


DKPro Core

Software components for NLP


Pedro Santos
Dr. Richard Eckart de Castilho

[DKPRO](#) [CORE](#) [DOWNLOADS](#) [DOCUMENTATION](#) [ISSUES](#) [SOURCE](#) [CONTACT](#) [ABOUT](#)


 **DKPro Core**

A collection of software components for natural language processing (NLP) based on the Apache UIMA framework.


Many NLP tools are already freely available in the NLP research community. DKPro Core provides Apache UIMA components wrapping these tools (and some original tools) so they can be used interchangeably in UIMA processing pipelines. DKPro Core builds heavily on [uimaFIT](#) which allows for rapid and easy development of NLP processing pipelines, for wrapping existing tools and for creating original UIMA components. [More](#)




Components
Find out more about our bundled components.



Models/Languages
Various models covering different languages accompany the components.




Formats
Reading and writing various formats is just one line of code away.




Typesystem
Our typesystem is comprehensive, yet simple.


Latest release: 1.7.0 (2014-11-28)



DKPro with Java
The original flavour. Use DKPro in your Java projects.



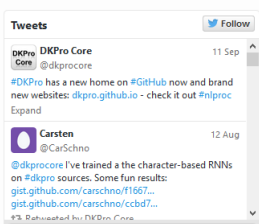
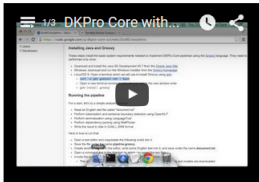
DKPro with Groovy
Create self-contained scripts using DKPro and Groovy!



DKPro with Jython
Easily integrate DKPro into your python projects!

How to cite

Many of the wrapped third-party components and the models used by them should be cited individually. We currently do not provide a comprehensive overview over citable publications. We encourage you to track down citable publications for these dependencies. However, you might find pointers to some relevant publications in the Model overview of the DKPro Core release you are using or in the JavaDoc of individual components.



Agenda

- What is a pipeline?
- Working with annotations
 - What is a type system?
 - What is the Common Analysis Structure (CAS)?
- Working with components
 - What is a reader?
 - What is an analysis engine?
 - What is a writer? (aka consumer)
- DKPro Core component collection



What is UIMA?

- Component-based architecture
 - Analysis of **unstructured data**
 - Structure from the unstructured data
- How?
 - Like an assembly line...
 - Raw material
 - Refinement, step by step
 - Nice car in the end



Apache UIMA™ – Some history

- 2003 – David Ferrucci and Adam Lally paper
 - *Accelerating corporate research in the development, application and deployment of human language technologies*
- 2004 – IBM alphaWorks project
 - IBM LanguageWare
- 2006 – Apache Incubator project
- 2009 – OASIS Standard
- 2010 – Full Apache project
- 2010 – IBM's *Watson* Jeopardy Challenge



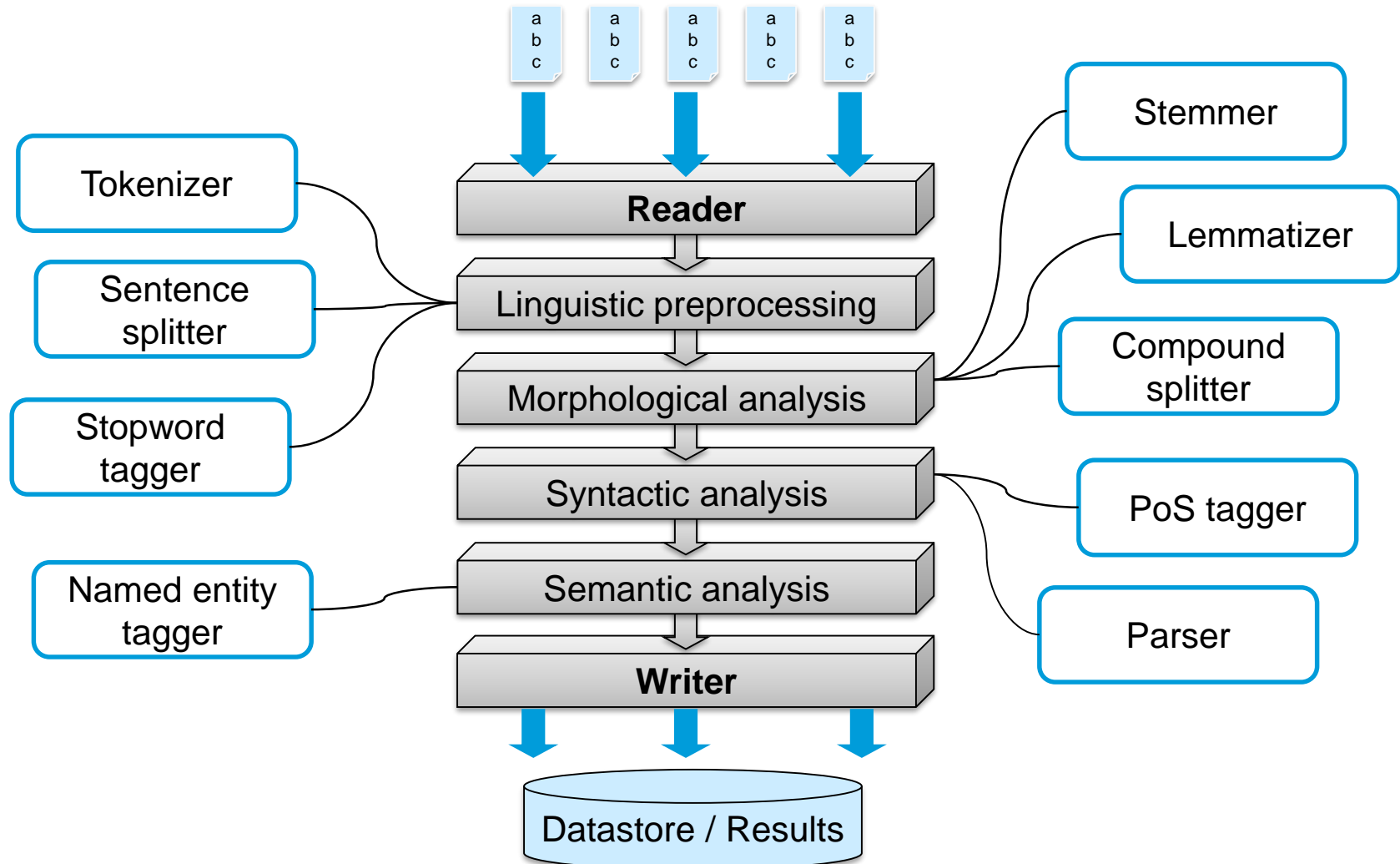
<http://uima.apache.org>

Pipelines



TECHNISCHE
UNIVERSITÄT
DARMSTADT

Pipeline Architecture



Component – Collection Reader

- Empty data structure (CAS) → Reader
- Reader → Text (SofA) and Meta-Data (e.g. language)

Reader

CAS

SofA	Language:	Latin
	DocumentText:	Ubi est Cornelia? Subito Marcus vocat: „Ibi Cornelia est, ibi stat!“

Component – Analysis Engine

- Structure → Analysis Engine (AE)
- Analysis Engine → Annotation

Reader

Tokenizer

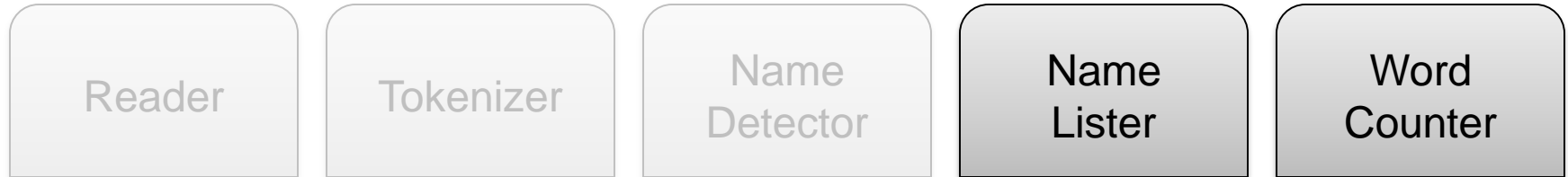
Name
Detector

CAS

SofA Language: Latin
DocumentText: Ubi est Cornelia?
 Subito Marcus vocat:
 „Ibi Cornelia est, ibi stat!“

Token(0, 3) Token(4, 7) Token(8,16) ...
Name(8, 16) Name(25, 31) ...

- Annotation -> CAS Consumer



CAS

SofA Language: Latin
DocumentText: Ubi est Cornelia?
 Subito Marcus vocat:
 „Ibi Cornelia est, ibi stat!“

Token(0, 3) Token(4, 7) Token(8,16)...

Name(8, 16) Name(25, 31) ...

Cornelia
Marcus

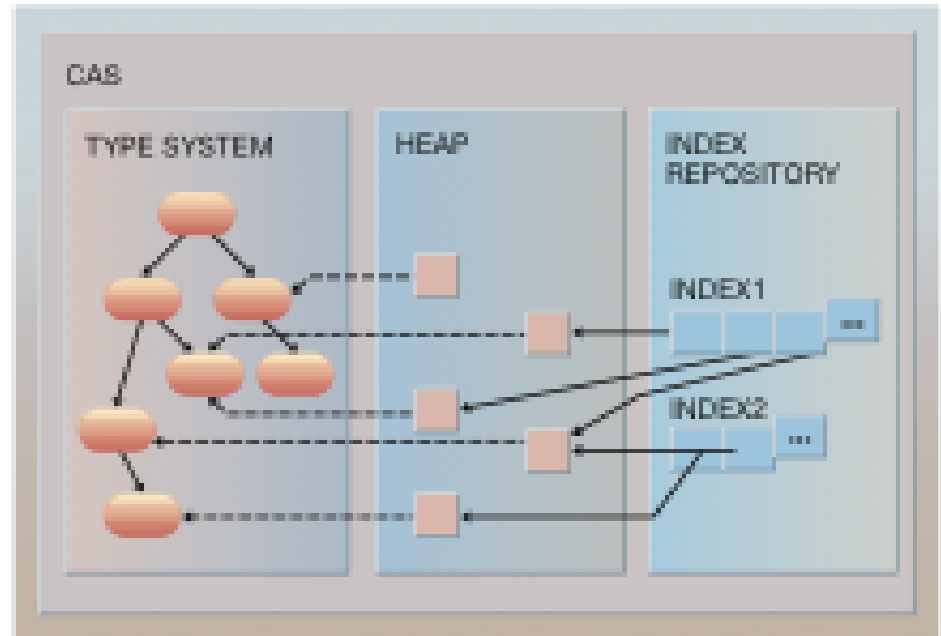
11 words
8 unique words



UIMA Data Structures

Common Analysis System (CAS)

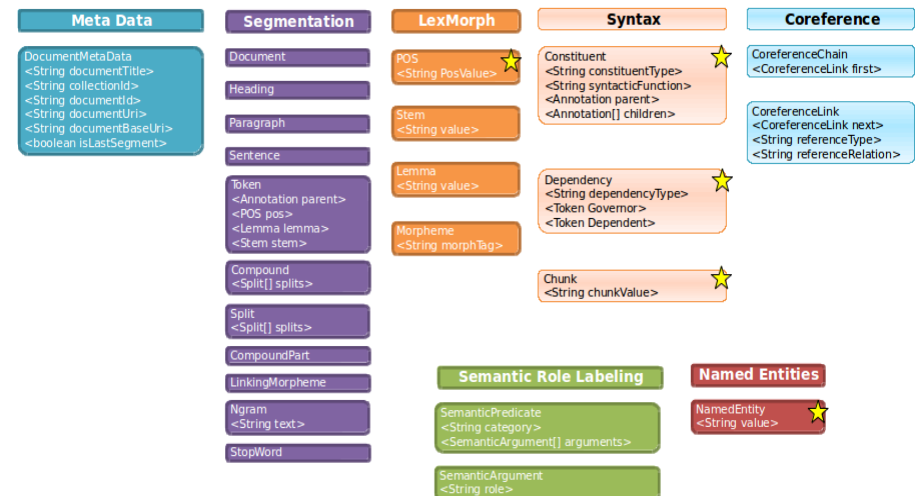
- Access to primary data
- Secondary data storage
 - a.k.a. Annotations
- In-memory database
 - Annotation types = tables
 - Indexes



Type System

- Platform-independent specification
- Object-oriented type system:
 - Type → class
 - Feature → class attributes
 - Feature Structure → instance
 - Single inheritance
 - No methods or encapsulation
- Primitive features: integer, float, boolean, string
- Built-in complex features: arrays, feature arrays...
- Communication contract

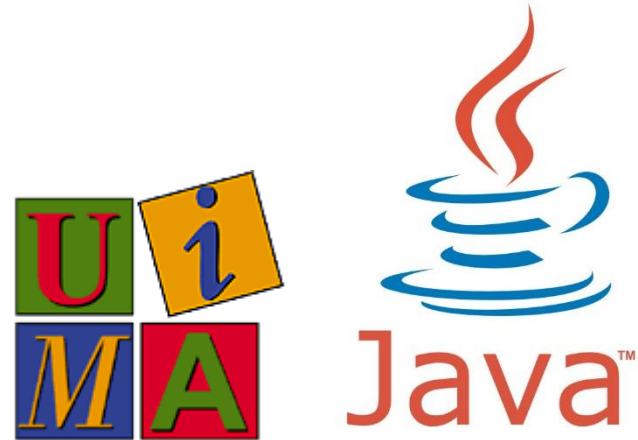
DKPro Core Type System (Top Level)



★ For these types, DKPro Core provides several specialized subtypes, e.g. *NP* for noun phrase constituents or *Location* for places.

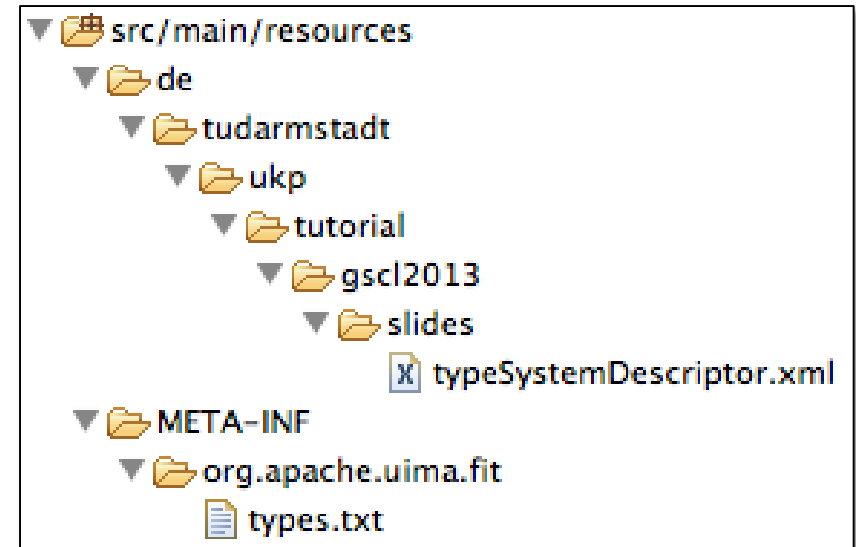
Java + CAS = JCas

- JCas = CAS types into the Java type system
- JCasGen
 - Java classes from XML type system descriptor



uimaFIT type system detection

- No explicit loading/creation of type system
- Type system detection mechanism
- Types defined in XML descriptor files
- Scanning of classpath for type system descriptor files



Type System Editor (Eclipse)

- JCasGen → UIMA types available as Java Classes

Type System Definition

▼ **Types (or Classes)**

The following types (classes) are defined in this analysis engine descriptor.
The grayed out items are imported or merged from other descriptors, and cannot be edited here. (To edit them, edit their source files).

Type Name or Feature Name	SuperType or Range	Element Type
<input type="checkbox"/> de.tudarmstadt.ukp.tutorial.gsc12013.slides.Token	uima.tcas.Annotation	
length	uima.cas.Integer	
de.tudarmstadt.ukp.tutorial.gsc12013.slides.Name	uima.tcas.Annotation	
de.tudarmstadt.ukp.tutorial.gsc12013.slides.Sentence	uima.tcas.Annotation	
de.tudarmstadt.ukp.tutorial.gsc12013.slides.Paragraph	uima.tcas.Annotation	

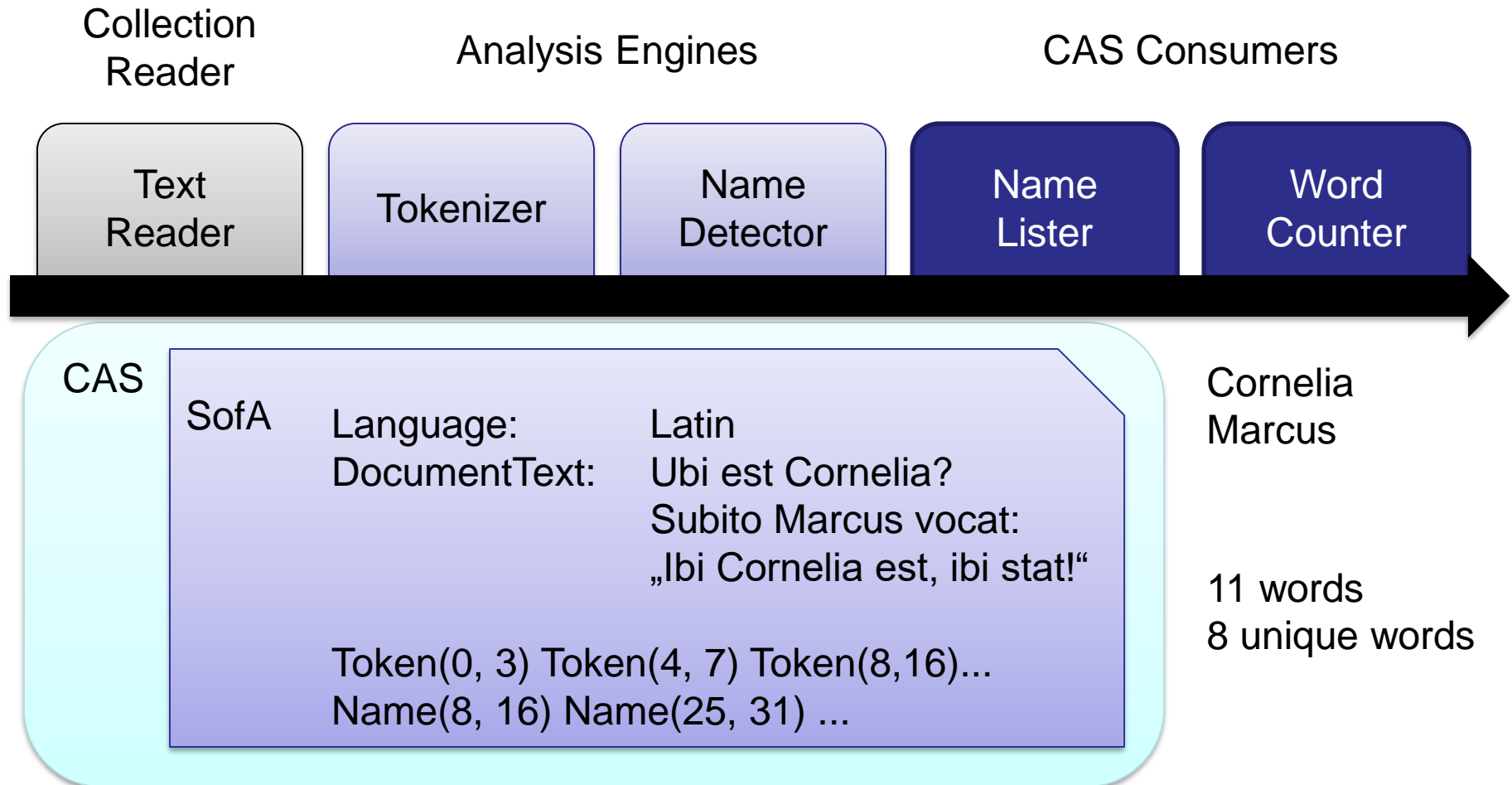
Add Type
Add...
Edit...
Remove
Export...
JCasGen
☐ limited

Components



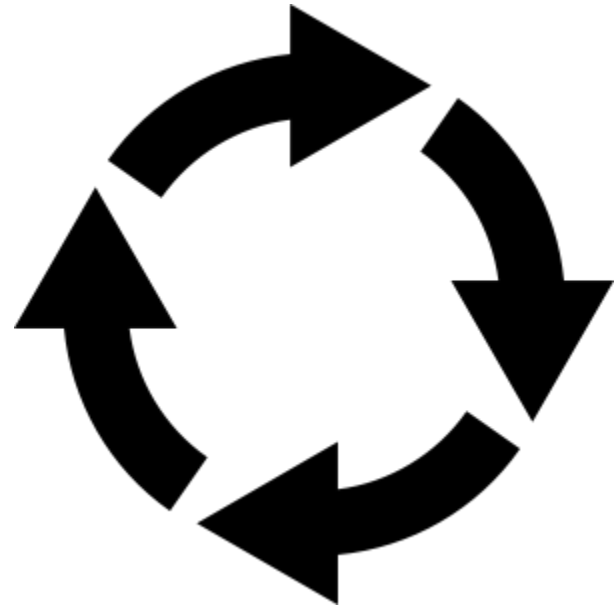
TECHNISCHE
UNIVERSITÄT
DARMSTADT

Components



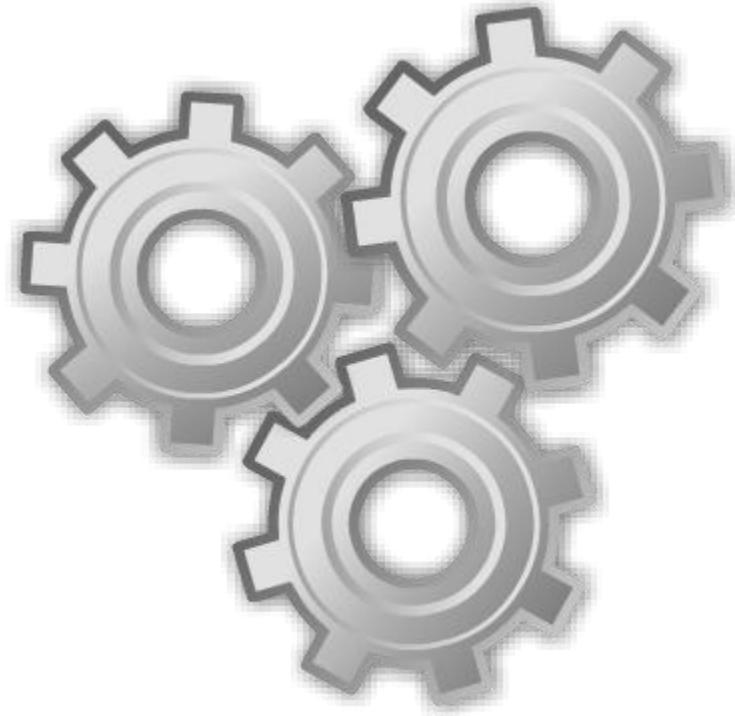
API – Life-Cycle Events

- Component life-cycle events
 - initialize()
 - reconfigure()
 - destroy()
- Processing life-cycle events
 - collectionProcessComplete()



API – Processing Methods

- **CollectionReader**
 - hasNext()
 - getNext()
 - getProgress()
- **AnalysisEngine**
 - process()
- **CasConsumer**
 - process(): no modifications to CAS

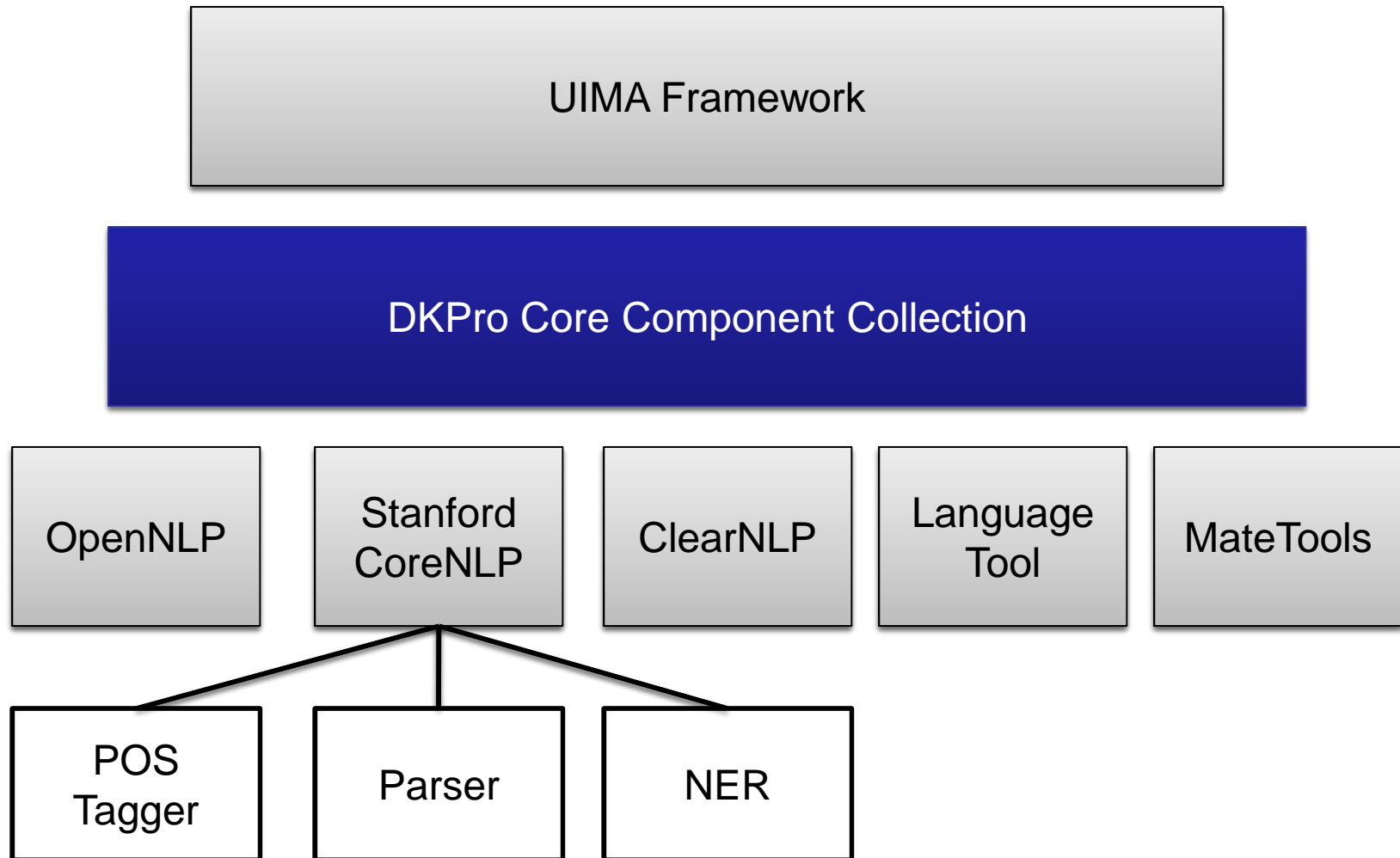


DKPro Core Component Collection



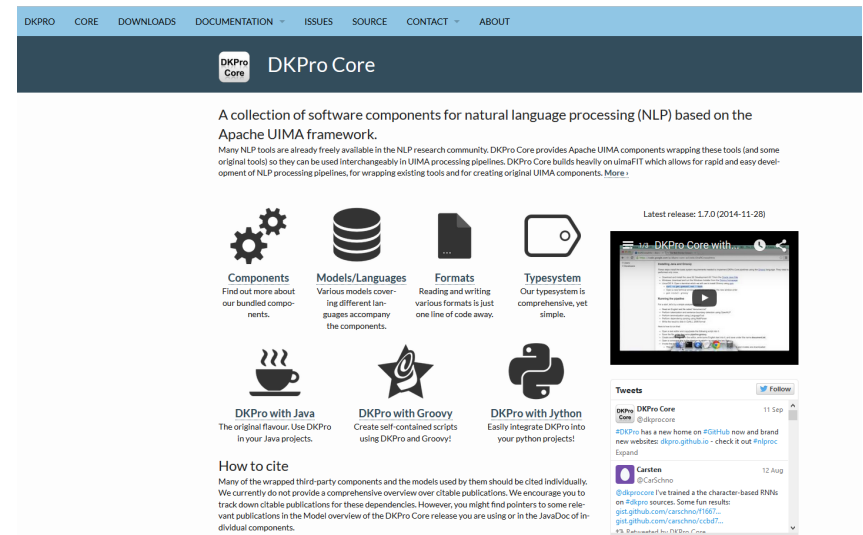
TECHNISCHE
UNIVERSITÄT
DARMSTADT

What's a component collection



DKPro Core

- Integration framework
 - Processing: tools and models
 - Primary data: corpora
 - Auxiliary data: other language resources (e.g. lexical resources)
- Primarily integration of existing work, not original work
- Contribution: integration itself
- Open Source under Apache Software License & GNU Public License



<https://dkpro.github.io/dkpro-core/>

- **Simplicity**

- Common data types
- Common set of parameters
- Sensible parameters defaults for minimal need for configuration
- Compose powerful pipelines with a few lines of code

- **Modularity**

- Usage of the necessary

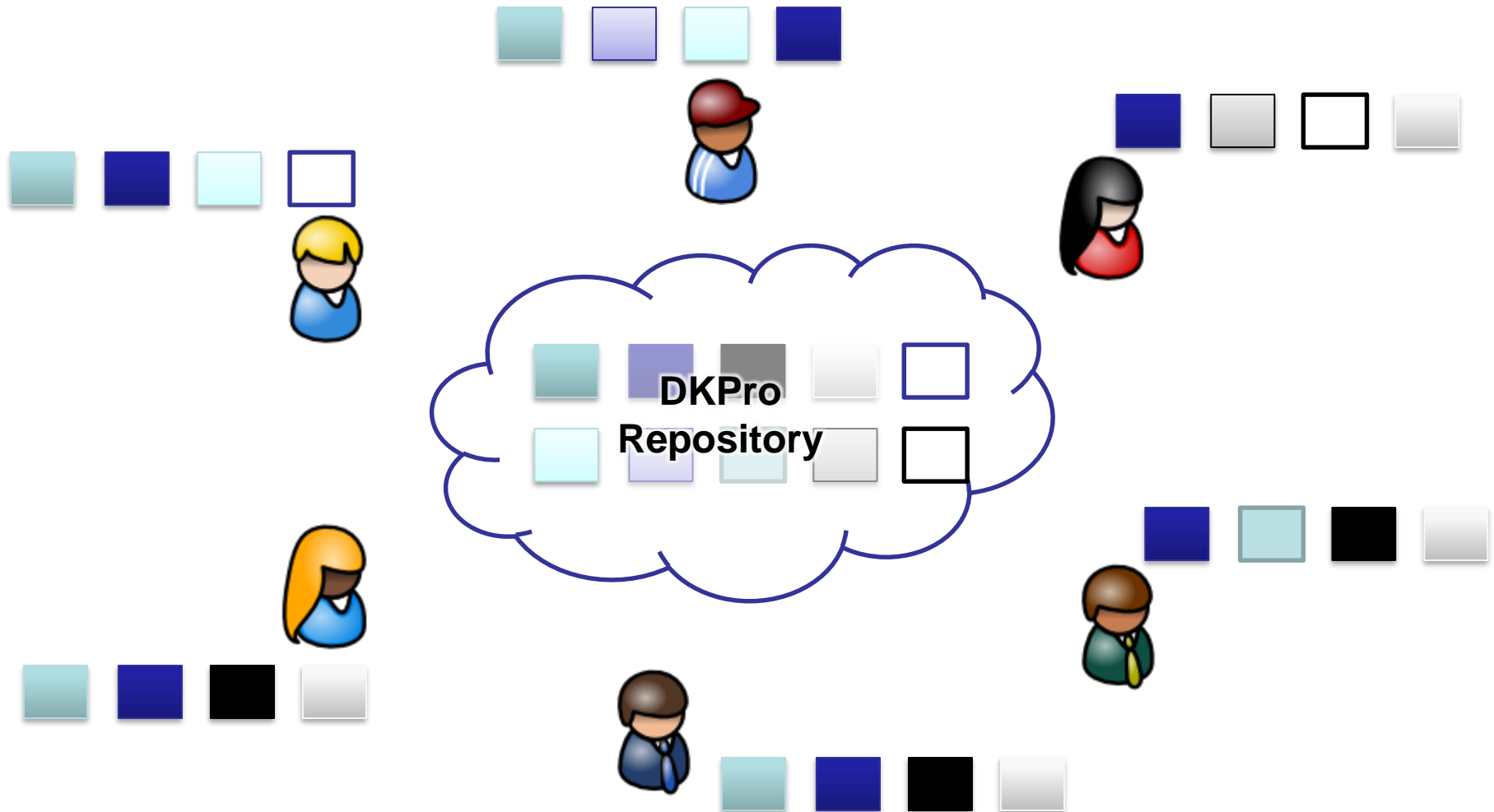
▪ ***Stuff has to “just work”, everywhere.***

- **Flexibility**

- Parameters override for fine-grained control
- Data types extension with custom fields
- Type mappings customization

▪ ***Stuff has to “just work”, everywhere.***

Managing Deployment



UKP OSS Component Repository

Publish component

Overview

Artifact

Group Id:

Artifact Id:*

Version:

Packaging:

Parent

Group Id:*

Artifact Id:*

Version:*

Relative Path:

Properties

Project

Name:

URL:

Description:

Inception:

Organization

SCM

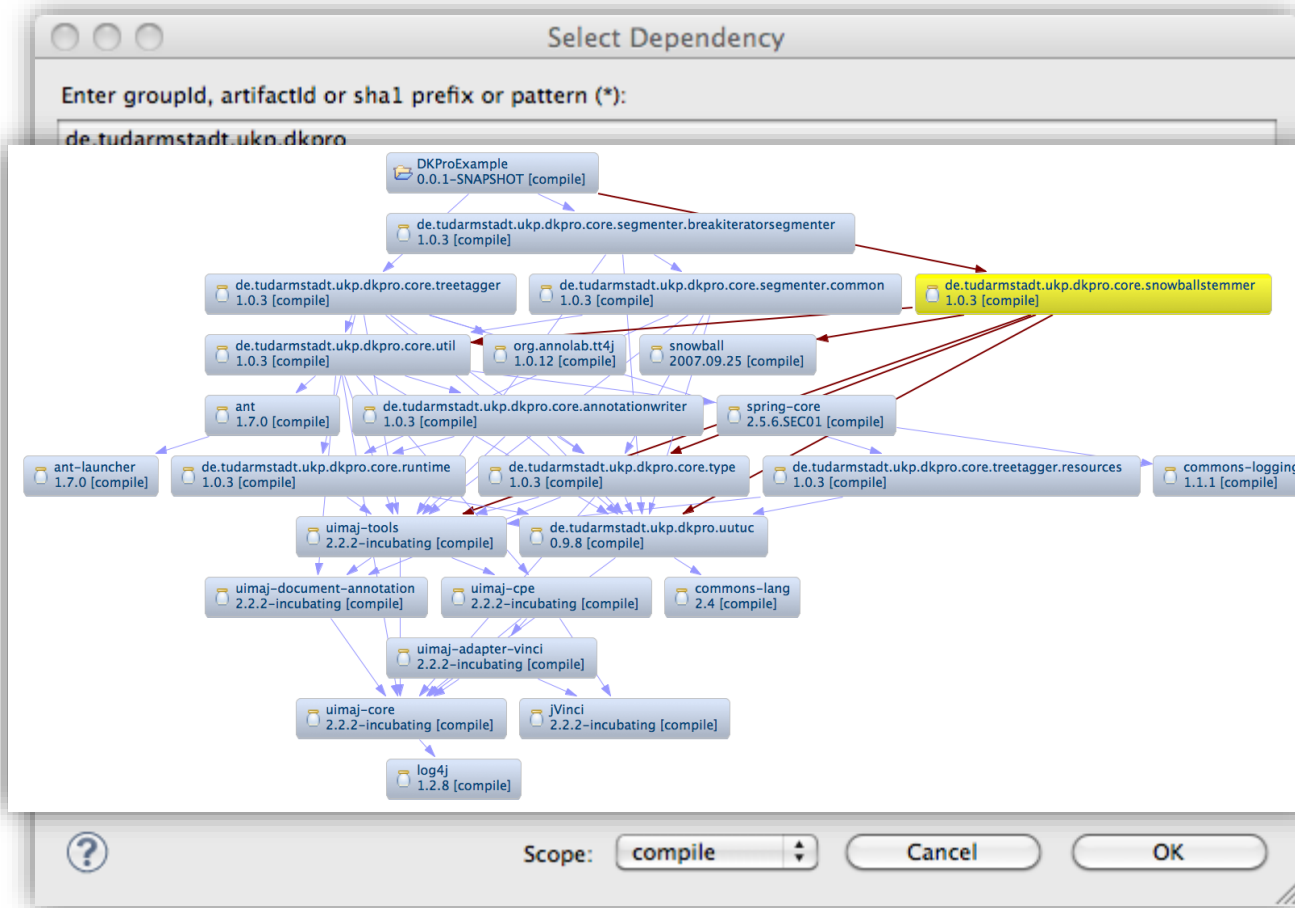
Issue Management

Continuous Integration



UKP OSS Component Repository Retrieving components

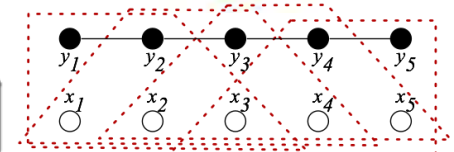
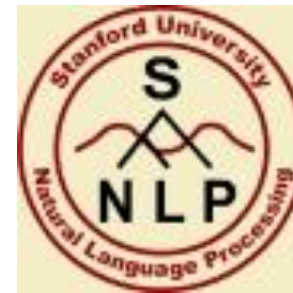
Component Repository



Tools and formats

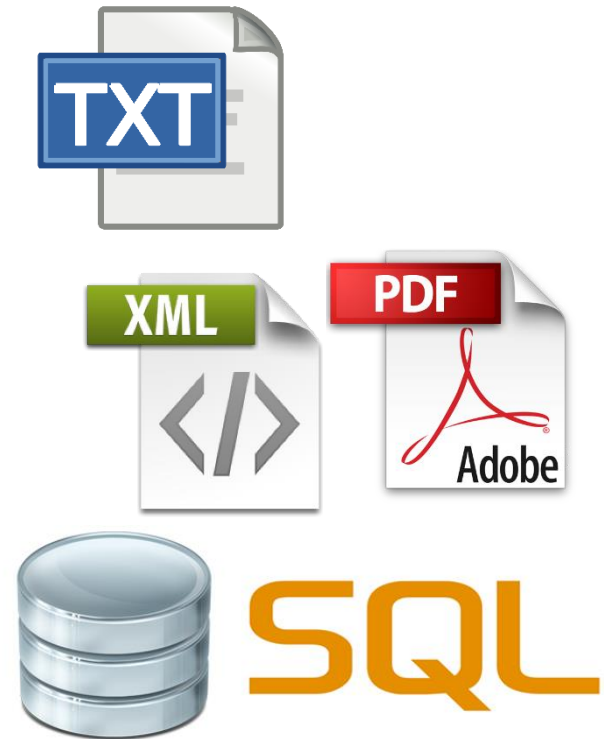
■ Integrated tools

- Stanford NLP
- OpenNLP
- Mate-Tools
- ClearNLP
- LanguageTool
- TreeTagger
- JWordSplitter
- Snowball Stemmer
- TextCat
- MaltParser
- MstParser
- BerkeleyParser
- ...



▪ Supported formats

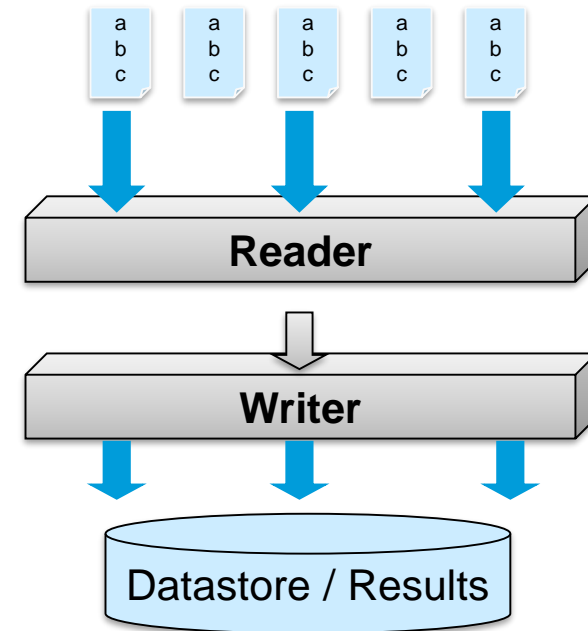
- Text
- PDF
- TIGER XML
- TEI XML
- BNC XML
- Negra Export
- SQL Databases
- Google web1t n-grams
- ...



Readers and Writers

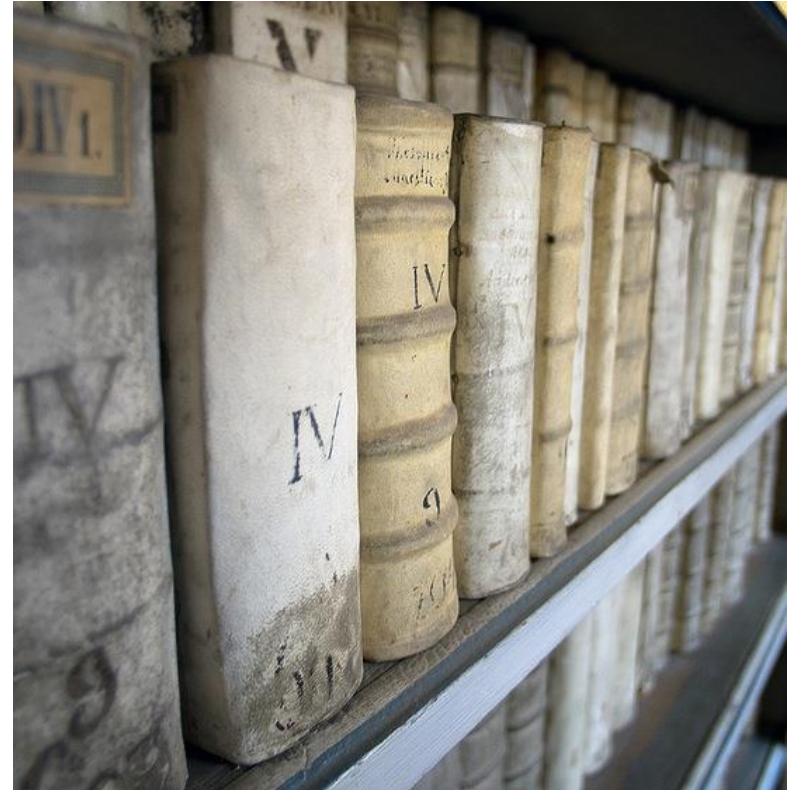
■ Common parameters

- Source / target location
- Source / target encoding
- Language (for readers)



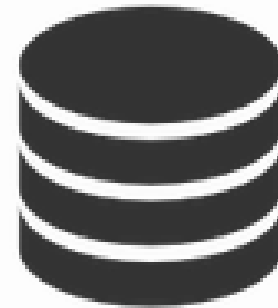
Some currently supported corpora/resources

- British National Corpus
- Wacky Corpora
- TüBa D/Z
- Tiger Corpus
- Digitale Bibliothek
- Brown Corpus
- ACL Anthology Reference Corpus
- ...
- Google Web1T n-grams



Good range of pre-trained models

- Upstream models packaged for convenient deployment and use
- Additional model meta-data
- 90+ models
- 20+ tools
- 15+ languages
- Best supported
 - English (Penn Treebank Tagset, Stanford Dependencies)
 - German (STTS Tagset, Negra/Tiger)



Models/Languages

Various models covering different languages accompany the components.

DKPro Type System Overview

Meta Data

```
DocumentMetaData
<String documentTitle>
<String collectionId>
<String documentId>
<String documentUri>
<String documentBaseUri>
```

Segmentation

Document

Token

LinkingMorpheme

Heading

Compound

Ngram

Paragraph

Split

StopWord

Sentence

CompoundPart

LexMorph

POS

Stem

Lemma

Morpheme

Syntax

Constituent

Dependency

Chunk

Coreference

CoreferenceChain

CoreferenceLink

Semantic Role Labeling

SemanticPredicate

SemanticArgument

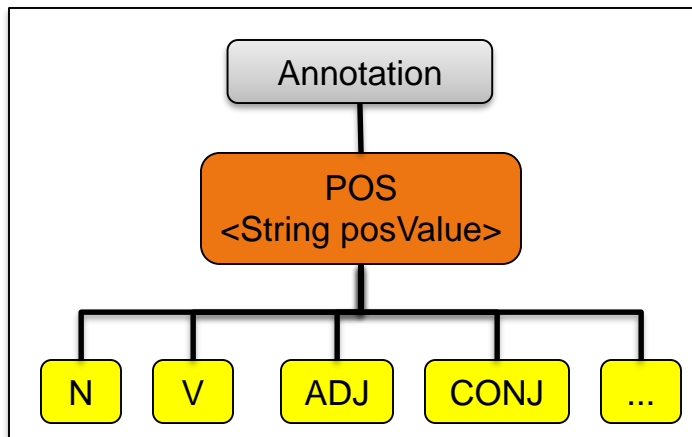
Named Entities

NamedEntity

Location

Person

...etc...

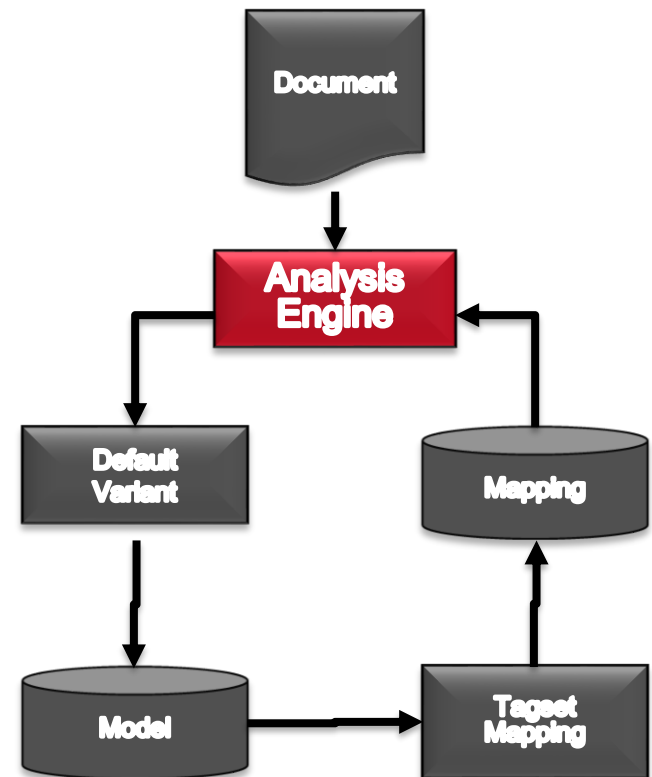


▪ Common parameters

- Model location
- Model encoding
- Model variant
- Mapping location
- Language

▪ Common features

- Model loading based on document language
- Print model tag set to log
- Default variants



Hands-on



TECHNISCHE
UNIVERSITÄT
DARMSTADT

