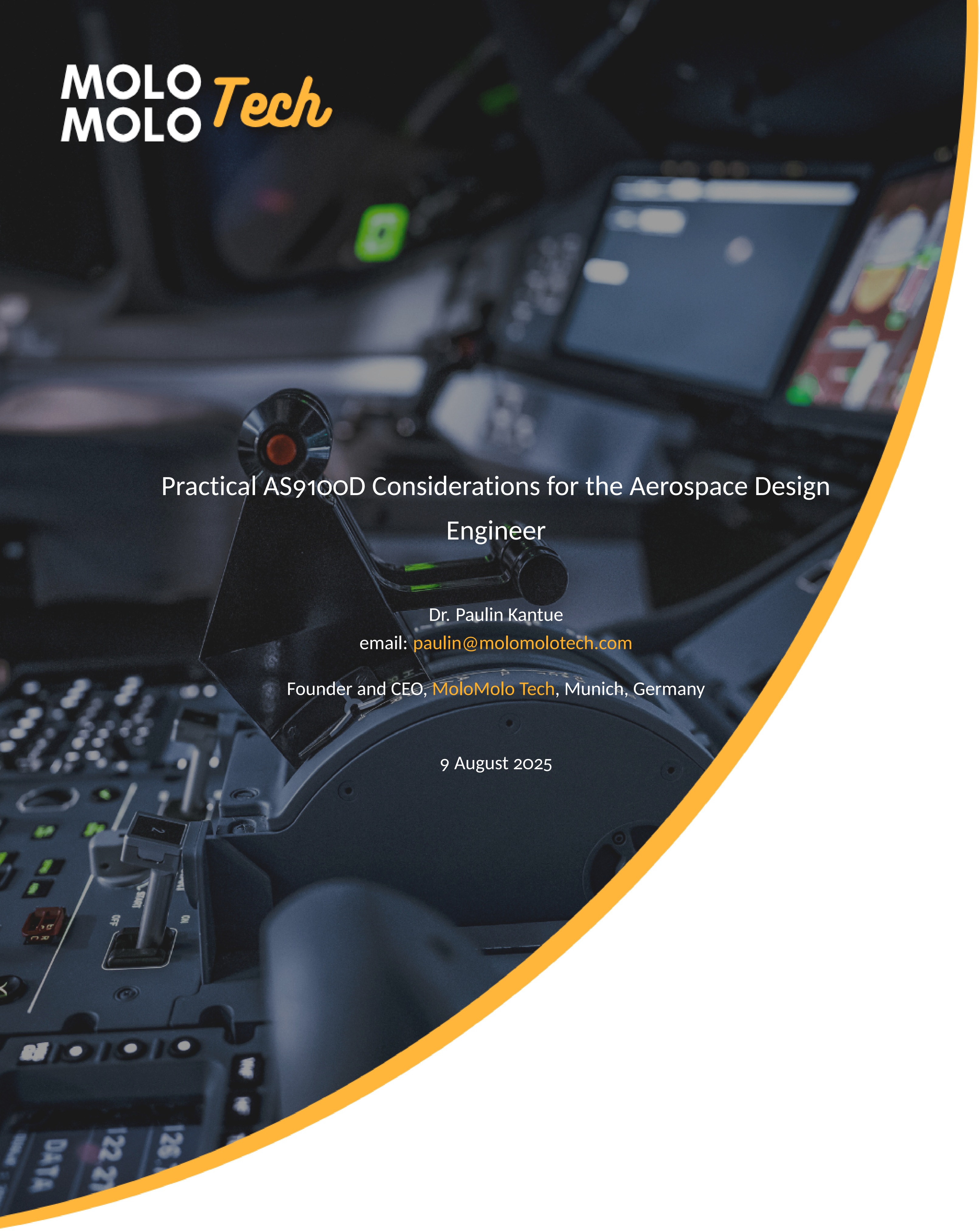

Practical AS9100D Considerations for the Aerospace Design Engineer

Dr. Paulin Kantue
email: paulin@molomolotech.com

Founder and CEO, MoloMolo Tech, Munich, Germany

9 August 2025

Introduction

Modern aerospace design engineers operate in an increasingly high-stakes environment, marked by greater system complexity, tighter development timelines, evolving certification demands, and expanding stakeholder expectations. Amid this environment, the design engineer must perform a balancing act between ensuring strict conformity to design requirements, accommodating dynamically evolving customer (e.g. chief engineering, pilots, software) input, and managing limited visibility into aircraft-level integration and operational criteria. This tension often introduces ambiguity and risk into early-stage decisions, where the cost of rework grows exponentially the further a project progresses.

A recurring challenge in this landscape stems from the partial or inconsistent application of Model-Based Systems Engineering (MBSE) practices. To their detriment, design engineers often conflate Model-Based Design (MBD) with MBSE. Although they both involve modelling tools, MBD focuses on simulating and validating detailed subsystem behaviour (e.g., control algorithms or aero models), MBSE is concerned with the higher-level architectural, functional, and requirement relationships that span across the entire system lifecycle. Moreover, it is common for engineers and stakeholders to misunderstand its purpose, treating models merely as documentation artefacts rather than decision-support tools. As a result, critical elements such as requirement traceability, interface clarity, and verification planning may be misrepresented or underdeveloped, compromising the usefulness of MBSE in supporting Quality Assurance (QA) and compliance objectives.

Simultaneously, the perceived distance between design activities and quality management goals, such as those defined in AS9100D:2016, results in a fragmented workflow. Design engineers may view quality as the domain of inspectors or auditors, rather than as a shared framework for ensuring functional safety, regulatory compliance, and customer satisfaction. This misalignment can be particularly evident when system-level requirements—such as flight control system (FCS) constraints, expected handling qualities, or maintenance accessibility, are not fully decomposed and understood at the subsystem or component level during early design phases.

In a multi-national unmanned aerial system (UAS) program, subsystem design teams implemented their deliverables based on incomplete system-level requirements due to delayed updates from the platform integrator. The teams employed MBSE tools, but without clear guidance on configuration management, modelling approach applicability or integration verification, resulting in several rework cycles during late-stage hardware/software integration. Moreover, QA checkpoints were reactive rather than embedded into the modelling process, which led to a series of last-minute corrective actions that could have been mitigated through earlier, model-informed quality planning.

This paper explores how aerospace design engineers can make tangible, proactive contributions to quality assurance by meaningfully integrating MBSE practices with the process-driven rigour of AS9100D:2016. It aims to bridge the operational gaps by offering practical strategies, examples, and mapping approaches that elevate the design process from merely satisfying requirements to enabling confident, compliant, and high-performing system development.

Understanding the Quality Landscape

What AS9100D Requires from a Design Perspective

The AS9100D:2016 standard, widely adopted across the aerospace industry, defines a comprehensive framework for quality management systems (QMS) that applies not only to manufacturing and delivery but also critically to the design and development phases of aerospace systems. From a design engineer's standpoint, AS9100D:2016 emphasises the importance of embedding quality assurance principles into the earliest phases of the lifecycle to ensure product conformity, traceability, and alignment with customer and regulatory expectations.

Clause 8 of the standard—*Operation*—is particularly relevant, as it outlines the requirements for design and development planning, inputs, controls, verification, validation, and change management. It requires that organisations establish documented processes to:

- Define clear design and development stages;
- Identify and control design inputs (e.g., functional, performance, interface, and regulatory requirements);
- Ensure that outputs meet input requirements and are suitable for verification;
- Implement robust review, verification, and validation activities at appropriate stages;
- Control changes and maintain configuration integrity throughout the design lifecycle.

Furthermore, AS9100D:2016 **Clause 6** encourages proactive planning by requiring the identification of risks and opportunities related to design activities, and ensuring that adequate resources, tools, and responsibilities are assigned. **Clause 7** outlines expectations for support functions, such as configuration management, documented information control, and the competence and awareness of personnel contributing to design outcomes.

Critically, design engineers are expected not only to deliver technically correct solutions but also to ensure that their design process supports:

- Full traceability of requirements;
- Prevention of nonconformities through design error detection and risk mitigation;

- Evidence-based decision-making;
- Seamless collaboration across interdisciplinary teams, including quality and systems engineering.

Despite the clear intent and structure provided by AS9100D, many of the design-related activities outlined in the standard such as: requirements traceability, risk-based design planning, or structured verification, are either inconsistently applied, poorly documented, or treated as box-ticking exercises rather than integral parts of the engineering workflow. This is often due to limited awareness or training across engineering teams, unclear ownership of quality responsibilities within the design function, and the perception that such activities lie outside the “real work” of engineering.

As a result, these quality-driven processes lack the visibility and cultural reinforcement necessary to embed them into the daily habits of design engineers. Documentation may exist, but it is not updated or referenced meaningfully during decision-making. Verification checklists may be created, but are disconnected from model artefacts or actual design behaviours. In many organisations, only a few individuals—often from the quality or systems engineering functions- understand the full scope of these requirements, leaving most team members unaware of how their design decisions impact audit readiness, process compliance, or customer satisfaction.

This lack of systemic engagement makes it nearly impossible to define or measure the effectiveness of the design-related quality processes outlined in AS9100D. Without consistent ownership, traceability, and data-driven feedback loops, improvement initiatives tend to be reactive, isolated, or based on anecdotal experience rather than empirical insight. Consequently, opportunities for learning, optimisation, and risk prevention are missed, while design quality becomes heavily reliant on downstream corrections instead of upstream engineering discipline.

Why Model-Based Systems Engineering Matters

Model-Based Systems Engineering (MBSE) has emerged as a foundational approach to manage the growing complexity, interdependencies, and compliance challenges in modern aerospace system development. Unlike traditional document-centric practices, MBSE enables teams to use formalised models to define, analyse, and validate system requirements, behaviour, architecture, and performance throughout the entire development lifecycle. For design engineers, MBSE offers a structured and integrated environment to move beyond disconnected spreadsheets and static drawings toward dynamic, traceable, and analysis-capable representations of the system.

At its core, MBSE supports the creation of a shared digital thread that connects high-level stakeholder needs to design solutions and verification artefacts. This facilitates early understanding of design intent, better alignment between systems engineering and design activities, and earlier identification of integration or requirement issues. MBSE allows design engineers to visualise how their subsystem or component fits into the broader system, enabling more informed trade-offs, interface considerations, and risk decisions.

Critically, MBSE provides mechanisms for maintaining consistency and traceability in the face of evolving requirements—one of the most persistent challenges faced by engineering teams. When properly applied, changes in requirements or constraints can be propagated through the model, highlighting impacts across design elements, test cases, and compliance targets. This creates a powerful feedback loop that supports both quality assurance and agile system evolution. MBSE also brings significant value in verifying and validating the system **before** physical integration begins. This should not be a surprise to MBD engineers, but is a constant gap for those who are more hardware inclined.

However, to realise these benefits, MBSE must be seen not merely as a modelling tool but as a design discipline. Misunderstandings still persist, particularly among design teams, that MBSE is the responsibility of systems engineers or architects alone. In reality, effective MBSE requires collaboration across domains, with design engineers playing a critical role in defining accurate behaviours, validating parameter ranges, and maintaining fidelity between model and implementation. When embraced this way, MBSE becomes a practical enabler of both engineering excellence and compliance, not an overhead burden.

ISO 15288, AS9100D, and the Role of MBSE in Enabling Compliance

The coexistence of ISO/IEC 15288 and AS9100D reflects a deliberate alignment between two complementary perspectives: one rooted in systems engineering best practices, the other in quality management and regulatory compliance. ISO 15288 defines the life cycle processes for systems engineering, covering technical, management, agreement, and enabling activities across the entire system life cycle. In parallel, AS9100D specifies the requirements for an aerospace organisation's Quality Management System (QMS), ensuring that products and services consistently meet customer and regulatory requirements.

While AS9100D establishes the "what" of quality management—such as maintaining traceability, managing change, and ensuring verification, ISO 15288 outlines the "how" from an engineering execution standpoint. When viewed together, these frameworks reinforce the need for engineering processes that are not only rigorous and technically sound, but also controlled, auditable, and aligned with compliance objectives.

Model-Based Systems Engineering (MBSE) acts as the connective tissue between these two standards. By formalising the way system life cycle processes are modelled, documented, and analysed, MBSE allows organisations to implement ISO 15288-compliant engineering activities in a way that simultaneously supports AS9100D quality requirements. For instance, MBSE can be used to ensure that each requirement has a clear source (traceability), that design decisions are captured and justified (configuration control), and that verification activities are directly linked to model artefacts (validation evidence).

More importantly, MBSE provides value that extends beyond the tactical concerns of daily design engineering. It enables organisations to create a digital backbone—a shared system model—that links together strategy, operations, design, and compliance. This central model becomes a single source of truth, allowing for more effective program management, streamlined audits, faster onboarding of new team members, and easier certification readiness.

By integrating ISO 15288 process rigour into MBSE workflows, and aligning those workflows with AS9100D expectations, aerospace organisations can reduce quality-related risks while improving agility and cross-functional communication. Design engineers benefit from this infrastructure by having better context and fewer late-stage surprises, but the true impact lies at the organisational level—where traceable, validated, and compliant engineering outcomes become the norm rather than the exception.

Practical QA Considerations for Design Engineers

The integration of quality assurance into the daily responsibilities of the design engineer requires not only procedural awareness but also the thoughtful use of tools and practices that bridge modelling, documentation, and compliance. This section outlines key considerations and enablers, with emphasis on real-world toolchains such as Capella, ReqIF, and Simulink, along with the digitisation of traditional QA practices to foster traceability, validation, and design assurance.

Requirements Management

Requirements management is foundational to both quality assurance and systems engineering. According to ISO/IEC 15288, Technical Processes such as *Requirements Definition* and *Requirements Analysis* emphasise the transformation of stakeholder needs into actionable and verifiable system requirements (Clauses 6.4.2 and 6.4.3). Simultaneously, AS9100D Clause 8.3.3 mandates the determination of design and development inputs, ensuring that functional, performance, regulatory, and interface requirements are explicitly defined and traceable throughout the development lifecycle.

In the context of Model-Based Systems Engineering (MBSE), textual requirements are no longer isolated in static documents; instead, they can be formally modelled, categorised, and linked directly to architecture components within a system modelling environment. **Capella**, an open-source modelling tool based on the Arcadia method, supports the structured modelling of requirements alongside functional, logical, and physical system architectures. Capella allows engineers to create or import textual requirements and connect them to model elements such as functions, interfaces, scenarios, and components.

To enable full traceability—particularly *directional traceability*—the use of the **Requirements Interchange Format (ReqIF)** is essential. ReqIF provides a standardised method for exchanging requirements between tools such as IBM DOORS, Jama, and Capella, ensuring that upstream (e.g., customer) and downstream (e.g., test) requirements can be linked and updated consistently. In Capella, ReqIF integration enables the import of external requirement packages as structured data, with each requirement assigned a unique identifier and metadata attributes such as criticality, source, and verification method.

Directional traceability, which connects requirements from source to realisation (forward traceability) and from implementation back to origin (backward traceability), supports both ISO 15288's call for technical coherence and AS9100D's focus on validation and audit readiness (Clauses 8.3.4 and 8.3.5). For example, requirements traced from a Capella component to its physical realization and test procedure can be exported through ReqIF and linked to verification tools such as Simulink Test or external QA systems. This creates a closed-loop digital thread that is both auditable and dynamically maintainable.

Modelling for Quality

Modelling is not merely a method for system visualization—it is a strategic enabler for quality assurance when aligned with both ISO 15288 and AS9100D. Clause 6.4 and 6.5 of ISO/IEC 15288 emphasize the role of architecture definition and design definition processes in establishing technical coherence, risk reduction, and stakeholder satisfaction. In parallel, AS9100D Clause 8.3 requires organisations to plan and control the design and development process with documented inputs, outputs, reviews, verification, validation, and change control mechanisms.

Tools like **Capella**, built on the Arcadia methodology, enable engineers to implement these standards by providing structured, multi-viewpoint modelling. Capella supports functional, logical, and physical architecture layers, allowing traceability from operational concepts to implementable system elements—thus satisfying both the *design definition* requirement of ISO 15288 and the *design output verification* of AS9100D (Clause 8.3.5).

- **Architecture modelling in Capella** helps enforce quality gates by ensuring that each system element is derived from validated operational needs. For instance, logical components must be traceable to system capabilities, while physical components inherit constraints and interface definitions from their logical predecessors—ensuring a top-down validation path.
- **Modelling assumptions and interface definitions** can be explicitly documented using Capella's Property Values, Exchange Items, and Functional Chains. This supports Clause 8.3.3 of AS9100D by ensuring that design inputs—including constraints, applicable standards, and interface requirements—are complete, unambiguous, and systematically verified.
- Where interoperability with other engineering domains is required, Capella models can be exported via ReqIF or SysML transformations, enabling downstream simulation, control algorithm design (e.g., in Simulink), or test automation. This linkage satisfies ISO 15288's process integration guidance (Clause 6.2.3) and supports AS9100D's requirement for design review and validation traceability (Clause 8.3.4).

Verification and Validation Support

Verification and validation (V&V) are foundational components of both systems engineering and quality management. ISO/IEC 15288 explicitly defines *Verification Process* (Clause 6.4.11) and *Validation Process* (Clause 6.4.12) as distinct activities aimed at ensuring that a system meets its specified requirements and fullfill its intended use, respectively. Similarly, AS9100D reinforces the importance of these activities in Clause 8.3.4.2 (Design and Development Controls), Clause 8.4.2 (Control of External Providers), and Clause 8.5.1 (Production and Service Provision), by requiring evidence of conformance to requirements and effective implementation of validation mechanisms.

Model-Based Systems Engineering (MBSE) facilitates these goals by enabling V&V activities to be designed, executed, and documented directly within the modelling environment. Tools such as **Capella** or **SysML**-based platforms allow for the embedding of test scenarios, interface checks, and performance criteria directly into architectural models. For example, in Capella, engineers can define operational scenarios and system functions in the *Operational Analysis* and *System Analysis* phases, and then trace those down to the logical and physical components to identify verification points.

This structured model hierarchy enables automated traceability from requirements to verification activities, making it easier to demonstrate compliance with AS9100D's requirement for validated design outputs and verified performance under Clause 8.3.4.2. Moreover, linking V&V artefacts directly to model elements satisfies ISO 15288's emphasis on bi-directional traceability and objective evidence of implementation (Clause 6.3.4).

Using Capella as a practical example, engineers can:

- Define **testable requirements** using the *Requirement VP (Viewpoint)* and associate them with specific functions, interfaces, or components.
- Build **test scenarios** using the *Scenario VP*, simulating system behaviour under expected operational conditions.
- Generate **traceability matrices** that connect requirements, design elements, and verification procedures—thereby supporting both quality audits and engineering reviews.

These capabilities not only streamline verification planning but also provide structured, model-based evidence to support regulatory and customer audits. They reduce the risk of late-stage test failures, non-conformances, and undocumented behaviour by enabling validation earlier in the lifecycle, consistent with AS9100D's quality planning expectations (Clause 6.1) and ISO 15288's iterative process guidance.

Design Risk Management

Design risk management ensures that potential sources of failure, non-compliance, or underperformance are identified, analysed, and mitigated early in the design lifecycle. From a standards perspective, **ISO/IEC 15288** explicitly addresses risk management within the *Technical Management Processes* group, particularly under the **Risk Management Process** (Clause 6.3.2.5), which calls for systematic identification, analysis, treatment, and monitoring of risks. In parallel, **AS9100D** emphasises risk-based thinking across the quality management system, with explicit references in **Clause 6.1** (Actions to Address Risks and Opportunities) and **Clause 8.3.5** (Design and Development Outputs), which mandates that product and process risks be addressed in design outputs.

A practical approach to integrating these requirements into the design workflow is through the use of **Capella**. Within this structure, design risks can be embedded into the system model itself—for instance, by tagging specific functions or interfaces with risk-related attributes (e.g., safety criticality, redundancy requirement, or failure detection method). Custom viewpoints or extensions can also be added to enable FMEA (Failure Mode and Effects Analysis) integration or hazard identification directly within the model.

Similarly, in a **SysML-based** modelling environment, design engineers can represent risks using «risk» stereotypes, custom properties, or parametric constraint blocks that model uncertainty or tolerance ranges. These risks can be directly associated with system requirements, components, or interfaces—enabling early visualization and impact analysis when system changes occur. By maintaining traceability between risks, mitigation requirements, and verification plans, the model supports both the intent of ISO 15288 and the auditable rigour required by AS9100D.

Furthermore, MBSE supports **risk-informed trade-off analysis**, a key component of Clause 6.3.2.6 in ISO 15288. Using Capella or SysML, engineers can define alternative architectures and simulate operational performance under varying conditions. These insights can then feed into design justification reports or risk registers, forming a traceable narrative that supports quality and safety audits.

Design Effectiveness and Workflow Monitoring

Ensuring design effectiveness is not merely about achieving functional performance—it involves verifying that the design process itself is systematic, traceable, and capable of producing consistent, quality outcomes. Both **ISO/IEC 15288** and **AS9100D** emphasise the need for structured process monitoring and performance evaluation to support system maturity and compliance.

Clause 9.1 of **AS9100D** requires organisations to determine what needs to be monitored and measured, including the performance of processes, and to ensure that these evaluations contribute to continual improvement. Similarly, **ISO 15288** outlines process quality and performance assessment under its Technical Management Process Group—specifically within the *Measurement Process* and the *Technical Assessment Process*—which call for regular measurement of design and development effectiveness against planned outcomes, resources, and stakeholder needs.

In practice, model-based systems engineering tools such as **Capella** or **SysML-based environments** provide a robust framework for embedding workflow monitoring directly into the design process. These tools can define and enforce modelling milestones, maturity levels, and completeness metrics. For instance, in Capella, viewpoint completion (functional, logical, physical) can be tracked using automated rules and dashboards, helping teams ensure that models are not only created, but validated at each abstraction level.

Additionally, by integrating SysML with verification tools or Jira-like workflow systems, organizations can monitor adherence to development stages, track unresolved requirements, or visualize model maturity through dashboards. These integrations support Clause 8.3.4 (AS9100D), which requires that monitoring and control be applied to design and development processes to ensure that planned results are achieved.

Moreover, metrics such as model element completeness, requirement coverage, unresolved interface definitions, or traceability violations can be extracted from SysML models using plug-ins or external analysis tools. These KPIs act as quantifiable indicators of design health and process efficiency, aligning with **ISO 15288's Measurement Process** and supporting the kind of objective evidence needed during internal or regulatory audits.

Conclusion

As aerospace systems grow in complexity and scrutiny, the role of the design engineer has evolved far beyond generating technical solutions. Today, the design engineer is a key agent of quality, compliance, and system integrity. Yet, the adoption of frameworks like AS9100D and ISO 15288, along with tools such as Capella, SysML, and Simulink, is only as effective as the willingness of engineering teams and organisations to embed these practices into their culture and workflows.

This paper has outlined how model-based systems engineering (MBSE) can serve as both a bridge and a blueprint—linking rigorous engineering practice to quality assurance expectations in a transparent, traceable, and actionable way. By leveraging modelling tools to embed requirements, risks, verification criteria, and change histories directly into the design environment, engineers can reclaim quality as a proactive design asset rather than a retrospective control mechanism.

For the design engineer, this is not just a compliance burden—it is an opportunity to influence how engineering is done. By advocating for clear traceability, by participating in the digitisation of quality processes, and by modelling with the whole system lifecycle in mind, engineers can help shape workflows that are not only more efficient, but more reliable, auditable, and sustainable.

Equally important is the organisational mindset. Quality cannot be the responsibility of a single department or function; it must be shared, visible, and valued across disciplines. When systems engineering and quality assurance are treated as part of the same mission—and when modelling is embraced as a collaborative language, organisations can move from fragmented compliance to integrated excellence.