# MATAWS : Multimodal Approach for Automatic WS Semantic Annotation
# Developer Guide for MATAWS 1.0.0

written by Cihan Aksoy

caksoy@uekae.tubitak.gov.tr

Galatasaray University, Computer Science Department, Istanbul, Turkey

TÜBİTAK - The Scientific and Technological Research Council of Turkey, Gebze/Kocaeli, Turkey

## License

# Table of contents

# 1 Introduction

MATAWS [1, 2] is an automatic semantic annotation tool for Web service (WS) collections. This document provides a brief overview of the program structure for developers. Also installation instructions are described to prepare the developing environment of the tool. A more detailed documentation is available inside the source code in the JavaDoc format. Developers should also refer to MATAWS *User Manual*.

# 2 Code Structure

The Primary class of MATAWS is `AnnotationManager`. It manages all components described in the *User Manual*. It is necessary to get an instance of this class to start the process.

`SineUtil` is used to extract the parameters from the WS collection by using the *SINE API* [3].

An implementation of `Annotator` interface is necessary for preprocessing and concept associating. Since the preprocessing module corresponds to *Strategy Pattern*, it is possible to make a collection-specific preprocessing set by implementing `PreprocessingSet`. Then it should be injected into an `Annotator` implementation. Since the *Sigma tool* [4] is used in this version of MATAWS, `SigmaAnnotatorImpl` is implementing `Annotator` interface and uses `SigmaUtil` to retrieve the concept associated to a given word.

The `Transformer` interface is prepared to generate a semantic WS collection. In MATAWS, `OWLSTransformerImpl` implements this interface by using the *OWL-S API* [5].

The `Statistics` interface is designed to calculate statistics concerning the results of the annotation process. With the `StatisticsImpl` implementation, all parameters and words are held after `Annotator` processes. Therefore, all statistical results may be prepared.

The `Output` interface takes the statistical results and may output into the desired form. In MATAWS, only `TextOutputImpl` outputs as a text file.

## 3 Installation

After having understood the code structure of the tool, you may develop the tool for any purpose. The basic instructions you should follow to avoid problems are explained in this section.

Firstly, create a Java project. Secondly, since MATAWS definitely requires the folders `configurations`, `input`, `kbs`, `output` and `statistics` under the root directory of the project, take these folders from the bundle and place them under the root directory of the project you just created. Thirdly, you should add the libraries found in the `lib` folder of the bundle and to the build path of your project. You can do that in a way by taking and placing the `lib` folder under the root directory of your project and by adding all libraries of the `lib` to the build path. After having followed these steps, you are ready for developing the program.

Once the environment is prepared as explained above, you may create a runner class to test the program. The runner class must get an instance of `AnnotationManager`, and call its `startProcess()`. Before running the runner class, you should place your WS collection(s) in the `input` folder. After having run the program, your WS collection(s) will be annotated and the corresponding semantic WS collection(s) will be generated in the `output` folder. If you do not have any problem during these operations, you may pass to the development of the tool.

Enjoy MATAWS!

## References

1. Aksoy, C., Labatut, V., Cherifi, C., Santucci, J.-F.: Mataws: A Multimodal Approach for Automatic Ws Semantic Annotation. In: 3rd International Conference on Networked Digital Technologies, (2011)
2. Aksoy, C., Mancuhan, K.: Annotation Automatique De Descriptions De Services Web. Thesis, Galatasaray University, Istanbul, TR. (2010)
3. Cherifi, C., Rivierre, Y., Santucci, J.-F.: Ws-Next, a Web Services Network Extractor Toolkit. In: 5th International Conference on Information Technology, (2011)
4. Pease, A.: Sigma Knowledge Engineering Environment, http://sigmakee.sourceforge.net/
5. Sirin, E., Parsia, B.: The Owl-S Java Api. In: 3rd International Semantic Web Conference, (2004)