XSLT Ingest Example: Accumulator Classes

The Accumulator Classes

Special attention must be given to the **foaf:Person** and the **foaf:Organiztion** classes. As time goes by and data sets are ingested into VIVO, instances of people and organizations will accumulate at a dramatic rate. We want there to be only one URI for an instance of a given person (or organization). This means that we will need to check for pre-existing people and organizations to ensure that we don't accidently create duplicates as we construct the ingest RDF. To this end we will employ two XML files **Per0.xml** and **Org0.xml** that contain the currently known people and organizations in our VIVO instance. In practice these files are created by SPARQL queries (See Appendix E). For purposes of this example, we have prepared greatly reduced sample files. Figures 3 and 4 illustrate the entries in these files. For **Per0.xml** we include name parts, label, netid and a URI. A netid is a unique string assigned to each person at Cornell to serve as a public identifier. Any uniquely assigned string by another name could be used instead. For **Org0.xml** we include the name of the organization and the assigned URI.

These two files will be fed into the XSLTs to supply sets of person and organization objects to match. The nature of the matching methods used will be described in significant detail later. Note that our process must also create RDF for any new persons or organizations that we encounter during the ingest process. This accumulator RDF must be asserted before the next round of ingest so that duplicates are not generated by other unrelated ingest processes that use similar methodologies.

Notice that the second **<person>** element has no netid. This situation can arise when publication data which lists co-authors from other institutions, who have no netid, is ingested. It can also arise when a person's netid is simply unknown or forgotten at the time **foaf:Person** data is being entered.

```
<ExtantPersons>
   <person>
      <uri>http://vivo.cornell.edu/individual/ask317</uri>
      <label>Killian, Andrea S</label>
      <lname>Killian</lname>
      <fname>Andrea</fname>
      <mname>S</mname>
      <netid>ask317</netid>
   </person>
   <person>
      <uri>http://vivo.cornell.edu/individual/dag065</uri>
      <label>Green, David Augustus</label>
      <lname>Green</lname>
      <fname>David</fname>
      <mname>Augustus</mname>
      <netid></netid>
   </person>
```

Per0.xml Fragment - Figure 3

It is expected that **Per0.xml** will be maintained for a given VIVO so that, ignoring case and whitespace differences:

- Duplicate **<person>** records are removed.
- A given netid will not be found in more than one **<person>** entry with distinct URIs.
- Among **<person>** entries with no netid, a given set of name parts will not be found in two or more **<person>** entries with distinct URIs.

```
<ExtantOrgs>
   <org>
      <uri>http://vivo.cornell.edu/individual/individual181</uri>
      <name>Cornell University</name>
   </org>
   <org>
      <uri>http://vivo.cornell.edu/individual/AI-0203F0900BA00000ED7</uri>
      <name>Pennsylvania State University</name>
   </org>
```

Org0.xml Fragment - Figure 4