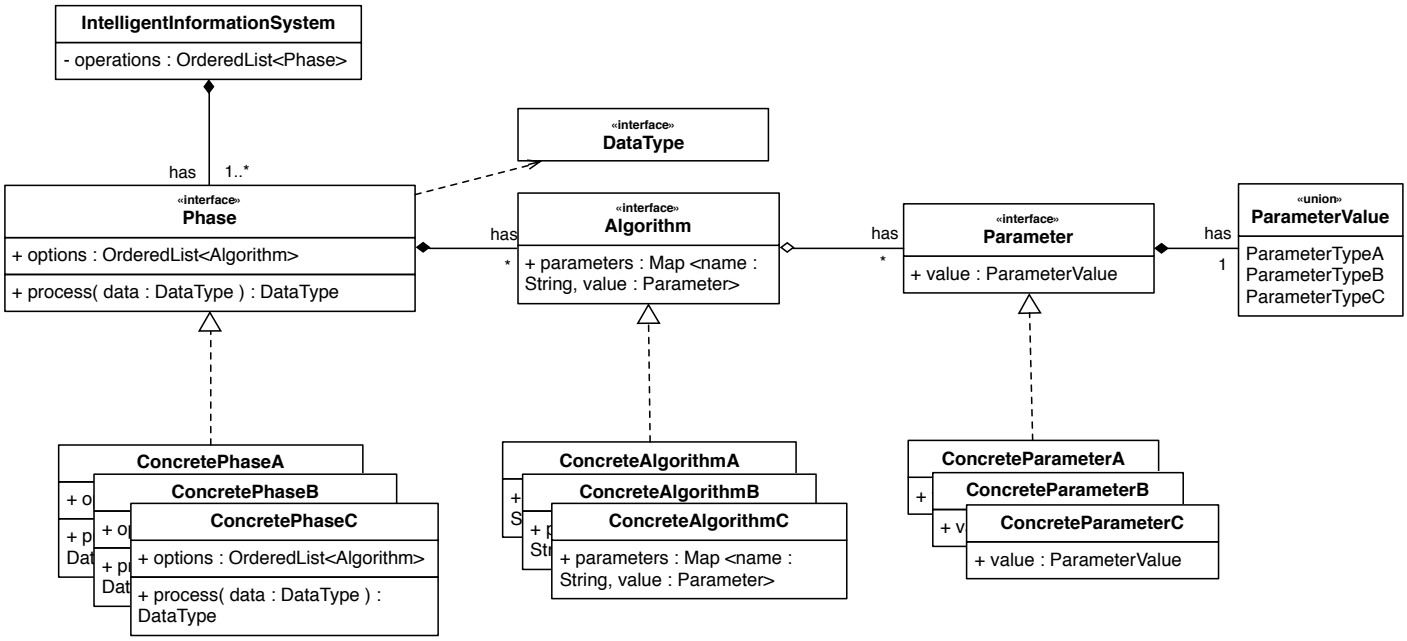
Joseph Chee Chang (josephc1)

11-791 Software Engineering of IIS

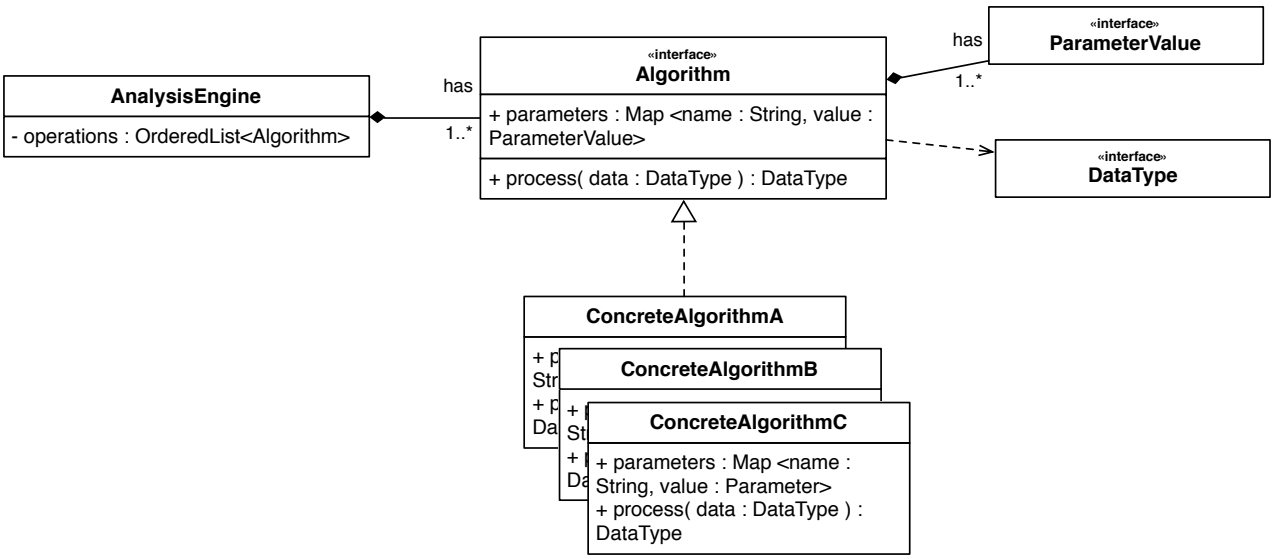
September 24, 2014

Homework 1

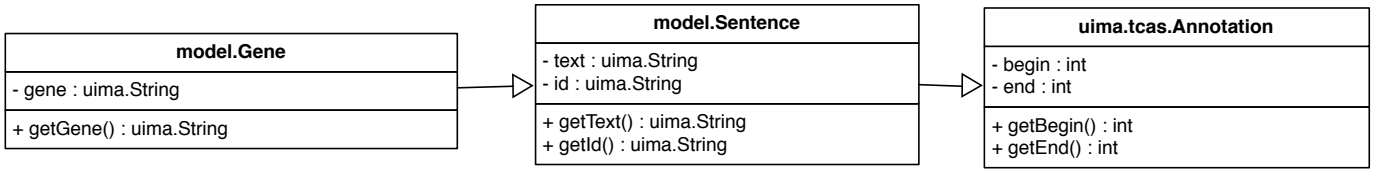
Task 1.1



Task 1.2



Type system



Two types are created for this system. First, model.Sentence inherits Annotation is used by the collection reader to store the input data (id and text). Range parameters (begin and end) is used to indicate the length of the sentence, in which begin always equals to zero. Second, model.Gene inherits model.Sentence with one additional field gene, where the name of the identified gene is stored. The range attributes (begin and end) stores the substring index (including white spaces) of the gene name in the text attribute. Note: only used attributes are shown in the UML graph.

Dataflow / workflow

The system contains three main components: collection reader (SingleFileCollectionReader), annotator (RangeExtractionAnnotator), and CAS consumer (GeneConsumer). Each in charge of: reading the input data, extracting gene names, output to file, respectively.

COLLECTION READER

Class SingleFileCollectionReader inherits the abstract class CollectionReader_ImplBase, which reads from the input file line by line. For each line in the input file, the ID and sentence part is parsed, and one model.Sentence data object is created to store

them. After each line read the data object is added to the jcas index and moved on to later components. **One object is created for each call to getNext()**. **No file contents nor data object is stored in the memory of the object. Progress is calculated by number of lines read and the total number of lines in the file**. This can give us a good idea of the progress during execution. However, it also requires determining the total number of lines in the input file in the initializer, which can be time consuming if the input file is large. The input file stream is opened once, when the object initializes, and closed when the closed function is called by the UIMA framework.

ANALYSIS ENGINE / ANNOTATOR

The system has only one primitive analysis engine, with one annotator. Class RangeExtractionAnnotator inherits the abstract class JCasAnnotator_ImplBase, which uses the pre-trained GENETAG NER tagger for identifying gene sequences in the input sentence. The input of this component is the output of the collection reader, i.e. model.Sentence objects each with a sentence ID and its text. For each of the model.Sentence object the component outputs zero or more model.Gene objects depending on how many gene sequence is identified. The output object has identical sentence ID and text of its corresponding input model.Sentence object, and an additional gene sequence name. Each model.Gene object indicates one identified gene sequence. The range attributes is now used to store the index range of where the gene sequence appeared in the sentence text. **The detail implementation uses the strategy pattern and the singleton pattern, see Design Pattern Used Section for more information.**

ALGORITHM / GENETAG NER TAGGER

The **HMM** chunker from the **LingPipe package** is used in the analysis engine. The model of the chunker is trained on the **Gene Tag corpus**, which can be freely obtained online.

Tanabe, L., Xie, N., Thom, L. H., Matten, W., & Wilbur, W. J. (2005). GENETAG: a tagged corpus for gene/protein named entity recognition. *BMC bioinformatics*, *6*(Suppl 1), S3.

http://alias-i.com/lingpipe/web/models.html

CAS CONSUMER

Class GeneConsumer inherits the abstract class CasConsumer_ImplBase, which receives model.Gene objects from the analysis engine as inputs. **The the range indexes (begin / end) is fixed here before writing to file.** The output file stream is opened once in the class initializer. Since processCas may be called multiple times by the UIMA framework, the output file stream is closed when when the destroy() function is called by the framework.

Design Patterns Used

THE STRATEGY PATTERN

The RangeExtractionAnnotator is a generic range annotator that for a given input text and an extractor object that implements the RangeExtractor interface, will used the input strategy to extract ranges in the input text. In our case, the realization of the extractor is the GeneRangeExtractor class that uses the LingPipe package with a pre-trained

model to extract gene mentions in the given text. However, any other classes that implements the RangeExtractor interface can also by plug into the annotator at runtime.

THE SINGLETON PATTERN

The GeneRangeExtractor strategy is also implemented according to the singleton pattern. Since it is a generic algorithm that can be used throughout different parts of a large system, and it requires loading a memory consuming model, we use the singleton pattern to insure that only one instance of this class can be created. This way, the model can only be load once into the memory.