EGLoad Design Document

Author: M Walker

Created: July 15, 2005

Last Modified: November 3, 2005 14:07

1 Purpose of Document

This document will describe the design of the Entrez Gene load for mouse genes. Human and rat genes are loaded by another product called the EntrezGeneLoad, which was the original product that loaded all mouse, human and rat genes. This new product, the EGLoad, re implemented the loading of mouse genes to utilize the new Java framework from lib_java_core for bucketizing data (see org.jax.mgi.shr.bucketizer). All of the details of the bucketizer process is covered in the design document for that Java package. This document will assume the reader is familiar with this framework and will cover aspects only particular to loading Entrez Gene mouse genes. This product is also meant to integrate with the existing EntrezGeneLoad product for human and rat. More specifically, it depends on EntrezGeneLoad to load raw provider input files into the RADAR database. It is assumed that the reader is also familiar with EntrezGeneLoad and any details regarding this product can be accessed in the design document for that product. Furthermore, this product implements the Data Load Architecture guidelines set forth in the DLA Standards document and it is assumed the reader is familiar with these standards.

2 Introduction

This load is responsible for comparing EntrezGene genes and MGI genes in bulk using the bucketizer process. This process groups the genes that are associated with each other (meeting the match criteria set forth in the bucketizer) into separate clusters. Any cluster where there is one Entrez Gene and one MGI marker and both these genes belong on the same chromosome is considered to be represent a match between the genes without any ambiguity. When this case is found, various process steps are performed which are discussed in detail in section 5. The other possible clusters are zero to one (MGI gene with no matching Entrez gene), one to zero (Entrez gene with no matching MGI gene), one to many (Entrez gene associated with more than one MGI genes), many to one (MGI gene associated to multiple Entrez genes), many to many (many MGI genes equating to many Entrez genes). These clusters are considered to be in conflict and they get reported only. See section 6 for more details in processing these clusters.

The bucketizer process uses data represented as **SVASets** to do its internal processing. **SVASets** are part of the lib_java_core product (see org.jax.mgi.shr.sva). They are basically named attributes with values represented as sets. This type of data for the EGLoad is encapsulated in the **EntrezGeneBuckitizable** object (see section 3). The **SVASet** definition is as follows:

- GENBANK - a set of genbank sequences associated with the gene
- MGIID - a set of MGI ids associated with the gene. There should only be zero or one for an Entrez gene and one or more for an MGI gene (primary and secondary MGI ids).

- XM - XM sequences from RefSeq associated with the gene.
- XR - XR sequences from RefSeq associated with the gene.

3 Class Overview

The **EntrezGeneLoader** is an extension of **DLALoader** from the lib_java_dla product (see org.jax.mgi.shr.dla.loader). It is responsible for calling the **EntrezGeneBucketizer**.

The **EntrezGeneBucketizer** is an extension of **AbstractBucketizer** from org.jax.mgi.shr.bucketizer. It implements the various abstract methods required by **AbstractBucketizer** for processing the results of the bucketizer process. This entails writing out the data to files where each file represents a bucket from the bucketizer process.

The **EntrezGeneBucketizable** is an extension of **SimpleBucketizer** from org.jax.mgi.shr.bucketizer. The **EntrezGeneBucketizer** is given two sets of **EntrezGeneBucketizables** and it bucketizes them accordingly. Application objects such as **EntrezGene** and **MGIMarker** extend this class so that they can be bucketized by the **EntrezGeneBucketizer**. The abstract methods implemented by the **EntrezGeneBucketizer** are called and provided with parameter objects of type **EntrezGeneBucketizable**. These objects can be cast within the process methods to the application superclass objects (**EntrezGene** or **MGIMarker**) for performing application level processing.

The **EntrezGeneQuery** class is responsible for obtaining the Entrez data from the RADAR database and creating **EntrezGene** application objects. These application objects are subsequently processed by the **EntrezGeneBucketizer** as objects of type **EntrezGeneBucketizable** (the base class). The **EntrezGeneLoader** instantiates and executes the **EntrezGeneQuery** and passes the results on to the **EntrezGeneBucketizer**.

The **MGIMarkerQuery** class is responsible for obtaining the MGI marker data from the MGD database and creating **MGIMarker** application objects. These application objects are subsequently processed by the **EntrezGeneBucketizer** as objects of type **EntrezGeneBucketizable** (the base class). The **EntrezGeneLoader** instantiates and executes the **MGIMarkerQuery** and passes the results on to the **EntrezGeneBucketizer**.

The **EntrezGene** is an inner class of **EntrezGeneQuery**. It is an extension of **EntrezGeneBucketizable** and therefore can be processed by the **EntrezGeneBucketizer**. Objects of this type are created by **EntrezGeneQuery**.

The **MGIMarker** is an inner class of MGIMarkerQuery. It is an extension of **EntrezGeneBucketizable** and therefore can be processed by the **EntrezGeneBucketizer**. Objects of this type are created by **MGIMarkerQuery**.

The **EntrezGeneCfg** class is an extension of **Configurator** from the lib_java_core product (see org.jax.mgi.shr.config)**.** It is responsible for reading the configuration file(s) along with additional command line arguments and any Java properties in order to configure the **EntrezGeneBucketizable** and **EntrezGeneLoader** classes.

The **AssocAccidLookup** class is an extension of **FullCachedLookup** from lib_java_core (see org.jax.mgi.shr.cache). It is used by the **MGIMarkerQuery** class to find associated sequence accession ids for a given marker. The associated sequence data becomes an attribute of the **MGIMarker** class.

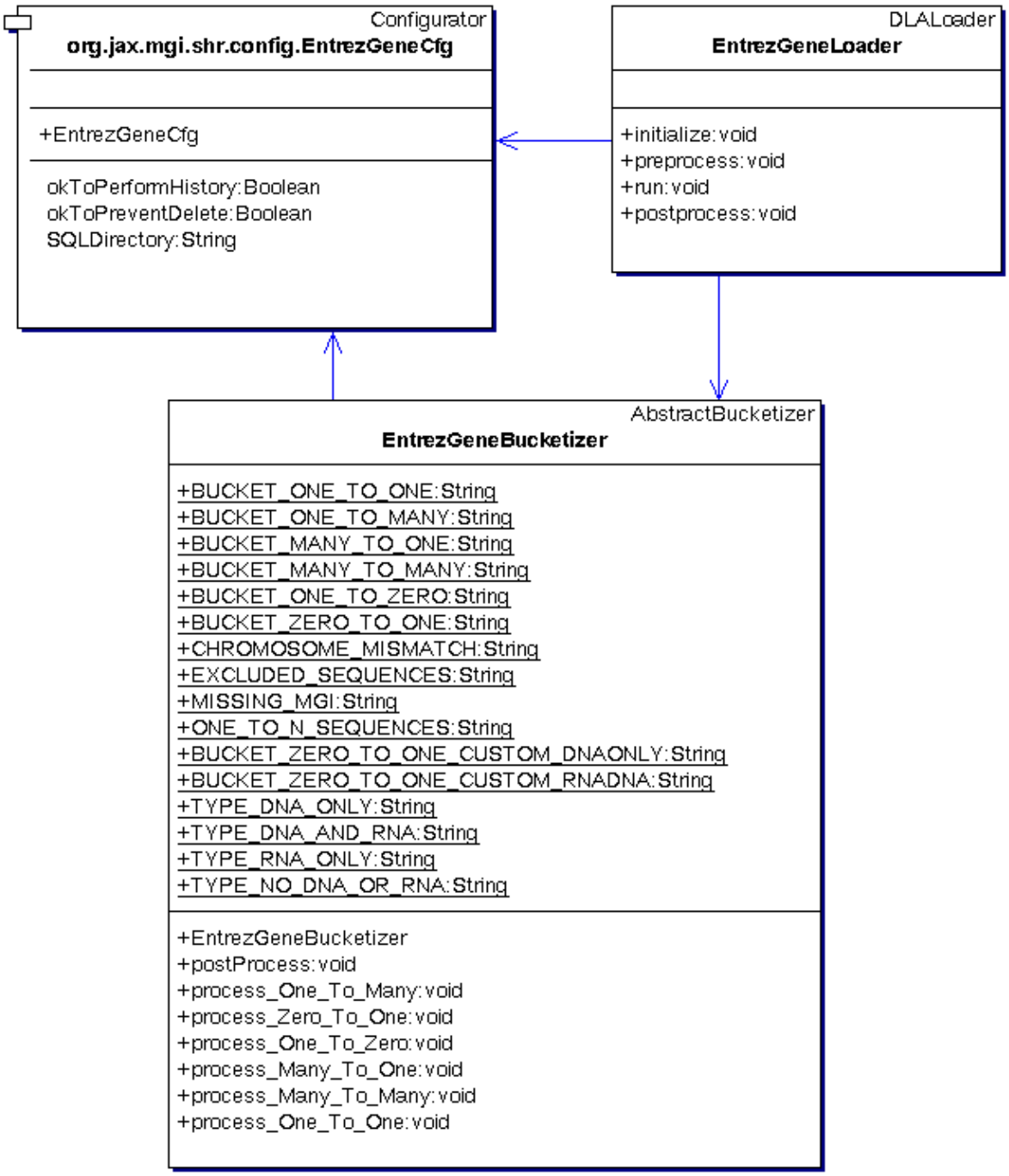
The **EntrezGeneHistory** class is an extension of **FullCachedLookup** from lib_java_core (see org.jax.mgi.shr.cache). It is used to cache previous Entrez gene/MGI marker associations. It is created and initialized by the **EntrezGeneLoader** before it does a delete of the previous data (a step required as part of its delete and reload strategy). This class is subsequently passed to the **EntrezGeneBucketizer** so that it can be used in reporting.

The **ProblemClonesLookup** is an extension of **FullCachedLookup** from lib_java_core (see org.jax.mgi.shr.cache). It is used to cache sequence accession records which are associated with clones that have been identified (via probe notes) to contain problem artifacts which prevent it from being associated to MGI markers. The **EntrezGeneBucketizer** uses this class before associating sequence data to a marker to assure that it is not already associated with a ”problem clone”.

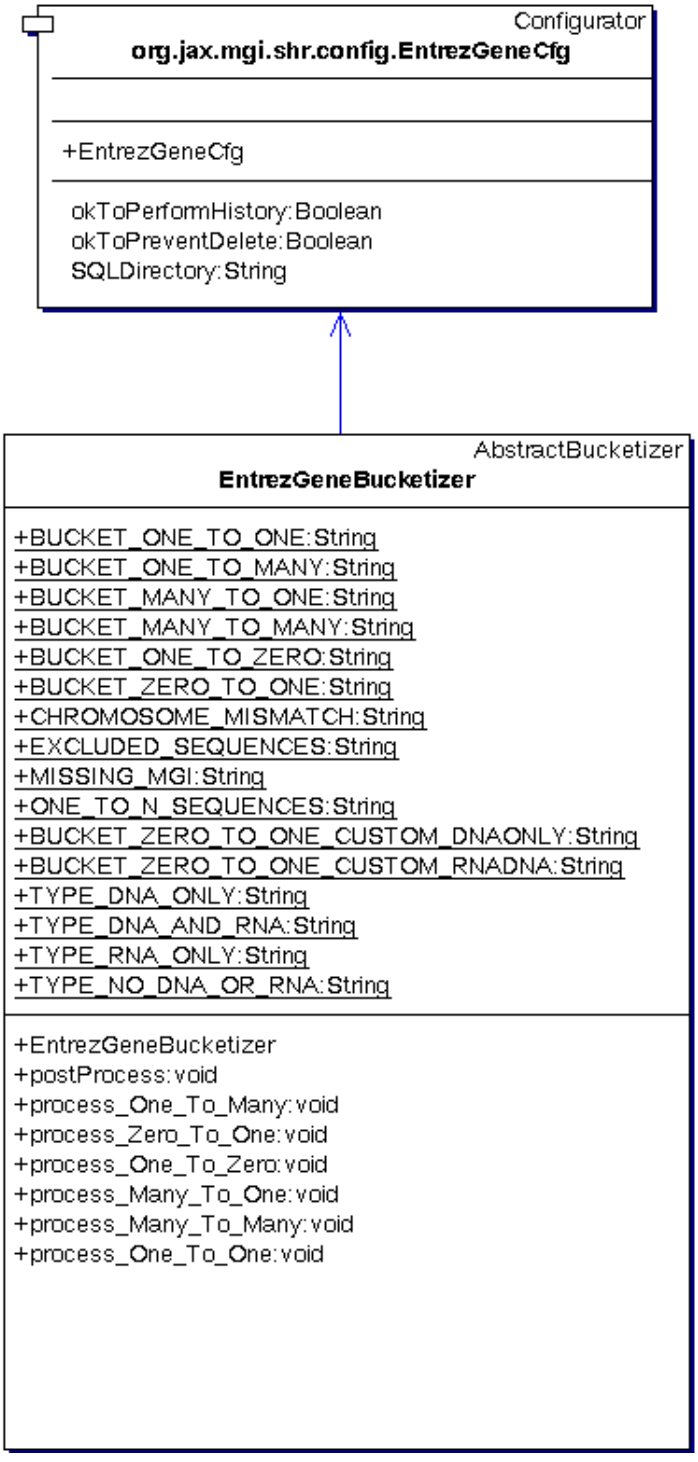
The **NCBIGeneModelLookup** is an extension of **FullCachedLookup** from lib_java_core (see org.jax.mgi.shr.cache). It is used by the **EntrezGeneBucketizer** to lookup up whether or not an Entrez gene id is designated as an NCBI gene model.

4 Class Diagrams

4.1 EntrezGeneLoader



4.2 EntrezGeneBucketizer



5 Processing one to ones

When a cluster is found from the bucketizer process to contain only one MGI gene and one Entrez gene then these genes are considered to match unambiguously. Firstly, the **EntrezGeneBucketizer** checks to make sure that these genes reside on the same chromosome. A chromosome designated as unknown is considered valid to match any chromosome. If this test fails, then the cluster is reported as a mismatched chromosome (see section 6). If this test passes, then the following actions are taken:

- An association is made in MGD between the MGI gene and the Entrez gene.
- Any GenBank sequences associated with the Entrez gene are associated with the MGI marker as long as that sequence is associated to only one Entrez gene.
- RefSeq sequences (XM, XR, XP, NM, NR, NP) associated with the Entrez gene are associated with the MGI marker.
- If the chromosomes match exactly (chromosomes designated as unknown do not count) and if the Entrez gene is represented as a NCBI gene model, then make an association between the MGI gene and the Entrez gene of type Entrez gene model.

6 Reports

6.0.1 bucket_one_to_one.txt

reports the one to one associations.

6.0.2 bucket_one_to_zero.txt

reports MGI genes with no associations to Entrez Gene.

6.0.3 bucket_zero_to_one.txt

reports EntrezGene genes with no corresponding MGI gene.

6.0.4 bucket_one_to_many.txt:

reports MGI genes that are associated to many EntrezGene genes.

6.0.5 bucket_many_to_one.txt

reports EntrezGene genes associated to many MGI genes.

6.0.6 bucket_many_to_many.txt

reports many EntrezGene genes and many MGI genes all associated to each other.

6.0.7 bucket_mismatched_chromosome

reports those one to one matches which were designated to reside on different chromosomes.

6.0.8 excludedSequences.txt

sequences which were excluded from bucketization because they were associated to more than one gene.

6.0.9 missingMGIIDs.txt

EntrezGene genes which matched on sequence alone and contain an MGI id not in MGI.

6.0.10 oneToNSequences

A list of sequences which contain only one MGI marker and many Entrez Gene associations.

6.0.11 HTML Formatted Reports

In addition to the above reports, html formatted versions of all but the one to one report are provided. These reports are named the same as above except they have the html file extension.