PharmaSUG 2015 - Paper DS07

The Best Practices of CDISC ADaM Validation Checks: Past, Present, and Future

Shelley Dunn, d-Wise, Morrisville, NC
Ed Lombardi, Agility Clinical, Carlsbad, CA

ABSTRACT

The CDISC ADaM Validation Check v1.3 document was recently published in March 2015 (following the initial submission of this abstract), with the previous version last revised in 2012 (v1.2). The v1.2 checks include specific guidelines and rules for validation of ADaM data sets based on the ADaM Implementation Guide V1.0 released in 2008. Since this time, the ADaM Team has published "The ADaM Basic Data Structure for Time-to-Event Analysis" (BDS-TTE) and the "ADaM Data Structure for Adverse Event Analysis" (ADAE). As new ADaM documents are released the ADaM Compliance Sub-Team has been preparing machine-testable failure criteria to support new requirements. Additionally, as new versions of the ADaMIG or other analysis documents are prepared, e.g., Occurrence Data Structure and Data Structure for Integration, the checks will be updated to reflect the current decisions made by the ADaM Team.

What are the best practices when it comes to ADaM validation? Will running validation checks, based on the CDISC ADaM Validation Check document, be sufficient to validate any analysis data structure? Many companies in the industry rely on a single set of checks provided by an outside vendor to validate their ADaM data sets. Unfortunately this practice alone is not sufficient to ensure all ADaM structure requirements are followed. ADaM validation is more than just running a tool and hoping for no errors. While the ADaM Validation Checks provide a comprehensive list of machine-testable failure criteria checks based on the ADaM documents mentioned above, not every nuance of ADaM particulars can be tested. Therefore, it becomes necessary to supplement these checks with both sponsor specific checks as well as manual data structure reviews. To understand what additional checks are required, one must first understand the scope of the current checks and what text from the ADaM documents allows for machine-testable criteria.

This paper will focus on ways to improve the quality of validating analysis data structures by providing a look at the past, present, and future of ADaM Validation Checks. Regardless of where you are on the spectrum, from being new to ADaM to being an author of the ADaMIG, this paper will guide you to understand what you can do to help implement best practices around the ADaM Validations checks.

Past: Starting with the difference between the ADaM Validation Check document and programmed checks will lead to an overview of the tools that have been available in the past to run validation checks

Present: In the present we will take a deeper dive into the scope of the ADaM Validation Checks and visit some of the challenges encountered by the ADaM compliance sub-team when adding the TTE and ADAE checks. As more ADaM documents are created and updated, there is an increasing need to shore up any loose language needed to write concrete checks

Future: The future of ADaM compliance is closely tied to the ADaM documents as well as educating users about the compliance tools and the necessity to supplement with sponsor defined checks

Picture 1. Back To The Future

INTRODUCTION

To understand ADaM validation one must first be able to draw a distinction between the ADaM Validation Check document, which is a specifications documents, and vendor tools or programs, which are designed to run validation checks against study data. Part of this paper is designed to dispel the myth that ADaM compliance is defined by running a single tool. Throughout this paper multiple methods of checking for ADaM compliance will be explained to illustrate why ADaM validation best practices require a diversified approach. Effective compliance relies on an intelligent combination of “out of the box checks,” sponsor defined programmed checks, and manual checks.

The terms “compliance” and “validation” are, at times, used interchangeably throughout this paper. In general, the terms are meant to convey the following:

- **Compliance:** Following the rules in the ADaMIG, ADAE, TTE, and any other published ADaM document
- **Validation:** Methods to check for compliance
- **CDISC ADaM Validation Checks:** Specification document outlining validation checks as defined by the CDISC ADaM validation sub-team
- **Validation Program(s):** A program or programs written to run validation checks
- **Validation Tool:** Software used to run validation checks

PAST

“We’re gonna go back in time.” Many people don’t know the source of ADaM Validation Checks. Contrary to SDTM, validation checks for ADaM were developed by the ADaM team. After publishing the ADaMIG v1.0, the ADaM compliance sub-team created the first version of the ADaM Validation Checks document. This document included machine-testable failure criteria to help users of the ADaM model test compliance for ADSL and BDS data sets. “This text is intended for use as a requirement specification which could be implemented in a variety of programming languages.” These checks served as specifications to allow users to produce their own compliance tools and validation software. The compliance sub-team produced two more versions of the document with the latest version (until recently) being published in 2012. The latest version published in March of 2015 is discussed later in this paper.

The validation checks were developed based on ADaMIG text. For example, the following text from the ADaMIG, “There is only one ADSL per study.” corresponds to the validation check #1 below.

ADAM DOCUMENTS

2.3.1 The ADaM Subject-Level Analysis Dataset ADSL
ADSL contains one record per subject, regardless of the type of clinical trial design. ADSL is used to provide the variables that describe attributes of a subject. This allows simple combining with any other dataset, including SDTM domains and analysis datasets. ADSL is a source for subject-level variables used in other analysis datasets, such as population flags and treatment variables. There is only one ADSL per study. ADSL and its related metadata are required in a CDISC-based submission of data from a clinical trial even if no other analysis datasets are submitted.

Display 2. Text from the ADaMIG (See Table 1)

VALIDATION CHECKS DOCUMENTS

Check Number	ADaMIG Section Number	Text from ADaMIG	ADaM Structure Group	Functional Group	ADaM Variable Group	Machine-Testable Failure Criteria
1	S2.3.1	There is only one ADSL per study	ADSL	Present/ Populated	General	ADSL dataset does not exist

Table 1. Example of ADaM Validation Check based on Text from ADaMIG (see Display 1)

In the case of compliance, it is essential to know the difference between validation checks and validation tools and what each of these do and do not do. The scope of the ADaM checks is best broken down into the following points:

- The validation checks are a specifications document
- The validation checks are based on machine-testable failure criteria
- Outside programming is required to implement the checks

- Non machine-testable compliance is not included
- Sponsor specific compliance is not included

COMPLIANCE TOOLS

Many companies rely on an outside vendor's tool (e.g., OpenCDISC, SAS Clinical Standards Toolkit (CST), home grown SAS programs, etc.) as their primary method to validate ADaM data sets. A major benefit of this approach is the time saved not having to program the validation checks. The widespread use of OpenCDISC (including use by the FDA) makes this "out of the box" tool almost essential to be run. However, companies have to understand these essential points when it comes to using outside vendor tools:

- They may not be implementing all of the validation checks published by ADaM
- The logic may not be implemented as intended
- They may performs additional checks outside of CDISC checks

Some companies' principle approach to ADaM compliance is focused on reducing the number of warnings and errors that come from their tools regardless of the cause. The ADaM Validation Checks are pass/fail and do not specify warnings or errors. Understanding when a warning or error, as defined by a tool, is relevant and should be fixed and when they should simply be explained in a Reviewer's guide is an important part of compliance. ADaM validation is more than running a tool and hoping for no errors. One of the most important aspects of good compliance is to understand the scope of the tool(s) you are using.

Always keep the big picture in mind when determining compliance of ADaM data sets. The FDA is interested in data quality and not whether study data passes a set of checks. Providing explanations to the FDA can assist them in reviewing the data. Whereas the need to fix study data based on the output of every check is missing the point.

The following examples are error messages from OpenCDISC versions 1.4 and 1.5. Notice that in each case, the check fired erroneously. The reader should be aware that the most current version of OpenCDISC is 2.0 where some of these issues are likely fixed.

Domain	Record	Variables	Values	Rule ID	Message
ADSL		VARIABLE, LABEL	RACEGR1N, Pooled Race Group 1 (N)	AD0018	ADaM dataset variable label mismatch
ADAE		VARIABLE	AVAL, AVALC	AD0198	Neither AVAL nor AVALC are present in dataset
ADLB	36553	BASE, CHG, AVAL	0.59, 0, 0.59	AD0223	Calculation error: CHG not equal to AVAL - BASE

Table 2. Example of OpenCDISC report messages

In the first example, the label correctly matches the implementation of RACEGRyN per the IG. In the second example, ADAE is not expected to have AVAL or AVALC. It appears that BDS checks are being run on the ADAE data set. In the third case, the calculation, 0.59 = 0.59 – 0, is also correct. All three cases are examples of false positives. Anyone simply trying to "fix all errors" without understanding the reason for these messages will waste time focusing on issues which may be caused by a bug in the tool and are not cases of incorrect implementation.

PRESENT

The most current version of the ADAM Validation Checks was published in March 2015 as v1.3.

Over the past year the CDISC ADaM compliance sub-team has been working together to implement changes, corrections, and/or updates identified in the published v1.2 checks as well as add new checks to cover the most recently published ADaM documents, The ADaM Basic Data Structure for Time-to-Event Analysis (BDS-TTE) and ADaM Data Structure for Adverse Event Analysis (ADAE).

THE ADAM BASIC DATA STRUCTURE FOR TIME-TO-EVENT ANALYSIS (BDS-TTE)

Time-to-event (TTE) checks are now included in the ADaM Validation Check document. Since TTE is a type of BDS structure, all current BDS checks apply to BDS-TTE checks. Additionally, new checks have been added that apply just to time-to-event data.

One of the first questions to be tackled with TTE is how to determine if a data set is a Time-to-Event data set or if certain records within a BDS data set are TTE records. The distinguishing variable is the CNSR (or Censor) variable. Minor differences are evident in the description of the CNSR variable in the ADaMIG and the CNSR variable in the ADaM BDS for Time-to-Event Analysis document.

Variable Name	Variable Label	Type	Codelist / CT	Core	CDISC Notes
CNSR	Censor	Num	*	Cond	CNSR is a required variable for a time-to-event analysis dataset, though it is a conditionally required variable with respect to the ADaM BDS. For example, CNSR = 0 for event and CNSR > 0 for censored records.

Table 3. Description of CNSR in the ADaM BDS for Time-to-Event Analysis document

While CNSR is technically a conditional variable in a BDS, the CDISC notes in the TTE document clearly indicate that the CNSR variable is a required variable for time-to-event analysis. This forms the basis of the TTE validation checks.

To run a validation check on a TTE data set or a TTE record, the first step is always checking that the variable CNSR is present in the data set.

Check Number	ADaMIG Section Number	Text from ADaMIG	ADaM Structure Group	Functional Group	ADaM Variable Group	Machine-Testable Failure Criteria
245	TTE S1	In these studies, the basis of analysis is the time from a defined starting point to the time of occurrence of the event of interest.	BDS	Present/ Populated	Time to Event Variables	CNSR is present and STARTDT is not present.

Table 4. Example of a TTE check (Note: Time-to-Event data sets are a subset of BDS)

ANALYSIS DATA MODEL (ADAM) DATA STRUCTURE FOR ADVERSE EVENT ANALYSIS (ADAE)

Unlike BDS, ADAE is a horizontal structure and contain a large number of possible variables. When generating validation checks for ADAE, each variable had to be considered. The result is a number of very similar checks for different variables. Below is an example of a check for a required ADAE variable.

Check Number	ADaMIG Section Number	Text from ADaMIG	ADaM Structure Group	Functional Group	ADaM Variable Group	Machine-Testable Failure Criteria
260	ADAE S4.1.3	Required Variable: AETERM	ADAE	Present/ Populated	ADAE Variables	AETERM is not present

Table 5. Example of an ADAE check

It is worthwhile to note that although separate checks are created for different variables within the specification document, it is probably more efficient to program a single check and apply it to all checks of a similar type. The "Functional Group" classification can aid in grouping checks. Functional group definitions can be found in the ADaM Validation Checks document.

- Value Consistency
- Metadata
- Present/Populated
- Controlled Terminology
- Valid Value

CHALLENGES

The ADaM compliance sub-team has encountered a number of challenges when writing ADaM Validation Checks including

- Vague language within the ADaM documents

- Different interpretations of ADaM text
- Basic logic used to link similar concepts
- Lack of rules to define data structures

To illustrate these points, consider the following examples.

Vague Language

ADAE, Section 4.1 contains the following text, “In general, the analysis version of an SDTM variable is named by replacing the “AE” prefix with an “A” for analysis.” Here the preface says “in general” and therefore does not allow for a rule to be created. This language implies that the analysis version of SDTM.AEACN should be called AACN in an ADAE data set; however, it is not a requirement and therefore cannot be included in the ADaM Validation Check document. However, this does not preclude one from adding this type of naming conventions to their own sponsor-defined checks.

Different Interpretations

The ADaMIG contains the variable ONTRTFL with the label “On Treatment Record Flag.” Based on the concept of “on treatment,” although not spelled out in the IG, the ADaM compliance sub-team implemented checks to ensure ONTRTFL is consistent with TRTSDT, Date of First Exposure to Treatment, and TRTEDT, Date of Last Exposure to Treatment. Namely that ONTRTFL will be “Y” when ADT falls between TRTSDT and TRTEDT and ONTRTFL will be null when ADT falls outside TRTSDT and TRTEDT. However, it is possible to interpret “on treatment” differently without a clear and concise definition. Is a subject always considered to be “on treatment” between the first and last dates of exposure? In some studies, on treatment might go beyond the final dose. In this situation, the sponsor has to decide to either create a custom variable or explain the failures in their Reviewer’s Guide.

Variable Name	Variable Label	Type	Codelist / CT	Core	CDISC Notes
ONTRTFL	On Treatment Record Flag	Char	Y	Perm	Character indicator of whether the observation occurred while the subject was on treatment.

Table 6. Description of ONTRTFL in the ADaMIG: Table 3.2.6.1 Flag Variables for BDS Datasets

Basic Logic

When two separate statements identified in an ADaM document are combined, they can lead to rules that may or may not be anticipated. For example the ADaMIG says the following:

The lower case letters “xx”, “y”, and “zz” that appear in a variable name or label must be replaced in the actual variable name or label using the following conventions. The letters “xx” in a variable name (e.g., TRTxxP, APxxSDT) refer to a specific period where “xx” is replaced with a zero-padded two-digit integer [01-99]. The lower case letter “y” in a variable name (e.g., SITEGRy) refers to a grouping or other categorization, an analysis criterion, or an analysis range, and is replaced with a single digit [1-9]. The lower case letter “zz” in a variable name (i.e., ANLzzFL) is an index for the **zz**th record selection algorithm where “zz” is replaced with a zero-padded two-digit integer [01-99].

Display 2. Text from the ADaMIG, Section 3, page 11

The ADaMIG goes on to define a number of variables containing the lower case letters “xx,” “y,” and “zz,” e.g., TRTxxP, SITEGRy, ANLzzFL. Applying logic to connect these two concepts leads to the implication that variables such as TRT**WW**P, SITEGR**W**, and ANL**WW**FL would be invalid. So while it is not reasonable for every document to explain every invalid variable name, the ADaM compliance sub-team applies certain logic to enable checks to be written to account for some naming conventions.

Lack of Rules

Another challenge is the lack of rules to determine the data set structure based on the name of the data set. With the exception of ADSL, it is not necessarily clear what data set structure is used based on the name of an ADaM data set. Similarly, if an ADaM data set has an ADAE structure or is a TTE version of a BDS structure, there are no requirements for how to name these ADaM data sets.

This is a challenge for ADaM Validation Checks because structure is variable dependent. That is, if an ADaM data set contains PARAMCD it is a BDS and if CNSR is always present and populated it is a BDS-TTE data set. This precludes being able to write checks to check for PARAMCD in a BDS or CNSR in a BDS-TTE. This could be resolved if the ADaM Team implemented standard naming conventions for data sets.

FUTURE

"Roads? Where we're going, we don't need roads."

Picture 2. Back To The Future II

But we will need more validation checks. There are numerous ADaM documents being developed and users of these documents will need to make sure they are complying with the intended implementation. These new documents include ADaMIG v1.1, Occurrence Data Structure (OCCDS), Data Structure for Integration, and numerous therapeutic area specific documents.

Each of these documents will contain new text where new validation checks could be hiding. Anyone who has run the OpenCDISC validation on their ADaM data sets is familiar with the message about the unknown configuration for certain data sets which means the data set is not being validated at all. Creating validation checks for new data structures is vital to ensuring quality implementation.

FDA study data validation rules for ADaM are likely to be published as well. As of March 2015, the FDA has published validation rule for SEND and SDTM. These rules are similar to already published rules. ADaM compliance is not stagnant and will continue to change and improve over time. New tools, paper, ideas, OpenCDISC versions, FDA validation rules and ADaM validation rules will change the landscape of ADaM compliance.

BEST PRACTICES FOR VALIDATION

PREREQUISITES

- Step 1: Understand the limitations of the documents and tools to be used
- Step 2: Consider additional checks based on any gaps in the documents and tools
- Step 3: Determine what tool(s) will be used to run validation checks
- Step 4: Define additional sponsor-defined checks (both programmable and manual) – See example below
- Step 5: Program sponsor-defined checks
- Step 6: Create a workflow to ensure that all checks are completed

PRACTICE

- Step 1: Run "out of the box checks." Keep in mind that these may not be entirely consistent with the CDISC ADaM Validation Checks
- Step 2: Review the "out of the box checks" output to determine what needs to be updated and what is acceptable
- Step 3: In the Reviewer's Guide, provide an explanation for all of the findings. Explain in the Reviewer's Guide why the findings are acceptable or unavoidable
- Step 4: Run programmed sponsor-defined checks
- Step 5: Review the sponsor-defined checks output to determine what needs to be updated and what is acceptable
- Step 6: Perform a manual review of the ADaM data sets based on pre-identified rules

As seen from the "Past," "Present," and "Future" sections, the checks are tied to the published ADaM documents and only contain checks that are machine-testable. To fully ensure ADaM compliance, companies will need to define additional checks and implement a variety of machine-testable and manual checks. The steps above are designed to give you a comprehensive plan for ADaM compliance. Below are examples of how to go about defining "sponsor-defined" and "manual" checks.

"HELLO MCFLY!"

ADaM Validation Checks cannot possibly check for every ADaM requirement in an ADaM data set. Without manual checks, you are setting yourself up for failure. But what manual checks should you check? The list is probably endless. However, anytime a rule cannot be tested by a software program it is a candidate for a manual check.

The following italicized text is from the ADaM Validation Check document Scope:

The following are examples of aspects of ADaM compliance that cannot be tested by a software program:

- *Within section 4.3.1 of the Implementation Guide the text says "Include all observed and derived rows for a given analysis parameter."*
- *Within section 4.6.1 of the Implementation Guide the text says, "To identify population-specific analyzed rows, use population-specific indicator variables."*
- *Many ADaM variables are conditionally required (required if a condition is true), but some conditions are not testable by a software algorithm.*
- *One of the key components of ADaM is the inclusion of thorough and well defined metadata. The thoroughness and clarity of metadata cannot be tested by a machine-readable algorithm but is necessary to enable the traceability that ADaM requires.*

While the examples above are rules that must be followed while implementing ADaM, they cannot be tested by a machine-readable algorithm. Instead, a complete assessment of compliance must be based on an understanding of the scope of the study data and the analyses which the datasets should support coupled with the published validation checks within this document and the general rules and principles published in the ADaM Implementation Guide.

The following italicized text is from the ADaMIG Fundamental Principles and describes the concepts of "traceability" and "analysis-ready," two concepts that cannot be tested by a software program.

- *Analysis datasets and associated metadata must provide traceability to allow an understanding of where an analysis value (whether an analysis result or an analysis variable) came from, i.e., the data's lineage or relationship between an analysis value and its predecessor(s). The metadata must also identify when analysis data have been derived or imputed.*
- *Analysis datasets should have a structure and content that allow statistical analyses to be performed with minimal programming. Such datasets are described as "analysis-ready." Note that within the context of ADaM, analysis datasets contain the data needed for the review and re-creation of specific statistical analyses. It is not necessary to collate data into "analysis-ready" datasets solely to support data listings or other non-analytical displays.*

EXAMPLE

Let's look at a specific example of ADaM text that cannot be used to define a machine-testable check.

- **Example:** Text from Time-To-Events: "It is generally recommended that time-to-event data be stored separately from non-time-to-event data even if they both fit within the ADaM BDS."

This example is not included in the ADaM Validation Checks for a number of reasons. The most obvious is the use of the words "generally recommended." Since "generally recommended" is not a black and white rule, this 'recommendation' cannot be an ADaM validation check. Below, we will consider this specific example and offer other ways to include this 'recommendation.'

Sponsor-Defined Check Example

In most companies, sponsors will have their own standards that build upon the ADaM documents. For example, many sponsors may define additional rules to use when generating ADaM data sets that go above and beyond the limitations of the general ADaM documents. Specific examples may include creating CT for variables where ADaM does not define CT, determining naming conventions for new ADaM variables not in the current documentation, defining rules regarding the ordering of variables, and so on. These types of sponsor specific rules are not included in the ADaM Validation Checks and must be programmed and checked independently.

- Based on the above example, a sponsor may decide to make this a requirement. That is, a sponsor might maintain the rule "It is 'required' that time-to-event data be stored separately from non-time-to-event data." In this case a Sponsor-Defined machine-testable check could be defined, programmed, and run on a specific TTE data set(s) to ensure non-TTE data is not included.

Manual Check Example

Alternatively, a sponsor may opt to require this rule in some cases and not in others. Therefore, they might opt to check this manually instead of programmatically.

- Based on the above example, a sponsor may have certain requirements with regards to TTE data sets for each different Therapeutic Area (TA). For TAs requiring different data sets for TTE and non-TTE data, a manual check can be done very easily to ensure that more than one BDS ADaM data set exists, one for TTE and one or more for non-TTE data.

CONCLUSION

To implement best practices of ADaM validation, it is vital to understand what the CDISC ADaM Validation Checks contain and what they do not contain. If your company uses "out of the box checks" to perform validation, it is imperative that you understand if the tool contains only the ADaM Validation Checks, if they contain any additional checks, and how these checks have been implemented.

The ADaM documents do not always contain black and white rules. When the text contains some grey areas, such as vague language, implementation options, undefined rules, etc., it is not possible to include these topics or concepts within the ADaM Validation Check document. It is critical to those carrying out compliance on ADaM data sets to understand the implications. That is, it is also necessary to define and check non-machine-testable criteria type checks. Some rules can be defined by the sponsor and programmatically checked; whereas, other checks will require a manual review. Only after all levels of checks have been completed can one be assured of ADaM compliance.

The best practice for ensuring ADaM compliance is to implement a series of validation checks to include all of the following: run checks for the ADaM Validation Checks (e.g., third party vendor checks), document discrepancies in your Reviewer's Guide, and carry out sponsor-defined checks (both programmatic and manual). Companies relying solely on one method are severely limiting themselves when it comes to guaranteeing all ADaM data sets are compliant with CDISC standards and FDA regulations.

REFERENCES

- CDISC Analysis Data Model Team. "Analysis Data Model (ADaM) Implementation Guide V1.0." CDISC web site. December 17, 2009. Available at www.cdisc.org\.
- CDISC Analysis Data Model Team. "Analysis Data Model (ADaM) Data Structure for Adverse Event Analysis." CDISC web site. May 10, 2012. Available at www.cdisc.org\.
- CDISC Analysis Data Model Team. "Analysis Data Model (ADaM) The ADaM Basic Data Structure for Time-to-Event Analyses." CDISC web site. May 8, 2012. Available at www.cdisc.org\.
- CDISC ADaM Validation Team. "Analysis Data Model (ADaM) Validation Checks v1.2." CDISC web site. July 5, 2012. Available at www.cdisc.org\.
- CDISC ADaM Validation Team. "Analysis Data Model (ADaM) Validation Checks v1.3." CDISC web site. March 17, 2015. Available at www.cdisc.org\.

CONTACT INFORMATION

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

Name: Shelley Dunn
Enterprise: d-Wise
Address: 1500 Perimeter Park Drive, Suite 150
City, State ZIP: Morrisville, NC 27560
Work Phone: 919-334-6089
Fax: 888-563-0931
E-mail: Shelley.Dunn@d-wise.com
Web: www.d-wise.com
Twitter: twitter.com/dWiseTech

Name: Ed Lombardi
Enterprise: Agility Clinical
Address: 6005 Hidden Valley Road, Suite 170
City, State ZIP: Carlsbad, CA 92011
E-mail: ELombardi@agility-clinical.com
Web: http://www.agility-clinical.com
Twitter: twitter.com/AgilityClinical