HW3: Execution Architecture with CPE and Deployment Architecture with UIMA-AS

Fernando Garza - 146368
March 28, 2014

1 INTRODUCTION

The purpose of this homework was to execute the pipeline developed on homework 2 with a CPE, and also deploy it as a service using UIMA-AS.

2 CPE

The CPE consists of three main stages:

- Collection reader
- Analysis engine
- Cas Consumer

2.1 COLLECTION READER

Dependening where the input files are located, the collection reader needs to establish a connection with the source files and then stream them to the CPE. Since our files are text files with q/a, we only need to read the filesystem and get the text files.

For this, one of uima's core components was used, called *FileSystemCollectionReader*, located in uima tools/components folder.

A descriptor for this component was created, which has the following configuration parameters:

- InputDirectory: the folder where the source q/a text files are located, by default this parameter takes the computer's location of the homework and the folder *inputData*.
- Analysis engine: the already developed aggregated analysis engine. For this homework the file is located in the resources folder with the name *hw3-146368-aae.xml*.
- Cas Consumer: the "printer" for the evaluations of the answers. For this, a code was developed in the java sources called *CasConsumerPrint.java*. Basically it does the same process as the Evaluator, but instead generates a file in the outputFolder (by default is referenced to the folder of the computer where the homework was done) called *print.txt*

After defining the stages, the UIMA CPE GUI was used to generate the CPE's descriptor file, which is located on the resources folder with the name *hw3-146368-CPE.xml*.

3 DEPLOYMENT

The idea behind using UIMA-AS was to deploy the service in order to be able to call it remotely. UIMA-AS consists of the UIMA framework with the ActiveMQ broker, which is used to get the capabilities of remote services.
The first step was to re-define the UIMA_HOME environment variable for the new UIMA-AS installation.

3.1 REMOTE SERVICE

In this part of the homework we were required to call the SCNL (Stanford CoreNLP). For this, a client descriptor had to be done, which is called *scnlp-146368-client.xml*, the definition of the descriptor complies with the other syntax used for the aggregate analysis and other descriptors. The descriptor was written and run.

The typesystem used for Stanford CoreNLP had to be redefined for the core functions used by stanford, but corresponds to the one we had been using on this project. A timeout was recie

3.2 UIMA-AS SERVICE DEPLOYMENT

In order for the aae to run as a service some changes had to be made to the environmental variables, such as the UIMA_CLASSPATH, because the deployment was made using command-line.

For the broker to start, the sheell command *$UIMA_HOME/bin/startBroker.sh* was called. Some difficulties were found while starting the broker, which helped to make sure that all the paths declarations were correctly stated, also, the UIMA-AS package had some structure errors,

because it took some dependencies that were already updated, some changes had to be made. In general, the files required to deploy the service were written in a very straightforward way, by looking at the UIMA-AS examples.

No bonus was taken for this homework.

4 REFERENCES

- Lane, Dale. Using UIMA-AS to run UIMA annotators in parallel, http://dalelane.co.uk/blog/?tag=uima-as
- UIMA Collection Processing Engine DeveloperâĂŹs Guide, http://uima.apache.org/d/uimaj-2.4.0/tutorials_and_users_guides.html#ugr.tug.cpe
- UIMA CPE GUI Manual, http://uima.apache. org/d/uimaj-2.4.0/tools.html#ugr.tools.cpe
- UIMA AS Manual: Getting Started: Apache UIMA Asynchronous Scaleout, http://uima.apache.org/doc-uimaas-what.html