

The Zoning Annotator

Annotations based on document location

Introduction

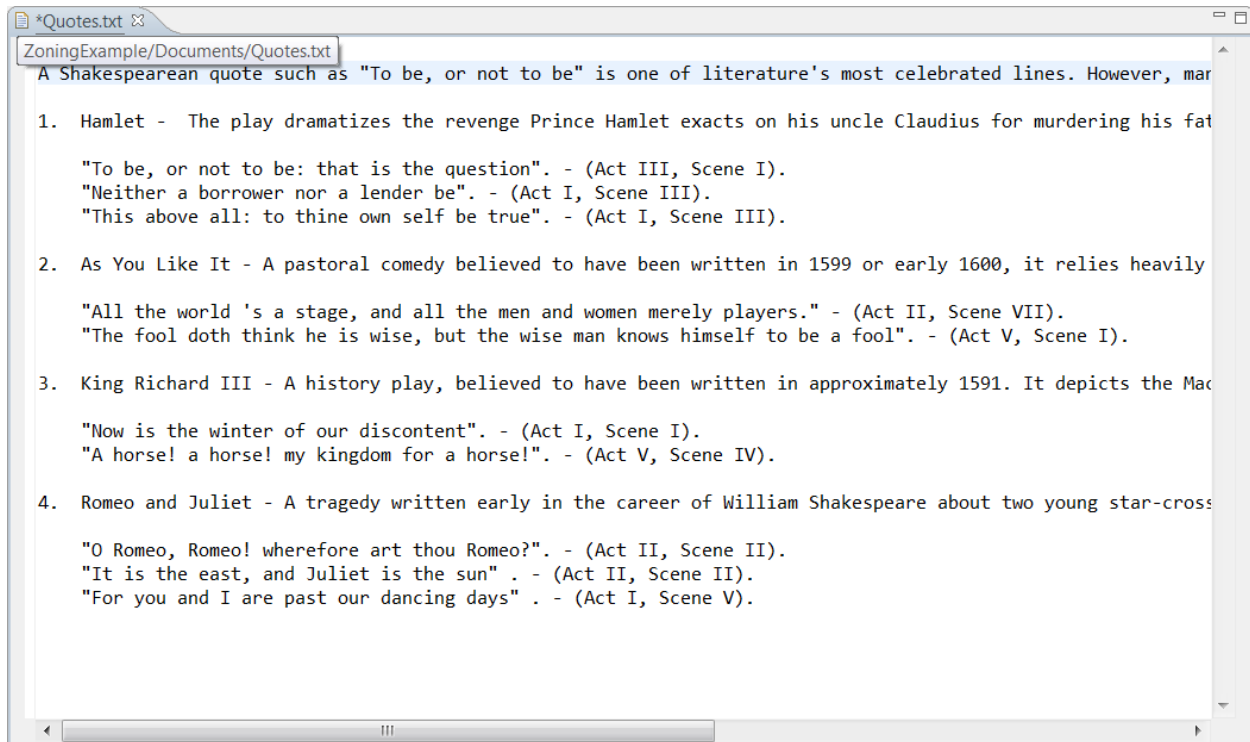
Most customer engagements involving IBM Content Analytics with Enterprise Search (ICAwES) require at least some level of customized text analysis in order to extract the entities, facts, concepts, abstractions, and relationships necessary to provide something close to optimal value for the customer's business case.

IBM uniquely offers the Content Analytic Studio (CA Studio) as a powerful tool for creating text analysis models quickly, easily, and without the need to learn difficult (for many of us) query language or Java programming code. In most cases, virtually all of these entities, facts, concepts, abstractions, and relationships of interest to the customer can be identified and extracted using annotators built with CA Studio.

However, as with all tools, CA Studio does not provide all the functionality one would desire. A good example of this is zoning - the process of creating an annotation based not only on its lexical properties and pattern, but also on its positioning within a defined area of text. Fortunately, there is a way to do this, without having to write any customized code.

The Problem, Illustrated

To get a better idea what kind of problem we are trying to tackle here, consider the following document:



In this example, imagine a situation where for a customer we are interested in extracting the Shakespearean quotes that are listed for each play. It is important to the customer that we not only identify each quote but also we must record which play it came from. As a human reader we get this information easily from the quote's position within the numbered sections of the document rather than from some lexical property or pattern that could be simply expressed in CA Studio. Obviously, we could probably think of a way to do this for this particular document. But imagine that there are many different documents that contain many different types of sections containing potentially interesting information where the positional context is important. This is, in fact, a situation we often run into and may be familiar to you if you have done a lot of text modeling for customers. Until now, we have had no effective way of doing this with CA Studio without writing a custom annotator.

The Solution

Fortunately, we now have a custom annotator that allows us to identify and extract only entities that appear in a certain section of a document. For example, In the text containing the Shakespearean quotes above, we can identify "A horse! a horse! my kingdom for a horse!" as a quote appearing in the section about the play King Richard III. Furthermore, it has been written as a generic annotator, so that it can be easily adapted to be used with any annotation type in any section of a document. This adaption requires only some simple configuration work and is done completely within CA Studio. No programming of any kind is required.

To start, as is always the case with text analysis modeling, it is necessary to have a clear understanding of what you are trying to model. In the example we use here, we want to extract quotes from the document. We will annotate each of these as a **"Quote"**, and the annotation will include a **"source"** feature that records which play the quote comes from. This feature can then be used as any other – in a selection rule to create a new annotation, mapped to a facet in the ICA Search Application or Content Miner, or be exported to a database.

The method for doing this is actually very simple. We will start with a high level description, and then go through a detailed, step by step procedure, with screen prints.

In order to establish a link between an entity and the zone (or section) of a document in which it is found, we must do four things:

1) **Define the annotation that identifies section (zone) headings**: The first step is to determine where a section or zone begins. This is usually a title, or heading and is often very easy to annotate. In the Shakespeare example above, the sections of interest all have the form *"Number. some title text –"*. We can easily create a rule to annotate these in a document.

2) **Define the annotation that identifies the text items of interest in the sections (zone)**: Next, we need to be able to successfully identify and annotate the item(s) of interest that occur within the sections. In the example we are using, these are the various quotes that are listed in the sections and they all follow a defined format. Again, we can easily create a rule to annotate these in a document.

3) **Define the feature to hold the section name that will be applied to the identified item(s)**: The purpose of this step is to define a new feature as part of the annotated text of interest within the document sections. This feature is the connection or relationship, so to speak, between the entity and the section. In our example, all instances of quotes within the numbered sections of the document will have the feature "source" which will contain the name of the play (i.e. the section or zone) that the quote is in. Thus the link between these entities (our quotes) and this specific section or zone of the document in which they appear (the play) has been established. In the first instance we just need to create the feature with a default constant value, the custom Where annotator will update it with the correct value.

4) **Configure the Where annotator to update the feature with the correct section values**: Finally we tell the Where annotator the name of our sections headings annotation (from step 1), the name of the

annotations we want to be identified within the sections (from step 2) and the name of the feature in those annotations to be updated with the section name (from step 3).

The result of doing this on our example document is shown below with a screenshot from CA Studio. The quote "A horse! a horse! my kingdom for a horse!" has been annotated and the highlight shows the the **source** feature of the annotation with the correct play name in it taken from the section of the document the quote is in.

"This above all: to thine own self be true". - (Act I, Scene III).

2. As You Like It - A pastoral comedy believed to have been written in 1599 or early 1600, it relies heavily on mistaken identity and desperate romance to induce humour.

"All the world 's a stage, and all the men and women merely players." - (Act II, Scene VII).

"The fool doth think he is wise, but the wise man knows himself to be a fool". - (Act V, Scene I).

3. King Richard III - A history play, believed to have been written in approximately 1591. It depicts the Machiavellian rise to power and subsequent short reign of Richard III of England.

"Now is the winter of our discontent". - (Act I, Scene I).

"A horse! a horse! my kingdom for a horse! ". - (Act V, Scene IV).

4. Romeo and Juliet - A tragedy about two young star-crossed lovers. In Shakespeare about two young families.

"O Romeo, Romeo! where art thou? Wherefore art thou Romeo?" - (Act I, Scene I).

"It is the east, and Juliet is the sun." - (Act I, Scene II).

"For you and I are past our dancing days". - (Act I, Scene V).

Annotation details:

- Covered text = "A horse! a horse! my kingdom for a horse! "
- Rule identifier = 35E15A05B4762DA96C6AA1265300F833
- source = King Richard III

Property	Value
Covered text	@ "A horse! a horse! my kingdom for a horse! "
Rule identifier	@ 35E15A05B4762DA96C6AA1265300F833
source	@ King Richard III
Type	com.ibm.ecmuk.en.Quote

Creating Zone-Based Annotations. Step by Step Guide

In the example we use here (and included in the Where project), we want to extract quotes from Shakespearean plays out of the document and include the name of the play the quote comes from. To do this we will follow our four step method.

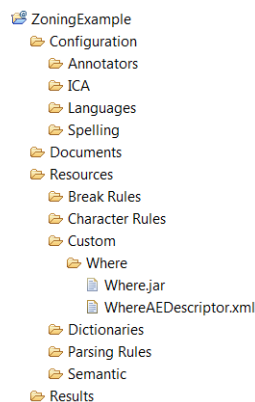
- We will annotate the start of each section about a specific play with an annotation called “**Play**”
- We will annotate each quote that appears in a section with an annotation called “**Quote**”
- We will add a feature to the **Quote** annotation called “**source**” to record which play the quote comes from. Initially this will be set to a constant value “unknown”.
- We will configure the Where annotator to process **Play** and **Quote** annotations and update the **source** feature in each **Quote** annotation.

1. Prepare your workspace by adding custom annotator file.

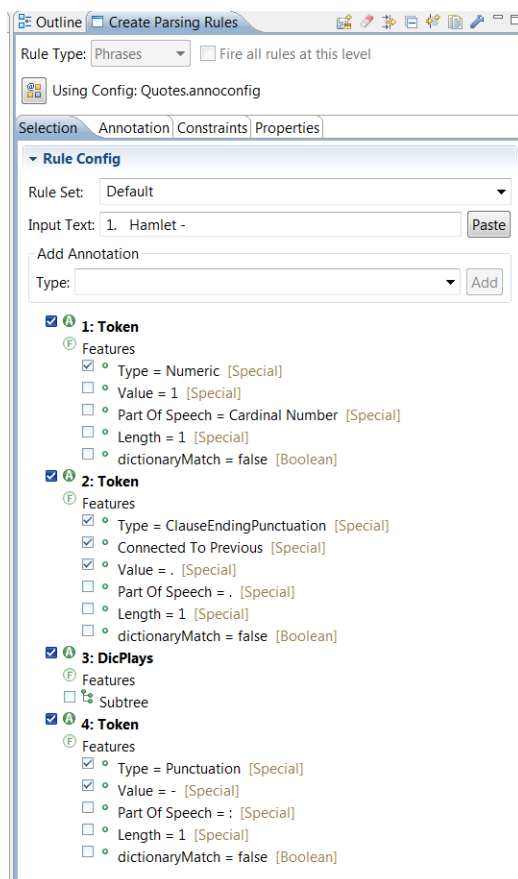
The easiest way to do this is to first import the **Where** project into your workspace. In the **Where** project, in the **Resources\Custom\Where** folder, you will find two files that are required:

- **Where.jar**: This is the compiled java code for the Where annotator
 - **WhereAEDescriptor.xml**: This file tells CA Studio what the Where annotator does and how it is configured.
- a) In your project add a Folder to the Resources Folder to hold the custom annotator (ie. the Where annotator). Typically, this would be something like Resources\Custom\Where.
 - b) Add the files **Where.jar** and **WhereAEDescriptor.xml** from the **Where** project to this folder.
 - c) Alternatively, you can simply copy and paste the Custom folder from the imported **Where** project to the Resources folder of your own project. This will copy everything at once.

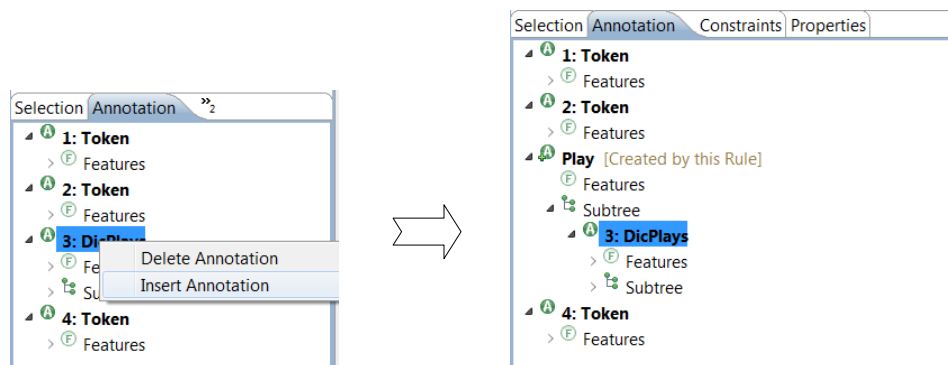
Following the above steps on a project named **ZoningExample** the result should look something like this:



2. Create the Section heading annotator: The first step is to determine where a section or zone begins. In our example, we created a dictionary **"DicPlays"** that contains a list of the titles of Shakespeare's plays and used it in a phrase rule that identifies the section heading for each play when a play name is seen following a numeric and before a **"-"** character. The rule is in a parsing rules database called **"QuoteRules"**. This rule creates the annotation **"Play"**. Here, we use the file **Quotes.anno**, (which is also included in the Where annotator) to create and test this rule:



Note that although the annotation uses the selection rule “*Numeric. title* –” it only annotates the play title itself. This is done on the Annotation tab by selecting just the **DicPlays** token to insert the annotation on.

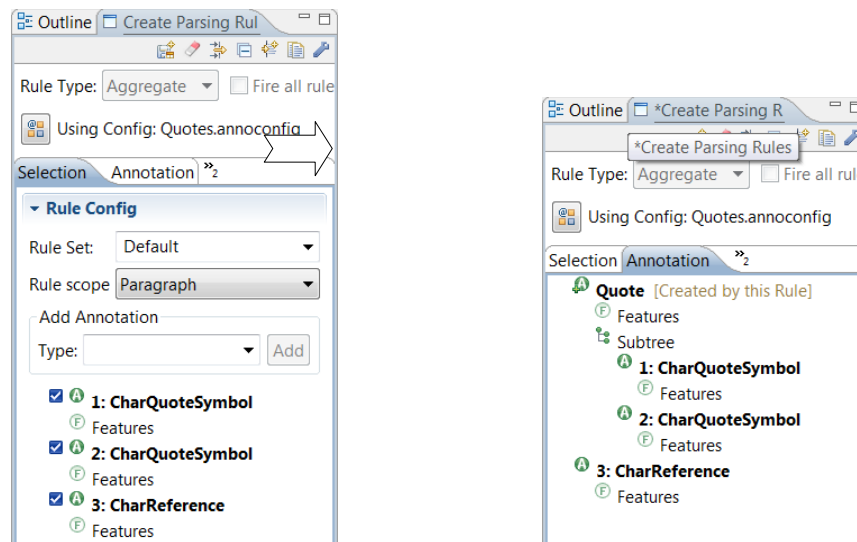


The result of applying our annotation rule on the example document is shown below.

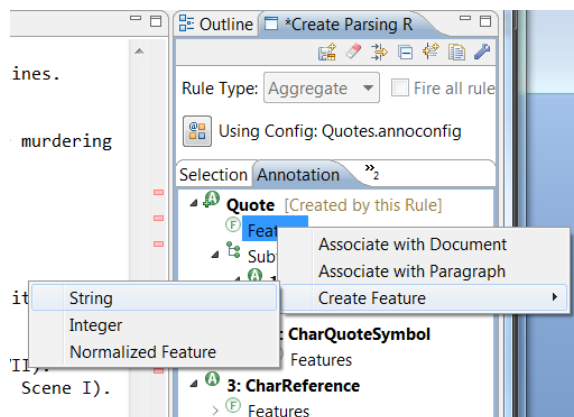
3. Define the text to be identified in the section or zone:

Next, we need to identify the text within the section we want to extract. In our example these are the quotes. We first created two character rules in a character rules database called “**CharQuotes**”, one rule identifies double quote symbols and the other references of the form “(Act X, Scene Y)”. In our **QuoteRules** parsing rules database we then created an Aggregate rule to identify text in quotes

followed by a reference which are then annotated as **"Quote"**. Again we only annotate the relevant part of the text as a **Quote** (the bit between the two quote marks), not the whole selection rule.

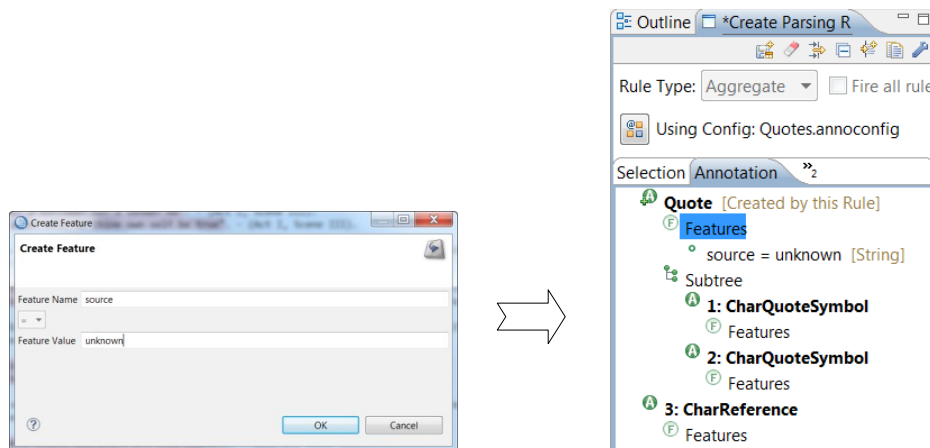


4. To the new **"Quotes"** annotation, we create a new feature, called **"source"**:
 - a. Go to the Right click on the feature entry annotation and choose **"Create Feature"** - **"String"**

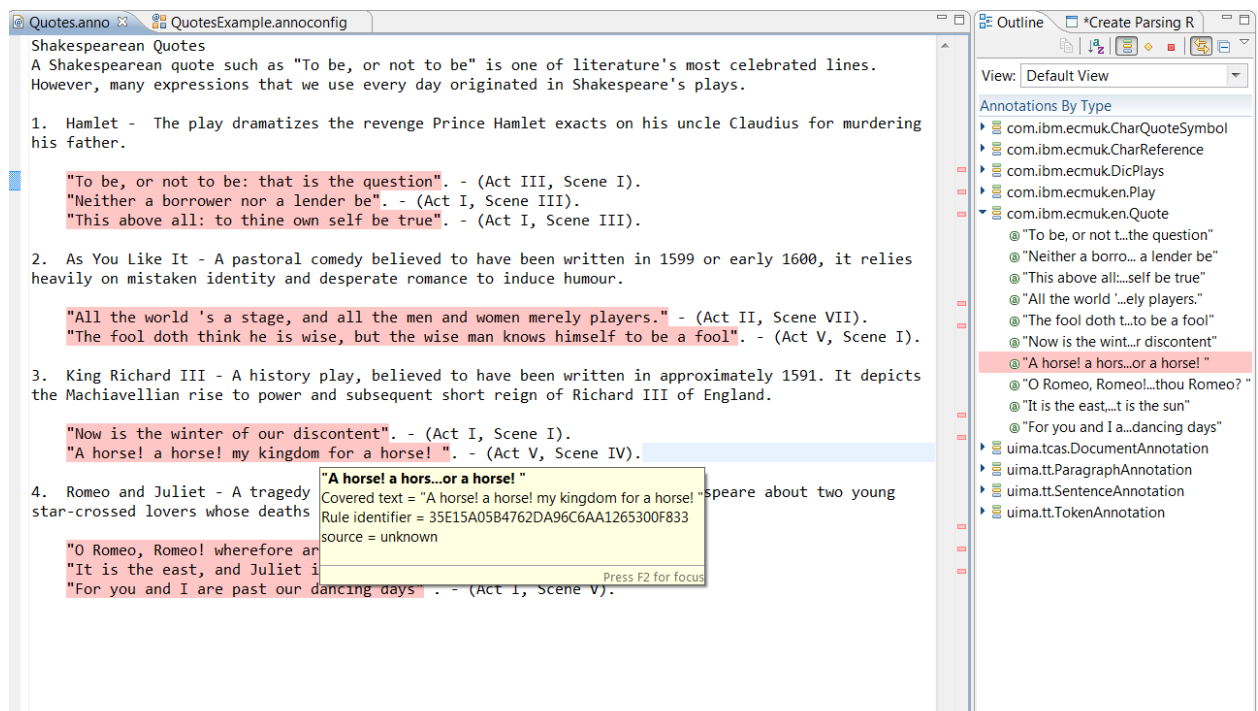


- b. Type in the feature name **"source"**¹. You will also need to enter a value. here, you can just enter a space or any string, in this case we type **"unknown"**. When the pipeline is run, the Where annotator will actually update this to the correct value based on the extracted **"Play"** annotation defined in the previous step. Then save the rule and rebuild the database.

¹ Features must start with a lower case character, so even if you type in **Source**, it will appear as **source**.

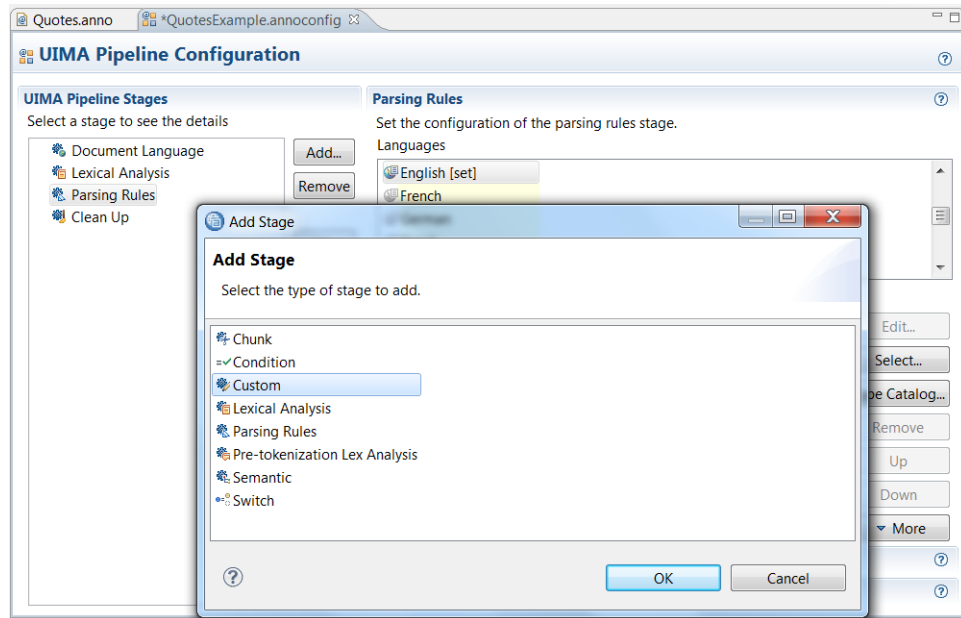


Re-analyse the example Quotes document and this will show all quotes as having been annotated with the source feature on all Quote annotations set to “unknown”.



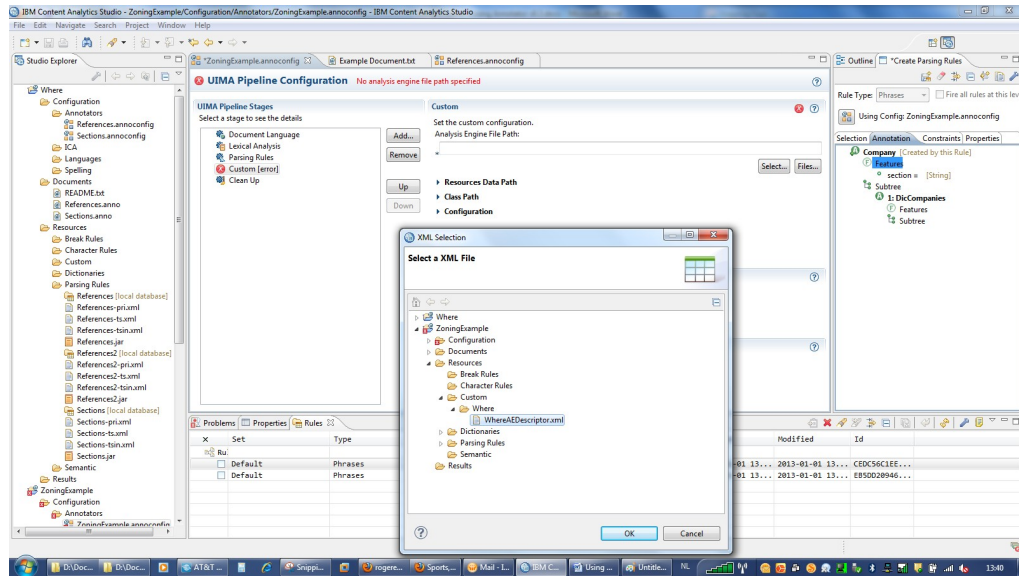
- Now we will configure the pipeline to include the custom Where annotator. Open the pipeline configuration file. In our example, it is called **QuoteExample.annoconfig**

- a. In the **Pipeline Stages** pane, click on **Parsing Rules**, then click on the **Add** button, then choose **Custom**²:

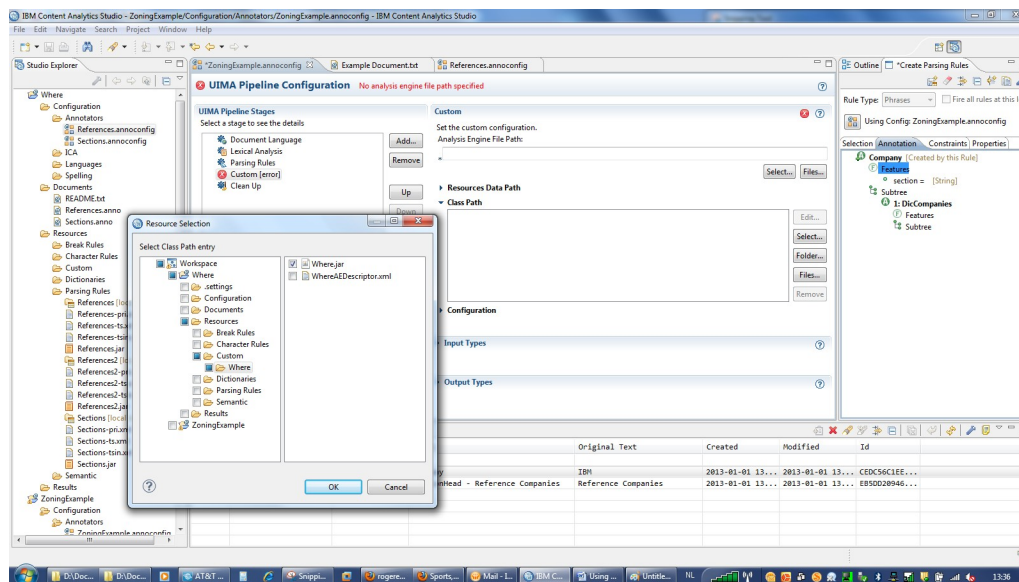


- b. Now, click on the **Custom** stage, and, in the Custom pane, locate the **Analysis Engine File Path** setting, located at the top of the page. Click on the **Select** button underneath it and navigate to the location where you stored the **WhereAEDescriptor.xml** file (in this example, `Resources\Custom\Where`).

² Clicking on Parsing Rules first will ensure that the custom stage comes after the Parsing Rules stage. This is important as the Parsing Rules database we defined generates annotations that are needed in the Custom Annotator.

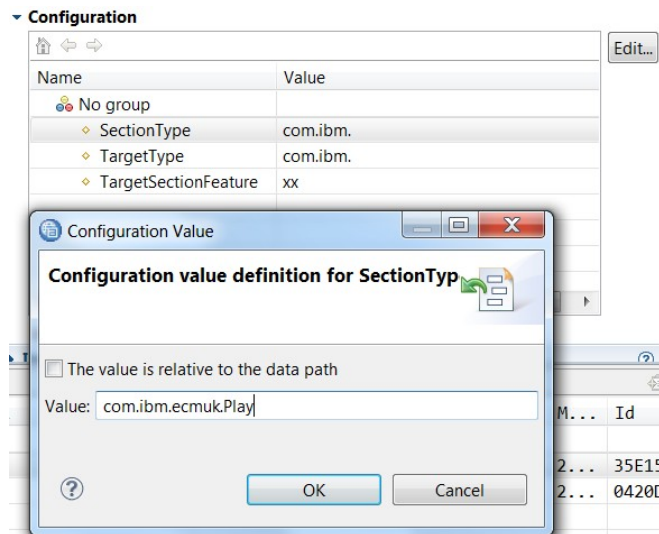


- c. Next, click on **Class Path** to expand its pane. Then click on the **Select** button and navigate to the location where you placed the Where.jar file (e.g. Resources\Custom\Where\Where.jar). Select only the **Where.jar** file.



- d. Now, expand the Configuration Pane. There are three configuration variables you must set here:

- i. **SectionType**: This is the UIMA name of the annotation that marks the start of a section of a document. in this example, this is the UIMA name for the **Play** annotation - **com.ibm.ecmuk.Play**³. Click on the Value cell and enter the UIMA name:

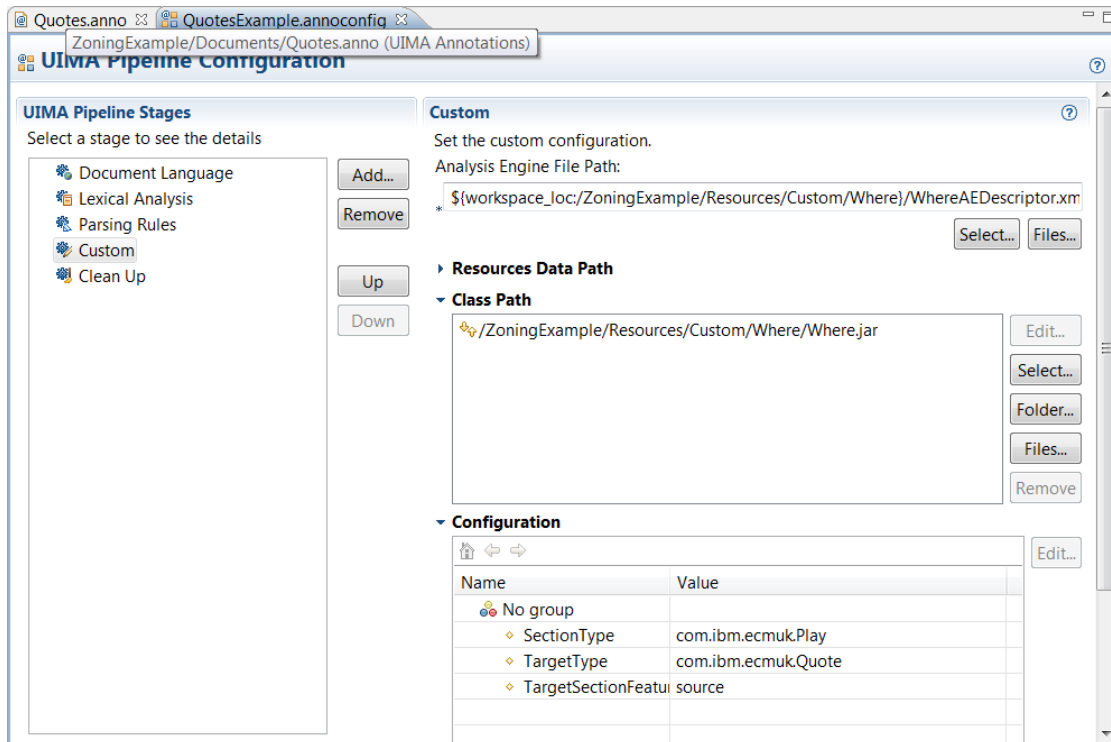


- ii. **TargetType** : The UIMA name of the annotation that you want the Where annotator to update with positional information. Here, this is the UIMA name for the **Quote** annotator - **com.ibm.ecmuk.Quote** Click on the value cell and enter the UIMA name:

- iii. **TargetSectionFeature**: The name of the feature you created in the TargetType that will be filled in with the section name by the Where annotator. In this example, we used **source**. Click on the Value cell and enter the feature name:

- e. When you are finished, the Custom annotator configuration page should look something like the screen shot below. Make sure to save the configuration pipeline.

³ These can be long names, as in this case. An easy way to copy and paste this name is to open the rule that creates the annotation, right click on it and choose "Edit Annotation Name". You can then simply copy the string.



- To test the annotator, we can re-analyse the example Quotes document. All of our existing **Quote** annotations should have had their source feature updated from a value of “unknown” to that of the section of the document they are in, which in this case is the name of the play.

Shakespearean Quotes
 A Shakespearean quote such as "To be, or not to be" is one of literature's most celebrated lines. However, many expressions that we use every day originated in Shakespeare's plays.

1. Hamlet - The play dramatizes the revenge Prince Hamlet exacts on his uncle Claudius for murdering his father.
 - "To be, or not to be: that is the question". - (Act III, Scene I).
 - "Neither a borrower nor a lender be". - (Act I, Scene III).
 - "This above all: to thine own self be true". - (Act I, Scene III).
2. As You Like It - A pastoral comedy believed to have been written in 1599 or early 1600, it relies heavily on mistaken identity and desperate romance to induce humour.
 - "All the world 's a stage, and all the men and women merely players." - (Act II, Scene VII).
 - "The fool doth think he is wise, but the wise man knows himself to be a fool". - (Act V, Scene I).
3. King Richard III - A history play, believed to have been written in approximately 1591. It depicts the Machiavellian rise to power and subsequent short reign of Richard III of England.
 - "Now is the winter of our discontent". - (Act I, Scene I).
 - "A horse! a horse! my kingdom for a horse! ". - (Act V, Scene IV).
4. Romeo and Juliet - A star-crossed lovers whose
 - "O Romeo, Romeo! where
 - "It is the east, and J
 - "For you and I are past our dancing days". - (Act I, Scene V).

Annotations By Type

- com.ibm.ecmuk.CharQuoteSymbol
- com.ibm.ecmuk.CharReference
- com.ibm.ecmuk.DicPlays
- com.ibm.ecmuk.Play
- com.ibm.ecmuk.Quote
 - @ "To be, or not t...the question"
 - @ "Neither a borro... a lender be"
 - @ "This above all...self be true"
 - @ "All the world '...ely players."
 - @ "The fool doth t...to be a fool"
 - @ "Now is the wint...r discontent"
 - @ "A horse! a hors...or a horse!"
 - @ "O Romeo, Romeo!...thou Romeo?"
 - @ "It is the east...t is the sun"
 - @ "For you and I a...dancing days"
- uima.tcas.DocumentAnnotation
- uima.tt.ParagraphAnnotation
- uima.tt.SentenceAnnotation
- uima.tt.TokenAnnotation

Property	Value
Covered text	@ "A horse! a horse! my kingdom for a horse! "
Rule identifier	@ 35E15A05B4762DA96C6AA1265300F833
source	@ King Richard III
Type	com.ibm.ecmuk.Quote

Advanced Example

In the Shakespearean quotes example used in the step by step guide the Where annotator was the final step in our processing pipeline that added positional information to previously created annotations. There are cases where to meet a customer's requirements you need to create annotations based on the positional information not just add it in at the end. This requires running parsing rules after the after positional information has been added by the custom stage.

As an example consider the following text that is included in the Where project as the document **XReferences.anno**.

1. Content

On Wednesday, IBM shook up its web conferencing strategy adding a hosted option by acquiring WebDialogs.

IBM today announced it has completed its acquisition of Cognos.

While it is true that one mega-deal, Novartis (NYSE: NVS) bid to buy Alcon (NYSE: ACL) for \$39.0 billion, the largest medical device deal ever announced, contributes significantly to the overall total, even without it, the remaining figures are impressive.

Roll Call Group, the publisher of Washington's Roll Call newspaper, said on Friday that it has acquired Capitol Advantage, an Internet lobbying services provider, for \$43 million.

2. References

IBM - Armonk, NY
WebDialogs - Billerica, Massachusetts
Cognos - Ottawa, Ontario
Alcon - Hünenberg, Switzerland
Capitol Advantage - Fairfax, VA





3. Contacts

IBM - J. Doe
WebDialogs - B. Smith
Cognos - H. Reinhart

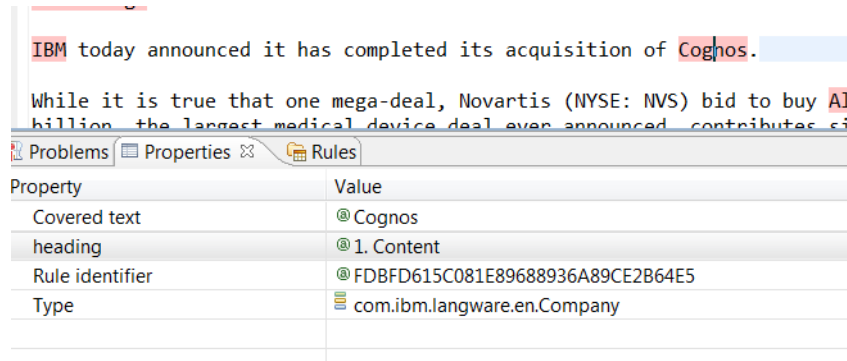
The document has three sections "Content" and "References" and "Contacts", and there are company names appearing in all sections. Our customer is only interested in extracting companies that are listed in the "References" section of the document.

As before we can use our four step method to add positional information to annotations that identify companies. In this case the specific four steps used in the example are

1. We define an annotation called "**Section**" that identifies the three section headings.
2. We define an annotation called "**Company**" that identifies all companies in the document
3. We add a feature called "**heading**" to the "**Company**" annotation.
4. We configure the Where annotator with the correct values from steps 1,2 and 3 as follows:

Configuration	
  	
 Edit...	
Name	Value
No group	
SectionType	com.ibm.langware.Section
TargetType	com.ibm.langware.Company
TargetSectionFeature	heading

This successfully creates all the **Company** annotations with their positional feature **heading** updated. For example:



The screenshot shows a text editor with the following text: "IBM today announced it has completed its acquisition of Cognos. While it is true that one mega-deal, Novartis (NYSE: NVS) bid to buy A1 billion, the largest medical device deal ever announced, contributes si". Below the text is a table with two columns: Property and Value.

Property	Value
Covered text	@ Cognos
heading	@ 1. Content
Rule identifier	@ FDBFD615C081E89688936A89CE2B64E5
Type	com.ibm.langware.en.Company

To meet our requirement of specifically identifying those companies in the References section we need to do an extra step

5. We can create a rule that will create a "**ReferenceCompany**" annotation for only those companies that have a **section** feature with the value "2. References". The rule to do this is very easy to make. It is important however, that this rule be fired only after the custom annotator has run, so that the feature has been populated with the correct value. It is necessary, then, to define the rule in a separate second rules database that is run after the custom annotator stage in the pipeline.



To summarize:

- a. Create a new rules database

- b. Create a new parsing rule stage in the UIMA pipeline. Make sure it runs after the Custom stage, and add the new rules database for it.
- c. In the new rules database, create a new rule in which only selects companies that have the value "2. References" in the **section** feature.
- d. Create an annotation for this rule called **ReferenceCompany**. Save the rule and rebuild it.

The Annotation **ReferenceCompany** now appears in the Outline pane. As you can see, it only contains companies that appear in the Reference Company section.

The screenshot shows the UIMA IDE interface. The main editor displays a document with three sections: 1. Content, 2. References, and 3. Contacts. The 'References' section is highlighted, showing a list of companies: IBM, WebDialogs, Cognos, Alcon, and Capitol Advantage. The Outline pane on the right shows the 'Annotations By Type' list, which includes 'com.ibm.langware.en.ReferenceCompany' as a new annotation type.

TargetSectionFeature: heading

1. Content

On Wednesday, IBM shook up its web conferencing strategy adding a hosted option by acquiring WebDialogs.

IBM today announced it has completed its acquisition of Cognos.

While it is true that one mega-deal, Novartis (NYSE: NVS) bid to buy Alcon (NYSE: ACL) for \$39.0 billion, the largest medical device deal ever announced, contributes significantly to the overall total, even without it, the remaining figures are impressive.

Roll Call Group, the publisher of Washington's Roll Call newspaper, said on Friday that it has acquired Capitol Advantage, an Internet lobbying services provider, for \$43 million.

2. References

IBM - Armonk, NY
WebDialogs - Billerica, Massachusetts
Cognos - Ottawa, Ontario
Alcon - Hünenberg, Switzerland
Capitol Advantage - Fairfax, VA

3. Contacts

IBM - J. Doe
WebDialogs - B. Smith
Cognos - H. Reinhart

Outline pane: Annotations By Type

- com.ibm.langware.DicCompanies
- com.ibm.langware.en.Company
- com.ibm.langware.en.ReferenceCompany**
- com.ibm.langware.en.Section
- uima.tcas.DocumentAnnotation
- uima.tt.ParagraphAnnotation
- uima.tt.SentenceAnnotation
- uima.tt.TokenAnnotation

Appendix A. Limitations

As is the case with most techniques, the Where custom annotator does have certain limitations:

1. The text of each “SectionType” annotations that identify the start of sections really needs to be unique in the document for the Where annotator to add value. If the sections are delimited by the same repetitive text or worst still by no text at all (bullets, page numbers, paragraph counts etc) then the Where annotator doesn’t help.
2. The annotator can only add positional information to instances of one named annotation defined as the “TargetType”. If you need positional information added to multiple annotation types then you can either:
 - a. In your model have an intermediate annotation type that can be subsequently refined with further rules to produce your multiple final types. Add the positional information to the intermediate type using the Where annotator and have subsequent rules stages to create the final types from the intermediate type. In these rules copy the positional feature up into the final types from the sub intermediate type. This is not always possible!
 - b. Add multiple custom stages, one per target type that needs positional information.
3. Creating the “SectionType” annotations with sufficient precision and recall to identify the start of sections is not always straightforward (or possible). It’s difficult to unambiguously identify section headings if they only contain the same tokens and patterns that are used in the main body of the text. Using the regex annotator over `uima.tcas.DocumentAnnotation` can often help here.