

Presenters

Ben Straub
GSK, RCSWG



Sam Parmar
Pfizer, RCSWG



Nick Masel
J&J, RCSWG, {datasetjson}



Breaking Free from the XPT

Exploration of Dataset-JSON as an Alternative Transport File to Regulatory Agencies

Ben Straub (GSK) Sam Parmar (Pfizer) Nick Masei (Johnson & Johnson)

Agenda

This talk covers three topics:

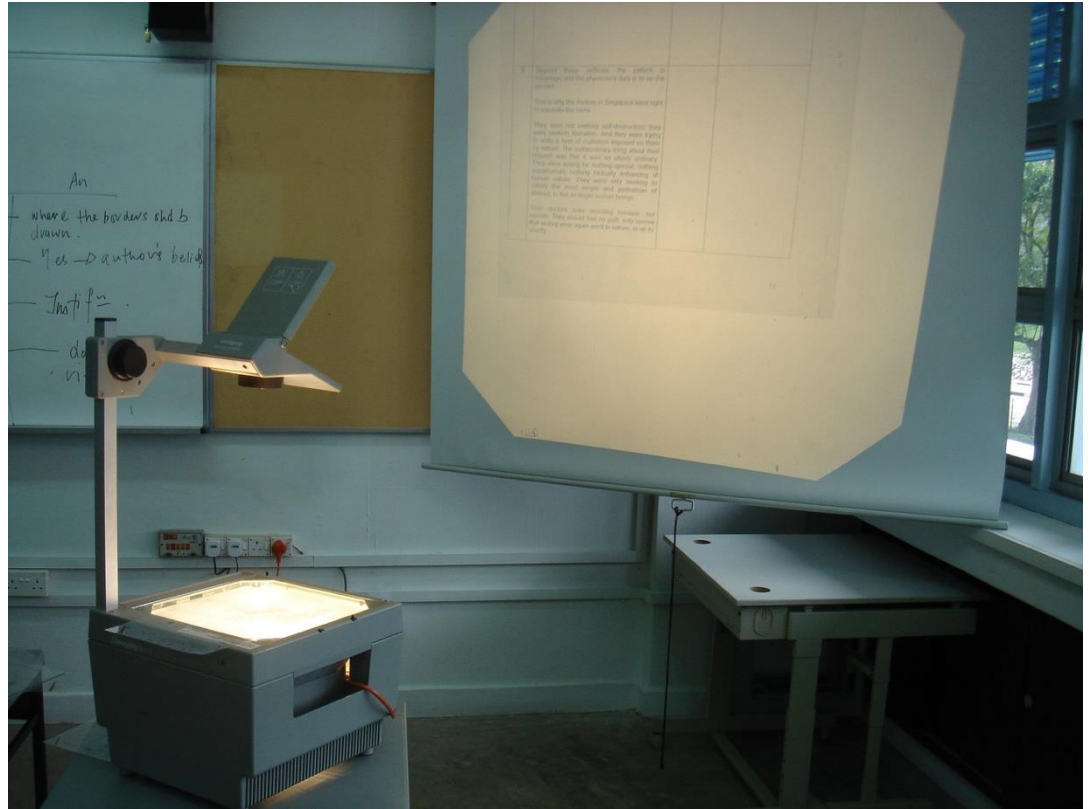
- Why the industry is rethinking XPT as the default transport file?
- What Dataset-JSON is, where it came from, and why it matters?
- The R Consortium R Submissions Working Group and Pilot 5

**1980s - A snapshot of
technology**

A time of bliss



A time of wonder



A time of memory



A time of xpt



Motivation: Why Talk About Transport Formats?

- All of these 1980s technologies have been replaced - except the XPT.
- XPTs has been the de facto submission transport for decades
- XPT works – but reflects 1980s constraints, not modern data practice
- Open-source tooling and modern standards are now mainstream
- Dataset-JSON offers a contemporary alternative

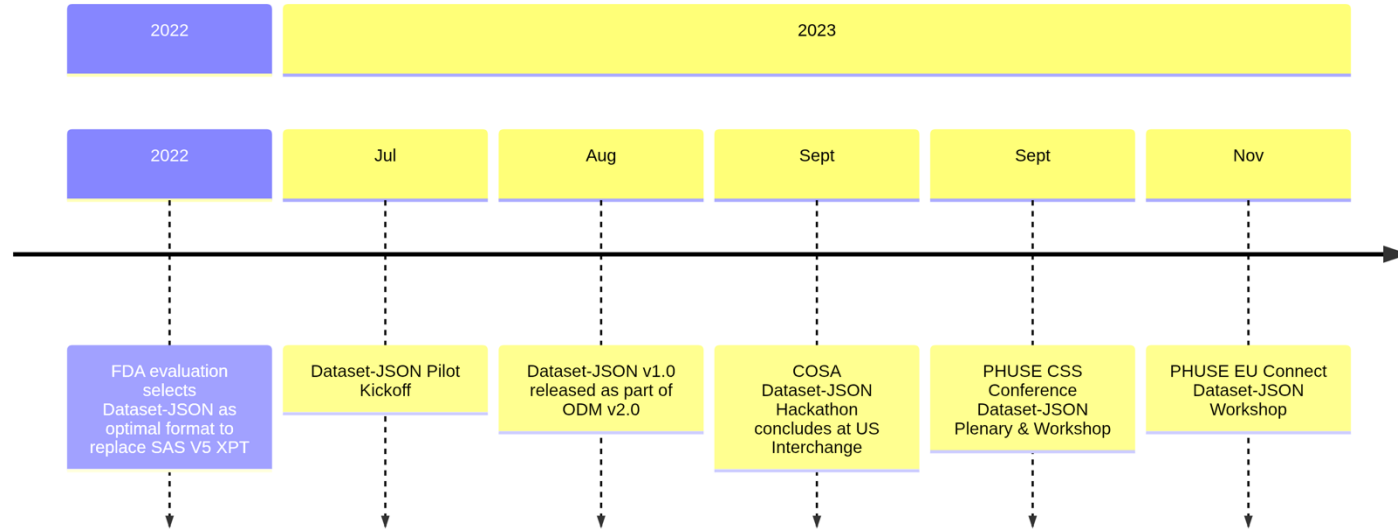
XPT: Yesterday's Solution to Yesterday's Problem

- **Outdated technology**
 - Defined in the 1980s for SAS-to-SAS transfer
 - Optimized for mainframes, not for modern APIs or big data
- **Constrained data characteristics**
 - 8-character variable names, 40-character labels
 - 200-character limit for character values
 - Limited data types
- **Poor handling of internationalization**
 - US-ASCII only; no native UTF-8/multibyte support

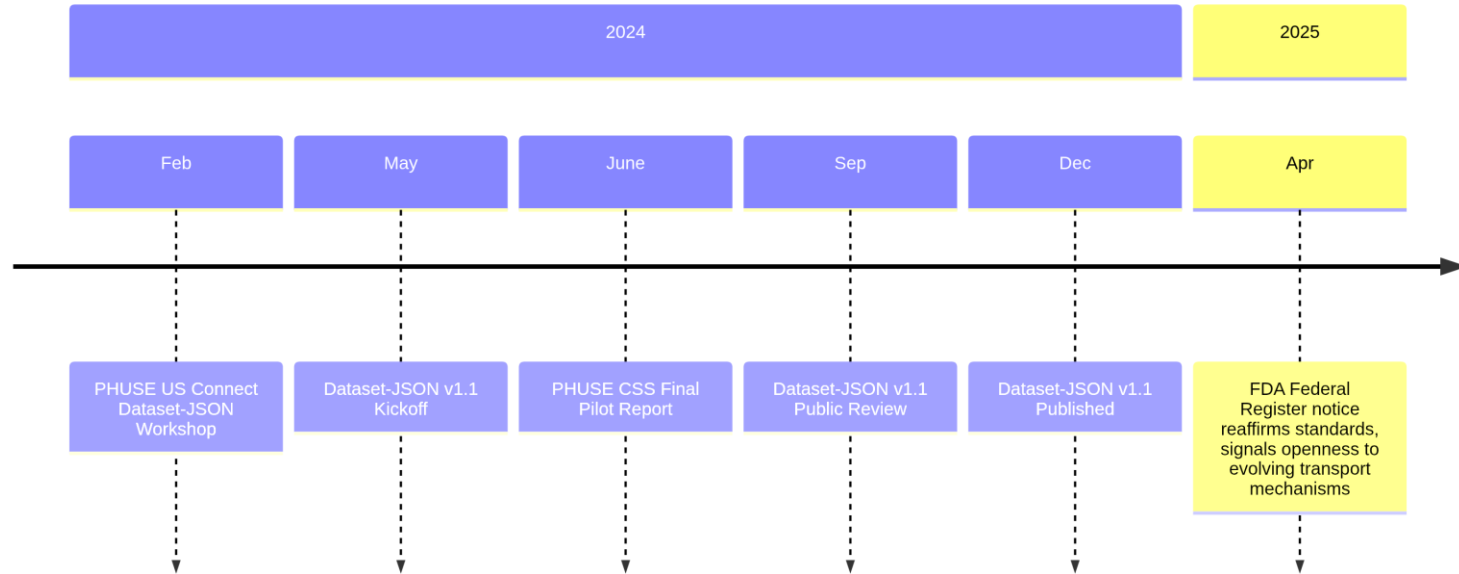
Why XPT Creates Friction Today

- **Weak embedded metadata**
 - No rich metadata inside the file
 - Reliance on external Define-XML; risk of misalignment
- **Inefficient storage**
 - Fixed-width allocations, lots of padding
 - No compression support
- **Ecosystem asymmetry**
 - Native in SAS; awkward in open-source and data engineering stacks

Dataset-JSON Timeline: 2022 - 2023



Dataset-JSON Timeline: 2024 - 2025



Why Dataset-JSON?

- **Self-contained metadata**
 - Variable labels/types and CDISC metadata embedded in the file
 - Designed to meet regulatory needs
- **Human-readable and audit-friendly**
 - Plain-text JSON; easy to inspect and search
- **Universally parseable**
 - Mature JSON parsers exist in all major languages
- **Supports APIs**
 - Same structure works for file transfer and REST endpoints
- **Interoperability**

Dataset-JSON in Practice

- One dataset per file
- Three main components:
 - **Top-level metadata**
 - Creation time, standard version
 - Optional link to Define-XML
 - **Column metadata**
 - Names, labels, data types, optional CDISC-specific attributes
 - **Row data**
 - Arrays of values; missing values as `null`
- Special handling:
 - Decimals as strings to preserve precision
 - ISO 8601 dates/times
 - Optional NDJSON variant for streaming large datasets

Regulatory and Collaboration Context

- Dataset-JSON sits in a collaborative ecosystem:
 - **CDISC**: defines standards
 - **PHUSE**: applied use cases and community practice
 - **FDA**: evaluates review impact and feasibility
 - **R Consortium Submission Working Group**: FDA–industry collaboration testing out R-based submissions.

R Consortium R Submissions Working Group and Pilot 5

R Consortium R Submissions Working Group

- Mission: demonstrate that **R-based submission packages** can:
 - Meet regulatory expectations
 - Be fully reproducible and reviewable
 - Experiment today to prepare for tomorrow's submissions
- Focus on:
 - Complete, public submission-like artifacts (data, code, outputs)
 - Practical workflows others can fork and adapt
 - Feedback loop with regulators to shape future processes

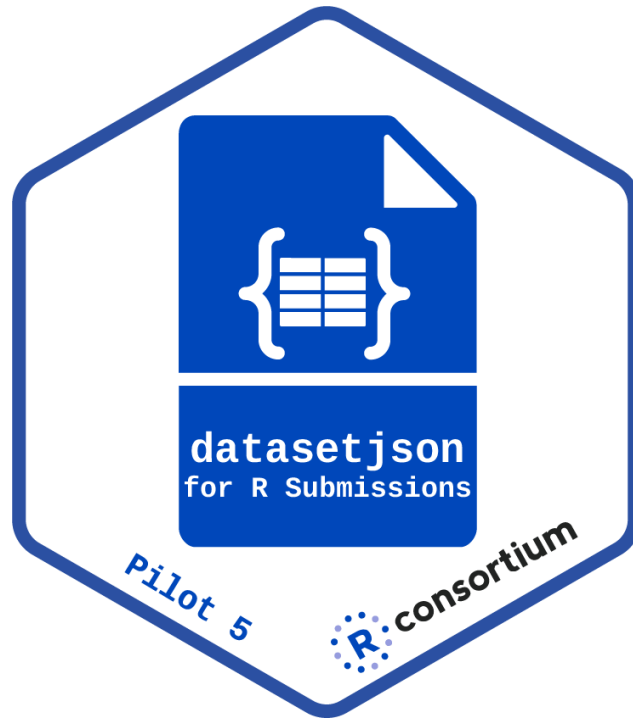
Pilots 1 - 4

- Evolution:
 - Pilot 1: TLFs in R, SDTM/ADaM from SAS XPT
 - Pilot 2: TLFs are packaged into a Shiny App
 - Pilot 3: Rebuilt 5 ADaM datasets in R, still output XPT
 - Pilot 4: Pilot 2 delivered as webassembly and as a container

What the Pilots Showed

- Demonstrated:
 - eCTD-style packaging and conventions
 - Reproducible execution from source data through TLFs
 - Reviewer-aligned documentation (ADRG) using open tools (e.g., Quarto)
 - Concrete interaction with FDA reviewers:
 - Installation, environment setup
 - Data and derivation questions
 - Pilots received a letter from FDA called “Statistical Review and Evaluation” which documented success of Pilot as public evidence.

Pilot 5



Pilot 5 Focus: Dataset-JSON + R-Generated ADaM

- Objectives:
 - Convert *all* XPTs in the package to Dataset-JSON
 - Build off Pilot 1 and Pilot 3
 - Deliver a publicly accessible R-based submission-like package
 - Use **Dataset-JSON v1.1** as the dataset transport format
 - Generate key ADaM datasets in R (not SAS) and ...
 - SDTM also regenerated as datasetjson but no program available
 - Use the [datasetjson](#) R package to do the heavy lifting

Pilot 5 Artifacts (What You Can See Today)

- [Public repository](#) for development work
 - R programs
 - Data: SDTM and ADaM
 - QC Reports from Bots
 - Closed Pull Requests with Discussion
 - Meeting Notes and Development ADRG (LLM-supported)
- [Public repository](#) for final submission work
 - Following M1/M5 structure
 - Final R programs
 - Final JSON data
 - Final ADRG and other supporting documents

Way of Working: Reproducibility-First

- Organized to be **re-runnable and inspectable**:
 - Programs: [pilot5-submission/pilot5-programs/](#)
 - e.g., `adsl.r`, `adae.r`, `adlbc.r`, `adtte.r`
 - Outputs: [pilot5-submission/pilot5-output/](#)
 - Documentation: ADRG + eCTD-style README
- Comparability treated as a first-class artifact:
 - [qcReport.qmd](#): dataset-to-dataset QC using `diffdf`
 - [tlf-qc.qmd](#): text and graphical diffing of outputs

Where AI Helped (But Didn't Replace Authors)

- Pilot 5 experimented with LLM-assisted documentation:
 - [adrg/llm-adrg-utils/llm_pipeline.qmd](#)
- Use cases:
 - Extract variables, datasets, filters from analysis code
 - Pre-populate reviewer-facing tables in ADRG
- Intent:
 - **Reduce manual effort and inconsistencies**
 - Keep humans firmly in the authoring and review loop

Challenges Faced

External Dependencies & Evolution

- Dataset-JSON moved from v1.0 to v1.1 during pilot
- Needed to understand and document v1.1 changes with FDA
- Pinnacle 21 Community did not yet support Dataset-JSON v1.1
- Required:
 - Workarounds and additional documentation
 - Direct discussion with FDA collaborators
 - Re-submission once P21 supports v1.1

Implementation & Tooling Gaps

- Learning how metadata is represented and persisted in Dataset-JSON
- Managing numeric precision:
 - Converting float variables to decimal data type in JSON
 - Building wrapper functions for consistent handling across datasets
- Package limitations:
 - [datasetjson](#) R package has a few open issues
 - Tracked and discussed publicly:
 - <https://github.com/atorus-research/datasetjson/issues>

What Pilot 5 Demonstrates

- Dataset-JSON is a **viable transport** for submission-like packages:
 - Produced fully in an R-driven pipeline
 - With explicit metadata management
- A reviewer-friendly, eCTD-style package can use Dataset-JSON instead of XPT
- Outputs (TLFs) can be regenerated from provided code and datasets
- QC comparisons:
 - Dataset-level and output-level
 - Scripted and reported for transparency

datasetjson R package Workhorse of Pilot 5



The Whole Game

Let's see the complete workflow with [{pharmaverseadam}](#) data:

1. **Start** with ADAM datasets
2. **Prepare** top-level metadata and column metadata
3. **Convert** to Dataset-JSON
4. **Share** the standardized file
5. **Read** it back anywhere

Step 1: Load ADAM Data and Metadata

```
library(datasetjson)
```

```
# Load data
```

```
adsl <- readRDS("data/adsl.rds")
```

Step 1a: Examine What We Have - ADSL

```
knitr::kable(head(adsl[1:6]),  
format = "html", table.attr =  
"style='font-size: 24px;'")
```

STUDYID	USUBJID	SUBJID	RFSTDTC	RFENDTC	RFXSTDT C
CDISCPIL OT01	01-701- 1015	1015	2014-01- 02	2014-07- 02	2014-01- 02
CDISCPIL OT01	01-701- 1023	1023	2012-08- 05	2012-09- 02	2012-08- 05
CDISCPIL OT01	01-701- 1028	1028	2013-07- 19	2014-01- 14	2013-07- 19
CDISCPIL OT01	01-701- 1033	1033	2014-03- 18	2014-04- 14	2014-03- 18
CDISCPIL OT01	01-701- 1034	1034	2014-07- 01	2014-12- 30	2014-07- 01
CDISCPIL OT01	01-701- 1047	1047	2013-02- 12	2013-03- 29	2013-02- 12

Step 2: Prepare Metadata

```
# Load column metadata  
adsl_meta <- readRDS("data/adsl_meta.rds")
```

Step 2a: Examine What We Have - adsl_meta

```
knitr::kable(head(adsl_meta), format = "html", table.attr = "style='font-size: 24px;'")
```

itemOID	name	label	dataType	targetDataType	length
STUDYID	STUDYID	Study Identifier	string	NA	12
USUBJID	USUBJID	Unique Subject Identifier	string	NA	11
SUBJID	SUBJID	Subject Identifier for the Study	string	NA	4
RFSTDTC	RFSTDTC	Subject Reference Start Date/Time	string	NA	NA
RFENDTC	RFENDTC	Subject Reference End Date/Time	string	NA	NA
RFXSTDTC	RFXSTDTC	Date/Time of First Study Treatment	string	NA	NA

Step 3: Create Dataset-JSON

```
# Convert ADSL to Dataset-JSON
## top-level metadata
## column/variable metadata
## rows
adsl_json <- dataset_json(
  adsl,
  name = "ADSL",
  dataset_label = "Subject Level Analysis
Dataset",
  columns = adsl_meta
)
```

CDISC Dataset-JSON v1.1 Specification

Step 3a: Inspect Data set-JSON Object

```
head(adsl_json)
  STUDYID   USUBJID SUBJID   RFSTDTC   RFENDTC   RFXSTDTC   RFXENDTC
1 CDISCPILOT01 01-701-1015      1015 2014-01-02 2014-07-02 2014-01-02 2014-07-02
2 CDISCPILOT01 01-701-1023      1023 2012-08-05 2012-09-02 2012-08-05 2012-09-01
3 CDISCPILOT01 01-701-1028      1028 2013-07-19 2014-01-14 2013-07-19 2014-01-14
4 CDISCPILOT01 01-701-1033      1033 2014-03-18 2014-04-14 2014-03-18 2014-03-31
5 CDISCPILOT01 01-701-1034      1034 2014-07-01 2014-12-30 2014-07-01 2014-12-30
6 CDISCPILOT01 01-701-1047      1047 2013-02-12 2013-03-29 2013-02-12 2013-03-09
  RFICDTC   RFPENDTC DTHDTC DTHFL SITEID AGE  AGEU SEX  RACE
1 <NA> 2014-07-02T11:45 <NA> <NA> 701 63 YEARS F WHITE
2 <NA> 2013-02-18 <NA> <NA> 701 64 YEARS M WHITE
3 <NA> 2014-01-14T11:10 <NA> <NA> 701 71 YEARS M WHITE
4 <NA> 2014-09-15 <NA> <NA> 701 74 YEARS M WHITE
5 <NA> 2014-12-30T09:50 <NA> <NA> 701 77 YEARS F WHITE
6 <NA> 2013-07-28 <NA> <NA> 701 85 YEARS F WHITE
  ETHNIC ARMCD ARM ACTARMCD
1 HISPANIC OR LATINO Pbo Placebo Pbo
2 HISPANIC OR LATINO Pbo Placebo Pbo
3 NOT HISPANIC OR LATINO Xan_Hi Xanomeline High Dose Xan_Hi
4 NOT HISPANIC OR LATINO Xan_Lo Xanomeline Low Dose Xan_Lo
5 NOT HISPANIC OR LATINO Xan_Hi Xanomeline High Dose Xan_Hi
6 NOT HISPANIC OR LATINO Pbo Placebo Pbo
  ACTARM COUNTRY DMDTC DMDY TRT01P
1 Placebo USA 2013-12-26 -7 Placebo
2 Placebo USA 2012-07-22 -14 Placebo
3 Xanomeline High Dose USA 2013-07-11 -8 Xanomeline High Dose
4 Xanomeline Low Dose USA 2014-03-10 -8 Xanomeline Low Dose
5 Xanomeline High Dose USA 2014-06-24 -7 Xanomeline High Dose
6 Placebo USA 2013-01-22 -21 Placebo
  TRT01A TRTSDTM TRTSTMF TRTEDTM TRTETMF
1 Placebo 2014-01-02 H 2014-07-02 23:59:59 H
2 Placebo 2012-08-05 H 2012-09-01 23:59:59 H
3 Xanomeline High Dose 2013-07-19 H 2014-01-14 23:59:59 H
4 Xanomeline Low Dose 2014-03-18 H 2014-03-31 23:59:59 H
5 Xanomeline High Dose 2014-07-01 H 2014-12-30 23:59:59 H
6 Placebo 2013-02-12 H 2013-03-09 23:59:59 H
```

Step 3b: Inspect Dataset-JSON Object Attributes

class, top-level, column

```
str(attributes(adsl_json))
List of 8
 $ names          : chr [1:54] "STUDYID" "USUBJID" "SUBJID" "RFSTDTC" ...
 $ row.names      : int [1:306] 1 2 3 4 5 6 7 8 9 10 ...
 $ label          : chr "Subject Level Analysis Dataset"
 $ class          : chr [1:3] "datasetjson_v1_1_0" "datasetjson" "data.frame"
 $ _xporttr.df_arg_ : chr "ADSL"
 $ datasetJSONVersion: chr "1.1.0"
 $ name           : chr "ADSL"
 $ columns        :List of 54
 ..$ :List of 5
 .. ..$ itemOID : chr "STUDYID"
 .. ..$ name    : chr "STUDYID"
 .. ..$ label   : chr "Study Identifier"
 .. ..$ dataType: chr "string"
 .. ..$ length  : int 12
 ..$ :List of 5
 .. ..$ itemOID : chr "USUBJID"
 .. ..$ name    : chr "USUBJID"
 .. ..$ label   : chr "Unique Subject Identifier"
 .. ..$ dataType: chr "string"
 .. ..$ length  : int 11
 ..$ :List of 5
 .. ..$ itemOID : chr "SUBJID"
 .. ..$ name    : chr "SUBJID"
 .. ..$ label   : chr "Subject Identifier for the Study"
 .. ..$ dataType: chr "string"
```

Step 4: Share Standardized Files

```
# Write to Dataset-JSON files  
write_dataset_json(adsl_json,  
"output/ADSL.json", pretty = TRUE) # use  
pretty when viewing
```

Now anyone can read these files!

- Regulatory reviewers
- External collaborators
- Different software (R, Python, SAS)

Step 5: Read Back Anywhere

```
# Read back - gets original data +  
metadata  
adsl_restored <-  
read_dataset_json("output/ADSL.json")
```

Step 5b: Inspect Dataset-JSON Object Attributes

class, top-level, column

```
str(attributes(adsl_restored))
List of 9
 $ names                : chr [1:54] "STUDYID" "USUBJID" "SUBJID" "RFSTDTC" ...
 $ row.names            : int [1:306] 1 2 3 4 5 6 7 8 9 10 ...
 $ class                : chr [1:3] "datasetjson_v1_1_0" "datasetjson" "data.frame"
 $ datasetJSONVersion   : chr "1.1.0"
 $ name                 : chr "ADSL"
 $ label                : chr "Subject Level Analysis Dataset"
 $ columns               :List of 54
 ..$ :List of 5
 .. ..$ itemOID : chr "STUDYID"
 .. ..$ name    : chr "STUDYID"
 .. ..$ label   : chr "Study Identifier"
 .. ..$ dataType: chr "string"
 .. ..$ length  : int 12
 ..$ :List of 5
 .. ..$ itemOID : chr "USUBJID"
 .. ..$ name    : chr "USUBJID"
 .. ..$ label   : chr "Unique Subject Identifier"
 .. ..$ dataType: chr "string"
 .. ..$ length  : int 11
 ..$ :List of 5
 .. ..$ itemOID : chr "SUBJID"
 .. ..$ name    : chr "SUBJID"
 .. ..$ label   : chr "Subject Identifier for the Study"
 .. ..$ dataType: chr "string"
 .. ..$ length  : int 4
```

Takeaways and Next Steps

- XPT served us well but is misaligned with:
 - Modern open-source ecosystems
 - Rich metadata and internationalization needs
 - Text-based, automated, reproducible workflows
- Dataset-JSON offers:
 - Self-contained, human-readable, API-ready transport
 - Stronger fit with open-source ecosystems
- Pilot 5:
 - Provides a public blueprint you can adapt
 - Invites further experimentation and feedback with regulators
- Call to action:
 - Explore the Pilot 5 repo
 - Try Dataset-JSON in your internal workflows
 - Engage via PHUSE / CDISC / R Consortium channels

References & Contact

Selected references

- Pilot 5 repo: <https://github.com/RConsortium/submissions-pilot5-datasetjson>
- eCTD overview: <https://www.fda.gov/drugs/electronic-common-technical-document-ectd>
- Dataset-JSON v1.1: <https://cdisc-org.github.io/DataExchange-DatasetJson/doc/dataset-json1-1.html>

Authors

- Ben Straub, GSK – Philadelphia, United States
- Sam Parmar, Pfizer – New York City, United States
- Nick Masel, Johnson & Johnson – Raleigh, United States