ENGINEERING FIELD GUIDE

Developing and Manufacturing Medical Devices

A practical guide for systems engineers

From intended purpose and system architecture to integrated verification, controlled manufacture and lifecycle assurance

General cross-market guidance | Edition 1.0 | August 2026

Table of Contents

How to use this guide

This guide explains how systems engineers develop, integrate, verify and support medical devices across the complete product lifecycle. It is intended for engineers coordinating mechanical design, electronics, embedded software, applications, cloud services, consumables, clinical workflows, usability, risk management and production. It describes how to connect intended purpose and user needs to architecture, requirements, interfaces, integration, objective evidence and controlled manufacture.

CORE MESSAGE The systems engineer owns coherence across the whole medical device. A collection of individually acceptable components does not demonstrate that the integrated product safely fulfils its intended medical purpose under representative conditions.

Scope and important limitations

- Establish the manufacturer's intended purpose, target users, patient population, operating environment, device boundary, target markets and applicable regulatory classification.
- Apply product-family requirements only where relevant: IEC 60601 essential performance and ISO 11608 primary functions are not interchangeable or universal concepts.
- Keep regulatory class, software safety class, clinical risk, cybersecurity risk, primary functions, critical characteristics and design outputs distinct.
- Systems engineering integrates specialist evidence but does not replace clinical evaluation, biological evaluation, usability engineering, risk management or regulatory assessment.
- Confirm the controlled edition, applicability and jurisdictional status of standards and guidance before relying on them for a specific product.

1. The systems-engineering role in medical-device development

Systems engineering provides the technical thread connecting clinical purpose, user needs, architecture, specialist design, manufacturing, verification and post-market operation. Its defining responsibility is to make interactions and assumptions visible

before they become patient-safety issues, integration failures or unsupported regulatory claims.

Understand the system rather than only its components

Systems responsibility	Practical engineering question
Clinical and user context	What medical outcome is intended, who relies on the device, and which conditions or decisions determine safe use?
Product definition	Which physical products, software, services, accessories, consumables and operational dependencies constitute the system?
Technical architecture	How are functions, performance budgets, interfaces, risk controls and responsibilities distributed?
Integrated evidence	Can the manufacturer show that the complete approved configuration meets requirements and intended use?
Lifecycle control	Can the product be manufactured, deployed, maintained, changed and monitored without losing control of safety or performance?

Maintain technical coherence across disciplines

- Translate stakeholder needs into measurable, allocated and verifiable system requirements.
- Define boundaries and interfaces across mechanics, electronics, software, connectivity, accessories, users and service processes.
- Ensure hazards and risk controls are assessed at system level and correctly allocated to implementing disciplines.
- Integrate design outputs and objective evidence into an approved, reproducible product configuration.
- Resolve gaps between what individual teams assume and what the finished device actually does.

2. Intended purpose, clinical context and system boundaries

The intended purpose describes the medical function for which the manufacturer supplies the device. It determines which claims require evidence, who the intended users are, which patients or specimens are relevant and how the system should behave within the intended clinical workflow.

Define the intended-use context precisely

- Describe the medical purpose, target condition, patient population, intended user and anticipated operating environment.
- Identify indications, contraindications, significant exclusions, clinical limitations and the expected role of professional judgement.
- Define whether the device diagnoses, monitors, delivers therapy, supports a clinical decision, processes an IVD specimen or performs another medical function.
- Capture home-use, professional-use, transport, storage, environmental and emergency scenarios where relevant.
- Identify reasonably foreseeable misuse and the situations in which the intended medical function is unavailable or unreliable.

Establish the regulated product boundary

A connected medical-device system may include a physical device, embedded firmware, mobile application, clinical algorithm, cloud service, accessory, disposable consumable and supporting interfaces. Define which elements are part of the regulated product, which are accessories, which are external systems and which provide essential support without being independently classified as devices. A supplier-hosted service does not automatically fall outside the controlled system merely because it is operated externally.

3. Regulatory framework and applicable lifecycle standards

Medical-device systems engineering operates within the manufacturer's quality system and the regulations of each intended market. The EU MDR and IVDR establish applicable general safety and performance requirements; the US Quality Management System Regulation became effective on 2 February 2026 and incorporates ISO 13485:2016 by reference together with additional FDA requirements.

Relate engineering practice to the controlled quality system

- Use ISO 13485 and the manufacturer's approved design-and-development processes to control inputs, outputs, reviews, verification, validation, transfer and changes.
- Apply ISO 14971 to identify hazards, analyse hazardous situations, allocate and verify risk controls, and review production and post-production information.
- Use ISO/IEC/IEEE 15288:2023 as a systems-lifecycle reference when appropriate; it complements but does not replace medical-device regulations or quality-system requirements.

- Select relevant software, usability, electrical safety, injection-system, biological evaluation, interoperability and cybersecurity references based on the actual product.
- Record the applicable edition, amendment, national adoption, recognition or harmonisation status, and jurisdiction-specific interpretation.

Keep classification and performance terminology distinct

Concept	What it means for systems engineering
Regulatory device class	Jurisdiction-specific product classification governing the applicable conformity-assessment or premarket pathway.
Software safety class	An IEC 62304 software-system classification determined using the applicable risk-based criteria.
Risk-control measure	A design, protective or information-based control identified within the product risk-management process.
Primary function	A product-family concept used for needle-based injection systems within the applicable ISO 11608 context.
Essential performance	An IEC 60601 concept applicable to relevant medical electrical equipment and particular-standard circumstances.
Critical characteristic	A characteristic that requires appropriate control because variation may affect safety, intended performance, compliance or quality.

4. Stakeholder needs and system-level design inputs

Stakeholder needs describe the outcomes required by patients, users, clinicians, manufacturing, servicing and the business. System requirements translate those needs into controlled, measurable statements against which the integrated device can be designed and verified.

Elicit needs from the complete stakeholder landscape

- Include patients, clinicians, laboratory personnel, carers, installers, service engineers and manufacturing operators where relevant.
- Capture clinical workflow, user competence, operating environment, maintenance, training, accessibility and product-disposal needs.
- Include regulatory obligations, market claims, interoperability expectations, privacy, security, availability and continuity requirements.

- Identify expected consumables, external systems, accessories, medicinal products or specimen characteristics.
- Resolve conflicting stakeholder expectations before they become contradictory design requirements.

Write measurable and testable system requirements

Weak statement	More useful system-level requirement
The system shall be reliable.	The approved system configuration shall meet its justified availability or functional reliability target under the defined operating, maintenance and environmental conditions.
The device shall be easy to use.	Representative intended users shall complete the defined critical tasks in the approved use environment without unacceptable use-related risk.
The result shall be accurate.	The system shall satisfy the justified accuracy, repeatability and acceptance criteria for the intended patient, specimen or operating population.
The system shall be secure.	The system shall enforce specified identity, access, integrity, confidentiality, update and recovery controls appropriate to its security and safety risks.

Distinguish requirements from solutions and targets

A requirement states what outcome must be achieved and the conditions under which it is judged. Avoid prematurely prescribing an implementation unless the design constraint is justified. Keep engineering ambitions, contractual targets, regulatory requirements, formal design inputs, risk-control requirements and production release criteria identifiable and separately controlled.

5. Functional decomposition and system architecture

Functional decomposition describes what the product must do before fixing the implementation. A system architecture then allocates those functions to elements such as mechanics, electronics, firmware, applications, cloud services, users and manufacturing controls.

Decompose the clinical function into system behaviour

- Identify the information, energy, material or physical interaction entering the system.

- Describe sensing, processing, decision-making, actuation, therapy delivery, indication, data storage and external communication as relevant.
- Define operating states, state transitions, enabling conditions, inhibition conditions and expected user interactions.
- Identify safety, security, availability and protective functions rather than treating them as generic non-functional add-ons.
- Show how functions combine to deliver the claimed medical purpose and which dependencies can interrupt or distort that outcome.

Choose architecture with explicit trade-offs

Architecture decision	Typical trade-off
Local versus remote processing	Latency, availability, privacy, cybersecurity, clinical continuity and dependence on connectivity.
Mechanical versus software control	Failure observability, independence, complexity, response time, verification and manufacturing variation.
Reusable versus disposable element	Cleaning, reprocessing, cost, supply continuity, biological safety and user handling.
Integrated versus modular product	Interface complexity, change flexibility, supplier control, installation and verification burden.
Automated versus user-driven action	Usability, human oversight, error prevention, workflow and risk of over-reliance.

Document architecture views appropriate to the audience

Use complementary views to explain functional decomposition, physical components, software deployment, data flows, trust boundaries, clinical workflow, external interfaces and manufacturing configuration. A single generic block diagram rarely communicates enough about safety, interoperability, operational dependencies and lifecycle ownership.

6. Requirement allocation, decomposition and performance budgets

Allocation translates system-level expectations into requirements for implementing elements. The systems engineer remains responsible for showing that the collection of allocated requirements is complete, compatible and capable of satisfying the original integrated requirement.

Allocate functions and risk controls deliberately

- Assign each system requirement to one or more mechanical, electronic, firmware, application, cloud, user-interface or process elements.
- Record joint ownership where the outcome depends on multiple disciplines rather than pretending a single component owns the whole function.
- Allocate risk controls to the element that implements them and specify how their effectiveness will be verified.
- Define assumptions, dependencies and interfaces needed for allocated requirements to work together.
- Preserve backward traceability to user needs, regulatory obligations and product risk analysis.

Manage end-to-end performance budgets

System budget	Possible allocated contributors
Measurement accuracy	Sensor error, sampling, calibration, environmental sensitivity, algorithmic processing and display resolution.
Therapy delivery	Mechanical tolerance, actuator output, control timing, fluid resistance, consumable variation and user positioning.
Response time	Sensing, processing, network transit, cloud execution, user acknowledgement and actuator reaction.
Battery life	Quiescent load, communications, sensing duty cycle, user interaction, temperature and ageing.
Availability	Device uptime, network access, backend services, planned maintenance, recovery time and service dependencies.

Verify decomposition rather than assuming it is correct

A mathematically neat allocation can still fail if component errors are correlated, environmental assumptions differ, safety functions share a common dependency or interface timing is unrealistic. Review the integrated budget against worst-case and representative conditions, component capability, operational context and the actual approved configuration.

7. Interface definition, ownership and interoperability

Interfaces are frequent sources of medical-device failures because each contributing discipline can believe that another team owns the behaviour. Systems engineering should define physical, electrical, software, data, workflow and organisational interfaces with explicit ownership and acceptance criteria.

Maintain an interface inventory and control definitions

- Identify connectors, mating geometries, electrical signals, power, network links, messages, file formats and external application programming interfaces.
- Define units, ranges, timing, state dependencies, error handling, identity, version compatibility and intended semantics.
- Capture user handoffs, installation steps, consumable compatibility, servicing activities and supplier operational responsibilities.
- Identify trust boundaries, authentication, authorisation, data protection and monitoring for connected interfaces.
- Specify interface owners, change control, verification evidence and the consequences of loss, corruption or incompatibility.

Assess interoperability as a system safety issue

Interoperability requires more than successfully exchanging a message. The receiving system must correctly interpret the data, associate it with the intended patient or specimen, recognise units and timestamps, and handle missing or conflicting information safely. FDA interoperability guidance emphasises design considerations, interface information and appropriate submission or labelling evidence for relevant interoperable medical devices.

Prevent unsafe assumptions at discipline boundaries

A sensor may report a position that does not represent the actual mechanical state; a cloud acknowledgement may arrive after the clinical decision point; a disposable cartridge may satisfy dimensions while altering fluid resistance. Record and challenge such interface assumptions before integrated verification begins.

8. System-level risk management and hazard analysis

Medical-device risk arises from sequences of events involving users, device behaviour, environment and clinical context. Systems engineering should expose interactions that

component-level analyses can miss and ensure that risk controls are allocated, implemented and verified coherently.

Identify hazards across the complete operating scenario

- Consider incorrect therapy, delayed treatment, misleading information, lost results, contamination, physical injury, electrical hazards and use-related errors.
- Assess component failure, integration faults, corrupted data, unavailable services, incorrect configuration, cybersecurity compromise and manufacturing variation.
- Include foreseeable misuse, installation errors, maintenance, consumable substitution, ageing and transport conditions.
- Consider common-cause failure, shared resources, cascading faults and apparently independent controls that depend on the same element.
- Relate every credible sequence of events to the relevant hazardous situation, potential harm and applicable risk acceptance approach.

Allocate controls and verify their combined effectiveness

Example initiating event	System-level concern	Potential control allocation
Sensor indicates completion incorrectly.	The user believes therapy finished although the physical function is incomplete.	Independent state confirmation, mechanism design, firmware plausibility checks and user-interface feedback.
Patient identity is mismatched.	A valid result is associated with the wrong patient or specimen.	Identity checks, interface constraints, workflow confirmation and traceable integration testing.
Network connectivity is interrupted.	A time-critical function or result becomes unavailable.	Local fallback, user notification, recovery handling and clearly defined operating limitations.
Disposable variation changes delivery.	The integrated product fails despite each supplier component passing isolated inspection.	Interface specification, supplier control, tolerance analysis and representative system verification.

Avoid reducing risk management to component FMEA

Design and process FMEA are valuable, but they do not automatically capture clinical workflow, human factors, security threats, network dependencies or patient-level harm.

Use complementary analysis techniques where necessary and maintain a direct trace from hazard to control, requirement, design implementation, verification evidence and residual-risk evaluation.

9. Safety concepts, primary functions and essential performance

Important performance terminology depends on the product and standard involved. Systems engineers should distinguish general safety-related functions from the specific concepts defined in product-family standards and avoid applying familiar labels outside their proper scope.

Use product-family terminology only when applicable

Term	Appropriate context	Systems-engineering responsibility
Safety-related function	General product architecture and ISO 14971 risk-control discussions.	Define required behaviour, implementation, independence, failure response and verification.
Primary function	Needle-based injection systems within the applicable ISO 11608 framework.	Identify relevant delivery or automated functions and allocate the outputs and controls needed to achieve them.
Essential performance	Medical electrical equipment within the applicable IEC 60601 framework.	Identify relevant performance whose degradation could create unacceptable risk and evaluate normal and fault conditions.
Critical characteristic	Product or process control where variation affects safety, performance, compliance or quality.	Specify the characteristic, acceptance limit, manufacturing control and verification or inspection method.

Define safe states, fault responses and independence

- Identify what safe or acceptably controlled behaviour means for each clinically relevant operating state.
- Specify detection, response time, annunciation, lockout, fallback, recovery and user intervention where appropriate.
- Assess whether risk controls depend on the same sensor, power rail, software element, cloud service or manufacturing assumption.

- Consider whether an apparently safe shutdown could itself create risk by interrupting therapy or hiding a result.
- Verify the integrated fault response under justified normal, abnormal and reasonably foreseeable conditions.

10. Operating states, use scenarios and clinical workflows

A medical-device system does not exist in a single steady-state condition. Storage, preparation, installation, connection, calibration, active use, cleaning, maintenance, update and disposal may each expose different hazards and performance requirements.

Define state-dependent system behaviour

- Describe startup, self-test, ready, active, suspended, completed, fault, recovery and shutdown states where relevant.
- Specify entry conditions, permitted actions, inhibited actions, indications and transitions for each state.
- Account for users interrupting a sequence, removing accessories, replacing consumables or losing connectivity.
- Identify safety-critical transitions that require confirmation, physical interlocks or independent checks.
- Ensure the user interface, mechanism, firmware and cloud services represent a consistent system state.

Model representative clinical and operational scenarios

Use scenario descriptions and sequence views to examine patient or specimen preparation, normal operation, calibration, servicing, software update and incident recovery. Evaluate realistic timing, responsibility handoffs, external-system interactions and consequences when the expected sequence is interrupted.

11. Human factors and use-related risk integration

Usability is a system property created by the combination of physical design, information presentation, user workflow, environment and training. Systems engineers must connect the usability-engineering process with architecture, product risk management and verification planning.

Connect user tasks to system requirements and controls

- Identify intended users, competence, physical abilities, working environment and clinically relevant time pressure.
- Map critical tasks to the physical user interface, information displayed, required decisions and possible use errors.
- Allocate appropriate design controls across mechanics, electronics, software, labelling and training.
- Ensure prompts, alarms and confirmations are comprehensible and represent actual system behaviour.
- Coordinate representative formative and summative evaluation with the approved product configuration and intended use.

Avoid relying on warnings to repair poor architecture

Where practical, use inherently safer design, physical incompatibility, interlocks, automation, meaningful feedback and error-resistant workflow before relying on user vigilance or written instructions. IEC 62366-1 provides an applicable framework for evaluating usability as it relates to medical-device safety.

12. Electronics, mechanics and embedded-software integration

Electromechanical behaviour emerges from the interaction of physical components, electronics and firmware. Systems engineers should ensure that allocated requirements are compatible across tolerance, timing, environmental exposure, calibration and failure conditions.

Make cross-discipline assumptions explicit

Interaction	System-level engineering question
Mechanism and sensor	Does the sensor reliably observe the clinically important physical state across tolerances, ageing and misuse?
Power and operation	Does battery or supply behaviour support the claimed operating modes, alarms, updates and safe shutdown?
Actuator and control software	Are force, travel, timing, stall handling and fault detection consistent across real manufactured configurations?
Enclosure and electronics	Do thermal, ingress, shock, EMC-related layout and servicing assumptions remain valid at the integrated product level?

Interaction	System-level engineering question
Consumable and algorithm	Are material, geometry, calibration and lot variation compatible with processing or delivery assumptions?

Define integration acceptance before assembling the system

Specify the intended configuration, interface versions, assembly state, instrumentation, calibration, environmental conditions and expected results. Integration testing should expose interface and interaction defects progressively rather than waiting until the formal system-verification phase to discover basic incompatibilities.

13. Connected systems, data flows and interoperability

A connected medical device may depend on mobile software, wireless communication, cloud services, clinical information systems and remote operational support. Systems engineers should define how those dependencies influence medical performance, safety, availability and regulatory boundaries.

Describe data meaning, ownership and clinical consequences

- Identify data origin, patient or specimen association, units, timestamp, expected update frequency and responsible source.
- Define transformation, storage, synchronisation, retention, presentation and transfer to external systems.
- Evaluate delayed, duplicated, missing, corrupted, stale or conflicting information.
- Specify offline behaviour, reconciliation, recovery and user notification when connectivity is interrupted.
- Control version compatibility, external interface contracts and the impact of supplier changes.

Distinguish supporting infrastructure from the regulated function

Infrastructure may be commercially or operationally external while still being necessary for a regulated product to function safely. Document the responsibility model, service assumptions, operational monitoring, availability, data protection and evidence needed to demonstrate control over relevant dependencies.

14. Cybersecurity as a system safety and lifecycle concern

Cybersecurity should be considered as part of the system architecture rather than added as a late technical checklist. Security failures may alter data, interrupt availability, expose sensitive information or defeat controls relied upon for medical-device safety.

Build a security-aware system architecture

- Identify assets, entry points, trust boundaries, external dependencies, privileged functions and clinically relevant data flows.
- Specify identity, authentication, authorisation, confidentiality, integrity, availability, logging and recovery requirements.
- Protect data appropriately at rest, in transit and during processing where justified by the architecture and risk.
- Control secure updates, vulnerability handling, software components, supplier dependencies and supported product lifetime.
- Connect threat modelling and cybersecurity controls to product risk management, verification, release and post-market monitoring.

Keep security evidence aligned with current expectations

FDA's February 2026 cybersecurity guidance describes quality-management and premarket-submission considerations for relevant medical devices, including cyber-device obligations where applicable. The systems engineer should coordinate security architecture views, interface controls, risk evidence, software component inventories, verification and lifecycle support without assuming that a single penetration-test report demonstrates complete security.

15. Data integrity, timing and end-to-end performance

System performance often depends on information remaining correct, timely and associated with the right patient, specimen or device throughout the complete operating sequence. Component-level accuracy is not sufficient if data changes meaning, arrives late or is displayed in a misleading context.

Control clinically relevant data behaviour

- Define authoritative sources, identifiers, units, precision, validity periods and expected clinical meaning.

- Specify timestamp handling, synchronisation, sequence ordering, duplicate detection and reconciliation.
- Assess missing values, stale results, interpolation, rounding, unit conversion and default behaviours.
- Verify the complete chain from acquisition or entry through processing, communication, storage and presentation.
- Consider auditability, access control, integrity protection and recovery following service interruption.

Manage latency, availability and degraded operation

Specify how quickly the clinical function must respond and what happens when timing, network access or an external service falls outside its defined limits. A technically successful response that arrives too late for the intended clinical decision can still represent a system failure.

16. Configuration management, product variants and change control

A medical-device configuration includes more than a hardware revision or software release number. The approved system may depend on component versions, firmware, applications, cloud configuration, interfaces, consumables, calibration, manufacturing settings and instructions for use.

Define the complete controlled product baseline

Configuration element	Examples requiring system-level control
Physical product	Mechanical parts, electronics, materials, accessories, approved suppliers and manufacturing revisions.
Software and firmware	Embedded builds, applications, algorithms, operating environments, dependencies and released settings.
Operational environment	Cloud deployment, identity services, integration endpoints, monitoring and support arrangements.
Clinical and production definition	Intended use, requirements, risk controls, calibration, labelling, packaging and approved test methods.
Product variants	Market versions, language packs, consumables, optional features and compatible external systems.

Assess changes across the complete system

- Identify affected requirements, interfaces, performance budgets, risk controls, usability, biological evaluation, security and regulatory commitments.
- Determine whether existing verification or validation remains representative after the proposed change.
- Assess manufacturing tooling, suppliers, process validation, installation, fielded devices and service compatibility.
- Define regression, integration and system-level testing proportionate to the changed behaviour and risk.
- Approve and communicate the new configuration before release or deployment.

17. Integration strategy and incremental system assembly

System integration should be planned as a progressive technical activity rather than treated as the moment when all teams deliver their components. The integration sequence should reduce the most important uncertainties early and provide objective evidence that interfaces behave as expected.

Build a risk-based integration sequence

1. **Define the integration baseline.** Identify approved hardware, firmware, software, external services, accessories, consumables and required test tools.
2. **Prioritise critical interactions.** Integrate interfaces and functions that carry the greatest safety, performance, interoperability or schedule uncertainty first.
3. **Establish acceptance conditions.** Specify expected behaviour, instrumentation, environment, representative data, fault cases and objective pass criteria.
4. **Integrate incrementally.** Add controlled elements in an agreed order while recording configurations, observed defects, deviations and corrective actions.
5. **Confirm system readiness.** Verify that unresolved anomalies, interfaces, representative configurations and prerequisite evidence are understood before formal verification.

Differentiate integration testing from formal verification

Integration testing establishes confidence that components and interfaces work together and helps discover defects. Formal verification provides approved objective evidence against controlled requirements. Integration evidence may support verification when the

configuration, protocol, traceability and acceptance criteria are suitable, but the two activities should not be assumed equivalent automatically.

18. System verification planning and execution

System verification confirms that the approved integrated design fulfils specified system requirements. The verification strategy should be developed early enough to influence requirement wording, architecture, instrumentation, sample availability and evidence planning.

Create requirement-driven verification methods

- Assign an appropriate method such as test, inspection, analysis or justified review to every controlled system requirement.
- Define the relevant approved configuration, samples, environment, operating state, interfaces and conditioning.
- Specify preconditions, procedures, measured outputs, calibration, objective acceptance criteria and traceability.
- Include risk-control effectiveness, degraded operation, fault handling, interoperability, security and relevant environmental extremes.
- Record deviations, failures, investigation, retesting decisions and approved conclusions without reconstructing evidence retrospectively.

Verify representative configurations and worst-case conditions

Choose samples and configurations that represent the production design, relevant software baseline, accessories, consumables, deployment environment and operating conditions. Consider tolerance combinations, ageing, packaging, transport, reprocessing or sterilisation where they influence the verified claim. A passing laboratory prototype is insufficient when its design or operating environment differs materially from the released system.

Manage traceability and unresolved anomalies

Connect each verification result to the exact requirement, test protocol, configuration, executed evidence and approval decision. Assess unresolved anomalies for their clinical, safety, usability and security impact before claiming that the system is ready for release.

19. System validation, clinical context and intended use

Validation confirms that the finished device meets user needs and intended use in the relevant operating context. It is not simply a second execution of requirement-based verification, and successful component tests do not establish that the complete product is suitable for its claimed medical purpose.

Design validation around the real use scenario

- Use intended users, relevant patient or specimen characteristics, representative workflows and realistic operating environments.
- Evaluate the approved finished configuration, including accessories, consumables, user information and necessary external dependencies.
- Address critical tasks, clinical timing, installation, training assumptions and foreseeable interruptions.
- Coordinate usability, clinical evaluation, performance evaluation and product-specific evidence with the overall validation strategy.
- Define acceptance criteria that demonstrate the device is suitable for its specified intended purpose.

Coordinate verification, validation and clinical evidence

Evidence type	Question answered
Design verification	Does the approved design fulfil specified system and allocated requirements?
Design validation	Does the finished device meet user needs and the intended use in the relevant context?
Clinical or performance evaluation	Is the clinical or diagnostic claim supported using the applicable product and regulatory evidence framework?
Usability evaluation	Can intended users complete relevant tasks safely in the expected use environment?
Risk-control verification	Have identified safety measures been implemented and shown to achieve their intended effect?

20. Design transfer, manufacturing and production controls

The approved system design must be translated into a manufacturing, deployment and support arrangement that consistently produces conforming devices. Systems engineers help ensure that production outputs remain faithful to the architecture, requirements, risk controls and verified configuration.

Transfer the entire approved product definition

- Provide approved system specifications, architecture, interface definitions, drawings, bills of materials and software or firmware baselines.
- Identify critical characteristics, manufacturing assumptions, calibration, process settings and acceptance criteria.
- Define supplier responsibilities, component traceability, incoming inspection, assembly, packaging, labelling and release controls.
- Transfer operational deployment, provisioning, service, cybersecurity support and update responsibilities where relevant.
- Verify that the production or deployment process can reproduce the system configuration used to support formal evidence.

Understand process validation and supplier dependencies

Where process outputs cannot be fully verified by subsequent inspection, validation may be required according to the applicable quality system and product risk. Coordinate with manufacturing and quality teams on relevant welding, sealing, sterilisation, calibration, automated assembly, software deployment or configuration processes. Supplier certification does not remove the manufacturer's responsibility for relevant interfaces, purchased components or outsourced services.

21. Post-market operation, maintenance and lifecycle learning

System responsibility continues after product release. Complaints, production trends, service reports, security vulnerabilities, software updates and external-system changes can reveal assumptions that were incomplete during development.

Connect operational evidence to product risk and design control

- Monitor complaints, malfunction reports, service activity, usage patterns, integration failures and relevant production trends.

- Investigate whether observed failures expose a new hazard, invalidate a risk-control assumption or affect other product variants.
- Assess software, supplier, material, infrastructure, cybersecurity and compatibility changes before release.
- Coordinate corrective action, field communication, regulatory assessment and verification of the approved remedy.
- Preserve traceability from post-market information to risk management, configuration control and design decisions.

Plan support, retirement and continuity

Define maintenance responsibilities, supported environments, replacement parts, security-update duration, data migration, service continuity and end-of-support arrangements. A connected system can become unsafe or ineffective if an external service, application platform or consumable becomes unavailable before the supported product lifetime ends.

22. Practical systems-engineering reviews and recurring failures

Many costly integration and regulatory problems arise because no individual discipline owned the relationship between intended purpose, architecture, risk, interfaces and evidence. Systems engineering should identify those gaps while corrective action is still technically and commercially feasible.

Recognise common systems-level failure patterns

Failure pattern	More effective systems response
Requirements are copied from a previous product.	Confirm intended purpose, user population, environment, interfaces and product-specific regulatory applicability.
Every component passes but the system fails.	Review end-to-end scenarios, performance budgets, interface assumptions and representative configurations.
A safety control is owned by nobody.	Allocate the control, identify dependencies, specify effectiveness and trace it to objective verification.
The cloud or consumable is treated as outside the product.	Define the real regulated boundary, operational dependencies and the evidence necessary to support safe use.

Failure pattern	More effective systems response
Validation is confused with verification.	Separate fulfilment of specified requirements from evidence that the finished device meets intended use.
Manufacturing receives incomplete technical outputs.	Transfer the complete approved system definition, process controls, configuration and release criteria.

Questions to ask at every development milestone

- Concept: What clinical problem does the system address, who uses it, and where are the product and operational boundaries?
- Inputs: Are needs, performance, interfaces, use conditions and safety requirements measurable and traceable?
- Architecture: Are functions, risk controls, dependencies and performance budgets allocated coherently?
- Development: Do mechanical, electronic, software and operational assumptions remain compatible?
- Integration: Are representative configurations, interface ownership, fault cases and acceptance criteria controlled?
- Verification and validation: Does the evidence demonstrate both requirement fulfilment and suitability for intended use?
- Transfer and support: Can the system be manufactured, deployed, maintained and changed without losing control of safety or performance?

PRACTICAL RULE When a medical outcome depends on several disciplines, the system requirement, interface assumptions, risk controls, verification evidence and approved configuration must be managed together.

Appendix A. Systems-engineering deliverables by lifecycle phase

Lifecycle phase	Representative systems-engineering deliverables
Concept and feasibility	Intended purpose, system boundary, stakeholder map, operating context, preliminary architecture, principal hazards and feasibility evidence.
Requirements	User needs, system requirements, standards applicability, interface inventory, performance budgets and risk-control allocation.

Lifecycle phase	Representative systems-engineering deliverables
Architecture and design	Functional and physical architecture, deployment views, state models, data flows, interface definitions and approved design reviews.
Implementation and integration	Allocated requirements, configuration baselines, integration strategy, interface results, anomaly tracking and cross-discipline decisions.
Verification	Verification strategy, requirement-method mapping, approved protocols, representative configurations, executed results and traceability.
Validation	Intended-use scenarios, representative users, finished-device configuration, validation evidence and clinical or performance coordination.
Design transfer	Approved product definition, production and deployment baselines, critical characteristics, supplier controls and release criteria.
Operation and maintenance	Post-market information, service evidence, approved changes, cybersecurity support, risk updates and lifecycle continuity plans.

Appendix B. Commonly relevant standards and frameworks

This matrix is a starting point, not a universal applicability determination. Confirm the controlled edition, scope, amendment, jurisdiction, recognised or harmonised status and product-specific requirements before claiming conformance.

Topic	Common reference	Systems-engineering relevance
Systems lifecycle	ISO/IEC/IEEE 15288:2023	Stakeholder needs, requirements, architecture, integration, verification, validation, operation and retirement.
Quality management	ISO 13485:2016; US 21 CFR Part 820 QMSR	Controlled design, reviews, purchasing, production, verification, validation, transfer and changes.
Risk management	ISO 14971:2019; ISO/TR 24971:2020	System hazards, risk-control allocation, residual risk and production or post-production information.

Topic	Common reference	Systems-engineering relevance
Software lifecycle	IEC 62304, applicable edition	Software safety classification, architecture, development, maintenance and controlled integration.
Usability engineering	IEC 62366-1, applicable edition	User-interface engineering, critical tasks, use-related risk and representative usability evidence.
Medical electrical equipment	IEC 60601 series, where applicable	Basic safety, relevant essential performance and system interactions for applicable electrical equipment.
Injection systems	ISO 11608 series, where applicable	Needle-based injection-system primary functions, automated functions and product-specific performance.
Biological evaluation	ISO 10993-1:2025 and relevant parts	Finished-device biological safety and integration of material and patient-contact assumptions.
Health-software security	IEC 81001-5-1; FDA cybersecurity guidance	Secure lifecycle, security architecture, threat controls, software components and ongoing support.
Manufacturer information	ISO 20417:2026; ISO 15223-1, as applicable	Information supplied with the device, instructions, labels and appropriate symbols.
EU device legislation	Regulation (EU) 2017/745; Regulation (EU) 2017/746	Applicable general safety and performance requirements and technical documentation.
Interoperability	FDA interoperability guidance; relevant interface standards	Interface specifications, data meaning, external dependencies, safe interaction and labelling.

Appendix C. Glossary

Term	Meaning in this guide
Allocation	Assignment of a system requirement, function or risk control to one or more implementing system elements.
Architecture	The organised description of system elements, functions, interfaces, dependencies and the rationale for their arrangement.

Term	Meaning in this guide
Configuration baseline	The approved combination of physical design, software, settings, interfaces, consumables and operational environment.
Critical characteristic	A product or process characteristic requiring control because variation can affect safety, performance, compliance or quality.
Design input	A controlled requirement describing a characteristic necessary for intended use, safety or regulatory fulfilment.
Design output	The controlled information defining the approved design and its manufacturing, deployment or support requirements.
Design transfer	The controlled translation of the approved design into production, deployment and associated operational arrangements.
Essential performance	An IEC 60601 concept applicable to relevant medical electrical equipment; not a generic label for all important functions.
Functional decomposition	Breaking an intended system outcome into defined functions and allocating them to suitable implementing elements.
Interface	A physical, electrical, software, information, workflow or organisational boundary between interacting elements.
Interoperability	The ability of connected systems to exchange and correctly interpret information or functions within the intended context.
Intended purpose	The manufacturer’s defined medical function, users, target population, operating context and product limitations.
Performance budget	The controlled allocation of an overall performance limit across contributing system elements.
Primary function	A product-specific concept within the applicable ISO 11608 needle-based injection-system context.
Risk control	A measure used to reduce a medical-device risk through design, protective measures or safety information.
System boundary	The defined separation between the controlled medical-device system, accessories, external systems and supporting services.
System integration	The controlled combination of system elements and assessment of their interfaces and joint behaviour.
Traceability	The ability to connect intended use, needs, hazards, requirements, architecture, verification, validation, production and changes.

Term	Meaning in this guide
Validation	Confirmation that requirements for a specified intended use or application have been fulfilled.
Verification	Confirmation that specified requirements have been fulfilled.

Appendix D. Authoritative starting sources

- ISO — ISO/IEC/IEEE 15288:2023, System life cycle processes
- ISO — ISO 13485:2016, Medical devices quality management systems
- ISO — ISO 14971:2019, Application of risk management to medical devices
- ISO — IEC 62366-1, Application of usability engineering to medical devices
- ISO — ISO 10993-1:2025, Biological evaluation of medical devices
- ISO — ISO 11608-1:2022, Needle-based injection systems
- ISO — ISO 20417:2026, Information supplied by the manufacturer
- FDA — Quality Management System Regulation
- FDA — Quality Management System Regulation: Frequently Asked Questions
- FDA — Design Considerations for Interoperable Medical Devices
- FDA — Applying Human Factors and Usability Engineering to Medical Devices
- FDA — Cybersecurity in Medical Devices: Quality Management System Considerations
- European Union — Regulation (EU) 2017/745 on medical devices
- European Union — Regulation (EU) 2017/746 on in vitro diagnostic devices
- European Commission — Medical devices guidance and regulatory information

CONTROLLED USE Consult the current controlled editions of applicable legislation, standards and regulatory guidance. The specific product architecture, intended purpose and approved quality-system processes determine which references and evidence are appropriate.