Internationalization scenarios for NIEM

A good internationalization (abbreviated: i18n) scenario contains enough detail to support a technical analysis: Can the NIEM technical framework do XYZ? An example message is often essential and always helpful. Here are some i18n scenarios with example messages, along with the NTAC analysis.

Support different languages in the content of a message

Scenario: We need to specify the language of different elements in the message. For example:

```
<my:Message xml:lang="fr">
  <nc:CommentText>De l'audace, encore de l'audace, toujours de l'audace</nc:CommentText>
  <nc:CommentText xml:lang="en">That means lots of audacity</nc:CommentText>
</my:Message>
```

Analysis: This is already supported in NIEM. The message designer decides where the `xml:lang` attribute should appear in his message, then includes the `xml:lang` attribute as needed in the complex types he defines. The NIEM model provides `xml:lang` on every element derived from `nc:TextType`, which is most of the string-valued simple content.

Scenario: As above, but this time we want to specify the language of attributes. For example:

```
<my:Message xml:lang="fr">
  <nc:PersonGivenName nc:personNameCommentTextText="nom stupide">
    Bozolicious
  </nc:PersonGivenName>
</my:Message>
```

Analysis: The XML specification says that `xml:lang` applies to attributes within lexical scope, and NIEM XML can be processed in that way. However, there is no good way to support language identification on attributes in the NIEM technical framework. It is not part of the RDF interpretation, and won't be carried through a translation from NIEM XML to any other NIEM serialization. If language identification is essential, use metadata or a child element, not an attribute. For example:

```
<my:Message xml:lang="fr">
  <nc:PersonGivenName structures:metadata="md">
    Bozolicious
  </nc:PersonGivenName>
  <nc:Metadata structures:id="md">
    <nc:CommentText>nom stupide</nc:CommentText>
    <nc:CommentText xml:lang="en">Stupid name</nc:CommentText>
</my:Message>
```

Scenario: Americans write "color", the English write "colour", and we need to support both flavors of English in the content of a message.

Analysis: This is already supported in NIEM. The message producer can put whatever he likes into his message. He can even specify the flavor of English in the `xml:lang` attribute (which might help a natural language translation program), like this:

```
<nc:CommentText xml:lang="en-US">The color is red</nc:CommentText>
<nc:CommentText xml:lang="en-GB">No, the colour is blue</nc:CommentText>
```

Support non-ASCII characters in the content of a message

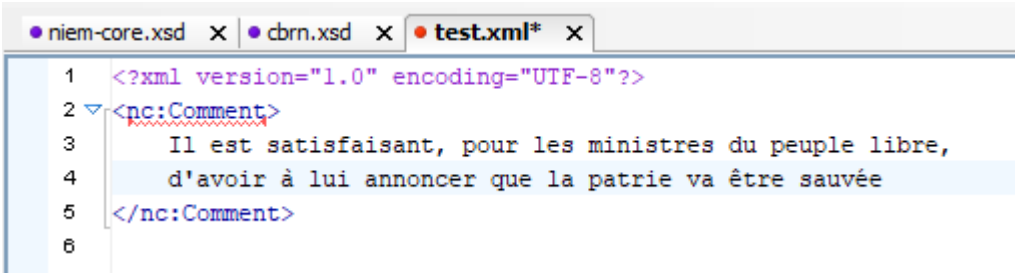
Scenario: We need to represent this quotation in a NIEM message, including the special characters:

> *Il est satisfaisant, pour les ministres du peuple libre, d'avoir à lui annoncer que la patrie va être sauvée*

Analysis: This is already supported in the NIEM technical framework. It's really not a NIEM thing at all. It's a Unicode thing, and it's up to your software and operating system to do the right thing. All you have to do is specify UTF-8 encoding in your message, like this

```
<?xml version="1.0" encoding="UTF-8"?>
```

Your tools should take it from there. For instance, in Oxygen, I see this:



The special characters appear as they should when I cut and paste. If I want to enter more of those special characters, I have to re-learn the right keyboard bindings, but that's my problem.

All of the magic happens behind the scene. If you look at the byte sequence in the file, you'll see this:

```
0000000   <   ?   x   m   l       v   e   r   s   i   o   n   =   "   1
0000020   .   0   "       e   n   c   o   d   i   n   g   =   "   U   T
0000040   F   -   8   "   ?   >  \n   <   n   c   :   C   o   m   m   e
0000060   n   t   >  \n                   I   l       e   s   t       s
0000100   a   t   i   s   f   a   i   s   a   n   t   ,       p   o   u
0000120   r       l   e   s       m   i   n   i   s   t   r   e   s
0000140   d   u       p   e   u   p   l   e       l   i   b   r   e   ,
0000160       d   '   a   v   o   i   r     303 240       l   u   i
0000200   a   n   n   o   n   c   e   r       q   u   e       l   a
0000220   p   a   t   r   i   e       v   a     303 252   t   r   e
0000240   s   a   u   v 303 251   e  \n   <   /   n   c   :   C   o   m
0000260   m   e   n   t   >  \n
```

The special characters are represented as two-byte code points. But that's nothing to do with the NIEM technical framework. Properly declare your character encoding in the message and in your software, and everything will just work.

Accommodate international practice in the NIEM model

Scenario: The NIEM model is missing data components required to represent a concept according to the practice of some nation. As a hypothetical example, suppose that a postal address in India requires a "locality name" that is distinct from city, county, or state name. We could do that with an augmentation, of course...

```
<nc:Address>
  <nc:AddressRecipientName>Dak Bhavan
  <nc:AddressCityName>New Delhi
  <nc:AddressStreet>
    <nc:StreetName>Lodhi
    <nc:StreetCategoryText>Road
  </nc:AddressStreet>
  <my:AddressAugmentation>
    <my:AddressLocalityName>Pitam Pura
  </my:AddressAugmentation>
</nc:Address>
```

[In all examples, closing tags may be omitted for brevity and clarity.]

However, data components that are essential for international practice should be part of the NIEM model, not an augmentation in an extension schema. We need to be able to do this instead:

```
<nc:Address>
  <nc:AddressRecipientName>Dak Bhavan
  <nc:AddressCityName>New Delhi
  <nc:AddressStreet>
    <nc:StreetName>Lodhi
    <nc:StreetCategoryText>Road
  </nc:AddressStreet>
  <nc:AddressLocalityName>Pitam Pura
</nc:Address>
```

Analysis: This is fully supported by the technical framework. If a domain or NIEM implementor needs additional components, they should follow the NIEM change request process and propose changes to the NBAC for review and adjudication. Gather the requirements, reach consensus on the solution, and change the next version of the schema.

Provide component definitions in other languages

Scenario: Definitions of NIEM data components are provided only in English. We need to provide additional, equivalent definitions in other languages. For example, the definition of `ag:AgriculturalProductionPlanType` is

> A data type that contains agriculture production plan related information including location, product, acreage, planting, practice and data modifications details.

We need to also supply a definition in French, perhaps:

> Type de données qui contient des informations relatives au plan de production agricole, y compris l'emplacement, le produit, la superficie, la plantation, la pratique et les détails des modifications des données.

Analysis: We don't have to change the NDR to support this in NIEM XSD. The `xs:documentation` schema element is repeatable, and has the `xml:lang` attribute. For example:

```
<xs:complexType name="AgricultureProductionPlanType">
  <xs:annotation>
    <xs:documentation xml:lang="en">
    A data type that contains agriculture production plan related information including
location, product, acreage, planting, practice and data modifications details.
    </xs:documentation>
    <xs:documentation xml:lang="fr">
    Type de données qui contient des informations relatives au plan de production agricole,
y compris l'emplacement, le produit, la superficie, la plantation, la pratique et les
détails des modifications des données.
    </xs:documentation>
  </xs:annotation>
  <xs:complexContent>
  ...
</xs:complexType>
```

(Of course, anyone doing this for real would do a better job than my online language translator. These are examples, not necessarily perfect examples.)

The NTAC and the domains decide whether their schema documents will include multiple documentation elements. In the NIEM 6 NDR, we will probably add a rule that in a reference schema document, the first instance of `xs:documentation` must have `xml:lang="en"`.

It is possible that the NTAC or some of the domains will not want the burden of choosing the languages for the alternate definitions, or of creating and maintaining those definitions. In that case, writing multiple definitions into the schema document is the wrong approach. Providing these additional definitions in a separate document is the alternative. In RDF Turtle syntax, this would look like:

```
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix ag: <http://release.niem.gov/niem/domains/agriculture/5.1/#> .
ag:AgriculturalProductionPlanType
  rdfs:CommentText
  "Type de données qui contient des informations relatives au plan de production agricole, y
compris l'emplacement, le produit, la superficie, la plantation, la pratique et les détails
des modifications des données."@fr .
```

Someone else might then come along and publish a separate definition bundle for Turkish:

```
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix ag: <http://release.niem.gov/niem/domains/agriculture/5.1/#> .
ag:AgriculturalProductionPlanType
  rdfs:CommentText
  "Konum, ürün, arazi, ekim, uygulama ve veri modifikasyonu ayrıntıları dahil olmak üzere
tarımsal üretim planı ile ilgili bilgileri içeren bir veri türü."@tr .
```

Extra language definitions could be provided in another syntax, of course -- perhaps a CSV file instead. It doesn't make a lot of difference right now, because at present we do not have tool support for extra definitions in XSD, Turtle, or any other syntax. So even if someone writes the definitions, no one will see them in the SSGT or any other tool until someone does some tool development work.

With this approach, anyone could publish component definitions in any language they chose. Users could then choose the definition bundles they want to apply.

Allow non-English component names in extension schemas

Scenario: All of the message designers and software developers for a particular message specification speak French as their first language. They want French names for the data components they create. For example:

```
<xs:element name="VéhiculeDommagesMontant" type="nc:AmountType" xml:lang="fr">
  <xs:annotation>
    <xs:documentation>
      Une somme d'argent évaluant le coût de réparation des dommages causés à un véhicule.
    </xs:documentation>
  </xs:annotation>
</xs:element>
```

instead of

```
<xs:element name="VehicleDamageAmount" type="nc:AmountType"">
  <xs:annotation>
    <xs:documentation>
      An amount of money evaluating the cost of repair of damage to a vehicle.
    </xs:documentation>
  </xs:annotation>
</xs:element>
```

Analysis: We don't encourage this. NDR rule 10-44 says "The name of any XML Schema component defined by the schema SHOULD be composed of words from the English language, using the prevalent U.S. spelling, as provided by [OED]." That rule is there to encourage interoperability and reuse. But if you really want to do it, and you're willing to ignore the warnings from Contessa, it doesn't break any conformance rule.

Provide non-English names for components in the NIEM model

Scenario: The developers in the above example want French names for the components they reuse from the NIEM core and domains. For example:

```
<xs:element name="VéhiculeDommagesMontant" type="nc:MontantArgent" xml:lang="fr">
  <xs:annotation>
    <xs:documentation>
      Une somme d'argent évaluant le coût de réparation des dommages causés à un véhicule.
    </xs:documentation>
  </xs:annotation>
</xs:element>
```

Analysis: The name of a data component forms a part of its identifier. For example, the URI for `nc:AmountType` in the 5.0 release is http://release.niem.gov/niem/niem-core/5.0/#AmountType. Allowing other names for a component would introduce additional identifiers for it. That would violate a design principle: one component for an identifier, one identifier for a component.

However, it is already possible to introduce convenient aliases for element names in NIEM JSON. By providing the appropriate `@context` object, a message designer can specify that runtime content such as:

```
"dommagesMontant": {
  "montant": 166.0,
  "codeMonnaie": "EUR"
}
```

is the equivalent of this canonical NIEM JSON:

```
"claim:VehicleDamageAmount": {
  "nc:Amount": 166.0,
  iso_4217:CurrencyCode: "EUR"
}
```

The NTAC is considering a similar alias mechanism for NIEM XML.

Summary

NIEM 5.0 already supports:

- Different languages in the message content, properly identified with `xml:lang`
- Message content containing characters not in ASCII, such as "être sauvée"
- A NIEM model with provision for international content, such as a special "locality" field in `nc:AddressType`
- Data component definitions in languages other than English
- Extension schemas with data component names in languages other than English
- In NIEM JSON, non-English aliases for data components in the NIEM model; hopefully soon also in NIEM XML

The NTAC is not aware of any other requirements for NIEM internationalization. If there are additional scenarios and examples that we have not considered, please send them forward and we will cheerfully provide our analysis of what can be done.

Author: Dr. Scott Renner | MITRE | sar@mitre.org | 28 June 2022