**PharmaSUG2011 - Paper IB04**

# Brave New World: How to Adapt to the CDISC Statistical Computing Environment

Jeff Abolafia, Rho, Inc., Chapel Hill NC
Frank DiIorio, CodeCrafters, Inc., Philadelphia PA

## INTRODUCTION

Recent years have seen CDISC standards such as SDTM, ADaM, and CDASH evolve from “what a good idea” to “must have for an FDA submission.” That is, if you’re in the FDA submission business, you’re now also in the CDISC business. Viewing these standards simply as a collection of data structures and documentation requirements is tempting, but ultimately problematic. Their impact extends throughout the statistical computing environment of any organization dealing with the FDA.

This paper presents an overview of the impact of CDISC standards, primarily SDTM and ADaM, on the traditional statistical computing environment. The paper divides the discussion into four major sections. The first, project operations, discusses the changes to work flow, personnel, client management, and skill set requirements required by the standards. The second section describes the technological requirements such as the metadata architecture and tools. The third section surveys validation requirements, emphasizing the increased scope and complexity of the validation task. The last section identifies expanded training required for both the standards and the changed workflow that they create.

The reader should come away from the paper with a realistic idea of the scope of organizational, technological, skill set and other changes necessary to effectively produce CDISC-compliant deliverables.

## WHO WE ARE

Before delving into the specifics of CDISC implementation, it’s important to understand how we (that is, Rho, Inc.) arrived where we are today. It helps frame the discussion, and may resonate with many readers. Rho was established as a Contract Research Organization (CRO) in 1985. Privately held, it has grown steadily over the years and currently has over 300 employees, attracting a variety of pharmaceutical clients ranging in size from small, one-compound startups to large, established companies.

There are several features of Rho’s history that have influenced [where we are today]:

- **Academic Origins.** Not unlike many CROs, the founders and early employees of the company came from academic backgrounds. This, of course, supplied the company with the requisite statistical and programming expertise. Another academic legacy that has endured over the years has been the company’s willingness to support research into new technologies and methodologies
- **Early Use of Metadata.** Early in the company’s history, senior personnel recognized the need to move dataset and display specifications out of word-processing files and into a file format that would support automation. Our experience with metadata - its design, user interface, and toolset – made us well prepared for the additional metadata requirements related to CDISC implementation.
- **SAS-Centric.** Given its Research Triangle, NC roots and personal connections with founders and early employees of SAS ® Institute, it isn’t surprising that much of the initial data entry, statistical, and submission-related software was developed using SAS. While this emphasis on use of SAS software has produced reliable, high-quality tools and programmers, it hasn’t blinded the company to exploring alternative and complementary technologies.
- **Mandatory Flexibility of the CRO.** A viable, competitive CRO must be able to economically satisfy the needs of different clients. This is a different environment than, say, a large pharmaceutical company, which can establish a single process for the life cycle of a project, from data collection to submission to the FDA. The CRO must be able to receive data in many formats, create or import metadata, and produce deliverables that meet the client’s. The adaptable metadata-focused architecture we developed enabled us to view CDISC standards as complementary to our current processes, rather than a completely new paradigm.

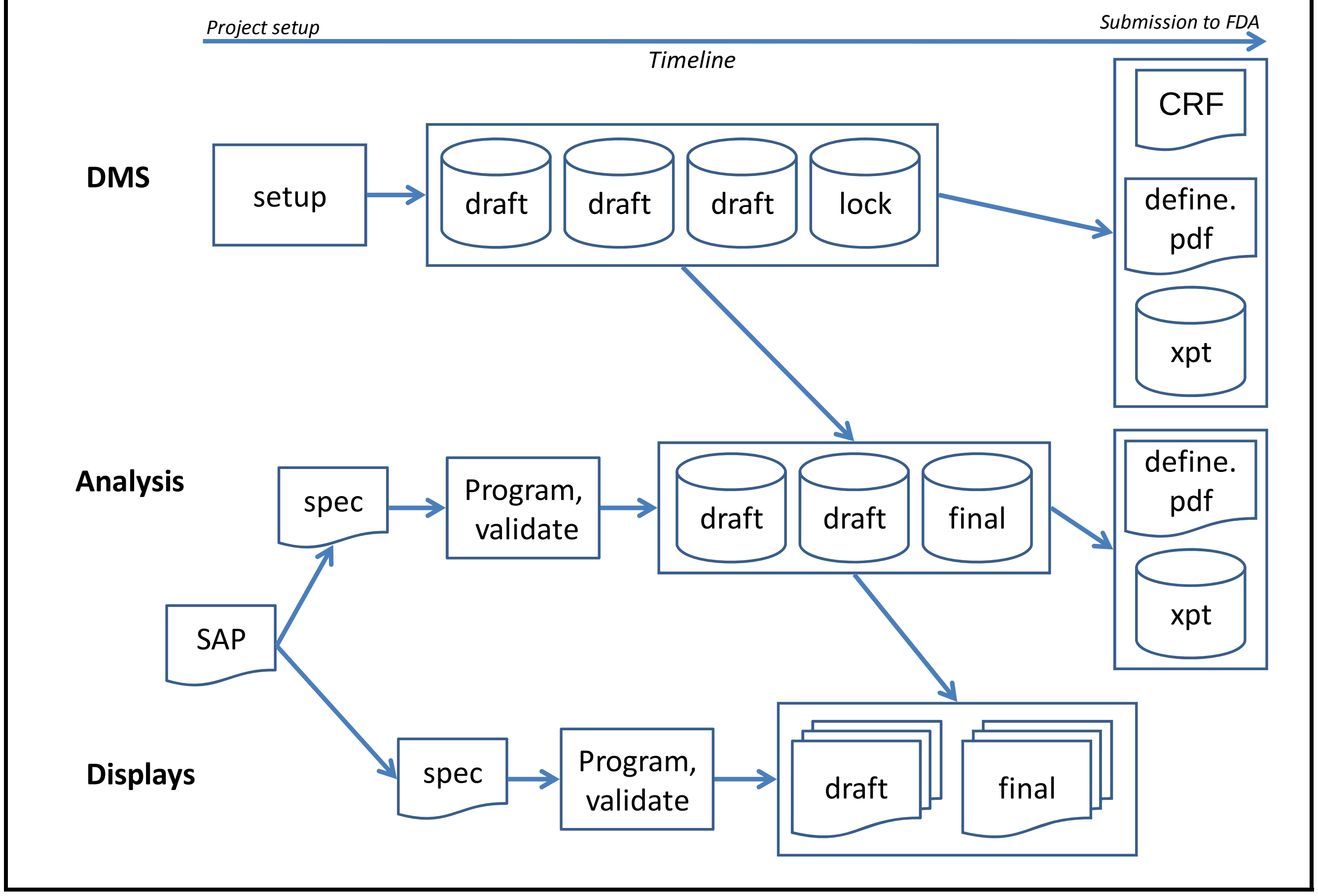
With this background in mind, let’s take a look at the statistical computing environment before the introduction of CDISC standards.

## HOW IT USED TO BE

This section presents an admittedly simplified overview, represented by **Figure 1**, of the pre-CDISC statistical computing environment. It focuses on work flow, technical requirements, skill sets, and validation. Statistical computing and statistics staff is typically responsible for creating analysis datasets, data displays, statistical analysis, submission databases, and supporting documentation. In some cases, statistical computing programmers may also be charged with various data management programming tasks.

Before SDTM, analysis datasets were created directly from the clinical data. Statisticians and statistical programmers, with input from data management, could begin the analysis datasets creation process while clinical data was being collected. Display creation could begin soon thereafter. The actual timing depended on the length of the data collection process and whether the Statistical Analysis Plan called for interim analysis. It should be noted that in this scenario there are very few hand-offs. Statisticians and statistical programmers use the clinical data from data management as the source for analysis datasets (source data may also come from labs or ECG reading centers). In some cases, feedback from statisticians leads to modifications in the data management system.

**Figure 1: Clinical Trial Workflow Before CDISC**



Statisticians and statistical programmers are typically charged with creating the submission database and supporting documentation. Prior to CDISC, these deliverables were prepared under the FDA guidance “Providing Regulatory Submissions in Electronic Format – NDAs” (1999). Under this guidance requirements were minimal:

- The clinical and analysis databases were usually prepared by merging a handful of identifiers onto each dataset
- Datasets were made SAS transport version 5 compatible
- Metadata requirements were minimal. At the dataset level: the name, description, structure and key variables were provided. At the variables level: variables attributes plus decodes and comments or derivations were required. Value-level metadata and controlled terminology were not even a consideration
- Define.pdf was required as documentation for both the clinical and analysis databases.

The tools and skill set needed to produce the deliverables described above were primarily SAS related. SAS (and perhaps other statistical packages) were used to generate datasets, displays, analysis, and in most cases define.pdf. Since the metadata requirements were minimal, specifications were most likely (about 75% of the time) Microsoft Word documents or to a lesser extent Excel spreadsheets, two very familiar formats.

The goals of validation in a non-CDISC environment were to ensure that datasets and displays were created according to specifications and they were programmed correctly. Validation was performed manually and by using tools developed in SAS. Validation to an external standard was rarely required.

## THE MOVE TO CDISC STANDARDS

The use of CDISC models has increased in the pharmaceutical industry and will continue to increase. The PDUFA five year plan includes the use of CDISC standards. The FDA is committed to using the CDISC SDTM and ADaM standards for data submitted to the FDA. The number of requests for CDISC submissions is increasing as many reviewers are developing a preference and familiarity with CDISC submissions. From a CRO perspective, there has been a sharp increase in the past couple of years in the number of requests for clinical data using the SDTM standard and analysis data modeled under the ADaM standard. If you are in the clinical trial business, it is now good business to adopt and integrate CDISC standards into your organization.

At this point in time integrating CDISC standards and producing CDISC deliverables has primarily fallen on statisticians and statistical programmers. These two groups are largely responsible for mapping clinical data to SDTM, programming and validating SDTM datasets, creating define.xml file as documentation for the SDTM database, creating and validating ADaM datasets, producing the define.pdf file that documents the ADaM database, and populating the extensive amount of metadata required by CDISC standards.

In the sections below we examine how the statistical computing environment must evolve to successfully incorporate CDISC standards into an organization and to produce high quality CDISC deliverables on time.

## PROJECT OPERATIONS

Introducing CDISC models, especially SDTM, into the Statistical Computing environment has a significant effect on work streams, work flow, and work processes. **Figure 2** (next page) shows how workflow changes by adding SDTM and ADaM components. Adding SDTM to the work flow means adding an entirely new work stream. The flow of work is no longer from the Data Management System (DMS) to analysis datasets. It is now from the DMS to SDTM to analysis datasets. This affects timelines, budget, and resources. In general more of everything is needed.

### TIMELINES

While there is now an increased workload and more deliverables to produce with SDTM, the metric for number of days from database lock to top line results and final displays has not changed. That being the case, more resources and coordination is necessary to meet project timelines.

### NEW TYPES OF WORK

Producing SDTM and ADaM deliverables adds new tasks to the pre-CDISC landscape. To create SDTM datasets and its required documentation:

- Specifications must be written to map data management data streams to SDTM domains
- Other SDTM required metadata at the dataset, variable, and value level must be populated
- SDTM datasets have be programmed and validated
- Trial design datasets have to be created and validated
- Define.xml must be generated
- The CRF has to be re-annotated for SDTM
- SDTM datasets and Define.xml must be validated to the SDTM standard.

Creating analysis datasets using the ADaM standard model also introduces new work streams, although not as voluminous as SDTM:

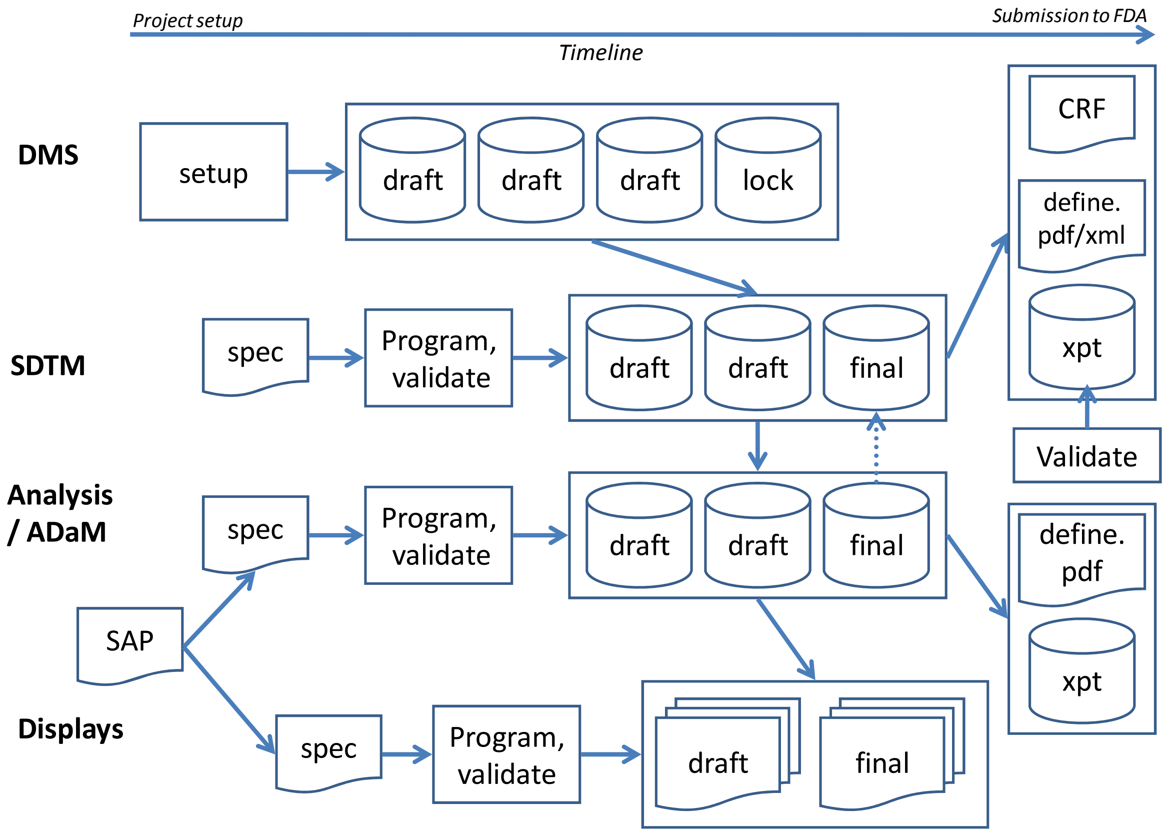
- ADaM datasets must be programmed and validated using SDTM as the data source;

- The metadata requirements for ADaM add a significant amount of work to the pre-CDISC requirements. As with SDTM, metadata must be specified at the dataset, variable, and value level;
- Results-level metadata has to be populated, which documents displays and analyses
- ADaM datasets must be validated to the ADaM standard.

## RESOURCES

While incorporating SDTM and ADaM adds new work streams and a significant amount of work, the timeline for producing analysis datasets and displays remains unchanged. As a result more resources are needed to produce CDISC deliverables. Staff must be added and then trained to write SDTM specifications, design and create trial design datasets, populate the metadata required by SDTM and ADaM, program and validate SDTM datasets, re-annotate CRFs for SDTM, produce the required define files, and validate SDTM and ADaM deliverables to their respective standard.

**Figure 2: Clinical Trial Workflow with CDISC**



## NEW SKILL SETS

In additional to requiring more personnel, CDISC also introduces a need to broaden the skill set beyond what was called for in the traditional statistical computing environment. It's just not SAS (or SAS and other statistical packages) anymore. Programmers and statisticians working on CDISC projects must develop expertise in SDTM, ADaM, and ISO standards. Creating the documentation file – define.XML – so that it works reliably and predictably with FireFox and Internet Explorer requires a knowledge not only of XML but also of XML Schema and, most bizarre of all, XSL, the tool by which the raw XML is transformed into the HTML-like screen displaying the metadata. Making the viewing experience effective and user-friendly also requires a working knowledge of HTML and JavaScript. Developing a robust metadata system to facilitate producing CDISC deliverables may require familiarity with Oracle, Microsoft Access, and version control software.

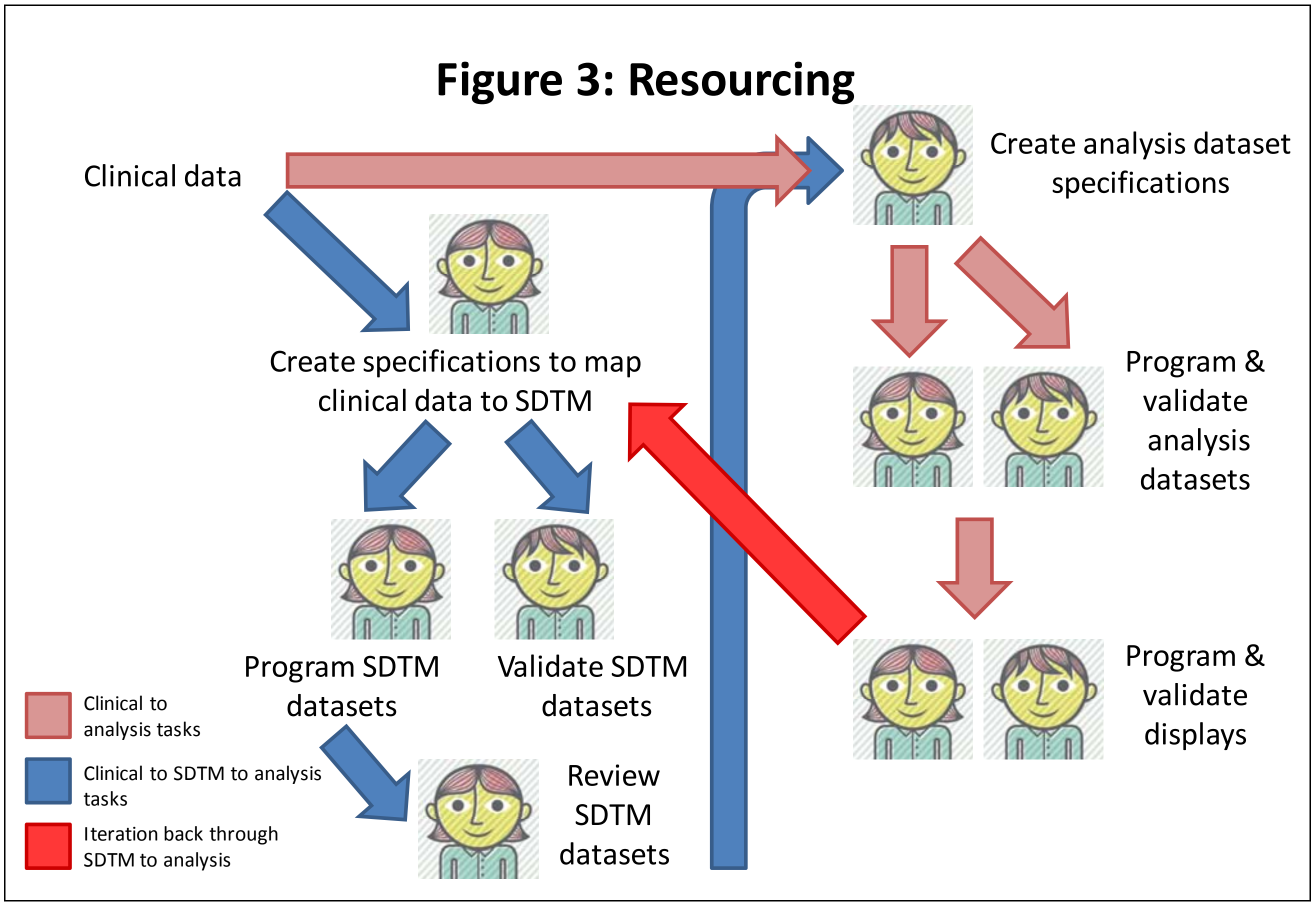


Figure 3: Resourcing
Clinical data
Create analysis dataset specifications
Create specifications to map clinical data to SDTM
Program & validate analysis datasets
Program SDTM datasets
Validate SDTM datasets
Program & validate displays
Review SDTM datasets
Clinical to analysis tasks
Clinical to SDTM to analysis tasks
Iteration back through SDTM to analysis

## PLANNING AND COORDINATION

**Figure 2** shows that there are more components to projects that incorporate the CDISC SDTM and ADaM models. One must plan for more of everything except time: more resources, project parts, and deliverables. The need increases for extensive up front planning between the SDTM team, data management, statistical programmers, and statisticians. **Figure 3** displays the workflow among these four groups. It shows that there are more handoffs and moving work streams to coordinate than in non-CDISC projects. If we follow the workflow and timeline displayed in **Figure 2**, timing for the following handoffs must be meticulously planned:

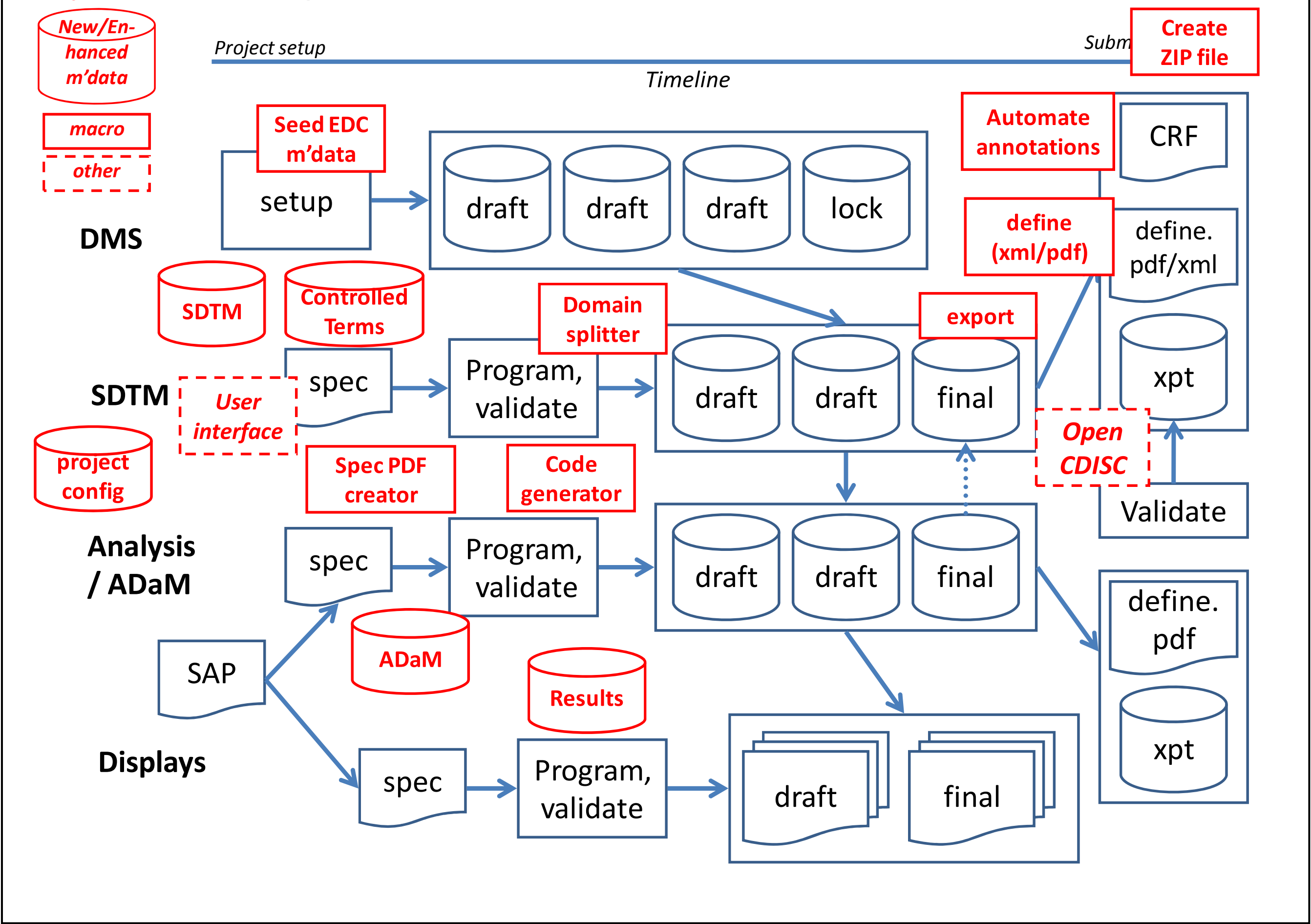
- The SDTM team must work with data management during the set up stage to begin the process of mapping the clinical data to SDTM domains
- While the clinical data is being collected, SDTM datasets must be programmed using interim clinical data as input
- Later in the data collection process, the SDTM team must work closely with the analysis team so that programming can begin for ADaM datasets and displays.
- Over time, structural changes to the clinical data or unexpected data may result in changes to SDTM specifications and datasets. These changes must be communicated effectively from data management to the SDTM team and in turn to the analysis team
- Validating the SDTM database to the SDTM standard will typically result in further modifications to SDTM datasets which will in turn affect the analysis database.

In summary, to produce high quality CDISC deliverables and meet project timelines, extensive coordination and communication among project teams is critical to success. In the following section, we outline the changes to the technological infrastructure, validation process, and corporate training that are necessary to successfully transition to be in the CDISC business.

## TECHNOLOGY

To satisfy the content and quality control requirements of the new standards, we had to enhance our use of existing technologies and become familiar with entirely new tools. This section describes changes to the metadata, new deliverables and metadata-related tools, and the introduction of the all-important XML technology. **Figure 4,** below, highlights the impact of the standards' implementation on metadata, macros, and related tools. Its intent is not to be a comprehensive display of all changes, but to give the reader a feel for the range of tools and processes affected *and* to illustrate that the changes are distributed throughout the project life cycle.

Figure 4: CDISC Tool, Metadata Impact



### METADATA AND INTERFACE

If an organization were 'metadata-less' before introduction of the CDISC standards, it would be almost impossible to stay that way and produce timely and compliant deliverables once the standards were in effect. Datasets now have to be created to conform not just to *internal* standards but to *external* ones (SDTM, ADaM) as well. The only feasible, practical way to describe dataset attributes and data values is via a collection of tables at the dataset, variable, and value levels. These tables describing the standards are metadata, and must be integrated into existing processes.

**Prepopulated Tables.** The most significant integration impact of the standards is, arguably, on the user interface that is used for maintaining the metadata. In the pre-standards era, the interface was simply a mechanism for the entry of information about variable attributes and derivation. The user started with the conceptual equivalent of a blank page and entered metadata in several tables.

SDTM is different, because with the exception of SUPPQUALed values and custom Domains, a high proportion of the metadata is predetermined. Domains have specific variable names and labels, and the order in which they appear in the dataset is fixed. The metadata database has to be seeded with the standard, so that when the user opens the interface for the first time, he/she is presented not with a screen that says "I'm ready to be populated," but instead says "I'm prepopulated – add where you can and mark the variables you won't be submitting."

**Controlled Terminology.** The interface also has to deal with Controlled Terminology. In a pre-standards environment, a subject's gender could be represented as "M" or "F" in one study and "Male", "Female", "Unknown" in another. (As an aside: this is not due to lack of internal standards, but rather the CRO perspective, where we serve multiple masters with multiple ways of expressing even the most basic concepts). With the CDISC standards comes Controlled Terminology, which establishes approved value lists for variables and says whether it is acceptable to add values to the list. The metadata user interface must not only make these terminology lists accessible, but must also allow or prohibit modifications.

**Model Change.** The interface software must also be flexible. Model implementation definitions change (think SDTM IG 3.1.1 and 3.1.2). A CRO has to be responsive to the needs of clients wishing to use older or newer standards. We cannot simply switch from one to the other, but must maintain both.

**"Upstream" Validation.** While not essential, interface-level validation of metadata as it is being entered is helpful, and prevents propagation of errors by tools that use the metadata. In addition to checks already made by the pre-standards interface (valid variable names, appropriate length for a given data type, etc.), new checks should be added. Did the user mark a Required or Expected variable as not part of the Domain? Did they not use the expected Controlled Terminology list for a variable? These and other tests are not hard to code, but do add to the cost of having the interface meet the greater set of [needs] imposed by the standard.

## NEW TECHNOLOGY: XML

If the standards themselves are viewed as a new *language* for conveying data, XML can be seen as the language's *grammar*. Although both data and its description (the "define" file) may be delivered as XML files, the prevailing current industry practice is delivery of data as SAS transport (.XPT) files and the define file as XML. Thus we focus here on discussion of define.xml.

Storing dataset and results-level metadata in the proscribed XML structure requires knowledge of XML basics. A solid metadata architecture greatly assists the task of creating define.xml. Even with the benefit of robust metadata, however, one quickly realizes that just as in real life, you don't marry the spouse, you marry the family. The "family" in the XML case is a collection of technologies that assist in describing and displaying the XML file. Family members include:

- **XSLT.** E**x**tensible **S**tylesheet **L**anguage **T**ransformations, used to transform XML into other formats, such as XHTML. It is a functional language in the tradition of Lisp, and so typically administers a double dose of bewilderment to the programmer, who has to not only master its arcane syntax but understand the mindset, limitations, and power of functional languages.
- **Xpath.** This is a sublanguage used by XSLT to navigate to nodes in the XML document.
- **Xschema.** Describes the structure of an XML document. While any web browser or other application that can read XML can determine whether the file is syntactically correct, Xschema goes a step further and is the basis for determining whether the file's contents are semantically correct. That is, Xschema helps answer the question "does the XML file have values that are valid?"

Note that it is possible to create define.xml and use a generic XSLT file for its transformation and display. However, the importance of a better-than-average comfort level with these technologies will only increase over time, as the FDA accepts both data and data descriptions in XML format.

An XML document, even one containing a generous amount of white space and indenting, is virtually impossible to read, a jumble of tags and text. Fortunately, there are many editors and viewers that not only color-code the file's contents, making it easy to distinguish tags and text, but also make clear the document's inherent tree-like structure. Delivering CDISC deliverables in XML thus requires some familiarity with the use of these tools. Notable among the tools are two free products: XMLpad and SAS's XML Mapper. The latter is especially useful, since it can read a (usually) hierarchical XML file and generate a "map" file that can be to represent the XML in the familiar rectangular form of a SAS dataset.

## NEW TOOLS

New standards, expressed via metadata, and a new way to document the deliverables required the modification of existing tools and development of entirely new ones. This section briefly describes some of the tools. The intent is not to describe the technical aspects of the tool, but rather to give the reader an idea of the range of tools and the necessity to invest time to develop them.

**Programming Spec.** The metadata is stored in a database and is attractively displayed via the user interface. As a practical matter, though, programmers who have to create the datasets want the metadata formatted differently, and want it in hard copy.  The spec-printing macro produces a PDF that lists only variables marked for submission, lays out value-level metadata specs neatly, and is highly configurable.

**Code Generator.**  This tool reads the metadata and constructs ATTRIB and KEEP statements for a dataset.  This relieves the programmer of manually transcribing or using cut and paste to create variable labels, set variable type and length, and so on. The elimination of this manual process ensures that attributes in the metadata are used exactly as specified in the program creating the dataset.

**Domain Generator.**  The SDTM model's separation of variables into *domain* and SUPP*domain*, while logically clean, can be difficult and cumbersome to code and validate.  The domain generator macro reads a single source dataset containing all variables (*domain* and SUPP*domain* alike).  It determines which variables belong in *domain* and which are intended for SUPP*domain*, and creates the two domains with their required variable attributes.

**CRF Annotations.**  In the ideal world, an EDC system would capture data as SDTM Domains.  Even if this were possible *part* of the time, the CRO business model requires the flexibility to read *any* type of data (CDISC-compliant or not) supplied by a client.  Annotated CRFs (aCRFs) will have the original data stream-variable name and will have to be reannotated to conform to the SDTM naming.  Our pre-CDISC metadata database already contained CRF page names and numbers.  These were used to create hyperlinks from the define file to a specific page in the aCRF.  We automated the bulk of the aCRF annotation process by creating a tool to create PDF annotation (XFDF) files.  (See Escobar, *et al.* in References, below, for one way to approach this metadata-driven task).

**PDF/PostScript Handling.**  Not specifically related to new standards, but relevant: PDF requirements for eSub software require setting link properties, linking by page number rather than Named Destinations. This requires use of PDF add-ins and/or modification of PostScript files

**define.XML Generator.**  Initially, define.xml was delivered with an XSLT file to transform it into HTML (and thus be readable in a standard Web browser).  The format of define.xml's transformed output was considerably less restricted than that of its earlier counterpart, define.pdf.  To take advantage of the less-structured requirements of displaying the XML meant learning not only XSLT but, inevitably, adding finishing touches that required an entry-level knowledge of HTML, JavaScript, and CSS.

**define.xml Rendered as PDF.**  The end result, while attractive (and satisfying to develop) had several limitations: there were implementation issues in Internet Explorer, Firefox, and Opera; navigation using the Back button was unreliable.  One approach was to modify the XSL, liberally inserting links to locations in the transformed HTML.  This was a satisfactory solution, but nothing overcame the biggest drawback of the HTML-like display of the define file – an inability to create a reasonably formatted printed document.  To do this, we created defineXML.pdf, using a program to read define.xml (using SAS XML Mapper), and format the Domain, Variable and Value-level tables.  Delivery of defineXML.pdf is now standard for most of our clients.  (See Lex Jansen's paper, in References, below, for a detailed description of the process).

**Deliverables ZIPper.**  Prior to delivery of the define file as XML, creation of the deliverables "package" for a client was straightforward: create a ZIP file of define.pdf, supplemental documentation (also in PDF format), and XPT files in the submission directory.  Packaging define.XML requires the file itself, its transform file (define.xslt), and various JavaScript and CSS files.  To ensure that only the relevant files are delivered, we automate creation of a ZIP file.  The tool to create the ZIP file also creates readMe.txt, which contains a ZIP file manifest and notes about known issues with using the XML (the Back button issue, restriction to Internet Explorer, and the like).

## VALIDATION

Introduction of SDTM and ADaM also imposes another layer of complexity when answering the question "is it right?" Previously, validation was a matter of comparing datasets written to an internal specification.  Now, the datasets must be created with consideration of an external specification's (SDTM, ADaM) expectation of variable name, data type, variable label, and in some cases, actual data values.  Additionally, the document describing the data (define.XML) is itself data, and must be validated.

For example, a pre-SDTM Adverse Events dataset might have contained an event severity variable (e.g., SEV), character, length 4, with values "mild", "mod", and "sev".  SDTM requires values for AESEV, length 8, be "Mild", "Moderate", and "Severe".  Mapping numeric 1 to "mild", 2 to "mod", and so on might correctly convey the source dataset's contents, but the SDTM standard adds the additional consideration of using a predefined variable name and using specific values.  Not only the predefined variable attributes need to be considered, but the presence and order of variables is set by the standard as well.

Another change from the pre-CDISC era is the need for careful construction of the description of deliverables.  Define.pdf, whose contents were only loosely restricted by column order and content, has been supplanted by define.xml.  A schema controls the node order, naming, and contents of the XML. The file can be generated from metadata, but will not be considered correct if it omits or misspecifies any elements. This reinforces the need for early (interface-level) catching of metadata entry errors mentioned earlier. Clearly, the scope and complexity of the validation task has expanded.

This potentially onerous task is greatly simplified by the availability of OpenCDISC (www.OpenCDISC.org). As its name implies, it is an open source product that in the most current version (1.2, released in early 2011) implements a set of *nnn* SDTM attribute and data checks and *nnn* ADaM checks. It will also validate a define.xml file written for SDTM IG Versions 3.1.1 or 3.1.2. Output is presented in several file formats, the most popular and information-rich being a Microsoft Excel spreadsheet.

The validation task cannot be completely turned over to OpenCDISC or similar products.

- A define.xml file can be well-formed and valid, but inconsistencies in browser XSLT implementation can result in incorrect or unacceptable transformed XHTML. The file must be opened in a browser, its presentation assessed, and its hyperlinks tested.
- Unlike define.pdf, which was simply a single file easily displayed and printed by Adobe Acrobat or Reader, define.xml isn't easily viewable without an XSLT file and, frequently, various support (CSS, Javascript) files. This imposes an additional QC check: ensuring all related support files are included in the deliverables package.
- The deliverables package may, at the client's request, also need to include the OpenCDISC report. If the report is run against the final version of a dataset and there are warning-level messages, a separate document explaining why the warnings are not cause for concern may need to be prepared.

This last point highlights a general comment about OpenCDISC (or *any* similar products). The checks are generic, and so are free of any study-specific context. If the client understands that certain controlled terminology checks are not appropriate for a study, checks for that terminology can be turned off. Fortunately, OpenCDISC enables usage of non-standard, customized validation configuration files. These files are written in XML, underscoring once again the need for a comfort level with this technology.

## TRAINING

In order to successfully implement CDISC standards, a corporate wide training program should be implemented. Training cannot be limited to just the groups that produce CDISC deliverables. CDISC standards have an impact on a broad spectrum of activities within a clinical trial. Furthermore, implementing CDSIC standards affects the work flow between many of the departments or teams that work together on a clinical trials project. Given the diversity of staff that requires CDISC training, a wide variety of CDISC training modules must be developed, each tailored to a unique audience. A corporate CDISC training program can be split into non-technical and technical training.

### NON-TECHNICAL TRAINING

Non-technical training is valuable for employees in almost all functional areas. It is useful for data managers, statistical programmers, statisticians, clinical staff, regulatory staff, project managers, business development staff, contract managers, and senior management. The goals for this type of training are to introduce staff to CDISC in general, delineate how CDISC models fit into the life cycle of a drug development program, show how CDISC standards fit into a regulatory submission strategy, demonstrate how work flow and resourcing are affecting, and to demonstrate the benefits to integrating CDISC models into work processes.

Topics that may be covered include:

- What is CDISC all about
- How do current regulations and guidances apply to CDISC
- CDISC and the eCTD
- Overview of each of the CDISC standards
- Where do each of the standards fit in the life cycle of a drug development process
- What are the CDISC deliverables
- How CDISC affects your work

### TECHNICAL TRAINING

This is the training for people actually doing the work. This typically includes the SDTM team, data management, statistical programming, and statistics. The goals here are to develop new skill sets, expose staff to new software and tools, and to develop a new work flow paradigm. At a minimum, we recommend the following training modules:

**SDTM.** This should be a detailed and thorough review of the SDTM model (all of the material in the CDISC SDTM guides) and also include an interactive exercise on mapping clinical data to SDTM, annotating a CRF for SDTM, the SDTM metadata model, controlled terminology, new validation requirements, and instruction of define.xml. We recommend this module for the SDTM team, data management, and statistical programmers if they are doing the programming or validation for SDTM.

We also recommend developing a dedicated SDTM team that maps clinical data to SDTM, populates additional metadata that is required, programs and validates SDTM datasets, produces define.xml, and validates CDISC deliverables.

**SDTM Lite.** Assuming statisticians are not creating SDTM datasets, this is primarily training for statisticians. This module presents the concepts of SDTM in less detail. The goal here is to present statisticians with enough information to effectively use SDTM data as input for ADaM datasets.

**ADaM.** This should be a detailed and thorough presentation of the ADaM model and also contain information on the ADaM metadata model (for both data and displays) requirements, how SDTM and ADaM are inter-related, controlled terminology, validation requirements, and the define file. This training is primarily for statisticians and statistical programmers (staff creating and validating analysis datasets).

**ODM/XML.** Creating the define.XML file, a key and required SDTM component, so that it works reliably and predictably with Internet Explorer requires a knowledge not only of XML but also of the ODM XML schema and of XSL (and other XML related technologies). This is likely a skill set that SAS programmers do not have. That being the case, we recommend that at least two members of the SDTM team undergo ODM/XML training. These two members will be responsible for creating a process to produce define.xml by developing a proprietary tool (easily done in SAS) or using third party software, as well as teaching the rest of the team how to produce and check define.xml.

**CDASH.** While CDASH is a standard for data collection and not a core function for statistical computing, some training and expose to CDASH is beneficial. Staff creating SDTM datasets may be using clinical data as input that follows the CDASH standard. Also, the SDTM team should develop standard specifications and programs to map clinical data that uses the CDASH standard to SDTM.

## CONCLUSION

The implementation of standards for the collection, compilation, and distribution of clinical trials data has quickly evolved into an essential part of the business model of any organization dealing with pharmaceutical companies and the FDA. This paper has identified the numerous project management, technological, and training requirements necessary for the effective use of CDISC's SDTM and ADaM models. It is imperative to realize that these data models have an impact throughout the life cycle of a project and are not isolated to data collection or analysis dataset creation. ADaM and, especially, SDTM introduce changes in processes that, while ultimately prove to be beneficial to an organization, may require costly and time-consuming allocation of resources for training and tool-building.

Be it with reluctance or enthusiasm, organizations that want to remain competitive have to embrace these and future standards. Organizations will not only have to adapt to integrate the current standards, but will have to continuously evolve to keep up with evolving standards. As CDASH and the Protocol Design models mature, much of the SDTM mapping work may be able to be done more efficiently by integrating it into data management systems.

There should also be recognition that even though the standards take some of the decision making from the statisticians and programmers, the need for thorough analysis of mappings, data definitions, and the like will always be present. Given the importance of good science, complete automation should not be a goal that is valued. Ultimately the combination of infrastructure, technology, management, plus having well-trained, well-equipped workers will be the difference between a merely compliant set of deliverables and one that clearly reflects the scientific process that is the heart of what we do.

## REFERENCES

FDA, 1999, "Providing Regulatory Submissions in Electronic Format – NDAs", available on FDA website at http://www.fda.gov/downloads/Drugs/DevelopmentApprovalProcess/FormsSubmissionRequirements/ElectronicSubmissions/UCM163187.pdf

www.w3schools.com is an excellent introduction to technologies such as XML and XSLT.

A very well-written overview of XSLT is Michael Kay's "What kind of language is XSLT?" http://www.ibm.com/developerworks/library/x-xslt

*The following papers can be found at www.LexJansen.com*

Creating a PDF from define.xml: Lex Jansen “Using the SAS XML® Mapper and ODS PDF to create a PDF representation of the define.xml (that can be printed)”

Automating CRF annotation: David Escobar, *et al.* “Case Report Form Auto-Annotation in PDF Files with a Metadata Database and SAS ODS XML Tagsets in a Data Management Unit”

Previous papers by the authors, focusing on metadata and tool development:

- Abolafia, Jeff, Susan Boyer, and Ben Vaughn,” Navigating the roadblocks: Constructing an analysis database from SDTM”, 2009. Paper presented at the CDISC North America Interchange.
- Abolafia, Jeff and Frank DiIorio,“Managing The Change And Growth Of A Metadata-Based System”, 2008. Proceedings of the SASGlobal Forum.
- Abolafia, Jeff, “What Would I Do Without PROC SQL And The Macro Language,” 2005. Proceedings of the Thirtieth Annual SAS® Users Group International Conference.
- DiIorio, Frank, “Controlling Macro Output or, ‘What Happens in the Macro, Stays in the Macro’,” 2006. Proceedings of the Fourteenth Annual SouthEast SAS® Users Group Conference.
- DiIorio, Frank, “Rules for Tools: The SAS Utility Primer,” 2005. Proceedings of the Thirtieth Annual SAS® Users Group International Conference.
- DiIorio, Frank and Jeff Abolafia, 'From CRF Data to Define.XML: Going “End to End” with Metadata', 2007. Proceedings of the Pharmaceutical SAS Users Group Conference.
- DiIorio, Frank and Jeff Abolafia, “The Design and Use of Metadata: Part Fine Art, Part Black Art,” 2006. Proceedings of the Thirty-first Annual SAS® Users Group International Conference.
- DiIorio, Frank and Jeff Abolafia, “Dictionary Tables and Views: Essential Tools for Serious Applications,” 2004. Proceedings of the Twenty-ninth Annual SAS® Users Group International Conference.

Also see http://www.CodeCraftersInc.com for other papers and resources not directly related to the current paper.

## CONTACT INFORMATION

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

Jeffrey Abolafia
Rho, Inc.
Jeff_abolafia@rhoworld.com

Frank DiIorio
CodeCrafters, Inc.
Frank@CodeCraftersInc.com