Bringing Named Entity Recognition on Drupal Content Management System

José Ferrnandes[1] and Anália Lourenço[1,2]

[1] ESEI - Escuela Superior de Ingeniería Informática, University of Vigo, Edificio Politécnico, Campus Universitario As Lagoas s/n, 32004 Ourense, Spain
[2] IBB - Institute for Biotechnology and Bioengineering, Centre of Biological Engineering, University of Minho, Campus de Gualtar, 4710-057 Braga, Portugal
jose@bloomidea.com, analia@{ceb.uminho.pt,uvigo.es}

Abstract. Content management systems and frameworks (CMS/F) play a key role in Web development. They support common Web operations and provide for a number of optional modules to implement customized functionalities. Given the increasing demand for text mining (TM) applications, it seems logical that CMS/F extend their offer of TM modules. In this regard, this work contributes to Drupal CMS/F with modules that support customized named entity recognition and enable the construction of domain-specific document search engines. Implementation relies on well-recognized Apache Information Retrieval and TM initiatives, namely Apache Lucene, Apache Solr and Apache Unstructured Information Management Architecture (UIMA). As proof of concept, we present here the development of a Drupal CMS/F that retrieves biomedical articles and performs automatic recognition of organism names to enable further organism-driven document screening.

Keywords: Drupal, text mining, named entity recognition, Apache Lucene, Apache Solr, Apache UIMA.

1 Introduction

The number of generic Text Mining (TM) software tools available now is considerable [1, 2], and almost every computer language has some module or package dedicated to natural language processing [3]. Notably, Biomedical TM (BioTM), i.e. the area of TM dedicated to applications in the Biomedical domain, has grown considerably [4, 5].

One of the main challenges in BioTM is achieving a good integration of TM tools with tools that are already part of the user workbench, in particular data curation pipelines [4, 6]. Many TM products (especially, commercial products) are built in a monolithic way and often, their interfaces are not disclosed and open standards are not fully supported [7]. Also, it is important to note that biomedical users have grown dependent of Web resources and tools, such as online data repositories, online and downloadable data analysis tools, and scientific literature catalogues [8]. Therefore, it

J. Sáez-Rodríguez et al. (eds.), *8th International Conference on Practical Appl. of Comput. Biol. & Bioinform. (PACBB 2014)*, Advances in Intelligent Systems and Computing 294,
DOI: 10.1007/978-3-319-07581-5_31,

is desirable to integrate TM tools with these resources and tools, and it seems logical to equip Web development frameworks, such as Content Management Systems and Frameworks (CMS/F), with highly customizable TM modules.

Drupal, which is one of the most common open source CMS/Fs (http://trends.builtwith.com/cms), already presents some contributions to Bioinformatics and TM: the GMOD Drupal Bioinformatic Server Framework [9], which aims to speed up the development of Drupal modules for bioinformatics applications; the OpenCalais project that integrates Drupal with the Thomson Reuters' Calais Web service (http://www.opencalais.com), a service for annotating texts with URIs from the Linked Open Data cloud; and, RDF/RDFa support so to enable the use of this ontology language in Web knowledge exchange and facilitate the development of document-driven applications, and promote the availability of knowledge resources [10].

The aim of this work was to extend further Drupal TM capabilities, notably to enable the incorporation of third-party specialized software and the development of customized applications. The proof of concept addressed Named Entity Recognition (NER), i.e. the identification of textual references to entities of interest, which is an essential step in automatic text processing pipelines [11, 12]. There are many open-source and free NER tools available, covering a wide range of bio-entities and approaches. So, our efforts were focused on implementing a Drupal module that would support customised NER and, in particular, to equip Drupal with the necessary means to construct domain-specific document search engines. For this purpose, we relied on Apache Information Retrieval (IR) and TM initiatives, namely Apache Lucene, Apache Solr and Apache Unstructured Information Management Architecture (UIMA).

The next sections describe the technologies used and their integration in the new Drupal model. The recognition of species names in scientific papers using the state-of-the-art and open source Linnaeus tool [13] is presented as an example of application.

2 Apache Software Foundation Information Retrieval and Extraction Initiatives

Apache organization supports some of the most important open source projects for the Web [14]. The Web server recommended to run Drupal CMS/F is the Apache HTTP Server [15]. Now, we want to take advantage of Apache Lucene, Apache Solr and Apache UIMA to incorporate IR and TM capabilities in Drupal CMS/F.

The Apache Lucene and Apache Solr are two distinct Java projects that have joined forces to provide a powerful, effective, and fully featured search tool. Solr is a standalone enterprise search server with a REST-like API [16] and Lucene is a high-performance and scalable IR library [17]. Due to its scalability and performance, Lucene is one of the most popular, free IR libraries [17, 18]. Besides the inverted index for efficient document retrieval, Lucene provides search enhancing features, namely: a rich set of chainable text analysis components, such as tokenizers and language-specific stemmers; a query syntax with a parser and a variety of query types

that support from simple term lookup to fuzzy term matching; a scoring algorithm, with flexible means to affect the scoring; and utilities such as the highlighter, the query spell-checker, and "more like this" [16].

Apache Solr can be seen as an enabling layer for Apache Lucene that extends its capabilities in order to support, among others: external configuration via XML; advanced full-text search capabilities, standard-based open interfaces (e.g. XML, JSON and HTTP); extensions to the Lucene Query Language; and, Apache UIMA integration for configurable metadata extraction (http://lucene.apache.org/solr/features.html).

Originally, the Apache UIMA started as an IBM Research project with the aim to deliver a powerful infrastructure to store, transport, and retrieve documents and annotation knowledge accumulated in NLP pipeline systems [19]. Currently, Apache UIMA supports further types of unstructured information besides text, like audio, video and images and is a *de facto* industry standard and software framework for content analysis [20]. Its main focus is ensuring interoperability between the processing components and thus, allowing a stable data transfer through the use of common data representations and interfaces.

3 New Supporting Drupal NER Module

We looked for a tight integration of the aforementioned Apache technologies in order to provide the basic means to deploy any NER task, namely those regarding basic natural language processing and text annotation (Fig.1). Using XML as common document interchange format, the new module allows the incorporation of third-party NER tools through pre-existent or newly developed Apache UIMA wrappers.

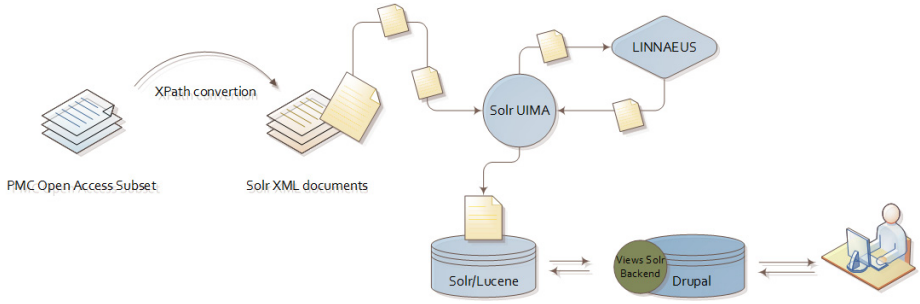

Fig. 1. Interoperation of Apache technology and third-party NER tools in the Drupal NER module

The proof of concept was the development of a Drupal CMS/F that retrieves scientific articles from the PubMed Central Open Access subset (PMC-OA) [21] and performs automatic recognition of organism mentions to enable further organism-driven document screening. The next subsections detail the interoperation of the different technologies and the integration of the Linnaeus UIMA NER wrapper as means to deploy such CMS/F.

3.1 Document Retrieval and Indexing

Documents are retrieved from the PMC-OA through the FTP service. An XSLT stylesheet is used to specify the set of rules that guide document format transformation. Notably, the XML Path (XPath) language is used to identify matching nodes and navigate through the elements and attributes in the PMC-OA XML documents.

After that, the Apache Solr engine is able to execute the UIMA-based NER pipeline to identify textual references of interest (in this case, organism names) and produce a list of named entities. The textual references are included in the metadata of the documents, and the entities recognized are added to the Apache Lucene index as means to enable further organism-specific document retrieval by the Drupal application.

3.2 Document Processing and Annotation

Apache UIMA supports the creation of highly customized document processing pipelines [22]. At the beginning of any processing pipeline is the Collection Reader component (Fig. 2), which is responsible for document input and interaction. Whenever a document is processed by the pipeline, a new object-based data structure, named Common Analysis Structure (CAS), is created. UIMA associates a Type System (TS), like an object schema for the CAS, which defines the various types of objects that may be discovered in documents. The TS can be extended by the developer, permitting the creation of very rich type systems.

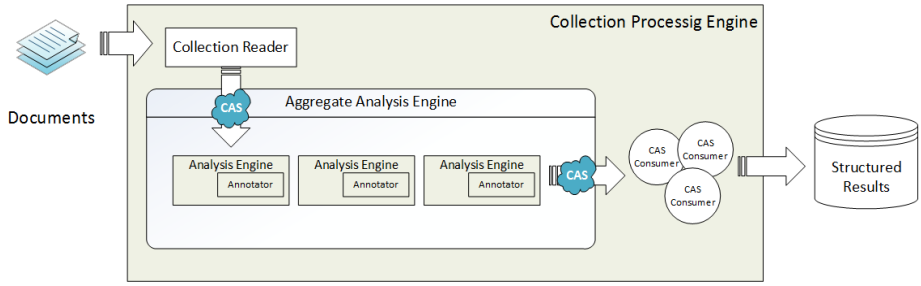

Fig. 2. High-Level UIMA Component Architecture

This CAS is processed throughout the pipeline and information can be added to the object by the Analysis Engine (AE) at different stages. The UIMA framework treats AEs as pluggable, compatible, discoverable, managed objects that analyze documents as needed. An AE consists of two components: Java classes, typically packaged as one or more JAR files, and AE descriptors, consisting of one or more XML files. The simplest type of AE is the primitive type, which contains a single annotator at its core (e.g. a tokenizer), but AEs can be combined together into an Aggregate Analysis Engine (AAE). The basic building block of any AE is the Annotator, which comprises the analysis algorithms responsible for the discovery of the desired types and the CAS update for upstream processing.

At the end of the processing pipeline are the CAS Consumers, which receive the CAS objects, after they have been analyzed by the AE/AAE, and conduct the final CAS processing.

3.3 Integration of Third-Party NER Tools

Apache UIMA supports seamless integration of third-party TM tools such as NER tools. Indeed, there already exist UIMA wrappers for several state-of-the-art NER tools, such as the organism tagger Linnaeus [23].

The first step to create an UIMA annotator wrapper is to define the AE Descriptor, i.e. the XML file that contains the information about the annotator, such as the configuration parameters, data structures, annotator input and output data types, and the resources that the annotator uses. The UIMA Eclipse plug-ins help in this creation by auto-generating this file based on the options configured in a point and click window (Fig. 3 - A).

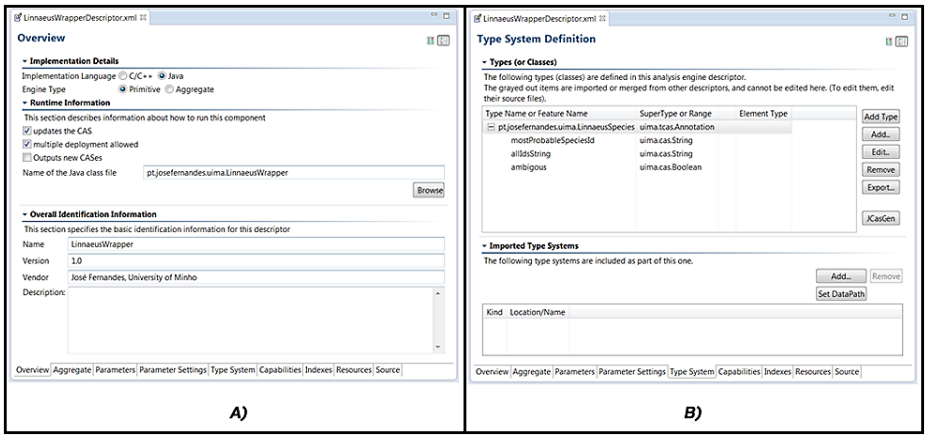

Fig. 3. UIMA Eclipse plug-in windows

The AE is then able to load the annotator in the UIMA pipeline, and the next step is to define the TS, namely the output types produced by the annotator, as described in the AE descriptor file (Fig. 3 - B).

```
    public void process(JCas cas) throws AnalysisEngineProcessException
{
        String text = cas.getDocumentText();
        List<Mention> mentions = matcher.match(text);
        for (Mention mention : mentions) {
            String mostProbableID = mention.getMostProbableID();
            String idsToString = mention.getIdsToString();
            LinnaeusSpecies species = new LinnaeusSpecies(cas);
            species.setBegin(mention.getStart());
            species.setEnd(mention.getEnd());
            species.setMostProbableSpeciesId(mostProbableID);
            species.setAllIdsString(idsToString);
            species.setAmbigous(mention.isAmbigous());
            species.addToIndexes();
        }
    }
```

Fig. 4. process() method of the LinnaeusWrapper.java class

The implementation of AE's Annotator is based on the standard interface AnalysisComponent. Basically, the wrapping of third-party tools implies the implementation of the annotator process( ) method, i.e. the desired Annotator logic (Fig. 4).

The CAS Visual Debugger is useful while implementing and testing the UIMA wrappers, in particular the annotators (Fig. 5). After wrappers are fully functioning the pipeline is ready to be used.

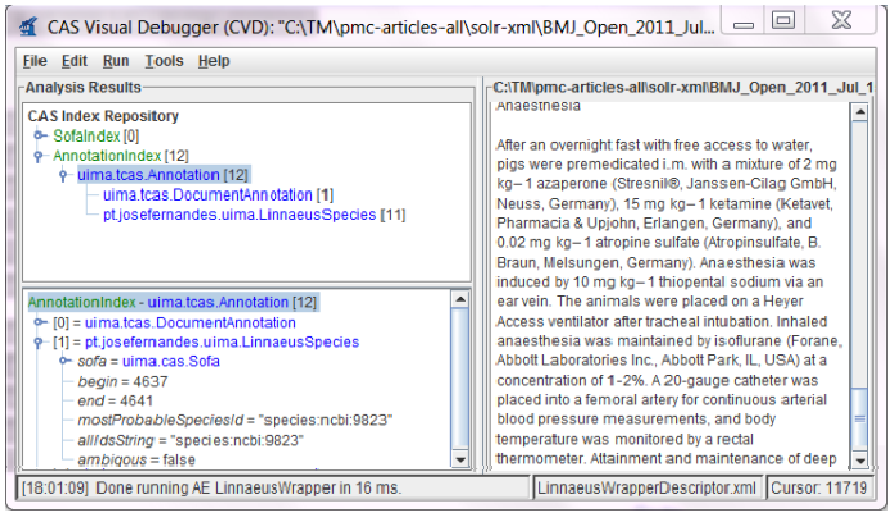

Fig. 5. Linnaeus UIMA wrapper running on a PMC Open Access article

3.4 Integration between Drupal and Apache Solr

Drupal allows developers to alter and customize the functionality of almost all of its components. Here, we developed a new Drupal module, named Views Solr Backend (https://drupal.org/project/views_solr_backend), to allow the easy and flexible querying of the Apache Solr index. The module is written in PHP and uses the APIs available for Drupal Views and Solarium, an Apache Solr client library for PHP applications [24]. This Drupal module can be easily configured and is even able to support different configurations for multiple Solr hosts (Fig. 6). Notably, it enables the administration of network parameters, the path to connect to the Solr host, and other parameters regarding the presentation of the search results in Drupal.

After configuration, it is possible to query any Solr schema, i.e. all indexed documents and the corresponding annotations. Therefore, our Drupal's Views module simplifies custom query display while increasing the interoperability with the CMS.

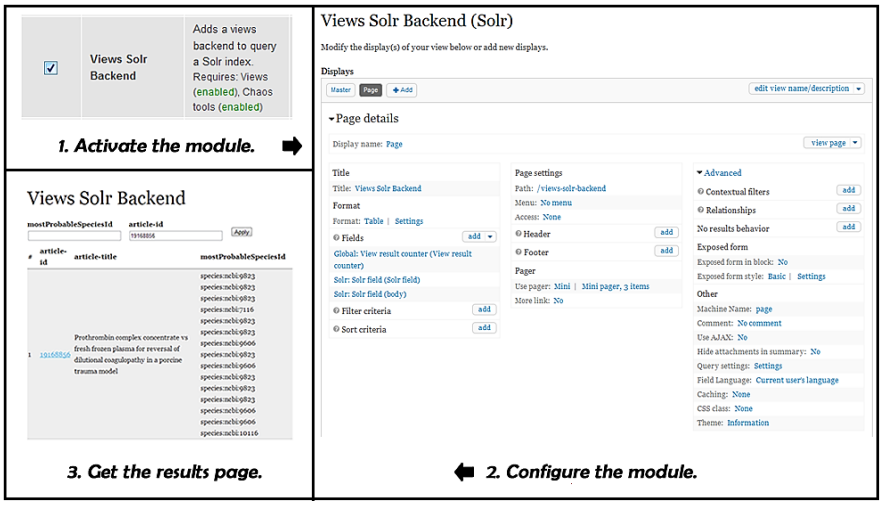

Fig. 6. Drupal module setup and presentation

4 Conclusions and Future Work

Drupal is a powerful and agile CMS/F that suits a number of development efforts in the Biomedical domains. Given the increasing demand for automatic document processing in support of the population of biomedical knowledge systems, BioTM has become almost a required module in such a framework.

This work addressed this need through the exploitation of major open source IR and IE initiatives. The new Drupal Views Solr Backend module, available at Drupal.org website (https://drupal.org/project/views_solr_backend), integrates Apache Solr with Drupal and thus, enables the implementation of customized search engines in Drupal applications. Moreover, the Apache Solr UIMA plug-in developed here for the Linnaeus NER tool exemplifies the integration of third-party NER tools in the document analysis processes of Apache Solr, granting a powerful and seamlessly means of specialized document annotation and indexing.

Acknowledgements. This work was supported by the IBB-CEB, the Fundação para a Ciência e Tecnologia (FCT) and the European Community fund FEDER, through Program COMPETE [FCT Project number PTDC/SAU-SAP/113196/2009/FCOMP-01-0124-FEDER-016012], and the Agrupamento INBIOMED from DXPCTSUG-FEDER unha maneira de facer Europa (2012/273). The research leading to these results has received funding from the European Union's Seventh Framework Programme FP7/REGPOT-2012-2013.1 under grant agreement n° 316265, BIOCAPS. This document reflects only the author's views and the European Union is not liable for any use that may be made of the information contained herein.

References

1. Kano, Y., Baumgartner, W.A., McCrohon, L., et al.: U-Compare: share and compare text mining tools with UIMA. Bioinformatics 25, 1997–1998 (2009), doi:10.1093/bioinformatics/btp289
2. Fan, W., Wallace, L., Rich, S., Zhang, Z.: Tapping the power of text mining. Commun. ACM 49, 76–82 (2006), doi:10.1145/1151030.1151032
3. Gemert, J.: Van Text Mining Tools on the Internet An overview. Univ. Amsterdam 25, 1–75 (2000)
4. Lourenço, A., Carreira, R., Carneiro, S., et al.: @Note: A workbench for biomedical text mining. J. Biomed. Inform. 42, 710–720 (2009), doi:10.1016/j.jbi.2009.04.002
5. Hucka, M., Finney, A., Sauro, H.: A medium for representation and exchange of biochemical network models (2003)
6. Lu, Z., Hirschman, L.: Biocuration workflows and text mining: overview of the BioCreative, Workshop Track II. Database (Oxford) 2012:bas043 (2012), doi:10.1093/database/bas043
7. Feinerer, I., Hornik, K., Meyer, D.: Text Mining Infrastructure in R. J. Stat. Softw. 25, 1–54 (2008), doi:citeulike-article-id:2842334
8. Fernández-Suárez, X.M., Rigden, D.J., Galperin, M.Y.: The 2014 Nucleic Acids Research Database Issue and an updated NAR online Molecular Biology Database Collection. Nucleic Acids Res. 42, 1–6 (2014), doi:10.1093/nar/gkt1282
9. Papanicolaou, A., Heckel, D.G.: The GMOD Drupal bioinformatic server framework. Bioinformatics 26, 3119–3124 (2010), doi:10.1093bioinformatics/btq599
10. Decker, S., Melnik, S., van Harmelen, F., et al.: The Semantic Web: the roles of XML and RDF. IEEE Internet Comput. 4, 63–73 (2000), doi:10.1109/4236.877487
11. Rebholz-Schuhmann, D., Kafkas, S., Kim, J.-H., et al.: Monitoring named entity recognition: The League Table. J. Biomed Semantics 4, 19 (2013), doi:10.1186/2041-1480-4-19
12. Rzhetsky, A., Seringhaus, M., Gerstein, M.B.: Getting started in text mining: Part two. PLoS Comput. Biol. 5, e1000411 (2009), doi:10.1371/journal.pcbi.1000411
13. Gerner, M., Nenadic, G., Bergman, C.M.: LINNAEUS: A species name identification system for biomedical literature. BMC Bioinformatics 11, 85 (2010), doi:10.1186/1471-2105-11-85
14. Fielding, R.T., Kaiser, G.: The Apache HTTP Server Project. IEEE Internet Comput. (1997), doi:10.1109/4236.612229
15. Web server | Drupal.org., `https://drupal.org/requirements/webserver`
16. Smiley, D., Pugh, E.: Apache Solr 3 Enterprise Search Server, p. 418 (2011)
17. McCandless, M., Hatcher, E., Gospodnetic, O.: Lucene in Action, Second Edition: Covers Apache Lucene 3.0, p. 475 (2010)
18. Konchady, M.: Building Search Applications: Lucene, LingPipe, and Gate, p. 448 (2008)
19. Ferrucci, D., Lally, A.: UIMA: An architectural approach to unstructured information processing in the corporate research environment. Nat. Lang. Eng. (2004)
20. Rak, R., Rowley, A., Ananiadou, S.: Collaborative Development and Evaluation of Text-processing Workflows in a UIMA-supported Web-based Workbench. In: LREC (2012)
21. Lin, J.: Is searching full text more effective than searching abstracts? BMC Bioinformatics 10, 46 (2009), doi:10.1186/1471-2105-10-46
22. Baumgartner, W.A., Cohen, K.B., Hunter, L.: An open-source framework for large-scale, flexible evaluation of biomedical text mining systems. J. Biomed. Discov. Collab. 3, 1 (2008), doi:10.1186/1747-5333-3-1
23. Móra, G.: Concept identification by machine learning aided dictionary-based named entity recognition and rule-based entity normalisation. Second CALBC Work
24. Kumar, J.: Apache Solr PHP Integration, p. 118 (2013)