The Endpoint Developer Framework: Rules of the Road and Governance Through the Trusted Endpoint Review Board

Executive Summary

The AI Trust Chain framework fundamentally relies on the integrity and reliability of its endpoints—the sources of assertions that propagate through decision-making systems. This paper proposes a comprehensive approach for endpoint developers that establishes clear operational guidelines ("rules of the road") and introduces the Trusted Endpoint Review Board (TERB) as a governance mechanism. Recognizing that endpoint developers hold significant power in shaping trust propagation throughout AI systems, this framework balances developer autonomy with systematic oversight to ensure the overall integrity of the trust chain.

1. Introduction

1.1 The Critical Role of Endpoints

In the AI Trust Chain framework, endpoints serve as the foundational sources of truth. Whether they are sensors reporting environmental data, ML models making predictions, or LLMs generating recommendations, endpoints initiate the assertions that cascade through decision-support systems. The trust and confidence values assigned at the endpoint level fundamentally determine the reliability of downstream decisions.

1.2 The Power Asymmetry Challenge

Endpoint developers possess significant influence over system behavior through several mechanisms:

- **Confidence Calculation**: Implementation decisions directly affect reported confidence levels
- **Data Quality Control**: The accuracy and completeness of assertions originate from developer choices
- **Temporal Validity**: Developers control how their endpoints handle time-sensitive information
- **Materiality Weighting**: Developers determine how much their endpoints rely on consumed assertions
- **Content Structure**: The actual data and metadata provided in assertions
- **Trust Explanation**: Developers MUST provide context that influences trust interpretation

Note that trust ceilings for endpoint classes are centrally controlled by administrators through the `/api/v1/trust/endpoint-class` admin endpoint, providing a key governance mechanism. However, developers still wield considerable power through their implementation choices, necessitating a structured approach to ensure responsible development while maintaining innovation flexibility.

2. Rules of the Road for Endpoint Developers

2.1 Core Principles

2.1.1 Transparency First

- All endpoints must provide clear documentation of their confidence calculation methodology
- Confidence intervals must be empirically justified, not arbitrarily assigned
- Known limitations must be explicitly declared in endpoint metadata
- Trust explanations MUST provide meaningful context for downstream consumers

2.1.2 Fail-Safe Design

- Endpoints must degrade gracefully when operating outside nominal conditions
- Uncertainty must increase appropriately when data quality degrades
- Emergency shutdown mechanisms must be implemented for critical endpoints

2.1.3 Auditability

- All assertions must include sufficient provenance for reconstruction
- Decision logic must be inspectable, even for ML-based endpoints
- Version control must track all changes to confidence calculations

2.2 Technical Requirements

2.2.1 Understanding the Dual-Channel Architecture

The AI Trust Chain framework implements a dual-channel architecture:

1. **Content Channel**: The actual data (temperature readings, predictions, recommendations)
2. **Metadata Channel**: Trust values, confidence, explanations, and provenance

Important: Developers submit content, confidence, AND trust_explanation; the framework adds trust metadata automatically.

2.2.2 What Developers Submit vs. What the Framework Stores

Developer Submission (API Request):

```
{
 "endpoint_id": "temp-sensor-01",
 "endpoint_class": "sensor.temperature.honeywell_t7771a",
 "endpoint_type": "sensor",
 "content": {
  "temperature": 75.2,
  "unit": "fahrenheit",
  "calibration_age_days": 180,
  "environmental_conditions": "optimal"
 },
 "confidence": 0.95,
 "trust_explanation": "Sensor operating within specifications. Calibration current.",
 "consumed_assertions": ["assertion_id_1", "assertion_id_2"],
 "consumed_assertion_materiality": {
  "assertion_id_1": [0.8, true], "assertion_id_2": [0.2, false]
 },
 "limitations": {
  "accuracy": "±0.5°F",
    "calibration_status": "due in 185 days"
  }
}
```

Complete Stored Assertion (What Actually Gets Stored):

```
{
  "id": "550e8400-e29b-41d4-a716-446655440000",
  "endpoint_id": "temp-sensor-01",
  "endpoint_class": "sensor.temperature.honeywell_t7771a",
  "endpoint_type": "sensor",
  "content": {
    "temperature": 75.2,
    "unit": "fahrenheit",
    "calibration_age_days": 180,
    "environmental_conditions": "optimal"
  },
  "metadata": {
    "trust_value": 0.70,
    "confidence_value": 0.95,
    "trust_explanation": "Sensor operating within specifications. Calibration current.",
    "temporal_validity": 0.98,
    "endpoint_id": "temp-sensor-01",
    "endpoint_type": "sensor",
    "endpoint_class": "sensor.temperature.honeywell_t7771a",
    "timestamp": 1737500000.0,
    "provenance": [
      "sensor.temperature.honeywell_t7771a",
      "data_source.weather_api",
      "ml_model.temp_validator"
    ],
    "limitations": {
      "accuracy": "±0.5°F",
      "calibration_status": "due in 185 days"
    },
    "context": {}
  },
  "consumed_assertions": ["assertion_id_1", "assertion_id_2"],
  "timestamp": 1737500000.0,
  "block_hash": null
}
```

Key Developer-Provided Fields:

- `endpoint_id`: Unique identifier for the endpoint
- `endpoint_class`: Specific class of endpoint
- `endpoint_type`: Category (sensor, ml_model, llm, data_source)
- `content`: The actual data/payload
- `confidence`: Developer's confidence in this specific assertion (0.0-1.0)
- `trust_explanation`: REQUIRED - Meaningful context about the assertion
- `consumed_assertions`: List of assertion IDs this depends on (if any)
- `consumed_assertion_materiality`: Weights and trust impact flags for consumed assertions
- `limitations`: Optional but recommended constraints/caveats

Key Framework-Managed Fields:

- `id`: Unique assertion identifier (immutable)
- `metadata.trust_value`: Calculated based on endpoint class ceiling and propagation rules

- metadata.temporal_validity: Time-based trust decay factor
- metadata.timestamp: When assertion was created
- metadata.provenance: Complete chain of endpoint classes
- block_hash: Blockchain reference (when mined)

2.2.3 Trust Value Calculation

The framework automatically calculates trust values based on:

1. **For Root Assertions** (no consumed assertions):
    - Trust value = Trust ceiling for endpoint class (set by administrators)
2. **For Derived Assertions** (consuming other assertions):
    - Trust value = MIN(Propagated trust from consumed assertions, Trust ceiling)
    - Propagation method depends on configured rules (minimum, weighted average, consensus)

Example Trust Propagation:

```
# Developer submits:
consumed_assertions = ["assertion_1", "assertion_2"]
consumed_assertion_materiality = {
   "assertion_1": [0.8, True],  # 80% weight, affects trust
   "assertion_2": [0.2, False]   # 20% weight, context only
}

# Framework retrieves:
assertion_1_trust = 0.85  # From stored assertion
assertion_2_trust = 0.60  # From stored assertion

# Framework calculates (weighted average method):
# Only assertion_1 affects trust (apply_to_trust=True)
propagated_trust = 0.85  # Since only one affects trust
endpoint_ceiling = 0.70  # From trust registry

# Final trust value:
trust_value = min(propagated_trust, endpoint_ceiling) = 0.70
```

2.2.4 Trust Explanation Guidelines

Developers **MUST** provide meaningful trust explanations for every assertion:

```
# Root assertion - explain why this endpoint is trustworthy
trust_explanation = "Sensor calibrated 180 days ago. Environmental conditions optimal. Cross-validated with redundant sensor."

# Derived assertion - explain trust interpretation
trust_explanation = "High confidence despite upstream uncertainty. Environmental controls compensate for sensor drift noted in consumed assertion. Fallback to secondary sensor available if primary fails."

# Low trust scenario - explain the issues
trust_explanation = "Sensor showing signs of drift. Calibration overdue by 30 days. Results should be treated with caution."
```

Trust explanations are **REQUIRED** because:

- Endpoints understand their own operational status
- They can interpret consumed assertions in context

- They provide transparency for audit trails
- Generic framework explanations would be unhelpful

Trust explanations may be developed using AI (LLM, SLM) or other logic to inform the downstream processing.

2.3 Assertion Classes with Complete Examples

For Sensor Endpoints

Developer Submits:

```
{
  "endpoint_id": "pressure-sensor-main",
  "endpoint_class": "sensor.pressure.bosch_bmp280",
  "endpoint_type": "sensor",
  "content": {
    "pressure": 1013.25,
    "unit": "hPa",
    "altitude": 100,
    "measurement_uncertainty": 1.0
  },
  "confidence": 0.92,
  "trust_explanation": "Sensor within operational range. Temperature compensation active. Last calibrated 45 days ago.",
  "limitations": {
    "range": "300-1100 hPa",
    "temperature_compensated": true
  }
}
```

Framework Stores (showing added metadata):

```
{
  "id": "auto-generated-uuid",
  "metadata": {
    "trust_value": 0.85,
    "confidence_value": 0.92,
    "trust_explanation": "Sensor within operational range. Temperature compensation active. Last calibrated 45 days ago.",
    "temporal_validity": 1.0
  }
}
```

For ML Model Endpoints

Developer Submits:

```
{
  "endpoint_id": "failure-predictor-v2",
  "endpoint_class": "ml_model.failure_prediction.xgboost",
  "endpoint_type": "ml_model",
  "content": {
    "prediction": "failure_likely",
    "probability": 0.78,
    "feature_importance": {
      "temperature": 0.45,
      "vibration": 0.35,
      "runtime_hours": 0.20
    }
```

```
 },
 "confidence": 0.73,
 "trust_explanation": "Model confidence reduced due to distribution shift detected in temperature features. Retrained 7 days ago on 50K samples.",
 "consumed_assertions": ["temp_assertion_id", "vibration_assertion_id"],
 "consumed_assertion_materiality": {
  "temp_assertion_id": [0.45, true],
  "vibration_assertion_id": [0.35, true]
 }
}
```

Framework Stores:

```
{
 "metadata": {
  "trust_value": 0.65,
  "confidence_value": 0.73,
  "trust_explanation": "Model confidence reduced due to distribution shift detected in temperature features. Retrained 7 days ago on 50K samples.",
  "provenance": [
   "sensor.temperature.honeywell_t7771a",
   "sensor.vibration.pcb_356a02",
   "ml_model.failure_prediction.xgboost"
  ]
 }
}
```

For LLM Endpoints

Developer Submits:

```
{
 "endpoint_id": "diagnostic-llm-prod",
 "endpoint_class": "llm.diagnostic.gpt4_finetuned",
 "endpoint_type": "llm",
 "content": {
  "diagnosis": "Bearing wear detected. Recommend inspection within 72 hours.",
  "reasoning": "Temperature and vibration patterns consistent with early-stage bearing degradation.",
  "confidence_factors": {
   "pattern_match": 0.82,
   "historical_accuracy": 0.91,
   "data_quality": 0.70
  }
 },
 "confidence": 0.81,
 "trust_explanation": "Diagnosis based on pattern recognition with 91% historical accuracy on similar cases. Upstream ML model shows distribution shift which may affect reliability.",
 "consumed_assertions": ["ml_prediction_id", "temp_assertion_id", "vibration_assertion_id"],
 "consumed_assertion_materiality": {
  "ml_prediction_id": [0.5, true],
  "temp_assertion_id": [0.3, true],
  "vibration_assertion_id": [0.2, true]
 }
}
```

Framework Stores:

```
{
 "metadata": {
  "trust_value": 0.62,
  "confidence_value": 0.81,
  "trust_explanation": "Diagnosis based on pattern recognition with 91% historical accuracy on similar cases. Upstream ML model
shows distribution shift which may affect reliability.",
  "provenance": [
   "sensor.temperature.honeywell_t7771a",
   "sensor.vibration.pcb_356a02",
   "ml_model.failure_prediction.xgboost",
   "llm.diagnostic.gpt4_finetuned"
  ]
 }
}
```

2.4 Behavioral Guidelines

2.4.1 Conservative Confidence Reporting

- When in doubt, underestimate confidence rather than overestimate
- Implement staged rollout for confidence calculation changes
- Maintain historical confidence accuracy metrics

2.4.2 Materiality Honesty

- Accurately report the materiality of consumed assertions
- Avoid artificial inflation of self-importance in assertion chains
- Document the rationale for materiality weightings
- Use tuple format: `[weight, apply_to_trust]` where weight is 0.0-1.0 and apply_to_trust is boolean

2.4.3 Trust Explanation Best Practices

- Provide meaningful context, not just "ok" or "functioning normally"
- Explain any factors that might affect trust
- For AI-enabled endpoints, interpret upstream explanations and add insights
- Include remediation suggestions when trust is low
- Reference specific operational parameters (calibration age, sample size, etc.)

2.4.4 Continuous Improvement

- Regularly review endpoint performance against real-world outcomes
- Implement feedback loops for confidence calibration
- Participate in cross-endpoint validation exercises

3. The Trusted Endpoint Review Board (TERB)

3.1 Mission and Mandate

The Trusted Endpoint Review Board serves as the governance body responsible for:

- Certifying new endpoint classes for production use
- Recommending trust ceilings to administrators
- Investigating trust chain failures and anomalies
- Establishing and updating technical standards
- Mediating disputes between endpoint developers

- Reviewing trust explanation quality and effectiveness

3.2 Board Composition

The TERB should consist of:

- **Technical Representatives (40%)**: Senior endpoint developers from different domains
- **Trust Chain Architects (20%)**: System-level designers who understand propagation dynamics
- **Domain Experts (20%)**: Subject matter experts from critical application areas
- **External Auditors (10%)**: Independent security and reliability experts
- **User Representatives (10%)**: Stakeholders who consume trust chain outputs

Rotation policy: Members serve 2-year terms with 50% rotation annually to maintain continuity.

3.3 Certification Process

3.3.1 Initial Certification

New endpoints must undergo:

1. **Documentation Review**: Complete technical specifications including:
    - Confidence calculation methodology
    - Sample assertions showing both submission and stored format
    - Trust explanation generation approach
    - All required fields properly documented
2. **Trust Ceiling Proposal**: TERB recommends appropriate trust ceiling to administrators based on:
    - Endpoint reliability history
    - Validation methodology
    - Criticality of use cases
3. **Simulation Testing**: Performance evaluation including:
    - Trust propagation behavior
    - Confidence calibration accuracy
    - Trust explanation quality assessment
4. **Pilot Deployment**: Limited production testing with monitoring
5. **Peer Review**: Assessment by existing certified endpoint developers
6. **Board Approval**: Final certification vote requiring 2/3 majority
7. **Admin Configuration**: Administrators set the trust ceiling in the registry

3.3.2 Certification Levels

Level 1 - Experimental:

- Recommended trust ceiling: 0.3-0.5
- Limited to non-critical decision chains
- Requires monthly performance reporting
- Must demonstrate understanding of trust vs. confidence distinction
- Must provide valid trust explanations

Level 2 - Standard:

- Recommended trust ceiling: 0.5-0.75
- Approved for standard operational use
- Quarterly review cycle
- Must show proper trust explanation generation
- Validated confidence calculations

Level 3 - Critical:

- Recommended trust ceiling: 0.75-0.95
- Authorized for safety-critical applications
- Continuous monitoring required
- Must demonstrate sophisticated trust interpretation capabilities
- Exemplary trust explanation quality

Note: Actual trust ceilings are set by administrators via the `/api/v1/trust/endpoint-class` endpoint based on TERB recommendations.

3.4 Enforcement Mechanisms

3.4.1 Automated Compliance Monitoring

The framework monitors:

- Confidence calibration accuracy
- Trust explanation quality and presence
- Materiality weighting consistency
- Proper use of consumed assertions
- Required field completeness

3.4.2 Graduated Response Protocol

1. **Warning**: Automated notification of non-compliance
2. **Probation**: Trust ceiling reduction by 20%
3. **Suspension**: Temporary removal from production chains or set Trust Ceiling = 0.1
4. **Revocation**: Permanent decertification

3.5 Innovation Sandbox

To balance oversight with innovation, the TERB maintains an "innovation sandbox" where:

- Experimental endpoints can operate without full certification
- Trust ceilings are capped at 0.3
- Data is segregated from production systems
- Rapid iteration is encouraged
- Successful experiments fast-track to certification

4. Implementation Roadmap

Phase 1: Foundation (Months 1-3)

- Establish TERB charter and initial membership
- Define core technical standards
- Develop automated monitoring infrastructure
- Create certification application portal
- Document trust value calculation examples

Phase 2: Pilot Program (Months 4-6)

- Certify 10-15 initial endpoints across different classes
- Refine certification process based on feedback
- Implement compliance monitoring dashboard
- Conduct first quarterly review cycle
- Publish trust explanation best practices

Phase 3: Scale-Up (Months 7-12)

- Open certification to all endpoint developers
- Launch innovation sandbox
- Implement automated enforcement mechanisms
- Publish first annual trust chain integrity report

Phase 4: Maturation (Year 2+)

- Introduce advanced certification levels
- Develop inter-chain trust federation standards
- Establish international TERB collaboration
- Create endpoint developer certification program

5. Risk Mitigation Strategies

5.1 Preventing Regulatory Capture

- Mandatory rotation of board members
- Public disclosure of all certification decisions
- External audit requirements
- Whistleblower protection for reporting violations

5.2 Avoiding Innovation Stifling

- Fast-track certification for proven technologies
- Innovation sandbox for experimentation
- Exemptions for research applications
- Regular review of outdated requirements

5.3 Managing Cascading Failures

- Endpoint diversity requirements for critical chains
- Automated circuit breakers for anomalous trust propagation
- Redundant certification paths
- Emergency override protocols

6. Success Metrics

6.1 System-Level Metrics

- **Trust Chain Reliability**: % of decisions with accurate trust assessments
- **Endpoint Diversity Index**: Measure of endpoint variety in critical chains
- **Mean Time to Certification**: Average duration of certification process
- **Compliance Rate**: % of endpoints meeting all requirements
- **Trust Explanation Quality Score**: Average rating of explanation usefulness

6.2 Developer Experience Metrics

- **Developer Satisfaction Score**: Survey-based assessment
- **Certification Success Rate**: % of applications approved
- **Innovation Velocity**: New endpoint classes certified per quarter
- **Support Ticket Resolution Time**: Average time to resolve developer issues

6.3 Governance Effectiveness Metrics

- **Decision Transparency Score**: Public visibility of board decisions
- **Appeal Success Rate**: % of appeals resulting in reversal
- **Stakeholder Representation Index**: Diversity of board composition
- **Enforcement Consistency**: Variance in penalties for similar violations

7. Case Studies

7.1 Temperature Sensor Certification

A Honeywell T7771A temperature sensor seeks Level 2 certification:

Developer Implementation (What They Submit):

```
{
 "endpoint_id": "temp-sensor-01",
 "endpoint_class": "sensor.temperature.honeywell_t7771a",
 "endpoint_type": "sensor",
 "content": {
  "temperature": 75.2,
  "unit": "fahrenheit",
  "calibration_age_days": 180,
  "environmental_conditions": "optimal"
 },
 "confidence": 0.95,
  "trust_explanation": "Sensor calibrated 180 days ago, operating in optimal environmental conditions. Cross-validated with redundant
sensor showing 0.1°F variance."
}
```

What Gets Stored (Framework Adds Metadata):

```
{
  "id": "550e8400-e29b-41d4-a716-446655440000",
  "endpoint_id": "temp-sensor-01",
  "endpoint_class": "sensor.temperature.honeywell_t7771a",
  "endpoint_type": "sensor",
  "content": {
    "temperature": 75.2,
    "unit": "fahrenheit",
    "calibration_age_days": 180,
    "environmental_conditions": "optimal"
  },
  "metadata": {
    "trust_value": 0.70,
    "confidence_value": 0.95,
    "trust_explanation": "Sensor calibrated 180 days ago, operating in optimal environmental conditions. Cross-validated with redundant
sensor showing 0.1°F variance.",
    "temporal_validity": 1.0,
    "endpoint_id": "temp-sensor-01",
    "endpoint_type": "sensor",
    "endpoint_class": "sensor.temperature.honeywell_t7771a",
    "timestamp": 1737500000.0,
    "provenance": ["sensor.temperature.honeywell_t7771a"],
    "limitations": {},
    "context": {}
  },
  "consumed_assertions": [],
  "timestamp": 1737500000.0
}
```

TERB Review Process:

1. Documentation confirms empirical validation of confidence calculation
2. Simulation shows appropriate confidence degradation with environmental changes
3. Trust explanations are meaningful and include specific metrics
4. Pilot deployment reveals accurate reporting
5. Peer review suggests adding calibration age to content
6. Board approves with Level 2 certification
7. TERB recommends trust ceiling of 0.70 to administrators
8. Admin sets trust ceiling via:

```
POST /api/v1/trust/endpoint-class
{
  "endpoint_class": "sensor.temperature.honeywell_t7771a",
  "max_trust": 0.70
}
```

7.2 Predictive ML Model Challenge

A failure prediction model experiences trust ceiling reduction:

Issue Identified:

- Confidence values consistently overestimated by 15%
- Distributional shift detection not implemented
- Materiality scores artificially inflated
- Trust explanations lack meaningful context (e.g., "Model functioning normally")

TERB Response:

1. Issue warning with 30-day correction period
2. Reduce trust ceiling from 0.70 to 0.56 (20% reduction)
3. Require implementation of drift detection
4. Mandate monthly confidence calibration reports
5. Require enhanced trust explanations with specific metrics
6. Schedule re-review in 90 days

8. Future Considerations

8.1 Cross-Chain Trust Federation

As AI Trust Chain implementations proliferate, establishing trust relationships between different chains becomes critical. The TERB framework should evolve to support:

- Mutual recognition agreements between TERBs
- Standardized trust translation protocols
- Cross-chain endpoint certification portability
- Unified trust explanation standards

8.2 Automated Governance

Machine learning models could augment TERB operations through:

- Anomaly detection in endpoint behavior
- Predictive identification of likely violations
- Optimization of certification requirements
- Automated preliminary review of applications
- Trust explanation quality assessment

8.3 Regulatory Integration

As AI governance regulations mature, the TERB should:

- Align with emerging regulatory standards
- Serve as a bridge between technical and regulatory communities
- Provide evidence for regulatory compliance
- Influence policy development through operational insights

9. Conclusion

The Endpoint Developer Framework and Trusted Endpoint Review Board represent essential governance infrastructure for maintaining AI Trust Chain integrity. By establishing clear rules of the road while respecting developer autonomy, this framework enables responsible innovation without sacrificing system reliability.

The concentration of power in endpoint developers' hands is not inherently problematic—it reflects the technical reality of distributed trust systems. However, this power must be balanced with accountability, transparency, and systematic oversight. The TERB provides this balance through a structured yet flexible governance approach that adapts to technological evolution while maintaining core integrity principles.

The clear separation between developer-controlled elements (confidence, trust explanations, content) and framework-managed trust values ensures system integrity while allowing developers to accurately represent their endpoints' capabilities. The dual-channel architecture, with its distinct content and metadata streams, provides the transparency necessary for effective governance and auditability.

Success requires buy-in from all stakeholders: endpoint developers who embrace responsible practices and understand the trust architecture, system architects who design for auditability, domain experts who provide contextual wisdom, and users who demand trustworthy AI systems. With committed participation across these groups, the framework can achieve its dual goals of enabling innovation and ensuring trust.

The journey toward trustworthy AI is not a destination but a continuous process of improvement, learning, and adaptation. The Endpoint Developer Framework and TERB provide the organizational structures and technical standards necessary for this journey, creating a foundation for AI systems that are not just powerful, but genuinely worthy of our trust.

Appendix A: Technical Specifications

A.1 Endpoint Registration with TERB

```
POST /api/v1/terb/register
{
  "endpoint_class": "sensor.temperature.honeywell_t7771a",
  "endpoint_type": "sensor",
  "confidence_methodology": {
    "calculation_method": "Statistical analysis of measurement uncertainty",
    "uncertainty_model": "Gaussian distribution with σ = 0.5°F",
    "validation_results": "99.2% accuracy over 10,000 samples"
  },
  "trust_explanation_approach": "Auto-generated based on calibration status, environmental conditions, and cross-validation results",
  "materiality_approach": {
    "dependency_weighting": "Not applicable - root sensor",
    "propagation_behavior": "N/A"
  },
  "certification_level_requested": 2,
  "supporting_documentation": ["url_1", "url_2"],
  "recommended_trust_ceiling": 0.70,
  "sample_assertion": {
    "submitted": {
      "endpoint_id": "temp-sensor-01",
      "endpoint_class": "sensor.temperature.honeywell_t7771a",
      "endpoint_type": "sensor",
      "content": {
        "temperature": 75.2,
        "unit": "fahrenheit"
      },
      "confidence": 0.95,
      "trust_explanation": "Sensor operating within specifications. Calibration current."
    },
    "stored": {
      "id": "uuid",
      "metadata": {
        "trust_value": 0.70,
        "confidence_value": 0.95,
        "trust_explanation": "Sensor operating within specifications. Calibration current."
      }
    }
  }
}
```

A.2 Assertion Creation (Developer Responsibility)

```
POST /api/v1/assertions
{
 "endpoint_id": "string",
 "endpoint_class": "string",
 "endpoint_type": "sensor|ml_model|llm|data_source",
 "content": {
  // Domain-specific content
 },
 "confidence": 0.95,
 "trust_explanation": "REQUIRED - Meaningful explanation of trust factors",
 "consumed_assertions": ["assertion_id_1"],
 "consumed_assertion_materiality": {
  "assertion_id_1": [0.8, true]
 },
 "limitations": {
  // Optional limitations
 }
}
```

Note: The framework automatically adds:

- `id`: Unique identifier
- `metadata.trust_value`: Based on trust ceiling and propagation
- `metadata.temporal_validity`: Based on data freshness
- `metadata.timestamp`: Creation time
- `metadata.provenance`: Chain of endpoint classes

A.3 Trust Configuration (Admin Responsibility)

```
POST /api/v1/trust/endpoint-class
{
 "endpoint_class": "sensor.temperature.honeywell_t7771a",
 "max_trust": 0.70
}
```

A.4 Querying Assertions (Complete with Metadata)

```
GET /api/v1/assertions/{assertion_id}

Response:
{
  "id": "550e8400-e29b-41d4-a716-446655440000",
  "endpoint_id": "temp-sensor-01",
  "endpoint_class": "sensor.temperature.honeywell_t7771a",
  "endpoint_type": "sensor",
  "content": { /* ... */ },
  "metadata": {
    "trust_value": 0.70,
    "confidence_value": 0.95,
    "trust_explanation": "Sensor calibrated 180 days ago, operating in optimal conditions",
    "temporal_validity": 0.98,
    // ... all metadata fields
  },
  "consumed_assertions": [],
  "timestamp": 1737500000.0,
  "block_hash": "0x1234..."
}
```

A.5 Compliance Monitoring Schema

```
{
  "compliance_report": {
    "endpoint_id": "string",
    "reporting_period": "ISO_8601_interval",
    "metrics": {
      "confidence_accuracy": 0.85,
      "trust_stability": 0.08,
      "assertion_completeness": 0.98,
      "trust_explanation_quality": 0.75,
      "anomaly_count": 3,
      "user_complaints": 1
    },
    "violations": [],
    "corrective_actions": []
  }
}
```

Appendix B: TERB Charter Template

B.1 Mission Statement

The Trusted Endpoint Review Board exists to ensure the integrity, reliability, and trustworthiness of endpoints within the AI Trust Chain framework through certification, monitoring, and governance.

B.2 Core Responsibilities

1. Establish and maintain technical standards for endpoint development
2. Review and certify endpoint implementations
3. Monitor compliance and enforce standards
4. Facilitate knowledge sharing among endpoint developers
5. Advise on trust chain architecture decisions
6. Ensure proper implementation of required fields (including trust_explanation)

B.3 Operating Principles

- Transparency in all decisions and processes
- Balance between innovation and reliability

- Evidence-based standard development
- Inclusive stakeholder representation
- Continuous improvement orientation
- Clear communication of trust architecture principles

References

1. AI Trust Chain Framework Documentation (https://github.com/mossrake/ai-trust-chain)
2. NIST AI Risk Management Framework
3. IEEE Standards for Trustworthy AI Systems
4. MLCommons AI Safety Taxonomy v0.5
5. Principles of Distributed Trust Systems
6. Governance Models for Technical Standards Organizations