

{sdm.oak}

- SDTM Programming in R
V0.1 Release



Pharmaverse/ CDSIC COSA



Rammprasad Ganapathy, Genentech/Roche

Table of contents

1. SDTM challenges

2. {sdm.oak}

- Package introduction
 - What is Roche Open-sourcing?
 - Reusable Algorithms Concept
 - Programming steps & Code walkthrough
 - In this Release
 - A path to Open-source SDTM Automation
 - Upcoming Events/Releases
-

SDTM - Challenges at Industry level

- **Raw Data structure** - Different EDC systems produce data in different structures, different variable names, dataset names etc.
- **Varying Data Collection standards** - Although CDASH is available, the companies can still develop varying eCRFs using CDASH standards.

With varying raw data structure, data collection standards it may seem impossible to come up with a common approach that can be used for programming SDTM datasets.

{sdm.oak} attempts to overcome this problem by providing a EDC agnostic, Standards agnostic solution.

{sdm.oak} - Introduction

- Sponsored by CDISC COSA, pharmaceutical companies, including Roche, Pfizer, Merck, GSK, Vertex and many more
- Part of the Pharmaverse.
- Inspired by the Roche's {roak} R Package.



{sdm.oak} will be an open-source R package



{sdm.oak} will be EDC-agnostic, data standards agnostic & provides a framework for modular programming of SDTM in R




{sdm.oak} can also automate SDTM dataset creation based on the metadata driven approach using standard SDTM specifications. We will pilot it standard CDASH eCRFs in the CDISC library. The concept can be extended to any sponsor MDR.

What is Roche Open-sourcing?



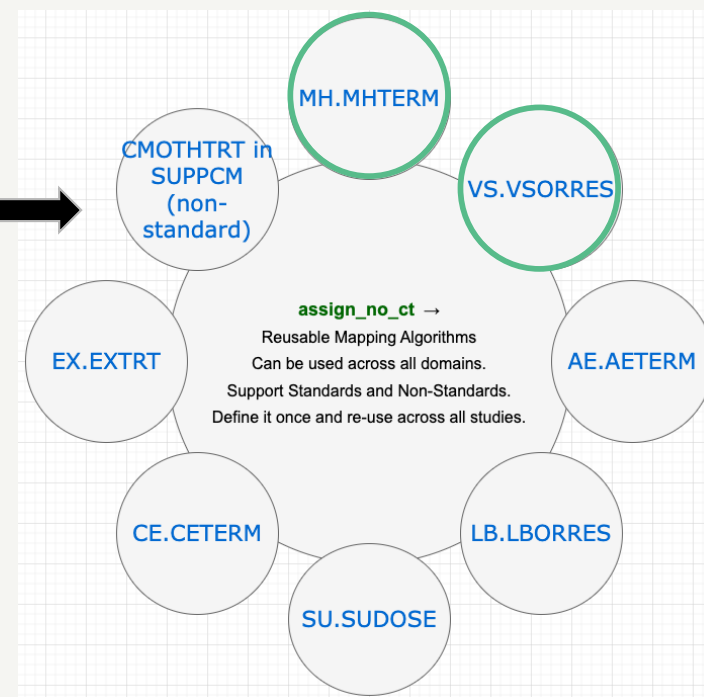
Core Concept - Reusable Algorithms

{roak} is based on the concept of reusable Algorithms.

-  Roche has used **22 algorithms** to automate **13,000 SDTM** mappings across **6 different Therapeutic Area standards**. Each Algorithms is programmed as a function in {roak}.
 -  Algorithms used for SDTM automation are being open-sourced by Roche.
 -  Source code has to be re-developed as EDC- and standards-agnostic so any company can use it.
-

Core Concept - Reusable Algorithms

Mapping Algorithm	Description
assign_no_ct	One to One mapping with no controlled terms.
assign_ct	One to One mapping with controlled terms.
hardcode_ct	Hard code the target based on the source with controlled terms.
hardcode_no_ct	Hard code the target based on the source without controlled terminology.
condition_add	Conditionally check a condition and apply a mapping



Read more about {sdm.oak} algorithms/functions [here](#)

Core Concept - Reusable Algorithms



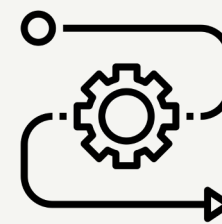
Reusable

Algorithms can be re-used across multiple SDTM domains.



Programming language-agnostic

{sdm.oak} team implemented them in R.



Automation-ready

Algorithms can be pre-specified for data collection standards in metadata repository (MDR).

{sdtm.oak} – Modular Approach

Provide a framework to modular programming of SDTM in R.

For example, a mapping of **CM.CMTRT** from raw data source `cm_raw.MDRAW` using **assign_no_ct** mapping algorithm which is programmed as **assign_no_ct()** function.

```
cm <-  
# Derive topic variable  
assign_no_ct(  
  raw_dat = cm_raw,  
  raw_var = "MDRAW",  
  tgt_var = "CMTRT"  
)
```



```
cm <-  
# Derive topic variable  
assign_no_ct(  
  raw_dat = cm_raw,  
  raw_var = "MDRAW",  
  tgt_var = "CMTRT"  
) %>%  
# Derive CMGRPID  
assign_no_ct(  
  raw_dat = cm_raw,|  
  raw_var = "MDNUM",  
  tgt_var = "CMGRPID",  
  id_vars = oak_id_vars()  
) %>%  
# DERIVE CMINDC  
assign_no_ct(  
  raw_dat = cm_raw,  
  raw_var = "MDIND",  
  tgt_var = "CMINDC",  
  id_vars = oak_id_vars()  
) %>%  
# Derive CMSTDTC. This function calls create\_iso8601
```


{sdtm.oak} – Programming Steps



- {sdtm.oak} is very close to the key SDTM concepts.
- Provide a straightforward way to do step-by-step SDTM programming in R, that is, mapping topic variable and its qualifiers.
- Programming steps are generic across SDTM domain classes like Events, Interventions, Findings
- The framework has the potential for automation (similar to Roche implementation)

{sdtm.oak} – Programming Steps

Read in Raw datasets

- Process one raw dataset at a time
- Similar raw data sources can be stacked or merged together.

Create oak_id_vars

- Unique identifiers for each record in the raw dataset
- Used to merge the qualifiers to topic variables

Read in Controlled terminology

- User prepared Controlled terminology file
- Can be in CSV or excel format

Map Topic

- Map Topic variable, usually one topic Variable at a time
- Using the mapping Algorithms/ functions

Map Rest

- Map Rest of the variables that defines the specific topic variable
- Qualifiers, Identifiers, timing, etc
- Using the mapping Algorithms/ functions

Repeat Map Topic and Map Rest

- Repeat Map Topic and Map Rest for all topic variables in the source.
- Map qualifiers that are common to all topic variables.

Create SDTM Derived variables

- Create derived variables like BLFL, study day

Add Labels and Attributes.

- SDTM Variable Labels and associated attributes.

{sdtm.oak} - Code walkthrough

- CM domain template program walkthrough.
- Demonstrates **modular** SDTM programming in R.
- Very similar to **{admiral}** style (easier for programmers).

CM Domain Example

https://pharmaverse.github.io/sdtm.oak/articles/events_domain.html

https://github.com/pharmaverse/sdtm.oak/blob/main/inst/template/create_cm_template.R



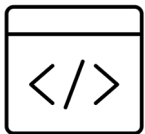
VS Domain Example:

https://pharmaverse.github.io/sdtm.oak/articles/findings_domain.html

https://github.com/pharmaverse/sdtm.oak/blob/main/inst/template/create_vs_template.R

{sdtm.oak} – V0.1 – In this Release

- Product documentation - <https://pharmaverse.github.io/sdtm.oak/index.html>
- Functions for Mapping Algorithms and required SDTM functions like sequence number, study day, baseline flag.
- Template program and Vignette to create CM domain and VS domain
- Ability program majority of the domains (Events, Interventions and Findings) . Not supported domains/Concepts - DM, RELREC, SE, SV, TDDs, metadata driven unit conversions



{sdtm.oak} V0.1 – Community can try and give us feedback via

Slack - oakgarden.slack.com

GitHub - <https://github.com/pharmaverse/sdtm.oak>

Thanks to Volunteers – Top Code/Review Contributors



Ramiro Magno – Director
Pattern Institute / Research
Software Engineer



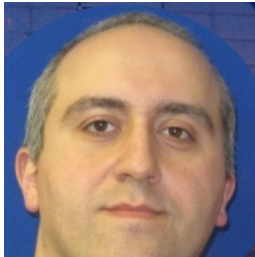
Shiyu Chen - Atorus
Research



Kamil Sijko - TTSI



Rosemary Li –
Roche/Genentech



Edgar Manukyan –
Roche/Genentech



Adam Forys –
Roche/Genentech

Venkata Maguluri – Pfizer
Yogesh Gupta – Pfizer
Preetesh Parikh – Pfizer
Aditya Parankusham – GSK
Susheel Arkala – Vertex
Phani Tata - Bayer
Omar Garcia – CDISC
Charles Shadle - CDISC

{sdm.oak} – Path to Open-source SDTM Automation



SDTM specification where users can define, raw data source, target sdm domain, target sdm variables and algorithms used for automation. A template is available in the Gitlab Repo (under development).



SDTM Controlled Terminology where the users can define the SDTM Controlled terms applicable to the study. A template is available in the Gitlab Repo.

Prepare SDTM Spec & Controlled Terminology in the format OAK expects



Automated way to read the spec and make {sdm.oak} function calls automatically.

GitHub Co-pilot is good in generating the code based on prompts.

{sdtm.oak} - Open-source SDTM Automation Vision

Modular Programming

```
cm_raw <- read.csv(system.file("raw_data/cm_raw_data.csv",
  package = "sdtm.oak"
)) %>%
  generate_oak_id_vars(
    pat_var = "PATNUM",
    raw_src = "cm_raw"
  )

dm <- read.csv(system.file("raw_data/dm.csv",
  package = "sdtm.oak"
))

# Create CM domain. The first step in creating CM domain is to create the topic variable

cm <-
# Derive topic variable
assign_no_ct(
  raw_dat = cm_raw,
  raw_var = "MDRAW",
  tgt_var = "CMTRT"
) %>%
# Derive CMGRPID
assign_no_ct(
  raw_dat = cm_raw,
  raw_var = "MDNUM",
  tgt_var = "CMGRPID",
  id_vars = oak_id_vars()
) %>%
# DERIVE CMINDC
assign_no_ct(
  raw_dat = cm_raw,
  raw_var = "MDIND",
  tgt_var = "CMINDC",
  id_vars = oak_id_vars()
) %>%
# Derive CMSTDTCT. This function calls create_iso8601
assign_datetime(
  raw_dat = cm_raw,
  raw_var = c("MDBDR", "MDBTM"),
  tgt_var = "CMSTDTCT",
  raw_fmt = c(list(c("d-m-y", "dd mmm yyyy")), "H:M"),
  raw_unk = c("UN", "UNK")
) %>%
# Derive qualifier CMSTRPT Annotation text is If MDPRIOR == 1 then CM.CMSTRPT = 'BEFORE'
hardcode_ct(
```

Create SDTM spec & automate

Prepare SDTM Spec in the format
{sdtm.oak} expects

```
library(sdtm.oak)
```

```
cm <- create_domain(
  domain = "cm",
  spec = "~/study_sdtm_spec.csv",
  ct = "~/study_ct.csv")
```

Initial focus is to develop {sdtm.oak} and pave way for SDTM programming in R.

{sdtm.oak} – Upcoming Events



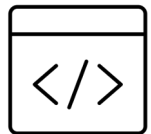
July – V 0.1 CRAN Release
Internal Hackathon – Volunteering Pharmaceutical companies.



August – Posit Blog
September – Pharmaverse Blog



October – Presentation at CDISC Interchange
October – Presentation/Virtual workshop at R in Pharma (Tentative)



We will continue to develop {sdtm.oak} and plan for a community hackathon by the end of this year.

{sdtm.oak} – Roadmap

- Functions to create the DM domain.
 - User Feedback.
 - Explore pathways for automation/code generation (copilot) based on a standard spec.
 - Metadata driven unit conversions for applicable like LB, MB, PC, IS
-

{sdtm.oak}

How to stay connected?

R package developers, Testers,
SDTM SMEs



Slack - oakgarden.slack.com

GitHub -

<https://github.com/pharmaverse/sdtm.oak>

Open to everyone to try {sdtm.oak} V0.1
and share your feedback in Slack or GitHub.



QUESTIONS
