cdisc

2023
JAPAN
INTERCHANGE
TOKYO | 10-11 JULY

Using R to generate Analysis Results Metadata (ARM)

Aik Hoe Seah, Principal Statistical Programmer, Cytel

Meet the Speaker

Aik Hoe Seah

Title: Principal Statistical Programmer

Organization: Cytel

Aik Hoe Seah, received his M.Sc (Statistics) from University of Bern, Switzerland.

He is currently a Principal Statistical Programmer at Cytel, where he has worked across many therapeutic areas (urology, gastroenterology, ophthalmology, stroke, osteoarthritis, COVID-19, oncology and many more), as well as leading submission activities for FDA and EMEA, using CDISC SDTM and ADaM standards.

He has been using SAS and R for more than 15 years and has presented in SAS Global Forum, PhUSE, PharmaSUG and local SAS forums.

Disclaimer and Disclosures

- *The views and opinions expressed in this presentation are those of the author(s) and do not necessarily reflect the official policy or position of CDISC.*

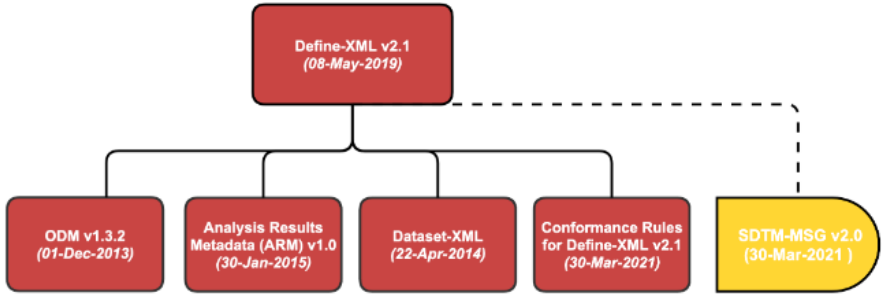
- *The author have no real or apparent conflicts of interest to report.*

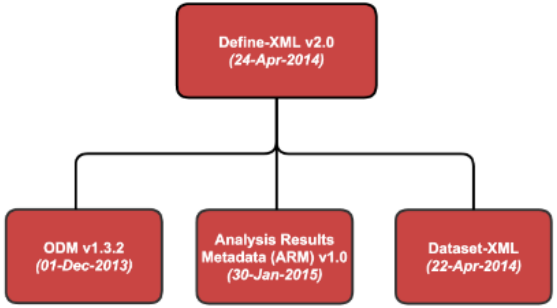
cdisc

Agenda

1. Introduction
2. Reasons to use R
3. Ingredients from ADaM specification
4. Adding ARM info back to XML
5. Conclusion

Introduction





Contains Nonbinding Recommendations

FDA U.S. FOOD & DRUG ADMINISTRATION

Separate data definition files should be included for each type of electronic dataset submission, i.e., a separate data definition file for the SDTM datasets of a given clinical study, a separate data definition file for the SEND datasets of a given nonclinical study, and a separate data definition file for the ADaM datasets of a given clinical study. The data definition file should be submitted in XML format, i.e., a properly functioning define.xml.[44] In addition to the define.xml, a printable define.pdf should be provided if the define.xml cannot be printed.[45] To confirm that a define.xml is printable within the CDER IT environment, it is recommended that the sponsor submit a test version to cder-edata@fda.hhs.gov prior to application submission. The Catalog lists the currently supported version(s) of Define-XML. It should be noted that Define-XML version 2.0 or later is strongly preferred . Sponsors should include a reference to the style sheet as defined in the specification (as listed in the Catalog) and place the corresponding style sheet in the same submission folder as the define.xml file. Within the eCTD study tagging file (STF), valid file-tags for define.xml are 'data-tabulation-data-definition' for SEND or SDTM datasets or 'analysis-data-definition' for ADaM datasets.

	A	B	C	D	E	F	G	H
1	申請電子データ提出に際して利用可能な規格一覧（令和５年2月28日）－申請電子データの標準							
2	用途	規格	バージョン	実装ガイドバージョン	形式	受付開始時期 (YYYY-MM-DD)	受付終了時期 (YYYY-MM-DD)	備考
3	データセット - 提出	SAS Transport (XPORT)	5	-	XPT	2016-10-01		
4	データセット	SDTM	1.7	3.3	XPT	2023-04-01		
5	データセット	SDTM	1.4	3.2	XPT	2016-10-01		
6	データセット	SDTM	1.3	3.1.3	XPT	2016-10-01		
7	データセット	SDTM	1.2	3.1.2 Amendment1	XPT	2016-10-01		
8	データセット	SDTM	1.2	3.1.2	XPT	2016-10-01		
9	データセット	ADaM	2.1	1.1	XPT	2022-01-01		
10	データセット	ADaM	2.1	1.0	XPT	2016-10-01		
11	定義ファイル	Define	2.0	-	XML	2016-10-01		
12	定義ファイル	Define	1.0	-	XML	2016-10-01	2025-03-31	
13	文書	PDF	1.4-1.7	-	PDF	2016-10-01		詳細は原則としてeCTDに含めるPDFの仕様に従うこと。

cdisc

Introduction

Programmers undergoing submission studies now have a few options:

1) to utilize functionality Pinnacle21 Enterprise edition (if available)
2) using their company's in-house solutions (if available)
3) writing their own R code
4) modifying SAS code available from conference proceedings

Example of ARM section of define-XML

Analysis Results Metadata - Summary

Table 14.1.1.1.1 Summary of Efficacy at Visit 4 (Observed data) - ITT
The absolute change from baseline at Week 4

Analysis Results Metadata - Detail

Table 14.1.1.1.1

Display	Summary of Efficacy at Visit 4 (Observed data) - ITT
Analysis Result	The absolute change from baseline at Week 4
Analysis Parameter(s)	PARAMCD = "ACBDEFGH"
Analysis Variable(s)	ADFA.CHG (Change from Baseline)
Analysis Reason	SPECIFIED IN SAP
Analysis Purpose	PRIMARY OUTCOME MEASURE
Data References (incl. Selection Criteria)	ADFA [PARAMCD = "ACBDEFGH" and DTYPE = "NULL" and ITTFL = "Y" and TRTPN IN (1, 2)]
Documentation	SAP Section 6.1.1: The primary efficacy analysis will be performed using an analysis of covariance (ANCOVA) model with the absolute change from baseline as the dependent variable, the randomized treatment group as a factor and the baseline value as a covariate. Least squares mean for each treatment group and for the difference between treatment groups will be presented along with two-sided p-values and 95% confidence intervals. Summary statistics will also be presented for actual values and mean change by treatment group and overall.

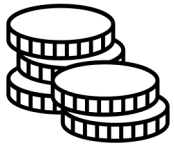
Programming Statements	[SAS version 9.4]
	`ods output lsmeans=lsm diff=diff lsmeandiffcl=diffcl;`
	`proc glm data=adfa order=data;`
	`  class trtpn;`
	`  model chg = trtpn base/solution;`
	`  lsmeans trtpn / pdiff cl;`
	`quit;`

Go to the top of the Define-XML document

Reasons to use R

Reasons generally fall within the categories of either time or budget constraints.
Some possible situations:

1) No budget to use Pinnacle21 Enterprise (P21E)
2) Having full customization ability internally within the organization.
3) Sponsor has P21E access, but define-XML is outsourced to Clinical Research Organization (CRO).
4) Only lead programmer has access to P21E but is not available currently (out of office? transitioned to other projects?), another acting lead statistical programmer must take care of the task.

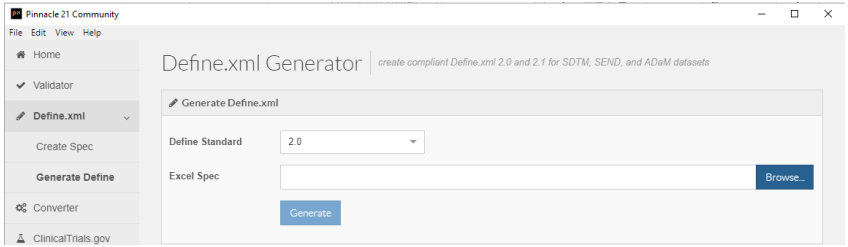


5) SAS server under maintenance but R server is working
6) And so on…

Ingredients from ADaM specification

Assumptions:

- ADaM specification template supported by Pinnacle 21 Community (P21C)
- Existing sections (non-ARM) related to ADaMs have been completed



- Define-XML v2.0 is generated using ADaM spec by P21C (if using Define-XML v2.1, adjustments to code can be made)

Steps

1) Import the ADaM specification (XLSX file) containing Analysis Displays, Results, Criteria, WhereClauses and Comments.
2) Create XML code by using ARM variables from ADaM specification.
3) Insert XML code back to the respective sections.

Step 1: Import the ADaM spec

Example of entries into the related ADaM spec tabs

- Analysis Displays
- Analysis Results
- Analysis Criteria
- WhereClauses
- Comments
- Documents

	Display	Result	Dataset	Variables	Where Clause
2	Summary of Efficacy at Visit 4 (Observed data) - ITT	The absolute change from baseline at Week 4	ADFA	CHG	PARAMCD='ABCDEFGH' and DTYPE='' and ITTFL='Y' and TRTPN in (1,2)
3					
4					
5					

... | Codelists | Methods | Comments | Documents | Analysis_Displays | Analysis_Results | **Analysis_Criteria**

	A	B	C	D	E
1	ID	Title	Document	Pages	
2	Table 14.1.1.1.1	Summary of Efficacy at Visit 4 (Observed data) - ITT			
3					
4					
5					

... | Methods | Comments | Documents | **Analysis_Displays** | Analysis_Results | Analysis_Criteria

	A	B	C	D	E	F	G	H	I	J	K
1	Display	ID	Description	Reason	Purpose	Join Comment	Documentation	Documentation Refs	Programming Context	Programming Code	Programming Document
2	Summary of Efficacy at Visit 4 (Observed data) - ITT	Table 14.1.1.1.1		SPECIFIED IN SAP	PRIMARY OUTCOME MEASURE		SAP Section 6.1.1: The primary efficacy analysis will be performed using an analysis of covariance (ANCOVA) model with the absolute change from baseline as the dependent variable, the randomized treatment group as a factor and the baseline value as a covariate. Least squares mean for each treatment group and for the difference between treatment groups will be presented along with two-sided p-values and 95% confidence intervals. Summary statistics will also be presented for actual values and mean change by treatment group and overall.		SAS version 9.4	ods output lsmeans=lsm diff=diff lsmeandiffcl=diffcl; proc glm data=adfa order=data; class trtpn; model chg = trtpn base/solution; lsmeans trtpn / pdiff cl; quit;	
3											
4											
5											

... | Methods | Comments | Documents | Analysis_Displays | **Analysis_Results** | Analysis_Criteria

Step 1: Import the ADaM spec

- **Example of entries into the related ADaM spec tabs**

Notes:

If there are additional comments or documents to support method/derivation of output, it can also be referenced to the Comments and Documents tabs

Analysis_Criteria was removed from ADaM spec template of P21C v4.0.2 (Previously existed in ~v4.0.1)

WhereClauses was removed from ADaM spec template of P21C v4.0.2 and now it's in ValueLevel (Previously existed in ~v4.0.1)

ID	Dataset	Variable	Compa	Value
Table_14.1.1.1.1.ADFA	ADFA	PARAMCD	EQ	ABCDEFGH
Table_14.1.1.1.1.ADFA	ADFA	DTYPE	EQ	null
Table_14.1.1.1.1.ADFA	ADFA	ITTFL	EQ	Y
Table_14.1.1.1.1.ADFA	ADFA	TRTPN	IN	1,2

... | ValueLevel | **WhereClauses** | Dictionaries | Codelists | Methods | Comments

Step 1: Import the ADaM spec

- Import ADaM spec in R
- There are a few ways to do so in R, using packages:

OpenXLSX, ReadXL, XLSX, XLConnect

In our example below, we are using OpenXLSX package

```
#Step 0: Import ADaM spec

library(openxlsx)

sheets <- openxlsx::getSheetNames("ADAMDefineSpecs2023-06-16_v1.xlsx")
data_frame <- lapply(sheets, openxlsx::read.xlsx, xlsxFile="ADAMDefineSpecs2023-06-16_v1.xlsx")
names(data_frame) <- sheets
```

- We will also need Tidyverse and dplyr package (for some data manipulation steps later, e.g. str_trim, left_join, write.table functions)

Step 2: Create XML code from ADaM specification

1) Generate WhereClause section

Note: If generating ADaM spec using P21C v4.0.2, WhereClause is rolled into ValueLevel

```
#Step 2.1: Generate WhereClause section
df_wc <- data_frame$`WhereClauses`
t0 <- df_wc[grep("Table|Listing|Figure", df_wc$ID), ]
#count the number of commas in Value to find out no. of records to split later on
t0$vcount <- lengths(regmatches(t0$Value, gregexpr(",", t0$Value)))
t0$cv <- ifelse(t0$vcount == 0, paste('<CheckValue>', str_trim(t0$Value), '</CheckValue>', sep=""),
                ifelse(t0$vcount == 1, paste('<CheckValue>', sub(",.*", "", t0$Value), '</CheckValue>,<CheckValue>',
                                                             sub(".*,", "", t0$Value), '</CheckValue>', sep=""),
t0$rc_st <- paste('<RangeCheck SoftHard="Soft" def:ItemOID="IT.', str_trim(t0$Dataset), ".",
                  str_trim(t0$Variable), '" Comparator="', str_trim(t0$Comparator), '">', sep="")
t0$rc_en <- " </RangeCheck>"
t0$vnum <- ave(t0$cv, t0$ID, FUN = seq_along)
t0$t2 <- ifelse(t0$vnum==1, paste('<def:WhereClauseDef OID="WC.', str_trim(t0$ID), '">,', sep=""), '')
t0$t2e <- ifelse(t0$vnum==4, ',</def:WhereClauseDef>', '')
rc1 <- data.frame(Armcode = paste(str_trim(t0$t2), str_trim(t0$rc_st), ',', str_trim(t0$cv), ',',
                                  str_trim(t0$rc_en), str_trim(t0$t2e), sep=""))
t_wc <- separate_rows(rc1, Armcode, sep = ",", convert = TRUE)
```

cdisc

Step 2: Create XML code from ADaM specification

2) To view the dataframe created, we simply run

View(t_wc)

The result is displayed on the right

t_wc

Filter

	Armcode
1	<def:WhereClauseDef OID="WC.Table_14.2.1.1.1.ADFA">
2	<RangeCheck SoftHard="Soft" def:ItemOID="IT.ADFA.PARAMCD" Comparator="EQ">
3	<CheckValue>ABCDEFGH</CheckValue>
4	</RangeCheck>
5	<RangeCheck SoftHard="Soft" def:ItemOID="IT.ADFA.DTYPE" Comparator="EQ">
6	<CheckValue>null</CheckValue>
7	</RangeCheck>
8	<RangeCheck SoftHard="Soft" def:ItemOID="IT.ADFA.ITTFL" Comparator="EQ">
9	<CheckValue>Y</CheckValue>
10	</RangeCheck>
11	<RangeCheck SoftHard="Soft" def:ItemOID="IT.ADFA.TRTPN" Comparator="IN">
12	<CheckValue>1</CheckValue>
13	<CheckValue>2</CheckValue>
14	</RangeCheck>
15	</def:WhereClauseDef>

Step 2: Create XML code from ADaM specification

The following sections can also possibly be skipped if not used:

- 3) Generate Comments section

```
# Step 2.3 Generate Comments section
com <- data_frame$`Comments`
df_com <- com[grep("Table|Listing|Figure", com$ID), ]
df_com1 <- data.frame(Armcode = paste('<def:CommentDef OID="COM.', str_trim(df_com$ID),
                                      '">,<Description>,<TranslatedText xml:lang="en">',
            str_trim(df_com$Description), '</TranslatedText>,</Description>,</def:CommentDef>', sep=""))
t_com <- separate_rows(df_com1, Armcode, sep = ",", convert = TRUE)
```

- 4) Generate Leafs (links) section

```
#  Step 2.4: Generate Leafs (links) Definition for ARM
leafs <- data_frame$`Documents`
df_leafs <- leafs[grep("Table|Listing|Figure", leafs$ID), ]
df_leafs1 <- data.frame(Armcode = paste('<def:leaf ID="LF.', str_trim(df_leafs$ID),
                                        '" xlink:href="', str_trim(df_leafs$Href),
                                        '">,<def:title>', str_trim(df_leafs$Title),
                                        '</def:title>,</def:leaf>', sep=""))
t_leafs <- separate_rows(df_leafs1, Armcode, sep = ",", convert = TRUE)
```

Step 2: Create XML code from ADaM specification

- 5) Generate ARM definition section (part 1)

```
#  Step 2.5:  Generate ARM definition Section
df_ad <- data_frame$`Analysis_Displays`
df_ar <- data_frame$`Analysis_Results`
df_ac <- data_frame$`Analysis_Criteria`

df_j1 <- left_join(df_ad, df_ar)
df_j2 <- left_join(df_j1, df_ac)
df_j2$temp <- gsub(' ', '_', df_j2$ID)

df2_st <- data.frame(Armcode=c("<!-- ******************************************************* -->",
                               "<!-- *****Analysis results metadata Definitions Section***** -->",
                               "<!-- ******************************************************* -->",
                               "<arm:AnalysisResultDisplays>"))
df2_en <- data.frame(Armcode=c("</arm:AnalysisResultDisplays>"))
rd_st <- data.frame(Armcode=paste(' <arm:ResultDisplay OID="RD.', str_trim(df_j2$temp), '" Name="', str_trim(df_j2$ID), '">', sep=""))
rd_en <- data.frame(Armcode=' </arm:ResultDisplay>')
rd_desc <- data.frame(Armcode = rbind(' <Description>',
                                      paste(' <TranslatedText xml:lang="en">', str_trim(df_j2$Display), '</TranslatedText>', sep=""),
                                      ' </Description>'))
ar_desc_ <- data.frame(Armcode = paste('<arm:AnalysisResult~', ' OID="AR.', str_trim(df_j2$temp),
                                       '"~ ParameterOID="IT.', str_trim(df_j2$Dataset), '.PARAMCD"~',
                                       ' AnalysisReason="', str_trim(df_j2$Reason), '"~',
                                       ' AnalysisPurpose="', str_trim(df_j2$Purpose), '">~',
                                       ' <Description>~',
                                       ' <TranslatedText xml:lang="en">', str_trim(df_j2$Result), '</TranslatedText>~',
                                       ' </Description>',
                                       sep=""))
ar_desc <- separate_rows(ar_desc_, Armcode, sep = "~", convert = TRUE)
ar_en <- data.frame(Armcode=c("</arm:AnalysisResult>"))
```

Step 2: Create XML code from ADaM specification

- 5) Generate ARM definition section (part 2)

```
ads1 <- data.frame(Armcode=c(" <arm:AnalysisDatasets>"))
ads1_ <- data.frame(Armcode=c(" </arm:AnalysisDatasets>"))
ads_body_ <- data.frame(Armcode = paste(' <arm:AnalysisDataset ItemGroupOID="IG.',
                                        str_trim(df_j2$Dataset), '">~',
                                        ' <def:WhereClauseRef WhereClauseOID="WC.',
                                        str_trim(df_j2$temp), '.', str_trim(df_j2$Dataset), '"/>~',
                                        ' <arm:AnalysisVariable ItemOID="IT.',
                                        str_trim(df_j2$Dataset), '.', str_trim(df_j2$Variables), '"/>~',
                                        ' </arm:AnalysisDataset>',
                                        sep=""))
ads_body <- separate_rows(ads_body_, Armcode, sep = "~", convert = TRUE)
ads <- rbind(ads1, ads_body, ads1_)
docu_ <- data.frame(Armcode = paste('<TranslatedText xml:lang="en">', str_trim(df_j2$Documentation),
                                    '</TranslatedText>', sep=""))
docu <- rbind(' <arm:Documentation>', ' <Description>', docu_, ' </Description>', ' </arm:Documentation>')
pgmcode_st <- data.frame(Armcode = paste(' <arm:ProgrammingCode Context="',
                                         str_trim(df_j2$Programming.Context), '">', sep=""))
pgmcode_body <- data.frame(Armcode = rbind(' <arm:Code>', str_trim(df_j2$Programming.Code), ' </arm:Code>'))
pgmcode_en <- data.frame(Armcode=c(" </arm:ProgrammingCode>"))
prgcode <- rbind(pgmcode_st, pgmcode_body, pgmcode_en)
ar <- rbind(ar_desc, ads, docu, prgcode, ar_en)
rd <- rbind(rd_st, rd_desc, ar, rd_en)
t_ard <- rbind(df2_st, rd, df2_en)
```

Step 2: Create XML code from ADaM specification

Once again, to view the dataframe created, we simply run View(t_ard)

The result is displayed as below

t_ard

Filter

	Armcode
1	<!-- ********** -->
2	<!-- *****Analysis results metadata Definitions Section***** -->
3	<!-- ********** -->
4	<arm:AnalysisResultDisplays>
5	<arm:ResultDisplay OID="RD.Table_14.1.1.1.1" Name="Table 14.1.1.1.1">
6	<Description>
7	<TranslatedText xml:lang="en">Summary of Efficacy at Visit 4 (Observed data) - ITT</TranslatedText>
8	</Description>
9	<arm:AnalysisResult
10	OID="AR.Table_14.1.1.1.1"
11	ParameterOID="IT.ADFA.PARAMCD"
12	AnalysisReason="SPECIFIED IN SAP"
13	AnalysisPurpose="PRIMARY OUTCOME MEASURE">
14	<Description>
15	<TranslatedText xml:lang="en">The absolute change from baseline at Week 4</TranslatedText>
16	</Description>
17	<arm:AnalysisDatasets>

t_ard

Filter

	Armcode
17	<arm:AnalysisDatasets>
18	<arm:AnalysisDataset ItemGroupOID="IG.ADFA">
19	<def:WhereClauseRef WhereClauseOID="WC.Table_14.1.1.1.1.ADFA"/>
20	<arm:AnalysisVariable ItemOID="IT.ADFA.CHG"/>
21	</arm:AnalysisDataset>
22	</arm:AnalysisDatasets>
23	<arm:Documentation>
24	<Description>
25	<TranslatedText xml:lang="en">SAP Section 6.1.1: The primary efficacy analysis will be performed using an analysis of covariance (ANCOVA) model with the absolute cha
26	</Description>
27	</arm:Documentation>
28	<arm:ProgrammingCode Context="SAS version 9.4">
29	<arm:Code>
30	ods output lsmeans=lsm diff=diff lsmeandiffcl=diffcl; proc glm data=adfa order=data; class trtpn; model chg = trtpn base/solution; lsmeans trtpn / pdiff cl; quit;
31	</arm:Code>
32	</arm:ProgrammingCode>
33	</arm:AnalysisResult>
34	</arm:ResultDisplay>
35	</arm:AnalysisResultDisplays>

Step 3: Insert XML code back to the respective sections

Method 1: Manual insertion

1) Print dataframe out to txt

```
#Step 3.1
write.table(t_wc, file = "define_arm_wc.txt", sep='', row.names=FALSE, col.names=FALSE, quote=FALSE)
write.table(t_ard, file = "define_arm_ard.txt", sep='', row.names=FALSE, col.names=FALSE, quote=FALSE)
```

2) Copy and insert created sections into the appropriate locations (end of each block)

define_arm_ard.txt - Notepad

File Edit Format View Help

```
<!-- ******************************************************* -->
<!-- *****Analysis results metadata Definitions Section***** -->
<!-- ******************************************************* -->
<arm:AnalysisResultDisplays>
 <arm:ResultDisplay OID="RD.Table_14.1.1.1.1" Name="Table 14.1.1.1.1">
 <Description>
 <TranslatedText xml:lang="en">Summary of Efficacy at Visit 4 (Observed data) - ITT</Transl
 </Description>
<arm:AnalysisResult
 OID="AR.Table_14.1.1.1.1"
 ParameterOID="IT.ADFA.PARAMCD"
```

Ln 41, Col 1 | 100% | Windows (CRLF) | UTF-8

*define-2023-06-06T04-31-40-658.xml - Notepad

File Edit Format View Help

```
        </def:leaf>
        <def:leaf ID="LF.adcm-file" xlink:href="../programs/adcm-sas.txt">
           <def:title>adcm SAS Program</def:title>
        </def:leaf>
        <def:leaf ID="LF.admh-file" xlink:href="../programs/admh-sas.txt">
           <def:title>admh SAS Program</def:title>
        </def:leaf>

     </MetaDataVersion>
  </Study>
</ODM>
```

Ln 14249, Col 1 | 100% | Unix (LF) | UTF-8

Step 3: Insert XML code back to the respective sections

Example of ARM section in Define-XML v2.0 viewed with XSL stylesheet

No differences were found vs Define-XML generated via SAS

ABCDEFG-101, ADaM-IG1.1

ABCDEFG-101

- Supplemental Documents
  - Analysis Data Reviewer's Guide
  - adsl SAS Program
  - adeg SAS Program
  - adexsum SAS Program
  - adfa SAS Program
  - adft SAS Program
  - adie SAS Program
  - adlb SAS Program
  - adpe SAS Program
  - adqs SAS Program
  - advs SAS Program
  - adae SAS Program
  - adcm SAS Program
  - admh SAS Program
- Analysis Results Metadata
  - Table 14.1.1.1.1
- Datasets
  - ADSL (Subject-Level Analysis Dataset)
  - ADEG (ECG Test Results Analysis Dataset)
  - ADEXSUM (Exposure Summary Analysis D
  - ADFA (Findings About Analysis Dataset)
  - ADFT (Functional Tests Analysis Dataset)
  - ADIE (Inclusion/Exclusion Criterion Datase
  - ADLB (Laboratory Tests Analysis Dataset)
  - ADPE (Physical Examination Analysis Data
  - ADQS (Questionnaire Analysis Dataset)
  - ADVS (Vital Signs Analysis Dataset)
  - ADAE (Adverse Events Analysis Dataset)
  - ADCM (Concomitant Medications Analysis
  - ADMH (Medical History Analysis Dataset)
- Controlled Terminology
  - CodeLists
    - ACNAE
    - ACTARM
    - ADEGPARAM
    - ADEGPARAMCD
    - ADEGPARCAT1
    - ADEXSUMPARAM
    - ADEXSUMPARAMCD
    - ADEXSUMPARAMN

Date/Time of Define-XML document generation: 2023-06-06T04:31:42-04:00
Define-XML version: 2.0.0
Stylesheet version: 2018-11-21

Standard	ADaM-IG 1.1
Study Name	ABCDEFG-101
Study Description	A Randomized, Double-Blind, Placebo-Controlled Pilot Study to Evaluate the Efficacy, Safety, and Tolerability of ABCDEFGH in Healthy Subjects
Protocol Name	ABCDEFG-101
Metadata Name	Study ABCDEFG-101 Data Definitions

Analysis Results Metadata - Summary

Table 14.1.1.1.1 Summary of Efficacy at Visit 4 (Observed data) - ITT
The absolute change from baseline at Week 4

Analysis Results Metadata - Detail

Table 14.1.1.1.1

Display	**Summary of Efficacy at Visit 4 (Observed data) - ITT**
Analysis Result	**The absolute change from baseline at Week 4**
Analysis Parameter(s)	PARAMCD = "ACBDEFGH"
Analysis Variable(s)	ADFA.CHG (Change from Baseline)
Analysis Reason	SPECIFIED IN SAP
Analysis Purpose	PRIMARY OUTCOME MEASURE
Data References (incl. Selection Criteria)	ADFA [PARAMCD = "ACBDEFGH" and DTYPE = "NULL" and ITTFL = "Y" and TRTPN IN (1, 2)]
Documentation	SAP Section 6.1.1: The primary efficacy analysis will be performed using an analysis of covariance (ANCOVA) model with the absolute change from baseline as the dependent variable, the randomized treatment group as a factor and the baseline value as a covariate. Least squares mean for each treatment group and for the difference between treatment groups will be presented along with two-sided p-values and 95% confidence intervals. Summary statistics will also be presented for actual values and mean change by treatment group and overall.

Programming Statements	[SAS version 9.4]
	`ods output lsmeans=lsm diff=diff lsmeandiffcl=diffcl;`
	`proc glm data=adfa order=data;`
	`  class trtpn;`
	`  model chg = trtpn base/solution;`
	`  lsmeans trtpn / pdiff cl;`
	`quit;`

Go to the top of the Define-XML document

Step 3: Insert XML code back to the respective sections

Proposing Method 2: Use R4DSXML package to convert Define-XML to R dataframe

Steps:

1) Import previously created Define-XML (without ARM section) using package XML

2) Convert Define-XML to R dataframe

3) Convert and append previously created dataframe (for individual sections) to their locations within the R dataframe

4) Convert R dataframe back to XML structure

5) Finally, output the XML object into Define-XML file

6) Open Define-XML file with stylesheet and review any missing elements

Step 3: Insert XML code back to the respective sections

- Functions from R4DSXML package:
- getAR
- getARDISP
- getCT
- getDLMD
- getStudyMD
- getValMD
- getVarMD
- read.dataset.xml

Example of Variable Level Metadata

```
definev3 <- paste0("define-2023-06-06T04-31-40-658_v3.xml")
v3varmd <- getVarMD(definev3)
View(v3varmd)
```

v3varmd

Filter

	IR_ItemOID	IGD_Name	IR_OrderNumber	IR_Mandatory	IR_KeySequence	ID_Name	ID_Length	ID_SignificantDigits	ID_DataType	ID_Label	ID_SASFieldName
429	IT.ADSL.STUDYID	ADSL	1	Yes	1	STUDYID	11	NA	text	Study Identifier	STUDYID
430	IT.ADSL.SITEID	ADSL	2	Yes	NA	SITEID	3	NA	text	Study Site Identifier	SITEID
431	IT.ADSL.SUBJID	ADSL	3	Yes	NA	SUBJID	3	NA	text	Subject Identifier for the Study	SUBJID
432	IT.ADSL.USUBJID	ADSL	4	Yes	2	USUBJID	19	NA	text	Unique Subject Identifier	USUBJID
433	IT.ADSL.AGE	ADSL	5	No	NA	AGE	8	NA	integer	Age	AGE
434	IT.ADSL.AGEU	ADSL	6	Yes	NA	AGEU	5	NA	text	Age Units	AGEU
435	IT.ADSL.SEX	ADSL	7	Yes	NA	SEX	1	NA	text	Sex	SEX
436	IT.ADSL.SEXN	ADSL	8	Yes	NA	SEXN	8	NA	integer	Sex (N)	SEXN
437	IT.ADSL.RACE	ADSL	9	Yes	NA	RACE	5	NA	text	Race	RACE
438	IT.ADSL.RACEN	ADSL	10	Yes	NA	RACEN	8	NA	integer	Race (N)	RACEN
439	IT.ADSL.ETHNIC	ADSL	11	Yes	NA	ETHNIC	22	NA	text	Ethnicity	ETHNIC
440	IT.ADSL.ETHNICN	ADSL	12	Yes	NA	ETHNICN	8	NA	integer	Ethnicity (N)	ETHNICN
441	IT.ADSL.COUNTRY	ADSL	13	Yes	NA	COUNTRY	3	NA	text	Country	COUNTRY
442	IT.ADSL.BRTHDTC	ADSL	14	Yes	NA	BRTHDTC	4	NA	text	Date/Time of Birth	BRTHDTC
443	IT.ADSL.SCRFL	ADSL	15	No	NA	SCRFL	1	NA	text	Screened Population Flag	SCRFL
444	IT.ADSL.SCFAILFL	ADSL	16	Yes	NA	SCFAILFL	1	NA	text	Screened Fail Population Flag	SCFAILFL
445	IT.ADSL.RANDFL	ADSL	17	Yes	NA	RANDFL	1	NA	text	Randomized Population Flag	RANDFL
446	IT.ADSL.ITTFL	ADSL	18	Yes	NA	ITTFL	1	NA	text	Intent-To-Treat Population Flag	ITTFL
447	IT.ADSL.ENRLFL	ADSL	19	Yes	NA	ENRLFL	1	NA	text	Enrolled Population Flag	ENRLFL
448	IT.ADSL.SAFFL	ADSL	20	Yes	NA	SAFFL	1	NA	text	Safety Population Flag	SAFFL

Showing 428 to 448 of 507 entries, 16 total columns

Future development

1) Improve efficiency of code for Define-XML v2.0 to cater to possibly more scenarios

2) Improve code to ensure it also runs smoothly with Define-XML v2.1

3) Automate process of inserting XML code back to the respective sections

(Have tried out some capabilities using XML package, but round trip of XML to List back to XML removes def tags, so need to figure out alternative way in combination with R4DSXML package)

4) Possibility of Rshiny app?

Conclusion

- We have suggested some methods for statistical programmers using R, to generate Analysis Results Metadata (ARM) by using available packages like OpenXLSX, Tidyverse, XML, R4DSXML, etc.
- Open-source software does the job too!
- Independent validation between different software is possible

cdisc

References

- Ippei Akiya. 2021. “R4DSXML, Read CDISC Data Files”. R Package version 0.6.3. Available at https://github.com/i-akiya/R4DSXML
- Diane Wold and Jeff Abolafia, 2021. “CDISC Update Analysis Results Standard”. Proceedings of the 2021 Pharmasug North Carolina SDE. Available at https://www.pharmasug.org/download/sde/rtp2021/PharmaSUG-NCSDE_2021-08.pdf
- Kiran K. Kundurapu and Nancy Baeur, 2021. “FDA and PMDA Study Data Submission Distinctions”. PhUSE Optimizing the Use of Data Standards Working Group White Paper. Available at https://phuse.s3.eu-central-1.amazonaws.com/Deliverables/Optimizing+the+Use+of+Data+Standards/WP047.pdf
- Srivathsa Ravikiran and Priscilla Gathoni, 2019. “One Click to Analysis Results Metadata”. Proceedings of the 2019 South East SAS User Group Conference. Available at https://www.lexjansen.com/sesug/2019/SESUG2019_Paper-259_Final_PDF.pdf
- Stackoverflow community

Thank You!

Your comments and questions are greatly valued and encouraged!
For any possible errors, please kindly contact me at:

Aik Hoe Seah
Cytel Singapore
aikhoe.seah@cytel.com

cdisc

Q&A