Unstructured Information Processing with Apache UIMA

NYC Search and Discovery Meetup

Pablo Ariel Duboue, PhD

IBM TJ Watson Research Center
19 Skyline Dr.
Hawthorne, NY 10603

February 24th, 2010

Outline

Outline

What is UIMA

- UIMA is a framework, a means to integrate text or other unstructured information analytics.
- Reference implementations available for Java, C++ and others.
- An Open Source project under the umbrella of the Apache Foundation.

Analytics Frameworks

- Find all telephone numbers in running text
  - `(((\([0-9]{3}\))|[0-9]{3})-?[0-9]{3}-?[0-9]{4}`
- Nice but...
  - How are you going to feed this further processing?
  - What about finding non-standard proper names in text?
  - Acquiring technology from external vendors, free software projects, etc?

In-line Annotations

- Modify text to include annotations
  - This/DET happy/ADJ puppy/N
- It gets very messy very quickly
  - (S (NP (This/DET happy/ADJ puppy/N) (VP eats/V (NP (the/DET bone/N)))
- Annotations can easily cross boundaries of other annotations
  - He said **<confidential>**the project can't go own. The funding is lacking.**</confidential>**

Standoff Annotations

- Standoff annotations
  - Do not modify the text
  - Keep the annotations as offsets within the original text
- Most analytics frameworks support standoff annotations.
- UIMA is built with standoff annotations at its core.
- Example:

```
He said the project can't go own.  The funding is lacking.
---------------------------------------------------------
0123456789012345678901235678901234567890123456789012345678
```

  - Sentence Annotation: 0-33, 36-58.
  - Confidential Annotation: 8-58.

Type Systems

- Key to integrating analytic packages developed by independent vendors.
- Clear metadata about
  - Expected Inputs
    - Tokens, sentences, proper names, etc
  - Produced Outputs
    - Parse trees, opinions, etc
- The framework creates an unified typesystem for a given set of annotators being run.

Many frameworks

- Besides UIMA
  - `http://incubator.apache.org/uima`
- LingPipe
  - `http://alias-i.com/lingpipe/`
- Gate
  - `http://gate.ac.uk/`

UIMA Advantages

- Apache Licensed
- Enterprise-ready code quality
- Demonstrated scalability
- Developed by experts in building frameworks
  - Not domain (e.g., NLP) experts
- Interoperable (C++, Java, others)

About The Speaker

- PhD in CS, Columbia University (NY)
  - Natural Language Generation
- Joined IBM Research in 2005
  - Worked in
    - Question Answering
    - Expert Search
    - DeepQA (Jeopardy!)
- Recently joined the UIMA group
  - `mailto:pablo.duboue@gmail.compablo`

Outline

UIMA Concepts

- Common Annotation Structure or CAS
  - Subject of Analysis (SofA or View)
  - JCas
- Feature Structures
  - Annotations
- Indices and Iterators
- Analysis Engines (AEs)
  - AEs descriptors

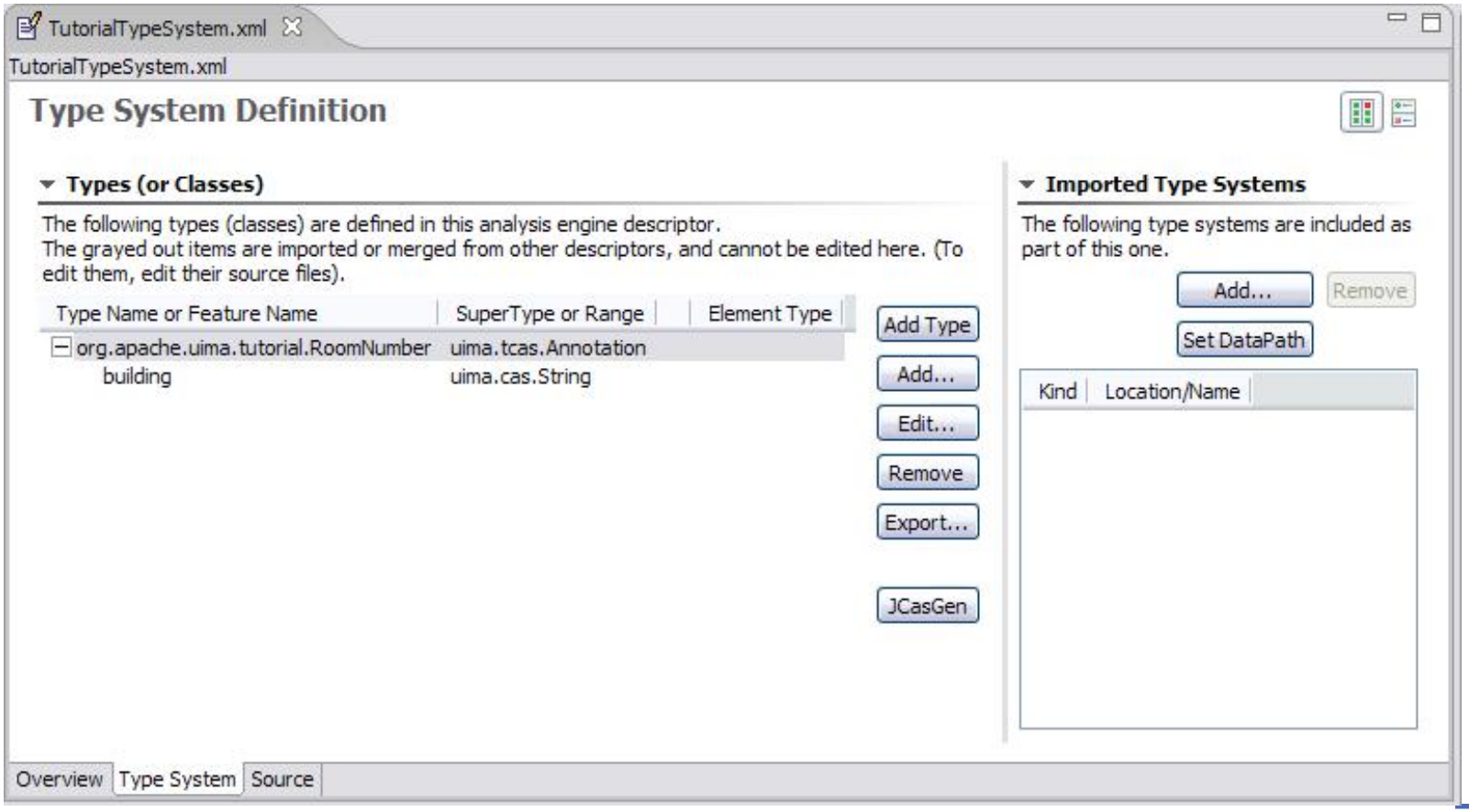
Room annotator

- From the UIMA tutorial, write an Analysis Engine that identifies room numbers in text.

  Yorktown patterns: 20-001, 31-206, 04-123 (Regular Expression Pattern: [0-9][0-9]-[0-2][0-9][0-9])
  Hawthorne patterns: GN-K35, 1S-L07, 4N-B21 (Regular Expression Pattern: [G1-4][NS]-[A-Z][0-9])

- Steps:
  1. Define the CAS types that the annotator will use.
  2. Generate the Java classes for these types.
  3. Write the actual annotator Java code.
  4. Create the Analysis Engine descriptor.
  5. Test the annotator.

Editing a Type System



The XML descriptor

```
<?xml version="1.0" encoding="UTF-8" ?>
  <typeSystemDescription xmlns="http://uima.apache.org/resourceSpecifier">
    <name>TutorialTypeSystem</name>
    <description>Type System Definition for the tutorial examples -
        as of Exercise 1</description>
    <vendor>Apache Software Foundation</vendor>
    <version>1.0</version>
    <types>
      <typeDescription>
        <name>org.apache.uima.tutorial.RoomNumber</name>
        <description></description>
        <supertypeName>uima.tcas.Annotation</supertypeName>
        <features>
          <featureDescription>
            <name>building</name>
            <description>Building containing this room</description>
            <rangeTypeName>uima.cas.String</rangeTypeName>
          </featureDescription>
        </features>
      </typeDescription>
    </types>
  </typeSystemDescription>
```

The AE code

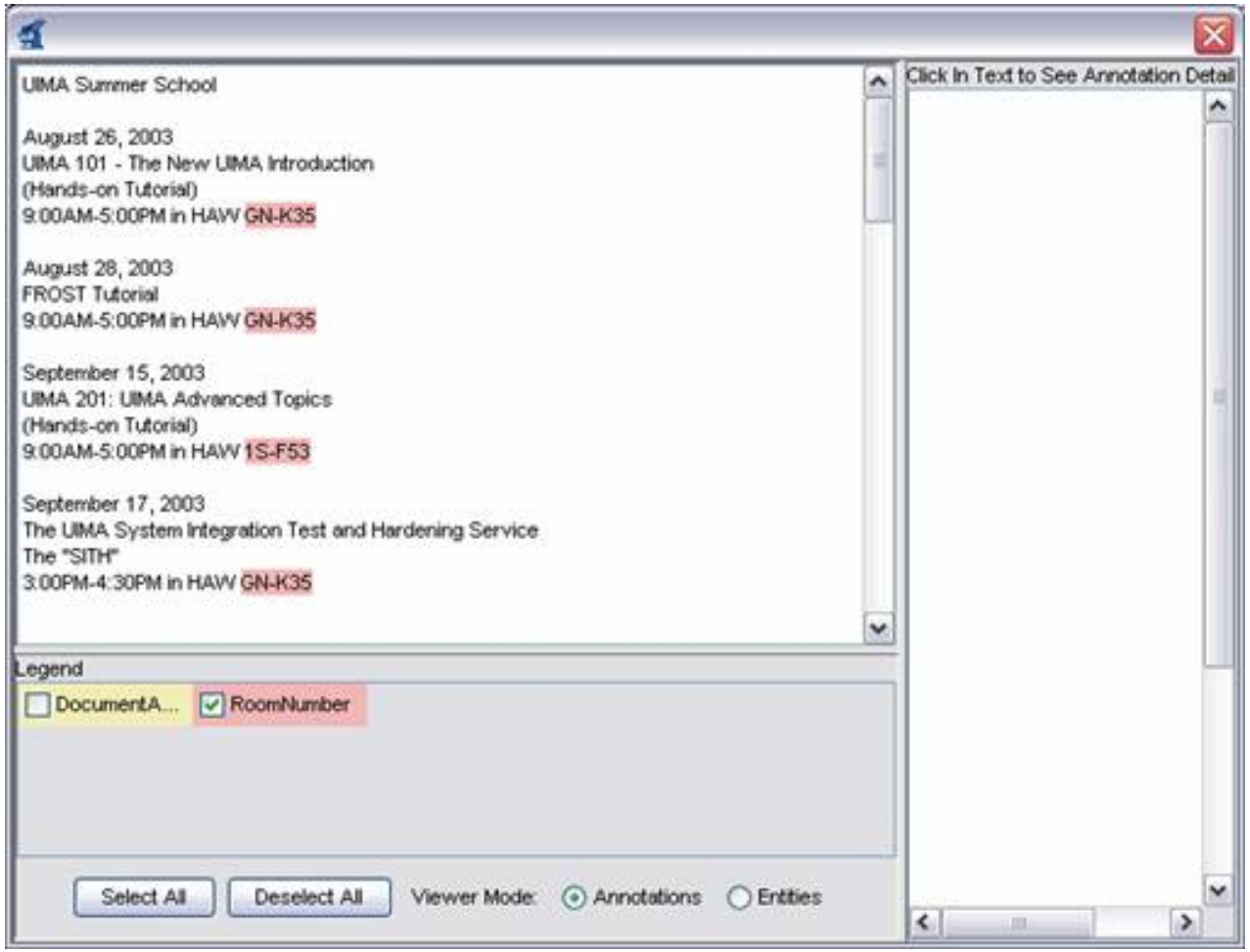
```
package org.apache.uima.tutorial.ex1;

import java.util.regex.Matcher;
import java.util.regex.Pattern;

import org.apache.uima.analysis_component.JCasAnnotator_ImplBase;
import org.apache.uima.jcas.JCas;
import org.apache.uima.tutorial.RoomNumber;

/**
 * Example annotator that detects room numbers using
 * Java 1.4 regular expressions.
 */
public class RoomNumberAnnotator extends JCasAnnotator_ImplBase {
  private Pattern mYorktownPattern =
        Pattern.compile("\\b[0-4]\\d-[0-2]\\d\\d\\b");

  private Pattern mHawthornePattern =
        Pattern.compile("\\b[G1-4][NS]-[A-Z]\\d\\d\\b");

  public void process(JCas aJCas) {
    // next slide
  }
}
```

The AE code (cont.)

```
public void process(JCas aJCas) {
  // get document text
  String docText = aJCas.getDocumentText();
  // search for Yorktown room numbers
  Matcher matcher = mYorktownPattern.matcher(docText);
  int pos = 0;
  while (matcher.find(pos)) {
    // found one - create annotation
    RoomNumber annotation = new RoomNumber(aJCas);
    annotation.setBegin(matcher.start());
    annotation.setEnd(matcher.end());
    annotation.setBuilding("Yorktown");
    annotation.addToIndexes();
    pos = matcher.end();
  }
  // search for Hawthorne room numbers
  // ..
}
```

UIMA Document Analyzer

UIMA Document Analyzer (cont)



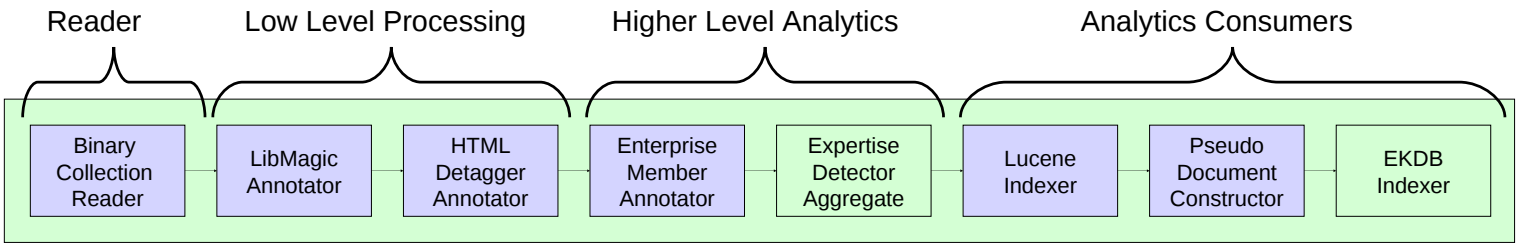
Outline

An Example

- TREC 2006 enterprise track
- Search for experts in W3C Website
  - Given topic, find expert in topic
- The IBM Enterprise Track 2006 Team
  - Guillermo Averboch, Jennifer Chu-Carroll, Pablo A Duboue, David Gondek, J William Murdock and John Prager .

Corpus Processing: Generalities

A simplified view of our TREC 2006 Enterprise Track Indexing Pipeline (actual pipeline has 23 components)



Pipeline

- Reader
  - Binary Collection Reader
- Low Level Processing
  - LibMagic Annotator
  - HTML Detagger Annotator
- Higher Level Analytics
  - Enterprise Member Annotator
  - Expertise Detector Aggregate
- Analytics Consumers
  - Lucene Indexer
  - Pseudo Document Constructor
  - EKDB Indexer

Binary Collection Reader

- Reads the TREC XML format
- 300,000+ documents (a full crawl of the w3.org site)
- Binary format, to allow auto-detection of file type, encoding, etc.

LibMagic Annotator

- Uses “magic” numbers to heuristically guess the file type.
- JNI wrapper to libmagic in Linux.
- Non-supported file types are dropped.
- UIMA can run this remotely from a Windows machine.

HTML Detagger Annotator

- For documents identified as HTML, parse them and extract the text.
- Perform also encoding detection (utf-8 vs. iso-8859-1).
- Other detaggers (not shown) are applied to other file formats.

Enterprise Member Annotator

- Detects inside running text the occurrence of any variant of the 1,000+ experts for the Enterprise Track.
- Dictionary extended with name variants.
- Simple TRIE-based implementation.

Expertise Detector Aggregate

- Hierarchical aggregate of 16 annotators leveraging existing technology into a new “expertise detection” annotator.
- Includes a named-entity detector and a relation detector for semantic patterns.

Lucene Indexer

- Integration with Open Source technology.
- Indexes the tokens from the text.
- The UIMA framework also contains JuruXML, an indexer for semantic information.

Pseudo Document Constructor

- Uses the name occurrences to create a “pseudo” document with all text surrounding each expert name.
- The pseudo documents are indexed off-line.

EKDB Indexer

- Stores extracted entities and relations in a relational database.
- Standards-based (JDBC, RDF).
- Employed in a variety of research applications for search and inference.

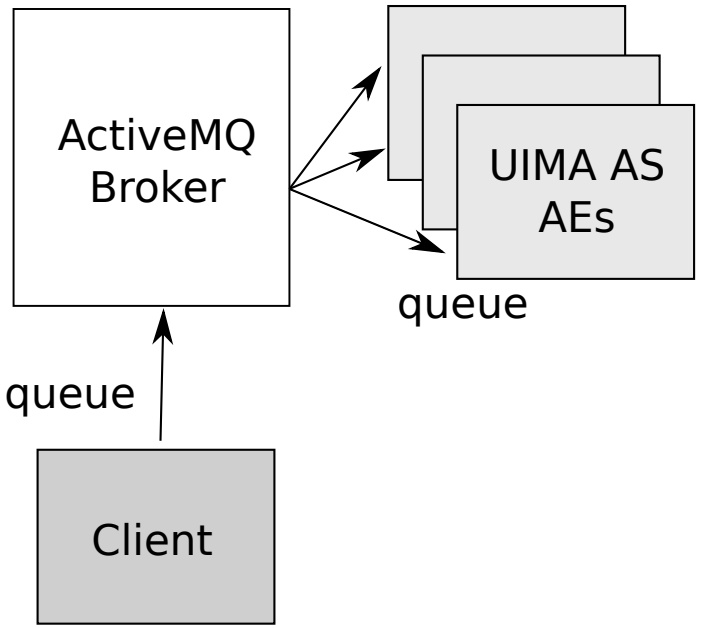
Outline

Custom Flow Controllers

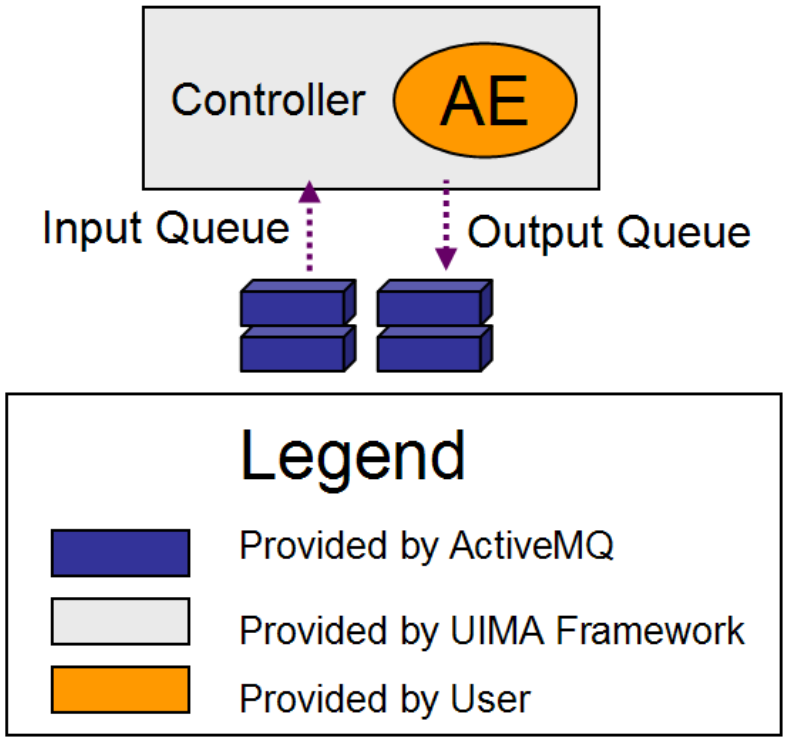
- UIMA allows you to specify which AE will receive the CAS next, based on all the annotations on the CAS.
- `examples/descriptors/flow_controller/WhiteboardFlowController.xml`
  - FlowController implementing a simple version of the "whiteboard" flow model. Each time a CAS is received, it looks at the pool of available AEs that have not yet run on that CAS, and picks one whose input requirements are satisfied. Limitations: only looks at types, not features. Does not handle multiple Sofas or CasMultipliers.

Outline

UIMA AS: ActiveMQ



UIMA AS: Wrapping Primitive AEs



UIMA AS: More information

- http://incubator.apache.org/uima/doc-uimaas-what.html
- http://svn.apache.org/viewvc/incubator/uima/uima\discretionary{-}{}{}as/trunk/uima-as-distr/src/main/readme/README?view=markup
- http://incubator.apache.org/uima/downloads/releaseDocs/2.3.0\discretionary{-}{}{}incubating/docs-uima-as/html/uima_async_scaleout/uima_async_scaleout.html

Outline

Descriptorless Primitive Analysis Engines

- Java 5 introduced annotations for metadata associated with Java classes.
  - The UIMA primitive AEs descriptor files fall squarely within their intended use cases.
- Example:

```
import org.apache.uima.annotation.*;

@UimaPrimitive(
   description="Identifies room numbers on text f
   typesystem="org/apache/uima/examples/roomtypes
)
public class RoomNumberAnnotator extends JCasAn
...
```

CASless JCas Types

- A common use case within UIMA is to do some processing and then kept some results outside the CAS.
  - These results are used with application level logic, outside the UIMA framework.
- Because the JCas types are only available when tied to a CAS, they cannot be used within application logic.
  - Copying information to POJOs that re-create the JCas types is a frequent and tedious task.
- Proposal: have JCasGen produce both CAS-backed and CAS-less implementation of the type defined in the type system.
  - With methods to bridge between the two.

Summary

- UIMA is a production ready framework for unstructured information processing.
- UIMA is a framework and it contains little or no annotators.
- It is an efficient framework that requires commitment on behalf of its practitioners.

- Outlook
  - As an open source project, new contributors are always welcomed.
  - There are a number of things I am personally interested in working with other people interested in UIMA.