JobOlize
Headhunting by Information Extraction in the era of Web 2.0

Christina Buttinger[†], Birgit Pröll[†], Jürgen Palkoska[†], Werner Retschitzegger[‡]
Manfred Schauer[±], Reinhold Immler[±]

[†]*Institute for Application Oriented Knowledge Processing (FAW),*
[‡]*Information Systems Group (IFS),*
Johannes Kepler University Linz, Austria
{christina.buttinger | birgit.proell | juergen.palkoska | werner.retschitzegger}@jku.at
[±]*JoinVision E-Services GmbH, Austria*
{manfred.schauer | reinhold.immler}@joinvision.com

Abstract

E-recruitment is one of the most successful e-business applications supporting both, headhunters and job seekers. The explosive growth of online job offers makes the usage of information extraction techniques to build up, e.g., job portals in a semi-automatic way a necessity. Existing approaches, however, hardly cope with the heterogeneous and semi-structured nature of job offers nor do they consider potentials offered by Web 2.0 technologies. This paper proposes an information extraction system called "JobOlize"[1], realized for arbitrarily structured IT job offers. To improve extraction quality, a hybrid approach is employed, combining existing NLP-techniques with a new form of context-driven extraction, incorporating layout, structure and content information. To allow users a proper adaptation of the extraction results while preserving the look and feel of the original Web pages, a rich client interface is provided. The improvements in extraction quality are justified on basis of a case study and the experiences gained are generalized and critically reflected by discussing lessons learned.

1. Introduction

The Internet has already become the most important medium for recruitment processes. Over the past years, the e-recruitment market grew steady. In the EU in 2007, 70% of all job offers were published online, and more than 56% of employments were results of online offers [6]. To manage the large amount of job offers on the Web and to adequately support, both, headhunters and job seekers, the number of job portals is increasing rapidly (cf., e.g., Monster [18], Stepstone [24] or Jobscout24 [12]), some of them even allowing matchmaking between offers and profiles of job seekers.

A crucial prerequisite for these systems is the usage of proper techniques for automating the extraction of relevant information from available job offers. Many existing information extraction techniques have in common that a target Web page is automatically atomized into its constituents called *tokens*, which further might be *annotated* on basis of a vocabulary defined by a common schema. Some of these techniques focus more on Web pages having a more or less common underlying structure and perform *screen scraping*, meaning that information is extracted based on, e.g., the HTML tags used, whereas others emphasize more on completely unstructured text and employ *natural language processing (NLP)*, analyzing, e.g., the grammar of the content's language (for a survey of extraction techniques, cf., e.g., [14]).

Considering the area of e-recruitment, job offers fall typically under both categories – although a common structure between them can seldom be found, there are at least structured parts, which could be used for improving the quality of the extraction process. Existing techniques hardly cope with this heterogeneous and semi-structured nature, thus decreasing the quality of extraction results. In addition, the environments provided by existing approaches for improving the extraction results manually, e.g., by manipulating the annotations of the extracted information, is most often realized as a heavy-weight desktop application, taking no advantage of upcoming Web 2.0 techniques [25].

[1] This prototype, which has been partly funded by the Austrian Research Promotion Agency FFG under grant 813202, has been used for further development of the Austrian online job portal `www.joinvision.com`.

This paper proposes an information extraction system called "JobOlize", which is realized for dealing with arbitrarily structured IT job offers. To improve extraction quality, a hybrid approach is proposed which combines existing NLP-techniques to deal with completely unstructured information, with a new form of context-driven extraction, incorporating layout, structure and content information. As demonstrated in a case study, this hybrid approach considerably improves the quality of the extracted information. To allow users a proper adaptation of the extraction results while preserving the look and feel of the original Web page, a rich client interface is provided using Mozilla's Firefox extensions, and the XML-based GUI language XUL [26]. The results and experiences gained during the development of JobOlize have been used for further development of the Austrian online IT job portal JoinVision (`www.joinvision.com`).

The paper is structured as follows. Section 2 presents the architecture of the JobOlize prototype, especially focusing on context-driven extraction together with the rich client interface. Section 3 discusses a case study evaluating the improvements in extraction quality and provides a comparison to related approaches. Section 4 concludes the paper with a critical reflection in terms of lessons learned.

2. The JobOlize Prototype

This section presents the architecture of JobOlize, especially focusing on the context-driven extraction process and describes functionality and implementation aspects of the system's rich client interface realized on basis of Web 2.0 techniques.

2.1 Context-Driven Information Extraction

The architecture of JobOlize, depicted in Figure 1, is divided into two core components, a *knowledge base* providing a domain ontology as well as an extraction rule base and a *pipeline* consisting of different extraction components. Parts of the system are realized on basis of the existing information extraction infrastructure *GATE (General Architecture for Text Engineering)* [3] which has been favored mainly because it is a mature open source implementation providing a highly extensible pluggable architecture.

e-Recruitment Domain Ontology. For representing the annotation vocabulary used by our extraction system, a *light-weight domain ontology* has been developed containing 15 core concepts of job offers which should be extracted (e.g., job title, IT-skill, language skill, operation area and graduation). A design goal in this respect was to heavily build on reasonable concepts of existing ontologies in the area of e-recruitment and human resource (HR) management, backed up by our experience in developing ontologies [20], [22], [23]. Most notably, from the ontology of Mochol et al. [17] which is based on the comprehensive HR-XML standard [10], we have adopted the concept of skill levels, whereas the work of Garcia et al. [7] was influential for our notion of offer characteristics. Our ontology also considers specialisation hierarchies of IT skills (e.g., "Oracle" and "MySQL" are specializations of "DBS") and equivalence relationships (e.g., "DB" is semantically equivalent to "DBS"), which are of particular interest for the extraction process. Finally, to consider also job offers in different languages, each concept contains a language property.

Extraction Rule Base. The second part of our knowledge base contains about 50 *extraction rules* ranging from very simple ones, responsible, e.g., for matching input tokens with ontology concepts to rather complex ones for, e.g., job title detection. The rules are in fact condition-action tuples formulated using the regular-expression-based language *JAPE* (*Java Annotation Pattern Engine*) provided by GATE and callouts to Java code for realizing more complex tasks.

Pipeline of Extraction Components. On basis of the second core part of our architecture, the pipeline, annotations of web pages are incrementally built up by streaming input pages through the different components of the pipeline, each of them being responsible for a certain annotation task, thereby adding new or manipulating already existing annotations. Parts of this pipeline are realized on basis of GATE which provides not only predefined, pluggable components for realizing basic extraction tasks but also allows to plug-in self-developed, customized ones. To be more specific, JobOlize reuses existing *NLP-based components* for tokenizing and stemming and furthermore realizes *customized components* on basis of Java for *preprocessing* the input Web pages (e.g., eliminating JavaScript code), for *post-processing* (e.g., exporting annotated Web pages as XML documents) and for the core task of identifying those tokens of the input Web page, which are relevant for our purposes, e.g., IT-skill with an associated skill-level like "basic knowledge in Java programming". The identification of relevant tokens is performed on basis of four different customized components, as described in the following, which are making use of the above mentioned knowledge base.

Initial Annotation. The first component is responsible for an *initial annotation* of the tokens of the input Web page with appropriate concepts defined by our domain ontology. For most of the tokens, this task is straightforward, but considering the domain concepts IT-skill and language skill, context-driven processing is required in order to determine their corresponding

levels. In particular, not only the position of the skill level with respect to the skill type itself is taken into account by means of appropriate JAPE rules, but also, e.g., if it is located within the same sentence or not.

Page Segmentation. The remaining three components promote the basic idea of context-driven extraction even further, with the ultimate goal to improve the quality of the extraction results. For this, the Web page

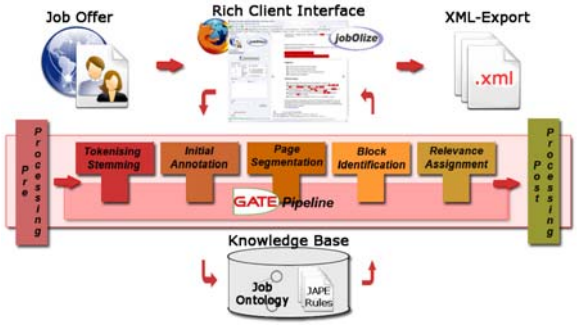

Figure 1. Overall Architecture of JobOlize

is first segmented into three parts, a *top part*, a *content part* and a *bottom part*, simply taking the *content* itself into account. This is done by identifying common text fragments between two job offers of the same Web site on basis of the well-known *Longest-Common-Subsequence algorithm (LCS)* [11]. These common text fragments represent the top and bottom parts of a page and contain tokens which are most probably irrelevant for further processing. For example, the occurrence of "powered by Typo3" within the bottom page would lead to an IT-skill annotation of "Typo3" during the initial annotation phase, being refined by page segmentation, i.e., classified as irrelevant.

Block Identification. The content part of the Web page identified before, is further divided into so-called *blocks*, representing a couple of tokens which "visually" belong together and are normally grouped under a header title. For the identification of such blocks, first, context in terms of *layout information* (e.g., a <b>-tag) and *structural information* (e.g., a -tag) is considered by appropriate JAPE rules. In a second step, context in form of *content information* is used to categorize the identified blocks. In particular, on basis of their header title and the corresponding domain concepts defined by our ontology, blocks are categorized and annotated as *requirements*, *responsibilities*, *offer characteristics* and *contact details*, i.e., those chunks of information most commonly found in job offers.

Relevance Assignment. The final component of our extraction process assigns pre-defined relevance values, between 0 and 1, to the initial annotations, depending on the block category, the annotated token is contained in. E.g., in case that a token annotated as IT-skill is part of the requirements block it gets a relevance of 1, whereas if it is part of the bottom part, it gets a relevance of 0,25, only. During post-processing, annotated IT-skills with a relevance lower than a certain threshold, are eliminated from the result.

2.2 Rich Client Interface for Annotations

For ensuring an acceptable quality of the extraction results, information extraction out of arbitrarily structured and heterogeneous job offers is reasonable in a semi-automatic way only. Thus, the results generated automatically by the extraction system, which are in our case, annotations of the target Web pages containing the job offers, have to be assessed by a user and potentially corrected accordingly. For this task, we provide a rich client interface which allows the user not only to initiate the extraction process by entering the URL's to the desired target Web pages, but especially to visualize the results of the extraction process in terms of the annotated Web pages and the possibility to manipulate existing annotations or to add additional ones directly in a Web browser[2]. The screenshot in Figure 2 depicts on the left hand side a menu sidebar and on the right hand side the original Web page.

Annotation Highlighting. Within the sidebar, the user can choose from an *annotation list* (cf. ①), representing the concepts of our domain ontology, which of the automatically generated annotations should be indicated within the Web page on the right hand side. To preserve the look and feel of the original Web page, annotations are indicated by marking the concerned parts of the Web page with corresponding colors and highlighting those, being currently focused on, by means of a flash effect.

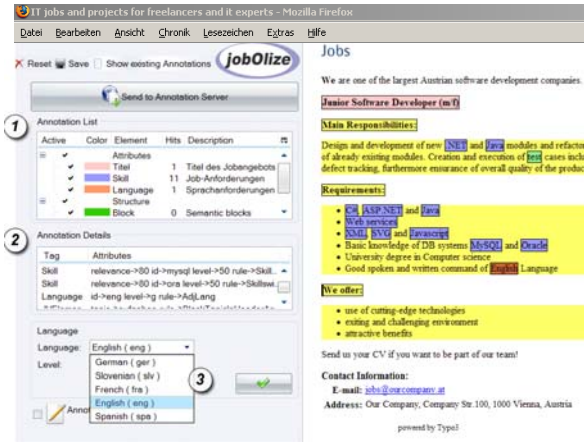

Figure 2. Rich Client Interface of JobOlize

[2] It has to be noted that the GUI provided by GATE is realized as a heavy-weight desktop application only, which among others, lack a proper visualization of the extraction results.

Annotation Details. The *details of annotations* (e.g., the relevance values assigned) are shown for each annotation, both, within the sidebar (cf. ②) and directly within the Web page, activated by a mouse over effect. This functionality also serves as a simple form of explanation component, which allows tracing back the genesis of a certain annotation.

Annotation Manipulation. Finally, the details of each of the automatically generated annotations can be *manipulated* in accordance with the underlying job ontology, the annotation can be completely *deleted* or new ones can be *added*. Thus, e.g., a language skill can be modified from German to English (cf. ③). Manipulations are immediately propagated to the Web page at the right hand side.

Implementation Aspects. Concerning implementation aspects, the rich client interface is realized as a *Firefox extension* using *XUL* (XML User Interface Language [26]), allowing for portability of the application, *JavaScript* and *CSS*. The primary reason to favor the Firefox extension mechanisms instead of alternatives like Apache's MyFaces [19], was the rather easy realization of a rich client interface experience as in native desktop applications, considerably reducing programming effort. Annotation highlighting as well as manipulation is based on DOM. As a pre-requisite for asynchronously initiating the extraction process (executed on the server) directly from within the rich client interface, the underlying extraction system had to be ported to the used Tomcat Web server.

3. Evaluation of JobOlize

This section is dedicated to an evaluation of JobOlize, first on basis of a case study and second by a comparison to related work.

3.1 Case Study

The goal of this case study is to evaluate the amount of improvement in extraction quality when employing our context-driven extraction process, compared to a simple ontology- and rule-based annotation (called "initial annotation" in Section 2.1).

Set-up of the Case Study. The test set of our case study consisted of 32 randomly chosen online IT job offers from 16 different Austrian recruitment Web sites, partly differing considerably in content, structure, and layout. The extraction quality has been tested for three different concepts of our job ontology, namely IT skill, operation area, and language skill, since each of them poses unique challenges to the extraction process. Finally, we employed the common metric used for evaluating the quality of information extraction, comprising three different measures [5]: First, *precision*, which reflects the share of extracted content which is relevant among *all extracted content*. Second, *recall*, which depicts the share of extracted content which is relevant among *all content*. Third, *F-measure*, which is a balanced measure, combining recall and precision. These measures were computed using the AnnotationDiff tool, provided by GATE [8].

Interpretation of Results. Figure 3 shows the results of the case study. Overall, it can be seen that our context-driven extraction process outperforms a non-context-driven extraction, as expected. The values for *recall* were equal in all three cases, simply due to the fact that the initial annotation determines all potentially relevant token candidates, while the subsequent components of our extraction pipeline do not further contribute to the recall values.

The most significant gain in extraction quality could be observed regarding the ontology concept *operation area*. Precision and F-measure showed an increase of 64% and 38%, respectively. The reason for this improvement is that, in several Web pages of the test set, operation areas also describe the business focus of the company, not only the profile required from a job seeker. This completely different intention of using operation areas can be detected by our context-driven extraction process, since operation areas of a company are naturally positioned in the top part of a Web page, whereas operation areas required from a job seeker occur within a dedicate block of the content part.

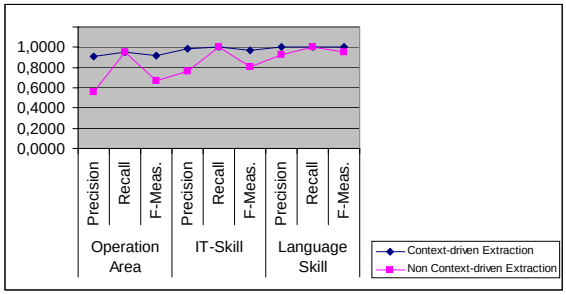

Figure 3. Results of the Evaluation

Precision and F-measure for *IT-skills* increased by 30% and 20%, respectively. The reason for this improvement is similar to that given for operation areas – there are job offers where pretended IT-skills show up at the bottom part of the Web page as already argued in Section 2.1. Our context-driven extraction process allows again eliminating these false positives, although there seems to be not that many false positive IT-skills than operations areas out there, as indicated by the lower improvement in extraction quality.

Finally, the least improvement could be observed for *language skills*, being nearly the same for precision (8%) and F-measure (6%). This rather marginal ad-

vance is due to the fact that language skills are more or less unambiguously used in job offers, seldom leading to a false positive content extraction.

3.2 Comparison to Related Work

There are already numerous information extraction techniques available [14]. Considering our rich client interface first, most existing systems build, as already mentioned, on desktop-based interfaces. The remaining ones, e.g., KIM [21] provide restricted functionality only, in that they visualize the annotated Web page only, but do not allow arbitrary manipulations.

Second, regarding the extraction itself, most existing techniques represent the targeted Web page in form of a DOM tree. On this basis, some of them just separate content from non-content parts (cf. e.g., [4]), often focusing on certain kinds of structure only, such as tables (cf., e.g.,[16]). Other approaches add additional context to improve extraction quality, either in terms of content information (cf. e.g., the "Pagelet" approach of [1]) or using structure (cf. the WISDOM approach of [13] and the template detection approach of [2]), or layout information (cf. e.g., the "Styletree" approach of [27] and the visual feature approach of [9]).

Compared to these systems, we approach the extraction process from a different angle, since we do not represent the input Web page as a DOM tree but let our extraction system directly work with the original Web page, thus preventing to be dependent on the decreased fidelity of DOM representations. Another major difference is that JobOlize does not focus on certain structures of the Web pages but rather deals with arbitrary structures, which is made possible by the combination of NLP techniques of the underlying GATE infrastructure and our context-driven extraction process. And finally, we do not consider a certain kind of context in isolation, but rather consecutively apply all kinds of context available, i.e., content, structure and presentation information to produce the resulting annotations and calculate corresponding relevance values.

4 Concluding Lessons Learned

The experiences gained in the course of developing our system JobOlize for the domain of e-recruitment are now critically reflected and generalized in order to provide valuable lessons learned also for other domains relying on information extraction techniques. According to the contributions of this paper, the lessons learned are grouped into three categories.

4.1 Context-Driven Information Extraction

Combining different kinds of context counteracts intricacies of HTML encodings. As one would expect, the liberty of users to ignore any accessibility or well-formedness criteria when designing HTML pages, using e.g., proprietary enumerations instead of explicit `<li>`-tags, different font sizes instead of explicit `<H>`-tags or HTML tables instead of CSS, considerably challenges the extraction process. It has been shown that the combination of different kinds of context as done in JobOlize eases the task to deal also with these intricacies of HTML encoding in a proper way.

Eliminating irrelevant parts of a web page considered harmful. One has to be aware that eliminating parts of a Web page which are presumably considered as irrelevant as done in JobOlize on basis of the LCS algorithm can also be counterproductive, e.g., in case that the eliminated part indeed contains links to further skills, self refreshing date items, or RSS-based newstickers. In addition, the accurateness of the used algorithm completely depends on a proper choice of the two Web pages used for initial comparison. When choosing two job offers with, e.g., similar skill demands, these blocks could, because of their overlap, be falsely classified as irrelevant.

4.2 Rich Client User Interface for Annotations

Firefox extensions and XUL proved to be adequate for rich client interfaces. Our experience of using Firefox extension mechanisms and XUL confirms that the development of a rich client interface is possible without much training and programming effort, being additionally facilitated by a comprehensive documentation and a steadily enlarging developer community. Nevertheless, as common in the Web Engineering area, one must expect permanent changes of the underlying SW-components which required, in our case, due to fundamental changes to the Firefox API from version 2 to 3, considerable re-coding.

Annotations partly conflicting with original HTML-tags. To visualize also the annotated Web page within the rich client interface, JobOlize currently merges the HTML-tags of the original page with the tags representing the annotations. There are, however, cases where this leads to irresolvable HTML validity conflicts, for example, if a title element overlaps an end-of-list HTML-tag. Currently, our rich client interface is not able to visualize annotations causing such conflicts, requiring more advanced visualization techniques such as an overlay concept.

4.3 Application of Extraction Systems

Guidelines how to apply information extraction systems required. Not least due to the high degree of extensibility provided by GATE, guidelines how to apply an extraction system depending on certain characteris-

tics of the domain are urgently needed. Especially the specification of extraction rules for JobOlize turned out as a trial and error process, getting completely lost in several rule dependencies which might be alleviated by a visual (pattern-based) rule-editor or by some kind of decision support system, applying further AI expertise to the area of information extraction.

Measures for evaluating the quality of information extraction needed. During the case study, it emerged that, e.g., for evaluating the skill levels, a simple Boolean measure in terms of true positives and false positives is not adequate. For example, instead of "perfect" for a skill level the slightly different value "very good" might have been extracted or instead of "Oracle" for a skill the more generalize concept "DBS" might have been extracted. These examples argue for a measure, which takes a similarity aspect into account, e.g., derived from relations defined within the ontology. These explorations confirm the argumentation of Lavelli et al. [15], that the field still lacks standard datasets, evaluation procedures and appropriate measures.

Acknowledgements

We kindly thank our colleague Simon Thalbauer for valuable contributions to the realization of Jobolize.

References

[1] Bar-Yossef, Z. and Rajagopalan, S. "Template detection via data mining and its applications". *In Proc. of the Int. World Wide Web Conf. (WWW)*, Honolulu, May 2002.

[2] Chakrabarti, D., Kumar, R., and Punera, K. "Page-level template detection via isotonic smoothing." *In Proc. of the 16th Int. World Wide Web Conf. (WWW)*, Alberta, May 2007.

[3] Cunningham, H., et al. "GATE: A Framework and Graphical Development Environment for Robust NLP Tools and Applications". *In Proc. of the 40th Anniversary Meeting of the Association for Comput. Linguistics (ACL),* Philadelphia, 2002.

[4] Debnath, S., Mitra, P. Pal, N. and Lee Giles, C. "Automatic Identification of Informative Sections of Web Pages", *IEEE Transactions on Knowledge and Data Enginering*, 17(09), Sept. 2005, pp. 1233-1246.

[5] Douthat, A., "The message understanding conference scoring software user's manual". *In Proc. of the 7th message Understanding Conference (MUC7)*, 1998

[6] Eckhardt, A., König, W., Weitzel, T. and Deininger, K. "Recruiting Trends 2007 – European Union – An empirical survey with the top 1.000-enterprises in the EU", No. 2007-798.

[7] Garcia-Sanchez, F. et al. "An ontology-based intelligent system for recruitment". In *Expert Systems with Applications*, 31(2), Aug. 2006, pp. 248-263.

[8] GATE User Guide, http://gate.ac.uk/sale (accessed May 2008)

[9] Gatterbauer, W., et al. "Towards domain-independent information extraction from web tables". *In Proc. of the 16th Int. World Wide Web Conf. (WWW),* Alberta, May 2007.

[10] HR XML, http://www.hr-xml.org (accessed May 2008)

[11] Hunt, J. W. and Szymanski, T. G. "A fast algorithm for computing longest common subsequences". *CACM 20(5)*, May 1977.

[12] JobScout24, http://www.jobscout24.de (accessed May 2008)

[13] Kao, H., Ho, J. and Chen, M. "WISDOM: Web Intrapage Informative Structure Mining Based on Document Object Model". *IEEE Transactions on Knowledge and Data Engineering*, 17(5), May, 2005, pp. 614-627.

[14] Kayed, M. and Shaalan, K. F. "A Survey of Web Information Extraction Systems". *IEEE Trans. on Knowledge and Data Engineering* 18(10), Oct. 2006.

[15] Lavelli, A., et al. "A critical survey of the methodology for IE Evaluation." *Int. Conf. on Lang. Resources & Evaluation, 2*004.

[16] Lin, S. and Ho, J. "Discovering informative content blocks from Web documents". In *Proc. of the 8th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining (KDD)*, Alberta, July, 2002, pp. 588-593.

[17] Mochol, M., Paslaru, E. and Simperl, B. "Practical Guidelines for Building Semantic eRecruitment Applications". *Proc. Of the Int. Conf. on Knowledge Mgmt. (iKnow)"*, Austria, Sept. 2006.

[18] Monster, http://www.monster.com (accessed May 2008)

[19] Myfaces, http://www.myfaces.org (accessed May 2008)

[20] Palkoska, J., et al. "On Cooperatively Creating Dynamic Ontologies". *In Proc. of Int. Conf. on Hypertext*, Salzburg, 2005.

[21] Popov, B., et al. "KIM - A Semantic Platform For Information Extraction and Retrieval". *In Journal of Natural Language Engineering*, 10(3-4), Sep 2004.

[22] Baumgartner, N., Retschitzegger, W. and Schwinger, W. "Lifting Metamodels to Ontologies – a Step to the Semantic Integration of Modeling Languages". *In Proc. of the 9th Int. Conf. on Model Driven Engineering Languages and Systems (MODELS)*. Springer, 2006.

[23] Baumgartner, N., Retschitzegger, W. and Schwinger, W. "A Software Architecture for Ontology-Driven Situation Awareness". *In Proc. of the 23rd ACM Symp. on Applied Comp. (ACM SAC)*, Fortaleza, March 2008.

[24] Stepstone, http://www.stepstone.com (accessed May 2008)

[25] Uren, V, et al. "Semantic Annotation for Knowledge Management: Requirements and a Survey of the State of the Art". *Web Semantics: Science, Services and Agents on the World Wide Web* 4(1), 2006, pp. 14-28.

[26] XUL, http://developer.mozilla.org (accessed May 2008).

[27] Yi, L., Liu, B., and Li, X. "Eliminating noisy information in Web pages for data mining". *In Proc. of the Int. Conf. on Knowledge Discovery & Data Mining (KDD)*, Aug., 2003.