`

Cloud Based Factory Operating Model Service Definition

GCloud15

`

Introduction

Cloud-Based DevSecOps Factory Operating Model provides a repeatable, secure, and scalable approach to designing, building, and operating cloud-native services. The service establishes a factory-based delivery model with shared standards, tooling, governance, and automation, enabling teams to work independently while maintaining consistency and control. It treats documentation and operational artefacts as first-class outputs alongside code, embedding quality, security, and compliance by design. AI-assisted DevSecOps capabilities improve codebase understanding, documentation, and developer productivity. The service supports predictable delivery, efficient operations, and safe transition to live service ownership at scale.

The Challenges of Delivering the Factory Operating Model

A factory operating model is a standardised, automated approach to delivering and operating cloud services. It uses shared tooling, governance, and repeatable processes to enable independent teams to deliver securely and consistently, with built-in quality, compliance, observability, and predictable scaling for live services.

Implementing the Factory Model

Delivering a Cloud-Based DevSecOps Factory Operating Model presents a number of technical, organisational, and cultural challenges. These challenges arise not because the factory model is complex in isolation, but because it requires a fundamental shift away from project-led, manual delivery towards a standardised, automated, and evidence-driven operating approach.

One of the primary challenges is **establishing a reliable baseline**. Many organisations begin factory adoption with limited observability, fragmented metrics, and inconsistent documentation. Without end-to-end visibility across environments, pipelines, and services, it is difficult to make confident decisions about automation, scaling, or risk. Early stages of delivery therefore focus on introducing monitoring, logging, cost visibility, and artefact standardisation, which can initially feel like overhead before benefits are realised.

Legacy architecture and technical debt also present significant challenges. Services built in a “garage” model often contain tightly coupled components, hard-coded pipelines, manual deployment steps, and outdated dependencies. These characteristics make it risky to introduce standardised pipelines or automation without careful sequencing. Factory delivery must therefore progress incrementally, refactoring and decoupling components over time while maintaining service stability and avoiding disruption to live users.

A further challenge is **balancing autonomy with governance**. The factory model enables teams to operate independently within defined standards, but this requires clear agreement on shared tooling, quality gates, and ways of working. Without alignment, teams may resist perceived constraints or attempt to bypass factory controls. Embedding governance into pipelines and delivery processes, rather than enforcing it through external approval stages, is essential but requires careful design and stakeholder buy-in.

Cultural change is often more difficult than technical change. Factory delivery reduces reliance on individual expertise and manual intervention, replacing them with automation, metrics, and repeatable processes. While this improves resilience and predictability, it can initially challenge established roles and behaviours. Teams may need time to adapt to clearer ownership,

`

evidence-based decision making, and the discipline required to maintain standard artefacts such as tests, documentation, and operational runbooks.

Introducing **documentation and operational artefacts as first-class outputs** also presents challenges. In many environments, documentation has historically been produced late, inconsistently, or not at all. The factory model requires documentation, diagrams, test evidence, and operational information to be version-controlled, generated automatically where possible, and maintained continuously. Establishing this discipline takes time and requires tooling, automation, and clear expectations.

The use of **AI-assisted DevSecOps capabilities** introduces additional considerations. While AI can significantly improve codebase understanding, documentation generation, and developer productivity, it must be implemented with appropriate controls. Ensuring model governance, data residency, security, and repeatability is critical, particularly in public-sector contexts. AI capabilities must enhance the factory model rather than introduce unmanaged risk or dependency.

Another challenge lies in **operational transition and ownership**. Moving from build-focused delivery to a run-state operating model requires clear definition of responsibilities, particularly where multiple suppliers or teams are involved. The introduction of Site Reliability Engineering practices, automated incident detection, and proactive cost control must be aligned with existing support and service management structures to avoid duplication or confusion.

Finally, sustaining the factory model over time is an ongoing challenge. Demand patterns, technologies, and organisational priorities evolve, and the factory must adapt without reverting to bespoke or manual practices. This requires continuous review of metrics, pipelines, tooling, and standards to ensure the operating model remains effective, scalable, and aligned to real service needs.

Overall, delivering a Cloud-Based DevSecOps Factory Operating Model requires more than implementing tools or pipelines. It demands disciplined sequencing, strong governance by design, cultural change, and sustained commitment to automation, transparency, and evidence-based delivery. When these challenges are addressed incrementally and pragmatically, the factory model provides a resilient foundation for predictable delivery, controlled cost, and scalable live service operation.

Key Benefits

The following benefits are expected from the servce.

- Enables predictable, repeatable delivery of cloud services at scale.
- Reduces delivery risk through automation, standardisation, and embedded quality controls.
- Improves service resilience and reliability through observability and proactive operations.
- Supports faster, safer releases with automated testing and deployment pipelines.
- Reduces reliance on individual expertise and manual intervention.
- Provides clear visibility of cost, quality, and service health.
- Enables independent team delivery within shared governance and standards.

`

- Improves readiness for live service operation and ownership transfer.
  Improves confidence in live service readiness through measurable quality and performance signals.

Return on Investment

The following demonstrates the expected return on investment.

- Reduced delivery and operational costs through automation and removal of manual processes.
- Faster release cycles reduce cost of delay and accelerate delivery of business value.
- Improved reliability lowers incident frequency and associated support and recovery costs.
- Standardised delivery reduces rework and inefficiency across teams and suppliers.
- Predictable scaling prevents over-provisioning while meeting real user demand.
- Reduced dependency on specialist individuals lowers long-term resourcing and escalation costs.
- Built-in quality and compliance reduce costly late-stage remediation and audit effort.
- Scalable operating model lowers marginal cost as service usage increases.

Approach

The following describes the activities that are available with this service.

Delivering a Cloud-Based DevSecOps Factory Operating Model typically begins with establishing a clear understanding of the current delivery, operational, and governance landscape. This includes reviewing existing cloud environments, architectures, pipelines, tooling, security controls, documentation, and ways of working to identify gaps, constraints, and areas of inconsistency.

The service then designs the target factory operating model. This activity defines shared standards, principles, and responsibilities across delivery teams, including how services are structured, how pipelines are implemented, how quality and security controls are enforced, and how governance is embedded into delivery rather than applied as an external process.

Implementation activities focus on establishing standardised, repeatable delivery foundations. This includes creating or aligning CI/CD pipelines, infrastructure-as-code patterns, environment provisioning, automated testing, security scanning, and compliance checks. Automation is introduced incrementally to reduce risk and ensure stability of existing services.

Observability and operational readiness are core activities. Monitoring, logging, alerting, and cost visibility are implemented consistently across services to provide real-time insight into performance, reliability, and spend. Site Reliability Engineering practices may be introduced to support proactive reliability, incident prevention, and operational efficiency.

`

Documentation and operational artefacts are treated as first-class outputs. Activities include defining standard documentation structures, automating generation of technical and operational artefacts, and ensuring documentation, tests, and runbooks are version-controlled alongside code.

Where appropriate, AI-assisted DevSecOps capabilities are introduced to improve codebase understanding, documentation generation, and developer productivity, with appropriate governance and controls in place.

The service also supports transition activities, including onboarding teams and suppliers to the factory model, clarifying ownership and responsibilities, and preparing services for live operation and ownership transfer. Delivery concludes with validation of outcomes, clear reporting, and a roadmap for continuous improvement, ensuring the factory model remains sustainable and scalable over time.

Training

Training is available and will take approximately 3 days.

Support

Support is available from 08:00 to 18:00 on working days in England. 1st

Line Support

1st line support will be available between the hours of 09:00 – 17:00 via the following channels;

- Web
- Phone
- email

2nd line support which will operate between the hours of 09:00 – 17:00.

Definition of SLA's (Will provide list)

- 0 – Critical – Within 1 Hour – Call back
- 1 – Severe – Within 2 Hours – Call back
- 2- High – 24 hours
- 3 – Medium – 48 hours
- 4 – Cosmetic – 72 hours