Upgrading VIVO from release 1.5 to release 1.8.1

Author: Jim Blake
Date: 04-Nov-2015 11:57
URL: https://wiki.duraspace.org/display/VIVO/Upgrading+VIVO+from+release+1.5+to+release+1.8.1

Table of Contents

1 Introduction 4
1.1 Why to upgrade from release 1.5 to 1.8 4
1.2 What you should do first 4
2 The process 5
2.1 Enable the migration code 5
2.2 Adjust your three-tier build 5
2.3 Split your properties files 5
2.4 Run the build script 5
2.5 Run the RDB migration tool 5
2.6 Create applicationSetup.n3 6
2.7 Start VIVO 6
2.8 Wait 6
3 What you should see 7

VIVO 1.6 introduced substantial changes in the ontology – VIVO 1.8 makes it easier for sites that are still at VIVO 1.5 to make the change.

1 Introduction

1.1 Why to upgrade from release 1.5 to 1.8

The transition from release 1.5 to 1.6 was a major one. VIVO 1.6 included migration code that restructured existing data to match the new ontology. This restructuring meant that VIVO also had to recompute all of the inferences in the data and rebuild the search index. The process was time-comsuming and resource-intensive.

The good news is that VIVO 1.8 can make this transition more quickly. Improvements have been made in the migration code, the inferencing code, and the search index builder.

Sites will find it much easier to upgrade from release 1.5 to release 1.8 in a single operation.

1.2 What you should do first

Read the individual upgrade documents.

These instructions should be sufficient for a standard VIVO installation. You may need to take additional actions if your site has modified the VIVO code, Review the "Noteworthy Changes" section in each of these documents. They may help you to keep your local modifications working.

Upgrade instructions for VIVO release 1.6.2

Upgrading VIVO to release 1.8.1

Upgrading VIVO to release 1.8.1

2 The process

2.1 Enable the migration code

The standard release of VIVO will not perform a multi-step migration. You must remove the comment indicators from two lines in the file `productMods/WEB-INF/resources/startup_listeners.txt`, in your VIVO distribution.

startup_listeners.txt as distributed

```
# Invokes process to perform updates to align with ontology changes if needed -->
# edu.cornell.mannlib.vitro.webapp.migration.rel16.Release16Migrator
# edu.cornell.mannlib.vitro.webapp.migration.rel17.Release17Migrator
edu.cornell.mannlib.vitro.webapp.migration.rel18.Release18Migrator
```

startup_listeners.txt, after enabling

```
# Invokes process to perform updates to align with ontology changes if needed -->
edu.cornell.mannlib.vitro.webapp.migration.rel16.Release16Migrator
edu.cornell.mannlib.vitro.webapp.migration.rel17.Release17Migrator
edu.cornell.mannlib.vitro.webapp.migration.rel18.Release18Migrator
```

2.2 Adjust your three-tier build

If your site uses a three-tier build, see "Auto-loaded RDF files move to the Home directory" in Upgrade instructions for VIVO release 1.6.2

2.3 Split your properties files

See "Separate your existing deploy.properties file into two files" in Upgrade instructions for VIVO release 1.6.2

2.4 Run the build script

Run this as you usually would.

2.5 Run the RDB migration tool

The section entitled "Run the RDB migration tool" in Upgrading VIVO to release 1.8.1 contains a full description of this tool and its options.

⚠ The bug that is described in that section has been fixed.

2.6 Create applicationSetup.n3

For most sites, simply go to the VIVO home directory and copy `config/example.applicationSetup.n3` to `config/applicationSetup.n3`

If you are using an alternative triple store, find more information in the VIVO 1.8 Installation Instructions, under "Using a Different Data Store".

2.7 Start VIVO

Start tomcat as you usually would.

2.8 Wait

The time required will depend on your hardware, and on the size and structure of your data. In our tests, migration of a small VIVO installation (25,000 individuals) took 25 minutes. Migration of a very large installation (1,500,000 individuals) took 26 hours.

These test migrations were performed on a MacBook Pro with 8 gigabytes of memory and a solid state drive . The `setenv.sh` file in Tomcat was modified to allocate 6 gigabytes of memory to Tomcat.

3 What you should see

When Tomcat is started, you can monitor the progress of the migration by watching the contents of `vivo.all.log`. The log will contain a variety of messages, but these are the most prominent:

Messages from	`FileGraphSetup`
Message content	The ontology files have changed dramatically from release 1.5 to 1.8. These messages describe the additions, deletions and changes.
Examples	Updating /Users/jeb228/Testing/instances/florida_migration/vivo_home/rdf/abox/filegraph/academicDegree.rdf because graphs are not isomorphic dbModel: 358 ; fileModel: 454
Time consumed (large site)	2 minutes
Fraction of total time	< 1%

Messages from	`KnowledgeBaseUpdater`
Message content	This is the actual migration code, adding and removing triples from the data store, according to the changes in the ontology.

Examples	`Performing any necessary data migration`
	
	`Added 2415 statements using the SPARQL construct query from file additions/AdviseeRole.sparql`
	
	...
	
	`Removed 12578 statements using the SPARQL construct query from file post/awardedDegreeDel.sparql`
	
	`Finished checking knowledge base in 38868 seconds`

Time consumed (large site)	11 hours
Fraction of total time	40%

Messages from	`RDFServiceJena`
Message content	Interspersed among the messages from `KnowledgeBaseUpdater`, you will likely see warnings about blank nodes being deleted. In daily operation, this would likely indicate a problem, but it is expected as part of the data migration.
Examples	`Deleting single triple with blank node: <ModelCom { 8adc9011c10bdaf219eb25e68baf942c @http://www.w3.org/2002/07/owl# onProperty http://vivoweb.org/ontology/core#authorInAuthorship} This likely indicates a problem; excessive data may be deleted.`

Messages from	`ABoxRecomputer`
Message content	Because of the changes in the ontology, all inference triples must be recomputed.

Examples	`Recomputing ABox inferences.`
	`Finding individuals in ABox.`
	`Recomputing inferences for 2021509 individuals`
	`Still recomputing inferences (1000/2021509 individuals)`
	`21 ms per individual`
	`...`
	`Still recomputing inferences (2021000/2021509 individuals)`
	`23 ms per individual`
	`Finished recomputing inferences`

Time consumed (large site)	13 hours
Fraction of total time	50%

Messages from	`IndexHistory`
Message content	As a final step, the Solr search index must be rebuilt.
Examples	`STARTUP, 3/11/15 4:59 PM, []`

	START_REBUILD, 3/12/15 5:02 PM, [documentsBefore=0, documentsAfter=0] START_URIS, 3/12/15 5:09 PM, [excluded=0, deleted=0, updated=0, remaining=745168, total=745168] PROGRESS, 3/12/15 5:09 PM, [excluded=267, deleted=0, updated=500, remaining=744401, total=745168] PROGRESS, 3/12/15 5:09 PM, [excluded=637, deleted=0, updated=1000, remaining=743531, total=745168] ... PROGRESS, 3/12/15 7:20 PM, [excluded=344432, deleted=0, updated=400000, remaining=736, total=745168] STOP_URIS, 3/12/15 7:20 PM, [excluded=344835, deleted=0, updated=400333, remaining=0, total=745168] STOP_REBUILD, 3/12/15 7:20 PM, [documentsBefore=0, documentsAfter=400333]
Time consumed (large site)	2 hours
Fraction of total time	10%

When the STOP_REBUILD message appears from IndexHistory, the migration is complete.