Graph-Driven Validation of CSR (TFLs) Using Semantic Technologies

Muthiah Giri Hanuragav[1], Viswanathan Gopinath[2]

[1]*Care2Data, Chennai, India*
[2]*Care2Data, Chennai, India*

Abstract
Regulators require that every number printed in a clinical-study report (CSR) be internally consistent across its TFLs and traceable to underlying observations. Scripted Quality Assurance around rich-text outputs (RTF) is brittle under schema drift and provides weak provenance. We present an ontology-centered workflow that converts RTF to JSON, maps JSON to RDF using compact YAML mappers read by a deterministic translator, enforces structure with SHACL, and applies SPARQL rule suites for content checks. Large-language models (LLMs) assist only in drafting YAML; the converter is deterministic and auditable. A pilot across three studies reduced manual TFL QC by up to 75% while surfacing discrepancies earlier.

Keywords
TFL, Clinical Study Report, Knowledge Graphs, SHACL, SPARQL, YAML, ETL

1. Industry Problem and Background

CSRs contain hundreds of TFLs that must be perfectly consistent. Common regulatory findings include: inconsistent denominators, misapplied baseline rules (baseline = last non-missing value before first dose), and cross-table disagreements. **Example:** A vital signs table showing "72 subjects at baseline" while the listing shows 71 can delay submission by weeks.

Statistical teams deliver TFLs as RTF files whose layout obscures the clinical hierarchy: Subject → Visit → Observation. Even renaming "Screening" to "Screen" breaks compare scripts, forcing re-qualification. **QC** (Quality Control) means manual verification that printed numbers match source data, a process consuming 25% of submission timelines.

2. Limitations of Traditional Solutions

Relational warehouses can load the tables but they do not encode relationships such as derivesFrom or hasBaselineFlag. They therefore cannot answer cross-artifact questions like "Is the baseline value in the Vital-Sign Table the same as in the Vital-Sign Listing?" Procedural rule engines break when schemas drift and version-controlling thousands of SQL files becomes unmanageable. Column labels alone cannot capture semantics such as "baseline is the last non-missing value before first dose."

ISWC 2025 Companion Volume, November 2–6, 2025, Nara, Japan
hanuragav.muthiah@care2data (M. G. Hanuragav); gopinath@care2data.com (V. Gopinath)

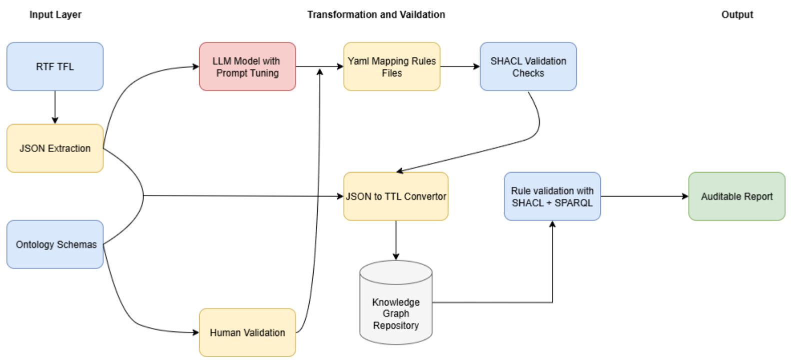

Figure 1: ETL workflow: RTF → JSON → RDF; YAML mappers drive a deterministic translator; SHACL gates structure; SPARQL checks content.

3. Why Semantic Technologies

An OWL vocabulary stabilizes concepts (subjects, visits, observations, summary cells), SHACL expresses structural constraints, and SPARQL encodes cross-artifact content rules. Because rules bind to *concepts* rather than column labels, validations remain stable even when table layouts change.

4. ETL Pipeline

Stage 1: RTF to JSON Conversion

A layout-aware parser extracts tables/listings from RTF files, normalizes headers, and emits canonical JSON while preserving complete provenance (file id, page, row, column). Figure 1 illustrates the complete ETL workflow.

Stage 2: JSON to RDF Transformation

A **deterministic Python translator** reads compact YAML mapper files to transform JSON into RDF. The translator mints namespaced IRIs following consistent patterns, performs type casting (e.g., `xsd:decimal`), threads `prov:wasDerivedFrom` relationships, and outputs idempotent N-Triples suitable for re-runs and diffs.

YAML mapper contents: Class instantiation targets, IRI templates, property mappings, and provenance passthrough rules. Only the translator consumes YAML; the triplestore receives the resulting RDF.

YAML creation process: Mappers are drafted from 2–3 representative JSON rows via constrained prompts, then human-reviewed and unit-tested before deployment.

Stage 3: Load and SHACL Validation

The repository accepts only graphs satisfying predefined shapes. Conformance and violations are recorded as machine-readable validation reports.

5. Automated Validation

The automated suite runs after the SHACL gate and mirrors established QC practice on the graph:

Structure. SHACL ensures well-formedness (typing and cardinalities; mandatory properties present; links such as `observes` point to `Observation`).

Content. A library of SPARQL templates re-computes denominators/percentages, verifies cross-table agreement (e.g., listings vs. summary tables), and enforces key business rules (e.g., baseline and change-from-baseline linkages; date constraints for dosing vs. events). Findings are emitted as RDF validation reports with click-through provenance to file/page/row/column.

Where rules come from. Rule templates are derived from sponsor QC checklists/SOPs, CDISC SDTM and ADaM implementation guides (variables, flags, and derivations), and ICH E3 guidance on CSR structure/consistency [1, 2, 3]. Shapes and queries are version-controlled and regression-tested prior to release.

6. LLMs: Draft Only, Never Execute

LLMs (GPT-4, Claude-3) are used *only* to draft YAML mappers, delivering ∼70% faster authoring; the converter never calls an LLM, keeping ETL deterministic and auditable. One of the many examples of hallucination is **Representative hallucination (inconsistent IRIs):**

- Input: Subject "P001" in multiple rows
- LLM: Generates both `ex:subject/P001` and `ex:subj/P-001`
- Impact: Duplicate subjects and incorrect aggregations

Why fine-tuning/APIs didn't fit; why prompt tuning did: public APIs lack fixed seeds, so drafts vary across runs (failing audit reproducibility), while on-prem fine-tuning strained GPUs (quantization degraded output; full precision was impractical). Constrained *prompt tuning* with a YAML schema and 2–3 golden JSON→triple examples yielded stable drafts; the deterministic translator preserved auditability.

7. Business Value and Pilot Results

Across three internal studies, manual TFL QC effort decreased by up to **75%** with earlier detection of denominator/baseline inconsistencies (internal benchmarks, Q4 2024). Gains stem from removing duplicate programming and enabling click-through lineage for auditors. Regulatory writers gain machine-readable provenance they can paste straight into submission dossiers

8. Conclusion

Separating semantics (ontology + shapes + rules) from layout (YAML mappers) yields robust, auditable TFL validation for the CSR. The graph layer provides stable checks under schema drift and click-through provenance from any printed number to its contributing rows.

Declaration on Generative AI

During the preparation of this work, we, the authors, used ChatGPT for grammar and spelling checks, paraphrasing, and citation management. After using this service, we reviewed and edited the content as needed and take full responsibility for the publication's content.

References

[1] CDISC. *Study Data Tabulation Model (SDTM) Implementation Guide*, Version 3.3, 2021.
[2] CDISC. *Analysis Data Model (ADaM) Implementation Guide*, Version 1.3, 2021.
[3] ICH. *E3: Structure and Content of Clinical Study Reports*, 1995 (R1).
[4] W3C. Shapes Constraint Language (SHACL). https://www.w3.org/TR/shacl/
[5] W3C. SPARQL 1.1 Query Language. https://www.w3.org/TR/sparql11-query/