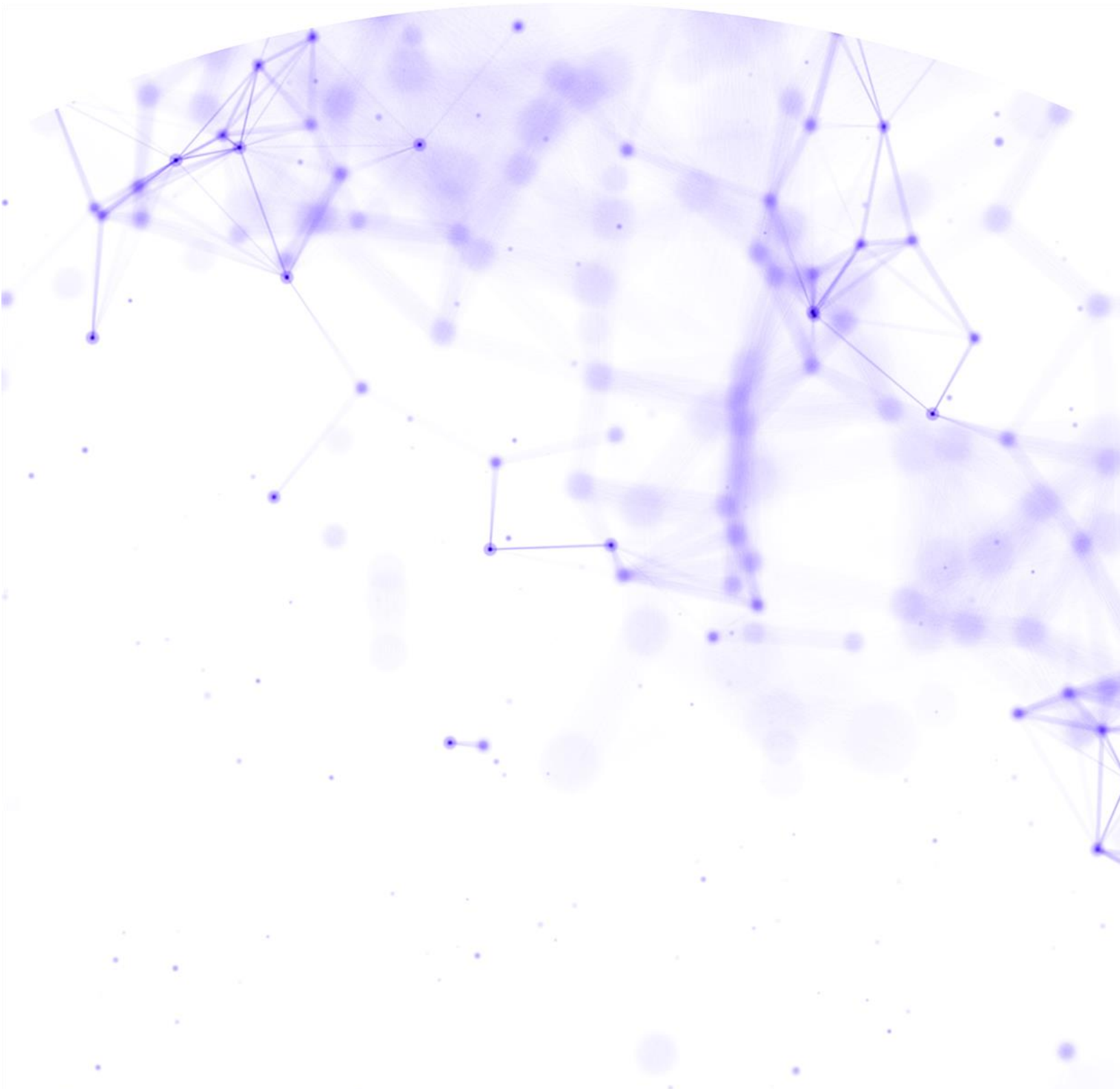
arm

Robotics Capability Framework

Non-Confidential

Copyright © 2026 Arm Limited (or its affiliates).
All rights reserved.

Issue 1.0
112293



Robotics Capability Framework

This document is Non-Confidential.

Copyright © 2026 Arm Limited or its affiliates. All rights reserved.

This document is protected by copyright and other intellectual property rights.
Arm only permits use of this document if you have reviewed and accepted Arm's Proprietary notice found at the end of this document.

The product revision is 1.0.

See also: Product and document information

Start reading

If you prefer, you can skip to the start of the content.

Intended audience

This document is intended for stakeholders who need a common reference for describing, comparing, specifying, assuring or evolving robotic system capabilities. This includes robot manufacturers, silicon and compute providers, software platform providers, system integrators, customers, procurers, regulators, insurers, safety and security practitioners, researchers, standards bodies, product teams, architects and technology strategists.

The document is particularly relevant to teams defining use cases, capability profiles, product roadmaps, architecture requirements, procurement language, deployment assumptions, assurance expectations or ecosystem collaboration around intelligent robotic systems. It is not intended to prescribe implementation architecture, mandate specific technologies or define a certification programme.

Inclusive language commitment

Arm values inclusive communities. Arm recognizes that we and our industry have used language that can be offensive. Arm strives to lead the industry and create change.

Feedback

Arm welcomes feedback on this framework and its documentation. To provide feedback on the Robotics Capability Framework or on this document, complete the form at https://www.arm.com/company/contact-us/robotics-capability-framework or email us at roboticsframework@arm.com.

Contents

1. Summary and context 4

2. The Robotics Capability Framework 5

2.1 Framework description and key principles 5

2.2 The Robotics Capability Framework levels of robotics intelligence 7

2.3 The Robotics Capability Framework Taxonomy 12

2.4 Capability layering: System 1, System 2 and System 3 23

2.5 How capabilities evolve across the Robotics Capability Framework levels 29

2.6 Alignment with the Robotics Capability Framework 41

3. Applying the Robotics Capability Framework 43

Appendix A Interpretation notes 48

Appendix B Example capability-profile matrix 51

B.1 Example profile C — Warehouse Logistics Assistant 54

Terms and definitions 58

Proprietary notice 59

Document information 60

Revision history 60

Copyright © 2026 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential
See Acknowledgements for Third-party Attributions

1. Summary and context

Robotic systems require a common reference for describing intelligence in physical systems. As robots differ by embodiment, task, environment, supervision model, operating context and deployment architecture, the same terms — such as autonomous, adaptive, contextual, cognitive or self-improving — can imply very different levels of capability, responsibility and assurance.

The Robotics Capability Framework provides a structured reference framework for describing this intelligence in a stable and implementation-agnostic way. It combines two complementary dimensions: a set of sophistication levels, describing how robotic systems progress from reactive behaviors toward deliberative, adaptive, contextual, cognitive and self-improving intelligence; and a capability taxonomy, describing the system capabilities required to realize those levels across robotics-specific, general-compute and ecosystem-facing domains.

The purpose of the Robotics Capability Framework is not to prescribe how robotic systems must be built. It does not define a product architecture, mandate a processor architecture, require a specific deployment model, or privilege any particular hardware, software, algorithm, AI model or toolchain. Instead, it defines a shared language for describing what a robotic system can do, how it behaves, what assumptions it depends on, what level of sophistication it demonstrates, and what expectations should apply.

The Robotics Capability Framework treats robotic intelligence as a system-level property arising from the combination of multiple capabilities. These capabilities include robotics-specific capabilities such as perception, localization and mapping, Physical Action, decision-making, learning and adaptation, and human–robot interaction; general-compute capabilities such as compute execution, determinism, lifecycle management, diagnostics, resilience and cybersecurity mechanisms; and ecosystem-facing capabilities such as interoperability and system integration. Safety and security are treated as cross-cutting properties and assurance expectations of these capabilities, rather than as standalone robotics-specific capabilities.

Because these capabilities may evolve at different rates, the Robotics Capability Framework represents intelligence as a capability profile across levels, rather than as a single monolithic label.

The framework is designed to remain stable even as technologies change. Hardware architectures, AI models, sensors, software stacks, middleware, safety mechanisms and deployment patterns will continue to evolve the Robotics Capability Framework remains anchored in capability, behavior and system properties, allowing different implementations and architectures to be compared without freezing today’s technologies into the framework.

In practice, the Robotics Capability Framework acts as a translation layer between market needs, system expectations and technology choices. It helps translate use cases, workloads, stakeholder expectations and assurance needs into solution-agnostic capability objectives. Those objectives can then guide architecture decisions, technology roadmaps, product planning, procurement language, deployment assurance and ecosystem collaboration.

In this way, the Robotics Capability Framework provides a future-proof reference model for describing where a robotic system sits, what it is expected to achieve, what assumptions govern its operation, and what capabilities must be present for higher levels of robotic intelligence to be realized safely, reliably and at scale.

Copyright © 2026 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential
See Acknowledgements for Third-party Attributions

2. The Robotics Capability Framework

2.1 Framework description and key principles

The Robotics Capability Framework is a robotics intelligence reference framework for describing the sophistication of robotic systems and the capabilities required to realize them. It maps robotic systems across six levels of sophistication, from RL0 Reactive to RL5 Self-improving, and combines those levels with a capability taxonomy that describes the system capabilities needed to achieve them.

The emergence of foundation models, agentic AI and embodied intelligence is moving robotics from deterministic automation toward probabilistic cognition, creating the need for a common reference that remains stable as implementation technologies evolve.

The framework provides a shared language for describing intelligent robotic systems across different embodiments, markets, operating environments and deployment models. It allows stakeholders to describe what a robotic system can do, how it behaves, what assumptions it depends on, what role humans play, what level of sophistication it demonstrates, and what expectations should apply as capability increases.

The Robotics Capability Framework treats intelligence as a system property that emerges from the combination of multiple capabilities. These include robotics-specific capabilities such as perception, localization and mapping, Physical Action, decision-making, learning and adaptation, and human–robot interaction; general-compute capabilities such as compute execution, determinism, timing, workload management, diagnostics, resilience, lifecycle management, updateability, rollback and cybersecurity mechanisms; and ecosystem-facing capabilities such as interoperability, integration, deployment support and assurance evidence. Safety and security expectations apply across the capability model. Functional safety is primarily treated as a dependency on the dependable compute and platform foundation required to execute those capabilities safely.

The framework is based on three key principles.

- **Future-proof.**

The Robotics Capability Framework is anchored in capabilities, behaviors, system properties, operating assumptions and assurance expectations that are intended to remain meaningful as robotics technologies evolve. It focuses on enduring characteristics of robotic systems rather than on specific technologies, product categories or market structures.

The framework therefore provides a stable reference against which new approaches can be understood. Foundation models, new control paradigms, novel sensing technologies, distributed intelligence, emerging compute platforms or future forms of robotic systems can change how capabilities are realized without requiring the underlying capability model to be redefined.

- **Implementation-and architecture agnostic.**

The Robotics Capability Framework describes what capabilities a robotic system must provide and the properties those capabilities must exhibit, not how they must be implemented.

Copyright © 2026 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential
See Acknowledgements for Third-party Attributions

It does not prescribe hardware, processor or instruction-set architecture, software frameworks, algorithms, AI models, middleware, toolchains, system partitioning, compute topology or product design choices. Nor does it prescribe whether intelligence executes on the robot, at the edge, on-premise, in the cloud or across a distributed robot–edge–cloud system.

The framework may describe behavioral and architectural roles where these are relevant to the properties of a capability, without prescribing their physical implementation. For example, System 1 captures fast, reflexive and safety-critical behavior; System 2 captures tactical, mission-bounded reasoning and adaptation; and System 3 captures strategic learning, training, shared knowledge and long-horizon improvement. These describe roles and behavioral properties within the system, not processors, software components, deployment locations or mandatory architectural partitions.

A System 1, System 2 or System 3 capability may therefore be realized through very different system architectures, depending on latency, safety, reliability, cost, power, scalability and deployment constraints.

This separation leaves vendors free to differentiate through architecture, implementation, performance, efficiency, cost, latency, safety design, user experience, lifecycle model and commercial proposition while remaining comparable at the capability level.

- **Embodiment-agnostic.**

The Robotics Capability Framework describes intelligence independently of the physical form through which it is expressed.

Capabilities are therefore defined in terms of their purpose, behavior and system properties rather than around a particular robot morphology, actuator arrangement or means of physical interaction. The same capability model should apply to a fixed industrial robot, autonomous mobile robot, drone, humanoid, quadruped, surgical robot, soft robot or robotic system whose physical configuration changes over time.

Different embodiments may realize the same capability in fundamentally different ways. Physical action, for example, may involve locomotion, manipulation, flight, rolling, grasping, pushing, deforming, positioning, force application or coordinated whole-body control depending on the embodiment. The framework describes the underlying capability and its sophistication without assuming which physical mechanism performs it.

Embodiment may influence the difficulty, performance envelope, constraints and assurance requirements associated with a capability, but it does not change the capability's fundamental place within the framework.

Together, these principles allow the Robotics Capability Framework to separate what a robotic system is capable of from how that capability is implemented and what physical form expresses it.

The framework is therefore intended to remain stable at the level of capability, behavior and system properties while allowing implementation technologies, system architectures, deployment models and robot embodiments to evolve independently.

By combining sophistication levels with a capability taxonomy, the Robotics Capability Framework also avoids reducing intelligence to a single label. A robotic system may demonstrate different levels of sophistication across different capability areas. For example, it may be advanced in perception but limited in learning, strong in physical action but constrained in human interaction, or highly integrated operationally while remaining dependent on bounded supervision.

Copyright © 2026 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential
See Acknowledgements for Third-party Attributions

The Robotics Capability Framework therefore describes robotic intelligence as a capability profile across levels, rather than as a one-dimensional maturity score.

- In practice, the Robotics Capability Framework acts as a translation layer:
- From use cases, workloads, operating assumptions and stakeholder expectations;
- Into implementation- and embodiment-agnostic capability objectives and system properties;
- Which can then guide architecture decisions, technology roadmaps, product planning, procurement language, assurance expectations and ecosystem collaboration.

This makes the framework useful for both engineering and business governance. For engineering teams, it provides a stable reference for analyzing capability requirements, workload intensity, safety expectations, compute needs, system partitioning, integration points and technology trade-offs. For strategy and business teams, it provides a way to reason about capability demand, differentiation, investment priorities, assurance needs and ecosystem friction.

2.2 The Robotics Capability Framework levels of robotics intelligence

The Robotics Capability Framework levels provide the horizontal dimension of the framework. They describe how robotic systems progress in sophistication, from systems that react to immediate stimuli to systems that can improve behavior over time under appropriate governance.

The levels are not intended to rank robots in a simplistic way, nor to imply that every system should aim for the highest level. A robot designed for a narrow, safety-critical or cost-sensitive task may be entirely appropriate at a lower Robotics Capability Framework level. The purpose of the levels is to describe the expected nature of the system's behavior, the degree of independence it can demonstrate, the role expected from humans, and the type of capability foundation required to support that behavior.

The Robotics Capability Framework therefore defines six levels: **RL0 Reactive**, **RL1 Deliberative**, **RL2 Adaptive**, **RL3 Contextual**, **RL4 Cognitive** and **RL5 Self-improving**. Each level represents a step change in how the system interprets inputs, selects actions, uses feedback, reasons over context, handles goals and improves over time.

The levels should be understood across three complementary perspectives:

System behavior describes how the robot behaves at that level, from reacting to stimuli to reasoning over goals and evolving behavior over time.

Typical output describes the kind of result expected from the system, from command execution to task execution, optimization, contextual action selection, justified planning and strategy refinement.

System capability describes the enabling capability foundation required at each level, from fixed rules with no memory or learning to rule-based logic, narrow generalization, context awareness, long-term memory, multi-goal reasoning and self-improving distributed intelligence.

Together, these perspectives prevent the levels from being interpreted as a single autonomy scale. They show that each level reflects a combination of behavior, output, human involvement and capability expectations. This is important because robotic intelligence is not only about reducing human control; it is about increasing the system's ability to act appropriately, safely and reliably within its intended operational context.

Copyright © 2026 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential
See Acknowledgements for Third-party Attributions

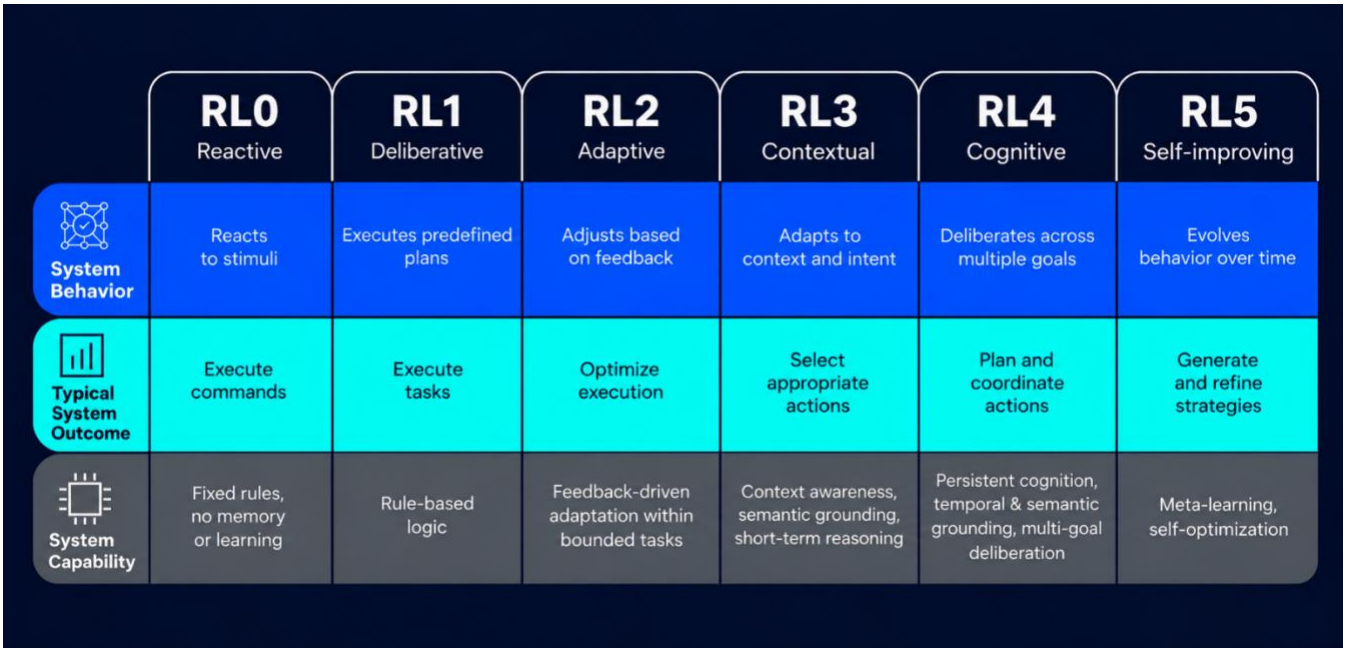


Figure 1. The Robotics Capability Framework levels of sophistication

The Robotics Capability Framework levels are therefore used as a reference for describing the sophistication expected from a robotic system or capability profile. They provide a common structure for comparing systems, framing requirements, aligning roadmaps, discussing assurance expectations and identifying what must change as systems move from reactive operation toward contextual, cognitive and self-improving behavior.

Copyright © 2026 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential
See Acknowledgements for Third-party Attributions

RL0 — Reactive

Overview

At **RL0**, the robotic system reacts to immediate stimuli or direct commands. Behavior is primarily driven by fixed rules, predefined responses or low-level control logic. The system has little or no memory, learning or interpretation beyond the current input.

RL0 should not be viewed as an immature system. It provides the deterministic execution foundation upon which all higher the Robotics Capability Framework levels depend, and remains the appropriate level for many safety-critical and tightly bounded tasks.

Characteristics

The system executes commands or responds to triggers. It does not reason about goals, context or alternative courses of action.

The capability foundation is based on **fixed rules, simple control logic, no meaningful memory and no learning**.

Use-case examples

Line-following robots; simple IR-activated vacuum robots; basic conveyor systems; fixed pick-and-place units; machines that stop, start or respond according to predefined sensor thresholds.

RL1 — Deliberative

Overview

At **RL1**, the robotic system can execute predefined plans, task sequences or workflows. Behavior is still bounded by explicit rules and programmed logic, but the system can follow a structured process rather than simply reacting to individual stimuli.

RL1 systems are useful where the task is known in advance, the environment is structured, and reliable execution of a predefined plan is the primary requirement.

Characteristics

The system executes tasks according to predefined plans, rules or command sets. It may follow a route, execute a programmed workflow, or complete a known task under expected conditions.

The capability foundation is based on **rule-based logic**, such as state machines, scripts, deterministic decision logic or symbolic planning structures. The system does not substantially adapt its behavior based on experience or changing context.

Use-case examples

AGVs operating in structured warehouses; rule-based robots with predefined command sets; household robots using trigger-action programming; mobile robots following fixed routes; industrial robots executing programmed task sequences.

Copyright © 2026 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential
See Acknowledgements for Third-party Attributions

RL2 — Adaptive

Overview

At RL2, the robotic system can adjust its execution based on feedback. It remains bounded by predefined objectives, operating assumptions and task structures, but it can modify how it performs a task in response to measured conditions or performance outcomes.

RL2 represents the transition from fixed execution to bounded adaptation. The system can improve or optimize execution within a known operating envelope, but it does not yet reason broadly about context, intent or competing goals. It is the last level where intelligence remains task-specific, deterministic and bounded, before the introduction of probabilistic foundation and generative AI models at RL3.

Characteristics

The system optimizes execution. It may tune parameters, compensate for variation, adapt motion, adjust perception thresholds, perform local replanning, refine timing or improve task performance while remaining within predefined objectives and operating limits.

The capability foundation is based on task-specific (narrow) AI using feedback-driven adaptation within bounded tasks. Unlike RL3, the system does not rely on foundation or generative AI models for perception, interaction or decision making, allowing behavior to remain deterministic, predictable and bounded within its intended operating envelope.

Use-case examples

Industrial robots adjusting grasping or motion parameters based on feedback; drones dynamically replanning routes around obstacles; autonomous mobile robots adapting local paths in changing warehouse layouts; assistant robots learning user-specific task preferences within bounded routines; robots improving execution through repeated experience while remaining within predefined task boundaries.

RL3 — Contextual

Overview

At **RL3**, the robotic system can adapt its behavior to context and intent. It does not merely optimize execution within a fixed task; it can interpret relevant situational information and select an appropriate action based on the current operating context.

RL3 is where robotic behavior begins to appear meaningfully contextual rather than merely adaptive. The system can operate with greater independence, but still within a defined capability and assurance envelope.

This is the first level where systems understand the meaning of their operating context rather than simply adapting execution. This indeed introduce new classes of system level challenges including for example AI safety, repeatability etc.. due to the non-deterministic behavior of AI models introduced at that level.

Characteristics

Copyright © 2026 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential
See Acknowledgements for Third-party Attributions

The system performs contextual action selection. It may choose between available behaviors, adapt to people, objects or environmental changes, respond to task intent, disambiguate user goals or modify its execution strategy based on short-term situational awareness.

The capability foundation combines context awareness, semantic grounding, episodic memory and short-horizon reasoning to enable contextual action selection.

Use-case examples

Urban delivery robots adjusting behavior around crowds; in-home assistants modulating interaction based on user emotion or intent; autonomous machinery adapting to environmental shifts; service robots selecting behaviors depending on user context; mobile manipulation systems adjusting task execution based on object state and surrounding activity.

RL4 — Cognitive

Overview

At **RL4**, the robotic system can reason over goals, constraints and trade-offs. It can consider multiple objectives, evaluate alternative actions, use longer-term memory and justify or explain why a course of action is appropriate within its operating context.

RL4 systems are not simply responding to context; they are reasoning about it. This introduces stronger expectations for explainability, consistency, assurance and trust.

This is the first level where systems reason persistently across multiple goals using world-model-guided cognition.

Characteristics

The system produces plans and justified actions. It may reason across tasks, negotiate conflicting objectives, anticipate consequences, use causal or semantic understanding, and plan hierarchically from task to skill to motion.

The capability foundation combines persistent cognition, world models, multi-goal deliberation and temporal reasoning to support hierarchical planning and explainable autonomous behavior.

Use-case examples

Co-working robots that schedule and coordinate with humans; embodied AI assistants reasoning across tasks; humanoid robots operating across multiple industrial or consumer interactions; mobile manipulators planning across dependent subtasks; robots explaining why they selected one action over another.

Copyright © 2026 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential
See Acknowledgements for Third-party Attributions

RL5 — Self-improving

Overview

At **RL5**, the robotic system can evolve its behavior over time under appropriate governance. It can refine strategies, improve performance, incorporate operational experience and contribute to shared learning across systems or contexts.

RL5 does not mean unconstrained autonomy or uncontrolled self-modification. It means that improvement over time becomes an explicit system capability, governed through validation, rollback, monitoring, assurance and human oversight.

This is the first level where system improvement itself becomes a governed capability.

Characteristics

The system generates and refines strategies. It may support meta-learning, self-optimization, fleet learning, distributed intelligence, long-horizon improvement or governed adaptation across deployments.

The capability foundation combines governed self-improvement, meta-learning, and continual policy evolution across systems and deployments.

Use-case examples

Drone swarms adapting mission logic over time; smart factories reconfiguring in real time based on operational learning; long-life space systems evolving behavior during extended missions; fleets of robots sharing learned strategies across deployments; general-purpose humanoid systems improving across a broad range of interactions under explicit governance.

2.3 The Robotics Capability Framework Taxonomy

The Robotics Capability Framework combines two complementary dimensions.

The first dimension is the Robotics Capability Framework **level**. This describes the expected sophistication of robotic intelligence, from **RL0 Reactive** to **RL5 Self-improving**.

The second dimension is the **capability model**. This describes the system abilities through which that sophistication is expressed.

The purpose of combining these two dimensions is to avoid treating robotic intelligence as a single label. A robot should not be described only as “adaptive”, “contextual” or “cognitive” without explaining which capabilities support that claim. Equally, a capability such as perception, motion or decision-making should not be described only as present or absent. What matters is the sophistication expected from that capability in the context of the system’s intended use.

The Robotics Capability Framework therefore uses the capability model as the vertical axis of the framework. The levels describe how sophisticated the system is expected to be; the capabilities describe where that sophistication is expressed.

A capability is a stable, solution-agnostic ability that a robotic system may need to possess. It describes what the system must be able to do, independently of how that ability is implemented. The capability set is intended to remain stable

Copyright © 2026 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential
See Acknowledgements for Third-party Attributions

across the Robotics Capability Framework levels. What changes across levels is not the existence of the capability, but the behavior, assurance expectation, autonomy horizon and sophistication required from it.

For example, **Perception** remains Perception across the framework. At lower levels, it may support immediate obstacle detection or simple trigger recognition. At higher levels, it may support contextual scene understanding, predictive awareness, behavioral interpretation or fleet-level perception improvement. The capability is stable; the sophistication changes.

The same principle applies across the framework. **Physical Action** may evolve from simple position control to adaptive trajectory execution, contextual manipulation, dexterous coordination or learned skill improvement. **Decision-Making** may evolve from rule-based action selection to tactical planning, contextual reasoning, goal trade-off analysis or governed strategy refinement. **Interaction** may evolve from simple signals and emergency triggers to contextual dialogue, intent recognition, personalized collaboration or long-term human–robot relationship modelling.

This is the core reason for blending the taxonomy and the Robotics Capability Framework levels into one framework. The taxonomy provides completeness and consistency. The Robotics Capability Framework levels provide progression and sophistication. Together, they create a structured way to describe robotic intelligence as a **capability profile across levels**, rather than as a single maturity badge.

This version of the Robotics Capability Framework defines the capability model at the level of **capability categories, capability definitions, capability objectives and level-specific expectations**. It does not standardize the internal component decomposition used to realize each capability. Component-level decomposition may vary by embodiment, product architecture, deployment model, safety case, technology stack and vendor implementation. Future profiles, implementation guides or conformance material may provide additional detail where needed, but the core framework remains focused on stable, solution-agnostic capability semantics.

Capability categories

The Robotics Capability Framework capability model is organized into three broad groups:

- **Robotics-specific capabilities**, which describe what the robotic system must do to operate intelligently in the physical world.
- **General-compute capabilities**, which describe the compute, platform, safety, security and lifecycle properties required to make robotics capabilities dependable, maintainable and deployable.
- **Ecosystem-facing capabilities**, which describe the properties required for robotic systems to integrate, interoperate, scale and remain stable across products, vendors, deployments and lifecycle phases.

This distinction is important because some capabilities, such as perception or motion, are primarily domain-specific robotics capabilities, while others, such as security, resilience, lifecycle management or interoperability, cut across the full system. The taxonomy explicitly separates robotics-domain capabilities from cross-cutting system capabilities so that the framework can describe both what the robot does and what makes those capabilities trustworthy, deployable, maintainable and scalable.

Copyright © 2026 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential
See Acknowledgements for Third-party Attributions

Robotics-specific capabilities

Robotics-specific capabilities describe the domain capabilities through which robotic intelligence is expressed in operation. They are the capabilities most directly associated with perceiving, understanding, moving, deciding, learning, interacting and operating in the physical world.

They are not alternative ways of classifying robots. They are the capability domains through which a Robotics Capability Framework level becomes meaningful. A declaration that a system is aligned with a given Robotics Capability Framework level should therefore be understood through the capability profile that supports that declaration.

Perception

Perception is foundational. Without some form of perception, higher-level planning, reasoning, adaptation and interaction cannot be grounded in the physical world. However, the required sophistication of Perception depends on the Robotics Capability Framework level and the use case. At lower levels, perception may be limited to simple triggers or obstacle detection. At higher levels, it may support semantic understanding, temporal continuity, predictive awareness, agent behavior interpretation and governed improvement of perception models over time.

Aspect	Description
Definition	Perception is the capability by which a robotic system transforms sensor data into actionable **observations** about the robot, the environment, humans, animals, objects, agents, events and relevant operating conditions.
Scope	Perception includes sensing, signal interpretation, feature abstraction, multimodal fusion, short-term observation continuity, uncertainty handling, behavior awareness and near-future scene prediction where required by the use case. It may use inputs from vision, distance sensing, tactile sensing, audio, inertial sensing, proprioception or other modalities.
Primary objective	The primary objective of Perception is to provide sufficiently timely, reliable and meaningful observations so that downstream capabilities — such as localization, motion, decision-making, interaction, learning and governance — can act appropriately within the intended operating context.
Boundary	Perception does not own the authoritative long-term world state, global spatial consistency or persistent map truth; those are primarily part of Localization & Mapping. The Robotics Capability Framework does not prescribe the sensors, models, algorithms, data formats, AI architecture or deployment location used to realize perception.

Copyright © 2026 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential
See Acknowledgements for Third-party Attributions

Localization & Mapping

Localization & Mapping is a foundational capability because robotic action depends on spatial grounding. At lower levels, it may support local position awareness or short-range pose estimation. At higher levels, it may support semantic spatial context, shared maps, persistent causal world models, environment evolution prediction and cross-agent spatial knowledge.

Aspect	Description
Definition	Localization & Mapping is the capability by which a robotic system determines its position, orientation and spatial relationship to its environment, and constructs, updates or consumes representations of that environment.
Scope	It includes local pose estimation, odometry, spatial reference frames, occupancy representations, map construction, semantic mapping, spatial consistency, map versioning, shared spatial representations and long-term world-model maintenance where required.
Primary objective	The primary objective of Localization & Mapping is to provide the spatial grounding required for navigation, motion, coordination, planning, interaction and operational assurance. It enables the robot not only to know where it is, but also to reason about the structure and evolution of its operating environment.
Boundary	Localization & Mapping consumes observations from Perception but is responsible for spatial consistency and persistent world-state representation. The Robotics Capability Framework does not prescribe whether localization is achieved through SLAM, visual-inertial methods, LiDAR, beacons, maps, GPS, semantic world models or any other implementation choice.

Physical Action

Physical Action is central to robotics because it is where intelligence becomes physical behavior. At lower levels, it may involve fixed actuation or deterministic task execution. At higher levels, it may involve adaptive trajectory adjustment, human-aware motion, predictive action selection, dexterous manipulation, multi-agent coordination or learned improvement of motion policies.

Aspect	Description
Definition	Physical Action is the capability by which a robotic system controls its body, actuators and effectors to move through space and physically interact with the environment.
Scope	It includes locomotion, navigation movement, arm and hand motion, grasping, placement, dexterous manipulation, posture control, stability, compliance, trajectory execution, physical interaction and coordinated multi-effector behavior where applicable.
Primary objective	The primary objective of Physical Action is to translate perception, spatial understanding, decisions and task intent into safe, effective and physically feasible action.

Copyright © 2026 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential
See Acknowledgements for Third-party Attributions

Boundary	Physical Action does not determine the overall mission objective or governance envelope; those are provided by Decision-Making and Governance Policy Control. The Robotics Capability Framework does not prescribe the control method, actuator architecture, planner, kinematic solver, learned policy, whole-body controller, hardware topology or execution location.

Decision-Making

Decision-Making must be understood separately from execution. The action selected by a decision capability may be realized through different workflows, controllers or behaviors. At lower levels, Decision-Making may be limited to fixed rules or predefined task logic. At higher levels, it may involve contextual reasoning, multi-goal trade-offs, causal inference, explainable planning, distributed coordination and governed policy evolution.

Aspect	Description
Definition	Decision-Making is the capability by which a robotic system selects, adapts and coordinates actions based on goals, constraints, context, environmental input, human input, policy and available resources.
Scope	It includes rule-based action selection, task sequencing, plan selection, tactical planning, contextual action selection, goal trade-off analysis, shared-control decisions, multi-agent coordination and strategic decision optimization where required.
Primary objective	The primary objective of Decision-Making is to determine what the system should do next, or how it should progress toward its goal, within the applicable operating envelope and available evidence.
Boundary	Decision-Making does not itself execute low-level physical control, own persistent spatial truth or define the full policy envelope. It consumes inputs from perception, localization, interaction, learning, governance and compute capabilities. The Robotics Capability Framework does not prescribe whether decision-making is implemented through rules, symbolic planning, optimization, AI models, agentic workflows, behavior trees, state machines, learned policies or hybrid approaches.

Learning & Adaptation

Learning & Adaptation becomes increasingly important as systems progress through the Robotics Capability Framework levels. At lower levels, it may be absent or limited to static configuration. At intermediate levels, it may support bounded tuning, task-specific learning and retention of task or context-specific experience. At higher levels, experience may be encoded into persistent memory and knowledge, retained and generalized across longer horizons, retrieved when relevant, and consumed by Perception, Decision Making, INTERACTION and other capabilities. At the highest levels, this may support continual, fleet-level and governed self-improvement.

Copyright © 2026 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential
See Acknowledgements for Third-party Attributions

Aspect	Description
Definition	Learning & Adaptation is the capability by which a robotic system improves, tunes or changes models, parameters, policies, behaviors or strategies through data, feedback, demonstration, simulation, operational experience or external updates.
Scope	It includes bounded parameter tuning, calibration, task-specific adaptation, user-guided correction, model improvement, policy refinement, transfer learning, continual learning, fleet learning and meta-learning. It also includes the acquisition, encoding, retention, consolidation, generalization and retrieval of memory and knowledge derived from experience, where these support future perception, reasoning, interaction or behavior.
Primary objective	The primary objective of Learning & Adaptation is to improve system performance, robustness, efficiency, personalization or generalization over time while remaining within controlled and auditable boundaries.
Boundary	Learning & Adaptation does not imply uncontrolled self-modification. Any adaptation that changes system behavior must remain subject to governance, validation, monitoring, rollback and applicable assurance expectations. The Robotics Capability Framework does not prescribe the learning method, training infrastructure, data pipeline, AI model, update mechanism or deployment architecture.

Interaction

Interaction evolves from simple signaling and fixed responses toward contextual, multimodal and collaborative behavior. At lower levels, it may involve buttons, indicators, emergency stops or predefined commands. At higher levels, it may involve intent recognition, dialogue, proactive clarification, personalized collaboration, social-norm alignment and trust calibration.

Aspect	Description
Definition	Interaction is the capability by which a robotic system communicates, collaborates and coordinates with humans, operators, other robots, agents or surrounding systems.
Scope	It includes signals, commands, feedback, alerts, human override, shared control, dialogue, gesture interpretation, intent recognition, clarification, collaboration, trust calibration, social cues and multi-agent negotiation where required.
Primary objective	The primary objective of Interaction is to enable the robotic system and relevant actors to exchange intent, state, instruction, feedback and authority in a way that supports safe, effective and understandable operation.
Boundary	Interaction does not own all perception of humans or all decision logic. It consumes perceptual and contextual inputs and provides interaction-relevant information to decision-making, motion, learning and governance capabilities. The Robotics Capability Framework does not prescribe the

Copyright © 2026 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential
See Acknowledgements for Third-party Attributions

	interface modality, user experience design, language model, dialogue system, communication protocol, display, haptic method or control device.

Safety & Trust governance

Safety & Trust Governance is critical because higher levels of robotic intelligence increase the need for bounded autonomy, explainability, evidence and controlled improvement. At lower levels, it may be expressed through fixed safety limits and emergency stops. At higher levels, it may support dynamic boundary enforcement, explainable safety runtime, decision provenance, continuous attestation, policy evolution and rollback across agents or fleets.

Aspect	Description
Definition	Safety & Trust Governance is the capability by which a robotic system defines, enforces and evolves the operating envelope within which autonomy is allowed to act.
Scope	It includes policy constraints, authority boundaries, runtime enforcement, safety arbitration, fallback rules, override conditions, evidence capture, auditability, decision provenance, policy lifecycle management, attestation, rollback and accountability mechanisms.
Primary objective	The primary objective of Safety & Trust Governance is to ensure that robotic autonomy remains bounded, accountable, traceable and aligned with applicable safety, security, operational, legal, ethical and organizational constraints.
Boundary	Safety & Trust Governance does not perform the task, generate motion or select every action itself. Instead, it constrains, authorizes, inhibits, records or modifies whether decisions, plans, updates or contextual influences may affect execution. The Robotics Capability Framework does not prescribe the policy language, enforcement mechanism, governance architecture, certification scheme, audit system or deployment location.

Copyright © 2026 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential
See Acknowledgements for Third-party Attributions

General-compute capabilities

General-compute capabilities describe the compute, platform and lifecycle properties required to make robotics capabilities dependable, maintainable and deployable. They are not unique to robotics, but they become increasingly important as robotic systems become more intelligent, connected, software-defined and lifecycle-managed.

These capabilities support the robotics-specific capabilities but are not themselves equivalent to robotic intelligence. They provide the execution foundation, assurance mechanisms and operational properties needed for robotic intelligence to run safely, reliably and at scale.

Aspect	Description
Definition	General-compute capabilities are the platform-level capabilities required to execute, isolate, observe, update, secure and manage the software, AI, data and control workloads that support robotic behavior.
Scope	They include compute execution, workload management, determinism, timing, memory, connectivity, monitoring, diagnostics, resilience, configuration, software lifecycle, updateability, rollback, orchestration, safety mechanisms and cybersecurity.
Primary objective	The primary objective of general-compute capabilities is to provide a dependable platform foundation for robotic capabilities across the full lifecycle of the system.
Boundary	General-compute capabilities do not define the domain-specific robotics behavior themselves. They enable, constrain and assure that behavior. The Robotics Capability Framework does not prescribe a processor architecture, operating system, middleware, accelerator, safety architecture, cybersecurity architecture or deployment topology.

Cybersecurity is treated primarily as a cross-cutting general-compute capability. It is concerned with protecting devices, data, models, communications, identities, update mechanisms and distributed robot-edge-cloud environments. Safety may appear both as a robotics-specific property, such as collision avoidance or safe separation, and as a platform-level property, such as diagnostics, isolation, fault detection, redundancy, watchdogs, timing guarantees and safe-state enforcement.

As the Robotics Capability Framework levels increase, general-compute capabilities become more demanding. Higher levels of sophistication typically require stronger compute performance, memory capacity, latency control, determinism, isolation, observability, update management, evidence generation, security, fault containment and lifecycle assurance.

Copyright © 2026 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential
See Acknowledgements for Third-party Attributions

Ecosystem-facing capabilities

Ecosystem-facing capabilities describe the properties required for robotic systems to integrate, interoperate, scale and remain stable across products, vendors, deployments and lifecycle phases.

Robotic systems rarely exist in isolation. They depend on sensors, actuators, middleware, operating systems, simulation tools, AI frameworks, data models, deployment pipelines, external services, safety processes, integration partners and operational environments. Ecosystem-facing capabilities ensure that these dependencies can be managed without locking the framework to one implementation or vendor architecture.

Interoperability

Aspect	Description
Definition	Interoperability is the capability of a robotic system to exchange data, commands, models, services or evidence across components, vendors, tools, platforms and deployment environments using stable and meaningful interfaces.
Scope	It includes device interfaces, data models, semantic contracts, middleware compatibility, service interfaces, tool integration, model exchange, versioning, compatibility handling and cross-platform portability.
Primary objective	The primary objective of Interoperability is to reduce integration friction and enable components, subsystems and services to work together across heterogeneous robotics ecosystems.
Boundary	Interoperability does not require all systems to use the same implementation or architecture. The Robotics Capability Framework does not prescribe a specific middleware, protocol, data schema, toolchain, interface standard or vendor ecosystem.

System Integration

Aspect	Description
Definition	System Integration is the capability to combine robotics-specific, general-compute and ecosystem-facing capabilities into a coherent system that can be designed, validated, deployed, maintained and operated.
Scope	It includes integration of hardware, software, sensors, actuators, compute, AI models, control loops, middleware, tools, safety mechanisms, security services, deployment infrastructure and operational processes.
Primary objective	The primary objective of System Integration is to ensure that the robotic system behaves coherently as a system, rather than as a collection of isolated capabilities.

Copyright © 2026 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential
See Acknowledgements for Third-party Attributions

Boundary	System Integration does not prescribe a single system architecture, supplier model, deployment topology or integration process. The Robotics Capability Framework defines the capabilities that must be considered, not the specific integration method used to realize them.

Stability

Aspect	**Description**
Definition	Stability is the capability of a robotic system and its ecosystem interfaces to remain dependable, compatible and predictable across change.
Scope	It includes interface stability, version management, lifecycle compatibility, regression control, behavior consistency, long-term support, update compatibility and operational continuity.
Primary objective	The primary objective of Stability is to allow robotic systems to evolve without breaking downstream consumers, deployed products, assurance evidence or operational expectations.
Boundary	Stability does not mean that systems cannot change. It means that change must be managed in a controlled, traceable and compatible way. The Robotics Capability Framework does not prescribe the release process, compatibility policy or lifecycle tooling.

Scalability

Aspect	**Description**
Definition	Scalability is the capability of a robotic system, platform or ecosystem to grow across robots, sites, fleets, workloads, markets or lifecycle phases without losing coherence, manageability or assurance.
Scope	It includes fleet scaling, workload scaling, deployment scaling, operational scaling, data scaling, multi-agent coordination, shared infrastructure, lifecycle management and ecosystem expansion.
Primary objective	The primary objective of Scalability is to support growth from individual robotic systems to larger deployments, product families, fleets or ecosystems while preserving performance, safety, security and operational control.
Boundary	Scalability does not prescribe whether growth is achieved through cloud, edge, on-robot compute, distributed architectures, centralized orchestration or hybrid models. The Robotics Capability Framework remains focused on the capability requirement rather than the deployment pattern.

Copyright © 2026 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential
See Acknowledgements for Third-party Attributions

Safety, security and assurance considerations

Safety and security are treated in the Robotics Capability Framework as cross-cutting properties and assurance expectations of capabilities, not as standalone robotics-specific capabilities.

Each robotics-specific capability should be understood as having safety and security expectations appropriate to its role, the Robotics Capability Framework level, operating context and risk profile. For example, Perception may need to express uncertainty, detect hazards and resist spoofed or corrupted inputs. Localization & Mapping may need to preserve spatial confidence, map integrity and safe degradation when localization confidence is low. Physical Action may need to maintain safe physical execution, force limits, separation, stability and fallback behavior. Decision-Making may need to enforce safe action selection and avoid unsafe plans or goal conflicts. Interaction may need to support override, consent, authority boundaries and understandable escalation. Learning & Adaptation may need validation, monitoring, rollback and protection against unsafe behavioral drift. Governance Policy Control may need auditability, accountability, policy enforcement and evidence capture.

In this sense, passive safety, active safety, SOTIF and cybersecurity are capability properties or assurance concerns that apply across the capability model. They describe expectations that should be considered when defining, assessing or declaring a capability at a given the Robotics Capability Framework level. They do not require the Robotics Capability Framework to define safety or security as separate robotics-specific capability categories.

Functional safety is treated differently. FuSa is primarily a dependency on the general-compute and platform foundation required to execute capabilities dependably. It concerns mechanisms such as diagnostics, fault detection, isolation, timing guarantees, redundancy, watchdogs, safe-state enforcement, safe degradation and evidence generation. These mechanisms may support every robotics-specific capability, but they are realized through the dependable compute and platform substrate rather than through the domain capability itself.

Cybersecurity also has a platform dimension. Security expectations apply across all capabilities, but many of the mechanisms that provide security — identity, authentication, authorization, isolation, secure boot, secure update, model integrity, data protection, communication security and protection of distributed robot-edge-cloud services — are part of the general-compute capability foundation. The Robotics Capability Framework therefore treats security both as a property expected of capabilities and as a platform capability required to make those properties enforceable.

This distinction allows the Robotics Capability Framework to keep safety and security visible at the capability level while avoiding a separate safety/security taxonomy that would duplicate existing standards or prescribe implementation architecture.

How the capability model works with the Robotics Capability Framework levels

The capability model should be read together with the Robotics Capability Framework levels.

A Robotics Capability Framework level is not achieved by the presence of a single feature. It is expressed through a coherent set of capabilities operating at an appropriate level of sophistication for a defined use case and operating context.

This means that a system may not be uniformly RL3, RL4 or RL5 across all capabilities. It may demonstrate contextual perception but only adaptive motion. It may support cognitive planning but bounded interaction. It may include fleet-level learning while retaining reactive safety control. The Robotics Capability Framework therefore supports a richer description: a system can be represented as a **capability profile**, showing how each relevant capability aligns to the sophistication expected by the target use case.

Copyright © 2026 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential
See Acknowledgements for Third-party Attributions

This avoids two common failure modes.

The first is **over-simplification**: reducing a complex robotic system to a single level without explaining what capabilities support the claim.

The second is **over-prescription**: treating the capability taxonomy as if every system must implement every capability in the same way.

The Robotics Capability Framework avoids both. The taxonomy defines the capability domains. The levels describe the expected sophistication. The use case determines which capabilities matter, how heavily they matter, and what level of sophistication is required.

A useful Robotics Capability Framework description should therefore state:

- the use case and operating context;
- the relevant capabilities;
- the assumptions, constraints and human role;
- the safety, security, governance and assurance expectations;
- the implementation choices only where they are needed for requirements, evidence or product differentiation.

In this way, the Robotics Capability Framework capability model provides the stable language through which robotic intelligence can be described, compared and evolved without prescribing how robotic systems must be built.

2.4 Capability layering: System 1, System 2 and System 3

Capabilities deployment model

A common but limiting approach is to define robotics architectures directly in terms of on-robot, edge, and cloud layers. While intuitive, this framing risks locking systems design too early into specific implementation choices and obscures the functional nature of the capabilities themselves.

Instead, this taxonomy treats on-robot, edge and cloud as deployment models, not architectural layers. Capabilities are defined independently of where they execute, allowing the same functional capability to migrate across deployment models as performance, cost, latency, safety, or scalability constraints evolve or dictated by the nature of specific use-cases and markets. This separation preserves architectural clarity while enabling flexible system realization.

Copyright © 2026 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential
See Acknowledgements for Third-party Attributions

Capabilities architecture layering

In robotics systems, architectural layering is essential to organize capabilities according to the role they play in system behavior. However, these layers should not be interpreted as physical partitions, product blocks, chips, subsystems, or deployment locations. Instead, they are architectural groupings based on common behavioral and operational properties, such as latency sensitivity, determinism, autonomy horizon, safety criticality, learning scope, and dependency on shared infrastructure.

For this reason, this taxonomy uses **System 1, System 2, and System 3** as capability-layering concepts rather than implementation boundaries. A capability assigned to System 1 shares reflex-like properties; a capability assigned to System 2 shares tactical autonomy properties; and a capability assigned to System 3 shares strategic, cross-context, or learning-oriented properties. These layers help describe the nature of a capability, not where it must execute or how it must be implemented.

This distinction is important because the same capability category may contain components that belong to different systems. For example, perception may include a System 1 obstacle detector, a System 2 semantic scene interpreter, and a System 3 fleet-level perception model improvement pipeline. These may run on the same chip, across multiple chips, on the robot, at the edge, or in the cloud. Deployment is therefore a separate design choice; the System 1/2/3 classification describes the capability's architectural role.

A more implementation-neutral framing comes from the concepts of System 1, System 2, and System 3, as shown in Figure 2. Widely adopted in cognitive science, System 1 represents fast, reflex intelligence, while System 2 reflects slower, tactical reasoning. In robotics, these describe complementary layers of intelligence and decision-making that can be mapped onto different system architectures and capabilities. To fully capture the collective and long-term dimensions of robotics intelligence, this taxonomy extends the model with a third layer, System 3,.

The three layers are defined as follows:

- **System 1 (Reflex Layer)**
  - **Nature**: Fast, automatic, stimulus-driven.
  - **Function in Robotics**: Provides immediate responses to sensory input, such as obstacle avoidance, reflexive safety stops, or stabilizing actions.
  - **Defining property**: deterministic, even hard real-time closed-loop control
  - **Characteristics**:
    - Extremely low latency and deterministic.
    - Relies on precomputed rules, policies, or lightweight learned models.
    - Essential for safety in dynamic environments.
  - **Deployment**: Predominantly on-robot, close to sensors and actuators.

Copyright © 2026 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential
See Acknowledgements for Third-party Attributions

- **System 2 (Tactical Autonomy Layer)**
  - **Nature**: Tactical, context-aware, and mission[1]-bounded autonomy -- slower, reasoning-based, and resource-intensive.
  - **Function in Robotics**: System 2 provides the tactical intelligence required to carry out an active mission safely and effectively. It reasons over the robot's current state, local environment, goals, and constraints to plan, adapt, and coordinate actions in real time or near real time. System 2 is responsible for deciding *how to act now* given the available information, including reacting to unexpected events, coordinating with nearby agents, and optimizing execution within bounded horizons
  - **Defining property:** mission-scoped decision-making that directly determines how the robot executes the current mission
  - **Characteristics**:
    - Higher computational demand than System 1.
    - Directly affects the execution of an active mission.
    - Operates over bounded spatial, temporal, and authority horizons.
    - Uses explicit reasoning, optimization, or learned policies.
    - Assumes local or provisional knowledge is sufficient to proceed.
    - May coordinate tactically with other robots when required by the mission.
    - Must be resilient to loss or delay of higher-level services.

    Deployment: Can be implemented on-robot, on a shared edge node, or as a distributed service, depending on latency, reliability, and mission requirements. System 2 capabilities may be shared across robots without implying full duplication, as long as they remain tightly coupled to mission execution

- **System 3 (strategic autonomy Layer)**
  - **Nature**: Strategic, cross-agent, and system-level autonomy.
  - **Defining property**: Determines under what conditions, priorities, and shared understanding tactical autonomy should operate
  - Function **in Robotics**: System 3 provides the strategic intelligence that shapes and optimizes robot autonomy beyond immediate tactical execution. It reasons over collective, historical, and contextual information across robots, missions, environments, and time horizons to determine **how autonomy itself should evolve and under what strategic envelope it should operate**.

    This includes **learning and training activities**, where learning refers to the generic mechanisms by which knowledge is acquired and retained, and training defines what domain-specific behaviors, models, or policies should be improved. Through these processes, System 3 updates shared representations, policies, constraints, skills, and priorities that influence future behavior and may also adjust the operating envelope of an ongoing mission.

[1] missions are the primary unit of operation which determines how to act in the moment given current goals, state, and constraints.

Copyright © 2026 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential
See Acknowledgements for Third-party Attributions

System 3 does not directly generate low-level actions, plans, or control signals. Instead, it produces strategic artefacts—such as learned models, trained policies, updated constraints, or shared knowledge—that are consumed by System 2 to make tactical decisions during mission execution.

- **Characteristics**:
  - Operates at a **strategic level of abstraction**, shaping how autonomy evolves rather than how actions are executed.
  - Reasons over **collective, historical, and cross-context information** (across robots, missions, environments, and time horizons).
  - Owns **learning and training activities**, where:
  - learning provides the mechanisms for acquiring and retaining knowledge, and
  - training defines what domain-specific behaviors, models, or policies should be improved.
  - Produces **strategic artefacts** such as learned models, trained policies, shared representations, constraints, priorities, or autonomy envelopes.
  - May influence an active mission by **adjusting constraints, priorities, or shared knowledge**, but does not generate immediate actions, plans, or control signals.
  - Is **not on the critical path for reflexive safety or tactical execution**; loss of System 3 degrades optimality and coordination, not basic operability.
  - Enables **consistency, optimization, adaptation, and alignment** at system, fleet, or ecosystem scale.
  - Supports progressive sophistication across the Robotics Capability Framework levels, from externally assisted optimization to fully self-improving systems.
- **Deployment**: System 3 capabilities are **deployment-agnostic** and may be realized in multiple forms depending on system sophistication, scale, and use-case constraints. They may be:
  - Embedded within a single robot (e.g., advanced or self-learning systems),
  - Distributed across a cohort of robots, where one or more agents assume strategic coordination roles,
  - Implemented on shared infrastructure (edge or cloud) to aggregate knowledge and optimize collective behavior.

  Deployment location does not define System 3. What defines System 3 is its **strategic intent**: shaping autonomy through learning, training, and cross-context reasoning rather than executing tactical actions. System 3 capabilities may be dynamically instantiated, migrated, or replicated without changing their architectural role.

Copyright © 2026 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential
See Acknowledgements for Third-party Attributions

How the layers support the combined framework

The System 1 / System 2 / System 3 layers make the combined the Robotics Capability Framework usable because they prevent the capability model from becoming a flat list.

They show that each capability may contain different architectural roles. For example:

- Perception may include System 1 hazard detection, System 2 semantic scene interpretation and System 3 perception-model improvement.
- Localization and Mapping may include System 1 local pose estimation, System 2 mission-scoped mapping and System 3 long-term world-model maintenance.
- Physical Action may include System 1 motor control, System 2 trajectory planning and System 3 skill learning.
- Decision-Making may include System 1 deterministic safety decisions, System 2 tactical action selection and System 3 policy optimization.
- Interaction may include System 1 emergency signals, System 2 contextual dialogue and System 3 personalized interaction strategy.
- Learning and Adaptation may include System 1 bounded parameter tuning, System 2 task-scoped adaptation and System 3 fleet-scale learning.
- Governance Policy Control may include runtime enforcement, tactical arbitration and strategic policy lifecycle management.

This structure allows the framework to describe how capabilities evolve without prescribing implementation. A Robotics Capability Framework level can therefore be interpreted as the sophistication expected across a capability profile, organized through System 1, System 2 and System 3 roles.

In this way, the combined Robotics Capability Framework can be read as a matrix:

- The Robotics Capability Framework **levels** describe the sophistication axis.
- **Capabilities** describe the system ability axis.
- **System 1 / System 2 / System 3** describe the capability-layering axis.

Figure 2 summarizes how the Robotics Capability Framework capability model and System 1 / System 2 / System 3 layers fit together. The outer capability categories describe the stable system abilities considered by the framework. The inner layers describe the behavioral role those capabilities play across reflexive, tactical and strategic autonomy. Market and workload inputs can then be interpreted through this structure to guide capability evolution across time horizons, supporting structured decision-making for technology roadmaps and product lines.

Copyright © 2026 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential
See Acknowledgements for Third-party Attributions

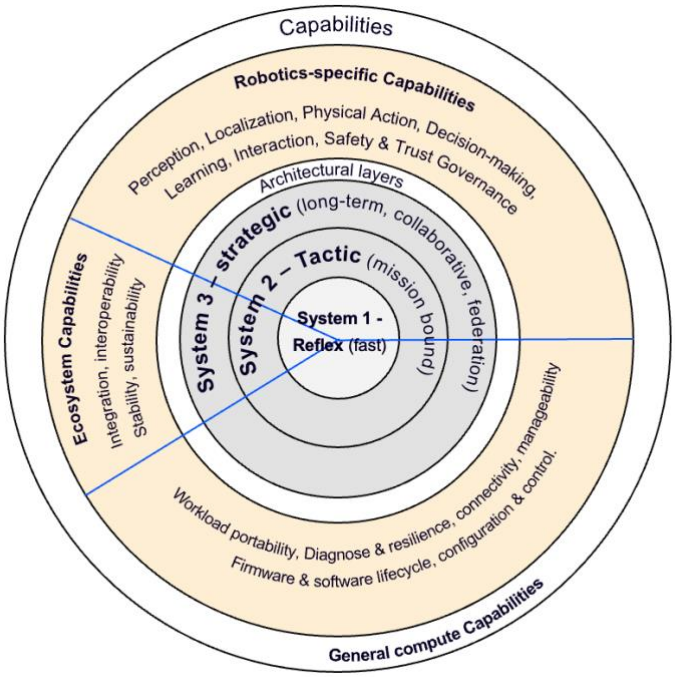


Figure 2. The Robotics Capability Framework capability model and System 1 / System 2 / System 3 layering

With the capability model and capability layers established, the next step is to describe how each capability evolves across the Robotics Capability Framework levels. This is where the framework brings the taxonomy and the sophistication levels together: each capability remains stable as a system ability, while its expected behavior, assurance needs and supporting system properties evolve from RL0 to RL5

Copyright © 2026 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential
See Acknowledgements for Third-party Attributions

2.5 How capabilities evolve across the Robotics Capability Framework levels

The Robotics Capability Framework brings the sophistication levels and the capability model together by describing how each capability is expected to evolve as the robotic system progresses from reactive behavior toward self-improving intelligence.

The levels do not replace the capabilities, and the capabilities do not replace the levels. They serve different purposes. The Robotics Capability Framework **levels** describe the expected sophistication of robotic intelligence. The **capability model** describes the system abilities through which that sophistication is expressed. This section explains how the two dimensions are read together.

A capability such as Perception, Physical Action, Decision-Making or Interaction is not simply present or absent. It can exist at different levels of sophistication. For example, perception may begin as simple object or scene triggering, evolve into multimodal sensing and bounded environmental adaptation, then progress toward semantic context fusion, predictive scene understanding and lifelong world-model refinement. Similarly, decision-making may begin as rule-based plan selection, then evolve into task-level planning, context-driven action selection, multi-goal reasoning and eventually cross-agent decision optimization.

This is the central idea of the combined Robotics Capability Framework: **robotic intelligence is the result of the coordinated combination of capabilities operating at different sophistication levels**.

Capability evolution, not capability replacement

The Robotics Capability Framework levels do not imply that lower-level capabilities disappear as a system becomes more sophisticated. Higher levels build on, coordinate and extend lower-level behaviors.

A robot operating at RL4 or RL5 still needs reactive safety mechanisms, deterministic control loops, local perception triggers, emergency stop behavior and bounded fallback modes. Sophisticated reasoning does not replace reflexive control. Instead, higher sophistication adds contextual awareness, longer-horizon reasoning, strategic learning and governance mechanisms above the reflexive and tactical foundations.

This means that capability evolution should be understood as cumulative. As a system progresses through the Robotics Capability Framework levels, capabilities expand in scope, context, memory, reasoning horizon, adaptability and governance expectations. The lower-level capability foundations remain necessary, especially where safety, timing and reliability are critical.

Reading the capability-level matrix

The capability-level matrix describes how each capability family evolves across the Robotics Capability Framework levels. Each row represents a capability family. Each column represents a Robotics Capability Framework level. Each cell describes the expected capability objective or behavioral expression for that capability at that level.

The purpose of the matrix is not to prescribe a specific implementation. It does not say which algorithms, processors, sensors, AI models, middleware or deployment architecture must be used. Instead, it describes the expected system behavior and capability sophistication in solution-agnostic terms.

Copyright © 2026 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential
See Acknowledgements for Third-party Attributions

Figure 3 illustrates the central matrix of the Robotics Capability Framework. The rows represent the capability model; the columns represent the Robotics Capability Framework levels of robotic intelligence. Each cell describes the expected sophistication of that capability at a given level. The purpose of the matrix is not to prescribe implementation choices, but to show how capability expectations evolve from rule-based behavior to contextual reasoning, cognitive planning and governed self-improvement.

Intelligence evolution	**Level 0 – Reactive**	**Level 1 – Deliberative**	**Level 2 – Adaptive**	**Level 3 – Contextual**	**Level 4 – Cognitive**	**Level 5 – Self-Improving**
Decision-making	Direct stimulus-response logic, no planning	Rule-based logic, finite-state decisions	Task-level planning, supervised adaptation	Probabilistic reasoning with context, memory, intent and goal understanding.	Persistent multi-goal deliberation with bounded, world-model-guided cognition.	Meta-learning and self-improving reasoning, with validated adaptation of strategies and policies
Perception	Simple event or threshold detection	Simple vision/object triggers	Multimodal perception (vision, audio,), object detection & segmentation	Foundation-model based perception with semantic understanding	Semantic & temporally grounded multimodal perception, predictive scene understanding,	Lifelong refinement of perception and world models from accumulated experience
Localization & mapping	Minimal local position or state awareness where required	Basic positioning for predefined routes, zones or tasks. The spatial model is limited and mostly static.	Updates local or mission-relevant spatial representations to support adaptive navigation, rerouting or changing layouts	Semantic spatial context, short-term local world-model memory and temporal continuity	Persistent spatial memory and structured world models, including semantic and episodic spatial alignment for planning over persistent context	Shared, evolving or causal world models across agents, sites or fleets, including prediction of environmental evolution
Interaction	Direct commands, buttons and status indicators	Scripted INTERACTION	Task-aware INTERACTION	Generative multimodal interaction with intent understanding	Context/ intent-aware dialogue, intention modeling, explainable interaction	Interaction strategies adapt over time from experience, feedback and evolving user/context understanding
Learning and Adaptation	No learning; fixed configuration and behavior	Limited parameter turning or externally defined updates, little or no persistent learned knowledge	Task-specific adaptation, calibration and short-term retention of learned parameters or experience	Context-aware learning with episodic/task memory; experience is retained and retrieved to influence future behavior within bounded contexts	Persistent long-term memory and structured knowledge; experience is consolidated, generalized and made available across perception, reasoning and interaction	Continual and self-directed learning; knowledge evolves across tasks, domains, agents or fleets, with goverenced consolidation, transfer and self-improvement
Safety and trust Governance	Deterministic execution, emergency stop, hard safety limits	Basic physical safety (joints limits, E-stop,…)	Runtime monitoring and anomaly detection (watchdog, fault detection, …)	Bounded behavior with runtime monitoring, interrupt and recovery (“AI safety island”)	Governed /trusted execution with explainability, provenance and bounded recovery	Adaptive governance that constrains and validates self-modification while maintaining safety and trust boundaries
Physical Action	Direct actuator control, fixed execution	Rule based task execution	Adaptive execution with feedback	Compliant, context-aware execution	Goal-directed execution with predictive planning	Execution strategies improve over time through experience and validated self-optimization
Representative AI architectures	Reactive controllers, finite-state machines, deterministic control loops	Classical planning (A*, D*), SLAM	Learning-based AI Pipelines (ML, RL, Adaptive Control)	Foundation Model Systems (LLMs, VLMs + Retrieval)	Agentic AI Systems (World Models, VLAs, Planners)	Self-Improving AI Systems (Meta-learning, Digital Twins, AutoML)

Figure 3. The Robotics Capability Framework levels to Capabilities alignment

Copyright © 2026 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential
See Acknowledgements for Third-party Attributions

RL0 – Reactive Capability Behavior

RL0 represents the reactive foundation of the Robotics Capability Framework. The system responds immediately to stimuli or direct commands using fixed rules, predefined responses or low-level control logic. It has no meaningful memory, contextual understanding or learning, and behavior is entirely bounded by deterministic execution.

Capability domain	System 1 — Reflex	System 2 — Tactical	System 3 — Strategic
Perception	Detect immediate signals, thresholds, hazards or simple triggers and expose safety-relevant cues.	No mission-level interpretation; may pass through or minimally transform cues needed by a fixed response or directly commanded behavior.	No meaningful model improvement or perception learning. Models and configuration are externally defined.
Localization & Mapping	Maintain minimal local pose, ego-state or proximity awareness required for immediate motion and safety.	Little or no mission-level mapping; may consume a fixed reference or local state required by the commanded action.	No cross-mission spatial model, shared map evolution or world-knowledge improvement.
Physical Action	Execute direct actuation, fixed motion, stop/start behavior and deterministic low-level control.	No meaningful tactical motion planning; execution follows directly commanded or hard-coded movement.	No skill improvement, motion-strategy reuse or strategic physical-action learning.
Decision-Making	Select deterministic immediate actions from fixed triggers, direct commands or hard-coded rules.	No tactical reasoning; behavior follows the explicitly commanded or predefined response.	No optimization or evolution of decision logic from experience.
Interaction	Use buttons, alarms, simple signals, emergency stop and fixed commands.	No structured dialogue or contextual collaboration; may expose fixed prompts or status responses where required.	No personalization, interaction-strategy learning or long-term user/agent modelling.
Learning & Adaptation	No learning; parameters are fixed except for deterministic calibration required to	No task-scoped adaptation.	No learning infrastructure required for system behavior.

Copyright © 2026 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential
See Acknowledgements for Third-party Attributions

Capability domain	**System 1 — Reflex**	**System 2 — Tactical**	**System 3 — Strategic**
	maintain basic control stability.		
Governance Policy Control	Enforce fixed limits, hard safety constraints, emergency stop and immediate inhibition.	Static operating constraints only; little or no runtime policy interpretation.	Policies are externally authored and statically configured; no autonomous policy lifecycle, policy learning or policy evolution.

Copyright © 2026 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential
See Acknowledgements for Third-party Attributions

RL1 – Deliberative Capability Behavior

RL1 introduces **deliberative execution**. The system executes predefined plans and deterministic workflows within structured environments, but behavior remains rule-based, predictable and largely static.

Capability	System 1 — Reflex	System 2 — Tactical	System 3 — Strategic
Perception	Immediate detection and safety cues remain deterministic.	Recognizes predefined objects, states or scene conditions needed to execute known tasks and plans.	Perception models remain predefined; updates are externally engineered and deployed.
Localization & Mapping	Local pose supports safe execution and stabilization.	Provides basic positioning and spatial information for predefined routes, zones or task sequences. Maps are largely static.	Maps/configuration may be stored and managed, but there is no autonomous cross-mission evolution.
Physical Action	Deterministic control, stability and immediate physical safety.	Executes predefined paths, motion sequences, grasps and manipulation behaviors within known task structures.	Skill definitions remain externally authored, validated and deployed.
Decision-Making	Fixed rules handle immediate action selection and safety continuity.	Selects between predefined plans, rules, task sequences or state-machine transitions.	Decision rules and plans are externally authored, versioned and updated rather than learned from operation.
Learning	Fixed configuration or limited calibration.	No meaningful online adaptation; task knowledge is predefined.	Training or configuration may happen externally, but operational experience does not autonomously evolve competence.
Interaction	Immediate signaling, override and fixed interaction responses.	Scripted commands, predefined responses, prompts and supervisory control.	Interaction models remain statically defined; no persistent personalization.
Safety & Trust Governance	Fixed limits and immediate safety inhibition remain authoritative.	Applies predefined operating envelopes, permissions and rule-based constraints to known workflows.	Policies are externally defined, validated and deployed through conventional lifecycle processes.

Copyright © 2026 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential
See Acknowledgements for Third-party Attributions

RL2 – Adaptive Capability Behavior

RL2 introduces **bounded adaptation**. The system adjusts execution using runtime feedback while remaining within predefined task objectives and operating limits.

Capability	System 1 — Reflex	System 2 — Tactical	System 3 — Strategic
Perception	Adjust low-level sensing/calibration within bounded limits while preserving deterministic safety cues.	Adapts interpretation to known variation: object pose, lighting, sensor conditions or multimodal evidence within known categories.	Supports controlled improvement or deployment of task-specific perception models under external validation; generalization remains narrow.
Localization & Mapping	Maintains local ego-state and performs short-horizon corrections as conditions vary.	Updates spatial representations to support local rerouting, changing layouts and adaptive navigation.	Stored maps or spatial knowledge may be updated between missions, but cross-context world-model evolution remains limited.
Physical Action	Feedback control adapts force, stability, trajectory or actuation while preserving hard execution limits.	Adjusts trajectories, grasps, timing, paths or motion parameters according to bounded feedback and execution outcomes.	Task-specific motion skills may be trained/refined and redeployed, but strategic self-improvement is not autonomous.
Decision-Making	Immediate actions remain deterministic where required for safety and continuity.	Adjusts task execution and locally replans using feedback while remaining inside predefined objectives and task structure.	Decision rules or policies may be tuned from historical data, but goals, authority boundaries and operating envelopes remain externally defined.
Learning	Bounded parameter tuning and calibration preserving stability and predictability.	Task-scoped adaptation from feedback, demonstration or recent operational data.	Creates or updates task-specific models under controlled training/validation; generalization remains narrow.
Interaction	Immediate override, signaling and safety interaction remain deterministic.	Supports task-aware guidance, correction, confirmation, preference capture and bounded shared control.	May retain task/user-specific configuration or preferences, but no broad long-term interaction strategy evolution.
Safety &Trust Governance	Hard limits continue to constrain adaptation.	Runtime health checks, bounded safety responses and monitoring ensure adaptation stays inside authorized limits.	Controlled update, validation and rollback mechanisms govern modified models/policies.

Copyright © 2026 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential
See Acknowledgements for Third-party Attributions

RL3 – Contextual Capability Behavior

RL3 introduces **context-aware intelligence**. The system interprets semantic context, understands intent and selects appropriate actions through probabilistic, short-horizon reasoning. This is the first level where the system interprets context and intent across key capability domains, using short-term reasoning and contextual memory to select appropriate actions.

Capability	System 1 — Reflex	System 2 — Tactical	System 3 — Strategic
Perception	Immediate hazards and motion-relevant cues remain available independently of higher-level interpretation.	Fuses semantic, multimodal and temporal observations to interpret people, objects, events and the current situation.	Improves and governs perception models using accumulated experience relevant to recurring contexts.
Localization & Mapping	Maintains fast local pose, occupancy and spatial integrity for reflexive motion.	Maintains semantic spatial context and short-term world-model continuity; understands zones, objects and recent spatial change.	Preserves and evolves spatial/semantic knowledge across missions sufficiently to support recurring contextual operation.
Physical Action	Maintains compliant, stable and physically safe execution under dynamic disturbances.	Produces context-aware and human-aware movement, adapting physical execution to object state, people and environmental conditions.	Stores and reuses validated context-conditioned skills for recurring task contexts; no autonomous strategic skill evolution.
Decision-Making	Fixed immediate safety decisions can pre-empt contextual reasoning when required.	Selects actions based on current context, intent, short-horizon reasoning and goal disambiguation.	Uses historical/contextual information to improve tactical decision policies, without independently generating immediate plans.; broad generalization remains limited**.**
Learning	Local adaptation remains bounded and predictable.	Performs task-specific, context-sensitive adaptation using recent experience and bounded memory.	Aggregates experience across missions/contexts and improves models or policies that System 2 subsequently consumes.
Interaction	Immediate signals, overrides and safety-related interaction remain available.	Interprets multimodal intent; supports contextual dialogue, clarification, correction and shared control.	Learns preferences and interaction patterns across encounters and improves interaction strategies under governance.
Safety & Trust Governance	Immediate inhibition and hard limits remain authoritative.	Performs dynamic boundary enforcement, context-aware constraint handling, safety arbitration and immediate fallback.	Maintains, validates and deploys policies that define the contextual operating envelope and permissible autonomy.

Copyright © 2026 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential
See Acknowledgements for Third-party Attributions

RL4 – Cognitive Capability Behavior

RL4 introduces cognitive deliberation. The system reasons over goals, constraints, trade-offs and consequences using longer-horizon memory and structured world models, enabling explainable planning and coordinated behavior.

Capability	System 1 — Reflex	System 2 — Tactical	System 3 — Strategic
Perception	Supplies trustworthy, time-bounded physical cues, hazard information and confidence required for immediate execution.	Provides semantically grounded, predictive perception supporting reasoning about agents, objects, dynamics and consequences.	Maintains and improves richer perception representations or models from accumulated experience under validation; deployed updates support cognitive operation but do not imply autonomous self-improvement.
Localization & Mapping	Maintains trusted local ego-state and spatial safety independently of higher-level reasoning.	Reasons over structured semantic and persistent spatial context when planning actions and evaluating alternatives.	Maintains authoritative persistent spatial/semantic world knowledge and longer-term environment evolution used by tactical cognition.
Physical Action	Executes goal-driven commands through stable, compliant and safety-bounded physical control.	Selects and coordinates feasible physical actions based on goals, predictions, constraints and trade-offs; supports multi-effector behavior where applicable.	Maintains validated skill libraries and motion/manipulation strategies informed by experience; updates remain governed and externally validated.
Decision-Making	Provides immediate deterministic pre-emption and continuity mechanisms where required.	Performs multi-goal reasoning, causal inference, hierarchical planning, consequence evaluation and explainable action selection.	Maintains and refines decision models, policies and reasoning artefacts using historical or collective context under governance; System 2 consumes these artefacts for cognitive planning.
Learning	Applies validated parameters or adaptations locally without compromising stability, timing or predictable execution.	Uses learned models, memory or knowledge to support planning, cognitive arbitration and task execution within the active mission.	Supports longer-horizon learning, transfer, model improvement and knowledge creation under governance and validation, feeding cognitive operation.
Interaction	Maintains safe authority transfer, immediate	Supports intent-aware dialogue, negotiation,	Maintains longer-term user or interaction knowledge,

Copyright © 2026 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential
See Acknowledgements for Third-party Attributions

Capability	System 1 — Reflex	System 2 — Tactical	System 3 — Strategic
	signaling and human override.	explanation, proactive clarification and sophisticated collaboration.	personalization and validated collaboration strategies across interactions under governance.
Safety & Trust Governance	Hard safety constraints remain capable of overriding cognitive plans.	Performs explainable runtime policy arbitration, risk reasoning, dynamic constraint enforcement and attested execution.	Owns policy definition, validation, versioning, evidence, provenance and controlled deployment/rollback of governance artefacts.

RL5 – Self-Improving Capability Behavior

RL5 introduces governed self-improvement. The system can refine models, knowledge, strategies, policies or behaviors through operational experience, shared learning or cross-context optimization, while remaining subject to validation, monitoring, rollback, attestation and accountability. Improvement itself becomes a managed system capability rather than an offline engineering activity

.

Capability	System 1 — Reflex	System 2 — Tactical	System 3 — Strategic
Perception	Safely executes with validated updated perception models while retaining deterministic hazard awareness and fallback.	Uses validated shared or evolving perception representations to interpret novel or changing operating contexts during the current mission.	Performs governed continual improvement of perception models, shared representations and cross-agent perceptual knowledge, with validation and rollback.
Localization & Mapping	Maintains reliable local pose and spatial integrity even as higher-level world models evolve.	Uses validated shared, evolving or causal spatial models when planning and coordinating the current mission.	Maintains shared, evolving, persistent or causal world models across missions, agents, sites or fleets and predicts environmental evolution.
Physical Action	Executes validated skills or updated physical-action parameters while maintaining invariant hard safety envelopes.	Composes and coordinates validated learned skills within current mission goals, constraints and safety envelope.	Improves motion policies, skill repertoires and coordination strategies over time; validates, versions, deploys and rolls them back when required.
Decision-Making	Immediate deterministic safety and continuity	Applies validated refined policies or strategies to current goals, and	Refines decision policies, reasoning strategies and optimization models using accumulated experience; supports

Copyright © 2026 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential
See Acknowledgements for Third-party Attributions

Capability	System 1 — Reflex	System 2 — Tactical	System 3 — Strategic
	remain independent of strategy evolution.	may coordinate across agents within the active mission.	cross-agent optimization under governance.
Learning	Applies approved adaptations locally within safe, bounded runtime constraints, without compromising deterministic safety behavior.	Selects and uses validated learned artefacts or task adaptations appropriate to the current mission and context.	Supports lifelong learning, fleet learning, continual learning, transfer and meta-learning; governs validation, deployment, monitoring, rollback, provenance and accountability.
Interaction	Preserves immediate human override, authority boundaries and safety signaling.	Uses personalized/shared interaction knowledge for adaptive collaboration and negotiation.	Evolves interaction strategies through long-term personalization, trust calibration, social alignment and cross-deployment learning.
Safety & Trust Governance	Maintains invariant hard boundaries and immediate inhibition despite evolving behavior.	Enforces evolving policies and autonomy envelopes at runtime, arbitrating what learned behavior is permitted to influence execution.	Continuously defines, validates, versions, attests, deploys and rolls back policy; maintains provenance, auditability and accountability across agents/fleets.

Copyright © 2026 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential
See Acknowledgements for Third-party Attributions

General-compute capability evolution

The robotics-specific capabilities described above depend on general-compute capabilities. These capabilities provide the platform foundation required to execute, isolate, monitor, update, secure and manage robotic intelligence.

As Robotics Capability Framework levels increase, general-compute expectations typically become more demanding:

General-compute property	Lower Robotics Capability Framework levels	Higher Robotics Capability Framework levels
Compute execution	Supports fixed control, predefined logic and bounded workloads.	Supports heterogeneous workloads, AI inference, planning, learning and distributed execution.
Timing and determinism	Provides predictable timing for direct control and safety reactions.	Provides mixed-criticality timing across reflexive control, tactical reasoning and strategic services.
Memory and data handling	Stores configuration, simple state and limited operational data.	Supports larger models, persistent memory, context history, logs, maps, evidence and learning data.
Observability and diagnostics	Provides basic status, faults and health monitoring.	Supports richer telemetry, provenance, anomaly detection, drift monitoring and assurance evidence.
Lifecycle management	Supports manual configuration and controlled software updates.	Supports managed updates, validation, rollback, policy-controlled deployment and long-term lifecycle assurance.
Cybersecurity	Protects basic device access, communication and software integrity.	Protects models, data, identities, update mechanisms, distributed services and robot-edge-cloud interactions.
Safety & Resilience	FuSa, fallback or simple recovery.	Supports graceful degradation, fault containment, redundancy, runtime assurance and governed recovery.

General-compute capabilities do not define the robotic behavior themselves, but they strongly influence whether higher Robotics Capability Framework levels can be deployed safely and reliably. A system cannot credibly claim higher levels of robotic sophistication if the compute foundation cannot support the required latency, isolation, observability, updateability, security, evidence and fallback expectations.

Ecosystem-facing capability evolution

Ecosystem-facing capabilities also evolve as robotic systems progress through the Robotics Capability Framework levels.

At lower levels, a system may operate as a relatively closed product with limited integration needs. Interfaces may be static, updates infrequent and interoperability limited to known devices or tools.

At higher levels, the system typically depends on richer ecosystems: sensors, actuators, simulation tools, middleware, data platforms, model repositories, cloud or edge services, fleet infrastructure, operator tooling, assurance systems and third-party software. As a result, interoperability, integration, stability and scalability become increasingly important.

Copyright © 2026 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential
See Acknowledgements for Third-party Attributions

Ecosystem-facing property	Lower Robotics Capability Framework levels	Higher Robotics Capability Framework levels
Interoperability	Integration with known components and predefined interfaces.	Stable semantic interfaces across heterogeneous components, tools, services, vendors and deployments.
System integration	Integration of fixed hardware and software elements for known tasks.	Integration across AI, control, safety, security, data, simulation, operations and lifecycle infrastructure.
Stability	Limited change, fixed configurations and controlled releases.	Managed change across versions, interfaces, behaviors, evidence and deployed systems.
Scalability	Individual robots or small deployments.	Product families, fleets, sites, multi-agent systems, shared services and ecosystem-wide scaling.

Capability profiles rather than single-level labels

Because capabilities may evolve at different rates, a robotic system should not always be described as if every capability sits at the same level.

A system may have RL3 perception but RL2 manipulation. It may have RL4 decision-making within a bounded task but RL1 interaction. It may have System 3 learning infrastructure, while its runtime behavior remains RL2 for safety or regulatory reasons.

For this reason, the Robotics Capability Framework supports the idea of a **capability profile**. A capability profile describes how each relevant capability aligns with the required Robotics Capability Framework level for a defined use case, operating context and deployment envelope.

This avoids treating the Robotics Capability Framework as a simplistic maturity badge. A declaration such as “this robot is RL4” is less meaningful than a structured statement such as: “for this use case and operating context, the system demonstrates RL4 decision-making and perception, RL3 interaction and motion, and RL2 learning, with RL1/RL0 safety fallbacks.”

This is especially important because not every system should aim for the highest level in every capability. A lower level may be more appropriate where predictability, cost, certification, determinism or safety constraints dominate.

Use-case determines required sophistication

The required level of sophistication is determined by the use case, not by the desire to maximize the level.

A simple pick-and-place robot in a fixed environment may not require contextual reasoning, long-term memory or self-improving behavior. A warehouse AMR operating near people may require adaptive navigation, local contextual awareness and stronger interaction mechanisms. A humanoid assistant operating across changing human environments may require contextual perception, goal disambiguation, human-aware motion and stronger governance.

This means that the Robotics Capability Framework should be used to reason about **fitness for purpose**. The relevant question is not “how high is the robot’s level?” but “what level of sophistication is required from each capability for this use case, and what evidence supports that claim?”

Copyright © 2026 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential
See Acknowledgements for Third-party Attributions

Relationship with System 1, System 2 and System 3

The evolution across the Robotics Capability Framework levels is also reflected through the System 1, System 2 and System 3 layers.

At lower levels, capability behavior is dominated by System 1 and basic System 2 functions: reflexive execution, fixed rules, predefined workflows and bounded control.

At intermediate levels, System 2 becomes increasingly important: task-level planning, feedback adaptation, contextual reasoning, mission-bounded decision-making and human–robot collaboration.

At higher levels, System 3 becomes more prominent: long-term memory, model improvement, policy evolution, shared knowledge, fleet learning, governance, validation and rollback.

However, this progression does not mean that the system moves from System 1 to System 3 and leaves the previous layers behind. A high-sophistication system coordinates all three layers. System 1 maintains reflexive safety and stability. System 2 manages tactical execution. System 3 improves, governs and aligns behavior over time.

2.6 Alignment with the Robotics Capability Framework

The Robotics Capability Framework is intended to be used by adopters as a common reference for describing robotic system capability, not as a prescription for how systems must be implemented. An adopter may use the Robotics Capability Framework to declare that a system, product, architecture, requirement set or roadmap is aligned with a given Robotics Capability Framework level when the declaration is made using the framework's terminology, structure and capability expectations.

At this stage, such a declaration should be understood as an **alignment or conformance statement**, not as formal certification. Certification may become future work, but the immediate purpose is to enable consistent communication between vendors, customers, integrators, regulators, insurers, researchers and technology providers.

Terminology alignment

A declaration of alignment begins with the consistent use of the Robotics Capability Framework terminology.

Adopters should describe robotic systems using the vocabulary defined by the framework, including use cases, functions, capability categories, capabilities, Robotics Capability Framework levels and System 1 / System 2 / System 3 capability layers. This ensures that systems are described using a shared language rather than vendor-specific or ambiguous terms.

Terminology alignment allows different stakeholders to discuss capability, autonomy, intelligence, supervision, safety, assurance and deployment expectations using the same reference model. It does not require adopters to expose implementation details or to use the same technologies. It only requires that claims are expressed in terms that are consistent with the framework.

Structural alignment

A declaration should also show that the system has been described through the Robotics Capability Framework structure.

Copyright © 2026 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential
See Acknowledgements for Third-party Attributions

This means that the adopter should be able to trace a system from its intended use case and operational context, through the relevant functions and capability categories, down to the relevant capabilities and level-specific capability expectations that support the declared level. The purpose is to show that the declaration is grounded in a structured capability analysis, rather than in a broad marketing claim.

Structural alignment does not imply that every capability in the taxonomy must be equally advanced or implemented in the same way. Instead, it means that the relevant capability domains have been considered, that omissions or constraints are understood, and that the declared level can be related back to the intended operational outcomes.

The taxonomy therefore provides a traceability mechanism. It helps adopters explain why a system is being described at a given Robotics Capability Framework level, which capabilities support that declaration, and which assumptions, limitations or dependencies are attached to it.

Capability-level alignment

The Robotics Capability Framework levels describe the expected sophistication of a robotic system. The taxonomy describes the capabilities that may contribute to that sophistication. A useful declaration should therefore connect the claimed Robotics Capability Framework level to the relevant capability expectations.

In practice, this means that adopters should state not only that a system is aligned with a given level, but also where that alignment applies. A system may be aligned with one level for a specific use case, task, operating environment or capability domain, while remaining at a different level elsewhere. For example, a robot may demonstrate contextual behavior in navigation while relying on more bounded or supervised behavior for manipulation, learning or human interaction.

For this reason, a Robotics Capability Framework declaration should be contextual. It should identify the use case, operating environment, human role, supervision model, fallback assumptions and relevant capability domains. This prevents a Robotics Capability Framework level from becoming a simplistic label and preserves the framework's role as a capability-based reference.

Future versions of the framework may introduce level-specific objectives and primitives for each capability category. Objectives would describe the expected outcomes for a capability at a given Robotics Capability Framework level, while primitives would provide observable indicators that those outcomes have been achieved. These extensions could support more objective and comparable declarations without constraining the underlying implementation choices.

Copyright © 2026 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential
See Acknowledgements for Third-party Attributions

3. Applying the Robotics Capability Framework

The Robotics Capability Framework is intended to be used as a practical framework for aligning market needs, robotic system capability, sophistication expectations, product strategy and technology development.

It is not a static catalogue of functions, nor a prescription for how robotic systems must be implemented. Its purpose is to provide a structured flow that connects external signals — such as market needs, stakeholder expectations, workloads and operating contexts — with internal decisions about capabilities, capability sophistication, requirements, architecture, technology hypotheses and product differentiation.

Figure 4 illustrates this flow.

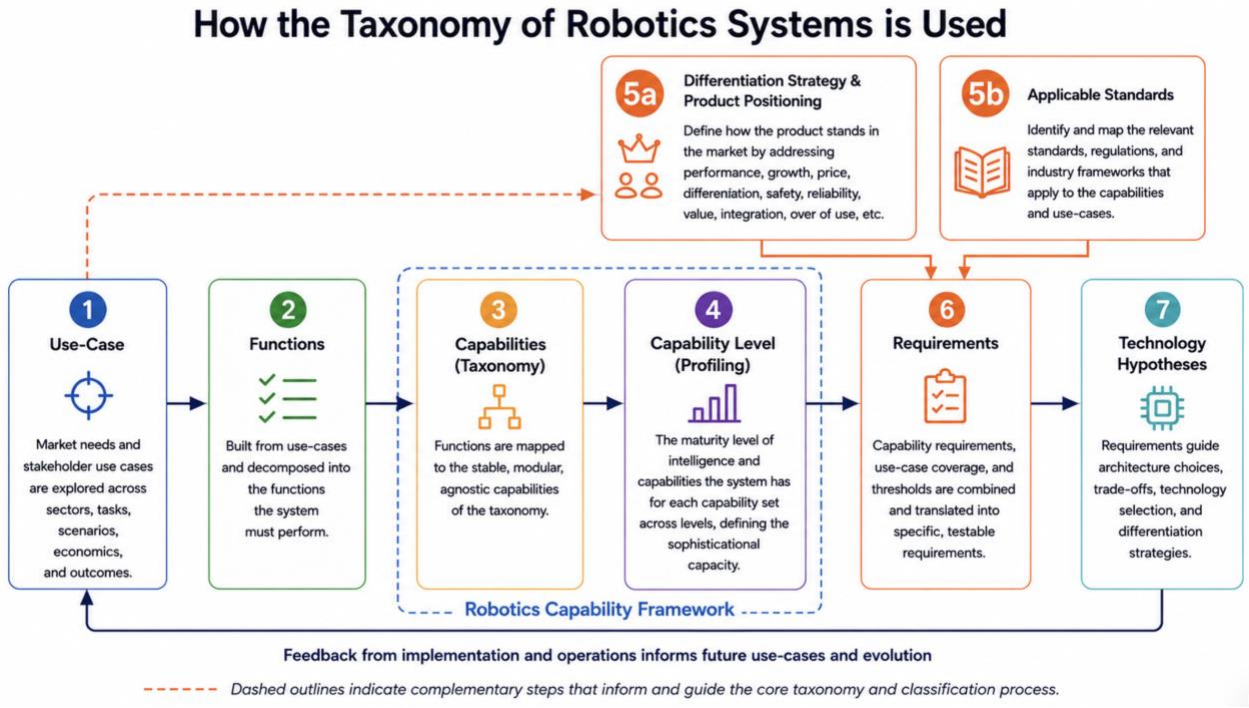


Figure 4. How the Robotics Capability Framework is applied from use case to technology hypotheses

1. Use case

The starting point is the use case.

Market needs and stakeholder expectations are captured across contexts, actors, scenarios, constraints and outcomes. A use case describes the real-world situation the robotic system is expected to support and the problem it is expected to solve.

A use case should not begin with an implementation assumption. It should describe the intended operational context, the relevant users or stakeholders, the operating environment, the expected outcome, the constraints and the success criteria.

Copyright © 2026 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential
See Acknowledgements for Third-party Attributions

For example, the use case may describe that a robot must move goods through a crowded warehouse, assist a person in a domestic environment, inspect assets in a hazardous location, collaborate with workers in a shared space or operate as part of a fleet.

This step grounds the Robotics Capability Framework in demand pull. The framework starts from the problem to be solved, not from a preferred technology choice.

2. Functions

The use case is then decomposed into the concrete functions the robotic system must perform.

Functions describe what the system does in operation. They are derived from the use case and reflect the behaviors required in that specific context.

Examples include detecting a human, maintaining localization, generating a motion plan, avoiding an obstacle, enforcing a policy constraint, interpreting a command, adapting to environmental change, handing over control or logging evidence for assurance.

This step is important because capability sophistication is not determined by a single feature or isolated function. The required sophistication emerges from the set of functions needed to satisfy the use case and from the behavioral expectations associated with those functions.

For example, immediate obstacle detection and predefined avoidance may require relatively low sophistication in Perception and Decision-Making. A use case requiring semantic scene understanding, intent interpretation, contextual action selection and governed adaptation may require higher levels in those same capabilities, while other capabilities may remain at lower levels.

3. Capabilities — taxonomy

The required functions are then mapped to the Robotics Capability Framework capability taxonomy.

Capabilities are the stable, solution-agnostic abilities that a robotic system may need to possess. They include robotics-specific capabilities such as Perception, Localization & Mapping, Physical Action, Decision-Making, Learning & Adaptation, Interaction and Governance Policy Control; general-compute capabilities such as security, manageability, connectivity, diagnostics, workload portability and lifecycle management; and ecosystem-facing capabilities such as interoperability, system integration, stability and scalability.

Capabilities do not change across Robotics Capability Framework levels. What changes is the sophistication expected from each capability.

For example, Perception remains Perception across the framework. At lower levels, it may support simple obstacle detection or object triggers. At higher levels, it may support semantic scene understanding, behavior awareness, predictive context, persistent knowledge or governed improvement of perception models.

This mapping step ensures that the framework remains complete and consistent. It prevents teams from jumping directly from a use case to a technology solution without first identifying the stable system capabilities that must be present.

Copyright © 2026 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential
See Acknowledgements for Third-party Attributions

4. The Robotics Capability Framework level — Profiling

Once the required functions have been mapped to the capability taxonomy, the RL0–RL5 sophistication scale is applied to each relevant capability according to the behavior required by the use case.

The purpose of this step is not to assign a single level to the robotic system. Different capabilities may require different levels of sophistication depending on the task, environment, human role, operating assumptions, autonomy boundary and assurance requirements.

The result is a **capability profile**.

For example, a warehouse logistics use case may require RL3 Contextual Perception and Decision-Making because the robot must interpret changing conditions and select actions according to current context, while RL2 Adaptive Localization & Mapping and Physical Action may be sufficient for local rerouting and bounded physical execution. Interaction may remain RL1 Deliberative where scripted commands and supervisory control are adequate.

The Robotics Capability Framework levels therefore provide a **common sophistication scale**, while the capability profile describes how that scale applies across the relevant capability domains.

A capability profile should not be reduced to an average or single maturity score. The shape of the profile is meaningful because it identifies where higher sophistication is necessary and where lower sophistication remains sufficient or desirable.

This step is the core of the combined Robotics Capability Framework: the taxonomy provides the capability structure, the RL0–RL5 levels provide the sophistication scale, and the use case determines the resulting capability profile.

5a. Differentiation strategy and product positioning

The Robotics Capability Framework defines what capabilities are required and how sophisticated they must be for the intended use case, but it does not prescribe how a product should compete in the market.

Product teams must therefore determine how the product stands in the market by addressing dimensions such as performance, growth, price, reliability, value, integration, usability and ease of use.

Examples of differentiation may include higher performance, lower power consumption, reduced cost, smaller physical footprint, faster deployment, improved safety or security, simplified programmability, stronger ecosystem compatibility, better integration experience, accelerated time to market or longer lifecycle support.

This step translates market intent into engineering emphasis. Two systems may target similar use cases and capability profiles while differentiating differently. One may compete on performance and autonomy. Another may compete on cost, safety, ease of integration or ecosystem openness.

The Robotics Capability Framework therefore defines the capability and sophistication baseline, while product strategy defines where to over-invest, differentiate or simplify.

5b. Applicable standards

The Robotics Capability Framework does not define the normative safety, security, regulatory or compliance obligations that apply to a specific use case.

Copyright © 2026 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential
See Acknowledgements for Third-party Attributions

Those obligations depend on the operating context, market, geography, sector, risk profile, embodiment, supervision model and deployment environment. Product teams must therefore identify and map the relevant standards, regulations and industry frameworks that apply to the capabilities and use case.

Examples may include functional safety, active safety, passive safety, SOTIF, cybersecurity, privacy, machinery safety, collaborative robot safety, medical device requirements, industrial automation standards, sector-specific regulation or public-space deployment guidance.

This step ensures that capability and sophistication expectations are interpreted through the correct assurance lens. It helps determine what evidence, validation, verification, traceability, diagnostics, fallback behavior, documentation, monitoring and governance may be required for the system to be acceptable in its intended market and use case.

The Robotics Capability Framework can support this process by clarifying what each capability is expected to do and at what level of sophistication. The applicable standards then define the normative obligations and evidence expectations associated with the operating context.

6. Requirements

The mapped capabilities, interpreted through the **capability sophistication profile** and refined by product differentiation and applicable standards, are translated into specific and testable requirements.

Requirements combine capability expectations, use-case context and applicable standards. They define what must be true for the system to be designed, validated, deployed and governed.

These requirements may cover system behavior, performance, latency, safety, security, reliability, fallback behavior, observability, explainability, data handling, interoperability, lifecycle management, update mechanisms and evidence generation.

This step gives architects and engineering teams clear guidance for defining interfaces, evaluating trade-offs, selecting technologies, partitioning functionality and establishing implementation constraints.

7. Technology hypotheses

Requirements then guide architectural exploration and technology selection.

Architects and engineers use them to formulate and evaluate technology hypotheses. These may include hardware and software architectures, interface contracts, deployment models, acceleration strategies, operating systems, middleware, AI models, toolchains, update mechanisms, safety mechanisms and integration approaches.

Multiple implementation options may satisfy the same requirements. This is intentional. The Robotics Capability Framework is implementation-agnostic, so it should allow different technical approaches to be explored without losing alignment with the system objectives.

For example, a requirement for contextual perception does not prescribe whether the solution must use a particular model architecture, sensor configuration or compute platform. It defines what the capability must achieve. Technology teams can then evaluate alternative implementation paths against the same capability and sophistication expectations.

Strategic alignment

Beyond engineering use, the Robotics Capability Framework provides an abstraction for strategic alignment.

Copyright © 2026 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential
See Acknowledgements for Third-party Attributions

By separating use cases, functions, capabilities, sophistication levels, differentiation strategy, applicable standards, requirements and implementation choices, the Robotics Capability Framework gives business leaders, architects, product teams and ecosystem partners a common way to discuss investments, roadmaps, partnerships and technology evolution.

Each element plays a distinct role:

Use cases provide context.
Functions describe what the system must do.
Capabilities provide completeness.
The Robotics Capability Framework **levels** provide sophistication.
Differentiation strategy provides competitive intent.
Applicable standards provide the assurance and compliance lens.
Requirements provide engineering direction.
Technology hypotheses provide implementation paths.

Used together, these elements create a scalable methodology for advancing robotic systems from market opportunity to deployed products while preserving clarity, traceability and implementation freedom

Copyright © 2026 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential
See Acknowledgements for Third-party Attributions

Appendix A Interpretation notes

A Robotics Capability Framework **level is not a product badge**

A Robotics Capability Framework level should not be used as a simplistic label such as “this robot is RL4” without qualification. A system may demonstrate different levels across different capabilities. For example, it may have RL3 perception, RL2 manipulation, RL4 decision-making within a bounded task and RL1 interaction. The framework therefore supports a **capability profile** rather than a single undifferentiated maturity claim.

A more precise statement is:

For this use case and operating context, the system demonstrates RL4 decision-making and perception, RL3 interaction and motion, RL2 learning, and RL0/RL1 safety fallbacks.

Lower levels remain necessary

Higher levels do not remove the need for lower-level behavior. A cognitive or self-improving system still requires reactive safety, deterministic control loops, bounded fallbacks, emergency stops and predictable failure behavior. The progression from RL0 to RL5 is cumulative: higher-level reasoning is added above reflexive and tactical foundations rather than replacing them.

The required level is determined by the use case

The Robotics Capability Framework should be used to reason about **fitness for purpose**. A fixed pick-and-place robot in a controlled environment may be entirely appropriate at RL0 or RL1. A warehouse AMR operating near people may require RL2 or RL3 capabilities. A domestic assistant operating in ambiguous human environments may require RL3 or RL4 interaction, perception and governance. A fleet that improves policies across deployments may require RL5 learning and governance mechanisms.

Level transition summary

Transition	What changes	Why it matters
RL0 → RL1	From immediate reaction to predefined task execution.	The system begins to follow structured workflows rather than only responding to triggers.
RL1 → RL2	From fixed execution to bounded adaptation.	The system can improve or adjust performance within a known envelope.
RL2 → RL3	From bounded adaptation to contextual action selection.	The system begins to interpret situation, intent and short-term context.
RL3 → RL4	From contextual response to cognitive reasoning.	The system can reason across goals, constraints, trade-offs and longer horizons.

Copyright © 2026 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential
See Acknowledgements for Third-party Attributions

RL4 → RL5	From cognitive planning to governed self-improvement.	Improvement over time becomes an explicit capability requiring validation, rollback and accountability.

Discontinuities between higher Robotics Capability Framework levels

The transitions between Robotics Capability Framework levels should not be read as a smooth linear increase in sophistication. Each level introduces a qualitative change in system behavior, but the transitions from **RL3 to RL4** and from **RL4 to RL5** are especially significant.

RL3 to RL4: from contextual action selection to cognitive deliberation

The transition from **RL3 Contextual** to **RL4 Cognitive** is not simply a move from “more context” to “better decisions”.

At RL3, the system interprets the current situation and selects appropriate actions within a bounded operating context. It may disambiguate intent, respond to environmental change, adapt to human presence and select behaviors based on short-term situational awareness.

At RL4, the system is expected to reason over goals, constraints, trade-offs and consequences. This introduces a different class of capability. The system is no longer only selecting the most appropriate action for the current context; it is deliberating across possible courses of action, evaluating competing objectives, considering longer-horizon effects and, where required, explaining or justifying why one plan is preferable to another.

This transition typically introduces expectations such as persistent multi-goal reasoning, hierarchical planning from goal to task to skill to motion, causal or semantic reasoning, interrupt-aware cognition, explainable goal arbitration, longer-horizon memory, traceable decision provenance and stronger assurance around why a decision was made.

The key distinction is that **RL3 selects contextually appropriate actions**, while **RL4 reasons about goals, alternatives and consequences**.

RL4 to RL5: from cognitive autonomy to governed self-improvement

The transition from **RL4 Cognitive** to **RL5 Self-improving** is different again. RL5 is not merely “more cognitive” or “more autonomous”. It introduces governed improvement over time as an explicit system capability.

At RL4, the system can reason, plan and justify action within its operating envelope. It may use long-term memory, causal inference and multi-goal reasoning, but the system’s behavioral model, policies and strategic competence are still primarily defined through controlled design, training, deployment and update processes.

At RL5, the system can refine strategies, models, policies or behaviors through operational experience, shared learning or cross-context optimization. This may occur across a single long-lived robot, a fleet, a site, a product family or a distributed robot-edge-cloud system.

This transition typically introduces expectations such as governed model or policy evolution, validation gates before behavioral updates affect operation, rollback of learned or deployed changes, continuous attestation, shared learning across agents, sites or fleets, long-term memory federation, digital-twin or simulation-based validation, policy update history, provenance and accountability for behavior that changes over time.

The key distinction is that **RL4 reasons within a governed operating envelope**, while **RL5 can improve that reasoning, behavior or strategy over time under governance**.

Copyright © 2026 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential
See Acknowledgements for Third-party Attributions

RL5 therefore does not mean unconstrained self-modification. It means that improvement itself becomes a managed, auditable and bounded capability. The system may evolve, but only through mechanisms that preserve validation, traceability, rollback, accountability and human-defined governance constraints.

Summary of higher-level discontinuities

Transition	Not merely…	But a shift toward…
RL3 to RL4	More context or better adaptation	Cognitive deliberation across goals, constraints, trade-offs, memory and explainable planning
RL4 to RL5	More autonomy or more advanced reasoning	Governed improvement of models, policies, strategies or behaviors over time

These discontinuities are important because they affect assurance expectations. RL4 increases the need for explainability, provenance and reasoning assurance. RL5 increases the need for validation, rollback, continuous attestation and governance of behavioral change.

Copyright © 2026 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential
See Acknowledgements for Third-party Attributions

Appendix B Example capability-profile matrix

This annex provides illustrative examples of how the Robotics Capability Framework can be used to describe a robotic system as a **capability profile**, rather than as a single undifferentiated level.

A capability profile identifies the level of sophistication required from each relevant capability for a defined use case, operating context and supervision model. It recognizes that a system may demonstrate different Robotics Capability Framework levels across different capabilities. For example, a robot may require contextual perception and decision-making, while retaining adaptive motion and reactive safety fallbacks.

The examples in this annex are illustrative only. They do not define mandatory requirements for all warehouse, logistics, eldercare or assistance systems. The required level of sophistication depends on the intended use case, operating environment, supervision model, risk profile, applicable standards and evidence expectations.

Purpose of a capability profile

A Robotics Capability Framework capability profile is used to describe:

Element	Purpose
Use case	The real-world scenario or outcome the robotic system is expected to support.
Operating context	The environment, actors, constraints and assumptions under which the system operates.
Target system behavior	The expected sophistication of the system in that context.
Capability profile	The Robotics Capability Framework level expected from each relevant capability.
Human role	The expected form of human control, monitoring, intervention, guidance or governance.
Safety and governance expectations	The fallback, oversight, auditability and assurance expectations associated with the profile.

A capability profile should not be read as a product architecture. It does not prescribe hardware, software, AI models, sensors, controllers, middleware, deployment location or component decomposition. It describes what the system is expected to achieve.

Copyright © 2026 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential
See Acknowledgements for Third-party Attributions

Example use-case evolution across the Robotics Capability Framework levels

A capability profile identifies the level of sophistication required from each relevant robotics-specific capability for a defined use case, operating context and supervision model. The purpose is not to assign a single Robotics Capability Framework level to the robot. Different aspects of the same use case may place very different demands on Perception, Localization & Mapping, Physical Action, Decision-Making, Learning & Adaptation, Interaction and Governance Policy Control.

The Robotics Capability Framework levels therefore provide a **common sophistication scale for each capability**, while the resulting capability profile describes the combination required to deliver the use case.

This is particularly important because changes in a use case do not necessarily increase all capability requirements together. For example:

- introducing changing routes may increase the required sophistication of Localization & Mapping without requiring richer Interaction;
- operating around people may increase Perception and Governance requirements while the physical transport task remains relatively simple;
- introducing dynamic operational priorities may increase Decision-Making without requiring more sophisticated Physical Action;
- allowing the system to reason across competing warehouse objectives may push Decision-Making and Governance toward cognitive behavior while Interaction or Learning remain at lower levels.

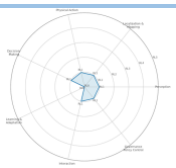
The capability profile is therefore represented as a **spider-web**, where each axis corresponds to a robotics-specific capability and RL0–RL5 provides the common sophistication scale. The shape of the profile is the meaningful result; capability levels should not be averaged into a single maturity score.

Copyright © 2026 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential
See Acknowledgements for Third-party Attributions

Variations of the Warehouse Logistics Assistant use case

The examples below illustrate how variations of the same broad Warehouse Logistics Assistant use case produce different capability profiles.

They are not intended to represent a mandatory maturity progression. Each is a different expression of the use case, with capability requirements selected according to the definitions and level-specific behaviors established by the Robotics Capability Framework.

Use-case variation	Capability profile	What changes in the use case
A — Structured goods transport		The robot moves goods along predefined routes in a structured warehouse. Objects, routes and workflows are known in advance. Human interaction is scripted and operational changes are handled externally. This requires primarily **RL1 Deliberative** behavior: predefined perception triggers, basic positioning, rule-based physical execution and predefined decision logic. Learning is not required and remains **RL0**.
B — Adaptive shared warehouse		The robot operates in a warehouse where obstacles, congestion and local layouts change. It must use feedback to reroute, adjust motion and adapt task execution while remaining within predefined objectives. This moves Perception, Localization & Mapping, Physical Action and Decision-Making to **RL2 Adaptive**, while Interaction can remain scripted at **RL1** and Learning can remain limited to externally managed tuning or updates.
C — Contextual warehouse assistant		The robot now operates around people, vehicles, changing traffic and operational priorities. It must interpret the meaning of the current situation and select actions accordingly. Perception and Decision-Making therefore move to **RL3 Contextual**, as does Governance Policy Control because autonomy must be constrained according to current context. Localization & Mapping and Physical Action remain **RL2 Adaptive**, because local rerouting and feedback-driven execution are sufficient. Interaction remains **RL1 Deliberative** where scripted worker interaction is adequate. This is the profile developed in Section B.1.
D — Cognitive warehouse coordinator		The robot is additionally expected to reason across competing operational objectives such as delivery urgency, congestion, battery state, resource availability and human priorities, and to explain or justify its decisions. Decision-Making therefore progresses to **RL4 Cognitive**, based on persistent multi-goal deliberation and evaluation of trade-offs. Governance Policy Control also reaches **RL4** to provide explainable policy arbitration, provenance and governed execution. Perception and Localization & Mapping remain **RL3 Contextual**, Physical Action remains **RL2 Adaptive** because goods transport is still physically bounded, Learning & Adaptation reaches **RL3** to make contextual experience available to future behavior, and Interaction reaches only **RL2** where task-aware coordination remains sufficient.

Copyright © 2026 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential
See Acknowledgements for Third-party Attributions

B.1 Example profile C — Warehouse Logistics Assistant

Use-case statement

A warehouse logistics assistant moves goods through a semi-structured warehouse environment shared with people, vehicles and other automation systems. The system is expected to transport items, adapt to local congestion, avoid obstacles, respect operational priorities and interact safely with human workers.

The capability requirements for this use case are intentionally **not uniform across the Robotics Capability Framework levels**. Different capabilities require different degrees of sophistication according to their role in delivering the use case.

For example, the system requires contextual Perception and Decision-Making because it must interpret changing warehouse conditions and select appropriate actions. However, its physical task remains comparatively constrained, so Physical Action can remain Adaptive. Rich human interaction is unnecessary, and Learning & Adaptation can remain Deliberative because operational learning is performed predominantly through externally managed engineering and deployment processes.

The result is a **use-case capability profile**, rather than a single system maturity score as shown in the following spider-web capability diagram.

Figure 5. Warehouse assistant capability profile

Copyright © 2026 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential
See Acknowledgements for Third-party Attributions

Operating assumptions

Dimension	Assumption
Environment	Semi-structured warehouse with mapped zones, workers, vehicles, dynamic obstacles and changing traffic patterns.
Task type	Goods transport, collection, delivery, queue handling and local task prioritization.
Physical interaction	Primarily locomotion and simple load handling; dexterous or highly general manipulation is not required.
Human role	Operational supervision and on-demand intervention.
Autonomy boundary	The robot can adapt execution and make local contextual decisions within an assigned task and operating zone. Strategic warehouse priorities remain externally governed.
Learning model	Models, policies and operational knowledge are primarily developed, validated and deployed externally. Limited runtime learning is required.
Fallback model	Safe stop, bounded local recovery, remote assistance or operator intervention when confidence, safety or policy limits are exceeded.

Copyright © 2026 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential
See Acknowledgements for Third-party Attributions

Capability-profile matrix

Capability	Target level	Use-case reason for selecting that level	Framework-defined capability behavior at that level
Perception	**RL3 — Contextual**	The robot must distinguish the significance of people, vehicles, goods, blocked aisles and changing warehouse conditions, not merely detect objects.	Foundation-model based perception with semantic understanding. At System 2, it "fuses semantic, multimodal and temporal observations to interpret people, objects, events and the current situation."
Localization & Mapping	**RL2 — Adaptive**	The warehouse is mapped and reasonably structured. The robot must handle blocked aisles, layout changes and local rerouting, but does not require semantic world-model continuity to accomplish the task.	Updates local or mission-relevant spatial representations to support adaptive navigation, rerouting or changing layouts
Physical Action	**RL2 — Adaptive**	The physical behavior is relatively bounded: transport goods, adjust trajectory, avoid obstacles and adapt execution from feedback. It does not require context-aware manipulation or general physical reasoning. .	Adaptive execution with feedback. At System 2, it "adjusts trajectories, grasps, timing, paths or motion parameters according to bounded feedback and execution outcomes."
Decision-Making	**RL3 — Contextual**	The robot must use warehouse context — such as traffic, urgency, human presence, blocked routes and task intent — to determine an appropriate action.	Probabilistic reasoning with context, memory, intent and goal understanding. At System 2, it "selects actions based on current context, intent, short-horizon reasoning and goal disambiguation."
Learning	**RL2 – Adaptive**	The system may adapt task execution or retain task-specific experience, but it does not need cross-context or persistent contextual learning.	Task-specific adaptation, calibration and short-term retention of learned parameters or experience. At System 2, it performs "task-scoped adaptation from feedback, demonstration or recent operational data."
Interaction	**RL1 - Deliberative**	Workers mainly need commands, status, warnings and predefined interactions. Contextual dialogue or intent interpretation is not essential to the use case.	Scripted INTERACTION. At System 2, it supports "scripted commands, predefined responses, prompts and supervisory control."
Safety & Trust governance	**RL3- Contextual**	Because contextual decisions can vary with traffic, people and operational state, the permitted autonomy envelope also needs contextual runtime enforcement.	At System 2, "dynamic boundary enforcement, context-aware constraint handling, safety arbitration and immediate fallback." System 3 maintains and deploys the policies defining the contextual operating envelope.

Copyright © 2026 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential
See Acknowledgements for Third-party Attributions

Profile interpretation

This example should not be interpreted as a uniformly RL3 system. Instead, it illustrates how a single use case can require different levels of sophistication across the robotics-specific capabilities defined by the Robotics Capability Framework.

For this warehouse logistics use case:

- **Perception is RL3 — Contextual**, because the robot must interpret semantically meaningful aspects of the current situation, including people, objects, events and changing warehouse conditions.
- **Localization & Mapping is RL2 — Adaptive**, because the system must update local or mission-relevant spatial representations to support rerouting and changing layouts, but does not require persistent semantic world-model continuity.
- **Physical Action is RL2 — Adaptive**, because execution must adjust trajectories, timing and movement according to feedback while remaining within bounded task objectives and physical constraints.
- **Decision-Making is RL3 — Contextual**, because action selection depends on current context, intent, short-horizon reasoning and operational conditions such as traffic, urgency or blocked routes.
- **Learning & Adaptation is RL2 — Adaptive**, because task-scoped adaptation and short-term retention of learned parameters or experience are sufficient; broader contextual or persistent learning is not required.
- **Interaction is RL1 — Deliberative**, because scripted commands, predefined responses, prompts and supervisory control are adequate for the worker interaction required by the use case.
- **Governance Policy Control is RL3 — Contextual**, because contextual autonomy requires dynamic boundary enforcement, context-aware constraint handling, safety arbitration and immediate fallback.

The purpose of the profile is not to maximize every capability to the highest level, nor to calculate an average maturity score. It is to identify the level of sophistication that each capability must provide for the use case to be satisfied.

The profile therefore makes visible where higher sophistication is necessary and where it would add little value. In this example, contextual perception and decision-making are essential to operate appropriately in a changing warehouse environment, while adaptive physical execution and mapping are sufficient for the bounded logistics task, and scripted interaction is adequate for the expected human role.

The spider-web representation should therefore be read as the **shape of the use-case requirement across capabilities**, not as a progression toward a uniform level. Different use cases will produce different shapes, reflecting different capability demands within the same Robotics Capability Framework.

Copyright © 2026 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential
See Acknowledgements for Third-party Attributions

Terms and definitions

Term	Meaning
Use-case	A real world scenario or outcome the robotics system is expected to support. It defines the context, actions, environments, constraints and success criteria
Function	A concrete behavior or operation the system performs within a use-case. Functions describe what happens in context, such as detecting a hazard, planning a path, or interpreting a command
Capability category	A grouping of related capabilities by their role in the robotics system, such as domain, general-purpose, or ecosystem capabilities.
Capability	A stable, solution-agnostic ability the system must possess. Capabilities describe what the system must be able to do, independent of implementation
Capability component	A logical building block within a capability architecture. Components decompose a capability for analysis, interfaces, requirements, and technology hypotheses
Objective	A level-specific outcome that a capability is expected to achieve at a given Robotics Capability Framework maturity level. Objectives describe what success looks like for the capability at that level. *Unless otherwise stated, objectives are defined per Robotics Capability Framework level.*
Primitive	A level-specific measurable building block required to achieve an objective. Primitives define how maturity is enabled and provide observable criteria for assessing compliance with a Robotics Capability Framework level. *Unless otherwise stated, primitives are defined per Robotics Capability Framework level.*
Requirement	A specific testable or constraint derived from a use-case, function, capability, and target the Robotics Capability Framework level. Requirements define what must be true for the system to be designed, validated, or governed
RLx	Robotics Capability Level x

Copyright © 2026 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential
See Acknowledgements for Third-party Attributions

Proprietary notice

This document is protected by copyright and other related rights and the practice or implementation of the information contained in this document may be protected by one or more patents or pending patent applications. No part of this document may be reproduced in any form by any means without the express prior written permission of Arm Limited ("Arm"). **No license, express or implied, by estoppel or otherwise to any intellectual property rights is granted by this document unless specifically stated.**

Your access to the information in this document is conditional upon your acceptance that you will not use or permit others to use the information for the purposes of determining whether the subject matter of the document infringes any third party patents.

The content of this document is informational only. Any solutions presented herein are subject to changing conditions, information, scope, and data. This document was produced using reasonable efforts based on information available as of the date of issue of this document. The scope of information in this document may exceed that which Arm is required to provide, and such additional information is merely intended to further assist the recipient and does not represent Arm's view of the scope of its obligations. You acknowledge and agree that you possess the necessary expertise in system security and functional safety and that you shall be solely responsible for compliance with all legal, regulatory, safety and security related requirements concerning your products, notwithstanding any information or support that may be provided by Arm herein. In addition, you are responsible for any applications which are used in conjunction with any Arm technology described in this document, and to minimize risks, adequate design and operating safeguards should be provided for by you.

This document may include technical inaccuracies or typographical errors. THIS DOCUMENT IS PROVIDED "AS IS". ARM PROVIDES NO REPRESENTATIONS AND NO WARRANTIES, EXPRESS, IMPLIED OR STATUTORY, INCLUDING, WITHOUT LIMITATION, THE IMPLIED WARRANTIES OF MERCHANTABILITY, SATISFACTORY QUALITY, NON-INFRINGEMENT OR FITNESS FOR A PARTICULAR PURPOSE WITH RESPECT TO THE DOCUMENT. For the avoidance of doubt, Arm makes no representation with respect to, and has undertaken no analysis to identify or understand the scope and content of, any patents, copyrights, trade secrets, trademarks, or other rights.

TO THE EXTENT NOT PROHIBITED BY LAW, IN NO EVENT WILL ARM BE LIABLE FOR ANY DAMAGES, INCLUDING WITHOUT LIMITATION ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL, PUNITIVE, OR CONSEQUENTIAL DAMAGES, HOWEVER CAUSED AND REGARDLESS OF THE THEORY OF LIABILITY, ARISING OUT OF ANY USE OF THIS DOCUMENT, EVEN IF ARM HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

Reference by Arm to any third party's products or services within this document is not an express or implied approval or endorsement of the use thereof.

This document consists solely of commercial items. You shall be responsible for ensuring that any use, duplication or disclosure of this document complies fully with any relevant export laws and regulations to assure that this document or any portion thereof is not exported, directly or indirectly, in violation of such export laws. Use of the word "partner" in reference to Arm's customers is not intended to create or refer to any partnership relationship with any other company. Arm may make changes to this document at any time and without notice.

This document may be translated into other languages for convenience, and you agree that if there is any conflict between the English version of this document and any translation, the terms of the English version of this document shall prevail.

The validity, construction and performance of this notice shall be governed by English Law.

The Arm corporate logo and words marked with ® or ™ are registered trademarks or trademarks of Arm Limited (or its affiliates) in the US and/or elsewhere. All rights reserved. Please follow Arm's trademark usage guidelines at https://www.arm.com/company/policies/trademarks. Other brands and names mentioned in this document may be the trademarks of their respective owners.

Arm Limited. Company 02557590 registered in England.
110 Fulbourn Road, Cambridge, England CB1 9NJ.
(PRE-1121-V1.0)

Copyright © 2026 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential
See Acknowledgements for Third-party Attributions

Document information

Revision history

These sections can help you understand how the document has changed over time.

Document release information

The Document history table gives the issue number and the released date for each released issue of this document.

Document history

Issue	Date	Confidentiality	Change
Version 1.0	September 2026	Non-Confidential	First non-confidential release.

Copyright © 2026 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential
See Acknowledgements for Third-party Attributions