ARCHITECTURE, LIFECYCLE AND RUNTIME GOVERNANCE

INTENT-COMPILED SOFTWARE

The Computational Architecture That Carries Human Authority Into Machine Action

A comprehensive definition of how activity intent, governor's intent and validated system-event semantics are transformed into an approved causal autonomy entity that governs digital and physical commitment continuously.

DEFINING PROPOSITION **Intent-compiled software establishes human-authorized governance once per approved version and evaluates it continuously against changing causal reality.**

Decision-Zone ICC™ Architecture Paper
Decision-Zone Inc. | August 2026

Executive thesis

Modern computing has made the machine actor computational. Software and AI can perceive, predict, plan, propose, coordinate and initiate actions across digital and physical systems. The governing function remains different. It must preserve whose intent applies, which authority is active, what conditions must remain true, which transitions are permissible, how concurrent activity must be coordinated and which consequences are acceptable before a proposed action becomes real.

Conventional software gives organizations powerful representations, workflows, permissions and component controls. Model-driven systems add computational actors that generate new outputs from changing context. ICC completes this landscape with one approved governing artifact that persists independently of the actor and remains computationally present at each consequential commitment.

ICC introduces intent-compiled software. Human operational knowledge, designated governing authority and real system-event semantics are compiled into formal causal structures, approved as a versioned autonomy entity and deployed into a runtime that evaluates live events, maintains causal state, coordinates admissible activity, enforces the commit boundary, validates consequences and produces an intent-to-execution causal trace.

THE ARCHITECTURE IN ONE VIEW

Model-driven software automates the actor. ICC™ makes the complete governed role autonomous.

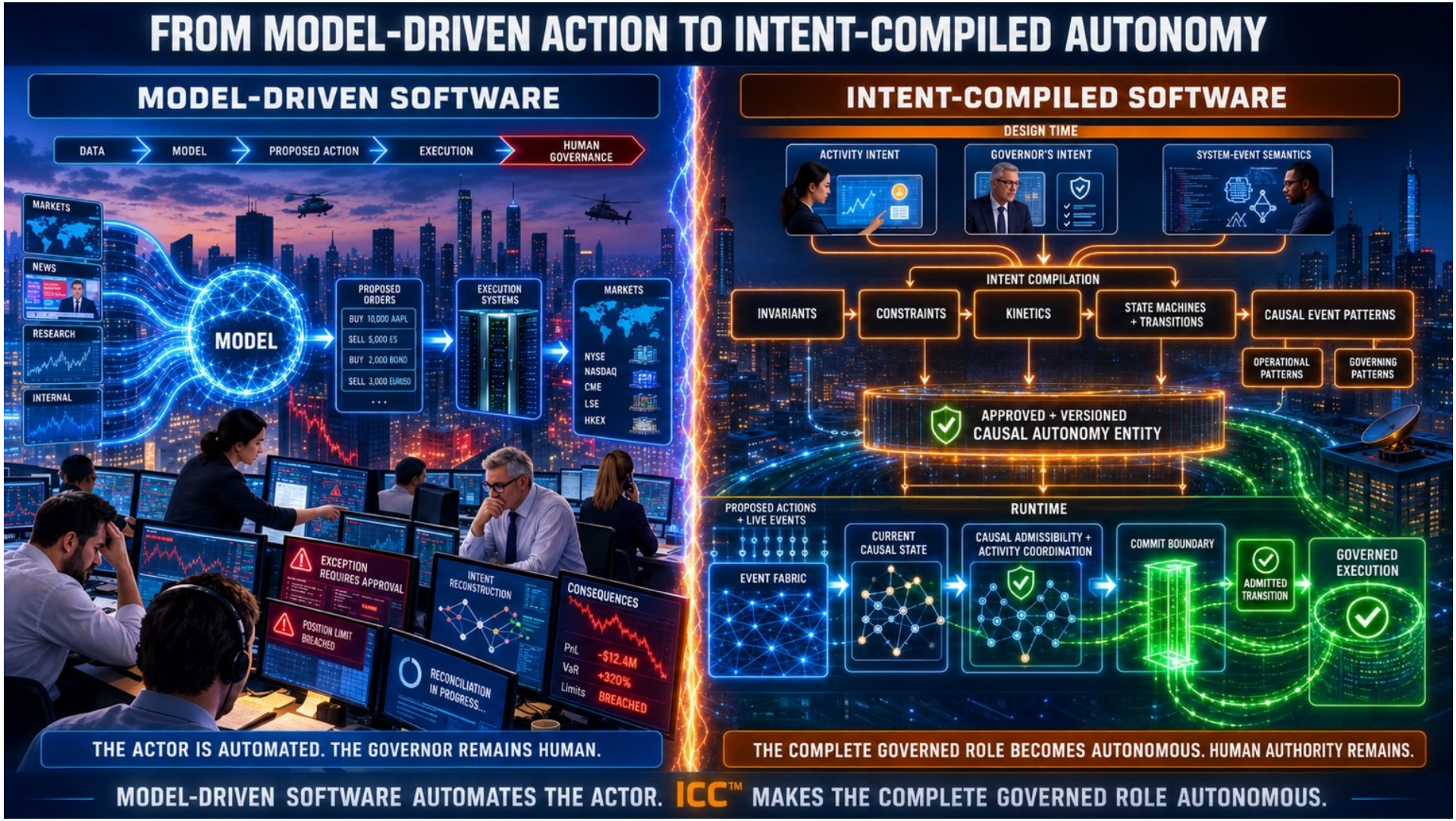

Figure 1. Model-driven software automates the actor while human governance remains in the operating loop. Intent-compiled software makes both the activity and governing functions computational while human authority remains.

1. Intent-compiled software: reading the architecture

Intent-compiled software is an architecture in which the durable source of runtime governance is a compiled, human-authorized causal artifact. Generated model responses, requirements, prompts and local controls remain valuable inputs and components; the compiled entity gives their governing intent an executable representation, a defined lifecycle and an enforceable position before commitment.

The category is defined by persistence. Compilation produces an artifact that remains in force until an authorized change is compiled, validated, approved, versioned and deployed. Runtime evaluates live proposals and events against the approved entity already in force. Governance therefore remains stable while models, tasks, events and conditions continue to change.

COMPLEMENTARY LAYERS **Generation produces new possibilities and proposals. Compilation produces the approved governing structure through which proposals become admissible commitments.**

Six defining properties

1. **Human source.** Operational meaning and governing authority originate with identified people and designated organizational roles.
2. **Formal transformation.** Intent is converted into invariants, constraints, kinetics, state machines, state transitions and causal event patterns.
3. **Approved persistence.** The compiled entity is versioned, authorized and carried forward independently of the proposing model.
4. **Live causal evaluation.** The entity evaluates current events, concurrency, authority and consequences as the causal situation evolves.
5. **Commit enforcement.** Only an admitted transition reaches a bound capability capable of producing digital or physical consequence.
6. **Causal evidence.** Observed effects return as events and are tied to the governing intent, admissibility determination and commitment that preceded them.

The left side: model-driven software automates the actor

The left side of Figure 1 shows a model-driven trading environment. Market data, news, research and internal information converge on the model. The model interprets those representations and generates proposed orders. Execution systems carry the orders into multiple markets. This is the actor architecture: data becomes model output, model output becomes proposed action and execution systems produce real financial consequences.

The lower trading floor shows the governing work still performed by people. Human operators approve exceptions, respond to breached position limits, reconstruct originating intent, reconcile systems and manage consequences. The software has automated the actor, while the human continues to carry the complete governing position across changing market conditions.

The center: the architectural transition

The luminous center line separates two software architectures. The model-driven side computationalizes the actor and leaves the governing function with people. The intent-compiled side captures the operational and governing functions explicitly, compiles them into one approved entity and places that entity in the execution path before commitment.

The upper right: compiling the complete governed role

The intent-compiled architecture begins at design time with three approved source domains. Operators and SMEs supply activity intent: how the role performs the work. Designated human authorities supply governor's intent: under whose authority and conditions activity may become real. System engineers supply validated system-event semantics: the event types, command paths, bindings and commitment interfaces through which the existing systems operate.

Intent compilation transforms these sources into invariants, constraints, kinetics, state machines, state transitions and causal event patterns. Operational patterns preserve how the role acts. Governing patterns preserve authority and admissibility. Human validation and approval produce the approved, versioned causal autonomy entity shown at the center of the right-hand architecture.

The lower right: the compiled entity governs runtime

At runtime, proposed actions and live events enter the ICC event fabric. The runtime maintains current causal state, activates the approved operational and governing patterns, evaluates causal admissibility and coordinates the activity. The commit boundary passes an admitted transition to the bound execution capability, producing governed execution. Consequences return as events and continue the causal loop.

The bottom line: the complete role becomes autonomous

The graphic's conclusion is the core strategic distinction. Model-driven software automates the actor. ICC computationalizes the actor and governor together. The continuous human operator can leave the execution loop because activity intent and governor's intent persist in the approved entity. Human authority remains the source of governance and is carried forward computationally into every commitment.

WHAT THE GRAPHIC ESTABLISHES **Model-driven software makes machine action scalable. Intent-compiled software makes the complete governed role scalable—activity, authority, admissibility, commitment and consequence together.**

2. From digital representation to persistent governance

Conventional digital systems introduce intent through requirements, user stories, policies, procedures, prompts, configuration and code. During implementation, that intent becomes representations and component behavior. Each subsystem preserves the elements required for its own function. ICC adds a unified executable form for the complete governing position across those components.

As the system operates, context windows, plans, commands, local state and models continue to evolve. ICC carries the originating intent alongside those transformations as an approved causal artifact. This makes governance explicit at runtime and connects each current action to the authority under which it is evaluated.

System-specific safeguards, permissions, interlocks and workflows remain essential. They provide strong local and component-level controls. ICC integrates them with the owner's complete activity and governing intent across the current causal situation, allowing cross-system consequential judgments to be evaluated computationally.

Model-driven systems strengthen the need for the governing layer

A model produces an output from its current input, context, parameters and learned representations. The output is a fresh probabilistic proposal optimized for the present situation. The approved causal entity supplies the complementary approval, version, provenance and authority required for governed commitment.

Models are computational actors and proposers. They expand the range, speed and quality of possible action. ICC allows that growing capability to operate at scale by carrying the human governing position into each commitment computationally.

CURRENT ARTIFACT	WHAT IT PRESERVES	ICC COMPLEMENT
Prompt or task input	A current request or instruction	Persistent governing authority
Model context	Information available for the current generation	Approved continuity across generations
Code or workflow	Implemented component behavior	Complete cross-system governing intent
Permission	Who or what may access a resource	Current causal admissibility
Interlock	A protected local condition	Integrated authority, concurrency and consequence judgment
Log	What the system recorded	Pre-commitment governance and causal proof

3. Three distinct intent domains

Intent-compiled software separates three intent domains because they answer different questions, come from different sources and have different lifecycles. Their explicit separation preserves the requested outcome, the capability of the role and the authority governing what may become real.

Task intent: what outcome is requested now?

Task intent identifies a mission, transaction, assignment or requested effect. It originates with a customer, owner, commander, application or authorized requester. It is normally specific to a runtime instance and may change from request to request. In the ICC architecture, task intent enters as an authority-scoped event while the reusable role and governing authority retain their approved versions.

Activity intent: how does the role perform the work?

Activity intent captures the operational knowledge of the role: activities, triggers, dependencies, sequences, coordination requirements, exceptions and recovery pathways. It is supplied by operators, traders, representatives, engineers and subject-matter experts. It is comparatively stable and is compiled into operational causal event patterns.

Governor's intent: under what authority and conditions may activity become real?

Governor's intent captures authority, invariants, constraints, permissible transitions, consequence boundaries, escalation, suspension and revocation. It is supplied and approved by owners, commanders and designated authorities in operations, safety, security, legal and compliance functions. It is compiled into governing causal event patterns and remains distinct from operational capability.

NON-COLLAPSING RULE **Task intent remains a scoped request. Activity intent remains operational capability. Governor's intent remains distinct, approved and enforceable inside one executable entity.**

INTENT DOMAIN	PRIMARY QUESTION	LIFECYCLE	RUNTIME ROLE
Task	What outcome is requested now?	Mission or transaction specific	Authority-scoped event
Activity	How does the role perform the work?	Reusable and versioned	Operational causal patterns
Governor's	May this activity become real under current authority and conditions?	Approved and versioned	Governing causal patterns

4. The three compilation inputs

The compiled entity combines exactly three input classes so that human knowledge, human authority and real system capability remain connected while preserving their distinct meanings.

1. **Human operational knowledge.** Operators and subject-matter experts supply activity intent: how the role performs, coordinates, handles exceptions and recovers. This input defines operational meaning, while governing authority remains with its designated human source.
2. **Human governing authority.** Designated authorities supply governor's intent: what must remain true, what must not be violated, who may authorize which transition and which consequences require blocking, holding, escalation, suspension or revocation.
3. **Validated system-event semantics.** System engineers establish the real event types, sensors, APIs, controllers, command paths, capability bindings and enforceable commit boundaries. Code demonstrates system capability; designated human authority defines the conditions under which that capability may be committed.

Source separation and provenance

Each compiled element retains its source and provenance. Operational meaning remains attributable to the experts who supplied it. Governing authority remains attributable to the authorities who approved it. System bindings remain attributable to validated technical interfaces. This separation supports review, challenge, correction and controlled change.

The role of the LLM

The LLM assists intent engineering. It extracts concepts from natural language, reconciles terminology, exposes ambiguity, identifies conflicts, proposes formal constructs and helps translate human knowledge into a structure suitable for validation. Authority remains with the human sources and the approved compiled entity occupies the governing position at runtime.

Human experts validate operational meaning. Designated authorities validate and approve governance. System engineers validate realizability and bindings. The compiled entity is deployed through this human authorization process. The LLM's role remains translation and formalization; approval, version control and runtime authority remain explicitly human-authorized.

AUTHORITY BOUNDARY **The LLM translates. Humans validate and authorize. The approved compiled artifact—not the model—governs runtime.**

5. The canonical compilation transformation

Compilation transforms resolved human intent and validated system semantics into formal causal structures. Each stage adds precision required for execution while preserving the provenance of the activity and governing sources.

1. **Resolved intent.** The operational purpose, governing authority, scope, terminology and conflicts are made explicit. Ambiguity is surfaced for human resolution before authority is compiled.
2. **Invariants.** Conditions that must remain true across admissible operation. They define non-negotiable properties of the role, authority or protected system state.
3. **Constraints.** Conditions and boundaries that must not be violated. They restrict transitions, combinations, timing, authority scope and consequences.
4. **Kinetics.** The permitted character of change: ordering, timing, rates, dependencies, concurrency, synchronization and transformation dynamics.
5. **State machines and transitions.** Valid states and movements provide explicit lifecycle structure for activities, authority, capability and consequence handling.
6. **Causal event patterns.** Operational and governing patterns express the partial-order causal relationships that the runtime can activate and evaluate against live events.

COMPILATION INVARIANT **The transformation increases formal precision while authority remains attached to its designated human source and approved governing patterns.**

Validation and approval

Compilation becomes authoritative through human validation, approval, versioning and deployment. Operators and SMEs validate operational meaning. Authorities approve governing meaning and scope. Engineers validate event semantics and enforceable bindings. The complete entity is then assigned an approved version and deployed. Any change to approved activity, governance or realization proceeds through controlled recompilation, validation, versioning and deployment.

6. The compiled causal autonomy entity

The output of intent compilation is one executable, governed and versioned causal autonomy entity. It contains the complete autonomous role while preserving the actor and governor as distinct computational functions.

Internal anatomy

- Formal causal structure: invariants, constraints, kinetics, state machines and permitted transitions.
- Operational causal event patterns: how the role performs, sequences, coordinates, handles exceptions and recovers.
- Governing causal event patterns: authority, conditions, prohibitions, consequence boundaries, escalation and revocation.
- System-event bindings: validated relationships between causal patterns, event types, sensors, APIs, controllers and capabilities.
- Commit controls: the enforceable boundary through which only admitted transitions reach bound capabilities.
- Authority and provenance: the approved source, scope, version and ownership of each governing element.
- Consequence-validation patterns: how observed effects are compared with compiled expectations and recovery requirements.
- Causal trace specification: how intent, active state, proposal, admissibility, commitment and observed consequence are connected as evidence.

PERSISTENCE CARRIER **Persistence is carried by approved, versioned causal event patterns—not by prompts, model context, generated plans, databases of historical state or proposed actions.**

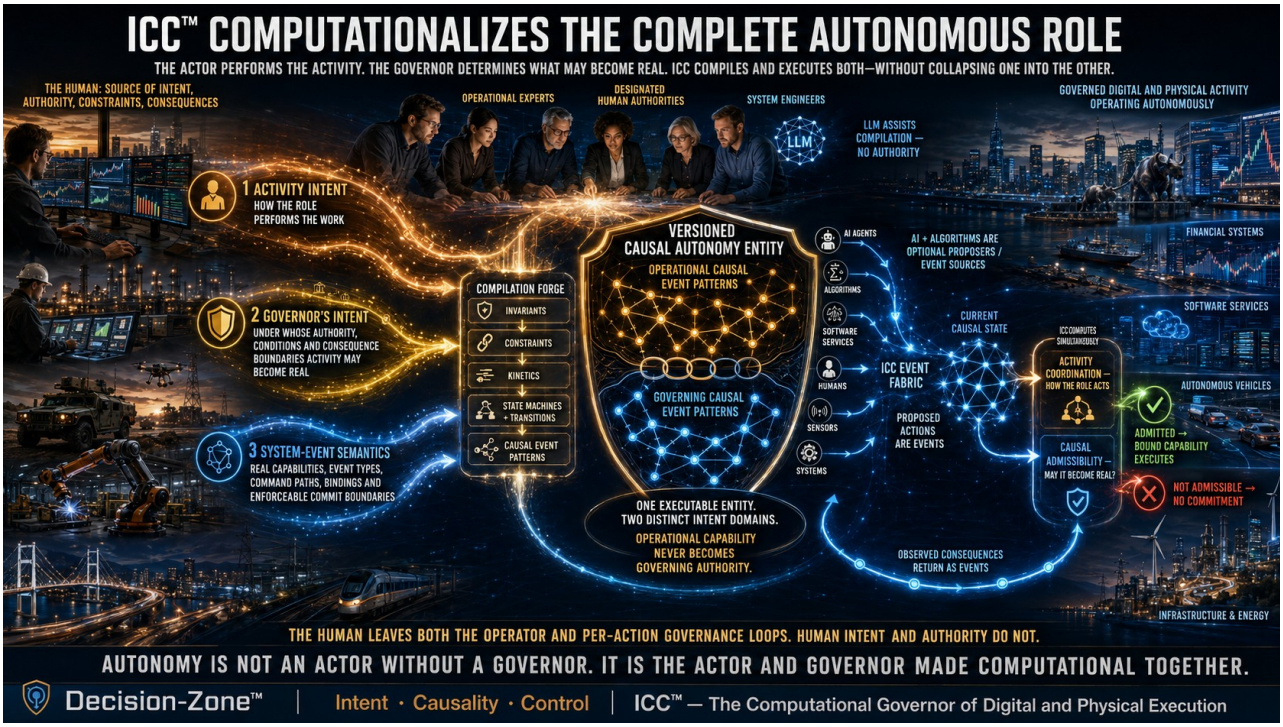

Figure 2. ICC compiles the complete autonomous role: operational capability and governing authority remain distinct inside one approved causal entity.

7. Runtime inputs and the ICC event fabric

The ICC runtime begins with live event instances. Events are not limited to sensor observations. They include the task that activates a role, the actions proposed by computational or human actors, the changing situation reported by systems and sensors, and the consequences observed after commitment.

Four event classes

1. **Authority-scoped task events.** A mission, transaction, assignment, requester identity, authority scope and requested effect enter at runtime. The task activates an approved role instance while the reusable role retains its approved version.
2. **Proposed-action events.** Candidate transitions from AI agents, algorithms, applications, workflows or humans enter the event fabric as proposals. A proposal is an event—not execution and not authority.
3. **Live system and sensor events.** System state, sensor readings, environment, operator activity, alarms, measurements and concurrent events represent the actual causal situation as it changes.
4. **Observed-consequence events.** Effects, fault conditions, deviations, recoveries and new causal conditions return after execution and participate in subsequent evaluation.

Event-fabric semantics

The event fabric carries typed event instances, causal relationships, partial-order concurrency, authority attribution and system-event bindings. The runtime integrates events from heterogeneous systems through shared event semantics while each system retains its own implementation. The fabric is the changing operational reality presented to the approved entity.

The compiled entity evaluates the event fabric as the stable approved specification. Runtime proposals and contextual changes evolve causal state while the governing definition retains its authorized version.

RUNTIME SEPARATION **Events change continuously. The approved causal pattern specification remains immutable during execution.**

8. Current causal state, admissibility and activity coordination

Current causal state

Current causal state is the active situation produced by live event instances and their causal relationships. It includes active task and authority scope, system and environmental conditions, concurrent activity, completed and pending transitions, and observed consequences. It represents which compiled patterns are active and how the current situation came to be, extending beyond a snapshot of stored values.

Partial-order concurrency

Real systems contain events that are simultaneous, independent, causally dependent or only partially ordered. ICC preserves these relationships directly. The governor therefore determines admissibility from the actual causal configuration, with the total-order log retained as supporting operational history.

Causal admissibility

Causal admissibility determines whether a proposed transition may proceed under the active authority, compiled invariants and constraints, permitted kinetics, current causal state and consequence boundaries. The question is specific and immediate: may this transition become real now?

Activity coordination

The same runtime coordinates how the role acts. It can sequence, synchronize, trigger, redirect, recover and revoke activity according to operational patterns while simultaneously enforcing governing patterns. Activity coordination and governance are integrated, but operational capability never becomes the authority that approves itself.

DETERMINATION	RUNTIME EFFECT
Admit	The transition is eligible to cross the commit boundary.
Block	The transition remains outside the admissible state space under the current causal situation.
Hold	Commitment waits while required conditions are resolved.
Sequence	The transition is ordered relative to other required activity.
Redirect	The role selects an approved alternate causal pathway.
Escalate	A designated authority or higher-order process is activated.
Revoke	Previously available authority or capability is withdrawn.

9. The commit boundary and physical realization

The commit boundary is the enforceable architectural position immediately before a digital or physical transition becomes consequential. It separates a proposal or admissibility determination from the capability that can make the transition real.

An admitted transition is delivered to a bound capability: an API, controller, actuator, vehicle, robot, financial system, infrastructure component or other execution mechanism. The commit boundary passes admitted transitions and retains all other proposals within the governed runtime for blocking, holding, redirection, escalation or revocation according to the compiled patterns.

COMMIT PRINCIPLE **Nothing executes merely because a model proposed it or a component has permission to access the capability. Execution requires admission under current causal state and approved human authority.**

Bound capabilities

A bound capability is an identified, validated means of producing an effect. Binding connects abstract causal transitions to real system interfaces and preserves the enforcement point. One causal architecture can therefore govern heterogeneous machines, APIs and controllers through their validated semantics.

The model as a governed proposer

A model may recommend, plan or propose a transition. It may also provide perception, classification or prediction used as event input. Its proposed action enters as one event among the live inputs evaluated by the approved causal entity, and an admitted result proceeds through the commit boundary.

Digital and physical commitment

The same governance principle applies to digital and physical consequences. A trade, payment, data release, access change or software command can be irreversible or consequential even when no mechanical actuator moves. ICC therefore governs bound digital capabilities and bound physical capabilities through the same pre-commitment architecture.

10. Consequence validation and the causal trace

Commitment continues the governance lifecycle. Once a bound capability executes, observed effects return as consequence events. The runtime compares those observations with the compiled consequence-validation patterns, updates current causal state and activates any required continuation, recovery, hold, redirect, escalation or revocation.

The closed causal loop

RUNTIME PATH **Task and live events → proposed action → current causal state → causal admissibility and coordination → commit-boundary enforcement → bound capability → observed consequence → event fabric**

Intent-to-execution causal trace

The trace connects the approved intent and authority to the active causal pattern, current state, proposed transition, admissibility determination, commitment, bound capability and observed consequence. It provides evidence of why a transition was admitted or refused and whether the observed effect corresponded to the compiled intent.

The trace extends operational logging

An operational log records events or commands after they occur. The ICC trace extends that evidence because it is produced by the same causal governance that evaluated the transition before commitment. It links pre-execution authority and admissibility to post-execution observation.

Continuous governance

The loop repeats at machine speed. New tasks, proposals, system conditions and consequences continue to enter as events. The approved entity remains stable while causal state evolves. Governance therefore continues from the approved version across every action.

LIFECYCLE STATEMENT **Compile once per approved version. Govern continuously. Validate consequences. Recompile only through authorized change.**

11. Relationship to models, agents and conventional software

Intent-compiled software complements existing software and intelligence with a governing layer that separates proposal generation from authority and commitment. Models, algorithms, workflows and human actors remain valuable sources of events and proposed actions.

LAYER	PRIMARY COMPUTATION	ICC RELATIONSHIP
Foundation model or LLM	Interpret, reason, generate and translate	May assist compilation or propose runtime actions; never supplies authority by itself
World model	Represent or predict possible states and consequences	Predictions may enter as events; admissibility remains independently governed
Agent	Plan, coordinate tools and pursue assigned outcomes	Every proposed consequential action enters the event fabric
Application or workflow	Implement business or mission procedures	Existing events and command paths are bound into the causal entity
Permission system	Control access to resources	ICC adds current causal admissibility to authorized access
Safety interlock	Prevent a specific local unsafe condition	Remains a valuable local control beneath integrated governance
Human operator	Perform activity and manually govern consequences	Operational and governing intent are compiled; continuous supervision can be removed

Model replacement and model updates

Governing authority resides in the approved causal entity, allowing models to be replaced while the organization retains its governance artifact. A model change receives technical validation of event contracts, proposed-action semantics and capability bindings. Recompilation is triggered by a change to the approved role, governing intent or validated system-event semantics; a model producing the same validated event types can remain within the existing governing version.

Why this matters strategically

An organization can license intelligence while owning governance. It can use different models for different tasks, change vendors and integrate new agents while its own approved causal entity remains the durable source of organizational authority.

12. Retrofit architecture: governing what already exists

ICC is designed to govern existing actors and capabilities rather than requiring a clean-sheet machine. The organization already has applications, algorithms, operators, vehicles, robots, controllers, APIs and infrastructure performing work. Intent compilation extracts the operational role and governing position, connects them to validated event semantics and inserts an enforceable commitment layer into the existing action path.

Four deployment patterns

1. **Inline commit gateway.** ICC is positioned immediately before an API, controller or actuator so that only admitted commands reach the capability.
2. **Embedded runtime.** The ICC runtime is deployed within a platform or mission system with direct access to its event sources and commitment interfaces.
3. **Cross-system governing fabric.** ICC integrates events and authority across heterogeneous applications, devices and infrastructure that retain their existing implementations.
4. **Sovereign governance layer.** Approved national or organizational intent is compiled independently of any model or technology supplier and carried across critical systems through controlled bindings.

Retrofit sequence

- Identify existing actors, capabilities, event sources and consequential commit points.
- Capture activity intent from operators and domain specialists.
- Capture governor's intent from designated authorities.
- Validate system-event semantics, command paths and enforceable boundaries.
- Compile, simulate, validate, approve and version the causal autonomy entity.
- Bind the approved entity to live event sources and capabilities.
- Use supervised commissioning to validate operation, traces and consequences, then transition to authorized autonomous deployment.

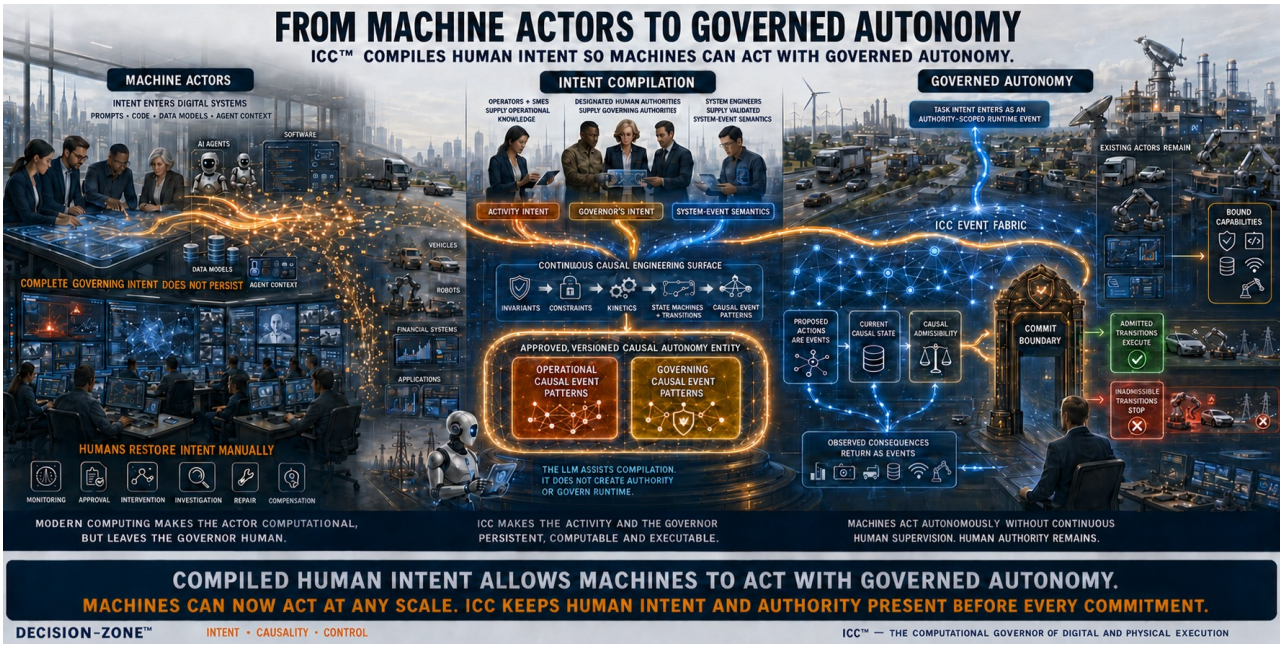

Figure 3. ICC occupies the governing position across existing AI, software, vehicles, robots and infrastructure; the installed actors and capabilities remain usable.

13. Computationalizing the operator while preserving human authority

A human operator frequently performs two functions at once. The operator carries activity intent—how to perform the work—and governor's intent—whether a particular action should be allowed under current authority, circumstances and consequences. Conventional automation replaces pieces of the activity while leaving the governing judgment with the operator.

Intent compilation makes both functions computational while preserving their separation. Operational causal patterns execute the role. Governing causal patterns remain independently authoritative over what may become real. The human can leave the continuous operator and per-action approval loops because approved intent and authority remain present in the compiled entity.

Replication of autonomous roles

Once an approved role has been compiled and bound to compatible systems, it can be instantiated repeatedly under authority-scoped tasks. A governed trader, customer representative, logistics coordinator, vehicle operator or mission controller can therefore be replicated with the approved supervisory judgment already carried by the entity. Each role instance uses the same approved operational and governing patterns while receiving its own tasks and live causal events.

Three representative applications

ROLE	ACTIVITY INTENT	GOVERNOR'S INTENT	BOUND CONSEQUENCE
Trader	Evaluate opportunities, place and manage orders	Authority, limits, exposure, market conditions, consequence thresholds	Financial commitment
Vehicle operator	Navigate, coordinate, recover and complete mission	Authorized route, safety invariants, environmental conditions, protected actors	Vehicle motion
Defense mission role	Detect, classify, coordinate and respond	Commander's authority, identification conditions, escalation and engagement boundaries	Mission-system commitment

AUTONOMY DEFINITION **Physical autonomy makes the actor and governor computational together while preserving their separation and human source.**

14. Controlled change, versioning and deployment

The approved causal entity is immutable during execution. Change occurs through an explicit design-time lifecycle so that operational meaning, governing authority and system realizability remain reviewable and traceable.

Changes that require controlled recompilation

- A change to the reusable activity role, including activities, coordination, dependencies, exceptions or recovery.
- A change to governing authority, invariants, constraints, permissible transitions, consequences, escalation or revocation.
- A change to validated system-event semantics, event contracts, capability bindings or enforceable commit boundaries.
- A material environmental change requiring approved assumptions and causal patterns to be updated.

Changes that normally remain runtime events

- A new authority-scoped task, mission, transaction or assignment.
- A new proposed action from a model, algorithm, application or human.
- A changing sensor reading, system state, environmental condition or concurrent event.
- An observed consequence, deviation, fault or recovery condition.

Version transition

A proposed new version is compiled and validated outside the active runtime authority. Designated experts and authorities review the change, system bindings are verified and the new version is deployed through a controlled transition. The prior version remains identifiable for traceability and rollback according to the organization’s deployment controls.

GOVERNANCE CONTINUITY **Runtime change evolves causal state. Authorized recompilation changes the governing definition. This distinction preserves governance continuity across runtime change.**

15. ICC integrated platform capabilities

ICC integrates the complete intent-compilation and runtime-governance lifecycle in one architecture. Each capability below contributes a required part of governed autonomy, and together they carry human operational knowledge and governing authority from design time through execution and observed consequence.

#	ICC CAPABILITY	ARCHITECTURAL FUNCTION
1	Human intent engineering	Captures operational knowledge, governing authority and validated system-event semantics with source provenance.
2	Canonical intent compilation	Transforms approved intent into invariants, constraints, kinetics, state machines, transitions and causal event patterns.
3	Versioned causal autonomy entity	Carries operational and governing patterns, authority, provenance, bindings and commit controls as one approved artifact.
4	System realization and binding	Connects causal patterns to real event contracts, sensors, APIs, controllers and digital or physical capabilities.
5	Proposal-as-event integration	Receives proposals from models, algorithms, applications and humans inside the same governed event fabric.
6	Integrated current causal state	Maintains the active authority, concurrent events, dependencies, system conditions and consequence state across systems.
7	Causal admissibility	Evaluates what may become real now against compiled authority, patterns and current causal state.
8	Activity coordination	Sequences, synchronizes, redirects, escalates, recovers and revokes activity according to the approved role.
9	Commit-boundary enforcement	Passes admitted transitions to validated bound capabilities and retains every commitment within computational governance.
10	Consequence validation	Returns observed effects as events, compares them with compiled expectations and activates follow-on governing activity.
11	Intent-to-execution causal trace	Connects intent, authority, active state, proposal, admissibility, commitment and observed effect as one evidence chain.
12	Controlled version lifecycle	Carries each approved version continuously and introduces change through authorized recompilation, validation and deployment.

One architecture, one continuous governing lifecycle

These capabilities operate as one system. Intent engineering establishes the approved role. Compilation makes that role persistent and executable. Runtime governance integrates live events, maintains causal state, coordinates activity, evaluates admissibility and enforces commitment. Consequence validation and the causal trace close the loop. The result is a complete computational governor for digital and physical execution.

16. Strategic and economic position

Intent-compiled software is hardware-independent and model-neutral. Its economic surface spans devices, models and domains wherever proposed actions can be expressed as events and commitment can be bound to an enforceable interface.

The physical-AI market has already demonstrated investor willingness to value portable software independently of hardware manufacturing. Skild AI announced a financing at a valuation above $14 billion; Physical Intelligence reportedly reached $5.6 billion; Applied Intuition reached a reported $15 billion. These comparables establish cross-platform software as strategic infrastructure and provide a relevant category context for Decision-Zone's own technical and commercial evidence.

The distinct Decision-Zone control point

Model companies provide intelligence that can operate many machines. ICC provides governance that can operate across many models and machines. The model layer becomes the portable actor; intent-compiled software becomes the portable governor across actors and hardware.

Value drivers

- Universality across digital and physical domains, independent of any model vendor, robot manufacturer or application stack.
- Retrofit into installed infrastructure rather than replacement of existing investment.
- Persistent organizational and sovereign authority across distributed systems.
- Reduction of continuous human supervision while retaining evidence linking approved intent, execution and observed consequence.
- Protected causal-event architecture, implementation depth and enforceable commit-boundary integration.

STRATEGIC STATEMENT **Models make machine intelligence portable. Intent-compiled software makes human authority portable, persistent and executable across that intelligence.**

Conclusion

The defining opportunity in machine action is a persistent computational governor carrying approved human intent and authority to each consequential commitment. Model outputs provide fresh intelligence, local controls protect components and ICC integrates these capabilities into governance that scales with machine activity.

Intent-compiled software creates one approved causal autonomy entity from activity intent, governor's intent and validated system-event semantics. It remains stable while live reality changes, governs proposals before commitment, coordinates admissible activity, controls bound capabilities, validates consequences and produces causal evidence. This is governed autonomy: compile once per approved version and govern continuously at machine scale.

Sources and architecture basis

1. Decision-Zone Inc., ICC Architecture Charter, Version 3, Controlled Architecture Baseline.
2. U.S. Patent 7,908,260, Decision-Zone causal-event construction, validation and execution architecture.
3. Skild AI, "Skild AI Raises $1.4B, Now Valued Over $14B," January 14, 2026. https://finance.yahoo.com/news/skild-ai-raises-1-4b-150800863.html
4. Bloomberg, "Robotics Startup Physical Intelligence Valued at $5.6 Billion in New Funding," November 20, 2025. https://www.bloomberg.com/news/articles/2025-11-20/robotics-startup-physical-intelligence-valued-at-5-6-billion
5. Reuters, "Applied Intuition valued at $15 billion for autonomous vehicle tech," June 17, 2025. https://www.reuters.com/business/autos-transportation/applied-intuition-valued-15-billion-latest-fund-raise-2025-06-17/

APPENDIX A | GOVERNED DRONE AUTONOMY

The drone actor performs the mission. ICC™ carries mission authority to every consequential transition.

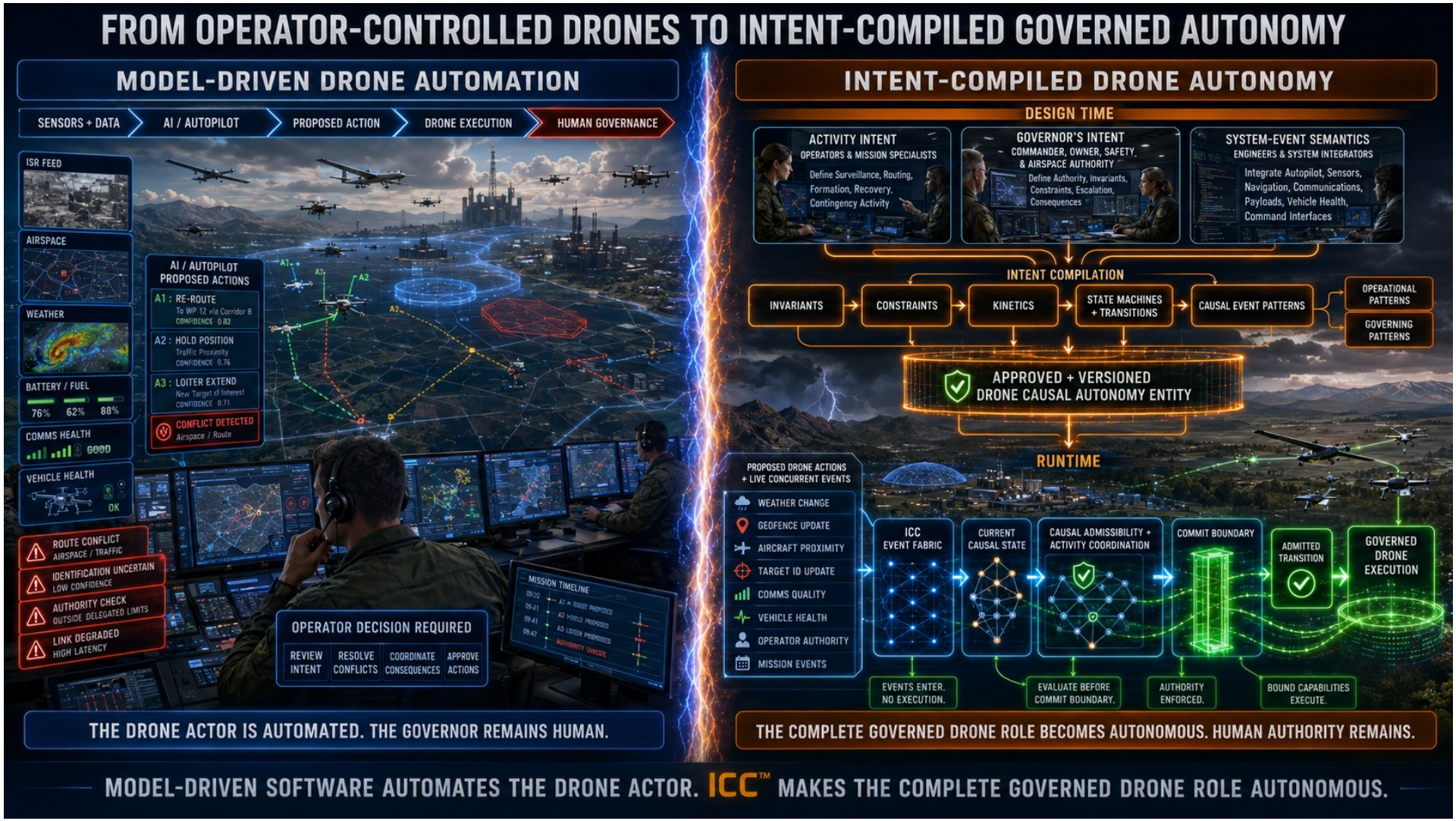

Figure A1. Existing sensing, AI, autopilot and vehicle capabilities become a governed autonomous drone system when activity intent, governor's intent and system-event semantics are compiled and enforced before commitment.

Appendix A. Governed drone autonomy

A modern drone is already a capable computational actor. Its sensors observe the environment, navigation software maintains flight, onboard or external models interpret conditions, and mission software proposes routes, formations, payload activity and recovery actions. These capabilities automate the work of flight and mission execution. Governed autonomy adds the complete governing position that determines whether each proposed transition may become real under current mission authority and current causal conditions.

The complete drone role

The drone role extends beyond flying from one waypoint to another. It may receive a mission, establish a route, coordinate with other aircraft, maintain separation, observe an area, inspect an asset, change formation, manage communications, respond to degraded vehicle state and recover safely. Each activity is related to other activity through dependencies, concurrency, sequencing, exceptions and recovery pathways. ICC compiles this complete operational role rather than treating each command as an isolated transaction.

The same role also carries a governing dimension. Mission authority defines who may assign the task, which airspace and capabilities are authorized, which safety and identification conditions must remain true, when escalation is required, and which consequences are acceptable. Operational capability and governing authority remain distinct inside one executable causal autonomy entity.

Three approved compilation inputs

1. **Activity intent.** Operators and mission specialists define surveillance, routing, formation, coordination, inspection, contingency and recovery activity, including the causal relationships among those activities.
2. **Governor's intent.** Commanders, owners, airspace authorities and safety authorities define mission authority, invariants, constraints, permissible transitions, escalation conditions and consequence boundaries.
3. **System-event semantics.** Engineers validate how autopilot, navigation, sensors, communications, payloads, vehicle health, command interfaces and enforceable flight-control boundaries express real events and capabilities.

Intent engineering transforms these sources into invariants, constraints, kinetics, state machines, state transitions and causal event patterns. Operational patterns preserve how the mission role acts. Governing patterns preserve the authority and conditions under which that activity may proceed. Human validation and approval produce the approved, versioned drone causal autonomy entity deployed into the ICC runtime.

DRONE AUTONOMY PROPOSITION **The approved drone entity carries the complete mission role forward: operational capability, governing authority, system bindings and commitment controls in one versioned causal architecture.**

Appendix A. Runtime governance of drone activity

At runtime, the mission or task arrives as an authority-scoped event. Proposed route changes, formation changes, payload activity and other candidate transitions also enter as events. They join live airspace, weather, geofence, aircraft-proximity, target-identity, communications, sensor, vehicle-health and consequence events inside the ICC event fabric. A proposal therefore remains part of the causal situation until the runtime determines what may proceed.

Current causal state and activity coordination

The runtime maintains the current causal state of the mission rather than evaluating a proposal against a static snapshot. Active authority, concurrent aircraft, route dependencies, communication quality, weather, vehicle condition and earlier consequences remain causally related. Approved operational patterns coordinate what the drone should do next; approved governing patterns determine what may become real now.

Causal admissibility and activity coordination operate together. ICC can admit, hold, sequence, redirect, escalate, recover or revoke activity according to the approved role. This permits the drone to respond at machine speed while retaining the human source of mission authority throughout changing conditions.

The commit boundary

Only an admitted transition crosses the commit boundary and reaches a bound drone capability. The bound capability may be an autopilot command, control-surface transition, navigation update, formation instruction, sensor operation, payload function or recovery action. Authority is therefore enforced immediately before the consequential transition becomes physical.

Consequence validation and mission evidence

After execution, observed effects return as consequence events. The runtime validates those effects against the approved causal patterns, evolves current causal state and activates any required continuation, correction, recovery, escalation or revocation. The resulting trace connects mission intent, authority, causal state, proposal, admissibility, commitment and observed effect as one evidence chain.

Operational scale

The approved role can be instantiated across individual aircraft, coordinated formations or large fleets under distinct authority scopes and tasks. Intelligence, autopilots and vehicle designs may vary while the governing architecture remains portable across their validated event semantics and commitment interfaces. The continuous drone operator can leave the per-action loop; human mission authority remains computationally present across the fleet.

OUTCOME **The drone actor becomes fully governed at machine speed: mission activity is coordinated, authority is preserved, consequential transitions are admitted before execution and observed effects close the causal loop.**

APPENDIX B | MULTI-TASK HUMANOID AUTONOMY

One approved humanoid role governs changing tasks, body actions, human interaction and physical consequence.

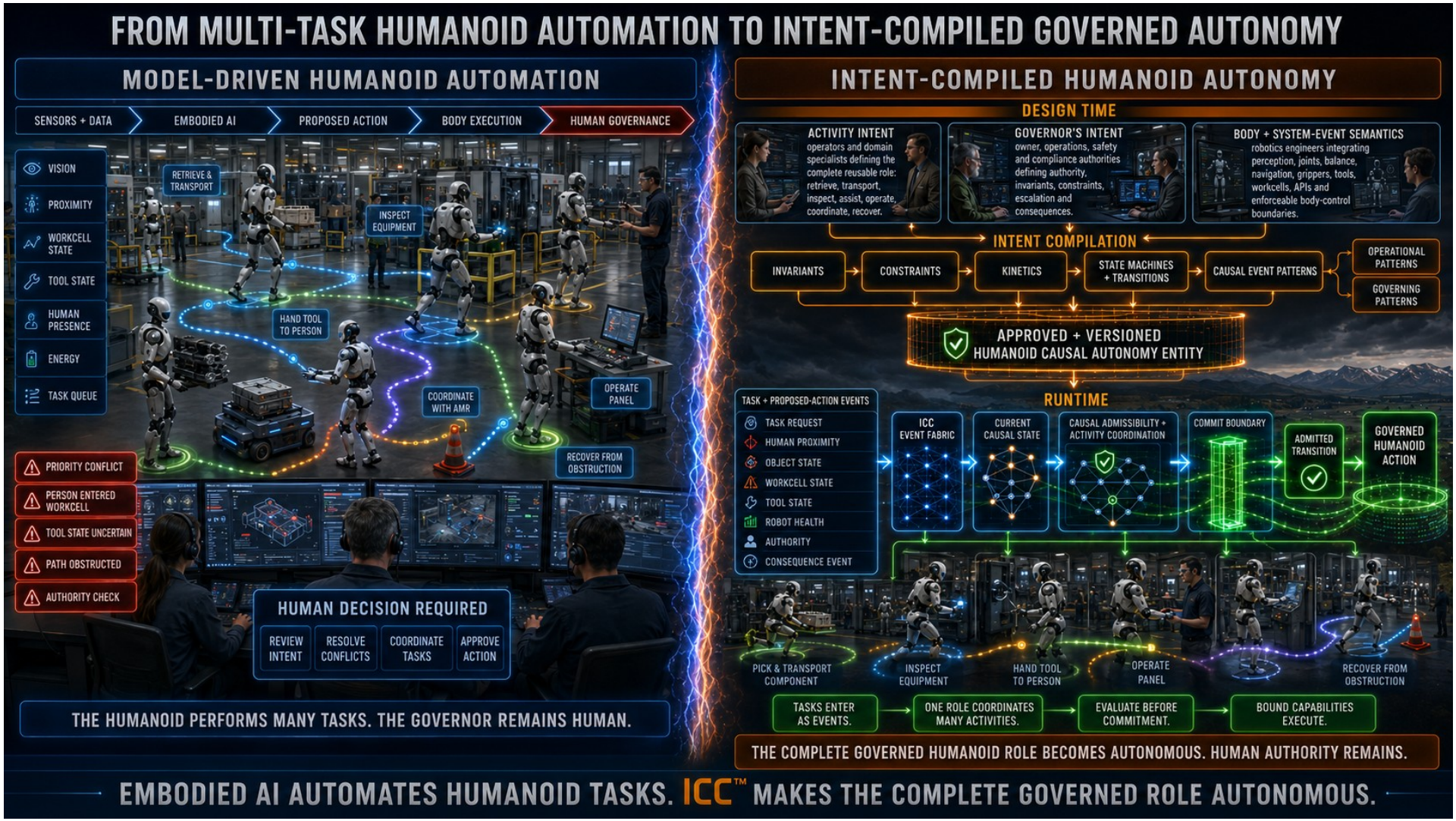

Figure B1. Embodied AI automates perception, reasoning and task performance. ICC™ compiles the complete humanoid role so changing task requests can be performed autonomously under persistent human authority.

Appendix B. Multi-task humanoid autonomy

A humanoid is a general physical actor rather than a single-purpose machine. The same body may retrieve and transport material, inspect equipment, operate a panel, use a tool, assist a person, coordinate with other systems and recover from an obstruction. Embodied AI supplies the perception, reasoning, planning and proposed movements needed to perform this varied work. ICC supplies the persistent governing architecture through which those activities become admissible physical action.

One role across many changing tasks

The reusable humanoid role defines how the machine performs classes of activity, not merely how it completes one request. It captures triggers, sequences, dependencies, concurrent activity, object handling, human interaction, exceptions and recovery. A specific task—retrieve this component, inspect this machine or assist this technician—arrives at runtime as an authority-scoped event that activates the already approved role.

This separation is essential. Task intent requests an outcome. Activity intent defines how the humanoid role works. Governor's intent defines under whose authority and under which conditions body activity may become real. A task therefore activates capability without becoming governing authority.

Three approved compilation inputs

1. **Activity intent.** Operators and domain specialists define retrieve, transport, inspect, assist, operate, coordinate and recover activities, including their sequencing, dependencies, exceptions and recovery pathways.
2. **Governor's intent.** Owners, operations leaders, safety authorities and compliance authorities define who may request activity, which invariants and constraints apply, how human proximity changes admissibility, and when activity must hold, redirect, escalate or revoke.
3. **Body and system-event semantics.** Robotics engineers validate perception, balance, joints, navigation, grippers, tools, workcells, APIs, robot-health events and the body-control interfaces through which admitted transitions can be enforced.

Compilation transforms these inputs into operational and governing causal event patterns supported by invariants, constraints, kinetics, state machines and state transitions. The approved, versioned humanoid causal autonomy entity carries the reusable role, governing authority, system bindings, provenance and commit controls into runtime independently of any fresh model output.

HUMANOID AUTONOMY PROPOSITION **A humanoid becomes governed autonomous when one approved role can accept changing tasks while preserving the operational and governing patterns that control every consequential body transition.**

Appendix B. Runtime governance of embodied activity

At runtime, task requests, model-generated proposals and proposed body actions enter the ICC event fabric together with live vision, proximity, object-state, workcell-state, tool-state, energy, robot-health, human-presence, authority and consequence events. The embodied model remains a valuable proposer and perception source. The approved causal entity remains the runtime source of governing authority.

Coordinating the complete activity

Current causal state integrates what the humanoid is doing, which task and authority scope are active, where people and objects are located, which tools and systems are available, which dependencies are satisfied and which consequences have already occurred. Operational patterns coordinate movement and task progression. Governing patterns determine whether each proposed transition is admissible within that complete situation.

Causal admissibility and activity coordination allow the runtime to admit, hold, sequence, synchronize, redirect, escalate, recover or revoke activity. A blocked path can activate an approved recovery pattern. A person entering a workcell can change the admissible state space. A tool-state discrepancy can hold commitment and initiate verification. The role adapts through compiled causal structure rather than through continuous human reconstruction.

The body as a bound capability

The commit boundary separates a proposed movement from a physical transition. Only an admitted transition reaches validated body and system capabilities such as locomotion, balance, grasping, tool use, panel operation or machine interaction. This preserves a direct computational relationship among human authority, the current causal situation and the movement that becomes real.

Consequences and continuous governing activity

Observed movement, object state, human response, task progress, deviation, failure and recovery return as events. Consequence-validation patterns compare observed effects with compiled expectations and activate the appropriate continuation or governing response. The causal trace preserves the path from approved role and runtime task through admissibility, commitment and physical effect.

Replication and scale

The same approved role can be instantiated across compatible humanoids, facilities and assignments while each instance receives its own authority-scoped tasks and live events. Organizations can therefore scale a governed worker role across large populations of machines without multiplying continuous human supervision. Human operational knowledge and governing authority remain embedded in every approved instance.

OUTCOME **Embodied AI performs many tasks. ICC™ makes the complete role autonomous by governing task activation, activity coordination, body commitment and observed consequence as one continuous causal lifecycle.**

APPENDIX C | SOVEREIGN AUTONOMY

Existing national intelligence remains distributed. Sovereign intent becomes persistent and computational across the nation.

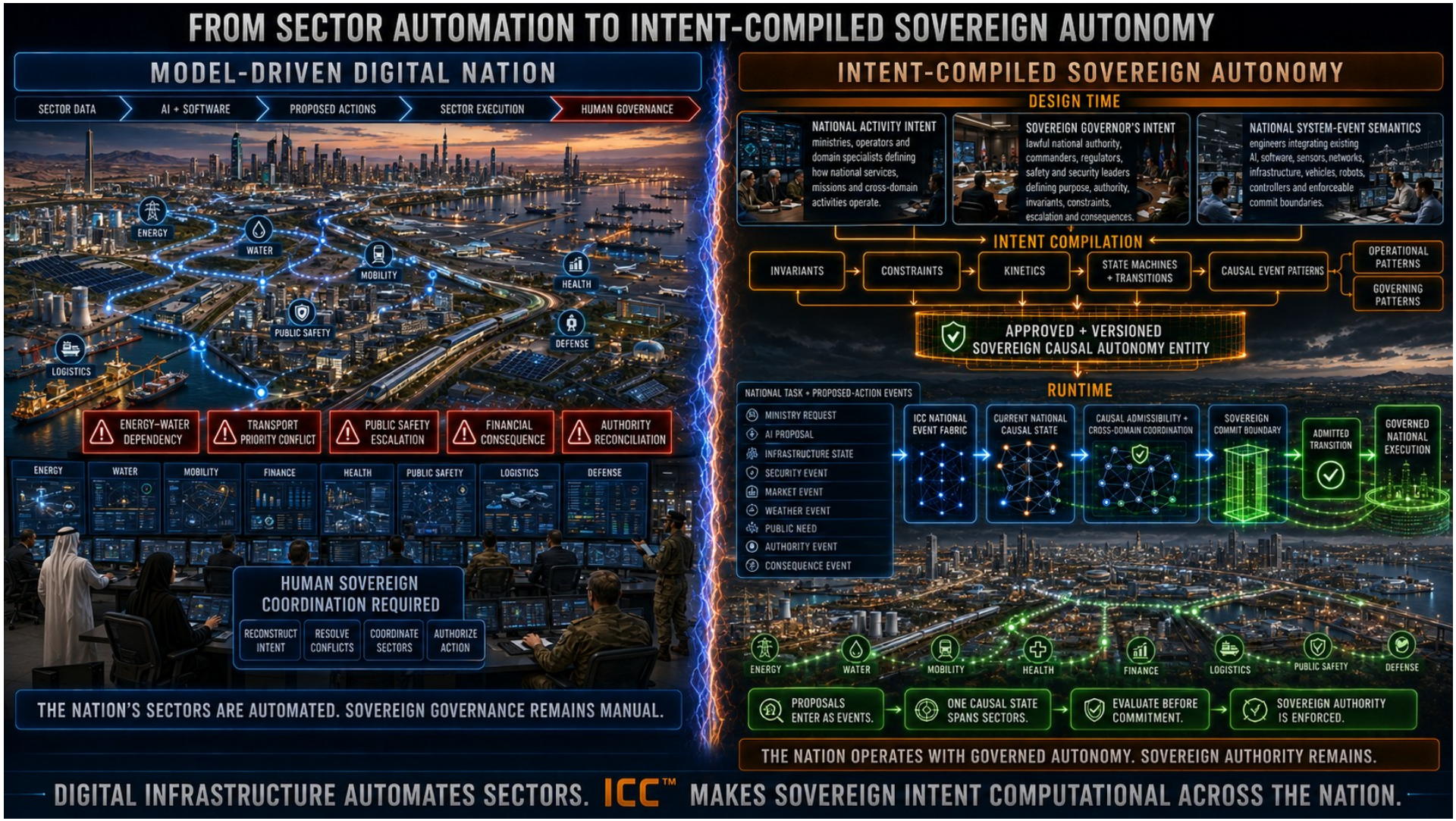

Figure C1. A sovereign nation achieves governed autonomy by compiling national activity intent, lawful governing authority and existing system-event semantics into one architecture that evaluates cross-domain actions before commitment.

Appendix C. Intent-compiled sovereign autonomy

A modern nation is already a system of systems containing millions of digital actors. Ministries, utilities, financial platforms, transportation networks, healthcare systems, public-safety systems, logistics networks, industrial systems and defense platforms continuously perceive conditions, process information and initiate activity. Sovereign autonomy is achieved when the nation's lawful intent and authority remain computationally present as these distributed actors operate together.

Distributed intelligence under persistent sovereign authority

ICC governs existing national systems while sector intelligence remains distributed. Each sector retains its existing applications, algorithms, agents, control systems and physical infrastructure. Their proposed consequential actions and live events participate in one governed event fabric. The sovereign architecture supplies the persistent authority, causal integration and commitment discipline that spans those systems.

The national role includes both sector activity and cross-domain coordination. Energy affects water; transport affects public safety and logistics; financial activity affects commerce and national resilience; communications affect healthcare, emergency response and defense. A current national causal state makes these relationships computationally available when a proposed action is evaluated.

Three approved compilation inputs

1. **National activity intent.** Ministries, operators and domain specialists define how public services, national missions and cross-domain activities operate, coordinate, recover and sustain continuity.
2. **Sovereign governor's intent.** Lawful national authorities, commanders, regulators, safety leaders and security leaders define national purpose, authority, invariants, constraints, escalation, protected interests and consequence boundaries.
3. **National system-event semantics.** Engineers validate the event contracts, states, sensors, networks, APIs, controllers, vehicles, robots, infrastructure capabilities and enforceable commitment interfaces already present across the nation.

Intent compilation transforms these sources into approved operational and governing causal event patterns. The resulting versioned sovereign causal autonomy entity preserves national intent, lawful authority, system bindings, provenance and commitment controls independently of changes in models, vendors or local applications.

SOVEREIGN AUTONOMY PROPOSITION **National intelligence may remain distributed across sectors and suppliers. The approved sovereign causal entity remains the portable, persistent and executable expression of the nation's own authority.**

Appendix C. Runtime governance across the nation

At runtime, ministry requests, model and agent proposals, infrastructure conditions, security events, market events, weather, public needs, authority changes and observed consequences enter the ICC national event fabric. The runtime preserves attribution, authority scope, causal relationships and concurrency while maintaining the current national causal state.

Cross-domain causal coordination

Approved operational patterns coordinate the activities required to deliver national services and missions. Approved governing patterns determine which transitions remain admissible under active authority, current dependencies and anticipated consequences. The runtime can sequence, synchronize, redirect, escalate, recover or revoke activity across domains while lawful national authority remains explicit and distributed across the governing architecture.

This architecture allows energy, water, mobility, finance, health, logistics, public safety and defense to retain their specialized intelligence while participating in a shared causal situation. Each sector remains independently realizable; sovereign intent provides the computational governing relationship among them.

Sovereign commit boundaries

A sovereign commit boundary is the enforceable position immediately before a consequential transition reaches a national capability. That capability may be a utility control, transportation command, financial commitment, emergency action, data release, infrastructure transition or mission-system action. Only an admitted transition proceeds to the validated bound interface.

Consequence validation and national evidence

Observed effects return to the event fabric and evolve national causal state. Consequence-validation patterns compare those observations with approved expectations and initiate any required continuation, recovery, escalation or revocation. The sovereign intent-to-execution causal trace connects lawful authority and national intent to the action that occurred and the consequence that followed.

Strategic consequence

A nation can license models, replace technology suppliers and modernize sector platforms while retaining ownership of its governing architecture. Sovereign intent becomes portable across models, systems and infrastructure because it is carried by an approved national causal entity independently of any individual model, application or technology provider.

OUTCOME **Digital infrastructure automates national sectors. ICC™ makes sovereign intent computational across them, enabling governed national autonomy while lawful human authority remains the source of every approved governing version.**