Computer Science • 13 (2) 2012 http://dx.doi.org/10.7494/csci.2012.13.2.21

Sandra Bergmann
Mathilde Romberg
Alexander Klenner
Marc Zimmermann

INFORMATION EXTRACTION FROM CHEMICAL PATENTS

Abstract *The development of new chemicals or pharmaceuticals is preceded by an in-depth analysis of published patents in this field. This information retrieval is a costly and time inefficient step when done by a human reader, yet it is mandatory for potential success of an investment. The goal of the research project UIMA-HPC is to automate and hence speed-up the process of knowledge mining about patents. Multi-threaded analysis engines, developed according to UIMA (Unstructured Information Management Architecture) standards, process texts and images in thousands of documents in parallel. UNICORE (UNiform Interface to COmputing Resources) workflow control structures make it possible to dynamically allocate resources for every given task to gain best cpu-time/real-time ratios in an HPC environment.*

Keywords information extraction, chemical patents, HPC, workflows, UNICORE, UIMA

1. Introduction

In life sciences the gathering of relevant information from the literature is a key issue. In order to build a new hypothesis all knowledge about an organism, a disease, a gene or a chemical reaction has to be collected and analyzed. Although corresponding publications can be quite easily searched in huge bibliographic databases like PubMed[1] or Google Scholar[2], the scientist still has to read large collections of research papers, PhD theses, books, and patents to get an overview of the state of the art. It is easier to find too many documents on a life science topic than to find the right information inside these documents [2]. Over the last decade several automatic methods have been developed to support information extraction in the life sciences. A number of commercial organizations (e.g., TEMIS, Linguamatics, Notiora, IBM, SureChem, and InfoChem) have developed algorithms for chemical named entity recognition [12], [13]. Chemicals can appear in documents as systematic chemical names (e.g. IUPAC), generic names, trade names abbreviations, and acronyms, company codes and database registry numbers (e.g. CASRN), and in chemical depictions. Systems for the identification and extraction of chemicals from text are described in [5]-[7]. Examples for extraction of chemicals from depictions are [1], [4], [10]. In order to extract all relevant information complex workflows have to be assembled [8]. Existing frameworks for the assembly of information extraction software are Apache UIMA[3] and GATE[4]. UIMA (Unstructured Information Management Architecture) defines standardized interfaces and allows for multithreading. In order to make use of large distributed computing resources grid middleware like Globus[5] and UNICORE [11] have been developed. For example UNICORE (Uniform Interface to Computing Resources) offers sophisticated workflow features as well as built-in application support.

In the life sciences often large collections of data have to be processed, therefore cooperation between industry partners offering computing services and clinical partners are quite common: IBM[6], a leading enterprise in software and different other fields, together with the Mayo Clinic, a nonprofit worldwide leader in medical care, established a shared project in the extraction of hidden knowledge from health data. For the simulation of diseases and treatment modalities they use algorithms and flexible computing power from IBM's computer Blue Gene [6].

In the UIMA-HPC project[7] we are focusing on a use case defined by our collaboration partner Taros Chemicals[8]. The following steps are processed to extract information from chemical patents:

[1]Web site: `http://www.ncbi.nlm.nih.gov/pubmed`
[2]Web site: `http://scholar.google.com/`
[3]Web site: `http://uima.apache.org/`
[4]Web site: `http://gate.ac.uk/ie/`
[5]Web site: `http://www.globus.org/`
[6]Web site: `http://www.ibm.com`
[7]Web site: `http://www.uima-hpc.de/en/about-uima-hpc.html`
[8]Web site: `http://www.taros.de`

- Finding all 'relevant' chemical terms and depictions, annotate their position, and reconstruct the chemical structure.
- Highlighting of annotations inside the original PDF document for visual inspection.
- Enrichment with additional information (i.e. references to online databases).
- Adding chemical bookmark structure for easier accessibility of compounds.
- Extraction of cross-references inside the document.

In this paper we present an example workflow based on the integration of UIMA in UNICORE and show first results for a test corpus of patent documents from the European Patent Office (EPO[9]).

The current section introduces the foundations of information extraction and discusses related work. Section 2 describes the basic frameworks, UIMA and UNICORE, and their interaction. The application scenario to be run on the prototype system and first results are covered in Section 3 and 4, respectively. Finally, Section 5 concludes the paper and gives an outlook of the next steps.

2. Method section

The distinctive feature of UIMA-HPC is the flexible generic approach which makes it applicable to any kind of UIMA-Pipeline and workflow thereof as well as any kind of computing resources, which are available.

2.1. Implementation of UIMA-Pipelines

UIMA-Pipelines are the basic building blocks of information extraction workflows. Apache UIMA provides a native Java[10] framework for mining unstructured data. An UIMA application is organized as a Collection Processing Engine (CPE) that consists of an UIMA Collection Reader (CR), one or more UIMA Analysis Engines (AEs), and one Collection Consumer (CC) (Figure 1). The analyzed artifact (e.g. text or binary data) is stored in the internal UIMA data structure Common Analysis Structure (CAS). The framework architecture also provides convenience methods for serializing CAS objects (XCAS) to store them persistently on hard disks. These stored XCAS files can then again be read by an CR. In our implementation we exploit this procedure to transport data between physically separated hardware nodes.

All UIMA components (CPE, CR, AE, and CC) are specified via XML file format descriptors, which contain consistent predefined internal routes. For a Grid system, we need a dynamic handling of network paths. Therefore we used the UIMAFit [9] implementation to generate all XML specifications at run-time of an UIMA-Pipeline. The necessary import of uniform resource identifiers (URIs) in all Java classes of UIMA can be dynamically adapted to any location using UIMAFit. All our integrated

[9]Web site: `http://www.epo.org/`
[10]Web site: `http://java.com/en/`

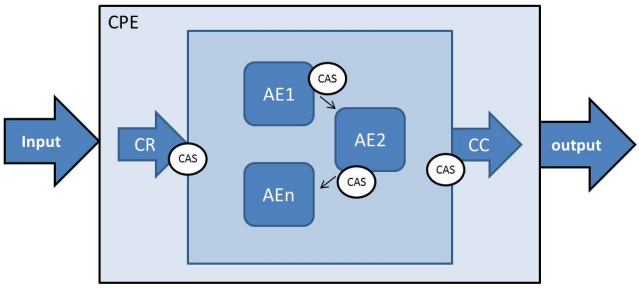

Figure 1. The input is converted into CAS by a collection reader (CR) and further processed by a number of analysis engines (AE) and finally written back by a collection consumer (CC)

pipelines are provided as Java archive files (jar) and run platform independent on different operating systems. The framework architecture UIMA makes it possible to easily integrate existing software and also exchange different AEs within different UIMA-Pipelines. Table 1 shows all UIMA-Pipelines that have been developed and their respective functions.

Table 1

Implemented UIMA-Pipelines to process documents with medical and chemical content

Input	**Integrated 3rd Party Software**	**Function**	**Annotations**	**Output**
PDF	CLI abbyy[11] finereader	OCR	SourceDocument Information	XCAS
PDF	PDFbox[12], iText[13]	Text extraction	SourceDocument Information	XCAS
XCAS	ProMiner[14]	Dictionary based Annotation	Chemistry, Diseases, Genes	XCAS
XCAS	Linda	Machine Learning (ML) based Annotation	Diseases, Genes, IUPAC-terms	XCAS
XCAS	OSCAR	Dictionary and ML based annotation of chemical terms	Chemical terms	XCAS
XCAS	iText, PDFBox	Generating annotated PDF	—	Enriched PDF

[11] Web site: `http://www.abbyy.de/`
[12] Web site: `http://pdfbox.apache.org/`
[13] Web site: `http://itextpdf.com/`
[14] Web site: `http://www.scai.fraunhofer.de/en/business-research-areas/bioinformatics/products/prominer.html`

2.2. UNICORE

UNICORE is an open source Grid middleware, which provides uniform access to distributed, heterogeneous resources in order to perform computations and manage data [11]. It is implemented in Java and consists of services for job and data handling, security, workflows, and orchestration on the server side and of a graphical client, a command line client, and a client API. In the context of UIMA-HPC with its goal to use HPC resources in order to significantly shorten time to solution, UNICORE's workflow and scheduling features are important. The UIMA-Pipelines described in the previous section can be combined in a workflow, which defines the steps to be taken from the raw document data to the fully annotated and searchable data. Through its workflow control functionality UNICORE is capable of starting as many pipelines as necessary to fully load a large computer system and as few pipelines as necessary to guarantee a certain run time per pipeline to make them comply with HPC system requirements. UNICORE supports applications through graphical interfaces on the client side, so called GridBeans, and application resource descriptions on the server side. When selecting a resource for execution the scheduling mechanism chooses those compute resources which offer the requested application and then takes into account criteria such as system load to determine the system to send the job to. In case of UIMA-HPC also the number of jobs and their respective set of input data are determined. UNICORE cares for controlling the workflow execution according to sequence and iteration definition and manages the necessary data transfers within the workflow.

2.3. UIMA and UNICORE

In order to make UIMA-Pipelines available on distributed, heterogeneous resources to be accessible through UNICORE they have to meet certain requirements:

- Installed on the target system.
- Executable as stand-alone applications.
- No hard-coded paths in file descriptors.

The overall architecture is shown in Figure 2. As UIMA is a native Java library it is cross platform compatible and can be installed on UNIX and Microsoft Windows based servers. The prerequisite is an installation of Java Virtual Machine 6 or higher. A UIMA-Pipeline is provided as a Java jar archive which has to be available on a server's file system. Input and output data format is defined in XML, serialised CAS objects (C). This must be unified to be free in the choice of annotations and their order in a workflow. The Java archive is made available through UNICORE by defining it as an application resource (B). Upon execution the jar archive is called by UNICORE via a system call using the standard arguments of the Java virtual machine. The XML application configuration files support any number of arguments that can be defined prior to execution separately for every job on the client side. UIMA provides the multithreading of embedded components. This allows us to exploit all cores of a node in the execution environment.

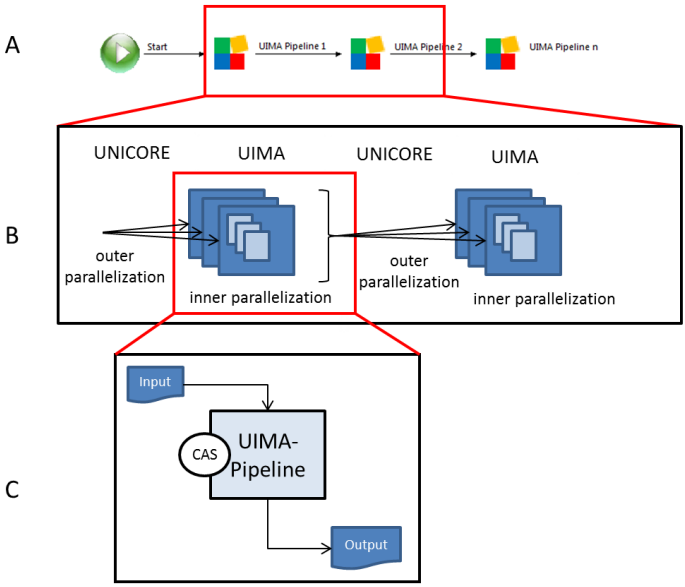

Figure 2. Complete architecture of the coupling between UIMA and UNICORE

To parallelize the execution of applications and exploit available nodes on HPC resources, UNICORE workflow control structures can be used. Each application can be addressed in a related GridBean in the UNICORE Rich Client (URC) [3] (A). Any application specific argument can be fulfilled via graphical panels provided by the GridBean. The UNICORE Rich Client offers a workflow editor to create and specify the execution order of applications. In (A) three UIMA-Pipelines (AE_1 to AE_n) are defined in sequential order. Each UIMA-Pipeline is created by the related GridBean, symbolised by the UIMA logo.

3. Application scenario

3.1. Annotations of IUPAC terms and trivial names

Working with patents in a scientific context has specific requirements that were evaluated with our partners from industry. We need to find all 'relevant' terms and depictions, annotate their position, reconstruct the chemical structure, and highlight all annotations for visual inspection. Additional information such as references to online data bases, cross references inside the document, and a chemical bookmark structure for easier accessibility are also required. In this context, the definition of 'relevant' terms is dependent on the specific requirements of the scientist. Here, we scan the documents for chemical structures (IUPAC terms and trivial names) in PDF documents.

To demonstrate the applicability of our approach we generate a text corpus consisting of 60 patents from the EPO (European Patent Office). The patents are chosen for their relevance, diverse scan quality, and different notations of IUPAC terms by a synthesis chemist expert from Taros Chemical GmbH. All 60 patents were analyzed according the automated extraction of IUPAC annotations. However, our modular architecture makes it possible to choose different UIMA-Pipelines to satisfy specific demands (i.e. a redefinition of 'relevant' terms) for other use cases such as the following one.

3.2. ProMiner applications

One of the successfully integrated and tested UIMA-Pipelines is ProMiner. The annotations are based on dictionaries, thesauri or large controlled vocabularies derived from ontologies. Some advantages of ProMiner are: context dependent disambiguation of biomedical termini, and the resolution of acronyms, specific handling of common English word synonyms, and the recognition of spelling variants of expressions in the source dictionary. Currently three different ProMiner annotation tools are integrated as UIMA-Pipelines, each of them depends on a different topic specific dictionary: ProMiner Human, ProMiner Disease, and ProMiner Drugbank. In order to demonstrate the exploitation of available cores, we performed ProMiner Human and ProMiner Drugbank UIMA-Pipelines on 60 patents from EPO.

4. Application scenario – results

4.1. Analysis of extracted chemical terms with IUPAC

As a preliminary result we are able to extract an absolute number of 58109 IUPAC terms and trivial names from the chosen 60 patents that were enriched with structure information. 9.45 percent of these represent single chemical elements (5491 occurences of elements) and the remaining 52.618 identified chemical entities represented 10523 unique chemical structures. For all molecules a unique InChI key has been generated in order to check for duplicates. All molecules have been converted into the SMILES format. SMILES is a linear non-unique notation of the connection table which lists all atoms in the order of a graph walk. Given that the length of the SMILES string is correlated with the complexity of the underlying molecule, we conclude that our method is capable of identifying and annotating complex chemistry in the scanned image-only PDFs. Figure 3 shows the distribution of extracted chemical terms according to their complexity for all extracted terms and unique terms, respectively. At this point we are not able to show values for the recall and precision of our annotation pipeline since we have no fully human annotated test corpus yet. The text and structure analysis has been executed on local computing resources and the current test bed, which consists of two computer systems, a server with 4 cores, and a 44 nodes cluster with 176 cores.

As a first result we decreased a runtime from 3.4 hours (single core) to 21 minutes (6 jobs in parallel, 4 cores per node). The runtime of 21 min is not yet optimal due to static parallelization through UNICORE. In the future we will develop a dynamic method to compute the necessary number of parallel jobs.

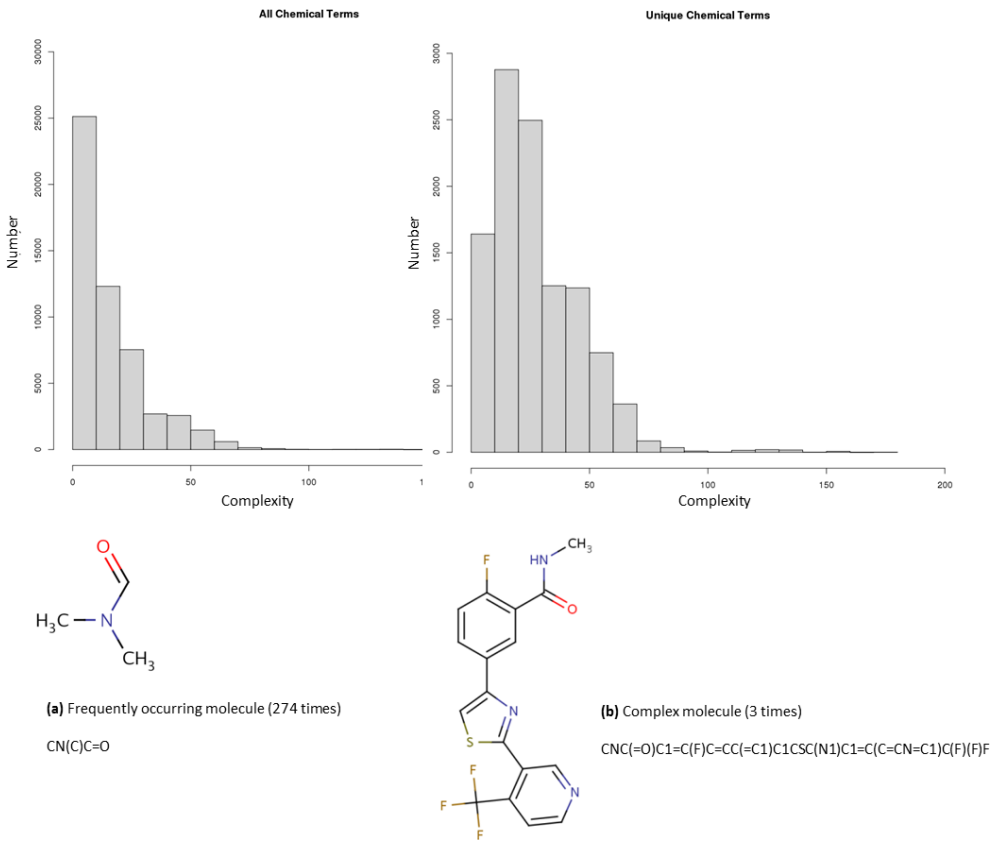

Figure 3. The frequency is measured as occurrences in the text in all documents. The complexity is measured as the length of the SMILES string. Complex molecules such as those shown in (a) occur more frequently than simple molecules like those presented in (b)

4.2. Results of run time analyses of ProMiner application

UIMA provides the multithreading characteristic for embedded applications. Figure 4 and Figure 5 show how a runtime, split into initialisation and processing time, changes with the number of exploited cores. 10 identical runs were executed per number of cores. The diagrams contain the mean time for each number of cores.

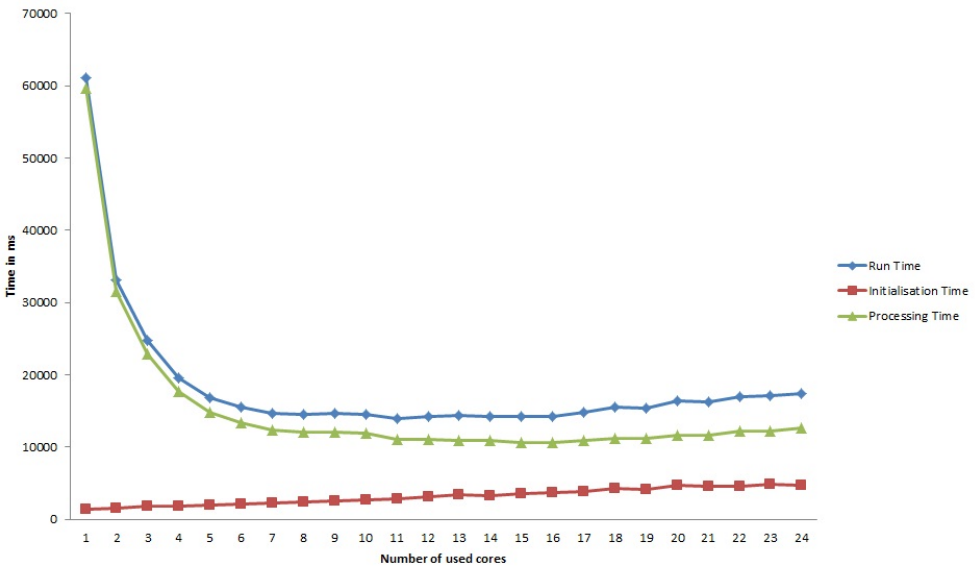

Figure 4. ProMiner Drugbank: On the x-axis the number of cores is plotted and on the y-axis the processing time (in milli seconds)

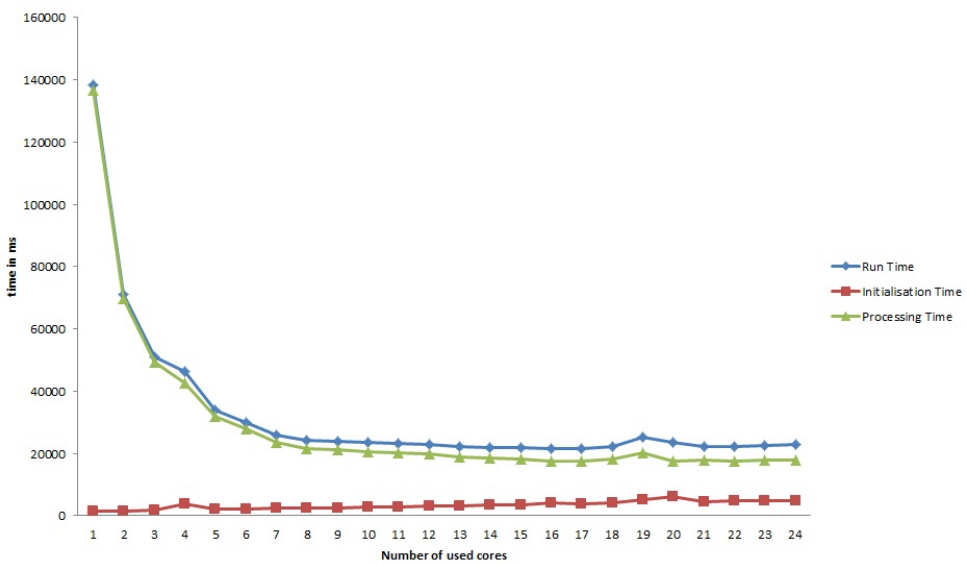

Figure 5. ProMiner Human: On the x-axis the number of cores is plotted and on the y-axis the processing time in milli seconds

For ProMiner Drugbank, Figure 4 shows that without any parallel execution a runtime of 1.02 minutes has been measured. By using two cores the runtime is reduced by 50%. Afterwards, the runtime reduced slower and as presented in Figure 4 the minimum is reached with 11 cores. To use more than 11 cores does not gain any further benefit.

For ProMiner Human, a runtime of 2.3 minutes has been measured without any parallel execution. By using two cores the runtime is reduced by 50% similar to the runtime of the ProMiner Drugbank application. Afterwards, the runtime is reduced slower as shown in Figure 5. The minimum is reached with 17 cores. The processing time descends similar to the runtime whereas the initialisation time ascends linear to the number of cores. A peak is recognizable at 19 cores for processing time and runtime, at 20 cores for initialisation time. The reasons for this peak are not yet known. The optimization of this application is another task of the UIMA-HPC project.

These analyses are useful to identify parameters for the optimization of runtime for jobs.

5. Conclusions and outlook

This paper shows the approach UIMA-HPC takes to enable information extraction on HPC resources. The central idea is to combine the two frameworks UIMA and UNICORE. The first is used to clearly define interfaces for all pieces of analysis software and multithreading capability while the second framework contributes workflow and scheduling features. In a first step UIMA and UNICORE have been combined and information extraction software components have been implemented as UIMA-Pipelines. Simple workflows consisting of control structures and UIMA-Pipelines have been executed. First experiments show a decreasing runtime by using UIMAs multithreading characteristics. In the next step more complex workflows will be developed, tested, and improved to achieve the goal of shorter time to solution. A scheduling algorithm making sure that the load is equally shared among the available resources will be developed. Gradually we will increase input data collections and add more computing resources to the test bed to demonstrate the scalability of the solution.

Acknowledgements

UIMA-HPC is partly funded by the German Ministry of Education and Research (BMBF) under grant id 01IH11012A-D.

References

[1] Algorri M.-E., Zimmermann M., Friedrich C. M., Akle S., Hofmann-Apitius M.: *Reconstrucition of chemical molecules from images.* [in:] *Conference Proceedings of the International Conference of IEEE Engineering in Medicine and Biology Society*, vol. 2007 of *Proceedings of Annual International Conference of the*

IEEE Engineering in Medicine and Biology Society, pp. 4609–4612. Department of Digital Systems, Instituto Tecnologico Autonomo de Mexico, Mexico City. algorri@itam.mx, 2007.

[2] Banville D. L.: *Mining chemical structural information from the drug literature. Drug discovery today*, 11(1–2):35–42, January 2006.

[3] Demuth B., Schuller B., Holl S., Daivandy J., Giesler A., Huber V., Sild S.: *The unicore rich client: Facilitating the automated execution of scientific workflows.* [in:] *e-Science (e-Science), 2010 IEEE Sixth International Conference on*, pp. 238–245, dec. 2010.

[4] Filippov I. V., Nicklaus M. C.: *Optical structure recognition software to recover chemical information: OSRA, an open source solution. Journal of chemical information and modeling*, 49(3):740–3, March 2009.

[5] Hettne K. M., Stierum R. H., Schuemie M. J., Hendriksen P. J. M., Schijvenaars B. J. A., Mulligen E. M. V., Kleinjans J., Kors J. A.: *A dictionary to identify small molecules and drugs in free text. Bioinformatics (Oxford, England)*, 25(22):2983–91, November 2009.

[6] J. Yuan M.: *Watson and healthcare: How natural language processing and semantic search could revolutionize clinical decision support. developerWorks*, April 2011.

[7] Jessop D. M., Adams S. E., Murray-Rust P.: *Mining chemical information from Open patents. Journal of cheminformatics*, 3(1):40, October 2011.

[8] Kolluru B., Hawizy L., Murray-Rust P., Tsujii J., Ananiadou S.: *Using workflows to explore and optimise named entity recognition for chemistry. PloS one*, 6(5):e20181, January 2011.

[9] Ogren P., Bethard S.: *Building test suites for UIMA components.* [in:] *Proceedings of the Workshop on Software Engineering, Testing, and Quality Assurance for Natural Language Processing (SETQA-NLP 2009)*, pp. 1–4, Boulder, Colorado, June 2009. Association for Computational Linguistics.

[10] Park J., Rosania G. R., Shedden K. A., Nguyen M., Lyu N., Saitou K.: *Automated extraction of chemical structure information from digital raster images. Chemistry Central journal*, 3:4, January 2009.

[11] Streit A., Bergmann S., Breu R., Daivandy J., Demuth B., Giesler A., Hagemeier B., Holl S., Huber V., Mallmann D., Memon A. S., Memon M. S., Menday R., Rambadt M., Riedel M., Romberg M., Schuller B., Lippert T.: *UNICORE 6 – A European Grid Technology*, vol. 18. 2009.

[12] Warr W. A.: *Chemoinformatics and Computational Chemical Biology*, vol. 672 of *Methods in Molecular Biology*. Humana Press, Totowa, NJ, 2011.

[13] Zimmermann M., Fluck J., Thi L. T. B., Kolárik C., Kumpf K., Hofmann M.: *Information extraction in the life sciences: perspectives for medicinal chemistry, pharmacology and toxicology. Current Topics in Medicinal Chemistry*, 5(8):785–796, 2005.

Affiliations

Sandra Bergmann
Forschungszentrum Jülich GmbH, 52425 Jülich, Germany, `s.bergmann@fz-juelich.de`

Mathilde Romberg
Forschungszentrum Jülich GmbH, 52425 Jülich, Germany

Alexander Klenner
Fraunhofer-Institute for Algorithms and Scientific Computing, 53754 Sankt Augustin, Germany

Marc Zimmermann
Fraunhofer-Institute for Algorithms and Scientific Computing, 53754 Sankt Augustin, Germany

Received: 9.12.2011
Revised: 16.04.2012
Accepted: 23.04.2012