Semantic Annotation of Web Pages Using Web Patterns

Milos Kudelka, Vaclav Snasel, Ondrej Lehecka,
Eyas El-Qawasmeh, Jaroslav Pokorny

Computer Science Dept., VSB – Technical University of Ostrava, Czech Republic
Computer Science Dept., Jordan University of Science and Technology, Irbid, Jordan
Department of Software Engineering, Charles University of Prague, Czech Republic

kudelka@inflex.cz, {ondrej.lehecka,vaclav.snasel}@vsb.cz,
eyas@just.edu.jo, pokorny@ksi.ms.mff.cuni.cz

Abstract. This paper introduces a novel method for semantic annotation of web pages. We perform semantic annotation with regard to unwritten and empirically proven agreement between users and web designers using web patterns. This method is based on extraction of patterns, which are characteristic for a particular domain. A pattern provides formalization of the agreement and allows assigning semantics to parts of web pages. We will introduce experiments with this method and show its benefits for querying the web.

1. Introduction

Semantic annotation of web pages concerns adding formal semantics (metadata, knowledge) to the web content for the purpose of more efficient access and management. Currently, the researchers are working on the development of fully automatic methods for semantic annotation (see, e.g., [2]).

For our research we consider semantic annotation and tracing user behaviour in the query-answering dialog: (1) to simplify querying; and (2) to improve relevance of answers. In the field of Internet search, we introduce a new perspective, which connects both goals in a native way. The key aspect of our perspective is smart focusing on the user and his expectations when searching information on the web. To be able to do this we need the user to share his expectation with us.

A simpler way is to turn our questions to professional web sites designers. Their primary mission is to fulfil user's expectation. A proof of this is that high-quality web pages and web solutions are widely accepted by users. Professional web designers apply practices, which come up from user's experiences. These practices relate with human sensation and allow simple orientation in supplied information. Solutions of the same problem are solved by different developers differently but at a certain level solutions are all the same. Similar web pages contain similar components. We can define this conformity such that there are similar web page components on the pages within the same application domain. These components are designated as web patterns.

Our method for semantic annotation of web pages is performed with regard to unwritten and empirically proven agreement between users and web designers. This method is based on extraction of patterns, which are characteristic for a particular domain. A pattern provides formalization of the agreement and allows assigning semantics to parts of web pages. We will introduce experiments with this method and show its benefits for querying the web.

Section 2 contains a short description of related works. Section 3 presents the patterns basics. Sections 4 and 5 present goals of our research. In Sections 6 and 7 we describe preparation of experiments and an analysis of results. Finally, Sections 8 and 9 focus on our future work and conclusions.

2. Related Work

There are two trends in the field of semantic analysis of the web today. One of them provides mechanism to semiautomatic (or manual) page annotation using ontology description languages and creation of semantic web documents [23], [10]. The second approach prefers an automatic annotation. In [3] there is a methodology based on a combination of information extraction, information integration, and machine learning techniques. The complex example of an automatic approach is the ambitious KIM project [15]. An application that performs automated semantic tagging of large corpora is described in [5]. It is based on the Seeker platform for large-scale text analysis. It marks up large numbers of pages with terms from a standard ontology.

Across all directions it is possible to see the use of ontology-based mechanisms, in the case of second approach along with knowledge bases. Our view on the problem of semantic analysis is in many cases similar to the presented techniques and tends to the automatic approach of semantic annotation. However our motivation of what and how to annotate is different. It seems to be on a higher abstraction level because we do not work with the semantics in meaning of the content of document, but more likely with the form of the document, which is related to the content and chosen domain.

The approach, which is similar to ours, is mentioned in [16]. It uses TXL language to analyze chosen parts of pages to obtain structured domain specific information (tourism domain). Other similar approaches are automatic transformation of arbitrary table-like structures into knowledge models [21], formalized methods of processing the format and content of tables to relevant reusable conceptual ontology [25] or domain-oriented approach to web data extraction based on a tree structure analysis [22]. Paper [1] presents techniques for supervised wrapper generation and automated web information extraction. The system generates wrappers, which translate relevant pieces of HTML pages into XML. The next similar approach to web data extraction is described in [18]. The principal component is a document schema (found in HTML code). Document schemata are patterns of structures embedded in documents.

On the side of query expression analysis and construction there are approaches, which are helpful for the user when formulating his requirement. One of the possible approaches is to use a cognitive search model [27]. There is a web search system prototype based on ontology that uses a cognitive model of the process of human

information acquisition. Another way is to help user with specification of query based on interaction with user using genetic algorithms and fuzzy logic, e.g., [4], [17], [11].

There is interesting conjunction with paper [12] whose authors analyze web pages focusing on web site patterns. In three time intervals authors observed how web designers have changed web design practices. They also realized that content of web pages remains the same whereas form is being developed so it better fulfils user's expectation. Our work confirms results mentioned in the paper. Important for us are such web pattern characteristics that are independent of a web page design.

3. Patterns Basics

According to [24] patterns are structural and behavioural features that improve the applicability of software architecture, a user interface, a web site or something another in some domain. They make things more usable and easier to understand.

GUI patterns supply a solution of typical problems within design of user interface. Typical examples are organization of user controls into lists or tabs and so on. GUI patterns describe on a general level how to make structure of information within user interface. They also tell which components to use, how they should work together and how to work with them (see [24], [9] for examples).

In [24] there is a set of idioms which give us a general answer to question which types of user interfaces we can come across (Forms, Text editors, Graphic editors, Spreadsheets, Browsers, Calendars, Media players, Information graphics, Immersive games, web pages, Social spaces, E-commerce sites). For our purposes we focused on Calendars, Web pages, Social spaces, and mainly on E-commerce sites.

In this paper we have chosen selling products domain for our experiments. Patterns identified for other domains are, e.g., in [26]. We can find common features in user interface within the selling products domain. These features express typical tasks with information (showing the price information, purchasing possibility, and the product detail information). When implementing a web site the web designers proceed the same way. They also use patterns even if they do not call them so (see [6]). During the analysis of selling products domain and even our experiments we worked with a ten of patterns, which come out from [6]. Figure 1 demonstrates an example where there is a cut of a web page with selling of product on eBay.com.

GUI and domain patterns are designated for web designers and domain experts. They are written with free text whereas structure of their description is formalized. For our purposes we do need to find a description of patterns which will be independent of the web designers and implementation and which will be useful for a semantic analysis of web pages.

Patterns on the page represent, in certain degree, what the user can expect (as a consequent of agreement between web designers and users). For us this is the key pre-requisite for technical usage of patterns. Our algorithms are able to determine whether pattern is on the page or not and as a consequence of this we can annotate this page and use this annotation next time. We have to also answer the question what can this bring to the user.

Fig. 1. A web page with marked patterns. The patterns found are graphically marked on the page: *Sign on possibility*, *Price information*, *Purchase possibility*, and *Rating*.

Obviously, there is a problem in formalization of this approach. Patterns on the page do not appear in exact form. A crucial feature of presence of the pattern on the page is that individual elements of the pattern appear more or less together. More formal description of this deduction is described again in [24], [19]. The visual systems usually implement so-called Gestalt principles:

- *Proximity* – related information tends to be close to each other.
- *Similarity* – similarly looking elements contain similar information.
- *Continuity* – the layout of the information is continuous.
- *Closure* – related information tends to be enclosed.

So we can suppose a web pattern as a group of characteristic technical elements (which are based on GUI patterns) and a group of domain specific elements for the domain we are involved in (typical keywords related to given pattern and other entities such as the price, date, percent, etc.).

Example. On the right bottom of Figure 1 it is clearly visible that there is instance of *Rating* pattern.

- For technical implementation of this item it is used *div layout* which is useful for good information structuring.
- In the pattern instance we can find characteristic words – *seller*, *feedback*, *score*, *positive*.
- Moreover we can find integer, date and percentage data type instances – *17*, *90.5%* and *Oct-20-04*.

4. Query Simplifying

Patterns should give us an understandable language, which we are able to use in communication with the user to settle what he expects on the page (see Figure 2). Using search engines he has to think about a set of key words, e.g. “price”, which he uses to specify his requirement. The pattern *Price information* contains much stronger information that “price” occurring on the page.

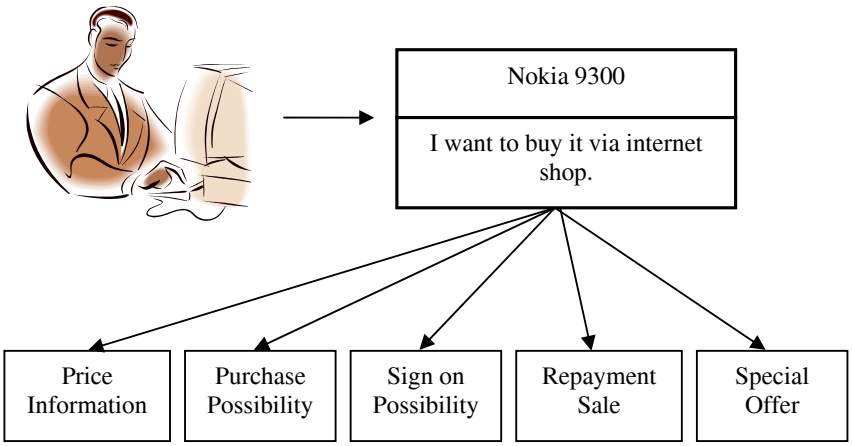

Fig. 2. Query from selling product domain. With the patterns we can set up catalogues and profiles which simplify the selection process for the user. It will remain to the user that he will have to enter subject of search but patterns will help him to specify expectations.

5. Improvement of Answer Relevance

When annotating pages we are working with the same information, which the user uses to make query. Moreover this information has stronger semantic content than, e.g., enumeration of keywords. Assume that we have annotated pages in a database with regard to patterns so there is information about which patterns are contained on each page. We can then use this information in two manners:

1. When showing web search engine results, for every shown page link in the returned set of links, we can add information about which patterns have been founded on the page. The user can recognize on the first look whether the page will fulfil his expectation.
2. We can count on the patterns already when performing search and sort page links with regard to weights of required patterns we have found on the page.

A side-effect in this situation is that there will be preferred those pages which have been created by high-quality web designers using recommended techniques described in patterns. Since patterns describe widely accepted solutions to users, a user will be given high-quality designed pages earlier in selection, i.e. on the first positions of the result set of page links.

6. Experiments Preparation

The key aspect of the pattern manifestation is that the introduced elements are close to each other. We can focus on the structure of the page during page analysis. It is not necessary to provide the deep analysis of page structure, because the technical elements provide just the environment for keeping the information together.

6.1. Choosing Domain and Domain Patterns

We consider the domain of e-commerce for testing purposes. Our goal was to integrate common users to testing so we have chosen one of the most used domains ever. During the domain analysis phase we sorted out nine domain web patterns – *Sign on possibility*, *Price information*, *Purchase possibility*, *Special Offer*, *Annuity selling*, *Product details*, *Comments and reviews*, *Discussion and FAQ*, *Advertising*. They are the semantic elements, whose are commonly expected by the user along the identification of the product.

6.2. Pattern Dictionary Preparation

For every one of the introduced web patterns we have manually chosen a group of words, which occurrence is characteristic for pattern on the web page. The words have been put into the database and serve as input for algorithms performing analysis of web pages. The set of chosen words needs to be understood as a starting set, which is automatically expanded according to deeper analysis of the pattern (we measure a frequency of words inside text segments of found patterns – the similar approach is presented in [3], [14]). Every pattern has its own dictionary of words.

We chose the words that are usually used by users when querying in the search engines – *price*, *eur*, *usd*, *offer*, *discount*, *stock*, *basket*, *buy*, *shop*, *review*, *forum*, *for sell*, *specifications*, … We have assigned more than five words to each pattern preferring most frequently used words in various associations.

These words can have multiple meanings and usually only one of the meanings is meant to represent a pattern. It is very important that the words in dictionaries are domain associated. We can then expect that [13]:

- The dictionary is not too large.
- The words occur in certain schemata.
- The meaning of words is more or less unambiguous.
- The words appear frequently in the text.

6.3. Patterns Extraction

The key task for solution of the problem we are dealing with is to find the mechanism of pattern detection on the page. We had to develop algorithms working with content of web page. They try to answer the questions about a weight of pattern on the page. In semantic analysis we need to find characteristics, which are

- dependent on the meaning of what a pattern represents for user, and
- independent of pattern implementation.

In our experiments we simplified pattern formalization problem using a set of words and data entities (e.g. date or percent), which are characteristic for the pattern. We can choose one concrete pattern (e.g. *Discussion*) and make analysis of big amount of web pages with discussions. After the analysis we can find out that there is quite small group of word and data entities by which it is possible to recognize the pattern. So if we suppose that we know the terminology for discussions (terms like *discussion*, *author*, *re*, etc.), then we can find segments in the plain text of the web page where the terms occur.

Let E be a set containing all *entities* they are characteristic for a given pattern (*pattern dictionary* – keywords and data types). On power set $P(E)$ we can define a binary relation δ so that pair (E,δ) makes a proximity space (see, e.g., [20]). Proximity space is used as the closeness model for groups of pattern entities. This defined structure is used as instrument for description and finding of page *text segments* S_i, which can (or does not have to) be a part of the given pattern. Let I be an instance of the given pattern. Then $I = \{S_1, ..., S_m\}$, where $m > 0$ and $S_i \in P(E)$.

A *pattern instance* is a set of analyzed text segments, which contain pattern entities (we do not focus on meaning of group of words but only on their presence). For discovery of algorithms that can be useful for finding and analyzing selected segments the Gestalt principles can do us a good turn. We developed methods for

- *proximity*: how to measure closeness (distance) between entities in searched text segments. We work with tree organization of entities representing a text segment and we suppose that in searched text segment entities must be close enough to each other (we have designated the distance based on analysis of text segments in searched pages).
- *similarity*: for measuring similarity of two searched text segments (for *Discussion* we are able to identify repetition of replies). We work with comparison of trees representing text segments.
- *continuity*: how to find out whether two or more found text segments make together instance of a pattern. We assume that two or more similar text segments (trees of entities from one pattern) match together.
- *closure*: for computation of the weight of one single searched text segment. In principle, we used two criteria. We rated shape of the segment tree (particularly, ratio of height and entity count) and quantity of all words and paragraphs in the text segment. On the overall computation of the weight also the proximity rate participates.

With complex usage of all mentioned principles we have implemented an algorithm, which offers excellent results in pattern extraction. The algorithm uses only plain text and regardless of the fact it is successful in more than 80% of cases.

6.4. Algorithm

Our algorithm is built on application of Gestalt principles. Before the algorithm takes place the page HTML code is preprocessed – the plain text (sequence of words) is extracted and data type instances (data entity) are found. The data type instances and sequence of words make list of page entities. Pattern dictionary is composed of characteristic words and expected data types (pattern entities).

Input for the algorithm is both set of entities, which represents each word, and data entity from the text of a web page and set of characteristic pattern entities. The algorithm compares these entities with characteristic pattern entities and creates representation of text segments called snippets [7], which can belong to (or compose) a pattern. These representations are then used for further computations and the result is value representing the weight of pattern occurrence on the page.

```
FOREACH page entity in all page entities
   IF page entity is pattern entity THEN
      IF not exist snippet to add page entity to THEN
         create new snippet in list of snippets
      ENDIF
      add page entity to snippet
   ENDIF
ENDFOR
FOREACH snippet in list of snippets
   compute proximity of snippet
   compute closure of snippet
   compute value(proximity, closure) of snippet
   IF value is not good enough THEN
      remove snippet from list of snippets
   ENDIF
ENDFOR
compute similarity of list of snippets
compute continuity of list of snippets
compute value(similarity, continuity) of pattern
RETURN value
```

Example: Let's go back to Figure 1 and instance of *Rating* pattern. After preprocessing of page HTML code there is only list of page entities (pattern entities are emphasized with bold font):

seller *information* <*par*> ***feedback score <num>*** <*par*> ***positive feedback <perc>*** <*par*> *member since* ***<date>*** *in united states* <*par*> *read* ***feedback*** *comments* <*par*> *add to favourite sellers* <*par*> *view* ***seller*** *other items*

After extraction of segments and evaluation of *proximity* and *closure* criteria there are only two remaining segments which are potential parts of pattern (in the case that there is more segments found they will be taken into account).

seller *information* <*par*> ***feedback score <num>***

positive feedback <perc> <*par*> *member since* ***<date>***

The evaluation of *similarity* and *continuity* criteria is performed on the found segments. In our example the two segments are not considered similar but supplementary to each other. In our example the computed final probability of pattern presence is higher than 80%.

6.5. Experimental Application

To test our approach we have implemented web application, which uses Google Web API. The system queries Google for a set of few tens of pages, which correspond to the user's conditions. Then the application downloads the pages and extracts plain text from the HTML code. On the plain text there are performed analysis, patterns extraction, and evaluation of pattern's weights. Then the page is evaluated as a whole. Pages are then sorted according to the overall computed value.

We tested our extract algorithms on PC Intel Pentium M 1.6 GHz with installed Windows XP. We extracted 9 patterns on each page. The performance of algorithms was 100 pages in about 1 second. The average time of 1 pattern extraction was approximately 10^{-3} seconds.

7. Result Analysis

There were more than 200 searches of products tested (cellular phones, computers, components and peripheries, electronics, sport equipment, cosmetics, books, CDs, DVDs, etc.) in four profiles. We usually worked with first thirty of found pages; for dozens of products we tested the set of first one hundred found pages. For wide testing we selected a miscellaneous group of people, e.g. students of different types of schools and also people dealing with product selling on the Internet.

During experiments we collected 31,738 various web pages that we got from the Google search engine using queries on products. After the analysis we discovered that on the 11,038 web pages there was not any extracted pattern even though our queries were focused on pages containing these patterns (queries from our application contained groups of elaborative words). Even though we must count with queries on which it is not possible to find enough relevant answers and also with inaccuracy of our algorithms it has to be noted one interesting thing. In spite of very precise query to searching engines the user has to count with approximately one quarter up to one third of irrelevant web pages, which do not contain expected information.

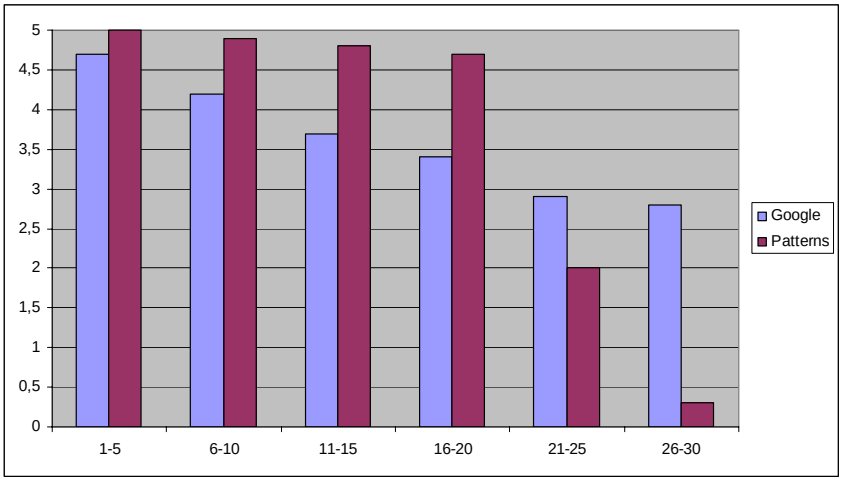

Fig. 3. First 30 pages (default and patterns sorted). On the horizontal axis, there are the first 30 found pages in groups of 5. On the vertical axis there is the number of relevant pages. The left columns show figures using Google search engine. Right columns show figures using our experimental application.

Figure 3 shows query results on concrete products. It includes only those queries on which there is multiply times more relevant pages than our first 30 analyzed web pages from search engine (it includes more than 200 various queries). We can observe that

- irrelevant web pages are moved to the end of the result set (our approach eliminates mistakes in order of pages),
- using our ranking the user reaches the expected pages earlier.

8. Future Work

Ability of inserting other web patterns into our system is opened. We are working on extraction of other web patterns from selling products domain (like *Rating*, *Ask the Seller*). It is also expected that the system will be extended with other domains in which it is able to identify web patterns. We are preparing *Tourism* and *Culture* domains.

During the query string analysis phase it is possible to recognize words which exists in our pattern dictionaries. Using this we are able to identify groups of patterns (called query profile) which user probably expects. We plan to use this observation during the result set links sorting and also during composing the user query expression.

We can see that patterns don't occur alone on pages but in certain groups. It means that there are page groups or clusters, which are characterized by the same web

patterns. We can suppose such group of web patterns as a page profile. For searching page profiles we plan to use clustering methods.

9. Conclusion

The crucial aspect of our approach is that we do not need to analyze page's HTML code. Our algorithms are based on analysis of plain text of the page. For page evaluation we do not use any meta-information about page (such as title, hyperlinks, meta-tags, etc.). We also confirmed that key characteristics of web patterns are independent of language environment. We tested our method in English and Czech language environment. The only thing we had to do was to change patterns dictionaries.

Our experiments show that it is very useful to consider gained data about pattern existence as a metadata stored along with the page. So now we have tools, which are able to discover whether the page contains certain pattern with about 80% accuracy.

Our approach is not universal. The reason is that its basic assumption is a domain with relatively formed rules of how web pages look like (we do not expect uniformity but we expect some synchronization between users and web pages designers).

Acknowledgment: This research was supported in part by GACR grant 201/04/2102 and the National programme of research (Information society project 1ET100300419).

References

1. Baumgartner, R., Flesca, S., Gottlob, G.: Visual Web Information Extraction with Lixto. In: Proc. of the 27th Int. Conference on Very Large Data Bases. (2001) 119–128
2. Chakrabarti, S.: Mining the Web: Discovering Knowledge from Hypertext Data. Morgan Kaufman Publishers (2003)
3. Ciravegna, F., Chapman, S., Dingli, A., Wilks, Y.: Learning to Harvest Information for the Semantic Web. ESWS 2004, LNCS 3053. Springer-Verlag Berlin Heidelberg (2004) 312–326
4. Cordón, O., Moya, F., Zarco, C.: Fuzzy Logic and Multiobjective Evolutionary Algorithms as Soft Computing Tools for Persistent Query Learning, in Text Retrieval Environments. IEEE Int. Conf. on Fuzzy Systems (FUZZ-IEEE 2004), Budapest (Hungary) (2004) 571-576
5. Dill, S., Eiron, N., Gibson, D., Gruhl, D., Guha, R., Jhingran, A., Kanungo, T., McCurley, K. S., Rajagopalan, S., Tomkins, A., Tomlin, J.A., Zien, J. Y.: A Case for Automated Large-Scale Semantic Annotation. Journal of Web Semantics, 1(1) (2003) 115-132
6. Van Duyne D. K., Landay J. A., Hong J. I.: The Design of Sites: Patterns, Principles, and Processes for Crafting a Customer-Centered Web Experience. Pearson Education (2002)
7. Ferragin, P., & Gulli, A. (2005). A personalized search engine based on Web-snippet hierarchical clustering. In: Proc. of 14th Int. Conf. on World Wide Web, Chiba, Japan. (2005) 801-810
8. Gamma, E., Helm, R., Johnson, R., Vlissides, J.: Design Patterns – Elements of Reusable Object-Oriented Software. Addison-Wesley (1995)

9. Graham, I.: A pattern language for web usability. Addison-Wesley (2003)
10. Handschuh, S., Staab, S., Ciravegna, F.: S-CREAM Semi-automatic CREAtion of Metadata. The 13th Int. Conf. on Knowledge Engineering and Management (EKAW2002), ed. Gomez-Perez, A., Springer Verlag (2002)
11. Husek, D., Owais S., Kromer, P., Snasel V., Neruda, R.: Implementing GP on Optimizing both Boolean and Extended Boolean Queries in IR and Fuzzy IR systems with Respect to the Users Profiles. 2006 IEEE World Congress on Computational Intelligence, CEC (2006) accepted.
12. Ivory, M. Y., Megraw, R.: Evolution of Web Site Design Patterns. ACM Transactions on Information Systems, Vol. 23, No. 4 (2005) 463–497.
13. Jianming Li, L. Z. Yu, Y.: Learning to generate semantic annotation for domain specific sentences. In: Knowledge Markup And Semantic Annotation Workshop in K-CAP 2001, (2001)
14. Karov, Y., Edelman, S.: Similarity-based Word Sense Disambiguation. Computational Linguistics 24(1) (1998) 41-59
15. Kiryakov, K., Popov, B., Ognyanoff, D., Manov, D., Kirilov, A., Goranov, M.: Semantic Annotation, Indexing, and Retrieval. ISWC 2003, LNCS 2870. Springer-Verlag Berlin Heidelberg (2003) 484–499
16. Kiyavitskaya, N., Zeni, N., Cordy, J. R., Mich, L., Mylopoulos, J.: Semantic Annotation as Design Recovery. In: Proc. of ISWC 2005, 4th Int. Semantic Web Conf., Galway, Ireland (2005)
17. Kraft, D. H., Petry, F. E., Buckles, B. P., Sadasivan, T.: Genetic Algorithms for Query Optimization in Information Retrieval: Relevance Feedback. In Sanchez, E., Shibata, T., and Zadeh, L.A. (eds.), Genetic Algorithms and Fuzzy Logic Systems, Singapore: World Scientific (1997)
18. Li, Z., Ng, W. K., Sun, A.: Web data extraction based on structural similarity. Knowl. Inf. Syst. 8(4) (2005) 438-461
19. Mullet, K., Sano, D.: Designing visual interfaces: Communication oriented techniques. Englewood Cliffs, NJ. Prentice Hall (1994)
20. Naimpally, S. A., Warrack, B. D.: Proximity Spaces. Cambridge University Press, Cambridge (1970)
21. Pivk, A.: Automatic Ontology Generation from Web Tabular Structures. PhD thesis, University of Maribor (2005)
22. Reis, D. C., Golgher, P. B., Silva, A.S., Laender, A. F.: Automatic web news extraction using tree edit distance. In: WWW '04: Proc. of the 13th Int. Conf. on World Wide Web. 502-511, New York, NY, USA. ACM Press (2004)
23. Sean, L., Lee, S., Rager, D., and Handler, J.: Ontology-based web agents. In: Proc. of the First Int. Conf. on Autonomous Agents (Agents'97) USA. ACM Press (1997) 59-68
24. Tidwell, J.: Designing Interfaces: Patterns for Effective Interaction Design. O'Reilly Media, Inc. (2006)
25. Tijerino, Y. A., Embley, D. W., Lonsdale, D. W., Ding, Y., Nagy, G.: Towards Ontology Generation from Tables. World Wide Web 8(3) (2005) 261-285
26. Wellhausen, T.: User Interface Design for Searching. A Pattern Language. http://www.tim-wellhausen.de/papers/UIForSearching/UIForSearching.html (May 29, 2005)
27. Wechsler, K., Baier, J., Nussbaum, M., Baeza-Yates, R.: Semantic Search in the WWW supported by a Cognitive Model. In: Int. Conf. Web-Age Information Management, LNCS, Springer, Dalian, China (2004)