



SEARCH

The National Consortium for Justice Information and Statistics

Transforming Hierarchical Structures into Associations

By

Scott Came

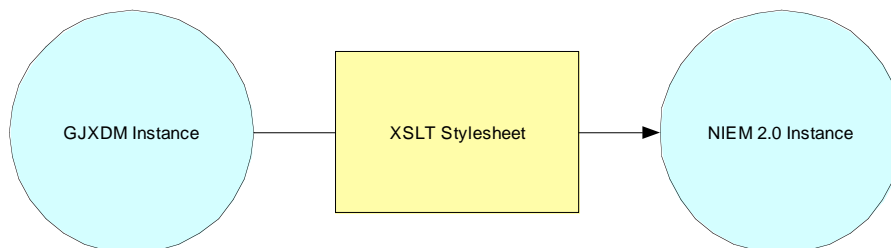
Director, Systems and Technology
and

Andrew Owen

Justice Information Systems Specialist
SEARCH

Introduction

This *Technical Brief* addresses the problem of transforming XML documents (also called “instances” or “messages”) between the Global Justice XML Data Model (GJXDM) 3.0.3 or the National Information Exchange Model (NIEM) 1.0 and NIEM 2.0.¹ This problem is relevant in any situation where business partners implemented an information exchange using GJXDM or NIEM 1.0, and wish to migrate that exchange to NIEM 2.0. Partners may wish to reuse infrastructure or exchange endpoints that they developed to participate in the original exchange, but not hold the other partners back from moving to NIEM 2.0. A practical solution in this situation is to develop transformations—generally with a technology such as the XML Stylesheet Transformation Language (XSLT²)—to translate one XML document into another.



¹ The hierarchical structures that motivate the techniques described in this paper appear in both NIEM 1.0 and GJXDM. This paper will focus on GJXDM, but the techniques described apply equally well to NIEM 1.0.

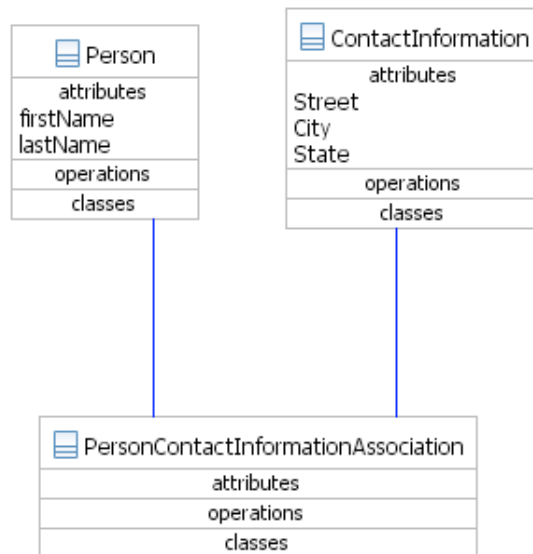
² See <http://en.wikipedia.org/wiki/Xslt> and <http://www.w3.org/TR/xslt>.

While simple in principle, the significant structural differences between GJXDM and NIEM 2.0 make the development of such transformations challenging. The release of the NIEM 2.0 included the “harmonization” of many concepts and structures in the model. One of the more significant instances of this harmonization involved breaking apart structures that were monolithic in the GJXDM, resulting in separate structures that are now related through association.

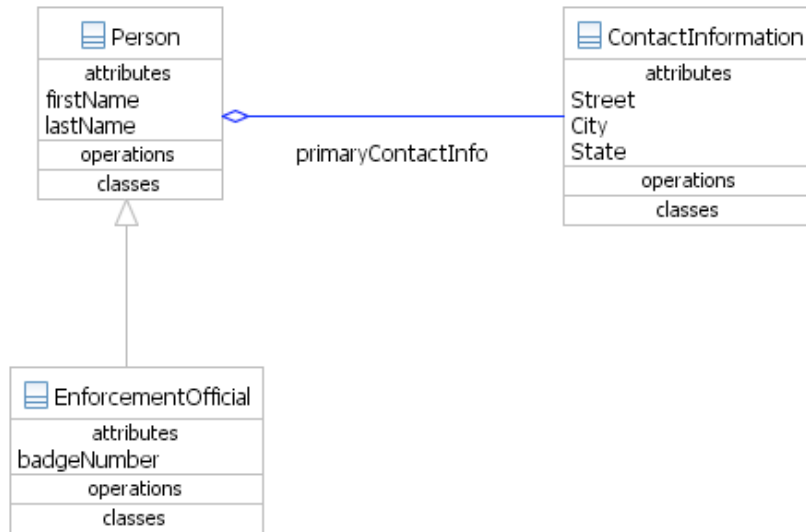
For example, in both GJXDM and NIEM 2.0, it is possible to represent the fact that a person has contact information (that is, a set of attributes that describe how to communicate with that person, such as address, telephone number, email address, etc.). However, the structure of this information in the two models is distinctly different. In GJXDM, contact information is a property of the *person* (the following diagram depicts this in UML through the use of the open diamond on the association, which indicates that the contact information is “contained within” the person):



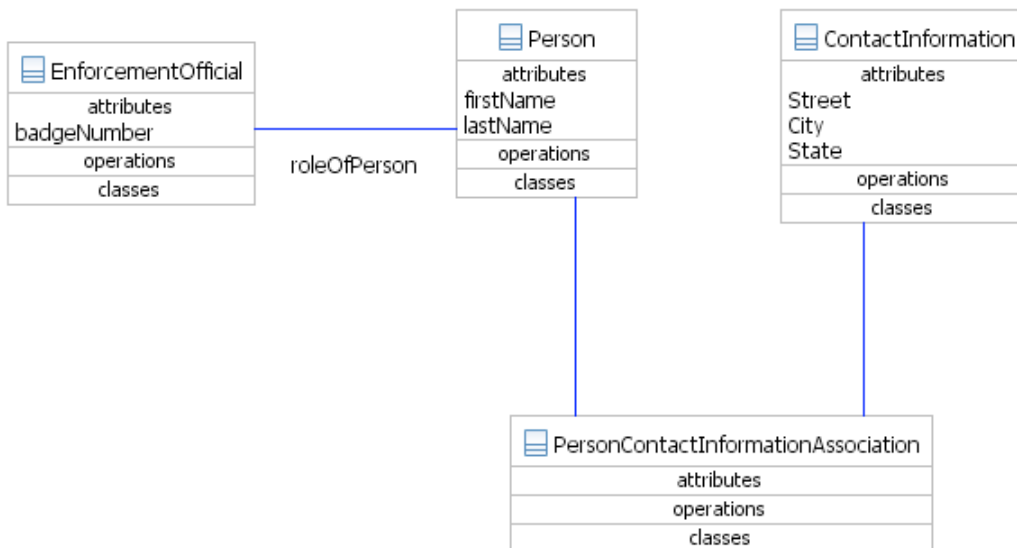
However, in NIEM 2.0, *person* and *contact* information are separate structures, connected by a third structure that associates them, as in the following UML diagram:



In addition to simple associations such as Person-Contact Information, NIEM 2.0 introduces a special kind of association called a “role.” GJXDM represents roles that people play (such as their jobs) through inheritance; that is, the role is viewed as a specialization of the person. For example, GJXDM represents the concept of a law enforcement officer as an extension of the concept of a person. The following diagram depicts this relationship, adding to the classes already introduced above:



NIEM 2.0 represents roles as separate objects, which connect to the “player” of the role through association. So, in NIEM 2.0, the concept of a law enforcement officer is its own structure (with its own attributes, such as a badge number), with an association to the person that plays that role, as in the following diagram:



Solution: XSLT Stylesheet

When faced with an information exchange involving these structures (or others like them), the developer may well choose to write an XSLT stylesheet to transform a GJXDM instance into a NIEM 2.0 instance. Writing this stylesheet would occur after the publication of the NIEM 2.0 Information Exchange Package Documentation (IEPD), because it relies on a clear understanding of the structures in the NIEM 2.0 instance. The IEPD establishes this understanding through an exchange content model, mapping artifact, and ultimately a set of XML Schemas.

The differences between GJXDM and NIEM 2.0 structures present a challenge to the development of the stylesheet. The nature of the challenge is that transforming the hierarchical structures of GJXDM into the flat, but associated structures of NIEM 2.0 requires two techniques in XSLT that, while possible, are not immediately obvious given the template-based, linear approach favored by XSLT. The two techniques are:

1. Generating unique identifiers for objects within the NIEM instance so that other objects (in particular, associations) can “point to” them.
2. Creating the association or role objects themselves from the “mixed” structures in the GJXDM instance.

Example Scenario

The following scenario will be used to illustrate these issues and techniques, and will build upon the domain concepts introduced in the previous section. The example exchange involves two Law Enforcement Officers. Each officer has a name (first and last), contact information (street address, city, state, and ZIP code), up to three birth dates (reported by different states), and a badge number.

Appendix A contains a GJXDM-conformant XML instance representing this scenario. Appendix B contains the equivalent NIEM 2.0-conformant instance. Appendix C contains a stylesheet used to transform the instance in Appendix A into the instance in Appendix B.

Stylesheet Overview

The basic outline of the stylesheet in Appendix C is as follows:

- The first nine lines contain the **xsl:stylesheet** element required by XSLT, and also set up the namespaces in the two instances. The **jj** and **gjxdm** prefixes appear in the input but not the output, so the output instance need not contain those prefixes; thus the exclude-result-prefixes attribute.
- Line 14 hints to the processor that the output of the stylesheet is XML and should be indented, if possible.
- Lines 16–24 drive the transformation. Line 17 effects a one-to-one transformation of the root of the input to the root of the output. Lines 18–22 structure the output by transforming portions of the input content.

- For most of the stylesheet, the transformation involves writing NIEM 2.0 content equivalent to the GJXDM input. The tricky parts correspond to the two techniques mentioned above; these are described in detail in the next two sections.

Generating Unique Identifiers

The “association” approach prevalent in NIEM 2.0 requires that many objects have unique identifiers, so that “association” objects can point to them. Returning to the example above, NIEM 2.0 represents a law enforcement officer as a separate object that references or “points to” a person object that plays the law enforcement officer role. The person object’s unique identifier is the essential link that makes this possible (these identifiers appear at lines 18 and 27 in the NIEM 2.0 example instance in Appendix B).

XSLT’s **generate-id()** built-in function is designed for just this kind of job. It accepts as an argument a node in the input (if there is no argument, the current context node is used) and generates an identifier that is guaranteed to be unique across all nodes in the input. This is true even if two nodes have the same name or string value.

In the example scenario, there are three types of objects referenced by associations: the person serving as a law enforcement officer, the contact information for each person, and the metadata representing the state(s) reporting the person’s birth date. There will be a one-to-one correspondence between contact information and metadata in the input and output; however, each person object in the output will correspond to a law enforcement official object in the input. This is an important consideration when determining the input node to use in generating the identifiers.

The stylesheet generates the identifier for each person object it encounters at lines 28–30. Note that it does so in the context of a `PersonName` element, and because the output `Person` object should correspond to the input `EnforcementOfficial` object, the stylesheet uses the `EnforcementOfficial` node (the parent (“..”) of the `PersonName` node) as the basis for generating the identifier. The stylesheet generates identifiers for the metadata and `ContactInformation` objects at lines 59–61 and 70–72, respectively.

Creating Association Objects

As noted above, NIEM 2.0 represents a law enforcement officer as an association between a role and a person who plays that role. The stylesheet creates the `EnforcementOfficial` role object at lines 112–125. At lines 115–117, it creates a pointer to the “role-playing” person, by generating an identifier from the same node, as at lines 28–30. Metadata objects are also handled by pointers, as at lines 33–35 and 48–50.

The case of contact information is slightly more complex, because handling it requires two passes through the input: one to write the contact information structure to the output,

and a second to associate the contact information to the person to whom it applies.³ The stylesheet accomplishes these two processing steps through use of “modes”—there are two templates matching the PrimaryContactInformation input structure (one starting at line 68 and the other at line 94), and the stylesheet invokes them sequentially as part of generating the output (lines 20 and 21). In the “association” pass, at lines 94–110, the stylesheet links the contact information structure to the appropriate person.

Conclusion

This paper has outlined XSLT techniques for transforming the “hierarchical” structures prevalent in GJXDM and NIEM 1.0 into the “flatter” and “associated” structures in NIEM 2.0. While there is no general XSLT-based approach for transforming any GJXDM instance into NIEM 2.0, these techniques can be applied in any situation where hierarchical structures need to be transformed into associations.

³ Note that this second pass through the input may be a “virtual” pass, depending on the implementation of the XSLT processor.

Appendix A: Example Instance (GJXDM 3.0.3)

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <gjxdm:gjxdm
3   xmlns:j="http://www.it.ojp.gov/jxdm/3.0.3"
4   xmlns:gjxdm="http://www.search.org/GJXDMtoNIEM/GJXDM">
5   <j:EnforcementOfficial>
6     <j:PersonName j:reportingOrganizationText="AZ">
7       <j:PersonGivenName>John</j:PersonGivenName>
8       <j:PersonSurName>Smith</j:PersonSurName>
9     </j:PersonName>
10    <j:PrimaryContactInformation>
11      <j:ContactMailingAddress>
12        <j:LocationStreet>
13          <j:StreetFullText>123 Main Street</j:StreetFullText>
14        </j:LocationStreet>
15        <j:LocationCityName>Los Angeles</j:LocationCityName>
16        <j:LocationStateName>California</j:LocationStateName>
17        <j:LocationPostalCodeID>
18          <j:ID>90210</j:ID>
19        </j:LocationPostalCodeID>
20      </j:ContactMailingAddress>
21    </j:PrimaryContactInformation>
22    <j:PersonBirthDate j:reportingOrganizationText="AZ">1980-08-15</j:PersonBirthDate>
23    <j:PersonBirthDate j:reportingOrganizationText="VA">1980-08-10</j:PersonBirthDate>
24    <j:PersonBirthDate j:reportingOrganizationText="CA">1975-08-15</j:PersonBirthDate>
25    <j:EnforcementOfficialBadgeID>
26      <j:ID>LA-9876</j:ID>
27    </j:EnforcementOfficialBadgeID>
28  </j:EnforcementOfficial>
29  <j:EnforcementOfficial>
30    <j:PersonName j:reportingOrganizationText="PA">
31      <j:PersonGivenName>Mary</j:PersonGivenName>
32      <j:PersonSurName>Simpson</j:PersonSurName>
33    </j:PersonName>
34    <j:PrimaryContactInformation>
35      <j:ContactMailingAddress>
36        <j:LocationStreet>
37          <j:StreetFullText>987 Main Street</j:StreetFullText>
38        </j:LocationStreet>
39        <j:LocationCityName>Sacramento</j:LocationCityName>
40        <j:LocationStateName>California</j:LocationStateName>
41        <j:LocationPostalCodeID>
42          <j:ID>98765</j:ID>
43        </j:LocationPostalCodeID>
44      </j:ContactMailingAddress>
```

```
45     </j:PrimaryContactInformation>
46     <j:PersonBirthDate j:reportingOrganizationText="AZ">1982-08-15</j:PersonBirthDate>
47     <j:EnforcementOfficialBadgeID>
48         <j:ID>LA-1234</j:ID>
49     </j:EnforcementOfficialBadgeID>
50 </j:EnforcementOfficial>
51 </gjxdm:gjxdm>
```

Appendix B: Example Instance (NIEM 2.0)

```
1 <niem:niem
2   xmlns:niem="http://www.search.org/GJXDMtoNIEM/NIEM"
3   xmlns:nc="http://niem.gov/niem/niem-core/2.0"
4   xmlns:s="http://niem.gov/niem/structures/2.0"
5   xmlns:j="http://niem.gov/niem/domains/jxdm/4.0"
6   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
7   xsi:schemaLocation="http://www.search.org/GJXDMtoNIEM/NIEM NIEM2.0/NIEMRootSchema.xsd">
8
9   <j:EnforcementOfficial>
10     <nc:RoleOfPersonReference s:ref="N10009"/>
11     <j:EnforcementOfficialBadgeIdentification>
12       <nc:IdentificationID>LA-9876</nc:IdentificationID>
13     </j:EnforcementOfficialBadgeIdentification>
14   </j:EnforcementOfficial>
15   <j:EnforcementOfficial>
16     <nc:RoleOfPersonReference s:ref="N10040"/>
17     <j:EnforcementOfficialBadgeIdentification>
18       <nc:IdentificationID>LA-1234</nc:IdentificationID>
19     </j:EnforcementOfficialBadgeIdentification>
20   </j:EnforcementOfficial>
21   <nc:Person s:id="N10009">
22     <nc:PersonBirthDate s:metadata="N1002E">
23       <nc:Date>1980-08-15</nc:Date>
24     </nc:PersonBirthDate>
25     <nc:PersonBirthDate s:metadata="N10032">
26       <nc:Date>1980-08-10</nc:Date>
27     </nc:PersonBirthDate>
28     <nc:PersonBirthDate s:metadata="N10036">
29       <nc:Date>1975-08-15</nc:Date>
30     </nc:PersonBirthDate>
31     <nc:PersonName s:metadata="N1000C">
32       <nc:PersonGivenName>John</nc:PersonGivenName>
33       <nc:PersonSurName>Smith</nc:PersonSurName>
34     </nc:PersonName>
35   </nc:Person>
36   <nc:Person s:id="N10040">
37     <nc:PersonBirthDate s:metadata="N10065">
38       <nc:Date>1982-08-15</nc:Date>
39     </nc:PersonBirthDate>
40     <nc:PersonName s:metadata="N10043">
41       <nc:PersonGivenName>Mary</nc:PersonGivenName>
42       <nc:PersonSurName>Simpson</nc:PersonSurName>
43     </nc:PersonName>
44   </nc:Person>
```

```

45 <nc:ContactInformation s:id="N10015">
46   <nc:ContactMailingAddress>
47     <nc:StructuredAddress>
48       <nc:LocationStreet>
49         <nc:StreetFullText>123 Main Street</nc:StreetFullText>
50       </nc:LocationStreet>
51       <nc:LocationCityName>Los Angeles</nc:LocationCityName>
52       <nc:LocationStateName>California</nc:LocationStateName>
53       <nc:LocationPostalCode/>
54     </nc:StructuredAddress>
55   </nc:ContactMailingAddress>
56 </nc:ContactInformation>
57 <nc:ContactInformation s:id="N1004C">
58   <nc:ContactMailingAddress>
59     <nc:StructuredAddress>
60       <nc:LocationStreet>
61         <nc:StreetFullText>987 Main Street</nc:StreetFullText>
62       </nc:LocationStreet>
63       <nc:LocationCityName>Sacramento</nc:LocationCityName>
64       <nc:LocationStateName>California</nc:LocationStateName>
65       <nc:LocationPostalCode/>
66     </nc:StructuredAddress>
67   </nc:ContactMailingAddress>
68 </nc:ContactInformation>
69 <nc:PersonContactInformationAssociation>
70   <nc:PersonReference s:ref="N10009"/>
71   <nc:ContactInformationReference s:ref="N10015"/>
72   <nc:ContactInformationIsPrimaryIndicator>true</nc:ContactInformationIsPrimaryIndicator>
73 </nc:PersonContactInformationAssociation>
74 <nc:PersonContactInformationAssociation>
75   <nc:PersonReference s:ref="N10040"/>
76   <nc:ContactInformationReference s:ref="N1004C"/>
77   <nc:ContactInformationIsPrimaryIndicator>true</nc:ContactInformationIsPrimaryIndicator>
78 </nc:PersonContactInformationAssociation>
79 <nc:Metadata s:id="N1000C">
80   <nc:ReportingOrganizationText>AZ</nc:ReportingOrganizationText>
81 </nc:Metadata>
82 <nc:Metadata s:id="N1002E">
83   <nc:ReportingOrganizationText>AZ</nc:ReportingOrganizationText>
84 </nc:Metadata>
85 <nc:Metadata s:id="N10032">
86   <nc:ReportingOrganizationText>VA</nc:ReportingOrganizationText>
87 </nc:Metadata>
88 <nc:Metadata s:id="N10036">
89   <nc:ReportingOrganizationText>CA</nc:ReportingOrganizationText>
90 </nc:Metadata>

```

```
91     <nc:Metadata s:id="N10043">
92         <nc:ReportingOrganizationText>PA</nc:ReportingOrganizationText>
93     </nc:Metadata>
94     <nc:Metadata s:id="N10065">
95         <nc:ReportingOrganizationText>AZ</nc:ReportingOrganizationText>
96     </nc:Metadata>
97 </niem:niem>
```

Appendix C: XSLT Stylesheet

```
1 <xsl:stylesheet version="1.0"
2   xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
3   xmlns:jj="http://www.it.ojp.gov/jxdm/3.0.3"
4   xmlns:gjxdm="http://www.search.org/GJXDMtoNIEM/GJXDM"
5   xmlns:j="http://niem.gov/niem/domains/jxdm/4.0"
6   xmlns:s="http://niem.gov/niem/structures/2.0"
7   xmlns:nc="http://niem.gov/niem/niem-core/2.0"
8   xmlns:niem="http://www.search.org/GJXDMtoNIEM/NIEM"
9   exclude-result-prefixes="jj gjxdm">
10
11   <!-- A stylesheet to transform the GJXDM instance (Appendix A) into an equivalent
12     NIEM 2.0 instance (Appendix B) -->
13
14   <xsl:output indent="yes" method="xml"/>
15
16   <xsl:template match="/gjxdm:gjxdm">
17     <niem:niem>
18       <xsl:apply-templates select="descendant::jj:EnforcementOfficial"/>
19       <xsl:apply-templates select="descendant::jj:PersonName"/>
20       <xsl:apply-templates mode="write-it" select="descendant::jj:PrimaryContactInformation"/>
21       <xsl:apply-templates mode="associate-it" select="descendant::jj:PrimaryContactInformation"/>
22       <xsl:apply-templates select="//@jj:reportingOrganizationText"/>
23     </niem:niem>
24   </xsl:template>
25
26   <xsl:template match="jj:PersonName">
27     <nc:Person>
28       <xsl:attribute name="s:id">
29         <xsl:value-of select="generate-id(..)"/>
30       </xsl:attribute>
31       <xsl:apply-templates select="../jj:PersonBirthDate"/>
32       <nc:PersonName>
33         <xsl:attribute name="s:metadata">
34           <xsl:value-of select="generate-id(@jj:reportingOrganizationText)"/>
35         </xsl:attribute>
36         <nc:PersonGivenName>
37           <xsl:value-of select="jj:PersonGivenName"/>
38         </nc:PersonGivenName>
39         <nc:PersonSurName>
40           <xsl:value-of select="jj:PersonSurName"/>
41         </nc:PersonSurName>
42       </nc:PersonName>
43     </nc:Person>
44   </xsl:template>
```

```

45
46 <xsl:template match="jj:PersonBirthDate">
47   <nc:PersonBirthDate>
48     <xsl:attribute name="s:metadata">
49       <xsl:value-of select="generate-id(@jj:reportingOrganizationText)"/>
50     </xsl:attribute>
51     <nc>Date>
52       <xsl:value-of select="."/>
53     </nc>Date>
54   </nc:PersonBirthDate>
55 </xsl:template>
56
57 <xsl:template match="@jj:reportingOrganizationText">
58   <nc:Metadata>
59     <xsl:attribute name="s:id">
60       <xsl:value-of select="generate-id()"/>
61     </xsl:attribute>
62     <nc:ReportingOrganizationText>
63       <xsl:value-of select="."/>
64     </nc:ReportingOrganizationText>
65   </nc:Metadata>
66 </xsl:template>
67
68 <xsl:template match="jj:PrimaryContactInformation" mode="write-it">
69   <nc>ContactInformation>
70     <xsl:attribute name="s:id">
71       <xsl:value-of select="generate-id()"/>
72     </xsl:attribute>
73     <nc>ContactMailingAddress>
74       <nc:StructuredAddress>
75         <nc:LocationStreet>
76           <nc:StreetFullText>
77             <xsl:value-of select="jj:ContactMailingAddress/jj:LocationStreet/jj:StreetFullText"/>
78           </nc:StreetFullText>
79         </nc:LocationStreet>
80         <nc:LocationCityName>
81           <xsl:value-of select="jj:ContactMailingAddress/jj:LocationCityName"/>
82         </nc:LocationCityName>
83         <nc:LocationStateName>
84           <xsl:value-of select="jj:ContactMailingAddress/jj:LocationStateName"/>
85         </nc:LocationStateName>
86         <nc:LocationPostalCode>
87           <xsl:value-of select="jj:ContactMailingAddress/jj:LocationPostalCode"/>
88         </nc:LocationPostalCode>
89       </nc:StructuredAddress>
90     </nc>ContactMailingAddress>

```

```

91     </nc:ContactInformation>
92 </xsl:template>
93
94 <xsl:template match="jj:PrimaryContactInformation" mode="associate-it">
95     <nc:PersonContactInformationAssociation>
96         <nc:PersonReference>
97             <xsl:attribute name="s:ref">
98                 <xsl:value-of select="generate-id(..)"/>
99             </xsl:attribute>
100         </nc:PersonReference>
101         <nc:ContactInformationReference>
102             <xsl:attribute name="s:ref">
103                 <xsl:value-of select="generate-id()"/>
104             </xsl:attribute>
105         </nc:ContactInformationReference>
106         <nc:ContactInformationIsPrimaryIndicator>
107             <xsl:text>true</xsl:text>
108         </nc:ContactInformationIsPrimaryIndicator>
109     </nc:PersonContactInformationAssociation>
110 </xsl:template>
111
112 <xsl:template match="jj:EnforcementOfficial">
113     <j:EnforcementOfficial>
114         <nc:RoleOfPersonReference>
115             <xsl:attribute name="s:ref">
116                 <xsl:value-of select="generate-id()"/>
117             </xsl:attribute>
118         </nc:RoleOfPersonReference>
119         <j:EnforcementOfficialBadgeIdentification>
120             <nc:IdentificationID>
121                 <xsl:value-of select="jj:EnforcementOfficialBadgeID/jj:ID"/>
122             </nc:IdentificationID>
123         </j:EnforcementOfficialBadgeIdentification>
124     </j:EnforcementOfficial>
125 </xsl:template>
126
127 </xsl:stylesheet>

```