JUNE 26, 2026

Building on Copilot: the automations and tools that create the value

Copilot is the floor. The return comes from the automations and agents you build on top — and from measuring the time saved honestly.

By **Dave Taylor**

You build Copilot automations on top of the licence: grounded knowledge agents in Copilot Studio, trigger-based flows in Power Automate, and custom integrations through the Graph API. Copilot itself is the floor. The return comes from those build-on-top tools, and from measuring the time they save honestly. Microsoft's 2023 Work Trend Index found early Copilot users were 70% more productive — but that headline only becomes a number on your P&L once you build the automations that turn scattered minutes into hours.

This closes the three-part series. Part 1 set out why Copilot is the floor, not the ceiling, and Part 2 explained why Microsoft Graph is the data layer it stands on. With the floor and the data layer in place, this part is about the storey you actually live in: what you build on top,

with which tools, how to choose between them, and how to prove the whole thing was worth the spend.

What do you build on top of Copilot, and with what?

Start from the build layer, because that's where the time savings live. Off-the-shelf Copilot answers questions inside the apps your team already uses. The value comes one rung up — when you wire a repeatable task into something that runs without a person babysitting it. In a professional-services firm that's the accounts-prep first draft, the working-paper summary, the standard client letter, the tax-query lookup, the file-review note. Each is a place where AI removes twenty or thirty minutes of someone's day, several times a week. The licence makes those savings possible. The build is what captures them.

There are four tools that earn a place here, and one rule for choosing between them. Microsoft's Copilot Studio builds grounded agents — an "ask-the-manual" assistant that answers from a curated set of golden-source documents, your tax manuals, scheme rules, or precedent letters, and nothing else. It's the right shape when the need is repeatable Q&A over vetted internal knowledge. Power Automate runs trigger-based flows across Microsoft 365: a file lands, it gets summarised, routed, and the right person is notified — the right shape for connecting Microsoft apps and approvals. The Graph API is for custom automation that reads and writes tenant data beyond what no-code flows reach. And custom connectors or Azure wire a line-of-business system into the flow when the standard set won't stretch that far. Each does one job well; the rule for picking between them is the next section.

How do you choose the right tool for a workflow?

Start from the workflow, not the tool. The mistake we see most often is a firm deciding it wants "an AI platform" and then hunting for tasks to justify it. Reverse that. Map the real job a person does today, find the single point where AI removes twenty to thirty minutes, then pick the lightest tool that captures exactly that. A grounded Q&A need is a Copilot Studio agent. A "when this happens, do that" need is a Power Automate flow. A read-write-into-our-data need is the Graph API. Resist building a platform when a flow will do — the platform costs more, takes longer, and usually solves a problem you don't have yet.

Two things make the choice harder, and both have the same answer: don't be dogmatic. The first is the pull to standardise on Power Automate for everything because it's already in the tenant. Some orchestration logic is genuinely painful to express there, and a non-Microsoft tool like n8n or Make is the smaller tool for that job. The second is the workflow that spans systems outside Microsoft 365 entirely — a practice-management package, a

billing platform, a bank feed. Those earn a tool that crosses boundaries cleanly. The principle holds across all of it: pick the smallest tool that does the job, and let the workflow, not the brochure, decide which one that is. This matters because every tool you add is a tool you later have to govern — which is exactly the map Part 3 of our security series tells you to keep.

Why does governed automation beat ad-hoc AI tools?

The alternative to building this deliberately is letting it happen by accident. It already is. Microsoft's 2024 Work Trend Index found 78% of employees bring their own AI tools to work, rising to 80% at small and mid-sized firms — where there's rarely an IT gatekeeper to slow it down. That's your team already automating their own work, just with personal accounts, no audit trail, and your client data passing through tools you've never seen and can't reach if a client asks where their file went. A governed Power Automate flow or a Copilot Studio agent does the same job inside your tenant, under your permissions, on data that never leaves a boundary you control, where you can see exactly what it touches and revoke it in a click. The shadow version is cheaper to start and far more expensive to live with.

Built-in automation also compounds in a way ad-hoc tools never do. When a Power Automate flow proves out the accounts-prep first draft for one client, the same flow serves every client of that type — the work was done once and the saving repeats across the volume, which is what makes the maths work later. A personal ChatGPT tab saves one person twenty minutes once and leaves nothing behind: no reusable flow, no record of what worked, nothing the next person inherits. The argument for building inside the platform isn't only about control, though control matters and the regulators increasingly require it. It's that a governed automation is an asset the firm owns and can hand to the whole team, while a shadow tool is a private habit you can't see, audit, or scale. The build layer is where you turn those private shortcuts into firm-wide capacity, on the tools in our working stack rather than a shelf of unused licences.

How do you measure the ROI honestly?

Here's the frame we use, and it's the part most vendors skip. The return on a Copilot build isn't two hours of magic on one heroic task. It's death by a thousand cuts, run in reverse: twenty or thirty minutes saved across four or five touchpoints inside a single job, repeated across every job of that type you run in a year. The maths only works at the level of the whole job, not the single demo. So the question to put to a finance leader isn't "how clever is the agent?" It's "across all the jobs we run, how many minutes per job do I have to save to pay for all of this?"

That question has an arithmetic answer, and it's deliberately simple: programme cost, divided by the number of jobs per year, divided by the loaded hourly rate, gives you the minutes per job you need to break even. When the answer comes back at roughly twenty to thirty minutes against a job that already runs to several hours, the bar is low — and a finance leader can see it's low without taking anything on faith. ROI here means return on investment, the cost set against the time recovered. The honest caveat is that every input to that sum starts as a placeholder: the job count, the rate, the minutes saved. Replacing those assumptions with measured baselines is the first month of the work, not a footnote you wave at. Measure the real numbers, and the break-even line moves to where it actually sits.

What does the time saved buy a business that can't hire?

For a chief executive, the number that lands isn't a percentage — it's capacity. Add up the minutes saved across the full volume of jobs in a year and they convert into full-time-equivalents, FTEs, of work absorbed without a new hire. Twenty minutes a job across a few thousand jobs is not a rounding error; it's a person's worth of time, recovered and put back into work the firm couldn't otherwise reach. In a market where the right people are hard to find and harder to keep, that's the argument that matters most to a CEO: the same team handles more work, or the same work in less time, and you haven't had to win a recruitment race to do it. The build layer doesn't replace people, and it isn't a headcount-cut argument dressed up as efficiency. It gives the people you already have their afternoons back, and gives the firm room to grow into work it would otherwise have turned away.

This is where the honesty rule pays off twice. A capacity claim built on placeholder assumptions falls apart the first time someone tests it against reality, and it should. A capacity claim built on measured baselines survives the test, which is why the measurement month isn't overhead — it's the thing that makes the rest of the argument defensible to a board. Microsoft's 2023 Work Trend Index put early Copilot users at 29% faster on the tasks they used it for. That's the ceiling the tool can reach on a task. Whether your firm turns "faster on a task" into "an FTE of capacity we didn't have to hire" depends entirely on which automations you build, and on whether you measured the real saving or guessed at it.

What it adds up to

The series comes down to three storeys. Copilot is the floor — the licence is the cheap, easy part. Microsoft Graph is the data layer it stands on — answers are only as good as what the model can reach. And the build layer is where the value actually lives: the Copilot Studio agents, the Power Automate flows, and the custom integrations that take scattered

twenty-minute savings and compound them into real hours, then into capacity. The single takeaway is that the tool was never the hard part. Buying Copilot takes an afternoon and a purchase order; the return takes a deliberate build and an honest measurement, and most firms stop after the purchase order. Choosing the right small automation for a real workflow, and measuring what it saves against an honest baseline rather than a hopeful one — that's the work, and it's the work that turns a licence into a return.

If you've bought Copilot and want to know which automations would actually pay for themselves, book a 30-minute working session and we'll map your highest-value workflow and the lightest tool to capture it — together, against your real numbers.