time. **Middleware** is the interpreter layer that sits between systems so each can speak its own language. An **iPaaS** (integration platform as a service) is the rentable toolkit, design, run, monitor, and govern all these connections from one place. And **EDI/B2B integration** extends the roads beyond the city limits, to suppliers, partners, and customers.

The newest context is the one that reframes everything: AI. The sources converge with our earlier research from both directions. Analytics and AI depend on unified, timely data, which is integration's product. And agentic AI, software that acts rather than merely answers, can only act *through* connections: observing events, reading governed data, invoking applications. The city's roads are about to carry a new kind of traffic, and their condition has never mattered more.

The Core Problem

The core problem of enterprise integration can be stated as a paradox: **every organisation is already integrated, badly, and the bad integration is invisible until it is expensive.**

No company actually runs on disconnected systems; work does flow between them, somehow. The question is *how*. In the unmanaged default, the "how" is a museum of improvisations: spreadsheets exported nightly and imported manually; a customer-service agent with six screens open, re-keying between them; a custom script written by a developer who left in 2019, silently copying records at 2 a.m.; an intern whose entire job is reconciling two systems' versions of the truth. Each improvisation was rational. Together they form what might be called the *invisible integration layer*, undocumented, unowned, unmonitored, and load-bearing.

The costs of this layer arrive disguised as other problems. **Data inconsistency**: three systems, three versions of the customer, and every report a negotiation, the same silo pathology our data-readiness report traced to its AI consequences. **Broken handoffs**: processes that stall at boundaries, the order captured but not fulfilled, the employee hired but not provisioned, each stall a customer experience or a compliance risk. **Manual workarounds**: skilled people spending their days as human middleware, with error rates and burnout to match. **Decision latency**: leaders steering on last month's numbers because assembling this month's requires archaeology. The sources' summary is exact: delayed information, inconsistent data, manual handoffs, and limited visibility, which slow execution, increase operational risk, and make it harder to respond to change.

The second layer of the problem appears when organisations do start connecting deliberately, and connect wrong. The **point-to-point trap** is the unanimous warning of the sources: direct, custom connections between each pair of systems that needs to talk. Each individual wire is quick to build and perfectly reasonable. The trap is arithmetic: connections grow with the *pairs* of systems, not the systems, ten systems can need dozens of wires, and every wire embeds assumptions about both ends. Change one system and an unknown number of wires break. The landscape becomes, in the industry's own affectionate term, spaghetti: brittle, undocumented, terrifying to touch, and, note the echo, a form of the technical debt our earlier report priced, accruing interest on every change.

The third layer is organisational, and by now familiar from every report in this series: **connections have no owner.** Integrations are built inside projects, by project teams, for project deadlines, and then the project ends. Nobody owns the flow; nobody monitors it; nobody documents it; nobody plans its lifecycle. The sources' best-practice language, treat integrations as managed assets, with ownership, documentation, and lifecycle management, is a precise negative image of the default reality, in which the enterprise's connective tissue is nobody's job.

Assembled: an invisible, improvised integration layer carrying the business; deliberate connections built in the pattern that scales worst; and no ownership of any of it. That is the condition into which companies are now attempting to introduce automation and AI agents, which is precisely why the subject has stopped being back-office.

What Leaders Often Get Wrong

Treating integration as an afterthought of system selection. The standard sequence: choose the application for its features, sign, and then discover what connecting it will cost. Integration effort is routinely the largest unbudgeted line in enterprise system projects, and "it has an API" on a datasheet is not an integration plan. The mature sequence reverses the emphasis: how a system will connect, to your data, your processes, your identity and security model, is a first-class selection criterion, weighted alongside features.

Buying wires when the business needs roads. Under deadline, the point-to-point connection is always cheaper *this quarter*, and the platform approach, shared interfaces, reusable flows, an integration layer, is always cheaper *from the third connection on*. Leaders who fund integration only inside individual projects guarantee the spaghetti outcome, because no project has an incentive to build the road; each needs only its own wire. Escaping the trap requires funding the shared layer as infrastructure, exactly the platform lesson our Paved Roads report drew for internal developer platforms.

Confusing data movement with process integration. Copying records between systems nightly keeps databases loosely aligned; it does not make a *process* flow. The sources distinguish carefully: data integration unifies information; process integration orchestrates work across systems, and the second is where customer-visible value lives. An organisation can have synchronised databases and still make customers wait three days for a handoff a real integration would complete in seconds.

Under-governing until the audit, over-governing after. The two failure modes of integration governance mirror each other. In the first, anyone connects anything: credentials in scripts, partner access unreviewed, data flowing across borders no one mapped, until a breach or audit exposes the seams. In the second, burned once, the organisation routes every connection through a central approval gauntlet, and delivery stalls while shadow integrations bloom around the blockage. The sources' phrasing of the resolution is worth adopting verbatim: governance as guardrails, not rigid constraints, standards, secure defaults, and monitored self-service, the same balance every report in this series has found at every layer.

Dismissing events as exotic. Most organisations default every connection to request-and-wait (one system calls another and blocks until it answers), because it is the intuitive pattern. But many business facts are broadcasts, not conversations: an order placed, a shipment delayed, a payment received, events that many systems care about. Event-driven integration, publish once, let subscribers react, decouples systems, absorbs bursts, and, per the sources, is precisely the pattern that lets AI agents *observe* the business in real time. Leaders who treat it as advanced plumbing are deferring the pattern the next decade runs on.

Measuring projects, never the capability. Integration is judged connection by connection, was the project on time?, and never as a capability with a maturity. The sources offer the better instruments: how much is reused rather than rebuilt; how long a new integration takes to design and deploy; how reliably the existing ones run; how easily they adapt when systems or regulations change. By those

measures, most organisations have no idea where they stand, which is why they keep repurchasing the same connections project after project.

The Evidence and Patterns

The taxonomy: one discipline, several jobs

The sources agree on the map of integration types, which is usefully read as a list of distinct jobs rather than competing philosophies. **Application integration** makes systems coordinate actions in real time (the order system tells the fulfilment system, now). **Data integration** unifies information across sources for operations and analytics (one view of the customer, at last). **Process integration** orchestrates end-to-end workflows across many systems (quote to cash, hire to onboard). **B2B/EDI integration** extends flows to partners, suppliers, and customers, the oldest form, and still the backbone of supply chains. **Cloud integration** bridges the hybrid reality of on-premises and cloud estates. **API integration** and **platform integration (iPaaS)** are the *means* that increasingly carry all of the above; **device integration** joins the physical world's sensors and machines to the party. A mature landscape uses most of these, deliberately matched to jobs, the fit-for-purpose habit, again.

The pattern consensus: reuse, decouple, govern

Across SAP's and IBM's best-practice material, the engineering consensus reduces to three verbs. **Reuse**: build integrations as shared assets, an API published once and consumed by many, an event stream any authorised system can subscribe to, rather than one-off wires; reuse is simultaneously the cost model and the speed model. **Decouple**: prefer loose connections (stable interfaces, asynchronous events) over tight ones, so systems can change, or be replaced, without dragging their neighbours down; this is what makes the architecture survivable, and it is the same seam-thinking our modernization report prescribed for renewing products without stopping them. **Govern**: secure data exchange, controlled access, versioned interfaces, monitored flows, and lifecycle management, because connections carry the business's most sensitive material across its most exposed seams, and because the automated and AI-enabled interactions now arriving must operate within established guardrails.

The organisational pattern: the integration centre of excellence

The sources' answer to the ownership vacuum is the **integration centre of excellence (ICoE)**, and its design brief repays attention because it deliberately avoids the failure modes our platform report catalogued. An effective ICoE is *not* a central delivery team through which all integration flows (the bottleneck), nor a standards police (the ivory tower). It defines shared patterns and standards; provides common platforms and tools; curates reusable assets; governs security and lifecycle; and enables other teams to build well, a paved-roads function for the connective tissue, evolving from basic standards toward optimisation and advanced automation as maturity grows. The maturity indicators the sources attach, asset reuse, time-to-integrate, reliability, adaptability, double as the capability scorecard leaders currently lack.

The proof cases

The named examples, with the usual caution that they are vendor-published, illustrate the shape of returns. **MNG Kargo**, a Turkish delivery company riding an e-commerce boom, automated its partner connections through managed APIs and a developer portal, letting e-commerce platforms connect themselves, seamless flow from sale to delivery, and then hosted a hackathon where third parties proposed service enhancements on those same APIs: integration as a growth surface, not plumbing. **Helsinki Regional Transport (HSL)**, serving 1.5 million people, rebuilt its ticketing integration on a modern platform and completed the migration, during a pandemic, without unplanned outages: integration engineering as continuity engineering. Both stories end the same way, pointedly: next step, microservices; next step, AI on the collected data. Integration was the foundation each ambition stood on. SAP's named customers, a global instrumentation manufacturer coordinating manufacturing, logistics, and finance; a Nordic retailer aligning inventory and customer data across channels; a construction giant coordinating multi-partner projects, repeat the moral across industries.

The agentic horizon

The most forward-leaning evidence deserves its own weight. SAP's analysis is unambiguous: agentic AI *depends* on enterprise integration, agents observe events, access data, and act on applications only through standardised APIs, event streams, and governed access, and integration supplies the security, lifecycle, and governance that make agent participation responsible. Read alongside our platform report's "agents are revealing users" finding and our architecture report's agent guardrails, a consistent picture forms: the enterprise that wants AI to *do* things, not merely draft things, is really asking whether its systems expose clean, governed, machine-usable connections. The roads decide the traffic.

Strategic Implications

Reframe integration spend as capability investment, and inspect it like one. The line items scattered across every project budget, connectors, custom scripts, middleware licences, integration consulting, are, summed, one of the larger technology investments most enterprises make, and among the least examined. Pulling them into view as a single capability, with the maturity scorecard (reuse, speed, reliability, adaptability) reported to leadership, converts invisible plumbing into a managed asset and reveals whether the organisation is compounding connections or endlessly repurchasing them.

Make integration readiness a gate for AI ambition. The strategic sequence the evidence supports: before funding agent-class AI initiatives, audit whether the processes they would act on are reachable, are the systems exposed through governed APIs, are the business events published anywhere, is access controlled at the right granularity? Where the answer is no, the integration work *is* the AI programme's first phase, and budgeting it as such avoids the expensive-experiment fate our whole series has documented for AI built on unready foundations.

Treat your API surface as a product with customers. The MNG Kargo lesson generalises: interfaces built for internal integration become, with modest additional care, a growth surface, partners

self-connecting, ecosystems forming, third parties adding value you didn't have to build. For SaaS operators this is doubly true: in B2B software, integration capability is now a purchase criterion, buyers ask what you connect to before they ask much else, and a well-designed public API with a developer portal is sales collateral, retention mechanism, and platform strategy in one artefact.

Fund the shared layer centrally; deliver at the edges. The point-to-point trap is an incentive problem before it is a technical one: no single project rationally builds the road. The resolution is structural: fund the integration platform, the standards, and the ICoE as shared infrastructure (the capability), while product and process teams build their own integrations on it (the delivery). Central enablement, distributed execution, the pattern that keeps both the bottleneck and the spaghetti at bay.

Read integration health as M&A due diligence, both ways. Acquisitions are integration events: the value thesis of most deals assumes systems and processes will join, and the cost of joining spaghetti to spaghetti has sunk many a synergy case. Buyers should probe the target's integration landscape (and their own readiness to receive it); sellers should recognise that a clean, documented, API-fronted estate is worth real money at diligence, the same demonstrability dividend our debt and modernization reports identified.

Use regulation as the forcing function it is. Data-protection law, financial-services rules on operational resilience, and AI regulation all reach through integrations: where data flows, who can access what, how actions are logged. Governed integration converts these from recurring emergencies into standing compliance, and, as our data report argued for governance generally, the discipline pays for itself in speed: pre-permissioned, documented flows are faster to build on, not slower.

Technical Implications

Establish the API layer as the stable contract. The foundational move: front core systems, especially the legacy ones, with clean, versioned, documented APIs, so consumers depend on stable contracts rather than internal structures. This is the same API-wrap pattern our modernization report found dominant, serving double duty: it decouples today's integrations and creates the surface tomorrow's agents require. API management (publication, security, versioning, monitoring, developer experience) is the discipline that keeps the layer trustworthy as it grows.

Add the event backbone for the facts many systems care about. Identify the business events, order placed, shipment dispatched, payment failed, customer churned, and publish them once to a shared stream that authorised systems subscribe to. This inverts the integration burden (producers announce; consumers self-serve), absorbs load spikes gracefully, and gives analytics and AI a real-time nervous system to observe. Request-response APIs and event streams are complements, not rivals: conversations for questions, broadcasts for facts.

Adopt a platform, but resist platform maximalism. An iPaaS earns its keep by centralising design, monitoring, connectors, and governance, the sources are clear on the value. The caution from every platform experience in this series applies: the platform is the means; the assets (APIs, events,

flows) and the standards are the capability. Choose for connector coverage of *your* estate, governance fit, and the ability to keep assets portable; avoid re-creating point-to-point spaghetti *inside* the platform, which is merely tidier spaghetti.

Engineer the seams for failure, because seams fail. Integrations cross networks, organisations, and clouds; partial failure is their normal weather. The reliability kit is the distributed-systems trio our architecture report named, retries with idempotency (safe to repeat), timeouts, graceful degradation, plus dead-letter queues for messages that cannot be processed, reconciliation jobs that verify the books balance across systems, and, above all, monitoring with alerting: the silent broken integration, discovered by a customer or an auditor, is the canonical incident of this domain.

Secure the connective tissue as a first-class surface. Integrations concentrate exactly what attackers want: credentials with broad access, data in motion, and paths between systems. The standing disciplines: no credentials in code (use vaults and managed identities); least-privilege scopes per integration, reviewed on a cadence; encryption in transit everywhere; partner access isolated and audited; and the lifecycle enforced, decommissioned integrations actually decommissioned, because the forgotten flow with live credentials is the breach that writes itself. Agent-era traffic raises every stake: the identity, budget, and audit machinery our platform report prescribed for agents rides on this layer.

Build the catalogue, because you cannot govern what you cannot list. The unglamorous prerequisite for everything above: an inventory of integrations, what connects to what, carrying which data, owned by whom, monitored how. Most organisations cannot produce this list today; the invisible integration layer is invisible precisely because nobody wrote it down. The catalogue converts spaghetti into a portfolio, the first step of every escape plan, and the twin of the metadata investment our data report demanded.

Decision Framework

Five questions, asked in order, turn integration from accumulated accident into managed capability. They work at both scales: a single new connection, or the enterprise programme.

Question 1: What does the business process need, end to end?

Start from the process, not the systems: map the flow (order to fulfilment, hire to productive, claim to settlement), the handoffs where it crosses systems, and where it currently stalls, re-keys, or breaks. Pass condition: the integration need stated as a process outcome ("orders reach the warehouse in seconds, with one version of the truth"), not as a technology wish ("connect A to B").

Question 2: Which pattern fits each connection?

Match the job to the tool: request-response APIs for questions and commands; events for facts many systems care about; bulk data movement for analytical unification; B2B channels for partners; and orchestration where a process must be actively coordinated across steps. Pass condition: each

connection in the design names its pattern and why, with "because it was quickest" flagged for what it is, borrowing.

Question 3: Is this a wire or a road?

The reuse test: will other consumers ever need this data, this event, this capability? If plausibly yes, build the shared asset (published API, subscribable event) even at modest extra cost; if genuinely no, a scoped connection is honest, recorded as such in the catalogue. Pass condition: the wire-vs-road decision is explicit, and every road built is registered where the next team will find it.

Question 4: Who owns it, and under what rules?

Every integration gets an owner, documentation, monitoring, and a lifecycle (versioning plan, deprecation path); the shared layer gets its ICoE-style stewardship, standards, platform, security defaults, enablement, governing as guardrails rather than gates. Pass condition: the catalogue entry is complete at go-live, and a named person will be paged when the flow breaks, because it will.

Question 5: How will we know the capability is improving?

Instrument the maturity scorecard: reuse rate (how often new needs are met by existing assets), time-to-integrate (idea to production for a standard connection), reliability (error rates, silent-failure detection), and adaptability (cost of the last system swap or regulatory change). Pass condition: the scorecard exists, is reviewed on a leadership cadence, and at least one investment decision per year cites it.

A standing check binds the framework, and points it forward: **could an agent use this?** For each significant capability and event: is it exposed through a governed, documented, machine-usable interface, with access control and audit? Today that question stress-tests your integration quality. Within a few years, it will simply be the requirements list.

Risks and Failure Modes

The spaghetti event horizon. Point-to-point accumulation reaches the state where nobody can change anything safely: every modification breaks unknown dependents, so change slows toward zero, the integration layer as the enterprise's binding constraint. The escape is the strangler discipline applied to wiring: catalogue, then progressively re-front the worst tangles with shared assets, highest-traffic seams first.

The silent failure. An integration breaks and nothing announces it: orders quietly stop flowing, records quietly stop syncing, and the discovery arrives via a customer complaint or an auditor's question, weeks later, with a reconciliation bill attached. Monitoring, alerting, and reconciliation jobs are not operational polish; they are the difference between an incident and an era.

The orphaned flow. The integration outlives its builders: undocumented, unowned, running on a forgotten server with embedded credentials, load-bearing and unknown. Every enterprise has these; the

catalogue-and-ownership discipline is the only cure, and the security review of exactly these flows is among the highest-yield audits available.

The middleware megaproject. The integration renewal framed as a multi-year, big-bang platform migration: enormous spend, delivery frozen meanwhile, and, per every big-bang lesson in this series, needs moved before the platform lands. The alternative is incremental: platform adopted, then flows migrated by traffic and pain, value banked continuously.

Governance theatre, or governance absence. The twin failures: standards documents nobody follows and gates everybody routes around, or no rules at all until the breach. Both stem from divorcing governance from delivery; the working pattern embeds it, secure defaults in the platform, review built into the paved path, monitoring always-on, so the compliant way is the easy way.

The partner seam surprise. B2B integrations carry unique risk concentrations: another organisation's security posture, contractual data obligations, and change cycles you don't control. Treating partner connections with internal-grade informality, shared credentials, unversioned formats, no isolation, invites both breaches and breakages. Partner seams deserve the most formal contracts, technical and legal, in the estate.

Integration debt compounding into AI failure. The forward-looking risk the sources make explicit: organisations attempting agent-class AI atop ungoverned, undocumented, point-to-point integration will find agents amplifying the chaos, acting on stale data, invoking fragile flows, evading audit, at machine speed. The amplifier logic of this entire series applies: AI multiplies the quality of the layer beneath it, and integration is that layer.

The capability that quietly decays. Even well-built integration erodes: interfaces drift, documentation staleness compounds, reuse rates fall as teams route around aging assets. Without the maturity scorecard and standing stewardship, the road network degrades the way all infrastructure does, invisibly, then suddenly. The never-finished principle governs here too.

Recommendations

1. Build the catalogue this quarter. Inventory every integration you can find, including the scripts, the exports, and the human workarounds: what connects to what, carrying which data, owned by whom, monitored how. Expect surprises. This single artefact converts the invisible layer into a manageable portfolio and grounds every other decision.

2. Stand up the maturity scorecard. Reuse rate, time-to-integrate, reliability, adaptability, baselined now, reported quarterly to leadership. What the organisation currently repurchases invisibly, it will start compounding visibly.

3. Fund the shared layer as infrastructure. A central budget for the integration platform, the API and event standards, and the ICoE-style enablement function, explicitly separated from project budgets, so roads get built even though every individual project only needs a wire.

4. Front the core with APIs; publish the key events. Prioritise stable, governed interfaces over the systems most processes touch, and stand up the event backbone for the business facts most systems care about. Sequence by process pain and AI ambition, the same moves serve both.

5. Charter the stewardship, guardrails-first. An ICoE with the explicit anti-charter our platform report prescribed: not a delivery bottleneck, not a standards police, an enabling function measured by other teams' speed and the reuse rate, with governance embedded in platform defaults rather than approval queues.

6. Engineer reliability into every seam. Monitoring and alerting as a condition of go-live; idempotent retries, timeouts, and dead-letter handling as standard patterns; reconciliation jobs across the flows where money or compliance ride. No integration ships silent.

7. Run the security sweep on the connective tissue. Credentials out of code and into vaults; least-privilege scopes reviewed on a cadence; partner seams isolated, contracted, and audited; decommissioned flows verifiably dead. The orphaned integration with live credentials is the finding to hunt first.

8. Ask the agent question of every new build. "Could a governed agent use this?", machine-readable documentation, clean authentication, scoped access, audit trails. It costs little at design time, dates nothing, and quietly assembles the foundation the next wave of ambition will stand on.

How SDTC Digital Thinks About This

SDTC Digital builds products and platforms whose value almost always depends on connections we don't control: our clients' systems, their partners' systems, and the accumulated wiring of decades. That vantage has made us unsentimental about integration, and oddly fond of it.

Unsentimental, because we have seen where the bodies are buried: the load-bearing script nobody knew about, the partner feed with shared credentials, the "quick connection" from 2017 that three business processes secretly depend on. Our practice therefore starts where this report starts, with the catalogue and the process map, because the honest inventory is the foundation of every good decision that follows, and because the most valuable early deliverable in most integration engagements is simply *visibility*.

Fond, because integration is where engineering discipline converts most directly into business motion. A clean API over a stubborn legacy core; an event stream that lets six systems, and next year, six agents, react to the business in real time; a partner interface that turns onboarding from weeks to hours: these are unglamorous artefacts with immediate, measurable business consequences. We build them as products, owned, documented, versioned, monitored, because the alternative, plumbing built project by project, is how every spaghetti estate we have ever untangled came to exist.

And we hold the forward view the evidence demands: the integration layer is becoming the action layer of the AI era. When we design connections now, we design them for the traffic that is coming, governed, auditable, machine-usable, because the companies whose systems can be safely acted upon will be the

companies whose AI amounts to more than demonstrations. The connective tissue is the strategy. We build it that way.

Conclusion

Enterprise integration has spent forty years as the least glamorous essential discipline in technology: the roads of the city, noticed only in their absence, funded only in fragments, owned by no one. The definition that opened the field, right information, right place, right time, has never needed updating. Only the traffic has changed.

And it is changing again, decisively. Automation put processes on the roads; analytics put decisions on them; and now agentic AI proposes to put *action* on them, software that observes, decides, and does, entirely through the connections an enterprise has built and governed, or failed to. The evidence of this report reduces to one strategic sentence: the quality of your integration layer is the ceiling on the quality of everything you intend to build above it.

The path is the same one every report in this series has found at every layer, because it is the same discipline wearing different clothes: make the invisible visible, own what you build, prefer roads to wires, govern with guardrails, measure the capability, and never declare it finished.

Your systems are already talking, somehow. The only question is whether by design.

Build the roads. The future arrives on them.

Sources referenced in this report include SAP's enterprise integration overview (definitions, integration types and technologies, best practices, the integration centre of excellence model, maturity indicators, customer examples, and the analysis of integration as the architectural foundation for agentic AI); IBM's enterprise integration analysis (integration approaches, middleware patterns, and the MNG Kargo and Helsinki Regional Transport case studies); Supermicro's taxonomy of integration types, challenges, and best practices; and Wikipedia's historical and academic grounding (including the Brosey et al. definition, the computer-integrated manufacturing lineage, Vernadat's enterprise modelling work, and the Enterprise Integration Act of 2002). Two scraped sources in the research dossier concerned an IT services company named "Enterprise Integration" rather than the discipline, and were excluded as off-topic. Vendor case-study outcomes are as published by the vendors; verify before publication.