MDR Requirements for Study Build and Implications for the CDISC 360 project

Presented by Dr. Philippe Verplancke
Head of Business Development, XClinical GmbH
8 May 2019
Disclaimer and Disclosures

• The views and opinions expressed in this presentation are those of the author(s) and do not necessarily reflect the official policy or position of CDISC.

• Philippe Verplancke is the founder and Head of Business Development of XClinical, an EDC / IWRS / eCOA vendor
Agenda

1. Business value of an MDR
2. Requirements for an MDR from the perspective of study builders
3. How the CDISC 360 project can contribute to extend the CDISC library towards these MDR requirements
4. Pragmatic next steps
1. Business Value of an MDR
What is a Metadata Repository (MDR), really?

1. A (central) library (= storage) of Metadata, this means a library of definitions of variables, datasets, forms, codelists, etc., including mapping rules between variables.

2. In addition to storage, an MDR typically provides:
   - User access control (who sees/modifies which metadata? who is the librarian?)
   - Version control (incl. “automated upgrades”)
   - Workflow & governance (creation, review, approval, decommissioning, …)
   - Reporting, e.g. what is the impact of a CDISC version upgrade to my study? How many times and in which studies did we use a specific variable?, etc.
   - Collaboration, Curation, …
   - Semantic linking and searching (e.g., what different kinds of blood pressure exist?)

3. The physical storage can be a file or a set of files (XML, SAS, Excel, …), a relational database, a graph database (RDF, …), etc.
A Metadata Repository (MDR), what’s in it for me?

• An MDR creates business value through:
  • Cost reduction: e.g. avoid redundancy and increase re-use
  • Quality increase: e.g. consistency with standards and regulatory requirements
  • Time reduction: e.g. accelerate review and approval processes, accelerate study set-up, accelerate SDTM and ADaM mapping

• An MDR does not need complex or expensive technology:
  • A set of files (Excel, XML, SAS, …) could be the basis for your MDR
  • But multi-user access, consistency, version control and searching is not easy using files, which is why an MDR is often using a database system

• An MDR is more than storage, it provides you with software functions to create, modify and manage metadata
2. Requirements for an MDR from the perspective of study builders
What does a DM need from an MDR to build a study?

- Find the right CRF variables for my research topic as soon as possible
  - intelligent search functions (text, meaning (semantic search), …)
- Avoid redundancies when creating new metadata
- Easy “copy-paste” into my study database (keeping the link to where I got it from)
  - Machine-readable metadata
- Pre-mapping of my CRF variables to SDTM and ADaM
- See differences between my study specification and standard (CDISC) specifications
- See differences between study versions (amendments)
- See impact of upgrading my study specification to the newest (CDISC) standard

Therefore: a corporate MDR should not only store (CDISC) standards but also all the study-specific metadata!
An MDR needs to implement an efficient version control mechanism for all elements of metadata

- When creating a study amendment or when reviewing newer versions of CDISC standards and their impact on the study, a data manager needs to have a transparent overview of the version history of metadata.
- CDISC currently only provides version control on a collection of metadata (stored in one file), not on each element of metadata.
- Here is how it could be improved in analogy to software version control.

**Example**

<table>
<thead>
<tr>
<th>V2</th>
<th>V1</th>
</tr>
</thead>
<tbody>
<tr>
<td>AE Form</td>
<td>AESEV variable</td>
</tr>
<tr>
<td>Mild</td>
<td>Medium</td>
</tr>
</tbody>
</table>

AE Form version 2 references all the elements with orange color or an orange border.
An MDR could use hashes to ensure non-ambiguity

- Every metadata element is defined by an OID, a version number and its content
- E.g., the metadata content of an item is defined as Name + Label + Datatype + Length + Measurement Unit OID/Version + CodelistRefOID/Version (+ any other relevant attributes) and this content results in a unique hash (e.g. “7171a9”)
- E.g., the metadata content of an itemgroup is defined as Name + Label + a list of ItemOIDs/Versions that it contains and this content results in a unique hash. The same applies on all other levels of metadata.
- Each OID is unique and each hash is unique and can be stored with a unique constraint in the database (across all studies)
- Each combination of OID and version number is unique and matches only one content, i.e. one hash
- This will enable faster checks for differences between two versions of a CDISC standard
- Apart from exact duplicates, an MDR should also detect and report probable duplicates: similar OIDs, similar labels, similar content, linked to the same biomedical concept, etc.
3. How the CDISC 360 project can contribute to extend the CDISC library towards these MDR requirements
What CDISC 360 wants to achieve (my understanding)

Invent / enrich modelling language and content of the CDISC library to do a proof-of-concept for:

- Use Case 1: Specify study metadata and mapping starting with the analysis endpoints all the way back to the data collection variables that are needed to provide evidence for the endpoints
  - pull the endpoint specification out of the CDISC MDR and everything else shall be automatically pulled out as needed

- Use Case 2: Specify study metadata starting with the data collection variables and start building CRF forms, SDTM datasets, ADaM datasets, analysis metadata
  - pull variables from the MDR and start building your study while reusing existing forward mappings or storing the new mappings that you created in the MDR for future reuse

- Use Case 3: Run a data collection system, run a data tabulation system, run a data analysis system based on the machine-readable specs from above (metadata-driven software procedures)
What CDISC 360 therefore needs to create to support the study build use cases (No. 1 and No. 2)

• Value-level metadata
• Unique identifiers
• Search functions
• Mapping language
• Correct XML exports

• Above all: a prototype web interface and prototype software to create all of the above, not just Excel files, not just a JSON API, because otherwise the whole treasure would remain underground and never be used

• Note: a version control mechanism is currently out of scope for the CDISC 360 project.
CDISC 360 – why value-level metadata are needed

- Data collection variables need the correct data type and length
- Data collection variables need the correct measurement units
- Data collection variables need a correct reference to a codelist (if relevant)
- Data collection codelists need a decode for every code

- Much of these are not yet available in the CDISC library and will need to be created (for the Diabetes therapeutic area) during the CDISC 360 project

- This is not only needed for data collection (set-up of EDC systems) but also for the correct transformation of raw data to SDTM and to ADaM
CDISC 360 – why unique identifiers are needed

• When referencing data collection variables and codelists in a mapping instruction to create SDTM or when referencing SDTM variables in a mapping instruction to create ADaM, we need an unambiguous reference to each variable and codelist.

• Otherwise, there is a risk that variations of the same variable are referenced erroneously because the data manager does not recognize he is picking the wrong variable definition.

• Currently there is no generally accepted and published unique identifier concept in the CDISC library. This will need to be created (for the Diabetes therapeutic area) during the CDISC 360 project. Unique identifiers are even more important as soon as new versions of the standards are published to recognize unambiguously which elements have changed.
Reality check on today’s status of the CDISC library

The current download files from the CDISC library (in Excel, pdf, define.xml or odm.xml format)

- Do not contain unique identifiers
- Do not contain correct references between variables and codelists (non-matching identifiers between codelist_ref and codelist_def)
- Do not contain variables with correct datatypes
- Do not contain measurement units
- Do not contain decodes for codelists
- Do not contain value lists, only codelists

Recommendation: assign a dedicated team of subject matter experts as the “Library Curation Taskforce” to create content, content, content
CDISC 360 – why search functions are needed

• A metadata repository – by definition – stores a volume of metadata that cannot be memorized by a human data manager. Therefore, a data manager accessing the MDR will not know whether or not she will find the variable she needs, e.g., to measure the volume of a solid cancer.

• It seems a trivial requirement to have a search function. However, the more intelligent the search, the better. E.g. wildcard search, filtered search, similarity search (“did you mean?”), semantic search (“which variables are related to cancer and are expressed in volume units?)

• Biomedical concepts describe relationships between entities and thus make it easier to find them through semantic search. The CDISC library may also store multiple variables that all implement the same biomedical concept (synonym variables representing variations of the same concept)
CDISC 360 – why a mapping language is needed

• Computational methods (pseudocode) describing
  • how to derive an SDTM variable from a set of data collection variables or
  • how to derive an ADaM variable from a set of SDTM variables
  are part of define.xml today

• But for the automation use case (No. 3), computer systems will need unambiguous
  instructions to execute the data transformations

• Biomedical concepts are a good step forward to describe relationships (= mapping)
  between variables across different CDISC standards. However, they too will still need
  to include algorithmic descriptions.

• Therefore, the current “pseudocode text” in computational methods needs to be
  replaced by a machine-readable code
CDISC 360 – why correct XML exports are needed

• Study builders want to load the CDISC library content directly into their EDC systems.

• Excel or PDF files are not really machine-readable (too much risk of unwanted modifications).

• Therefore, a machine-readable export from the CDISC library will be needed, whether through an API or via downloads. Modern options such as JSON or RDF may be out of reach of many organizations in terms of IT capabilities.

• Recommendation: provide CDISC library exports as XML (using the define.xml and the odm.xml standard as appropriate; refer to “reality check” in a previous slide).
4. Pragmatic next steps
Pragmatic next steps

• Corporations building their own MDR should start with a database repository including
  • unique identifiers
  • a simple version control concept adapted from well-known version control mechanisms in the world of software development
  • a direct referencing mechanism (instead of making duplications/copies)
  • import and export procedures using define.xml and odm.xml

• CDISC 360 should assign dedicated staff to create content, content, content

• Before inventing a mapping language (computational algorithms), CDISC 360 should consider existing solutions (e.g. XSL and XPath)

• In an ideal world, CDISC 360 could hire software developers to do rapid prototyping, starting small with an agile development process
Questions?

Thank You!

Contact: philippe.verplancke@xclinical.com