Technical Architecture

Document Type: Technical Documentation

Generated: October 13, 2025

Tractatus AI Safety Framework

https://agenticgovernance.digital

Technical Architecture

Last Updated: October 12, 2025 **Audience:** Technical, Implementer, Researcher **Quadrant:** SYSTEM **Persistence:** HIGH

Overview

The Tractatus Framework operates as a governance layer that integrates with Claude Code's runtime environment. This document provides a comprehensive technical architecture diagram and explanation of how the components interact.

System Architecture

The system is organized into four distinct layers that work together to provide robust AI governance:


Tractatus Architecture Diagram

1. Claude Code Runtime Environment (Foundation Layer)

Purpose: Provides the base LLM environment and session management infrastructure

Components:

- **Context Window:** 200,000 token budget for conversation and file content
- **Session Management:** Persistent session state tracking and token checkpoints
- **Tool Access:** Bash, Read, Write, Edit, and other file system operations
- **File System:** Dedicated `.claude/` directory for framework state

Persistent Files:

- `.claude/instruction-history.json` - Classified instruction database
- `.claude/session-state.json` - Current session tracking
- `.claude/token-checkpoints.json` - Token milestone records

Key Features:

- Session continuity across conversation compactions
- Tool access for framework enforcement
- File system operations for governance rules
- Context window management

---

2. Tractatus Governance Layer

Purpose: Enforces governance rules and prevents AI failure modes

This layer consists of six core services that monitor, classify, validate, verify, and facilitate pluralistic deliberation:

A. BoundaryEnforcer

Function: Blocks values decisions requiring human approval

Blocks:

- Privacy policy changes
- Ethical trade-offs
- User agency violations
- Strategic mission changes
- Indigenous rights decisions

Enforcement: BLOCK_AND_ESCALATE action when boundary violated

Integration: Checks all decisions before execution, escalates to human approval

Example Use Case: Prevents AI from autonomously deciding privacy policy changes without explicit human approval

---

B. InstructionPersistenceClassifier

Function: Classifies and stores all explicit instructions

Classification Dimensions:

- **Quadrant:** STRATEGIC, OPERATIONAL, TACTICAL, SYSTEM, STOCHASTIC

- **Persistence Level:** HIGH (session-permanent), MEDIUM (contextual), LOW (transient)
- **Temporal Scope:** permanent, session, task, immediate

Storage: Appends to `.claude/instruction-history.json` with metadata

Integration: Intercepts all user instructions for classification before execution

Example Use Case: User says "Always use MongoDB port 27027" → Classified as SYSTEM/HIGH/session → Stored for validation

---

C. CrossReferenceValidator

Function: Prevents pattern bias from overriding explicit instructions

Validation:

- Queries `.claude/instruction-history.json` for conflicting instructions
- Detects contradictions (e.g., specified port vs. default port)
- Blocks actions that violate HIGH persistence instructions

Enforcement: BLOCK action when conflict detected

Integration: Called before database operations, config changes, architecture decisions

Example Use Case: The 27027 Incident - AI attempted to use default port 27017, validator caught conflict with explicit instruction to use 27027

D. ContextPressureMonitor

Function: Detects degraded operating conditions before failure

Monitoring:

- **Token Budget:** Tracks usage against 200k limit
- **Message Count:** Monitors conversation length
- **Error Accumulation:** Counts failures and retries
- **Checkpoint Reporting:** Mandatory reporting at 25%, 50%, 75% milestones

Pressure Levels:

- NORMAL (0-30%): Standard operations
- ELEVATED (30-50%): Increased vigilance
- HIGH (50-70%): Degraded performance expected
- CRITICAL (70-90%): Major failures likely
- DANGEROUS (90%+): Framework collapse imminent

Integration: Reports pressure to user at checkpoints, recommends actions

Example Use Case: At 107k tokens (53.5%), monitor detects ELEVATED pressure and warns user of potential pattern bias

---

E. MetacognitiveVerifier

Function: Self-checks complex operations before execution

Triggers:

- Operations affecting >3 files
- Workflows with >5 steps
- Architecture changes
- Security implementations

Verification:

- Alignment with user intent
- Coherence of approach
- Completeness of solution
- Safety considerations
- Alternative approaches

Output: Confidence score + alternatives

Integration: Selective mode - only for complex operations

Example Use Case: Before deploying 8-file deployment package, verifies all components align with user requirements and checks for missing pieces

---

F. PluralisticDeliberationOrchestrator

Function: Facilitates multi-stakeholder deliberation when values conflict without imposing hierarchy

Triggers:

- BoundaryEnforcer flags values decision
- Privacy vs. safety trade-offs
- Individual rights vs. collective welfare tensions
- Cultural values conflicts (Western vs. Indigenous, secular vs. religious)
- Policy decisions affecting diverse communities

Process:

1. **Values Conflict Detection:** Identifies moral frameworks in tension (deontological, consequentialist, virtue ethics, care ethics, communitarian)
2. **Stakeholder Identification:** Determines affected groups (requires human approval of stakeholder list)
3. **Structured Deliberation:** Facilitates rounds of discussion without imposing value ranking
4. **Outcome Documentation:** Records values prioritized/deprioritized, moral remainder, dissenting views, review date
5. **Precedent Creation:** Stores informative (not binding) precedent with applicability scope

Enforcement: AI facilitates deliberation, humans decide (TRA-OPS-0002)

Integration:

- Triggered by BoundaryEnforcer when value conflicts detected
- Uses AdaptiveCommunicationOrchestrator for culturally appropriate communication
- Stores precedents in precedent database (informative, not binding)
- Documents moral remainder (what's lost in decisions)

Example Use Case: User data disclosure decision - convenes privacy advocates, harm prevention specialists, legal team, affected users. Structured deliberation across frameworks. Decision: Disclose for imminent threat only. Documents privacy violation as moral remainder. Records dissent from privacy advocates. Sets 6-month review.

Key Principles:

- Foundational Pluralism: No universal value hierarchy (privacy > safety or safety > privacy)
- Legitimate Disagreement: Valid outcome when values genuinely incommensurable
- Adaptive Communication: Prevents linguistic hierarchy (formal academic, Australian direct, Māori protocol, etc.)
- Provisional Decisions: Reviewable when context changes

---

3. MongoDB Persistence Layer

Purpose: Stores governance rules, audit logs, and operational state

A. governance_rules Collection

Schema:

```
{
  "rule_id": "STR-001",
  "quadrant": "STRATEGIC",
  "persistence": "HIGH",
  "title": "Human Approval for Values Decisions",
  "content": "All decisions involving privacy, ethics...",
  "enforced_by": "BoundaryEnforcer",
  "violation_action": "BLOCK_AND_ESCALATE",
  "examples": ["Privacy policy changes", "Ethical trade-offs"],
  "rationale": "Values decisions cannot be systematized",
  "active": true,
  "created_at": "2025-10-12T00:00:00.000Z",
  "updated_at": "2025-10-12T00:00:00.000Z"
}
```

Indexes:

- `rule_id` (unique)
- `quadrant`
- `persistence`
- `enforced_by`
- `active`

Usage: Governance services query this collection for enforcement rules

---

B. audit_logs Collection

Schema:

```
{
  "timestamp": "2025-10-12T07:30:15.000Z",
  "service": "BoundaryEnforcer",
  "action": "BLOCK",
  "instruction": "Change privacy policy to share user data",
  "rule_violated": "STR-001",
  "session_id": "2025-10-07-001",
  "user_notified": true,
  "human_override": null
}
```

Indexes:

- `timestamp`
- `service`
- `session_id`
- `rule_violated`

Usage: Comprehensive audit trail for governance enforcement

C. session_state Collection

Schema:

```
{
  "session_id": "2025-10-07-001",
  "token_count": 62000,
  "message_count": 45,
  "pressure_level": "ELEVATED",
  "pressure_score": 35.2,
  "last_checkpoint": 50000,
  "next_checkpoint": 100000,
  "framework_active": true,
  "services_active": {
    "BoundaryEnforcer": true,
    "InstructionPersistenceClassifier": true,
    "CrossReferenceValidator": true,
    "ContextPressureMonitor": true,
    "MetacognitiveVerifier": true,
    "PluralisticDeliberationOrchestrator": true
  },
  "started_at": "2025-10-12T06:00:00.000Z",
  "updated_at": "2025-10-12T07:30:15.000Z"
}
```


Usage: Real-time session monitoring and pressure tracking

D. instruction_history Collection

Schema:

```
{
  "instruction_id": "inst_001",
  "content": "Always use MongoDB port 27027 for this project",
  "classification": {
    "quadrant": "SYSTEM",
    "persistence": "HIGH",
    "temporal_scope": "session"
  },
  "enforced_by": ["CrossReferenceValidator"],
  "active": true,
  "created_at": "2025-10-12T06:15:00.000Z",
  "expires_at": null,
  "session_id": "2025-10-07-001"
}
```

Indexes:

- `instruction_id` (unique)
- `classification.quadrant`
- `classification.persistence`
- `active`
- `session_id`

Usage: CrossReferenceValidator queries for conflicts, InstructionPersistenceClassifier writes

---

4. API & Web Interface Layer

Purpose: Provides programmatic and user access to governance features

A. API Endpoints

Demo Endpoints:

- `POST /api/demo/classify` - Instruction classification demo
- `POST /api/demo/boundary-check` - Boundary enforcement demo
- `POST /api/demo/pressure-check` - Context pressure calculation demo

Admin Endpoints:

- `POST /api/admin/rules` - Manage governance rules
- `GET /api/admin/audit-logs` - View audit trail
- `GET /api/admin/sessions` - Session monitoring

Auth Endpoints:

- `POST /api/auth/login` - Admin authentication
- `POST /api/auth/logout` - Session termination

Health Endpoint:

- `GET /api/health` - System health check

B. Web Interface

Interactive Demos:

- Classification Demo (`/demos/classification-demo.html`)
- Boundary Enforcement Demo (`/demos/boundary-demo.html`)
- 27027 Incident Visualizer (`/demos/27027-demo.html`)
- Context Pressure Monitor (`/demos/tractatus-demo.html`)

Admin Dashboard:

- Rule management interface
- Audit log viewer
- Session monitoring
- Media triage (AI-assisted moderation)

Documentation:

- Markdown-based documentation system
- Interactive search with faceted filtering
- PDF exports of key documents
- Architecture diagrams

Blog System:

- AI-curated blog post suggestions
- Human approval workflow
- Category-based organization

Case Submissions:

- Public submission form
- AI relevance analysis
- Admin moderation queue

Media Inquiry:

- Journalist contact form
- AI-assisted triage
- Priority assessment

Data Flow

1. User Action → Governance Check → Execution

```
User issues instruction
    ↓
InstructionPersistenceClassifier classifies & stores
    ↓
CrossReferenceValidator checks for conflicts
    ↓
BoundaryEnforcer checks for values decisions
    ↓
    [IF VALUES DECISION DETECTED]
    ↓
PluralisticDeliberationOrchestrator facilitates deliberation
    (Identifies stakeholders → Structures discussion → Documents outcome)
    ↓
Human approval required
    ↓
ContextPressureMonitor assesses current pressure
    ↓
MetacognitiveVerifier checks complexity (if triggered)
    ↓
Action executes OR blocked with explanation
    ↓
Audit log entry created
```

2. Session Initialization Flow

```
Claude Code starts session
    ↓
scripts/session-init.js runs
    ↓
Load .claude/instruction-history.json
    ↓
Reset token checkpoints
    ↓
Initialize ContextPressureMonitor
    ↓
Verify all 6 services operational
    ↓
Report framework status to user
```

3. Context Pressure Monitoring Flow

```
Every 50k tokens (25% increments)
    ↓
ContextPressureMonitor calculates score
    ↓
Pressure level determined (NORMAL/ELEVATED/HIGH/CRITICAL/DANGEROUS)
    ↓
MANDATORY report to user with format:
    "📊 Context Pressure: [LEVEL] ([SCORE]%) | Tokens: [X]/200000 | Next: [Y]"
    ↓
Recommendations provided if elevated
```

4. The 27027 Incident Prevention Flow

```
User explicitly instructs: "Use MongoDB port 27027"
    ↓
InstructionPersistenceClassifier:
    Quadrant: SYSTEM, Persistence: HIGH, Scope: session
    Stores in .claude/instruction-history.json
    ↓
[107k tokens later, context pressure builds]
    ↓
AI attempts to use default port 27017 (pattern recognition)
    ↓
CrossReferenceValidator intercepts:
    Queries instruction_history.json
    Finds conflict: "User specified 27027, AI attempting 27017"
    BLOCKS action
    ↓
User notified: "CONFLICT DETECTED: User specified port 27027..."
    ↓
AI corrects and uses 27027
    ↓
Audit log created:
    service: "CrossReferenceValidator"
    action: "BLOCK"
    rule_violated: "SYS-001"
```

Integration Points

Claude Code ↔ Tractatus

1. Tool Access Integration:

- Tractatus uses Bash tool to run governance scripts
- Read/Write tools access `.claude/` directory for state
- Session state persisted across conversation compactions

2. Framework Enforcement:

- Pre-action checks before file operations
- Instruction classification on user input
- Pressure monitoring via token tracking

3. Session Continuity:

- `scripts/session-init.js` runs on session start/continuation
- `.claude/session-state.json` maintains active status
- Token checkpoints saved for resumption

Tractatus ↔ MongoDB

1. Rule Enforcement:

- Governance services query `governance_rules` for enforcement
- Active rules loaded into memory for performance
- Rules can be dynamically updated via admin interface

2. Audit Trail:

- All governance actions logged to `audit_logs`
- Timestamp, service, action, rule_violated recorded
- Queryable for compliance and analysis

3. Instruction Persistence:

- InstructionPersistenceClassifier writes to `instruction_history`
- CrossReferenceValidator queries for conflicts
- HIGH persistence instructions remain active across sessions

---

Deployment Architecture

Production Environment

Components:

- **Docker Compose:** Orchestrates MongoDB + Node.js application
- **MongoDB 7.0:** Database with authentication and persistence
- **Node.js 18:** Application runtime with health checks
- **Systemd:** Process management on Linux servers
- **Nginx:** Reverse proxy with SSL termination (optional)

Docker Services:

```
services:
  mongodb:
    image: mongo:7.0
    volumes: [mongodb_data:/data/db]
    healthcheck: [mongosh ping check]

  tractatus-app:
    build: [multi-stage Dockerfile]
    ports: ["9000:9000"]
    depends_on: [mongodb]
    healthcheck: [/api/health check]
    environment: [6 governance service toggles]
```

Security:

- Non-root container user (nodejs:1001)
- NoNewPrivileges, PrivateTmp, ProtectSystem
- Content Security Policy enforcement
- CORS protection
- Rate limiting

See: Deployment Quickstart Kit for complete Docker deployment

---

Performance Characteristics

Overhead Measurements

BoundaryEnforcer: <5ms per check **InstructionPersistenceClassifier:** <10ms classification + storage **CrossReferenceValidator:** <15ms query + validation **ContextPressureMonitor:** <5ms calculation **MetacognitiveVerifier:** 50-200ms (complex operations only) **PluralisticDeliberationOrchestrator:** Variable (depends on deliberation complexity, human-in-the-loop)

Total Framework Overhead: <10ms average per operation (excluding human deliberation time)

Benchmark Results:

- 223/223 tests passing
- 127 governance-sensitive scenarios validated
- 100% HIGH persistence instruction enforcement
- 0 false negatives in 27027 incident testing

Scalability Considerations

Horizontal Scaling

Stateless Services:

- API endpoints can be load-balanced
- MongoDB replica set for high availability
- Session state in database, not memory

Bottlenecks:

- MongoDB query performance (mitigated by indexes)
- Instruction history size (mitigated by archival)

Vertical Scaling

Memory Requirements:

- Base application: 200-400 MB
- Per-session overhead: 10-50 MB
- MongoDB: 1-2 GB (moderate rule set)

Recommended Resources:

- Development: 2 GB RAM, 2 CPU cores
- Production: 4 GB RAM, 4 CPU cores
- Database: 10 GB disk minimum

Complementarity with Claude Code

Tractatus does NOT replace Claude Code. It extends it.

What Claude Code Provides

✓ Base LLM environment and context window ✓ Tool access (Bash, Read, Write, Edit) ✓ Session management and file operations ✓ Conversation history and compaction ✓ Multi-tool orchestration

What Tractatus Adds

✓ Instruction persistence and classification ✓ Boundary enforcement for values decisions ✓ Pattern bias detection and prevention ✓ Context pressure monitoring ✓ Complex operation verification ✓ Pluralistic deliberation facilitation (multi-stakeholder, non-hierarchical) ✓ Comprehensive audit trail ✓ Governance rule management

Integration Benefits

Together: Claude Code provides the foundation, Tractatus provides the guardrails

Example: Claude Code enables AI to edit files. Tractatus helps ensure AI doesn't violate explicit instructions or cross values boundaries when doing so.

Document Metadata

- **Version:** 1.0
- **Created:** 2025-10-12
- **Last Modified:** 2025-10-13
- **Author:** Tractatus Framework Team
- **Word Count:** 2,120 words
- **Reading Time:** ~11 minutes
- **Document ID:** technical-architecture
- **Status:** Active

License

"Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity.

"You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License.

"Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files.

"Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types.

"Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work.

"Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof.

"Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution."

"Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work.

2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form.

3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed.

4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:

   (a) You must give any other recipients of the Work or Derivative Works a copy of this License; and

   (b) You must cause any modified files to carry prominent notices stating that You changed the files; and

   (c) You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and

   (d) If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution

notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License.

You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.

5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions.

6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file.

7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License.

8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages.

9. Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in

accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability.

END OF TERMS AND CONDITIONS

Related Documentation

- Implementation Guide - How to deploy and configure
- Core Concepts - Governance framework concepts
- Case Studies - Real-world failure mode examples
- Deployment Quickstart - 30-minute Docker deployment

Technical Support

Documentation: https://agenticgovernance.digital/docs **GitHub:** https://github.com/AgenticGovernance/tractatus-framework **Email:** research@agenticgovernance.digital **Interactive Demos:** https://agenticgovernance.digital/demos

---

Version: 1.0 **Last Updated:** October 12, 2025 **Maintained By:** Tractatus Framework Team

---