GOVERNED AUTONOMY

IC-POAR: From intended behavior to an accountable running product

A paper in the Governed Autonomy collection

Andrew Durden
Independent research
Publication edition 2026.10.6

Research on authority, evidence, and continuity in autonomous systems.

Content identifier: 058a680e5b3eee8e57a28486

All rights reserved · Research exposition

Contents

IC-POAR: From intended behavior to an accountable running product 3

Abstract . . . 3

1. The gap between generated parts and a running product . . . 3

2. Intended behavior, realization, and observation . . . 4

3. Interpretation does not confer authority . . . 5

4. Discovering requirements without inventing intent . . . 5

5. Experience is part of behavior . . . 6

6. Continuing through implementation change . . . 7

7. Collaboration and verification . . . 7

8. Four illustrative cases . . . 8

9. Evidence boundaries and open questions . . . 9

Source guide . . . 9

All rights reserved · Research exposition

IC-POAR: From intended behavior to an accountable running product

Governed Autonomy · Publication 2026.10.6
Research study: conceptual exposition of the Intent Compiler and Projected Operating Artifact Runtime architecture
Source: Founding Architecture Corpus v0.1.0, 14 July 2026

Abstract

The Intent Compiler and Projected Operating Artifact Runtime (IC-POAR) investigates a different unit of software authorship: the intended behavior of a product, together with the authority, experience, state, and evidence needed to sustain that behavior. Its proposed output is a running product whose implementation can change while its intended meaning remains identifiable and reviewable. Source code remains important, but a particular codebase is one realization of that meaning.

The architecture addresses a recurring gap in automated development. A system can generate plausible interfaces and services without establishing that they form a complete product, that their controls reach authorized effects, or that the product can continue through interruption and change. IC-POAR makes these relationships part of the intended outcome. It also distinguishes interpreting a request from having authority to enact an interpretation, and distinguishes reconstructing software resources from repeating consequential actions in the outside world.

This paper develops the proposal through definitions, experience and authority boundaries, and four illustrative cases. The founding corpus specifies requirements and a reference trace whose observations and receipts are illustrative; implementation qualification remains to be demonstrated. The central research questions concern faithful interpretation, discovery of missing requirements, affordable verification, human comprehension, and continuity through changes in implementation.

1. The gap between generated parts and a running product

A product request describes an outcome in the user's world. A user may expect work to continue after an application closes, a decision to remain subject to approval, or a report to remain retrievable after a service is replaced. Those expectations concern behavior through time. Producing a screen and a service that look appropriate does not establish that the expectations are satisfied.

Some of the missing work is connective. A visible control needs an operation with an owner, authority, state, failure handling, and an observable result. A background activity needs continuity beyond the interface that started it. An update needs a way to preserve existing records and in-progress work. A polished success display needs evidence that the relevant outcome actually occurred. These are parts of the product rather than optional explanations attached after construction.

All rights reserved · Research exposition

IC-POAR proposes that such obligations belong in the maintained account of intended behavior. They should survive changes in framework, service boundaries, storage, and deployment. This does not mean that a product description can contain all knowledge in advance. It means that missing knowledge should remain visible as missing, and that decisions made during implementation should remain connected to the requirements they serve.

The proposal's ambition is therefore broader than producing code from a prompt. It studies whether intended meaning can remain the continuing basis of construction, activation, observation, and change. The collection thesis explains the surrounding account of governed autonomy, while the architecture study places this proposal among other work on authority and continuity. Source guide: foundations.

2. Intended behavior, realization, and observation

Three concepts organize the argument. **Intended behavior** describes what a product is required, permitted, or prohibited from doing in a declared context. A **realization** is a particular arrangement of software and resources intended to deliver that behavior. An **observation** records something about what happened in an operating environment.

Their relationships matter because none can simply replace the others. An existing implementation is evidence of how the product currently works; its existence does not settle how the product ought to work. An observation can show that an expectation is unmet without granting permission to lower the expectation. An approved requirement can establish what should happen without proving that the current realization achieves it.

An **operating projection**, in the source's terminology, is a concrete realization for an environment, including the relevant executable parts, state relationships, operational controls, and evidence. The word projection does not imply a disposable imitation. The intended result is a real operating product. It emphasizes that the same product meaning may have more than one acceptable implementation, each subject to its own constraints.

Continuity also requires identity. A product capability can remain the same continuing responsibility when its name or implementation changes. A data record can remain the same subject when its storage representation changes. Equal bytes do not establish that two subjects have the same authority or history. Conversely, changed bytes do not necessarily imply a new product responsibility.

The source's use of “canonical” is deliberately limited. It means that the current account has been resolved under declared rules and authority. It does not mean that the account is complete, infallible, or free of unknowns. This distinction prevents formal organization from becoming a claim of universal understanding. Source guide: semantic model.

All rights reserved · Research exposition

3. Interpretation does not confer authority

Human intentions reach a development system through mixed material: conversation, written requirements, prototypes, screenshots, existing code, observed behavior, and institutional policy. These materials can be helpful without having the same status. A screenshot may establish an approved arrangement of visible elements while saying little about persistence. Existing code may reveal a compatibility constraint while also containing accidental behavior. A model may propose a reasonable interpretation while lacking authority to decide that it is the intended one.

IC-POAR treats this distinction as a governing boundary. Interpretation produces candidates for acceptance. Authority depends on the subject, scope, context, and decision being made. A person responsible for appearance need not be able to authorize a new external action. An operator's emergency intervention need not amend the product's long-term requirements. A confident model explanation does not settle a conflict among competent decision makers.

History matters here. A later statement can clarify, narrow, supersede, or temporarily override an earlier one. Recency alone cannot determine which relationship applies. Nor can a later correction automatically provide authority for an action already performed. The record needs to preserve what was accepted at the time and what is now understood about that period.

The practical challenge is to keep this account useful to people. Ordinary technical choices should be resolved within existing authority. Decisions that materially alter the outcome, data use, risk, experience, or permitted consequences need appropriate human judgment. Determining that boundary reliably is itself a research problem. The proposal does not assume that asking more questions automatically improves governance; unnecessary questions can obscure the consequential ones. Source guide: authority and decisions.

4. Discovering requirements without inventing intent

Product requests leave many implications unstated. If work must survive an interface closing, its continuation cannot depend solely on that interface remaining alive. If an action requires approval, the approved action must remain identifiable when it is carried out. If users must be able to recover their records after an update, continuity of those records becomes part of the update's obligations.

IC-POAR calls the accounting for explicit and derived requirements **closure**. An obligation is a requirement that must receive an adequate disposition before the relevant outcome can be accepted. The concept is useful because it directs attention to missing connections and consequences, rather than treating a list of completed components as sufficient.

Derivation nevertheless has a danger: a familiar implementation pattern can be mistaken for a logical necessity. A promise of durable work may require durable responsibility and recovery, but it does not by itself select a particular vendor or framework. A successful pattern in one environment may be unsuitable in another. The source distinguishes requirements from preferences, assumptions, and proposed hypotheses to keep these differences reviewable.

All rights reserved · Research exposition

The scope of completeness must remain bounded. No finite catalogue can be presumed to discover every requirement in every domain. A system may account completely for the obligations it has represented while still missing an important implication of the original intention. The research agenda therefore includes independent evaluation of missed requirements and unsound derivations, not only checking whether a generated plan satisfies its own description.

This is one of the proposal's central tensions. Greater automation depends on deriving more of the necessary work. Greater accountability depends on refusing to silently turn uncertain interpretation into authoritative meaning. A practical system must improve both together. Source guide: requirements and construction.

5. Experience is part of behavior

The experience of a product includes what people can perceive, understand, navigate, operate, and recover from. It includes focus, keyboard access, responsive behavior, content, motion, status, and the availability of actions under different conditions. A screenshot covers only part of this subject.

IC-POAR's **experiential contract** expresses the approved aspects of that experience and connects them to product behavior. A control has a purpose beyond responding to a click. A completion message makes a claim about an outcome. A loading state, interrupted connection, empty workspace, expired session, or unavailable action affects what the user can do next. These relationships should remain coherent when the implementation changes.

Prototypes create a particular interpretation problem. A prototype may use sample data, timers, local state, or scripted feedback to demonstrate an interaction. Approval of the experience does not automatically approve those mechanisms as the production behavior. At the same time, replacing them with real operations must preserve the aspects of the experience that were actually approved. Functional completeness and experiential fidelity constrain the same product.

Accessibility makes the limitation of visual similarity especially clear. Two screens can look alike while differing in keyboard operation, reading order, focus recovery, or the information available to assistive technology. Conversely, two platform-specific controls may look different while providing an appropriately equivalent action. Acceptable variation depends on the declared experience and the people using it.

Status also has meaning. An estimate can be useful if presented as an estimate. A pending request can be useful if its completion remains distinguishable. When observations become stale, continued visual motion should not imply that fresh progress has been observed. The source seeks evidence-connected presentation while leaving the effectiveness of its proposed verification approach to be established through implementation and evaluation. Source guide: experience.

All rights reserved · Research exposition

6. Continuing through implementation change

If intended behavior is to outlast a particular implementation, change must preserve more than a feature list. Existing records, authority, in-progress activity, and previous consequential actions must remain intelligible. A replacement service that starts correctly can still be incompatible with data written by its predecessor. Routing users back to an older version can still be unsafe after a newer version has changed that data.

The architecture therefore treats continuity as a property to examine during change. State has a continuing semantic subject; storage is one binding of that subject. A migration must account for what it preserves, transforms, or loses. Information loss and irreversible change require explicit treatment. Operational rollback, data repair, and a decision to reverse intended behavior are different actions, even when an incident makes them closely related.

External effects impose an additional boundary. Recreating an owned software resource is different from repeating an email, payment, deletion, or physical command. A historical record may show that an action was attempted while leaving its result uncertain. Restoring a software configuration does not restore the outside world, and absence from a local view does not prove that the action never happened.

Recovery consequently depends on what can be observed and on the actual guarantees of the external destination. A safely repeatable operation may permit another attempt under its conditions. An uncertain irreversible action may require further reconciliation or a decision that the system cannot make alone. Compensation is a new action with its own consequences, rather than erasure of the earlier one.

The proposal also considers partial connectivity. Disconnected work may continue within a bounded authority and information context. It cannot receive immediate revocations or assume that remote conditions stayed unchanged. The eventual reconciliation must respect those limitations. These are requirements of the proposed architecture, not operational guarantees demonstrated by this paper. Source guide: continuity.

7. Collaboration and verification

Parallel contributors can produce changes that are individually sensible and jointly incompatible. Their files may not overlap while their assumptions about ownership, authority, timing, or permitted behavior conflict. Agreement on text is therefore insufficient to establish agreement on a product's meaning.

IC-POAR studies contributions as proposed changes to that meaning. A contribution has an intended objective and a basis that can become stale. Updating it for a changed environment should preserve the objective or make the need for reinterpretation explicit. Historical reputation can inform review effort, but it cannot replace evidence or confer authority over a different subject. Useful incomplete work may remain useful without being treated as a completed capability.

Verification then asks what evidence is adequate for the particular obligation. Tests, bounded analysis, mathematical proofs, accessibility checks, operational observations, and human acceptance support different claims. A strong result for one component does not

All rights reserved · Research exposition

make every dependency equally strong. A signed receipt identifies a statement and its provenance; it does not make the statement universally correct.

The proposed certification claim is correspondingly conditional. It concerns stated inputs, assumptions, methods, bounds, and an environment. Evidence can lose present applicability when one of those conditions changes while remaining part of the historical record. Reusing it requires support for continued applicability, rather than matching the name of an earlier test.

Verification must also examine prohibited behavior. A test run that never encountered an unauthorized action does not necessarily establish that the action is unreachable. The strength of an absence claim depends on the space examined and the method used. Similarly, a system that passes narrow tests may still exploit gaps in what those tests express. Independent review, counterexamples, and evaluation against intended meaning remain essential research concerns. Source guide: contributions and assurance.

8. Four illustrative cases

Four hypothetical cases illustrate the relationship between intended behavior and its realization. They are examples for analysis rather than reports of executed systems.

A continuing research task. A user requests a research task that must continue after the interface closes and remain available when the user returns. A complete realization would preserve the work's identity, authorized scope, results, and recovery behavior beyond the initiating interface. Returning to an attractive dashboard alone would not establish success. The relevant success case is continued authorized work followed by an accessible result whose displayed status has an adequate basis. The example leaves the implementation choice open.

A useful interpretation is denied authority. A proposed enhancement would automatically send the completed research result to a wider audience. The enhancement may appear convenient and technically feasible, but it changes the consequence of the user's request. The system can retain the proposal for review without treating it as permission to send. Work already authorized can continue, and the result can be made available through its approved route. The boundary concerns the additional action, rather than the value of the suggestion.

A replacement preserves the user's records. A product moves to a different implementation while users have existing records and ongoing activity. The intended outcome is continuity of the subjects and their required behavior. A successful installation is insufficient if records disappear, authority changes silently, or earlier actions are repeated. If the transition encounters a recoverable inconsistency, the system needs an authorized continuation that respects the actual state. Whether that continuation is a compatible rollback or forward repair depends on what has already changed.

An apparent success lacks evidence. A prototype shows a completed operation after a short animation. During construction, the animation is replaced by an actual service call, but the interface still announces completion before the outcome is observed. Visual similarity has preserved the wrong aspect of the prototype. The appropriate correction connects

All rights reserved · Research exposition

presentation to the intended operation's status and gives the user a usable experience while completion remains pending or uncertain. This illustrates why approved appearance and approved behavior must be evaluated together.

9. Evidence boundaries and open questions

The founding corpus is an architectural specification and research programme. Its reference trace follows an intended product through meaning, construction, evidence, and operation. The example digests, signatures, observations, and receipts illustrate the proposed form; they are not records of an executed deployment. Proposed integration with another system similarly requires evidence about that system and the resulting operation.

The corpus classifies design problems as buildable, research-assisted, or open in their universal form. These classifications concern the availability of possible mechanisms. The research thresholds are proposed evaluation criteria. Compiler performance, independent conformance, live activation, migration behavior, and human comprehension still need to be measured in an identified implementation.

The threat assumptions are substantial. Sources and models can be malicious; generators and verifiers can be defective; authority roots can be compromised; observations can be forged or incomplete; external providers can change behavior. A design that separates these roles still needs evidence that its boundaries are enforced. Shared failure causes can also weaken the apparent independence of multiple checks.

Several questions are especially consequential. Can people approve the meaningful consequences of a change more accurately than they can review conventional specifications and source? Can a system preserve materially different interpretations without overwhelming either the user or the computation? Can it discover missing requirements without treating preferred architecture as necessity? Can verification remain affordable as behavior spans interfaces, services, state, and external effects? Can localized updates reuse unaffected evidence without overlooking hidden dependencies?

The proposal draws on compiler verification, model-driven engineering, capability security, incremental computation, provenance, declarative control, and accessibility practice. Its contribution is the proposed relationship among lasting product meaning, authorized change, complete operation, and evidence of consequence. Bounded implementations and independent evaluation would test whether that relationship improves outcomes and makes consequential decisions easier to understand. The evidence guide describes the standards appropriate to those claims.

Source guide

All references below are to *Intent Compiler and Projected Operating Artifact Runtime: Founding Architecture Corpus v0.1.0*, dated 14 July 2026. They identify the basis of the conceptual argument; proprietary construction procedures are excluded. Shared terms appear in the glossary.

Foundations. *Foundational System Definition*, §§A.1–A.5, A.9, and A.14–A.15; *Complete Problem Ledger*, “Classification,” “Dependency interpretation,” and “Ledger governance.”

All rights reserved · Research exposition

Semantic model. *Canonical Semantic Model*, §§C.1–C.5, C.10–C.15, C.19–C.22, and C.27.

Authority and decisions. *Foundational System Definition*, §A.11; *Canonical Semantic Model*, §§C.16–C.19; *Research and Closure Agenda*, §§M.8 and M.18.

Requirements and construction. *Compiler Architecture*, §§D.1, D.3–D.8, and D.10–D.13; *Research and Closure Agenda*, §§M.3–M.7 and M.11–M.15.

Experience. *Experiential Contract Architecture*, §§E.1–E.3, E.9–E.22; *Verification and Proof System*, §H.5.

Continuity. *Projection Runtime*, §§G.3, G.6–G.12, and G.17–G.21; *Research and Closure Agenda*, §§M.15–M.17 and M.21–M.23.

Contributions and assurance. *Multi-Agent Contribution Protocol*, §§F.1, F.4–F.10, and F.12–F.15; *Verification and Proof System*, §§H.1–H.4 and H.9–H.18; *Security and Threat Model*, §§I.1–I.5 and I.13; *Reference Compilation Trace*, §L.1; *Codebase-Agent Handoffs*, opening evidence statement; *Primary Sources and Inherited Mechanisms*, opening scope statement and §§R.1–R.7.

All rights reserved · Research exposition