Contents

1. Disclaimer................................................................................................................................. 4
3. Overview ................................................................................................................................... 6
   3.1. Conformance Testing Support .......................................................................................... 6
   3.2. Requirements ................................................................................................................... 7
4. Updates from Previous Major Version ................................................................................... 9
   4.1. Updates to Conformance Support ..................................................................................... 9
   4.2. Updates to Features and Functionality ............................................................................ 10
5. Guide ....................................................................................................................................... 13
   5.1. Download and Installation .............................................................................................. 13
   5.2. Running the Conformance Test Architecture ................................................................. 13
   5.3. Conformance Test Architecture Features ...................................................................... 13
       5.3.1. Traditional transaction Batch Testing ........................................................................ 13
       5.3.2. Traditional Transaction Editor .................................................................................. 19
           Transaction-Level Editing .............................................................................................. 20
           Record-Level Editing ................................................................................................... 23
           Field-Level Editing ....................................................................................................... 26
           Subfield-Level Editing ................................................................................................. 27
       5.3.3. NIEM XML Transaction Batch Testing ..................................................................... 28
       5.3.4. Options ................................................................................................................... 29
   5.4. Elements of a Result ......................................................................................................... 32
   5.5. Troubleshooting and Errors ............................................................................................ 33
       Parsing (Traditional Encoding) .......................................................................................... 33
       Batch & Testing ............................................................................................................... 37
       Editor ................................................................................................................................. 38
       Options .............................................................................................................................. 39
5.6. References .......................................................................................................................... 40
6. Annex A: Schema Modifications .............................................................................................. 41
Figures

Table 1 – User Guide Version History Summary

Figure 2 - Empty Batch Test Tab

Figure 3 - Files Loaded into the Batch Test Tab

Figure 4 - Batch Test tab showing Overall Results

Figure 5 - Batch Test Tab with a Transaction Selected and Results Displayed in Right Pane

Figure 6 - Batch Test Tab with a Transaction Selected and Statistics Displayed in Right Pane

Figure 7 - The Editor with a Passing Transaction Loaded

Figure 8 – The Editor Add Record Dialog

Figure 9 – The Editor Organize Records Dialog

Figure 10 - The Editor Tab with a Passing Transaction Loaded - Displaying Transaction Level Results

Figure 11 - Editor Tab with Record Expanded

Figure 12 - The Editor Tab with Expanded Field

Figure 13 - The Add New Field Dialog for ASCII/Unicode Records

Figure 14 – The Editor Organize Fields Dialog

Figure 15 - The Editor Tab with Expanded Field

Figure 16 - The Modify Subfields Dialog

Figure 17 – The Editor Tab with Expanded Subfields and Information Items

Figure 18 - The NIEM XML Transaction Batch Testing Tab with Output

Figure 19 - Text Output Result Example

Figure 20 - XML Output Result Example

Figure 21 – Error: Transaction-TableOfContentsRecordLength (Transaction too short / missing data)

Figure 22 – Error: Transaction-TableOfContentsRecordLength (Transaction too long / additional data)

Figure 23 – Error: Additional data saved as Unknown Record Type

Figure 24 – Error: Text Output Log File Cannot Be Found / Missing

Figure 25 – Error: Testing Underway - A test is currently being performed

Figure 26 – Editor Message: File Modified / Save Changes to Open File?

Figure 27 – Editor Message: Data Length Modified – Adjust Record Header Length

Figure 28 – Editor Message: Append Data when adding Binary Field

Figure 29 – Editor Message: Modify Subfields Loss of Data
1. Disclaimer

NIST/ITL BioCTS

For ANSI/NIST-ITL

October 2010

The software was developed by the National Institute of Standards and Technology (NIST), an agency of the Federal Government. Pursuant to Title 15 United States Code Section 105, works of NIST are not subject to copyright protection in the United States and are considered to be in the public domain. Thus, the software may be freely reproduced and used. Please explicitly acknowledge the National Institute of Standards and Technology as the source of the software.

This software is released by NIST as a service and is expressly provided "AS IS." NIST MAKES NO WARRANTY OF ANY KIND, EXPRESS, IMPLIED OR STATUTORY, INCLUDING, WITHOUT LIMITATION, THE IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, NON-INFRINGEMENT AND DATA ACCURACY. NIST DOES NOT REPRESENT OR WARRANT THAT THE OPERATION OF THE SOFTWARE WILL BE UNINTERRUPTED OR ERROR-FREE, OR THAT ANY DEFECTS WILL BE CORRECTED.

NIST does not warrant or make any representations regarding the use of the software or the results thereof, including but not limited to the correctness, accuracy, reliability or usefulness of the software. By using this software or by incorporating this software into another product, you agree to hold harmless the United States Government for any and all damages or liabilities that arise out of such use.

Certain trade names and company products are mentioned in the text or identified. In no case does such identification imply recommendation or endorsement by the National Institute of Standards and Technology, nor does it imply that the products are necessarily the best available for the purpose. With the exception of material marked as copyrighted, information presented in this document is considered public information and may be distributed or copied. Use of appropriate byline/photo/image credits is requested.
## 2. User Guide Version History

As updates and corrections are made to this user guide, new versions may be released. The following table summarizes the changes made during each revision.

<table>
<thead>
<tr>
<th>Version</th>
<th>Date</th>
<th>Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0</td>
<td>August 2015</td>
<td>Initial release of the User Guide</td>
</tr>
<tr>
<td>1.1</td>
<td>October 2015</td>
<td>-Correction to the Record Types supported. For the AN-2013 CTS:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Removed support for Type 13, 14, 15, and 17 for XML. For AN-2011</td>
</tr>
<tr>
<td></td>
<td></td>
<td>and AN-2013, added support for Type 2.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-Updated Annex A: Schema modifications to reflect changes to the</td>
</tr>
<tr>
<td></td>
<td></td>
<td>modified Schema.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-Updated references to GUI functionality due to slight changes</td>
</tr>
</tbody>
</table>

*Table 1 – User Guide Version History Summary*
3. Overview

This document describes the features available in the latest release of Biometric Conformance Test Software (BioCTS) for ANSI/NIST-ITL Version 2 (AN-ITL v2) developed by NIST/ITL Computer Security Division, and will be updated as minor revisions to the software are released. Note that in this document “Version 2” and “BioCTS for AN-ITL v2” describe any BioCTS software released using the major Version 2; all subsequent minor releases and updates are also included in this description (e.g. Beta 2.x.xxxx.xxxxx). The last major BioCTS version released prior to BioCTS for AN-ITL v2 was Beta 1.2.5353.15785 on September 10, 2014. For an overview of the changes made since the previous major version, see Updates from Previous Major Version.

BioCTS for AN-ITL v2 is a desktop application which tests electronic biometric data files, known as transactions, for conformance to NIST Special Publication (SP) 500-290 Data Format for the Interchange of Fingerprint, Facial & Other Biometric Information [1], and NIST Special Publication 500-290 Rev1 (2013) Data Format for the Interchange of Fingerprint, Facial & Other Biometric Information [2]. ANSI/NIST-ITL 1-2011 (AN-2011), ANSI/NIST-ITL 1-2011 Update: 2013 (AN-2013), and related information on the standard can be found at the ANSI/NIST-ITL Standard Homepage [3].

BioCTS for AN-ITL v2 contains many useful features for testing transactions individually or as groups, viewing and analyzing conformance errors and test results, and modifying transaction data to correct conformance errors. Some examples of the software’s functionality include:

- Testing a significant number of transactions (tested at 1,000s+) in a single batch test
- Evaluating the batch or individual transaction results at a high level with visual and textual statistical analyses
- Loading transactions into the BioCTS Editor (Traditional Encoding only), which provides tools for viewing, expanding, and modifying the Records, Fields, Subfields, and Information Items which comprise a transaction

Test Results from each transaction are displayed on-screen and are saved to a user-specified folder with a time stamp.

3.1. Conformance Testing Support

All record types defined in AN-2011 and AN-2013 are supported for conformance testing of Traditional Encoded Transactions, and selected record types are supported in NIEM XML Encoded Transactions.

BioCTS for AN-ITL v2 supports four Conformance Test Suites (CTS) shown below. For each CTS, the supported assertions and record types are also listed.

**CTSs for AN-2011**

The CTSs for AN-2011 implement all test assertions specified in NIST SP 500-295 Revision 1 - Conformance Testing Methodology for ANSI/NIST-ITL 1-2011, Data Format for the Interchange of Fingerprint, Facial & Other Biometric Information [4], plus additional assertions developed to address record types not supported in previous versions of BioCTS for AN-ITL.
• **CTS for AN-2011 Traditional Encoded Transactions**
  - Full Support (Transaction Level and All record types defined in AN-2011)

• **CTS for AN-2011 NIEM XML Encoded Transactions**
  - Transaction Level Assertions
  - Type 1, Transaction Information Record
  - Type 2, User-defined descriptive text record
  - Type 4, High-resolution grayscale fingerprint image
  - Type 10, Facial and SMT image
  - Type 13, Variable-resolution latent image
  - Type 14, Variable-resolution fingerprint image
  - Type 15, Palm print image
  - Type 17, Iris image

**CTSs for AN-2011 Update: 2013**

The CTSs for AN-2011 Update: 2013 implement the test assertions specified in *NIST Special Publication NIST SP 500-304 – Conformance Testing Methodology Framework for ANSI/NIST-ITL 1-2011 Update: 2013, Data Format for the Interchange of Fingerprint, Facial & Other Biometric Information* [5] as well as additional required test assertions specific to the 2013 Update that were not published at the time of writing.

• **CTS for AN-2011 Update: 2013 Traditional Encoded Transactions**
  - Full Support (Transaction Level and All record types defined in AN-2011 Update: 2013)

• **CTS for AN-2011 Update: 2013 NIEM XML Encoded Transactions**
  - Transaction Level Assertions
  - Type 1, Transaction information record
  - Type 2, User-defined descriptive text record
  - Type 4, High-resolution grayscale fingerprint image
  - Type 10, Facial and SMT image
  - Type 12, Forensic dental and oral data
  - Type 22, Non-photographic imagery

Note that Type-2 support in the above CTS’s does not include support for specific profiles nor any user-defined fields. Future releases will add support for additional record types.

### 3.2. Requirements

• Supported Microsoft® Operating Systems
  - Windows® XP™ Service Pack 3
  - Windows® Vista™ Service Pack 2
  - Windows®  7™ Service Pack 1
  - Windows® 8.1™
- Windows® 10™

- Microsoft® .NET 4.0 Framework

Links working as of 8/4/2015
4. Updates from Previous Major Version

For users already familiar with the previous version of BioCTS for AN-ITL, this section provides an overview of changes in functionality and conformance support. All changes in this section are in reference to the previously released major version (Beta 1.2.5353.15785, September 10, 2014). The most notable change from the previous version of the software is the addition of full Traditional Encoding support for all record types defined in AN-2011 and AN-2013. Additionally, several new features have been added to the Editor to allow greater control over modifications to the Record, Field, Subfield, and Information Item structure and organization. The parser has also been improved to provide more meaningful feedback when transactions are malformed or when records are of incorrect length.

Any minor changes not related to adding conformance support or improving functionality and features are not listed in this document. Minor modifications to software are listed on the BioCTS website as a changelog that is updated over time.

4.1. Updates to Conformance Support

The information in this section lists only the conformance support that has been added since the previous release.

AN-2011 Traditional Encoded Transactions

BioCTS for AN-ITL v2 fully supports all record types defined in AN-2011. Support for the following record types was added since the previous version:

- Type 2, User-defined descriptive text*
- Type 7, User-defined image
- Type 8, Signature image
- Type 9, Minutiae data
- Type 16, User-defined variable-resolution testing image
- Type 18, DNA data
- Type 19, Variable-resolution plantar image
- Type 20, Source representation
- Type 21, Associated context
- Type 22, Non-photographic imagery
- Type 98, Information assurance
- Type 99, CBEFF biometric data record

AN-2011 NIEM XML Encoded Transactions

No changes were made to the supported Record Types since the previous version.

Annex A of this document provides details regarding the modifications that had to be made to the XML Schema provided with ANSI/NIST-ITL 1-2011.
AN-2013 Traditional Encoded Transactions

BioCTS for AN-ITL v2 fully supports all record types defined in AN-2013. Support for the following record types was added since the previous version:

- Type 2, User-defined descriptive text*
- Type 7, User-defined image
- Type 8, Signature image
- Type 9, Minutiae data
- Type 11, Forensic and investigatory voice data
- Type 12, Forensic dental and oral data
- Type 13, Variable-resolution latent friction ridge image
- Type 14, Variable-resolution fingerprint image
- Type 15, Variable-resolution palm print image
- Type 16, User-defined variable-resolution testing image
- Type 17, Iris Image
- Type 18, DNA data
- Type 19, Variable-resolution plantar image
- Type 20, Source representation
- Type 21, Associated context
- Type 22, Non-photographic imagery
- Type 98, Information assurance
- Type 99, CBEFF biometric data record

AN-2013 NIEM XML Encoded Transactions

Support for the following record types was added since the previous version:

- Type 12, Forensic dental and oral data
- Type 22, Non-photographic imagery

*Note that Type-2 support in the above CTS’s does not include support for specific profiles nor any user-defined fields.

4.2. Updates to Features and Functionality

There are several new features included in the BioCTS User Interface since the last release:

- **Usability Updates** – Several improvements have been made to the tool’s usability and workflow. Some example include:
• Removal of pop-up prompts after testing and various actions. The user interface now displays a label at the bottom right of the window to show messages that do not interrupt the user's progress. For example:
  - Testing Complete: ❌ Test Complete! Test Time: 00:00:08.5787139
  - Tests In Progress: Testing...
• Option to have output window minimized by default to avoid disrupting user.
  - Shown highlighted in yellow:

- Several usability improvements to the Editor (see below).

- **Advanced Editor Features** – Powerful new Editor features have been added to provide more robust editing and control over transaction data. For details on using these new features, refer to the [Editor Guide](#). New features include:
  - Add Record
  - Remove Record
  - Sort Records (sort records numerically)
  - Arrange Records (reorder using up and down arrows)
  - Show Record Image
  - Sort By Field Number (sort fields numerically within a record)
  - Organize Fields (arrange using up and down arrows)
  - Add Binary Field - Revised (specify byte index for insertion)
  - Modify Subfields
  - View Data Pop-out Window

- **Parsing improvements** – The parsing methods in the previous version of the tool provided poor error messages in some cases by not providing enough detail, such as “Error: Data passed in but not parsed”. Some of the parser improvements are listed below; for more detailed information on understanding errors when parsing, see [Troubleshooting and Tips: Parsing](#).
  - The errors generated by the parser are more informative in the latest version. Parsing error messages now describe how much data was missing or how much extra data was present in relation to the Transaction Contents found in Field 1.003.
  - Previous versions of the tool would “short circuit” during parsing and not continue if any errors were detected during parsing. The latest version attempts to parse and report on as much data as possible, even if some errors are detected during parsing.
  - When modifying data in the Editor and re-running tests, the previous versions of the tool did not properly report errors at times because the data was not reparsed—it was instead created from the transaction saved in memory. The new version forces the data in memory to be reparsed just like a new transaction.
5. Guide

5.1. Download and Installation
Download the installer from the website http://www.nist.gov/itl/csd/biometrics/biocta_download.cfm.

After the download completes, run the install program and follow the on screen instructions presented in the dialog boxes.

5.2. Running the Conformance Test Architecture
To run the CTA software from the Start menu:
Select **All Programs** then select **NIST BioCTS**, then select **ANSI NIST ITL**. The application will be located in this directory for access after it is successfully installed.

After starting BioCTS for AN-ITL v2, specific Conformance Test Suites may be selected from the Options Tab (See Section 5.3.4 for more details). Four Conformance Test Suites are present, and are separated into two categories:

- **Traditional Encoding Conformance Test Suites**
  - ANSI/NIST-ITL 1-2011 Traditional Encoding Conformance Test Suite
  - ANSI/NIST-ITL 1-2011 Update: 2013 Traditional Encoding Conformance Test Suite

- **NIEM XML Encoding Conformance Test Suites**
  - ANSI/NIST-ITL 1-2011 NIEM XML Encoding Conformance Test Suite

Within the Options Tab, a single Conformance Test Suite can be selected for each category.

5.3. Conformance Test Architecture Features

5.3.1. Traditional transaction Batch Testing
The “Traditional Transaction Batch Test” tab allows multiple traditionally encoded transactions (files) to be tested in groups, and displays the overall results for each transaction in the “Files Under Test” pane.

- **Open Files**: loads files into the Batch File List and removes any previously loaded files
- **Add Files**: loads files into the Batch File List by adding them after any previously loaded files
- **Test Files**: tests all files loaded in the Batch File List
- **Stop Testing**: immediately stops all testing of the files in the Batch File List
- **Clear Results**: clears all testing results from the Batch File List (but does not remove the files)
- **Clear Files**: clears all files from the Batch File List (and removes the test results)
Find/Search: Opens text-search window for searching test results (also Ctrl + f). The F3 key locates the next matching result. The Esc key closes the Find/Search Window.

Figure 2 - Empty Batch Test Tab
A significant number of files (1,000s+) can be loaded at in a single batch. The more powerful the hardware used, the more files that can be tested in batch mode. BioCTS Team members perform the tests on conventional Desktop Computers (e.g., Intel® Xeon® Processors, 8 GB of Memory).

Figure 3 - Files Loaded into the Batch Test Tab
The “Batch Test” tab will display the transaction’s overall result with either:

- ❌ - Overall Result of Fail
- ✓ - Overall Result of Pass

The status banner at the bottom right side of the window displays messages regarding the tool’s current actions. While testing, the banner displays “Testing...” in yellow.

![Batch Test tab showing Overall Results](image)

*Figure 4 - Batch Test tab showing Overall Results*
Once all files in the Batch File List have completed testing, the status banner at the bottom right side of the window displays “Test Complete! ...” as well as the testing time in green.

Textual output results for each transaction can be viewed by clicking on the desired filename in the “File Under Test” pane. The complete textual results are displayed in the pane to the right.
In addition to the detailed Textual Output Log, each file has a statistical breakdown of the results. There is also a high level graphical view of the errors per Record within a Transaction. These graphs can be viewed within BioCTS itself, or saved as an image.

![Figure 6 - Batch Test Tab with a Transaction Selected and Statistics Displayed in Right Pane](image-url)
5.3.2. Traditional Transaction Editor

The Traditional Transaction Editor is capable of interactively displaying all entities contained in a traditionally encoded transaction, allowing a user to edit, add and remove records and their contents.

The Editor is designed to display as little or as much data as desired by the user by making use of expander sections, which can be expanded to display more information by pressing the button within sections. Below is a brief description of the tools available in the Editor:

- **Open File**: Loads a single file into the Editor and automatically tests it
- **Save File**: Saves the file to disk (location chosen by user) with any modifications from Editor
- **Test Files**: Tests or re-tests the file currently loaded into the Editor
- **Reload from Disk**: Reloads the current file from disk, discarding any changes from the Editor
- **Auto Fix**: Attempts to fix common file issues with Record Length and Transaction Contents
- **Sort**: Sorts fields or records numerically (context dependent)
- **Organize/Arrange**: Manually repositions fields or records using arrows (context dependent)
- **Add**: Add field or record (context dependent)
- **Remove/Delete**: Removes a field or record (context dependent)
- **Modify Subfields**: Change number of subfields and information items (with or without data)
- **View/Modify Data**: Displays data in a pop-out window and allows editing
- **Show Record Image**: Attempts to display the image found in the record (context dependent)

---

1 Many Editor tools and actions, including adding or removing Records, Fields, and Subfields, cause the transaction to be re-tested automatically. In some cases, the Auto Fix feature will also be run to ensure the Transaction is formatted properly after the change takes place. For example, the removal of a Field will necessitate an adjustment to the Record Header Length to prevent parsing errors.
**Transaction-Level Editing**

The Editor displays the data and associated results in an expandable, hierarchical format, and allows editing of existing data using text fields.

![Image of the Editor with a Passing Transaction Loaded](image)

**Figure 7 - The Editor with a Passing Transaction Loaded**

At this level, new Records may be added by clicking the green plus icon (the user will be prompted for a Record Type number).

![Image of the Editor Add Record Dialog](image)

**Figure 8 – The Editor Add Record Dialog**

The Records may also be reorganized within the Transaction by using the numerical sort or the manual rearrange buttons.
Figure 9 – The Editor Organize Records Dialog
The Editor displays results in expander sections where appropriate. In this example, the Transaction-level results are shown. This method of viewing results applies to all constructs in the Transaction, including Records, Fields, Subfields, and Information Items. Fields that do not contain Subfields or Information Items\(^2\) show only a Results list and a data field.

Figure 10 - The Editor Tab with a Passing Transaction Loaded - Displaying Transaction Level Results

\(^2\) Technically, all Fields contain at least one Subfield that contains one Information Item. See “How separators are used” in Annex B of NIST SP 500-290 Rev. 1. BioCTS displays such cases as “Fields containing a single value”, which is the way such Fields are listed in NIST SP 500-290 Rev. 1 in Record Layout Tables.
**Record-Level Editing**

When a record is expanded, the Editor displays a list of the fields contained in the record. The selected or expanded Record may be deleted by clicking the “Remove Record” button. In any expanded Record, the Add, Sort, and Organize Field options are available. If the Record contains an image, the option to “Show Record Image” will be available (not shown in Record-Type 1 below).

![Figure 11 - Editor Tab with Record Expanded](image)

The Add New Field prompt is context-dependent and will provide different options based upon whether the field is in an ASCII/Unicode-encoded Record Type or a Binary Record Type. The window provides visual feedback of what the Field will look like.

For ASCII/Unicode Records, the Visual Display will show the breakdown of the Subfield and Information Item organization.
For Binary Records, the Visual Display will show the inserted binary field data in the context of the binary Record data.

The Fields may be reorganized within the Transaction by using the numerical sort or the manual rearrange buttons.
If the selected Record contains an image, the “Show Record Image” button will be available. Clicking the button will show the image and its metadata. If the image type is not supported or cannot be read by BioCTS, a “No Preview Available” message will be displayed.
**Field-Level Editing**

When expanded, a field displays expanders for the field-level results and the data that is held within the field. At this level, a Field can be deleted from a record. The values in the data text fields can be edited. Three buttons are provided to assist when a large amount of data is being manipulated (such as in an image field):

- **Load Data**: Loads data into the contents of a field/subfield/information item
- **Save Data**: Saves the contents of a field/subfield/information item to a file
- **Pop-Out Data**: Displays data in a large pop-out window for easy viewing and editing

![Figure 15 - The Editor Tab with Expanded Field](image)

The “Modify Subfields” allows the user to modify the number of Subfields and Information Items, and decide whether to keep the current data. In the example below, a second Information Item is being added, and the previous data is retained in the first Information Item.

![Figure 16 - The Modify Subfields Dialog](image)
**Subfield-Level Editing**

When a Field is expanded that contains multiple Subfields or Information Items, the containing Subfields and Information Items are displayed with their own set of data and Results. The Subfields and Information Item data can be edited in the same manner as Field data, by using the provided text field or the provided buttons for loading, saving, and modifying data.

![Editor Tab with Expanded Subfields and Information Items](image.png)

**Figure 17 – The Editor Tab with Expanded Subfields and Information Items**
5.3.3. **NIEM XML Transaction Batch Testing**

The NIEM XML Transaction Batch Testing tab contains the same features found within the Traditional Transaction Batch Testing Tab, with the exception of the Graphical Display of Statistics. In XML Transaction Batch, the results display a summary of the SHA-2 Hash values of the Schema files used during testing of the files.

![The NIEM XML Transaction Batch Testing Tab with Output](image-url)

*Figure 18 - The NIEM XML Transaction Batch Testing Tab with Output*
5.3.4. Options

The “Options” tab provides the testing and output customization choices described below.

Note that the Options tab is disabled during testing to prevent changes that may compromise the integrity of the Test Results.

- **ANSI/NIST-ITL Conformance Test Suite Selection**: The Options tab is where the Conformance Test Suites are selected. There are drop-down menus that allow for selection of a Traditional Conformance Test Suite, and a NIEM XML Conformance Test Suite.

- **Output Directory**: The Options tab also includes a directory selection of where BioCTS saves log files. Within this directory, time-stamped folders are generated for each batch test performed.

  - The time-stamped folder has the following format:
    - yyyy – 4 digit year (e.g. 2012)
    - MM – 2 digit month (e.g. 10)
    - dd – 2 digit day (e.g. 31)
    - HH – 2 digit hour in 24-hour scale (e.g. 13)
    - mm – 2 digit minutes (e.g. 59)
    - ss – 2 digit seconds (e.g. 22)

  - In the example provided below:
    - Text Output will be generated in the directory:
      - C: \ Users\dyaga\Desktop\BioCTS for ANSI NIST ITL Output\2012.10.31.13.59.22\Text Output
    - XML Output will be generated in the directory:
      - C: \ Users\dyaga\Desktop\BioCTS for ANSI NIST ITL Output\2012.10.31.13.59.22\XML Output

- **ANSI/NIST-ITL 1-2011 XML Schema Extension or Constraint Location / ANSI/NIST-ITL 1-2011 Update: 2013 XML Schema Extension or Constraint Location**: The Options tab is also where the Schema file location is specified. BioCTS provides a set of default Schema Files which are based on the official ANSI/NIST-ITL Schemas released with the standard. However, the 2011 Schema files have been modified for use in the BioCTS AN-2011 CTS; see Annex A of this document, which provides details regarding the XML Schema modifications and rationale for the changes. A Constraint or Extension Schema may be used in place of the Schema files provided with the software. The features provided (and test outcome for each usage are detailed below):

  - **Extension Schemas**: Support for Extension Schemas is provided to allow substitution elements to be specified for user-defined information in the standard such as the user-defined abstract elements found in Record Type-2.

---

3 The Apply button has been removed based upon user feedback—changes made in the Options Tab are automatically applied.
- **Constraint Schemas**: Support for Constraint Schemas is provided to allow constraints and restrictions to be defined for use in particular applications such as standard profiles like the FBI EBTS.

- **User Defined Fields Tested Against Default Schemas**: When XML files with User-Defined fields are tested with the Default Schema Files, the XML files may fail the Schema validation test (A User-Defined Schema should be used).

- **Tests against User-Defined Schemas**: When a user-defined (constraint or extension) Schema is used, the validity of the test results reported by BioCTS depends upon the validity of the User-Defined Schema. User-Defined Schemas are valid only if they conform to the requirements in Section C.3 of NIST SP 500-290 and Section C.3 of NIST SP 500-290 Rev1 (2013). BioCTS does not verify that User-Defined Schemas are compliant with NIST SP 500-290—only use Schema files from trusted sources.
  - If the user-defined Schema is not syntactically correct, BioCTS will warn the user.
  - If the XML files tested contain User-Defined Fields that the user-defined Schema does not contain, the XML files will fail the Schema validation test.

- **User-Defined Field Override File Location**: A New feature to this release is the specification of a User-Defined Field Override File. There are many “User Defined” Fields specified in the ANSI/NIST-ITL standards (e.g., Type 02 Fields), and by default they are shown as “User Defined Field” within the Editor and Test Output Logs. This feature allows a user to override the names of these fields so that they are more descriptive than the default “User Defined Field”.

- **Output Options**: Options exist to display the Log folder after a test, the collection of enhanced statistics of the Tests performed within the software, and the option to generate XML logs or not. An option is available to force the Log Folder to be opened minimized to avoid interrupting the user with a pop-up window.
Figure 16 - The Options Tab
5.4. **Elements of a Result**

The result of any test is comprised of 4 elements:

- **Test Name** – A string to define what test took place
- **Test Level** – Testing can take place at multiple levels
  - Parse – A parsing error occurred with this element
  - L1 – A level 1 test
  - L2 – A level 2 test
  - L3 – A level 3 test
- **Test Result**
  - Ok – The test passed
  - Warning – The test passed, but warranted a warning statement (i.e. an “Unspecified” value, etc.)
  - Error – The test failed
  - Critical Error – The test failed and may be preventing other tests from being performed
- **Test Message** – A description of what went on during the test, and possible error message

The Text Output Log’s format has been altered since the initial release of BioCTS for ANSI/NIST-ITL. The information presented in a Test Result has been significantly expanded; the test message is generated with as much information as possible, including various operators and operands used within the test. The following is an example of the current Test Results, where each element of the Result is clearly labelled within the Text Output Log:

```text
<table>
<thead>
<tr>
<th>Level</th>
<th>Test Name</th>
<th>Result</th>
<th>Test Message</th>
</tr>
</thead>
<tbody>
<tr>
<td>L1</td>
<td>1.001-LEN-CharCount</td>
<td>Ok</td>
<td>Greater Than or Equal Test: Is the character count present (3) greater than or equal to the minimum non-negative Integer value (1)?</td>
</tr>
</tbody>
</table>
```

*Figure 19 - Text Output Result Example*

The same exact Result is encoded in the XML Log as follows:

```xml
<Result>
  <Level>L1</Level>
  <Message>
    Greater Than or Equal Test: Is the character count present (3) greater than or equal to the minimum non-negative Integer value (1)?
  </Message>
  <Results>Ok</Results>
  <Test>1.001-LEN-CharCount</Test>
</Result>
```

*Figure 20 - XML Output Result Example*
5.5. Troubleshooting and Errors
This section contains information to assist in understanding the normal operation of BioCTS for AN-ITL v2 and common errors encountered when using the software. This is not a comprehensive listing of all errors that may be reported.

Parsing (Traditional Encoding)
The parsing engine will continue to parse the transaction even when it encounters some critical errors such as data length discrepancies. Errors messages are reported to assist in discovering why the transaction is malformed.

Operation: The parsing engine relies heavily on the contents of Field 1.003-CNT (Transaction Content) and the Record Header (Field xx.001) for each Record. The general parsing procedure follows:

- Each pair of REC (record category code) and IDC (information designation character) values found in 1.003-CNT is read and saved to a list.
- The value of Field 1.001 (the record length) is used to progress to the next Record location.
- The parser uses each REC value (beginning with the first) to determine if the Record is binary or ASCII/Unicode.
  - If ASCII, the field at the location is parsed and the Field value is saved as the record length.
  - If binary, the first 4 bytes are read in as the Field value, which is saved as the record length.
- The IDC value (Field xx.002) is also saved, and the record length value from the previous step is used to progress to the next Record location. The process is repeated until all REC values have been exhausted or the end of the file is encountered. If the end of the file is encountered before processing all REC values, or if extra data is discovered after processing all REC values, an error is reported.
- The list of REC values, record lengths, and IDC values is used to create Records and parse their containing Fields. These Records and IDC values are compared against the list obtained from 1.003-CNT. Any discrepancies are reported as errors.
Data Too Short / Missing Data:

If the data in the Transaction is too short when compared to the values found in 1.003-CNT and Field xx.001 of each Record, the tool will report the error in the Transaction-related results under the test name “Transaction-TableOfContentsRecordLength”.

Figure 21 – Error: Transaction-TableOfContentsRecordLength (Transaction too short / missing data)
Data Too Long / Extra Data:

If the data in the Transaction is too long when compared to the values found in 1.003-CNT and Field xx.001 of each Record, the tool will report the error in the Transaction-related results under the test name “Transaction-TableOfContentsRecordLength”.

![Figure 22 – Error: Transaction-TableOfContentsRecordLength (Transaction too long / additional data)](image-url)
The software will save any additional data to an “Unknown Record Type”, shown below in the Editor:

![Figure 23 – Error: Additional data saved as Unknown Record Type](image-url)
**Batch & Testing**
When testing under batch mode, there are a few common errors that are encountered due to the way the tools processes and reports logged test results.

**Log File Missing**: The Batch Tester relies on saved text data log files to report results to the screen in the user interface. If those log files are deleted or moved, or if permission settings do not allow access to the folder where they are saved, the software may report the following error:

![Figure 24 – Error: Text Output Log File Cannot Be Found / Missing](image)

**Testing Underway**: The Batch Tester is designed to be responsive, allowing the user to interact with the tool while tests are running. However, some features cannot run concurrently with Batch testing. For example, if the user attempts to run the Editor while the Batch Testing is in progress, the following error message may be displayed:

![Figure 25 – Error: Testing Underway - A test is currently being performed](image)
Editor
The Editor provides robust features for viewing and editing data at all levels of the transaction, which may introduce some confusion in the way it processes handles modified data. It is important to note that many options cause the Auto Fix to run automatically to avoid parsing and data length errors.

Operation: The Editor works by reading in data from the file and creating a copy of that data as structured Records, Fields, Subfields, and Information Items in the computer memory. When the data is modified, it is changed only in memory, not on disk. Any changes must be saved to disk using the Save button, or they will be lost after closing the tool. When the test/run button is pressed, the data in memory is written out to binary/ASCII form in memory and reparsed to simulate reading the newly modified data from a file. This method increases test time by a small margin, but ensures that all errors are caught, even if they are testing during parsing. When the Auto Fix button is pressed, common errors are checked, including Record Header Lengths and Transaction Content (Field 1.003). If these common parsing errors are found, the contents of the Records are used to correct the header/metadata. For example, the length of the modified record is inserted into Field xx.001 / Record Header Length, and its IDC value is inserted into 1.003-IDC.

Save Modified File: If modifications were made to the transaction in the Editor and the changes were not saved to disk, the software will provide the following prompt if the user attempts to leave the Editor tab:

Data Length Modified: If modifications made in the Editor may have caused the data length to change and the user attempts to re-test the data, the software will prompt the user to verify that the Record Header lengths have been updated (with the option to automatically adjust it).
Append Binary Data: When adding a binary field, if the data insertion index exceeds the end of the field, the user will be prompted to append the data to the end of the field. Specifying an index beyond the end of the field is also the recommended method for appending new field data.

Modify Subfields – Loss of Data: When modifying Subfields and Information Items, if the new total number of Information Items is less than the original total, the following warning will be displayed to avoid unintentional data loss:

Options
The Options tab is disabled during testing in the Editor or Batch to prevent undesired changes that may negatively affect the Test Results. Also, the Apply Button has been removed due to user feedback; all changes made in the Options Tab are automatically applied.
6. References

1. NIST Special Publication 500-290 Data Format for the Interchange of Fingerprint, Facial & Other Biometric Information
3. ANSI/NIST-ITL Standard Homepage
   http://www.nist.gov/itl/iad/ig/ansi_standard.cfm
4. NIST SP 500-295 Revision 1 – Conformance Testing Methodology for ANSI/NIST-ITL 1-2011, Data Format for the Interchange of Fingerprint, Facial & Other Biometric Information
   http://csrc.nist.gov/groups/ST/BiomResCenter/CTA_BETA/DRAFT_NIST_SP_500_295_REV_1_DRAFT.pdf
7. SHA-2 Wikipedia Summary
Annex A: Schema Modifications

Modifications are made only when the Schema interferes with conformance of the base standard requirements or is more restrictive than the base standard. Some examples include but are not limited to invalid value restrictions and enumerated values, missing or invalid element definitions, and improper ordering of elements. All modifications to the Schema files are tagged in the Schema files with a comment <!--COMMENT--> that includes the tag “BioCTS”. Searching the modified Schema for “BioCTS” will locate all of the modifications.

Current Modified Schema Files - All releases AFTER Beta 2.0.5724.16724:

For AN-2013, no modifications are made to the XML Schema files posted on the ANSI/NIST-ITL Standard Website (http://biometrics.nist.gov/cs_links/standard/ansi_2015/Schema-MRT/ANSI-NIST-ITL-2013_FINAL.zip). Modifications are no longer necessary because many of the BioCTS team’s recommendations were adopted in the latest update to the XML Schema.

For AN-2011, the modifications in this XML Schema are marked with <!--[BioCTS]-->. Searching for [BioCTS] will located the changes. The modifications are located in biometrics.xsd, which is created by using biometrics.errata.xsd (http://biometrics.nist.gov/cs_links/standard/ansi_2012/biometrics.errata.xsd) and adding the following modifications:

- **Modified**: Modified the element `biom:FaceImageAcquisitionProfile` (a child element of the complex type `FaceImageType`) to include `minOccurs="0"`
  
  *Reason*: Field 10.013 has a min occurrence of 0, max occurrence of 1. By not having the `minOccurs="0"` the Schema was requiring this element to always be present.

  **BEFORE:**
  ```xml
  <xsd:element ref="biom:FaceImageAcquisitionProfile"/>
  ```

  **AFTER:**
  ```xml
  <xsd:element ref="biom:FaceImageAcquisitionProfile" minOccurs="0"/> <!--BioCTS added minOccurs="0", see field 10.013 descriptions in Table 58 or Annex G-->
  ```

- **Added**: The value 18 for “Unknown friction ridge” to the Simple Type `FingerPositionCodeSimpleType`.
  
  *Reason*: Value 18 is specified as a valid value in Table 8 of the AN-2011 standard, but was not present in the Schema.

  **ADDED:**
  ```xml
  <xsd:enumeration value="18"/> <!--BioCTS Added enumeration 18, see Table 8-->
  <xsd:annotation>
    <xsd:documentation>Unknown friction ridge</xsd:documentation>
  </xsd:annotation>
  </xsd:enumeration>
  ```

Previous Versions - Beta 2.0.5724.16724 and earlier:

The following sections provide detailed justifications for the XML Schema modifications.
AN-2011 Schema Modifications (applicable to 2013 Update):

The following list of modifications was made to the AN-2011 version of the Schema and also retained for the AN-2013 update.

I. Additions and Modifications to:
subset\niem\domains\biometrics\1.0\biometrics.xsd

- **Modified**: Modified the element `biom:FaceImageAcquisitionProfile` (a child element of the complex type `FaceImageType`) to include `minOccurs="0"`
  
  **Reason**: Field 10.013 has a min occurrence of 0, max occurrence of 1. By not having the `minOccurs="0"` the Schema was requiring this element to always be present.

  **BEFORE**:
  ```xml
  <xsd:element ref="biom:FaceImageAcquisitionProfile"/>
  ```

  **AFTER**:
  ```xml
  <xsd:element ref="biom:FaceImageAcquisitionProfile" minOccurs="0"/>
  ```

- **Added**: The value 18 for “Unknown friction ridge” to the Simple Type `FingerPositionCodeSimpleType`.
  
  **Reason**: Value 18 is specified as a valid value in Table 8 of the AN-2011 standard, but was not present in the Schema.

  **ADDED**:
  ```xml
  <xsd:enumeration value="18">
    <xsd:annotation>
      <xsd:documentation>Unknown friction ridge</xsd:documentation>
    </xsd:annotation>
  </xsd:enumeration>
  ```

- **Modified**: The Element `FingerprintImageStitchedIndicator` to only accept the character “Y”.
  
  **Reason**: The Element was defined in the Schema as a Boolean type (`true` or `false`). This prevented the base requirement of only allowing a single alphabetic character of “Y” (Field 14.027 in Table 71 of the AN-2011 standard).

  **BEFORE**:
  ```xml
  <xsd:element name="FingerprintImageStitchedIndicator" type="niem:xsd:boolean">
    <xsd:annotation>
      <xsd:documentation>True if the image was formed by stitching together separately captured images; false otherwise.</xsd:documentation>
    </xsd:annotation>
  </xsd:element>
  ```

  **AFTER**:
  ```xml
  <xsd:element name="FingerprintImageStitchedIndicator" type="niem:xsd:boolean">
    <xsd:annotation>
      <xsd:documentation>Y if the image was formed by stitching together separately captured images; not present otherwise.</xsd:documentation>
    </xsd:annotation>
  </xsd:element>
  ```
Additional AN-2011: Update 2013 Schema Modifications:

The following list of modifications was made to the AN-2011: Update 2013 version of the Schema to address instances where the Schema conflicted with the base standard in a way that interfered with conformance testing.

- **Added:** Before line 2397, added support for Field 10.039
  
  **Reason:** The base standard text states that 10.039 should be allowed for any image type (it is not allowed for FaceImageType in the original Schema).

  **Added XML Content:**
  
  ```xml
  <xsd:element ref="biom:PhysicalFeatureReferenceIdentification" minOccurs="0"/>
  <!--BioCTS added (10.039 can be used for any IMT type) -->
  ```

- **Added:** After line 2397, added support for Fields 10.034, 10.046, 10.047, 10.048, and 10.051.
  
  **Reason:** The base standard text allows these fields for any image type (they are not allowed for FaceImageType in the original Schema).

  **Added XML Content:**
  
  ```xml
  <xsd:element ref="biom:ImageCaptureDateEstimateRange" minOccurs="0"/>
  <xsd:element ref="biom:SubjectExistentialDetails" minOccurs="0"/>
  <xsd:element ref="nc:OrganizationUnitName" minOccurs="0"/>
  <xsd:element ref="biom:PatternedInjuryDetail" minOccurs="0"/>
  <xsd:element ref="biom:RulerScalePresenceInformation" minOccurs="0"/>
  ```

- **Modified:** Starting at line 12469, modified enumeration values.
  
  **Reason:** The value <> (less than greater than) is indicated by the base standard, but the Schema only allows LTGT. This modification allows the <> value instead.

  **BEFORE:**
  
  ```xml
  <xsd:restriction base="xsd:token">
    <xsd:enumeration value="I">
      <xsd:annotation>
        <xsd:documentation>A clear cut groove running vertically across the lip</xsd:documentation>
      </xsd:annotation>
    </xsd:enumeration>
  </xsd:restriction>
  ```

  **AFTER:**
  
  ```xml
  <xsd:restriction base="xsd:token">
    <xsd:enumeration value="I">
      <xsd:annotation>
        <xsd:documentation>A clear cut groove running vertically across the lip</xsd:documentation>
      </xsd:annotation>
    </xsd:enumeration>
  </xsd:restriction>
  ```
A data type for lip print characterization code

- **I** - A clear cut groove running vertically across the lip
- **IP** - Partial-length groove of Type I
- **II** - Branched groove
- **III** - An intersected groove
- **IV** - A reticular groove
- **LTGT** - Indicator for center of the lip
- **O** - Other pattern(s)

---

---
An intersected groove

A reticular groove

Other pattern(s)

- **Modified**: Starting at line 12647, modified incorrect enumerated values.

**Reason**: The enumerated values were incomplete/invalid.

**BEFORE**:

- `<xsd:enumeration value="10">`
  - `<xsd:annotation>`
    - `<xsd:documentation>`Cleft lip (cheiloschisis) – bilateral complete`<xsd:documentation>`
  - `<xsd:annotation>`
- `<xsd:enumeration value="11">`
  - `<xsd:annotation>`
    - `<xsd:documentation>`Piercing – upper lip`<xsd:documentation>`
  - `<xsd:annotation>`
- `<xsd:enumeration value="12">`
  - `<xsd:annotation>`
    - `<xsd:documentation>`Piercing – lower lip`<xsd:documentation>`
  - `<xsd:annotation>`
- `<xsd:enumeration value="13">`
  - `<xsd:annotation>`
    - `<xsd:documentation>`Tattoo – upper lip`<xsd:documentation>`
  - `<xsd:annotation>`
- `<xsd:enumeration value="14">`
  - `<xsd:annotation>`
    - `<xsd:documentation>`Tattoo – lower lip`<xsd:documentation>`
  - `<xsd:annotation>`
- `<xsd:enumeration value="15">`
  - `<xsd:annotation>`
    - `<xsd:documentation>`Other`<xsd:documentation>`
AFTER:
<xsd:enumeration value="10"><!
<xs:annotation>
  <xsd:documentation>Cleft lip (cheiloschisis) – unilateral complete - left</xsd:documentation>
</xsd:annotation>
</xsd:enumeration>
<xsd:enumeration value="11">
  <xsd:annotation>
    <xsd:documentation>Cleft lip (cheiloschisis) – unilateral complete - right</xsd:documentation>
  </xsd:annotation>
</xsd:enumeration>
<xsd:enumeration value="12">
  <xsd:annotation>
    <xsd:documentation>Cleft lip (cheiloschisis) – bilateral incomplete</xsd:documentation>
  </xsd:annotation>
</xsd:enumeration>
<xsd:enumeration value="13">
  <xsd:annotation>
    <xsd:documentation>Cleft lip (cheiloschisis) – bilateral complete</xsd:documentation>
  </xsd:annotation>
</xsd:enumeration>
<xsd:enumeration value="14">
  <xsd:annotation>
    <xsd:documentation>Piercing – upper lip</xsd:documentation>
  </xsd:annotation>
</xsd:enumeration>
<xsd:enumeration value="15">
  <xsd:annotation>
    <xsd:documentation>Piercing – lower lip</xsd:documentation>
  </xsd:annotation>
</xsd:enumeration>
<xsd:enumeration value="16">
  <xsd:annotation>
    <xsd:documentation>Tattoo – upper lip</xsd:documentation>
  </xsd:annotation>
</xsd:enumeration>
<xsd:enumeration value="17">
  <xsd:annotation>
    <xsd:documentation>Tattoo – lower lip</xsd:documentation>
  </xsd:annotation>
</xsd:enumeration>
<xsd:enumeration value="99">
  <xsd:annotation>
    <xsd:documentation>Other (describe in LPPT)</xsd:documentation>
  </xsd:annotation>
</xsd:enumeration>

• **Modified**: Starting at line 12736, modified incorrect enumerated value.
• **Reason**: The enumerated value in LipPrintSurfaceCodeSimpleType was invalid.

BEFORE:
<xsd:enumeration value="4">
  <xsd:annotation>
    <xsd:documentation>Piercing – upper lip</xsd:documentation>
  </xsd:annotation>
</xsd:enumeration>

AFTER:
<xsd:enumeration value="9"><!
  <xs:annotation>
    <xsd:documentation>Piercing – upper lip</xsd:documentation>
  </xsd:annotation>
</xsd:enumeration>

• **Modified**: Starting at line 12783, modified incorrect enumerated value.
• **Reason**: The enumerated value in LipPrintMediumCodeSimpleType was invalid.
BEFORE:
<xsd:enumeration value="4"/>

AFTER:
<xsd:enumeration value="9"/>

- **Modified**: Starting at line 15784, modified the last element in the sequence.
  
  **Reason**: The element name `biom:CaptureOrganization` should not be used to represent field 12.047. Instead, `nc:OrganizationUnitName` should be used because it matches the description in the base standard text, and is used in other record types to represent field xx.047.

BEFORE:
<xsd:complexType name="DentalCaptureType">
  <xsd:annotation>
    <xsd:documentation>A data type for a set of information regarding the dental image</xsd:documentation>
  </xsd:annotation>
  <xsd:complexContent>
    <xsd:extension base="s:ComplexObjectType">
      <xsd:sequence>
        <xsd:element ref="biom:CaptureLocation" minOccurs="0"/>
        <xsd:element ref="biom:CaptureUTCDateTime" minOccurs="0"/>
        <xsd:element ref="biom:CaptureOrganization" minOccurs="0"/>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>

AFTER:
<xsd:complexType name="DentalCaptureType">
  <xsd:annotation>
    <xsd:documentation>A data type for a set of information regarding the dental image</xsd:documentation>
  </xsd:annotation>
  <xsd:complexContent>
    <xsd:extension base="s:ComplexObjectType">
      <xsd:sequence>
        <xsd:element ref="biom:CaptureLocation" minOccurs="0"/>
        <xsd:element ref="biom:CaptureUTCDateTime" minOccurs="0"/>
        <xsd:element ref="nc:OrganizationUnitName" minOccurs="0"/>-- BioCTS modified (was previously biom:CaptureOrganization-->
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>

- **Modified**: Starting at line 177, modified the last element in the sequence.
  
  **Reason**: The element name `biom:CaptureOrganization` should not be used to represent field 22.047. Instead, `nc:OrganizationUnitName` should be used because it matches the description in the base standard text, and is used in other record types to represent field xx.047.

BEFORE:
<xsd:complexType name="BiometricCaptureType">
  <xsd:annotation>
    <xsd:documentation>A data type for a set of information regarding the captured biometric information</xsd:documentation>
  </xsd:annotation>
  <xsd:complexContent>
    <xsd:extension base="s:ComplexObjectType">
      <xsd:sequence>
      </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
Modified: Starting at line 16036, modified the type of the element to match the standard description.

Reason: The element biom:SubjectEstimatedDeathDate should be of type niem:xsd:dateTime to accommodate the time values (as indicated by the base standard description for Field 12.006-DEDD).

BEFORE:

```xml
<xsd:element name="SubjectEstimatedDeathDate" type="nc:DateType">
       <xsd:annotation>
          <xsd:documentation>A date of the subject death</xsd:documentation>
       </xsd:annotation>
   </xsd:element>
```

AFTER:

```xml
<xsd:element name="SubjectEstimatedDeathDate" type="niem:xsd:dateTime">
       <xsd:annotation>
          <xsd:documentation>A date of the subject death</xsd:documentation>
       </xsd:annotation>
   </xsd:element>
```

Modified: Starting at line 16031, modified the type of the element to match the standard description.
**Reason:** The element `biom:SubjectDateRangeText` should be of type `xsd:duration` to accommodate the values indicated by the base standard description for Field 12.006-DRDE.

**BEFORE:**

```
<xsd:element name="SubjectDateRangeText" type="nc:TextType">
    <xsd:annotation>
        <xsd:documentation>A kind of subject range of death date estimate</xsd:documentation>
    </xsd:annotation>
</xsd:element>
```

**AFTER:**

```
<xsd:element name="SubjectDateRangeText" type="xsd:duration">
    <xsd:annotation>
        <xsd:documentation>A kind of subject range of death date estimate</xsd:documentation>
    </xsd:annotation>
</xsd:element>
```

- **Added:** After line 16972, added support for Field 22.999 (`nc:BinaryBase64Object`)
  
  **Reason:** The base standard allows for Field 22.999 to contain an image, but the Schema does not.

  **Added XML Content:**
  
  ```xml
  <!--BioCTS added 22.999-->
  <xsd:element ref="nc:BinaryBase64Object" minOccurs="0"/>
  ```

- **Added:** After line 16983, added support for Field 22.994 (`biom:SourceExternalFileReferenceText`)
  
  **Reason:** The base standard allows for Field 22.999 to contain an image, but the Schema does not.

  **Added XML Content:**
  
  ```xml
  <!--BioCTS added 22.994-->
  <xsd:element ref="biom:SourceExternalFileReferenceText" minOccurs="0"/>
  ```

**II. Additions and Modifications to: exchange\itl.xsd**

- **Modified:** Starting at line 87, modified ordering of elements.
  
  **Reason:** T2C comes before user defined fields, not after (according to Annex G). Changed ordering of elements defined in `CrossReferencePackageDataRecordType`

  **BEFORE:**
  
  ```xml
  <xsd:complexType name="CrossReferencePackageDataRecordType">
      <xsd:annotation>
          <xsd:documentation>A data type for a biometric record including Type 2 cross reference</xsd:documentation>
      </xsd:annotation>
      <xsd:complexContent>
          <xsd:extension base="itl:PackageDataRecordType">
              <xsd:sequence>
                  <xsd:element ref="biom:Type2CrossReferenceIdentification" minOccurs="0"/>
              </xsd:sequence>
          </xsd:extension>
      </xsd:complexContent>
  </xsd:complexType>
  ```

  **AFTER:**
  
  ```xml
  <xsd:complexType name="CrossReferencePackageDataRecordType">
      <xsd:annotation>
          <xsd:documentation>A data type for a biometric record including Type 2 cross reference</xsd:documentation>
      </xsd:annotation>
      <xsd:complexContent>
          <xsd:extension base="itl:PackageDataRecordType">
              <xsd:sequence>
                  <xsd:element ref="biom:Type2CrossReferenceIdentification" minOccurs="0"/>
              </xsd:sequence>
          </xsd:extension>
      </xsd:complexContent>
  </xsd:complexType>
  ```
AFTER:

```xml
<complexType name="CrossReferencePackageDataRecordType"> <!-- BioCTS modified -->
  <annotation>
    <documentation>A data type for a biometric record including Type 2 cross reference.</documentation>
  </annotation>
  <complexContent>
    <extension base="s:ComplexObjectType">
      <sequence>
        <element ref="biom:RecordCategoryCode"/>
        <element ref="biom:ImageReferenceIdentification"/>
        <element ref="biom:Type2CrossReferenceIdentification" minOccurs="0"/>
        <element ref="itl:UserDefinedFields" minOccurs="0" maxOccurs="unbounded"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
```

- **Modified**: Starting at line 99, modified ordering of elements.
  
  **Reason**: Cross references come before user defined fields, not after (according to Annex G).
  Changed ordering of elements defined in
  
  MultipleCrossReferencePackageDataRecordType

BEFORE:

```xml
<complexType name="MultipleCrossReferencePackageDataRecordType">
  <annotation>
    <documentation>A data type for a biometric record including Type 2, 10 and 22 cross references</documentation>
  </annotation>
  <complexContent>
    <extension base="itl:PackageDataRecordType">
      <sequence>
        <element ref="biom:Type2CrossReferenceIdentification" minOccurs="0"/>
        <element ref="biom:Type10CrossReferenceIdentification" minOccurs="0" maxOccurs="99"/>
        <element ref="biom:Type22CrossReferenceIdentification" minOccurs="0" maxOccurs="99"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
```

AFTER:

```xml
<complexType name="MultipleCrossReferencePackageDataRecordType"> <!-- BioCTS modified -->
  <annotation>
    <documentation>A data type for a biometric record including Type 2, 10 and 22 cross references</documentation>
  </annotation>
  <complexContent>
    <extension base="s:ComplexObjectType">
      <sequence>
        <element ref="biom:RecordCategoryCode"/>
        <element ref="biom:ImageReferenceIdentification"/>
        <element ref="biom:Type2CrossReferenceIdentification" minOccurs="0"/>
        <element ref="biom:Type10CrossReferenceIdentification" minOccurs="0" maxOccurs="99"/>
        <element ref="biom:Type22CrossReferenceIdentification" minOccurs="0" maxOccurs="99"/>
        <element ref="itl:UserDefinedFields" minOccurs="0" maxOccurs="unbounded"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
```
Modified: Starting at line 506, removed the reference to biom:SourceExternalFileReferenceText.
Reason: biom:SourceExternalFileReferenceText user is already listed in biom:NonPhotographicImageryDetail, this listing was a duplicate.
REMOVED:
<xsd:element ref="biom:SourceExternalFileReferenceText" minOccurs="0"/>

Modified: Starting at line 507, removed the reference to biom:SourceImage.
Reason: biom:SourceImage user is a segmented image type that is not used in Type-22 records. Also, nc:BinaryBase64Object is already specified for the image field 22.999.
REMOVED:
<xsd:choice>
  <xsd:element ref="biom:SourceImage"/>
</xsd:choice>