XML-IR Users and Use Cases

Andrew Trotman[1], Nils Pharo[2], Miro Lehtonen[3]

[1]Department of Computer Science, University of Otago, Dunedin, New Zealand
andrew@cs.otago.ac.nz
[2]Faculty of Journalism, Library and Information Science, Oslo University College, Norway
nils.pharo@jbi.hio.no
[3]Department of Computer Science, University of Helsinki, Finland
miro.lehtonen@cs.helsinki.fi

Abstract. We examine the INEX *ad hoc* search tasks and ask if (or not) it is possible to identify any existing commercial use of the task. In each of the tasks: thorough, focused, relevant in context, and best in context, such uses are found. Commercial use of CO and CAS queries are also found. Finally we present abstract use cases of each *ad hoc* task. Our finding is that XML-IR, or at least parallels in other semi-structured formats, is in use and has been for many years.

1. Introduction

Many of the teething troubles in setting up realistic XML-IR experiments have been blamed on the lack of user grounding. Trotman [19] suggests that the standard methodology for XML-IR experiments should be based on examining user behavior and that it is essential to identify such a group of users. Unfortunately, he stops there and does not identify a single user.

At INEX 2006 Trotman and Pharo ran a thought experiment called the Use Case Studies Track [22]. They asked participants to hypothesize users of XML-IR and to imagine how such users would interact with an information system. They assumed such users did not exist and would not exist for several years.

Almost immediately Dopichaj [7] identified the use of XML-IR in book search and separately Lehtonen [13] identified commercial systems dating back to several years before the first INEX workshop. In the words of Lehtonen "Rather than trying to find the users, we are tempted to ask an even more interesting question: How did we lose sight of the users of XML retrieval?".

In this contribution we examine the existing INEX *ad hoc* tasks and demonstrate that for each track there is already a commercial use – they are already grounded in a user base. We take those search engines identified in the studies of Dopichaj and of Lehtonen, and connect the methodology of INEX. We then present hypothetical use cases for each INEX task,

Ironically, just as Dopichaj and Lehtonen were demonstrating the long-term prior existence of XML-IR users, Trotman was suggesting that users might prefer passages to elements [20] and emphasizing the importance of Passage Retrieval. We do not believe that the method of identifying the semantic unit is of material value in the case studies herein – that is, we do not believe the user is concerned with whether they are presented with a series of paragraphs or an element.

2. INEX *ad hoc* tasks

INEX initially identified only one *ad hoc* task, the thorough task. Subsequent rounds added the focused task, then the fetch & browse task (which became the relevant in context task). Finally the best in context task was added in 2006. The best in context task (also known as the best entry point (BEP) task) was one of the tasks studied in the Focus experiments [12] prior to the first INEX, and the task for which it could be easiest for the established search engines to adopt on the web.

2.1 Thorough Retrieval

The purpose of the thorough task is to identify all relevant elements in the document collection and to rank those relative to each other. That is, regardless of nesting, and regardless of the document in which the element has been identified, the task is to rank all elements with respect to topical relevance.

At the beginning of INEX 2006 there was no absolute certainty of the credibility of the task to a user community. It is clearly stated that “there are no display-related assumptions nor user-related assumptions underlying the task” [5]. However, as soon as these assumptions are made, the task may reduce to finding BEPs or to focused retrieval.

The task continues because it is considered important by some participants as a benchmark for improvements [20]. It is the only task that has been held at every INEX. It is also considered by some to be a system oriented approach to XML element ranking: from the thorough set of relevant elements, a set of disjoint elements (for focused retrieval) could be chosen, or a set of good entry points into a document could be identified.

2.2 Focused Retrieval

An endeavor to create a more user-centered task resulted in a task in which overlapping elements were forbidden from occurring in the result list. Evidence from the INEX Interactive track supported the view that users did not want to see the same text repeated in their results lists [18]. Adding the exclusion of overlap from the results list added a new problem to the search strategy, the identification of the *just right* sized element to return to the user.

At INEX 2006, and for the focused task, a hypothetical user of an XML-IR system was described. This user views the results list top-down, they prefer smaller elements to larger ones, and they are mostly concerned with elements ranked highly in the results list [5], they also do not want to see the same information returned multiple times.

The focused task has been criticized for being context-free [16] – that is, the user is presented with the results of their search but cannot evaluate the authority of the information because no context is given[1]. As an example, a section of an academic paper might fulfill the user's information need, but without knowing who wrote the paper (and were it was published) the user cannot be certain of the accuracy of the content. The context is important, and is needed.

2.3 Relevant In Context (Fetch & Browse)

In the fetch & browse task the search engine must first identify relevant documents and rank these relative to each other and with respect to topical relevance. Within this ranking, those relevant elements within the document are identified. The subtle difference between relevant in context and fetch & browse is that the former is focused whereas the latter is thorough.

At INEX 2006 a real user was imagined. This user considers the document to be the natural unit of retrieval, but wants the relevant elements within the document identified for quick perusal and browsing [5] – perhaps by subtle highlighting.

2.4 Best in Context

Considering how an information system using the relevant in context strategy might be implemented leads to the final task. The user performs a search, chooses a result from the results lists, and expects to be taken to the relevant passage within the chosen document. Again, at INEX 2006 a real user was theorized [5].

This task was previously investigated in the Focus experiments [12]. There they asked assessors to identify the best entry points from which a reader should start reading in order to satisfy their information need.

2.5 Theoretical Tasks

Since the first INEX workshop tremendous effort has been made to identify a realistic task for XML-IR, and it is hard to argue such a task has been found. We believe that relevant in context would be of enormous interest if a web search engine provider chose to implement similar technology for HTML on their cached results. Best in context might be easily implemented in a search engine that considered anchors inside a web page as well as whole web pages. Alternatively, if and when semantically and structurally stronger XML-languages are widely adopted in place of HTML, such technology would be valuable. Although XHTML is based on XML rather than SGML, the web still is based on quite a weak markup language, and sadly search engines do not identify relevant parts of documents.

This leaves a gap. INEX has identified four tasks; three are believed to be user driven. But, are these tasks purely theoretical or are there any users?

[1] The counter argument is that focused retrieval is about finding elements and not about displaying elements, and as such there is no context from which to be free.

3. INEX *ad hoc* Queries

It is poignant to ask how a user might phrase their query before even being presented with their search results – this is a question that has also received much attention (see, for example, van Zwol [25]). We believe it is likely to remain a topic of debate as different users are likely to want to interact with an XML search engine in different ways. A database administrator might prefer SQL to interact with a given database whose users prefer a point and click interface. Similarly, an administrator of an XML information system might prefer XPath [4] while a user of said system might prefer a keyword interface.

INEX identifies two types of queries, those that contain Content And Structural hints (CAS queries) and those that contain Content Only (CO queries) words. The former are specified in the INEX language NEXI [23], and the latter are simple keyword lists.

Several other methods of specifying queries were tested at INEX (see Trotman [24] for details), but it is the loose semantics and the simplicity that make NEXI an appealing language for XML-IR.

3.1 Content Only then Content And Structure

Again at INEX we see a hypothetical user. This user does not know (or might not want to use) the structure of the documents in the collection. This user issues a query containing only keywords and is returned a set of elements (ordered according to the task).

Just as a user of a web search engine might refine their query by adding keywords, the hypothetical user adds not keywords but structural constraints to refine their query. Of course, they might choose to include the constraints from the beginning if they suspected it would help.

3.2 Other models

Van Zwol examined a graphical user interface for choosing structural constraints for CAS queries [25]. In his interface the user selects structures from lists and is able to specify keywords to be constrained to those structures. These graphical queries are easily converted to CAS queries in the NEXI language. Baeza-Yates [1] used block-like graphical language to represent the nesting of structures. Query by fragment, the specification of XML fragments containing keywords was examined by Carmel *et al*. [3].

3.2 Theoretical queries

Comparative studies of the methods of asking the queries are hard to do in an environment in which there are no known users. What can be done, however, is to examine the requirements of the query language. It should optionally allow the user to specify structure, it should optionally allow the user to specify keywords, and it should optionally allow the user to specify the granularity (or target elements) of the search result that is, the following kinds of queries:

1. Keywords only (no structural constraints or target elements specified)
2. Keywords limited to structures (no target elements specified)
3. Keywords limited to structures with target elements specified
4. Structural constraints and target elements (no keywords)
5. Structural constraints (no keywords or target elements)
6. Target elements (no keywords or structural constraints)

The last three of which (typically) require no interpretation of the user's information need and are so outside the realm of the INEX *ad hoc* task. The term *keyword* refers to single keywords as well as phrases of several words or any other string-valued search condition.

User interfaces for queries need to be limited to text-only interfaces. In the preceding section graphical query languages and list-box query interfaces were discussed. Such interfaces do not alter the kinds of queries that can be asked, only how they are asked. We believe there is room for investigation of novel query interfaces for XML, a field in its infancy.

3.4 Result Granularity

Two models for specifying result granularity exist within INEX, either it is specified explicitly (or vaguely in a CAS query) or it is left to the search engine to determine (in a CO query or a CAS query targeting //*). Previous justification of the vague interpretation has been based on the user not knowing the DTD, or alternatively not having an expectation of the best size of a result to answer their questions.

The best size of a result may be dictated not only by the user but also by equipment used to render the information. Some web sites provide different interfaces depending on the surfing device being used. Yahoo, for example, provides an interface for mobile phones as well as desktop computers. The search engine could take advantage of the user context as part of the search process. In the case of XML-IR the search engine could interrogate the user's display characteristics and specifically target elements that are of a granularity to fit the device. In the case of a desktop computer this might be a section of a book whereas on a palmtop it might be several consecutive paragraphs, but on a mobile phone only a single paragraph.

4. Existing Search Models

In this section some search engines already identified as being (or hypothesized as being) XML-IR are examined. In some cases it is not possible to state with any certainty that XML is used as they are proprietary and vendors are not at liberty to disclose this information.

4.1 Version Control (Thorough)

To find an example of thorough retrieval it is necessary to find a collection of documents in which both the whole document and the document elements are atomic. That is, the document must be a loosely coupled collection of elements. [16]. Two such genre have been identified: books (discussed below); and version control.

A version control system such as RCS allows the user to manage individual source code files and also to manage whole software development projects. In some systems the user is able to search through the comments associated with a file, with a directory of files, or with a project.

In the Microsoft version control system SourceSafe, for example, the user can add comments to (label) a file, or a directory. Any comments associated with the directory also apply to any subdirectories and files of that directory. A directory label will be seen when a file's comments are examined. These comments could be (but are likely not) stored in an XML file in which the structure of the XML matches the structure of the file system.

A user searching these comments expects the version control system to identify each and every relevant comment, regardless of where in the structure that comment occurs. If a comment is given to a file, and to a directory then it is reasonable to expect it to occur multiple times in the results list. If the comment applies to a directory then the user expects to see the comment in a results list only once. In other words, the search engine should be thorough. In a version control system the context is implicit (the given archive) and so it is not needed.

4.2 News Summarization (Focused)

The awkwardness in finding an application of focused retrieval is that the context is not presented. As with thorough retrieval, if the context is implicit or otherwise already known, then context is not needed. The credibility of news stories published by BBC news, for example, can be taken for granted. The BBC news reader need not concern themselves with the authorship as the editor has already ensured the credibility of the content.

Not needing to know the greater context of a document does not mean elements are meaningful outside the context of the body text. To find an application of this aspect of focused retrieval it is necessary to look beyond the user, and at an information system as a whole. Such a system would present the user with information and not documents. News summarization is one example.

The Columbia Newsblaster summarizes news pages from multiple sources, often in HTML. To do this it must extract the text of the document from the decoration. Evans *et al*. [8] describe the elaborate mechanism that is used to do so. Focused retrieval could be used to separate the relevant text from the decoration.

Multi-document text summarizers such as MultiGen [15] are used by Newsblaster to summarize a collection of documents on a single topic. Focused retrieval could be used to identify parts of documents for summarization. Such an approach might result in an increase of summarization performance as there could be an associated reduction in non-relevant material being summarized.

4.3 Question Answering (Focused)

Question Answering has been examined at TREC, CLEF and NTCIR, but not yet at INEX. In this case the search engine must identify a very short (certainly less than a paragraph) snippet from a document that answers a user question – a focused retrieval strategy. Although it is unusual to see sentence and entity level markup in a document, applications that insert them do exist (for example, POS taggers).

4.4 Book Search (Relevant in Context or Fetch & Browse)

Dopichaj [7] identifies book search as a commercial (and in-use) application of XML retrieval. Specifically he identifies Books24x7 and Safari as successful Internet sites that rely on this (or very similar) technology. His examples could not have been further from expectation; they use a thorough retrieval strategy.

Books24x7 provides an interface in which books are first ranked by topical relevance, and then within that, parts of books are ranked. Dopichaj provides an example where (within a single book) both a chapter and a section of that chapter are listed. This interface is relevant in context but results are thorough; it is the Fetch & Browse strategy. It is not clear why Fetch & Browse was dropped from INEX, but perhaps it should be reinstated.

The granularity of search on Books24x7 is not known. It is possible that only three levels of the document tree are searchable: book, chapter, and section. Some INEX participants [21] take the view that many of the common elements (such as typographically marked elements or section headings) seen in the document tree are of no material value for retrieval and should not even be indexed. Others [14] take the view that the best elements for retrieval can be pre-determined and so only index a small subset of the elements in a document. Of the latter group, some [14] build a separate index for each element type and merge after searching each.

When examining the table-of-contents of a single book on Books24x7, chapters of that book which are relevant to the user's needs are marked as being so. By extension, if the relevance of each chapter (and section of each chapter) were also indicated then the table of contents would be a heat map of relevance across the document.

We believe that the document heat map is an important grounding for XML-IR. At each point in the document a relevance score is computed and this is displayed either alongside the document or with a table-of-contents like overview of the document. The heat map is a thorough paradigm. A given document has a certain heat and then within that each section and each subsection of that has a given heat. For evaluation purposes it is convenient to order these elements first by document then by topical relevance within document.

4.5 Extending Web Search (Best in Context)

Lehtonen [13] makes the point that the user does not need to be aware of the fact that XML is used in the underlying system for it to be XML retrieval. He provides examples of web interfaces to XML collections. It is reasonable to ask what other aspects of the web could be likened to XML retrieval, or more to the point what aspects of XML retrieval are already in use on the web. The best in context task is one example.

It is not uncommon to see a web page with many internal links. Particularly in long documents (for example academic papers) a table-of-contents with links to entry points in that document is the norm. The web rendering of papers by BioMedCentral [2] for example, includes this kind of table-of-contents which they refer to as an outline. In some cases these links are inserted automatically by software that converts XML to HTML.

It is also not uncommon for web search engines such as Google [9] to present the user with a list of results that includes one sentence query-biased summaries (or snippets) of the identified documents. This is done so that the user can see, in context, where the search terms lie within the document.

It is, however, uncommon for information systems to provide both behaviors. It is not possible for a web search engine to instruct the web browser to load a given page and locate an arbitrary snippet at the top of the window – such behavior does not exist in the browser. But, given a set of entry points into the document (each marked as an HTML anchor) it is entirely possible for the search engine to instruct the browser to load the page and to locate the last entry point before the snippet.

In this example the entry points are chosen by the author of the web page and inserted manually. But this does not make it a reasonable criticism of the analogy. There are essentially no true structures in an HTML document and the best an author can do is to create these with anchors (cascading style sheets (CSS), div and span elements, and XHTML aside). Of course, if the authors were using XML then they would have structures available to use, but just as with the HTML example, they would manually have to choose to insert the structures (and the same is true with CSS). There seems to be no getting around the issue that entry points are manually chosen in both HTML and XML.

5. Existing Query Models

One is tempted to suggest that identifying any single implementation of XPath in a commercial product is sufficient evidence of its user base but it is of more value to identify mass use of the searching paradigms.

5.1 Keyword Only

Examples of keyword only interfaces abound the Internet (for example Google) and before this they abounded CD-ROMs.

5.2 Keywords Limited to Structures

Close inspection of the search interface provided by Google reveals that searches can be restricted to specific meta-data fields. This behavior is in use by search engines that utilize this search engine to search only their site. A query can, for example be restricted to the University of Otago web site by adding the search term "site:otago.ac.nz", requiring that the meta-data for the web page contain a structure called site and that that structure contain the expression otago.ac.nz.

Structured information systems such as PubMed [17] also provide similar behavior, in this case a user can restrict their search terms to occurring in a multitude of different structures including the title, the keywords, or even the abstract of a document.

5.3 Keywords Limited to Structures with Target Elements Specified

Although now defunct, BioMedNet once used an SGML [11] search engine that relied on fully normalized reference concrete syntax (it was essentially an XML search engine). The site provided a number of services including job listings, a store for biology supplies, magazines, academic journals, and abstracts from Medline.

The user interface to the search engine provided the user with the ability to search only the journal collection, only a single journal, only the jobs, only the store, or everything. Of course, the user also supplied keywords and could limit these to any structure seen in any of the SGML elements. This behavior is directly analogous to the user supplying the target elements and provided keywords optionally limited to structures (even though not implemented in this way).

5.4 Search Forms

Many search engines provide a so-called advanced interface in which the user specifies their information need by selecting from list boxes. Lehtonen [13] provides examples of such interfaces in which keywords are not needed, as well as ones in which they are.

One can easily lose sight of a goal when it lies directly under one's own nose. The bibliographic citation manager used by the authors of this contribution (EndNote) provides search functionality in which the user chooses fields from a list box and enters keywords to be limited to those fields.

6. XML-IR Use Cases

Formal use-cases are used to document the interaction of a user and an information system. Such use cases are often written in an environment in which the system does not yet exist, or in which the system is being modified (and consequently does not fully exist). They are design documents.

XML-IR is already in use, but this need not detract from the benefits of writing use cases. For one, a description of how it is being used could lead to insights as to how it might be used.

Cockburn [6] examines effective and ineffective use cases and provides a set of reminders. Reminder 6 is to “Get the Goal Level Right”. By this he is reminding us not to get bogged down in the detail of how exactly something happens, but rather to describe that it does. The exact details can sometimes be filled in with another use case while at other times it is obvious. How the user enters their query in a web search engine is less important than that they must do so.

Much of the fine detail of the interaction between the user and the information system is identical for the different XML retrieval tasks identified at INEX. The user sits at a computer, enters a query, and is presented with results. Because these details are the same each time, and essentially the same is seen in web search, it is unnecessary to include or repeat them – the level is wrong.

Cockburn [6] also examines several different formats for presenting use cases. These range from the *fully dressed* format that is highly formal with numbered steps to a drawn *diagram* format to a *casual* format that is paragrammatic. It is important to choose the appropriate style for both clarity and to avoid ambiguity.

The use cases included herein are in the *casual* format. They serve not to dictate the design of a system but rather are illustrative of the user’s needs, and why an XML-IR system of a particular kind is needed. As the interaction of the user and the search engine is obvious and ubiquitous, the use cases reduce to a description of the user’s need and why a particular strategy satisfies their needs.

6.1 The Thorough Use Case

The user needs maintenance information to maintain some modular engineering equipment, but cannot be certain of the exact modules until in the field. Such is the case when a modular system evolves over time but old and new pieces remain interchangeable. The product has an electronic set of service manuals that are constructed hierarchically with the details of different versions of the same module compiled into the same XML document.

The user believes it is possible to identify the component in the field from the service manuals, but to do so needs exhaustive details on each variant of a given components. The service manual is designed to extract the appropriate XML elements to form a coherent page regardless of the variant of that component.

The user consults the collection, which provides a ranked list of information elements for all variants of the component. These are chosen to best fit the rendering capacity of a handheld computer.

The user browses and selects from the result list the document fragments that match the modular component in the field.

The user identifies the version of the component.

The information system constructs the service pages for the component.

Sure of a valid identification and having the service manual page the user services the part and is finished.

6.2 The Focused Use Case

The user has been asked to present a report summarizing the many different views of a given debated topic. They are, consequently, interested in reading opinions without being interested in whose opinion it is. Such might be the case if a secondary school pupil were asked to present a report on the different views of the effects of global warming.

The user believes it is necessary to obtain the information from multiple sources and from many different collections.

They consult such a meta-search engine which provides a ranked list of information elements from multiple sources.

The user browses and selects from the result list and views accompanied information.

Uninterested in the source of the information, as it is opinion they are seeking, they take notes from relevant information elements.

The user combines the notes from the multiple sources into a coherent report.

Sure of having covered the multitude of opinions, the user is finished.

6.3 The Relevant in Context (and Fetch & Browse) Use Case

The user needs factual information to solve a dispute. Such was the original reason for the collection of facts now readily available as the Guinness World Records [10] (but previously under different names).

The user believes it is possible to obtain the information from a single source in a community built collection of facts (such as Wikipedia, or Guinness).

They consult such a collection which provides a ranked list of information elements.

The user browses and selects from the result list and views accompanied information.

Knowing the facts are under dispute, the user chooses to verify the credibility of the information. The information system provides two methods for doing this:

Either the user consults a discussion list associated with the information, and uses their personal experience and judgment in following the postings

Or the user consults the context of the information and uses their personal experience to judge the reliability of the source and therefore the credibility of the information.

Sure of the facts and the credibility of the source, the user is finished.

6.4 The Best in Context Use Case

The user is writing an academic paper and recalls some facts they wish to cite, but is unable to recall the source. The user is a regular and frequent user of an online academic digital library which houses many of the journals they frequently read. Such was the case for the authors of this paper during preparation.

The user is looking for a single source of the information from a reputable source (such as a journal on ScienceDirect).

They consult such a collection which provides a ranked list of information elements. As the information units are long (typically 8-30 printed pages), the result list includes query biased summaries (snippets).

The user browses and selects from the result list.

The information system presents the user with an article, pre-scrolled to the right location so that the snippet is on-screen in context, for the user to read.

The user discovers that their understanding of the details is not sufficiently detailed and chooses to read more of the context.

As the information unit is on-screen, the user scrolls and reads at their leisure to ensure they sufficiently understand the details.

Sure of their facts, and with a citation to them, the user is finished.

7. Conclusions

There are currently four *ad hoc* tasks at INEX: thorough, focused, relevant in context, and best in context. For each of these tasks it is reasonable to ask if it is a purely academic task being studied for academic reasons, or if such systems exist and therefore might be improved by research in XML-IR.

For thorough retrieval, internet book search as well as version control are identified as existing uses of the technology. For focused an application is news summarization. The relevant in context task is already in use in internet book search, and an application of best in context is suggested. Use cases for each of these four tasks are presented.

Accepting that it is reasonable to rank these tasks on credibility, we suggest the relevant in context task is most credible (as it has been show to exist), followed by best in context, and then focused, then thorough.

XML-IR users certainly exist. We believe the use cases are reasonable, and we hope that future research in XML-IR will consider the existing users as well as the hypothetical users for whom some of the INEX tasks were targeted.

References

[1] Baeza-Yates, R., Navarro, G., & Vegas, J. (1998). A model and a visual query language for structured text. In *Proceedings of the String Processing and Information Retrieval: A South American Symposium*, (pp. 7-13).

[2] BioMedCentral. (2006). Biomedcentral. Available: http://biomedcentral.com [2006, 21 November].

[3] Carmel, D., Maarek, Y. S., Mandelbrod, M., Mass, Y., & Soffer, A. (2003). Searching XML documents via XML fragments. In *Proceedings of the 26th ACM SIGIR Conference on Information Retrieval*, (pp. 151-158).

[4] Clark, J., & DeRose, S. (1999). XML path language (XPath) 1.0, W3C recommendation. The World Wide Web Consortium. Available: http://www.w3.org/TR/xpath.

[5] Clarke, C., Kamps, J., & Lalmas, M. (2006, to appear). INEX 2006 retrieval task and result submission specification. In *Proceedings of the INEX 2006 Workshop*.

[6] Cockburn, A. (2000). *Writing effective use cases* (1st edition ed.): Addison-Wesley.

[7] Dopichaj, P. (2006). Element retrieval in digital libraries: Reality check. In *Proceedings of the SIGIR 2006 Workshop on XML Element Retrieval Methodology*, (pp. 1-4).

[8] Evans, D., Klavans, J. L., & McKeown, K. R. (2004). Columbia newsblaster: Multilingual news summarization on the web. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics(HLT-NAACL-2004).*

[9] Google. (2006). Google. Available: http://google.com [2006, 21 November].

[10] Guinness. (1975). *Guinness book of records* (22nd ed.). London: Guinness Superlatives Ltd.

[11] ISO8879:1986. (1986). *Information processing - text and office systems - standard generalised markup language (SGML).*

[12] Kazai, G., & Ashoori, E. (2006). What does shakespeare have to do with INEX? In *Proceedings of the SIGIR 2006 Workshop on XML Element Retrieval Methodology*, (pp. 20-27).

[13] Lehtonen, M. (2006). Designing user studies for XML retrieval. In *Proceedings of the SIGIR 2006 Workshop on XML Element Retrieval Methodology*, (pp. 28-34).

[14] Mass, Y., & Mandelbrod, M. (2004). Component ranking and automatic query refinement for XML retrieval. In *Proceedings of the INEX 2004 Workshop*, (pp. 73-84).

[15] McKeown, K. R., Barzilay, R., Evans, D., Hatzivassiloglou, V., Kan, M.-Y., Schiffman, B., & Teufel, S. (2001). Columbia multi-document summarization: Approach and evaluation. In *Proceedings of the Document Understanding Workshop (DUC 2001).*

[16] O'Keefe, R. A. (2004). If INEX is the answer, what is the question? In *Proceedings of the INEX 2004 Workshop*, (pp. 54-59).

[17] PubMed. (2006). Pubmed. Available: http://ncbi.nlm.nih.gov [2006, 21 November].

[18] Tombros, A., Larsen, B., & Malik, S. (2004). The interactive track at INEX 2004. In *Proceedings of the INEX 2004 Workshop*, (pp. 410-423).

[19] Trotman, A. (2005). Wanted: Element retrieval users. In *Proceedings of the INEX 2005 Workshop on Element Retrieval Methodology, Second Edition*, (pp. 63-69).

[20] Trotman, A., & Geva, S. (2006). Passage retrieval and other XML-retrieval tasks. In *Proceedings of the SIGIR 2006 Workshop on XML Element Retrieval Methodology*, (pp. 43-50).

[21] Trotman, A., & O’Keefe, R. A. (2003). Identifying and ranking relevant document elements. In *Proceedings of the 2nd workshop of the initiative for the evaluation of XML retrieval (INEX)*.

[22] Trotman, A., & Pharo, N. (2006). User case studies track. INEX. Available: http://inex.is.informatik.uni-duisburg.de/2006/usercase.html [2006, 13 November].

[23] Trotman, A., & Sigurbjörnsson, B. (2004). Narrowed Extended XPath I (NEXI). In *Proceedings of the INEX 2004 Workshop*, (pp. 16-40).

[24] Trotman, A., & Sigurbjörnsson, B. (2004). NEXI, now and next. In *Proceedings of the INEX 2004 Workshop*, (pp. 41-53).

[25] van Zwol, R., Baas, J., van Oostendorp, H., & Wiering, F. (2005). Query formulation for XML retrieval with bricks. In *Proceedings of the INEX 2005 Workshop on Element Retrieval Methodology, Second Edition*, (pp. 80-88).