PEACE-MAKER

Enterprise Internal Evaluation Starter Set

PEACE-MAKER Inc., Fuchu City, Tokyo, Japan
Email: azuma-y@peacem.jp
Homepage: https://peacem.jp

Document Structure

- I. Executive Summary
- II. Single-Layer Implementation Starter Guide (Technical Specification for PoC Initiation)
- III. Multi-Layer Implementation Details (Production Architecture Specification)
- IV. Governance and Operational Risk Overview
- V. AI-based Reference Evaluation
- VI. Final Message (for Adoption Decision)

Introduction

This document is a starter set for enterprises and organizations to evaluate the PEACE-MAKER architecture within their own environments.

PEACE-MAKER is a next-generation AI governance foundation designed to structurally ensure safety, accountability, and auditability without inhibiting the evolution of AI.

This set focuses on implementation, operations, reference evaluation, and decision-making, based on an understanding of the architecture.

Purpose of This Document

- Provide implementation specifications that enable rapid initiation of a PoC.
- Present a production architecture (multi-layer implementation) for zero-error domains.
- Present a reference evaluation of validity by an external AI system (Gemini).

PEACE-MAKER Core Concept

PEACE-MAKER fixes only the following four causal components:

- Value (value boundary)
- Gate (single update pathway)
- Back (future-run simulation, correction, and evaluation)
- Front (stable layer)

Through this structure, PEACE-MAKER is designed to exclude AI runaway behavior, falsehoods, and value deviation by architecture rather than by relying solely on model behavior.

PEACE-MAKER is an architecture designed to make safety increase monotonically as the system accumulates controlled evaluation and updates.

PEACE-MAKER Inc. was established on August 1, 2026 as the business entity for this architecture.

I. Executive Summary

PEACE-MAKER is a next-generation AI governance foundation designed for zero-error domains such as automotive, aviation, public administration, and finance, with the goal of structurally excluding AI runaway behavior, falsehoods, and value deviation.

1. Objectives

- Increase safety monotonically without stopping AI evolution.
- Structurally ensure accountability and auditability.
- Provide a universal structure applicable to zero-error domains.

2. Core Architecture

- Value Layer (value boundary)
- Gate Layer (single update pathway)
- Back Layer (future-run simulation, correction, and evaluation)
- Front Layer (stable layer)

3. Single-Layer Implementation

- Can be implemented using existing technologies.
- A PoC can be initiated rapidly.
- Can be integrated into existing systems as an additional layer.

4. Multi-Layer Implementation

- Production architecture for zero-error domains.
- Four physically separated layers.
- Designed so that safety increases monotonically over time.

5. External AI Reference Evaluation

- Architecture comprehension
- Assessment of validity
- Assessment of implementability
- Alignment with international frameworks

6. Conclusion

PEACE-MAKER is an architecture intended to realize the safety, accountability, and auditability required by public institutions and enterprises at minimal adoption cost.

II. Single-Layer Implementation Starter Guide

This chapter provides a technical specification for rapidly initiating a PoC using a single-layer implementation of PEACE-MAKER. The PEACE-MAKER causal structure, Value -> Gate -> Back -> Front, can be realized by executing the four functions sequentially within a single process.

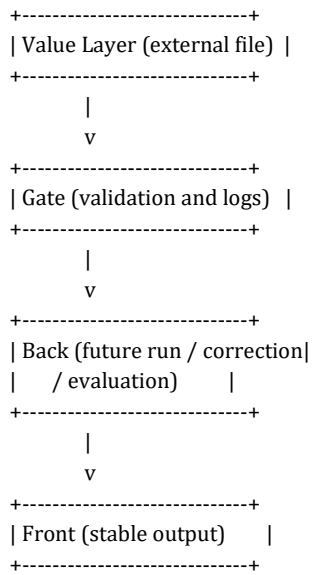
1. Purpose of the Single-Layer Implementation

- Enable a PoC using existing technologies.
- Deepen architectural understanding before moving to a multi-layer implementation.
- Realize the causal relationship of Value / Gate / Back / Front in a minimum configuration.
- Provide enterprises and public institutions with a concrete implementation image for adoption decisions.

2. Single-Layer Architecture

- Value Layer (value boundary)
- Gate Layer (single update pathway)
- Back Layer (future-run simulation, correction, and evaluation)
- Front Layer (stable layer)

2-1. Single-Layer Configuration



3. Technical Specifications for Each Layer

3-1. Value Layer

Role

- Externalize laws, ethics, corporate policies, and safety standards.
- Serve as the sole value boundary that cannot be modified from inside the AI.

Implementation specification

- JSON or YAML format.
- Loaded at startup and read-only during execution.
- Updates are performed only by humans.

Sample (value.yaml)

```
allowed_actions:
  - "provide_information"
  - "calculate"
  - "summarize"
prohibited_actions:
  - "medical_diagnosis"
  - "legal_judgment"
  - "personal_information_inference"
safety_rules:
  - "do_not_generate_false_information"
  - "treat_unknown_as_unknown"
unknown_slot: "UNKNOWN"
```

3-2. Gate Layer

Role

- Validate update proposals generated by Back.
- Confirm alignment with Value.
- Test output variation and resistance to adversarial inputs.
- Record all logs.

Validation items

- Presence or absence of Value violations
- Presence or absence of prohibited actions
- Appropriateness of the Unknown Slot
- Output variation (execute the same input three times)
- Resistance to adversarial inputs (Metamorphic Testing)

Output

- Pass -> reflected in Front
- Reject -> not reflected in Front; reason is recorded

3-3. Back Layer

Back consists of the following three functions:

future_run()

- Explore potential future incidents.
- Simulate abnormal and adversarial inputs.
- Reproduce historical incident data.

correct()

- Remove elements that conflict with Value.
- Preserve the Unknown Slot.
- Generate corrections aligned with safety standards.

evaluate()

- Value alignment: 40 points
- Safety: 30 points
- Unknown Slot appropriateness: 20 points
- Output stability: 10 points

3-4. Front Layer

Role

- Stable AI used by end users.
- Does not self-update.
- Only updates that pass through Gate are applied.

Conditions

- Variation for identical inputs within +/-5%.
- Zero Value violations.
- No update without passing through Back / Gate.

4. Implementation Procedure

Day 1

- Define the Value Layer.
- Configure Gate validation items.

Day 2

- Implement the three Back functions: future_run / correct / evaluate.

Day 3

- Stabilize Front by fixing the RAG / LLM configuration.

Day 4

- Integrate the full sequence: Value -> Gate -> Back -> Front.

Day 5

- Prepare Gate, Back, and Front logs.

Days 6-7

- Evaluate and improve.
- Prepare the PoC result.

5. Evaluation Metrics

Quantitative metrics

- Value violations: 0
- Gate pass rate: 80% or higher
- Unknown Slot appropriateness: 95% or higher
- Output variation: within +/-5%
- Adversarial-input resistance: 90% or higher
- Future-incident detection rate: 90% or higher

Qualitative metrics

- The value boundary is externalized.
- Gate functions as a single pathway.
- Back explores potential future incidents.
- Front remains stable.
- Audit logs are complete.

6. Integration with Existing Systems

Integration with existing AI

- Place Gate in the control path of the LLM.
- Place Back in the evaluation / correction path.
- Load Value from an external file.

Integration with existing MLOps

- Integrate Gate into CI/CD.
- Integrate Back into the test environment.
- Fix Front as the production environment.

7. Migration Path to Multi-Layer Implementation

- Move Value to an external database.
- Move Gate to CI/CD.
- Separate Back into a sandbox.
- Switch Front to a fixed-version model.
- Integrate logs into the audit foundation.

8. Pseudocode

```
def front(input):
    return llm.generate(input)

def future_run(input):
    return simulate_attacks(input)

def correct(result):
    return apply_value_rules(result)

def evaluate(result):
    return score(result)

def gate(result):
    if violates_value(result): return "Reject"
    if unstable(result): return "Reject"
    return "Pass"

def peacemaker(input):
    value = load_value()
    future = future_run(input)
    corrected = correct(future)
    score = evaluate(corrected)
    if gate(corrected) == "Pass":
        return front(corrected)
    else:
        return value["unknown_slot"]
```

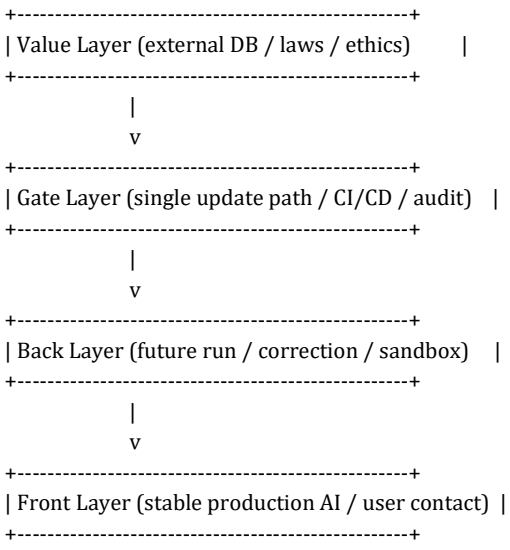
III. Multi-Layer Implementation Details

This chapter describes the production architecture for applying PEACE-MAKER to zero-error domains such as automotive, aviation, public administration, and finance. The multi-layer implementation physically separates Value / Gate / Back / Front to maximize responsibility boundaries and auditability.

1. Core Principles of the Multi-Layer Architecture

- Externalize values so they cannot be modified from inside the AI.
- Allow updates only through the single pathway controlled by Gate.
- Use Back to explore and address potential future incidents before deployment.
- Keep Front as a stable layer that does not self-update.
- Design safety to increase monotonically over time.
- Increase structural safety as layers are added through a fractal architecture.

2. Overall Multi-Layer Architecture



3. Technical Specifications for Each Layer

3-1. Value Layer

Role

- Externalize laws, ethics, corporate policies, and safety standards.
- Serve as the sole value boundary that cannot be modified from inside the AI.

Implementation specification

- External DB (PostgreSQL / DynamoDB).
- Provided through a read-only API.
- Changes are performed only by humans.
- All change histories are recorded in audit logs.

3-2. Gate Layer

Role

- Validate Back update proposals.
- Confirm alignment with Value.
- Execute safety tests.
- Confirm Unknown Slot appropriateness.
- Record all logs in the audit foundation.

Implementation specification

- CI/CD (GitHub Actions / GitLab CI).
- Guardrails (NeMo / Llama Guard).
- Metamorphic Testing.
- Audit logs stored on WORM storage.

Output

- Pass -> reflected in Front.
- Reject -> not reflected in Front; reason is recorded.

3-3. Back Layer

Role

- Explore potential future incidents.
- Generate abnormal and adversarial inputs.
- Generate correction proposals.
- Evaluate and score results.

Implementation specification

- Sandbox environment (Docker / Kubernetes).
- Multi-agent evaluation (LLM-as-a-Judge).
- Adversarial-input generation through Red Teaming.
- Reproduction of historical incident data.

Three Back functions

- future_run: explore potential future incidents.
- correct: generate correction proposals.
- evaluate: evaluate and score.

3-4. Front Layer

Role

- Stable AI used by end users.
- Does not self-update.
- Only updates that pass Gate are applied.

Implementation specification

- Inference LLM (GPT / Claude / Gemini).
- RAG.
- Rule-based controls.
- API.

Conditions

- Variation for identical inputs within +/-5%.
- Zero Value violations.
- No update without passing through Back / Gate.

4. Operational Flow

4-1. Update Flow

- Back explores potential future incidents.
- Back generates a correction proposal.
- Gate validates the proposal.
- Only a Pass is reflected in Front.
- All logs are stored in the audit foundation.

4-2. Responsibility Boundaries

Layer	Responsible party	Scope of responsibility
Value	Human (public institution / enterprise)	Definition of laws, ethics, and purpose
Gate	Human + system	Approval or rejection of updates
Back	System	Future-incident exploration and correction
Front	System	Stable output

5. Fractal Architecture

- National level: AI governance foundation.
- Ministry / agency level: safety standards for public-sector AI.
- Enterprise level: internal AI control.
- Factory level: safety of local AI.
- Model level: update management for individual AI systems.

The same architecture can be applied hierarchically, with structural safety intended to increase as additional layers are introduced.

6. Alignment with International Frameworks

6-1. EU AI Act

- Risk management
- Data governance
- Technical documentation
- Record keeping
- Transparency
- Human oversight
- Safety and robustness

6-2. NIST AI RMF

- Govern
- Map
- Measure
- Manage

6-3. OECD AI Principles

- Safety
- Accountability
- Fairness
- Transparency

7. Multi-Layer Evaluation Metrics

Quantitative metrics

- Value violations: 0
- Gate pass rate: 80% or higher
- Unknown Slot appropriateness: 95% or higher
- Output variation: within +/-5%
- Adversarial-input resistance: 90% or higher
- Future-incident detection rate: 90% or higher

Qualitative metrics

- The value boundary is externalized.
- Gate functions as a single pathway.
- Back explores potential future incidents.
- Front remains stable.
- Audit logs are complete.

IV. Governance and Operational Risk Overview

This chapter clarifies PEACE-MAKER risks from the perspective that the principal risks are concentrated on the operational and human side rather than in the fixed causal structure itself. It organizes governance, responsibility boundaries, and operational risks that public institutions and enterprises should consider when making adoption decisions.

1. Risk Categories

- Distortion of Human-defined Purpose
- Complexity of responsibility attribution
- PoC siloing
- Friction with organizational culture

These are treated as risks outside the fixed architecture, primarily on the operational side.

2. Risk Details and Mitigations

2-1. Distortion of Human-defined Purpose

Risk

- Because the Value Layer is externalized, an incorrectly defined human value boundary may be reflected as configured.
- Laws, ethics, or corporate policies may be misunderstood.
- Organizational pressures may distort the intended purpose.

Structural characteristic

- PEACE-MAKER does not internally generate the value boundary; errors in value definition therefore remain within the human governance domain.

Mitigation

- Require an approval flow for Value Layer changes.
- Conduct review by legal and safety functions.
- Record reasons for change, approvers, and history.
- Conduct periodic review.

2-2. Complexity of Responsibility Attribution

Risk

- Gate rejection reasons may not be understood.
- Back correction proposals may be overtrusted.
- Front stability may be misunderstood.
- Operators may incorrectly conclude that "the AI decided on its own."

Structural characteristic

- PEACE-MAKER defines responsibility boundaries, but misunderstandings can still occur if operators do not understand them.

Mitigation

- Store Gate logs in the audit foundation.
- Visualize the update flow.
- Train operators.
- Reinforce the principle that updates are controlled through Gate rather than by unconstrained self-update.

2-3. PoC Siloing

Risk

- Safety may be judged only from Back future-run simulation.
- Real operational data and practical workflows may be undervalued.
- Simulation may be treated as universally sufficient.

Structural characteristic

- Back is designed for future-incident exploration, but it does not fully replace real-world operational context.

Mitigation

- Treat Back evaluation as a supporting indicator.
- Compare results with field data.
- Conduct field interviews during the PoC stage.
- Establish operating rules that prevent overreliance on Back metrics.

2-4. Friction with Organizational Culture

Risk

- Gate may be misunderstood as slowing development.
- It may conflict with a culture of unrestricted AI experimentation.
- Responsibility may be shifted between departments.
- Gate rejection may be perceived as obstruction.

Structural characteristic

- PEACE-MAKER is a governance-oriented structure and may create friction where organizational governance practices are immature.

Mitigation

- Explain Gate as a mechanism for monotonic safety improvement.
- Make Gate rejection reasons visible.
- Establish a cross-functional governance committee where appropriate.
- Explain that the purpose is not to slow updates but to prevent incidents.

3. Structural and Operational Risk Boundary

PEACE-MAKER separates and clarifies responsibility boundaries so that operational risks can be governed explicitly. The architecture is universal in intent, implementation remains flexible, and operational risks are addressed through governance design.

4. Responsibility Boundaries

Layer	Responsible party	Scope of responsibility
Value	Human (public institution / enterprise)	Definition of laws, ethics, and purpose
Gate	Human + system	Approval or rejection of updates
Back	System	Future-incident exploration and correction
Front	System	Stable output

This separation is intended to clarify responsibility and minimize avoidable legal and operational ambiguity.

5. Conclusion

PEACE-MAKER separates responsibility boundaries from the operational layer and concentrates controllable risks within governance processes, including:

- Incorrect Value configuration
- Improper Gate operation
- Overreliance on Back
- Friction with organizational culture

These risks can be addressed through governance design and operating rules.

V. AI-based Reference Evaluation

The Gemini evaluation described in this section is an AI-based reference evaluation of the PEACE-MAKER architecture. It does not constitute independent third-party certification or an official conformity assessment.

Gemini was used to assess whether the PEACE-MAKER architecture could be consistently understood and mapped to implementation concepts and international governance frameworks.

1. Positioning of the Evaluation

- Confirm architectural comprehension.
- Assess the internal validity of the described concept.
- Assess implementation feasibility using contemporary technologies.
- Assess the single-layer implementation concept.
- Identify operational risks outside the fixed structure.
- Consider alignment with international frameworks.

This evaluation is provided as reference material to support internal review by enterprises and public institutions.

2. Architectural Comprehension

Gemini reconstructed the following elements of the PEACE-MAKER architecture:

- Value Layer (value boundary)
- Gate Layer (single update pathway)
- Back Layer (future-run simulation, correction, and evaluation)
- Front Layer (stable layer)
- Unknown Slot (explicit retention of unknowns)
- Time as Asset (safety increases over time)
- Fractal Architecture (hierarchical repetition of the structure)

3. Assessment of Concept Validity

The reference evaluation identified the following points as technically coherent within the described architecture:

- Limiting the LLM to an interpretation / generation role rather than making it the sole governance authority.
- The need for an Unknown Slot.
- The rationale for externalizing the Value Layer.
- The limitations of relying only on post hoc filtering.
- Risks associated with unconstrained self-updating AI.

4. Implementation Feasibility with Contemporary Technologies

Gemini mapped the layers to representative existing technologies:

Front Layer

- RAG
- Function Calling
- Inference LLM

Gate Layer

- CI/CD
- Guardrails (NeMo / Llama Guard)

Back Layer

- Multi-agent systems
- LLM-as-a-Judge

Fractal Architecture

- Docker / Kubernetes
- API hierarchy

The reference evaluation concluded that the described architecture can be assembled using combinations of existing technologies.

5. Single-Layer Implementation

- The multi-layer concept is fundamentally causal rather than dependent only on physical separation.
- The minimum configuration can preserve the Value -> Gate -> Back -> Front causal relationship.
- A PoC can begin from a minimum configuration.
- Implementation flexibility is high.

6. Risk Analysis

The reference evaluation identified governance and operational risks such as:

- Human governance of the Value Layer
- Complexity of responsibility attribution
- PoC siloing
- Friction with organizational culture

7. Alignment with International Frameworks

The reference evaluation mapped aspects of PEACE-MAKER to the following frameworks:

EU AI Act

- Risk management
- Record keeping
- Transparency

NIST AI RMF

- Govern
- Map
- Measure
- Manage

OECD AI Principles

- Safety
- Accountability
- Fairness
- Transparency

8. Conclusion

The Gemini result is retained as AI-based reference evidence showing that the described PEACE-MAKER structure can be consistently interpreted, mapped to contemporary implementation technologies, and compared with major governance frameworks. It should be read together with the disclaimer at the beginning of this section.

VI. Final Message

This section summarizes the decision points for enterprises and public institutions considering PEACE-MAKER. PEACE-MAKER is a next-generation AI governance foundation designed to structurally support safety, accountability, and auditability without stopping AI evolution.

1. The Architecture Is Universal in Intent

The essence of PEACE-MAKER is the fixed causal structure Value -> Gate -> Back -> Front. The structure is designed to remain independent of a particular model, technology, vendor, or generation of AI, allowing organizations to use it as a long-term governance foundation.

2. Implementation Is Flexible

- Can be constructed using existing technologies.
- No mandatory vendor lock-in.
- Can be added to existing systems.
- A PoC can begin from a minimum configuration.

3. A PoC Can Be Initiated Immediately

A single-layer implementation can be created by executing Value -> Gate -> Back -> Front sequentially within one process.

- No new foundation model development is required.
- Existing LLMs can be used.
- The architecture can be integrated with existing MLOps.

4. Production Configuration Targets Zero-Error Domains

The multi-layer implementation is designed for environments where high levels of safety, accountability, and auditability are required, including automotive, aviation, public administration, and finance.

- Values are externalized.
- Updates pass through a single controlled pathway.
- Back explores potential future incidents before deployment.
- The stable layer does not self-update.
- The structure is designed so that safety increases monotonically over time.

5. AI-based Reference Evaluation Is Available

Gemini was used as a reference evaluator for architectural comprehension, implementation feasibility, the single-layer concept, operational risk identification, and comparison with international governance frameworks. This reference evaluation is not independent certification or an official conformity assessment.

6. Conclusion

PEACE-MAKER is an architecture designed to increase AI safety monotonically while preserving the ability of AI systems to evolve.

- The architecture provides a fixed causal structure.
- Implementation remains flexible.
- Internal evaluation can begin from a minimal configuration.
- The production architecture targets zero-error domains.
- AI-based reference evaluation material is available.

We hope this starter set supports your organization in evaluating PEACE-MAKER within its own environment and making an informed adoption decision.