

CHAPTER 17

Cloud Operating Models and Platform Engineering

The Engine of Enterprise Agility

“Technology is the accelerator, but the operating model is the engine. A world-class platform does not merely host applications; it empowers people.”

— Sazzad Khan

EXECUTIVE SUMMARY

For Nordic Ventures Group, moving workloads to Azure while retaining the behaviours of twelve distributed datacentres would reproduce the past in a different location. Ticket queues, centralised approvals, fragmented ownership and manual provisioning would continue to constrain delivery. The cloud would be present, but cloud agility would remain absent.

This chapter defines a modern Cloud Operating Model built around shared responsibility, platform engineering and product thinking. NVG creates an internal platform that gives teams in fourteen countries secure, repeatable and observable ways to build and operate services. The platform provides paved roads rather than rigid gates: subscription vending, identity patterns, network connectivity, policy guardrails, logging, secrets, deployment pipelines, and approved workload archetypes.

The objective is not central control for its own sake. It is to reduce cognitive load, remove repetitive infrastructure work, improve developer experience, and allow product, data, AI and factory engineering teams to focus on business outcomes. Platform engineering becomes the mechanism through which governance and speed reinforce one another.

CHAPTER OBJECTIVES

- Define a Cloud Operating Model that aligns business, platform and workload responsibilities.
- Explain platform engineering as an enterprise capability rather than a tooling initiative.
- Design an internal developer platform around self-service, golden paths and reusable products.
- Reduce cognitive load while preserving security, resilience and financial accountability.
- Measure platform adoption, developer experience and business value.

WHY TRADITIONAL IT OPERATING MODELS FAIL IN THE CLOUD

Traditional infrastructure organisations were designed for scarce, physical capacity. Servers, networks and storage required centralised specialists, long procurement cycles and scheduled change windows. Control was exercised through queues and hand-offs because each change could affect shared equipment.

Cloud platforms change the economics and speed of provisioning, but they do not automatically change the organisation. When the old model is carried forward, teams experience cloud through service tickets. A developer requests a subscription, waits for networking, waits for identity, waits for monitoring and then waits for security approval. The technology can provision in minutes, while the organisation still delivers in weeks.

Traditional Pattern	Enterprise Consequence	Modern Response
Central ticket queue	Platform team becomes a bottleneck	Self-service within approved guardrails
Project funding	Capabilities decay after project closure	Platform treated as a long-lived product

Traditional Pattern	Enterprise Consequence	Modern Response
Manual configuration	Drift, inconsistency and audit difficulty	Infrastructure and policy as code
Siloed ownership	Slow incident resolution and unclear accountability	Explicit platform and workload responsibilities
One-size-fits-all standards	Teams bypass the platform	Golden paths with controlled extensibility
Activity metrics	Success measured by tickets closed	Outcome, adoption and flow metrics

THE NORDIC VENTURES GROUP CHALLENGE

NVG operates across fourteen European countries, supports approximately six thousand users and depends on five factories, SAP, Dynamics 365, Microsoft 365, industrial systems and a growing portfolio of data and AI initiatives. The historical estate was organised around twelve datacentres and regional teams with different processes, tools and operational practices.

The target operating model must preserve necessary local knowledge while establishing enterprise consistency. A fully centralised model would create delay and disconnect decisions from business context. A fully decentralised model would recreate fragmentation. NVG therefore adopts shared management: platform teams provide reusable enterprise capabilities, while workload teams own application-specific delivery and operations within defined boundaries.

THE CLOUD OPERATING MODEL

A Cloud Operating Model defines how people, processes, technology, governance and financial accountability work together to create and sustain cloud value. It

answers who owns the platform, who owns workloads, how decisions are made, how risk is accepted and how services improve over time.

Domain	Platform Responsibility	Workload Responsibility
Identity	Enterprise identity patterns, privileged access, baseline controls	Application roles, user entitlements, access reviews
Networking	Hub connectivity, DNS, firewall patterns, private connectivity	Workload segmentation, endpoint requirements, traffic intent
Governance	Management groups, policy baselines, subscription vending	Resource compliance, exception remediation, evidence
Security	Security tooling, posture management, incident integration	Threat modelling, secure configuration, vulnerability remediation
Operations	Shared monitoring, logging, backup patterns, service standards	SLOs, application telemetry, runbooks, on-call ownership
FinOps	Cost allocation model, budgets, reporting, optimisation standards	Forecasting, rightsizing, unit economics, product trade-offs
Delivery	Reusable modules, pipelines, platform interfaces	Application code, deployment frequency, release quality

Centralised, Decentralised and Shared Management Models

A centralised model can provide consistency but often becomes a bottleneck at enterprise scale. A

decentralised model increases autonomy but can duplicate tooling and weaken control. Shared management divides responsibilities intentionally. The platform team owns common capabilities and guardrails; workload teams own the services they build and the outcomes those services deliver.

For NVG, shared management is not a compromise between control and speed. It is a design that assigns each decision to the team best positioned to make it. Enterprise-wide risks remain centrally governed. Product-specific decisions remain close to the business.

PLATFORM ENGINEERING AS A STRATEGIC CAPABILITY

Platform engineering is the discipline of designing and operating internal platforms that reduce friction for the teams that build and run digital products. The platform encapsulates recurring architecture, security and operational practices into reusable capabilities. It turns expert knowledge into accessible products.

The goal is not to create another infrastructure team or a portal that hides complexity without changing it. A successful platform improves the entire path from idea to production. It provides interfaces, documentation, automation, support, telemetry, and feedback. Teams should be able to discover what is available, understand when to use it and receive a predictable outcome.

Platform as a Product

NVG treats the enterprise platform as a product with internal customers. It has a product owner, a roadmap, defined service levels, versioned capabilities, and measurable adoption. Platform priorities are determined by user needs and enterprise strategy rather than by the preferences of the platform team.

- A documented product vision connected to NVG business outcomes.
- Named personas such as application teams, data engineers, AI teams, and factory engineers.
- A transparent backlog and roadmap.
- Usage analytics and qualitative feedback.

- Lifecycle management for modules, templates and supported versions.
- Support, documentation and onboarding as product features.

Internal Developer Platform

The internal developer platform is the practical interface through which teams consume platform capabilities. It may combine a portal, source repositories, reusable infrastructure modules, CI/CD workflows, policy checks, service catalogues, and observability integrations. The interface is less important than the experience: a user should be able to obtain a secure, supported environment without understanding every implementation detail.

Platform Product	What the User Receives	Enterprise Control Embedded
Application landing zone	Subscription, network, identity, logging and budgets	Management group placement, policy, RBAC and tagging
Secure virtual machine pattern	Hardened image, backup, monitoring and patch integration	Approved image, Defender configuration and recovery controls
Container platform path	AKS baseline, registry, secrets, ingress and observability	Network isolation, policy, identity and supply-chain controls
AI experimentation environment	Workspace, data access pattern, quotas and model governance	Cost limits, approved data boundaries and responsible AI checks
Data product environment	Storage, Fabric or analytics pattern, catalogue integration	Classification, lineage, access governance and retention

Platform Product	What the User Receives	Enterprise Control Embedded
Factory edge pattern	Connected edge environment and cloud management	Segmentation, certificate management, monitoring and resilience

GOLDEN PATHS AND CONTROLLED EXTENSIBILITY

Golden paths are supported, opinionated routes for common delivery scenarios. They reduce decision fatigue and encode proven practices. They are not mandatory answers for every workload. Teams may leave the path when a justified requirement exists, but deviations become explicit architecture decisions rather than accidental variation.

A strong golden path provides speed, documentation, operational support, and a clear upgrade path. A weak golden path merely standardises technology while ignoring user experience. NVG therefore evaluates golden paths according to both control effectiveness and developer adoption.

REDUCING COGNITIVE LOAD

Every team has limited capacity to understand business logic, architecture, security, operations, and organisational process. When teams must master every infrastructure detail before delivering value, cognitive load becomes a strategic constraint. Platform engineering reduces this load by moving recurring complexity into well-designed products.

Team Topologies distinguishes intrinsic domain complexity from extraneous complexity created by tooling and organisational friction. NVG cannot remove the complexity of manufacturing, supply chains or European regulation. It can remove unnecessary complexity in environment provisioning, identity integration, monitoring, and deployment.

Cognitive Load Source	Platform Response
Selecting and configuring basic infrastructure	Curated workload archetypes and modules
Interpreting security standards repeatedly	Automated controls and early pipeline feedback
Discovering ownership and support channels	Service catalogue and named product teams
Integrating monitoring and logging	Default observability packages
Understanding cloud cost after deployment	Budgets, forecasts and unit-cost views from the beginning
Relearning patterns across countries	Consistent platform interfaces with localised guidance

DEVELOPER EXPERIENCE AND SELF-SERVICE

Developer experience is the quality of the environment in which teams discover, build, test, deploy, and operate software. It includes more than speed. A good experience provides clarity, confidence, fast feedback, and low friction. A platform can be technically sophisticated and still fail if users cannot understand or trust it.

Self-service does not mean unrestricted access. It means that approved actions can be performed without unnecessary human hand-offs. The control moves from manual approval to automated evidence, policy and traceability.

AUTOMATION AS THE DEFAULT

NVG implements platform capabilities through version-controlled code. Bicep, Terraform, GitHub Actions or Azure DevOps pipelines can be selected according to organisational standards and workload needs. The

principle is more important than a single tool: repeatable changes should be automated, reviewed and observable.

- Infrastructure as Code for landing zones and shared services.
- Policy as Code for preventive and detective controls.
- Configuration automation for operating standards.
- Pipeline security checks and artefact provenance.
- Automated testing for modules and platform releases.
- Automated evidence collection for assurance and audit.
- Automated decommissioning to prevent abandoned resources.

ORGANISING PLATFORM TEAMS

A single platform team cannot own every enterprise capability indefinitely. NVG establishes product-aligned teams around domains such as cloud onboarding, identity, connectivity, security, developer enablement, data platforms, and observability. Coordination occurs through shared principles, architecture forums, and inner-source practices.

Team Type	Primary Mission	Typical Outputs
Enterprise Platform Team	Provide landing zones and common Azure foundations	Subscription vending, management groups, policy, shared services
Identity Platform Team	Deliver trusted access and lifecycle capabilities	Authentication patterns, PIM, governance, federation
Connectivity Platform Team	Provide secure and resilient connectivity	Hub services, DNS, firewall, private connectivity

Team Type	Primary Mission	Typical Outputs
Developer Platform Team	Accelerate build and release workflows	Templates, pipelines, service catalogue, developer portal
Data and AI Platform Team	Enable governed analytics and AI	Data products, model access, catalogue, responsible AI controls
Enabling Team	Help workload teams adopt new practices	Coaching, migration support, architecture facilitation

CLOUD CENTRE OF EXCELLENCE: FROM CONTROL TOWER TO ENABLING NETWORK

The Cloud Centre of Excellence should not become a permanent approval board for every technical decision. Its role is to align strategy, operating principles, standards, skills, and transformation priorities. As maturity increases, execution moves into platform and workload teams while the CCoE focuses on enablement, measurement and cross-enterprise learning.

PLATFORM GOVERNANCE

Platform governance determines how products are approved, changed, measured, and retired. It includes architecture principles, security baselines, service ownership, release management, and exception handling. Governance must also protect users from sudden breaking changes. Versioning and deprecation are therefore essential product disciplines.

MEASURING PLATFORM SUCCESS

Platform success is not measured by the number of modules created. It is measured by improved flow, reliability, adoption, and business outcomes. Metrics

should combine operational evidence with user experience.

Metric	What It Reveals
Time to first compliant environment	How quickly a team can begin useful work
Platform adoption rate	Whether teams choose the provided products
Lead time for change	How quickly ideas reach production
Deployment frequency	Whether delivery capability is improving
Change failure rate	Whether speed is achieved safely
Mean time to restore	Whether teams can recover service rapidly
Developer satisfaction	Whether the platform reduces friction
Golden path escape rate	Where platform products do not meet legitimate needs
Cost per environment or transaction	Whether scalability is economically sustainable
Policy exception age	Whether risk exceptions are actively managed

BUSINESS VALUE

The Cloud Operating Model creates value by converting enterprise standards into reusable delivery capability. NVG can reduce provisioning time, improve consistency, lower operational effort and make security evidence available earlier. Teams gain autonomy without recreating infrastructure silos. The result is not simply

faster deployment; it is a higher organisational capacity for change.

TRUSTED ADVISOR PERSPECTIVE

The Trusted Advisor does not begin by asking which portal or infrastructure language NVG should adopt. The first question is where delivery friction, risk and cognitive load prevent business progress. Technology decisions follow from the operating model and user experience required.

NORDIC VENTURES GROUP LESSON

NVG initially measured the cloud platform by the number of subscriptions created and policies assigned. Adoption remained uneven because teams still depended on tickets and undocumented expertise. The platform team then shifted to product management, interviewed internal users and prioritised three high-volume journeys: application onboarding, AI experimentation and secure virtual machines. Provisioning time fell, support demand became more predictable and policy exceptions decreased. The lesson was clear: a platform creates value only when teams can and choose to use it.

ARCHITECTURE DECISION RECORD

ADR-018 — Adopt a Shared-Management Cloud Operating Model and Product-Oriented Platform Engineering

Context

NVG requires enterprise consistency across fourteen countries while enabling workload teams to deliver at cloud speed. A centralised ticket model would constrain scale, while unrestricted decentralisation would recreate fragmentation.

Decision

NVG will adopt a shared-management Cloud Operating Model. Product-oriented platform teams will provide reusable landing zones, identity, connectivity,

governance, security, observability, and developer enablement. Workload teams will own application delivery and operations within defined guardrails. Platform capabilities will be delivered through self-service, Infrastructure as Code, documented golden paths, and measurable service levels.

Positive Consequences

- Clear division of enterprise and workload responsibilities.
- Reduced provisioning time and cognitive load.
- Improved consistency, auditability and security-by-default.
- Greater autonomy for product, data, AI, and factory teams.
- Platform investment becomes measurable through adoption and flow.

Trade-offs

- Requires sustained product management rather than one-time project funding.
- Demands new skills in platform engineering, automation and developer experience.
- Poorly designed golden paths may be bypassed.
- Shared responsibility requires explicit support and escalation models.

KEY TAKEAWAYS

- Cloud agility depends on the operating model, not only the platform.
- Shared management balances enterprise control with workload autonomy.
- Platform engineering turns recurring expertise into reusable internal products.
- Golden paths should make the secure route the easiest route.
- Self-service replaces unnecessary hand-offs with automated evidence and guardrails.
- Developer experience and cognitive load are strategic concerns.

- Platform success is measured through adoption, flow, reliability, and business outcomes.

REFLECTION QUESTIONS

1. Which cloud decisions must remain enterprise-wide, and which should move to workload teams?
2. Where do ticket queues currently create delay without reducing material risk?
3. Which three user journeys would create the greatest platform value?
4. Can teams obtain a compliant environment without specialist intervention?
5. How are platform products measured, versioned and retired?
6. Do platform teams understand their internal customers?
7. Which metrics prove that the platform improves business delivery?

REFERENCES

1. Microsoft. Cloud Adoption Framework: Choose a cloud operating model and shared management cloud operations. Microsoft Learn.
<https://learn.microsoft.com/azure/cloud-adoption-framework/>
2. Microsoft. What is Platform Engineering? Microsoft Learn. <https://learn.microsoft.com/platform-engineering/what-is-platform-engineering>
3. Microsoft. Azure landing zone conceptual architecture. Microsoft Learn.
<https://learn.microsoft.com/azure/cloud-adoption-framework/ready/landing-zone/>
4. Microsoft. Azure Well-Architected Framework - Operational Excellence. Microsoft Learn.
<https://learn.microsoft.com/azure/well-architected/operational-excellence/>

5. Skelton, M., & Pais, M. (2019). Team Topologies: Organizing Business and Technology Teams for Fast Flow. IT Revolution.
6. Forsgren, N., Humble, J., & Kim, G. (2018). Accelerate: The Science of Lean Software and DevOps. IT Revolution.
7. Google Cloud. DORA research and software delivery performance. <https://dora.dev/>
8. Kim, G., Humble, J., Debois, P., & Willis, J. (2021). The DevOps Handbook (2nd ed.). IT Revolution.

TRUSTED ADVISOR PRINCIPLE

A platform succeeds when teams choose it, not when they are forced onto it.

Golden paths reduce cognitive load without removing accountability.

Platform engineering is a product discipline, not a ticketing queue.

The Trusted Advisor measures the operating model by developer experience and business outcome, not by control retained.

Now that the operating model can scale, one question inevitably follows: can it absorb a company that was never designed to fit inside it?