cdisc®

2026 Europe Interchange

THE FUTURE IS CONNECTED: STANDARDS AND AI POWERING DIGITAL TRANSFORMATION

MILAN, ITALY | MAIN CONFERENCE: 20-21 MAY | TRAININGS & WORKSHOPS: 18, 19, & 22 MAY

Closing the Loop: Validating AI-Generated SDTM Mappings using CDISC CORE and Synthetic Data

Speakers

Constantin Weberpals

Machine Learning Researcher

CDTM

Ara Baghumyan

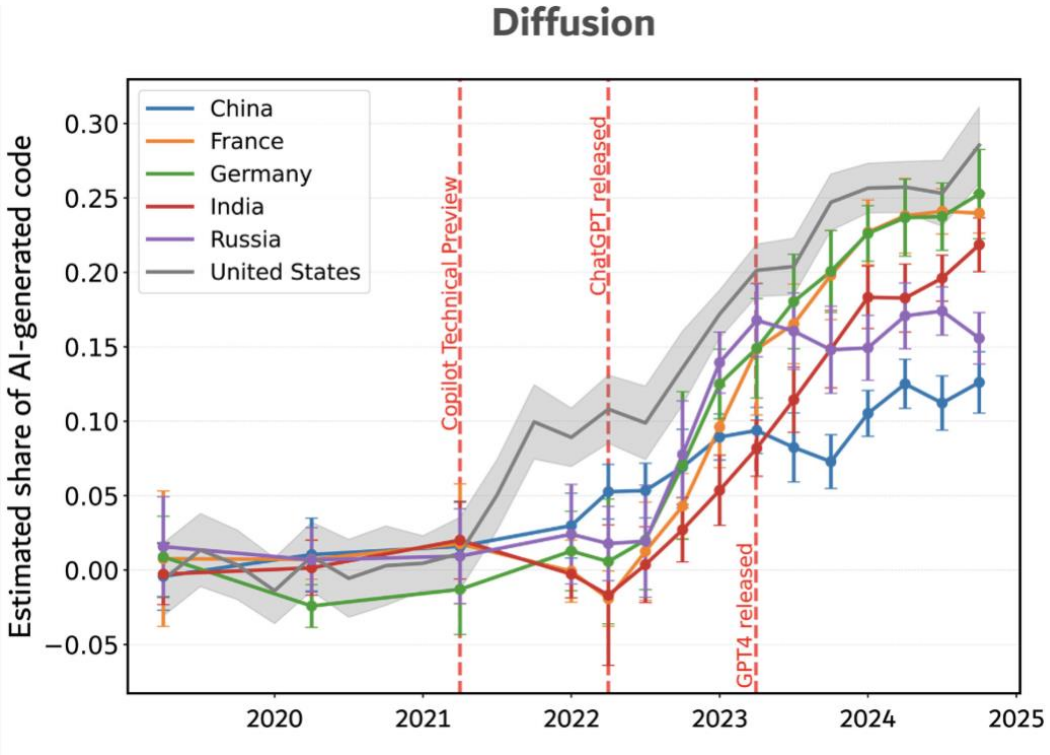
Co-Founder

BioBrain PRO

Agenda

1. The Inflection Point - Coding Agents
2. Applying a Closed-Loop Framework to SDTM
3. Key Components: Metadata Extraction & AI Mapping
4. Synthetic Data Generation
5. CDISC CORE Validation
6. Target Error Categories & Key Takeaways

Code Generation Reached a Tipping Point in Software Engineering



BUSINESS INSIDER

Subscribe

DOW -0.13% NASDAQ +0.01% S&P 500 +0.12% OIL +3.31% AAPL -0.84% NVDA +0.95% MSFT -0.94% TSLA -1% AMZN -2.59% META -0.4%

TECH

Google says 75% of the company's new code is AI-generated

https://www.science.org/doi/10.1126/science.adz9311

cdisc®

Why is Code Generation a Strong Use Case for LLMs?

- Defined syntax: the compiler is binary and output is structurally valid or it isn’t

- Machine-verifiable correctness: tests pass or fail

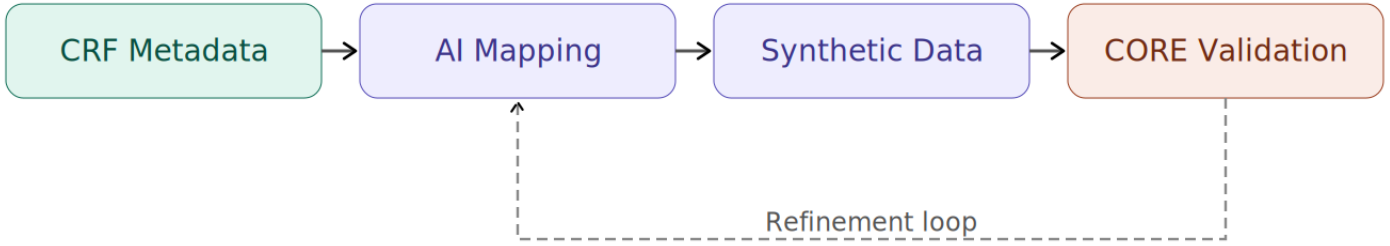
➔ Fast feedback loop: the signal arrives in milliseconds, Iteration is essentially free
➔ AI doesn't need to be 100% right the first time, it needs a signal it can iterate against

cdisc®

SDTM Has a Similar Structure

	Coding	SDTM mapping
Language rules	Language specification	SDTMIG
Defined syntax	Compiler / syntax rules	Controlled Terminology + Variable rules
Oracle / verifier	Compiler & Unit tests	CDISC CORE
Error signal	Error + line number	Rule ID, domain, variable, record

Framework: Core Components



Metadata Extraction & AI Mapping

① Metadata Extraction

- aCRF / eCRF, EDC exports, data transfer specs
- Central lab / ePRO, conditional logic

- **Output: JSON metadata catalog**
  - Per field: name, type, codelist, visit context

② AI-Driven Mapping

- **Relevant context**
  - SDTMIG specs, CT codelists, org conventions, prior study examples
- **Output: Structured JSON mapping spec**
  - Source-to-target mapping, derivation logic, CT mapping, VLM conditions

cdisc®

Synthetic Data Generation & CDISC CORE Validation

③ Synthetic Data Generation

- **Why synthetic data?**
  - CORE requires data records to execute, a spec alone cannot be validated
- **Probes known failure modes**
  - Boundary-condition timing values (--DY, EPOCH)
  - Mixed-case CT entries, realistic TESTCD/unit combos
  - Cross-domain subject IDs and intentional optional-field gaps

④ CDISC CORE Validation

- **Rules cover**
  - Structural requirements, CT usage, cross-domain consistency, SDTMIG logic
- **Output per violation**
  - Rule ID, severity, domain/variable, record ID, description

cdisc®

⑤ Iterative Refinement Loop

- Violations → structured feedback (mapping + rule + CORE definition)
- LLM proposes minimal targeted mapping revision
- Repeat until exit condition met

- **Triage layer**
  - Deduplicates cascading errors by root cause before LLM feedback
- **Circuit breaker**
  - Persistent failures routed to human review; expected non-conformance documented in SDRG

Validation shifts from a late quality gate to an integral part of the specification process

Avoiding Late-Stage Validation

- **Common Mapping Errors**
  - Controlled terminology mismatches
  - Value Level Metadata (VLM) violations
  - Derivation logic errors (--DY, EPOCH)
  - Cross-domain inconsistencies

- **Traditional pipeline**
  - Manual spec → code → datasets → late validation → expensive rework

Anatomy of a CORE Rule: CG0272

What it checks (plain English)

- If TS contains a record with TSPARMCD=HLTSUBJI and TSVAL=Y (healthy subjects study), then the TDIGRP record's TSVAL must be empty or equal the literal "HEALTHY SUBJECTS"

- **Why an LLM would miss it**
  - Cross-record dependency: one record's value constrains another record's required value
  - The correct value is a controlled phrase ("HEALTHY SUBJECTS"), not free-form English
  - An LLM mapping "the population is healthy" would happily write something plausible like "Healthy adults" — structurally invalid
- **Source**
  - cdisc-org/cdisc-rules-engine: tests/resources/rules/CG0272.yml — CDISC.SDTMIG.CG0272 • SDTMIG 3.2 & 3.3 • Status: Published

The rule itself (excerpted YAML)

```yaml
Core:
  Id: CORE-000473
  Status: Published

Operations:
  - filter:
      TSPARMCD: HLTSUBJI
      TSVAL: "Y"
    operator: record_count
    id: $HLTSUBJI_Y

Check:
  all:
    - name: TSPARMCD
      operator: equal_to
      value: TDIGRP
    - name: $HLTSUBJI_Y
      operator: greater_than_or_equal_to
      value: 1
    - all:
      - name: TSVAL
        operator: not_equal_to
        value: "HEALTHY SUBJECTS"
      - name: TSVAL
        operator: non_empty
```

Target Error Categories

- **Controlled Terminology Mismatches**
  - LLMs use semantically correct but non-exact CT submission values
  - CORE CT rules catch these; refinement loop resolves via CT tool
- **Value Level Metadata Violations**
  - LB/VS/EG: correct values depend on TESTCD x variable combination
  - E.g., hemoglobin g/dL vs mg/dL
- **Derivation Logic Errors**
  - LLMs handle typical cases but fail at boundaries (--DY, EPOCH)
  - Synthetic boundary records expose edge-case failures via CORE
- **Why manual review falls short**
  - All three categories are context-dependent and subtle — only CORE rules executing against realistic data reliably surface them at spec time

cdisc®

What Worked: CORE as a Feedback Engine

- **A loop instead of the gate**
  - spec → synthetic data → Dataset-JSON → CORE → LLM repair → new spec version
  - CORE becomes mid-spec feedback, not a late quality check
- **Versioned, auditable spec evolution**
  - AE added missing day variables and proposal-flagged vars
  - Each iteration persisted as a canonical spec version
- **Domain profiles as abstraction**
  - DomainProfile: expected vars, topic/result/unit vars, CT vars, VLM candidates
  - Additional domains could be added without rewriting the pipeline
- **Two-layer feedback design**
  - Domain-local validation works well across DM, VS, LB, AE, CM, …
  - Cross-domain dependency findings

cdisc®

What We Learned: Three Practical Gotchas

- **Synthetic data quality drives the signal**
  - Can create fake findings, e.g., bare-numeric CMDOSTXT triggered a CORE issue (numeric value)
  - Synthetic data needs to be good
- **Value-level metadata needs its own pass**
  - VSORRESU where VSTESTCD IN (SYSBP, DIABP) → mmHg
  - Define-XML necessary to run full value level checks
- **Not every finding should auto-fix**
  - safe_auto_apply: CT submission value normalisations
  - proposal_requires_review: medical coding (AEBODSYS / AESOC)
  - missing_context: cross-domain findings, unresolvable in a single-domain pass
- **CT needs tooling**
  - Model uses CT tool to extract accurate terms
  - Spec stores canonical codelist refs and term-level evidence

cdisc®

Key Takeaways

- **Feedback loop beats linear pipeline**
  - Integrating CORE validation into spec creation eliminates costly late-stage rework
- **Synthetic data is the key enabler**
  - CORE requires data records, targeted edge-case generation makes spec-time validation possible
- **LLMs need guardrails**
  - RAG + triage layer + circuit breaker = iterative refinement
- **Expected non-conformance is a feature**
  - Document expected violations in SDRG rather than force-resolving them

cdisc®

Thank You!

Questions?