EDIT - bug #7818

TaxonRelationshipsDTOTest leaves test environment in unpredictable state

10/11/2018 04:04 PM - Andreas Kohlbecker

Status:	Feedback	**Start date:**	
Priority:	Priority14	**Due date:**	
Assignee:	Andreas Müller	**% Done:**	0%
Category:	cdmlib	**Estimated time:**	0:00 hour
Target version:	Release 5.4		
Severity:	normal	**Found in Version:**	

Description

as stated in #7648#note-14 the TaxonRelationshipsDTOTest causes problems for other tests.

e.g.: Running the following tests in sequence

```
mvn  -Dtest=eu.etaxonomy.cdm.api.service.dto.TaxonRelationshipsDTOTest,eu.etaxonomy.cdm.api.service.taxonGraph.TaxonGraphHibernateListenerTest test
```

will end up with TaxonGraphHibernateListenerTest to fail due to:

```
Tests in error: 
  testChangeSpecificEpithet_of_InfraSpecific(eu.etaxonomy.cdm.api.service.taxonGraph.TaxonGraphHibernateListenerTest): object references an unsaved transient instance - save the transient instance before flushing: eu.etaxonomy.cdm.model.taxon.TaxonRelationshipType
  testChangeNomRef(eu.etaxonomy.cdm.api.service.taxonGraph.TaxonGraphHibernateListenerTest): object references an unsaved transient instance - save the transient instance before flushing: eu.etaxonomy.cdm.model.taxon.TaxonRelationshipType
  testChangeSpecificEpithet_of_Species(eu.etaxonomy.cdm.api.service.taxonGraph.TaxonGraphHibernateListenerTest): object references an unsaved transient instance - save the transient instance before flushing: eu.etaxonomy.cdm.model.taxon.TaxonRelationshipType
  testChangeRank(eu.etaxonomy.cdm.api.service.taxonGraph.TaxonGraphHibernateListenerTest): object references an unsaved transient instance - save the transient instance before flushing: eu.etaxonomy.cdm.model.taxon.TaxonRelationshipType
  testNewTaxonName(eu.etaxonomy.cdm.api.service.taxonGraph.TaxonGraphHibernateListenerTest): object references an unsaved transient instance - save the transient instance before flushing: eu.etaxonomy.cdm.model.taxon.TaxonRelationshipType
  testChangeGenus(eu.etaxonomy.cdm.api.service.taxonGraph.TaxonGraphHibernateListenerTest): object references an unsaved transient instance - save the transient instance before flushing: eu.etaxonomy.cdm.model.taxon.TaxonRelationshipType
  testNewGenusName(eu.etaxonomy.cdm.api.service.taxonGraph.TaxonGraphHibernateListenerTest): object references an unsaved transient instance - save the transient instance before flushing: eu.etaxonomy.cdm.model.taxon.TaxonRelationshipType
```

This is clearly caused by the TaxonRelationshipsDTOTest which run in the up started application context of the previously run CdmTransactionalIntegrationTests but not cleaning up the context afterwards.

The following changes to the TaxonRelationshipsDTOTest solve the problem:

```
public class TaxonRelationshipsDTOTest extends CdmTransactionalIntegrationTest 

//    /**
//     * @throws java.lang.Exception
//     */
//    @BeforeClass
//    public static void setUpBeforeClass() throws Exception {
//        if (Language.DEFAULT() == null){
//            new DefaultTermInitializer().initialize();
//        }
//    }
```

running the DefaultTermInitializer().initialize() outside of the CdmIntegrationTest-Context seems to have cause the side effect.

The problem becomes only relevant if the TaxonRelationshipsDTOTest is run as first test, before others. **For test like this one we should implement some means to reset the terms to the initial virgin state.** <<<< TODO

History

#1 - 10/11/2018 04:07 PM - Andreas Kohlbecker

- Description updated

#2 - 10/11/2018 04:08 PM - Andreas Kohlbecker

- Assignee changed from Andreas Müller to Andreas Kohlbecker

#3 - 10/11/2018 06:30 PM - Andreas Kohlbecker

- Description updated

- Status changed from New to Feedback

- Assignee changed from Andreas Kohlbecker to Andreas Müller

#4 - 10/11/2018 09:39 PM - Andreas Müller

- Assignee changed from Andreas Müller to Andreas Kohlbecker

There are multiple possibilities.
The simplest is to add a logger saying that this test should not run as first which is usually not the case in the suite but might be the case when debugging. This way one may now that term loading is the reason for unexpected behavior.

Also the TermLoader has an unloadAllTerms method. We could call this in the AfterClass method, but ONLY if the terms were loaded before in BeforeClass.

A further solution will be to load terms before each test. Once we reduce the number of base terms a lot (e.g. TDWG Areas etc. will be removed) term loading will not take much time and it will be acceptable to load them at least before each class.
Maybe even now new termloading and DB cleaning in BeforeClass might be a solution as it happens only per class and not per test. This way testing becomes predictable independent on the order of the test classes in the test suite.

#5 - 10/12/2018 01:58 PM - Andreas Kohlbecker

> Also the TermLoader has an unloadAllTerms method. We could call this in the AfterClass method, but ONLY if the terms were loaded before in BeforeClass.

To me this looks like a preferable solution, we would need static access to the Termloader in order to accomplish that. But how can this be done without application context. This test is a pure Unittest.

#6 - 10/23/2018 01:51 PM - Andreas Kohlbecker

- Assignee changed from Andreas Kohlbecker to Andreas Müller

#7 - 10/24/2018 05:37 PM - Andreas Müller

- Priority changed from Highest to Priority14