

Sourcing AI capability: buy, build or assemble

Article 5 of 6 · aicfopartner.ai · PDF issued 9 September 2026

EXECUTIVE SUMMARY

- Sourcing AI capability comes down to four options: review what is owned, buy a product, assemble around the company's data and systems, or write and maintain custom software.
- The starting point is to establish how much of that AI capability is already available through the ERP, the Microsoft or Google estate and the wider technology stack.
- Buying AI capability as a product is strongest for transaction work at scale, where the volumes justify the cost.
- Assembling means configuring a vendor maintained workflow or agent platform, such as n8n, Workato or Copilot Studio, and connecting it to the company's data and systems to automate specific processes. It suits requirements that are particular to the organisation, or where assembly costs less than an off the shelf product.
- Writing and maintaining custom software should be rare. It is justified only where the team can support that software for years and the capability genuinely differentiates the business.
- A business case is needed whatever the sourcing route. It nets time, working capital and quality gains against the full running cost, while also weighing time to value and reversibility.
- For purchased AI capability, fit must be proven before buying. Trial the product on last month's real data, including the awkward cases, with error types and pass thresholds agreed first.

01 — THE FOUR SOURCING OPTIONS

The fluency article covered the introduction of AI within the finance team, working with an AI assistant on routine tasks, the first hours recovered and the best techniques [captured as Skills the whole team can run](#). Beyond this stage, workflows evolve to the point where they require proper orchestration, rather than a saved Skill. The question is how the next stage of AI capability should be developed. The answer is usually one of four choices, each outlined as follows:

- Review current functionality: the capability may already sit inside applications the business pays for, including the ERP
- Buy: an established product covering the capability

- Assemble: an AI tool such as Claude, connected to the company's data inside a vendor maintained platform, configured by finance
- Build: custom software the team is fully responsible for

These options can overlap. For example a purchased cash management platform will often include some AR and AP functionality, such as collections chasing and supplier payment scheduling. Conversely, a single process can draw on more than one option, so matching in a reconciliation may stay with the ERP while the debtor chasing becomes part of an assembled solution.

The first area to explore is the capability the organisation already pays for or has access to, which often reaches beyond the ERP. Microsoft or Google subscriptions, for example, can carry workflow and automation tools, and many finance systems hold licensed modules not currently in use. In a small business, readily available options are usually quicker to check because the application estate is smaller. For larger organisations, what is already available varies by instance, subscription tier and territory and is worth a detailed discussion with current providers. In addition, if a process is already outsourced, payroll or AP processing for example, the provider's scope belongs in the same review. As an initial step from the ERP perspective, it's worth a call with the support team to understand the AI and automation options available to the team.

If the ERP falls short on capability, that in itself is a finding. Sometimes the product is weak. More often the cause is a poor implementation, an ageing instance or a subscription tier never upgraded. An internal review of the suitability of a legacy ERP is generally recommended at this stage, given the finance function is built around it. Post assessment options range across: upgrading what is already there, keeping the ERP as the system of record and assembling AI capability around it or moving to a modern cloud ERP. The latter is usually the conventional answer, certainly at scale. However, for a currently narrow profile of company, adopting an AI native ERP built around the ledger can be a compelling proposition. That profile of organisation is typically a high growth software business with a simple structure yet complex revenue recognition and forecasting. The development of AI native ERPs will be interesting to watch over the coming years, and incumbents will no doubt adapt. Even today, for many businesses implementation is alongside an existing ERP system. For buyers outside the US in particular, weighing the AI native ERP route certainly includes exploration of the following topics: where the data sits, whether the auditor will rely on the system's controls and traceability, and whether it supports UK GAAP or IFRS rather than US GAAP alone. The ERP assessment matrix on the Resources page is [a useful initial reference point for organisations making decisions relating to their ERP.](#)

Where material capability gaps remain after an assessment of current infrastructure, reviewing the options that exist through pre built product offerings is a sensible next step. Such products are usually quick to deploy and specialised, have improvements driven by a broad client base and come with vendor support and maintenance. The main drawback is of course cost, so buying tends to be strongest for transaction work at scale: invoice capture, matching, collections, cash application, and reconciliations for example. Products priced for enterprises, with platform fees, volume commitments and minimum contracts, are naturally hard to justify at lower transaction volumes.

Where existing infrastructure has gaps and purchasing an off the shelf solution isn't viable the next option to consider is assembling the required capability. In simple terms this means using an AI tool such as Claude or Copilot, connected to the company's own data and systems, with finance setting up and owning the configuration. This is the option that solves for the elements of process and systems that are unique to a particular organisation. These range from approval chains, risk tolerances and operational systems sitting outside the ERP, such as a media booking or practice management system. A connection between these systems and the ledger usually exists, but it is

rarely complete, resulting in rekeying and checking around the edges and where no connection exists the gap is filled by hand. That joining work is what assembly covers well. Conventional approaches said buy the standardised workflows and build the bespoke. However, building used to mean custom software, so many organisations did not, and the bespoke layer often stayed manual. The logic here has not changed, but AI has made this kind of automation and assembly far easier to implement, so the bespoke path has a more practical outlet.

Assembly comes in two workable patterns. In the first, a workflow tool such as n8n runs the process and the LLM is called only at the judgement points. This tends to win where the steps are known, volumes are high and the auditor needs a reliable evidence trail. It also tends to be cheaper to run, since the deterministic steps avoid model charges and LLM calls happen only where judgement is needed. The workflow platform's own subscription is often the main direct running cost. In the second case, an AI agent given a defined goal runs the workflow itself, governed through scoped permissions, activity logs, a spend ceiling and human review at any point that posts to a live system. This tends to win where inputs are messy, the path cannot be fully written down in advance and volumes are lower. It is faster to stand up but requires more development expertise and bandwidth. It's also generally more expensive to run as an agent tends to make more model calls, with less predictability, particularly when planning its path and handling exceptions. At scale these costs compound quickly. The market is loudest about agents, yet much finance work follows fixed rules, so the workflow pattern wins more often than the noise suggests. Even so, agentic capability is improving rapidly, token prices are falling, and agents offer flexibility, speed and seamless integration into many existing applications. Agentic workflows nevertheless look like the direction of travel.

02 — WORKFLOW LED AND AGENT LED ASSEMBLY

DIMENSION	WORKFLOW LED	AGENT LED
Cost to run	Lower and more predictable, model charges only at the judgement points, platform charges remain.	More model calls, less predictable, compounding at volume.
Speed to stand up	Slower, the steps must be mapped first.	Faster to trial, though testing and guardrails absorb much of the saving.
Audit story	Consistent, traceable evidence.	Activity logs, with human approval before anything posts to a live system.
Where it breaks	Messy inputs and unmapped paths, and a flow can break when a connected system changes.	High volumes and strict repeatability, and needs tight permissions and monitoring to keep actions within intent.

The fourth option is a true build, where the team is responsible for the software itself and therefore for everything that goes wrong with it. The distinction from assembly is where that responsibility lies. Assembling means configuring platforms the vendors maintain, with the business responsible for what it configures. The same can apply to agents, since on the current platforms one is often set up through instructions, connections and permissions rather than code. The moment the team is writing and maintaining its own software, whether a full application or a hand coded agent, the work has become a build. However, build is seldom the answer as it's genuinely difficult. The test of a build is whether the team can maintain it in a sustainable way, manage security and improve it at scale. The

test of a vendor is whether the product can flex to the organisation's needs and manage the security. A bought product improves on the feedback of its whole client base. A build improves only on the team's own experience. The honest exception is capability that genuinely differentiates the business. Building it takes people committed to the software for its whole life, not a project team that moves on, and returns that cover the running and maintenance cost, not just the cost of creating it. This boundary is easy to cross without noticing. A quick script produced with an LLM that ends up running a recurring finance task is, in effect, a small build: someone now owns its upkeep, usually whoever wrote it. These scripts cost next to nothing to produce and are fine for internal convenience, a one off analysis or a file tidy. They sit uneasily though if used in areas where an auditor needs a complete record, or where output feeds into reported numbers or the ledger. Such use cases need proper engineering ownership and tested controls: either the script is hardened into a proper build, or the work moves to a bought, assembled or native solution. [The governance article covers the control side.](#)

Whichever option is chosen, the capability can be implemented and run in house, implemented by a partner and handed over, or operated as a managed service. The expensive failures come from choosing the wrong option: paying enterprise subscription prices for work a simple assembled workflow could have handled, or committing the team to build and maintain something a proven product already does well.

03 — THE FOUR OPTIONS COMPARED

OPTION	PROS	CONS
Review current functionality	Often already paid for, quickest to check, no new supplier.	Coverage varies by instance, tier and territory, and capability can be thin or expensive.
Buy	Quick to value, specialist product, improved by client base feedback, well supported.	Cost often fails to justify the economics at lower volumes.
Assemble	Fits the bespoke layer, can be cheaper than a specialist product at suitable volumes, finance owns the configuration.	Run cost and maintenance continue after go live and need a named owner.
Build	Exactly what the business needs, genuine best in class differentiation.	Cost. The team owns everything that breaks, for years not months.

04 — THE BUSINESS CASE

Whichever option is chosen, a business case follows, typically based on annual cost comparisons. For a purchased product, the annual cost is generally a mix of fixed subscription fees and variable charges, with enterprise pricing often negotiated by tier. For an assembled product the main costs are the platform and workflow subscriptions, the seat fees, the LLM token charges and any hosting. Token charges vary sharply by model, so model selection at each step is a cost decision as well as a quality one. Some open weight models, including several from Chinese labs, now deliver strong performance on many tasks at materially lower token prices than leading commercial models. Within a well governed portfolio of commercial and open weight models, an enterprise can route each step to the lightest

approved model that meets the required standard, while reserving the strongest models for the judgement points. Implementation sits on top of either. Savings typically come from three areas: time, working capital and quality. Time is hours of manual work removed: keying invoices, chasing payment, reconciling accounts, preparing reports, etc. In a growing business, future headcount avoided also has to be estimated and weighed into these considerations. Working capital benefits manifest themselves in cash collected sooner, early payment discounts captured and fewer duplicate or late payments. Quality is fewer errors reaching the ledger, better control and faster decisions. A purchase that saves nothing, avoids nothing and improves nothing clearly has no case, and every case should state which spreadsheets and workarounds will be switched off, and when.

Where the realistic choice is between assembling and buying, two further factors often determine it: ownership and cost certainty. An assembled capability needs two named people: an owner who decides what it does, and a maintainer who fixes it when it breaks. Their time is part of the cost consideration. In a smaller team, if nobody can honestly take that on, a managed service or a thinner native feature usually beats a superior assembled product. Certainty also differs between the routes. An assembled or built capability is an estimated cost until it is working, and internal estimates tend to run optimistic. The cost of a bought product is known with more certainty, though not completely: usage charges can be layered onto subscriptions, and renewals can bring increases and repricing. Both exposures are worth sensitivity testing in any multi year comparison.

Nevertheless cost, while critical, does not always decide the choice of approach, and two considerations can take primacy. Time to value is one: an adequate native feature available today often beats a cheaper assembly six months away. Reversibility is the other: how easily a tool can be withdrawn. For a bought product that is partly the minimum term and the notice period, but mostly it is operational, since a dependency embedded in the close or a payment run is disruptive to remove whatever the contract says. Where both routes are open, the reversible one is usually the better place to start.

05 — BUYING WELL

When the decision is to buy, the market offers plenty of credible products and the shortlist can look interchangeable. What separates them is functionality and fit. Functionality, what a product does and how intuitive it is to use, usually shows itself in demonstrations. Fit is harder: it has to be proven on this organisation's systems, data and volumes. The starting point here is the connection: the product must connect to an organisation's ERP instance, version and tier actually in use, not the flagship configuration the vendor shows. Proving fit starts with a trial. Before any data leaves the business, the trial environment is approved. It then runs on last month's real data and should include the awkward cases: rare transactions, high values and month end exceptions, with the material error types and pass threshold agreed before it starts. The demonstration is also not the process. The written procedure rarely captures the small corrections people make without noticing, and a tool configured to the documented steps meets the undocumented ones in its first week. The trial therefore has to sit through the process as it actually runs, not as it is described. Quoted match rates may be achieved on clean data, so it is also worth asking upfront what level of data hygiene those figures assume to avoid trials that were never going to succeed. The preparation behind these checks is [covered in the implementation roadmap](#).

The contract terms deserve the same scrutiny as the product. The pricing should still make sense as volumes grow. Exit terms need to be explicit, with the data, configurations and logs secured for the business when the agreement ends. Support means knowing exactly who owns an incident when something fails, and how product changes are notified and tested before they reach production. Claims should be tested through reference customers who can be called and assurance evidence such as SOC reports and ISO certificates, checked for scope and currency rather than

taken on trust. Model flexibility deserves a question too: whether the product allows a choice of underlying LLM, and how model changes are priced, notified and tested before they reach production. Regulated firms face stricter duties on material third party arrangements, [covered in the governance article](#).

06 — A TYPICAL CAPABILITY PATTERN

Applied to core finance processes (AR, AP, FP&A, cash management, close), the approach taken to develop AI capability tends to settle into a recognisable pattern, particularly at the mid market:

- For the close, the ERP itself is generally the answer, since that is where the audit trail lives and where new capability is increasingly shipped
- Cash is usually the first purchase worth serious consideration. A cash management platform gives the business current and forward visibility of cash, and the stronger products include useful AR and AP functionality. At the mid market, the pricing can also be proportionate. A cash platform needs little data preparation as well, working from the bank feeds and the general ledger
- FP&A is normally assembled rather than bought, because the commentary, the board pack and the forecast assumptions are particular to individual businesses. The strength of that assembly rests on the data beneath it: a governed data platform, a warehouse or lakehouse, is where the function's data is usually best consolidated as it scales. This groundwork is [covered in the implementation roadmap](#)
- AR can often be handled through the ERP, the cash management platform or both, with assembly an option for individual steps such as invoice chasing. A dedicated collections engine tends to earn its place where volumes, messy remittances or cash application complexity outgrow that, with the DSO trend one of the clearest signals
- AP often follows AR where cash collection is the more immediate working capital constraint, though AP automation can also improve the position through better timed payments. The ERP's own capture and matching, with light assembly around any gaps, is often adequate until invoice volume, approval complexity or control pain justifies a dedicated product

Solutions are also available beyond the core processes and can include revenue recognition, treasury and statutory e invoicing. These tend to be adopted in response to a specific trigger, a new revenue model, a financing event or a country mandate, rather than as part of the efficiency case above. All of this is a starting point rather than a rule, and [the live shortlists on the Resources page track how quickly it moves](#).

[How the shortlists are curated is set out alongside them](#). Working through these decisions alongside CFOs, from the first review of current functionality to the final shortlist, is what AI CFO Partner does. If the next purchase is on the horizon, [book a call](#).

THE AI IN FINANCE SERIES

- 01 AI in finance: what it is and how it works aicfopartner.ai/ai-in-finance
- 02 AI strategy for finance: the six choices aicfopartner.ai/ai-strategy-for-finance
- 03 The roadmap to the AI enabled finance function aicfopartner.ai/ai-finance-roadmap
- 04 Developing an AI enabled finance team aicfopartner.ai/ai-finance-team-development
- 05 Sourcing AI capability: buy, build or assemble aicfopartner.ai/ai-buy-build-or-assemble
- 06 Governance that speeds AI adoption aicfopartner.ai/ai-governance-for-finance

Questions or comments: tim@aicfopartner.ai