TriageCounsel Technical Executive Brief

Deterministic Legal Reasoning. AI Explanations.

Updated 2026-08-02 | Ruleset v7.2.0 | Production build verified live at triagecounsel.com

Executive Summary

TriageCounsel is a contract intelligence platform built around a deterministic legal reasoning engine. Unlike conventional AI-first contract review tools, legal risk determinations are made through explicit, auditable rules. The LLM is deliberately placed outside the decision path and is used only to explain finalized findings in natural language. Since the prior brief, the ruleset has grown to 189 deterministic rules, six new deterministic analysis layers have shipped (clause-quality scoring, risk dashboards, negotiation reasoning, and more), the full engine has been validated against 4,190 real SEC-filed contracts with a GO verdict, and a production-blocking deployment defect has been found and fixed, with the fix confirmed live.

CORE PRINCIPLE	**Deterministic rules decide. AI explains.**

At a Glance

189	**813**	**4,190 / 4,190**	**100%**
Deterministic Rules (v7.2.0)	Automated Tests Passing	SEC Contracts Benchmarked	DOCX / Export Reliability
GO	**97 / 100**	**0**	**LIVE**
Engine Benchmark Verdict	Commercial Readiness Score	Critical Blockers	Confirmed in Production

The Problem

Most legal AI systems ask a probabilistic model to identify, classify, and explain legal risk in a single step. That design can produce inconsistent outputs and makes it difficult to reproduce, audit, trace, or defend a finding in a high-stakes legal workflow.

Architecture and Trust Boundary

TriageCounsel separates deterministic analysis from probabilistic language generation. Parsing, normalization, rule execution, clause-level evidence, clause-quality scoring, cross-finding consistency checks, suppression,

deduplication, scoring, redline authoring, and audit-trail creation are completed before the LLM is invoked. The LLM receives structured findings and produces a human-readable explanation; it does not create or alter the underlying legal determination, and it never authors redline language.

Deterministic layer	Explanation layer
• Explicit policy rules (189) • Clause-level evidence • Clause-quality & risk scoring • Reproducible outputs • Consistency validation • Rule-mapped redline templates • Rule trace and audit trail	• Plain-English summaries • Natural-language explanations • Human-review support • No replacement of legal judgment • No authority to change findings • Never authors redline text

Research Foundation

The architecture is based on the peer-reviewed ICCS 2026 (Springer LNCS) paper *Deterministic Execution Frameworks for Hybrid Symbolic-Probabilistic Computational Pipelines*. The research demonstrates a hybrid design in which deterministic execution preserves reproducibility while the LLM is constrained to explanation.

Current Scope

Commercial NDAs, MSAs, SaaS agreements, vendor contracts, employment & contractor agreements, consumer and creator/marketplace terms, commercial leases, loans/promissory notes/guaranties, franchise & distribution agreements, M&A/partnership agreements, settlement agreements, construction contracts, real estate purchase & sale, insurance policies, lending/financial-services compliance, government contracting, healthcare provider agreements, and IP licensing — 22 practice-area scopes in total.

TriageCounsel | Deterministic Legal AI

New Since the Last Brief: Deterministic Analysis Layers

Six additive deterministic layers have shipped on top of the core rule engine — each is independent of, and never reduces the authority of, the presence/absence rule findings.

Layer	What it does
Clause Quality Engine	Deterministic 0-100 completeness scoring for six clause types (Arbitration, Liability, Confidentiality, Indemnification, Termination, IP) — independent of and additive to presence/absence rules.
Three-Score Risk Dashboard	Legal Risk / Business Risk / Negotiation Difficulty, each computed from existing finding data, no new detection.
Risk Allocation & Balance Score	Of the one-sided clauses the engine can classify directionally, what percentage favor the counterparty vs. the reviewing party.
Lawyer Confidence Index	Restructures per-finding confidence into an explicit, checkable breakdown instead of a bare percentage.
Negotiation Reasoning Chain	For deficient clause elements: Issue → Legal Risk → Business Risk → Market Standard → Suggested Redline → Expected Counterparty Objection → Fallback Position.
Defined-Terms & Cross-Reference Checker	Flags undefined referenced terms and broken internal cross-references.

Redlines Became Real Word Track Changes

Every rule_id now maps to exactly one reviewed default redline — the LLM may explain a redline, it never authors the language. Accepted/edited redlines are written as native OOXML Track Changes (*w:ins*/*w:del*), each with a Word comment citing the rule ID and legal rationale; rejected findings get a "reviewed and declined" comment quoting the attorney's own reasoning. Track-changes authorship is the reviewing attorney; explanatory comments are attributed to the deterministic engine, keeping AI-assistance fully visible without the edit itself appearing AI-authored. A unified Review Workflow now keys every decision by finding occurrence (not bare rule ID), fixing a defect where deciding on one occurrence of a repeated rule silently applied that decision to every other occurrence of the same rule in the document. Decisions feed a Deterministic Replay and a Negotiation Package export, and the PDF/DOCX exporters were wired to carry every one of these layers through to the downloaded document.

Ruleset Growth

Severity	Rule count
Critical	9
High	42
Medium	109
Low	29

Severity	Rule count
Total (v7.2.0)	189

v7.2 added four rules closing a gap with no prior coverage: AI-generated output ownership left unaddressed, no human-review requirement for AI-generated output, no recognized security certification (SOC 2 / ISO 27001 / PCI-DSS) despite discussing security controls, and an SLA that never states a response time. Every new rule in the current ruleset — this release and prior ones — is scored against an 11-factor severity framework and passed through an automated severity gate before shipping; hand-set severities are never asserted independently of that framework.

TriageCounsel | Deterministic Legal AI

Severity Scoring Model

Every finding's CRITICAL / HIGH / MEDIUM / LOW severity is itself computed deterministically — never hand-picked. All 189 rules carry an explicit, individually-justified 11-factor vector; a rule's severity is a pure function of that vector, recomputed by CI on every change and never independently overridden.

11 intrinsic factors, three harm tiers. A factor qualifies only if it is determinable from the clause's own text, its party-direction grammar, and general legal doctrine — never from the specific parties' identities, governing law, deal size, or a probability estimate.

Tier (weight)	Factors
A — Fundamental rights & personal harm (×4)	PE Personal Exposure to a Natural Person · RW Waiver of Forum/Process Rights · CR Penal Exposure
B — Structural financial & legal exposure (×2)	FB Financial Boundedness & Certainty · RS Named Regulatory/Statutory Exposure · AT Ownership/Title Transfer · UD Unilateral Discretion Over a Fundamental Term
C — Operational & temporal exposure (×1)	REV Legal Reversibility of Harm · SC Structural Breadth of Exposed Persons · OC Notice/Cure Absence · DUR Temporal Boundedness

Scoring algorithm. Eight ceiling rules are checked first, in order; the first match wins and sets the tier directly — e.g. an uncapped natural-person guaranty (PE = 3) or a confession of judgment (RW = 3) is CRITICAL on that fact alone. If no ceiling fires, a Weighted Aggregate Score is computed and mapped to a tier:

$$WAS = (PE + RW + CR) \times 4 \;+\; (FB + RS + AT + UD) \times 2 \;+\; (REV + SC + OC + DUR) \times 1$$

Hard invariant: CRITICAL is reachable only through a ceiling rule, never through an aggregate score alone — so every CRITICAL finding has a one-sentence, citable reason ("fired because PE = 3"), never an opaque high weighted sum. Below the ceiling, v1.1 (current default) bands each rule's WAS against that rule's own *practical_max* — the WAS it would reach if every factor it structurally touches were maxed out — at 35% / 70% cut points, rather than a single fixed score across every clause type. This replaced an earlier fixed-threshold model after a family-clustering study found the fixed cutoff produced genuine differentiation in only 2 of 18 practice-area families (16 were either fully saturated or fully locked out, because a clause type's achievable score range is set by how many factors it structurally touches, not by its drafting severity); the relative model raised that to 16 of 18 families.

Deliberately excluded from scoring

- Likelihood of litigation/enforcement
- Bargaining power / negotiation leverage
- Estimated dollar damages (FB is the bounded/unbounded proxy)
- Deal size or commercial stakes
- Governing-law enforceability (handled separately, below)

Integrity controls (CI-enforced)

- Recomputed tier must equal stored tier or the build fails
- Monotonicity across rules sharing a clause category
- Mutuality gate blocks UD scoring on symmetric rights
- Blind second-scorer re-score gate on every new rule
- Frozen golden-set regression on every scoring-function change

Jurisdiction is layered on top, never baked in. Intrinsic severity is computed identically regardless of governing law (an identical non-compete clause is banned in one state and standard in another; that fact lives in a separate jurisdiction-modifier table keyed by clause category, not inside the severity score). A jurisdiction modifier may adjust enforceability and apply a bounded ±1-tier adjustment, but it can never downgrade a finding whose severity was set by a ceiling rule — a confession of judgment stays CRITICAL even where it is unenforceable, because the drafting itself is still the defect.

TriageCounsel | Deterministic Legal AI

Full-Corpus Engine Validation: TriageBench

TriageBench runs the complete ten-stage deterministic pipeline (Upload → Analysis → Rule Engine → Evidence → Review → Verify → Negotiation Package → DOCX Export → Replay → Audit) against every contract in a 4,190-document corpus of real SEC-filed exhibits (Employment, Purchases, Lease, Services), independent of the FastAPI/database/Stripe application surface and with no dependency on the non-deterministic LLM layer. The most recent full-corpus run is the reference baseline:

Metric	Result
Contracts scheduled / accounted for	4,190 / 4,190 (100%)
Parser success rate (no P0/P1 defects)	100.0%
Verify / replay success rate	100.0%
DOCX export success rate	100.0%
Negotiation Package success rate	100.0%
Critical blockers	0
Engine reliability score	99 / 100
Parser reliability score	100 / 100
Determinism score	100 / 100
Evidence-anchoring score	100 / 100
DOCX / export reliability score	100 / 100
Commercial readiness score	97 / 100
Go / No-Go verdict	GO

This baseline run followed a fix to a real crash TriageBench itself discovered: a legacy page-break artifact (a form-feed character) in one SEC filing's extracted text was illegal under the XML 1.0 character spec and crashed DOCX export. The fix implements the exact XML 1.0 *Char* production (verified character-by-character against lxml's own acceptance behavior) and is applied everywhere raw contract text enters a generated document — covered by 15 new targeted tests plus independent cross-validation against Mammoth, a third-party OOXML parser, confirming exported documents remain valid and human-meaning is preserved, not silently truncated or hidden.

The only open item from this run is 59 contracts (1.41%) flagged by an independent, secondary HTML-text-extraction cross-check used purely to sanity-check corpus parsing consistency — not a rule-engine, DOCX-export, or application defect, non-blocking, and documented as a known external-corpus characteristic for engineering follow-up.

Live Production Validation: TriageBench Live

A separate suite validates the deployed application exactly as a customer would — real signup, real session cookies, real file upload through the real form, real page loads and downloads, real clicks in a real Chromium

browser. It never imports an engine or application module; that boundary is enforced automatically by an AST-based test, not just by convention. Coverage: the full 11-step customer workflow, security and failure-injection probes, concurrency load, real-browser regression, and a dedicated database-session-lifecycle proof suite.

TriageBench Live's concurrency suite surfaced a real production defect: 59 route and exception-handler call sites obtained a database session via *next(get_db())* outside FastAPI's dependency injection, so nothing guaranteed the session closed — under sustained traffic the connection pool exhausted permanently and every request hung for the full pool timeout. The fix converts every call site to *Depends(get_db)* (closed automatically by FastAPI's own request exit-stack) with an explicit context-manager form for the two exception handlers Starlette invokes directly, and adds SQLite WAL mode with an explicit busy timeout. A targeted proof suite exercises sequential, at-capacity, and above-capacity concurrent bursts specifically to distinguish a genuine leak (permanent, never-recovering) from ordinary capacity backpressure (bounded, self-recovering) — the fixed build shows zero leakage under both sequential and concurrent sustained load.

Automated Test Suite

813 / 813	13 / 13	30 / 30
Core application tests passing	TriageBench framework tests passing	TriageBench Live framework tests passing

TriageCounsel | Deterministic Legal AI

Production Deployment: Found, Fixed, Confirmed Live

A full audit of the live deployment found that the benchmark corpus used by TriageBench had been committed to the repository without an exclusion rule, so it was being bundled directly into the deployed serverless function — pushing the build to 795 MB against a 500 MB platform ceiling. Every deployment since that commit landed had been silently failing to build, meaning production had been serving a stale, pre-benchmark build the entire time this round of fixes was developed. The exclusion rule was corrected, the build was confirmed to complete successfully, and the resulting production deployment was independently verified: the live domain returns HTTP 200, the homepage renders, and the registration and login flows both respond correctly.

Check	Result
Production build status	READY (was failing to build)
triagecounsel.com homepage	HTTP 200 — verified
Registration flow (/register)	HTTP 200 — verified
Login flow (/login)	HTTP 200 — verified
Session-lifecycle fix deployed	Confirmed live
DOCX export fix deployed	Confirmed live

Execution Pipeline

The production pipeline below shows the operational separation between legal reasoning and language generation. In the current build, all 189 deterministic rules are evaluated before the explanation layer is called.

- ✓ **Parser** — DONE
  Document text extracted and normalized
- ✓ **Normalizer** — DONE
  Text cleaned and structured for analysis
- ✓ **Deterministic Rule Engine** — DONE
  189 rules executed against document
- ✓ **Evidence Engine** — DONE
  Clause-level evidence anchored to findings
- ✓ **Clause Quality & Risk Scoring** — DONE
  Completeness, balance, and risk-dashboard scores computed
- ✓ **Consistency Engine** — DONE
  Cross-finding validation completed
- ✓ **Workflow Engine** — DONE
  Suppression, deduplication, and per-occurrence review decisions applied
- ✓ **Redline & Export Engine** — DONE
  Rule-mapped redlines, Track Changes, comments, and Negotiation Package assembled
- ✓ **LLM Explanation Layer** — DONE
  Findings explained for human review

✓ **Verification Report Generated** DONE
Report assembled with full audit trail

All findings are fixed before the LLM explanation layer is invoked. The verification report combines deterministic findings, clause evidence, clause-quality and risk scores, rule trace, consistency checks, redline/Track Changes state, and the resulting natural-language explanation.

Why This Architecture Matters

The architecture is designed for environments where a user must know not only what the system concluded, but also which rule fired, what contractual evidence supported it, whether related findings were consistent, and how the final report was assembled. This moves the LLM from decision-maker to constrained interface.

Design Principles

Determinism: same inputs and rules produce the same findings. **Auditability:** every finding is tied to evidence and an execution trace. **Constrained AI:** the model explains finalized outputs and never authors redline language; it does not decide. **Human review:** reports support professional judgment rather than replace it. **Verified in production:** claims in this brief are backed by a passing test suite, a full-corpus engine benchmark, and a confirmed-live deployment — not by design intent alone.

Santhosh Guntupalli | Founder, TriageCounsel | triagecounsel.com | santhoshguntupalli06@gmail.com