CLOUDBOLT

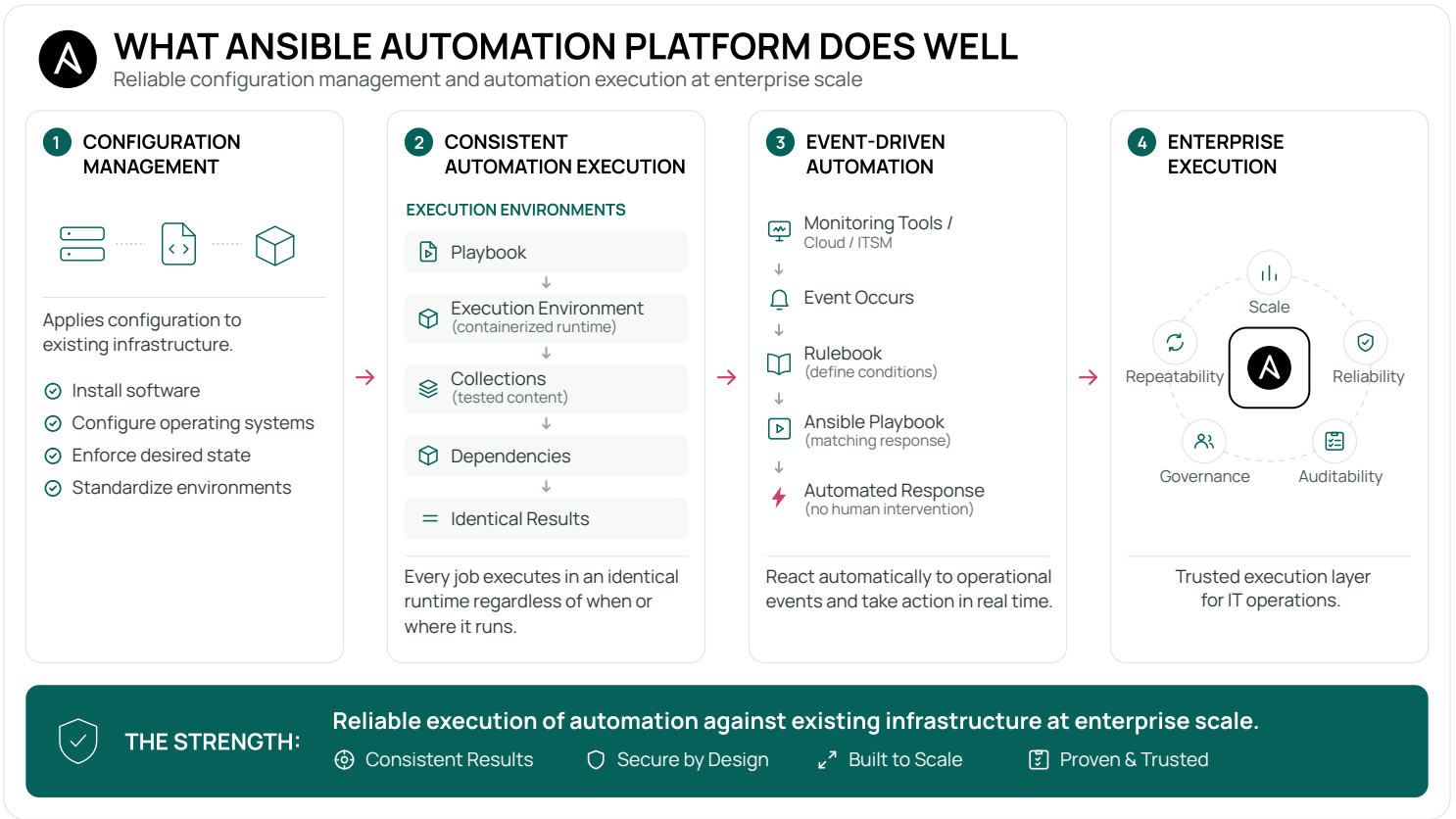
COMPLEMENTARY STACK SERIES: PART 2

Ansible Automation Platform and CMP: Where the Execution Layer Ends

The pitch usually goes something like this: a Red Hat account team mentions that Ansible Automation Platform now has a self-service portal with its own catalog, so why keep paying for a separate cloud management platform. It is a reasonable question, and it deserves a real answer instead of a defensive one. Ansible Automation Platform's newest self-service features are genuinely useful. They are also scoped in a specific, documented way that makes the platform an excellent complement to a cloud management platform, not a substitute for one. Knowing exactly where that scope ends is the difference between a defensible budget conversation and a guess.

WHAT ANSIBLE AUTOMATION PLATFORM DOES REALLY WELL

Ansible built its reputation on configuration management: applying settings, installing software, and enforcing state on systems that already exist, using a large ecosystem of tested playbooks and collections. Ansible Automation Platform packages that into an enterprise-grade execution engine. Automation execution environments are containerized runtimes, built on a Red Hat Enterprise Linux base image, that bundle the automation engine with the exact collections and dependencies a playbook needs, so a job runs the same way whether it is triggered today or eighteen months from now. Event-Driven Ansible adds a reactive layer on top: a rulebook monitors events from monitoring tools, cloud providers, or ITSM systems and hands the matching response to Ansible Automation Platform's existing execution tooling automatically, without waiting on a person to notice. Red Hat's own recent positioning calls this combination the trusted execution layer for IT operations, and for the specific job of running automation reliably at scale, that description holds up.



WHAT THE SELF-SERVICE PORTAL ADDS…AND WHAT IT DOESN'T

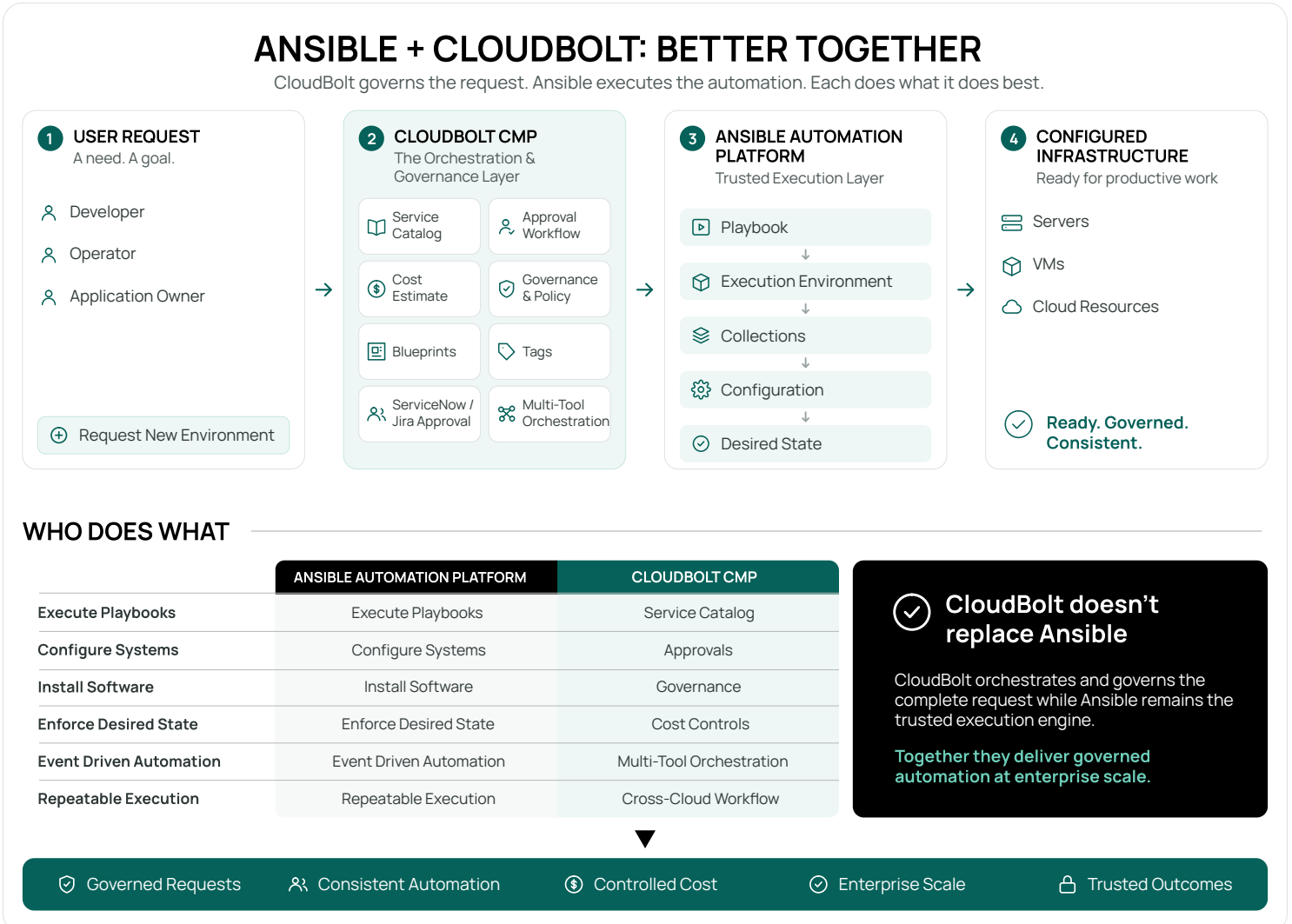
The self-service automation portal gives less technical users a simpler front end: a centralized content catalog that synchronizes existing Ansible job templates so someone can request a piece of automation without opening a playbook. It is a real improvement for teams that only need to run Ansible-based tasks. It is also, by design, scoped to exactly that. Only job templates accessible by the portal's configured token get synchronized into its catalog, and Ansible Automation Platform's own role-based access control remains the source of truth for who can see and run what. A user who cannot log into Ansible Automation Platform cannot log into the self-service portal either, because the portal depends on that same underlying authentication to work. None of that is a flaw. It is a configuration management tool's self-service layer, built to make Ansible easier to consume, not a general-purpose service catalog built to govern requests that span multiple tools and clouds.

WHERE CMP PICKS UP

A cloud management platform's catalog starts from a different premise: the request a person submits is rarely just one thing from one tool. Provisioning a new environment might mean creating cloud infrastructure, configuring it with Ansible, registering it in ServiceNow, and applying a budget check, all before anything is handed back to the requester. CloudBolt's catalog is built to be that unified layer: cross-cloud orchestration, automated approval chains that can span ServiceNow, Jira, and 200-plus other systems, and cost estimates enforced before a resource ever deploys, regardless of which tool underneath does the actual work. Unlike a native catalog that only works inside a single ecosystem, CMP's catalog is designed to sit above all of them at once. Ansible remains one of the execution engines CMP calls, not a competing catalog it has to choose between.

THE PRACTICAL PAIRING

In practice, the two sit together cleanly. A CMP blueprint provisions the underlying infrastructure, triggers an Ansible Automation Platform job template to configure it exactly the way the organization's playbooks already do, and then wraps the whole sequence in the approval, tagging, and cost logic that job template was never built to provide on its own. Ansible still runs the playbook. CloudBolt still governs the request. Nothing about adding a governance layer requires touching the automation content a team has already invested years in building.



WHO DOES WHAT

	ANSIBLE AUTOMATION PLATFORM	CLOUDBOLT CMP
Execute Playbooks	Execute Playbooks	Service Catalog
Configure Systems	Configure Systems	Approvals
Install Software	Install Software	Governance
Enforce Desired State	Enforce Desired State	Cost Controls
Event Driven Automation	Event Driven Automation	Multi-Tool Orchestration
Repeatable Execution	Repeatable Execution	Cross-Cloud Workflow

CloudBolt doesn't replace Ansible

CloudBolt orchestrates and governs the complete request while Ansible remains the trusted execution engine.

Together they deliver governed automation at enterprise scale.

Governed Requests · Consistent Automation · Controlled Cost · Enterprise Scale · Trusted Outcomes

THE BOTTOM LINE

Ansible Automation Platform's self-service portal solves a real problem: making an excellent execution engine easier for more people to use. It does not solve a different problem: governing, costing, and approving requests that span more than Ansible itself.

A team that already trusts its Ansible content does not have to choose between keeping that investment and gaining that governance. The next chapter in this series looks at the same question from Terraform's side, where the shape of the overlap is different, but the underlying answer looks very similar.

Talk to CloudBolt cloudbolt.io/demo