Use of Metadata to Automate Data Flow and Reporting

Gregory Steffens

Novartis

PhUSE 13 June 2012

NOVARTIS

Stages of Metadata Evolution I

- In the beginning ... No corporate or industry level data or reporting standards
- Data Standards defined in each company, or often in each therapeutic area, inconsistently complied with
- Data standards and study data specifications were stored in documents or unstructured excel files. Programmers re-enter information into SAS program files.
- Claims for “scientific freedom” required in data design
- Lots of reinvention, inefficiencies, inconsistent data that can’t be easily pooled, re-entry of information into documents and program files. Expensive in time and money.

NOVARTIS

Evolution of Metadata II

- Data Standards defined for the industry, most recently by CDISC
- Begin to store data standards and specifications in formats approaching metadata.
  - Starts with excel in formats that are inconsistent, not designed for programmatic access and don't have a clear distinction between data and metadata sometimes (e.g. why isn't suppqual a flag in metadata instead of a separate, physical data domain?)
  - But metadata not playing nearly as primary a role as it should. Data standards not published in standard metadata (e.g. define.xml) and software tools not yet in place to use metadata)
- No industry standard metadata used to publish standards or study specification with yet

Stages of Metadata Evolution III

- Rigorously standardized metadata design
- Implementation of corporate **meta-programming** – programs that need no modification as it is used in every study to implement database attributes defined in metadata.
- Metadata and meta-programming should be data standard neutral - no assumptions about what the data standard is – and programming language neutral and process neutral. The industry is not generally here yet. Still thinking about out to automate SDTM or ADaM or SUPPQUAL instead of thinking about true meta-programming.
- We need to evolve to the implementation of industry-level meta-programming, driven by industry-standard metadata design. We are starting to get there!

Stages of Metadata IV

- The next big thing is to standardize **map metadata** that defines the relationships between a source metadatabase and a target metadatabase. A standardized representation of data **flow**. Map metadata should be separate tables from metadata, to allow for mapping from any source to any target and multiple targets.
- Create corporate meta-programming that automates data flow - a **Data Transformation Engine (DTE)**
- Implement an industry DTE – with meta-programming driven by metadata and map metadata – that is shared by industry, CROs and regulatory agencies.

Stages of Metadata Evolution V

- The next phase of metadata evolution is not strictly metadata, but is Study Information Data (SID), that is a standard structure to store study design, treatment arms, visit definitions, schedule of events, TFL design, etc. We need to continue our journey out of the world of documents and into the world of metadata.
- SID will enable meta-programming for the generation of standard tables, figures and listings as well as analysis results metadata that enables navigation through TFLs like the define file enables navigation through the data sets.
- SID is starting with trial design standards in CDISC and in companies (e.g. Jeff's presentation about Rho). But there is a mix of SID in data domains, ODM and metadata.
- Documents, like the protocol and SAP, will be generated from metadata in this phase of evolution.

Metadata Constituents

- A standard list of database attributes to include in any description of a database or of a data standard
- Put in a standard set of data structures that can be read by programming code
- The attributes must be highly structured in order to be usable by program code
- To define a standard for defining data standards and study data specifications
- Enables easy publication in different formats, html, word, pdf, xml, etc. Generate documents from metadata, not metadata from documents!

Standard Database Attributes

- Data Set Level
  - Short/long names, data set location, order in define
- Variable level
  - Short/long name, type, length label, primary key flag, format, value list name, suppqual flag, code/decode relationship, order, aCRF location, etc.
- Valid values
  - Value list name, start/end value, short decode, long decode, rank
- Descriptions
  - Source name, derivation description
- Row-level attributes
  - Identical to variable level attributes but for subsets of rows defined by a parameter variable value. Defines “virtual variables”, variables whose attributes change in different type of rows in the table.

NOVARTIS

Row-Level Metadata

- Necessary to fully describe tall-thin data set structures

USUBJID	SYSBP	BPSYSLOC	BPSYSU	HEIGHT	HEIGHTU	WEIGHT	WEIGHTU	BMI	BMIU
1	120	STANDING	mm Mg	185	CM	90	KG	26.3	Kg/m**2

USUBJID	VSTESTCD	VSLOC	VSORRES	VSORRESU
1	SYSBP	STANDING	120	Mm Mg
1	HEIGHT		185	CM
1	WEIGHT		90	KG
1	BMI		26.3	Kg/M**2

Metadata Structure

- Structured content to enable programmatic access to the list of attributes
- Storage structure is separate from publication structure – maximize programmatic access in the metadata design and user friendly access by people in metadata publication formats
- Storage structure is also separate from the data entry format
- Maximize sharing of information within the metadata, e.g. values lists and descriptions. Normalize the metadata design.
- There are a lot of errors and inefficiencies out there yet, in the design and implementation of metadata

Some Principles of Metadata Design

- Rigorously standardized for all database and standard descriptions, no metadata design change is required for different database standard types or study specification!
- Metadata should not impose a process or a data flow, like SDTM to ADaM to IDB. Process and flow belong in map.
- Maximize structured information and programmatic access, e.g. primary keys flagged instead of listed
- Enter once; use many. e.g. descriptions and values; meta-programming
- Complex derivation logic in descriptions and subroutines, though. Data transformation automation is implemented differently than data derivation automation.

NOVARTIS

Objectives of Metadata

- It is critical to explicitly define the objectives. Many disagreements arise from an unstated difference in assumed objectives
- Objectives allow evaluation of the success of the metadata design; e.g. retrospective description for esubmission or prescriptive enabler of automation
- Data standards and metadata are a means to an end and that end includes an efficient and transparent data flow that leads to good decisions about safety and efficacy

Objectives …

- Prescriptive metadata drives meta-programming, no more merely description, post-facto metadata
- Meta-programming must be able to assume a standard **metadata** structure in order to minimize its assumptions about **data** structures.
- Don't automate each domain, automate all domains and all standards with a single set of macros that read metadata that tells them what to do. The DTE meta-programming.
- Include enough attributes to enable the automation of every transform
- Store data **standards**, standards **templates** and study **specifications** in the same metadata design

Industry Metadata Standard

- We need an industry metadata standard to exchange information about data standards, data specifications and the way one database is created from another (e.g. ADaM from SDTM)
- Current practice is to use metadata that is quasi-standardized at each company or to use old-fashioned word documents
  - This causes great inefficiencies
    - Translating between metadata standard structures and attribute lists causes large amounts of unnecessary work

Some of the Problems Could be Solved by an Industry Standard Metadata

- Excel often used, with un-typed columns, not 2-dimensional and confusion between storage, entry and presentation structures
- Inconsistent metadata structures even within a company, between different standards, specifications and versions of the same standard
- Unstructured information like …
  - controlled terminology concatenated in large character variables
  - Primary key variables in lists instead of flags
- Inconsistent attribute lists, metadata structure
- CDISC excel workbooks have these problems too
- Including mapping information in metadata
- Assumptions about process, data flow and data standards

What could be …

- An industry metadata standard does exist – the define.xml. This has a standard list of attributes and a standard structure
- But the standard structure is xml and difficult to access programmatically
- A solution is a standard relational metadata structure that contains the list of attributes in the define.xml schema but in a programmatically accessible format.
  - This approach was used in the two CDISC pilot projects with success, using my relational metadata design and some meta-programs.
- All data standards and specifications would be stored and publicized in this standard metadata structure
- Standard GUI for entry and modification of metadata content
- A set of standard presentations of metadata content

What to do with Standard Metadata

- Data standards published in a standard way
- Study data specifications exchanged between organizations and software systems using the same metadata design
- Automation that uses metadata to inform the code about the database, instead of the code making assumptions about the database. Metadata **is** code.
- A metadata standard is more important that data standards!

NOVARTIS

A Process

- Submit data standards in an industry standard metadata structure.
- Create a study data specification by subsetting the metadata-resident data standard
- Compare the study specification to an IDB standard so that integrating the study data will be easier. Using multiple CROs for different studies is less of a problem.
- Create the define.xml / pdf / html / rtf from metadata in minutes, including all the hyperlinks to data and aCRFs
- Send the source data and specification to the programming team
- The team uses meta-programs to build and validate the database
- Validation of the data by automated comparison of the data to the metadata-resident specification

Principles of the Process

- Metadata is ***prescriptive*** rather than merely descriptive
  - Prescriptive metadata created at the start has much more value than descriptive metadata created at the end
- Metadata is populated at the start of the project and supports automation throughout the process from creation to FDA submission
  - Publish the plan
  - Check compliance to standard
  - Build the database
  - Validate the data
  - Create define file for the FDA
  - Metrics – measure compliance of requirements to standard and the data to requirements
- Enter once; use many!
- Metadata structure is identical in all applications to support sharing of content

Other Kinds of Metadata

- After metadata comes map metadata that supports even more complex automation of the transformation of data from source to target structures, like creating SDTM, ADaM or integrated databases to support ISS/ISE
- A Data Transformation Engine requires metadata and map metadata and provides huge efficiency gains and transparency in the data flow (transforms not “hidden” in code or documents)
- The term “metadata” is often used more broadly to also mean data that describes trial design, treatment arms, tables, figures and listings, titles/footnotes, etc. A more general term is “data driven applications”, which include metadata driven applications.

Map Metadata

- Map metadata must be standardized
- Map metadata “connects” an observation in the source metadata with an observation in the target metadata.
- It’s structure is simple – one map metadata set for each metadata set. It contains the primary key variables of the metadata sets for the source and the target. A columns metadata set is keyed by TABLE and COLUMN, so the map metadata structure contains SOURCE_TABLE SOURCE_COLUMN, TARGET_TABLE and TARGET_COLUMN. This is enough to support meta-programming of the flow of data from one structure to another. Map describes no DB attributes.

Meta-programming to implement data flow

- %dtmap(
  - source_mdlib=m,source_prefix=raw_,
  - target_mdlib=m,target_prefix=target_,
  - maplib=m,
  - inlib=raw,
  - outlib=sdtm,
  - suppqual_make=yes)

Study Information Data (SID)

- Standard, structured data sets that describe information required for TFL generation and the creation of some of the protocol and SAP sections.
- Visits, epochs, schedule of events, baseline visits
- Treatment arms, treatments, schedule of treatments
- TFL titles and footnotes – meta-programming creates all the titles and footnotes and analysis results metadata can be automatically created, just like the define file.
- TFL summary statistics for each TFL and a style sheet functionality to create the TFLs from that.

Examples of Macros that Implement Meta-programming

List of some of the macros and their functionality which help to achieve efficiency and ensure good quality:

Mdprint/md2odm	Publish in html or xml format
Mdatribs	Apply attributes defined in metadata to a data library
Ut_find_decodes	Finds decode variables and their attributes
Dt_make_decodes	Creates decode variables
Dt_copy_headers	Copies header variables from source to target data sets
Mdcompare / mdcompare_print	Compares metadatabases to each other, such as a study requirement to a standard or a study to a study
mdcheck	Checks data and reports discrepancies with the metadata
mdbuild	Builds metadata to describe an existing data library
mdfreqvals	Creates the values metadata set (supplements mdbuild)

Examples of Meta-Programming

dtmap	Top level macro that users call to transform data from one format to another, e.g. raw to SDTM to ADaM to IDB
Dt_thin2wide	Convert tall-thin to short-wide
Dt_wide2thin	Convert short-wide to tall-thin
Tool_code_lib	Documents program code
Ut_saslogcheck	Checks SAS logs for disallowed messages
Ut_age_years	Computes age in years
Ut_truncate_long_chars	Truncates long character variable lengths to least length to hold longest value
mdport	Creates a transport file of a metadatabase to archive versions
md2excel / excel2md	Converts metadata between SAS and excel
mdmkdsn	Creates 0-observation data sets as defined in metadata

Examples of Meta-Programming

Suppqual_make	Creates the suppqual data sets, by reading the suppqual flag in the metadata to identify supplementary qualifiers
Suppqual_get	Gets supplementary qualifier variables from the suppqual data sets and adds them to their proper domain
Dtmap_values	Changes the value of variables by reading value map metadata
Mdformats	Create user formats from values metadata set
Missvars	Report variables that have a missing value in all observations
Missobs	Report observations where all variables have a missing value