

ARD Generator

Metadata-driven analysis using the Analysis Results Standard



COSA Spotlight 12 December 2023

Karl Wallendszus, Oxford Population Health



CDISC Foundational Standards

Data Collection
CDASH



Data Aggregation
SDTM



Analysis
ADaM



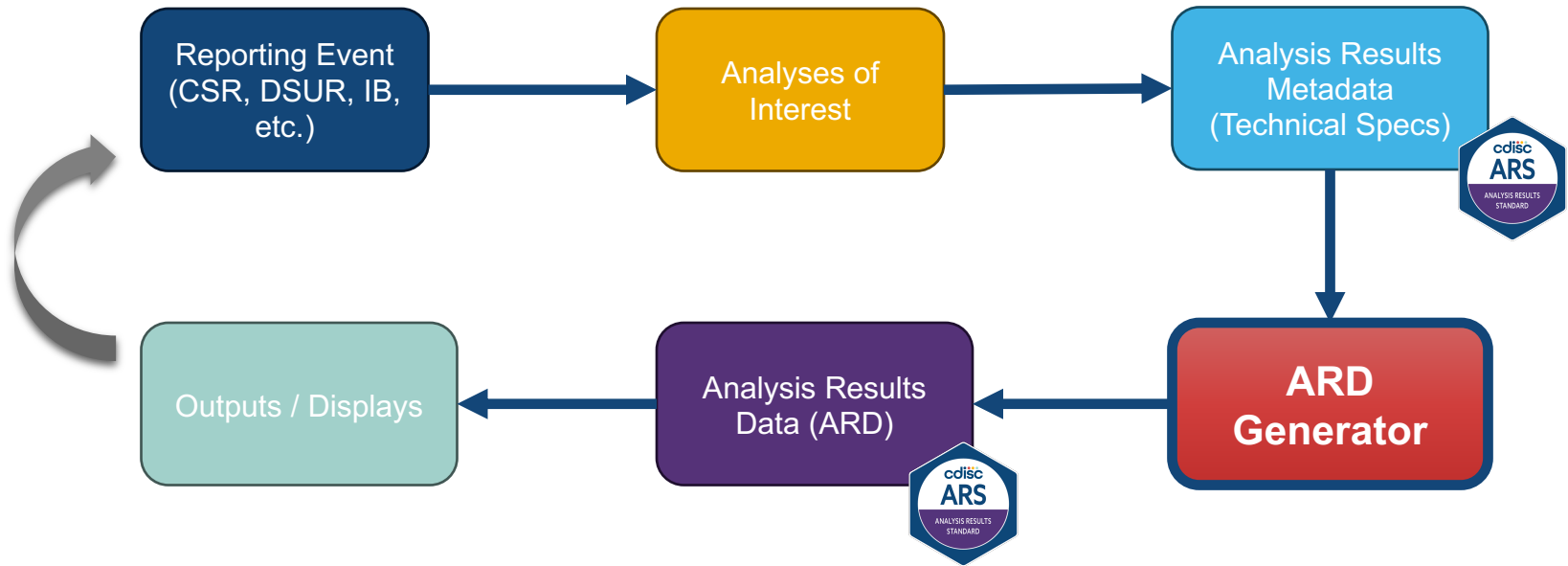
Results
???

Table 4.2.2: HbA1c Longitudinal Repeated Measures Analysis Results Metadata	
Metadata Field	Metadata
DISPLAY IDENTIFIER	Table 4.2.1/Figure 4.2.1
DISPLAY NAME	Mean Change from Baseline in HbA1c (Percent) Longitudinal Repeated Measures Analysis, 24-Week Short-term Double-blind Treatment Period, Intention-to-treat Population
RESULT IDENTIFIER	Treatment difference results (LSMean, confidence interval, p-value)
PARAM	HbA1c (%)
PARAMCD	HBA1C
ANALYSIS VARIABLE	CHG (Change from baseline)
ANALYSIS REASON	SPECIFIED IN SAP
ANALYSIS PURPOSE	PRIMARY OUTCOME MEASURE
ANALYSIS DATASET	ADHBA1C

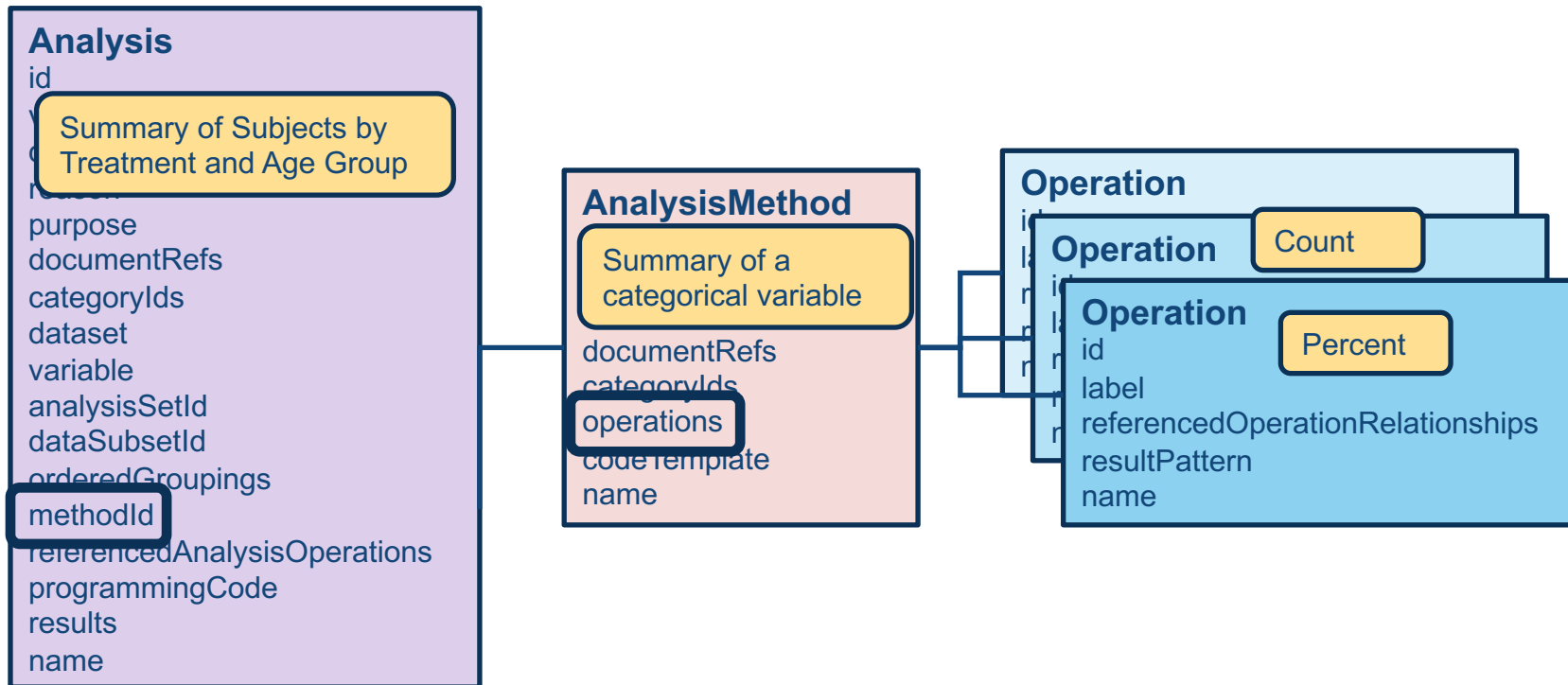


ARM for Define.XML

ARS Model Supported Workflow and Entry Points



Analysis – Method - Operation





Structure of code

- Written in SAS
- Ingest ARS metadata from JSON → SAS datasets
- Process analyses specified in metadata
- Macro library
 - Analysis
 - Method
 - Operation
 - Particular operations
- Output to Analysis Results Dataset (ARD)

Structure of code

The screenshot displays the GitHub interface for the 'ard-generator' repository. At the top, the repository name and owner 'KarlWallendszus / ard-generator' are visible, along with a search bar and navigation links for Code, Issues, Pull requests, Zenhub, Actions, Projects, Security, Insights, and Settings. The repository is public and has 1 branch (main) and 1 tag. The file list includes folders for 'data', 'log', 'macros', and 'programs', and files for '.gitignore', 'README.md', 'getversion.cmd', 'locate_libs.sas', 'setbase_stem.sas', 'setbasedir.cmd', 'setprogdtdt.sas', 'setup.sas', and 'setup_version.cmd'. The right sidebar shows repository statistics: 13 commits, 0 stars, 1 watching, and 0 forks. The 'Releases' section shows version v0.1 as the latest release from July 21. The 'Packages' section indicates no packages are published. The 'Languages' section shows SAS at 99.2% and Batchfile at 0.8%.

File/Folder	Description	Last Commit
data	Separate running analysis, method and operation into library macros.	last month
log	Add programs, macros and test data	2 months ago
macros	Handle names and descriptions of analyses, methods and operations...	4 days ago
programs	Handle names and descriptions of analyses, methods and operations...	4 days ago
.gitignore	Add macro for categorical variable summary by group percentages	last week
README.md	Initial commit	2 months ago
getversion.cmd	Add programs, macros and test data	2 months ago
locate_libs.sas	Add programs, macros and test data	2 months ago
setbase_stem.sas	Add programs, macros and test data	2 months ago
setbasedir.cmd	Add programs, macros and test data	2 months ago
setprogdtdt.sas	Add programs, macros and test data	2 months ago
setup.sas	Add programs, macros and test data	2 months ago
setup_version.cmd	Add programs, macros and test data	2 months ago

Structure of code

The screenshot shows the GitHub interface for the repository 'KarlWallendszus / ard-generator'. The left sidebar displays the file tree with the 'programs' directory selected. The main content area shows the commit history for the 'programs' directory, listing files and their commit messages.

Repository: KarlWallendszus / ard-generator

Navigation: Code, Issues, Pull requests, Zenhub, Actions, Projects, Security, Insights, Settings

Files sidebar:

- main
- data
- log
- macros
 - source
 - append_addcols.sas
 - build_expression.sas
 - build_work_dataset.sas
 - define_analset.sas
 - op_catvar_count_bygrp_n.sas
 - op_catvar_summ_bygrp_pct....
 - outline_ard.sas
 - run_analysis.sas
 - run_method.sas
 - run_operation.sas
 - standardize_grouping.sas
 - standardize_groupings.sas
- test
- programs

Commit history for 'ard-generator / programs':

Name	Last commit message	Last commit date
..		
import_adam.sas	Add programs, macros and test data	2 months ago
import_json_reportingevent.sas	Separate running analysis, method and operation into library macros.	last month
reportingevent.map	Add programs, macros and test data	2 months ago
reportingevent_md.map	Separate running analysis, method and operation into library macros.	last month
run_analyses.sas	Handle names and descriptions of analyses, methods and operations wit...	4 days ago

Sign in now to use Zenhub

Run analyses

```
.....;
* Main code
.....;

* Create dataset for expressions in SAS syntax for conditions;
data jsonmd.expressions;
  length id $40 label $200 dsconds $32 expression $200;

  * Derive expressions in SAS syntax for conditions;

  * Run analyses;
  %run_planned_analyses(mdlib=jsonmd, datalib=adam, ardlib=ard);
  /*
  %run_analysis(mdlib=jsonmd, datalib=adam, ardlib=ard,
    analid=An01_05_SAF_Summ_ByTrt, debugfl=Y);
  %run_analysis(mdlib=jsonmd, datalib=adam, ardlib=ard,
    analid=An03_02_AgeGrp_Summ_ByTrt, debugfl=Y);
  %run_analysis(mdlib=jsonmd, datalib=adam, ardlib=ard,
    analid=An03_05_Race_Summ_ByTrt, debugfl=Y);
  %run_analysis(mdlib=jsonmd, datalib=adam, ardlib=ard,
    analid=An07_09_Soc_Summ_ByTrt, debugfl=Y);
  */
run_
  a
run_
  a
run_
  a
analid=An07_09_Soc_Summ_ByTrt, debugfl=Y);
*/
```

Run analyses

```
/**
 * Run all planned analyses.
 *
 * @param mdlib      Library containing metadata datasets.
 * @param datalib    Library containing data to be analysed.
 * @param ardlib     Library containing analysis results datasets.
 * @param debugfl    Debug flag (Y/N).
 */
%macro run_planned_analyses ( mdlib=, datalib=, ardlib=, debugfl=N );

    * Get list of planned analyses;
    %local analids;
    proc sql;
        select distinct analysisid into :analids separated by '|'
            from &mdl lib..listofplannedanalyses
            where analysisid ^= ''
            order by 1;
    quit;

    * Loop through analyses;
    %local ianal analid;
    %let ianal = 1;
    %do %while(%scan(&analids., &ianal., '|') ne );
        %let analid = %scan(&analids., &ianal., '|');

        * Run this analysis;
        %run_analysis(mdlib=&mdl lib., datalib=&dat alib., ardlib=&ard lib.,
            analid=&analid., debugfl=&debugfl.);

        %let ianal = %eval(&ianal.+1);
    %end;

%mend run_planned_analyses;
```

Run a single analysis

```
⊞ %macro run_analysis ( mdlib=, datalib=, ardlib=, analid=, debugfl=N );

    * Get analysis details;
    %local analname analdesc analver analreas analpur analds analvar analsetid
        datasubsetid method
        ngroupings groupingords groupingids groupingresbys
        nrefops refopords refoprelids refopanalids
        docrefs catids;
    proc sql;
        select name, description, version, reason, purpose, dataset, variable,
            analysisSetId, dataSubsetId, method_id
            into :analname, :analdesc, :analver, :analreas, :analphur,
                :analds, :analvar, :analsetid, :datasubsetid, :method
            from &mdlid..analyses
            where id = "&analid.";
        select count(*) into :ngroupings
            from &mdlid..analysesordgroupings
            where id = "&analid.";
        select order, groupingId, resultsByGroup
            into :groupingords separated by '|', :groupingids separated by '|',
                :groupingresbys separated by '|'
            from &mdlid..analysesordgroupings
            where id = "&analid."
            order by 1;
        select count(*) into :nrefops
            from &mdlid..analysesrefoperations
            where id = "&analid.";
        select order, referencedOperationRelationshipI, analysisId
            into :refopords separated by '|', :refoprelids separated by '|',
                :refopanalids separated by '|'
            from &mdlid..analysesrefoperations
            where id = "&analid."
            order by 1;

    quit;
```

Run a single analysis

```
* Show analysis details;
%put NOTE: =====;
%put NOTE: Analysis &analid.: &analname.;
%put NOTE: Description: %bquote(&analdesc.);
%put NOTE: Version: &analver.;
%put NOTE: Reason: &analreas.;
%put NOTE: Purpose: &analpurp.;
%put NOTE: Dataset: &analds.;
%put NOTE: Variable: &analvar.;
%put NOTE: Analysis set: &analsetid.;
%put NOTE: Data subset: &datasubsetid.;
%put NOTE: Method: &method.;
%do ig = 1 %to &ngroupings.;
  %put NOTE: Grouping &&groupingord&ig.: &&groupingid&ig. (result by group: &&groupingresby&ig.);
%end;
%do ir = 1 %to &nrefops.;
  %put NOTE: Referenced operation &&refopord&ir.;
  %put NOTE: Relationship &&refoprelid&ir.;
  %put NOTE: Analysis: &&refopanalid&ir.;
%end;

* Run the analysis method;
%run_method(mdlib=&mdlid., datalib=&datalib., ardlid=&ardlib.,
  method=&method., analid=&analid., analsetid=&analsetid.,
  datasubsetid=&datasubsetid., analds=&analds., analvar=&analvar.,
  groupingids=&groupingids., debugfl=&debugfl.);

* Write completion message to log;
%put NOTE: Analysis &analid. completed;
%put NOTE: =====;
```

Run an analysis method

```
%macro run_method ( mdlib=, datalib=, ardlib=, method=, analid=,
    analsetid=, datasubsetid=, analds=, analvar=, groupingids=, debugfl=N );

    /* Get operation details;
    %local method methname methlabel methdescr
        noperations opords opids;
proc sql;
    select name, label, description into :methname, :methlabel, :methdescr
        from &mdlid..analysismethods
        where id = "&method.";
    select count(*) into :noperations
        from &mdlid..methodoperations
        where id = "&method.";
    select operation_order, operation_id
        into :opords separated by '|', :opids separated by '|'
        from &mdlid..methodoperations
        where id = "&method."
        order by 1;

quit;
```

Run an analysis method

```
%* Build a work dataset including all relevant variables;
%build_work_dataset(mdlib=&mdlib., datalib=&datalib., analds=&analds.,
    analvar=&analvar., analsetid=&analsetid.,
    datasubsetid=&datasubsetid., groupingids=&groupingids.,
    fmtlib=&ardlib., debugfl=&debugfl.);

%* Loop through analysis operations;
%do iop = 1 %to &noperations.;

    %* Execute this operation;
    %run_operation(mdlib=&mdlib., datalib=&datalib.,
        opid=&&opid&iop., methid=&methid., analid=&analid.,
        analsetid=&analsetid., datasubsetid=&datasubsetid.,
        groupingids=&groupingids., analds=workds, analvar=&analvar.,
        ard=&ardlib..ard, debugfl=&debugfl.);

%end;
```


Run an analysis operation

```
/* Create a work version of the ARD with a row for each expected result;
%outline_ard(ardlib=work, mdlib=&mdlbr., analid=&analid., opid=&opid.,
groupingids=&groupingids., dsin=&analds., dsout=work.ard);
```

	resultGroup1_groupLabel	resultGroup1_groupValue	resultGroup2_groupingId	resultGroup2_groupId	resultGroup2_groupLabel	resultGroup2_groupValue	rawValue
1	Placebo	AnlsGrouping_01_Trt_1	AnlsGrouping_03_AgeGp	AnlsGrouping_03_AgeGp_1	< 65 years	AnlsGrouping_03_AgeGp_1	
2	Xanomeline Low Dose	AnlsGrouping_01_Trt_2	AnlsGrouping_03_AgeGp	AnlsGrouping_03_AgeGp_1	< 65 years	AnlsGrouping_03_AgeGp_1	
3	Xanomeline High Dose	AnlsGrouping_01_Trt_3	AnlsGrouping_03_AgeGp	AnlsGrouping_03_AgeGp_1	< 65 years	AnlsGrouping_03_AgeGp_1	
4	Placebo	AnlsGrouping_01_Trt_1	AnlsGrouping_03_AgeGp	AnlsGrouping_03_AgeGp_2	≥ 65 years	AnlsGrouping_03_AgeGp_2	
5	Xanomeline Low Dose	AnlsGrouping_01_Trt_2	AnlsGrouping_03_AgeGp	AnlsGrouping_03_AgeGp_2	≥ 65 years	AnlsGrouping_03_AgeGp_2	
6	Xanomeline High Dose	AnlsGrouping_01_Trt_3	AnlsGrouping_03_AgeGp	AnlsGrouping_03_AgeGp_2	≥ 65 years	AnlsGrouping_03_AgeGp_2	

```
groupingids=&groupingids., dsin=&analds., dsout=work.ard,
debugfl=&debugfl.);
```

```
%end;
```

```
%else %if &opid. = Mth01_CatVar_Summ_ByGrp_2_pct %then %do;
```

Run an analysis operation

```
%op_catvar_summ_bygrp_pct(analid=&analid., method=&method., opid=&opid.,  
    groupingids=&groupingids.,  
    num_analid=&num_analid., num_opid=&&relopid&irel_num.,  
    den_analid=&den_analid., den_opid=&&relopid&irel_den.,
```

	resultGroup1_groupLabel	resultGroup1_groupValue	resultGroup2_groupingId	resultGroup2_groupId	resultGroup2_groupLabel	resultGroup2_groupValue	rawValue
1	Placebo	AnlsGrouping_01_Trt_1	AnlsGrouping_03_AgeGp	AnlsGrouping_03_AgeGp_1	< 65 years	AnlsGrouping_03_AgeGp_1	14
2	Xanomeline Low Dose	AnlsGrouping_01_Trt_2	AnlsGrouping_03_AgeGp	AnlsGrouping_03_AgeGp_1	< 65 years	AnlsGrouping_03_AgeGp_1	8
3	Xanomeline High Dose	AnlsGrouping_01_Trt_3	AnlsGrouping_03_AgeGp	AnlsGrouping_03_AgeGp_1	< 65 years	AnlsGrouping_03_AgeGp_1	11
4	Placebo	AnlsGrouping_01_Trt_1	AnlsGrouping_03_AgeGp	AnlsGrouping_03_AgeGp_2	≥ 65 years	AnlsGrouping_03_AgeGp_2	72
5	Xanomeline Low Dose	AnlsGrouping_01_Trt_2	AnlsGrouping_03_AgeGp	AnlsGrouping_03_AgeGp_2	≥ 65 years	AnlsGrouping_03_AgeGp_2	76
6	Xanomeline High Dose	AnlsGrouping_01_Trt_3	AnlsGrouping_03_AgeGp	AnlsGrouping_03_AgeGp_2	≥ 65 years	AnlsGrouping_03_AgeGp_2	73

```
* Append work ARD to main ARD;
```

```
%append_addcols(dsbased=&ard., dsnew=work.ard);
```



Future development

- Handle more operations
- Implement display patterns
- Output formats
 - JSON
 - XPT?

Contact Details



Karl Wallendszus

Clinical Trial Service Unit, Oxford Population Health
University of Oxford, UK

karl.wallendszus@ndph.ox.ac.uk

<https://github.com/KarlWallendszus/ard-generator>

