Rybiński, Henryk; Koperwas, Jakub; Skonieczny, Łukasz; Kozłowski, Marek; Struk, Wacław

Article

Omega-Psir: From the repository to the research knowledge base experience of Warsaw University of Technology

Foundations of Management

Provided in Cooperation with:
Faculty of Management, Warsaw University of Technology

Suggested Citation: Rybiński, Henryk; Koperwas, Jakub; Skonieczny, Łukasz; Kozłowski, Marek; Struk, Wacław (2013) : Omega-Psir: From the repository to the research knowledge base experience of Warsaw University of Technology, Foundations of Management, ISSN 2300-5661, De Gruyter, Warsaw, Vol. 5, Iss. 1, pp. 53-68, https://doi.org/10.2478/fman-2014-0006

This Version is available at:
https://hdl.handle.net/10419/184548

Foundations of Management, Vol. 5, No. 1 (2013), ISSN 2080-7279
DOI: 10.2478/fman-2014-0006

OMEGA-PSIR: FROM THE REPOSITORY TO THE RESEARCH KNOWLEDGE BASE EXPERIENCE OF WARSAW UNIVERSITY OF TECHNOLOGY

Henryk RYBIŃSKI, Jakub KOPERWAS, Łukasz SKONIECZNY, Marek KOZŁOWSKI, Wacław STRUK

Institute of Computer Science, Warsaw University of Technology, Warsaw, Poland
e-mail: {h.rybinski, j.koperwas, l.skonieczny, m.kozlowski, w.struk}@ii.pw.edu.pl

Abstract: This paper is devoted to a solution for building a research knowledge base for the university. An experience in developing a specialised software for building such a knowledge base is presented. We present the functionality of the system, as well as, sketch applied AI technologies aiming at providing features attractive for the system beneficiaries. It is shown that although a classical repository is the main part of the system, the essential value of the solution is in providing analytical tools for the 'research management'. First lessons learned from deploying the software at Warsaw University of Technology are also discussed. The platform has been developed under the SYNAT project, aimed at building nation-wide scientific information infrastructure.

Key words: digital library, knowledge base, scientific resources, repository.

1 Introduction

In 2010, a dedicated project, SYNAT, has been launched in order to address deficiencies of scientific information infrastructure in Poland. The main SYNAT construction is based on three levels of scientific repositories, its ultimate goal is to ensure the dissemination of the Polish scientific achievements and to improve integration and communication of the scientific community, while leveraging existing infrastructure assets and distributed resources. The three levels of scientific repositories are shown in Figure 1.

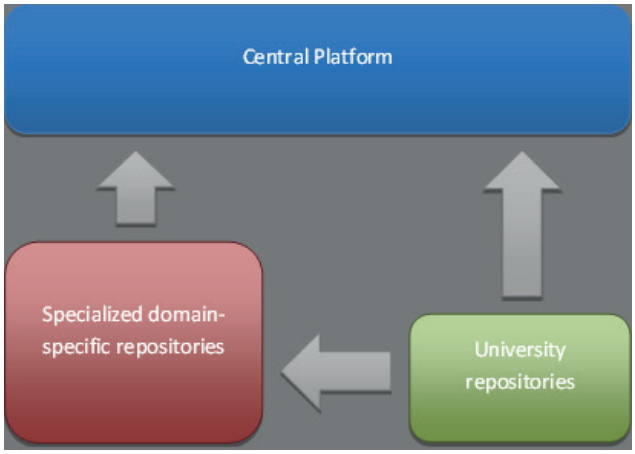

Figure 1. General SYNAT architecture

- the central level (the SYNAT platform and the INFONA portal),
- the domain level (specialised domain-specific repositories),
- the university level (repositories held by the universities).

More information about the research concerning SYNAT can be found in [2, 3]. In this document, we focus on the university level and postulate that it should be more of a knowledge base rather than just a repository. The purposes of the university 'Research Knowledge Base' is to provide:

- an entry point of the entire scientific information (together with full texts) concerning research performed at the university, with accordance to the principle that the information about publication, research project, etc., is entered to the system in the place where the research was performed. This ensures that this information is correct and up-to-date,
- an access point to this information with limiting factors depending on the access type (logged user, access from university, access from the world), and the publishers' copyright policies,
- an easy way to store cleaned, unambiguous and interconnected information of various types, e.g. researchers, institutions, conferences, journals, book series, projects, publications, diplomas etc.,
- administrative tools for reporting and evaluating scientific achievements of the university as a whole, the university institutions, as well as, individual researchers, according to the requirements of the university executive level, State authorities (Ministry of High Education), non-governmental bodies, etc.,
- interoperability, i.e. both side communication channels with various university services and systems (e.g. homepages, system supporting diploma processes, report generators, employee database) and various external services like Google Scholar, publishers' websites, etc.

We have been developing such a 'knowledge base' for almost 2 years now. The project originated from analysing the needs of Faculty of Electronics and Information Technology, and was first implemented to suit one faculty requirements, but soon it became clear that the system should become an university-wide one. Hence, it is now under deployment at the whole university.

The paper consists of three sections. In Section 2 we present general features of the software. Then in Section 3 we present special developments for improving data acquisition processes. In Section 4 we briefly present novel technologies incorporated within the software, making more valuable information retrieval parameters, as well as, analytical functionalities of the system.

2 Ω-Ψ^R Software

2.1 General assumptions

The last decade has shown an increased interest of many universities around the world in the systems concerning research data management and access to publicly funded research information. Also in Poland, at universities various approaches can be observed – some has been very enthusiastic in building infrastructure for research repositories storage, some other rather reluctant in supporting academic staff in meeting more demanding requirements for research practice, quite often due to lack of idea on how to provide suitable motivation on one side, and assistance on the other, leading to a successful university research knowledge database. It turns out that important success factors are harmonising solutions concerning both organisational issues, as well as functional features of the software.

The starting requirements for the university knowledge base were rather typical, focused on the repository functions. The main aim of the repository was to build institutional publication repository services, based on the open access idea to the most possible extent. However, since the very beginning it was clear that the system cannot be limited to the repository functions only. The university's research knowledge base should cover a vast and heterogeneous repertory of data concerning various aspects of the research activities, and the knowledge base should be a central entry point for information about researchers and their activity records, including inter alia the projects run at the university, along with various project documents and data, but also presentations, patents, etc., as well as various level diplomas and theses, starting from B.Sc. through M.Sc. to Ph.D. theses. All these data types should be strongly interconnected, building a semantically rich database. With such variety of the content in the university knowledge base, the software should be flexible enough, so that demands for new object types and new relationships would be fairly easy to implement. The flexibility should be reflected in providing administrative tools that enable defining new data structures, and then for the new objects make it easy to define new data entry worksheets and new search screens. As a result of a thorough evaluation of the needs of various levels of the academic community, the fundamental features of the implemented software are as follows:

- the system provides an easy way for:
  - storing any typed metadata along with digital content,
  - an easy way for defining custom types of stored metadata,
  - defining relationships between records of different types,
- with the defined data structures (rich in relationships between various objects), there are simple yet powerful means for the maintenance and control of the semantic data network stored in the knowledge database,
- the system is able to preserve 'historical values' of linked objects in the course of changes,
- the system provides multiversioning of the data,
- an efficient full-text search capabilities in both metadata and digital objects is available,
- the system provides means for automatic generation of highly ergonomic and customisable GUI,
- the system provides means for an easy integration with external systems and exposure for external search engines,
- the system provides means for extensive access control mechanisms,

With all the features above, the last but not least, is the multilinguality of the system.

Bearing in mind dynamics in changing the users' needs, all the tools of this kind have been implemented within the system in such a way that in most cases the system development does not require programmers' intervention. With the use of the implemented software, we have built the University Research Knowledge Base, which has following functionalities:

- storing of the university organisational structure,
- storing knowledge resources metadata and their correlation, among others:
  - publications (books, articles, etc.),
  - patents,
  - diplomas (B.Sc., M.Sc., Ph.D.),

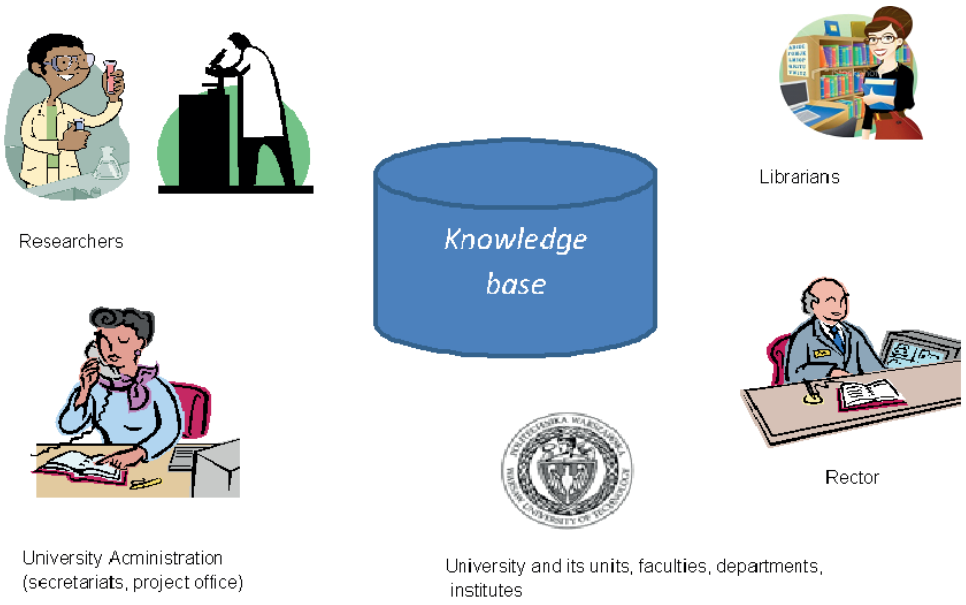

Figure 2. System beneficiaries

- research projects, along with the project documents, multimedia presentations, as well as research data, benchmarks, etc.,
- other scientific documents (reports, reviews, etc.),
- information about authors and their affiliations (CVs, other external and/or internal activities),

- storing employees data together with their achievements,
- evaluation of employees and organisational subunits,
- reporting for the internal purposes and for the purpose of the authorities,
- import/export in various formats.

Clearly, with the flexibility feature of the system it is fairly easy to define other objects within the knowledge base.

2.2 Users and beneficiaries of the system

The groups of the foreseen system users and beneficiaries are very heterogeneous. The first classification splits the users into the external and internal ones. As for the internal users, we can distinguish the following groups (Figure 2):

- researchers,
- students (graduates, undergraduates),
- university administration,
- scientific bodies (faculty councils, senate, promotion commissions, etc),
- university leaders, responsible for the research strategies.

For the external users, the role of the system is also multifold, though a bit different. The system integrates various functions, but the main function that should be emphasised are:

- provide a complete and up-to-date information about the research areas of the university researchers, and their strength to the potential external partners for building scientific cooperation links,
- provide means to the governmental authorities concerning the research potential of the university, and the currents achievements, inter alia for the evaluation and assessment reasons,
- provide complete and up-to-date information about the research areas of the university researchers, and their strength to the international evaluating bodies.

2.3 Main functionalities

In this Section we briefly present the functionality of the system.

2.3.1 Repository functions

With any defined object in the knowledge database it is possible to predefine various ‘digital attributes’. The digital attributes are devoted to store digital objects, which then are accessible by a unique ‘object identifier’. The text objects are subject to indexing, so that the index for full-text retrieval is built automatically with the new objects added to the database. Also the updates of text documents are automatically reflected in the indexes. The repository functions are implemented by means of the JCR [1] programming tools.

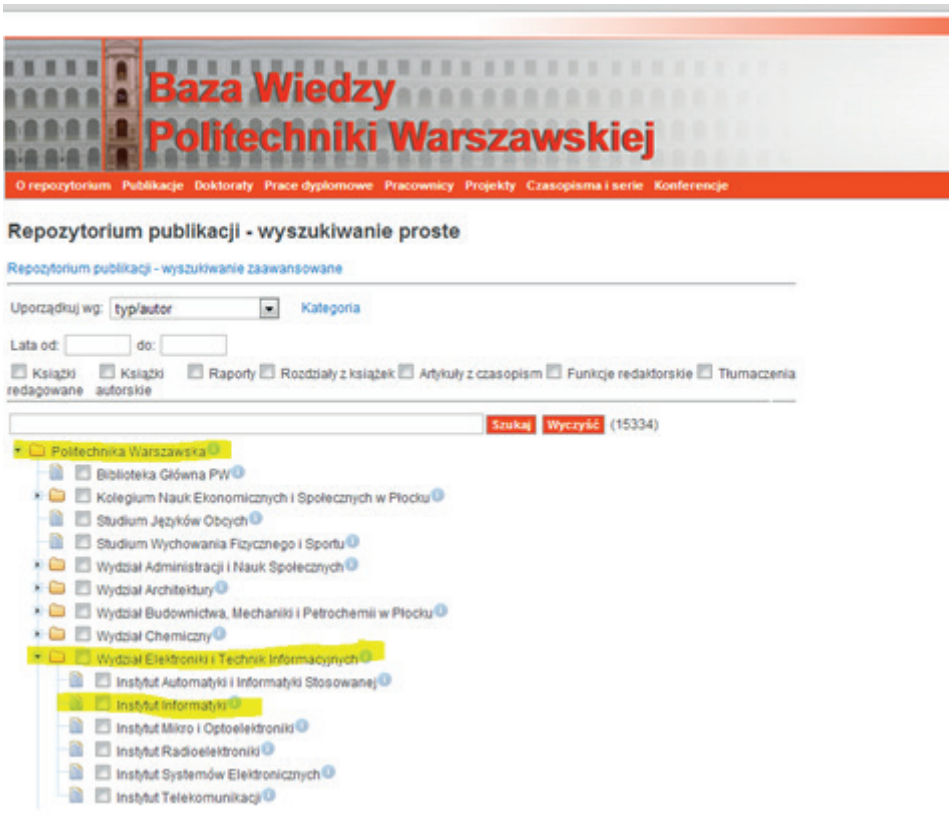

Figure 3. A search screen for a central retrieval point

2.3.2 Information retrieval

As Jackrabbit implementation of JCR uses Lucene [6], the powerful full-text retrieval capabilities of Lucene become a standard way to access the text objects and accompanying digital objects. The digital objects are 'values' of the digital attributes, so that if a document being value of an attribute of a given record fits to a query, the record is classified as relevant to the query. Among the objects' attributes, there are usually the attributes entered by the data entry clerk (or imported from external sources), or the ones, which are 'automatically' generated as a result of 'text mining' (automatic classifications/categorisations of the objects). Also the 'automatically' filled attributes are used for building the search index, and then used for information retrieval. For building the information retrieval screens, special tools have been implemented within the system. An example of the search screen is presented on Figure 3.

2.3.3 Multilinguality

Multilinguality is implemented in two layers:

- the GUI layer,
- the data layer,

For the GUI layer, in order to build a screen the system uses a dictionary for a given language, replacing the 'coded' messages/texts into the proper language equivalents. For the data layer, one has to foresee some attributes that are language-related, so that for a specific interface language appropriate values can be displayed.

2.3.4 Reporting

The system has a variety of tools to 'define' specific reports. Two ways of reporting are foreseen:

- building a simple report from a search result,
- building an aggregate report from a search result.

Both ways of reporting can be 'defined' in such a way that no reprogramming of the software is needed. It is just a repository of reports stored in the system base, and for a given GUI page we can invoke a predefined report.

2.3.5 Interoperability

The interoperability of the system is reflected by means of the following functions:

- data import,
- data export,

- harvesting tools for the external servers,
- linkages to the external resources.

Data import can be done in many ways. There are database administrator possibilities to import bulk of data for a startup loading of given objects, there are also means for the data entry clerk to import a BibteX file for loading some bibliographic records. What is very important, any kind of the data import is performed in such a way that the imported data cannot spoil the 'semantic network' stored in the knowledge base. To this end the following functionalities have been developed:

- the import controlled by the system for duplicates,
- the imported data are automatically 'linked' to appropriate objects, for example the new publication is linked to the author(s), relevant journal, conference, etc.

The data import is a part of the acquisition functionality. For the importance of this functionality for the system, it is described in more details in Section 3.

Data export is also provided in various ways. There are possibilities just to dump a part of the database in the form of the XML (in a proprietary format). Additionally there are tools for end-users to export from the system data in the form of BibTeX or Excel.

The harvesting functionality has been implemented for exposing the knowledge base data to the harvesting processes of the external systems. The external systems can harvest data from the knowledge base by means of OAI PMH [11] (in the Dublin Core format) or by means of the system proprietary services.

Another important form of interoperability is the connectivity to the resources of other systems. In particular, while presenting the bibliographic results, the system provides a link to the Google Scholar record. Additionally, whenever available, the DOI value provides an easy access to the publisher version of the publication. In the case of missing DOI, the data may contain URL links to the external digital contents.

2.3.6 System flexibility and configuration

As mentioned above, with a variety of the content in the university research knowledge base, and difficulties in defining all the software needs at the design phase, the software should be flexible enough, so that demands for new functionalities would be fairly easy to implement. The main idea behind this requirement was to provide easiness in defining custom installations and make possible expanding to the system in course of its life, without the need of (re)programming the software. The flexibility should be reflected in providing administrative tools that enable defining new data structures, and then for the new objects make it easy to define new data entry worksheets and new search screens. The proposed solutions for an upgraded flexibility of the system moved further. Actually, the system additionally provides scripting means for:

- validation tools supporting data entry process quality,
- means for visualising search results, custom views, defining sorting rules, etc.,
- means for defining analytical views,
- access rights rules, defining access to various objects, as well as synthetic information that can be obtained from the system.

All the tools of this kind have been implemented within the system in such a way that in most cases the system development does not require programmers' intervention. More details concerning the administrator tools are presented in [13].

2.3.7 Access control/protection

Another issue tightly coupled with the organisation structure is access control and users privileges. The system provides means for controlling access to the functions, as well as to the digital resources. There are some levels of protection, depending on the user role and its affiliation. The privileges control has been implemented by introducing to the metadata the attributes storing the 'record owner' and its 'affiliation'. Additionally there are special means implemented for controlling access to the digital resources. For specifying the privileges various metadata attributes can be used. The metadata and the user defined accessibility parameters are then combined within business rules that compute access rights. The rules can be easily modified, as they are kept in a separate configuration file. The administrative means for the access control are presented in more details in [13]. Below we provide an example, which illustrates how the rules can be built:

Example

A user that has 'dataentry' or 'superdataentry' role assigned can modify and delete his/her own records. Additionally, if the user enters a publication co-authored with an author from another institute, the other institute will also have access to the record modification (except deletion).

```
rule DataEntryAndSuperDataEntryCanAd-
dCanDeleteAndModfiyItsOwnRecords
when
  check: PermissionCheck(granted
==false,action ==”edit” ||
action==”delete”)
  role: Role(name == ‘dataentry’ ||
name==’superdataentry’)
  identity:Identity(name:principal.
name, affiliation:affiliation)
  entity:Entity(owner==name,affilia-
tionowner==affiliation)
then
check.grant();
end
rule DataEntryCanModifyPublications-
OfOwnedAuthors
when
  check: PermissionCheck(granted
==false,action ==”edit”)
  role: Role(name == ‘dataentry’ ||
name==’superdataentry’)
  identity:Identity()
  art:Article()
  a:Author(affiliation
!=null,affiliation.
accronymPL==identity.affiliation) from
art.author
then
check.grant();
end
```

2.3.8 Knowledge base functions

As a result of integration of various types of information under one system, quite new and powerful functionalities can be implemented. The most spectacular are the possibilities to perform:

- looking for expert,
- aggregating and visualising knowledge area for researchers, and/or for the university units,
- providing statistics about the research activities.

Below we present the first two functions

Looking for expert

One of the interesting features implemented within the system is the functionality of looking for experts, potential candidates for the teams and projects, within the university, as well as for the external partners who are looking for the cooperation.

The applied search is not based on what the staff declares or writes to the CV. Instead, it is based on what the knowledge base contains concerning various activities of the researcher. It means that the implementation of the functionality makes sense if the following conditions are satisfied:

- the knowledge base represents a kind of ‘semantic network’ by means of variety of interconnected objects, such as publications, patents, projects directed and/or participated, expert’s involvement in the conference program committees, etc.1 ,
- the knowledge base is as much complete as possible.

The expert search is based on the following components:

- major fields of the expertise suggestion; there is a module which extracts the main areas of expertise from the publications data (keywords provided by the authors), tags extracted automatically from the papers contents (by semantic processing of the objects, see the next subsection), the research maps2, which are built based on journals subject area, assigned to the papers by the Ontology for Scientific Journals (see [12], and the next subsection), as well as the researcher’s affiliation description, which in turn can also contain area tags, aggregated from the tags of publications, assigned to the ‘affiliation’ unit,
- search engine: this relies on the Ω-Ψ^R search engine, build on top of the Apache Lucene library; it provides full-text search and a rank on the basis of the well-known TF/IDF measure,
- a ranking module, which is definable by the system administrator with a special scripting language; such a module can implement a very specific rank algorithm, which can provide specific weights to the particular evaluation elements (e.g. impact factor of the journals, number of citations, special ranks for managing projects, etc.); one can have many ranking algorithms defined, so that the end-user can specify the ranking, depending on his/her specific needs; for example, a student looking for his Ph.D. supervisor can provide a search for experts in a given domain but then can sort them by number of supervised theses,
- result presentation: the results are presented in the form of table with the authors portfolios, containing all the details about the authors activities; the results are enriched with the ranking score bar.

---

[1] This information is planned for being applied when the acquisition process based on web mining is extended on searching for the involvement into conferences PC.

[2] For the internal needs, the module presents the tags in the form of a vector, and it visualizes it for the end-users as a word cloud. The word cloud can be “calculated” for the authors, and for the affiliations by aggregating cloud vectors assigned to the papers, supervised theses, run projects etc. This helps the user to pick the most probable area of expertise rather than test the casual phrases

The ranking algorithm can be roughly presented as follows:

- all the knowledge base resources are searched with a specified search phrase (formulated the same way as for searching publications, theses, etc.),
- for all the result items returned, the persons being the publication authors, theses supervisors, project leaders, etc. are extracted and a table for authors is built with all the 'achievements' for each person (publications, theses, projects, affiliations, etc.),
- for each 'achievement' record for each person in the table the score is calculated according to the selected algorithm,
- for each author the calculated achievements scores are added and the final 'authors score' is provided,
- the table is sorted by the final scores and presented as output.

A simple example for calculating the one 'achievement record' score is given below[3]:

Multiply the following parameters:

- the Lucene relevance score,
- the author role weight ('article author', 'book author', 'book editor', 'phd author', 'phd supervisor', 'bachelor author', 'bachelor supervisor', 'master author', 'master supervisor', 'project member', 'project leader', 'author profile'),
- impact Factor (IF) of the journal (for the journal papers), IF for the book series (for the chapters or books), a defined value for a book chapter or book if the IF are not known.

Aggregating the research activities

The main advantage of integrating various aspects of research activities of the researchers is a possibility to aggregate all the activities, so that the presentation of a department, faculty or an institute can be provided with the aggregated values summarising various activities. The aggregated 'activity vector' can be then visualise in the form of a table or as a 'word cloud' of research areas. Examples of visualising research areas are presented in Figure 4.

2.4 Functionally defined users

For running the knowledge base, a number of roles have to be assigned to the users. In this respect the system is also flexible and it is possible to make it in various ways, depending on the organisational conditions and human resources.

In general, in addition to the actual beneficiary end-users one can distinguish the following 'functional groups' of users:

- data entry staff,
- administrators of the database(s),
- system administrators,
- exploitation and maintenance staff.

Data entry: At WUT the data entry functions are assigned to the specially trained staff, usually to faculties. The decisions in this respect depend on the faculties and deans. In some cases the roles are dedicated to the institutes, in other, to the faculty librarians. It is however also expected that in smaller units, the data entry role can be assigned to the researchers directly. For such cases, special control of data quality will be given to the Main Library staff. The data entry role is limited to the role scope: as a rule, the user can correct and modify its own record, only the user with special privileges can update records of 'subordinated users'.

Administrators of the database(s) (DBA): The database administrator role provides the functionality which makes possible to change the values of 'central databases', like the journals base or conferences. At WUT, only an assigned librarian will have rights to (re)define the scores assigned to the journals (based on the Ministry scoring system), control the conference database (flagging on and off the conferences with the Web of Science flag). The other functions of the DBA cover, i.e.:

- assigning/changing privileges to the users,
- controlling the statistics of the database contents,
- generating special reports for decision makers,
- controlling data quality,
- verifying data entry (cataloguing) rules,
- maintaining the contents of auxiliary dictionaries (languages, countries, affiliations, and the university structure).

In Figure 4, a screen for searching for an expert and the result screen are provided.

System administrators (SA): The system administrators have at their disposal a number of functions that make possible changes to the software without programming. The system administrator can change the system configuration (switching on or off some functionalities), add new reports to GUI, change the rules for the users privileges. The other changes of SA cover:

[3] This algorithm causes that publications where the keyword occurred frequently (for example in full text, extracted paper keywords, journal name, journal keyword) are scored higher, moreover the journal impact factor increases the ranking.

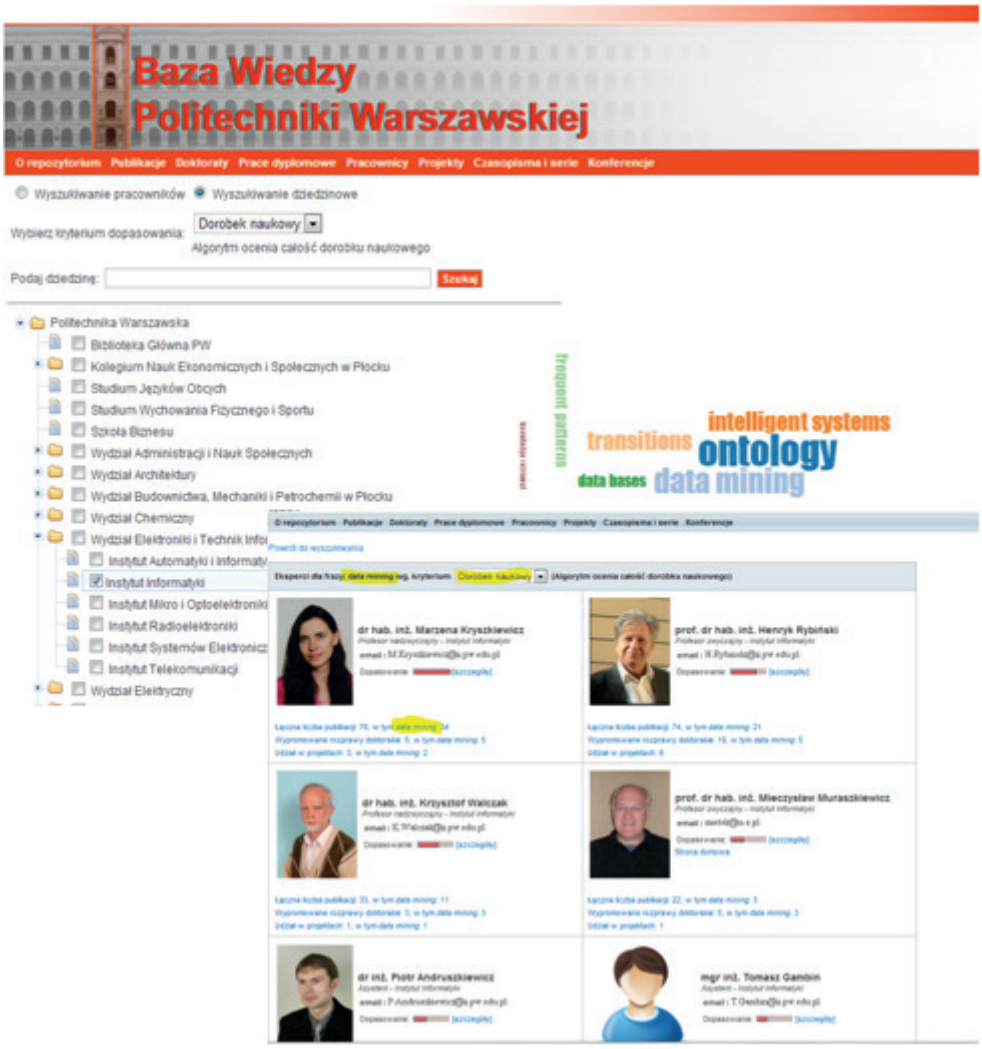

Figure 4. A search for experts in a domain and the result page

- defining new data objects and modifying structures of the existing ones,
- defining new screens (data entry, search, results) and modifying the old ones,
- defining new sorting rules, reports, etc. and modifying the old ones,
- defining/modifying validation procedures,
- additionally the database administrators can add new or change the existing scoring algorithms for the criteria of the research quality assessing.

Exploitation manager (EM): The EM is responsible for the correct work of the system environment and smooth work of the system. In particular, EM is responsible for:

- checking the CPU workload of the processor,
- usage of the disk space,
- backing up the information resources and the system,
- planning reorganisation/maintenance breaks and recovering the system after breaks,
- traffic control and statistics of the usage of the system,
- controlling the network parameters.

2.5 Architecture

The architecture of the system is schematically presented on the Figure 5.

One can distinguish here the following components:

- Application – the framework applied, Seam (http://seamframework.org/), makes easier building ergonomic GUI, and simplifies an overall integration of the system;
- Data – the data system is based on the XML formats. The data structure is verified by the XML Schema Definition (XSD). Such XML data are then converted to the object model by means of the JAXB technology (http://jaxb.java.net/). The data persistency is gained by means of the repository system Jackrabbit (http://jackrabbit.apache.org/), compliant with the standard of Java Content Repository (JSR 170 and JSR 283). The JCR standard provides *inter alia* the following functionality:

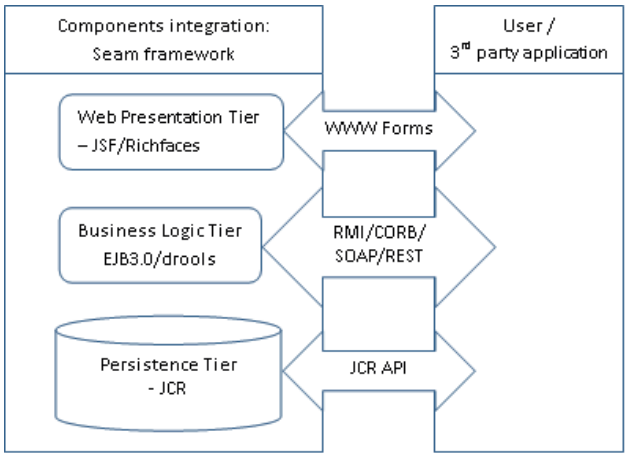

Figure 5. System architecture

- indexing,
- versioning,
- access control
- integration on the data level,
- control of the changes of the contents,
- a fairly simple switch to another technology provider.

For technical reasons, the system also uses a typical relational database, mainly for storing addresses of remote repositories as well as keeping data necessary for the users' authorisations.

- Presentation layer – this layer is supported by the technology provided by Java Server Faces, with the use of the library RichFaces (http://www.jboss.org/richfaces), which is based on the AJAX approach. The basic user interface is automatically generated, based on the XSD definitions, then it can be freely adopted/tuned to specific needs of the particular functionalities. More details are in [13].
- Business logic layer – it is based on the components of EJB 3.0 (see http://jcp.org/aboutJava/community process/final/jsr220/), so that transactionality of the system is guaranteed. Additionally, it simplifies exposing the system to the other (external) systems.
- Run environment – the system is run under the control of a virtual Java machine, ver. 6, and the application server JBoss[4].

The system Ω-Ψ^R has been implemented in such a way that it can applied as a central system, or in a distributed environment. As at a given point it has been decided that at WUT we apply a centralised version. This solution is in general much cheaper and easier to maintain.

[4] If needed, it can be replaced by another product compliant with the Java Enterprise Edition standard.

Figure 6. Data acquisition process

3 Data acquisition

This section presents a solution for automatic acquisition of bibliographic entries from the web. This process consists of three steps, depicted in Figure 6:

- searching for publications,
- extracting bibliographic metadata and finally,
- merging entries into university knowledge base.

The first step is realised by the Ψ^R module. The module can be seen as a focused web crawler. It delegates user-given queries to various search engines, executes them in a periodical manner and consecutively refines them to improve precision and recall of results. We use it to search the Internet for the publications of the WUT authors. Besides publications acquisition, the Ψ^R module is going to be used to retrieve up-to-date information about conferences and journals.

The second step is performed by the Zotero software [14]. Zotero is a free and open-source Firefox extension, which helps managing bibliographic data and related research materials. Notable features include automatic detection of bibliographic entries on websites, retrieving these entries and converting between many formats e.g. BibTex, MODS, RefWorks. Zotero supports bibliography detection in a variety of sources including publishers' databases, like Springer, Scopus or IEEE, as well as, publication search engines like Google Scholar or CrossRef.

It is worth mentioning that the system Ω-Ψ^R itself is supported by Zotero as well. In particular, for the purpose of data acquisition we implemented the Zotero-based web server, named bib2py. We use it to convert websites containing information about WUT publications into BibTex format. As Zotero was developed as a browser extension, it was not straightforward to build the web-server applica-

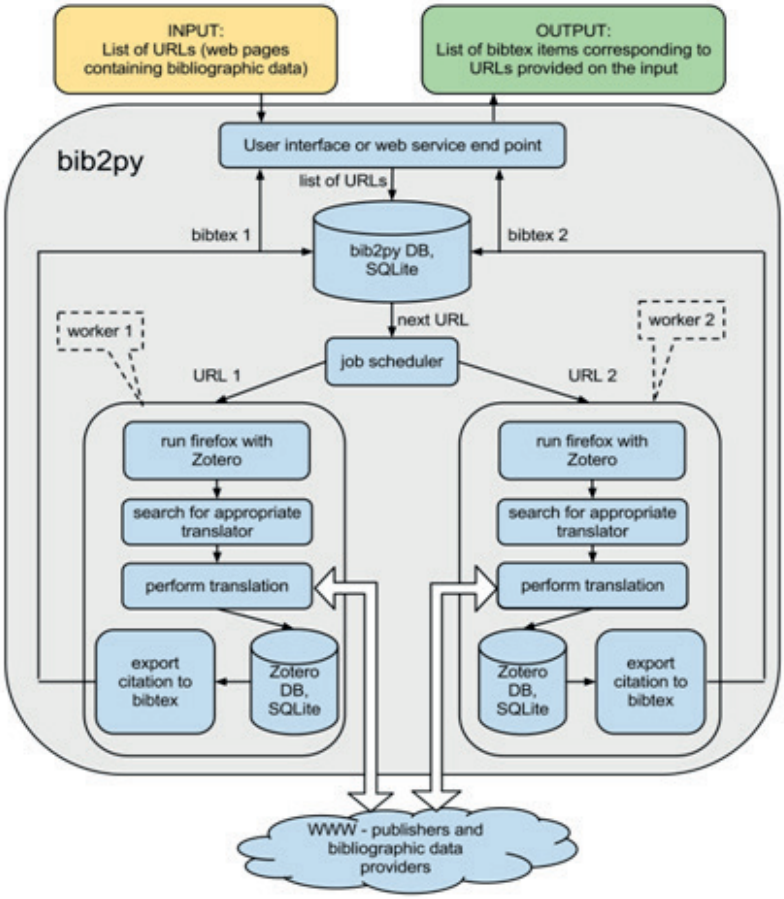

Figure 7. Zotero as a server – bib2py architecture

tion on the top of it. Several changes in the Zotero source code were made in order to automate the process of bibliographic data extraction and to eliminate the necessity of the user interaction. Moreover, in our web-server we utilised a plugin to Zotero, named 'Scaffold', which is a Firefox extension designed for developing and testing the Zotero translators.

The bib2py module provides the functionality that enables user to manage the process of bibliography extraction from the predefined collection of research resources. The service takes as an input either a single URL or the list of links that point to the websites containing bibliographic data, e.g.: http://link.springer.com/chapter/10.1007/978-3-642-32826-8_40, or http://dx.doi.org/10.1016/0306-4379%288 4%2990025-5.

For each URL, it tries to extract the bibliographic metadata existing on the webpage using an appropriate Zotero translator. When the bibliographic data are successfully extracted from the website, they are exported into the specified citation format (e.g. bibtex). The details of the bib2py architecture are presented on Figure 7. Note that in order to speed up the bibliographic data retrieval we can process different resources in parallel.

The last step, i.e. importing BibTex into the repository, is performed by the Ω-Ψ^R software. It converts BibTex into a native XML format, which represents publications in the form of a tree-like structure (see Figure 8). The tree nodes represent bibliographic elements, which might be shared between many publications, e.g. authors, books, journals, series.

Each tree node might have its own properties (e.g. title, name, surname). As mentioned earlier, the repository data structure (based on the JCR data model) is composed of objects (or nodes). Therefore, while importing an XML tree, the tree nodes are stored in the repository independently (like objects), so that they can be reused in linking to other publications, as well as, they can be embedded locally in the 'document tree', so that to allow keeping 'historical data'. The splitting of 'global data' from the local ones makes possible to modify data on either site, e.g. authors affiliation change or surname change can be performed at a global level of the PERSON object, whereas the local author subtree at a given publication tree stores the historic value of the affiliation). During the import, every element of the tree has to be matched against the contents in the repository, then merged and integrated

Koperwas Jakub Janusz, Skonieczny Łukasz, Rybiński Henryk, Struk Wacław: Development of a University Knowledge Base, w: Intelligent Tools for Building a Scientific Information Platform: Advanced Architectures and Solutions / Bembenik Robert *[i in.]* (*red.*), Studies in Computational Intelligence, vol. 467, 2013, ISBN 978-3-642-35646-9, ss. 97-110, DOI:10.1007/978-3-642-35647-6_8

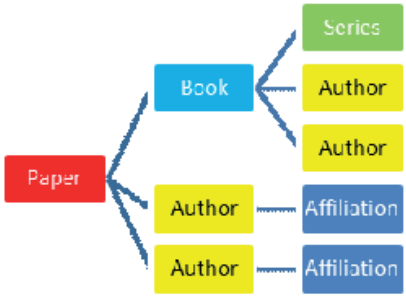

Figure 8. Bibliographic entry and its tree-like representation

with the existing 'objects'. We have implemented a number of general methods for the tree matching, as well as specific matching methods for identifying publications and authors.

It is worth noting that the presented data acquisition functionality does not cover all the possible functions of the module Ψ^R. This functionality addresses two main needs:

- it can be used to gather missing publications from web and load a bulk to the system,
- it can be used as a tool for the staff to perform a very easy data entry for their new publications; actually, with the use of ZOTERO the data input received from the publisher side is of a very high quality, and the obtained BibTeX file can be loaded automatically to the system. As the import of data is controlled for duplicates, there is no risk to spoil the repository.

4 Semantic processing server

Semantic processing in the WUT Knowledge Base is thought as a process aiming at enriching the objects by adding semantically meaningful descriptions and/or modifications. The processing is performed on the repository documents (publications, theses, patents, etc), and it consists in:

- scientific domain classification – adding subject tags that classify the objects to given scientific domains,
- extraction of semantically meaningful descriptors from the text and assigning them to the 'semantic description' field,
- identifying synonyms, relevant acronyms and other close meaning terms based on the semantically meaningful descriptors,
- looking for translations of the extracted descriptors and acronyms (from the English texts to Polish, from the Polish texts to English).

The target goal of those activities is manifold, the most important reasons are: the processing (1) is mainly used for building maps of research areas for individual researchers, and then, with the use of their research characteristics, for propagating the researchers interest to the affiliation related university units descriptions, and building the research maps for these units (from laboratories to faculties); the processing (2) and (3) are also used for building interest vectors of the researchers (used for building the word clouds), but mainly they are used for improving the search parameters, such as precision and recall; The processing (4) is used to improve multilingual information retrieval.

In the system Ω-Ψ^R the semantic processing is using two special semantic resources:

- ontology for Scientific Journal [12] (in the sequel OSJ), mainly used for performing task (1) above,
- the Wikipedia resources, mainly used for the tasks (2–4).

Below we describe the usage of the resources in more details.

4.1 Publication classifier

Scientific domain classification is the task consisting of providing a publication with one or more relevant tags, assigning the publication to one or more scientific classes. In Ω-Ψ^R we have decided to use the OSJ ontology as a classification schema. OSJ is a three level hierarchy, ended with the leaves on the last (fourth) level, which are simply scientific journal titles, so the path in OSJ from the root to a leaf (i.e. a journal title) assigns domain tags to the papers from the journal. The OSJ classification schema covers 15,000 peer-reviewed scientific journals, and it is translat-

ed by more than 22 international experts who volunteered their time and expertise, making the tools available to worldwide scientists. The levels in the OSJ hierarchy are respectively domain, field, and subfield. For example the journal 'Artificial organs' in OSJ is classified as follows:

- domain – *Applied Sciences*,
- field – *Engineering*,
- subfield – *biomedical engineering*.

Clearly, OSJ can be used straightforward for assigning tags to all the papers published in the journals that are present in the OSJ list. The problem appears for the publications out of the OSJ journal lists, as well as theses, publications being conference papers, chapters in the books, etc. To this end, we have designed and implemented Bayesian classifier, which was trained on the OSJ papers. So, the science domain classifier works as follow for each document:

- if the document is a paper from the OSJ list, take the tags assigned by OSJ to the journal,
- otherwise, use the Bayesian classifier on the available metadata, preferably including title, keywords and abstract, and use the result tags to classify the document.

The classifier provides only the tags from the second level of the ontology[5]. While experimenting, we verified two solutions: one classifier for all the OSJ fields or a tree of specific classifiers, each node representing a 'specialised' classifier. The experiments have shown that the solution with the tree of 'specialised' classifiers outperforms one common classifier. The tree of classifiers is a hierarchical structure with the depth 1, where each node represents a specialised classifier. The root is a classifier for the first OSJ level, its children are composed by six classifiers at level 2 (for each OSJ domain there is one fields classifier constructed). An average accuracy (10-fold cross validation) in a tree mode has reached 85%, whereas in the case of a single classifier for all the OSJ fields the accuracy was about 60%.

4.2 Semantic indexing

For implementing semantic indexing, special semantic resources, like domain ontologies, or thesaurus, play a crucial role in enhancing the intelligence of Web, mainly by means of enterprise search and in supporting information integration. Nowadays, most of the semantic resources cover only specific domains. They are created by relatively small groups of knowledge engineers and are very cost intensive to keep up-to-date with the domain changes. At the same time, Wikipedia has grown into one of the central knowledge sources, maintained by thousands of contributors. Bearing in mind that the whole university research domain cannot be covered by one specific domain ontology, we have decided to apply Wikipedia (Polish and English) as a semantic knowledge resource and implement Wikipedia-based semantic indexing of documents in the Ω-Ψ^R system information included in Wikipedia.

Wikipedia contains 30 million articles in 286 languages, including over 4.2 million in the English part. Additionally, it extensively uses ontology called DBpedia, which is a crowd-sourced community effort to extract structured information from Wikipedia and make this information available on the Web. DBpedia allows one to ask for sophisticated queries against Wikipedia, and to link various data sets on the Web to Wikipedia data. Since a few years, both Wikipedia and DBpedia are used in research in many areas involving natural language processing, in particular for information retrieval and information extraction (see e.g. [4, 5, 7]). Below we present how we use Wikipedia for the needs of semantic processing in the Ω-Ψ^R system.

Wikipedia articles consist of free text, with structured information embedded, such as *infobox tables* (the pull-out panels that appear in the top right of the default view of some Wikipedia articles, or at the start of the mobile versions), categorisation information, images, geo-coordinates and links to external Web pages. This structured information can be extracted and arranged in a uniform dataset, which can be queried. In our project we use Wikipedia in two approaches – *term oriented* and *text oriented*. The first one describes a given term by other words, which come from the Wikipedia content. The module responsible for this task is named WikiThesaurus. The text oriented approach consists of extracting keywords from a text using dictionary of terms (defined as Wikipedia article's titles). The module responsible for extraction is called WikiKeywordsExtractor. Both modules are working on a previously prepared database, built from the Wikipedia content.

WikiThesaurus is a module providing semantic information about an analysed term. It is inspired by the Milne and Medelyan works [9, 10]. It processes data in two steps. First, given a term extracted from the processed

[5] The first level of OSJ is too general, it has six broad categories: Natural Sciences, Applied Sciences, Health Sciences, Economics and Social Sciences, Arts and Humanities and General, whereas the third level is too detailed, and there is a problem with finding out a training set with a uniform distribution of categories and representative number of examples per category.

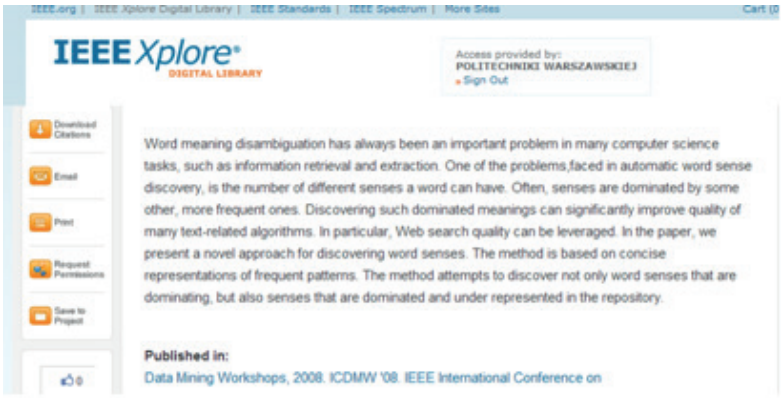

Figure 9. The example of abstract used in keywords extraction process

document, it searches for an article with the title equal to or at least containing this term. Then, the found article is processed in order to extract its labels[6], senses, translation or first paragraphs. In order to retrieve related topics to a given term we retrieve all the articles that are siblings in a domain, links-in and links-out. The functionalities provided by WikiThesaurus are as follows:

- retrieving senses for a term,
- retrieving short description for a term,
- retrieving alternative labels for a term,
- retrieving translations for a term,
- retrieving related topics to a term.

WikiKeywordsExtractor is used to extract keywords from a text using knowledge included in Wikipedia. The main idea and some partial solutions come from [7]. The module processes the input text in three steps:

- text preprocessing,
- terms filtering,
- terms evaluation.

4.2.1 Text preprocessing

KeywordExtractor processes the input text in such a way that for each term it looks in the Wikipedia preprocessed base for anchors, i.e. links to other articles or proper names. With the found anchors it builds a list of candidate keywords.

4.2.2 Term filtering

KeywordExtractor filters the candidates in order to remove the irrelevant terms (like articles describing pronunciation), or solve the polysemy problems. The latter one refers to the terms referring to the articles being the disambiguation pages, containing multiple meanings of the given term. For this case, the module has to choose only one meaning. The decision depends on categories matching, and the number of other candidates having link-in/out to a given meaning. The cleaned list of candidate articles is passed to the final phase.

4.2.3 Term evaluation

KeywordExtractor ranks terms (being candidate titles) using measures based on the common descriptors. Each candidate article is described by a bag of words (bow in the sequel), constructed from titles of related articles (as siblings, links-out, links-in). In the same way we describe the analysed text – a bag of words is built from terms, which are titles of candidates returned from the filtering phase. Next, each candidate is compared to the bow of the analysed text. The final measure is a derivative of cardinality of the intersection of the two sets (see Formula 1).

$$\text{Rank (candidate, text)} = \frac{|Bow(candidate) \cap Bow(text)|}{|Bow(text)|} \quad (1)$$

where *Bow(candidate)* is the set of terms describing the candidate article, and *Bow(text)* is the set of terms describing the analysed text.

Generally, those candidates, which have higher cardinality measure, are ranked higher[7]. Finally, only the candidates ranked above a given threshold are accepted. The titles of those candidates are the final list of keywords.

The example of keywords extraction is demonstrated in Figure 9.

[6] They are manually edited, and assigned to the articles by Wikipedia editors.

[7] Other similarity measures are now under tests.

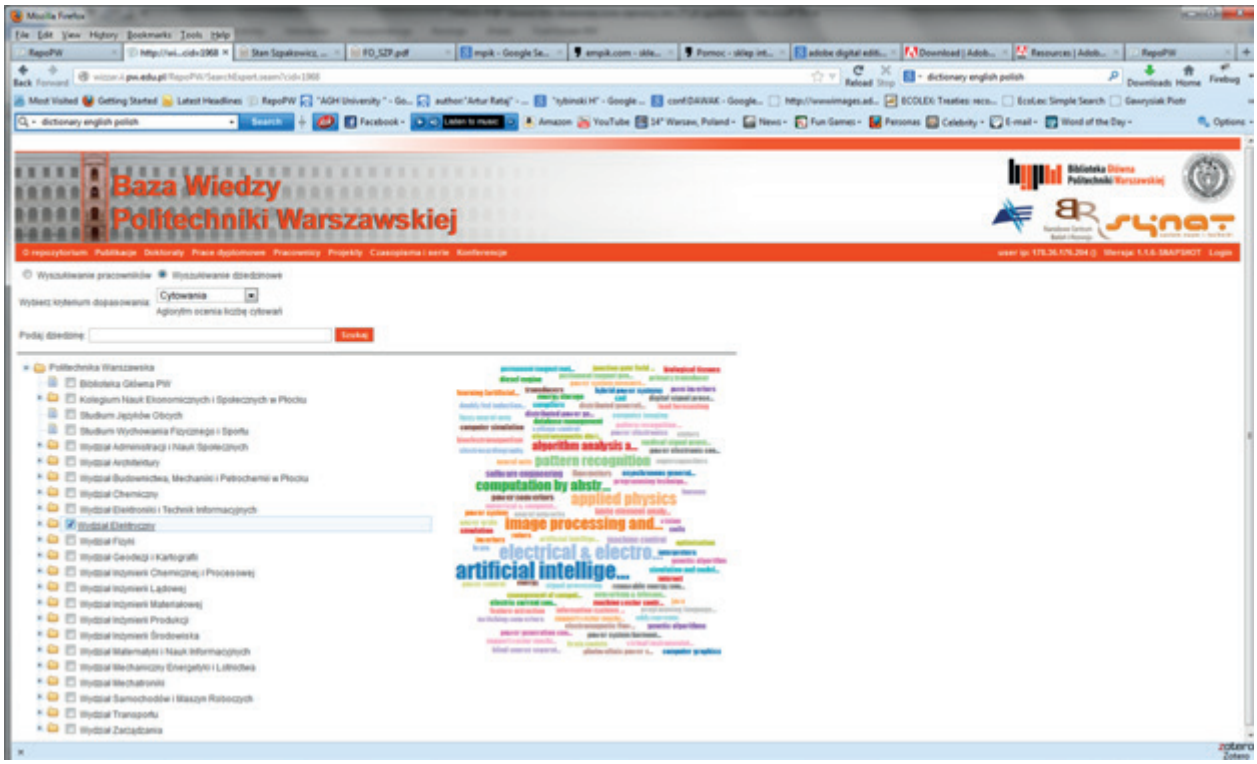

Figure 10. A map of the research areas for a selected faculty

The input text is the abstract from Figure 10. The extracted keywords are: *information, information retrieval, web search engine, computer science, word sense, World Wide Web.*

The extracted keywords are used to create research maps which facilitate expert search described in previous section. Publication keywords are aggregated on the authors' level and further on the affiliations' level creating cloud of words which visually depicts research areas of interests for the given person or institution. An example of a research map for a faculty and a researcher is presented in the Figure 11.

5 Conclusions

First, a simpler version of the system was deployed only at the Institute of Computer Science and was used only for scientific publications. Then, it was implemented at the Faculty of Electronics and Information Technology (FEIT), and it has been used for more than 2 years. In the meantime it gained the development level suitable for building the university knowledge base and the functionality of the current version of Ω-Ψ^R. As such, a year ago the system has been moved to the university level, and after a series of courses and training it starts working for all the faculties.

It turns out that the WUT Knowledge Base installation is positively accepted by the university faculties. Especially, as the functionality of Ω-Ψ^R goes beyond the typical functionality of institutional repository, providing appropriate means for many various groups of users it has chances to become a central knowledge source about the university research activities, the more that, due to applied intelligent tools (acquisition tools, reporting functionalities), the maintenance efforts of Knowledge Base are essentially reduced compared to the typical solutions.

The efforts devoted to the dissemination of university scientific achievements start bringing initial results. The usage statistics of WUT Knowledge Base show increasing interests from visitors from all around the world, especially Western Europe and North America. It is expected that those effects could be enforced after some time, when the database contents reaches a certain level. In order to speed up the saturation processes, the system will provide in the near future more crowd-sourcing features, in particular making possible uploading full texts by the authors themselves providing social media integration etc.

It also turns out that the Ω-Ψ^R software system itself is very successful and attracting much attention. In particular, the system was presented to other Polish universities, who requested the possibility to evaluate the system and are seriously considering using it. Installing the system on more than one university could enable cross-university

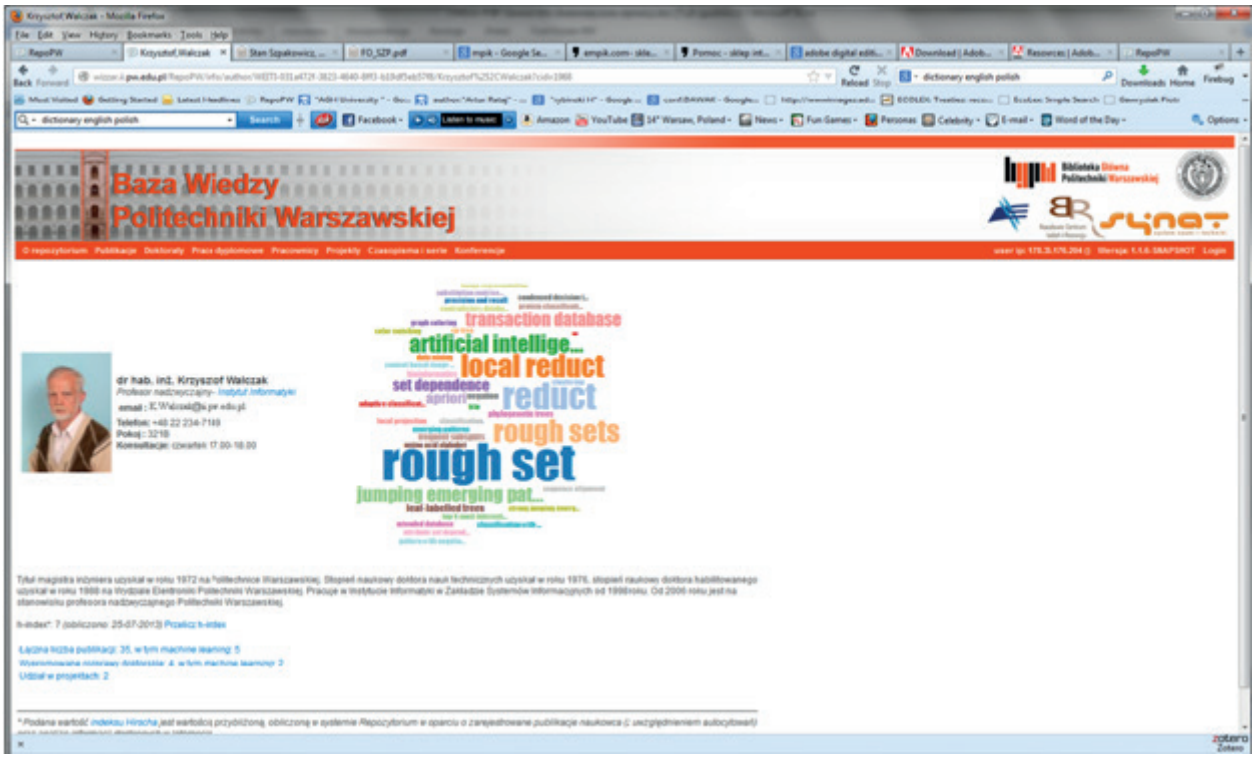

Figure 11. A map of the research areas for a selected faculty staff

resources sharing, and thus could lead to the development of completely new functionalities increasing synergy in Polish science. For instance, quick access to peers work from the universities network will be possible. In addition, the functionalities of looking for a team will eventually help building project teams consisting of researchers from various universities. Some comparative statistics would also be possible, which would stimulate the competitiveness.

Both, the Ω-Ψ^R system and its implementation as the WUT Knowledge Base seem to significantly contribute to the irreversible global trend of aggregating and sharing the scientific achievements. The initiative of the universities in Poland in implementing the university level research knowledge bases will provide means for building a modern nation-wide scientific information platform.

6 References

[1] Apache Jackrabbit, Getting Started with Apache Jackrabbit, The Apache Software Foundation, http://jackrabbit.apache.org/getting-started-with-apache-jackrabbit.html

[2] Bembenik R., Skonieczny Ł., Rybiński H., Niezgódka M. (eds.) - *Intelligent Tools for Building a Scientific Information Platform*. Studies in Computational Intelligence, vol. 390, 2012, Springer,

[3] Bembenik Robert, Skonieczny Łukasz, Rybiński Henryk, Kryszkiewicz Marzena, Niezgódka Marek (eds.) - *Intelligent Tools for Building a Scientific Information Platform: Advanced Architectures and Solutions*. Studies in Computational Intelligence, vol. 467, 2013, 548 p.

[4] Gabrilowich E., Markovitch S. - *Overcoming the brittleness bottleneck using Wikipedia: Enhancing text categorization with encyclopedic knowledge*, AAAI, 2006.

[5] Gabrilowich E., Markovitch S. - *Wikipedia-based semantic interpretation for natural language processing*. Journal of Artificial Intelligence Research, vol. 34, (443–498), 2009.

[6] Lucene, Welcome to Lucene, The Apache Software Foundation, http://lucene.apache.org/

[7] Medelyan O., Milne D., Legg C., and Witten Ian H. - *Mining meaning from Wikipedia*. Int. J. Hum. Comput. Stud., 67(9): 716–754, 2009.

[8] Medelyan O. - *Human-competitive automatic topic indexing*, *PhD Thesis*. University of Waikato. 2009.

[9] Milne D., Medelyan O., Witten I.H. - *Mining domain-specific thesauri from Wikipedia: A case study* [in] Proc. IEEE/WIC/ACM International Conference on Web Intelligence, Hong Kong, China, 2006, pp. 442–448.

[10] Milne D., Witten I.H. - *An effective, low-cost measure of semantic relatedness obtained from Wikipedia links* [in] Wikipedia and Artificial Intelligence: An Evolving Synergy, Chicago, IL, 2008, pp. 25–30.

[11] Open Archives Initiative Protocol for Metadata Harvesting, http://www.openarchives.org/pmh/

[12] Ontology of Scientific Journal, Classification of Scientific Journals, http://www.science-metrix.com/eng/tools.htm

[13] OMEGA-PSIR Administrator Manual (Podręcznik Administratorów), SYNAT, Politechnika Warszawska, 2013 (in Polish).

[14] Zotero, http://www.zotero.org/