

Meaningful Use Stage 3 Validation Tool for Reporting Antimicrobial Resistance and Antimicrobial Use to the National HealthCare Safety Network (NHSN)

Introduction

For 2018, NHSN Antimicrobial Resistance and Antimicrobial Use reporting has been identified as a new option for public health registry reporting under Meaningful Use Stage 3. See

<https://www.federalregister.gov/articles/2015/03/30/2015-06612/2015-edition-health-information-technology-health-it-certification-criteria-2015-edition-base>. See certification criterion (§ 170.315(f)(6)). In order to qualify as certified technology an EHR or EHR Module must be capable of creating Clinical Document Architecture (CDA) documents for Antimicrobial Use and Resistance conformant to the HL7 Implementation Guide for CDA® Release 2—Level 3: Healthcare Associated Infection Reports, Release 1—US Realm—August 2013. See http://www.hl7.org/implement/standards/product_brief.cfm?product_id=20. The following CDAs must all be successfully generated.

- Antimicrobial Resistance Option (ARO) Report (Numerator) specific document template in Section 2.1.2.1 (pages 69-72);
- Antimicrobial Resistance Option (ARO) Summary Report (Denominator) specific document template in Section 2.1.1.1 (pages 54-56);
- Antimicrobial Use (AUP) Summary Report (Numerator and Denominator) specific document template in Section 2.1.1.2 (pages 56-58)

The validator uses an open source tool called Probatron as well as custom scripting. Probatron.jar is included in the installation zip file. The validator checks for three levels of conformance.

- Check for valid XML
- Check that the file validates against the cda.xsl schema
- Check that the file validates against the hai.sch Schematron

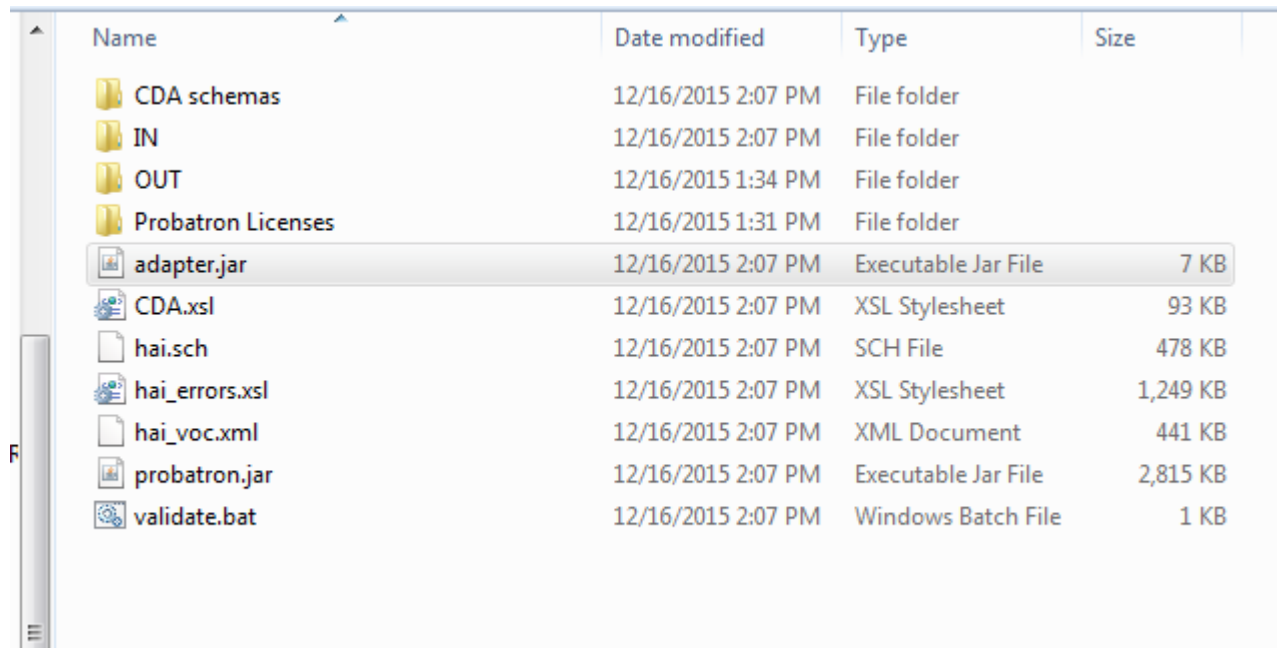
Pre-conditions for Running the Validator

- A server or workstation with a recent version of Windows.
- Rights to the server or workstation allowing a user to install, test and run software applications.
- A recent version of the Java Runtime Environment (JRE) installed on the machine. To see if Java is installed, open a terminal window and type > Java -version. You should get a response back with the current version of Java running. If not go to <http://www.oracle.com/technetwork/java/javase/downloads/index.html> and install the Java JRE.

Installation Instructions

Download the zip file from https://github.com/brhoAtCDC/HAI_Validator_4_MU3

Extract the files to a directory of your choice. Make sure you have execute rights to the directory you choose. If you open an Explorer window you should see the following:



Name	Date modified	Type	Size
CDA schemas	12/16/2015 2:07 PM	File folder	
IN	12/16/2015 2:07 PM	File folder	
OUT	12/16/2015 1:34 PM	File folder	
Probatron Licenses	12/16/2015 1:31 PM	File folder	
adapter.jar	12/16/2015 2:07 PM	Executable Jar File	7 KB
CDA.xsl	12/16/2015 2:07 PM	XSL Stylesheet	93 KB
hai.sch	12/16/2015 2:07 PM	SCH File	478 KB
hai_errors.xsl	12/16/2015 2:07 PM	XSL Stylesheet	1,249 KB
hai_voc.xml	12/16/2015 2:07 PM	XML Document	441 KB
probatron.jar	12/16/2015 2:07 PM	Executable Jar File	2,815 KB
validate.bat	12/16/2015 2:07 PM	Windows Batch File	1 KB

Running the Validator

The validator is very easy to run.

1. Put the CDA files to be tested in the "IN" directory. Note that the "IN" directory comes preloaded with a set of example CDAs to be tested, some valid and some invalid. You should delete (or move those files prior to an actual test). Place three CDAs in the IN folder. One Antimicrobial Use CDA, one Antimicrobial Resistance Numerator CDA and one Antimicrobial Resistance denominator CDA.
2. If necessary, insure these files are the correct CDAs by looking in the XML for the title and template IDs. The values should be as follows for each CDA:

a. Antimicrobial Use, Pharmacy Option (AUP) Summary Report

```
<!-- Conformant to Antimicrobial Use (AUP) Summary Report -->
<templateId root="2.16.840.1.113883.10.20.5.44"/>
```

b. Denominator for Antimicrobial Resistance Option (ARO)

```
<!-- Conformant to Antimicrobial Resistance Option (ARO) Summary Report (R1) -->
<templateId root="2.16.840.1.113883.10.20.5.46"/>
```

c. Antimicrobial Resistance Option (ARO) report

```
<!-- Conformant to the HAI Antimicrobial Resistance Option (ARO) Report -->
<templateId root="2.16.840.1.113883.10.20.5.31"/>
```

3. Run validate.bat. You can do this two ways, one by simply double clicking on the validate.bat file in Explorer. The other way is to open a terminal window and move the cursor to the directory containing the validate.bat file. Using the Explorer window is easier for most people, but the command line gives you greater flexibility.

- a. **Running validate.bat in an Explorer Window:**

By doubling clicking on validate.bat in Explorer, you will get a report generated in the "OUT" directory. The file name will be a text file called "Result" + the data and time. An example file name looks like "Result_2015_12_16_13_04_53.txt". Assuming a single instance of the validator is running on a machine, this should insure no files get overwritten.

- b. **Running validate.bat on a Command Line:**

Open a Terminal Window by clicking the start button and typing in cmd. Hit return and a terminal window should open. Change directories to where the validate.bat file is located. If you simply type validate.bat, you will get the exact same result file as using the Explorer window. However you can customize the result file name by typing in a command line parameter. So typing

Validate.bat "Acme Systems Version 4.3.2" your result file will now look like

Acme _Systems Version 4.3.2_2015_12_16_13_04_53.txt

Note that the quotes around the file name parameter are only necessary if you put spaces in the name.

Be sure to clean out any old CDAs from the "IN" folder as the validator will validate any file found there.

Interpreting the Results

The result file is a text file and if double clicked, will open in Notepad. However the output looks much better if you open the result file in **WordPad** (free with all versions of Windows) or most any good text editor. To do so in Explorer, right click on the result file and go to “Open with” and choose WordPad. The file in WordPad looks like:

```
Overall Test Result: Failed

Summary:
arobad.xml Invalid
Bad_R1Norm_AR-denom_Format-Issue.xml Invalid
Bad_R1Norm_AR-num_Vocab-Issue.xml Invalid

#####
arobad.xml

Invalid

=====

<<<CDA XML Schema with SDTC Extensions>>>

<Error : 1>
org.xml.sax.SAXParseException; cvc-enumeration-valid: Value 'zzzzz' is not
facet-valid with respect to enumeration '[XCRPT, COMP, RSON, SPRT, CAUS,
GEVL, MFST, REFR, SAS, SUBJ]'. It must be a value from the enumeration.
-----

=====

...
```

In order to pass validation, all three files must be valid. “The Overall Test Result:” will be “Passed” only if all three files pass validation.

The “Summary:” section lists each file by file name and displays the results of the test. Again in order to pass, all files must be valid. If any are not, scroll down to the section with that file name and there will be the actual errors listed out. The validator checks for three levels of conformance.

- Check for valid XML
- Check that the file validates against the cda.xsl schema
- Check that the file validates against the hai.sch Schematron

If a file fails at any level it is invalid overall.

If a testing system cannot interpret the results, please contact the NHSN CDA help desk at nhsncda@cdc.gov.

Technical Issues:

The validator uses an open source product called Probatron as well as custom scripting. If the validator fails to run, please contact the NHSN CDA help desk at nhsncda@cdc.gov