# 11-791: Homework 1

**Name: Bo Ma, Andrew id: bom**

Logical Data Model and UIMA Type System Design & Implementation

# 1. Introduction.

I design 9 types of system shown as figure 1. In the first part I will give a whole structure .In the second part, I will talk about the detailed information of each type.

Annotation is the base system type for all other type. BaseAnnotationType is the uima cas type extended by the annotation type. BaseAnnotationType extend all other type. This mean that each type will have the four feature: begin, end, casprocessId, confidence. Token type is the basic and smallest type in the document and it extend the NGram type and DocumentAnnotation type. Furthermore, the Answer type will extend the Answerscore type. In addition, the answer score type extend the AnswerTopN type. The evaluation type is annotated by the AnswerTopN Type.

Please refer to the figure 1 to get more clear information about dependency of each type.

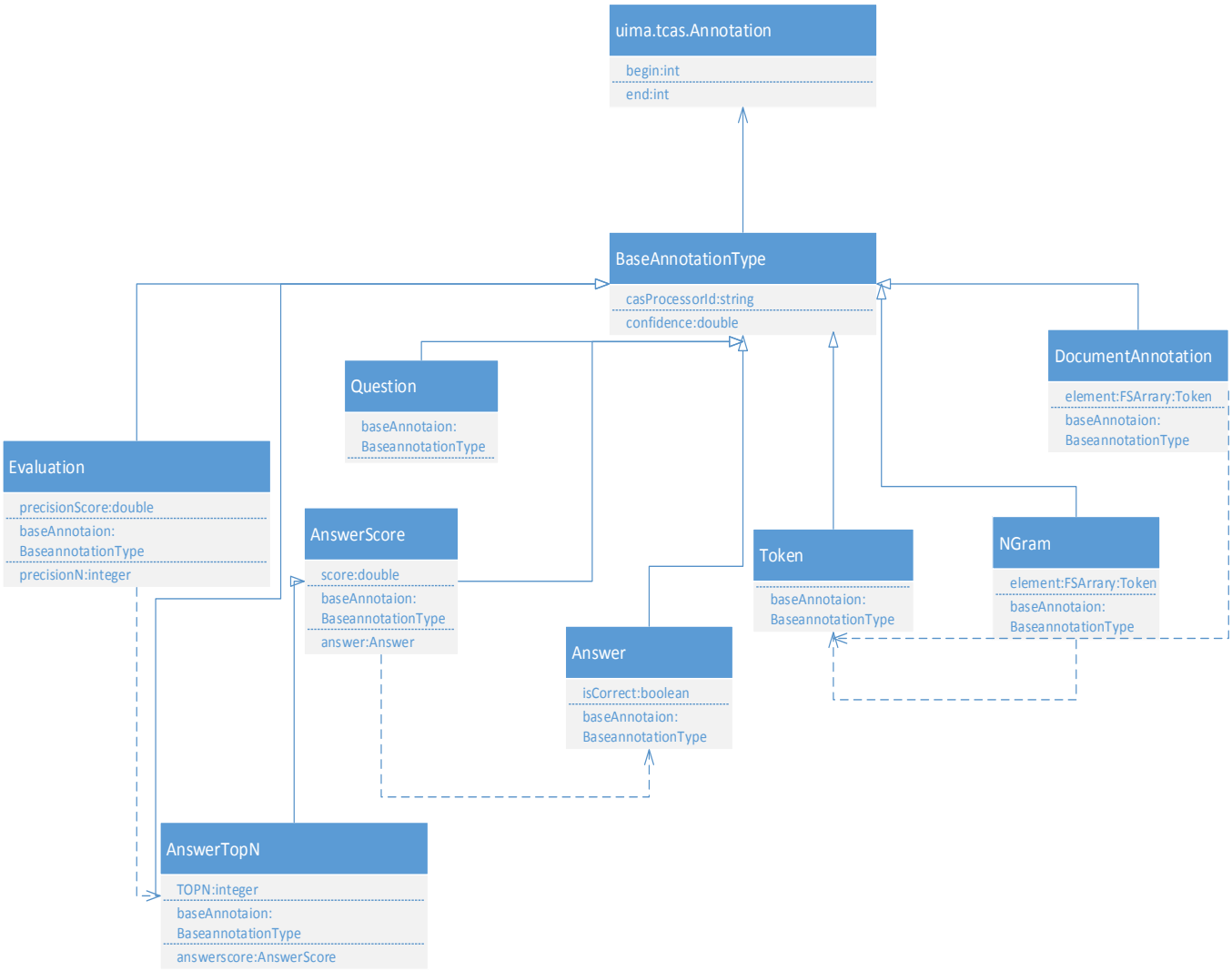


Figure 1: The Diagram of Type Dependence and Extension

The solid and dash line indicate the dependency.(one class depend on the other class)

# 2. Information about each type.

### 2.1 BaseAnnotationType

- **Typename**: uima.BaseAnnotationType.
- **Description**: This kind of type is a base type in the system. It inherits from the uima.tcas.Annotation type. The other two features this type has are casProcessorId and confidence. The other type will be the sub class of this one.
- **Feature**:
    - casProcessorId:
        - **Range Type**: uima.cas.String

        - **Description**: this is a string that we can get where the annotator is produced and provide information for annotator.
    - Confidence:
        - Range Type: uima.tcas.Double
        - Description: Each annotation need a confidence score, so this will provide confidence score for each annotation.
    - Annotation:
        - **Range Type**: uima.tcas.Annotation
        - **Description**: this is a base annotation that we can get the begin and end feature, so we can store the begin and end character of each sentence.

### 2.2 Answer

- **Type name**: uima.Answer.
- **Description**: This kind of type is to show a instance of one answer. Also, it answer whether this kind of answer is correct answer or just a wrong answer.
- **Feature**:
    - BaseAnnotationType
        - **Range Type**: uima.BaseAnnotationType
        - **Description**: This kind of feature is to inherited from the BaseAnnotation to get the feature of casProcessorId and confidence, so we can get the annotation where it is generated.
    - isCorrect:
        - **Range Type**: uima. cas.Boolean
        - **Description**: We can know whether this answer is a correct answer or a wrong answer.

### 2.3 AnswerScore

**Type name**: uima. AnswerScore.

- **Description**: This kind of type is give the calculating score of each answer.
- **Feature**:
    - BaseAnnotationType
        - **Range Type**: uima.BaseAnnotationType
        - **Description**: This kind of feature is to inherited from the BaseAnnotation to get the feature of casProcessorId and confidence, so we can get the annotation where it is generated.
    - Score:
        - **Range Type**: uima. cas.Double
        - **Description**: we get the score of each answer after we run the processor through annotator.

- Answer:
    - **Range Type**: uima.Answer
    - **Description**: this feature is to know the answer sentence that we get our answer score.

### 2.4 DocumentAnnotation

**Type name**: uima. DocumentAnnotation.

- **Description**: This kind of type is to show a instance of document. It stores the token information so it get a token feature.
- **Feature**:
    - BaseAnnotationType
        - **Range Type**: uima.BaseAnnotationType
        - **Description**: This kind of feature is to inherited from the BaseAnnotation to get the feature of casProcessorId and confidence, so we can get the annotation where it is generated.
    - element:
        - **Range Type**: uima.cas.FSArray
        - **Element Type**: token
        - **Description**: we get the token in this document.

### 2.5 Evaluation

**Type name**: uima. Evaluation.

- **Description**: This kind of type is to evaluate the precision of the results
- **Feature**:
    - BaseAnnotationType
        - **Range Type**: uima.BaseAnnotationType
        - **Description**: This kind of feature is to inherited from the BaseAnnotation to get the feature of casProcessorId and confidence, so we can get the annotation where it is generated.
    - PrecistionN:
        - **Range Type**: uima.cas.integer
        - **Description**: The number of correct answer N.
    - PrecistionScore:
        - Range Type: uima.cas.double
        - Description: the result of measuring performance by Precision@N (how many of the top N are correct).

### 2.6 NGram

**Type name**: uima. NGram.

- **Description**: This kind of type is a number of token to form the Ngram type.
- **Feature**:
    - BaseAnnotationType
        - **Range Type**: uima.BaseAnnotationType
        - **Description**: This kind of feature is to inherited from the BaseAnnotation to get the feature of casProcessorId and confidence, so we can get the annotation where it is generated.
    - element:
        - **Range Type**: uima.cas.FSArray
        - **Element Type**: uima.Token
        - **Description**: Each gram is separate by the token in the array. With the length of the array we can get the exact N.

### 2.7 Question

**Type name**: uima. Question.

- **Description**: This kind of type is annotating the question in document so that we can get the answer from the document
- **Feature**:
    - BaseAnnotationType
        - **Range Type**: uima.BaseAnnotationType
        - **Description**: This kind of feature is to inherited from the BaseAnnotation to get the feature of casProcessorId and confidence, so we can get the annotation where it is generated.

### 2.8 Token

**Type name**: uima. Token.

- **Description**: This kind of type is the smallest type in the document. The system will annotate each token span in each question and answer (break on whitespace and punctuation).
- **Feature**:
    - BaseAnnotationType
        - **Range Type**: uima.BaseAnnotationType
        - **Description**: This kind of feature is to inherited from the BaseAnnotation to get the feature of casProcessorId and confidence, so we can get the annotation where it is generated.

### 2.9 AnswerTopN

**Type name**: uima. AnswerTopN.

- **Description**: This kind of type to annotate the TOP N answer . The system will sort the answers according to their scores, and calculate precision at N. So this type will store the top N answer.
- **Feature**:
  - BaseAnnotationType
    - **Range Type**: uima.BaseAnnotationType
    - **Description**: This kind of feature is to inherited from the BaseAnnotation to get the feature of casProcessorId and confidence, so we can get the annotation where it is generated.
  - elementAnswer
    - **Range Type**: uima.cas.FSArray
    - **Element Type**: uima.AnswerScore
    - **Description**: the Array contain the answer score and AnswerScore type contain the Answer sentence information, so we can get the answer and answer score through this array.

# 3. Analysis Design Engine for next use

1. Element Annotation: In this step , the annotation parse the document and split each sentence. Get the question and answer type.
2. Token Annotation : In this step , the annotation parse the question and answer to get the token. It token is separated by space or some punctuation.
3. NGram Annotation: This step, the token generated from the previous step will fill in the 1-Gram, 2-Gram, 3-Gram. Thus, we get the N Gram of each answer and question.
4. Scoring Annotation: In this step, the processor will take the NGram as the input and get the answerScore of each answer, the formula is probably like this questionN-Grams found / total ansewr N-Grams. We get the ground truth of each sentence and now we can calculate the TOP N answer and get the precision at top N.
5. Evaluation Annotation: This step, we use the score result as the input and sort each answer and get the TopN answer , so we can caclculate the precision at Top N.