Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 38 (2014) 92 – 99

Procedia Computer Science

10th Italian Research Conference on Digital Libraries, IRCDL 2014

Information inference in scholarly communication infrastructures: the OpenAIREplus project experience

Mateusz Kobos[a]*, Łukasz Bolikowski[a], Marek Horst[a], Paolo Manghi[b], Natalia Manola[c], Jochen Schirrwagen[d]

[a]*Interdisciplinary Centre for Mathematical and Computational Modelling, University of Warsaw, Pawinskiego 5a, 02-106 Warsaw, Poland*
[b]*Istituto di Scienza e Tecnologie dell'Informazione "A. Faedo", Consiglio Nazionale delle Ricerche, via G. Moruzzi 1, 56124 Pisa, Italy*
[c]*Department of Informatics and Telecommunications, University of Athens, Panepistimiopolis, 15784 Ilissia, Athens, Greece*
[d]*Department of Library Technology and Knowledge Management, Bielefeld University, Universitätsstr. 25, 33615 Bielefeld, Germany*

Abstract

The Information Inference Framework presented in this paper provides a general-purpose suite of tools enabling the definition and execution of flexible and reliable data processing workflows whose nodes offer application-specific processing capabilities. The IIF is designed for the purpose of processing big data, and it is implemented on top of Apache Hadoop-related technologies to cope with scalability and high-performance execution requirements. As a proof of concept we will describe how the framework is used to support linking and contextualization services in the context of the OpenAIRE infrastructure for scholarly communication.

Peer-review under responsibility of the Scientific Committee of IRCDL 2014

Keywords: OpenAIRE infrastructure; data processing system; data mining; text mining; big data

1. Introduction

OpenAIREplus[1,2] delivers a scholarly communication infrastructure[† 3] devised to populate a graph-shaped information space of interlinked metadata objects describing publications, datasets, persons, projects, and

* Corresponding author. Tel.: +48 22- 87-49-419.
E-mail address: mkobos@icm.edu.pl
† http://ww.openaire.eu

Peer-review under responsibility of the Scientific Committee of IRCDL 2014
doi:10.1016/j.procs.2014.10.016

organizations by collecting and aggregating such objects from publication repositories, dataset repositories, and Current Research Information Systems. The infrastructure supports scientists by providing tools for dissemination and discovery of research results, as well as funding bodies and organizations, by providing tools for measuring and refining funding investments in terms of their research impact. In particular, the infrastructure offers tools to enrich metadata objects using inference mechanisms. Such mechanisms, by mining the graph of metadata and original files they describe (e.g. the publication), are capable of identifying new relationships between existing objects, e.g. by enriching them with research funding information, supporting data curation, e.g. fixing the title name, or identifying new objects, e.g. by finding missing authors of a publication object.

In this paper we present the Information Inference Framework (IIF) which was originally designed to underpin the OpenAIREplus infrastructure inference mechanisms and evolved into a general-purpose solution for processing large amounts of data in a flexible and scalable way. The IIF can be regarded as a platform for defining and executing data processing workflows of possibly distributed modules. To this aim, it provides generic tools for: (1) defining domain-specific workflow nodes, e.g. in case of OpenAIRE project: node for extracting metadata from PDF files, node for classifying documents; (2) providing well-defined way to pass the data between the nodes; (3) defining sequence of execution of the nodes as a concise workflow; and (4) executing the workflow in a distributed, reliable, and scalable computational environment. In order to obtain the benefits described in the last point, we use the Apache Hadoop[‡] system that allows for running applications on a computing cluster. IIF is an open source project and its source code is available in the project SVN repository[§].

In this paper we outline the design decisions and describe the components of this processing infrastructure framework for big data. After comparing our work in the context of other popular systems for processing scholarly data in the next section, we show how the framework is applied to enrich scholarly information in the OpenAIREplus project.

2. Related work

The IIF is a novel solution. Its contribution can be viewed from two perspectives: introducing a new generic framework for processing big data (see Sect. 3), and introducing a new system for processing and extracting knowledge from objects related to scientific activity: documents, author information etc. (see Sect. 4).

2.1. Data processing frameworks

The concepts underlying IIF were inspired by the Rapid Miner[4] open source software for data analytics. The tool allows for creating data workflows consisting of various workflow nodes using a user-friendly graphical interface. Its library contains many predefined workflow nodes related to Extract, Transform and Load (ETL) tasks and machine learning algorithms. An interesting feature of this tool is that it checks whether the data declared to be consumed by a workflow node matches the data declared to be produced by a preceding workflow node, which is done as early as during design time of the workflow. A disadvantage of Rapid Miner is that it was not designed to process large data sets. Our system, on the other hand, was designed from the beginning with scalability and big data in mind. Note that there is a plugin for Rapid Miner, called Radoop[5] that allows for processing big data by Rapid Miner, but this is neither an open source nor free solution and thus not as flexible and customizable as we would want it to be when applying it to our specific domain.

With relation to the first perspective, we can also compare the introduced solution with the ones that were devised to solve similar problems. Google Scholar[**], Microsoft Academic Search[††], ArnetMiner[‡‡], and CiteSeerX

‡ http://hadoop.apache.org/

§ The project is divided into Maven subprojects with names having "-iis-" infix that are available at https://svn-public.driver.research-infrastructures.eu/driver/dnet40/modules

** http://scholar.google.com

†† http://academic.research.microsoft.com

‡‡ http://arnetminer.org

which is based on SeerSuite[§§] are all popular projects dealing with harvesting and providing access to scientific publications and related data. The data processing architecture of ArnetMiner is described in references[6,7], and information related to CiteSeerX can be found in references[8,9]; unfortunately, we were not able to find any reliable publication describing the architecture of neither Microsoft Academic Search nor Google Scholar, so we cannot compare our solution with them. Generally, these projects follow a Service Oriented Architecture (SOA) approach where each processing module is a separate web service with its own data processing tools. The modules communicate with each other through well-defined networking channels. Our approach is different: each processing module is akin to a cluster application that uses the same Hadoop-based computational framework to execute its tasks and communicate with other modules. The first approach assures that the modules are very loosely coupled; however, the disadvantage is that each module has to define its own communication protocol and solution of processing and storing the data in an efficient way and to handle hardware failures. In our approach, the modules process the data in the way supported by the Hadoop cluster (note that this does not preclude using a separate web service as a processing module, though it is not as effective as using a native, Hadoop-based approach). This way we can circumvent the mentioned problems with the SOA architecture. See Sect. 3 for a more detailed discussion of various solutions applied in the IIF along with comparison with other projects.

2.2. Inferring knowledge over scholarly communication content

A number of frameworks with objectives and/or architectures similar to IIF can be mentioned. Arguably the two most prominent, complementary architectures for content analysis are Apache UIMA[10] and GATE[11], which are mature, open-source suites of tools for text engineering. Several projects and higher-order frameworks are using UIMA or GATE, for example: Behemoth[***]; or XAR[12], an open-source framework for information extraction. The IIF addresses the need of a large-scale, flexible, open-source architecture to execute complex workflows for processing documents (e.g. scientific publications) and finding links between extracted entities; none of the existing solutions can fully satisfy these requirements of the OpenAIREplus project.

Another set of solutions related to the second perspective is formed by popular projects related to scientific publications mentioned previously (Google Scholar, Microsoft Academic Search, ArnetMiner, and CiteSeerX). When comparing data extraction and inference functionality of these systems with the IIF as used in OpenAIREplus, we can see that some types of features are shared – although the details differ – between them (extracting metadata from PDF documents, creating citation links between documents, finding similar documents and clustering documents, labeling the documents, generating statistics), some features are not present in the IIF, e.g. finding expert in given field (ArnetMiner), finding topics relevant to the query (ArnetMiner), organization comparison (Microsoft Academic Search), showing co-authorship graph (Microsoft Academic Search Engine), but some features are present in the IIF only: extracting links to projects and related funding source from a document, finding similarities between documents based on activity of users of the OpenAIRE portal, extracting links to related data sets from a document. The last one allows OpenAIREplus users to easily access research data described in publications and is of paramount importance since the scientific research and scholarly communication is becoming increasingly data-centric.

3. The Information Inference Framework

The IIF is essentially a flexible data processing system for big data. It was designed to fulfil the requirements of the OpenAIREplus project, but built to be generic and not restricted to this specific application domain. The IIF provides tools for defining data processing workflows which consist of: (1) workflow nodes responsible for data processing, (2) data stores that serve as means of communication between workflow nodes, and (3) tools for executing a workflow.

[§§] http://citeseerx.ist.psu.edu, http://citeseerx.sourceforge.net

[***] https://github.com/DigitalPebble/behemoth

As mentioned above, the framework was implemented on top of Apache Hadoop in order to benefit from its out-of-the-box facilities to scale in terms of size and in terms of distributed computation. In Apache Hadoop, data files are stored in its distributed file system (HDFS) which is designed to flexibly scale by simply integrating new nodes to the underlying cluster. Moreover, the business logic of workflow nodes is written using the MapReduce programming paradigm and their execution is automatically distributed over the cluster nodes. Last but not least, hardware failures of the machines belonging to the cluster are handled automatically.

In the following part, we will describe workflow nodes, data stores, and workflows along with related design and technological decisions and compare them with alternative solutions applied in other systems.

3.1. Data stores

The data objects passed between workflow nodes are named "data stores". Each data store corresponds to a list of objects, where each object is of the same type, i.e. it is defined by the same schema. We considered a few alternative serialization formats for data stores: plain CSV files, JSON, Thrift, Protocol Buffers, and Avro. Among these we have chosen the Avro datafile[†††] since it is characterized by a combination of useful features which are unique among these technologies: it defines a schema being kept with the data it describes, it uses an efficient binary storage format, it is forward/backward compatible, and it is well integrated with the Hadoop ecosystem (e.g. it is handled out-of-the-box by MapReduce jobs and Hadoop Pig scripts).

Data stores are immutable, i.e. they cannot be modified after they are created. This approach has two main advantages. First, since the data is read-only, it removes the need for synchronization. This is very important since problems with synchronization of access to a shared file system (e.g. in CiteSeerX) result in frequent freezes of the system (see[8]). Second, it makes reasoning about the state of the system as well as debugging and restoring of the workflow state easier, thereby facilitating workflow nodes development. This solution is akin to the approach of "sharing memory by communicating" applied in the concurrency-centered system language Go[‡‡‡]. The side-effect of immutability is data replication, which can be tackled by a garbage collector set to remove useless data stores.

Data stores are kept in the Hadoop distributed file system (HDFS). Each data store is represented by a single HDFS directory which contains a set of Avro datafiles with the same schema. Before chosing HDFS, we considered and compared several types of databases. For example ArnetMiner stores its data in a RDF data base backed by a relational MySQL database, while CiteSeerX stores the data in a MySQL relational database and in a distributed Global File System. A disadvantage of both approaches is that they rely on RDMSs which are not designed to arbitrarily scale unless expensive manually distributed solutions (e.g. horizontal/vertical partitioning) are adopted. We also considered using freely available RDF databases, but they seemed to be not scalable enough for our needs[§§§].

3.2. Workflow nodes

A workflow node is defined by: (1) the schemas of input and output data stores, and (2) its business logic (see Fig. 1 for an example), which processes input data stores and generates output data stores conforming to the given schemas. Note that this solution is closely related to the "pipes and filters" design pattern[13]. The schemas, in a way similar to static typing in programming languages, define input and output interfaces of the workflow node. This approach reduces somewhat flexibility but increases modularity and enables static workflow correctness control; i.e. checking, before the execution of the workflow, whether the data store consumed by a downstream workflow node is compatible with the data store produced by an upstream workflow node. This modularity results in loose coupling of workflow nodes, an important feature of every system; for comparison, in CiteSeerX and ArnetMiner this is achieved by applying a Service Oriented Architecture approach where each module is a separate web service.

---

††† http://avro.apache.org

‡‡‡ http://blog.golang.org/share-memory-by-communicating

§§§ http://www.w3.org/wiki/LargeTripleStores website contains list of sizes of such databases in real-life deployments

To specify the input and output interfaces, we use Avro schemas. This approach results in an interesting functionality akin to polymorphic behavior of arguments of methods in object-oriented languages: a workflow node can handle not only the data stores that have exactly the same schema as defined by its input, but also data stores with a schema that is a superset of the expected schema. This way the workflow node can define that it needs only a part of a possibly complex data store and only this part is visible to the node.

Types of workflow nodes. The system supports the definition of various kinds of workflow nodes; they can be generally divided into two categories: atomic and composite workflow nodes.

An atomic node is indivisible from the point of view of the IIF. It might be a map-reduce Hadoop job or a chain of such jobs that uses the underlying Hadoop system; it can also be a wrapper for some external system, e.g. a web service that processes the data using its own tools like a neo4j database.

A composite node consists of a combination of workflow nodes. Currently we define the following composite workflow nodes: subworkflow, conditional execution workflow node (one of the control paths is chosen to be executed based on whether a certain condition is satisfied), parallel execution workflow node (several control paths are executed in parallel).

Note that it is often the case that a data store produced by one workflow node does not match exactly the data store expected by a node that is supposed to consume it, e.g. a certain attribute in one data store is represented by two attributes in the other data store. Thus, the data store produced by one workflow node has to be somehow adapted before it can be consumed by the subsequent node. A workflow node that is responsible for such transformation is named "transformer" – note that its role is similar to the "message translator" pattern[13]. Transformers are usually written as Pig scripts which provide a succinct data transformation language similar to SQL. When executed on a cluster, the scripts are automatically translated into a series of MapReduce jobs.

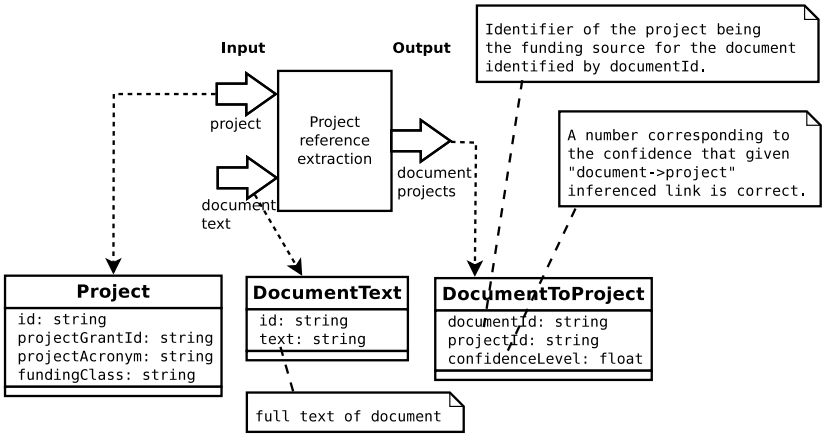

Fig. 1. A sample OpenAIREplus workflow node that ingests data stores corresponding to projects and full text of documents and produces a data store containing links between documents and projects. The node specifies its inputs and outputs with Avro schemas, which are represented here as UML class diagrams.

3.3. Definition and execution of a data processing workflow

After concrete workflow nodes appropriate for a given application domain are implemented, the workflow designer creates a workflow definition which connects outputs of certain workflow nodes with inputs of other workflow nodes, thus defining a data processing pipeline. Finally, the framework creates an archive containing the workflow definition file and implementations of workflow nodes (which are stored in *.jar files). Next, such archives are automatically deployed and executed on the cluster.

When choosing the workflow engine specific to our use case, a couple of alternatives have been considered: generic ones (Apache ODE, Sarasvati) and based on Hadoop (Apache Oozie, LinkedIn's Azkaban, Hamake). We decided to choose probably the most popular engine from the second group: Apache Oozie. We were convinced by

its balanced mix of properties: relative maturity of the project; queuing and coordination of the workflow done directly on the Hadoop cluster (through Oozie service); possibility to restart the workflow from a certain point; and variety of supported interfaces (command-line, REST, Java API, web interface for browsing executed workflows). Note that Apache Oozie is a generic workflow engine, without any notion of the data passed between workflow nodes. To implement our idea of a data processing workflow, we impose certain conventions on the way of defining our Oozie workflows: each workflow node (1) must clearly specify what kind of data it consumes/produces (2) has a place to store its intermediary computations results, (3) stores output data in well-defined data stores, and (4) receives input parameters in a well-defined manner. In order to hide such conventions from the developers, we are currently in the process of designing a data workflow description language on top of Oozie to write workflows according to such conventions; this approach results in a more concise, uniform, legible, and easy to maintain workflow descriptions.

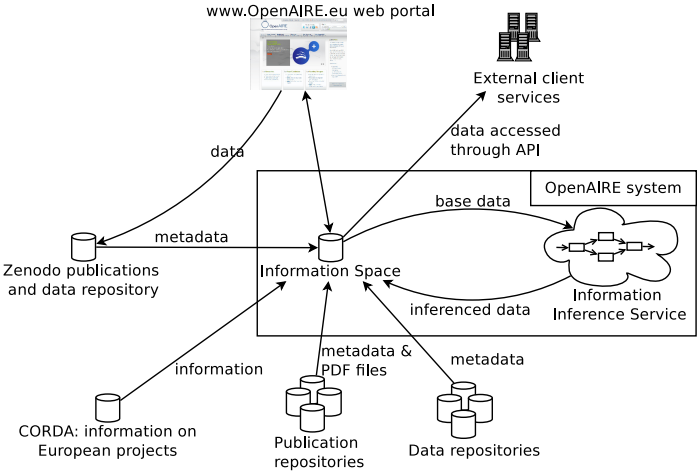

Fig. 2. Architecture of OpenAIREplus system. Arrows show the direction of data flow along with information about type of data being transferred.

4. The IIF in OpenAIREplus

The original motivation behind developing the IIF was implementing information inference service for scholarly communication infrastructure of the OpenAIREplus project. In this section we present the OpenAIREplus project itself and the role of the IIF in it.

4.1. Architecture of the OpenAIREplus system

A high-level architecture of the OpenAIREplus infrastructure is presented in Fig. 2. Its core components are:

- **Data aggregation and persistence**. The data aggregation framework is based on the D-NET Software Toolkit[14], which offers tools for defining and executing metadata processing workflows. In OpenAIREplus, the framework has been configured to collect XML-serialized versions of publications, datasets, and projects from data sources of various types, such as publication repositories, dataset repositories, CRIS systems, entity registries (e.g. OpenDOAR, re3data.org). Subsequently, such serializations are harmonized and transformed to become objects of a graph of entities (i.e. publications, datasets, funding schemes, projects, organisations, persons, and data sources). The graph is stored in the Information Space which can be regarded as the central database of the system.
- **Data access**. The data stored in the Information Space can be accessed using the OpenAIRE portal and standard machine communication protocols (e.g. OAI-PMH, CQL).
- **Information inference**. An instance of the IIF is responsible for enriching the data stored in the Information Space with new information using various types of data mining algorithms.

The IIF includes several workflow nodes and integrates them in one data inference workflow. The subset of data from the Information Space that is required by the workflow nodes is retrieved by the IIF importer module. Next, the IIF executes the workflow and produces an output which is subsequently merged with the original data in the Information Space, in order to enrich or adjust its content. The data stored and produced by the IIF forms a secondary, non-authoritative storage, i.e. at any moment, all data in the IIF can be recreated from the Information Space by re-running the IIF workflow. IIF is stateless in this sense.

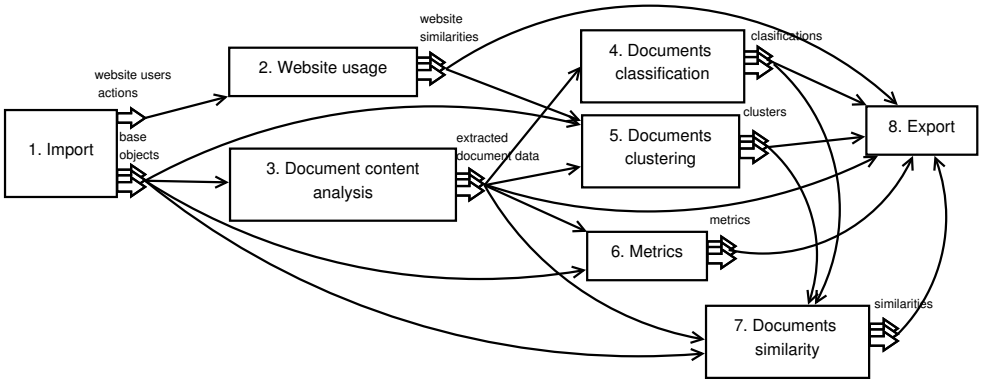

Fig. 3. General workflow of the IIF. Thick arrows correspond to outputs of workflow nodes, thin arrows correspond to connections between inputs and outputs.

4.2. Information Inference Framework in OpenAIREplus

The instance of the IIF deployed in OpenAIREplus defines specific workflow nodes and related data stores. In general, it processes plain text of papers retrieved from the Information Space and their related metadata and produces various inferred relationships and data. See Fig. 1 for a sample definition of an OpenAIREplus workflow node. The functionalities provided by the IIF instance to OpenAIREplus system correspond directly to the following top-level workflow nodes that are bound together by OpenAIREplus inference workflow (see Fig. 3):

1. Import – retrieves relevant data from the Information Space and saves it in data stores. The data stores contain information about persons, projects, documents, datasets, and logs from the OpenAIREplus web portal.
2. Website usage – computes similarities between documents and persons based on the activity of the OpenAIREplus web portal users, e.g. if a user views two documents in a short time, they might be related.
3. Document content analysis – extracts metadata from PDF documents and uses it to update the original information about documents. This node consists of a few smaller subworkflows that provide the following functionalities: extracting basic metadata from raw content of documents (PDF files); creating "cited by" relationships between documents based on the references in document's bibliography section; creating "funded by" relationships between document and projects retrieved from the Information Space; creating "references" relationships between a document and its data sets stored in selected external databases (e.g. DataCite, EuropePMC).
4. Documents classification – assigns predefined labels (e.g. based on ArXiv taxonomy) to documents based on their contents, metadata, and activity of OpenAIREplus web portal users.
5. Documents clustering – creates clusters of similar documents. The similarity is computed based on document content and its metadata as well as the activity of OpenAIREplus web portal users.
6. Metrics – computes basic metrics for authors and documents, such as number of documents produced by a given author and number of citations of a given document.
7. Documents similarity – computes which documents are similar and thus are possibly related or duplicated based on the data gathered so far.
8. Export – exports inferred data to the Information Space.

4.3. Current state of development and future plans

The IIF is still actively developed. However, the most important framework elements are already functional and we currently integrate the nodes provided by OpenAIREplus project partners into one workflow. These include CERMINE**** system for metadata extraction and MadIS†††† system for extraction of references from documents and text mining. In the next months, other nodes providing the functionality described above will be incrementally introduced to complete the system by the end of 2015. In the same time-frame, we are considering to introduce an additional "training" workflow. Its goal would be to automatically adjust machine learning models (responsible, e.g. for classification and clustering functionalities) to changes in Information Space data. Currently these models have to be provided by the developers.

Acknowledgments

This work is funded by the European commission (FP7-INFRA-2011-2, Grant Agreement no. 283595). This work would have not been possible without the contributions and experiences of the OpenAIREplus technical team members: M. Artini, C. Atzori, A. Bardi, S. La Bruzzo, M. Mikulicic (ISTI-CNR, Italy), L. Nielsen (CERN, Switzerland), B. Companjen (DANS-KNAW, Netherlands), T. Giannakopoulos, H. Dimitropoulos, K. Iatropoulou, A. Lempesis, O. Metaxas (University of Athens, Greece), M. Loesch (Bielefeld University, Germany).

References

1. Manghi P, Bolikowski L, Manola N, Schirrwagen J, Smith T. OpenAIREplus: the European Scholarly Communication Data Infrastructure. *D-Lib Magazine* 2012; 18.
2. Manghi P, Manola N, Horstmann W, Peters D. An Infrastructure for Managing EC Funded Research Output – The OpenAIRE Project. *International Journal on Grey Literature* 2010; 6: 31-40.
3. Castelli D, Manghi P, Thanos C. A vision towards scientific communication infrastructures. *International Journal on Digital Libraries* 2013; 13: 55-169.
4. Mierswa I, Wurst M, Klinkenberg R, Scholz M, Euler T. Yale. Rapid prototyping for complex data mining tasks. In: Ungar L, Craven M, Gunopulos D, Eliassi-Rad T, editors. *KDD '06: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, New York: ACM; 2006, p. 935-940.
5. Prekopcsák Z, Makrai G, Henk T, Gáspár-Papanek C. Radoop. Analyzing big data with rapidminer and hadoop. In: Fisher S, Mierswa I, editors. *Proceedings of the 2nd RapidMiner Community Meeting and Conference*, Aachen: Shaker Verlag; 2011, p. 31-42.
6. Tang J, Zhang J, Zhang D, Yao L, Zhu C, Li JZ. ArnetMiner: An Expertise Oriented Search System for Web Community. In: Golbeck J, Mika P, editors. *Proceedings of the Semantic Web Challenge* 2007, CEUR-WS.org; 2007.
7. Tang J, Zhang J, Yao L, Li J, Zhang L, Su Z. ArnetMiner. extraction and mining of academic social networks. In: *Proceedings of the 14th ACM SIGKDD international conference on knowledge discovery and data mining. KDD '08*, New York: ACM; 2007, p. 990-998.
8. Teregowda PB, Councill IG, Fernández RJP, Khabsa M, Zheng S, Giles CL. SeerSuite: developing a scalable and reliable application framework for building digital libraries by crawling the web. In: *Proceedings of the 2010 USENIX conference on Web application development. WebApps'10*, Berkeley: USENIX Association; 2010, p. 14-14.
9. Li H, Councill I, Lee WC, Giles CL. CiteSeerx: an architecture and web service design for an academic document search engine. In: *Proceedings of the 15th international conference on World Wide Web*, New York: ACM; 2006, p. 883-884.
10. Ferrucci D, Lally A. UIMA: an architectural approach to unstructured information processing in the corporate research environment. *Nat Lang Eng* 2004; 10: 327-348
11. Cunningham H, Maynard D, Bontcheva K, Tablan V, Aswani N, Roberts I, Gorrell G, Funk A, Roberts A, Damljanovic D, Heitz T, Greenwood MA, Saggion H, Petrak J, Li Y, Peters W. *Text Processing with GATE (Version 6)*. California: Gateway Press; 2011.
12. Ashish N, Mehrotra S, Pirzadeh P. XAR: An Integrated Framework for Information Extraction. In: *Computer Science and Information Engineering, 2009 WRI World Congress on*. Los Angeles: IEEE; 2009, p. 462–466.
13. Hohpe G, Woolf B. *Enterprise Integration Patterns: designing, building, and deploying messaging solutions*. Addison-Wesley 2004.
14. Manghi P, Artini M, Atzori C, Bardi A, Mannocci A, La Bruzzo S, Candela L, Castelli D, Pagano P. *The D-NET software toolkit: A framework for the realization, maintenance, and operation of aggregative infrastructures*. Journal Program, Emerald Insight 2014;48(4):322-354.

**** https://github.com/CeON/CERMINE
†††† http://code.google.com/p/madis