Article
Amy Brown · Sep 12, 2017 6m read

Preprocessing Support for C-CDA 2.1 Import Transformations

HealthShare HealthConnect and Information Exchange version 15.03 support import transformations from C-CDA 2.1 to SDA. You can find these transforms in your installation's `csp/xslt/SDA3` directory. For general information about import transforms, see "CDA Documents and XSL Transforms in HealthShare" in *Overview of Health Connect*.

Among the enhancements to import functionality added in connection with C-CDA 2.1 support is **the ability to preprocess your C-CDA input files prior to the transformation done for import**.

Preprocessing support can greatly simplify and reduce total processing time for transformations. Some possible use cases are:

- Missing or malformed elements
- Datestamp translation

The examples and instructions below show how to ensure that (1) certain punctuation within node values is removed before import and (2) all `effectiveTime` elements for certain encounters have `high` child nodes that are populated with timestamps. The `high` node is converted to an SDA `ToTime` node on import.

Full working examples of a preprocessor transform, a modified top-level transform, and an input file are attached to this article.

Preliminary Setup

1. Working with the administrator of your organization's domain names, obtain a subdomain name to be used for code extensions. The example used in these instructions is: `extensions.your-healthcare.org`.
2. Choose a prefix and a name for a custom mode for preprocessing. The examples used in these instructions are `mo` for prefix and `MyMODEName` for mode.

Create a Transform for Preprocessing

Create your XSL file for preprocessing as follows.

1. The stylesheet element should be of the form:

```
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0"
xmlns:hl7="urn:hl7-org:v3" xmlns:custom_mode_prefix=subdomain
exclude-result-prefixes="hl7 custom_mode_prefix">
```

   Using the examples in the Preliminary Setup section, your stylesheet element would be:

```
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0"
xmlns:hl7="urn:hl7-org:v3" xmlns:mo="http://extensions.your-
healthcare.org" exclude-result-prefixes="hl7 mo">
```

2. Include templates that replicate the top-level element and copy all attributes, elements, and text nodes. Both templates must use your custom mode.

```
<!-- Entry Point; replicate the top element -->

<xsl:template match="/hl7:ClinicalDocument" mode="mo:MyMODEname">
  <hl7:ClinicalDocument>
      <xsl:apply-templates select="@*" mode="mo:MyMODEname"/>
      <xsl:apply-templates select="node()" mode="mo:MyMODEname"/>
  </hl7:ClinicalDocument>
</xsl:template>

<!-- copy all attributes, elements, and text nodes anywhere in the document -->

<xsl:template match="node()" mode="mo:MyMODEname" priority="1">
   <xsl:copy>
      <xsl:copy-of select="@*"/>
         <xsl:apply-templates select="node()" mode="mo:MyMODEname"/>
   ?</xsl:copy>
</xsl:template>
```

3. Create a template for each distinct modification you want to make. These templates must also use your custom mode. Here we are creating two templates, one to strip certain punctuation from node values and the other to ensure that certain elements contain `low` and `high` timestamp nodes.
The first template removes question marks and exclamation points from node values.

```
<xsl:template match="text()" mode="mo:MyMODEname" priority="2">
   <xsl:value-of select="translate(., '?!', '')"/>
</xsl:template>
```

The second template searches `entry.encounter.effectiveTime` elements for missing `high` nodes. If a `low` node is present and populated and the corresponding `high` node is missing, the preprocessor copies the timestamp in the `low` node and inserts a `high` node with that timestamp value. Note that this template has the highest priority.

```
  <!-- For certain hl7:encounter/hl7:effectiveTime elements, ensure that it has
 both low and high timestamp, or neither -->
  <xsl:template match="/hl7:ClinicalDocument/hl7:component/hl7:structuredBody/h
l7:component/
       hl7:section[hl7:templateId/@root = '2.16.840.1.113883.10.20.22.2.22']/
       hl7:entry/hl7:encounter/hl7:effectiveTime" mode="mo:MyMODEname" priority
="3">

    <xsl:copy>
      <xsl:copy-of select="@*"/>
      <xsl:apply-templates select="node()" mode="mo:MyMODEname"/>

      <!--If test adds a high node where low node is present and high node is m
issing. It copies the value of the low node to the new high node. -->

      <xsl:if test="string-length(hl7:low/@value) > 0 and not(string-
length(hl7:high/@value) > 0)">
        <xsl:comment>hl7:timestamp was augmented in pre-
processing phase</xsl:comment>
        <xsl:element name="high" namespace="urn:hl7-org:v3">
```

```
          <xsl:attribute name="value">
            <xsl:value-of select="hl7:low/@value"/>
          </xsl:attribute>
        </xsl:element>
      </xsl:if>

    </xsl:copy>
  </xsl:template>
```

4. Copy the XSL file for preprocessing to `../CDA-Support-Files/Import/`.

Create a Custom Top-Level Transform

Clone and modify `CCDAv21-to-SDA.xsl` or `CCDAv21-nonXML-to-SDA.xsl` as follows:

1. The stylesheet element should include your custom namespace:

```
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
xmlns:mo="http://extensions.your-healthcare.org" xmlns:isc="http://extension-fu
nctions.intersystems.com" xmlns:hl7="urn:hl7-org:v3" xmlns:sdtc="urn:hl7-org:sd
tc" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:exsl="http://ex
slt.org/common" exclude-result-prefixes="isc hl7 sdtc xsi exsl">
```

2. Add an include statement that references the custom import file that names the XSL file to use for preprocessing:

```
<xsl:include href="CDA-Support-Files/Import/CCDAPreProcessor.xsl"/>
```

3. Replace the line

```
<xsl:variable name="input" select="/hl7:ClinicalDocument"/>
```

with the block

```
<xsl:variable name="inputRTF">


   &lt;xsl:apply-templates select="/hl7:ClinicalDocument" mode="<em>custom_mode
_prefix</em>:<em>mode</em>"/&gt;



&lt;/xsl:variable&gt;

<xsl:variable name="input" select="exsl:node-
set($inputRTF)/hl7:ClinicalDocument"/>
```

.
Using the example above, the calling code block would be:

```
<xsl:variable name="inputRTF">
```

```
&lt;xsl:apply-
templates select="/hl7:ClinicalDocument" mode="mo:MyMODEname"/&gt;



&lt;/xsl:variable&gt;

  <xsl:variable name="input" select="exsl:node-
set($inputRTF)/hl7:ClinicalDocument"/>
```

Set Up Ensemble and Import the C-CDA File

1. In your production, find the business service that imports SDA. In Information Exchange this service is named `HS.Gateway.ECR.SDA3XML.FileService` and it is part of your Edge production.
2. Set the value of the **InputXSL** additional setting of that service to the filename of your custom top-level transform. The top-level transform in the attached code examples is named `CCDAv21-to-SDA-PreProc.xsl`, so the **InputXSL** setting value would be `CCDAv21-to-SDA-PreProc.xsl`.
3. Import your C-CDA file as you normally do.

#HealthShare #XML

Source URL:https://community.intersystems.com/post/preprocessing-support-c-cda-21-import-transformations