

1. Problem Analysis: Technical Overview

1.1 Summary

This code section is a problem evaluation framework which represents a paradigm shift beyond mere traditional risk assessment, integrating AI-powered analysis with advanced systems thinking to provide comprehensive problem characterization and intervention strategy development. The architecture combines statistical reliability with contextual intelligence to deliver actionable insights for complex problem resolution.

1.2 Architecture and Orchestration

1.2.1 System Orchestration Framework

This section starts with `problem_pipe.py`, which serves as the comprehensive workflow coordinator, managing sequential analysis execution and extracting critical performance metrics from distributed analysis components. This orchestrator currently operates defaulting to Context 1 (a "standard implementation") configuration while maintaining architecture provisions for future dynamic selection capabilities, depending of in which context the problem is taking place. This is a factor that affects the analyses that take place afterwards, introducing leanings towards certain benchmarks depending on the context in which the problem takes place (**more about this below, in point X**).

```
problem_pipe.py → llama_problem.py → problem_dimensions.py →  
problem_solved.py
```

The system generates structured outputs in both JSON and TXT formats, capturing several essential synthetic metrics generated during the analysis, including certain proprietary Risk-Reward-Certainty Ratios (RRCR), and comprehensive dimensional assessments that form the foundation for the strategic decision-making process and the cost-benefit analysis of the future stages.

1.2.2 Dual-Engine Analysis Architecture

At the analytical core lies a dual-pathway evaluation system implemented through `llama_problem.py`, which processes problem statements across relevant aspects of the problem, using complementary methodologies that ensure comprehensive coverage and analytical robustness.

Archetype Classification Engine categorizes recurrent problems as established patterns in a dictionary (proprietary). These providing contextual framework for dimensional interpretation and establishing baseline expectations for solution complexity.

Dimensional Assessment evaluates problems across specialized dimensions (likewise, a proprietary feature logic). Each relevant dimension identified by the code receives comprehensive valence with an associated confidence of analysis.

The integration methodology employs a calibrated dynamic weighting scheme between both evaluations. It is enhanced with median-based outlier removal and multi-run averaging several iterations, alongside with several consensus validation constraints.

1.2.3 Reliability Enhancement System

The `llama_problem_reliable.py` module implements a variant statistical reliability measure through extended iterated runs of the LLM alongside with consensus validation requirements. The system demands 70% or greater agreement across analysis runs within certain ranges of tolerance (removes wild outliers), ensuring an increased consistency and confidence.

Result Caching Architecture utilizes problem statement hashing for efficient reanalysis avoidance, while **confidence scoring algorithms** calculate statistical confidence metrics based on inter-run variance analysis. The system incorporates intelligent early stopping protocols that terminate analysis when confidence exceeds 90% threshold, optimizing computational resource utilization.

1.3 Dimensional Processing Framework

1.3.1 Dynamic Correlation Management

The `problem_dimensions.py` component orchestrates a comprehensive dimensional processing engine incorporating multiple specialized analysis modules that model complex system interactions and dependencies.

Correlation Matrix Engine (`problem_corels.py`) manages dynamic relationships between all the problem dimensions using correlation coefficients. The system implements context-aware correlation adjustments that adapt based on problem scenarios—crisis contexts, which strengthen the sensitivity of specific correlations between problem dimensions when appropriate.

Interaction Weight Matrix (`problem_interactions.py`) defines a comprehensive directional influence matrix. This enables a sophisticated cascade modeling where changes in individual problem dimensions propagate through the system according to established influence patterns.

1.3.2 System Dynamics Analysis

Critical Threshold Analysis (`problem_thresholds.py`) identifies critical tipping points where dimensional relationships undergo significant behavioral changes, enabling detection of phase transitions and systemic instability indicators that signal the appropriated intervention requirements.

Dimension Clustering Engine (`problem_clusters.py`) employs statistical clustering methodologies based on correlation patterns to identify the problem dimension groupings that exhibit coordinated movement, revealing systemic relationships and co-dependencies critical for intervention planning.

1.4 Risk Assessment and Synthesis Engine

1.4.1 Analytics Framework

The `problem_solved.py` module represents the analytical culmination, performing sophisticated risk assessment and value analysis through multiple integrated evaluation methodologies.

Primary Analysis Engine conducts dimension fit analysis by comparing problem versus solution scores across all dimensions, tracking movement patterns and calculating improvement or deterioration metrics. The system implements a sophisticated weighted value change formula that combines direct fix effects with synergy interactions, providing comprehensive improvement assessment.

2nd Order Effects Modeling represents a breakthrough capability that calculates leverage coefficients indicating how much each dimension influences others, vulnerability assessments showing dimensional susceptibility to external changes, and net influence calculations balancing leverage against vulnerability. This cascade analysis identifies dimensions most likely to trigger system-wide chain reactions.

1.4.2 Derived Factor Analysis System

The system generates over a dozen composite factors from the dimensions and the archetype combinations. These factors themselves combine into meaningful strategic indicators that support, suggest and lean towards certain executive decisions and resource allocations for the problem.

Sensitivity Analysis Protocols test system responses to sudden changes, identifying intervention leverage points and stability thresholds critical both for successful problem resolution strategies and unexpected circumstances that might play out.

1.4.3 Multi-Layer Risk Assessment

The comprehensive risk assessment engine produces sophisticated multi-layer evaluations including Systemic Risk Scores, Overall Risk Assessments, and Risk Potential Indicators. The **Risk-Reward-Certainty Ratio (RRCR)** provides balanced comparison of perceived benefits against associated risks plus an estimate of how much certainty these analyses entail, incorporating context-based adjustments derived from archetype patterns and problem dimensions distributions.

The system applies intelligent bonuses and penalties based on score distribution patterns, ensuring realistic risk assessment that accounts for both statistical patterns and contextual considerations.

1.5 Technical Implementation and Configuration

1.5.1 AI Engine Specifications

The system leverages **Llama 3.2** through local Ollama implementation (for data privacy concerns) with precision-optimized parameters including ultra-low temperature settings (0.01) for maximum analytical consistency, extended token limits (8192) for comprehensive analysis depth, and sophisticated error handling protocols.

1.5.2 Context-Adaptive Intelligence

The framework supports five specialized problem contexts with dynamic adaptation capabilities:

- **Standard Context:** Baseline analysis without special adjustments
- **Crisis Context:** Strengthens time sensitivity and risk correlations for emergency scenarios
- **Innovation Context:** Emphasizes novelty dimensions while reducing certainty impact requirements
- **Policy Context:** Prioritizes stakeholder impact and ethical considerations for governmental scenarios
- **Technical Context:** Emphasizes knowledge requirements and structural considerations for engineering