CLINICAL STANDARDS TOOLKIT 1.5
HOW DO I KNOW MY METADATA IS RIGHT?

GENE LIGHTFOOT, SAS, CARY, NC, US
WHAT DO WE MEAN BY METADATA?

• In the context of Clinical Standards Toolkit
• Information about the Toolkit
• Information about CDISC Models (SDTM, ODM, etc.)
• Information about Customer Standards
• Information about where things are.
WHY DO WE NEED TO VALIDATE METADATA?

- Validation is an industry requirement
- Customers want and require this
- Good clinical programming practice – affects clinical data
- Want to ensure product is behaving the way SAS intended.
HOW IS IT DONE WITH PRE-CLINICAL STANDARDS TOOLKIT 1.5?

• No systematic assessment of the validity of Clinical Standards Toolkit metadata.
• Code undergoes separate SAS testing before it is shipped
• Running the cstutil_checkds framework macro assesses structure and content
• SAS provides an IQ/OQ document that customer’s can use.
  Problem: IQ/OQ is a baseline created by SAS, sometimes the customer installation is a little different, causing some mismatches.
SAS Health & Life Sciences R&D and Product Management saw importance of providing a mechanism (process) for Internal Validation of the Clinical Standards Toolkit.

Objectives:
- To ensure SAS supplied metadata files are consistent and correct
- To provide functionality to customers to help validate custom standards

3 Areas to validate:
- Global and sample library
  - Framework and sample studies
- Standard reference metadata
  - CDISC standards
- Standard source metadata
  - Study level
Range of functionality

- Support IQ/OQ assessment and reporting
- Support registration of a new standard and/or updates to an existing standard
- Assess metadata consistency across files
- Assess structural validity of a metadata file
- Assess content of a metadata file
- Validate a SASReferences data set
- Evaluate validation check metadata
- Perform a process pre-check
- Perform run-time process validation
- Evaluate code module header compliance
HOW TO DO THIS?

- Clinical Standards Toolkit is standard/model based.
- Load information about Clinical Standards Toolkit as a Standard
- Populate the Metadata files
- Create validation_master data set containing internal validation checks.
- Use the Framework validation macros to validate itself
Clinical Standards Toolkit cstGlobalLibrary and cstSampleLibrary folder structure created for Framework Standard.
• CST_FRAMEWORK is registered (standards.sas7bdat)
• Standard supports validation is Y

<table>
<thead>
<tr>
<th>Name of standard</th>
<th>Mnemonic for standard</th>
<th>Root path for the standard</th>
<th>(Sample) study root path</th>
</tr>
</thead>
<tbody>
<tr>
<td>CST-FRAMEWORK</td>
<td>CST</td>
<td>&amp;_cstGRoot./standards/cst-framework-1.5</td>
<td>&amp;_cstSRoot./cst-framework-1.5</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Is this the default version for the standard [Y/N]?</th>
<th>Is this standard part of the CST framework [Y/N]?</th>
<th>Is this a data standard [Y/N]</th>
<th>Standard supports validation [Y/N]?</th>
<th>Is this an xml-based standard [Y/N]?</th>
<th>The revision of the standard-standardversion</th>
</tr>
</thead>
<tbody>
<tr>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>N</td>
<td>1.5</td>
</tr>
</tbody>
</table>
Reference (Standard) Metadata is created dynamically for internal validation:

- reference_tables and reference_columns
- Different from other standards which are static
- More flexible for internal validation since framework metadata can change as new standards and/or customized standards are added.
Toolkit uses global macro variables extensively. Introduction of new metadata tables allows us validate these macros.

Standard macro variables data set – contains metadata about all macro variables used throughout Clinical Standards Toolkit.

<table>
<thead>
<tr>
<th>Macro variable name</th>
<th>Macro variable label</th>
<th>Macro description</th>
<th>Is macro variable required? (Y/N)</th>
<th>Is macrovalue case sensitive? (Y/N)</th>
<th>Value type (ANY, DISCRETE, INTEGER, DATASET)</th>
<th>Source file for macro variable</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 _cstCheckSortOrder</td>
<td>Specifies the order in which validation checks are to be run</td>
<td>The variable enables specification of the order in which the checks are to be run. The _DATA_value indicates that checks are to be processed in the order defined in the Validation Control data set. You can specify a set of space-delimited keys from Validation Control columns (for example, check-source checked)</td>
<td>N</td>
<td>N</td>
<td>ANY</td>
<td>validation</td>
</tr>
<tr>
<td>2 _cstColumnMetadata</td>
<td>Data set containing column-level metadata supporting validation</td>
<td>Data set that is used during processing that contains column-level metadata (derived from either the reference or study column metadata) that is used by the process.</td>
<td>Y</td>
<td>N</td>
<td>DATASET</td>
<td>validation</td>
</tr>
<tr>
<td>3 _cstDebug</td>
<td>Turns debugging on or off for the session</td>
<td>If on, then _cstDebug options are set. Many tabs remain in the Work library at process conclusion. Note that when _cstDebug=1, the size of the SAS log is significantly larger.</td>
<td>Y</td>
<td>N</td>
<td>DISCRETE</td>
<td>initialize</td>
</tr>
<tr>
<td>4 _cstDebugOptions</td>
<td>SAS session debugging options when _cstDebug=1</td>
<td>SAS system options set when _cstDebug=1.</td>
<td>N</td>
<td>N</td>
<td>ANY</td>
<td>initialize</td>
</tr>
</tbody>
</table>
Standard macro variable details data set – contains metadata about all macro variable values used throughout Clinical Standards Toolkit.

<table>
<thead>
<tr>
<th>Macro variable name</th>
<th>Macro variable value</th>
<th>Macro variable value label</th>
<th>Is this value the default? (Y/N)</th>
</tr>
</thead>
<tbody>
<tr>
<td>_cstCheckSortOrder</td>
<td><em>DATA</em></td>
<td>Use order defined in the Validation Control data set</td>
<td>Y</td>
</tr>
<tr>
<td>_cstColumnMetadata</td>
<td>work._cstcolumnmetadata</td>
<td></td>
<td></td>
</tr>
<tr>
<td>_cstDebug</td>
<td>0</td>
<td>Off</td>
<td>Y</td>
</tr>
<tr>
<td>_cstDebug</td>
<td>1</td>
<td>On</td>
<td>N</td>
</tr>
<tr>
<td>_cstDebugOptions</td>
<td>mprint mlogic symbolgen mautolocdisplay</td>
<td></td>
<td></td>
</tr>
<tr>
<td>_cstFMTLibraries</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>_cstMessageOrder</td>
<td>APPEND</td>
<td>Append records from multiple message data sets</td>
<td>Y</td>
</tr>
<tr>
<td>_cstMessageOrder</td>
<td>MERGE</td>
<td>Merge records from multiple message data sets</td>
<td>N</td>
</tr>
</tbody>
</table>
Messages data set – contains Framework Specific messages used by validation and reporting processes. Prefaced by CSTxxxx and CSTVxxx

<table>
<thead>
<tr>
<th>Result identifier</th>
<th>Standard version</th>
<th>Source of check</th>
<th>Record identifier used by checksource</th>
<th>Severity of check</th>
<th>Message text</th>
<th>Basis or explanation for result</th>
</tr>
</thead>
<tbody>
<tr>
<td>81</td>
<td>CST0201</td>
<td>*** CST</td>
<td>CST0201</td>
<td>Warning</td>
<td>&amp;_cstParm1</td>
<td></td>
</tr>
<tr>
<td>82</td>
<td>CST0202</td>
<td>*** CST</td>
<td>CST0202</td>
<td>Error</td>
<td>&amp;_cstParm1</td>
<td></td>
</tr>
<tr>
<td>83</td>
<td>CSTV001</td>
<td>*** CST</td>
<td>CSTV001</td>
<td>Warning</td>
<td>Multiple records detected for standard &amp;_cstParm1</td>
<td></td>
</tr>
<tr>
<td>84</td>
<td>CSTV002</td>
<td>*** CST</td>
<td>CSTV002</td>
<td>Error</td>
<td>Standard is defined as XML but location of import XSL is missing Example: XML standard without XSL/schema paths</td>
<td></td>
</tr>
<tr>
<td>85</td>
<td>CSTV003</td>
<td>*** CST</td>
<td>CSTV003</td>
<td>Error</td>
<td>Standard is defined as XML but location of export XSL is missing Example: XML standard without XSL/schema paths</td>
<td></td>
</tr>
</tbody>
</table>
Addition of new /Templates folder containing metadata data sets. Used to validate data set structure. Each standard has its own set of templates.
Validation_master – contains ALL checks created by SAS to validate that the installation is correct and required metadata is in place.

<table>
<thead>
<tr>
<th>Validation check identifier</th>
<th>Severity of check</th>
<th>SAS macro module name</th>
<th>Domains/data sets to which check applies</th>
<th>Columns to which check applies</th>
<th>Code logic used within code</th>
</tr>
</thead>
<tbody>
<tr>
<td>1   CSTV001</td>
<td>Warning</td>
<td>cstcheck_notunique</td>
<td>glmeta.standards</td>
<td>standard+mnemonic</td>
<td>data work._cstProblems;set</td>
</tr>
<tr>
<td>2   CSTV002</td>
<td>Error</td>
<td>cstcheck_columncompare</td>
<td>glmeta.standards</td>
<td>[isxmlstandard][importxsl]</td>
<td>&amp;_cstDSName;if (upcase(&amp;_cstColumn1) eq &quot;Y&quot; and &amp;_cstColumn2 eq &quot;&quot;)].run;</td>
</tr>
<tr>
<td>3   CSTV003</td>
<td>Error</td>
<td>cstcheck_columncompare</td>
<td>glmeta.standards</td>
<td>[isxmlstandard][exportxsl]</td>
<td>data work._cstProblems;set</td>
</tr>
<tr>
<td>4   CSTV004</td>
<td>Error</td>
<td>cstcheck_columncompare</td>
<td>glmeta.standards</td>
<td>[isxmlstandard][schema]</td>
<td>&amp;_cstDSName;if (upcase(&amp;_cstColumn1) eq &quot;Y&quot; and &amp;_cstColumn2 eq &quot;&quot;)].run;</td>
</tr>
<tr>
<td>5   CSTV005</td>
<td>Error</td>
<td>cstcheck_columncompare</td>
<td>glmeta.standards</td>
<td>[isxmlstandard][importxsl]</td>
<td>data work._cstProblems;set</td>
</tr>
<tr>
<td>6   CSTV006</td>
<td>Error</td>
<td>cstcheck_columncompare</td>
<td>glmeta.standards</td>
<td>[isxmlstandard][exportxsl]</td>
<td>&amp;_cstDSName;if (upcase(&amp;_cstColumn1) eq &quot;Y&quot; and &amp;_cstColumn2 ne &quot;)&quot;].run;</td>
</tr>
<tr>
<td>7   CSTV007</td>
<td>Error</td>
<td>cstcheck_columncompare</td>
<td>glmeta.standards</td>
<td>[isxmlstandard][schema]</td>
<td>data work._cstProblems;set</td>
</tr>
</tbody>
</table>
New metadata columns have been added to handle Internal Validation

- IOtype - (input, output, or both) determines if a file must be present and have r/w access.
- Filetype – Folder, dataset, catalog, or file
- Allowoverwrite – Can we write to the file (Y/N)
- Studylibraryrootpath – root directory for study library
- ...

New SAS macros have been written

- Cstutilcheckwriteaccess – If an output file can we write to the folder
- Cstutilfindvalidfile – Does the required file exist?
- ...

CLINICAL STANDARDS TOOLKIT 1.5 AS A STANDARD (CSTGLOBALLIBRARY)
A sample library has been created for Framework validation containing framework specific programs and data.

Driver programs
- `validate_data.sas` – run everything specified in `validation_control`
- `validate_framework` – framework specific checks.
- `validate_iqoq` – IQ/OQ specific checks.
- `validate_standard` – standard specific metadata checks
• Validation_control – Usually a copy or subset of validation_master allowing the user to select their checks.
• New to 1.5 – Validation_control VIEWS subset by checktype from validation_master:
  • FW-framework
  • STD – standard
  • STDIQOQ – standard iq/oq
INTERNAL VALIDATION WORKFLOW FOR IQ/OQ

VALIDATE_IHQQ

- Choose Validation control_F W VIEW
- Dynamically generate reference tables and columns
- Select Standards for IQ/OQ checking
- Process Setup Macro
- Framework checks run
- Macro to create job stream for each standard
- Generate Results data set

Framework IQ/OQ Standards IQ/OQ
Results are stored in the validation results data set
• Accessible by programmers
• Can be used in reports

<table>
<thead>
<tr>
<th>Result identifier</th>
<th>Source data</th>
<th>Resolved message text from message file</th>
<th>Result severity [e.g., warning, error]</th>
<th>Problem detected? (0=no, otherwise yes)</th>
<th>Process status [Nonzero, aborted]</th>
<th>Actual value observed</th>
<th>Record-level keys + values</th>
</tr>
</thead>
<tbody>
<tr>
<td>CSTV260</td>
<td>REFMETA.REFERENCE_TABLES</td>
<td>Required column cannot be null</td>
<td>Error</td>
<td>1</td>
<td>0</td>
<td>keys=</td>
<td>SASrel=REFDATA, table=ItemGroupLeafTitles</td>
</tr>
<tr>
<td>CSTV260</td>
<td>REFMETA.REFERENCE_TABLES</td>
<td>Required column cannot be null</td>
<td>Error</td>
<td>1</td>
<td>0</td>
<td>keys=</td>
<td>SASrel=REFDATA, table=MUTranslatedText</td>
</tr>
<tr>
<td>CST0100</td>
<td>REFMETA.REFERENCE_COLUMNS</td>
<td>No errors detected in REFMETA.REFERENCE_COLUMNS</td>
<td>Info</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CST0004</td>
<td>CSTCHECK_COLUMN</td>
<td>No columns evaluated - check validation control specification</td>
<td>Warning: Check not run</td>
<td>-1</td>
<td>0</td>
<td>tableScope=glcntl.s</td>
<td></td>
</tr>
</tbody>
</table>
Internal Validation by the Numbers*

- ~ 121 Internal Validation checks
- 15 new Clinical Standards Toolkit macros
- 8 new Framework metadata columns
- Number of new metadata tables
  - Framework – 2
  - Templates – 49 (currently across 9 standards)

*Numbers may vary as development was ongoing when slide was produced.
• SAS Clinical Standards Toolkit information can be found at:
  • http://support.sas.com/rnd/base/cdisc/cst/index.html

• SAS Knowledge Base/Focus Area
  • http://support.sas.com/rnd/base/cdisc/cst/index.html

• Availability of updates will generally be posted to the SAS and Clinical Trials Community, available at:
  • http://communities.sas.com/community/sas_and_clinical_trials
Please come see me in the SAS booth
Talk to your sales representative
Let SAS Technical Support know.
Contact me through e-mail
Questions?

THANK YOU

Contact information:

Gene Lightfoot
+1-919-531-3103
gene.lightfoot@sas.com
(Clinical Standards Toolkit Development)

Bill J. Gibson
+1-919-531-0996
billj.gibson@sas.com
(Product Management)