LanguageWare Technology

Introduction

Introduction

- **Course Overview**
  - Introduction of LanguageWare® technology
- **Target Audience:**
  - All audiences
- **Prerequisites:**
  - None

Course objectives

After this course you will be able to:

- Understand what LanguageWare is
- Understand what NLP (Natural Language Processing) is
- Understand what UIMA (Unstructured Information Management Architecture) is
- Get start with LanguageWare and LanguageWare Resource Workbench

Course roadmap

- **Introduction of LanguageWare**

  NLP (Natural Language Processing)

- **LanguageWare Technology**

  Runtime

  Linguistic Resource

  Tools

- **Summary**

LanguageWare
What is it?

- **General**
  – Technology Component for Natural Language Processing

- **Specific**
  – Analyze “text expressed in a language we speak/write everyday (natural language)”
    - English, French, German, Portuguese, Spanish, Japanese, Chinese, Arabic, etc.
    - Not artificial language (Java™, Perl, Ruby, etc.)

- **For example (in terms of analysis for ICA):**

Text	According to finance report, IBM Corp. ’s EPS increased by 10.1%.	
Identify Language	English	Indexing
Segment Sentence	⟷	
Identify Token	According to finance report IBM Corp. ’s EPS increased by 10.1%	
Normalize Character Case	according	
Lemmatize Token	corporation increase	
Assign Part of Speech Tag	preposition noun(singular) noun(singular) noun(singular) preposition adjective noun(singular) noun(proper) posessive verb(past tense) numeral	Built-in Facet
Identify Domain Specific Term	IBM Corp. EPS	Custom Facet
Extract Domain Specific Phrase	IBM Corp. ’s EPS 10.1% Positive (finance – increase)	

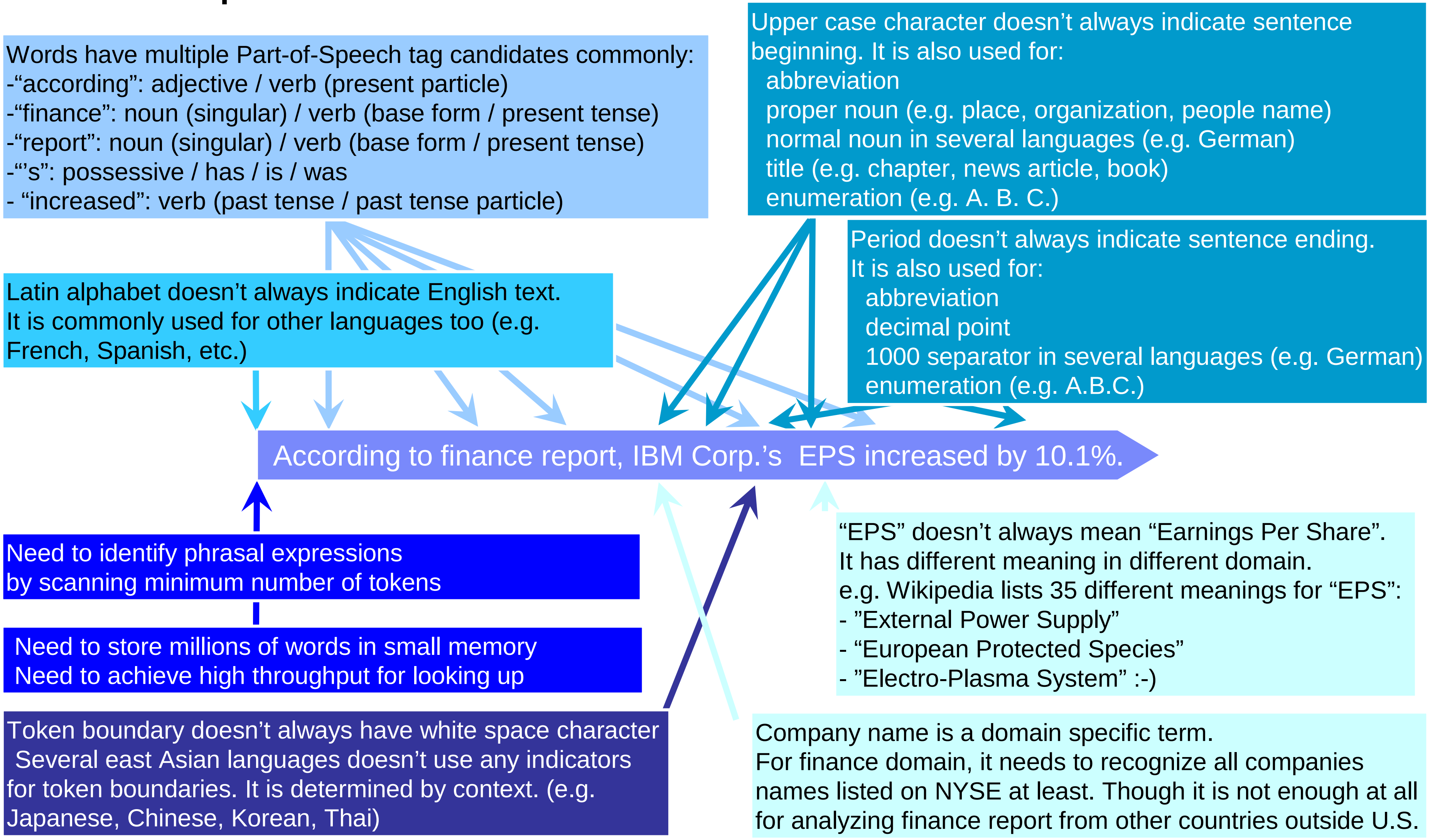
Natural Language Processing (NLP)
What is difficult? Isn't it an easy task? (1 of 2)

- **General**
  - – No, it is not a trivial task for computer, even though humans can do it easily.

- **Specific**
  - – It is not a simple string matching operation
  - – Computer needs to:
    - Resolve many ambiguities in text
    - Recognize domain specific terms / expressions
    - Deal with grammatical characteristics of each language
  - – In addition:
    - It has to achieve high throughput with low memory footprint

Natural Language Processing (NLP)
What is difficult? Isn't it an easy task? (2 of 2)

- **For example:**



Words have multiple Part-of-Speech tag candidates commonly:
-“according”: adjective / verb (present particle)
-“finance”: noun (singular) / verb (base form / present tense)
-“report”: noun (singular) / verb (base form / present tense)
-“’s”: possessive / has / is / was
- “increased”: verb (past tense / past tense particle)

Upper case character doesn’t always indicate sentence beginning. It is also used for:
abbreviation
proper noun (e.g. place, organization, people name)
normal noun in several languages (e.g. German)
title (e.g. chapter, news article, book)
enumeration (e.g. A. B. C.)

Period doesn’t always indicate sentence ending.
It is also used for:
abbreviation
decimal point
1000 separator in several languages (e.g. German)
enumeration (e.g. A.B.C.)

Latin alphabet doesn’t always indicate English text.
It is commonly used for other languages too (e.g. French, Spanish, etc.)

According to finance report, IBM Corp.’s EPS increased by 10.1%.

Need to identify phrasal expressions
by scanning minimum number of tokens

Need to store millions of words in small memory
Need to achieve high throughput for looking up

Token boundary doesn’t always have white space character
Several east Asian languages doesn’t use any indicators for token boundaries. It is determined by context. (e.g. Japanese, Chinese, Korean, Thai)

“EPS” doesn’t always mean “Earnings Per Share”.
It has different meaning in different domain.
e.g. Wikipedia lists 35 different meanings for “EPS”:
- ”External Power Supply”
- “European Protected Species”
- ”Electro-Plasma System” :-)

Company name is a domain specific term.
For finance domain, it needs to recognize all companies names listed on NYSE at least. Though it is not enough at all for analyzing finance report from other countries outside U.S.

Course roadmap

- **Introduction of LanguageWare**

  NLP (Natural Language Processing)

  UIMA (Unstructured Information Management Architecture)

- **LanguageWare Technology**

  Runtime

  Linguistic Resource

  Tools

- **Summary**

LanguageWare Technology

What does it provide to resolve the task?

- **General**
  - 1. Runtime, 2. Linguistic Resource, and 3. Tools
- **Specific**

  1. Runtime (Analysis Engine)
     - Single runtime, Multiple languages support (20+ languages)
     - State of the art performance, Low memory footprint (23 patents & 9 disclosures)
     - Built on open standard framework (Apache UIMA)
  2. Linguistic Resource (Dictionary, Parsing Rule)
     - Derived from deep linguistic expertise (10+ years experiences)
       - Many different language speakers are on board the team
  3. Tools
     - Eclipse Workbench to rapidly adapt analysis engine & linguistic resource to customer's business domain
       - Examples: health care, life science, finance, insurance, voice of customer, SNS

1. Runtime

What is UIMA?

- **General**
  - UIMA stands for “Unstructured Information Management Architecture”
  - A platform to orchestrate various analysis engines to discover vital knowledge from unstructured information

- **Specific**
  - Software infrastructure to:
    - Provide common data representation (CAS) for the artifact being analyzed (e.g. text)
    - Coordinate workflow of analysis engines in concert
    - Package analysis engines in a generic, portable way for deployment (PEAR)
  - Reliable and scalable
    - Originate from IBM Watson Research (2004)
    - Developed by IBM Software Group
  - Open & standard
    - Donated to Apache Software Foundation (2006) http://uima.apache.org/
    - Approved as OASIS Standard (2009)
  - Embedded in a lot of IBM products
    - IBM Content Analytics, OmniFind® Enterprise Edition, Lotus® Notes®, and many others!

1. Runtime
UIMA - Annotators

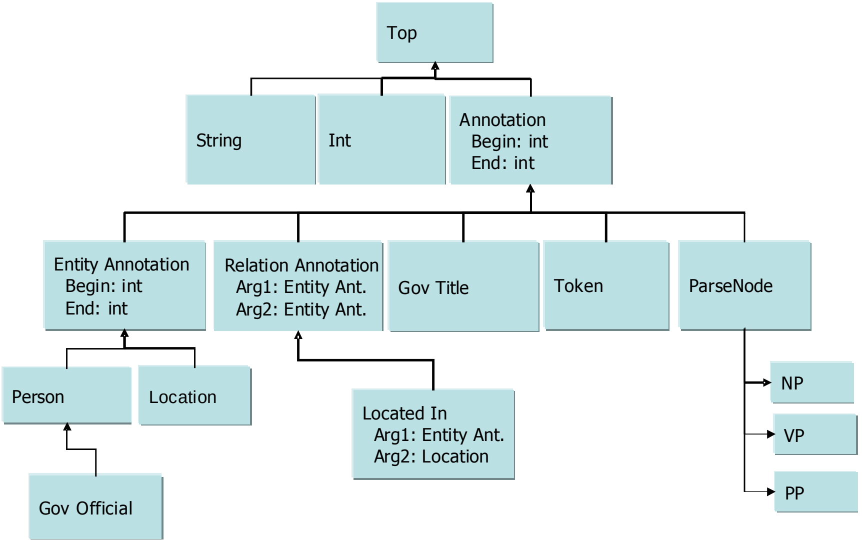
- **General**
  - – The basic building blocks in UIMA are called **Analysis Engines** (AEs). They are composed to analyze a document and produce analysis results.
  - – Analysis Engines are built by the framework using basic components that holds the core analysis algorithms running inside AEs. These components are called **Annotators**.
- **Specific**
  - – An Analysis Engine (AE) is a program that analyzes artifacts (e.g. documents) and infers information from them.
  - – Analysis Engines are constructed from building blocks called Annotators.
  - – An annotator is a component that contains analysis logic and uses all resources needed to execute that logic.
  - – Annotators produce their analysis results in the form of typed Feature Structures.

1. Runtime

UIMA - Type system

- **General**
  - A type system defines the various types of objects that may be discovered in documents by AE's that conform to that type system.

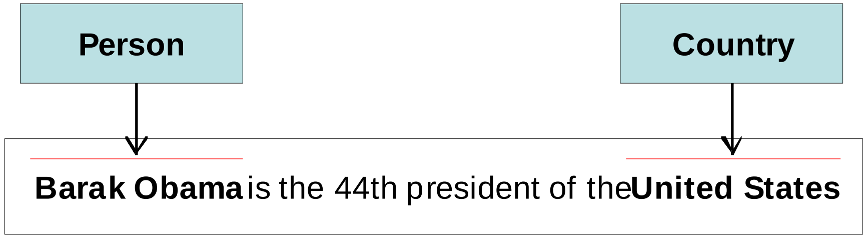
- **Specific**
  - UIMA defines a few basic types and allows to extend these to define an arbitrarily rich Type System.
  - Object types may be related to each other in a single-inheritance hierarchy.
  - There are no limits to the different types that may be defined in a type system.
  - A type system is domain and application specific.



1. Runtime

UIMA - Type system: Annotation type

- **General**
  - The **Annotation** type is used to identify and label or “annotate” a specific region of an artifact.
- **Specific**
  - The Annotation type for text includes two features, namely **begin** and **end**.
  - The Annotation type is a general and common type used in artifact analysis and from which additional types are often derived.
  - For Example, A **Person** type can be used to annotate mentions of person entities in text.

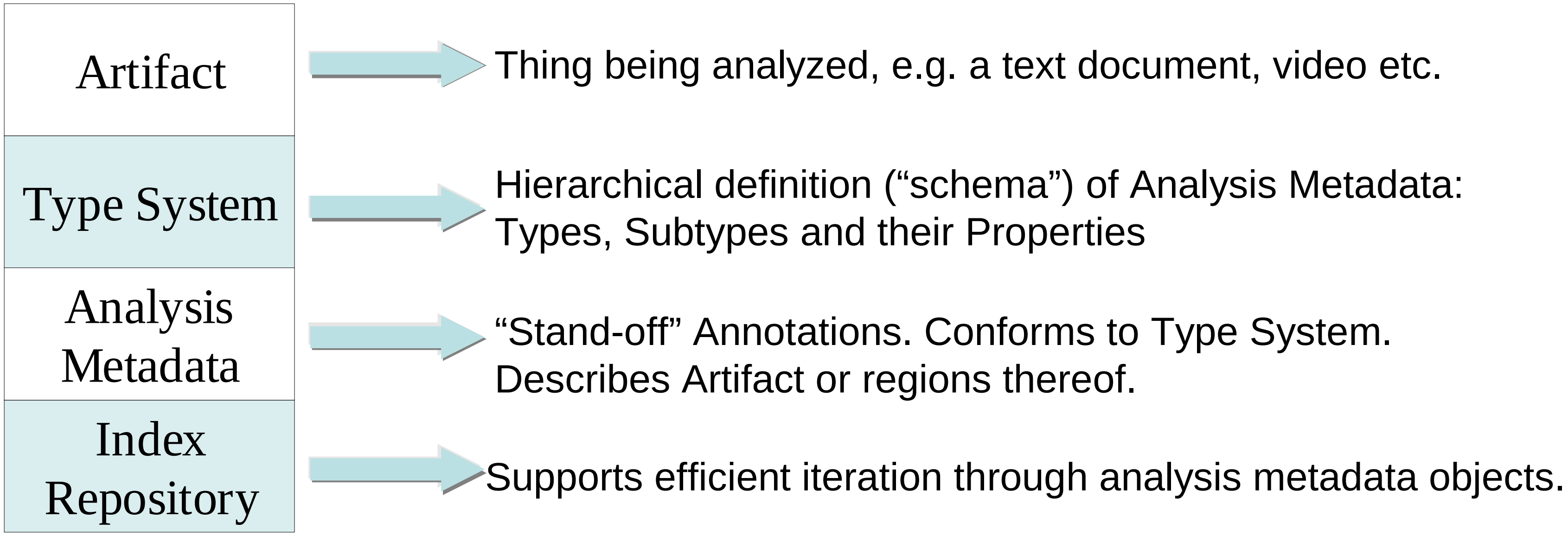

Person
Country
Barak Obama is the 44th president of the **United States**

1. Runtime

UIMA - Common Analysis Structure (CAS)

- **General**
  - UIMA defines a **C**ommon **A**nalysis **S**tructure (**CAS**) for annotators represent and share their results.

- **Specific**
  - The CAS is an object-based data structure that allows the representation of objects, properties and values.
  - The **Type system** represents an object schema for the CAS.



Artifact → Thing being analyzed, e.g. a text document, video etc.

Type System → Hierarchical definition (“schema”) of Analysis Metadata: Types, Subtypes and their Properties

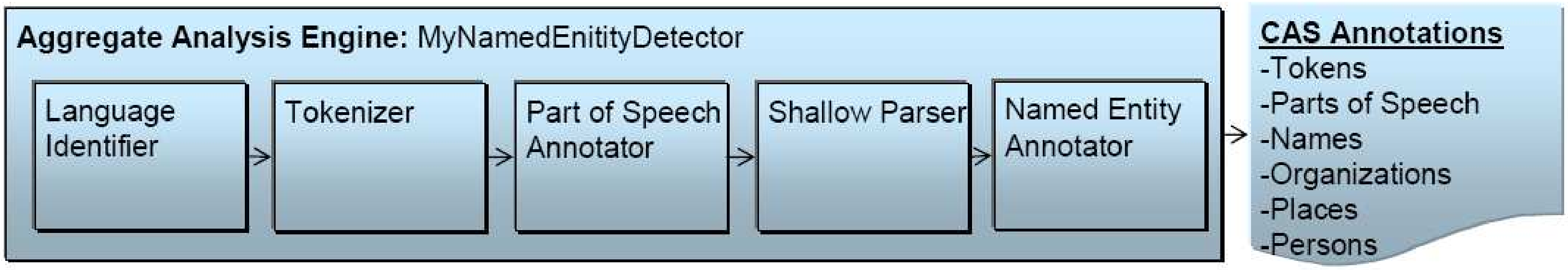
Analysis Metadata → “Stand-off” Annotations. Conforms to Type System. Describes Artifact or regions thereof.

Index Repository → Supports efficient iteration through analysis metadata objects.

1. Runtime

UIMA - Aggregate Analysis

- **General**
  - UIMA allows defining special type of Analysis Engines to contain other Analysis Engines organized in a flow. These more complex analysis engines are called **Aggregate Analysis Engines**.
- **Specific**
  - Annotators tend to perform fairly granular function.
  - A workflow of component engines may be orchestrated to perform more complex tasks.
  - For Example, An AE for named entity detection may include a pipeline of annotators starting with language detection and followed by tokenization, part-of-speech tagging, grammatical parsing and then finally named-entity detection.
  - Each step in the pipeline is required by the subsequent analysis.



Aggregate Analysis Engine: MyNamedEnitityDetector

Language Identifier → Tokenizer → Part of Speech Annotator → Shallow Parser → Named Entity Annotator →

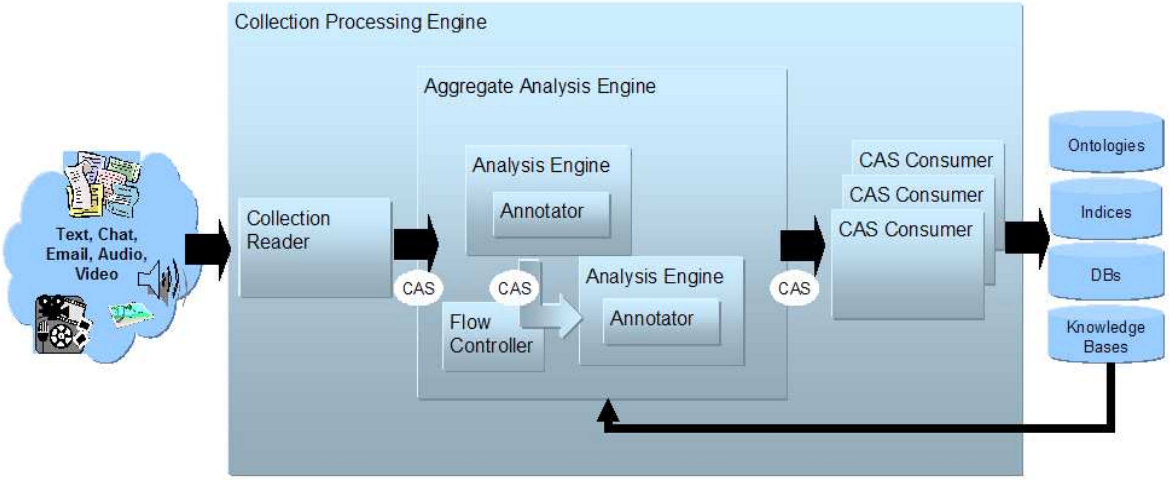
CAS Annotations
-Tokens
-Parts of Speech
-Names
-Organizations
-Places
-Persons

1. Runtime

UIMA - Collection processing (1 of 2)

- **General**
  - UIMA defines a collection processing architecture for the analysis of collections of documents from source to sink.

- **Specific**
  - The **Collection Processing Architecture** defines additional components for reading data from collections, preparing the data for processing by Analysis Engines, executing the analysis and extracting analysis results.



1. Runtime

UIMA - Collection processing (2 of 2)

- A **Collection Reader** connects to and iterates through a source collection, acquiring documents and initializing CASes for analysis.

- **CAS Consumers** function at the end of the flow to do the final CAS processing. For example, a CAS Consumer can be implemented to index CAS contents in a search engine.

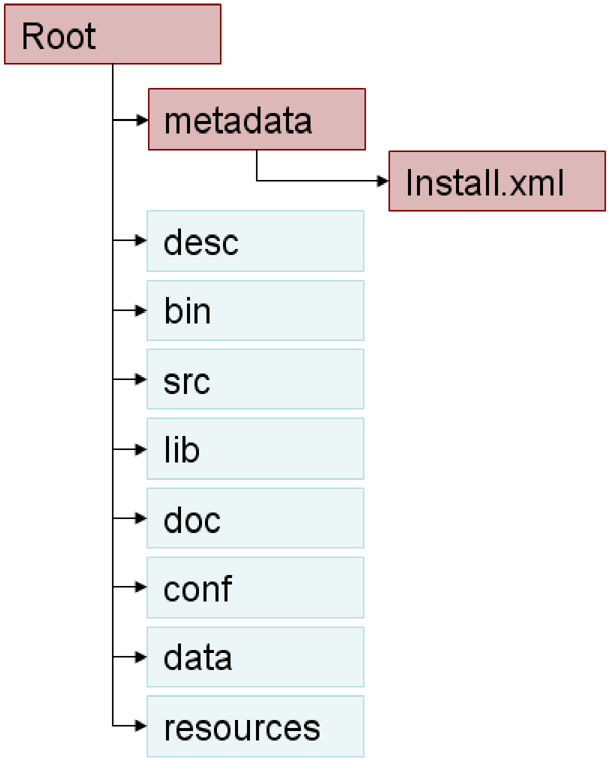
- A UIMA **C**ollection **P**rocessing **E**ngine (**CPE**) is an aggregate component that specifies a “source to sink” flow from a Collection Reader though a set of analysis engines and then to a set of CAS Consumers.

1. Runtime

UIMA - PEAR packages

- **General**
  - A **PEAR** (**P**rocessing **E**ngine **Ar**chive) file is a standard package for UIMA components.

- **Specific**
  - PEAR files are used for distribution and reuse of UIMA components by other components or applications.
  - A PEAR file is an archive file for an analysis engine that includes all required resources for installing that analysis engine in another UIMA environment.
  - An installed PEAR can be used as a component within a UIMA pipeline, by specifying the pear descriptor that is created when installing the pear file.

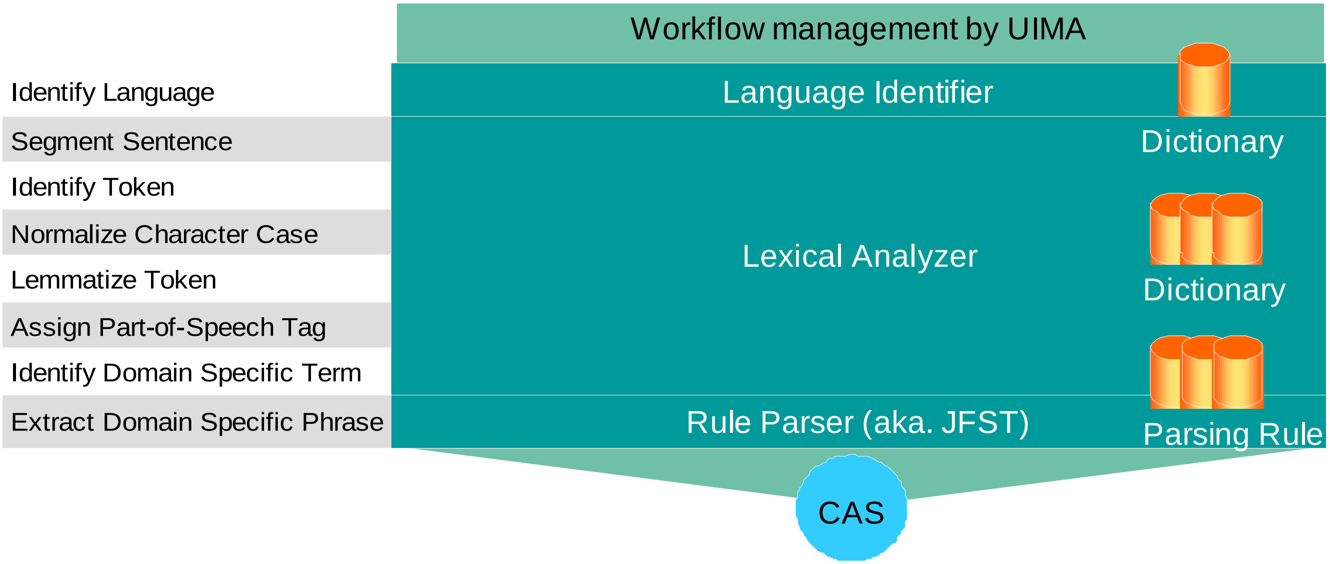


1. Runtime

How does LanguageWare use UIMA?

- **General**
  - Runtime is implemented as three analysis engines conforms to UIMA specification
    a)Language Identifier
    b)Lexical Analyzer
    c)Phrasal Rule Parser

- **Specific**
  - Analysis engines store the result of analysis in CAS object, and pass it to next engine
  - Workflow of analysis engines is managed by UIMA
  - Runtime is interoperable with other analysis engines implemented by 3rd party
  - Runtime is exported to ICA as UIMA PEAR (portable archive format)



2. Linguistic Resource

What languages are supported?

- **General**
  - 20+ languages

- **Specific**
  - Language Identifier
    - Afrikaans, Arabic, Balinese, Basque (Euskera), Bulgarian, Catalan, Chinese (Simplified and Traditional), Czech, Danish, Dutch, English, Finnish, French, German, Greek, Hebrew, Hungarian, Icelandic, Irish (Gaelic), Italian, Japanese, Korean, Malay, Norwegian Bokmal, Norwegian Nynorsk, Polish, Portuguese, Romanian, Russian, Spanish, Swedish, Tagalog, Thai, Turkish, Vietnamese
  - Lexical Analyzer
    - With Part-of-Speech tagging

      Arabic, Chinese, Danish, Dutch, German, English, Spanish, French, Italian, Japanese, Portuguese
    - Without Part-of-Speech tagging

      Afrikaans, Catalan, Czech, Greek, Korean, Norwegian (Nynorsk and Bokmal), Polish, Russian, Swedish, Finnish
  - Rule Parser
    - Any languages (as long as supported by preceding lexical analyzer)

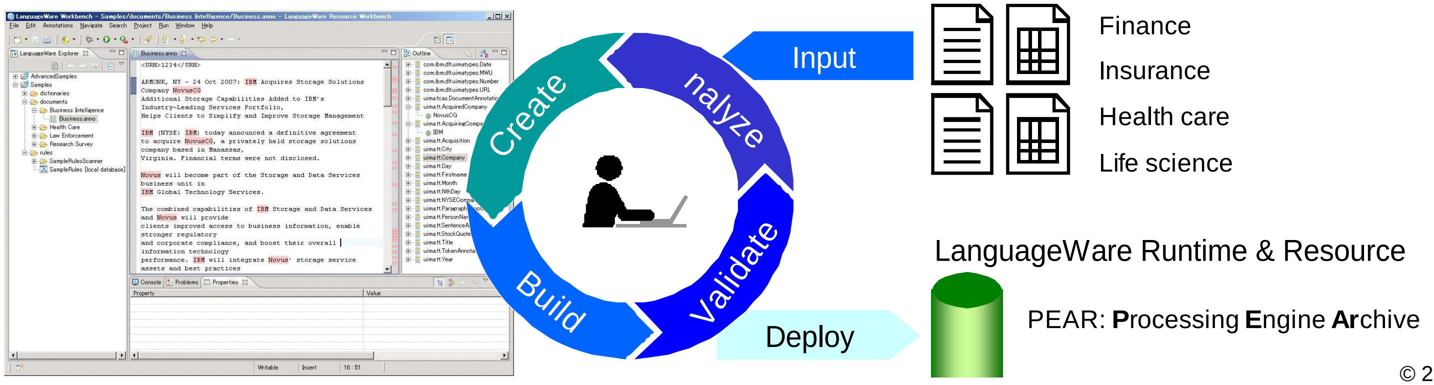
3. Tools

What is LanguageWare Resource Workbench?

- **General**
  - – Eclipse application to customize LanguageWare for target business domain
    - One size does not fit all !!

- **Specific**
  - – Configure UIMA Workflow
  - – Create dictionaries for target domain
  - – Create parsing rules to extract phrases in interest
  - – Verify the correctness of analysis for target domain
  - – Export LanguageWare runtime and resources to ICA on the fly

LanguageWare Resource Workbench



```
<URN>1234</URN>

ARMONK, NY - 24 Oct 2007: IBM Acquires Storage Solutions
Company NovusCG
Additional Storage Capabilities Added to IBM's
Industry-Leading Services Portfolio,
Helps Clients to Simplify and Improve Storage Management

IBM (NYSE: IBM) today announced a definitive agreement
to acquire NovusCG, a privately held storage solutions
company based in Manassas,
Virginia. Financial terms were not disclosed.

Novus will become part of the Storage and Data Services
business unit in
IBM Global Technology Services.

The combined capabilities of IBM Storage and Data Services
and Novus will provide
clients improved access to business information, enable
stronger regulatory
and corporate compliance, and boost their overall
information technology
performance. IBM will integrate Novus' storage service
assets and best practices
```

PEAR: **P**rocessing **E**ngine **Ar**chive

Course roadmap

- **Introduction of LanguageWare**

  NLP (Natural Language Processing)

- **LanguageWare Technology**

  Runtime

  Linguistic Resource

  Tools

- **Summary**

Course summary

You have completed this course and can:

- Get started with LanguageWare and LanguageWare Resource Workbench

See the LanguageWare help for more tips and advanced use cases.

Contacts

- If you have any questions, comments or suggestions, contact us using the LanguageWare email address *EMEALAN@ie.ibm.com* or developerWorks® Forum.

Trademarks, copyrights, and disclaimers

IBM, the IBM logo, ibm.com, developerWorks, LanguageWare, Lotus, Lotus Notes, and OmniFind are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of other IBM trademarks is available on the web at "Copyright and trademark information" at http://www.ibm.com/legal/copytrade.shtml

Java, and all Java-based trademarks and logos are trademarks of Oracle and/or its affiliates.

Other company, product, or service names may be trademarks or service marks of others.

THE INFORMATION CONTAINED IN THIS PRESENTATION IS PROVIDED FOR INFORMATIONAL PURPOSES ONLY. WHILE EFFORTS WERE MADE TO VERIFY THE COMPLETENESS AND ACCURACY OF THE INFORMATION CONTAINED IN THIS PRESENTATION, IT IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED. IN ADDITION, THIS INFORMATION IS BASED ON IBM'S CURRENT PRODUCT PLANS AND STRATEGY, WHICH ARE SUBJECT TO CHANGE BY IBM WITHOUT NOTICE. IBM SHALL NOT BE RESPONSIBLE FOR ANY DAMAGES ARISING OUT OF THE USE OF, OR OTHERWISE RELATED TO, THIS PRESENTATION OR ANY OTHER DOCUMENTATION. NOTHING CONTAINED IN THIS PRESENTATION IS INTENDED TO, NOR SHALL HAVE THE EFFECT OF, CREATING ANY WARRANTIES OR REPRESENTATIONS FROM IBM (OR ITS SUPPLIERS OR LICENSORS), OR ALTERING THE TERMS AND CONDITIONS OF ANY AGREEMENT OR LICENSE GOVERNING THE USE OF IBM PRODUCTS OR SOFTWARE.