Integration of Job Portals by Meta-search

Jürgen Dorn and Tabbasum Naz
Vienna University of Technology, Institute of Information Systems, A-1040 Wien
{dorn,naz}@dbai.tuwien.ac.at

Abstract
In the area of Human Resource Management, the trend is towards online exchange of information about human resources. For example, online applications for employment become standard and job offerings are posted in many job portals. However, there are too many job portals to monitor all of them if someone is interested in a new job.

We developed a prototype for integrating information of different job portals into one meta-search engine. First, existing job portals were investigated and XML schemes were derived automated from these portals. Second, translation rules for transforming each schema to a central HR-XML-conform schema were determined. The HR-XML-schema is used to build a form for searching jobs. The data supplied by a user in this form is now translated into queries for the different job portals. Each result obtained by a job portal is sent to the meta-search engine that ranks the result of all received job offers according to user's preferences.

Keywords: Business applications and case studies of interoperability, Meta-data and meta-models for interoperability, Interoperability for knowledge sharing

1 Introduction

Unemployment is not only a serious problem of developing nations i.e. Asia, Africa, Latin America but also a problem of developed nations. In Europe, unemployment rate increases sharply and almost continuously since the early 1970s. It increased further in the 1980s, to reach a plateau in the 1990s. It is still high today [1] [2]. According to [3] unemployment rate in January 2006 is 11.60 in Germany, 6.60 in Pakistan, 5.10 in Austria, 5.10 in United States and 4.70 in United Kingdom. One reason of the high unemployment is the problem of the inefficient distribution of job offers. Job opportunities are available but people are unable to access them. To drop the unemployment rate an improved search for job offerings may help.

The Web has drastically changed the online availability of data and the amount of electronically exchanged information. Many Web portals provide a search in

databases. The traditional search for jobs investigates newspapers, trade press, job fairs and employment recruitment agencies. All these methods were adequate in the past. Access to Internet has proven that these methods are too slow, expensive and lacking in their ability to deliver high quality candidates in the shortest possible time in the modern employment market. Thus, for example, the German Federal Employment Office (BA) has launched a "Virtual Employment Market" platform in 2003 to overcome the problems.

"The importance of the Internet for job procurement is increasing for the reason that three quarters of the people in the employment age are online." [4]. For a certain company, publishing online their job offers is a sign of good economic health, in that way e-recruitment becomes a sign of institutional publicity and ever more companies are publishing their job offers in the Web. Recruiters are interested to automate the pre-selection of candidates and to decrease transactions costs for publishing job postings and for pre-selecting [5]. But still people are facing problems in searching jobs due to the large number of online job search portals. Job offers also lack semantically meaningful annotation therefore search and integration into databases are highly difficult [4].

We describe a meta-search prototype that integrates job portals so that users can access more then one job portal at a time. The paper focuses on the problem of automatically integrating job search interfaces.

The prototype consists of number of components shown in Fig. 1. First, an interface extractor derives attributes from job search interfaces. Second, an XML-Schema generator creates XML schemes. Third, each schema is translated into a HR-XML-conform schema by explicite translation rules. The system solves problems of different representational concepts in different used search engines. Since the search engines use different data structures, different concepts and different granularities of knowledge we use a domain-ontology to translate between concepts.

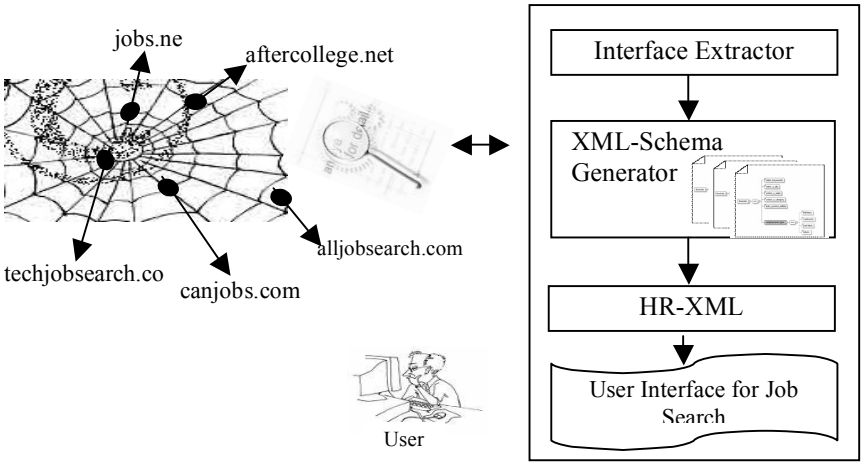


Fig 1. Meta-Search in Human Resource Management

The rest of the paper is organized as follows. Section 2, is related work in meta-search domain, HR-XML and job search portals. In section 3, the interface

extractor and the XML schema generator is described. Section 4 describes integration of schemes, meta-data and HR-XML schema into a unified interface. Finally, section 5 contains the conclusion and outlook.

2 Related Work

Meta-search, information extraction and integration are important problems. Schema extraction, matching and then integration have received much recent attention. He et al. [6][7][8] worked in meta-search and developed WISE: iExtractor for interface extraction and WISE-Integrator for automatic schema, attribute values, format and layout integration. WISE-Integrator deals with e-commerce based search engines but not specifically for job search engines. WISE-Integrator uses a positive and predictive match based clustering approach for the identification of matching attributes. They apply a majority rule for choosing global attribute names of the cluster.

Lixto suite [9] provides a hardwired meta-search solution. It consists of a visual wrapper and a transformation server. Lixto's visual wrapper is used for creating wrapper that extract the relevant information from HTML documents and translate it into XML, which can be queried and further processed. The Lixto transformation server provides data flow processes like collecting, transforming, concatenating, sending and storing of XML documents [10].

MetaQuerier [11] [12] is a tool for developing schema models and for extraction and matching Web query interfaces. MetaQuerier applies a statistical /probabilistic approach for schema matching. The authors claim that their system fully automates all tasks in streamline to output semantic matching. MetaQuerier considers only element labels but other meta information about the interface like domain, value, relationship types have not been discussed.

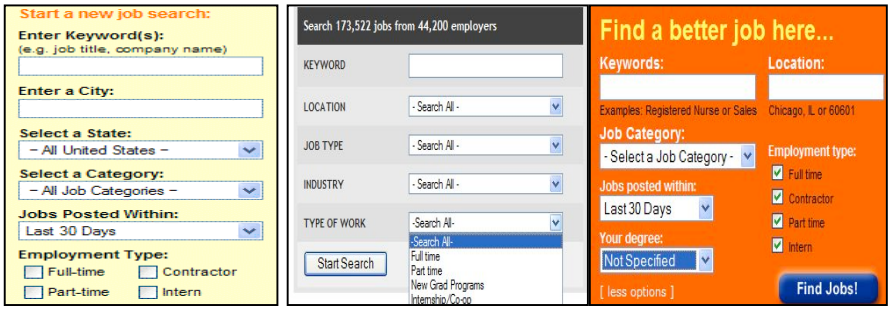
The KnowledgeNets project [4][13][14] introduces another approach to solve the problems in recruitment process by technologies from the Semantic Web. Using Semantic Web technologies, the data exchange between employers and job portals can be based on a set of controlled vocabularies which provide shared terms for describing occupations, required skills and educational background to perform semantic matching [13]. All job portals can operate on the same information and postings would reach more applicants, resulting in higher market transparency. Job portals could offer semantic matching services, which would calculate the semantic similarity between job postings and applicant's profiles. In [13], an human resource ontology (HR ontology) integrating the existing widespread used standards is described. This ontology is divided into sub-ontologies which are used in both job posting and job application descriptions.

Peig et al. [15] discuss a problem of interoperability while developing meta-search and presented an idea of a central agent mapping meta-data schemes between users and content providers. They state that ontologies provide primitives, needed to retrieve information about some categories of contents.

HR-XML organization (www.hr-xml.org) is dedicated to the development and promotion of a standard suite of XML specifications to enable e-business and the automation of human resources-related data exchanges. By developing and

publishing open data exchange standards based on XML, the Consortium provides the means for any company to transact with other companies without having to establish, engineer, and implement many separate interchange mechanisms. XML Schemas define the data elements for particular HR transactions, as well as options and constraints governing the use of those elements. The HR-XML Consortium has produced schemas covering major processes, as well as component schemas, used across multiple business processes [16].

The integration of schemes in a certain domain seems to be easier as between different domains because sources are more homogeneous and are thus easier to be integrated [17]. However, different search interfaces in the same domain can contain different number of attributes, different names for representing the same type of elements and organize the attributes in different way as is shown in Fig 2.



a b c

Fig. 2. Job Search Interfaces a. jobs.net; b. aftercollege.com; c. careerbuilder.com

Jobs.net and aftercollege.com have interfaces with different attributes (see Fig 2). Jobs.net has an attribute "Employment Type" and aftercollege.com has an attribute "TYPE OF WORK" to represent the same concept. Jobs.net and aftercollege.com do not ask about "Degree" of a job seeker but careerbuilder.com asks about it. So there are problems in the integration of different job search portal's interfaces in the job domain as interfaces are representing the schemas in different ways.

3 Extraction of Data Structure

First step in meta-search is to construct a machine understandable format of the search interface of each portal. The process of generating machine understandable format is completed in two phases "Interface Extraction" and "XML Schema Generation". Interface extraction phase is implemented almost in the same way as given in [6] and [7] but it has been developed specifically for job search portals with some extensions.

3.1 Interface Extraction

In the "Interface extraction" phase we identify logic attributes by grouping related labels and elements and then derive meta information of these attributes. It is further divided into two phases i) Attribute Extraction ii) Attribute Analysis.

3.1.1 Attribute Extraction

During attribute extraction, logic attributes are identified and then related elements and labels are grouped together. First, we give the URL of an portal to the interface extractor. The HTML form may be of devoted type or co-existing type [8]. If the interface is of devoted type, the program can take the form as a target form otherwise user has to select the target form. <FORM> and </FORM> tags of HTML are used to identify a form.

Second, individual labels and form control elements (e.g. inpur fields, check-boxes, and radio buttons are extracted. Text between elements is extracted to determine labels for control elements. Moreover
, <P> and </TR> tags are also extracted to determine the physical location of elements and labels. Extra scripting and styling information i.e. font sizes, styles is ignored. Logically, elements and their associated labels together form different attributes. Attributes can have one or more labels and elements. For example, the "Skills/ Keywords" attribute in Fig. 3 has four elements including a text area and three radio buttons.

Third, an interface expression (IEXP) to provide a physical layout of a search interface is constructed. 't' is used to represent a text or label, 'e' to represent an element and '|' to represent line separation. IEXP for the interface in Fig. 3 is te|te|t|ttt|teeee|t|ttete|tt|t|te|ee|ee|ee|e|t|te|t|te|te where first "t" represents a label/text "Job Category", first 'e' represents a select list element next to label "Job Category", first "|" represents a line separator and so on.

Fourth step is to group the labels and elements that semantically correspond to the same attribute, to find an appropriate attribute label/name for each group. In Fig. 3, a label Skills/Keywords", a text area, three radio buttons and their values all belong to the same attribute. For grouping labels and elements, a LEX (layout-expression-based extraction) technique described in [7] is implemented. LEX finds an appropriate attribute label for each element, either in the same row or above the current row. During the grouping process, if the neighbor text of an element is not considered as an attribute label of e, then it is assumed to be an element label of e. For example in Fig. 3, "From" and "To" are recognized as element labels for text boxes instead of attribute label. Our interface extractor uses heuristics measures i.e. 'Ending Colon", 'Distance of element and text', 'Vertical Alignment', 'Priority of current row' from [7] with minor changes to identify an appropriate label for elements. One heuristic measure is introduced to count number of labels with colons and without colons from an IEXP. If a number of labels with colons is greater then number of labels without colons then apply the ending colon heuristic otherwise choose closest label for an element.

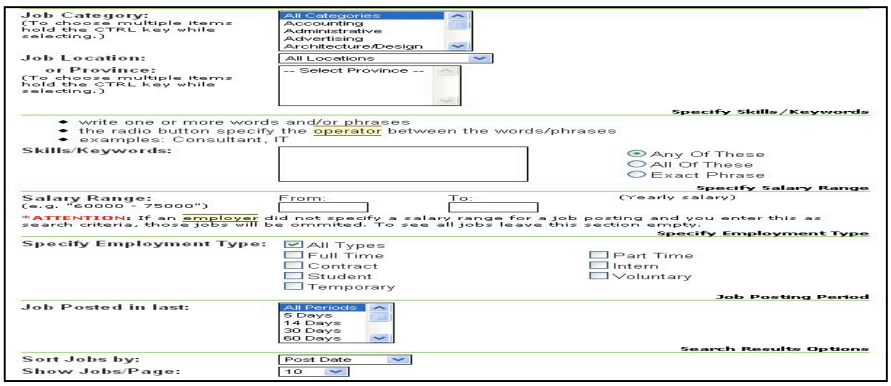


Fig. 3. Job Search Interface of jobshejobs.com

3.1.2 Attribute Analysis

Aim of the "Attribute Analysis" phase is to collect information about elements i.e. relationship type (RT), domain type (DT), default value (DV), value type(VT) and unit. Semantics of domain elements and meta information is also identified.

RT can be of "Group type", "Range type" or "Part type". Check if all elements of attributes are check boxes or radio buttons and are greater then one in number then RT is of group type. In Fig. 3 "Specify Employment Type" attribute is an example of group type elements. Check if labels contain some keywords or patterns e.g. "between", "from", "to", then RT is of range type. Few job search portals contain "from", "to" labels with salary range attribute as in Fig 3. Elements that are not of group and range type are treated as a part type.

Next step is to extract meta information attributes i.e. domain type (DT), default value (DV), value type (VT) and unit. DT indicates how many values can be specified on an attribute for each query. DT can be range, infinite, boolean or finite. If RT type of elements is of range type then DT is recognized as range. If relationship type is not range and there involves a textbox or text area then DT is infinite. If attribute has a single checkbox, then DT is boolean. Otherwise, if selection list is involved then DT is finite. DV only occurs if there is a selection list, radio buttons or checkboxes and is always marked as checked or selected in an element. If an attribute contains just textboxes or text areas then attribute has no default value. VT can be determined by analysis of attribute name. It can detect date, time, currency, integer and string data type. If an attribute name contains "range" or "number" then value type is numeric. Otherwise if the value type is not date, currency, and number then it is considered as string. In job search portals, sometimes salary attribute contains a unit in label or values of some attribute. Interface extractor can detect unit if label contains "EUR", "€" etc. Table 1 represents the attribute names and meta information collected during interface extraction and attribute analysis phase for job interface shown in Fig. 3.

Table 1. Meta Information for jobshejobs.com

Attribute Name	RT	DT	DV	VT	Unit
job_ category	None	Finite	All categories	String	Nil
job_ location	None	Finite	All locations	String	Nil
or_ province	None	Finite	-Select Province-	String	Nil
skills_keywords	Group	Infinite	Any of These	String	Nil
salary_ range	Range	Range	Nil	Integer	Nil
specify_ employment_ type	Group	Finite	All Types	String	Nil
job_ posted_in_ last	None	Finite	All Periods	String	Nil
sort_ jobs_by	None	Finite	Post Date	String	Nil
Show_ jobspage	None	Finite	10	String	Nil

3.2 XML-Schema Generation

Schema model developed in section 3.1 is used in schema generation process to define the legal building blocks of an XML document. An XML Schema defines the elements, attributes, child elements, order of child elements, data types of elements and attributes, default and fixed values for elements and attributes [19]. Character set for schema is collected from the HTML page, if there is "charset" attribute otherwise consider "iso-8859-1" as a default character encoding. During this process schema elements and XML schema equivalents for HTML elements are identified and is given in more detail in our work [20].

3.2.1 Schema elements

<RootJob> is automatically created with a sequence indicator as the root element of schema and it contains all other elements from the search interface as child elements. An XML Schema may contain simple and complex elements.

A simple element is an XML element that contains only text. It cannot contain any other element or attribute. Textboxes can be of the simple types. In Fig. 2.a, a text box with label "Enter Keywords(s)" is considered as a simple element because it contains only text and does not contain any other element or attribute. In an XML Schema it can be represented as "<xs:element name="enter_keywords" type = "xs:string"/>". Default or fixed value for elements can also be specified.

A complex element is an XML element that contains other elements and/or attributes. There are four kinds of complex elements i.e. empty elements, elements that contain only other elements, elements that contain only text, elements that contain both other elements and text. Complex elements may contain attributes as well. In Fig. 3, "Salary Range" is an example of complex element that contains s "from" and "to" as child elements. If labels "from" and "to" are on the HTML page for textboxes, these labels are used as name of elements. Sometimes when labels are not available, internal names of elements can be used. Elements can have a "type" attribute that refers to the name of complex type to use.

3.2.2 XML Schema Equivalents for HTML Elements

In this section, we explain how each HTML elements i.e. text filed, text area, radio button, check box, select list from the HTML search interface can be represented in XML Schema .

1. Text boxes and text areas on the search interface are represented as simple elements.
2. A group of multiple radio buttons on search interface is also a simple element having a default value, restriction and enumeration list. Text/label associated with radio button is taken as a value for that radio button.
3. Multiple checkboxes on a search interface with domain type "group" are treated as complex type element with attributes "fixed" and "minOccurs". The <minOccur> specifies, how many values are selected for a checkbox.
4. If select element in HTML does not contain attribute "multiple" then select list is single-select list otherwise it is multiple select list. A single-select list on search interface is treated as a simple element having a default value, restriction and enumeration list in the same way as radio buttons. But multiple-select list on search interface is treated as complex type element and it must contain a "type" attribute that refers to an element of complex type. In Figure 3,"Jobs Posted in last" is a single-select list and "Job category" is a multiple-select list [20][21].

A complete XML schema for search interface can be developed by combining XML equivalents for each HTML element.

4 Integration of Meta-Data

Integration of meta-data involves three steps i) schema integration ii) form generation iii) and data integration.

4.1 Schema Integration

During schema integration, schemes generated for different job portal's interfaces are translated into a HR-XML schema for a meta-search of jobs.

Table 2 shows a list of common attributes, available in different job portals. An asterisk (*) in a cell marks the presence of the attribute. "Job Category" represents a grouping of jobs under one or more classification schemes that is meaningful to an organization i.e. IT, Accounting, Education. "Job Type" represents the type of hours i.e. Full Time, Part Time. In some portals "Job Type" also represents the nature of the position i.e. Contract, Temporary, Volunteer. Some job portals provide more specific concepts for job category and represent it as "Industry". Some other attributes found are "Travel" that is information regarding if the person is willing to travel, "Experience" information about work experience or education in years, "Posted within" when job was being posted. Some attributes are related to decide on the ranking.

Table 2. Common Attributes in job domain

Attributes / Job Websites	Keyword	Location	State	Country	Province	City	Job Category	Job Type	Industry	Degree	Salary
Jobs.net	*		*			*	*	*			
Aftercollege.com	*	*					*	*	*		
Clearchannel.com	*		*	*		*	*				
Jobmonkey.com	*	*					*				
Careerbuilder.com	*	*					*	*		*	
Techjobsonline.com	*	*					*	*			*
Promotions.	*		*			*					
Brightspyre.com	*						*				
Careerscafe.com	*		*	*	*	*		*		*	*
Jobinterviewonline.com	*		*			*	*	*			*
Topconsultant.com		*					*				*
Jobshejobs.com	*	*			*		*	*			*
Search2.workopolis.c	*	*				*	*		*		
Directjobs.com	*	*					*	*			
Alljobsearch.com	*			*		*	*	*			

Integration of interface schemes is divided into two parts: schema matching and schema merging. During schema matching, semantic correspondence between interface attributes is identified and each schema is translated to the HR-XML schema. The HR-XML schema used is derived from "Job and Position Header", "Worksite", "Educational History", "Postal Address", "HR" schemes. Table 3 represents the name of HR-XML schemes and attributes in HR-XML schemes that are used for capturing common attributes of job search portals. During schema merging, a single scheme is derived for the meta-search user interface. A domain-ontology contains the HR-XML attributes and attributes from job portals. For an attribute of the job portal interface, a corresponding attribute from HR-XML is obtained by using the similarity relation of the ontology. The HR-XML attributes are further used in the construction of the unified user interface.

Table 3. HR-XML attributes and schemas for common job portal's interfaces

Job Portal's Attributes	HR-XML Schema	Attributes in HR-XML Schemes
Job Category, Industry	Job and Position Header	JobCategory
Job Type	Job and Position Header	PoistionType, TypeOfHours
Qualification	Education History	SchoolName, Degree
Location, State, Country, Province, City	Postal Address	Region, Municipality, CountryCode
Travel	Human Resource	Preferences

4.2 Form Generation

We need to construct a user interface which contains all distinct fields. Dragut et a. [23] emphasize the importance of meaningful labeling of elements and state that the labels assigned to their elements must be carefully chosen to convey the meaning of each individual element. For-example, one job portal use "Employment Type" to represent the job preferences and other may use "Type of work" or "Job Type". From different attribute names, the user interface must contain the most meaningful and appropriate one. The problem of carefully choosing the meaningful label is solved by HR-XML schemes. During the form generation, a unified form given the above discovered schema matching and merging is constructed. The order of elements in the user interface has also some importance, so common attributes are placed at higher position. The form generation is supported by XForms which enables the generation of the form from an XML schema and also the easy adaptation to different user clients [24].

4.3 Data Integration

The aim of the data integration is to determine the values of different attributes for the user interface. We have to choose the values that are semantically unique. These values should also be compatible with the local values. Since the search engines use different data, different concepts and different granularities of knowledge we use a domain-ontology to translate between concepts.

After analyzing job search portals, we found that there are two types of semantic relations: synonymy and hypernymy between concepts. Synonymy means

that two terms x_1 and x_2 are synonyms, if they have same meaning. For example, "programmer" is synonym for "coder". Hypernymy means that a term x_2 is hypernym of x_1, if x_1 is more generic concept of x_2. For example, "IT" is hypernym of "IT-Hardware".

For finding synonyms and hypernyms in job portals, again our ontology is used. Normally, hypernymy relationships exist for attribute "Job Category" or "Industry" in job search interfaces. In the job domain, most of the attributes take alphabetic values and are of finite type, so we are focusing on the merging of alphabetic domains. Only "Salary" and "Experience" attributes can take numeric values and salary display also require currency conversion. If alphabetic values are synonyms, then we have to choose which to represent in the user interface. To solve this problem, we maintain a list of distinct values and then follow majority rule i.e. choose the most frequent one from the synonyms for unified interface.

If values are hypernymy (mostly in case of job category), then we find a semantic relationship between values by using the taxonomy for job categories. Global values for the user interface may represent a generic concept or a specific concept. Both of them have their pros and cons. If generic concepts are chosen then query against the unified interface may need to be mapped to multiple values in some local interfaces. If we keep specific concepts, then for users who are interested in more generic concepts may have to submit multiple queries using the more specific concepts, resulting in less user-friendly interface. So a combined approach is used to solve this problem providing a hierarchy of values, including both generic and specific concepts. Multiple categories may be formed for the values corresponding to each global and a value hypernymy hierarchy is created for each category [6].

Fig. 4 represents that interface placementindia.com has only generic concepts for attribute job category i.e. Computers/IT whereas interface clickjobs.com has specific concepts i.e. IT-Hardware, IT-Networking and IT-Software.

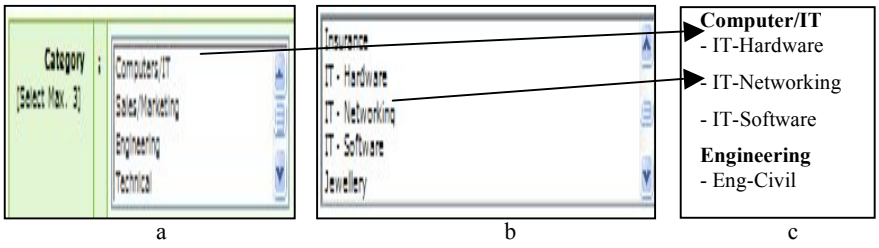


Fig 4. a. Generic concept by placementindia.com; b. Specific concept by clickjobs.com; c. Combined concept

A domain-specific ontology identifies the relationship between the values of two interfaces. If in the user interface the generic concepts "Computer/IT" is represented then a user interested only in a specific field i.e. IT-Hardware will get irrelevant results. So query against the user interface may need to be mapped to multiple values in some individual interfaces. If only specific concepts are represented in the user interface i.e. "IT-Hardware, IT-Networking, IT-Software" then a user is interested in all categories he will make three queries to get the desired result. So, the solution is a combined approach providing a hierarchy of

values, including both generic and specific concepts as in Figure 4 c. If a user is interested in Computer/IT related jobs then the meta-search engine can generate one query for Figure 4.a interface and three queries for Figure 4.b interface. But if a user is interested in IT-Software the meta-search can generate one query for Figure 4.a interface and one for Figure 4.b interface. This solution can solve the problem and helps the user in job search.

5 Conclusion and Outlook

We have presented an approach for integrating data from different job portals in a meta-search in order to support job seeking people to master the large number of available job portals. Our focus in this paper was on the automated extraction of the structure of provided information. If this structure is available it can be used in meta-search to integrate the different sources. The vision would be to generate agents that can supply available jobs dynamically with a Web service interface as recommended in [25].

The main difference between our work and existing works [6] [7] [8] [11] is that we used HR-XML for schema integration. Each scheme is translated to a HR-XML-conform schema. There is no published work for integration of machine readable schemas and HR-XML schema for meta-search in human resource management. Moreover no research paper discusses about how to represent XML Schema equivalents for HTML elements i.e. text boxes, text areas, radio buttons, check boxes, select lists and generation of XML Schema for HTML search interface.

Modern human resource management focuses more on competencies than on job titles or job positions. At the moment only few job portals reflect this trend. However, in the near future this will change and the detailed description of required competencies will gain impact. This can be modelled with HR-XML, too. If job offerings are based on the specification of required competencies and job applicants submit queries with their detailed competencies (possibly part of CV) then the matching will be more complex and fuzzy-based.

6 References

[1] Bertola G, (2006), Europe's Unemployment Problem. For Artis M and Nixson F, Economics of the European Union 3rd edition, Oxford University Press

[2] Olivier B, (2006), European Unemployment: The Evolution of Facts and Ideas, Economic Policy, Vol 21, Issue 45:5

[3] http://www.photius.com/rankings/economy/unemployment_rate_2006_0.html

[4] Falk.T, Heese R, Kaspar C, Mochol M, Pfeiffer D, Micheal T,Tolksdorf R, (2006), Semantic Web Technologien in der Arbeitsplatzvermittlung. Informatik-Spektrum 29: 201-209

[5] Harzallah M, Lecl`ere M, Trichet F, (2002), CommOnCv: Modelling the Competencies Underlying Curriculum Vitae. ACM Proceedings of 14th International Conf on Software Engineering and Knowledge Engineering, Italy: 65-71

[6] He H, Meng W, Yu C, Wu Z, (2004), Automatic Integration of Web Search Interfaces With WISE-Integrator. VLDB Journal, Vol. 13, No. 3: 256-273

[7] He H, Meng W, Yu C, Wu Z, (2005), Constructing Interface Schema For Search Interfaces of Web Databases, 6th International Conference on Web Information Systems Engineering (WISE05), New York City:29-42
[8] Peng Q, Meng W, He H, Yu C, (2004), WISE Cluster: Clustering E-Commerce Search Engines Automatically. 6th ACM International Workshop on Web Information and Data Management (WIDM 2004), Washington, DC:104-111
[9] Jaura O, (2006), A Scalable Special-Purpose Metasearch Engine. PhD Thesis, DBAI, Institute of Informatics, Vienna University of Technology, Austria
[10] Gottlob G, Koch C, Baumgartner R, Herzog M, Flesca S, (2004), The Lixto Data Extraction Project - Back and Forth Between Theory and Practice. In Symposium on Principles of Database Systems: 1-12.
[11] B. He, Z. Zhang, Chang KC, (2004), Towards Building a MetaQuerier: Extracting and Matching Web Query Interfaces. In NSF (IDM Workshop, Boston, Massachussett.
[12] He B, Chang KC, (2003), Statistical Schema Matching across Web Query Interfaces. In Proceedings of the 2003 ACM SIGMOD Conference California
[13] Bizer C, Heese R, Mochol M, Oldakowski R, Tolksdorf R, Eckstein R , (2005), The Impact of Semantic Web Technologies on Job Recruitment Process. International Conference on Economical Informatics, Germany: 1367–1383
[14] Mochol M, Oldakowski R, Heese R, (2004), Ontology based Recruitment Process. Workshop over Semantic technologies for Information Portals, Germany
[15] Peig E, Delgado J, Pérez I, (2001), Metadata Interoperability and Metasearch on the Web. Proceedings of the International Conference on Dublin Core and Metadata Applications 2001, Tokyo, Japan: 247-254
[16] Allan, Ch.; Pilot, L. (2001), HR-XML: Enabling pervasive HR eBusinees, XML Europe 2001, Berlin, Germany
[17] Chang KC, He Bin, Zhang Z, (2005), Toward Large Scale Integration: Building a MetaQuerier over Databases on the Web. In Proceedings of the Second Conference on Innovative Data Systems Research (CIDR 2005), Asilomar, California: 44-55
[18] http://metaquerier.cs.uiuc.edu/repository/datasets/tel-8/index.html
[19] http://www.w3schools.com/schema/schema_intro.asp
[20] Naz T, (2006), An XML Schema Generator for HTML Search Interfaces, Technical Report, DBAIEC, Institute Faculty of Informatics, TUWien, Austria.
[21] http://www.w3.org/TR/xmlschema-0/
[22] Wu W, Doan A, Yu C, (2005), Merging Interface Schemas on the Deep Web via Clustering Aggregation. Proceedings of the Fifth IEEE ICDM,USA: 801–804
[23] Dragut EC, Yu C, Meng W, (2006), Meaningful labeling of Integrated Query Interfaces, interface. Proceedings of the 32nd international conf on VLDB: 679–690
[24] Rainer,A., J. Dorn and P. Hrastnik, Strategies for Virtual Enterprises using XForms and the Semantic Web, Proceedings of International Workshop on Semantic Web Technologies in Electronic Business (SWEB2004), Berlin, October
[25] J. Dorn, P. Hrastnik, and Rainer, A. An Advanced Meta-Search Engine, submitted I-ESA 2007