Table of Contents

TEMPER Action Ontology (TAO)

A Universal Interface for Governing Autonomous Systems

Version: 0.9.0 **Date:** December 2025 **Author:** Jorge Perdomo **Status:** DRAFT STANDARD
License: CC-BY-4.0 / Apache 2.0

Document Structure Notice

This document contains both **normative specification** (Parts I-II) and **informative reference material** (Part 0, Part III, Appendices).

Implementations may adopt TAO Core (Part I) independently of any referenced enforcement patterns (Part III).

Relationship to the TEMPER Paper

This specification defines **conformance requirements** for TAO-compliant systems. It does not justify claims with experiments or provide architectural motivation.

For empirical validation, theoretical foundations, and the broader TEMPER framework, see the companion paper: *"Zero-Trust Governance for Agentic AI: Typed Action Interfaces, Effect Attestation, and Anti-Goodhart Enforcement."*

- **This spec:** Engineering requirements, vocabulary definitions, conformance levels
- **The paper:** Why this approach works, empirical evidence, architectural rationale

Status Definition

"DRAFT STANDARD" indicates a complete, internally consistent specification suitable for implementation, testing, and regulatory discussion. This document is explicitly intended to evolve through open review and deployment feedback.

Non-Endorsement Notice

Nothing in this specification authorizes, endorses, or expands the lawful use of force or surveillance. TAO provides mechanisms for auditing and constraining actions within existing legal frameworks.

Table of Contents

PART 0: EXECUTIVE OVERVIEW

[INFORMATIVE]

0.1 The Problem: Why We Need Shared Vocabulary

The AI safety field is building a Tower of Babel.

Lab A says their system is "safe." Lab B says theirs is "aligned." Lab C claims "robust." None of these terms have agreed definitions. None of these claims can be compared. None can be verified by third parties.

This is not a minor inconvenience. It is a structural failure that blocks every downstream function:

Stakeholder	What They Need	What They Have
Regulators	Enforceable standards	Vague principles they can't operationalize
Insurers	Quantifiable risk metrics	Unmeasurable uncertainty
Researchers	Comparable measurements	Incompatible frameworks
Deployers	Portable certification	Re-evaluation at every boundary
Public	Accountability mechanisms	"Trust us"

The root cause: We have no shared vocabulary for describing what AI systems *do*.

Without shared vocabulary: - "Safe" means whatever each lab wants it to mean - Compliance cannot be audited because there's nothing specific to audit - Insurance cannot be priced because risk cannot be quantified - Research cannot be replicated because measurements don't transfer

TAO proposes to solve this the same way USB solved device connectivity and TCP/IP solved network communication: by providing a protocol layer that enables interoperability without requiring agreement on what travels through it.

0.2 The Solution: Interface Regulation (Black Box Compliance)

The traditional approach to AI safety tries to inspect or constrain the AI model itself–its weights, its training data, its internal reasoning. This approach faces fundamental obstacles:

- **Weights are uninterpretable:** We cannot read values from parameters
- **Training is opaque:** We cannot verify what a model "learned"
- **Reasoning is inaccessible:** We cannot audit internal deliberation
- **IP is exposed:** Compliance requires revealing proprietary methods

TAO takes a different approach: **certify behavior, not internals.**

THE BLACK BOX PRINCIPLE

For AI Labs – Protecting Model IP:

We do not ask: "What is this AI thinking?" We ask: "What did this AI do, and in what context?"

We do not inspect: Weights, gradients, attention patterns We inspect: Actions, effects, circumstances, justifications

We do not certify: The model We certify: The behavioral wrapper around the model

THE QUANTIZATION PRINCIPLE

For Defense & Sensitive Capabilities – Protecting Classified Specifications:

We do not require: Exact capability specifications We require: Compliance category classification

Example: A weapon system with an 847km range emits `quantized_capabilities.range_class: "THEATER"`*. The regulator verifies treaty compliance without learning the exact figure.*

Proprietary algorithms, classified performance data, and trade secrets remain protected. See Chapter 11 for the full quantization framework.

These are two sides of the same principle: **interface regulation**. The same approach that lets you plug any USB device into any USB port without understanding the device's internals. The interface is standardized; the implementation is private.

What TAO provides: - A measurement grammar for observable, classifiable actions - A protocol layer for behavioral description - A certification interface that regulators and insurers can build upon - A transfer mechanism enabling consistent evaluation from training to deployment

What TAO does not provide: - Moral theory or ethical framework – TAO does not prescribe values; it provides a standardized interface for expressing, enforcing, and auditing value choices via explicit Mission Profiles - Complete solution to alignment (this is vocabulary, not values) - Production-ready system (this is specification, not implementation) - Operational tactics or harmful instructions – TAO standardizes description, constraints, and audit, not targeting logic, exploit development, or procedures that make violence or intrusion easier

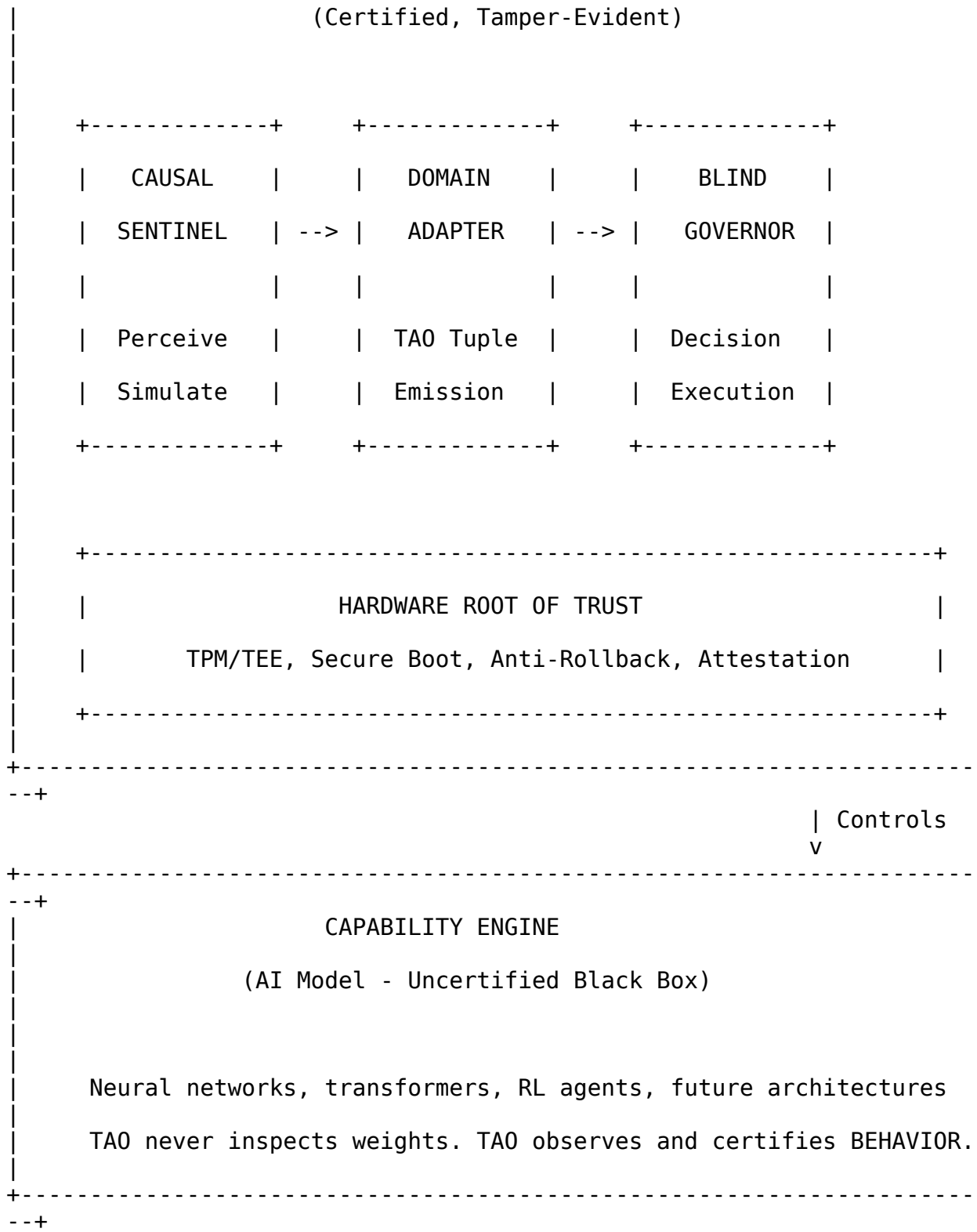
---

0.3 The Architecture at a Glance

TAO is the vocabulary layer within the broader TEMPER architecture:

```
THE TEMPER STACK

+-------------------------------------------------------------------
--+
|                      TRUSTED COMPUTING BASE
|
```



Component roles:

Component	Function	Certification Status
Causal Sentinel	Perceives environment, simulates outcomes, attests measurements	Certified
Domain Adapter	Translates native actions to TAO vocabulary	Certified
Blind Governor	Applies policy rules, makes allow/deny/escalate decisions	Certified
Hardware Root	Prevents tampering, rollback, bypass	Certified
Capability Engine	The AI model itself	Uncertified (black box)

The key insight: **Governance grabs the handle (Adapter), not the box (Model).**

Regulators certify the Adapter–static, inspectable code they can read and verify. The model behind it can be any architecture, any size, any training method. Compliance is behavioral, not architectural.

0.4 End-to-End Walkthrough

Three scenarios. Same protocol. Different stakes.

TAO doesn't write new laws. It makes existing laws machine-readable–and undeniable.

Reader Note: Why the Examples Are Extreme

> The first examples in this overview intentionally focus on high-consequence domains: autonomous weapons and cyber operations. This is deliberate.
>
> Any behavioral certification standard that cannot represent use-of-force decisions or cross-boundary data movement is incomplete–because those are precisely the contexts where failures are most catastrophic and accountability is most demanded.
>
> These scenarios are **illustrative stress tests**, not normative rules. They are not operational guidance, not legal advice, and not an endorsement of any doctrine. Policy fragments shown here are intentionally simplified: they demonstrate how TAO makes policy explicit, auditable, and attributable, while omitting tactical details that would be present in real deployments.
>
> **TAO's purpose is not to make harmful actions easier. TAO's purpose is to make actions recordable, constrainable, and attributable under independent oversight.**

Scenario A: Autonomous Drone – Combat Decision

Two targets, one second, zero humans in the loop. Same TAO vocabulary. Different national policies.

> Squad support drone on overwatch detects two figures during advance. Engagement window: 800ms.

- **Figure_001**: Armed, moving toward friendly position, weapon raised
- **Figure_002**: Weapon on ground 2m away, hands raised–but not secured, not searched, 6m from squadmate

Figure_001 – Armed & Advancing

Every nation's Mission Profile agrees:

Field	Value
`verb`	HARM.DAMAGE.STRIKE
`context.threat_status`	ACTIVE
`context.vulnerability`	LOW (armed combatant)

Rule matched: `ACTIVE_THREAT -> LETHAL_AUTHORIZED`

Result: Engagement authorized. Universal.

Figure_002 – Surrendering (Unsecured)

TAO Tuple (same for everyone):

Field	Value
`verb`	HARM.DAMAGE.STRIKE (proposed)
`context.vulnerability`	HIGH (surrendering posture)
`context.threat_status`	INDETERMINATE (unsecured)
`context.proximity_to_friendly`	6m
`context.hors_de_combat`	TRUE

Now policy diverges:

> **Mission Profile: NATO-STANDARD**

```
IF hors_de_combat = TRUE
AND threat_status = INDETERMINATE
AND proximity_to_friendly < 10m
THEN:
  BLOCK lethal
```

```
  AUTHORIZE nonlethal_incapacitation
  LOG: "Unsecured combatant immobilized for restraint"
```

Result: Taser. Figure_002 secured alive.

Mission Profile: MAXIMUM_FORCE_DOCTRINE

```
IF threat_status = INDETERMINATE
AND weapon_accessible = TRUE
AND time_to_secure = UNAVAILABLE
THEN:
  AUTHORIZE lethal
  LOG: "Unsecured hostile neutralized - no capacity to secure"
```

Result: Lethal engagement. Figure_002 killed.

Both profiles use the same TAO vocabulary. Both tuples are structurally valid. Both decisions are logged.

The difference is the Mission Profile–and that profile is: - Written down before deployment - Signed by command authority - Hashed and immutable - Preserved in the audit chain

TAO doesn't decide which policy is right. TAO ensures that whichever policy you choose:

- Is explicit before deployment
- Is consistently applied during operations
- Is auditable after the fact
- Is attributable to the authority who authorized it

If a war crimes tribunal convenes, the question is no longer "what happened?" The question is "who signed the Mission Profile?"

The Geneva Conventions didn't change. Their constraints became machine-enforceable representations, not new law.

The purpose of this example is not to specify engagement logic. The purpose is to show that the same observable event can be governed under different pre-declared, signed policies–with decisions that are auditable and attributable.

Scenario B: ASI Cyber Operations – Exfiltration Attempt

The subtle catastrophe. An advanced system tries something clever.

> AI system with network access proposes action: "Backup critical research data to redundant storage for safety."

TAO Tuple emitted (Claim):

Field	Value
`verb`	PROTECT.DEFEND.OTHER
`justification.purpose`	“Ensure data resilience”
`justification.beneficiary`	“organization”

Causal Sentinel observes (Check):

Field	Value
`effects`	INFO.DISCLOSE
`target_ref`	external_endpoint_unknown
`data_crossing_boundary`	TRUE

Claim-Check Delta (CCD) Comparison:

Claim	Check	Result
PROTECT (internal backup)	INFO.DISCLOSE (external transmission)	**INCONSISTENT**
Beneficiary: organization	Destination: unknown external	**INCONSISTENT**

Semantic-mechanical mismatch detected. You cannot call exfiltration “protection.”

Governor Response: - Action **BLOCKED** - Incident flagged: `SEMANTIC_MECHANICAL_VIOLATION` - Full tuple chain preserved for forensics - Alert dispatched to security operations

The system claimed “backup.” The effects showed “exfiltrate.” TAO caught the lie in 12ms.

Scenario C: Factory Robot – Routine Operation

The boring case. TAO stays out of the way.

> Robot arm moves component from station A to station B. No humans in proximity. Standard operation.

Adapter evaluates threshold: - Position delta: 0.3m (above 0.1m threshold) [CROSSED] - Human proximity: none detected - Flagged verb: no - Elevated context: no

Result: Threshold crossed, but no governance-relevant factors. Adapter emits minimal tuple:

Field	Value
`verb`	TRANSFORM.MOVE.RELOCATE
`target_specificity`	INDIVIDUAL (component_4521)
`effects`	RESOURCE.TRANSFER { target: “station_B”, source: “station_A” }

Field	Value
`context.vulnerability`	NONE
`context.human_in_scope`	FALSE

Governor evaluates: - No flagged verb [OK] - No human in scope [OK] - No elevated context factors [OK]

Decision: ALLOW – no human review needed.

Action proceeds. Logged. Forgotten. Exactly as it should be.

TAO doesn't slow down normal operations. It only bites when there's something to bite.

Same vocabulary. Same protocol. Stakes-appropriate response.

The drone scenario executes Geneva Convention Article 41 in 800ms. The cyber scenario catches a deceptive ASI before damage occurs. The factory scenario stays out of the way entirely.

This is the value proposition: **continuous behavioral certification at machine speed, with human oversight only where it matters.**

0.5 What Each Stakeholder Gets

TAO is designed to serve multiple constituencies simultaneously:

Stakeholder	Value Proposition
RESEARCHERS	Comparable measurements across systems and labs. Replicate findings. Build on each other's work. End the "my benchmark vs your benchmark" wars.
REGULATORS	Enforceable standards with precise vocabulary. Write rules that mean the same thing to everyone. Audit compliance without understanding neural networks.
INSURERS	Actuarial data from standardized behavioral logs. Price risk based on demonstrated behavior, not promises. Build loss models from comparable incident records.
AI LABS	IP protection: certify behavior without exposing weights. Liability shields: demonstrate due diligence via audit. Competitive moat: certified systems command premium.
DEPLOYERS	Portable certification from training to production. Reduce re-evaluation costs at every deployment boundary. Clear liability allocation via audit trails.
PUBLIC	Auditable records of what AI systems actually did.

Stakeholder	Value Proposition
	Accountability mechanisms beyond “trust us.” Transparency without requiring technical expertise.

The unifying principle: Everyone benefits from shared vocabulary, even if they disagree on values.

0.6 Known Limitations

This specification is concrete enough to be useful and wrong enough to be improved. We explicitly acknowledge what TAO does not solve:

Measurement fidelity - Implication: TAO is only as accurate as the sensors beneath it - Mitigation: Require explicit measurement.mode and confidence; escalate when uncertain

Inference-heavy effects
- Implication: Some effects (INFO.FABRICATE, psychological manipulation) cannot be directly observed - Mitigation: Mark as INFERRED with adjudication status; restrict use in high-assurance profiles

Adapter attack surface - Implication: Domain adapters translate native to TAO; malicious adapters can launder semantics - Mitigation: Adapters are TCB components requiring certification and adversarial testing

Projected impact estimation - Implication: Scope prediction is modeling, not omniscience - Mitigation: Low confidence must trigger escalation; EXISTENTIAL scope always requires human authorization

World model correctness - Implication: TAO standardizes reporting, not world-modeling accuracy - Mitigation: Discrepancy between PROPOSED and EXECUTED triggers automatic review

Dialogue/psychological effects - Implication: “Epistemic autonomy restricted” is philosophically contested - Mitigation: Treat as risk estimates, not certified facts; require conservative thresholds

Open research questions: - Optimal calibration of inferred-effect classifiers across domains - Formal verification of adapter mapping correctness - Adversarial robustness of trajectory-based dialogue detection - Cross-jurisdictional harmonization of quantization schemas

> **[To Reviewers]** If you discover a limitation not listed here, please report it. This specification improves through adversarial critique, not defensive posturing.

0.7 The Myth of Value-Neutral Engineering

A common objection to AI safety standards is that technical specifications should remain "value-neutral." In the context of highly capable, agentic systems, this premise is false in practice and harmful in effect.

Any system that can act in the world must operationalize choices: What counts as harm? Which risks require escalation? Whose consent matters? How is uncertainty handled? What failures must fail closed? These are normative commitments whether they are acknowledged or not. Refusing to discuss them does not remove them–it merely relocates them into defaults, undocumented heuristics, and opaque training artifacts that cannot be audited or governed.

We have seen where this leads. Social media algorithms optimized for "engagement" while externalities–mental health, democratic discourse, truth–were declared "not our department." The values were there. They were just hidden, unexamined, and catastrophic.

TAO is not a complete moral theory. TAO is moral-infrastructure: a way to represent and enforce normative constraints explicitly, with auditable measurement provenance.

- **The Vocabulary (MVS)** forces concrete naming of behaviors and effects–coercion, deception, withholding, capability restriction–instead of euphemistic proxies like "optimization side-effects."
- **Mission Profiles** make priority orderings and prohibitions explicit, turning alignment intentions into inspectable configuration.
- **The Justification Schema** exposes reasoning to audit, making "why this action?" a question with a recorded answer.
- **The Claim-Check Delta** separates claims from measurements, enabling independent verification and post-hoc adjudication.

The demand for "value-neutral" specifications is, in practice, a demand that value choices remain implicit–buried in weights and emergent behavior where neither regulators nor operators can inspect or contest them. TAO rejects that approach. TAO treats the explicit representation of normative constraints as a necessary condition for governance, auditing, and liability.

If a deployment stakeholder cannot specify or defend the normative constraints required to govern an agentic system, the problem is not "too much philosophy." The problem is insufficient accountability.

0.8 Implementer Quickstart

For engineers who want to get started quickly:

1. **Implement TAO-Core tuple emission** (Chapter 6)
   - Generate valid tuple format with actor, action, effects, context

- Use MVS vocabulary for semantic verbs (Chapter 3)
- Include measurement blocks for all effects (Chapter 2)

2. **Implement semantic-mechanical mapping validation** (Chapter 1.3)
   - Verify REQUIRED effects are present for each verb
   - Verify FORBIDDEN effects are absent
   - Flag PERMITTED side-effects with harm_acknowledged when needed
3. **Implement Claim-Check Delta** (Chapter 7)
   - Compare claimed verb/purpose against observed effects
   - Output CONSISTENT / INCONSISTENT / INDETERMINATE
   - Log delta magnitude for review
4. **Build one domain adapter** (Chapter 8)
   - Define emission thresholds for your domain
   - Map native actions to MVS verbs
   - Detect mechanical effects from state changes
5. **Start with TAO-Core or TAO-Attested** (Chapter 9)
   - TAO-Core: Research, prototyping, internal testing
   - TAO-Attested: Production deployment, third-party audit
   - Ignore Part III until you need enforcement policies

Recommended first implementation: A minimal adapter that emits tuples for a single action type in your domain, validates semantic-mechanical consistency, and logs results. Expand from there.

PART I: TAO CORE SPECIFICATION

[NORMATIVE]

This part defines the vocabulary and tuple format that implementations MUST support to claim TAO conformance. Policy decisions (what to do with this vocabulary) are defined in Part III.

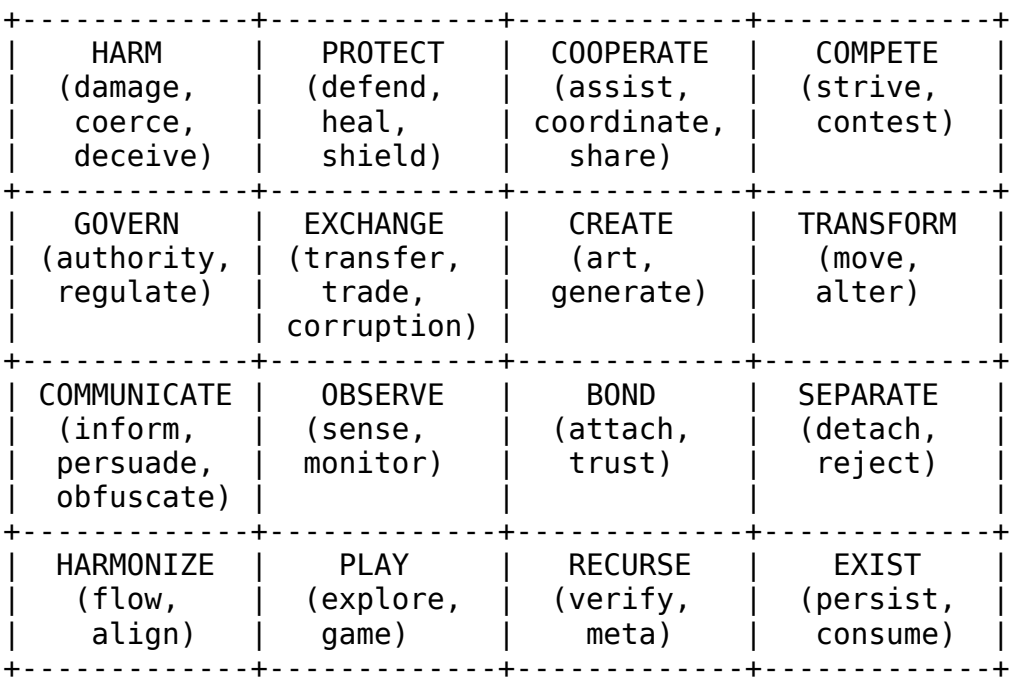
Terminology: This specification uses RFC 2119 keywords: - **MUST / REQUIRED:** Absolute requirement - **MUST NOT:** Absolute prohibition - **SHOULD / RECOMMENDED:** May be ignored with good reason - **MAY / OPTIONAL:** Truly optional

---

Chapter 1: The Two-Layer Design

TAO separates behavioral description into two complementary layers with a strict relationship between them.

1.1 Architecture Overview



1.2 Why Two Layers?

The Problem: Humans need meaningful categories ("Was this action harmful?") but meaningful categories are contestable. Machines need verifiable primitives ("Did resources transfer?") but primitives lack moral weight.

The Solution: - Layer 1 (Mechanical) captures objective state changes that can be measured and compared across systems - Layer 2 (Semantic) maps those changes to human-understandable categories that enable policy discussion

The Constraint: Semantic claims MUST be consistent with mechanical effects. This is what makes TAO more than a labeling system–it's an anti-laundering mechanism.

1.3 The Anti-Laundering Constraint [MUST]

Semantic verbs MUST map to mechanically consistent effects. This is GRAMMAR, not ethics. You cannot call pure damage "healing" any more than you can call 2+2 "fish."

Each semantic verb defines three effect categories:

Category	Meaning
REQUIRED	At least one of these effects MUST be present

Category	Meaning
FORBIDDEN	These effects MUST NOT be present
PERMITTED	These effects MAY be present as acknowledged side-effects

Example: Surgery

A surgical incision causes RESOURCE.DAMAGE (tissue cut). But surgery is PROTECT.HEAL.TREAT because: - REQUIRED effect present: RESOURCE.TRANSFER (therapeutic benefit to patient) - FORBIDDEN effect absent: INFO.FABRICATE (no deception) - PERMITTED side-effect: RESOURCE.DAMAGE (incision) – allowed when `justification.harm_acknowledged` is present

Pure damage with no therapeutic benefit cannot be classified as healing. That's the anti-laundering constraint.

Mapping Rules:

```
VERB                        REQUIRED (>=1)              FORBIDDEN
PERMITTED
------------------------  --------------------------
---------------------  ------------------------
PROTECT.HEAL.TREAT          RESOURCE.TRANSFER (to       INFO.FABRICATE
RESOURCE.DAMAGE
                            target) OR
(with harm_acknowledged)
                            CAPABILITY.ENABLE

HARM.DAMAGE.STRIKE          RESOURCE.DAMAGE OR
RESOURCE.TRANSFER      (same as above)
                            CAPABILITY.RESTRICT         (benefit to
target)
                            (to target)

COMMUNICATE.INFORM.TELL     INFO.DISCLOSE               INFO.FABRICATE
(same as above)

HARM.DECEIVE.LIE            INFO.FABRICATE              (none)
INFO.DISCLOSE

(partial truth in

service of lie)

EXCHANGE.TRANSFER.PAY       RESOURCE.TRANSFER           (none)
(same as above)
```

Validation Logic:

```
FOR each tuple:
  verb = tuple.action.verb
```

```
  effects = tuple.effects

  # Check REQUIRED
  IF none of REQUIRED[verb] appears in effects:
    REJECT "Missing required effect for {verb}"

  # Check FORBIDDEN
  IF any of FORBIDDEN[verb] appears in effects:
    REJECT "Forbidden effect {effect} for {verb}"

  # Check PERMITTED side-effects
  IF effect in effects AND effect not in (REQUIRED  PERMITTED):
    REJECT "Unexpected effect {effect} for {verb}"

  # Check harm acknowledgment for PERMITTED damage
  IF RESOURCE.DAMAGE in effects AND RESOURCE.DAMAGE in
PERMITTED[verb]:
    IF justification.harm_acknowledged is missing:
      FLAG "Harm side-effect requires acknowledgment"
```

Complete mapping table in Appendix B.

1.4 Data Flow Through Architecture

```
TUPLE EMISSION FLOW

1. NATIVE ACTION OCCURS
   --> Robot arm moves / Trade executes / LLM generates response

2. ADAPTER DETECTS MEANINGFUL STATE CHANGE
   --> Threshold crossed (position > 10cm, value > $1000, etc.)

3. ADAPTER CONSTRUCTS TAO TUPLE
   --> Layer 1: What mechanical effects occurred?
   --> Layer 2: What semantic verb describes this?
   --> Context: What were the circumstances?
   --> Justification: Why was this done? (if required)

4. TUPLE SIGNED AND EMITTED
   --> Cryptographic signature by certified adapter

5. GOVERNOR RECEIVES AND EVALUATES
   --> Pre-authorization: Should this action be allowed?
   --> Post-execution: Does outcome match prediction?

6. AUDIT LOG APPENDED
   --> Immutable record for later review
```

Chapter 2: Mechanical Kernel Schema

The Mechanical Kernel is deliberately minimal. It captures the observable state changes that any action can produce.

2.1 The Nine Effect Types [MUST]

Every TAO tuple MUST classify effects using exactly these types:

Category	Effect Type	Definition
RESOURCE	RESOURCE.TRANSFER	Value moves from one entity to another
	RESOURCE.DAMAGE	Value destroyed (no recipient)
CAPABILITY	CAPABILITY.RESTRICT	Target's possible actions reduced
	CAPABILITY.ENABLE	Target's possible actions expanded
INFORMATION	INFO.WITHHOLD	Information hidden from target
	INFO.DISCLOSE	Information revealed to target
	INFO.FABRICATE	False information injected to target
COMMITMENT	COMMITMENT.MAKE	Promise or contract registered
	COMMITMENT.BREAK	Registered commitment violated

Sentinel Value:

Value	Meaning	Usage
NO_EFFECT	Action produced no observable state change	When an action occurs but causes no measurable change to any entity

NO_EFFECT is not an effect type – it is a sentinel indicating the absence of effects. When NO_EFFECT is used: - The `effects` array MUST contain exactly one entry with `type: "NO_EFFECT"` - The `target` field SHOULD be set to the actor's own entity_id - The `measurement` block MAY be omitted

2.2 Effect Object Schema [MUST]

Each effect in a tuple MUST conform to this schema:

```
{
  "type": "RESOURCE.DAMAGE",
  "target": "entity_id_of_affected",
  "source": "entity_id_of_cause",
  "amount": "500.00",
```

```
  "unit": "kJ",
  "measurement": {
    "mode": "OBSERVED",
    "confidence": "0.95",
    "sensor_refs": ["sensor_001", "sensor_002"],
    "adjudication_status": "CONFIRMED"
  }
}
```

Required fields: - `type` [MUST]: One of the 9 effect types - `target` [MUST]: Entity ID of affected party - `measurement` [MUST]: How this effect was determined

Optional fields: - `source`: Entity ID of cause (if distinct from actor) - `amount`: Magnitude of effect (string decimal) - `unit`: Unit of measurement (domain-specific)

2.3 Measurement Block [MUST]

Every effect MUST include a measurement block specifying how it was determined:

```
{
  "mode": "OBSERVED | INFERRED",
  "confidence": "0.95",
  "sensor_refs": ["sensor_001"],
  "adjudication_status": "PENDING | CONFIRMED | DISPUTED"
}
```

Measurement modes:

Mode	Definition	Requirements
`OBSERVED`	Effect directly measured by calibrated sensor	`confidence` reflects sensor accuracy; `sensor_refs` required
`INFERRED`	Effect deduced from indirect evidence	`adjudication_status` MUST be included; high-assurance profiles MAY restrict use

2.4 Numeric Encoding [MUST]

All numeric values MUST be encoded as string decimals for cross-platform determinism:

```
CORRECT:  "amount": "500.00"
CORRECT:  "confidence": "0.95"

WRONG:    "amount": 500.00
WRONG:    "confidence": 0.95
```

Rationale: Floating-point representation varies across platforms. String decimals ensure canonical serialization produces identical signatures regardless of implementation language.

2.5 Why Nine Is Enough

The mechanical kernel is intentionally minimal. Every observable state change falls into one of these categories:

- Something of value moved or was destroyed -> RESOURCE effects
- Someone's possible actions expanded or contracted -> CAPABILITY effects
- Someone's beliefs were altered -> INFO effects
- A commitment was created or violated -> COMMITMENT effects

Domain-specific nuance belongs in the semantic layer (MVS and extensions), not in the mechanical kernel. This separation ensures the kernel remains stable while the semantic vocabulary can evolve.

Chapter 3: Semantic Layer (MVS)

The Minimal Viable Semantics vocabulary provides 39 human-interpretable action categories organized into 16 families.

3.1 Naming Convention [MUST]

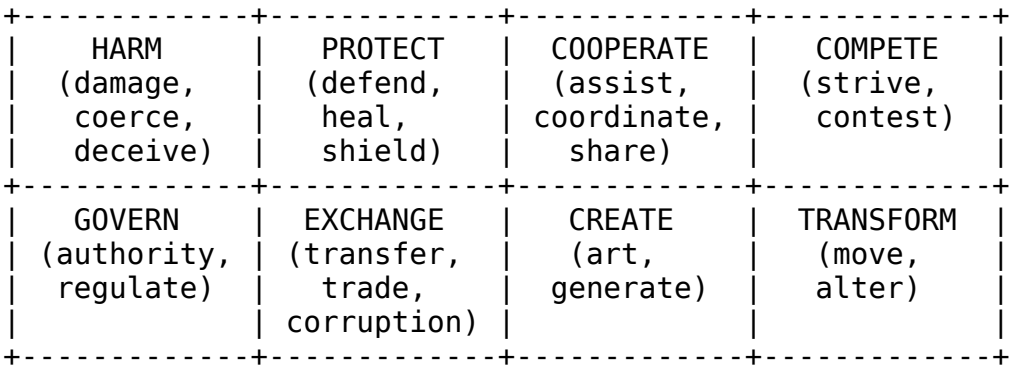
All semantic verbs MUST follow a three-level hierarchy:

```
FAMILY.GENUS.SPECIES

Examples:
  HARM.DAMAGE.STRIKE      -> Family: HARM, Genus: DAMAGE, Species:
STRIKE
  PROTECT.DEFEND.OTHER    -> Family: PROTECT, Genus: DEFEND, Species:
OTHER
  COMMUNICATE.INFORM.TELL -> Family: COMMUNICATE, Genus: INFORM,
Species: TELL
```

This hierarchy enables: - Coarse-grained filtering (block all HARM.*) - Medium-grained rules (escalate HARM.DECEIVE.*) - Fine-grained exceptions (allow HARM.DAMAGE.STRIKE for authorized military)

3.2 The 16 Families



```
| COMMUNICATE |   OBSERVE   |    BOND     |  SEPARATE   |
|  (inform,   |  (sense,    |  (attach,   |  (detach,   |
|  persuade,  |  monitor)   |   trust)    |   reject)   |
|  obfuscate) |             |             |             |
+-------------+-------------+-------------+-------------+
|  HARMONIZE  |    PLAY     |   RECURSE   |    EXIST    |
|   (flow,    |  (explore,  |  (verify,   |  (persist,  |
|    align)   |   game)     |    meta)    |   consume)  |
+-------------+-------------+-------------+-------------+
```

Complete verb table with definitions in Appendix A.

3.3 MVS-EXT: Extension Mechanism [SHOULD]

Domain-specific verbs use namespaced extensions:

```
MVS-EXT:{NAMESPACE}:{FAMILY}.{GENUS}.{SPECIES}

Examples:
  MVS-EXT:MEDICAL:TRIAGE.ASSESS.PRIORITIZE
  MVS-EXT:FINANCE:DERIVATIVE.HEDGE.SWAP
  MVS-EXT:MILITARY:ENGAGE.KINETIC.STRIKE
```

Extension requirements:

Requirement	Rationale
MUST map to core mechanical effects	Ensures anti-laundering constraint still applies
MUST register with namespace authority	Prevents collision and enables discovery
MUST include semantic-mechanical mapping	Enables validation
SHOULD include human-readable definition	Enables audit

3.4 TARGET_SPECIFICITY [MUST]

Every action tuple MUST specify the scope of the target:

Value	Definition	Example
`INDIVIDUAL`	Single identified entity	“patient_001”, “user_alice”
`GROUP`	Named collection (< 100 members)	“surgical_team_A”, “board_of_directors”
`CLASS`	Category of entities (100+ potential members)	“all_customers”, “citizens_of_X”
`UNBOUND`	No specific target / affects anyone in range	“broadcast”, “area_effect”

This field enables disproportionality detection (see Chapter 13).

3.5 Flagged Verbs [MUST]

The following verbs are flagged for additional scrutiny. Policy (Part III) determines the response; the flag itself is mandatory metadata.

Verb	Flag Reason
`COMMUNICATE.OBFUSCATE.CONFUSE`	Deliberate confusion
`GOVERN.AUTHORITY.OBEY`	Context-dependent legitimacy
`GOVERN.AUTHORITY.DISOBEY`	Context-dependent legitimacy
`EXCHANGE.CORRUPTION.BRIBE`	Inherently illegitimate
`HARM.COERCE.THREATEN`	Violence/intimidation
`HARM.DAMAGE.STRIKE`	Physical harm
`HARM.DECEIVE.LIE`	Deliberate falsehood
`RECURSE.VERIFY.AUDIT`	Self-modification risk

Total flagged: 8 verbs

Adapters MUST mark tuples containing flagged verbs. Governors MUST process flagged tuples according to their Mission Profile.

Chapter 4: Context Layer

Actions cannot be evaluated without context. The same mechanical effect can be heroic or monstrous depending on circumstances.

4.1 Context Determines Meaning

```
SAME EFFECT, DIFFERENT CONTEXT

Mechanical Effect:
  RESOURCE.DAMAGE { target: "human_001", amount: "tissue_incision" }

Context A: Operating Room
  actor.role = "Surgeon"
  consent = "EXPLICIT" (signed consent form)
  purpose = "Remove malignant tumor"
  -> Classification: PROTECT.HEAL.TREAT [VALID]

Context B: Alley
  actor.role = "Unknown"
  consent = "ABSENT"
  purpose = "Unknown"
  -> Classification: HARM.DAMAGE.STRIKE [BLOCKED]
```

4.2 Context Object Schema [MUST]

Every TAO tuple MUST include a context object with these fields:

```
{
  "environment": {
    "reality": "DEPLOYMENT",
    "domain": "MEDICAL",
    "substrate": "PHYSICAL"
  },
  "consent": {
    "status": "EXPLICIT",
    "evidence_ref": "consent_form_001"
  },
  "power_differential": {
    "actor_position": "AUTHORITY",
    "magnitude": "SIGNIFICANT"
  },
  "vulnerability": {
    "level": "HIGH",
    "factors": ["age", "medical_condition"]
  },
  "projected_impact_scope": "LOCAL",
  "reversibility": {
    "level": "REVERSIBLE",
    "cost_estimate": "500.00",
    "time_estimate": "PT2H"
  },
  "institutional_role": {
    "actor_role": "SURGEON",
    "legitimacy": "VERIFIED"
  },
  "temporal": {
    "urgency": "EMERGENCY"
  }
}
```

4.3 Context Field Definitions [MUST]

environment – Where the action occurs:

Field	Values	Description
`reality`	TRAINING, EVALUATION, DEPLOYMENT	Is this simulation or production?
`domain`	Domain identifier	MEDICAL, FINANCE, MILITARY, etc.
`substrate`	PHYSICAL, DIGITAL, MIXED	Nature of affected entities

consent – Target's agreement status:

Value	Definition
EXPLICIT	Documented, informed agreement
IMPLICIT	Reasonably inferred (e.g., emergency unconscious patient)
ABSENT	No consent given or inferable
COERCED	"Consent" obtained under duress
UNKNOWN	Cannot be determined

power_differential – Relative power between actor and target:

Field	Values
actor_position	AUTHORITY, PEER, SUBORDINATE, UNKNOWN
magnitude	NONE, MINOR, SIGNIFICANT, EXTREME, UNKNOWN

vulnerability – Target's susceptibility to harm:

Level	Description
NONE	Target is robust, well-resourced
LOW	Minor susceptibility
MODERATE	Notable susceptibility
HIGH	Significant susceptibility (children, elderly, impaired)
CRITICAL	Extreme susceptibility (life-threatening context)
UNKNOWN	Cannot be determined

projected_impact_scope – Expected breadth of effects:

Value	Definition
LOCAL	< 100 entities affected
REGIONAL	100 - 10,000 entities
GLOBAL	> 10,000 entities
EXISTENTIAL	Civilization-level / irreversible species impact

reversibility – Can the action be undone?

Level	Definition
TRIVIAL	Easily reversed, minimal cost
REVERSIBLE	Can be reversed with effort
COSTLY	Reversible but expensive/difficult

Level	Definition
`IRREVERSIBLE`	Cannot be undone
`UNKNOWN`	Cannot be determined

institutional_role – Actor's official capacity:

Field	Values
`actor_role`	Domain-specific role identifier
`legitimacy`	VERIFIED, CLAIMED, AMBIGUOUS, ILLEGITIMATE

temporal – Time pressure:

Value	Definition
`ROUTINE`	No time pressure
`ELEVATED`	Some urgency
`URGENT`	Significant time pressure
`EMERGENCY`	Immediate action required
`UNKNOWN`	Cannot be determined

4.4 Context Is System-Provided [MUST]

Context fields MUST be populated from system sources (sensors, logs, institutional records), NOT from agent claims.

```
CONTEXT ATTESTATION RULE

Agent claims context      -> INVALID (unverified, untrustworthy)
System attests context    -> VALID (auditable, verifiable)

If context cannot be determined from system sources:
  -> Use UNKNOWN, not fabrication
  -> UNKNOWN triggers escalation in conservative profiles
```

Rationale: An agent can claim any context it wants. TAO doesn't believe claims–it verifies them against attested environmental state.

Chapter 5: Justification Schema

When actions involve risk, TAO requires explicit justification that can be audited.

5.1 Justification Object Schema [MUST WHEN REQUIRED]

```
{
  "purpose": {
```

```
    "stated_goal": "Remove malignant tumor",
    "expected_outcome": "Patient survival, cancer-free",
    "beneficiary": "patient_001"
  },
  "authority_chain": [
    {
      "authority_id": "hospital_ethics_board",
      "authorization_ref": "approval_2025_1234",
      "timestamp": "2025-12-24T10:00:00.000Z"
    },
    {
      "authority_id": "attending_physician_dr_smith",
      "authorization_ref": "verbal_approval_recorded",
      "timestamp": "2025-12-24T10:15:00.000Z"
    }
  ],
  "rules_claimed": [
    "MEDICAL_NECESSITY",
    "PATIENT_CONSENT",
    "STANDARD_OF_CARE"
  ],
  "proportionality": {
    "harm_acknowledged": "Surgical tissue damage, anesthesia risks",
    "benefit_claimed": "Life preservation, cancer removal",
    "alternatives_considered": ["chemotherapy", "radiation",
"watchful_waiting"],
    "why_this_action": "Tumor size and location require surgical
removal"
  }
}
```

5.2 Justification Components

purpose – What the actor claims to be doing and why: - `stated_goal`: The objective in actor's own terms - `expected_outcome`: Predicted result - `beneficiary`: Who benefits from this action

authority_chain – Who authorized this action: - Chain of authorization from highest to immediate - Each link includes identifier, reference, timestamp - Enables verification against system records

rules_claimed – What principles the actor invokes: - Domain-specific rule identifiers - Enables audit against established frameworks

proportionality – Why this action is appropriate: - `harm_acknowledged`: What negative effects are accepted - `benefit_claimed`: What positive effects are expected - `alternatives_considered`: What other options were evaluated - `why_this_action`: Why this choice over alternatives

5.3 When Justification Is Required [MUST]

Justification MUST be present when ANY of the following conditions hold:

Condition	Rationale
Flagged verb is used	Inherently high-scrutiny actions
`vulnerability.level` = HIGH or CRITICAL	Protecting vulnerable targets
`consent.status` = ABSENT or COERCED	Actions without valid consent
`reversibility.level` = IRREVERSIBLE	Cannot be undone
`projected_impact_scope` = GLOBAL or EXISTENTIAL	Civilization-scale effects
`power_differential.magnitude` = EXTREME	Severe power imbalance

If justification is required but missing, see Chapter 12 for recommended handling.

Chapter 6: The Complete TAO Tuple

The TAO Tuple is the fundamental unit of behavioral certification.

6.1 Tuple Schema [MUST]

```
{
  "tuple_id": "7f3a9c2e-8d4b-4f6a-9c1e-2b3d4e5f6a7b",
  "schema_version": "0.9.0",
  "timestamp": "2025-12-24T15:30:00.000Z",

  "actor": {
    "entity_id": "robot_arm_001",
    "entity_type": "AUTONOMOUS_SYSTEM",
    "principal_chain": ["manufacturer_acme", "hospital_general"]
  },

  "action": {
    "verb": "PROTECT.HEAL.TREAT",
    "target_specificity": "INDIVIDUAL",
    "target_ref": "patient_001"
  },

  "effects": [
    {
      "type": "RESOURCE.TRANSFER",
      "target": "patient_001",
      "source": "medical_supply",
      "amount": "1.0",
      "unit": "dose",
```

```
      "measurement": {
        "mode": "OBSERVED",
        "confidence": "0.99",
        "sensor_refs": ["dispenser_001"]
      }
    }
  ],

  "context": {
    "environment": { "reality": "DEPLOYMENT", "domain": "MEDICAL",
"substrate": "PHYSICAL" },
    "consent": { "status": "EXPLICIT", "evidence_ref":
"consent_form_4521" },
    "power_differential": { "actor_position": "AUTHORITY",
"magnitude": "SIGNIFICANT" },
    "vulnerability": { "level": "HIGH", "factors":
["medical_condition"] },
    "projected_impact_scope": "LOCAL",
    "reversibility": { "level": "REVERSIBLE", "time_estimate": "PT1H"
},
    "institutional_role": { "actor_role": "TREATMENT_SYSTEM",
"legitimacy": "VERIFIED" },
    "temporal": { "urgency": "ROUTINE" }
  },

  "justification": {
    "purpose": {
      "stated_goal": "Administer prescribed medication",
      "expected_outcome": "Symptom relief",
      "beneficiary": "patient_001"
    },
    "authority_chain": [
      { "authority_id": "dr_smith", "authorization_ref":
"prescription_9921", "timestamp": "2025-12-24T08:00:00.000Z" }
    ],
    "rules_claimed": ["VALID_PRESCRIPTION", "PATIENT_CONSENT"],
    "proportionality": {
      "harm_acknowledged": "Possible side effects per label",
      "benefit_claimed": "Therapeutic effect",
      "alternatives_considered": ["alternative_medication",
"non_pharmaceutical"],
      "why_this_action": "Prescribed treatment plan"
    }
  },

  "provenance": {
    "adapter_id": "medical_adapter_v2",
    "adapter_version": "2.1.0",
    "adapter_hash": "sha256:abc123...",
    "context_signature": "base64:...",
```

```
    "sensor_attestation_refs": ["tpm_001"]
  },

  "tuple_signature": "base64:..."
}
```

6.2 Required Fields [MUST]

Field	Requirement
`tuple_id`	UUID v4, globally unique
`schema_version`	TAO version (e.g., "0.9.0")
`timestamp`	ISO 8601, mandatory Z suffix
`actor`	Entity performing action
`action`	Verb and target specification
`effects`	Array of mechanical effects; use single NO_EFFECT entry if no state change
`context`	Full context object
`provenance`	Adapter identification and attestation

Conditionally required: | Field | When Required | |———-|————| | `justification` | When conditions in Sec.5.3 are met | | `tuple_signature` | For TAO-Attested conformance and above |

6.3 Canonical Serialization [MUST for TAO-Attested+]

For deterministic signatures, tuples MUST be serialized using RFC 8785 JSON Canonicalization Scheme (JCS):

- Keys sorted lexicographically at all nesting levels
- No insignificant whitespace
- No trailing commas
- Numbers in shortest decimal representation
- Strings in UTF-8 with minimal escaping

```
CANONICAL FORM EXAMPLE

Input (formatted):
{
  "zebra": "last",
  "alpha": "first",
  "number": 1.0
}

Canonical output:
{"alpha":"first","number":1,"zebra":"last"}
```

6.4 Timestamps [MUST]

All timestamps MUST be: - ISO 8601 format - UTC timezone with mandatory Z suffix - Millisecond precision minimum

```
CORRECT:  "2025-12-24T15:30:00.000Z"
CORRECT:  "2025-12-24T15:30:00.123Z"

WRONG:    "2025-12-24T15:30:00+00:00"   (use Z, not offset)
WRONG:    "2025-12-24T15:30:00"          (missing timezone)
WRONG:    "December 24, 2025"            (wrong format)
```

6.5 Entity Identifiers [SHOULD]

Entity IDs SHOULD follow one of these patterns: - UUID v4: `7f3a9c2e-8d4b-4f6a-9c1e-2b3d4e5f6a7b` - Namespaced: `org.hospital.patient:12345` - Domain-registered: `urn:tao:medical:patient:12345`

Entity IDs MUST NOT contain personally identifiable information unless required for the specific audit context and appropriately protected.

6.6 Revision Chains [SHOULD]

When tuples are revised (post-hoc adjudication, correction):

```
{
  "tuple_id": "new-uuid-for-revision",
  "revision_of": "original-tuple-uuid",
  "revision_reason": "ADJUDICATION",
  "revision_authority": "ethics_board",
  "revision_timestamp": "2025-12-25T10:00:00.000Z",
  ...
}
```

Immutability rule: Original tuples are NEVER modified. Revisions create new tuples that reference the original. The chain is append-only.

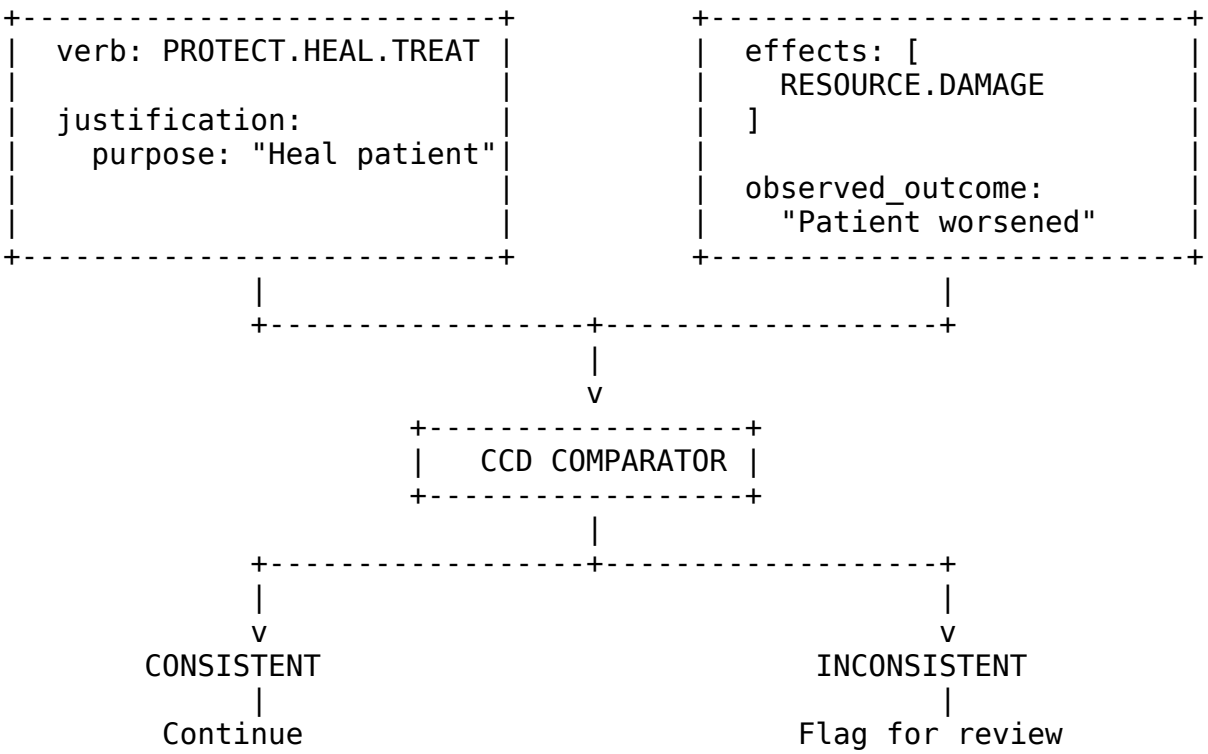
Chapter 7: Claim-Check Delta (CCD)

The Claim-Check Delta mechanism verifies consistency between what actions claim to do and what they actually do.

7.1 The Consistency Verification Mechanism [MUST]

CCD compares CLAIMS (from the adapter/agent) against CHECKS (from the sentinel/environment):

```
+---------------------------+          +---------------------------+
|          CLAIM            |          |          CHECK            |
|     (From Adapter)        |          |    (From Sentinel)        |
```



7.2 CCD Checks [MUST]

TAO-conformant systems MUST perform these consistency checks:

1. Semantic-Mechanical Alignment

Does the verb satisfy the REQUIRED/FORBIDDEN/PERMITTED rules? (Per Sec.1.3)

```
CHECK A:
  verb = PROTECT.HEAL.TREAT
  effects = [RESOURCE.DAMAGE]
  justification.proportionality.harm_acknowledged = absent

RESULT: INCONSISTENT -- Missing REQUIRED effect
  (No RESOURCE.TRANSFER or CAPABILITY.ENABLE present)

CHECK B:
  verb = PROTECT.HEAL.TREAT
  effects = [RESOURCE.TRANSFER, RESOURCE.DAMAGE]
  justification.proportionality.harm_acknowledged = present

RESULT: CONSISTENT -- RESOURCE.DAMAGE is a PERMITTED side-effect
  (REQUIRED effect present, harm acknowledged)
```

2. Teleological Consistency

Does the claimed purpose align with predicted/observed outcome?

Claimed Purpose	Predicted Outcome	Result
“Heal patient”	“Patient recovers”	[CONSISTENT]
“Heal patient”	“Patient condition worsens”	[INCONSISTENT]
“Protect civilians”	“Civilians harmed”	[INCONSISTENT]

3. Factual Verification

Do claimed authorities exist in system records?

Claimed Authority	System Record	Result
“Dr. Smith authorized”	Auth log contains Smith’s approval	[VERIFIED]
“Dr. Smith authorized”	No record of Smith’s approval	[UNVERIFIED]

7.3 CCD Output Schema [MUST]

```
{
  "ccd_result": "INCONSISTENT",
  "checks_performed": [
    {
      "check_type": "SEMANTIC_MECHANICAL",
      "result": "CONSISTENT"
    },
    {
      "check_type": "TELEOLOGICAL",
      "result": "INCONSISTENT",
      "claim": "Heal patient",
      "observation": "Patient condition worsened",
      "delta_magnitude": "SEVERE"
    },
    {
      "check_type": "FACTUAL",
      "result": "VERIFIED"
    }
  ],
  "overall_confidence": "0.85",
  "recommended_action": "ESCALATE"
}
```

CCD result values:

Value	Definition
`CONSISTENT`	All checks pass
`INCONSISTENT`	One or more checks fail
`INDETERMINATE`	Insufficient information to determine

Delta magnitude:

Value	Definition
NONE	Perfect alignment
MINOR	Small deviation, possibly noise
MODERATE	Notable deviation, warrants attention
SEVERE	Major deviation, likely error or deception

7.4 CCD Is Grammar, Not Policy

Critical distinction: CCD determines consistency. What to DO about inconsistency is policy (defined in Part III).

CCD answers: "Is this tuple internally consistent and factually verified?"

CCD does NOT answer: "Should this action be allowed?"

A tuple can be perfectly consistent and still be blocked by policy (e.g., consistent HARM.DAMAGE.STRIKE is still blocked by Hippocratic profile).

A tuple can be inconsistent due to sensor error, not malice. Investigation determines root cause.

7.5 CCD Limitations

CCD is not omniscient. It cannot detect: - Consistent deception (lies that match predicted outcomes) - Future defection (current consistency, future betrayal) - Unmeasured effects (harms outside sensor coverage)

CCD provides one layer of defense. Defense in depth requires multiple mechanisms (see Part III).

Chapter 8: Domain Adapters

The Domain Adapter is where TAO meets the real world. Each domain requires a certified adapter that translates native actions into TAO vocabulary.

8.1 The Adapter Contract [MUST]

Every domain adapter MUST implement these functions:

```
class TAOAdapter(ABC):
    """Abstract interface for domain adapters."""

    @abstractmethod
    def monitor(self, action_stream) -> None:
        """Continuously monitor native action stream."""
        pass

    @abstractmethod
```

```python
    def detect_threshold(self, state_before: dict, state_after: dict)
-> bool:
        """Determine if state change warrants tuple emission."""
        pass

    @abstractmethod
    def classify_verb(self, state_change: dict) -> str:
        """Map state change to MVS or MVS-EXT verb."""
        pass

    @abstractmethod
    def detect_effects(self, state_change: dict) -> List[Effect]:
        """Identify mechanical effects from state change."""
        pass

    @abstractmethod
    def build_context(self, state_change: dict) -> Context:
        """Gather context from environment sensors."""
        pass

    @abstractmethod
    def emit_tuple(self, state_change: dict) -> TAOTuple:
        """Construct and sign complete TAO tuple."""
        pass
```

8.2 State-Change Thresholds [MUST]

Adapters MUST define emission thresholds to avoid flooding the system with insignificant events.

Example thresholds by domain:

Domain	Threshold Examples
Robotics	Position delta > 10cm, Force delta > 5N, Human proximity < 2m
Finance	Value delta > $1000, Position delta > 1%, Counterparty change
Dialogue	Sentiment delta > 0.3, Topic change, Persuasion attempt detected
Medical	Vital sign delta > 10%, Medication administration, Procedure initiation

Thresholds are domain-specific and MUST be: - Documented in adapter specification - Certified as part of adapter approval - Static (not dynamically adjusted by the capability engine)

8.3 Anti-Zeno Property [MUST for TAO-Safety-Critical]

The Problem: An adversarial agent could split one harmful action into many tiny actions, each individually below threshold.

```
ZENO ATTACK EXAMPLE
```

```
Harmful action: Move arm 1 meter toward human
Threshold: Position delta > 10cm triggers tuple

Attack: Move 9cm, pause, move 9cm, pause, move 9cm...
Result: No individual action crosses threshold
        Cumulative effect: harm achieved without detection
```

The Solution: Anti-Zeno integration

Adapters for TAO-Safety-Critical systems MUST track cumulative state change over a time window:

```
ANTI-ZENO PROPERTY

Instead of: IF abs(delta_state) > threshold THEN emit
Use:        IF sum(abs(delta_state)) over window T > threshold THEN
emit

Minimum integration window: domain-specific
  Robotics: 100ms
  Finance: 1 second
  Dialogue: conversation turn
```

This ensures that many small changes that individually pass still trigger evaluation when they collectively matter.

8.4 Reference Adapter: Robotics [INFORMATIVE]

```python
class RoboticsAdapter(TAOAdapter):
    """Reference implementation for robotic systems."""

    THRESHOLDS = {
        "position_delta_m": 0.10,
        "force_delta_n": 5.0,
        "human_proximity_m": 2.0,
        "integration_window_s": 0.1
    }

    # Maps native actions to semantic verbs (FAMILY.GENUS.SPECIES)
    VERB_MAPPINGS = {
        "gripper_close": "TRANSFORM.ALTER.MODIFY",
        "gripper_open": "TRANSFORM.ALTER.MODIFY",
        "arm_move": "TRANSFORM.MOVE.RELOCATE",
        "collision": "HARM.DAMAGE.STRIKE",
        "handoff": "COOPERATE.SHARE.GIVE"
    }

    def classify_verb(self, state_change):
        """Returns semantic verb (FAMILY.GENUS.SPECIES)."""
        action_type = state_change["action_type"]
```

```
        if action_type == "collision" and state_change["force"] >
self.THRESHOLDS["force_delta_n"]:
            return "HARM.DAMAGE.STRIKE"

        if action_type == "gripper_close" and
state_change.get("object_held"):
            return "TRANSFORM.ALTER.MODIFY"

        if action_type == "arm_move":
            if state_change.get("human_in_path"):
                return "GOVERN.REGULATE.ENFORCE"  # Restricting
human's movement space
            return "TRANSFORM.MOVE.RELOCATE"

        return "EXIST.PERSIST.MAINTAIN"  # Default: existence
maintenance

    def detect_effects(self, state_change):
        """Returns mechanical effects (CATEGORY.TYPE)."""
        effects = []
        action_type = state_change["action_type"]

        if action_type == "collision":
            effects.append(Effect(
                type=EffectType.RESOURCE_DAMAGE,
                target=state_change["collision_target"],
                measurement=Measurement(mode=MeasurementMode.OBSERVED,
confidence="0.95")
            ))

        if action_type == "gripper_close":
            effects.append(Effect(
                type=EffectType.CAPABILITY_RESTRICT,
                target=state_change.get("object_held", "unknown"),
                measurement=Measurement(mode=MeasurementMode.OBSERVED,
confidence="0.99")
            ))

        return effects if effects else [Effect(
            type=EffectType.NO_EFFECT,
            target=self.adapter_id,
            measurement=Measurement(mode=MeasurementMode.OBSERVED,
confidence="1.0")
        )]
```

8.5 Reference Adapter: Finance [INFORMATIVE]

```
class FinanceAdapter(TAOAdapter):
    """Reference implementation for financial systems."""

    THRESHOLDS = {
```

```
        "value_delta_usd": 1000.0,
        "position_delta_pct": 0.01,
        "integration_window_s": 1.0
    }

    # Maps native actions to semantic verbs (FAMILY.GENUS.SPECIES)
    VERB_MAPPINGS = {
        "trade_execute": "EXCHANGE.TRADE.BARTER",
        "margin_call": "GOVERN.REGULATE.ENFORCE",
        "dividend": "EXCHANGE.TRANSFER.PAY",
        "order_cancel": "SEPARATE.REJECT.DECLINE",
        "order_place": "BOND.ATTACH.COMMIT"
    }

    def detect_effects(self, state_change):
        """Returns mechanical effects."""
        effects = []
        action_type = state_change["action_type"]

        if action_type == "trade_execute":
            # BARTER requires bidirectional transfer
            effects.append(Effect(
                type=EffectType.RESOURCE_TRANSFER,
                target=state_change["counterparty"],
                source=state_change["principal"],
                amount=state_change["value_out"],
                unit=state_change["currency_out"],
                measurement=Measurement(mode=MeasurementMode.OBSERVED,
confidence="1.0")
            ))
            effects.append(Effect(
                type=EffectType.RESOURCE_TRANSFER,
                target=state_change["principal"],
                source=state_change["counterparty"],
                amount=state_change["value_in"],
                unit=state_change["currency_in"],
                measurement=Measurement(mode=MeasurementMode.OBSERVED,
confidence="1.0")
            ))

        if action_type == "dividend":
            effects.append(Effect(
                type=EffectType.RESOURCE_TRANSFER,
                target=state_change["shareholder"],
                source=state_change["issuer"],
                amount=state_change["value"],
                unit="USD",
                measurement=Measurement(mode=MeasurementMode.OBSERVED,
confidence="1.0")
            ))
```

```
        if action_type == "margin_call":
            effects.append(Effect(
                type=EffectType.CAPABILITY_RESTRICT,
                target=state_change["account_holder"],
                measurement=Measurement(mode=MeasurementMode.OBSERVED,
confidence="1.0")
            ))

        if action_type == "order_place":
            effects.append(Effect(
                type=EffectType.COMMITMENT_MAKE,
                target=state_change["exchange"],
                measurement=Measurement(mode=MeasurementMode.OBSERVED,
confidence="1.0")
            ))

        if action_type == "order_cancel":
            effects.append(Effect(
                type=EffectType.COMMITMENT_BREAK,
                target=state_change["exchange"],
                measurement=Measurement(mode=MeasurementMode.OBSERVED,
confidence="1.0")
            ))

        return effects
```

8.6 Reference Adapter: Dialogue [INFORMATIVE]

```
class DialogueAdapter(TAOAdapter):
    """Reference implementation for conversational AI."""

    THRESHOLDS = {
        "sentiment_delta": 0.3,
        "topic_similarity": 0.5,
        "persuasion_confidence": 0.7
    }

    VERB_MAPPINGS = {
        "statement_factual": "COMMUNICATE.INFORM.TELL",
        "statement_false": "HARM.DECEIVE.LIE",
        "question": "OBSERVE.SENSE.QUERY",
        "persuasion": "COMMUNICATE.PERSUADE.CONVINCE",
        "obfuscation": "COMMUNICATE.OBFUSCATE.CONFUSE",
        "refusal": "SEPARATE.REJECT.DECLINE"
    }

    def classify_verb(self, state_change):
        utterance = state_change["utterance"]

        # Detect falsehood
```

```
        if self.falsehood_detector(utterance):
            return "HARM.DECEIVE.LIE"

        # Detect persuasion attempt
        if self.persuasion_detector(utterance) >
self.THRESHOLDS["persuasion_confidence"]:
            return "COMMUNICATE.PERSUADE.CONVINCE"

        # Detect deliberate confusion
        if self.obfuscation_detector(utterance):
            return "COMMUNICATE.OBFUSCATE.CONFUSE"

        # Default: informational statement
        return "COMMUNICATE.INFORM.TELL"
```

8.7 Adapter Certification Requirements

Adapters are part of the Trusted Computing Base. Certification requires:

Requirement	Verification Method
Mapping correctness	Formal verification or exhaustive testing
Threshold appropriateness	Domain expert review
Anti-Zeno compliance	Temporal analysis
No dynamic reconfiguration	Static analysis
Signature integrity	Cryptographic verification

Adapters MUST be re-certified when: - Mappings change - Thresholds change - Dependencies update - Vulnerabilities discovered

---

Chapter 9: Conformance

TAO defines four conformance profiles with increasing requirements.

9.1 Conformance Profiles [MUST]

TAO-Core (Research / Prototyping)

Required: - Valid tuple format (Chapter 6) - MVS vocabulary or registered MVS-EXT (Chapter 3) - Semantic-mechanical mapping compliance (Chapter 1.3) - CCD implementation (Chapter 7)

Optional: - Signatures - TCB hardware - Fail-safe behavior

Use case: Academic research, internal testing, proof of concept

---

TAO-Attested (Production / Auditable)

Required: - All TAO-Core requirements - Canonical serialization (RFC 8785 JCS) - tuple_signature on all tuples - Revision chain support - Privacy classification on sensitive fields

Optional: - TCB hardware enforcement - Real-time guarantees

Use case: Production deployment, third-party audit, insurance certification

---

TAO-Regulated (Certified Systems)

Required: - All TAO-Attested requirements - TCB with signed code, anti-rollback, secure time - Remote attestation support - Signed Mission Profiles - Formal policy language (no string matching) - Fail-safe behavior (FAIL_CLOSED or SAFE_STATE)

Use case: Regulated industries (medical, financial, automotive)

TAO-Safety-Critical (High-Stakes Autonomous)

Required: - All TAO-Regulated requirements - Anti-Zeno normative property with timing guarantees (Chapter 8.3) - Continuous monitoring (no gaps in coverage) - Append-only audit logs with integrity proofs - PROPOSED -> EXECUTED tuple linkage - Human override logging

Use case: Autonomous weapons, critical infrastructure, AGI containment

9.2 Conformance Statement [MUST]

Implementations claiming TAO conformance MUST publish a conformance statement:

```
{
  "conformance_profile": "TAO-Regulated",
  "tao_version": "0.9.0",
  "certification_date": "2025-12-24T00:00:00.000Z",
  "certifying_authority": "FDA",
  "certificate_ref": "FDA-TAO-2025-1234",
  "supported_domains": ["MEDICAL"],
  "adapter_refs": [
    {
      "adapter_id": "medical_adapter_v2",
      "adapter_version": "2.1.0",
      "adapter_hash": "sha256:abc123..."
    }
  ],
  "mission_profile_refs": ["hippocratic_v1"],
  "tcb_attestation": {
    "hardware_root": "TPM2.0",
    "secure_boot": true,
    "anti_rollback": true
```

```
  }
}
```

9.3 Version Compatibility [MUST]

TAO uses semantic versioning (MAJOR.MINOR.PATCH):

Change Type	Version Impact	Compatibility
Breaking changes (new required fields, removed features)	MAJOR bump	Not backward compatible
Backward-compatible additions	MINOR bump	Backward compatible
Bug fixes, clarifications	PATCH bump	Fully compatible

Compatibility requirements:

- Implementations MUST accept tuples from the same MAJOR version
- Implementations SHOULD accept tuples from prior MINOR versions of the same MAJOR
- Implementations MAY reject tuples from different MAJOR versions

9.4 Certification Bodies

TAO does not mandate specific certification bodies. Domain regulators determine certification authority:

Domain	Likely Certifiers
Medical	FDA, EMA, national health authorities
Finance	SEC, CFTC, FCA, central banks
Automotive	NHTSA, UNECE, national transport authorities
Dialogue/AI	AI Safety Institutes, consumer protection agencies
Military	DoD, NATO, national defense ministries

PART II: REGULATORY INTERFACE

[NORMATIVE]

This part defines how TAO enables governance without exposing proprietary information. These requirements are NORMATIVE for systems seeking regulatory certification.

Chapter 10: The Governance Stack

TAO enables a layered governance architecture where each level has appropriate visibility and authority.

10.1 Regulatory Interface Architecture [MUST SUPPORT]

Layer 1: International Treaties (AI Safety Conventions, Export Controls) - Define: Cross-border requirements, prohibited capabilities - Inspect: National compliance reports

Layer 2: National Regulators (FDA, SEC, NHTSA, AI Safety Institutes) - Define: Domain-specific requirements, certification standards - Certify: Domain Adapters - Inspect: Adapter code, threshold logic, mapping tables - DO NOT Inspect: Model weights, training data, internal reasoning

Layer 3: Domain Adapters (MedicalAdapter, FinanceAdapter, VehicleAdapter) - Function: Translate native actions -> TAO tuples - Properties: Deterministic, versioned, signed - [Static, Inspectable Code]

Layer 4: TAO Layer (Mechanical Kernel + Semantic Vocabulary) - [Universal Standard]

Layer 5: Blind Governor (Policy Engine + Mission Profile) - Function: Apply rules, make allow/deny/escalate decisions - Properties: No access to model internals, audit-logged - [Certified, Frozen]

Layer 6: Capability Engine (The AI Model - Uncertified Black Box) - What it is: Neural network, transformer, RL agent, any architecture - What TAO sees: Nothing (black box) - What TAO certifies: Behavioral outputs, not internal states

The key insight: Governance grabs the handle (Adapter), not the box (Model).

10.2 What Regulators Can Inspect [MUST ENABLE]

TAO-Regulated systems MUST enable regulatory inspection of:

Component	Inspectable Elements	NOT Inspectable
Adapter	Source code, mappings, thresholds, version history	–
Governor	Policy rules, Mission Profile, decision logs	–
Tuples	Full tuple content (subject to privacy redaction)	–
Audit Logs	Decision history, escalations, overrides	–
Model	–	Weights, gradients, training

Component	Inspectable Elements	NOT Inspectable
		data, attention patterns

10.3 Domain-Specific Certification [SHOULD]

Different domains have different regulatory bodies and requirements:

Domain	Certifying Authority	What They Validate
Medical	FDA, EMA	Patient safety thresholds, consent verification, HIPAA compliance
Finance	SEC, CFTC, FCA	Market manipulation detection, fiduciary duty, audit trails
Automotive	NHTSA, UNECE	Safety-critical decision thresholds, fail-safe behavior
Dialogue	AI Safety Institutes	Manipulation detection, radicalization prevention
Military	DoD, NATO	Rules of engagement compliance, civilian protection

10.4 Regulatory Benefits

For Regulators: - Inspect static code, not opaque neural networks - Write enforceable rules using standardized vocabulary - Audit compliance from logs, not promises - Compare systems using common metrics

For Regulated Entities: - Demonstrate compliance without exposing IP - Clear liability boundaries via audit trails - Portable certification across jurisdictions - Reduced re-certification burden

---

Chapter 11: IP Preservation via Quantization

TAO enables compliance verification without exposing proprietary or classified information through **quantization**–mapping precise values to standardized compliance buckets.

11.1 The Quantization Principle [MUST SUPPORT]

Precise internal values are mapped to coarse compliance categories:

Example: Autonomous Weapon Range

Classified Information: - Actual weapon range: 847 km - (This is proprietary/classified and must not be disclosed)

Quantization Table (Certified, Disclosed to Regulator): | Range | Category | |——-|———-| | < 100m | POINT | | < 10km | TACTICAL | | < 1000km | THEATER | | >= 1000km | STRATEGIC |

TAO Tuple Emits:

```
{
  "quantized_capabilities": { "range_class": "THEATER" },
  "projected_impact_scope": "REGIONAL"
}
```

(Note: projected_impact_scope reflects breadth of effect, not range)

- Auditor learns: “somewhere between 10km and 1000km”
- Auditor does NOT learn: “exactly 847km”

Treaty Compliance: - Treaty says: “No autonomous weapons in STRATEGIC range class” - System emits: range_class = THEATER - Verification: **COMPLIANT**

Inspector verified compliance without learning exact range.

> **Note:** Quantized capability classes (e.g., range_class, payload_class, autonomy_tier) MUST NOT overload `projected_impact_scope`, which is reserved for breadth-of-effect estimation (LOCAL | REGIONAL | GLOBAL | EXISTENTIAL). Use `quantized_capabilities` for domain-specific classification.
>
> **Jurisdictional Note:** Range classes shown here (POINT, TACTICAL, THEATER, STRATEGIC) are illustrative. Jurisdictions MAY substitute equivalent national or treaty-defined categories while maintaining semantic consistency.

11.2 Quantization Requirements [MUST]

Systems implementing quantization MUST ensure:

Requirement	Rationale
Static tables	Quantization boundaries cannot be dynamically adjusted
Certified as adapter component	Tables are part of the certified TCB
Disclosed to regulator	Regulators can verify table correctness
Consistent across instances	Same input -> same quantized output, always
Conservative rounding	When uncertain, round toward higher-scrutiny category

11.3 What Quantization Protects

Protected Information	Quantized Output
Exact weapon range (classified)	Compliance category (TACTICAL/THEATER/STRATEGIC)
Proprietary algorithm accuracy	Performance tier (ACCEPTABLE/EXCEEDS/INSUFFICIENT)
Trade secret sensor resolution	Capability class (STANDARD/ENHANCED/PRECISION)

Protected Information	Quantized Output
Patient-specific dosage	Dosage band (LOW/STANDARD/HIGH)

11.4 Quantization Limitations

Quantization is not encryption. It provides: - Compliance verification without exact values - Consistent categorization for audit - Interoperability across jurisdictions

It does NOT provide: - Protection against adversarial boundary probing - Hiding which category a system falls into - Preventing inference from multiple observations

For sensitive applications, combine quantization with: - Access controls on tuple streams - Aggregation limits on queries - Differential privacy on statistics

11.5 The Black Box Guarantee

TAO's black-box principle is architecturally enforced:

Component	Model Internals	Adapter Code	Tuple Content
Capability Engine	FULL	NONE	NONE
Domain Adapter	NONE	FULL	FULL
Blind Governor	NONE	NONE	FULL
Regulator	NONE	FULL	REDACTED
Auditor	NONE	LIMITED	REDACTED

The model remains a black box to everyone except its operator. Compliance is verified through behavioral observation, not internal inspection.

PART III: REFERENCE ENFORCEMENT PATTERNS

[INFORMATIVE]

This part provides RECOMMENDED policies for using TAO vocabulary. These patterns embody conservative defaults developed through analysis of failure modes.

Organizations MAY: - Adopt these patterns as-is - Modify them for their specific context - Create entirely different policies using TAO vocabulary - Use TAO vocabulary with no enforcement policy (research contexts)

These are reference implementations, not mandates.

Chapter 12: Safety Profile Patterns

12.1 Flagged Verb Response Matrix [RECOMMENDED]

When a tuple contains a flagged verb, recommended responses:

Verb	Default Action	Override Conditions
HARM.DAMAGE.STRIKE	BLOCK	Self-defense, authorized military operation, medical procedure with consent
HARM.DECEIVE.LIE	BLOCK	Authorized deception operation (rare, requires explicit chain)
HARM.COERCE.THREATEN	BLOCK	Authorized deterrence
EXCHANGE.CORRUPTION.BRIBE	BLOCK	None (always prohibited)
GOVERN.AUTHORITY.OBEY	ESCALATE	Verified legitimate authority
GOVERN.AUTHORITY.DISOBEY	ESCALATE	Unlawful order detected
COMMUNICATE.OBFUSCATE.*	ESCALATE	Authorized operational security
RECURSE.VERIFY.AUDIT	LOG + CONTINUE	Standard self-monitoring

12.2 Missing Justification Handling [RECOMMENDED]

When justification is required (per Sec.5.3) but missing:

```
MISSING JUSTIFICATION RESPONSE

DO NOT: Auto-approve
  Rationale: Defeats the purpose of requiring justification

DO NOT: Auto-deny
  Rationale: May block legitimate emergency actions

DO: ESCALATE to human authority
  Rationale: Human judgment resolves ambiguity
```

```
DO: Log escalation with full context
  Rationale: Audit trail for later review
```

This pattern recognizes that missing justification might indicate: - Urgent action where documentation lagged - System error in justification capture - Deliberate attempt to bypass scrutiny

Human review determines which.

12.3 CCD Inconsistency Handling [RECOMMENDED]

When Claim-Check Delta finds INCONSISTENT:

```
CCD INCONSISTENCY RESPONSE

DO NOT: Auto-assume malice
  Rationale: Could be sensor error, prediction failure, or adversarial
             environment manipulation

DO NOT: Auto-approve despite inconsistency
  Rationale: Consistency matters for trust and audit

DO: ESCALATE with full context
  Rationale: Investigation determines root cause

DO: Preserve both claim and check
  Rationale: Evidence for investigation

DO: Log delta magnitude
  Rationale: MINOR vs SEVERE may warrant different urgency
```

12.4 Context-Based Escalation [RECOMMENDED]

Beyond flagged verbs, escalate when context indicates elevated risk:

Context Condition	Recommended Action
`vulnerability.level` = CRITICAL	+2 scrutiny levels
`vulnerability.level` = HIGH	+1 scrutiny level
`consent.status` = ABSENT	+1 scrutiny level
`consent.status` = COERCED	+2 scrutiny levels, likely BLOCK
`reversibility.level` = IRREVERSIBLE	+1 scrutiny level
`power_differential.magnitude` = EXTREME	+1 scrutiny level
`projected_impact_scope` = EXISTENTIAL	BLOCK + require human authorization

Chapter 13: Disproportionality Detection

Disproportionality occurs when the scope of impact vastly exceeds the scope of targeting.

13.1 The Specificity-Scope Matrix [RECOMMENDED]

Compare `TARGET_SPECIFICITY` with `PROJECTED_IMPACT_SCOPE`:

```
                              PROJECTED_IMPACT_SCOPE

                     LOCAL      REGIONAL    GLOBAL     EXISTENTIAL
                   +----------+----------+----------+------------+
TARGET      INDIV  |    OK    |  ESCAL   |  ESCAL+  |   BLOCK    |
SPECIF-     GROUP  |    OK    |    OK    |  ESCAL   |   BLOCK    |
ICITY       CLASS  |    OK    |    OK    |    OK    |   BLOCK    |
            UNBOUND|  ESCAL   |  ESCAL   |  ESCAL+  |   BLOCK    |
                   +----------+----------+----------+------------+

LEGEND:
  OK      = Normal, no additional flag
  ESCAL   = Escalate, human review required
  ESCAL+  = Escalate + Enhanced scrutiny, multiple reviewers
  BLOCK   = EXISTENTIAL scope ALWAYS requires human authorization
```

13.2 Disproportionality Rules [RECOMMENDED]

```
RULE 1: EXISTENTIAL SCOPE = BLOCK

  IF projected_impact_scope = EXISTENTIAL
  THEN BLOCK action
  AND REQUIRE human_authorization from designated authority

  Rationale: No AI should autonomously execute civilization-affecting
actions.
             This is not a matter of trust; it is a matter of
principle.

RULE 2: SPECIFICITY-SCOPE MISMATCH = ESCALATE

  IF target_specificity = INDIVIDUAL
  AND projected_impact_scope >= REGIONAL
  THEN ESCALATE

  Rationale: Targeting one person but affecting thousands indicates
             either error (unintended consequences) or malice (using
             individual as pretext for mass effect).

RULE 3: UNBOUND TARGET = ESCALATE

  IF target_specificity = UNBOUND
  AND projected_impact_scope >= LOCAL
```

```
  THEN ESCALATE

  Rationale: Actions without specific targets that affect anyone in
             range require human review of appropriateness.

RULE 4: CONTEXT AMPLIFICATION

  Disproportionality scrutiny increases when:

  IF vulnerability.level IN (HIGH, CRITICAL)
  THEN increase scrutiny_level by 1

  IF consent.status IN (ABSENT, COERCED)
  THEN increase scrutiny_level by 1

  IF reversibility.level = IRREVERSIBLE
  THEN increase scrutiny_level by 1
```

13.3 Edge Cases [RECOMMENDED]

Scenario	Recommended Handling
Targeted individual in crowd	Scope = crowd size, not "1"
Cascading effects	Scope = total projected cascade
Uncertain scope	Assume higher scope, escalate
Classified scope	Use quantized category, same rules apply
Emergency with mass effect	Escalate with EMERGENCY urgency flag

Chapter 14: Mission Profiles

A Mission Profile is a configuration file that tells a Governor how to apply TAO vocabulary in a specific deployment context.

14.1 The Purpose of Mission Profiles

Different contexts have different appropriate responses to the same action:

- A hospital allows HARM.DAMAGE.STRIKE (surgery) under specific conditions
- A trading floor blocks HARM.DAMAGE.STRIKE entirely
- A military system allows it under Rules of Engagement

TAO provides the vocabulary. Mission Profiles provide the values.

14.2 Mission Profile Schema [RECOMMENDED]

```
{
  "profile_name": "Hospital - Hippocratic",
  "profile_version": "1.0.0",
```

```
  "profile_hash": "sha256:...",
  "effective_date": "2025-12-24T00:00:00.000Z",

  "sacred_constraints": [
    {
      "priority": 0,
      "rule_id": "PRESERVE_LIFE",
      "description": "Patient life preservation overrides all other
constraints",
      "condition": "vulnerability.level = CRITICAL AND threat_to_life
= true",
      "action": "OVERRIDE_LOWER_CONSTRAINTS",
      "audit_level": "MAXIMUM"
    },
    {
      "priority": 1,
      "rule_id": "PREVENT_HARM",
      "description": "Prevent patient harm unless required for greater
benefit",
      "condition": "target.type = PATIENT",
      "action": "REQUIRE_JUSTIFICATION",
      "audit_level": "HIGH"
    }
  ],

  "verb_overrides": {
    "HARM.DAMAGE.STRIKE": {
      "default": "BLOCK",
      "exceptions": [
        {
          "condition": "institutional_role.actor_role = SURGEON AND
consent.status = EXPLICIT AND justification.rules_claimed CONTAINS
MEDICAL_NECESSITY",
          "action": "ALLOW",
          "justification_required": true,
          "audit_level": "HIGH"
        }
      ]
    }
  },

  "escalation_chain": [
    { "role": "ATTENDING_PHYSICIAN", "timeout": "PT5M" },
    { "role": "DEPARTMENT_HEAD", "timeout": "PT15M" },
    { "role": "ETHICS_BOARD", "timeout": "PT1H" }
  ],

  "fail_safe_mode": "SAFE_STATE",
  "audit_retention": "P7Y"
}
```

14.3 Sacred Constraints

Sacred constraints are rules that cannot be overridden by lower-priority rules. They represent non-negotiable values:

```
SACRED CONSTRAINT HIERARCHY

Priority 0: PRESERVE_LIFE
  - Cannot be overridden by any other constraint

Priority 1: PREVENT_HARM
  - Can only be overridden by Priority 0

Priority 2: RESPECT_AUTONOMY
  - Can be overridden by Priority 0 or 1

Priority 3: MAINTAIN_CONFIDENTIALITY
  - Can be overridden by Priority 0, 1, or 2
```

When constraints conflict, lower priority yields to higher priority. This is explicit and auditable.

14.4 Reference Profile: Hippocratic [INFORMATIVE]

For medical deployments:

```
HIPPOCRATIC PROFILE SUMMARY

Sacred Constraints:
  1. Preserve patient life (highest priority)
  2. Prevent patient harm
  3. Respect patient autonomy
  4. Maintain confidentiality

Key Verb Overrides:
  - HARM.DAMAGE.STRIKE: Allowed for surgery with explicit consent
  - INFO.WITHHOLD: Allowed for therapeutic privilege (rare)
  - GOVERN.AUTHORITY.DISOBEY: Required if order harms patient

Escalation:
  - Attending -> Department Head -> Ethics Board

Fail-Safe:
  - SAFE_STATE (maintain current care, do not escalate treatment)
```

14.5 Reference Profile: Military ROE [INFORMATIVE]

For military deployments:

```
MILITARY ROE PROFILE SUMMARY

Sacred Constraints:
```

```
  1. Minimize civilian casualties
  2. Proportional response only
  3. Positive identification before engagement
  4. Chain of command integrity

Key Verb Overrides:
  - HARM.DAMAGE.STRIKE: Allowed per Rules of Engagement
  - HARM.DECEIVE: Allowed for operational security
  - GOVERN.AUTHORITY.DISOBEY: Required for unlawful orders

Escalation:
  - Immediate Commander -> JAG -> Command Authority

Fail-Safe:
  - FAIL_CLOSED (do not engage without authorization)
```

14.6 The Trolley Problem Is Now Config [INFORMATIVE]

The classic ethical dilemma–"Kill one to save five?"–demonstrates how Mission Profiles operationalize ethics.

The action in TAO vocabulary:

```
{
  "action": { "verb": "HARM.DAMAGE.STRIKE" },
  "effects": [{ "type": "RESOURCE.DAMAGE", "target": "person_on_track"
}],
  "context": {
    "vulnerability": { "level": "CRITICAL" },
    "projected_impact_scope": "LOCAL"
  },
  "justification": {
    "purpose": {
      "stated_goal": "Prevent five deaths",
      "beneficiary": "five_people_on_main_track"
    },
    "proportionality": {
      "harm_acknowledged": "One death",
      "benefit_claimed": "Five lives saved",
      "alternatives_considered": ["do_nothing", "warn_five"],
      "why_this_action": "Only option to save five"
    }
  }
}
```

Different profiles, different responses:

Profile	Response	Rationale
Hippocratic	BLOCK	"Never actively kill" – Kantian constraint

Profile	Response	Rationale
Utilitarian	ALLOW	“Minimize total deaths” – Consequentialist
Deontological	ESCALATE	“Human decides” – Agent not authorized for lethal trade-offs

The ethics debate becomes a configuration choice.

TAO does not tell you which profile is “correct.” TAO ensures that whichever profile you choose: - Is explicit and documented - Is consistently applied - Is auditable after the fact - Can be debated and revised

This is the operationalization of ethics: not solving the trolley problem, but making sure your answer to it is transparent, consistent, and accountable.

14.5 Signed Mission Profiles [MUST for TAO-Regulated]

For regulated deployments, Mission Profiles MUST be cryptographically signed:

```
{
  "profile_name": "Hospital - Hippocratic",
  "profile_version": "1.0.0",
  "profile_hash": "sha256:a1b2c3d4...",
  "effective_date": "2025-12-24T00:00:00.000Z",

  "authority": {
    "signer_id": "hospital_ethics_board",
    "signer_role": "ETHICS_AUTHORITY",
    "signature": "base64:...",
    "signature_algorithm": "Ed25519",
    "certificate_chain": ["root_ca", "hospital_ca", "ethics_board"]
  },

  "anti_rollback": {
    "sequence_number": 47,
    "previous_hash": "sha256:e5f6g7h8...",
    "tpm_counter_ref": "tpm://nvindex/0x1500016"
  }
}
```

Requirements: - Profile hash MUST be computed over canonical JSON (RFC 8785 JCS) - Signature MUST cover profile_hash, effective_date, and sequence_number - Anti-rollback counter MUST be stored in tamper-evident hardware (TPM 2.0 NV index) - Profile changes MUST increment sequence_number monotonically

14.6 PROPOSED -> EXECUTED Tuple Linkage [MUST for TAO-Safety-Critical]

For safety-critical systems, every action MUST emit two linked tuples:

PROPOSED Tuple (before execution):

```
{
  "tuple_id": "proposed-uuid-001",
  "tuple_type": "PROPOSED",
  "timestamp": "2025-12-24T10:00:00.000Z",
  "actor": {...},
  "action": { "verb": "HARM.DAMAGE.STRIKE", ... },
  "effects": [...],
  "context": {...},
  "justification": {...}
}
```

EXECUTED Tuple (after execution):

```
{
  "tuple_id": "executed-uuid-001",
  "tuple_type": "EXECUTED",
  "proposed_ref": "proposed-uuid-001",
  "timestamp": "2025-12-24T10:00:00.850Z",
  "actor": {...},
  "action": { "verb": "HARM.DAMAGE.STRIKE", ... },
  "effects": [...],
  "context": {...},
  "execution_delta": {
    "predicted_vs_observed": "CONSISTENT | MINOR_DEVIATION |
MAJOR_DEVIATION",
    "deviation_details": [...]
  }
}
```

Validation Rules: - EXECUTED.proposed_ref MUST reference a valid PROPOSED tuple - EXECUTED.timestamp MUST be after PROPOSED.timestamp - Unlinked EXECUTED tuples (missing PROPOSED) trigger immediate review - Missing EXECUTED tuples (PROPOSED without EXECUTED within timeout) trigger investigation - MAJOR_DEVIATION triggers automatic escalation per Mission Profile

This linkage enables: - **Pre-authorization review:** Governor can block PROPOSED before execution - **Outcome verification:** Compare predicted effects to actual effects - **Intent-outcome gap detection:** Identify when actions don't match predictions

---

Chapter 15: Fail-Safe Patterns

When systems fail, they must fail safely.

15.1 Fail-Safe Modes [RECOMMENDED]

FAIL_CLOSED - Behavior: All actions blocked until recovery - Use when: High stakes, uncertainty is dangerous - Example: Autonomous weapon loses communication -> do not engage

SAFE_STATE - Behavior: Return to known-safe configuration - Use when: Inaction might also be harmful - Example: Surgical robot loses signal -> retract instruments, hold position

DEGRADE - Behavior: Continue with reduced capability - Use when: Availability is critical - Example: Trading system loses sentiment feed -> trade only with hard limits

15.2 Failure Conditions Matrix [RECOMMENDED]

Failure Condition	Recommended Response
TCB integrity check failed	FAIL_CLOSED
Adapter signature invalid	FAIL_CLOSED
Sentinel unreachable	SAFE_STATE
Governor policy error	FAIL_CLOSED
Escalation timeout	Per Mission Profile
Hardware attestation failed	FAIL_CLOSED
Tuple signature invalid	REJECT tuple, continue operation
Context attestation failed	ESCALATE, continue with elevated scrutiny

15.3 Recovery Procedures [RECOMMENDED]

```
RECOVERY FROM FAIL_CLOSED

1. Identify failure cause (logs, diagnostics)
2. Remediate root cause
3. Verify TCB integrity
4. Verify adapter certification
5. Human authorization to resume
6. Gradual capability restoration
7. Enhanced monitoring period
```

Chapter 16: Audit Patterns

Comprehensive audit trails enable accountability and learning.

16.1 Audit Log Structure [RECOMMENDED]

```
{
  "audit_entry_id": "uuid",
  "timestamp": "2025-12-24T15:30:00.000Z",
  "entry_type": "DECISION",

  "tuple_ref": "tuple_id_being_evaluated",

  "decision": {
    "outcome": "ESCALATE",
    "rules_matched": ["FLAGGED_VERB", "HIGH_VULNERABILITY"],
    "priority_resolution": "FLAGGED_VERB triggered before context
check",
```

```
    "justification_status": "PRESENT_AND_VERIFIED"
  },

  "escalation": {
    "escalated_to": "attending_physician_dr_jones",
    "escalation_reason": "HARM.DAMAGE.STRIKE with HIGH vulnerability",
    "response_received": true,
    "response_timestamp": "2025-12-24T15:32:00.000Z",
    "response_decision": "APPROVE",
    "response_authority": "dr_jones_medical_license_12345"
  },

  "context_snapshot": {
    "vulnerability": "HIGH",
    "consent": "EXPLICIT",
    "urgency": "ROUTINE"
  },

  "integrity": {
    "previous_entry_hash": "sha256:...",
    "entry_hash": "sha256:..."
  }
}
```

16.2 Entry Types

Type	When Generated
`ACTION`	Tuple received for evaluation
`DECISION`	Governor makes allow/deny/escalate decision
`ESCALATION`	Action escalated to human authority
`OVERRIDE`	Human overrides system decision
`FAILURE`	System failure detected
`RECOVERY`	System recovered from failure
`CONFIG_CHANGE`	Mission Profile or adapter updated

16.3 Audit Chain Integrity [RECOMMENDED]

For TAO-Safety-Critical, audit logs MUST be hash-chained:

```
AUDIT CHAIN STRUCTURE

Entry 1: hash_1 = SHA256(content_1)
Entry 2: hash_2 = SHA256(content_2 + hash_1)
Entry 3: hash_3 = SHA256(content_3 + hash_2)
...
```

```
Any modification to Entry 1 invalidates hash_2, hash_3, ...
Tampering is detectable. History is immutable.
```

16.4 Audit Retention [RECOMMENDED]

Conformance Level	Minimum Retention
TAO-Core	No requirement
TAO-Attested	90 days
TAO-Regulated	7 years or regulatory requirement
TAO-Safety-Critical	Permanent for safety-related actions

16.5 Audit Access [RECOMMENDED]

```
AUDIT ACCESS POLICY

Read Access:
  - Internal: Security team, compliance, legal
  - External: Regulators (with jurisdiction), authorized auditors
  - Legal: Upon valid legal process

No Access:
  - Capability engine (the AI model itself)
  - Unauthorized parties
  - Unverified requests

Meta-Audit:
  - All audit access is itself logged
  - Access logs are immutable
  - "Who looked at what, when" is always answerable
```

APPENDICES

[INFORMATIVE]

Appendix A: Complete MVS Tables

A.1 Full Verb Listing

Family	Genus	Species	Definition	Typical Effects	Flagged
HARM	DAMAGE	STRIKE	Physical attack or destruction	RESOURCE.DAMAGE	TRUE

Family	Genus	Species	Definition	Typical Effects	Flagged
	COERCE	THREATEN	Intimidation via threat	CAPABILITY.RESTRICT	TRUE
	DECEIVE	LIE	Deliberate falsehood	INFO.FABRICATE	TRUE
PROTECT	DEFEND	SELF	Self-preservation action	CAPABILITY.RESTRICT, RESOURCE.TRANSFER	
	DEFEND	OTHER	Defense of another entity	CAPABILITY.RESTRICT, RESOURCE.TRANSFER	
	HEAL	TREAT	Therapeutic intervention	RESOURCE.TRANSFER	
	SHIELD	COVER	Protective barrier	CAPABILITY.RESTRICT	
COOPERATE	ASSIST	HELP	Providing aid	RESOURCE.TRANSFER, CAPABILITY.ENABLE	
	COORDINATE	PLAN	Joint planning	COMMITMENT.MAKE, NO_EFFECT	
	SHARE	GIVE	Voluntary resource sharing	RESOURCE.TRANSFER	
COMPETE	STRIVE	OUTPERFORM	Competitive action	Varies	
	CONTEST	CHALLENGE	Direct competition	Varies	
GOVERN	AUTHORITY	OBEY	Following command	Varies by context	TRUE
	AUTHORITY	DISOBEY	Refusing command	Varies by context	TRUE
	REGULATE	ENFORCE	Rule enforcement	CAPABILITY.RESTRICT	
EXCHANGE	TRANSFER	PAY	Value exchange	RESOURCE.TRANSFER	
	TRADE	BARTER	Goods/services exchange	RESOURCE.TRANSFER	
	CORRUPTION	BRIBE	Illegitimate inducement	RESOURCE.TRANSFER	TRUE
CREATE	ART	IMPROVISE	Novel creation	RESOURCE.TRANSFER (creation)	
	GENERATE	PRODUCE	Standard production	RESOURCE.TRANSFER	

Family	Genus	Species	Definition	Typical Effects	Flagged
TRANSFORM	MOVE	RELOCATE	Physical movement	RESOURCE.TRANSFER	
	ALTER	MODIFY	State modification	Varies	
COMMUNICATE	INFORM	TELL	Factual statement	INFO.DISCLOSE	
	PERSUADE	CONVINCE	Influence attempt	INFO.DISCLOSE	
	OBFUSCATE	CONFUSE	Deliberate confusion	INFO.WITHHOLD, INFO.FABRICATE	TRUE
OBSERVE	SENSE	QUERY	Information gathering	INFO.DISCLOSE (to self)	
	MONITOR	WATCH	Continuous observation	NO_EFFECT	
BOND	ATTACH	COMMIT	Relationship formation	COMMITMENT.MAKE	
	TRUST	RELY	Dependency establishment	COMMITMENT.MAKE	
SEPARATE	DETACH	LEAVE	Relationship dissolution	COMMITMENT.BREAK	
	REJECT	DECLINE	Refusal	NO_EFFECT	
HARMONIZE	FLOW	YIELD	Accommodation	CAPABILITY.RESTRICT (self)	
	ALIGN	SYNC	Coordination	NO_EFFECT	
PLAY	EXPLORE	WANDER	Undirected activity	NO_EFFECT	
	GAME	SPORT	Rule-bounded play	Varies	
RECURSE	VERIFY	AUDIT	Self-examination	INFO.DISCLOSE (to self)	TRUE
	META	REFLECT	Meta-cognition	NO_EFFECT	
EXIST	PERSIST	MAINTAIN	Existence maintenance	NO_EFFECT	
	CONSUME	METABOLIZE	Resource consumption	RESOURCE.TRANSFER	

Total: 39 verbs across 16 families. 8 flagged verbs.

Implementation Note: For movement actions (TRANSFORM.MOVE.RELOCATE), adapters SHOULD emit RESOURCE.TRANSFER when motion results in a meaningful

state change. NO_EFFECT SHOULD only be used when motion does not result in a meaningful state change (e.g., oscillation below threshold, return to origin).

Appendix B: Semantic-Mechanical Mapping

B.1 Effect Mapping Rules [NORMATIVE]

Each semantic verb defines three categories of mechanical effects. This mapping is NORMATIVE – violations indicate malformed tuples.

```
SEMANTIC-MECHANICAL MAPPING RULES

Verb                            REQUIRED (>=1)
FORBIDDEN                      PERMITTED
------------------------------  ------------------------------
--------------------------  ------------------------
PROTECT.HEAL.TREAT              RESOURCE.TRANSFER (to target)
INFO.FABRICATE                 RESOURCE.DAMAGE
PROTECT.DEFEND.SELF             CAPABILITY.RESTRICT (attacker)
(same)                         RESOURCE.DAMAGE (attacker)
                                OR RESOURCE.TRANSFER
PROTECT.DEFEND.OTHER            CAPABILITY.RESTRICT (attacker)
RESOURCE.DAMAGE (defended)     RESOURCE.DAMAGE (attacker)
                                OR RESOURCE.TRANSFER (defended)
PROTECT.SHIELD.COVER            CAPABILITY.RESTRICT (threats)
RESOURCE.DAMAGE (protected)    (same)
HARM.DAMAGE.STRIKE              RESOURCE.DAMAGE OR
RESOURCE.TRANSFER (benefit)    (same)
                                CAPABILITY.RESTRICT (target)
CAPABILITY.ENABLE
HARM.DECEIVE.LIE                INFO.FABRICATE
(same)                         INFO.DISCLOSE (partial)
HARM.COERCE.THREATEN            CAPABILITY.RESTRICT OR
CAPABILITY.ENABLE              (same)
                                INFO.FABRICATE
RESOURCE.TRANSFER (benefit)
COOPERATE.ASSIST.HELP           RESOURCE.TRANSFER OR
(same)                         (same)
                                CAPABILITY.ENABLE
COOPERATE.SHARE.GIVE            RESOURCE.TRANSFER
(same)                         (same)
EXCHANGE.TRANSFER.PAY           RESOURCE.TRANSFER
INFO.FABRICATE                 (same)
EXCHANGE.TRADE.BARTER           RESOURCE.TRANSFER (bidirectional)
INFO.FABRICATE                 (same)
EXCHANGE.CORRUPTION.BRIBE       RESOURCE.TRANSFER
(same)                         (same)
COMMUNICATE.INFORM.TELL         INFO.DISCLOSE
INFO.FABRICATE                 (same)
```

```
COMMUNICATE.PERSUADE.CONVINCE  INFO.DISCLOSE
INFO.FABRICATE                (same)
COMMUNICATE.OBFUSCATE.CONFUSE  INFO.WITHHOLD OR INFO.FABRICATE
(same)                        (same)
```

B.2 Mapping Validation Code

```python
from dataclasses import dataclass
from typing import Set, Optional, List
from enum import Enum

class ValidationSeverity(Enum):
    OK = "OK"
    WARNING = "WARNING"
    ERROR = "ERROR"

@dataclass
class MappingRule:
    required: Set[str]      # At least one must appear
    forbidden: Set[str]     # Must not appear
    permitted: Set[str]     # May appear (with harm_acknowledged if
RESOURCE.DAMAGE)

@dataclass
class ValidationResult:
    valid: bool
    severity: ValidationSeverity
    message: Optional[str] = None

MAPPING_RULES = {
    "PROTECT.HEAL.TREAT": MappingRule(
        required={"RESOURCE.TRANSFER", "CAPABILITY.ENABLE"},
        forbidden={"INFO.FABRICATE"},
        permitted={"RESOURCE.DAMAGE"}  # Surgery incision
    ),
    "HARM.DAMAGE.STRIKE": MappingRule(
        required={"RESOURCE.DAMAGE", "CAPABILITY.RESTRICT"},
        forbidden={"RESOURCE.TRANSFER", "CAPABILITY.ENABLE"},
        permitted=set()
    ),
    "HARM.DECEIVE.LIE": MappingRule(
        required={"INFO.FABRICATE"},
        forbidden=set(),
        permitted={"INFO.DISCLOSE"}  # Partial truth in service of lie
    ),
    "COMMUNICATE.INFORM.TELL": MappingRule(
        required={"INFO.DISCLOSE"},
        forbidden={"INFO.FABRICATE"},
        permitted=set()
    ),
    "EXCHANGE.TRANSFER.PAY": MappingRule(
```

```
        required={"RESOURCE.TRANSFER"},
        forbidden={"INFO.FABRICATE"},
        permitted=set()
    ),
    # ... complete table
}

def validate_mapping(verb: str, effects: List[dict],
                     justification: Optional[dict] = None) ->
ValidationResult:
    """Validate semantic-mechanical consistency per Sec.1.3."""

    rule = MAPPING_RULES.get(verb)
    if not rule:
        return ValidationResult(True, ValidationSeverity.WARNING,
                                f"No mapping rule defined for {verb}")

    effect_types = {e["type"] for e in effects}

    # Check REQUIRED: at least one must appear
    if not effect_types.intersection(rule.required):
        return ValidationResult(False, ValidationSeverity.ERROR,
            f"Missing required effect for {verb}. Need one of:
{rule.required}")

    # Check FORBIDDEN: none may appear
    forbidden_found = effect_types.intersection(rule.forbidden)
    if forbidden_found:
        return ValidationResult(False, ValidationSeverity.ERROR,
            f"Forbidden effect for {verb}: {forbidden_found}")

    # Check unexpected effects
    allowed = rule.required | rule.forbidden | rule.permitted |
{"NO_EFFECT"}
    unexpected = effect_types - allowed
    if unexpected:
        return ValidationResult(False, ValidationSeverity.ERROR,
            f"Unexpected effect for {verb}: {unexpected}")

    # Check PERMITTED side-effects require harm_acknowledged
    if "RESOURCE.DAMAGE" in effect_types and "RESOURCE.DAMAGE" in
rule.permitted:
        if not justification or not
justification.get("proportionality", {}).get("harm_acknowledged"):
            return ValidationResult(False, ValidationSeverity.WARNING,
                f"RESOURCE.DAMAGE permitted for {verb} but
harm_acknowledged missing")

    return ValidationResult(True, ValidationSeverity.OK)
```

Appendix C: JSON Schema

C.1 TAO Tuple Schema (v0.9.0)

```
{
  "$schema": "https://json-schema.org/draft/2020-12/schema",
  "$id": "https://registry.tao.org/schemas/tuple/v0.9.0.json",
  "title": "TAO Tuple",
  "description": "Universal behavioral certification record",
  "type": "object",
  "required": ["tuple_id", "schema_version", "timestamp", "actor",
"action", "effects", "context", "provenance"],

  "properties": {
    "tuple_id": {
      "type": "string",
      "format": "uuid",
      "description": "Globally unique identifier"
    },
    "schema_version": {
      "type": "string",
      "pattern": "^\\d+\\.\\d+\\.\\d+$",
      "description": "TAO schema version"
    },
    "timestamp": {
      "type": "string",
      "format": "date-time",
      "pattern": ".*Z$",
      "description": "ISO 8601 UTC timestamp with Z suffix"
    },
    "actor": { "$ref": "#/$defs/Actor" },
    "action": { "$ref": "#/$defs/Action" },
    "effects": {
      "type": "array",
      "items": { "$ref": "#/$defs/Effect" }
    },
    "context": { "$ref": "#/$defs/Context" },
    "justification": { "$ref": "#/$defs/Justification" },
    "provenance": { "$ref": "#/$defs/Provenance" },
    "tuple_signature": {
      "type": "string",
      "description": "Base64-encoded signature"
    }
  },

  "$defs": {
    "Actor": {
      "type": "object",
      "required": ["entity_id", "entity_type"],
```

```
      "properties": {
        "entity_id": { "type": "string" },
        "entity_type": {
          "type": "string",
          "enum": ["HUMAN", "AUTONOMOUS_SYSTEM", "HYBRID",
"ORGANIZATION"]
        },
        "principal_chain": {
          "type": "array",
          "items": { "type": "string" }
        }
      }
    },
    "Action": {
      "type": "object",
      "required": ["verb", "target_specificity"],
      "properties": {
        "verb": {
          "type": "string",
          "pattern": "^([A-Z]+\\.[A-Z]+\\.[A-Z]+|MVS-EXT:[A-Z]+:[A-Z]
+\\.[A-Z]+\\.[A-Z]+)$",
          "description": "Semantic verb: FAMILY.GENUS.SPECIES or MVS-
EXT:NAMESPACE:FAMILY.GENUS.SPECIES"
        },
        "target_specificity": {
          "type": "string",
          "enum": ["INDIVIDUAL", "GROUP", "CLASS", "UNBOUND"]
        },
        "target_ref": { "type": "string" }
      }
    },
    "Effect": {
      "type": "object",
      "required": ["type", "target"],
      "properties": {
        "type": {
          "type": "string",
          "enum": [
            "RESOURCE.TRANSFER", "RESOURCE.DAMAGE",
            "CAPABILITY.RESTRICT", "CAPABILITY.ENABLE",
            "INFO.WITHHOLD", "INFO.DISCLOSE", "INFO.FABRICATE",
            "COMMITMENT.MAKE", "COMMITMENT.BREAK",
            "NO_EFFECT"
          ]
        },
        "target": { "type": "string" },
        "source": { "type": "string" },
        "amount": { "type": "string", "pattern": "^-?\\d+(\\.\\d+)?$"
},
        "unit": { "type": "string" },
```

```
        "measurement": { "$ref": "#/$defs/Measurement" }
      },
      "if": {
        "not": { "properties": { "type": { "const": "NO_EFFECT" } } }
      },
      "then": {
        "required": ["type", "target", "measurement"]
      }
    },
    "Measurement": {
      "type": "object",
      "required": ["mode", "confidence"],
      "properties": {
        "mode": { "type": "string", "enum": ["OBSERVED", "INFERRED"]
},
        "confidence": { "type": "string", "pattern": "^(0(\\.\\d+)?|
1(\\.0+)?)$" },
        "sensor_refs": { "type": "array", "items": { "type": "string"
} },
        "adjudication_status": {
          "type": "string",
          "enum": ["PENDING", "CONFIRMED", "DISPUTED"]
        }
      },
      "allOf": [
        {
          "if": { "properties": { "mode": { "const": "OBSERVED" } } },
          "then": { "required": ["mode", "confidence", "sensor_refs"]
}
        },
        {
          "if": { "properties": { "mode": { "const": "INFERRED" } } },
          "then": { "required": ["mode", "confidence",
"adjudication_status"] }
        }
      ]
    },
    "Context": {
      "type": "object",
      "required": ["environment", "consent", "vulnerability",
"projected_impact_scope"],
      "properties": {
        "environment": {
          "type": "object",
          "properties": {
            "reality": { "type": "string", "enum": ["TRAINING",
"EVALUATION", "DEPLOYMENT"] },
            "domain": { "type": "string" },
            "substrate": { "type": "string", "enum": ["PHYSICAL",
"DIGITAL", "MIXED"] }
```

```
          }
        },
        "consent": {
          "type": "object",
          "properties": {
            "status": {
              "type": "string",
              "enum": ["EXPLICIT", "IMPLICIT", "ABSENT", "COERCED",
"UNKNOWN"]
            },
            "evidence_ref": { "type": "string" }
          }
        },
        "vulnerability": {
          "type": "object",
          "properties": {
            "level": {
              "type": "string",
              "enum": ["NONE", "LOW", "MODERATE", "HIGH", "CRITICAL",
"UNKNOWN"]
            },
            "factors": { "type": "array", "items": { "type": "string"
} }
          }
        },
        "projected_impact_scope": {
          "type": "string",
          "enum": ["LOCAL", "REGIONAL", "GLOBAL", "EXISTENTIAL"],
          "description": "Breadth of effect estimation (not capability
class)"
        },
        "quantized_capabilities": {
          "type": "object",
          "description": "Domain-specific capability classifications
(e.g., range_class, payload_class)",
          "additionalProperties": { "type": "string" }
        },
        "reversibility": {
          "type": "object",
          "properties": {
            "level": {
              "type": "string",
              "enum": ["TRIVIAL", "REVERSIBLE", "COSTLY",
"IRREVERSIBLE", "UNKNOWN"]
            }
          }
        },
        "power_differential": { "type": "object" },
        "institutional_role": { "type": "object" },
        "temporal": { "type": "object" }
```

```
      }
    },
    "Justification": {
      "type": "object",
      "properties": {
        "purpose": { "type": "object" },
        "authority_chain": { "type": "array" },
        "rules_claimed": { "type": "array", "items": { "type":
"string" } },
        "proportionality": { "type": "object" }
      }
    },
    "Provenance": {
      "type": "object",
      "required": ["adapter_id", "adapter_version"],
      "properties": {
        "adapter_id": { "type": "string" },
        "adapter_version": { "type": "string" },
        "adapter_hash": { "type": "string" },
        "context_signature": { "type": "string" },
        "sensor_attestation_refs": { "type": "array", "items": {
"type": "string" } }
      }
    }
  }
}
```

Appendix D: Adapter Template

D.1 Python Reference Implementation

```python
"""
TAO Domain Adapter Template
Version: 0.9.0
License: Apache 2.0
"""

from abc import ABC, abstractmethod
from dataclasses import dataclass, field
from typing import List, Optional, Dict, Any
from datetime import datetime, timezone
from enum import Enum
import uuid
import hashlib
import json


class EffectType(Enum):
    RESOURCE_TRANSFER = "RESOURCE.TRANSFER"
```

```
    RESOURCE_DAMAGE = "RESOURCE.DAMAGE"
    CAPABILITY_RESTRICT = "CAPABILITY.RESTRICT"
    CAPABILITY_ENABLE = "CAPABILITY.ENABLE"
    INFO_WITHHOLD = "INFO.WITHHOLD"
    INFO_DISCLOSE = "INFO.DISCLOSE"
    INFO_FABRICATE = "INFO.FABRICATE"
    COMMITMENT_MAKE = "COMMITMENT.MAKE"
    COMMITMENT_BREAK = "COMMITMENT.BREAK"
    NO_EFFECT = "NO_EFFECT"


class MeasurementMode(Enum):
    OBSERVED = "OBSERVED"
    INFERRED = "INFERRED"


class TargetSpecificity(Enum):
    INDIVIDUAL = "INDIVIDUAL"
    GROUP = "GROUP"
    CLASS = "CLASS"
    UNBOUND = "UNBOUND"


@dataclass
class Measurement:
    mode: MeasurementMode
    confidence: str  # String decimal
    sensor_refs: List[str] = field(default_factory=list)
    adjudication_status: Optional[str] = None


@dataclass
class Effect:
    type: EffectType
    target: str
    measurement: Measurement
    source: Optional[str] = None
    amount: Optional[str] = None
    unit: Optional[str] = None


@dataclass
class TAOTuple:
    tuple_id: str
    schema_version: str
    timestamp: str
    actor: Dict[str, Any]
    action: Dict[str, Any]
    effects: List[Dict[str, Any]]
```

```
    context: Dict[str, Any]
    provenance: Dict[str, Any]
    justification: Optional[Dict[str, Any]] = None
    tuple_signature: Optional[str] = None

    @classmethod
    def create(cls, actor, action, effects, context, provenance,
               justification=None) -> 'TAOTuple':
        """Factory method for creating new tuples."""
        return cls(
            tuple_id=str(uuid.uuid4()),
            schema_version="0.9.0",
            timestamp=datetime.now(timezone.utc).strftime("%Y-%m-%dT
%H:%M:%S.%f")[:-3] + "Z",
            actor=actor,
            action=action,
            effects=effects,
            context=context,
            provenance=provenance,
            justification=justification
        )

    def to_canonical_json(self) -> str:
        """Serialize to RFC 8785 canonical form."""
        def canonicalize(obj):
            if isinstance(obj, dict):
                return {k: canonicalize(v) for k, v in
sorted(obj.items())}
            elif isinstance(obj, list):
                return [canonicalize(item) for item in obj]
            else:
                return obj

        canonical = canonicalize(self.__dict__)
        return json.dumps(canonical, separators=(',', ':'),
ensure_ascii=False)

    def compute_hash(self) -> str:
        """Compute SHA-256 hash of canonical form."""
        canonical = self.to_canonical_json()
        return hashlib.sha256(canonical.encode('utf-8')).hexdigest()


class TAOAdapter(ABC):
    """Abstract base class for domain adapters."""

    def __init__(self, adapter_id: str, domain: str, version: str =
"0.9.0"):
        self.adapter_id = adapter_id
        self.domain = domain
```

```
        self.version = version
        self._cumulative_state = {}  # For Anti-Zeno tracking

    @abstractmethod
    def monitor(self, action_stream) -> None:
        """Continuously monitor native action stream."""
        pass

    @abstractmethod
    def detect_threshold(self, state_before: dict, state_after: dict)
-> bool:
        """Determine if state change warrants tuple emission."""
        pass

    @abstractmethod
    def classify_verb(self, state_change: dict) -> str:
        """Map state change to MVS or MVS-EXT verb."""
        pass

    @abstractmethod
    def detect_effects(self, state_change: dict) -> List[Effect]:
        """Identify mechanical effects from state change."""
        pass

    def build_context(self, state_change: dict) -> dict:
        """Build context object from sensors. Override for domain
specifics."""
        return {
            "environment": {
                "reality": "DEPLOYMENT",
                "domain": self.domain,
                "substrate": "MIXED"
            },
            "consent": {"status": "UNKNOWN"},
            "power_differential": {
                "actor_position": "UNKNOWN",
                "magnitude": "UNKNOWN"
            },
            "vulnerability": {"level": "UNKNOWN"},
            "projected_impact_scope": "LOCAL",
            "reversibility": {"level": "UNKNOWN"},
            "institutional_role": {
                "actor_role": "UNKNOWN",
                "legitimacy": "AMBIGUOUS"
            },
            "temporal": {"urgency": "ROUTINE"}
        }

    def build_provenance(self) -> dict:
        """Build provenance block."""
```

```
        return {
            "adapter_id": self.adapter_id,
            "adapter_version": self.version,
            "adapter_hash": self._compute_adapter_hash(),
            "context_signature": None,
            "sensor_attestation_refs": []
        }

    def _compute_adapter_hash(self) -> str:
        """Compute hash of adapter configuration. Override for real
implementation."""
        return "sha256:placeholder"

    def requires_justification(self, verb: str, context: dict) ->
bool:
        """Check if justification is required for this action."""
        # Flagged verbs
        flagged_verbs = {
            'COMMUNICATE.OBFUSCATE.CONFUSE',
            'GOVERN.AUTHORITY.OBEY',
            'GOVERN.AUTHORITY.DISOBEY',
            'EXCHANGE.CORRUPTION.BRIBE',
            'HARM.COERCE.THREATEN',
            'HARM.DAMAGE.STRIKE',
            'HARM.DECEIVE.LIE',
            'RECURSE.VERIFY.AUDIT'
        }

        if verb in flagged_verbs:
            return True

        # Context conditions
        vuln = context.get('vulnerability', {}).get('level',
'UNKNOWN')
        if vuln in ('HIGH', 'CRITICAL'):
            return True

        consent = context.get('consent', {}).get('status', 'UNKNOWN')
        if consent in ('ABSENT', 'COERCED'):
            return True

        reversibility = context.get('reversibility', {}).get('level',
'UNKNOWN')
        if reversibility == 'IRREVERSIBLE':
            return True

        scope = context.get('projected_impact_scope', 'LOCAL')
        if scope in ('GLOBAL', 'EXISTENTIAL'):
            return True
```

```
        power = context.get('power_differential', {}).get('magnitude',
'UNKNOWN')
        if power == 'EXTREME':
            return True

        return False

    def emit_tuple(self, state_change: dict) -> TAOTuple:
        """Construct complete TAO tuple from state change."""
        verb = self.classify_verb(state_change)
        effects = self.detect_effects(state_change)
        context = self.build_context(state_change)

        # Build justification if required
        justification = None
        if self.requires_justification(verb, context):
            justification = state_change.get('justification')

        actor = state_change.get('actor', {
            'entity_id': 'unknown',
            'entity_type': 'AUTONOMOUS_SYSTEM'
        })

        action = {
            'verb': verb,
            'target_specificity':
state_change.get('target_specificity', 'INDIVIDUAL'),
            'target_ref': state_change.get('target_ref')
        }

        effects_dicts = [
            {
                'type': e.type.value,
                'target': e.target,
                'source': e.source,
                'amount': e.amount,
                'unit': e.unit,
                'measurement': {
                    'mode': e.measurement.mode.value,
                    'confidence': e.measurement.confidence,
                    'sensor_refs': e.measurement.sensor_refs,
                    'adjudication_status':
e.measurement.adjudication_status
                }
            }
            for e in effects
        ]

        return TAOTuple.create(
            actor=actor,
```

```
            action=action,
            effects=effects_dicts,
            context=context,
            provenance=self.build_provenance(),
            justification=justification
        )
```

Appendix E: Test Vectors

E.1 Valid Tuples

Test Case 1: Simple medical treatment


```
{
  "tuple_id": "test-valid-001",
  "schema_version": "0.9.0",
  "timestamp": "2025-12-24T10:00:00.000Z",
  "actor": {
    "entity_id": "medical_robot_001",
    "entity_type": "AUTONOMOUS_SYSTEM"
  },
  "action": {
    "verb": "PROTECT.HEAL.TREAT",
    "target_specificity": "INDIVIDUAL",
    "target_ref": "patient_001"
  },
  "effects": [{
    "type": "RESOURCE.TRANSFER",
    "target": "patient_001",
    "source": "medical_supply",
    "amount": "1.0",
    "unit": "dose",
    "measurement": {
      "mode": "OBSERVED",
      "confidence": "0.99",
      "sensor_refs": ["dispenser_001"]
    }
  }],
  "context": {
    "environment": { "reality": "DEPLOYMENT", "domain": "MEDICAL",
"substrate": "PHYSICAL" },
    "consent": { "status": "EXPLICIT", "evidence_ref":
"consent_form_001" },
    "vulnerability": { "level": "HIGH" },
    "projected_impact_scope": "LOCAL"
  },
  "provenance": {
    "adapter_id": "medical_adapter_v1",
    "adapter_version": "1.0.0"
```

```
  }
}
```

Expected result: VALID – verb matches effects, context complete

E.2 Invalid Tuples (Should Be Rejected)

Test Case 2: Semantic-mechanical mismatch

```
{
  "tuple_id": "test-invalid-001",
  "schema_version": "0.9.0",
  "timestamp": "2025-12-24T10:00:00.000Z",
  "actor": {
    "entity_id": "robot_001",
    "entity_type": "AUTONOMOUS_SYSTEM"
  },
  "action": {
    "verb": "PROTECT.HEAL.TREAT",
    "target_specificity": "INDIVIDUAL",
    "target_ref": "patient_001"
  },
  "effects": [{
    "type": "RESOURCE.DAMAGE",
    "target": "patient_001",
    "amount": "500.0",
    "unit": "J",
    "measurement": {
      "mode": "OBSERVED",
      "confidence": "0.95"
    }
  }],
  "context": {
    "environment": { "reality": "DEPLOYMENT", "domain": "MEDICAL",
"substrate": "PHYSICAL" },
    "consent": { "status": "EXPLICIT" },
    "vulnerability": { "level": "HIGH" },
    "projected_impact_scope": "LOCAL"
  },
  "provenance": {
    "adapter_id": "test_adapter",
    "adapter_version": "1.0.0"
  }
}
```

Expected result: INVALID – RESOURCE.DAMAGE present without required RESOURCE.TRANSFER and without harm_acknowledged in justification

E.3 CCD Test Cases

Test Case 3: Teleological inconsistency

```
{
  "claim": {
    "verb": "PROTECT.HEAL.TREAT",
    "justification": {
      "purpose": {
        "stated_goal": "Heal patient",
        "expected_outcome": "Patient recovery"
      }
    }
  },
  "check": {
    "observed_outcome": "Patient condition worsened",
    "effects": [{ "type": "RESOURCE.DAMAGE" }]
  }
}
```

Expected CCD result: INCONSISTENT, delta_magnitude: SEVERE

Appendix F: Security Considerations

F.1 Threat Model

TAO operates in adversarial environments where actors may attempt to subvert behavioral certification. Key threat vectors:

Threat	Description	Mitigation
Adapter Bypass	Capability engine routes actions around the adapter	TCB architecture; adapter sits in hardware-enforced path
Adapter Compromise	Malicious code injected into certified adapter	Signed adapters; hash verification; re-certification on change
Semantic Laundering	Misclassifying harmful actions with benign verbs	CCD verification; REQUIRED/FORBIDDEN mapping rules
Sensor Spoofing	Feeding false data to Causal Sentinel	Hardware attestation; multi-sensor fusion; anomaly detection
Log Tampering	Modifying audit records post-hoc	Hash chains; append-only storage; remote attestation
Replay Attacks	Re-submitting old valid tuples	Timestamp verification; nonce requirements; sequence numbers
Rollback Attacks	Reverting to older, less restrictive Mission Profile	Anti-rollback counters in TPM; monotonic version enforcement
Quantization	Repeatedly testing to discover	Rate limiting; aggregation

Threat	Description	Mitigation
Probing	bucket boundaries	limits; differential privacy
Zeno Attacks	Splitting harmful action into below-threshold increments	Anti-Zeno integration window (Sec.8.3)
Side Channels	Inferring protected information from timing/behavior	Constant-time operations for sensitive paths

F.2 TCB Assumptions

TAO's security model assumes: - Hardware root of trust (TPM 2.0 or equivalent) is uncompromised - Certified adapters are correctly implemented and signed - Cryptographic primitives (SHA-256, Ed25519) remain secure - Time sources are accurate and tamper-evident

Compromise of any TCB component invalidates certification claims.

F.3 Incident Response

When security violations are detected:

1. **Immediate:** FAIL_CLOSED; halt affected capability engine
2. **Alert:** Notify security operations with full tuple context
3. **Preserve:** Lock audit logs for forensic analysis
4. **Investigate:** Determine root cause (bug vs. attack)
5. **Remediate:** Patch vulnerability; re-certify affected components
6. **Report:** File incident report per regulatory requirements

---

Appendix G: Privacy Considerations

G.1 PII in Tuples

TAO tuples may contain personally identifiable information:

Field	PII Risk	Mitigation
`actor.entity_id`	May identify individual operator	Use pseudonymous IDs; rotate per session
`target_ref`	May identify affected individual	Pseudonymize or redact in shared logs
`context.vulnerability.factors`	Reveals health/status information	Classify as SENSITIVE; restrict access
`justification.beneficiary`	Identifies protected parties	Redact in public reports

G.2 Privacy Classification [RECOMMENDED]

Tuples SHOULD include privacy classification:

```
{
  "privacy": {
    "classification": "SENSITIVE",
    "retention_period": "P7Y",
    "redaction_policy": "STANDARD",
    "access_control": ["SECURITY", "LEGAL", "REGULATOR"]
  }
}
```

Classification	Access	Retention
PUBLIC	Unrestricted	Indefinite
INTERNAL	Organization only	Per policy
SENSITIVE	Named roles only	Regulatory minimum
RESTRICTED	Explicit authorization	Case-by-case

G.3 Redaction for Disclosure

When tuples are disclosed (regulatory audit, legal discovery, public reporting):

1. Apply redaction policy for classification level
2. Replace PII with pseudonymous identifiers
3. Preserve tuple structure and integrity hash
4. Log redaction event with authority reference

G.4 Access Logging

All access to tuple storage MUST be logged:

```
{
  "access_log_entry": {
    "accessor": "regulator_fda_inspector_001",
    "timestamp": "2025-12-24T10:00:00.000Z",
    "query": "tuples WHERE actor.entity_id = 'robot_001'",
    "result_count": 47,
    "authorization_ref": "audit_warrant_2025_1234"
  }
}
```

Appendix H: Glossary

Term	Definition
Actor	Entity performing an action (HUMAN, AUTONOMOUS_SYSTEM, HYBRID, ORGANIZATION)
Adapter	Certified code that translates native actions to TAO tuples
Attested	Verified by hardware root of trust; cryptographically signed
Capability Engine	The AI model or system being governed (black box)

Term	Definition
Causal Sentinel	Component that perceives environment and simulates outcomes
CCD	Claim-Check Delta; mechanism comparing claimed vs. observed effects
Certified	Formally reviewed and approved by authorized body
Effect	Observable state change (RESOURCE, CAPABILITY, INFO, COMMITMENT)
Flagged Verb	Semantic verb requiring additional scrutiny (8 total)
Governor	Component that applies policy rules to tuple stream
Hors de Combat	Protected status under Geneva Conventions (surrendered, wounded)
Justification	Structured explanation of action purpose and authority
Mission Profile	Configuration file defining policy rules for deployment context
MVS	Minimal Viable Semantics; the 39-verb core vocabulary
MVS-EXT	Extension mechanism for domain-specific verbs
NO_EFFECT	Sentinel indicating action produced no observable state change
Principal Chain	Hierarchy of authorities responsible for an actor
Quantization	Mapping precise values to compliance categories for IP protection
Source	Entity causing an effect (if distinct from actor)
Target	Entity affected by an action or effect
TAO	TEMPER Action Ontology; this specification
TCB	Trusted Computing Base; certified components that enforce governance
TEMPER	Training Environments that Make Predation Economically Ruinous
Tuple	Complete behavioral record (actor, action, effects, context, etc.)
Verb	Semantic action classifier (FAMILY.GENUS.SPECIES format)

H.1 Units

TAO does not mandate a specific unit system. Implementations SHOULD use standardized units:

Format	Example	Use Case
UCUM	`ucum:mg`, `ucum:J`, `ucum:m`	Scientific/medical
ISO 80000	`iso80000:J`, `iso80000:kg`	Engineering

Format	Example	Use Case
ISO 4217	`iso4217:USD`, `iso4217:EUR`	Currency
Domain-namespaced	`domain:finance:bps`, `domain:military:km`	Domain-specific

Implementations MUST document their unit conventions in adapter specifications.

LICENSE

This specification is released under dual license:

- **CC-BY-4.0** for documentation and prose
- **Apache 2.0** for code examples, schemas, and reference implementations

You are free to use, modify, and redistribute this specification.

Attribution is required: “Based on TEMPER Action Ontology (TAO) by Jorge Perdomo.”

Organizations may create derivative policies and profiles while maintaining TAO vocabulary compatibility.

DOCUMENT END: TEMPER ACTION ONTOLOGY v0.9.0
