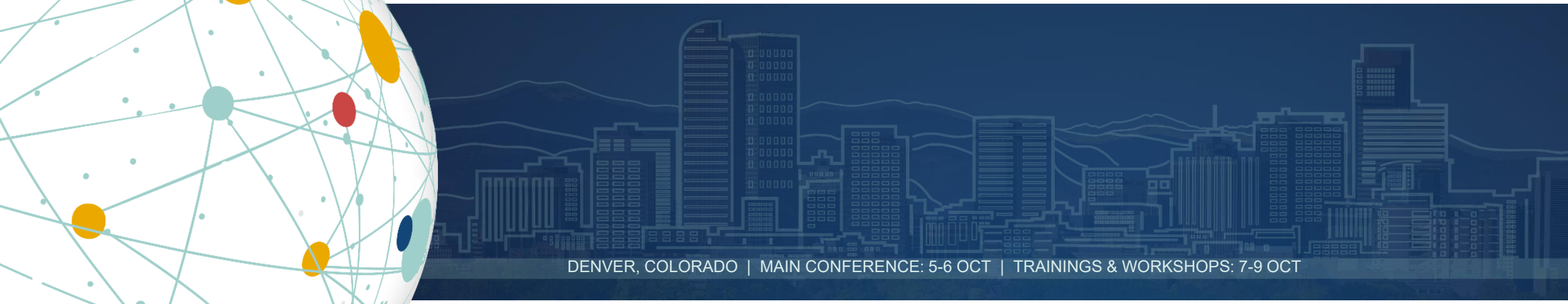


cdisc®

2026 US Interchange

THE FUTURE IS CONNECTED: STANDARDS AND AI POWERING DIGITAL TRANSFORMATION



Interrupting the Conversation: AI-Assisted Clinical Submission Artifacts as a Model for Targeted AI Adoption

Carlo Radovsky, Immanant

Speakers



Carlo Radovsky

President

Immanant

Clinical data standards consultant with over three decades of pharmaceutical industry experience in standards implementation, governance, and systems.

In some ways, isn't this the worst?

The essence of Vibe Slides

- Color and composition signal unearned effort
- Decoration risks crowding out information



But…

ain’t it tempting?



aCRF Considerations

Some Use Cases

Up-versioning per MSG v2.0	Copying Across Studies	Generating Based on Metadata Sources
• Differences in both style and content	• Form Order and Layout Differences	• Interaction between EDC and SDTM specifications

- Each has distinct technical challenges
- Some lend themselves to repeatable tooling more than others

cdisc®

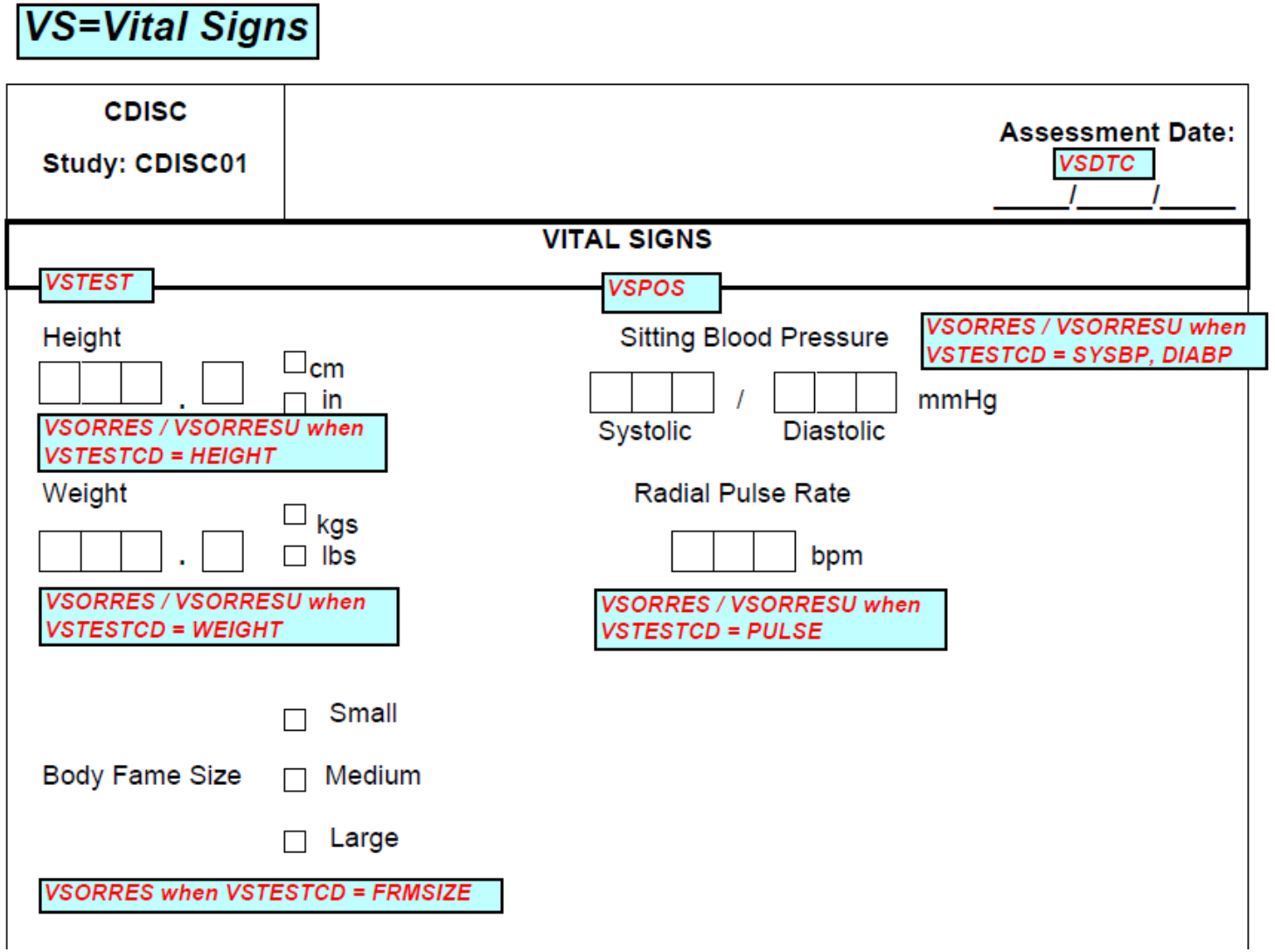
Some Use Cases

Up-versioning per MSG v2.0
• Differences in both style and content

- Let's take the first use case

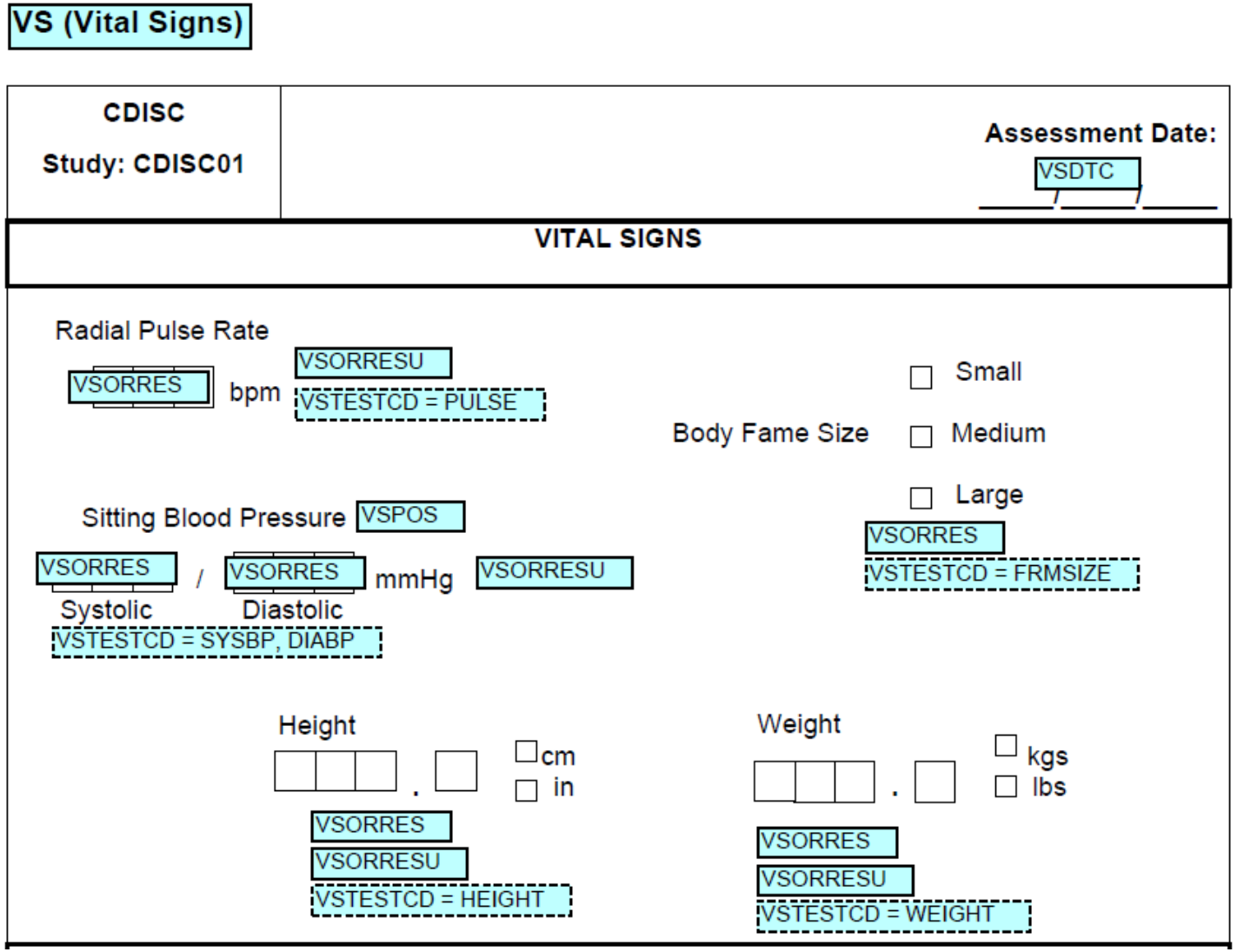
Use Case 1: Challenge

Turn this …



Use Case 1: Challenge

Into this..



Use Case 1: Challenge

Using nothing but this…



Context & Motivation

The aCRF Annotation Technical Problem

Poor Tooling / Manual Effort

Without a metadata driven solution, reverts to out-of-the box software solutions.

Adobe functionality (comment & bookmark management, FDF/xFDF import/export) woefully inadequate

Third-party commercial tools feasible, but technically disjointed

Solution

- **Python libraries solving a range of technical needs**
- **Generate updated PDF as an output, code secondary**

Shifting Rules / Conventions

SDTM-MSG v1.0 vs. v2.0 significantly different

Historical mappings insufficient

Evolve within organizations

Solution

- **Evaluate annotation text and establish rules for parsing and appearance**

Fuzzy Location

Use of fixed coordinates impossible

Content shifts with within and across studies

Both within a page, and overall page order

Solution

- **Anchor each annotation to its associated CRF prompt text — not to a fixed position on the page**

Architecture

Two-Phase Design

Phase 1 — Extract

Input
Annotated source CRF
+ blank target CRF

Process
Apply splits
Detect anchor words
Record offsets + colors
Connect bookmarks to Forms

Output
Annotation_bookmark_catalog.json
Annotation_bookmark_catalog.xlsx
Annotation_bookmark_catalog.csv

`extract_annotations_bookmarks.py`

JSON / Excel / CSV catalog

Phase 2 — Apply

Input
Annotation_bookmark_catalog.json
+ target blank CRF PDF

Process
Map pages (fuzzy headers)
Annotation Processing: Look up anchors in target, stack splits, align columns, overlap avoidance
Bookmark Processing (Link to new pages)

Output
Annotated target CRF PDF
(submission-ready acrf.pdf)

`apply_annotations_bookmarks.py`

Phase 1: Extract

Extracting & Upgrading Annotations

For each FreeText annotation on each page of the source annotated CRF:

1	Read /Rect, /Contents, /C	Extract native PDF coordinates, text, and background color from annotation dictionary
2	Upgrade to v2.0 format	Map domain labels (DM=Demographics → DM (Demographics)), fix typos, normalize spacing
3	Detect color	Map source /C float triple to nearest v2.0 palette color (Blue/Yellow/Green/Orange/White)
4	Detect anchor word	Select semantically relevant nearby word: proximity + semantic bonus − boilerplate penalty
5	Apply splitting	Split 'when' and ' / ' patterns → 206 source → 273 v2.0 annotations
6	Export catalog	Write JSON + Excel + CSV: text, color, font, split metadata, anchor, offset, fallback %

Phase 1: Extract

The Annotation Catalog (JSON / Excel / CSV)

The catalog is the single handoff artifact between Phase 1 and Phase 2. It can be inspected, edited, and reused for multiple target CRF versions.

Key Fields per Record

`text`	v2.0 annotation text (post-upgrade, post-split)
`color_name`	Blue / Yellow / Green / Orange / White
`fs + bold`	Font size (8/10/12) and bold flag
`dashed`	True = dashed border (conditional qualifier)
`anchor_text`	Anchor word in source blank PDF
`offset_dx/dy`	Distance from anchor to annotation center (pts)
`anchor_x/y%`	Proportional fallback if anchor not found in target
`split_group`	ID linking splits to same source annotation

Why This Matters

✓ **Reviewable**	Open in Excel: inspect every annotation before applying
✓ **Editable**	Fix incorrect anchors or split decisions by editing cells
✓ **Reusable**	Apply the same catalog to v2, v3, v4 of the target CRF
✓ **Auditable**	Excel Summary sheet shows per-page counts and domain labels
✓ **Portable**	JSON is environment-agnostic; works cross-platform, in CI/CD
✓ **Traceable**	orig_text field shows exactly what was in the source aCRF

Phase 2: Apply Annotations

Semantic Anchor Word Selection

Each annotation stores its anchor word from Phase 1. Phase 2 locates that word in the target page via a 4-stage fallback chain.

Stage 1
Exact match
Case-insensitive exact text match on anchor_text

→

Stage 2
Fuzzy match
SequenceMatcher ratio ≥ 0.75 on anchor_text

→

Stage 3
Context word
Exact match on any stored context word (up to 3 neighbours)

→

Stage 4
Semantic keyword
Match annotation keywords (e.g. HEIGHT, PULSE) in page text

Phase 1 Anchor Scoring Formula

```
score(word) = distance
            – semantic_bonus  (up to 350 pt for keyword match)
            + boilerplate_pen (200 pt for 'study', 'date', 'yes' …)
```

Result:

"VSTESTCD = HEIGHT" anchors to word **"Height"** — not "tests" or "Yes".

Boilerplate penalty prevents generic page words from being selected even when geometrically closest.

Phase 2: Apply Annotations

Single Coordinate Frame Architecture

The most impactful fix in the entire development: establish one canonical coordinate space for ALL computation.

Native PDF	pdfplumber	Display (rot)
y = 0 at BOTTOM	y = 0 at TOP	y = 0 at TOP
pW × pH (unrotated)	Post-rotation display	Same as pdfplumber
Final /Rect output only	ALL internal computation	Equivalent for rot≠0 pages

Rule: plumb_center() to read source positions → ALL placement in pdfplumber → plumb_to_native() once at output

Bug (before fix): FRMSIZE annotation placed at y-mirror of correct position. Annotation for 'Body Frame Size' appeared at bottom of page instead of middle.

Fix: plumb_center() correctly converts native rect (bottom-up) to pdfplumber coords (top-down) for all 4 rotation values. Single-transform, verified for rot = 0/90/180/270.

Phase 2: Apply Annotations

Five-Step Placement Pipeline

1	**Initial position**	Anchor position + stored offset. Clamped to page bounds.
2	**Stack split groups**	split parts aligned to same x0, stacked 2 pt apart.
3	**Column alignment**	Annotations within 25 pt x0 and same 8-char prefix → width-normalised.
4	**UC3 domain labels**	Domain labels placed left-margin, below header band. Scan right/down for clear slot.
5	**Overlap avoidance**	Nudge 7 directions at 4/8/16/32 pt. Fall back to text wrapping at narrower width.

```
python apply_annotations.py  catalog.json  v1_blank.pdf  v2_blank.pdf  acrf_output.pdf
```

Results

Migration Results — CDISC01 Study

22	7	206	273	77
Pages matched	Pages reordered	Source annotations	v2.0 output anns	Bookmarks

- ✓ All 22 pages correctly matched; 7 reordered pages (including 3-page disposition/CGI/inventory shift) auto-resolved
- ✓ 3 landscape pages (rot=90) — annotations horizontal, coordinates correct
- ✓ 273 v2.0-compliant annotations from 206 source: 67 new boxes created by splitting
- ✓ 77 bookmarks across 2 hierarchies and 4 levels carried forward; 7 reordered page links updated
- ✓ FRMSIZE (Body Frame Size) correctly placed after coordinate inversion fix
- ✓ Runtime: < 10 seconds for full 22-page migration

Results

Advantages & Limitations

✓ Advantages

- Hours → seconds: full migration in < 10 seconds
- Two-phase: catalog inspectable / editable before apply
- Same catalog → multiple target CRF versions
- Interpretable: debug log shows anchor for every annotation
- v2.0 compliant: colors, fonts, border styles, splitting
- Handles page reorder, layout change, rotation automatically

⚠ Limitations & Future Work

- Color mapping heuristic: 4+ domain pages may need manual fix
- No new page or field detection: pages and fields added in target need manual annotation and bookmarking
- Column alignment by 8-char prefix: heterogeneous text may group poorly
- Overlap avoidance is greedy — not globally optimal for dense pages
- Freeform note detection (lowercase/comma heuristic) has edge cases

Results

Key Lessons from Script Iterations

01

Anchor to content, not coordinates

Coordinates are meaningless across CRF versions. The right unit of position is 'relative to the word this annotation labels' — not '(x, y) on the page'.

02

One coordinate frame, convert once

Mixing pdfplumber (top-down) and native PDF (bottom-up) mid-pipeline produces mirror-image errors. Establish one canonical space; convert at the boundary only.

03

Proximity isn't enough for anchoring

The nearest word to any clinical annotation is almost always a boilerplate word. Semantic scoring — boosting annotation-relevant words — is the key differentiator.

04

Join all headers for page matching

Matching on the first header alone fails for pages that share a title. The joined multi-header key correctly distinguishes MMSE from Physical Exam despite the same opening line.

Results

Key Lessons from Script Iterations

05

Algorithmic Drift

LLMs are stochastic processes that generate text from a probability distribution

Nuanced and valid code can be “optimized” away

Context window limitations

06

Independent Technical Knowledge

Early and informed understanding of issues reduced trial-and-error updates with specific examples and algorithm modifications targeting root causes

Results

Complete Deliverables Package

acrf_utils.py
Shared utilities: font metrics, coord transforms, anchor scoring, splitting rules, header extraction

extract_annotations_bookmarks.py
Phase 1: read source aCRF → upgrade → split → anchor → export JSON/Excel/CSV catalog

apply_annotations_bookmarks.py
Phase 2: read catalog + target blank → map pages → place annotations → place bookmarks → write annotated PDF

annotation_bookmark_catalog.xlsx
273+77-row Excel workbook with content type, annotations and bookmarks, per-page summary, auto-filter

Change Report.docx
Summary of the changes from source to target relating to annotations and bookmarking.

Annotation_bookmark_catalog.json
Machine-readable JSON catalog: all 273 annotation and 77 bookmark records with full placement data

Annotation_bookmark_catalog.csv
Flat CSV equivalent of the Excel catalog for data pipeline integration

Technical Reference.docx
Full API documentation: function signatures, parameters, JSON schema, algorithm detail

presentation.ppt
High-level technical summary

Results

Automated aCRF Annotation Migration

Automated aCRF annotation migration is achievable with three Python modules, zero manual intervention, and < 10 seconds runtime. The two-phase design makes the catalog a reviewable, editable artifact — not a black box.

206 source annotations → 273 v2.0-compliant annotations	**| 22 pages | 7 reordered | < 10 seconds**
77 source bookmarks → 77 target bookmarks	

THOUGHTS ON NEXT STEPS FOR THIS UTILITY

- Derive new annotation colors from domain order-of-appearance rather than source color heuristics
- Detect newly added CRF pages and fields in the target and flag them for manual processing
- Constraint-based placement solver for globally optimal overlap avoidance on dense pages
- Extend to multi-study annotation library: learn anchor patterns across studies
- Upscale bookmark management, leveraging existing visit schedule

Define-XML

Define-XML

Five Targeted Utilities, One Define-XML

Each utility reads one source the study already has and writes one slice of the Define-XML. The aCRF catalog from the first use case is the entry point.

	Source you already have	AI-generated Python utility	Define-XML elements written
1	Annotated CRF (acrf.pdf)	Page metadata extractor	`def:Origin Type="Collected" + def:PDFPageRef`
2	Datasets (XPT / Dataset-JSON)	Base template generator	`ItemGroupDef, ItemDef, ItemRef, KeySequence`
3	Mapping specification (Excel)	Spec attribute extractor	`Origin Type (Derived / Assigned / Protocol), CodeListRef`
4	SAS programs (.sas)	Derivation reviewer	`MethodDef + def:CommentDef text`
5	Submitted PDFs (cSDRG, aCRF)	Document link builder	`def:leaf, def:DocumentRef, NamedDestination`

Same pattern as the aCRF work: every utility writes an inspectable intermediate (JSON / Excel) before anything is merged into define.xml.

Define-XML

1 · aCRF Page Metadata → Origin Page References

The annotation catalog already knows every annotation, its domain colour and its page. Define-XML just needs it pivoted by variable.

Input	Process	Output
`acrf.pdf` — FreeText annotations, one colour per domain **`Annotation_bookmark_catalog.json`** from Phase 1 (text, colour, page, split group) Domain labels ("VS (Vital Signs)") anchor each page to its dataset	Parse annotation text: DATASET.VARIABLE, "VAR when TESTCD = X", "VAR / VARU" Resolve domain by colour link when the annotation omits the prefix Collect distinct physical page numbers per (dataset, variable) Flag variables annotated on the CRF but absent from the datasets — and vice versa	**`acrf_pages.xlsx`** — Dataset · Variable · Pages · Annotation text Merged into ItemDef as a Collected origin with page refs Page list re-generated whenever the aCRF is re-issued (no hand edits)

```
<ItemDef OID="IT.VS.VSORRES" Name="VSORRES" DataType="text" Length="8">
  <def:Origin Type="Collected" Source="Investigator">
    <def:DocumentRef leafID="LF.acrf">
      <def:PDFPageRef PageRefs="12 14 27" Type="PhysicalRef"/>
    </def:DocumentRef>
  </def:Origin>
</ItemDef>
```

Why this is the "interconnected" piece

The catalog was built to migrate annotations. Re-read as metadata, it is also the authoritative list of what was collected and where

Define-XML

2 · Base Template from Dataset Metadata

Start from what is submitted, not from what was planned. The datasets fix the variable list, order, types and lengths.

1	**Read the datasets**	XPT (xport) or Dataset-JSON: names, labels, types, lengths, variable order per dataset
2	**Classify with the IG / Spec metadata**	Look up domain class, structure and Repeating / IsReferenceData from the SDTMIG reference metadata or the specification
3	**Derive key sequence**	Sort keys from the IG default keys, confirmed against actual uniqueness in the data
4	**Size lengths against metadata or source datasets**	Length = max observed for text (per FDA TCG), not the SAS allocation; numeric SignificantDigits from data
5	**Emit the skeleton**	MetaDataVersion, Standards, ItemGroupDef / ItemRef / ItemDef, def:leaf for each dataset; origins left blank on purpose

```
<ItemGroupDef OID="IG.VS" Name="VS"
  Repeating="Yes" IsReferenceData="No"
  SASDatasetName="VS" Purpose="Tabulation"
  def:Structure="One record per vital
    sign measurement per visit per subject"
  def:Class="FINDINGS"
  def:ArchiveLocationID="LF.VS">
  <ItemRef ItemOID="IT.VS.STUDYID"
    OrderNumber="1" Mandatory="Yes"
    KeySequence="1"/>
  ...
</ItemGroupDef>
```

What this deliberately does not do: guess origins, methods or comments. A skeleton with empty slots is more useful than one with plausible-looking text, because every blank is a visible to-do for utilities 3–5.

cdisc®

3 · Spec-Based Attributes: Collected vs. Derived

The mapping specification is where the sponsor already recorded intent. The utility reads it, reconciles it, and refuses to invent what is missing.

Read the spec

Excel mapping spec: one row per dataset · variable

Columns vary by sponsor — a small header map (JSON) tells the script which column is Origin, Codelist, Comment

Fuzzy header match with a review report when confidence is low

Reconcile

Join spec rows to the skeleton ItemDefs by dataset · variable

Cross-check against utility 1: a variable with CRF pages but spec origin "Derived" is a conflict, not a choice

Unmatched on either side → exception list, never silently dropped

Write attributes

`def:Origin Type` — Collected · Derived · Assigned · Protocol · Not Available

`CodeListRef` from the spec codelist column

`def:CommentOID` when the spec carries free-text notes

Variable	CRF pages (util. 1)	Spec origin	Resolution
VSORRES	12, 14, 27	Collected	Agree → Collected with PDFPageRef
VSSTRESN	—	Derived	Agree → Derived; MethodOID to be filled by utility 4
VSDTC	12	Derived	**Conflict → exception list for human decision**

Define-XML

4 · SAS Program Review → Derivation & Comment Text

The one step where the model reads code rather than metadata — and the one step that most needs a human on the other side.

```
/* vs.sas */
if VSTESTCD in ("SYSBP","DIABP") and
   VSORRESU = "mmHg" then do;
  VSSTRESN = input(VSORRES, best.);
  VSSTRESU = "mmHg";
end;
else if VSTESTCD = "TEMP" and
   VSORRESU = "F" then
  VSSTRESN = round((input(VSORRES,best.)
             - 32) * 5/9, 0.1);
```

Generated VLM MethodDef (draft — for review)

<**MethodDef OID**="MT.VS.TEMP.VSSTRESN" **Name**="Derivation for VSSTRESN for TEMP" **Type**="Computation">
<**Description**> <**TranslatedText** xml:lang="en">Height converted to centimetres. When VSORRESU = "cm",VSSTRESN = VSORRES converted to numeric. When VSORRESU = "in", VSSTRESN = VSORRES × 2.54,rounded to one decimal place.".</**TranslatedText**> …
</**MethodDef**>

How it reads the code

Locate every assignment to the target variable; follow macro calls and %include one level

Summarise the branch logic, not the syntax; keep the line references

Guard rails

VLM granularity anchored on the spec document (distinct inputs based on VSTESTCD).

Never paraphrase a derivation it could not fully trace — emit "REVIEW: partial" instead

Also produced

`def:CommentDef` for dataset-level notes found in program headers

Side-by-side Excel: code excerpt · generated text · reviewer column

Define-XML

5 · Links from Submitted PDF Documents

Bookmarks already exist in the aCRF and the Reviewer's Guide. The utility turns them into leaves, document references and named destinations.

Read the PDFs

`acrf.pdf` — bookmarks by form and by visit (carried over by the migration utility)

`csdrg.pdf` — Reviewer's Guide section headings, dataset tables, issue summaries

Any other leaf: protocol, SAP, external dictionaries

Match to metadata

Bookmark titles ↔ ItemGroupDef names and def:Structure

cSDRG dataset-specific sections (e.g. 4.x) ↔ dataset-level CommentDef

Named destinations preferred over physical pages — they survive a re-paginated PDF

Write links

`def:leaf` with xlink:href and def:title

`def:AnnotatedCRF / def:SupplementalDoc`

`def:DocumentRef` + **`def:PDFPageRef`** `Type="NamedDestination"`

```
<def:leaf ID="LF.csdrg" xlink:href="csdrg.pdf">
  <def:title>Clinical Study Data Reviewer's Guide</def:title></def:leaf>
<def:CommentDef OID="COM.VS">
  <Description><TranslatedText xml:lang="en">See cSDRG 4.4 VS</TranslatedText></Description>
  <def:DocumentRef leafID="LF.csdrg">
    <def:PDFPageRef PageRefs="Sect_4_4_VS" Type="NamedDestination"/>
  </def:DocumentRef></def:CommentDef>
```

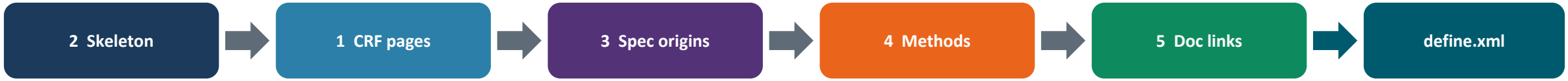
Link check as a by-product

Every generated reference is opened and verified against the PDF before the define is assembled — dangling links are reported, not shipped.

Define-XML

Assembly, Validation and the Human in the Loop

Five small utilities and one merge step. Each intermediate is a file a reviewer can open; the final define.xml is produced last, not first.



1	**Merge in a fixed order**	Skeleton first; every later utility only fills empty slots or appends — it never overwrites a value a human already reviewed
2	**Validate before viewing**	Schema + Define-XML 2.1 conformance rules (P21 / CDISC CORE); render with the stylesheet; every link opened
3	**Review the exceptions, not the file**	Exception lists from utilities 1, 3 and 4 are the review workload; a 400-variable define collapses to a few dozen decisions
4	**Regenerate, don't patch**	A new aCRF, spec or program re-runs the affected utility; the reviewed decisions live in the intermediates and survive the re-run

The point is not that the model writes Define-XML. It is that a domain expert who could not have written five Python utilities can now specify, test and own them.

AI's Role

Sure!?!?!?

Benefits

- **Rethink solutions, processes, skill sets**
  Reduce dependencies, both staff and licensed software
- **Reduced reliance on technical staff/skills**
  Robust requirements are the driver
- **Agile Development**
  Iterate in real-time
- **Reproducibility**
  Lock in capability with extracted code, requirements, prompts

Risks

- **TANSTAAFL**
  AI usage costs rising
- **Dependency**
  Reliance on the model and its surrounding tooling
- **Skill erosion**
  Risks loss of internal technical capabilities
- **Delayed or hidden scale-up costs**
  Governance, auditability, other impacts in moving from prototype to robust solutions

AI's Role

Broader View

Considerations
No Regulatory Precedent An emerging space, with the potential for variability across agencies "Typewriter Clause" vs. the use of non-deterministic solutions
When is it Enough "The difference between art and skill is knowing when to stop" – Ben Affleck on AI This is not just for AI, but for us. The danger of "Sure" and sycophantic AIs

AI Capabilities

AI's Role

Vibe Coding: AI as a Coding Assistant

Coined by Andrej Karpathy in early 2025 — a style of programming where you informally describe the task, and the LLM generates the code

How It Works	Best For	The Output
You describe intent in plain language The model writes the implementation, able to generate a vast swath of code	Prototypes, weekend projects, and one-off tools	The deliverable is code You run it in an IDE or another installed framework to produce a result

NOTE: the generated code is often beyond your available time (or your skills) to critically evaluate.

AI's Role

Scale-Up of Model Capabilities

Agentic use of AI is not new — but it is now built directly into the generally available models

2025 AI as Assistant	2026 AI as Agent
"let me write that code"	*"let me do that for you"*
The model returns code; you wire it up and run it.	The model plans, calls tools, and completes the task.

THE TRAJECTORY

2025 Vibe Coding → **2026** Agentic Automation + Software as a Service

AI's Role

Two Modes: AI as Labor and AI as Service

As capability scales, the role AI plays shifts from doing the work to being the work — a service others consume

AI as Labor

AGENTIC AUTOMATION

- The LLM coordinates tools and transformations to complete a workflow end to end.
- Code becomes secondary — or unnecessary; the model is the worker, not the author.
- Effort shifts from writing software to directing it.

AI as Service

SOFTWARE AS A SERVICE

- Agentic automation is formalized and distributed as a repeatable function or workflow.
- A proven solution is packaged once, then reused across studies and teams.
- Consumers invoke the capability — they do not rebuild it.

AI's Role

AI-Assisted Code Development

Generative AI produced, refined, and tested the code behind this solution. The trade is real capability against real risk

Benefits

- **Speed**
  Hours of manual work compressed into minutes
- **Flexibility**
  Explore and discard approaches at low cost
- **Transparency**
  Every step is inspectable, not hidden in a tool
- **Reproducibility**
  The same prompts and inputs regenerate the same result

Risks

- **Accuracy**
  Generated code can be subtly wrong
- **Dependency**
  Reliance on the model and its surrounding tooling
- **Validation burden**
  AI-written code still requires human review
- **Fragility**
  Correctness depends on consistent input structure

AI's Role

Context Injection: Grounding the Model

A general model knows little about your standards. Supplying the right reference material is what makes its output trustworthy

1 Curate references

- The CDISC Metadata Submission Guide v2.0
- Exemplars
- Structured references (e.g. Excel)

2 Inject as context

Supply materials to the model alongside the task it is asked to perform

3 Ground the output

Responses are anchored to authoritative standards, not the model's untethered priors.

RAG

Retrieval-Augmented Generation. Formalized context can be stored externally and retrieved in real time — so guidance is always current without retraining the model.

AI's Role

A Repeatable Working Process

The same five steps move a one-off prompt toward a reliable, reusable capability.

1	**Identify the ask**	Frame the problem conceptually before writing a single prompt.
2	**Provide references**	Give the model the relevant standards, guides, and worked examples.
3	**Build detailed instructions**	Turn the concept into explicit, testable requirements.
4	**Iterate**	Refine prompts and code until the solution is robust and verified.
5	**Extract & parameterize**	Capture the streamlined prompt, code, and supporting materials for reuse.

Integrating AI into the Submission Pipeline

Recommendations for using AI-assisted automation where conformance and auditability are non-negotiable.

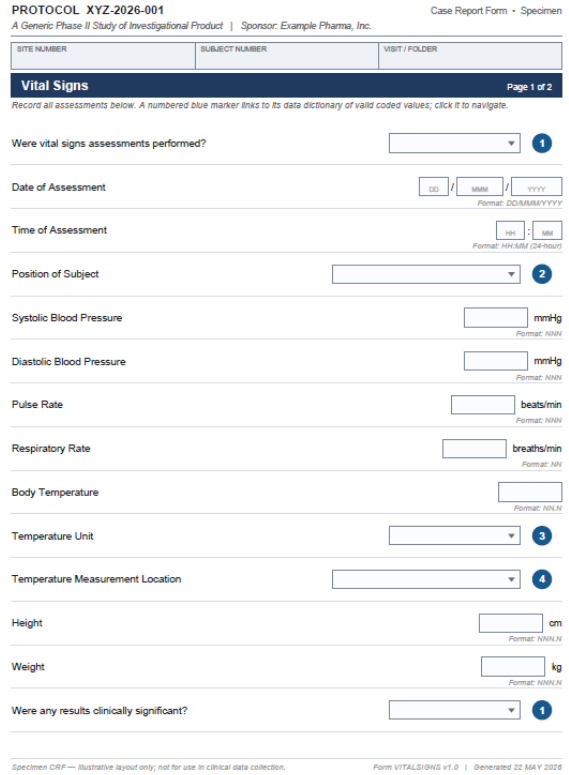
Treat output as a reviewable artifact	Validate AI-generated code and results — never accept a black box.
Keep a human in the loop	Conformance-critical decisions stay with the qualified reviewer.
Version-control prompts and references	Capture the context that produced a result so it can be reproduced.
Standardize the injected context	Shared guides and exemplars make output consistent across studies.
Add verification tooling	Visual checks and version tracking close the loop on every migration.

AI's Role

The Ask

- Give me a sample unannotated aCRF to work from
- Incorporate functionality like RAVE annotations, with the hyperlink to the data dictionary page

Sure



PROTOCOL XYZ-2026-001 — Case Report Form · Specimen

A Generic Phase II Study of Investigational Product | Sponsor: Example Pharma, Inc.

SITE NUMBER	SUBJECT NUMBER	VISIT / FOLDER

Vital Signs — Page 1 of 2

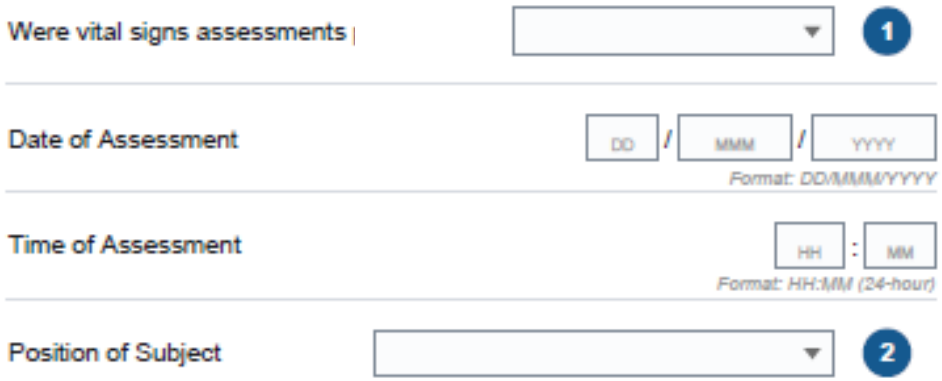
Record all assessments below. A numbered blue marker links to its data dictionary of valid coded values; click it to navigate.

Were vital signs assessments performed?	▾ 1
Date of Assessment	DD / MMM / YYYY — Format: DD/MMM/YYYY
Time of Assessment	HH : MM — Format: HH:MM (24-hour)
Position of Subject	▾ 2
Systolic Blood Pressure	mmHg — Format: NNN
Diastolic Blood Pressure	mmHg — Format: NNN
Pulse Rate	beats/min — Format: NNN
Respiratory Rate	breaths/min — Format: NN
Body Temperature	Format: NN.N
Temperature Unit	▾ 3
Temperature Measurement Location	▾ 4
Height	cm — Format: NNN.N
Weight	kg — Format: NNN.N
Were any results clinically significant?	▾ 1

Specimen CRF — illustrative layout only; not for use in clinical data collection. *Form VITALSIGNS v1.0 | Generated 22 MAY 2026*

AI's Role

- Hyperlinked (for some)
- Source Variable Name (for some)
- Usefully formatted metadata (for some)



1 No Yes (dictionary code: NY)

Referenced by: Vital Signs Performed; Clinically Significant

USER VALUE (displayed)	CODED DATA (stored)
No	N
Yes	Y

2 Subject Position (dictionary code: POSITION)

Referenced by: Position of Subject

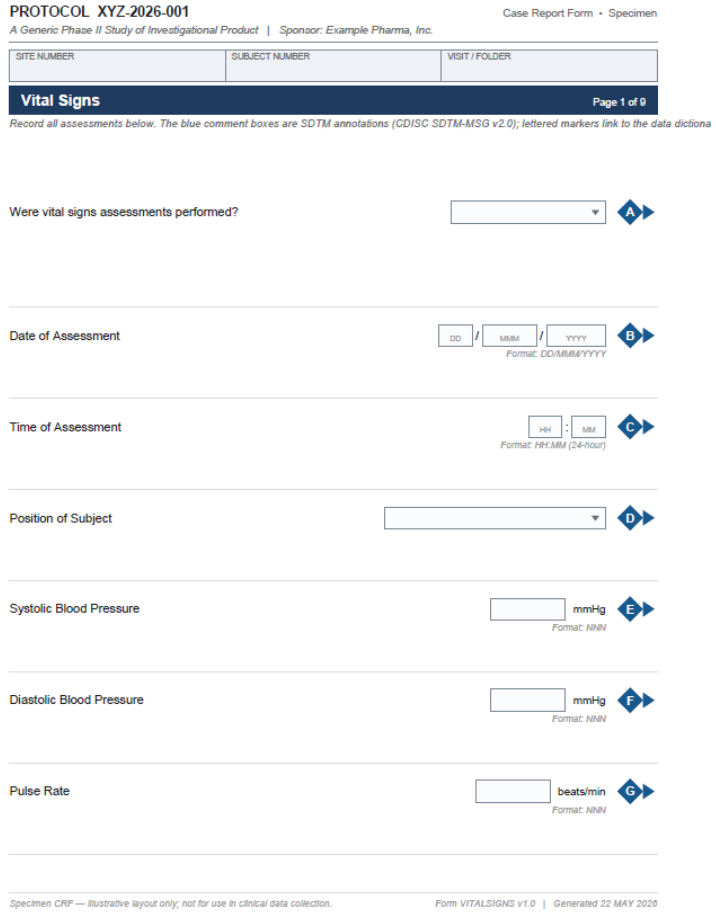
USER VALUE (displayed)	CODED DATA (stored)
Supine	SUPINE
Sitting	SITTING
Standing	STANDING
Semi-Recumbent	SEMI-RECUMBENT

AI's Role

The Ask

- Give me a sample unannotated aCRF to work from
- Incorporate functionality like RAVE annotations, with the hyperlink to the data dictionary page
- Link and full data dictionary for ALL VARIABLES
- Let's jump ahead and for each variable add space for annotations
- Also, lets reformat the links to as to be a bit more distinct…

Sure



PROTOCOL XYZ-2026-001 — Case Report Form - Specimen

A Generic Phase II Study of Investigational Product | Sponsor: Example Pharma, Inc.

SITE NUMBER	SUBJECT NUMBER	VISIT / FOLDER

Vital Signs — Page 1 of 9

Record all assessments below. The blue comment boxes are SDTM annotations (CDISC SDTM-MSG v2.0); lettered markers link to the data dictiona

Were vital signs assessments performed? [ ] A

Date of Assessment [DD] / [MMM] / [YYYY] B — Format: DD/MMM/YYYY

Time of Assessment [HH] : [MM] C — Format: HH:MM (24-hour)

Position of Subject [ ] D

Systolic Blood Pressure [ ] mmHg E — Format: NNN

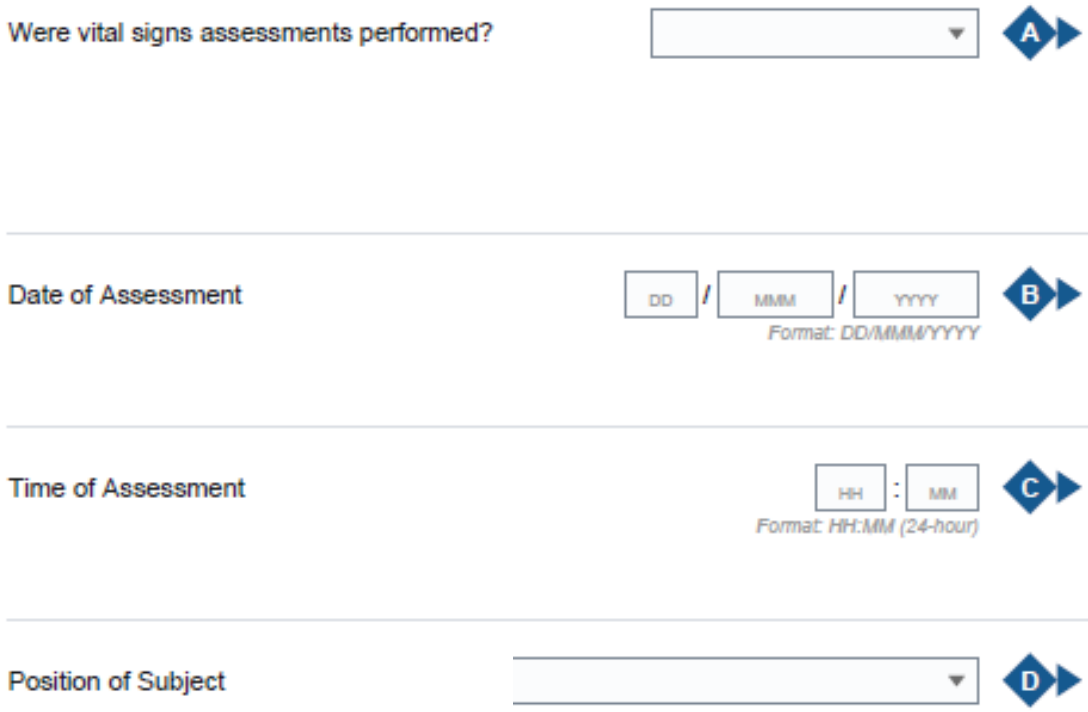
Diastolic Blood Pressure [ ] mmHg F — Format: NNN

Pulse Rate [ ] beats/min G — Format: NNN

Specimen CRF — Illustrative layout only; not for use in clinical data collection. *Form VITALSIGNS v1.0 | Generated 22 MAY 2026*

AI's Role

- Cleaned up… maybe a bit too far



Field A — Vital Signs Performed

Field specification and attributes

ATTRIBUTE	SPECIFICATION
SDTM Variable	VSSTAT
SDTM Label	Completion Status
SDTM Annotation	[NOT SUBMITTED]; VSSTAT = NOT DONE when VSTESTCD = VSALL
Origin	Derived (VSSTAT); prompt response not submitted
Data Type	text
Length / Format	8
Controlled Terminology	No Yes (NY) — prompt response
Units	—
Souce Variable	VSYN

Controlled terminology — No Yes (NY)

USER VALUE (displayed)	CODED DATA (stored)
No	N
Yes	Y

Field B — Date of Assessment

Field specification and attributes

ATTRIBUTE	SPECIFICATION
SDTM Variable	VSDTC
SDTM Label	Date/Time of Measurements
SDTM Annotation	VSDTC
Origin	Collected
Data Type	date (ISO 8601)
Length / Format	10 — collected as DD/MMM/YYYY
Controlled Terminology	—
Units	—
Source Variable	VSDAT

AI's Role

The Ask

- Annotate with SDTM mappings…

Sure



PROTOCOL XYZ-2026-001 — Case Report Form • Specimen

A Generic Phase II Study of Investigational Product | Sponsor: Example Pharma, Inc.

SITE NUMBER | SUBJECT NUMBER | VISIT / FOLDER

Vital Signs — Page 1 of 9

Record all assessments below. The blue comment boxes are SDTM annotations (CDISC SDTM-MSG v2.0); lettered markers link to the data dictionary.

VS (Vital Signs)

Were vital signs assessments performed? (A)

[NOT SUBMITTED]

VSSTAT = NOT DONE when VSTESTCD = VSALL

Date of Assessment (B) — DD / MMM / YYYY — *Format: DD/MMM/YYYY*

VSDTC

Time of Assessment (C) — HH : MM — *Format: HH:MM (24-hour)*

VSDTC

Position of Subject (D)

VSPOS

Systolic Blood Pressure (E) — mmHg — *Format: NNN*

VSORRES / VSORRESU when VSTESTCD = SYSBP

Diastolic Blood Pressure (F) — mmHg — *Format: NNN*

VSORRES / VSORRESU when VSTESTCD = DIABP

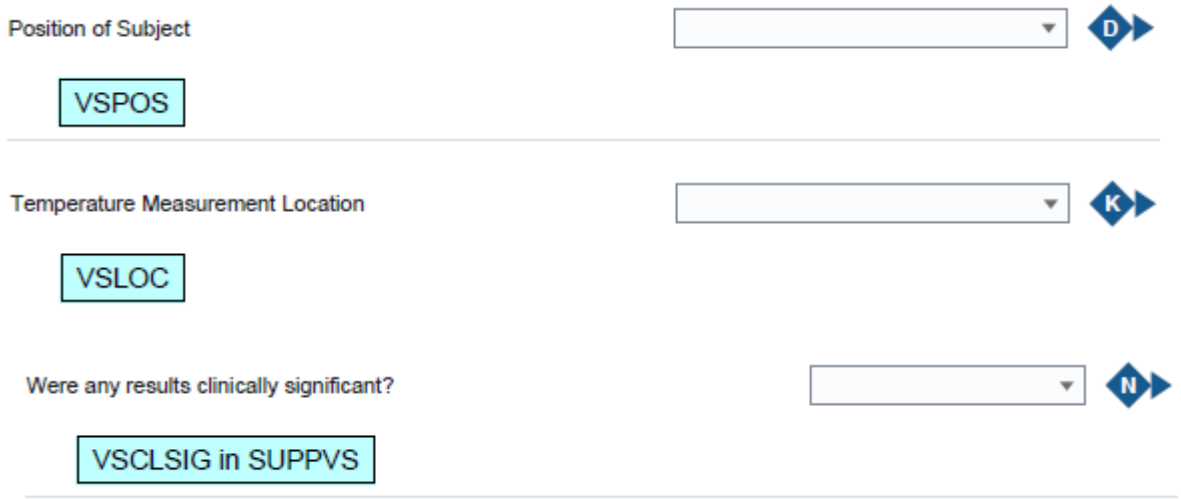
Pulse Rate (G) — beats/min — *Format: NNN*

VSORRES / VSORRESU when VSTESTCD = PULSE

Specimen CRF — Illustrative layout only; not for use in clinical data collection. *Form VITALSIGNS v1.0 | Generated 22 MAY 2026*

AI's Role

- Pretty good, room for improvement
- Position and location need to be anchored to a specific test
- Clinical Significance
  - Is a standard variable, not SUPP
  - Should be anchored to a specific test, probably derived (sub-optimal CRF design)

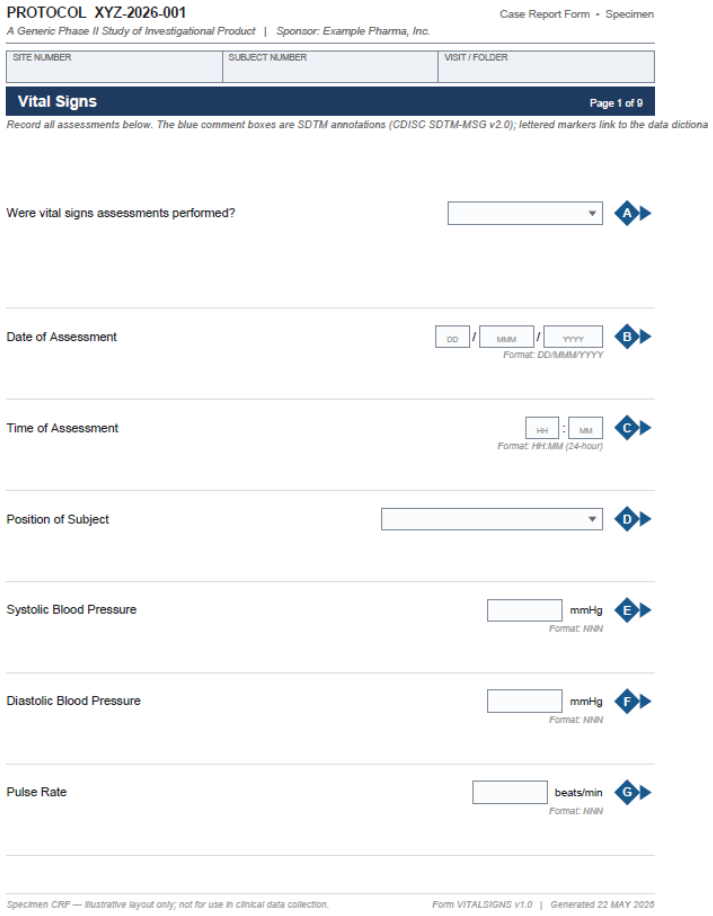


AI's Role

The Ask

- I've provided the metadata reference that links the form name to the source dataset name
- Place source dataset names in the top right corner
- Place source variable names (follow the hyperlink to determine)

Sure



PROTOCOL XYZ-2026-001 — Case Report Form • Specimen

A Generic Phase II Study of Investigational Product | Sponsor: Example Pharma, Inc.

SITE NUMBER	SUBJECT NUMBER	VISIT / FOLDER

Vital Signs — Page 1 of 9

Record all assessments below. The blue comment boxes are SDTM annotations (CDISC SDTM-MSG v2.0); lettered markers link to the data dictiona

Were vital signs assessments performed? [ ] A

Date of Assessment [DD] / [MMM] / [YYYY] B — Format: DD/MMM/YYYY

Time of Assessment [HH] : [MM] C — Format: HH:MM (24-hour)

Position of Subject [ ] D

Systolic Blood Pressure [ ] mmHg E — Format: NNN

Diastolic Blood Pressure [ ] mmHg F — Format: NNN

Pulse Rate [ ] beats/min G — Format: NNN

Specimen CRF — illustrative layout only; not for use in clinical data collection. *Form VITALSIGNS v1.0 | Generated 22 MAY 2026*

AI's Role

The Ask

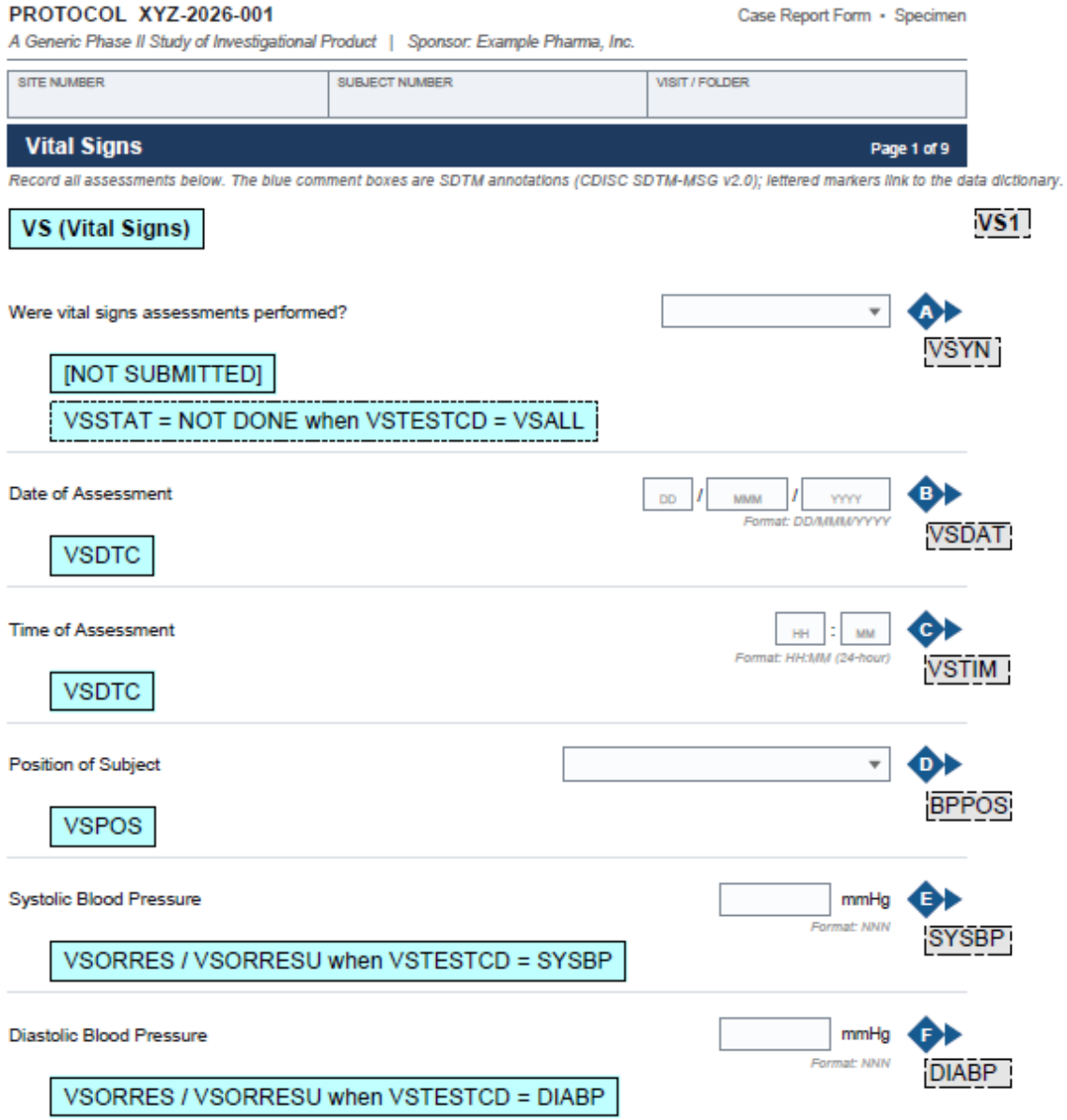
- I've provided the metadata reference that links the form name to the source dataset name
- Place source dataset names in the top right corner
  - Use a gray background, and a dot-dashed border, make bold

The Ask

- Place source variable names below the hyperlink
  - Follow the hyperlink to determine the name
  - Use a gray background and a dot-dashed border

AI's Role

Sure



PROTOCOL XYZ-2026-001 — Case Report Form • Specimen

A Generic Phase II Study of Investigational Product | Sponsor: Example Pharma, Inc.

SITE NUMBER	SUBJECT NUMBER	VISIT / FOLDER

Vital Signs — Page 1 of 9

Record all assessments below. The blue comment boxes are SDTM annotations (CDISC SDTM-MSG v2.0); lettered markers link to the data dictionary.

VS (Vital Signs) — VS1

Were vital signs assessments performed? — A — VSYN

[NOT SUBMITTED]

VSSTAT = NOT DONE when VSTESTCD = VSALL

Date of Assessment — DD / MMM / YYYY — *Format: DD/MMM/YYYY* — B — VSDAT

VSDTC

Time of Assessment — HH : MM — *Format: HH:MM (24-hour)* — C — VSTIM

VSDTC

Position of Subject — D — BPPOS

VSPOS

Systolic Blood Pressure — mmHg — *Format: NNN* — E — SYSBP

VSORRES / VSORRESU when VSTESTCD = SYSBP

Diastolic Blood Pressure — mmHg — *Format: NNN* — F — DIABP

VSORRES / VSORRESU when VSTESTCD = DIABP

AI's Role

The Ask

- Extract the annotation page numbers into Excel, per domain based on linking by color
- Compare the annotations against
  - Specifications
  - Programming
- Identify any unannotated fields
  - Give me a list
  - place suggested annotations on the CRF, formatted to note their status

The Ask

- Generate the Define-XML
  - Here's the source metadata, SDTM programs and the aCRF
  - Don't forget the CDISC terminology reference, code tables, biomedical concepts, and specializations
  - Include the appropriate C-Codes
  - Determine VLM

AI's Role

Sure!!!!

AI's Role

Sure!?!?!?

Benefits

- **Rethink solutions, processes, skill sets**
  Reduce dependencies, both staff and licensed software
- **Reduced reliance on technical staff/skills**
  Robust requirements are the driver
- **Agile Development**
  Iterate in real-time
- **Reproducibility**
  Lock in capability with extracted code, requirements, prompts

Risks

- **TANSTAAFL**
  AI usage costs rising
- **Dependency**
  Reliance on the model and its surrounding tooling
- **Skill erosion**
  Risks loss of internal technical capabilities
- **Delayed or hidden scale-up costs**
  Governance, auditability, other impacts in moving from prototype to robust solutions

AI's Role

Broader View

Considerations
No Regulatory Precedent An emerging space, with the potential for variability across agencies "Typewriter Clause" vs. the use of non-deterministic solutions
When is it Enough "The difference between art and skill is knowing when to stop" – Ben Affleck on AI This is not just for AI, but for us. The danger of "Sure" and sycophantic AIs

Ain’t it tempting?



Thank You!

Name: Carlo Radovsky
Organization: Immanant
E-mail: Carlo.Radovsky@immanant.com

