Napat Luevisadpaibul ID:nluevisa

Homework 3

1 Document Reference

referenceDoc/hw2-nluevisa-report.pdf: The design of previous version of this system. Sometimes I will compare the current design with the previous design contained in this document

2 Type System Design

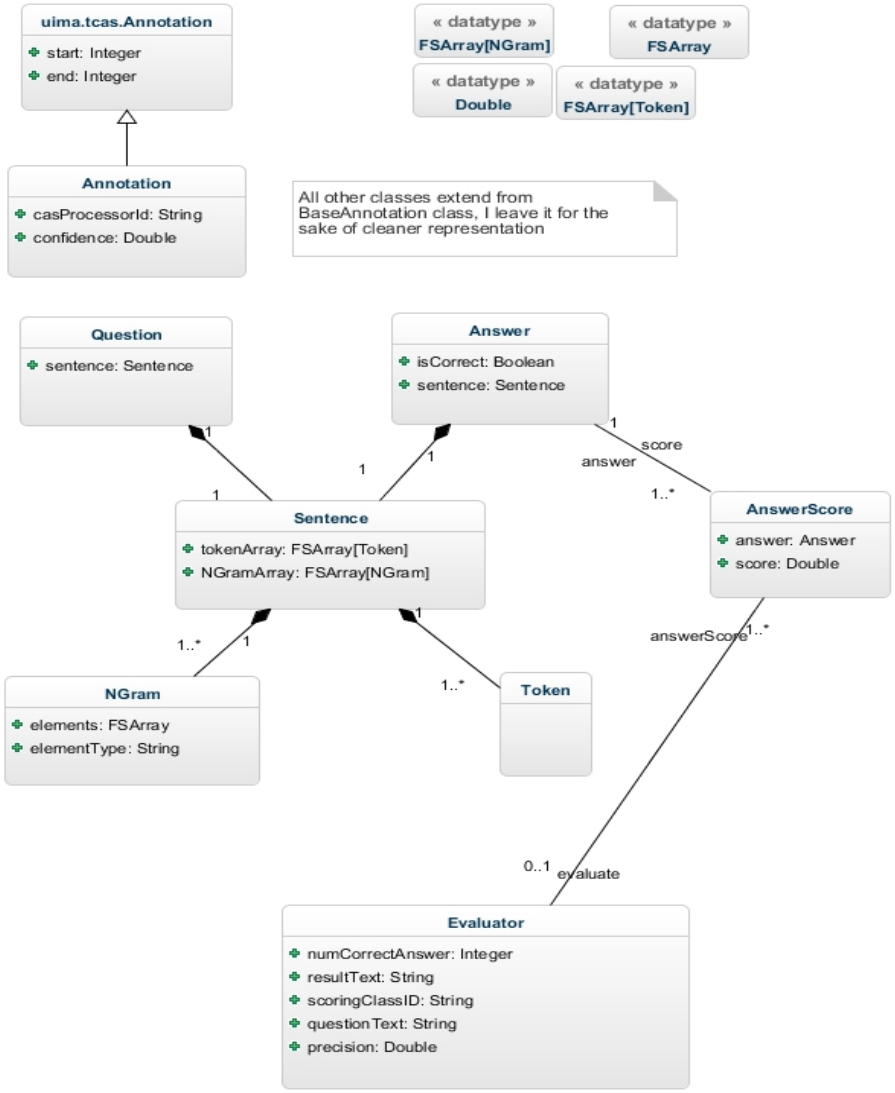


Figure 1 Type System UML Class Diagram

The only difference of my current design and previous design of type system is that type "Evaluator" now has additional features. This is because I want to store the result of each scoring methods for each questions in annotations in order to reduce the computational time if I have multiple Cas consumers and want to show results in different formats (ex. Output screen, XMI, database etc). I don't have to calculate the precision or iterate through AnswerScore every times in each Cas Consumers. Instead the duty my Cas consumer is to define the way to show output to user. "EvaluatorAnnotator" is the class that produce "Evaluator" type which store data ready to show to user (ie question, scoring method, result of each answers and precision of this method to this question).

Note that my evaluator will process every scoring method in analysis engine so we can compare the result of every scoring methods we have in just a single run including their performance using CPE GUI.

3 Annotator Class Design

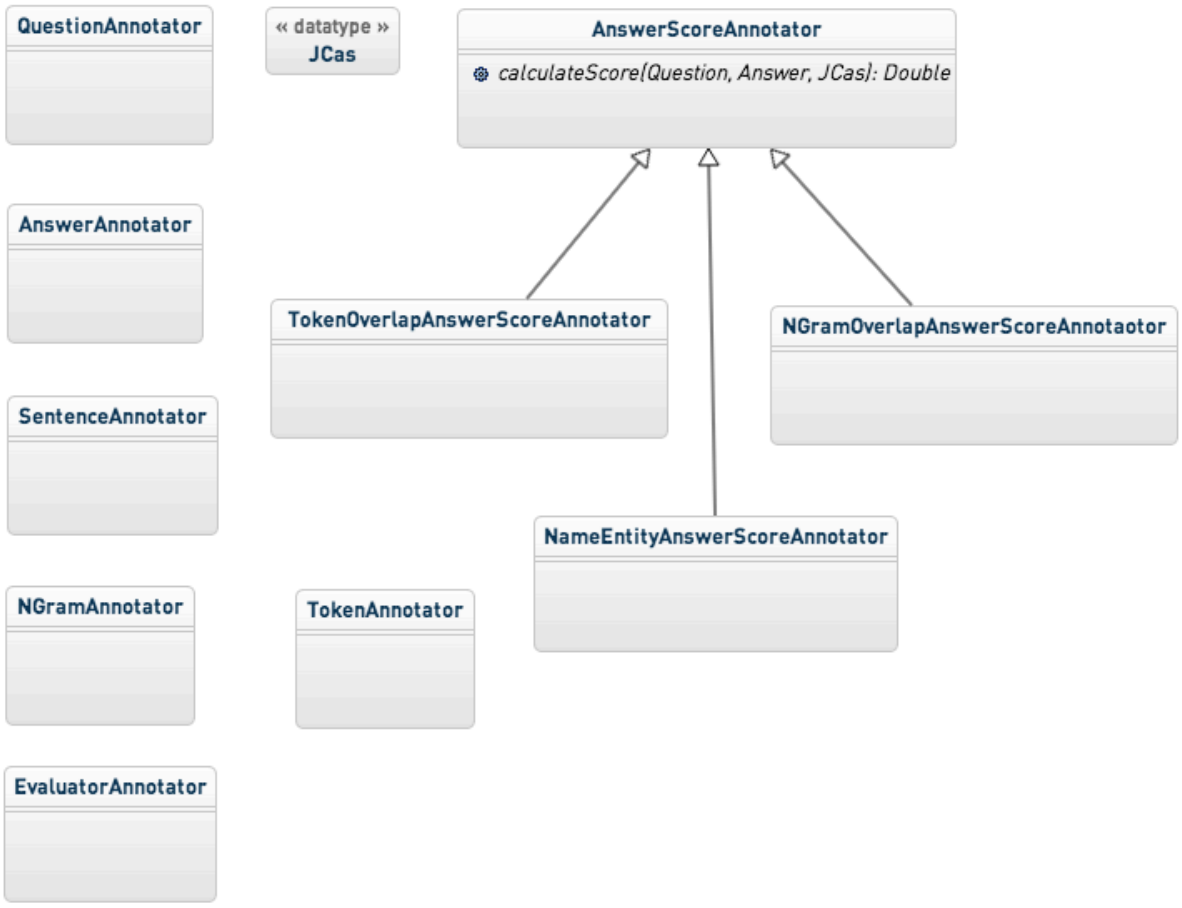


Figure 2 Annotator UML Class Diagram

In HW2 report I also point out that the most notable thing of this diagram is to show that "AnswerScoreAnnotator" is an **abstract class**. The only abstract method it has is calculateScore. "AnswerScoreAnnotator" has implementation that annotated the answer associated with it, set casProcessorId and set the score to value return by calculateScore method. This means we can have as

many methods to calculate score as we want by just simply create a new subclass that extends "AnswerScoreAnnotator", then implement only one method "calculateScore". This uses concept of **program to interface** and **loosely coupling.** The class we newly add will not affect the code in other classes at all so we can provide more capability to our system really easily. In previous system (HW2), I created two scoring methods, NGram overlapping and Token overlapping to illustrate this point. In HW3 I have to integrate a new scoring method using **Name Entity** annotation from Stanford Core NLP service. I can simply do this by just creating new subclass of AnswerScoreAnnotator called **NameEntityAnswerScoreAnnotator,** then implemented few lines of code to produce score without changing anything in other classes at all. Note that I add one parameter of JCas type to calculateScore interface to make it general enough for every AnswerScore class implementing it.

4 Aggregate Analysis Engine Design

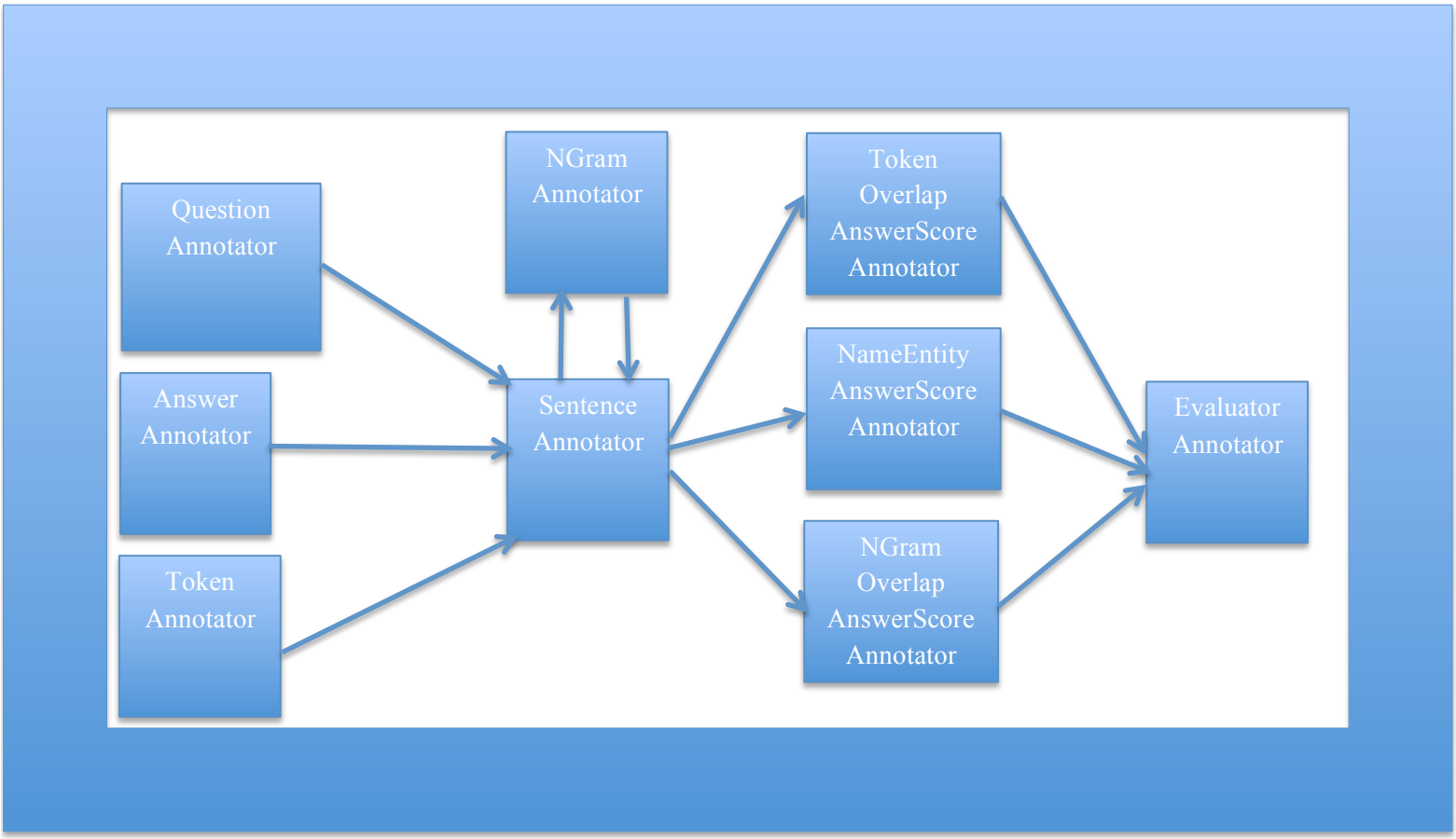


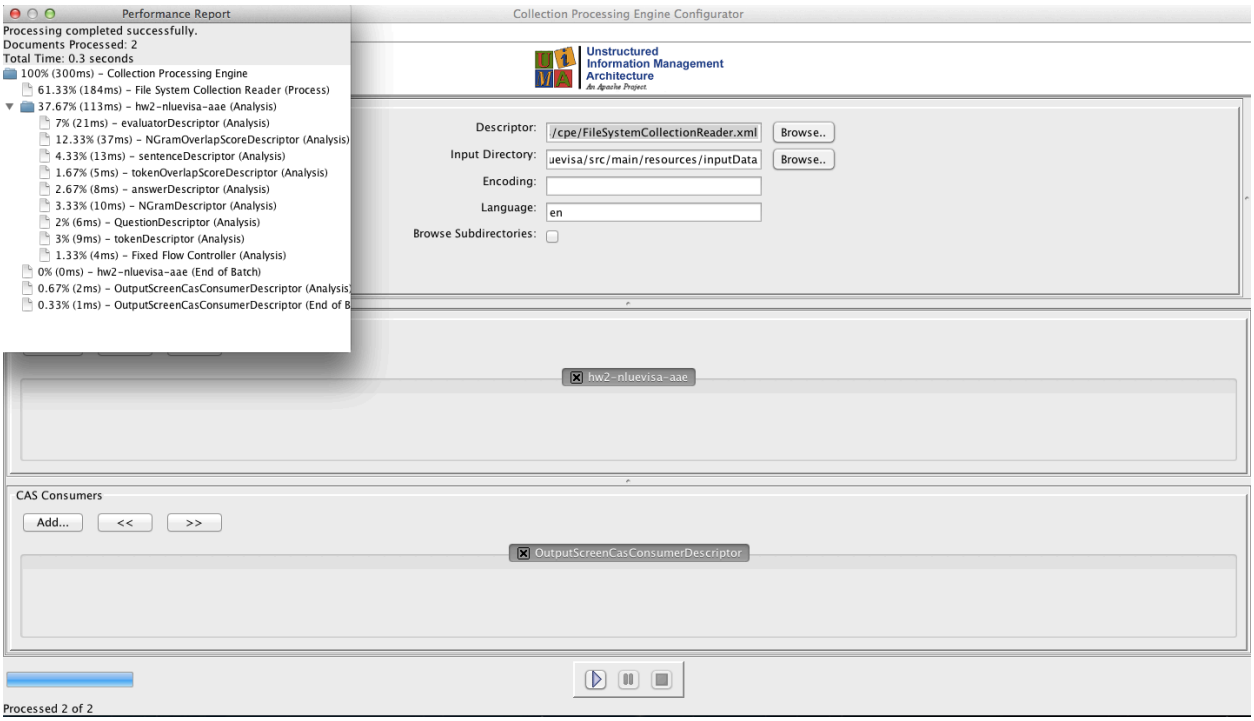
Figure 3 HW3 aggregate analysis engine design

In HW2 report, I mentioned that I don't want to change EvaluatorAnnotator code after I add some more calculating class so I use HashMap, which map the class name to AnswerScore produce by that class. This way I can always print and compare results of every score calculating methods in our pipeline without affecting the code of this class. I think that was a good decision because I don't need to change much of my code when I add Name Entity analysis scoring method to my system (the only change is to adapt Evaluator Annotator to CasConsumer)

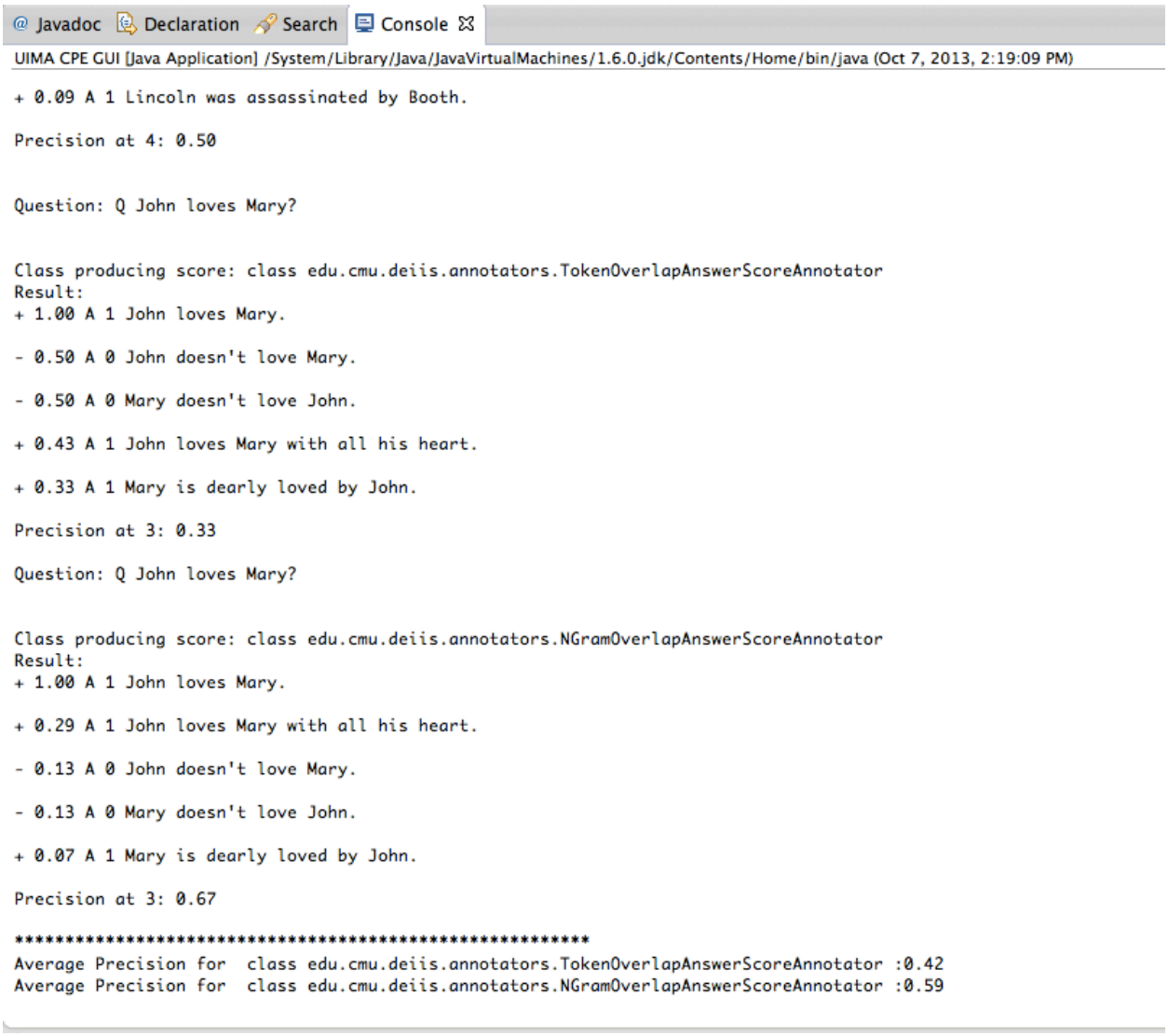
5 Analysis of Program Result

5.1 Task 1: Creating and Running My CPE

Running CPE GUI with hw3-nluevisa-cpe.xml

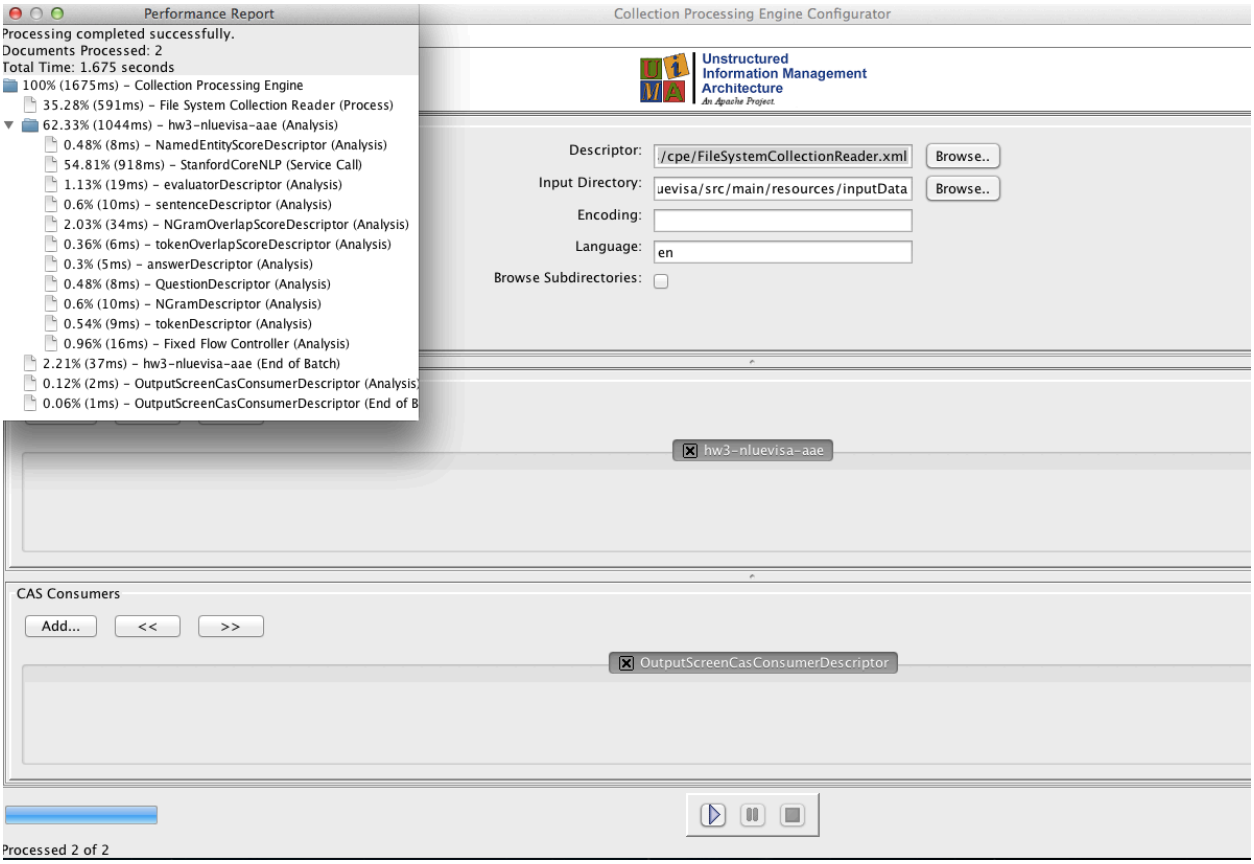


From the performance report, we can see that most of the system running time(61.33%) is spent on File System Collection Reader, which read the input file and initialize our Cas. In Analysis pipeline, the most time consuming part in hw2 analysis engine is NGramOverLapScoreAnnotator because we need to compare all Unigram, Bigram and Trigram of each questions and answers to produce score in this class. The second most time consuming in analysis pipeline is EvaluatorAnnotator while OutputScreenCasConsumer consume only 0.67% of our process time (I delegate almost all computations to EvaluatorAnnotator so every CasConsumer will process really fast).



This result on screen will show the question, class to compute score, answer with score ant + to indicate that it's correct or – if wrong, precision at top N answer and the average precision for each scoring methods.

5.2 Task 2.2: Creating an UIMA-AS client to integrate Name Entity from Stanford Core NLP



You can see that StanfordCoreNLP (Service Call) consumed half of my processing time. That is to be expected because we need to call the service from outside our machine.

The console result looks like this:

```
Question: Q Booth shot Lincoln?


Class producing score: class edu.cmu.deiis.annotators.TokenOverlapAnswerScoreAnnotator
Result:
- 1.00 A 0 Lincoln shot Booth.

+ 1.00 A 1 Booth shot Lincoln.

- 0.67 A 0 Lincoln assassinated Booth.

+ 0.67 A 1 Booth assassinated Lincoln.

- 0.60 A 0 Booth was shot by Lincoln.

+ 0.60 A 1 Lincoln was shot by Booth.

+ 0.40 A 1 Lincoln was assassinated by Booth.

- 0.40 A 0 Booth was assassinated by Lincoln.

Precision at 4: 0.50

Question: Q Booth shot Lincoln?


Class producing score: class edu.cmu.deiis.annotators.NGramOverlapAnswerScoreAnnotator
Result:
+ 1.00 A 1 Booth shot Lincoln.

- 0.30 A 0 Lincoln shot Booth.

- 0.20 A 0 Lincoln assassinated Booth.

+ 0.20 A 1 Booth assassinated Lincoln.

- 0.14 A 0 Booth was shot by Lincoln.

+ 0.14 A 1 Lincoln was shot by Booth.

- 0.09 A 0 Booth was assassinated by Lincoln.

+ 0.09 A 1 Lincoln was assassinated by Booth.

Precision at 4: 0.50
```

```
Precision at 4: 0.50

Question: Q Booth shot Lincoln?


Class producing score: class edu.cmu.deiis.annotators.NameEntityAnswerScoreAnnotator
Result:
+ 0.33 A 1 Booth assassinated Lincoln.

+ 0.33 A 1 Booth shot Lincoln.

+ 0.25 A 1 Lincoln was shot by Booth.

+ 0.25 A 1 Lincoln was assassinated by Booth.

- 0.25 A 0 Booth was shot by Lincoln.

- 0.25 A 0 Lincoln shot Booth.

- 0.25 A 0 Booth was assassinated by Lincoln.

- 0.25 A 0 Lincoln assassinated Booth.

Precision at 4: 1.00


Question: Q John loves Mary?


Class producing score: class edu.cmu.deiis.annotators.TokenOverlapAnswerScoreAnnotator
Result:
+ 1.00 A 1 John loves Mary.

- 0.50 A 0 John doesn't love Mary.

- 0.50 A 0 Mary doesn't love John.

+ 0.43 A 1 John loves Mary with all his heart.

+ 0.33 A 1 Mary is dearly loved by John.

Precision at 3: 0.33

Question: Q John loves Mary?
```

```
+ 0.43 A 1 John loves Mary with all his heart.

+ 0.33 A 1 Mary is dearly loved by John.

Precision at 3: 0.33

Question: Q John loves Mary?


Class producing score: class edu.cmu.deiis.annotators.NGramOverlapAnswerScoreAnnotator
Result:
+ 1.00 A 1 John loves Mary.

+ 0.29 A 1 John loves Mary with all his heart.

- 0.13 A 0 Mary doesn't love John.

- 0.13 A 0 John doesn't love Mary.

+ 0.07 A 1 Mary is dearly loved by John.

Precision at 3: 0.67

Question: Q John loves Mary?


Class producing score: class edu.cmu.deiis.annotators.NameEntityAnswerScoreAnnotator
Result:
+ 0.50 A 1 Mary is dearly loved by John.

- 0.50 A 0 John doesn't love Mary.

- 0.50 A 0 Mary doesn't love John.

+ 0.50 A 1 John loves Mary.

+ 0.33 A 1 John loves Mary with all his heart.

Precision at 3: 0.33

************************************************************
Average Precision for  class edu.cmu.deiis.annotators.TokenOverlapAnswerScoreAnnotator :0.42
Average Precision for  class edu.cmu.deiis.annotators.NameEntityAnswerScoreAnnotator :0.67
Average Precision for  class edu.cmu.deiis.annotators.NGramOverlapAnswerScoreAnnotator :0.59
```

Looking at the result, I've successfully integrated Name Entity from Stanford NLP annotation to compute score using Name Entity Overlap scoring method to improve my average precision from 0.59 in NGram Overlap to 0.67 with this new method. This method produce astonishing result in Question "Booth Shot Lincoln? " (perfect score) but do poorly on the question "John Loves Mary"

Note that when I pass Question and Answer to method "calculateScore" in every AnswerScore classes, I get rid of prefix Q/A and 1/0. If I don't do this, the average precision will drop significantly (around 0.3 for Name Entity Overlap method)

5.3 Task 2.3: Creating and Deploying my own UIMA AS service

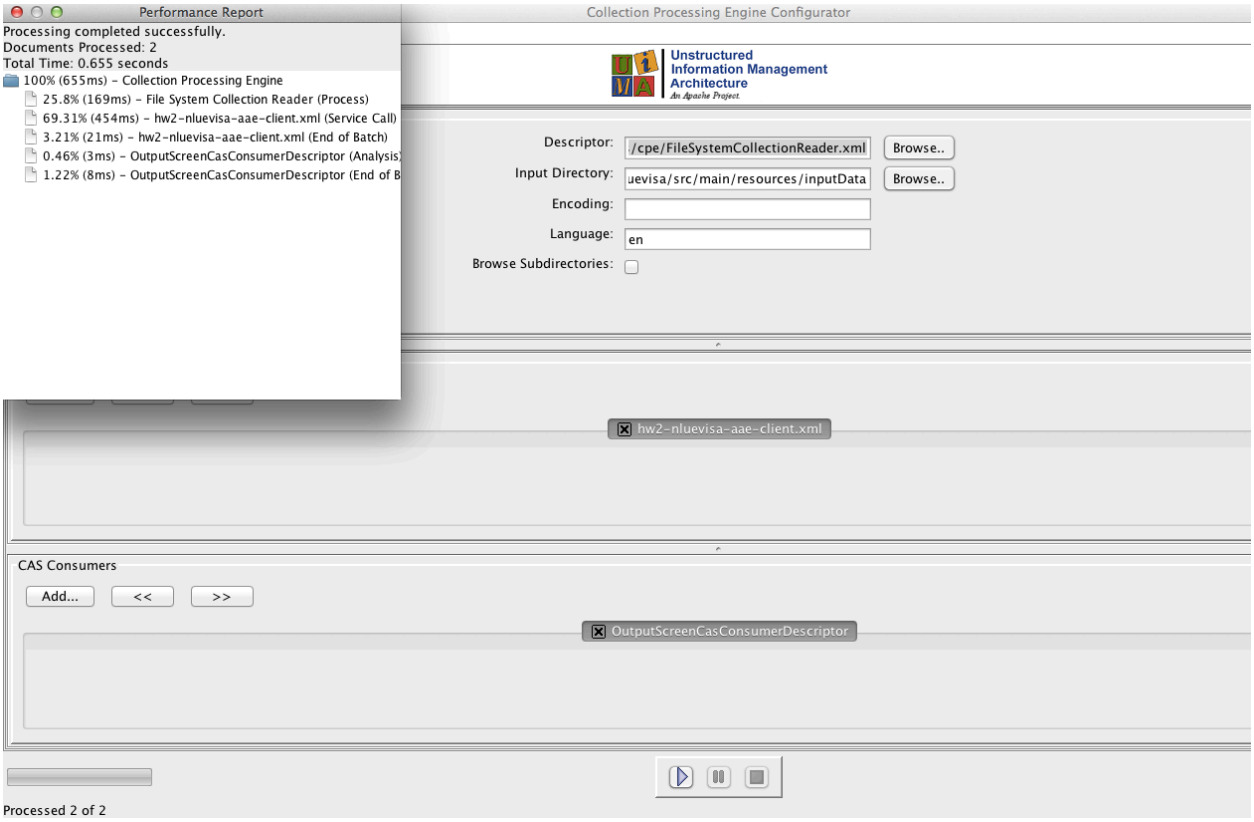
First, run startBroker.sh result

```
INFO: Using java '/System/Library/Frameworks/JavaVM.framework/Home/bin/java'
Java Runtime: Apple Inc. 1.6.0_51 /System/Library/Java/JavaVirtualMachines/1.6.0.jdk/Contents/Home
  Heap sizes: current=83008k  free=81619k  max=126912k
    JVM args: -Dactivemq.classpath=amq/conf;/Users/napatluevisadpaibul/Desktop/uima-as/apache-activemq-5.4.1/conf; -Dactivemq.home=/
Users/napatluevisadpaibul/Desktop/uima-as/apache-activemq-5.4.1 -Dactivemq.base=amq
ACTIVEMQ_HOME: /Users/napatluevisadpaibul/Desktop/uima-as/apache-activemq-5.4.1
ACTIVEMQ_BASE: amq
Loading message broker from: xbean:file:amq/conf/activemq-nojournal.xml
INFO  BrokerService                 - Using Persistence Adapter: MemoryPersistenceAdapter
INFO  BrokerService                 - ActiveMQ 5.4.1 JMS Message Broker (localhost) is starting
INFO  BrokerService                 - For help or more information please see: http://activemq.apache.org/
INFO  ManagementContext             - JMX consoles can connect to service:jmx:rmi:///jndi/rmi://localhost:1099/jmxrmi
INFO  TransportServerThreadSupport  - Listening for connections at: tcp://Napats-MacBook-Air.local:61616
INFO  TransportConnector            - Connector openwire Started
INFO  BrokerService                 - ActiveMQ JMS Message Broker (localhost, ID:Napats-MacBook-Air.local-50196-1381172558480-0:0)
started
```

Second, deployAsynchService

```
Napats-MacBook-Air:bin napatluevisadpaibul$ deployAsyncService.sh /Users/napatluevisadpaibul/git/hw2-nluevisa/hw2-nluevisa/src/main/
resources/hw2-nluevisa-aae-deploy.xml
Service:hw2-nluevisa-aae Initialized. Ready To Process Messages From Queue:HW2AAEQueue
Press 'q'+'Enter' to quiesce and stop the service or 's'+'Enter' to stop it now.
Note: selected option is not echoed on the console.
```

Finally, run CPE GUI with hw3-nluevisa-aee-as-cpe.xml

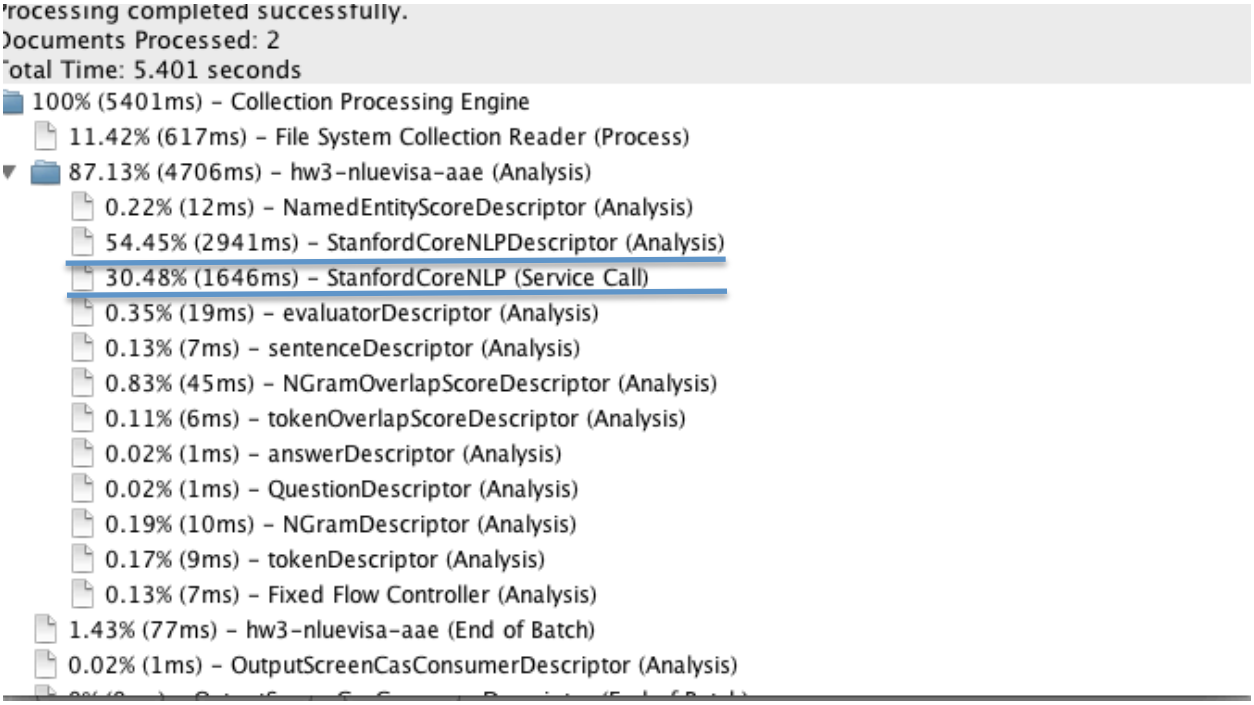


The console result output the same thing as in task 1

5.4 Task 2.4: Run Stanford Core NLP Locally and compare with the remote one

I've created StanfordCoreNLPDescriptor that will initialize annotator in clearTK. I did an experiment by put both this descriptor and the remote one from task 2.2 into the same aggregate analysis engine so that I can compare the performance using CPE GUI.

This is the result from running CPE GUI



The result of this experiment may be counter-intuitive because the locally built one is actually slower than the remote one (2941ms VS 1646ms). This is because of overhead in initializing step. Each time CPE run we have to initialize resource for local built analysis engine, which in this case outweighs the cost of transfer data over network. There is one more factor to consider which is memory. Locally built Stanford Core NLP consumes much more memory in our machine than the remote one.