Yesod AI

AI-Powered Multiple-Agent Pipeline for Automating ADaM Dataset Generation

Lucas Liu, Ph.D.
Bo Ci, Ph.D.

ML09 @ PHUSE Connect 2026

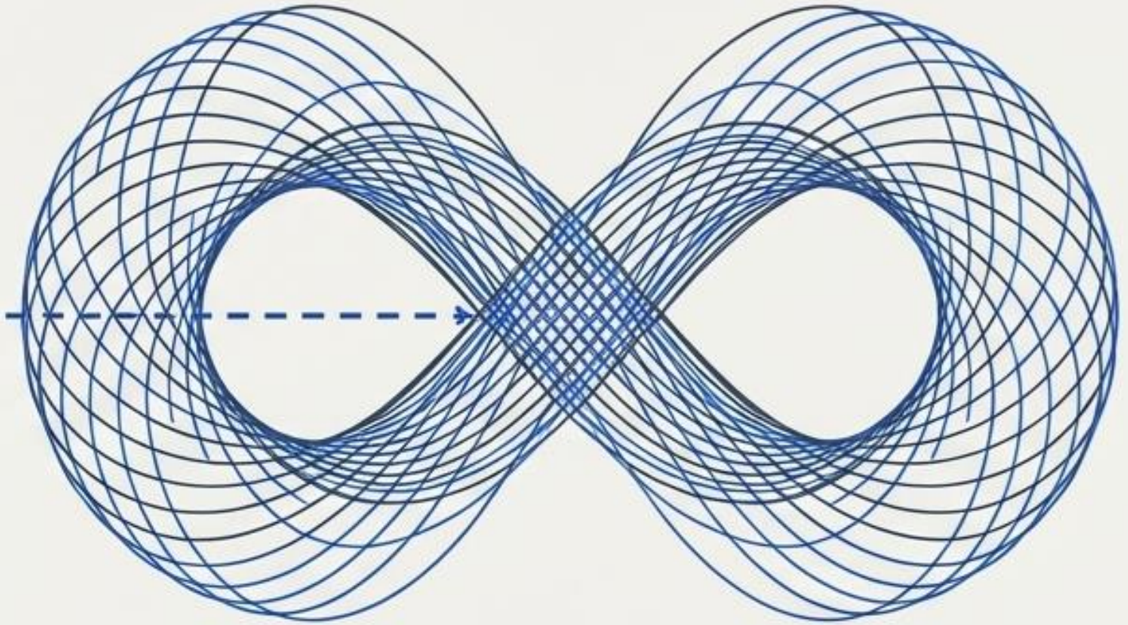
A Typical ADaM Programming Scenario

Your Mission (Should you choose to accept it): Derive a reliable ADaM dataset.

Understand:
Read the specification to grasp the derivation logic.

Code:
Write program to implement the derivations.

Validate:
A QC programmer independently implements the same logic.

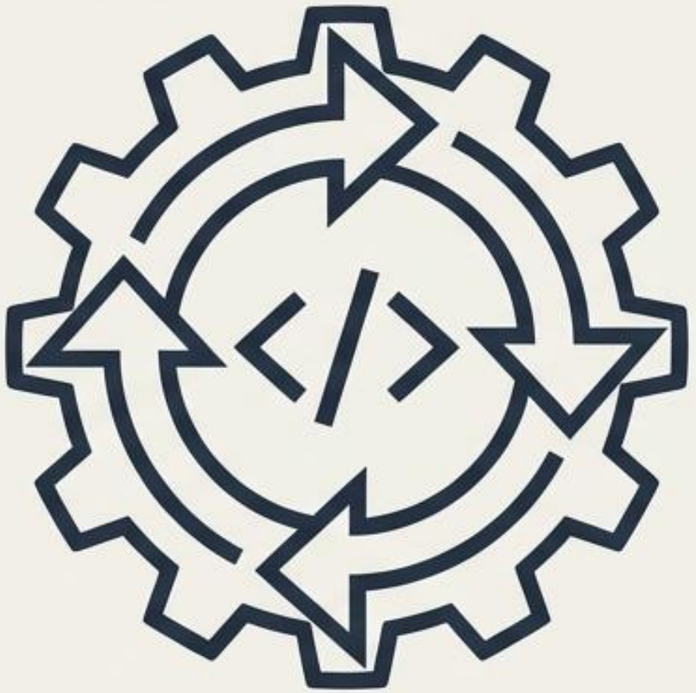


Then discrepancies appear.
Like an endless loop.

Pain points

Time-consuming

Labor-intensive



Repetitive work for QC

Why so exhausting?

Because in the current workflow,

humans are the only driving force.

So naturally, we start to look toward **automation**—especially now, as AI makes it increasingly possible.

So the question becomes: how far can automation go?

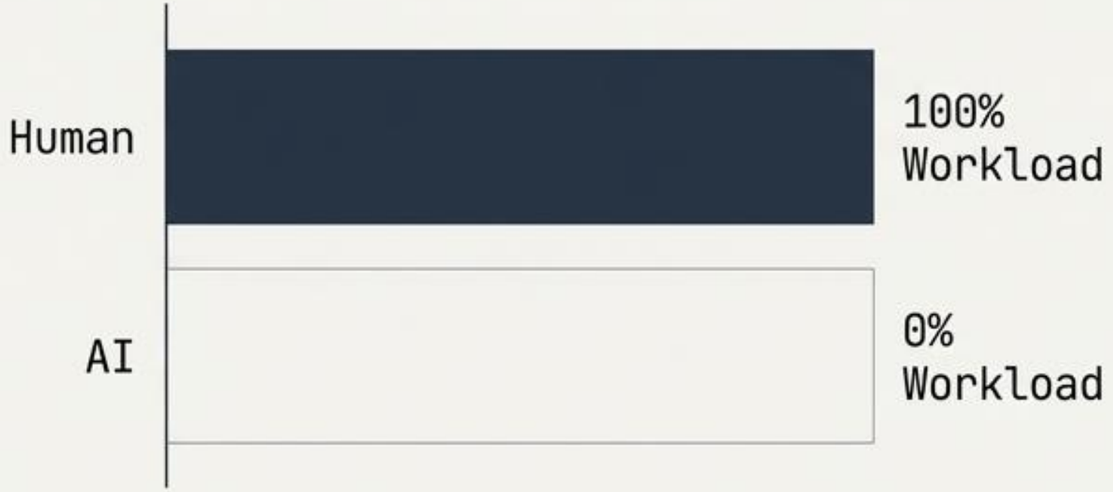
To measure that, we borrow a familiar framework:
The Levels of Autonomy

Level 0: Manual programming

Human completes the entire workflow

- Interpret specifications
- Write code
- Run code
- Debug errors
- Perform QC and reconciliation

Human plans route and drives

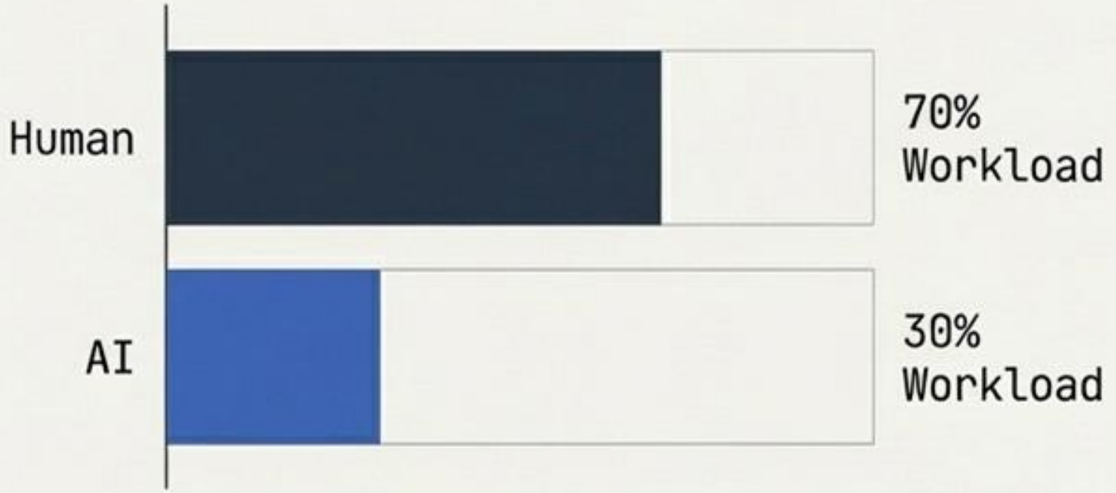


Level 1: AI-assisted programming

AI helps write code, but human drives the process

- Interpret specifications
- Write code
- Run code
- Debug errors
- Perform QC and reconciliation

Google Maps ->
Human driving



Level 2: AI-coding agents

AI write code & execute partial tasks, but humans remain in the loop

- Interpret specifications
- Write code
- Run code
- Debug errors
- Perform QC and reconciliation

Google Maps + cruise control

Level 3-4: High autonomy

AI runs the entire workflow. Humans only step in when needed.

Interpret specifications	Write code
Run code	Run code
Debug errors	
Perform QC and reconciliation	

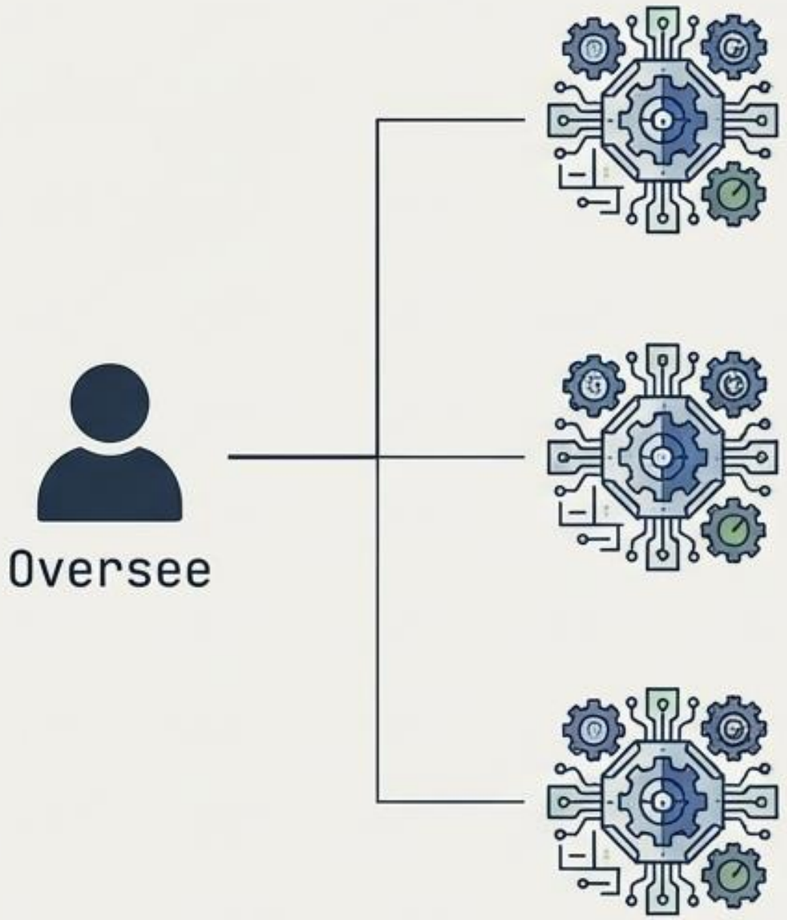
Waymo: just say the destination

Workload Bar

Human 10% Workload

AI 90% Workload

Our Goal: Level 3-4 AI teammates

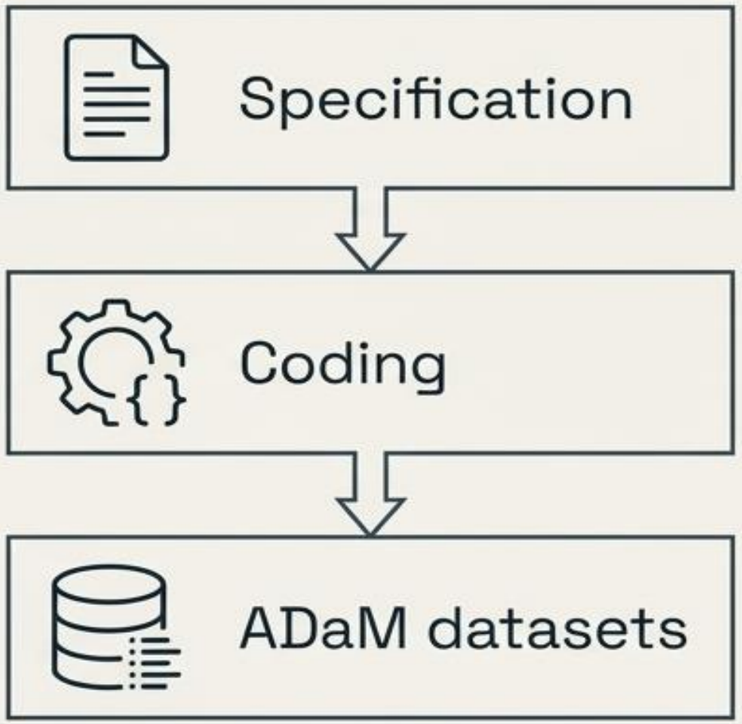


End-to-end delivery:
From ADaM specification to codes & ADaM datasets.

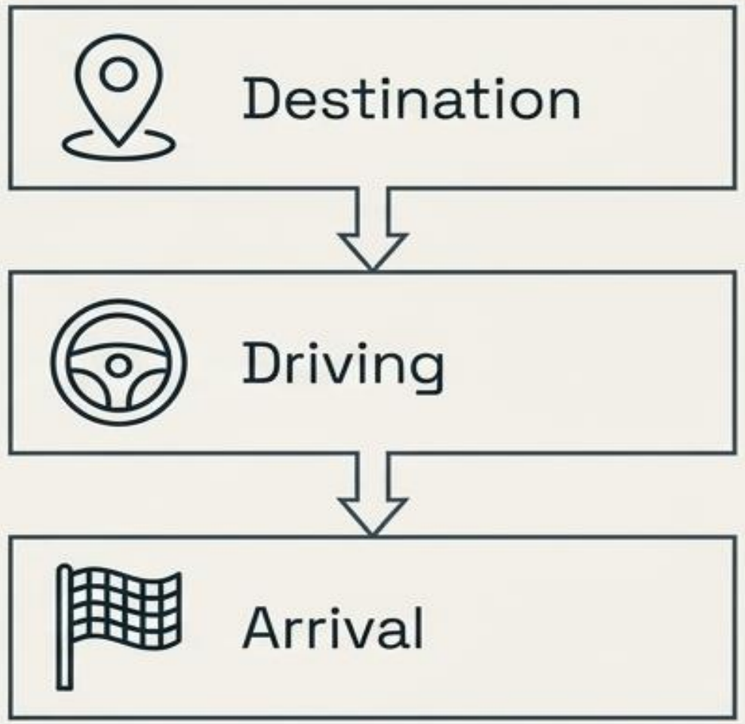
Less human involvement:
AI teammates only reach out when necessary.

80-90% Reduction in Turnaround Time in PoC cases

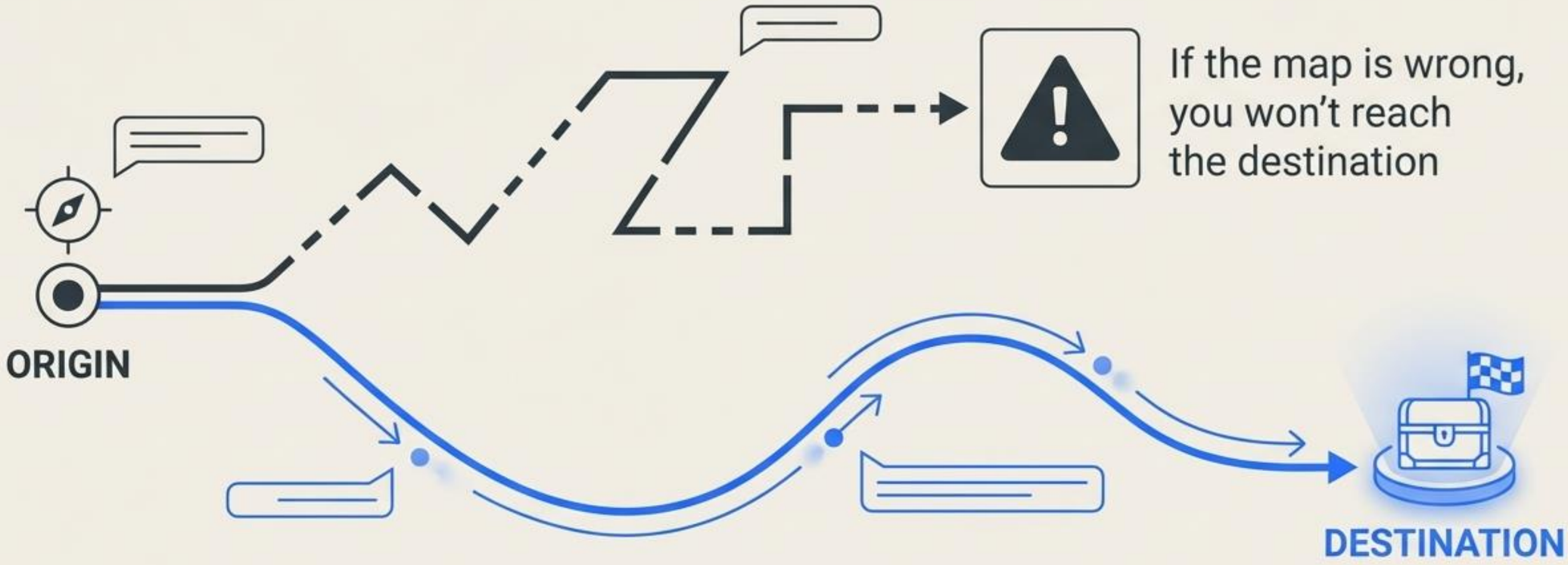
ADaM Generation Workflow



A Driving Analogy

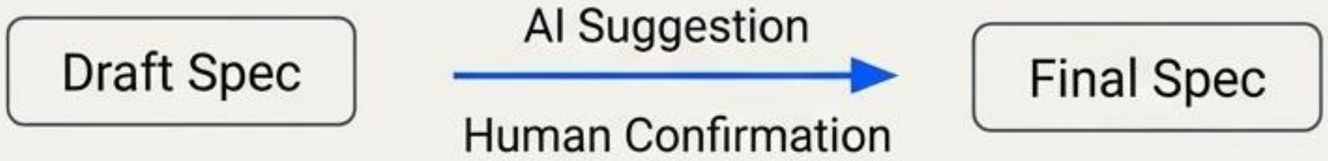


ADaM Spec is the treasure map



High-quality specifications enable reliable automation

AI-Driven ADaM Spec Review



What the Review Agent Detects

- Ambiguous languages
- Inconsistent variable name vs ID
- Typos
- Missing logics: Handling missing values

From Ambiguity to Precision

Initial Draft	AI Suggestion
Set to datetime of the first exposure observation start datetie of treatment and convert it to numeric datetime when sorted in datetime order.	Set to the numeric value of the earliest Start Date/Time of Exposure [EX.EXSTDTC] for the subject. If the subject has no exposure records, set it to null.

Typo

Clarify implicit logic

Add missing variable

Add edge case handling

Self-Correction Mechanism

We believe a smart system isn't the one that never makes mistakes, but the one that corrects itself when they occur.

Just like navigation — when you take a wrong turn, it doesn’t fail. It reroutes.

Code Generation

Designed for flexible code generation

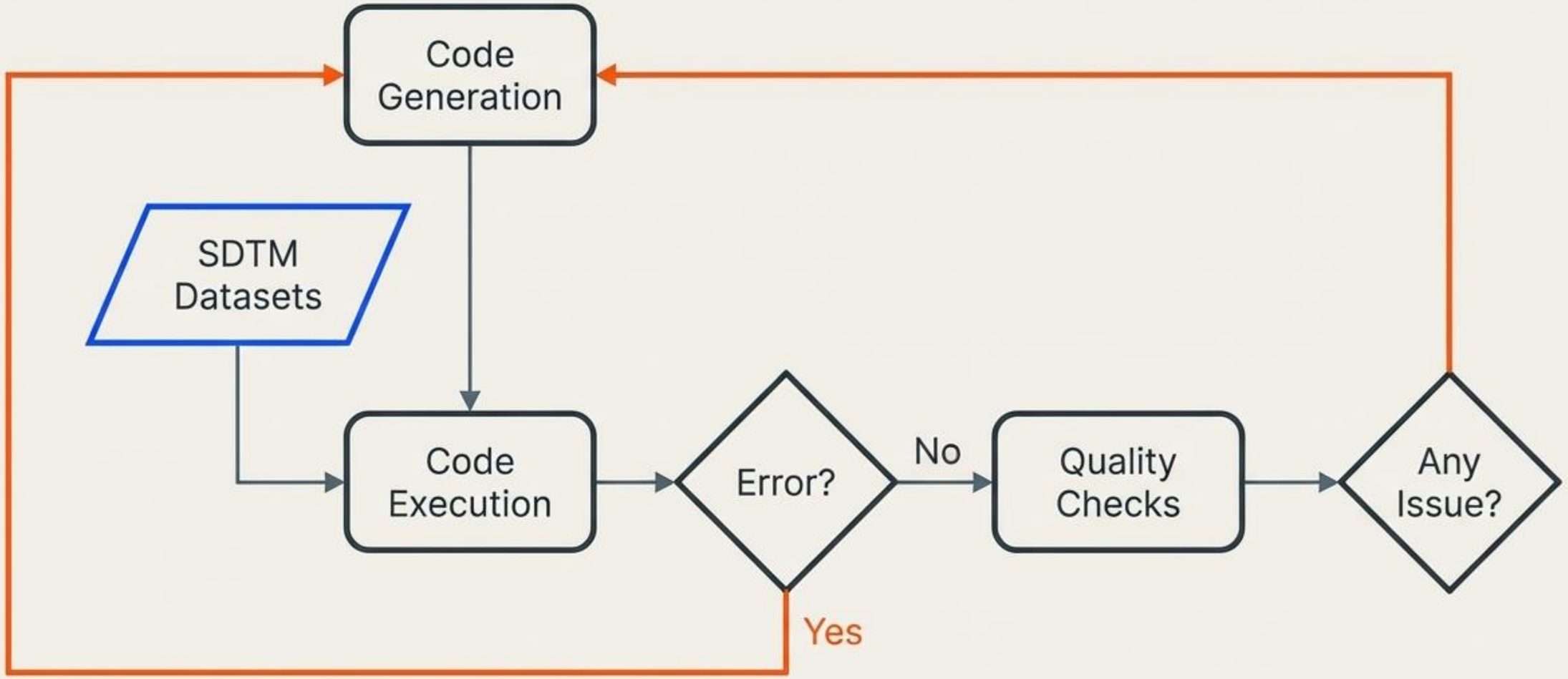
Language-agnostic: capability to support R, SAS, Python

Template-aware: can leverage existing code templates and macros

Generated Code

```r
tmp_BRTHDT <- DM %>%
  derive_vars_dt(
    dtc = BRTHDTC,
    new_vars_prefix = "BRTH",
    date_imputation = "06-15",
    highest_imputation = "M",
    preserve = TRUE
  ) %>% mutate(
    USUBJID = as.character(USUBJID),
    BRTHDT = as.Date(BRTHDT)
  ) %>% select(USUBJID, BRTHDT) %>%
  distinct()
```

Code Iteration



Summary

AI-driven Spec Review

Identify ambiguity, missing logic, and implementation risks before programming begins.

Autonomous execution, iteration, and QC

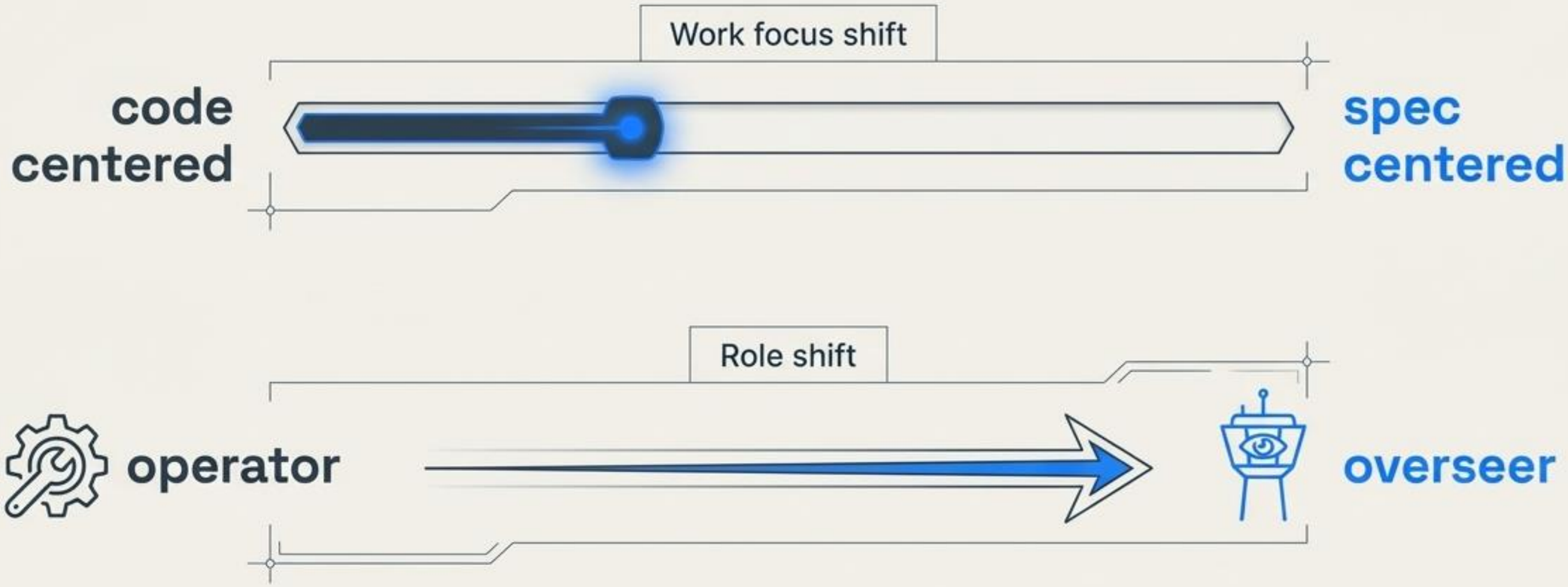
Generated code is automatically executed, debugged, refined, and quality-checked.

Significant efficiency potential

> 80% reduction

in delivery time for selected workflows.

Some thoughts on where the field is heading



Partner with us to reshape the future of statistical programming

Bo Ci, CEO

Phone: (469) 601-7522

Email: ci.bo@yesodai.com