Application RDF Files

When VIVO is started for the first time, its empty triple stores ("triple sources") are initialized with data from RDF files deployed to the rdf subdirectory of the VIVO home directory. These RDF files are organized into a number of subdirectories in order to separate different types of data and to identify which parts of the data will be updated and maintained via edits to the corresponding triple store and which will be modified by subsequent changes to these RDF files.

ABox, TBox and Configuration Data

The top-level subdirectories in the rdf directory can be organized into three general categories:

1. Data about the structure of ontologies (the “terminological box” or TBox)
    a. Example: foaf:Person is a class and it is a subclass of foaf:Agent.
    b. Directory:
        i. `/tbox`

2. Data about specific named “individuals” and their relationships (the “assertion box” or ABox)
    a. Example: Botswana is a country and it is located on the continent of Africa.
    b. Directory:
        i. `/abox`

3. Data for configuring the behavior of the Vitro application
    a. Examples:
        i. There is a class group called “People” and it has a rank of 1 for sorting in a list of class groups.
        ii. The SPARQL Update API is restricted to the root user.
        iii. The “relatedBy” property, when the object is of class vivo:Authorship, should be rendered on a profile page with the heading “selected publications.”
        iv. A custom form should be used when editing a certain configuration item.
        v. display:NavigationElement is a class used in the configuration data.
    b. Directories:
        i. `/applicationMetadata`
        ii. `/auth`
        iii. `/display`
        iv. `/displayDisplay`
        v. `/displayTbox`

Why keep these separate? In a typical VIVO, the data about specific individuals (the ABox) is much larger than the ontology structure data (the TBox) or the data used for application configuration. The latter two are accessed repeatedly and benefit from being cached in memory. By keeping these areas of concerns separate it is easy to load the ontology definitions and the configuration data into memory without performing a complicated process of extracting them from the rest of the data in the triple store.

Content and Configuration Triple Sources

VIVO uses two triple sources as configured in applicationSetup.n3 in the VIVO home directory. Data from these directories is stored in the content triple source:

```
/abox
/tbox
/applicationMetadata
```

Data from these directories is stored in the configuration triple source:

```
/auth
/display
/displayDisplay
/displayTbox
```

Firsttime, Everytime and Filegraph

Within the directories that delineate the above categories of RDF data, there are further divisions that specify where the respective data should be stored and updated in the future.

Firsttime

RDF files in "firsttime" directories are loaded when the triple store is initialized or upgraded: that is, the very first time VIVO is started with an empty triple store, or the first time a new version of VIVO is started with an existing triple store if the new version of VIVO contains an upgraded version of the ontology.

"Firsttime" data is expected to be modifiable by end users or system administrators through VIVO's Web interface -- either using interactive editing forms or by loading or removing RDF data via the Site Admin page. Firsttime files are the standard way of distributing the parts of ontologies that do not have semantic meaning (do not affect the inferences made by reasoners). Prior to version 1.10, these annotations also included the human-readable labels applied to classes and properties.

First time a new VIVO is started

When a new VIVO is started, each RDF file in each firsttime directory is loaded in its entirety into one of the application's triple stores. The data are stored in graphs that can be written to by VIVO's GUI editing interface or modified by the Add/Remove RDF feature.

First time a new version of VIVO is started with an upgraded ontology

When a new version of VIVO is started with a new version of the ontology, the upgrade process compares the previous version of the "firsttime" files against the current state of the triple store. For a given subject and predicate, if the value in the triple store still matches the value from the previous version of firsttime -- that is, end users have not made a change to this data element -- then any changes found in the current version of firsttime will be propagated to the triple store.

Example: how VIVO applies RDF updates when changes have been made using the user interface

1. Version 0.1 rdf/tbox/firsttime/vitroAnnotations.n3 contains the triples 'foaf:Person vitro:displayRankAnnot "-1"^^xsd:int and 'foaf:Organization rdfs: label vitro:displayRankAnnot "-1"^^xsd:int'.
2. VIVO is installed with a new empty triple store
3. Triple from step 1 is loaded into the triple store.
4. VIVO is restarted any number of times; the initialTBoxAnnotations.n3 file is no longer consulted.
5. A user edits the display rank in the VIVO UI for the foaf:Person class, changing it to "9".
6. Version 0.2 is installed, where the vitroAnnotations.n3 file now contains the triples 'foaf:Person vitro:displayRankAnnot "5"^^xsd:int and 'foaf: Organization vitro:displayRankAnnot "2"^^xsd:int'. The new code installation continues to use the existing triple store with all of its data intact.
7. VIVO is restarted with the new version.
8. VIVO's upgrade process consults the current and previous vitroAnnotations.n3 files and discovers that the foaf:Person class has a new display rank.
9. It checks the triple store and sees that "9" differs from the previous value of "-1" -- that is, someone has edited it.
10. It keeps the value that someone has entered and ignores the new value of "5" found in the file.
11. The upgrade process discovers that foaf:Organization has a new display rank.
12. It checks the triple store and sees that the display rank is "-1" and matches the value from the previous version of vitroAnnotations.n3 -- that is, no one has edited it.
13. It removes the value "-1" from the triple store and adds the new value of "2".

Everytime

"Everytime" files are used for configuration data RDF files that are loaded each time VIVO starts and whose statements are added to a temporary in-memory RDF model. It is not possible to identify which statement came from which file. Data in everytime files are not intended to be editable by end users using VIVO's GUI interface. Any further updates to the data are to be made on the filesystem, followed by a restart of VIVO.

Filegraph

"Filegraph" files are loaded each time VIVO starts, and the contents of each file are loaded into a separate graph in the triple store. Thus, it is possible to query the triple store and discover from which file a given statement was loaded -- or, for example, to output all of the statements that came from a given ontology. Triples in the filegraph files are not deletable by end users using VIVO's GUI interface. Any further updates to the data that require removal of the original triples are to be made on the filesystem, followed by a restart of VIVO. Files in the filegraph directory are the standard method of distributing all parts of existing ontologies that have semantic meaning (affect the inferences made by reasoners). For example, the class hierarchy of the VIVO ontology is not intended to be modified by end users because it would make their data incompatible with that of other VIVO users.

When a filegraph file is modified and VIVO is restarted, a background recomputation of all of VIVO's inferred triples will be performed. This is done in order to take into account the logical consequences of any new ontology axioms that might have been added via the modification. (Note that if you shut down VIVO while this background process is running, it will not automatically resume but may be started again manually via the "Recompute inferences" link on the Site Admin page.)

Example

When VIVO is started, the contents of the file at rdf/tbox/filegraph/vivo.owl are compared to the contents of the graph http://vitro.mannlib.cornell.edu /filegraph/tbox/vivo.owl in the triple store. (Specifically, it is tested to see if the two graphs are isomorphic or have the same shape.)

If the contents differ, the graph in the triple store is emptied and the statements from the file are loaded into it. After a reload, a complete recomputation of VIVO's inferences is triggered.