UK Civil Aviation Authority

Safety and Security of Cloud Computing

CAP 3254

CAP 3254

Published by the Civil Aviation Authority, 2026

Civil Aviation Authority
Aviation House
Beehive Ring Road
Crawley
West Sussex
RH6 0YR

First published May 2026
First edition

Enquiries regarding the content of this publication should be addressed to: cyber@caa.co.uk

The latest version of this document is available in electronic format at: www.caa.co.uk/

Contents

Scope

This Civil Aviation Publication (CAP) provides guidance on the safety and security assurance of cloud-integrated architectures used within aerospace operational technology systems.

The guidance applies to civil aviation and space domain applications where cloud computing, virtualised infrastructure, or distributed digital services support functions that are safety-related or safety-influencing.

This publication is applicable to:

- Operators conducting Beyond Visual Line of Sight (BVLOS) or remotely piloted aircraft operations supported by cloud services.
- Air Traffic Management (ATM) and U-space service providers adopting virtualised or cloud-hosted systems.
- Organisations implementing cloud-based telemetry, tracking and control (TT&C) or mission data processing services within space operations.
- Hybrid, multi-cloud, and federated cloud environments where safety-relevant services are distributed across organisational or technical boundaries.

This publication:

- Describes architectural principles for integrating cloud infrastructure within safety-relevant systems.
- Provides guidance on the allocation of safety and security responsibilities between system operators and cloud service providers.
- Sets out analytical approaches, including Functional Hazard Assessment (FHA), Failure Modes and Effects Analysis (FMEA), and System-Theoretic Process Analysis (STPA and STPA-Sec), for the assessment of distributed cloud-enabled systems.
- Addresses the treatment of Service Level Agreements (SLAs) and Service Level Objectives (SLOs) as elements of the safety and security assurance framework.

This publication does not:

- Replace, supersede, or modify existing certification requirements, regulatory obligations, or applicable airworthiness standards.
- Mandate specific technologies, vendors, or commercial cloud service models.
- Provide detailed cyber threat intelligence or standalone cybersecurity certification guidance.

The guidance is intended for system designers, operators, service providers, cloud infrastructure providers supporting safety-relevant services, and regulatory personnel responsible for oversight of cloud-enabled aerospace systems.

Assumptions

This publication assumes that the reader:

- Is familiar with established aerospace safety assessment processes, including Functional Hazard Assessments (FHA), Failure Modes and Effects Analysis (FMEA).
- Possess working knowledge of cybersecurity principles relevant to operational technology environments
- Understands basic cloud computing concepts including virtualisation, containerisation, service orchestration, and Service Level Agreements (SLAs)
- Is engaged in the design, assurance, operation, or oversight of safety-relevant aerospace systems.

This publication does not provide introductory training in safety engineering, cybersecurity, or cloud computing fundamentals.

Acronyms

Advanced Message Queuing Protocol: AMQP

Air Traffic Management: ATM

Application Programming Interface: API

Artificial Intelligence: AI

Beyod Visual Line of Sight: BVLOS

Detect and Avoid: DAA

Failure Mode and Effects Analysis: FMEA

Functional Hazard Assessment: FHA

Machine Learning: ML

Message Queuing Telemetry Protocol: MQTP

Quality of Service: QoS

Service Level Agreements: SLA

Service Level Objectives: SLO

Software Defined Networks: SDN

System-Theoretic Process Analysis: STPA

Telemetry, Tracking and Control: TT&C

Time to Live: TTL

Unmanned Air System: UAS

Introduction

The aerospace sector is undergoing a rapid digital transformation, driven by advances in connectivity, automation, and data analytics. Across both aviation and space domains, operations are becoming increasingly networked and data centric. Cloud computing now plays a vital role in this development by providing scalable computational resources, secure data storage, and globally accessible analytics for safety critical systems.

Cloud-based architectures are being adopted for a wide range of aerospace applications. In the aviation domain, they are enabling Beyond Visual Line of Sight (BVLOS) operations, predictive maintenance through machine learning and Detect and Avoid (DAA) systems for Unmanned Air System (UAS) traffic management. Conventional Air Traffic Management (ATM) systems are also increasingly adopting cloud and virtualised infrastructure to support modernisation, data distribution, operational resilience. In the space domain, they support satellite Telemetry, Tracking & Control (TT&C) processing, and mission data management, allowing ground operations to manage resources using globally distributed cloud infrastructure and ground stations.

These developments mark a significant shift from traditional architectures, which have historically been deterministic, tightly controlled, and physically isolated systems. Cloud environments are by contrast elastic, shared, and continuously developing, thus introducing new complexity and interdependence.

These developments raise several critical questions:

- How can cloud platforms be safely integrated into safety-critical aerospace systems?
- How can safety and security assurances be maintained across externally managed distributed infrastructures?
- What analytical methods can effectively capture the interactions between physical systems, software, and cloud services?

This CAP addresses these questions by linking cloud architectures with safety and security analysis frameworks. It demonstrates how both traditional reliability-based methods and system theoretical approaches can be applied to assess and manage the risks of cloud integrated systems. The focus will not be on describing generic cloud technologies but on understanding their implications for operational safety, security, and continued assurance in aerospace applications.

Cloud Architecture in Aerospace Applications

Cloud computing has become an enabler of modern aerospace operations, providing the computational elasticity, data integration, and global reach once limited by physically co-located systems. For safety critical applications however, cloud architectures must be designed not only for performance, but also for predictability, verifiability, and controlled development.

Deployment Models

Different deployment models govern how control, visibility, and assurance responsibilities are distributed between the operating organisations and cloud providers. Each model presents a different balance between control, scalability, and transparency.

Public Cloud

Operated by third-party providers, shared across multiple tenants.

Advantages: Elastic scalability, rapid access to advanced computing and Artificial Intelligence (AI) capabilities.

Challenges: Limited infrastructure transparency, varying data jurisdiction rules.

Private Cloud

Dedicated infrastructure operated by a single organisation, either on-premises or exclusive use through a third-party host.

Advantages: Deterministic performance, complete configuration control, full auditability.

Challenges: Higher maintenance and lifecycle cost.

Hybrid Cloud

Combines Private and Public resources via secure interfaces.

Advantages: Balances scalability and control; supports selective segregation of workloads.

Challenges: Requires synchronisation and unified policy enforcement.

Multi-Cloud

Uses multiple independent providers for resilience and flexibility.

Advantages: Reduces vendor dependence and common-mode failures.

Challenges: Complex identity and configuration management.

Federated Cloud

A cooperative network of autonomous clouds governed by shared trust, identity, and policy frameworks.

Advantages: Facilitates collaboration white retaining domain autonomy.

Challenges: Requires consistent governance, and interoperable assurance policies.

Edge and Fog integration

Extend cloud resources closer to data sources; airborne, ground, or orbital.

Advantages: Reduced latency and continued operation during network disruption.

Challenges: Ensuring state synchronisation and coherent decision authority across distributed nodes.

Core Cloud Technologies

Modern cloud ecosystems rely on a portfolio of technologies that together provide scalability, isolation, and verifiable control.

- **Virtualisation and Hypervisors**: partition hardware into isolated environments, ensuring determinism and containment.
- **Containerisation and Microservices**: modularise systems for independent deployment, enabling localised verification and rapid recovery.
- **Orchestration and Service Meshes**: manages containerised workloads and enforces communication policy & provide predictable routing and telemetry visibility.
- **Data Streaming and Message Queues**: guarantees ordered, reliable data exchange (AMQP, MQTT, Kafka) to maintain telemetry integrity.
- **Edge and Fog Computing**: provides local processing for autonomy and pre-validation of data.
- **Storage and Data Management**: use versioned, replicated, and immutable storage for traceable mission data.
- **Serverless and Event Driven Computing**: execute on demand tasks such as alerting or log processing, outside of time critical loops.
- **Software Defined Networks and Network Function Virtualisation**: enables programmable redundant network paths with controlled latency.
- **Security, Identity, and Key Management**: implement zero trust architectures, cryptographic signing, and multi-factor authentication.

Architectural overview

Across deployments and operational contexts, cloud systems in operational technology environments typically adopt a three-layer hybrid architecture, defining how computation, communication, and controls are distributed.

- **Edge layer**: onboard avionics, ground control stations, and mission computers executing time-critical functions such as stabilisation and collision avoidance.
- **Communication layer**: terrestrial and satellite networks providing redundant, encrypted, latency-controlled connectivity.
- **Cloud layer**: scalable computation, storage, and analytics platforms hosting mission data management, simulation, and coordination systems.

These layers form an integrated control loop. The edge generates the data and immediate control actions, the communication layer transmits and synchronises the information, and the cloud aggregates, analyses, and redistributes decisions or insights.

From a safety perspective, this structure provides fault containment and clear authority boundaries. However, control logic that spans layers could often have misconfiguration or timing faults that can propagate through other layers, requiring holistic, cross layer assurance methods.

Cloud-Hosted Applications and Services

Cloud infrastructure does not operate in isolation; it hosts a portfolio of applications and services that support, augment, or coordinate aerospace operations. These cloud-hosted applications vary significantly in their criticality, timing sensitivity, and assurance requirements, and must therefore be explicitly identified and classified within the system architecture.

Typical cloud hosted applications in aerospace operations include:

- **Mission data management services,** including telemetry aggregation, storage, replay, and post-mission analysis.
- **Situational awareness and coordination services,** such as traffic information distribution, airspace status updates, and collaborative planning tools.
- **Predictive analytics and machine-learning services,** supporting maintenance forecasting, anomaly detection, or advisory decision support.
- **Simulation and digital-twin environments,** used for operational rehearsal, contingency evaluation, or performance monitoring.
- **Identity, authorisation, and trust services**, including credential management, certificate authorities, and policy enforcement points.
- **Operational monitoring and observability services**, aggregating logs, metrics, and security events across distributed components.

While many of these applications are not directly safety-critical in isolation, they frequently influence safety-critical decisions by providing advisories, constraints, or shared situational context. As a result, their failure, corruption, or delay may still contribute to hazardous system states.

Each cloud-hosted application must therefore be:

- Mapped to its supported operational functions
- Assigned an appropriate safety and integrity classification
- Analysed within the safety-security analysis activities as part of the overall system
- Subject to architectural constraints that prevent non-critical services from degrading safety-critical behaviour

This explicit identification of cloud applications prevents implicit reliance on services whose failure modes or assurance boundaries are otherwise obscured by the underlying infrastructure abstraction.

Architectural Principles

Cloud systems must apply redundancy, determinism, and segregation within a distributed digital system. The following principles form the foundation of safe and secure cloud architecture.

- **Domain Separation**: logical and physical isolation between mission critical and non-critical workloads prevents cascading faults and exposures to unverified components.

- **Redundancy**: redundant compute nodes, storage replicas, and communication paths enable continued functions under component or link failures.

- **Data Integrity and Provenance**: each transaction – telemetry or control – must be verifiable, timestamped, and auditable to reconstruct system state and decision.

- **Deterministic Performance within Elasticity**: resource scaling must maintain predictable timing behaviour. Reserved resources or edge co-processing mitigate the non-deterministic nature that is typical of shared clouds.

- **Secure by Design**: authentication, encryption, and access controls are embedded at every layer, ensuring that security mechanisms directly support safety objectives.

- **Continuous Observability**: unified monitoring of performance, integrity, and security provides early detection of anomalies and assurance continuity throughout the operation.

Safety and Security of Cloud Architecture

Digitisation of the aerospace ecosystem blurs the boundary between security and safety. As functions migrate into software, virtualised infrastructure, and cloud services both random, and malicious actions act on the same digital pathways; message buses, Application Programming Interfaces (APIs), orchestration layers, and identity systems. As a result, the mechanisms that produce safety hazards and security breaches increasingly overlap.

Therefore, cloud infrastructure will depend on three properties that are shared by both safety and security: Integrity, Availability and Authenticity.

- **Integrity**: unaltered data and commands across virtualisation, message queues, and storage.
- **Availability**: reliable execution of compute and communication services when required.
- **Authenticity**: verified identity of data sources, services, and operators.

Loss of any of these properties whether by random failure or malicious actions can create hazardous states. In cloud-integrated systems, a loss of timeliness can have the same operational effect as loss of availability. Excessive latency, jitter, or delayed delivery of otherwise valid data or services can invalidate safety assumptions and lead to hazardous states, even when infrastructure components remain nominally operational.

Building on the architectural principles, the safety and security principles that should govern the cloud architecture of an operational technology system can be described as a set of three layers.

Cloud layer

The cloud layer provides high-level processing, where the coordination and storage functions support the overarching operational system. If both safety and non-safety critical services are running in the cloud, they must operate with logical and physical separation to limit the impact of a compromise on safety critical function. Critical services running on this layer also require deterministic computation, which involves operation within predictable resource and timing boundaries; therefore, safety process cannot be dependent on latency variation or shared-resource contention.

Data integrity and provenance of all data used in safety must be verifiable and traceable to its source. Configuration and change management should be adhered to across system configurations, with all software releases undergoing rigorous change control and impact assessment to verify that no update introduces unsafe or insecure conditions.

Continuous monitoring of cloud performance and early detection of faults or intrusions can help identify issues, such as latency drift or data corruption that could impact safety. Only authorised personnel and processes should be able to modify, access, or operate safety-

related components; the principle of least privilege, combined with authentication, prevents unauthorised modifications or intrusions.

Service Level Agreements and Assurance Responsibilities

The safety and security of cloud-integrated aerospace systems depend not only on technical architecture, but also on the explicit assurance commitments between cloud service providers and system operators. These commitments are typically formalised through Service Level Agreements (SLAs) and Service Level Objectives (SLOs), which define the expected performance, availability, and support characteristics of cloud services.

For safety-relevant applications, SLAs must be treated as assurance artefacts rather than commercial conveniences. Availability guarantees, latency bounds, recovery times, data durability, and change notification obligations directly influence whether safety objectives can be met in operation.

SLAs should explicitly address:

- **Availability and resilience**, guarantees for safety-supporting services, including redundancy, geographic diversity, and uptime for services.
- **Performance and latency objectives**, especially where cloud services influence time-bounded decisions or advisories.
- **Change management and notification**, ensuring that infrastructure updates, maintenance activities, or configuration changes do not invalidate safety assumptions without prior assessment.
- **Incident response and escalation**, including timelines for fault notification and joint investigation.
- **Auditability and transparency**, allowing operators and regulators to verify compliance with declared assurance claims.

The allocation of responsibility between cloud provider and system operator must be unambiguous i.e. a RACI matrix for operations. While the provider may be responsible for infrastructure availability and baseline security controls, the operator retains responsibility for ensuring that safety-critical functions remain within validated operational bounds. This division of responsibility must be reflected in both technical design and contractual terms.

Where safety objectives depend on cloud service behaviour, those objectives must be traceable to enforceable SLA parameters. If an SLA cannot guarantee the necessary integrity, availability, or timeliness, then the system architecture must incorporate compensating controls, such as local autonomy, buffering, or degraded-mode operation.

Treating SLAs as part of the safety-security assurance framework ensures that contractual boundaries do not become hidden hazard sources within distributed cloud-enabled systems.

Communication layer

This layer connects the cloud services with the edge systems. It must guarantee data integrity, availability, and timeliness across potentially unreliable or contested networks. The integrity of the communication system must be maintained by preventing unauthorised command injections, spoofing, or message alteration. This can be achieved by digital signatures, mutual authentication, and message level integrity checks. Another method to ensure minimal transmission errors is by detecting and correcting them via checksums, sequence validation, and ensuring retransmission policies are in-place.

The availability is maintained by having system redundancy and path diversity, which are achieved by using multiple independent communication paths. This will mitigate against link failures or intereferences, by using multiple network technologies (mobile or satellite); however, when the system is affected in this layer, it must not lead to an unsafe condition. The system must be able to revert to a predined autonomous mode to ensure safety is maintained. If it cannot be maintained, safety and non-safety traffic should be isolated to prevent bandwidth contention. If availability is reduced, it must still be able to maintain its operation in a safe condition. This segregation is achieved via logical and physical network segmentation.

Timeliness of safety critical data must be achieved through deterministic delivery mechanisms, such as network slicing or dedicated channels which prevents data from being delayed by non-critical traffic or congestion (deliberate or non-deliberate). Time synchronisation ensures data relevance for safe operations; system clocks must be securely synchronised using trusted sources and delayed or stale messages must be rejected to prevent replay attacks or confusion.

Edge Layer

Edge layer is the closest to the operatonal environment and typically encompasses onboard systems, sensors, actuators, and local process nodes. This is where real time safety functions reside. Safety-critical decisions, such as emergency actions or fallback models, must remain executable locally without dependency on the cloud or external communication. This ensures that a loss of connectivity or an erroneous remote command cannot endanger the system, thereby ensuring local autonomy and independence.

The input validation of all incoming sensor data must undergo plausibility and range checks to prevent unsafe reactions to erroneous or falsified data. Security mechanism such as cryptographic authentication of sensor feeds and anti-spoofing checks strengthen the integrity of the data.

Verifying the authenticity and the integrity of the firmware and software before execution, using cryptographic signatures and hardware root of trust will prevent malicious code from

compromising critical control loops. Along with these, self-checking algorithms and redundant control paths will detect and isolate failures. Redundant sensing and computational paths ensure continued operation even if a component fails or becomes compromised. The edge system must have safe state transition in the event of anomalies, detected faults, or communication losses. This transition must be protected against manipulation or denial.

Safety and Security Co-Engineering

Cloud enabled aerospace systems are too dynamic to be able to treat safety and security as two separate sequential approvals. Services scale up and down, network paths change, data schemas evolve, Machine Learning (ML) models are retrained, yet the safety case must still hold. Integrated safety-security co-engineering means that safety and security are:

- Defined together – with the same function, same hazards.
- Implemented together – with the same architecture, same policies.
- Verified together – with the same monitoring, same evidence.

In classic aerospace systems, safety is about component reliability and fault tolerance, security is about keeping attackers out. In a cloud system:

- A security failure such as an unguarded API, expired certificate, or poisoned data stream can look like a safety failure such as a stale command, missing telemetry, or wrong advisory.
- A safety mechanism like the throttling of non-critical traffic can create security gaps like an operator becoming forced to bypass controls.
- A benign cloud behaviour like auto scale, node eviction, or route change can invalidate safety timing assumptions.

Integrated safety-security assurance for cloud-based aerospace systems relies on combining analytical methods that address various levels of system abstraction. No single technique can model both the functional purpose of the system and the dynamic, distributed interactions that are characteristic of modern cloud platforms.

Analysis Techniques

The three methods this section will address are Functional Hazard Assessment (FHA), Failure Modes and Effects Analysis (FMEA), and System-Theoretic Process Analysis (STPA and STPA-Security).

FHA is the natural starting point because cloud environments change what counts as a function in an aerospace system. Traditional avionics might treat flight control as a single function; however, a cloud integrated system could deconstruct it into on-board control, data relay, remote planning, and AI-based advisory services.

FHA forces explicit definition of which functions are safety-critical, what the operational consequences are in case of delay, corruption, etc and what safety objectives or integrity levels each function must satisfy. In the cloud context, this is vital because a function’s failure may result not from a physical fault but from virtualisation, network partitioning, or provider reconfiguration. Therefore, this technique anchors the entire assurance process in functional intent, defines the quantitative bounds for latency, freshness, and trust.

This provides the first mapping of security events, such as credential compromise which leads to safety outcomes like unauthorised control.

While FHA describes the what and the so what, FMEA investigates the how. Cloud architecture contains many moving parts such as message brokers, orchestrators, virtual machines, APIs, etc; each with its own failure and recovery modes.

FMEA suits this environment because it manages a large number of components systematically. It links failure modes like pod evictions, queue overflow, token expiry, etc to system level effects defined in FHA. FMEA quantifies likelihood, detectability, and severity, enabling prioritisation of mitigations; and translates the qualitative hazards into specific design or configurations controls.

FMEA also bridges the safety-security gap. Many failures in cloud systems are non-malicious manifestations of security lapses like expired certificates or broken trust chains. By treating these within the same framework, we avoid duplication of analysis and ensure consistent risk scoring.

Even with FHA and FMEA, some failures will not be captured because they involve emergent behaviours rather than discrete component faults. STPA models the system as a hierarchy of control loops and examines how unsafe actions arise when control, feedback, or contextual information is missing, incorrect, delayed, or untrusted.

In the cloud system, controllers include autonomy modules, cloud planners, orchestration policies, and human operators. Feedback paths include telemetry streams, observability data, and security alerts. Unsafe actions may stem from delayed data, loss synchronisation, competing authority, or policy drift – none which will appear in conventional FMEA tables. STPA-Security (STPA-Sec) extends this reasoning to security triggered hazards such as issuing valid commands with forged credentials, misapplying safety logic due to tampered configuration, or losing safe control when trust or policy information becomes inconsistent across providers. In this system, these unsafe control actions rarely arise in isolation but are typically the result of identifiable adversarial behaviours acting on exposed digital control surfaces.

Because cloud infrastructure exposes safety-critical functions to digital attack surfaces, the STPA-Sec activity should incorporate inputs from a focused cyber-threat assessment. This allows malicious triggers such as API misuse, token theft, data poisoning, or cloud orchestrator manipulation to be treated as causal factors within the co-engineered FHA-FMEA-STAP workflow, rather than a separate security analysis.

Integrating the techniques

Each of these techniques feed into the next, ensuring that safety and security information remains consistent across layers and through time. FHA identifies critical functions and hazard severities whilst FMEA maps those functions to physical and virtual components, exploring how failures in hardware, software, or service configurations might produce functional loss. The severity classifications from FHA guide the weighting and

prioritisation within FMEA. The high-risk failure modes identified in FMEA are inserted as disturbances in the STPA model to evaluate how the system control logic responds. For example, the message broker lag or orchestrator eviction from FEMA become contextual causes in STPA causal scenarios. This step tests whether the system's control structure has sufficient feedback, validation, and degraded mode behaviour to remain safe. The constraints derived from STPA are then formalised into design requirements, orchestration policies, or runtime checks. This closes the loop by feeding back into FHA/FMEA documentation as verified mitigations.

This allows safety and cybersecurity engineers to work on the same functional definitions rather than different taxonomies. Every runtime policy or configuration parameter maps back to a specific hazard or constraint.

Mapping Co-Engineering to Cloud Layers

Edge layer

If there is a vehicle in motion, the requisite safety need would be to keep the vehicle safe - even if everything else is on fire. The linking security need would therefore be to reject bad or forced instructions. Thus, the co-engineering rule would be to apply cloud/remote commands only if: the freshness is within the current threshold, the signature is valid, the source is authorised, and the timing budget has not been breached. If these bounds are not satisfied, then the system should stay in a local, conservative mode.

This would be an STPA-Sec constraint that has been derived from an FHA hazard around applying unsafe commands, and FMEA items such as broker lag and IdP outage.
It involves both safety and security, do not act on old data and do not act on untrusted data.

To implement this, the cloud system would use controls such as signed containers, a local mission cache, or key storage in the hardware.

Communication layer

The communication system has a safety need of predictable, ordered, bounded latency delivered for control. The security need that is linked to this would be that data delivered on the network is authenticated, the system is replay-protected and has path-controlled delivery. Therefore, the co-engineering rule would be that the control flow must use Quality of Service (QoS) guaranteed rule, mutually authentic channels; if QoS cannot be met, the system must enter a degraded but safe mission state.

The co-engineering rationale here would start with a safety argument that states that if the latency is greater than the latency threshold, then enter safe mode; the security argument would be that if the certificate is expired, drop the connection.

To implement this, Software Defined Networks (SDN) intent needs to be defined, define multipath, control the Time to Live (TTL) for a topic, and network Service Level Objective (SLO) monitors tied to degraded-mode triggers at the edge.

Cloud layer

A safety need in the cloud layer would be orchestration, and analytics must not starve, evict, or reorder critical services. The linked security need would be that only verified code, and identities are able to affect critical services. Therefore, the co-engineering rule would be that no change to safety classified services may occur unless the change is policy validated, the identity has been authenticated, and the change preserves minimum capacity and timing SLO.

This was derived from the STPA unsafe action; the orchestrator scales down safety service below minimum, and the FMEA item level failure would be autoscaler misconfiguration and node pressure.

To implement the safety and security need, the system would have admission controllers like Open-Policy-Agent/Gatekeeper, PodDistributionBudgets, resource reservations, signed image, pre-service identities, and audit logs.

Co-engineering as a lifecycle activity

Traditional safety and security assurance processes have often been built as document-driven, one-time exercises that end with certification or deployment. For cloud-enabled aerospace systems, this approach is no longer viable. The underlying infrastructure changes continuously; orchestration policies evolve, network topologies shift, and data models or machine-learning components are retrained. The assurance argument therefore cannot remain static and must evolve with the system.

Integrated safety-security co-engineering treats assurance as a living lifecycle rather than a fixed review activity. The process begins at the concept and architecture phase, where FHA defines what functions are safety-critical and what hazards emerge when they fail. This early analysis establishes critical functions such as command-and-control continuity, data freshness, and trust enforcement which defines quantitative safety objectives for latency, availability, and authentication integrity.
These objectives function as boundary conditions for all later engineering decisions.

The design phase transforms the FHA outputs into detailed FMEA analysis, identifying specific cloud, edge, and communication components whose faults could violate the safety objectives. The resulting ranked failure modes guide architectural mitigations such as redundant carriers, time-to-live limits, and admission-control policies. At this stage, STPA and STPA-Sec are applied to evaluate how those failure modes could interact within the overall control structure, revealing unsafe control actions that cannot be discovered through component centric analysis alone.

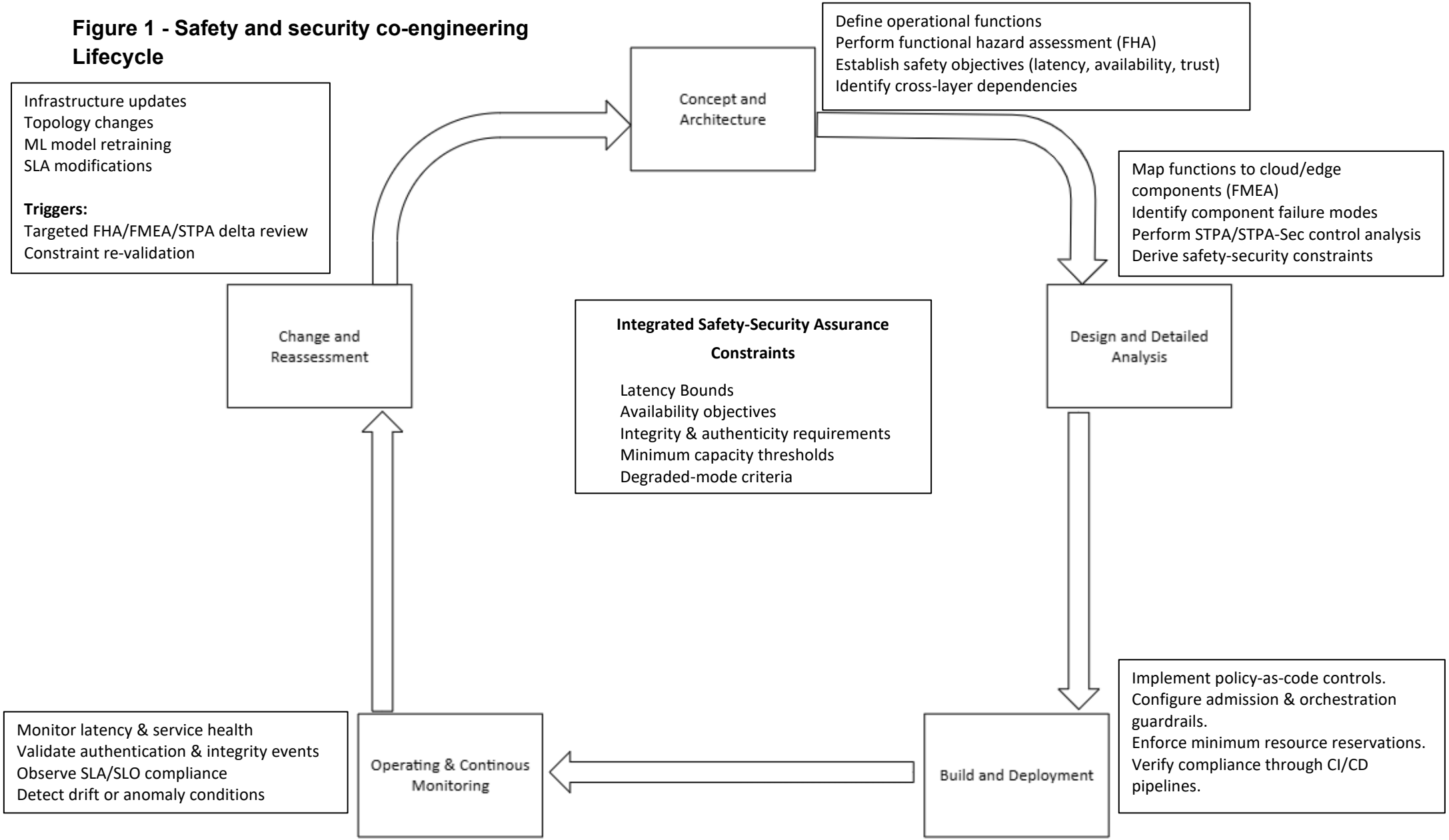
In the build and deployment phase, co-engineering principles manifest through automation. The constraints derived from STPA such as maximum allowable latency, minimum replication of safety services, or command freshness limits are implemented as executable infrastructure policies. Continous integration pipelines verify compliance with these constraints before any change is deployed. Safety, security and operational

configuration therefore evolve together, and every modification therefore carries a verifiable assurance trail.

In the operational phase, runtime observability metrics replace periodic audits as the primary means of assurance. Telemetry and event data are evaluated against the same safety-security constraints defined during the design stage. A breach in latency, authentication, or service capacity thresholds produces measurable evidence in both safety monitoring and cybersecurity dashboards.

Finally, future changes such as an upgrade of infrastructure, data flows, or model behaviour should trigger an automatic re-evaluation of the affected analysis. Lightweight, tool assisted FHA, FMEA, or STPA updates can be run to verify that constraints remain valid under new conditions. This continuous feedback loop ensures that assurance is not an artifact of a single moment in time but a sustained, traceable property of the operational system.

Figure 1 - Safety and security co-engineering Lifecycle



Co-engineering in Federated and Multi-Cloud Systems

Modern aerospace and UAS rarely operate within a single technical or organisational boundary. Airspace management, cloud coordination, and communication services are increasingly spread across multiple providers, operators, and national infrastructures. In such an environment, the assurance problem extends beyond technology as it becomes an issue of inter-domain trust and accountability.

Integrated safety-security co-engineering approaches this challenge by treating each participating domain – whether an operator, a U-space service provider, a telecommunications carrier, or a cloud host as a semi-autonomous control entity with its own hazards, failure modes, and responsibilities. The process begins by performing an FHA within each domain, identifying the functions that directly affect cross domain safety (for instance, time synchronisation, identity management, or telemetry distribution). These analyses are then connected through an inter-domain STPA-Sec exercise that focuses specifically on the trust boundaries and the points where data, commands, or assurances cross from one organisation's control plane into another's.

This approach acknowledges that no single domain can guarantee end-to-end safety on its own. Instead, safety emerges from the explicit contracts of assurance that exist between domains. For example, a U-space coordination service may trust positional data from an operator's cloud only if it is signed, timestamped, and verified within a known freshness window. Similarly, a ground operator's system may treat all external advisories as low-trust inputs until validated locally.

Through this mechanism, co-engineering creates a federated assurance architecture; each domain enforces local safety and security controls, but shared constraints such as freshness, authenticity, minimum capacity, and bounded latency remain consistent across the system-of-systems. If any participating domain violates those constraints, others can immediately identify degraded trust and switch to safe local modes.

The result is a system that preserves the continuity of assurance even when no single organisation or infrastructure is in full control. This is particularly relevant for urban BVLOS operations, where command, control, and situational awareness depend simultaneously on mobile networks, cloud services, and shared airspace information exchanges. By applying co-engineering across organisational and technical boundaries, aerospace operators can maintain a coherent safety case that is both distributed and resilient; a necessary property for the next generation of cloud-enabled aviation ecosystems.