Paper CT07

SAS to R: Opportunities and Challenges in SDTM and ADaM Transformation

Jalendhar Ettedi, Atorus Research, Bangalore, India
Padmashree Jahagirdar, Atorus Research, Bangalore, India

ABSTRACT

The clinical trial programming ecosystem is evolving as organizations seek flexible, cost-effective, and reproducible alternatives to traditional SAS-based workflows. R has emerged as a credible option for regulatory-compliant implementation of CDISC SDTM and ADaM standards, supported by a rapidly maturing pharmaverse ecosystem. This conference paper presents practical experiences in transitioning from SAS to R, highlighting workflow changes, validated packages, and real-world derivation examples. We demonstrate how R enables metadata-driven, transparent, and reproducible pipelines while meeting regulatory expectations, and we discuss challenges and mitigation strategies encountered during adoption.

INTRODUCTION

For decades, SAS has been the dominant platform for clinical trial data programming, particularly for the creation of CDISC-compliant SDTM and ADaM datasets. However, increasing licensing costs, limited extensibility, and macro-heavy programming models have prompted organizations to evaluate open-source alternatives. R, with its expressive syntax and strong community support, has gained significant traction in regulated clinical environments.

This paper focuses on a practical comparison of SAS and R workflows for SDTM and ADaM development, demonstrating how validated R packages can support production-grade outputs suitable for regulatory submission.

WHY SAS TO R?

The primary drivers for adopting R include its open-source nature, elimination of licensing costs, and strong support for reproducible research.

KEY ADVANTAGES

- Open-source with no licensing costs
- Strong, readable data manipulation via the dplyr and tidyverse
- Metadata-driven programming using {metacore} and {xportr}
- Growing pharmaverse ecosystem with validated clinical packages
- Highly reproducible and modular pipelines

KEY CHALLENGES

- Transition from macro-heavy SAS workflows to functional R paradigms
- Requirement for formal validation frameworks
- Building sponsor and regulatory confidence

R ECOSYSTEM FOR CLINICAL PROGRAMMING

A typical R-based clinical programming environment includes below core packages and it is depicted by Figure 1:

- **envsetup**: Standardizes project structure and R environments
- **dplyr / tidyr**: Core data manipulation and reshaping
- **metacore / metatools**: Centralized CDISC metadata handling
- **admiral**: Validated ADaM derivation functions
- **lubridate**: Robust date and time handling
- **xportr**: Application of CDISC metadata and FDA-ready XPT export
- **readxl**: Reading Excel-based specifications
- **diffdf**: Audit-ready dataset comparisons for QC

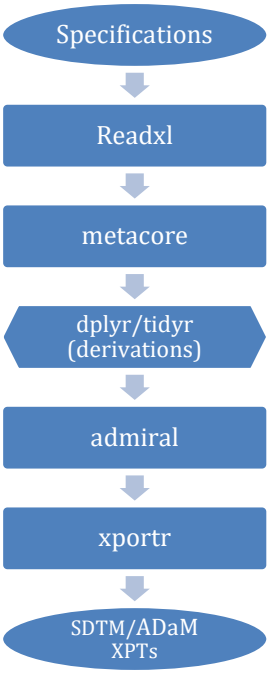

Figure 1: R based Clinical Programming Ecosystem

SDTM IMPLEMENTATION IN R

WORKFLOW COMPARISON

In SAS, SDTM derivations typically rely on DATA steps, PROC SQL, custom macros, and manual metadata assignments. In contrast, R workflows use tidyverse pipelines for transformations, metacore for structured metadata, and xportr for applying labels, controlled terminology, and variable attributes. This is depicted in Figure 2 below.

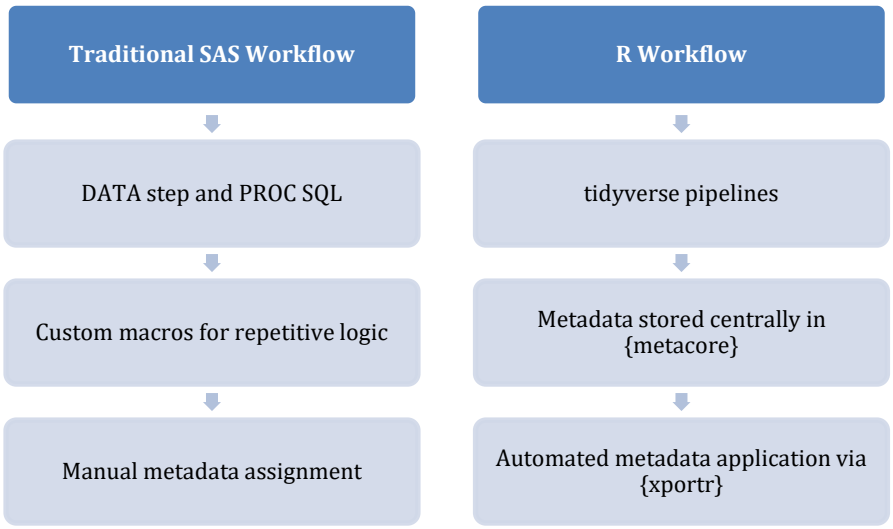

Figure 2: Workflow Comparison

KEY SDTM DERIVATIONS

1. **DM Domain**: Deriving RFSTDTC & RFENDTC

Reference start and end dates (RFSTDTC and RFENDTC) are derived using grouped summarization across exposure records. R's group_by and summarise functions provide a clear and concise alternative to PROC SQL or macro-based solutions.

```
dm_dates <- ex %>%
group_by(USUBJID) %>%
summarise(
RFSTDTC = min(EXSTDTC, na.rm = TRUE),
RFENDTC = max(EXENDTC, na.rm = TRUE)
)
```

Advantages: Grouped summarization is intuitive and Simplified readable transformation workflow.

2. **AE Domain:** Common derivations include study day variables (AESTDY, AEENDY), treatment-emergent flags (TRTEMFL). Vectorized date comparisons in dplyr enable transparent logic for identifying treatment-emergent events.

```
ae <- ae %>%
left_join(dm %>% select(USUBJID, RFSTDTC)) %>%
mutate(TRTEMFL = if_else(ASTDTC >= RFSTDTC, "Y", "N"))
```

Advantages: R (with dplyr) expresses the Clear, vectorized date comparison in a single pipeline

3. **LB Domain:** Study day derivation is simplified through standardized date comparison, enabling consistent calculation of relative day values without manual conditional logic.

```
lb <- lb_raw %>%
derive_vars_dy(
reference_date = RFSTDT,    source_vars = exprs(LBDT)  )
```

Advantages: With admiral::derive_vars_dy(), R reliably calculates study day values (e.g., --DY) relative to a reference date through a standardized, validated, and reusable approach.

4. **SUPPQUAL:** metatools::build_qnam(), R programmatically converts non-standard domain variables into SUPPQUAL key–value records, ensuring standardized linkage, naming consistency, and SDTM compliance.

```
suppdm <- bind_rows(
build_qnam(dm, "RACEOR1", "", "", "CRF"),
build_qnam(dm, "RACEOR2", "", "", "CRF"))
```

Advantages: With metatools::build_qnam(), R simplifies SUPPQUAL creation by automating variable pivoting, record linkage, and QNAM/QLABEL assignment through a reusable function.

CHALLENGES AND MITIGATION

R handles character and date-time objects differently than SAS, and partial date handling requires explicit rules. These challenges are mitigated by using lubridate for date parsing, xportr for XPT-compliant formatting, and pre-validation of date fields prior to derivation.

ADAM IMPLEMENTATION IN R

R-based workflows commonly support the creation of ADSL, ADAE, ADLB (BDS), and ADTTE datasets using admiral and tidyverse tools.

KEY ADAM DERIVATIONS

1. **ADSL:** Treatment start and end dates are derived using admiral helper functions that ensure consistency across studies.

```
adsl <- adsl %>%
derive_vars_dt(new_vars_prefix = "TRTS", dtc = RFXSTDTC) %>%
derive_vars_dt(new_vars_prefix = "TRTE", dtc = RFXENDTC)
```

Advantages: Admiral abstracts common logic, Consistency across studies

2. **ADAE:** On-treatment flags are derived using validated admiral functions that automatically manage treatment windows and edge cases, reducing custom logic.

```
adae <- derive_var_ontrtfl(
dataset = adae,
start_date = ASTDT,
end_date = AENDT,
ref_start_date = TRTSDT,
ref_end_date = TRTEDT
)
```

Advantages: Using admiral::derive_var_ontrtfl(), R provides a standardized, validated, and reusable function that automatically handles treatment dates and edge cases with minimal code.

3. **Partial Date Imputation**: admiral provides standardized functions for controlled imputation of start and end dates, including appropriate imputation flags.

Start Date Derivation

derive_vars_dt("AST", AESTDTC, highest_imputation = "Y",min_dates = exprs(TRTSDT))

End Date Derivation

derive_vars_dt("AEN", AEENDTC, highest_imputation = "M", date_imputation = "last")

Advantages: With admiral::derive_vars_dt(), R performs standardized date conversion and controlled imputation (day/month/year flags) in a single validated call.

4. **ADLB:** Baseline records and baseline values are identified using restrict_derivation and derive_var_base, eliminating manual sorting and BY-group processing.

Baseline:

restrict_derivation(derivation = derive_var_extreme_flag, args = params(by_vars = exprs(STUDYID, USUBJID, PARAMCD), order = exprs(ADT, ADTM), new_var = ABLFL, mode = "last"), filter = ((AVALC != "" | !is.na(AVAL)) & ((!is.na(ADTM) & ADTM < TRTSDTM) | (is.na(ADTM) & !is.na(ADT) & ADT <= TRTSDT))))

Base:

adlb <- derive_var_base (dataset = adlb_raw, new_var = BASE, by_vars = exprs(USUBJID, PARAMCD))

Advantages: With admiral::restrict_derivation(), R automatically and consistently identifies baseline records across parameters using a validated, reusable function, while derive_var_base() assigns the BASE value for each subject and parameter.

5. **ADTTE:** Time-to-event parameters are derived declaratively by specifying start dates, event conditions, and censoring rules within a single validated function.

```
adtte <- derive_param_tte(
dataset_adsl = adsl,
start_date = TRTSDT, event_date = DTHDT, event_conditions = DTHFL == "Y")
```

Advantages: Using admiral::derive_param_tte(), R enables a declarative, end-to-end TTE derivation (start date, event logic, censoring) in a single validated function.

CHALLENGES IN TRANSITIONING TO R

TECHNICAL CONSIDERATIONS

- Memory management differences between SAS and R
- Explicit handling of partial dates and time zones
- R exports XPT but not SAS7BDAT without SAS

PROCESS CONSIDERATIONS

- Unit testing and reproducibility culture
- Dual programming during transition
- Regulatory documentation alignment

CONCLUSIONS

R offers a powerful, transparent, and reproducible platform for SDTM and ADaM dataset creation. Its strong metadata-driven capabilities, validated pharmaverse packages, and open-source ecosystem make it a viable alternative to SAS for regulatory submissions. While challenges remain—particularly around validation and legacy mindset—R is fully capable of delivering production-grade clinical datasets when supported by compliant processes and governance frameworks. While organizational change and validation investment are required, R offers a transparent, reproducible, and future-ready foundation for clinical programming.

ACKNOWLEDGMENTS

The authors acknowledge the open-source pharmaverse community for developing and maintaining validated tools that enable regulatory-compliant clinical programming in R.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Author Name: Jalendhar Ettedi
Company: Atorus Research India Pvt. Ltd.
Email:Jalendhar.ettedi@atorusresearch.com

Author Name: Padmashree Jahagirdar
Company: Atorus Research India Pvt. Ltd.
Email:Padmashree.jahagirdar@atorusresearch.com