Requirements analysis of DeftEval

1.0 OBJECTIVE/GOALS

1.1 Objectives

Successful completion of the Requirements Analysis Phase should comprise:

- Definition of approved requirements
- Describing the major components

1.2 Goal

Our project must recognize and extract all definitions from a given text. To do that, we annotate each definition with B.I.O. tags. That means the application will know with those tags that B is for begin, I for inside and O for the part of the statement that is not bound by definition, which is on the outside. Those tags help us to identify special cases when a definition can contain one or more definition, or when exist information that does not refer to the definition and this increase the ambiguity.

2.0 Components

Majors components that are combined to conclude the project start with know neuro-linguistic programming technique:

2.1 Separation of sentence one by one. This is a mandatory action that help us to have a separated list and filter in that way to reduce the number of “noises” in the input text.

2.2 The second component is about classify the sentence or the group of sentence to be a definition. To visualize definitions in a contract it is necessary to first extract them and clauses from the contract. We use relatively trivial regular expressions which are applied in three stages: (1) identification and segmentation of the definition clause and other clauses in the text of the contract; (2) segmentation of definitions from each other; and (3) extraction of the defined term and its defining text from a definition. By requiring users to apply simple rules which are widely used in Australian industry practice, such as ensuring definitions ends with a full stop and using standard ‘key words’ for definition relations (particularly the words ‘means’ and

‘includes’) essentially 100% accuracy can be attained on typical contract texts. We can also try to come with a Bayes Naive approach to increase the accuracy .

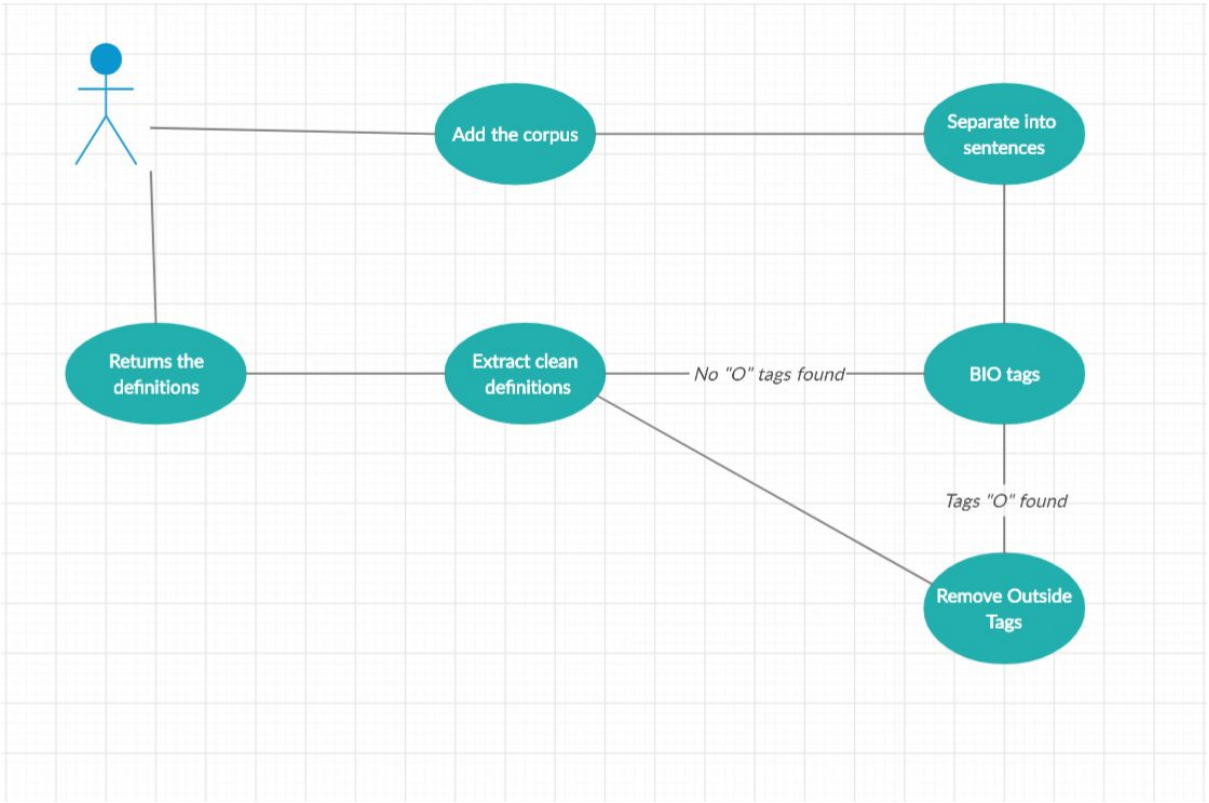
2.3 Label each token with BIO tags according to the corpus' tag specification. Each definition must be tagged with BIO. We should annotate the beginning, inside and what is outside in our definition.This task is important for the next step of the project that is relation classification.

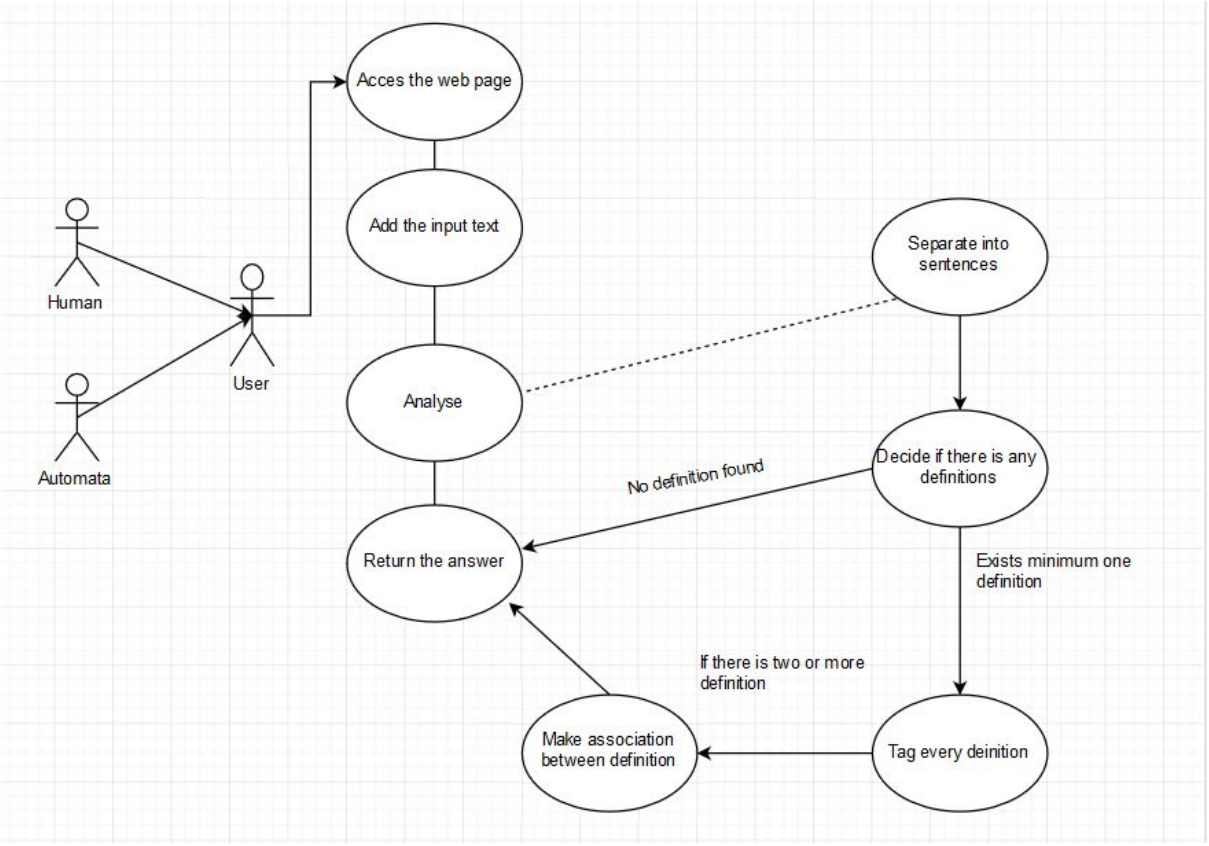
The “outside” tag will help us to eliminate parts of sentences that are not part of the definition, keeping the definition clean. For example, if we have the next sentences: “The chicken, which eats wheat, is a bird.” , the B tag will be put before the word “the”, the I tag will be “The chicken is a bird” and the O tag will be “which eats wheat” .

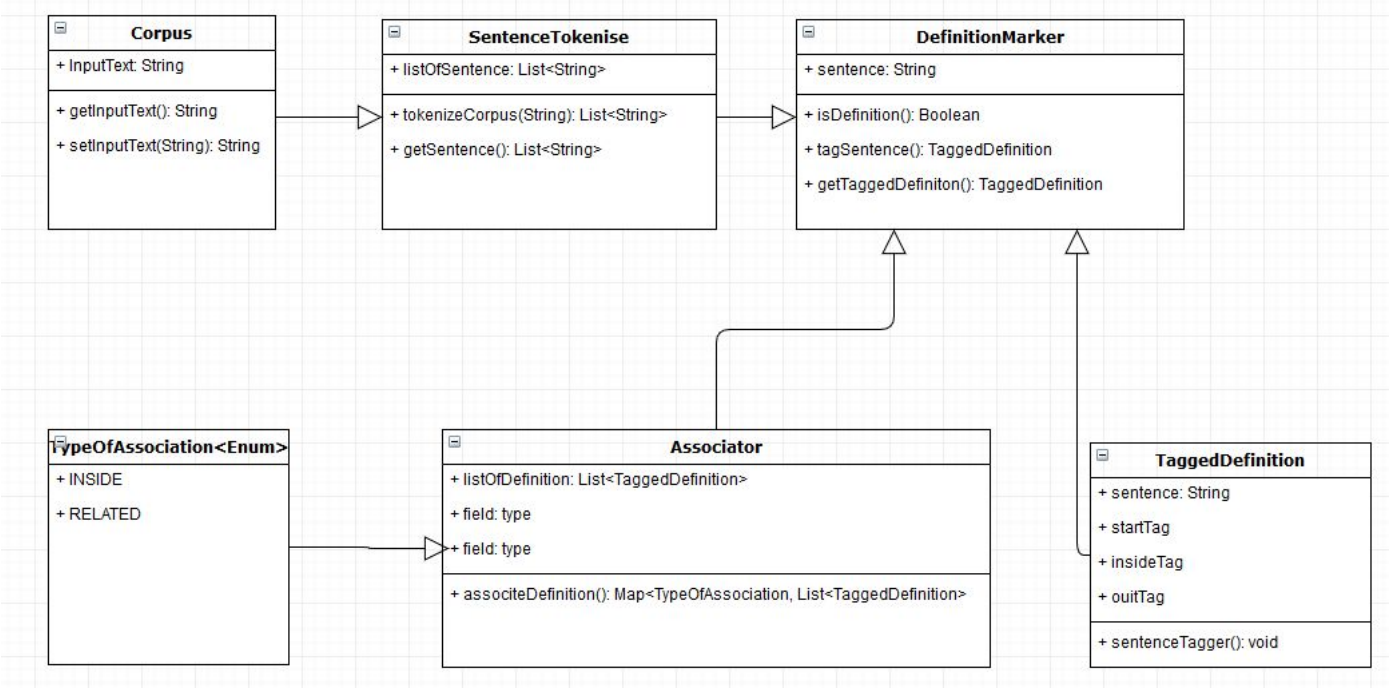
2.4 Given the tag sequence labels, label the relations between each tag according to the corpus relation specification. After this task is complete, we can now extract all definition and all connection between definition from a given corpus.

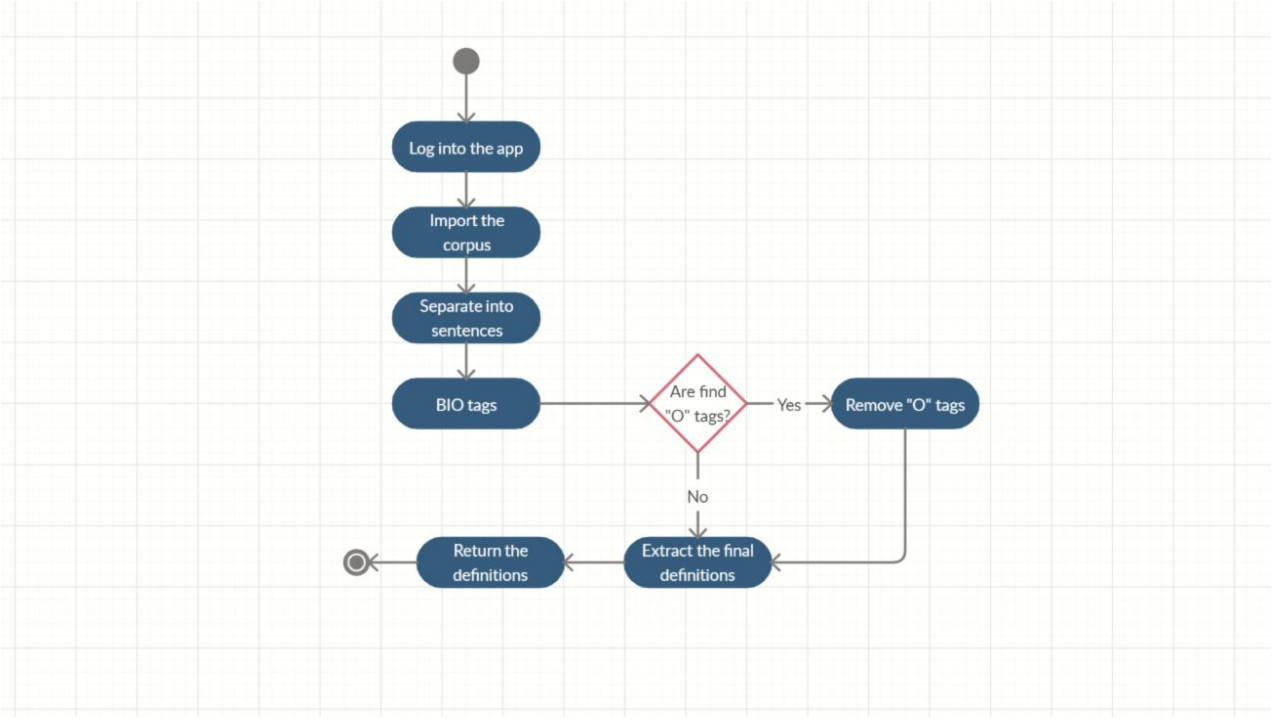
3.0 Use Case Scenarios

- **Actors**:
  - Students who want to extract definitions from a large text to better understand a lesson or a literary work.
  - Teachers or proofreaders who want to verify the correctness of the definitions in a large text written by unqualified persons.
- **Scenarios**
  - The student / teacher accesses the project and imports the corpus which he wants to extract the definitions.
  - The application separates the corpus into sentences.
  - BIO tags are placed for each sentence and parts of sentences that are not necessary are eliminated.
  - The application extract all definition from corpus.
  - The application returns the definitions in the entered corpus.


Add the corpus
Separate into sentences
Returns the definitions
Extract clean definitions
No "O" tags found
BIO tags
Tags "O" found
Remove Outside Tags


Acces the web page
Add the input text
Human
User
Automata
Analyse
Separate into sentences
Decide if there is any definitions
No definition found
Return the answer
Exists minimum one definition
If there is two or more definition
Make association between definition
Tag every deinition


Corpus
+ InputText: String
+ getInputText(): String
+ setInputText(String): String
SentenceTokenise
+ listOfSentence: List<String>
+ tokenizeCorpus(String): List<String>
+ getSentence(): List<String>
DefinitionMarker
+ sentence: String
+ isDefinition(): Boolean
+ tagSentence(): TaggedDefinition
+ getTaggedDefiniton(): TaggedDefinition
TypeOfAssociation<Enum>
+ INSIDE
+ RELATED
Associator
+ listOfDefinition: List<TaggedDefinition>
+ field: type
+ field: type
+ associteDefinition(): Map<TypeOfAssociation, List<TaggedDefinition>
TaggedDefinition
+ sentence: String
+ startTag
+ insideTag
+ ouitTag
+ sentenceTagger(): void


Log into the app
Import the corpus
Separate into sentences
BIO tags
Are find "O" tags?
Yes
Remove "O" tags
No
Extract the final definitions
Return the definitions