INTRODUCING SIGN

Sigil Intelligence Graph Notation

The Knowledge Contract Layer for the Agentic Enterprise

This whitepaper introduces SIGN — Sigil Intelligence Graph Notation — a purpose-built notation language that enables AI agents to read, reason over, and respect enterprise knowledge with full governance, auditability, and token efficiency. We describe the problem SIGN solves, its design principles and specification, its application across three enterprise domains, and the case for an open standard as the foundation for the agentic enterprise.

SIGN is the Rosetta Stone of Agentic Knowledge — unlocking the institutional knowledge that organizations already possess, in a form every agent can read, apply, and be held accountable to.

Authors: Dr. Joe Shepherd · April 2026

Version: 1.0 · First publication

ABSTRACT

Abstract

Enterprise AI agents are rapidly moving from experimental to operational. Yet the knowledge they act on — business rules, governance constraints, domain definitions, inference logic — remains locked in forms optimized for human readers: policy documents, spreadsheets, prompt instructions, and configuration files. None of these were designed for an agent context window as the primary consumer.

The result is a structural gap at the heart of enterprise AI deployment. Agents that cannot reason over governed knowledge cannot be trusted with consequential decisions. Enterprises that cannot audit why an agent made a decision cannot satisfy regulators, boards, or their own governance standards.

SIGN — Sigil Intelligence Graph Notation — is a notation language designed to close this gap. It is the contract layer between what an organization knows and what its agents are permitted to do with that knowledge. SIGN expresses structure, relationships, inference rules, constraints, clusters, and provenance in a form agents can reason over — not just retrieve from — while remaining human-readable and token-efficient.

This paper introduces SIGN v1.0, describes its design principles and specification, demonstrates its application across three enterprise domains — healthcare compliance, financial services risk policy, and brand governance — and makes the case for SIGN as an open standard for governed agent knowledge. SIGN was originated and validated at Career Highways in the context of a large-scale enterprise workforce intelligence platform. The specification is available at github.com/sign-lang.

SECTION 1

1. The Enterprise Agent Knowledge Problem

Every team that has deployed AI agents into production enterprise workflows eventually encounters the same wall. The models are capable. The infrastructure is available. The use cases are clear. But the knowledge the agents need to operate correctly — the rules, constraints, definitions, and governance logic that govern acceptable decisions — is not available to them in any form they can reliably reason over.

This is not an intelligence problem. It is a knowledge representation problem. The knowledge exists in the organization. It has been documented, reviewed, and maintained by domain experts over years or decades. But it exists in forms optimized for human readers, not for agent consumers. The gap between how organizations store knowledge and how agents need to consume it is the central unsolved problem in enterprise AI deployment today.

1.1 How Organizations Store Knowledge Today

Enterprise knowledge — the rules, policies, definitions, and logic that govern how the organization operates — exists in three predominant states. Each is problematic for agent consumption in different ways.

Storage form	Why agents cannot reliably consume it
Policy and reference documents (PDF, Word, SharePoint)	Dense prose optimized for human comprehension. Agents can retrieve passages but cannot reason over rules, enforce constraints, or apply inference logic encoded in narrative form. No versioning, governance, or auditability at the rule level.
Structured data systems (HRIS, ERP, LMS, CRM)	Machine-readable but semantically opaque to agents. Agents can query data but cannot access the business logic, governance rules, or domain definitions that give the data meaning. The 'why' behind the data is inaccessible.
Human expertise and institutional memory	The richest and most accurate form — and entirely inaccessible to agents. Senior practitioners hold nuanced judgment, exception logic, and contextual rules that exist nowhere else. This knowledge evaporates when people leave.
System prompts and inline instructions	Fast to write but ungoverned by design. Rules embedded in prompts are unversioned, unauditable, non-reusable across agents, and invisible to any governance framework. Break silently when context changes.

1.2 The Approaches That Fall Short

Approach	What it solves	What it doesn't solve
Retrieval-Augmented Generation (RAG)	Gives agents access to large document corpora without context window limits.	Does not structure knowledge for reasoning. Agents retrieve relevant passages but cannot enforce constraints, apply inference rules, or reason over relationships. Auditability is at the document level, not the rule level.
JSON / structured data	Machine-readable with defined schema. Parseable by code.	Agent-hostile. 50–60% of tokens are structural noise. Designed for parsers, not attention mechanisms. No relationship semantics, no inference model, no constraint expression.
RDF / OWL / semantic web	Formally correct semantic model with rich relationship types and inference support.	Failed the adoption test. Academic syntax with heavy toolchain requirements. Near-zero presence in LLM pretraining data. Agents cannot parse Turtle or Manchester syntax without domain-specific fine-tuning.
Prompt engineering	Immediate, accessible, requires no infrastructure.	Fundamentally ungoverned. No versioning, no auditability, no reuse across agents. Rules embedded in prompts cannot be audited, cannot trigger breaking-change governance, and cannot be maintained at corpus scale.

The design-target problem: Every existing approach was designed for a different primary consumer — a machine parser, a human reader, an academic reasoner, or a single-use prompt. None were designed for the LLM agent context window as the unit of consumption. SIGN starts from that constraint and works backward to the format.

1.3 What Governed Agent Knowledge Requires

Requirement	Rationale
Token efficiency	Every token in an agent context window has a cost — financial, latency, and quality. A format that wastes tokens on structural noise limits the scope of knowledge accessible to an agent or inflates operational cost at scale.
Structural expressiveness	Agents need more than definitions. They need rules they can apply, constraints they must enforce, inference logic they can execute, and relationships they can traverse. A format limited to key-value structure cannot carry this semantic load.
Governance primitives	Enterprise knowledge is versioned, reviewed, approved, and updated on defined cycles. The format must carry governance metadata as first-class structure — not as optional annotations on top of content.
Breaking change management	When a rule changes, every agent that depends on that rule needs to be re-validated. The format must signal breaking changes so dependent systems can respond. Silently propagating breaking changes to production agents is unacceptable.
Auditability at the rule level	Governance requires knowing which specific rule, constraint, or inference the agent applied, in what version, and with what provenance. The format must support this granularity — document-level attribution is insufficient.
Domain agnosticism	The same format must be capable of expressing workforce intelligence, regulatory compliance, brand governance, financial risk policy, and any other domain without modification.
Human authorship with agent consumption	Knowledge is maintained by humans. It must be readable, authorable, and reviewable without specialized tooling. The compilation step produces the agent-optimized form; the source must remain accessible to domain experts.

SECTION 2

2. Introducing SIGN

SIGN — Sigil Intelligence Graph Notation — is a strongly typed, human-readable, token-optimized notation language designed for the agentic consumer. It is the contract layer between a governed knowledge corpus and the AI agents that consume it.

SIGN is not a general-purpose serialization format and is not competing with JSON. JSON is machine-readable with parser overhead designed for programmatic consumption. SIGN is designed specifically around how agents read: by pattern recognition, sigil semantics, and attention over a token budget. Every design decision follows from that constraint.

2.1 The Core Design Insight

LLM agents parse by pattern recognition and semantic attention. A sigil like @constraints or @infer carries meaning that agents recognize immediately from pretraining patterns on structured documents. A JSON key like “structural_rules” carries the same meaning but costs more tokens and relies on the agent correctly inferring intent from an arbitrary prose label.

The sigil system is the core innovation of SIGN. Each sigil is a short @-prefixed token that declares the type and purpose of the block that follows. The sigil system is self-describing — an agent encountering SIGN for the first time can infer complete structural meaning from sigils alone. No schema document required in context.

2.2 The Sigil Vocabulary

Sigil	Group	Purpose
@canon	Document structure	Bundle header — version and SHA-256 integrity hash of the compiled artifact
@doc	Document structure	Document identity — type token, status, audience scope, related-document edges
##	Document structure	Agent summary line — one per document in the always-injected index layer
@links	Document structure	Related document graph edges — builds the navigable knowledge graph
@def	Content	Primary canonical definition of the subject concept
@props	Content	Named core properties using + prefix
@include	Content	Positive inclusion criteria using ? prefix
@exclude	Content	Negative exclusion criteria using ! prefix
@rules	Content	Numbered operational or classification rules
@boundary	Content	Identity and scope boundary — prevents semantic overlap with adjacent concepts
@constraints	Governance	Structural rules agents must enforce: mutex, requires, forbids
@infer	Governance	Governed inference rules — derive new facts from existing knowledge state
@cluster	Governance	Named typed sets with threshold semantics — first-class knowledge graph nodes
@attrs	Governance	Required and optional attribute declarations with type information
@rel	Relationships	Schema-level relationship type declarations
@edges	Relationships	Instance-level graph edges with typed predicates and edge properties
@xwalk	Relationships	Crosswalk identity resolution — maps external or legacy identifiers to canonical IDs
@vocab	Relationships	Governed definitions of every relationship predicate used in the document
@status	Lifecycle	Current-state key-value snapshot for operational reference documents
@phases	Lifecycle	Lifecycle phase plan with [done]/[next]/[later] tokens

Sigil	Group	Purpose
@cycle / @ reviewed	Lifecycle	Review cadence and last-reviewed date — governance scheduling primitives
@reserved	Extension	Forward roadmap sigil declarations — governs the extension namespace

2.3 Relationship Operators

Operator	Semantics
->	Outbound directed edge: subject → predicate → target.
<-	Inbound directed edge: inverted for authoring readability.
<=>	Symmetric bidirectional edge: valid only for inherently symmetric predicates such as adjacent_to.
=>	Crosswalk resolution: source term maps to canonical target, with optional confidence and method annotations.

2.4 Token Efficiency — Measured

The token efficiency of SIGN relative to JSON was measured on a production enterprise knowledge corpus. The reduction comes from three sources: elimination of structural noise characters; sigil compression of block-type declarations; and prose-efficient list syntax (+, ?, !) replacing array notation.

Measurement	Result
Single full knowledge document (SIGN vs equivalent JSON)	379 tokens vs 978 tokens — 61% reduction
9-document knowledge index (SIGN vs equivalent JSON)	406 tokens vs 818 tokens — 50% reduction
Daily savings at 1,000 agent invocations (index injection only)	~400,000 tokens per day before document retrieval costs

2.5 Two-Layer Delivery Architecture

Layer	Content
Layer 1 — Index	Document identity, status, graph edges, and one ## summary line per document. Approximately 400 tokens for a 40-document corpus.
Layer 2 — Full document	Complete SIGN structure: definitions, rules, constraints, inference logic, clusters, relationships.

The governing principle: Agents should always know what knowledge exists. Agents should only load the knowledge they need. Layer 1 satisfies the first. Layer 2 satisfies the second. The index is the map; the document is the territory.

2.6 Governance as First-Class Structure

Governance in SIGN is not metadata bolted on to content. It is first-class structure encoded in the sigil vocabulary. Three governance properties distinguish SIGN from every other knowledge representation format.

Breaking change detection is built into the format. Any change to @def, @include, @exclude, @rules, @rel, @constraints, or @infer in an existing active document is a breaking change requiring a major version bump. Every agent package declaring the affected document must be re-validated before production promotion.

Provenance is carried at the fact level. Every fact is marked as asserted — validated by a human authority — or inferred — derived by a rule or model. Agents know what to trust and what to treat with appropriate uncertainty. LLM-estimated facts are required to carry type:inferred and cannot be promoted to type:asserted without human review.

Audit trails are generated at the decision level. When an agent applies a @constraints rule or executes an @infer inference, the specific rule, its version, and its provenance are recorded in the execution trace. Every governed decision is traceable to the specific knowledge that authorized it.

SECTION 3

3. SIGN in Practice — A Worked Example

The following example demonstrates a complete SIGN document. It covers a generic domain concept definition and illustrates how all major sigil categories compose into a coherent knowledge document that an agent can reason over. The structure shown here is representative of any domain — the sigils carry identical meaning whether the subject is a capability definition, a regulatory requirement, a brand standard, or a risk policy.

3.1 The Layer 1 Index Entry

Every SIGN document contributes a single entry to the always-injected index. This is the agent's first contact with the document — everything it knows before deciding whether to retrieve the full content.

```
@canon v1.0 sha:a3f9c2

@doc CONCEPT-001 [std, active, agents+employees] v1
@links CONCEPT-002,CONCEPT-003,REF-001
  ## Canonical definition; inclusion/exclusion criteria; governed con-
straints; inference rules
```

Layer 1 index entry — 38 tokens. Document identity, status, audience scope, graph edges, and a self-contained summary. No rule content loaded until the agent needs it.

3.2 The Full Layer 2 Document

When an agent needs to apply classification logic, it retrieves the full document. The complete document provides everything required to make a governed decision with a full audit record.

```
@doc CONCEPT-001 [std, active, agents+employees] v1 sha:7d4a1f
@links CONCEPT-002,CONCEPT-003,REF-001
@cycle 90d  @reviewed 2026-04-01

@vocab
  depends_on     | subject cannot function effectively without target
present
  adjacent_to    | subjects frequently co-occur and complement; symmetric
  progression_to | target represents natural advancement from source; not
symmetric

@def core-concept
  Atomic, distinct, demonstrable unit of knowledge or capability with sta-
ble identity.
  Acquired through learning or experience. Observable through behavior or
outcomes.
  Identity stable across organizational contexts.

@include
  ? single bounded unit with a clear definition and scope
  ? demonstrable through execution, behavior, or outcomes
  ? intentionally developable or acquirable
  ? stable identity independent of specific organization or context

@exclude
  ! a broad bundle comprising multiple distinct units
  ! a process step, workflow state, or outcome
  ! a role, title, or organizational designation
  ! a platform or tool without distinct underlying capability

@constraints
  concept-identity
  mutex:   concept.type = bundled AND atomicity-validation = passed
  forbids: external label promoted to canonical without crosswalk valida-
tion
  forbids: process step used as identity anchor
  enforced-by: ontology-governance-layer
```

Illustrative Layer 2 document — 379 tokens vs 978 tokens for JSON equivalent (61% reduction). All sigil groups demonstrated: definition, criteria, constraints, inference, relationships.

3.3 What the Agent Can Now Do

With this document loaded, the agent can perform governed operations:

- **Apply governed classification:** Determine whether a candidate concept qualifies using @include and @ exclude criteria — traceable rule application, not ad hoc judgment.
- **Enforce structural constraints:** Reject proposals that violate @constraints, with an audit record citing the specific constraint and its version.
- **Execute governed inference:** Apply the concept-trajectory rule automatically when conditions are met, producing a derived fact tagged confidence:derived with its source rule recorded.
- **Traverse the knowledge graph:** Follow @rel declarations to retrieve related documents without hard-coded navigation logic.
- **Produce a complete audit trail:** Every decision cites the document version, the specific rule applied, the canon SHA at time of execution, and whether the fact was asserted or inferred.

SECTION 4

4. Domain Validation — Three Case Studies

A notation language designed as infrastructure must be domain-agnostic. The sigil primitives of SIGN — definitions, rules, constraints, inference, clusters, relationships — are universal structural properties of any governed enterprise knowledge domain. The following three case studies demonstrate this across materially different domains: healthcare compliance, financial services risk policy, and brand governance. Each has different buyers, different knowledge structures, and different compliance requirements. The format is identical across all three.

SIGN was originated at Career Highways in the context of a large-scale enterprise workforce intelligence platform. That domain is not detailed here in order to protect proprietary ontology design. It is referenced in the introduction and conclusion as the proof of concept from which the specification was derived.

4.1 Case Study: Healthcare Compliance

Healthcare organizations operate under a dense and frequently updated regulatory environment — HIPAA, CMS conditions of participation, state licensure requirements, accreditation standards, and payer-specific contract obligations. Compliance teams spend significant resources ensuring operational decisions respect these overlapping constraints. AI agents operating in clinical or operational workflows must apply this same logic — but it currently exists only in documents, not in any form an agent can reason over.

The SIGN contract for a healthcare compliance corpus demonstrates how regulatory logic translates directly into the sigil structure:

```
@doc HIPAA-003 [pol, active, agents+employees] v1 sha:9f2c44
@links HIPAA-001,HIPAA-002,CMS-001
@cycle 60d  @reviewed 2026-03-01

@def minimum-necessary-standard
  The principle that PHI disclosed or accessed must be limited to the min-
imum
  necessary to accomplish the intended purpose of the use or disclosure.

@rules
  1. Access to PHI must be scoped to the role's defined minimum-necessary
permissions
  2. Bulk export of PHI requires documented clinical or operational justi-
fication
  3. Agent actions touching PHI must record access purpose in the audit
log
  4. De-identified data does not trigger minimum-necessary constraints

@constraints
  phi-access
  mutex:   phi-access-scope = unrestricted AND role-type = non-clinical
  requires: audit-log-entry IS PRESENT when phi-accessed = true
  forbids: phi-transmission to non-covered-entity without executed BAA
  enforced-by: privacy-compliance-layer

@infer phi-violation-risk
  applies-to: agent-action
  when:  action.type = data-export
  and:   data.contains = PHI
  and:   justification = absent
  then:  violation-risk = high
  and:   alert-target = compliance-officer
  confidence: derived
  src:   hipaa-rule-engine-v2
```

Healthcare compliance SIGN document. The @constraints block enforces structural rules agents cannot override. The @infer block triggers automatic escalation on violation risk.

This example illustrates what SIGN enables that currently does not exist: an AI agent deployed in a healthcare workflow can operate with genuine regulatory constraint enforcement, not prompt-level suggestions. The constraints are in the runtime, not in the prompt. Violations produce an auditable record. Escalation is automatic and traceable. The compliance team can replay any decision and see exactly which rule version governed it.

4.2 Case Study: Financial Services Risk Policy

Banks and asset managers operate under investment policy statements, risk frameworks, counterparty limits, regulatory capital requirements, and trading constraints. Agents making portfolio decisions, generating client recommendations, or executing transactions must operate within precisely defined policy boundaries. Today those boundaries live in policy PDFs and the judgment of risk officers — neither of which is accessible to an agent in an enforceable form.

SIGN maps financial risk policy with precision because the domain is structurally identical to what SIGN was designed for: rules that must be applied, constraints that must be enforced, inference logic that must trigger escalation, and cluster semantics for exclusion lists and approved counterparty sets.

```
@doc RISKPOL-001 [pol, active, agents] v1 sha:3c8f21
@links RISKPOL-002,RISKPOL-003,REGCAP-001
@cycle 30d  @reviewed 2026-04-01

@def concentration-risk
  The risk arising from lack of diversification in a portfolio — excessive
exposure
  to a single issuer, sector, geography, or instrument type relative to
policy limits.

@rules
  1. Single issuer exposure must not exceed 5% of AUM without CIO approval
  2. Emerging market allocation above 15% requires Investment Committee
approval
  3. Illiquid assets must not exceed 20% of open-ended fund NAV
  4. ESG-screened mandates must exclude all issuers in @cluster:exclud-
ed-sectors

@constraints
  portfolio-limits
  forbids: equity-allocation > 0 when mandate-type = capital-preservation
  requires: CIO-approval IS RECORDED when issuer-concentration > poli-
cy-limit
  mutex:   leverage = active AND risk-rating = conservative
  forbids: trade-execution when real-time-breach-check = failed
  enforced-by: risk-management-layer

@cluster excluded-sectors [policy-exclusion]
  + sector:weapons-manufacturing
  + sector:tobacco
  + sector:thermal-coal
  [review-cycle:annual, authority:investment-committee]

@infer breach-approaching
  applies-to: portfolio-position
  when:  issuer.concentration > 4.5%
  and:   market.direction = adverse
  then:  breach-risk = elevated
  and:   alert-target = risk-officer
```

Financial services risk policy SIGN document. @cluster for exclusion lists, @infer for breach escalation, @constraints for hard portfolio limits. All enforced at runtime.

The financial services domain is arguably the purest expression of the SIGN design target. Auditability requirements are stronger here than in almost any other domain — regulators can demand a full decision trail for any trade or recommendation. The combination of @constraints (hard limits), @infer (early warning escalation), @cluster (exclusion and approved-instrument sets), and @xwalk (regulatory framework alignment) covers the full risk governance vocabulary in a form that is both agent-enforceable and human-auditable.

The @cycle and @reviewed governance blocks also earn their value here. Risk policy thresholds change on regulatory schedules that no agent should track ad hoc. A 30-day review cycle on RISKPOL-001 means the governing document is validated monthly, and any agent operating on a stale version is flagged automatically.

4.3 Case Study: Brand and Marketing Governance

Brand governance is one of the highest-value and least-solved problems in enterprise AI deployment. Marketing teams have access to content generation tools that produce output at scale — but brand standards, approved claims, prohibited language, and legal review requirements are documented in brand guidelines PDFs that no agent reads reliably. The result is either severely constrained agent use in marketing or expensive manual review pipelines that eliminate the speed advantage.

Career Highways' own brand corpus provides a direct production illustration. The organization maintains a governed canon covering visual identity, brand voice and tone, and approved power statements for each channel tier. Each document is expressed in SIGN and registered with the governed control plane. Any agent producing brand-aligned content retrieves and applies these documents before generating output.

```
@doc GTM-009 [ref, active, agents+employees] v1 sha:2b066b
@links GTM-001,GTM-002,GTM-010,REF-006
@cycle 180d  @reviewed 2026-04-28

@def visual-identity
  Career Highways visual brand identity. The durable visual constants — logo system,
  color palette, typography, photography direction — that govern every brand-aligned
  artifact. Voice and written tone are governed by GTM-002, not this document.

@props
  + brand-midnight  | #004150 — primary brand color; headlines and main blocks
  + brand-mint      | #4BBD9C — accent; pin icon fill
  + brand-grey      | #E8E8E8 — neutral surface; dividers

@rules
  1. Maintain clear space equal to lowercase 'e' in 'Career' on all sides of any logo
  2. CMYK and HEX values in @props are authoritative for all digital surfaces
  3. Do not lead any enterprise surface with 'AI-powered' framing
  4. Use brand tagline at brand layer only — never as a headline or power statement

@constraints
  brand-visual
  requires: brand color hex values from @props for all primary surfaces
  forbids: logo recoloring, rotation, stretching, drop shadows, or visual effects
  forbids: tagline used as headline or power statement
  forbids: cluster colors used for non-cluster encoding
  enforced-by: brand-governance-policy
```

Visual identity SIGN document from the Career Highways production brand corpus (GTM-009). Brand hex values, logo rules, and visual constraints as first-class governed structure.

```
@doc GTM-010 [ref, active, agents+employees] v1 sha:bdbc18
@links GTM-001,GTM-002,GTM-009
@cycle 180d  @reviewed 2026-04-28

@def power-statement
  Short, memorable phrase used as a headline in marketing surfaces — so-
cial posts,
  deck breakers, hero panels, OOH, brochures, web hero banners.

@cluster tier-a [approved-enterprise]
  + 'Talent is currency.'
  + 'Digitally activate job architecture at scale.'
  + 'Decision quality fuels organizational resilience.'

@cluster tier-c [approved-learner]
  + 'Your path is your power.'
  + 'You are in demand.'
  + 'Careers in the fast lane.'

@constraints
  power-statements
  requires: exactly one power statement per surface
  requires: tier matches channel — enterprise -> Tier A; product -> Tier
B; learner -> Tier C
  forbids: Tier C learner statements in enterprise sales, board, or PR
contexts
  forbids: rewording, partial use, or punctuation changes to approved
statements
  enforced-by: brand-governance-policy
```

Power statement governance (GTM-010). Approved statements as @cluster members. Constraints enforce tier-channel matching at runtime — no manual review required.

With these documents in the governed corpus, an agent generating brand content does not rely on a prompt that says 'follow our brand guidelines.' It operates within a runtime that enforces specific hex values, prohibits specific language patterns, constrains power statement selection to approved tier-matched options, and produces an audit record of every brand governance decision made. The difference between suggestion and enforcement is the difference between a prompt and a constraint.

The three case studies together demonstrate the key architectural property: the sigil primitives carry the same semantic weight across radically different knowledge domains. @constraints in healthcare compliance enforces HIPAA minimum-necessary requirements. @constraints in financial services enforces hard portfolio limits. @constraints in brand governance enforces logo and language standards. The format is identical. The domain is different. The governance guarantee is the same.

5. The Governed Knowledge Stack

SIGN does not operate in isolation. It is a layer in a complete governed knowledge architecture for enterprise AI. Understanding where SIGN sits in that architecture clarifies both its scope and its relationship to other components.

5.1 Architecture Overview

Layer	Description
Knowledge source (Layer 0)	Existing enterprise knowledge in native forms: HRIS exports, policy documents, brand guidelines, regulatory frameworks, competency models, SharePoint, Confluence. Pre-SIGN. The raw material.
SIGN specification layer (Layer 1)	Knowledge compiled into governed SIGN documents. Versioned, hashed, governed. The contract layer. This is what this paper specifies.
Canon registry and delivery (Layer 2)	The compiled SIGN bundle registered with a governed canon service. Two-layer delivery: index always injected, full documents retrieved on demand. Version management, breaking change enforcement, tenant isolation.
Agent runtime and policy (Layer 3)	The control plane governing agent access to canon documents. Tool registration, authority policy enforcement, delegation rules, escalation on denial. No decision logic lives in agents — it lives in the governed runtime.
Agent intelligence (Layer 4)	The AI agents that consume governed knowledge and produce outputs. Agents reason over SIGN-expressed knowledge but do not own it, modify it, or make governance decisions about it.

5.2 The Underlying Data Model as Black Box

One of the most important architectural properties of the SIGN stack is that agents never need to know what backing systems exist. An agent receives governed rules from the SIGN canon and governed tool access from the control plane. The underlying data models are completely opaque to it. This is not a limitation — it is the correct design. The knowledge contract expressed in SIGN is the interface. Everything below that interface is an implementation detail that can change without affecting the agent's behavior.

5.3 What SIGN Deliberately Does Not Do

Scope clarity is important for a specification paper. SIGN governs knowledge delivery to agents — not data serialization, configuration, or API contracts. SIGN is not a query language: agents retrieve documents, they do not query the graph via SIGN syntax. SIGN is not a replacement for agent decision frameworks: it defines what the decision means, not how the decision machinery executes. These are adjacent concerns with clean boundaries.

6. The Case for an Open Standard

The question of whether to open-source SIGN is a strategic one that deserves a principled answer. We make the case here that an open standard for the governed knowledge contract layer is the correct outcome for the industry — and that the historical precedent is unambiguous.

6.1 Why Proprietary Formats Fail at the Infrastructure Layer

Every enterprise infrastructure category that has succeeded at scale has done so through open standards. SQL became the standard for relational data access. HTTP became the standard for web communication. OpenAPI became the standard for API specification. In every case, standardization of the interface layer enabled the commercial infrastructure layer to scale — because adopters could build against the standard without committing to a specific vendor.

A proprietary knowledge contract format faces a fundamental adoption barrier: every organization that encodes its knowledge in a proprietary format is betting on the survival and alignment of the format's owner. For regulated industries with long institutional memories of vendor lock-in, this bet is not acceptable. An open standard removes it.

6.2 What Open Source Includes — and Does Not Include

Open source (MIT)	Proprietary / commercial
SIGN specification v1.0	Canonical knowledge corpus content — validated definitions, relationship graphs, inference rules
Reference parser (@sign-lang/parser)	Governed canon registry — the authoritative namespace adopters crosswalk against
Compiler CLI (@sign-lang/compiler)	Canon delivery infrastructure — control plane, policy engine, audit and replay systems
Linter and diff tools (@sign-lang/lint, @sign-lang/diff)	Client-specific corpora — all client knowledge expressed in SIGN
VS Code extension — syntax highlighting, diagnostics	Enterprise platform — the full governed agent infrastructure

The distinction is precise: open-sourcing SIGN is open-sourcing SQL, not open-sourcing the database. Anyone can write SQL. The enterprise database business remains worth hundreds of billions of dollars.

6.3 How Open Standards Create Commercial Value

The open-source layer creates commercial value through a specific mechanism. Adoption of the format requires crosswalk alignment with a canonical namespace. Every organization that encodes their knowledge in SIGN — mapping their internal labels to canonical IDs — has created a structural dependency on the namespace they crosswalk against. When they want to reason across that mapping at scale, keep it current as definitions evolve, or deploy agents that respect it with full audit trails, they need the platform that maintains the canonical namespace and the infrastructure that serves it.

> **The compounding advantage:** Every organization that adopts SIGN and builds their corpus against a canonical namespace makes that namespace more valuable — because it becomes the reference point that more crosswalk mappings point to. The network effect compounds in the direction of the authoritative content, not the format.

SECTION 7

7. Implications and Open Problems

SIGN v1.0 is a production specification validated against a real enterprise corpus. It is not a research prototype. That said, several important questions remain open — both in the specification and in the broader problem space SIGN addresses.

7.1 What SIGN Enables That Did Not Exist Before

The most important implication of SIGN is organizational, not technical. SIGN is the first format that allows enterprise knowledge to function as a first-class asset. Today organizational knowledge cannot be versioned at the rule level, queried by an agent, or made to govern anything automatically. It evaporates when the people who hold it leave. A SIGN corpus changes all of this. The knowledge is versionable, auditable, queryable, agent-governing, and persistent. It does not leave when people leave. It compounds in value as the agent systems that consume it become more capable.

7.2 The Ingestion Problem

Manual SIGN authoring will not drive adoption at scale. The path to widespread adoption requires automated ingestion: an organization should be able to point at existing knowledge assets — policy documents, HRIS exports, competency frameworks, brand guidelines — and receive a validated SIGN corpus as output, subject to human review and approval. This is an AI agent problem. The quality of structural inference from prose source content improves with every reviewed corpus. The feedback loop from human review to extraction intelligence is where durable competitive advantage accumulates in any platform built on SIGN.

7.3 Index Scaling

The two-layer delivery model operates correctly for corpora of up to approximately 200 documents. Above that threshold the Layer 1 index exceeds the token budget for always-injected content. A three-layer extension — meta-index, selective index, and full document — is required for production corpora at enterprise scale. This architecture is defined and will be addressed in SIGN v1.1.

7.4 Open Problems

Open problem	Description
Formal inference semantics	SIGN's @infer rules are deliberately informal — human-readable conditions rather than formal logic expressions. This makes them authorable by domain experts but limits formal verification. A formal semantics layer is a research-worthy extension.
Cross-corpus crosswalk resolution	When two organizations adopt SIGN and both build corpora against the same canonical namespace, their @xwalk mappings should be composable. The mechanics of multi-tenant crosswalk resolution at scale are not yet fully specified.
Temporal knowledge	SIGN documents are versioned but not temporally qualified. Facts that are true in a specific time window require an extension to the governance model that does not yet exist in v1.0.
Agent contribution to canon	Agents can read and apply SIGN but cannot propose updates to it. A governed contribution model — where agent-inferred facts can be promoted to canon through a defined human review workflow — is a valuable extension not yet specified.

CONCLUSION

8. Conclusion

The challenge of governed agent knowledge is not a niche infrastructure problem. It is the central constraint on enterprise AI deployment at scale. Organizations that cannot give agents governed, auditable access to institutional knowledge cannot trust agents with consequential decisions. Organizations that cannot trust agents with consequential decisions will not realize the value that AI systems are capable of delivering.

SIGN addresses this constraint directly. It is not a better serialization format. It is a purpose-built contract layer — the missing interface between what organizations know and what their agents are permitted to do with that knowledge. Its design principles are grounded in the operational realities of enterprise AI deployment: token budgets are real, governance requirements are non-negotiable, auditability is not optional in regulated industries, and domain experts must be able to maintain the knowledge that agents depend on.

SIGN was originated at Career Highways in the context of a large-scale enterprise workforce intelligence platform — a domain where the consequences of ungoverned agent decisions are both measurable and consequential. The three case studies in this paper demonstrate that the sigil abstraction holds across materially different domains: healthcare compliance, financial services risk policy, and brand governance. They share nothing in content but share everything in structure.

We publish SIGN v1.0 as an open specification under an MIT license. The specification is available at github.com/sign-lang. We invite the agent infrastructure community to adopt it, critique it, extend it, and build on it. The governed knowledge contract layer for the agentic enterprise is a problem the industry needs to solve together. SIGN is a starting point.

APPENDIX

Appendix A — Complete Sigil Quick Reference

Sigil	Layer	Purpose
@canon	Bundle	Version and SHA-256 integrity hash
@doc	Both	Document identity, type, status, audience scope
##	Index only	Agent summary line — one per document
@links	Both	Related document graph edges
@cycle	Full	Review cadence in days
@reviewed	Full	ISO date of last authoritative review
@vocab	Full	Governed relationship predicate definitions
@def	Full	Primary canonical definition
@props	Full	Named core properties
@include	Full	Positive inclusion criteria
@exclude	Full	Negative exclusion criteria
@rules	Full	Numbered operational rules
@boundary	Full	Identity and scope boundary
@constraints	Full	Structural rules agents must enforce
@cluster	Full	Named typed set with threshold semantics
@infer	Full	Governed inference rule
@rel	Full	Schema-level relationship declarations
@edges	Full	Instance-level graph edges
@xwalk	Full	Crosswalk identity resolution mappings
@attrs	Full	Required and optional attribute declarations
@disallowed	Full	Prohibited structural or naming patterns
@phases	Full	Lifecycle phase plan
@status	Full	Current-state key-value snapshot
@reserved	Full	Forward roadmap sigil declarations
->	Full	Outbound directed edge
<-	Full	Inbound directed edge
<=>	Full	Symmetric bidirectional edge
=>	Full	Crosswalk resolution mapping
+	Full	Cluster member or property entry
?	Full	Inclusion criterion
!	Full	Exclusion or disallowed entry
[...]	Full	Inline property bag
|	Full	Name/description separator

Appendix B — SIGN vs Existing Formats

Format	How SIGN differs
JSON	JSON serializes data for machine parsers. SIGN declares knowledge for agent consumers. SIGN eliminates 50–60% of structural noise while adding relationship semantics, inference rules, and constraint expression that JSON cannot carry.
YAML	YAML is a configuration format for human-readable data serialization. SIGN is a knowledge notation with relationship semantics, inference rules, governance primitives, and agent-optimized compression. Different purpose class entirely.
RDF / OWL	RDF has the correct semantic ambitions — formal relationships, inference support, graph model. It failed the adoption test: academic syntax, heavy toolchain, near-zero presence in LLM pretraining data. SIGN delivers equivalent semantic expressiveness at engineering-team adoption cost.
Markdown	Markdown is human-first and structurally untyped. SIGN compiles from markdown source but adds sigil typing, relationship structure, inference rules, and token compression for the agent consumption layer.
Prompt engineering	Embedding rules in prompts is ungoverned, unversioned, unauditable, and non-reusable across agents. SIGN makes knowledge a governed artifact separate from the prompt. The difference is auditability.
GraphQL	GraphQL is a query language for data graphs. SIGN is a declaration language for knowledge graphs. SIGN describes what the graph means and what agents may do with it — not a query interface.

Appendix C — Token Efficiency Detail

The following measurements were taken on a production enterprise knowledge corpus using a tokenizer calibrated for current LLM token counting conventions.

Document	SIGN tokens	JSON equivalent	Reduction
Single full knowledge document	379	978	61%
9-document knowledge index	406	818	50%
Single governance constraint block	24	51	53%
Single inference rule (@infer)	31	74	58%
Single cluster declaration (@cluster)	18	44	59%