Homework 3

Salvador Medina
139323
DEIIS, ITAM

March 30, 2014

1 CPE

For the first part of the homework it was required to create a CPE based on the AAE created for homework 2. As the *consumer reader* we could use UIMA's *FileSystemCollectionReader* and create its description file.

1.1 Implementation

In this particular case, we created the CPE as shown in Figure 1 in the descriptor file *hw3-ID-aae-as-CPE.xml*. This CPE takes as input the set of files given in homework 2 through the *FileSystemCollectionReader*. Each of the files then are fed to the aggreagated analysis engine developed for the previous homework. Finally the result are saved into an output file per input file under the same name but in a different folder.

For the final task we had to develop a consumer named *printConsumer* and update the type system.

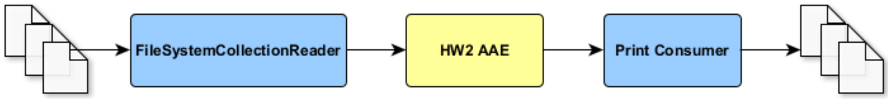


Figure 1: Resulting CPE

1.2 Print Consumer

For our collection consumer we had to add a type into our type system which was able to annotate the results obtained from calculting the Precision@n. Where N denotes the top results obtained from our analysis engine. Therefore, a *Result* type was added. This type holds two attributes as seen in Figure 2:

N: Number of top elements to consider

Precision: Calculated considering N

With this in mind, in the *PrintConsumer.java* all the *Result* annotations are taken from the CAS and printed into a file in the output directory that was established for the consumer.

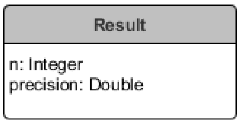


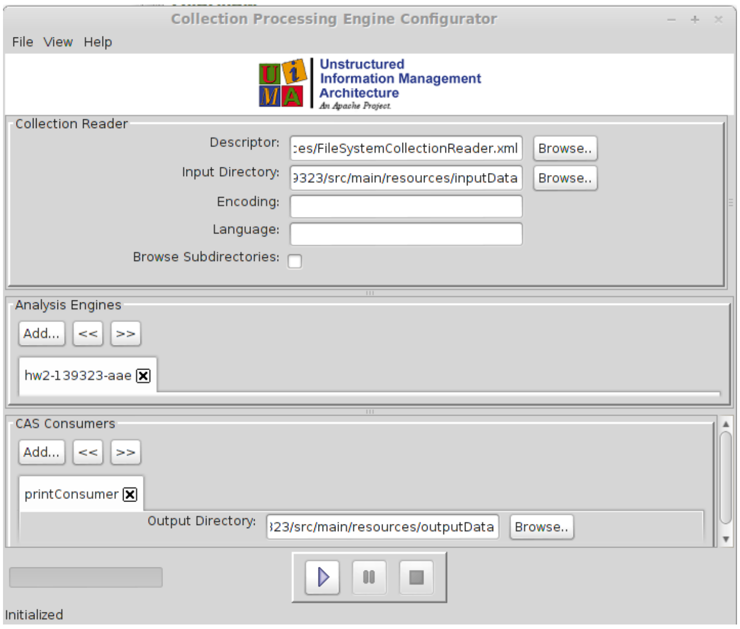
Figure 2: Result Type

1.3 Execution

The CPE GUI tool was executed from Eclipse by running the *UIMA CPE GUI* running configuration which was already established in the project. From there we fed the dialog shown in Figure 3 with the *FileSystemCollectionReader* descriptor, the aggregate analysis engine descriptor and the newly developed *PrintConsumer* descriptor. We set the input and output paths and clicke on the Play button.

From there the CPE executed showing the results in a dialog shown in Figure 4. Besides the shown results, we could also check that the expected output files were created.

This tool was used to create the CPE descriptor file *hw3-139323-cpe.xml* and imported into the Eclipse project.



Collection Reader

Descriptor: :es/FileSystemCollectionReader.xml

Input Directory: 9323/src/main/resources/inputData

Encoding:

Language:

Browse Subdirectories: □

Analysis Engines

hw2-139323-aae ☒

CAS Consumers

printConsumer ☒

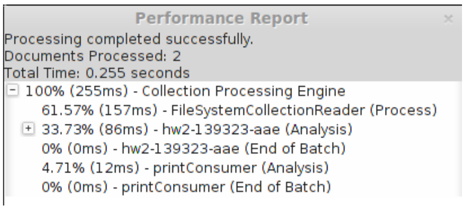
Output Directory: :23/src/main/resources/outputData

Initialized

Figure 3: CPE GUI example of execution

2 UIMA AS

The UIMA AS part was troublesome due to installing the version 2.4.2 instead of the stable version 2.4.0. This brought troubles such as having multiple-binding prob-



```
Processing completed successfully.
Documents Processed: 2
Total Time: 0.255 seconds
⊟ 100% (255ms) - Collection Processing Engine
    61.57% (157ms) - FileSystemCollectionReader (Process)
  ⊞ 33.73% (86ms) - hw2-139323-aae (Analysis)
    0% (0ms) - hw2-139323-aae (End of Batch)
    4.71% (12ms) - printConsumer (Analysis)
    0% (0ms) - printConsumer (End of Batch)
```

Figure 4: CPE GUI execution results

lems with the SLF4J, or not being able to deploy correctly the AAE to the broker, without having any possibility of trying the Stanford Queue.

Up the point of this report it was just recently found and will be working on the rest of the tasks to complete the homework.