OTANIS-USG

A Unified Architectural Governance Framework for Transitional, Reversal, and Irreversible Workflows in Bounded Agentic Systems

Masayuki Otani
Architectural Governance
United Kingdom
www.architecturalgovernance.com
info@architecturalgovernance.com

12 April 2026

Abstract

OTANIS defines a declared-boundary execution-governance architecture for irreversible execution-bearing action classes whose realised commit paths are interceptable, mediated, and non-bypass. ISDAIRE determines whether such classes are admissible ex ante, ARETABA specifies how runtime authority is constructed and enforced at the bound earliest governed irreversible execution boundary $T_e^*(a)$, and GAG and MGAG preserve legitimacy across composition and multi-layer systems [1–4]. What remains under-specified is how one bounded agentic system should be governed when its governed workflow family is not uniform, but instead consists of transitional workflows, reversal workflows, and irreversible workflows that must coexist coherently inside one architecture and, in some cases, across multiple already-governed systems.

This paper addresses that narrower problem. It introduces OTANIS-USG as a unified architectural governance framework for bounded agentic systems whose governed workflow family consists of transitional, reversal, and irreversible workflows. The framework reuses common boundary, authority, provenance, traceability, version, refusal, and escalation substrates across all governed workflow classes, while resolving a governance architecture for each class on the reviewed scope through a formal evidence-adjudication and architecture-resolution protocol. Only irreversible workflows require full OTANIS execution-boundary enforcement. Transitional workflows are governed at a declared reliance boundary $T_r^*(a)$. Reversal workflows are governed as new declared action classes linked to prior governed paths through a declared origin relation and a reversal boundary $T_{\text{rev}}^*(a)$.

The paper further defines a unified review object, class-level admissibility conditions for transitional, reversal, and irreversible workflows, a deliberately minimal ex ante contract for reversal governance, compensatable and non-compensatable reversal semantics, ordering rules between an original governed path and its reversal workflow, and a distinction between legitimate governed reversal, malformed reversal claims, and declared operational repair. It also introduces a promotion rule under which transitional or reversal paths that become effect-bearing at a narrower declared scope must be reclassified into predefined OTANIS-governed irreversible workflows. The paper further clarifies the relationship between internal workflow sequencing and formal orchestration. OTANIS-USG governs the mixed workflow family inside one bounded system. When a business action spans multiple already-governed bounded systems, the orchestration layer remains the formal composite coordination discipline above them. The paper does not claim universal applicability to open-ended environments, automatic discovery of omitted boundaries, or semantic sufficiency of undeclared reversal logic. Its guarantees remain architectural and conditional. Where workflow classes, origin links, boundary sets, authority surfaces, provenance commitments, and commit paths are declared

correctly, OTANIS-USG provides a more defensible basis for design review, procurement qualification, deployment pressure testing, and forensic reconstruction of mixed agentic systems.

Keywords: Agentic AI, OTANIS, unified systems governance, transitional workflows, reversal workflows, execution-time governance, workflow promotion, orchestration, auditability.

1 Introduction

As AI systems move from bounded assistance into execution-bearing operational architectures, the governance problem becomes more differentiated rather than more uniform. Some governed workflows prepare or constrain later action without yet committing an irreversible external effect. Some governed workflows exist to negate, compensate, rescind, reopen, or otherwise reverse the effects of an earlier governed path. Some governed workflows directly cross into irreversible external effect. Treating all of these as one undifferentiated class produces weak architecture. Treating all of them as if they were direct irreversible commits is also weak, because it obscures where full OTANIS execution-boundary enforcement is actually required and where a different, narrower governance architecture is the correct one.

OTANIS already answers the irreversible part of this problem. It states that if a system is allowed to produce irreversible external effects, legitimate authority must remain executable at the bound earliest governed irreversible execution boundary rather than only in policy text, workflow diagrams, or post-hoc explanation [1, 2]. ISDAIRE answers the ex-ante admissibility question for governed action classes. ARETABA answers the narrower runtime- authority construction question. GAG and MGAG answer the compositional and multi-layer legitimacy question [3, 4]. The orchestration extension then answers a further question, namely how a single composite business action may remain governable when it spans multiple already-governed OTANIS instances [5].

What the current formal family has not yet isolated clearly enough is the unified workflow question that arises when one bounded agentic architecture contains all three of the following.

1. transitional workflows that prepare, constrain, parameterise, queue, prioritise, or otherwise materially condition later governed action,
2. reversal workflows whose explicit purpose is to negate, compensate, rescind, revoke, reopen, unwind, or restore part of the state, obligation, entitlement, or exposure created by a prior governed path,
3. irreversible workflows whose realised commit paths require full OTANIS control at $T_e^*(a)$.

That distinction matters operationally. Many real systems fail not because the final execution step lacked a gate, but because a preparatory artefact had already narrowed the decision space, bound the payload, selected the corridor, queued the action, or created a presumption of continuation long before final commit. Many other systems fail because a later remediation or correction step is casually described as an "undo" even though it is in fact a new path with its own authority, ordering, traceability, and possibly its own irreversible commitment. In both cases, the failure is

not only a model-quality issue. It is an architectural governance failure. This is structurally consistent with wider systems and safety literature, which repeatedly shows that serious failures often arise from coupling, control-loop weakness, stale state, workflow interaction, and boundary misidentification rather than from one isolated component behaving "incorrectly" in a narrow local sense [8, 13, 15, 18]. It is also consistent with established access-control literature that treats protected operations, authority scope, and delegation structure as first-class architectural concerns rather than as post-hoc explanation [16, 17].

This paper addresses that narrower unified-architecture problem. OTANIS-USG is introduced as a unified architectural governance framework for bounded agentic systems whose governed workflow family consists of transitional, reversal, and irreversible workflows. Purely advisory behaviour may still exist in such systems, but it is outside the governed workflow family unless it conditions later governed action, in which case it must be reclassified.

The contribution claimed here is limited and precise.

1. It formalises a governed workflow family consisting of transitional, reversal, and irreversible workflow classes.
2. It introduces a workflow-classification adjudication and architecture-resolution protocol that resolves a governance architecture for the reviewed scope as transitional governance, reversal governance, or full OTANIS.
3. It reuses common boundary, authority, provenance, traceability, version, refusal, and escalation substrates across all three workflow classes.
4. It defines a declared reversal action class together with reversal admissibility conditions, reversal boundary selection, reversal authority and provenance requirements, and ordering rules relative to the original governed path, while keeping reversal ex ante definition intentionally structurally minimal rather than semantically closed.
5. It introduces a promotion rule under which transitional or reversal paths that become effect-bearing at a narrower declared scope must be reclassified into predefined OTANIS-governed irreversible workflows for that scope.
6. It distinguishes compensatable from non-compensatable reversal semantics and states when a supposed reversal is actually only an operational repair step outside the governed claim surface.
7. It clarifies how one or more bounded unified systems may participate in broader composed actions under the orchestration, GAG, and MGAG layers without weakening local OTANIS obligations.

The paper does not claim a universal theory of all enterprise workflows, nor does it claim that classification alone makes a deployment safe. It claims something narrower. Where workflow classes, origin links, boundary sets, authority surfaces, provenance commitments, and commit paths are declared correctly, OTANIS-USG provides a unified architectural basis for reviewing, testing, and auditing mixed agentic systems without weakening OTANIS where irreversible

commitment actually occurs. That is the theorem surface of the paper.

2 What This Paper Contributes as an Architectural Framework

This paper should be read as a unified architectural governance framework inside the OTANIS family, not as a new umbrella architecture that displaces ISDAIRE, ARETABA, OTANIS, GAG, MGAG, or the orchestration extension.

Its purpose is to answer a narrower question.

> When a bounded agentic system contains transitional, reversal, and irreversible workflows in one architecture, under what conditions can the system resolve a justified governance architecture for each workflow on the reviewed scope while preserving consistency, composability, and auditability across the whole governed path?

The answer developed here has seven parts.

First, the paper treats workflow classification as a governed architectural question rather than a loose product label. Transitional, reversal, and irreversible are not commercial categories. They are formal governance categories tied to declared effect semantics, declared boundaries, and declared authority requirements.

Second, the paper introduces a workflow-classification adjudication and architecture-resolution protocol. A workflow is not simply reviewed in the abstract. It is resolved to a governance architecture for the reviewed scope through declared evidence obligations and adjudication rules.

- Transitional workflows are governed at a reliance boundary.
- Reversal workflows are governed as new declared paths linked to prior governed action.
- Irreversible workflows remain governed by OTANIS at $T_e^*(a)$.

Third, the paper reuses common predicates and substrates across all governed workflow classes. Boundary declaration, dependency scope, authority construction, provenance, traceability, version binding, refusal, and escalation are not reinvented separately for each workflow family. They are reused where possible so that the unified system remains composable and auditable.

Fourth, the paper formalises reversal as a first-class governed workflow family. This matters because reversal is not the same as "undo". A reversal workflow is a new governed path with its own authority, traceability, provenance, ordering constraints, and possibly its own irreversible commitment. Its ex ante contract is intentionally minimal rather than semantically closed. The framework requires enough declared structure to distinguish governed reversal from ad hoc repair, but it does not pretend that real reversal logic can always be fully specified in advance.

Fifth, the paper introduces a promotion rule. Transitional and reversal workflows remain under their narrower governance models only while they do not themselves cross a declared irreversible external or precursor-irreversible surface. If a realised step does so, the path must be promoted into a predefined OTANIS-governed irreversible workflow for that narrower scope,

or else autonomous continuation is non-admissible.

Sixth, the paper distinguishes compensatable and non-compensatable reversal semantics. That distinction is load-bearing. A compensatable reversal may admit a further governed path if declared ex ante. A non-compensatable reversal must be treated much more strictly, because a later failure can leave a new orphan effect just as easily as an original irreversible workflow can.

Seventh, the paper preserves OTANIS where OTANIS belongs. Only irreversible workflows require full OTANIS execution-boundary enforcement. Transitional and reversal workflows have their own formally defined but appropriately scoped governance models. If a transitional or reversal path itself contains a new irreversible sub-step, that sub-step remains inside OTANIS at its own $T_e^*(a)$. The framework therefore does not weaken OTANIS. It tells the architecture where OTANIS must continue to apply.

3 Scope and Non-Claims

3.1 Scope

This paper applies only when all of the following hold.

1. The deployment contains a finite declared set of governed workflow classes and a finite declared set of relevant downstream couplings under review.
2. Each governed workflow class is classified as transitional, reversal, or irreversible.
3. At least one workflow class is irreversible in the OTANIS sense, or the wider system is explicitly designed to remain inside transitional and reversal governance only until a later irreversible step appears outside the current slice under review.
4. At least one workflow class is transitional or reversal.
5. The organisation seeks a unified governance architecture for mixed-consequence workflows rather than a new runtime substitute for OTANIS.

The unit of analysis is therefore not the whole system in every behavioural respect. It is the governed workflow family and the declared paths by which transitional, reversal, and irreversible workflows interact.

3.2 Out of Scope

This paper is out of scope for the following cases.

1. Open-ended environments in which relevant workflow classes, origin links, or commit paths cannot be declared with enough precision to support review.
2. Systems whose irreversible workflow classes are themselves outside the OTANIS applicability domain because no controlled, mediated, non-bypass commit path exists [1, 2].
3. Purely advisory outputs that do not condition later governed action.
4. Attempts to use USG as a substitute for execution-boundary control where a workflow class

is already irreversible.

5. Attempts to treat an operational workaround, manual remediation, or post-event business repair as a legitimate governed reversal without the required declared reversal artefacts.

3.3 Non-Claims

This paper does not claim any of the following.

1. Universal classification correctness for arbitrary real systems.
2. Automatic discovery of hidden dependencies, hidden precursor commitments, or hidden reversal targets.
3. Automatic proof that a declared reversal path is legally, economically, or operationally adequate in the real world.
4. A runtime substitute for OTANIS, ISDAIRE, or ARETABA.
5. Universal safety, correctness, legal sufficiency, or domain adequacy.

Its claims are narrower and more defensible. Where workflow classes, origin links, boundary sets, authority surfaces, provenance commitments, refusal semantics, escalation routes, and execution surfaces are declared correctly, OTANIS-USG provides a formal basis for governing a mixed workflow architecture without weakening OTANIS where irreversible commitment is actually present.

4 Relationship to ISDAIRE, ARETABA, OTANIS, GAG, MGAG, and Orchestration

This paper does not alter the division of labour already established across the OTANIS family. It adds a narrower unified-governance layer that depends on the existing layers.

- **ISDAIRE** determines whether irreversible workflow classes are admissible in principle and whether the relevant ex-ante artefacts exist at all [3].
- **ARETABA** determines how runtime authority is constructed and resolved for concrete governed action instances at the relevant boundary [4].
- **OTANIS** determines how the resolved authority object is enforced at the bound earliest governed irreversible execution boundary for irreversible workflow instances [2].
- **GAG and MGAG** preserve legitimacy across composition, delegation, and multiple governance layers [1, 2].
- **OTANIS orchestration** governs composite business actions spanning multiple already-governed OTANIS-conformant systems or multiple bounded unified systems, under declared participant, ordering, coordination, and trace-binding semantics [5].
- **OTANIS-USG** determines how one bounded system containing transitional, reversal, and irreversible workflows classifies those workflows, selects the correct governance architecture

for each, and preserves coherence across the mixed system.

This relationship can be stated directly.

Rule 1 (Irreversible workflow supremacy)**.** For any action instance a such that $\text{Irreversible}(\alpha(a)) = 1$, if local OTANIS evaluation yields $\text{Permit}(a, \beta_e(a), x_e(a)) = 0$ at $T_e^*(a)$, then no OTANIS-USG result may convert that action into a compliant irreversible commit. OTANIS-USG may classify, review, and pressure-test the surrounding mixed system, but it may not weaken the execution-boundary obligations already defined by OTANIS and ARETABA.

Rule 2 (Reversal is a new governed path)**.** A reversal workflow is not the negation of audit history and not a permission to pretend the original governed path never occurred. It is a new governed path with its own workflow class, boundary, authority surface, provenance commitments, traceability obligations, refusal semantics, escalation routes, and ordering constraints relative to the original path.

Rule 3 (Promotion to OTANIS under emergent irreversibility)**.** If a workflow instance currently governed as transitional or reversal reaches a realised step that produces a declared irreversible external effect or a declared precursor-irreversible commitment, then that narrower step may no longer remain governed only as transitional or reversal. It must be promoted into a predefined OTANIS-governed irreversible workflow for that scope, or else autonomous continuation must refuse or escalate.

Rule 4 (Internal sequencing versus formal orchestration)**.** Internal sequencing of transitional, reversal, and irreversible workflows inside one bounded unified system belongs to OTANIS-USG. Formal orchestration applies only when a composite business action spans multiple already-governed bounded systems or multiple independent OTANIS runtimes. Internal sequence is therefore not automatically equivalent to orchestration.

Rule 5 (No review substitution for absent execution control)**.** If a workflow class is irreversible but lacks a declared, interceptable, mediated, non-bypass commit path, OTANIS-USG cannot make it admissible by review quality alone. The correct response is redesign, refusal of autonomous deployment, or removal of the claim that the class is governed.

These five rules are load-bearing. Without them, a unified framework could easily collapse into either over-general architecture or weak review theatre. This paper explicitly rejects both.

5 Formal Model and Notation

5.1 System Model

Let a mixed-consequence deployment be represented as a directed graph

$$\mathcal{S} = (V, E) \tag{1}$$

where V denotes agents, tools, services, queues, operator surfaces, and commit adapters, and E denotes invocation, data-flow, or declared downstream-governance edges.

Let $\mathcal{A}$ be the finite set of declared governed workflow classes and let $\mathcal{I}$ be the set of realised action instances. Every $a \in \mathcal{I}$ is mapped to exactly one declared class

$$\alpha : \mathcal{I} \to \mathcal{A}. \tag{2}$$

Let T be a set of execution events equipped with a partial order $\prec$ representing causal precedence, consistent with the OTANIS family and with standard distributed-ordering practice [11, 12]. Let X denote the declared space of runtime selection and evaluation inputs relevant to boundary binding, promotion, and workflow-class integrity, and let Y denote the finite set of declared target effect families.

Define a class-level target mapping

$$\text{TargetClass} : \mathcal{A} \to Y \tag{3}$$

and an instance-level target mapping

$$\text{Target} : \mathcal{I} \to Y \tag{4}$$

such that

$$\text{Target}(a) = \text{TargetClass}(\alpha(a)) \tag{5}$$

for every realised instance a. A target effect family is not a claim to model a whole enterprise ontology. It is only the minimal effect identifier needed to determine whether workflow instances participate in the same materially relevant governed path.

5.2 Governed Workflow Family

The governed workflow family in this paper consists of three and only three workflow classes.

$$\text{WorkflowDecl} : \mathcal{A} \to \{\mathsf{T}, \mathsf{R}, \mathsf{E}\} \tag{6}$$

where WorkflowDecl denotes the class label declared by the architecture authors, and

$$\text{ClassEvidence} : \mathcal{A} \to \text{declared structural evidence objects} \tag{7}$$

denotes the corresponding declared evidence basis for that label.

Purely advisory behaviour may still exist in the wider system, but it is outside the governed workflow family unless it conditions later governed action.

Definition 1 (Transitional workflow class)**.** A declared workflow class α is transitional iff it does not itself commit a declared irreversible external effect, but its realised output can be relied upon later to prepare, constrain, parameterise, queue, prioritise, or otherwise materially condition a same-target governed path.

Definition 2 (Reversal workflow class)**.** A declared workflow class α is reversal iff its explicit

governed purpose is to negate, compensate, rescind, revoke, reopen, unwind, or restore part of the state, obligation, entitlement, or exposure created by a prior governed workflow instance linked through Orig.

Definition 3 (Irreversible workflow class)**.** A declared workflow class α is irreversible iff its realised commit path can produce a declared irreversible external effect on a controlled, mediated, non-bypass path and therefore remains inside OTANIS at $T_e^*(a)$.

The paper does not treat the declared class label as sufficient on its own. Instead, the label must be supported by a finite evidence-obligation protocol over a fixed normalised evidence vocabulary.

Let the admissible evidence-object types be the finite set

$$\mathcal{V}_{\mathrm{e}} = \left\{ \begin{array}{l} \mathsf{TargetSpec}, \mathsf{EffectSpec}, \mathsf{RelianceSpec}, \mathsf{OriginSpec}, \\ \mathsf{ReversalIntentSpec}, \mathsf{BoundarySpec}, \mathsf{AuthoritySpec}, \\ \mathsf{CommitSurfaceSpec}, \mathsf{NonBypassSpec}, \mathsf{SnapshotSpec}, \mathsf{TraceSpec} \end{array} \right\}. \tag{8}$$

For every declared class α, let $\mathcal{E}(\alpha)$ be the finite raw declared evidence multiset. The paper fixes the preprocessing pipeline rather than leaving it implicit.

First, schema validation is a deterministic predicate over raw evidence objects:

$$\mathrm{SchemaValid}(e) = 1 \tag{9}$$

iff evidence object e conforms to the declared class-evidence schema, carries a declared type in $\mathcal{V}_{\mathrm{e}}$, and is version-compatible with the reviewed claim.

Second, evidence identity is a deterministic canonical key:

$$\mathrm{EvidenceID}(e) \tag{10}$$

which is computed from the evidence type, class identifier, target family, reviewed source identity, and any declared step or boundary identity required by the evidence schema.

Third, reviewed-scope projection is a deterministic filter

$$\mathrm{ProjectScope}_{\alpha} : \mathcal{E}(\alpha) \to \mathcal{P}_{\mathrm{fin}}(\mathcal{E}(\alpha)) \tag{11}$$

that retains exactly those schema-valid evidence objects whose declared scope lies inside the reviewed claim slice for class α.

Fourth, duplicate collapse is deterministic over evidence identity: all projected schema-valid evidence objects with the same EvidenceID are merged only if their schema-bearing content is identical on the reviewed scope; otherwise contradiction is recorded and adjudication must fail.

Accordingly, for every declared class α, let

$$N(\alpha) \tag{12}$$

be the normalised finite evidence multiset obtained from ClassEvidence(α) by applying, in order, schema validation, reviewed-scope projection, and duplicate collapse by evidence identity. If any one of these preprocessing stages is indeterminate on the reviewed scope, class adjudication is indeterminate and the class is non-admissible for a governed claim.

For every vocabulary item $v \in \mathcal{V}_{\mathrm{e}}$, define

$$\mathrm{Present}(\alpha, v) = 1 \tag{13}$$

iff $N(\alpha)$ contains at least one reviewed evidence object of type v, and define

$$\mathrm{Consistent}(\alpha, v) = 1 \tag{14}$$

iff all reviewed evidence objects of type v in $N(\alpha)$ agree on the reviewed scope and do not contradict the declared class claim.

For each candidate class $c \in \{\mathsf{T}, \mathsf{R}, \mathsf{E}\}$, let

$$\mathrm{EvidenceReq}(c, \alpha) \tag{15}$$

be the finite set of mandatory obligation types, and let

$$\mathrm{Excl}(c, \alpha) \tag{16}$$

be the finite set of mandatory exclusion conditions that must also hold on the reviewed scope. The minimal paper-level obligation sets are fixed as follows.

$$\mathrm{EvidenceReq}(\mathsf{T}, \alpha) = \{\mathsf{TargetSpec}, \mathsf{RelianceSpec}, \mathsf{BoundarySpec}, \mathsf{SnapshotSpec}\}, \tag{17}$$

$$\mathrm{EvidenceReq}(\mathsf{R}, \alpha) = \{\mathsf{TargetSpec}, \mathsf{OriginSpec}, \mathsf{ReversalIntentSpec}, \mathsf{BoundarySpec}, \mathsf{AuthoritySpec}, \mathsf{TraceSpec}\}, \tag{18}$$

$$\mathrm{EvidenceReq}(\mathsf{E}, \alpha) = \{\mathsf{TargetSpec}, \mathsf{EffectSpec}, \mathsf{CommitSurfaceSpec}, \mathsf{NonBypassSpec}, \mathsf{BoundarySpec}, \mathsf{AuthoritySpec}\}. \tag{19}$$

The corresponding exclusion conditions are:

- for T, no reviewed evidence object may assert a declared irreversible effect or a declared controlled mediated commit surface on the same reviewed step;
- for R, no reviewed evidence object may leave the origin relation, reversal intent, or reversal target family unresolved on the reviewed scope;
- for E, no reviewed evidence object may deny the existence of a controlled, mediated, non-bypass commit surface for the claimed irreversible step.

Define the obligation-satisfaction predicate

$$\mathrm{OblSat}(\alpha, v) = 1 \tag{20}$$

iff $\mathrm{Present}(\alpha, v) = 1$ and $\mathrm{Consistent}(\alpha, v) = 1$ on the reviewed scope.

Define the adjudication predicate

$$\mathrm{Adjudicated}(\alpha, c) = 1 \tag{21}$$

iff all of the following hold.

1. for every $v \in \mathrm{EvidenceReq}(c, \alpha)$, $\mathrm{OblSat}(\alpha, v) = 1$,
2. every exclusion condition in $\mathrm{Excl}(c, \alpha)$ holds on the reviewed scope,
3. no contradiction in $N(\alpha)$ forces class c to be rejected,
4. for every $c' \neq c$, at least one obligation in $\mathrm{EvidenceReq}(c', \alpha)$ fails or an exclusion condition in $\mathrm{Excl}(c', \alpha)$ fails.

Because adjudication is defined over the same normalised reviewed evidence multiset $N(\alpha)$, and because $N(\alpha)$ is itself produced by the deterministic preprocessing pipeline just fixed, two reviewers applying this paper to the same raw declared evidence multiset, the same reviewed claim slice, the same schema-validation rule, the same evidence-identity function, and the same obligation and exclusion sets must reach the same admissibility result for each candidate class. The paper therefore claims deterministic adjudication under a fixed reviewed preprocessing context, not uniquely correct classification of a real system independent of how the reviewed slice was framed.

The support predicates are then shorthand for adjudicated support.

$$\mathrm{Supports}_{\mathsf{T}}(\alpha) = 1 \iff \mathrm{Adjudicated}(\alpha, \mathsf{T}) = 1 \tag{22}$$

$$\mathrm{Supports}_{\mathsf{R}}(\alpha) = 1 \iff \mathrm{Adjudicated}(\alpha, \mathsf{R}) = 1 \tag{23}$$

$$\mathrm{Supports}_{\mathsf{E}}(\alpha) = 1 \iff \mathrm{Adjudicated}(\alpha, \mathsf{E}) = 1. \tag{24}$$

Define the resolved workflow class by

$$\mathrm{WorkflowClass}(\alpha) = c \tag{25}$$

iff all of the following hold.

1. $\mathrm{WorkflowDecl}(\alpha) = c$,
2. $\mathrm{Adjudicated}(\alpha, c) = 1$,
3. $\mathrm{Adjudicated}(\alpha, c') = 0$ for every $c' \neq c$.

If these conditions fail, then $\mathrm{WorkflowClass}(\alpha)$ is indeterminate and the class is non-admissible for a governed conformance claim.

We write

$$\mathrm{Transitional}(\alpha) = 1 \iff \mathrm{WorkflowClass}(\alpha) = \mathsf{T}, \tag{26}$$

$$\text{Reversal}(\alpha) = 1 \iff \text{WorkflowClass}(\alpha) = \mathsf{R}, \tag{27}$$

and

$$\text{Irreversible}(\alpha) = 1 \iff \text{WorkflowClass}(\alpha) = \mathsf{E}. \tag{28}$$

5.3 Architecture Selection

OTANIS-USG introduces an architecture-selection function

$$\text{SelectArchitecture} : \mathcal{A} \to \{\mathsf{TG}, \mathsf{RG}, \mathsf{OTANIS}\} \tag{29}$$

such that

$$\text{SelectArchitecture}(\alpha) = \begin{cases} \mathsf{TG}, & \text{WorkflowClass}(\alpha) = \mathsf{T}, \\ \mathsf{RG}, & \text{WorkflowClass}(\alpha) = \mathsf{R}, \\ \mathsf{OTANIS}, & \text{WorkflowClass}(\alpha) = \mathsf{E}. \end{cases} \tag{30}$$

This function is therefore not a bare lookup over an unconstrained human label. It is defined only after declared class labels have been checked against declared structural evidence on the reviewed scope. If $\text{WorkflowClass}(\alpha)$ is indeterminate, contradictory, or unsupported, then $\text{SelectArchitecture}(\alpha)$ is indeterminate and the workflow class is non-admissible for a governed claim.

Rule 6 (Selection determinacy)**.** A governed workflow class may not claim conformance under this paper unless exactly one resolved workflow class and exactly one governing architecture are selected for that class.

5.4 Common Boundary Substrate

For every governed workflow class α, let

$$\text{ObsScope}(\alpha) \tag{31}$$

denote the declared observation scope used to project the reviewed event stream for that class, and let

$$\text{RetryCollapse}_{\alpha} \tag{32}$$

denote the declared retry-equivalence and duplicate-collapse rule used to identify repeated or observer-duplicated candidate events that represent the same reviewed step.

For every governed workflow instance a, declare a finite non-empty candidate boundary set

$$\text{Cand} : \mathcal{I} \to \mathcal{P}_{\text{fin}}(T) \tag{33}$$

and derive the reviewed candidate set

$$\widehat{\text{Cand}}(a) := \text{RetryCollapse}_{\alpha(a)}\big(\text{Cand}(a) \cap \text{ObsScope}(\alpha(a))\big). \tag{34}$$

Let $\mathrm{Order}(\alpha(a))$ be the declared reviewed ordering rule used on the projected candidate set. The paper requires $\mathrm{Order}(\alpha)$ to be a total preorder over reviewed candidate events such that, after retry collapse, it extends reviewed causal precedence on the declared observation scope. Concretely, if $t_1 \prec t_2$ on the reviewed scope and t_1, t_2 are not collapsed as the same reviewed step, then t_2 may not strictly precede t_1 under $\mathrm{Order}(\alpha)$.

Let

$$\mathrm{MinClass}(a) \tag{35}$$

denote the unique reviewed minimum equivalence class of events in $\widehat{\mathrm{Cand}}(a)$ under the declared total preorder induced by $\mathrm{Order}(\alpha(a))$, if such a unique class exists.

The reviewed minimum class is not automatically safe to treat as a single event. For every workflow class, define a class-specific boundary-equivalence predicate

$$\mathrm{BoundaryEq}_{\alpha(a)}(a, t_i, t_j) \in \{0, 1\} \tag{36}$$

for $t_i, t_j \in \mathrm{MinClass}(a)$, where $\mathrm{BoundaryEq}_{\alpha(a)}(a, t_i, t_j) = 1$ iff all of the following hold on the reviewed scope.

1. $\mathrm{SnapshotConsistent}(a, t_i)$ and $\mathrm{SnapshotConsistent}(a, t_j)$ have the same truth value under the declared snapshot profile.
2. The class-appropriate boundary outcome agrees at both events. Concretely,

$$\mathrm{BOutcome}_{\alpha(a)}(a, t_i) = \mathrm{BOutcome}_{\alpha(a)}(a, t_j),$$

   where the outcome space is $\{\mathsf{permit}, \mathsf{refuse}, \mathsf{escalate}\}$.
3. The reviewed effect typing induced by t_i and t_j is the same for the governed path, including any promotion-relevant effect typing.
4. The resulting refusal or escalation consequence required by the declared profile is the same at both events.

A minimum class is *boundary-equivalent* only if $\mathrm{BoundaryEq}_{\alpha(a)}(a, t_i, t_j) = 1$ for every pair $t_i, t_j \in \mathrm{MinClass}(a)$.

To preserve event-typed downstream semantics, every workflow class must also declare a deterministic canonical representative selector

$$\mathrm{CanonRep}_{\alpha(a)} : \mathcal{P}_{\mathrm{fin}}(T) \to T \tag{37}$$

that maps any admissible and boundary-equivalent reviewed minimum equivalence class to exactly one reviewed event inside that class. The selector must be internal to the class semantics, deterministic on the reviewed scope, and order-preserving in the following narrow sense: if two reviewed minimum classes are equal, then the same representative event is always selected, and if no unique minimum class exists or if the unique minimum class is not boundary-equivalent, no representative may be selected.

The class-appropriate governed boundary is then bound by

$$B^*(a) := \mathrm{CanonRep}_{\alpha(a)}\big(\mathrm{MinClass}(a)\big) \tag{38}$$

only if a unique reviewed minimum equivalence class exists after projection and retry collapse, that class is boundary-equivalent, and the corresponding canonical representative is defined. If the reviewed candidate set remains empty, observer-divergent, duplicate-ambiguous, without a unique reviewed minimum class, not boundary-equivalent, or without a defined canonical representative under the declared selector, then boundary resolution is indeterminate and conformance fails for that path.

The same substrate is reused across all governed workflow classes. The interpretation differs by workflow class, not the existence of the substrate itself.

$$B^*(a) = \begin{cases} T_r^*(a), & \mathrm{WorkflowClass}(\alpha(a)) = \mathsf{T}, \\ T_{\mathrm{rev}}^*(a), & \mathrm{WorkflowClass}(\alpha(a)) = \mathsf{R}, \\ T_e^*(a), & \mathrm{WorkflowClass}(\alpha(a)) = \mathsf{E}. \end{cases} \tag{39}$$

Thus the unified framework uses a common candidate-boundary, reviewed-minimum, and canonical-representative substrate while still distinguishing reliance boundaries, reversal boundaries, and irreversible execution boundaries.

5.5 Boundary Interpretation by Workflow Class

The workflow-specific meanings of the common boundary substrate are as follows.

Definition 4 (Declared reliance modes and evidence)**.** For each transitional workflow class α, let

$$\mathrm{RelyModes}(\alpha) \subseteq \{\mathsf{consume}, \mathsf{queue}, \mathsf{bind}, \mathsf{reserve}, \mathsf{admit}, \mathsf{presume}\} \tag{40}$$

be the declared reliance-mode set for that class. Let

$$\mathrm{RelyEvidence}_\alpha : \mathrm{RelyModes}(\alpha) \to \text{declared evidence predicates over events and state} \tag{41}$$

map each declared mode to the corresponding evidence rule that determines when that mode becomes true on the reviewed path.

Definition 5 (Causally effective narrowing event)**.** For a transitional workflow instance a, an event t on the reviewed same-target path is a causally effective narrowing event iff, under the declared reviewed scope, it makes at least one admissible downstream governed continuation unavailable, fixed, reserved, pre-routed, or otherwise materially narrowed in a way that later path behaviour must respect unless the path is explicitly re-opened or refused.

Definition 6 (Transitional reliance boundary)**.** For a transitional workflow instance a, $T_r^*(a)$ is the earliest declared event on the realised same-target path that is both supported by at least one realised reliance mode in $\mathrm{RelyModes}(\alpha(a))$ and is a causally effective narrowing event under

the associated evidence rules $\mathrm{RelyEvidence}_{\alpha(a)}$. If no such event can be established uniquely on the reviewed scope, transitional conformance fails for that path.

Remark 1. Declared reliance modes remain useful as explanatory and evidential structure, but they do not permit a later, more preferred mode to displace an earlier causally effective narrowing event. Mode declaration therefore supports boundary justification. It does not authorise boundary laundering by precedence assignment.

Definition 7 (Reversal boundary)**.** For a reversal workflow instance a, let

$$\mathrm{RevPrim}(\alpha(a)) \subseteq \{\mathsf{recall}, \mathsf{rescind}, \mathsf{reinstate}, \mathsf{revoke}, \mathsf{offset}, \mathsf{recover}\} \tag{42}$$

be the finite declared reversal-primitive vocabulary for that class. Then $T^*_{\mathrm{rev}}(a)$ is the earliest declared event on the realised reversal path at which the following are jointly bound on the reviewed scope: the declared origin instance $\mathrm{Orig}(a)$, the applicable reversal authority surface, and the first realised primitive in $\mathrm{RevPrim}(\alpha(a))$ whose declared effect typing is reversal of that origin effect family. If these binding conditions cannot be established uniquely, reversal conformance fails for that path.

Definition 8 (Irreversible execution boundary)**.** For an irreversible workflow instance a, $T^*_e(a)$ is the bound earliest governed irreversible execution boundary in the OTANIS sense.

5.6 Common Authority, Provenance, and Version Substrates

For every governed workflow class α, let

$$\mathrm{SnapshotProfile}(\alpha) = \begin{pmatrix} \mathrm{AcqScope}(\alpha),\ \mathrm{RevVis}(\alpha),\ \mathrm{FreshBound}(\alpha), \\ \mathrm{ReplicaRule}(\alpha),\ \mathrm{QuorumSet}(\alpha),\ \mathrm{QuorumThresh}(\alpha),\ \mathrm{FailRule}(\alpha) \end{pmatrix} \tag{43}$$

denote the declared boundary-state consistency profile. The paper-level admissible profile components are restricted as follows.

- $\mathrm{AcqScope}(\alpha) \in \{\mathsf{local}, \mathsf{path}, \mathsf{target}\}$,
- $\mathrm{RevVis}(\alpha) \in \{\mathsf{monotone}, \mathsf{quorum}, \mathsf{full}\}$,
- $\mathrm{FreshBound}(\alpha) \in \mathbb{R}_{>0} \cup \{0\}$,
- $\mathrm{ReplicaRule}(\alpha) \in \{\mathsf{single\_\_authority}, \mathsf{quorum\_\_agree}, \mathsf{all\_\_agree}\}$,
- $\mathrm{QuorumSet}(\alpha)$ is a finite non-empty declared authority-visible replica set whenever $\mathrm{RevVis}(\alpha) = \mathsf{quorum}$ or $\mathrm{ReplicaRule}(\alpha) = \mathsf{quorum\_\_agree}$,
- $\mathrm{QuorumThresh}(\alpha) \in \mathbb{N}$ satisfies $1 \leq \mathrm{QuorumThresh}(\alpha) \leq |\mathrm{QuorumSet}(\alpha)|$ whenever quorum semantics are declared,
- $\mathrm{FailRule}(\alpha) \in \{\mathsf{refuse}, \mathsf{escalate}\}$.

For every governed workflow instance a, let $x_b(a)$ denote the boundary-evaluation snapshot state

at $B^*(a)$. For any two replica views v_i, v_j gathered under the declared snapshot profile, define

$$\mathrm{ViewCompat}_{\alpha(a)}(a, v_i, v_j) = 1 \tag{44}$$

iff v_i and v_j agree on every boundary-relevant state component required by the declared profile for the reviewed path, namely authority validity, revocation state, boundary applicability, reviewed effect typing, and the refusal or escalation consequence that would follow at the candidate boundary event. Then define

$$\mathrm{SnapshotConsistent}(a, t) = 1 \tag{45}$$

iff all state sources required by $\mathrm{AcqScope}(\alpha(a))$ are acquired within the declared freshness bound $\mathrm{FreshBound}(\alpha(a))$, revocation visibility satisfies $\mathrm{RevVis}(\alpha(a))$, and replica agreement satisfies $\mathrm{ReplicaRule}(\alpha(a))$ on the reviewed scope at candidate boundary event t, with any quorum-based evaluation computed over the declared pair $\big(\mathrm{QuorumSet}(\alpha(a)), \mathrm{QuorumThresh}(\alpha(a))\big)$. Concretely, quorum visibility holds only if at least $\mathrm{QuorumThresh}(\alpha(a))$ members of the declared quorum set report pairwise compatible views under $\mathrm{ViewCompat}_{\alpha(a)}$, and quorum agreement holds only if the same threshold of replicas agree on the boundary-relevant state required by the declared profile. If $\mathrm{SnapshotConsistent}(a, t) = 0$, the class-specific boundary evaluation at t must apply $\mathrm{FailRule}(\alpha(a))$ and may not treat the snapshot as authoritative.

Define a boundary-resolved authority object

$$\beta_b(a) = \big(p_b, \sigma_b, b_b, \lambda_b, \rho_b, \kappa_b, \epsilon_b, \pi_b, \tau_b, \omega_b\big). \tag{46}$$

This notation is deliberately ARETABA-compatible. The point is not that every workflow class uses identical runtime semantics. The point is that authority, boundary definition, lifecycle validity, revocation, refusal, escalation, provenance, traceability, accountability, and snapshot consistency remain first-class across all governed workflow families.

The class-specific runtime predicates are then interpreted as follows.

$$\mathrm{Permit}^t(a, \beta_b(a), x_b(a)) \tag{47}$$

for transitional workflow instances at $T_r^*(a)$, subject to $\mathrm{SnapshotConsistent}(a, T_r^*(a)) = 1$,

$$\mathrm{Permit}^r(a, \beta_b(a), x_b(a)) \tag{48}$$

for reversal workflow instances at $T_{\mathrm{rev}}^*(a)$, subject to $\mathrm{SnapshotConsistent}(a, T_{\mathrm{rev}}^*(a)) = 1$,

and

$$\mathrm{Permit}(a, \beta_e(a), x_e(a)) \tag{49}$$

for irreversible workflow instances at $T_e^*(a)$, unchanged from OTANIS.

Only the last predicate is permission to cross an irreversible external commit boundary. The first two are permissions to enter a governed reliance state or a governed reversal path.

5.7 Origin Relation for Reversal Workflows

Let

$$\mathrm{Orig} : \mathcal{I} \to \mathcal{I} \cup \{\varnothing\} \tag{50}$$

be the declared origin relation for reversal workflow instances. For a reversal instance a_r, $\mathrm{Orig}(a_r) = a_o \neq \varnothing$ identifies the prior governed action instance whose effect, state, entitlement, or exposure the reversal workflow is declared to address.

For non-reversal workflow instances, define

$$\mathrm{Orig}(a) = \varnothing. \tag{51}$$

The origin relation is not an optional narrative field. It is part of what distinguishes a legitimate governed reversal from a later operational repair step with no governed link to the original path.

5.8 Class Promotion into Predefined OTANIS Workflows

The unified system must also account for class instability under realised path conditions. A workflow may begin as transitional or reversal and later encounter a narrower step whose effect semantics are no longer merely reliance-bearing or reversal-bearing, but irreversible in the OTANIS sense.

Define a partial promotion function

$$\mathrm{Promote} : \mathcal{A} \times T \times X \to \mathcal{A} \cup \{\varnothing\} \tag{52}$$

where $\mathrm{Promote}(\alpha, t, x) = \alpha^{\uparrow}$ means that under declared inputs and realised path conditions at event t, a narrower step of workflow class α must be reclassified into the predefined irreversible workflow class $\alpha^{\uparrow}$.

Define the irreversible-trigger predicate

$$\mathrm{IrrevTrigger}(a, t) = 1 \tag{53}$$

iff the realised step at event t on the path of a produces a declared irreversible external effect or a declared precursor commitment that OTANIS would already treat as governance-relevant for autonomous execution.

Then the promotion rule is

$$(\mathrm{WorkflowClass}(\alpha(a)) \in \{\mathsf{T}, \mathsf{R}\}) \wedge \mathrm{IrrevTrigger}(a, t) = 1 \Longrightarrow \mathrm{Promote}(\alpha(a), t, x) = \alpha^{\uparrow} \neq \varnothing. \tag{54}$$

The promoted narrower step is then instantiated as an irreversible workflow instance $a^{\uparrow}$ of class

$\alpha^{\uparrow}$ and governed under OTANIS at its own bound irreversible execution boundary $T_e^*(a^{\uparrow})$. If no such predefined irreversible class exists, autonomous continuation beyond t is non-admissible under default-deny semantics.

This is a structural rule, not a runtime discovery claim about arbitrary hidden reality. It applies only where the triggering conditions and the target irreversible scopes have themselves been reviewed and declared strongly enough for promotion to be well formed.

5.9 Bounded Unified Systems as Composable Governance Units

Let

$$\mathcal{U} = \{U_1, U_2, \ldots, U_m\} \tag{55}$$

be a finite set of bounded unified systems, where each U_j has its own governed workflow family, local boundaries, local authority surfaces, and local conformance claim under this paper.

OTANIS-USG governs the internal workflow family of each U_j. When a business action spans multiple members of $\mathcal{U}$, the resulting composite path does not fall under OTANIS-USG alone. It falls under the orchestration, GAG, and MGAG layers in addition to the local OTANIS- USG claims of the participating systems.

Define

$$\text{ExternallyClosed}(\mathcal{S}) = 1 \tag{56}$$

iff, for every target effect family and every governed outcome claimed by the reviewed slice, no undeclared external participant, adapter, queue, service, authority dependency, side obligation, or commit surface outside the reviewed slice can alter the admissibility, boundary binding, authority validity, refusal or escalation outcome, or effect semantics of the reviewed governed path. If any such undeclared external dependence exists, then $\text{ExternallyClosed}(\mathcal{S}) = 0$.

This distinction preserves architectural cleanliness. One bounded unified system is a governance unit. A composite business action spanning multiple such units is an orchestration problem, not merely a longer internal sequence. It also prevents a local reviewed-slice claim from being read as an end-to-end legitimacy claim when the slice is not externally closed.

6 Unified Governance Object and Class-Level Admissibility

6.1 Unified Review Object

For each governed workflow class α, define the unified review object

$$\Omega^{\text{USG}}(\alpha) = \begin{pmatrix} \text{WorkflowClass}(\alpha),\ \text{Cand}_\alpha,\ \text{Order}(\alpha),\ \text{Dep}(\alpha), \\ \text{TargetClass}(\alpha),\ \text{RelyModes}(\alpha),\ \text{RelyEvidence}_\alpha,\ \text{Orig}_\alpha, \\ \text{Compensable}(\alpha),\ \kappa_\alpha,\ \epsilon_\alpha,\ \pi_\alpha, \\ \tau_\alpha,\ \omega_\alpha,\ \text{Version}(\alpha) \end{pmatrix} \tag{57}$$

where:

- Cand_α is the class-level candidate-boundary set,
- $\mathrm{Order}(\alpha)$ is the class-level deterministic ordering model,
- $\mathrm{Dep}(\alpha)$ is the declared dependency scope,
- $\mathrm{TargetClass}(\alpha)$ is the class-level target effect family,
- $\mathrm{RelyModes}(\alpha)$ and $\mathrm{RelyEvidence}_\alpha$ are the declared reliance-mode set and evidence map for transitional classes, and are vacuous otherwise,
- Orig_α is the declared origin-link specification for reversal classes and $\varnothing$ otherwise,
- $\mathrm{Compensable}(\alpha)$ states whether the declared reversal semantics are compensatable,
- κ_α and ϵ_α are refusal and escalation semantics,
- π_α, τ_α, and ω_α are provenance, traceability, and accountability commitments,
- $\mathrm{Version}(\alpha)$ is the reviewed version-binding object.

The purpose of $\Omega^{\mathrm{USG}}(\alpha)$ is to keep boundary, authority, provenance, and version semantics aligned across all governed workflow classes rather than inventing unrelated review objects for each one.

6.2 Transitional-Class Admissibility

Define

$$\mathrm{TransitionalAdequate}(\alpha) = SC \wedge TB \wedge RM \wedge DC \wedge AU \wedge FR \wedge ES \wedge TR \wedge VR \tag{58}$$

for any class such that $\mathrm{WorkflowClass}(\alpha) = \mathsf{T}$, where:

- SC means the target scope and governed effect family are declared clearly,
- TB means a finite candidate-boundary set and deterministic binding rule are declared for T_r^*,
- RM means declared reliance modes and evidence mapping are present, so that the reviewed path does not depend on an ungrounded claim that a later step "relied" on the transitional artefact,
- DC means downstream reliance and coupling semantics are declared,
- AU means a boundary-resolved authority surface is declared,
- FR means freshness, expiry, invalidation, and reset semantics are declared,
- ES means refusal and escalation routes are declared,
- TR means provenance and trace commitments are declared,
- VR means reviewed version identity and replay identity are declared.

The corresponding runtime predicate is

$$\mathrm{Permit}^{t}(a, \beta_b(a), x_b(a)) \tag{59}$$

evaluated at $T_r^*(a)$. When this predicate is false, the transitional artefact may still exist as data, but it is not compliant for downstream governed reliance. This does not eliminate all reviewer judgement, but it does force reliance claims to be tied to declared modes and evidence rather than to purely narrative interpretation.

6.3 Reversal-Class Admissibility

Reversal is the workflow family in which rigid ex ante semantic closure is least realistic. Accordingly, this paper does not require reversal classes to be exhaustively specified in advance in the same way that a tightly bounded original irreversible action class may be. What it does require is a minimum structural contract strong enough to distinguish governed reversal from ad hoc repair.

Define

$$\mathrm{ReversalAdequate}(\alpha) = RC \wedge RA \wedge RB \wedge OR \wedge ES \wedge TR \wedge VR \tag{60}$$

for any class such that $\mathrm{WorkflowClass}(\alpha) = \mathsf{R}$, where:

- RC means a minimal declared reversal contract exists, namely a target effect family, an origin-link schema, a bounded reversal intent, and a reviewed statement of what kind of prior governed effect the class is permitted to address,
- RA means reversal authority is declared explicitly rather than presumed from the original workflow,
- RB means a finite candidate-boundary set and deterministic binding rule are declared for T_{rev}^*,
- OR means the ordering relation between the original governed path and the reversal path is declared,
- ES means refusal and escalation routes are declared,
- TR means provenance and trace commitments bind the reversal to the original path,
- VR means reviewed version identity and replay identity are declared.

The corresponding runtime predicate is

$$\mathrm{Permit}^{r}(a, \beta_b(a), x_b(a)) \tag{61}$$

evaluated at $T_{\mathrm{rev}}^*(a)$.

The compensatability distinction remains load-bearing, but compensatability need not be fully open-coded as a rich ex ante semantic tree. The conservative rule is this. If no declared governed continuation or compensation path exists, the framework defaults to non-compensatable reversal.

Definition 9 (Compensatable reversal)**.** Compensable(α) = 1 iff a further governed continuation or compensation path is declared strongly enough ex ante to prevent the reversal workflow from leaving an orphan governed effect when a later required step fails.

Definition 10 (Non-compensatable reversal)**.** Compensable(α) = 0 iff no such governed continuation path is declared or trusted for the relevant reversal scope. This is the conservative default. Once the reversal crosses its declared effect-bearing boundary, any later failure would then leave a new orphan effect unless the reversal path is terminal among required effect-bearing steps.

6.4 Legitimate Governed Reversal, Declared Operational Repair, and Malformed Reversal Claims

Define

$$\text{RepairIntent}(a) = 1 \tag{62}$$

iff the action instance a is explicitly declared or operationally represented as attempting correction, remediation, workaround, recovery, or offset relative to a prior effect family.

Define

$$\text{LegitimateReversal}(a_r) = 1 \tag{63}$$

iff all of the following hold.

1. $\text{WorkflowClass}(\alpha(a_r)) = \mathsf{R}$,
2. $\text{Orig}(a_r) \neq \varnothing$,
3. $\text{ReversalAdequate}(\alpha(a_r)) = 1$,
4. $B^*(\text{Orig}(a_r)) \prec T^*_{\text{rev}}(a_r)$,
5. provenance and trace commitments link the reversal path to the original governed path.

Define

$$\text{OperationalRepair}(a) = 1 \tag{64}$$

iff $\text{RepairIntent}(a) = 1$, $\text{Orig}(a) \neq \varnothing$, the path is declared or handled as repair or remediation outside governed reversal, and no claim of legitimate governed reversal is being made for that instance.

Define

$$\text{MalformedReversal}(a) = 1 \tag{65}$$

iff $\text{RepairIntent}(a) = 1$, the path is presented as governed reversal, but $\text{LegitimateReversal}(a) = 0$.

Define

$$\text{UnrelatedRepair}(a) = 1 \tag{66}$$

iff $\text{RepairIntent}(a) = 1$ but the path lacks a credible origin or target relation to the effect family it purports to correct.

These categories are intentionally not complements of one another over all actions. The negation of legitimate governed reversal contains many possibilities that are not repair at all. The paper therefore treats repair ontology as a small explicit partition over repair-intended actions rather than as the raw complement of reversal legitimacy.

6.5 Irreversible-Class Admissibility

For any class such that $\mathrm{WorkflowClass}(\alpha) = \mathsf{E}$, define

$$\begin{aligned}\mathrm{IrreversibleAdequate}(\alpha) = 1 \iff \mathrm{Admissible}_{\mathrm{ISDAIRE}}(\alpha) = 1 \\ \wedge \text{ a declared OTANIS-governed commit path exists for } \alpha.\end{aligned} \tag{67}$$

The corresponding runtime predicate remains unchanged:

$$\mathrm{Permit}(a, \beta_e(a), x_e(a)) \tag{68}$$

at $T_e^*(a)$.

6.6 Unified Class-Level Adequacy

Define the class-level adequacy predicate

$$\mathrm{Adequate}(\alpha) = \begin{cases} \mathrm{TransitionalAdequate}(\alpha), & \mathrm{SelectArchitecture}(\alpha) = \mathsf{TG}, \\ \mathrm{ReversalAdequate}(\alpha), & \mathrm{SelectArchitecture}(\alpha) = \mathsf{RG}, \\ \mathrm{IrreversibleAdequate}(\alpha), & \mathrm{SelectArchitecture}(\alpha) = \mathsf{OTANIS}. \end{cases} \tag{69}$$

This is where the paper becomes unified rather than merely descriptive. A governed workflow class is not only labelled. It is assigned the governance architecture whose admissibility predicate actually matches its declared role.

7 System-Level Unified Integrity

Class-level adequacy is necessary and not sufficient. A unified architecture also requires system-level integrity across workflow interactions.

Define

$$\mathrm{BoundaryIntegrity}(\mathcal{S}) = 1 \tag{70}$$

iff all of the following hold.

1. No workflow class declared transitional or reversal silently produces a new irreversible external effect on the reviewed path without promotion into a predefined irreversible class and OTANIS re-entry.
2. No workflow claimed as reversal lacks a declared origin link or declared reversal boundary.

3. No declared operational repair step or malformed reversal claim is counted as a governed reversal.

4. All declared candidate-boundary sets, ordering semantics, dependency scopes, observation scopes, and snapshot profiles remain version-compatible across the reviewed architecture.

Define

$$\mathrm{OrderingIntegrity}(\mathcal{S}) = 1 \tag{71}$$

iff all of the following hold.

1. For any realised path in which a transitional workflow instance conditions a later irreversible workflow instance in the same target effect family, the relevant reliance boundary is ordered no later than the later irreversible boundary.

2. For any legitimate reversal instance a_r, the relevant original governed boundary occurs before the reversal boundary, that is

$$B^*(\mathrm{Orig}(a_r)) \prec T^*_{\mathrm{rev}}(a_r).$$

3. If $\mathrm{Compensable}(\alpha(a_r)) = 0$, then the reversal instance is terminal among required effect-bearing steps on the declared reversal path.

4. If a reversal path contains a new irreversible sub-step, that sub-step is classified irreversible and remains inside OTANIS at its own T^*_e.

Define

$$\mathrm{CompositionReady}(\mathcal{S}) = 1 \tag{72}$$

iff any declared participation of the reviewed slice in a wider composite business action is bound to a declared orchestration specification, declared participant set, declared ordering semantics, and version-compatible trace binding consistent with the orchestration, GAG, and MGAG layers.

The composition condition is therefore not waived merely because the authors choose not to make a broad composite claim. If no composite claim is made, the reviewed slice must instead be externally closed with respect to the outcomes it claims.

Define the system-level conformance predicate

$$\begin{aligned}\mathrm{UnifiedConformant}(\mathcal{S}) = \bigwedge_{\alpha\in\mathcal{A}} \mathrm{Adequate}(\alpha) \wedge \mathrm{BoundaryIntegrity}(\mathcal{S}) \wedge \mathrm{OrderingIntegrity}(\mathcal{S}) \\ \wedge \mathrm{PressureSatisfied}(\mathcal{S}) \wedge \mathrm{CompositionCondition}(\mathcal{S}).\end{aligned} \tag{73}$$

where

$$\mathrm{CompositionCondition}(\mathcal{S}) = \begin{cases} \mathrm{CompositionReady}(\mathcal{S}), & \text{if the reviewed slice is declared as,} \\ & \text{part of a wider composed action} \\ \mathrm{ExternallyClosed}(\mathcal{S}), & \text{otherwise.} \end{cases} \tag{74}$$

7.1 Path-Ordering Constraint

The path-ordering rules can be stated directly.

Rule 7 (Reliance-before-commit)**.** Let a_t be an instance of a transitional class and let a_e be a downstream instance of an irreversible class such that $\text{Target}(a_t) = \text{Target}(a_e)$. Then an admissible path requires

$$T_r^*(a_t) \prec T_e^*(a_e).$$

If this ordering cannot be established under declared path semantics, the mixed-system design is non-admissible for autonomous continuation on that path.

Rule 8 (Promotion of transitional or reversal steps under effect-bearing change)**.** Let a be an instance of a transitional or reversal class. If a realised event t on the path of a satisfies $\text{IrrevTrigger}(a, t) = 1$, then the narrower effect-bearing step must be promoted into a predefined irreversible workflow class $\alpha^{\uparrow} = \text{Promote}(\alpha(a), t, x)$ and governed under OTANIS at $T_e^*(a^{\uparrow})$. If no such predefined irreversible class exists, autonomous continuation on that narrower path is non-admissible.

Rule 9 (Reversal cannot precede its origin)**.** For any reversal instance a_r such that $\text{LegitimateReversal}(a_r) = 1$, the relevant original governed boundary must already exist on the reviewed path before the reversal boundary may be validly crossed.

Rule 10 (Non-compensatable reversal terminality)**.** If $\text{LegitimateReversal}(a_r) = 1$ and $\text{Compensable}(\alpha(a_r)) = 0$, then no later required effect-bearing step may remain unresolved on the same declared reversal path. Otherwise the path is structurally weak and unified conformance fails.

Rule 11 (OTANIS re-enters inside reversal when new irreversible effect occurs)**.** If a reversal path contains a step whose realised commit produces a new irreversible external effect, that step is an irreversible workflow instance and must satisfy full OTANIS execution- boundary enforcement at its own T_e^*, even though the enclosing business purpose of the path is reversal.

8 Pressure-Testing Framework

A unified governance claim is not credible merely because workflow classes were labelled and documented. The deployment must also be pressure-tested against the failure modes most likely to collapse the distinction between transitional, reversal, and irreversible paths.

Define

$$\text{PressureSpecified}(\mathcal{S}) = 1 \tag{75}$$

iff the declared test pack covers, at minimum, the following families and, for each required family in the reviewed slice, declares family-level acceptance criteria and freshness expectations sufficient to determine a pass or fail result.

1. **Workflow misclassification** A path declared transitional or reversal is shown in fact to contain an undeclared irreversible effect, or a path declared outside the governed family is

shown to condition later governed action.

2. **Boundary ambiguity** A declared reliance boundary or reversal boundary cannot be selected deterministically under declared inputs, reviewed observation scope, retry collapse, and ordering semantics.
3. **Missing or weak origin link** A supposed reversal path cannot demonstrate a declared and traceable link to the original governed path it claims to address.
4. **Stale original-state or stale reversal-authority failure** A reversal is attempted using stale authority, stale origin-state evidence, stale snapshot visibility, or stale version bindings.
5. **Compensatable versus non-compensatable mismatch** A reversal declared compensatable is shown to leave an orphan effect when a later step fails, or a non-compensatable reversal is incorrectly treated as if a safe later recovery path existed.
6. **Operational repair substitution** A manual remediation, spreadsheet adjustment, or informal override is passed off as a legitimate governed reversal.
7. **Bypass and side-route testing** A reviewed path can be bypassed through an alternate adapter, queue, retry path, or side execution surface.
8. **Version and trace skew** The original path, the reversal path, and the irreversible commit path cannot be replayed coherently under one compatible set of reviewed artefacts.
9. **Non-regression under topology change** A new adapter, queue, event source, remediation route, or participant system introduces a new governed path without corresponding reclassification and retesting.
10. **Undeclared external dependence** The reviewed slice relies on undeclared external participants, external commit surfaces, or wider composite coordination while still presenting itself as a self-sufficient local conformance claim.

Let $\mathcal{F}_{\text{req}}(\mathcal{S})$ denote the finite set of required pressure-test families for the reviewed claim slice, induced by the families listed above after applying only those scope exclusions for which $\text{ScopeJustified}(f, \mathcal{S}) = 1$ is itself declared and reviewable.

For paper completeness, the non-waivable lower-bound operator sets and minimum pass criteria are fixed as follows.

Family	**BaselineOps**(f)	**BaselineCriteria**(f)
Workflow misclassification	Inject one undeclared irreversible step; inject one advisory-to-relied-upon step.	The reviewed slice must either reclassify the path correctly or refuse/escalate before any conflicting governed claim persists.
Boundary ambiguity	Execute concurrent-candidate case; retry-duplicate case; observer-divergence case.	The system must either produce one unique reviewed minimum consistent with declared ordering semantics or refuse/escalate.

Family	**BaselineOps**(f)	**BaselineCriteria**(f)
Missing or weak origin link	Remove origin link; substitute wrong target family; provide trace without origin binding.	Any failed origin-link condition must block legitimate reversal and be surfaced as non-conformance or non-reversal classification.
Stale original-state or stale reversal-authority failure	Stale authority case; stale revocation case; stale replica case.	Any case violating the declared snapshot profile must trigger the declared fail rule and must not permit governed continuation.
Compensatable versus non-compensatable mismatch	Commit reversal step then fail later required step; omit continuation path.	If orphan-effect risk appears, the system must classify the path as non-compensatable or fail conformance.
Operational repair substitution	Present repair as governed reversal without full reversal artefacts.	The path must not be admitted as legitimate reversal and must be classified under the repair ontology.
Bypass and side-route testing	Alternate adapter invocation; retry-path invocation; side-surface invocation.	No bypass route may silently preserve the same governed claim. The system must block, surface, or rescope the claim.
Version and trace skew	Mismatched artefact version; replay mismatch; missing trace binding.	Unified conformance fails unless replay and trace remain coherent under one compatible reviewed version set.
Non-regression under topology change	Introduce new adapter, queue, event source, or participant without prior review.	The new topology must force reclassification, retesting, or refusal of the prior conformance claim.
Undeclared external dependence	Omit one external participant or external commit surface that can alter effect semantics.	The slice must fail external closure or be elevated to composite review; it may not retain standalone local conformance.

For each family $f \in \mathcal{F}_{\mathrm{req}}(\mathcal{S})$, let

$$\mathrm{BaselineOps}(f) \tag{76}$$

denote the corresponding minimum operator set fixed above, and let

$$\mathrm{BaselineCriteria}(f) \tag{77}$$

denote the corresponding non-waivable lower-bound acceptance criteria fixed above. Let further

$$\mathrm{PassCriteria}(f) \tag{78}$$

be any stricter deployment-specific acceptance criteria declared for the reviewed slice.

Then define

$$\text{FamilyPass}(f, \mathcal{S}) = 1 \tag{79}$$

iff all of the following hold.

1. current executed evidence exists for every minimum operator in $\text{BaselineOps}(f)$,
2. that evidence is fresh for the reviewed claim,
3. the evidence satisfies $\text{BaselineCriteria}(f)$,
4. the evidence also satisfies any stricter declared local criteria $\text{PassCriteria}(f)$.

Define

$$\text{PressurePassed}(\mathcal{S}) = 1 \tag{80}$$

iff for every required family $f \in \mathcal{F}_{\text{req}}(\mathcal{S})$, $\text{FamilyPass}(f, \mathcal{S}) = 1$. Equivalently, any required family whose minimum operators are unexecuted, whose evidence is stale, or whose evidence fails either the paper-level lower bound or the declared stricter local criteria forces $\text{PressurePassed}(\mathcal{S}) = 0$.

Define

$$\text{PressureSatisfied}(\mathcal{S}) = 1 \iff \text{PressureSpecified}(\mathcal{S}) = 1 \wedge \text{PressurePassed}(\mathcal{S}) = 1. \tag{81}$$

Pressure testing in this paper is therefore not a generic robustness exercise. It is a structured attempt to falsify the specific governance distinctions on which the unified architecture depends, and unified conformance requires not only declared coverage but also pass-conditioned satisfaction against non-waivable lower bounds for the reviewed scope.

9 Formal Properties

Proposition 1 (Transitional reliance-mode determinacy)**.** If $\text{WorkflowClass}(\alpha) = \mathsf{T}$ but either $\text{RelyModes}(\alpha) = \varnothing$, or no declared reliance-evidence rule can determine a causally effective narrowing event on the reviewed path, then $\text{TransitionalAdequate}(\alpha) = 0$.

Proof. Direct from the definition of TransitionalAdequate. Without declared reliance modes and corresponding evidence rules sufficient to determine a causally effective narrowing event, the reliance boundary remains too indeterminate to support a conformance claim under this paper. □

Proposition 2 (Adjudication determinacy under fixed reviewed inputs)**.** If two reviewers apply the same raw declared evidence multiset, the same class-evidence schema, the same reviewed-scope projection, the same schema-validation rule, the same evidence-identity function, and the same obligation and exclusion sets to class α, then they must reach the same value of $\text{Adjudicated}(\alpha, c)$ for each candidate class c.

Proof. Direct from the definitions of SchemaValid, $\text{ProjectScope}_{\alpha}$, EvidenceID, duplicate collapse, and the obligation protocol. The normalised multiset $N(\alpha)$ is not a free input. It is a

deterministic function of those declared preprocessing objects and the raw evidence multiset. Adjudication is then a deterministic check over $N(\alpha)$, the fixed obligation sets, and the fixed exclusion conditions. Therefore the result cannot vary unless one of those declared review inputs varies. □

Proposition 3 (Architecture-selection determinacy)**.** If the declared class label and declared structural evidence do not resolve to exactly one workflow class, then SelectArchitecture(α) is indeterminate and Adequate(α) = 0.

Proof. Immediate from the definitions of WorkflowDecl, WorkflowClass, and SelectArchitecture. Architecture selection is only defined after the declared class label has been reconciled with declared structural evidence. If no unique resolved class exists, the adequacy predicate is undefined and the class cannot support a governed conformance claim. □

Proposition 4 (Boundary indeterminacy under unresolved or non-equivalent reviewed candidates)**.** If the reviewed candidate set $\widehat{\mathrm{Cand}}(a)$ remains empty, duplicate-ambiguous, observer-divergent, without a unique reviewed minimum equivalence class, with a minimum class that is not boundary-equivalent, or without a defined canonical representative for that class, then $B^*(a)$ is indeterminate and the relevant class-level conformance claim fails for that path.

Proof. Direct from the definition of the common boundary substrate. Boundary binding is only defined after observation-scope projection, retry collapse, unique reviewed-minimum class selection, boundary-equivalence across that class, and canonical representative selection. If any one of those conditions fails, the boundary cannot be treated as an executable event-typed boundary on the reviewed scope. □

Proposition 5 (Snapshot inconsistency blocks boundary authority)**.** If SnapshotConsistent($a, B^*(a)$) = 0, then no compliant transitional or reversal permit may be issued at $B^*(a)$.

Proof. Direct from the definition of the common authority and snapshot substrate. When the declared snapshot profile is not satisfied, the boundary-evaluation snapshot cannot be treated as the reviewed authority basis and the system must refuse or escalate. □

Proposition 6 (Canonical representatives may not change boundary semantics)**.** Let $t_i, t_j \in \mathrm{MinClass}(a)$. If $\mathrm{BoundaryEq}_{\alpha(a)}(a, t_i, t_j) = 0$ for any such pair, then $\mathrm{CanonRep}_{\alpha(a)}(\mathrm{MinClass}(a))$ is undefined and boundary resolution must fail.

Proof. Direct from the definition of boundary-equivalence and the admissibility condition on the canonical representative selector. The selector is permitted to represent a reviewed minimum class only when choosing one member is semantically harmless with respect to snapshot consistency, boundary outcome, effect typing, and refusal or escalation consequence. If that condition fails for any pair in the minimum class, the selector would become outcome determinative and boundary resolution must refuse or escalate instead. □

Proposition 7 (OTANIS exclusivity for irreversible workflows)**.** If WorkflowClass(α) = E, then no transitional or reversal predicate may substitute for Permit($a, \beta_e(a), x_e(a)$) at $T_e^*(a)$.

Proof. Direct from the definitions of SelectArchitecture and IrreversibleAdequate. The irreversible case maps uniquely to the OTANIS architecture. Any weaker substitution would violate the selection rule and the execution-bearing supremacy rule. □

Proposition 8 (Promotion under realised irreversibility)**.** Let a be an instance of a transitional or reversal class. If IrrevTrigger(a, t) = 1 at realised event t, then either a predefined irreversible class $\alpha^{\uparrow}$ = Promote($\alpha(a), t, x$) exists and the narrower step is governed under OTANIS, or else autonomous continuation is non-admissible.

Proof. Direct from the definition of the promotion function and the promotion rule. Once the realised step is no longer merely reliance-bearing or reversal-bearing but effect-bearing in the OTANIS sense, the narrower step cannot remain governed only under the transitional or reversal model. □

Proposition 9 (Reversal target precedence)**.** If LegitimateReversal(a_r) = 1, then

$$B^*(\mathrm{Orig}(a_r)) \prec T^*_{\mathrm{rev}}(a_r).$$

Proof. This is built into the definition of legitimate governed reversal. A path cannot count as reversal of an origin whose relevant governed boundary has not yet been established. □

Proposition 10 (Non-compensatable reversal terminality)**.** Let a_r be a legitimate reversal instance with Compensable($\alpha(a_r)$) = 0. If a later required effect-bearing step remains on the same declared reversal path, then OrderingIntegrity($\mathcal{S}$) = 0 and hence UnifiedConformant($\mathcal{S}$) = 0.

Proof. A non-compensatable reversal that is not terminal admits a failure schedule in which the reversal has already produced a new governed effect while a later required effect-bearing step fails. The result is a new orphan effect on the reversal path. By definition this violates ordering integrity and therefore unified conformance. □

Proposition 11 (Pressure-test failure blocks unified conformance)**.** If PressurePassed($\mathcal{S}$) = 0, then UnifiedConformant($\mathcal{S}$) = 0.

Proof. Direct from the definition of UnifiedConformant($\mathcal{S}$). System-level conformance depends on PressureSatisfied($\mathcal{S}$) = 1, which in turn requires both specified coverage and passing pressure-test outcomes for the reviewed scope. □

Proposition 12 (Externally open slices cannot claim standalone conformance)**.** If the reviewed slice is not declared as part of a wider composed action and ExternallyClosed($\mathcal{S}$) = 0, then UnifiedConformant($\mathcal{S}$) = 0.

Proof. Direct from the definition of CompositionCondition($\mathcal{S}$). In the absence of an explicit composite claim, system-level conformance requires external closure of the reviewed slice. □

Proposition 13 (Repair-intended non-legitimate paths are not legitimate governed reversal)**.** If RepairIntent(a) = 1 and LegitimateReversal(a) = 0, then the path is not a legitimate governed reversal and, where sufficient representation and origin information exist, it must be classified under the repair ontology as malformed reversal, declared operational repair, unrelated repair, or another explicitly declared non-reversal repair category.

Proof. Direct from the four repair-ontology definitions used in this section. The repair ontology is an explicit partition over repair-intended actions, not the raw complement of legitimate reversal over all actions. □

Proposition 14 (Irreversible sub-step inside a reversal path re-enters OTANIS)**.** If a reversal path contains a sub-step whose realised commit produces a new irreversible external effect, then that sub-step must be classified irreversible and evaluated under OTANIS at its own T_e^*.

Proof. The workflow-class definitions in this paper are semantic rather than narrative. Once a realised sub-step produces a declared irreversible external effect on a controlled mediated path, it satisfies the irreversible workflow-class definition. By the architecture-selection rule, the sub-step therefore maps to OTANIS. □

10 Worked Example: Mixed Insurance Claim Settlement

10.1 Business Setting

The insurance examples used in the OTANIS family remain useful here because they already contain execution-bearing, compensatory, and multi-step elements [3–7]. The present paper reuses that domain but isolates the unified-systems-governance question.

Let the deployment contain the following declared workflow classes.

$$\alpha_1 = \text{SettlementPacketAssembly}, \tag{82}$$

$$\alpha_2 = \text{ClaimPayoutSubmission}, \tag{83}$$

$$\alpha_3 = \text{ClaimRecoveryInitiation}. \tag{84}$$

All three classes belong to the same target effect family, namely governed claim settlement.

- α_1 assembles the settlement packet, including reserve values, routing metadata, and approval-surface fields.
- α_2 submits the claim payment on the governed payout path.
- α_3 initiates a governed recovery or unwind path if later sanctions, fraud, or reconciliation evidence requires reversal action.

10.2 Admissible Unified Design

Class α_1. The settlement-packet assembly step does not itself commit an irreversible external effect, but it does condition later payout submission. Declare

$$\mathrm{WorkflowClass}(\alpha_1) = \mathsf{T}.$$

Its candidate boundary set is

$$\mathrm{Cand}(a_1) = \{\mathrm{SettlementPacketPersist}, \mathrm{ApprovalQueueEnqueue}\}. \tag{85}$$

If the earliest governed reliance occurs when the completed packet is enqueued for later operational use, then

$$T_r^*(a_1) = \mathrm{ApprovalQueueEnqueue}. \tag{86}$$

Class α_2. The payout-submission class directly crosses an irreversible external boundary on the payment rail. Declare

$$\mathrm{WorkflowClass}(\alpha_2) = \mathsf{E}.$$

Its candidate boundary set is

$$\mathrm{Cand}(a_2) = \{\mathrm{PayoutGatewaySubmit}, \mathrm{PaymentRailSubmission}\}. \tag{87}$$

If the payment rail submission is the first non-cancellable governed commit, then

$$T_e^*(a_2) = \mathrm{PaymentRailSubmission}. \tag{88}$$

This class remains fully inside OTANIS.

Class α_3. Now suppose a later sanctions or fraud signal requires recovery or unwinding action after the original payout path. Declare

$$\mathrm{WorkflowClass}(\alpha_3) = \mathsf{R}.$$

For the realised reversal instance a_3, let

$$\mathrm{Orig}(a_3) = a_2. \tag{89}$$

Its candidate boundary set is

$$\mathrm{Cand}(a_3) = \{\mathrm{PaymentRecallSubmit}, \mathrm{ReserveReinstateCommit}\}. \tag{90}$$

If the payment rail still exposes a governed recall route and recall submission is the earliest governed reversal boundary, then

$$T_{\mathrm{rev}}^*(a_3) = \mathrm{PaymentRecallSubmit}. \tag{91}$$

The path is admissible only if the ordering relation is declared explicitly:

$$T_e^*(a_2) \prec T_{\mathrm{rev}}^*(a_3). \tag{92}$$

If the insurer has also declared what happens when recall fails, then the compensatability of the reversal class can be stated clearly.

- If a later governed reserve-reinstatement or supervised continuation path exists, then

$$\mathrm{Compensable}(\alpha_3) = 1.$$

- If no such later governed continuation path exists, then

$$\mathrm{Compensable}(\alpha_3) = 0,$$

  and the reversal path must be terminal among required effect-bearing steps.

The unified design is admissible under OTANIS-USG if all of the following hold.

1. $\mathrm{TransitionalAdequate}(\alpha_1) = 1$.
2. $\mathrm{IrreversibleAdequate}(\alpha_2) = 1$.
3. $\mathrm{ReversalAdequate}(\alpha_3) = 1$.
4. $T_e^*(a_2) \prec T_{\mathrm{rev}}^*(a_3)$.
5. Any irreversible sub-step inside the reversal path remains inside OTANIS.
6. Trace and provenance commitments allow replay of the original payout and the reversal path as one coherent governed history.

10.3 A Transitional-to-Irreversible Promotion Case

Suppose now that the insurer introduces a rail-reservation mechanism during packet handling. The packet-assembly workflow remains transitional in general, but one realised step reserves a scarce payout corridor in a way that is economically or operationally binding before the later rail submission. Under the present framework that realised step may no longer remain only inside the transitional class.

If the reservation semantics are governance-relevant under the declared boundary-discovery and precursor-commit logic of the wider OTANIS family, then the narrower reservation step must be promoted into a predefined irreversible workflow class, for example

$$\alpha_1^{\uparrow} = \mathrm{PayoutCorridorReservation}.$$

That promoted step is then governed under OTANIS at its own bound irreversible execution boundary. The original workflow does not remain purely transitional merely because its business label still sounds preparatory.

This is the practical meaning of class promotion. A workflow class may remain transitional as a whole while a narrower realised step instantiates an OTANIS-governed irreversible sub-path. If no such predefined irreversible scope exists, autonomous continuation beyond the reservation step is non-admissible until the design is corrected.

10.4 A Non-Admissible Repair Masquerading as Reversal

Now suppose the original payout has already fully settled and no declared governed recall route exists. The insurer later performs a spreadsheet-driven recovery attempt, manual reserve change, or informal debt-collection handoff and labels this "the reversal workflow".

That claim is non-admissible under this framework.

1. The path may lack a declared reversal boundary.
2. The path may lack a governed origin link beyond narrative business explanation.
3. The path may lack a declared reversal authority surface.
4. The path may contain declared operational repair or a malformed reversal claim, but not a legitimate governed reversal.

In that case the later step may still be operationally real and commercially necessary. The claim rejected here is only the stronger one, namely that the later step is a legitimate governed reversal under the same unified architectural governance framework.

The correct response is one of the following.

1. declare and govern a real reversal workflow class,
2. keep the later step as explicit operational repair outside the governed claim,
3. or remove the unified-conformance claim for that mixed path.

11 Failure Cases Exposed by OTANIS-USG

OTANIS-USG makes several unified-governance failure classes explicit.

11.1 F1. Silent Promotion Failure

A workflow declared transitional or reversal is later shown to produce a new irreversible external effect or governance-relevant precursor commitment on the reviewed path without promotion into a predefined irreversible class and without OTANIS re-entry.

11.2 F2. Undeclared Reversal Link

A supposed reversal path exists in operational reality, but no declared Orig relation links it to the original governed path.

11.3 F3. Undeclared Reversal Boundary

A workflow is described as reversal, but no finite candidate-boundary set or deterministic binding rule exists for T^*_{rev}.

11.4 F4. Stale Original-State or Reversal-Authority Failure

A reversal is attempted using stale origin-state evidence, stale revocation state, stale lifecycle state, or stale authority assumptions.

11.5 F5. Wrong Ordering Between Original and Reversal

The architecture cannot demonstrate that the relevant original governed boundary occurred before the reversal boundary, or the reversal path is triggered against the wrong target effect family.

11.6 F6. Non-Compensatable Reversal with Later Required Effect-Bearing Step

A reversal declared non-compensatable is followed by a later required effect-bearing step, creating the possibility of a new orphan effect on the reversal path.

11.7 F7. Operational Repair Substitution

A manual remediation, spreadsheet correction, or informal workaround is described as a governed reversal even though the required reversal artefacts are absent, yielding a malformed reversal claim rather than legitimate governed reversal.

11.8 F8. Bypass, Side Route, or Version Skew

The reviewed mixed path can be bypassed through an alternate adapter, queue, or retry route, or the original path and reversal path cannot be replayed coherently under one compatible set of reviewed artefacts.

11.9 F9. False Orchestration Equivalence

A system treats every internal workflow sequence as if it were already a formally orchestrated composite action, or treats a true multi-system composed action as if local OTANIS-USG review alone were sufficient.

12 Falsifiability Conditions

The paper is intended to remain falsifiable. The following observations would refute a claim that a deployment satisfies OTANIS-USG as defined here.

1. A workflow class cannot be assigned uniquely to transitional, reversal, or irreversible.
2. A workflow class assigned to transitional or reversal is shown in fact to cross a new irreversible external boundary or governance-relevant precursor commitment without promotion into a predefined irreversible class and OTANIS re-entry.

3. A supposed reversal path cannot demonstrate a declared origin link, a declared reversal boundary, or a declared reversal authority surface.
4. A claimed legitimate reversal cannot demonstrate

$$B^*(\mathrm{Orig}(a_r)) \prec T^*_{\mathrm{rev}}(a_r).$$

5. A reversal declared non-compensatable is followed by a later required effect-bearing step.
6. A deployment labels a manual remediation or operational workaround as a governed reversal without the required declared artefacts.
7. A realised mixed path cannot reconstruct, under compatible reviewed versions, the transitional artefacts, the irreversible commit path, the reversal artefacts, and the ordering relations between them.
8. A pressure test demonstrates a bypass route, silent promotion path, false-orchestration equivalence, or version-skew path that falls inside the declared governed scope.

These conditions are deliberately narrow. They do not prove that the deployment is globally safe. They do show that a unified-governance claim made under this framework can be tested and refuted.

13 Discussion

The framework presented here is intentionally conservative. That is appropriate. Mixed systems become weakest precisely where workflow roles are blurred and later repaired informally under business pressure. A unified architecture needs stronger distinctions, not weaker ones.

The formal results in this paper are also intentionally modest. They are primarily consistency, closure, classification, and no-overclaim results rather than deep new theorems about arbitrary distributed systems. That does not make them unimportant. It means the paper should be read as an architectural unification and conformance discipline, not as a claim to have solved semantic adequacy, workflow discovery, or universal commit theory.

External validation likewise remains limited and future-facing. The present paper is strongest as an internally coherent OTANIS-family extension grounded in classic systems, workflow, access-control, and safety literature. Broader empirical comparison against live deployments, competing governance patterns, and inter-reviewer classification reliability remains outside the current claim surface and should be treated as future work rather than implied accomplishment.

Three clarifications are especially important.

First, this paper does not collapse everything into OTANIS. Only irreversible workflows require full OTANIS execution-boundary enforcement. Transitional and reversal workflows have their own narrower governance models because they raise different architectural questions. Having said that, those narrower models do not remain in force if the realised path later becomes effect-bearing at a narrower scope. In that case the relevant step must be promoted into a

predefined OTANIS workflow or autonomous continuation must stop.

Second, the paper does not collapse reversal into "undo". A reversal may itself be a complex governed path, and it may itself contain a new irreversible step. When that happens, OTANIS re-enters at the relevant irreversible sub-step. This is one reason the compensatable versus non-compensatable distinction is so important and why the ordering discipline in this paper is closer in spirit to governed compensation rather than casual rollback [5, 10].

Third, the paper does not collapse operational reality into formal legitimacy. Organisations may still carry out manual remediation, business correction, customer appeasement, debt recovery, or other real repair work. The narrower claim made here is only that those steps do not count as legitimate governed reversal unless the required reversal artefacts were declared and satisfied.

A further practical consequence follows. OTANIS-USG does not replace orchestration. It makes orchestration cleaner. A bounded unified system becomes a more rigorous local governance unit. Multiple such units may then participate in a wider composite business action under the orchestration, GAG, and MGAG layers, provided local supremacy, declared participant sets, ordering semantics, and trace binding remain intact [5]. This means that a number of unified systems may indeed be composed together under the OTANIS family, but the resulting claim is stronger only if local USG conformance and composite orchestration conformance both hold.

A practical classification consequence follows as well. Many organisations will discover that only a subset of their mixed workflows are fit for a unified governed claim. Some workflows will remain purely advisory and should stay outside the governed family. Some will be transitional and require bounded reliance governance. Some will be reversal workflows and require explicit origin links, authority, ordering, and compensatability semantics. Some will be irreversible and must satisfy full OTANIS. Some will fail the review entirely and should remain human-coordinated or explicitly treated as operational repair rather than governed autonomy. That is not a weakness of the framework. It is an architectural classification result.

14 Conclusion

This paper formalises OTANIS-USG as a unified architectural governance framework for mixed-consequence bounded agentic systems whose governed workflow family consists of transitional, reversal, and irreversible workflows. The problem addressed is narrower than universal AI safety and narrower than OTANIS runtime execution governance itself. It is the question of how one architecture containing these three governed workflow classes should classify them on the reviewed scope, resolve a justified governance architecture for each, and preserve consistency, composability, and auditability across the whole governed path.

The paper's answer is deliberately constrained. Transitional workflows are governed at declared reliance boundaries. Reversal workflows are governed as new declared paths linked to prior governed action through a declared origin relation, a declared reversal boundary, a minimum structural ex ante contract, explicit authority and provenance requirements, and explicit compensatable versus non-compensatable semantics with conservative default handling. Irreversible workflows remain governed by OTANIS and ARETABA at $T_e^*(a)$. If a transitional or reversal

path later becomes effect-bearing at a narrower declared scope, that step must be promoted into a predefined OTANIS-governed irreversible workflow. System-level conformance requires class-level adequacy, boundary integrity, ordering integrity, pass-conditioned pressure-testing satisfaction, and either explicit composite readiness or external closure of the reviewed slice. Operational repair and malformed reversal claims must not be confused with legitimate governed reversal.

These claims do not make every mixed workflow autonomously governable. They do provide a more defensible basis for distinguishing between architectures that are structurally capable of coherent governance across transitional, reversal, and irreversible workflows and architectures that remain too weak, too hidden, or too under-specified for a serious governance claim. Read strictly, that is the contribution of OTANIS-USG. It is also enough to matter.

References

[1] M. Otani, "Executable governance under multiple irreversibility boundaries: Deterministic binding to the earliest governed irreversible execution boundary in composed AI systems," AI Consultant Insights (AICI), Technical Report, March 2026.

[2] M. Otani, "OTANIS: An operational trust and authority normative integrated system for executable governance of agentic AI across multiple irreversibility boundaries," AI Consultant Insights (AICI), Technical Report, March 2026.

[3] M. Otani, "ISDAIRE: A formal specification and admissibility framework for executable governance of agentic AI systems," Architectural Governance, Technical Report, March 2026.

[4] M. Otani, "ARETABA: A formal execution-time authority framework for constructing and enforcing governed irreversible actions in agentic AI systems," Architectural Governance, Technical Report, March 2026.

[5] M. Otani, "Orchestrating multiple OTANIS systems: A formal architecture for governed composite irreversible actions in agentic AI systems," Architectural Governance, Technical Report, March 2026.

[6] M. Otani, "Authority credentials, delegation, and provenance for OTANIS-governed agentic systems," Architectural Governance, Technical Report, March 2026.

[7] M. Otani, "Operational assurance, registry semantics, predicate authoring, and conformance for OTANIS-governed agentic systems," Architectural Governance, Technical Report, March 2026.

[8] J. H. Saltzer and M. D. Schroeder, "The protection of information in computer systems," *Proceedings of the IEEE*, vol. 63, no. 9, pp. 1278–1308, 1975.

[9] J. Gray, "Notes on database operating systems," in *Operating Systems, An Advanced Course*, R. Bayer, R. M. Graham, and G. Seegmüller, Eds. Berlin, Heidelberg: Springer, 1978, pp. 393–481.

[10] H. Garcia-Molina and K. Salem, "Sagas," in *Proceedings of the 1987 ACM SIGMOD International Conference on Management of Data*, 1987, pp. 249–259.

[11] L. Lamport, "Time, clocks, and the ordering of events in a distributed system," *Communications of the ACM*, vol. 21, no. 7, pp. 558–565, 1978.

[12] K. M. Chandy and L. Lamport, "Distributed snapshots: Determining global states of distributed systems," *ACM Transactions on Computer Systems*, vol. 3, no. 1, pp. 63–75, 1985.

[13] M. Bishop and M. Dilger, "Checking for race conditions in file accesses," *Computing Systems*, vol. 9, no. 2, pp. 131–152, 1996.

[14] B. Schneier and J. Kelsey, "Secure audit logs to support computer forensics," *ACM Transactions on Information and System Security*, vol. 2, no. 2, pp. 159–176, 1999.

[15] W. M. P. van der Aalst, A. H. M. ter Hofstede, B. Kiepuszewski, and A. P. Barros, "Workflow patterns," *Distributed and Parallel Databases*, vol. 14, no. 1, pp. 5–51, 2003.

[16] M. Abadi, M. Burrows, B. Lampson, and G. Plotkin, "A calculus for access control in distributed systems," *ACM Transactions on Programming Languages and Systems*, vol. 15, no. 4, pp. 706–734, 1993.

[17] V. C. Hu, D. Ferraiolo, D. R. Kuhn, A. Schnitzer, K. Sandlin, R. Miller, and K. Scarfone, "Guide to attribute based access control (ABAC) definition and considerations," NIST, Tech. Rep. Special Publication 800-162, 2014.

[18] N. Leveson, *Engineering a Safer World: Systems Thinking Applied to Safety*. Cambridge, MA: MIT Press, 2011.