
# ADaM Data in High Definition

Varia Cartledge, Alexion Pharmaceuticals, Boston, USA

## ABSTRACT

Define.xml is a critical submission component to allow regulatory reviewers to understand the clinical evidence underlying clinical study reports. In producing define.xml for ADaM, programmers and statisticians have the opportunity to provide insight and clarity into the flow of the study from data collection through the analysis. But many times, this opportunity is lost to spec writing that does not meet the needs of the define and lack of engagement from study programmers and statisticians in define creation. I will present straightforward guidance for best practices in spec writing to support define.xml for ADaM and share experience in creating Analysis Results Metadata through Pinnacle21 Enterprise with a cross-functional team of programmers and statisticians.

## INTRODUCTION

What story does your data tell a regulatory agency? When a reviewer moves from the messaging presented in Clinical Study Reports and Integrated Summaries to dive in and explore analysis data, are you providing the best tools possible to ensure ease and speed of review? According to the FDA Technical Conformance Guide, “An insufficiently documented data definition file is a common deficiency that reviewers have noted.” (Study Data Technical Conformance Guide, U.S. Department of Health and Human Services Food and Drug Administration, October 2017). But a well-documented data definition file is not just a regulatory obligation, it is an asset to your submission.

Best practices for producing define.xml, whether automated or manual, rely on the flow of information from the programming specifications to the define.xml origins and derivations. But many programming processes treat specifications purely as a communication tool, not as an input for submission deliverables. In the current submissions climate, the programming specifications must have a double function and contain information that will allow both accurate creation of data and high-quality documentation.

While the creation of define.xml has been a requirement for several years, the additional strategic approach of providing Analysis Results Metadata (ARM) increases the urgency of managing specifications and inputs to define.xml. It’s important that while we develop best processes for creating ARM, that we also use this opportunity to ensure that the right stakeholders have input and oversight on this information.

## QUALITY IN, QUALITY OUT

For the purposes of this paper the ADaM specification process will be described as a tabular collection of information, e.g. an excel spreadsheet, with columns describing the properties of variables within datasets.

A common assumption is that programmers most familiar with the study data are providing the derivation definition content for define.xml. This assumes that the specifications that programmers write are sufficiently detailed and appropriate to be transferred directly into the tool that is being used to create the define.

In reality, a programmer with less in-depth knowledge of the study may be responsible for the creation of the define. The define programmer may be faced with spec material that is written in casual language, or contains numerous special characters, or consists mostly of SAS code, or is in some other way inappropriate for submission. In order to meet timelines, it may be the case that the define programmer cannot mark up all the inadequate variable definitions for revision by the study programmer. This means that a less experienced programmer may do time and labor-intensive work that risks misinterpreting the original variable definitions.

The best practice to ensure study programmers are fully engaged in creating define content is to capture the information needed for the define at the same time as the specifications to communicate to the dataset programmer. This information is not always going to be presented identically. To accommodate this, it is recommended to use two separate columns in the specification, one for the definition to program from, and one for the define contents. Study programmers should also be trained on style and content guidance.

## STYLE GUIDANCE FOR DEFINITIONS

The purpose of considering style when creating definitions is to ensure that any assumed reader can read a given definition and follow the logic of traceability from collected data through to the variables used to create tables, listings, and figures. Some of the following guidelines, including guidelines around using standard English punctuation conventions, may seem unnecessary. After all, SAS programmers are generally comfortable with seeing semi-colons at the end of statements. It is the responsibility of teams to assess the overall visual appearance of the define and the definitions therein to ensure that the presentation lends over-all readability and clarity to the content. Regardless of what conventions a team adopts, consistency is a must.

- Write all definitions using complete sentences. Use a capital letter at the start and a period at the end. Do not use semicolons to end sentences.
- Cross-references to other documents or other tabs of the specs (e.g. tabs defining value level metadata or AVISIT calculations) cannot be used verbatim in the define. Any material needed to define the variable should be provided in the dataset tab.
- Cross-references to other variable definitions should not be used. Avoid defining a variable using “same as (other variable)” or “see (other variable)”.
- Informal language should not be used.

- Reference the variable being derived in the definition comment. For example, instead of “='DERIVED' for the derived parameters”, write “PARAMTYP = 'DERIVED' for the derived parameters.”
- Use consistent punctuation. Particularly, pick a style of quotation marks to use throughout. For example, you could standardize on using single, non-directional quotation marks to denote values.
- Use consistent if/then statement language.
- Use spaces before and after equal signs.
- Do not use extended ascii or non-ascii characters such as slanted quotes, double quotes, carriage returns, extra spaces, bullets. Do not use Greek characters or mathematical symbols such as the degree symbol.
- Do not use special formatting such as bold, italics, colored text, or strikethroughs. These will not be carried across into the define, and in the case of strikethroughs, the text may inadvertently be presented as part of the definition.
- Do not use SAS code language in definitions. For example, “in” should be replaced with “contains” or “includes” using the most appropriate word choice for the definition. “Min” and “max” should be described as the minimum value or the maximum value.
- Dataset and variable references should be provided as DATASET.VARIABLE. For extension studies or integrations, dataset references may also need to include a study reference and a reference to whether the data is SDTM or ADaM. These should be provided as follows: nnn.DATASET.VARIABLE, SDTMnnn.DATASET.VARIABLE, or ADAMnnn.DATASET.VARIABLE.
- Definitions must not reference internal directory pathways when referencing input data or documents.

EXAMPLES OF CLEAR DEFINE LANGUAGE

| Specification for Programmer | Comments Column for Define |
|---|---|
| if records from 301 then 'DRUG-INDIC-301 '\|\|' '\|\|SDTM301.MH.EPOCH<br>if records from 302 then '' DRUG-INDIC '\|\|' '\|\|SDTM302.MH.EPOCH<br>when EPOCH from SDTM301 or SDTM302 is missing, then missing | If records are from 301 then SEPOCH = 'DRUG-INDIC-301' concatenated with SDTM301.MH.EPOCH.<br>Else if records are from 302 then SEPOCH = 'DRUG-INDIC' concatenated with SDTM302.MH.EPOCH.<br>When EPOCH from SDTM301 or SDTM302 is missing then SEPOCH is missing. |

All statements are presented as a complete sentence with a leading capital letter and a final period. All values are presented in single, non-directional quotation marks. The variable being defined is referenced in the definition. Equal signs are consistently presented with a space before and after them.

| Specification for Programmer | Comments Column for Define |
|---|---|
| if ASTDT lt TR01SDT then ASTDT - TR01SDT;<br>else if ASTDT>=TR01SDT then ASTDT - TR01SDT +1; | If ASTDT < than TR01SDT then AST01DY = ASTDT – TRT01SDT.<br>Else if ASTDT >= to TR01SDT then AST01DY = ASTDT – TRT01SDT + 1. |

All statements are presented as a complete sentence with a leading capital letter and a final period. Semi-colons are not used to divide statements. The variable being derived is referenced in the definition. Symbols are used consistently.

| Specification for Programmer | Comments Column for Define |
|---|---|
| if AEREL='NOT RELATED' then 1<br>else if AEREL='UNLIKELY RELATED' then 2<br>else if AEREL='POSSIBLY RELATED' then 3<br>else if AEREL='PROBABLY RELATED' then 4<br>else if AEREL='DEFINITELY RELATED' then 5 | If AEREL = 'NOT RELATED' then AERELN = 1.<br>Else if AEREL = 'UNLIKELY RELATED' then AERELN = 2.<br>Else if AEREL = 'POSSIBLY RELATED' then AERELN = 3.<br>Else if AEREL = 'PROBABLY RELATED' then AERELN = 4.<br>Else if AEREL = 'DEFINITELY RELATED' then AERELN = 5. |

All statements are presented as a complete sentence with a leading capital letter and a final period. Equal signs have consistent spacing before and after. Variable being derived is referenced in the definition.

| Specification for Programmer | Comments Column for Define |
|---|---|
| ‘DRUG-INDIC-301 Baseline', will include all records from ADaM301 ADPE and SDTM302 PE where PECAT='Physical Examination' and PEORRES ne '';<br>‘DRUG-INDIC-302 Baseline', will include 302 baseline and 302 post baseline records; | For all records from ADAM301.ADPE and SDTM302.PE where PECAT = 'Physical Examination' and PEORRES is not missing, BASETYPE = ‘DRUG-INDIC-301 Baseline'.<br>For all 302 baseline and 302 post baseline records, BASETYPE = ‘DRUG-INDIC-302 Baseline'. |

The fragment “ne "" is replaced with “is not missing”. References to source data match the style guide. Variable being derived is included in definition. Based on the specification for the programmer there was not enough information to specify variables to determine the 302 baseline basetype. This would be improved by including more specific information about what 302 baseline and post baseline records are.

| Specification for Programmer | Comments Column for Define |
|---|---|
| ' ="Y" if ABLFL='Y' or AVISITN in (19999, 29999)<br><br>Note: only baseline and on study will be summarized. | ANL01FL = 'Y' if ABLFL = 'Y' or if AVISITN is 19999 or 29999.<br>Only baseline and on study will be summarized. |

Double quotation marks are replaced with single quotation marks. The variable being derived is included in the definition. The note is still insufficient and for preparation of the define should be revised to include more information for the reviewer.

| Specification for Programmer | Comments Column for Define |
|---|---|
| =ADAM301.ADCE.MHRATMP; This is a relapse level variable for historic relapse. | The Origin column in the P21 define module should be set to 'PREDECESSOR'. 'ADAM301.ADCE.MHRATMP' should be included as the predecessor, no comment or method would be created for this variable. The note about this being a relapse level variable does not belong on a predecessor variable. |

| Specification for Programmer | Comments Column for Define |
|---|---|
| = SDTM.QS.QSTESTCD<br><br>keep QS records with QSCAT in ('C-SSRS SINCE LAST VISIT', 'C-SSRS BASELINE/SCREENING VERSION')<br>For the derived PARAMCDs, please see AVLM. The following derived PARAMCDs will be created:<br>SUIDB<br>SUIDB1<br>SUIBB<br>SUIBB1<br>SUIDBB<br>SUIDBB1<br>SUIDF<br>SUIBF<br>SUIDBF<br><br>*Example from VLM for PARAMCD SUIDB:*<br>=1 if any of AVALC='Y' when PARAMCD in ('CSS0401A','CSS0402A','CSS0403A','CSS0404A','CSS0405A');<br>=0 otherwise both AVALC='N' when PARAMCD in ('CSS0401A', 'CSS0402A') by USUBJID and ADT | See Computational Methods |

Create this definition in the computational methods document. It may be possible or even desirable to take the information from the value level metadata and turn it into a lookup table to present in the computational methods document.

Reference to the **Value Level tab** is inappropriate for a comment defining the variable.

| Specification for Programmer | Comments Column for Define |
|---|---|
| 'For records from 301, then ADaM301.ADQS.ANL02FL;'<br>For records from 302 then:<br>This analysis flag is only applicable to ADSL.REL02FL='Y'.<br>'="Y" if record is the analysis record for baseline and the scheduled relapse VISITs. Refer Tab ANLxxFL_SV for visit info.<br>1. If ABLFL = 'Y' then ANL02FL ='Y';<br>2. If for the same AVISIT, there are more than 1 records, please flag the one with min(ADT ~~where ADT<=ADSL.EOS02DT~~);<br>3. Don't flag if SDTM302.QS.QSSCAT ='Converted Neurostatus EDSS'<br>4. if for the same visit, there are both records with SDTM302.QS.QSSCAT ='Complete EDSS' and SDTM302.QS.QSSCAT ='Original EDSS', flag only the records with SDTM302.QS.QSSCAT ='Complete EDSS'<br>5. if AVISITN=2900, ANL02FL will be null;<br>6.If ADT>EOS02DT then check:<br>If AVISIT contains ‘First Relapse’ and ADSL.REL102DT= . then null;<br>If AVISIT contains ‘Second Relapse’ and ADSL.REL202DT= . then null;<br>If AVISIT contains ‘Third Relapse’ and ADSL.REL302DT= . then null | For records from 301 ANL02FL = ADAM301.ADQS. ANL02FL.<br>For records from 302 apply the following logic:<br>This analysis flag is only applicable to ADSL.REL02FL = 'Y'.<br>ANL02FL = 'Y' if record is the analysis record for baseline and the scheduled relapse VISITs.<br>If ABLFL = 'Y' then ANL02FL = 'Y'.<br>If for the same AVISIT, there is more than one record, flag the record with the lowest value of ADT.<br>If SDTM302.QS.QSSCAT = 'Converted Neurostatus EDSS' then the record should not be flagged.<br>If for the same visit, there are both records with SDTM302.QS.QSSCAT = 'Complete EDSS' and SDTM302.QS.QSSCAT = 'Original EDSS', flag only the records with SDTM302.QS.QSSCAT ='Complete EDSS'.<br>If AVISITN = 2900 then ANL02FL will be null.<br>If ADT is greater than EOS02DT then check that the following logic applies:<br>If AVISIT contains 'First Relapse' and ADSL.REL102DT is missing then ANL02FL will be null.<br>If AVISIT contains 'Second Relapse' and ADSL.REL202DT is missing then ANL02FL will be null.<br>If AVISIT contains ‘Third Relapse’ and ADSL.REL302DT is missing then ANL02FL will be null. |

Extraneous quotation marks have been removed. The variable being derived is referenced in the definition. Strikethrough text was removed.

The reference to the ANLxxFL_SV tab was removed. Information from a separate worksheet tab should be rolled into the comment for the define. Grammar was updated. Spacing before and after equals signs was added. Mathematical symbols were removed. Directional quotation marks were replaced. Semicolons were replaced with periods. Instances of “=.” were replaced with “is missing”. Sentence fragments were rewritten as full sentences. If it was not appropriate to display the logic without the numerical steps, then this definition would have to be displayed in the computational methods document.

## ANALYSIS RESULTS METADATA

Analysis Results Metadata (ARM) is provided to give reviewers a clear and concise way to understand the traceability and algorithms used to create key endpoint results. The first step in creating robust and useful ARM is establishing which study results are most important for reviewers to understand. It may be tempting to provide ARM contents to display the full narrative of the study, however, selective use of ARM allows reviewers to have direct insight into the methods used to identify critical results in a study. Presenting too much information in the form of an over-extensive use of ARM can detract from clarity. The study statistician or overall program lead statistician should have the ultimate responsibility for selecting which results to prepare ARM for.

To that end, statisticians who may not have interacted much with the define document will need to be trained to understand what the importance of ARM is, how it is presented in the define, and what supporting information is needed to create ARM. While there are examples of ARM provided in the CDISC ARM standards release, it may be beneficial to create examples using studies that statisticians are familiar with to give a clearer view of how the study results will be presented.

Both programmers and statisticians determining what results to include in ARM should be aware that an “analysis result” may be a subset of a display output such as a table or figure.

“When analysis results metadata is provided for a specific result within (a) display, there is no requirement to describe all individual results within a display. For example, analysis results metadata may be provided for complex statistics (e.g., p-values) but not necessarily for descriptive statistics like mean or median.” (Analysis Results Metadata Specification, Version 1.0, CDISC ADaM Metadata Sub-Team, January 27, 2015)

## ARM EXPERIENCE

Our study team comprised study statisticians, study programmers, and a data standards administrator who also served as the define programmer for the process.

The team made the decision to pilot ARM for two Phase III studies being prepared for submission to FDA and PMDA at the end of 2018.

While neither agency required submission of ARM, the utility was considered high enough to make submission practical and desirable. Preparing ARM in advance allowed statisticians and programmers to provide information that anticipated regulatory questions that might arise.

Because the creation of ARM was novel to everyone involved on the team, we developed the process to support the activity as we were working on the project.

To facilitate communication and cross-functional collaboration, the team determined that we needed a template that would contain the information necessary for the generation of ARM. We created a template from Pinnacle21 Enterprise by exporting the excel spec for a define, with the intention of using the template directly as input back into Pinnacle21.

Regardless of which tool is being used to develop ARM, we would recommend using a template as a worksheet for gathering the inputs for ARM.

We began the process with a kick-off meeting to provide programmers and statisticians with visual examples of how ARM would be presented in the define, and to discuss the scope and detail of information that would need to be gathered and entered in the template.

In January 2019 the PMDA issued guidance in their Technical Conformance Guide stating the following:

> Therefore, the definition documents of the ADaM datasets should preferably include Analysis Results Metadata, which shows the relationship between the analysis results and the corresponding analysis dataset and the variables used, for the analyses performed to obtain the main results of efficacy and safety and clinical study results that provide the rationales for setting of the dosage and administration, shown in 4.1.1.3. The Analysis Results Metadata of each analysis

(Revision of Technical Conformance Guide on Electronic Study Data Submissions, Pharmaceuticals and Medical Devices Agency, January 24, 2019)

This provides a good basis for selecting results to include in the Analysis Results Metadata. It would be sensible to establish a base set of expected results at a corporate global standard level; however, decisions would still need to be made at a study level. For example, a company might decide to include “Primary efficacy results only”, but study teams would still need to identify those results in their study.

The team drafted the list of results to appear in the ARM during the kick-off meeting, and subsequently shared the template via email to refine the selection. A process improvement to this step would be to use a shared editing space to maintain the template. It is worth spending extra time on this step to ensure that all stakeholders are satisfied with the planned content of ARM before proceeding.

To support the list of results, the corresponding tables were entered in the Analysis Displays tab of the template.

| | A | B | C | D | E |
|---|---|---|---|---|---|
| 1 | ID | Title | Document | Pages | |
| 2 | Table 14.2.1.1.1 | Time to First Adjudicated On-Trial Relapse Full Analysis Set | | | |
| 3 | Table 14.2.2.2.1.1 | Change from Baseline in EDSS Score at the End of Study Full Analysis Set | | | |
| 4 | Table 14.3.1.2.1.3 | Overview of All Treatment Emergent Adverse Events (TEAEs) by Treatment Group Safety Set | | . | |
| 5 | | | | | |
| 6 | | | | | |

7 ... Documents | **Analysis Displays** | Analysis Results | Analysis Criteria | D ...

For each Analysis Result, an ID was assigned by adding “R”, indicating “Result”, and an incremented number. After some trial and error with importing the template into the tool and generating a define with ARM, we determined that it was helpful to add a leading zero to support results where more than nine results came from a given table.

| Display | ID | Description |
|---|---|---|
| Table 14.1.1.2.1 | Table 14.1.1.2.1 R01 | Baseline EDSS |
| Table 14.1.1.2.1 | Table 14.1.1.2.1 R02 | Baseline HAI S |
| Table 14.1.1.2.1 | Table 14.1.1.2.1 R03 | Baseline mRS |
| Table 14.1.3.2.1 | Table 14.1.3.2.1 R01 | Number of rel |
| Table 14.1.3.2.1 | Table 14.1.3.2.1 R02 | Historical Ann |

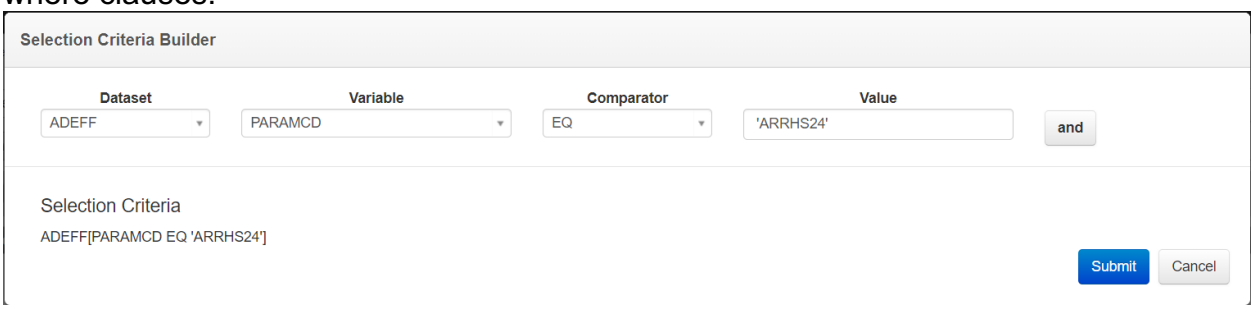
To support each Analysis Result, study statisticians identified which section of the Statistical Analysis Plan (SAP) or Protocol was appropriate to provide documentation. Because this involved selecting the specific text from the SAP relevant to the result, this task belonged to a study team member rather than a define programmer.

Because the SAP itself is not contained within module five of the Electronic Common Technical Document (eCTD), we decided to create an extract of the SAP containing only the sections relevant to the Analysis Results. This document was created as a PDF and a bookmark structure was added. This document was stored in the directory with the define so that links from the define did not have to traverse the eCTD structure.

The programming statements to support each Analysis Result were identified by the study programmers. While it is possible to link directly to program code stored elsewhere, the presentation of the relevant portions of code directly in the define added to the immediate utility of ARM.

The Reason and Purpose fields for Analysis Results could easily be determined by the define programmer based on the protocol and SAP. Controlled terminology is available for these fields and the application of the terminology was self-evident. These did not need to be created by a study programmer or statistician but were reviewed with the rest of the finalized output. The programming context field was set to the version of SAS used for the study.

The most challenging piece of interoperability was managing the Where Clause within the Analysis Criteria tab. The information required to complete this tab was readily available for the define programmer within the annotated tables, listings, and figures. However, the Pinnacle21 Enterprise define tool has a separate logic tool for managing where clauses.



The Where Clause column was left blank on the template intended for importation into Pinnacle21, and the define programmer used the Where Clause Tool to enter the information.

Although the majority of the metadata needed to complete ARM appear on the Analysis Displays, Analysis Results, and Analysis Criteria tabs, any result that takes input from two or more datasets must have a “Join” comment assigned to it to explain the join logic for the datasets:

| ID | Description |
|---|---|
| JOIN | Get denominators for percentages from ADSL (Safety Population) and counts and numerators from ADAE (Treatment Emergent events for Safety Population). Join ADAE with ADSL based on USUBJID keeping only records in ADAE for the numerator. |

This comment must be treated the same way as all comment elements within the define; it should be included in the comments tab of the define specification and the ID used as a reference on the Analysis Results tab. Because the template that the team was using to complete ARM did not include all the tabs of the define specification, this comment was added after the rest of the template metadata had been imported into Pinnacle21 Enterprise.

The define tool provided internal validation and checking to assure that the information imported from the template fit the ARM model. The define programmer revised items within the tool as needed. However, for any revisions that affected wording or content provided by the study programmers and statisticians, it was necessary to communicate outside of the template -> input -> edit process.

In a perfect world, the initial import of the template into the Pinnacle21 Enterprise define tool and addition of the where clause logic would have created a final, submission ready define. However, while it is possible to do a great deal of review on the contents intended for inclusion in the ARM while in spreadsheet form, exporting a viewable define.xml for further review allowed study team members to examine the contents in context as regulatory reviewers would see them.

Once the define.xml with ARM was created and reviewed by study team members, the define template was re-exported from Pinnacle21 Enterprise to be revised and subsequently re-imported into the tool. Any revisions needed to the contents of the where clauses were managed within the tool, not within the template.

While the technical creation of the ARM contents in Pinnacle21 Enterprise had challenges, including the lack of 1:1 mapping from the exported define spec in excel to the GUI inside the tool, the biggest challenges were training and communicating the requirements of ARM to the team while we piloted the process. Building a shared expertise within the community of biostatisticians and statistical programmers will make this a much smoother process in the future.

## CONCLUSION

Define.xml for ADaM is a powerful tool for creating data submissions that will be easy to navigate and understand. The construction of a well-formed define requires a level of technical expertise that often involves a dedicated define programmer. However, the work of developing the key content of the define belongs in the hands of the study programmers and study statisticians. Engaging team members in the define process and ensuring that programmers and statisticians across organizations have a fundamental understanding of the content and purpose of the define is key to building the value of the define and creating workflows that focus on what is meaningful in define creation.

## REFERENCES

**Study Data Technical Conformance Guide**, U.S. Department of Health and Human Services Food and Drug Administration, October 2017

**Analysis Results Metadata Specification**, Version 1.0, CDISC ADaM Metadata Sub-Team, January 27, 2015

**Revision of Technical Conformance Guide on Electronic Study Data Submissions**, Pharmaceuticals and Medical Devices Agency, January 24, 2019

## ACKNOWLEDGMENTS

With great thanks to Frank Menius at Covance, whose collaboration on developing define processes and brainstorming around define were critical to the writing of this paper.

## CONTACT INFORMATION

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

Varia Cartledge
Alexion Pharmaceuticals
121 Seaport Blvd
Boston, MA 02210
Work Phone: +1 857.293.3329
Email: varia.cartledge@alexion.com