Query Formulation for XML Retrieval with Bricks

Roelof van Zwol, Jeroen Baas, Herre van Oostendorp, and Frans Wiering

Centre for Content and Knowledge Engineering
Utrecht University
Utrecht, the Netherlands
roelof, ajbaas, herre, fransw}@cs.uu.nl

ABSTRACT

XML retrieval, also referred to as Structured Document Retrieval is a discipline of information retrieval that focusses on the retrieval of relevant document fragments for a given information need that contains both structural and textual components.

In this article we will focus on the theory behind Bricks, a visual query formulation technique for XML retrieval that aims at reducing the complexity of the query formulation process and required knowledge of the underlying document structure for the user, while maintaining full expression power, as offered by the NEXI query language for XML retrieval.

In addition, we present the outcome of a large scale usability experiment, which compared Bricks to a keyword-based and a NEXI-based interface. The results showed that participants were more successful at completing a number of search assignments using Bricks or NEXI. Furthermore, we observed that the participants were also able to successfully complete their assignments in a significantly shorter period of time, when using the Bricks interface.

1. INTRODUCTION

The recent popularity of XML retrieval, or Structured Document Retrieval, is caused by the widespread use of the eXtensible Markup Language (XML) in digital libraries, intranet environments. Contrary to the well-know Internet search engines, XML retrieval systems try to exploit the structure of the documents during the retrieval process.

For XML retrieval systems to work in practice, it is crucial that users are capable to adequately used the structure of a document in all facets of the retrieval process. Not only do we need good retrieval strategies, but the offered functionality should correspond to the user's need.

In this article the focus is on the query formulation process for XML retrieval. The large scale search engines that are available on the Internet allow easy access to large quantities of information that is available on-line. Using a few keywords a user can formulate his information need and retrieve a list of relevant documents, which needs to be browsed for the relevant information. This approach is satisfactory for most users, but for digital libraries and large intranets, where the information need is usually more specific and large amounts of information on the subject is available more sophisticated query formulation techniques are desired.

Current approaches in XML retrieval allow a user to either specify his information need using keywords (content only), or by using a combination of structural constraints and keywords. This is formalized in the NEXI query language [14], where a user can specify his information request through an XPath-like expression [4], that combines both the structural and content-based aspects of the user information need.

Using such a query language for the retrieval provides powerful expression mechanisms, but also has its impact on the query formulation process. The user should be able to express his information need using the syntax of the query language, and in addition the user should have knowledge of the structure of the document.

Consider the information need of Example 1, where a user visiting the Lonely Planet Web-site wants to:

Example 1
Find historical information about ***revolutions*** *for destinations with a* ***constitutional monarchy*** *as government.*

Using a (NEXI-CO) content-only approach, the user is likely to use the following keyword combination to formulate his information need:

history revolutions destination government "constitutional monarchy"

Without any path directives in the information request a XML retrieval system can literally retrieve any document fragment that contains one or more of the given terms. For example, this can be a piece of text that is emphasized, or the entire document.

Taking a closer look at the information need, we can see that the objective is to retrieve historical information. Furthermore suppose that the user is familiar with the (semanti-

cal) structure of the document collection, he is then able to identify the structural conditions of the information need. In Example 1 the structural conditions of the information request are underlined, while the emphasized terms form the content-based aspects of the information need. If we make the transition of the information need to a formal specification, we will end up with the following NEXI content and structure (NEXI-CAS) query:

//destination[about(.//government, "constitutional monarchy")]//history[about(., revolutions)]

This NEXI query consists of two parts, a request query and a support query. The request query specifies the type of document fragment that should be returned by the system:

//destination//history[about(., revolutions)],

while the support query is used to specify additional conditions that should be met:

//destination[about(.//government, "constitutional monarchy")]

A NEXI-CAS query always consists of a request query that has a request path and a filter with one or more *about*-clauses. The request path specifies the desired element of retrieval, while the filter is used to specify the structural and textual conditions. Each about-clause has two arguments, a path directive and a list of terms. The path directive specifies where within the request path to search for the specified terms. Similarly the support query consists of a support path and filter that can contain one or more about-clauses.

The NEXI query language provides exactly the necessary expression power for XML retrieval. Although the syntax of the NEXI query language is relatively simple, a user needs to learn the syntactical features. This makes it hard, if not impossible, for the average user to express their information need in NEXI.

To overcome these limitations we have developed *Bricks*, a visual query formulation technique for XML retrieval that aims at:

1. Reducing the complexity of the query formulation process.
2. Reducing the required knowledge of the document structure.
3. Maintaining maximum expression power, as offered by the NEXI query language.

To realize this, Bricks uses a graphical approach that allows the user to specify his information need using small building blocks ('bricks'), starting with the specification of the desired element of retrieval. As a result, Bricks is guiding the user in a more natural way through the query formulation process. Not only does it solve the syntactical formulation issues, it also prevents possible information overload, when the document structure is large and complex. This is realized by using a priority for the different document elements. Elements with a low priority are not visible for the user early in the query formulation process. In Figure 1 the information need of Example 1 is expressed with Bricks.

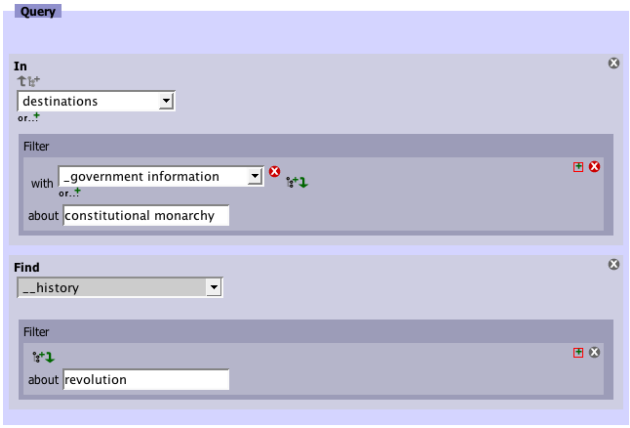


Figure 1: Example information request in Bricks

To validate our ideas, we have designed and implemented the Bricks interface on top of the XML retrieval system that was developed for participation in INEX, the INitiative for the Evaluation of XML retrieval [6]. INEX provides an international platform for the evaluation of XML retrieval strategies, allowing researchers to measure the retrieval performance of their system.

Finally we have set-up and performed a usability experiment to evaluate our ideas. In the experiment, we have compared Bricks with a keyword-based (NEXI-CO) approach and a 'content and structure'-based (NEXI-CAS) approach. We will briefly discuss the outcome of the experiment in terms of effectivity and efficiency.

1.1 Organization

In Section 2 we discuss the related work on structured document retrieval, and in particular on query formulation techniques. It also provides additional background information on the experimental setting that was used to evaluate our theory on structured query formulation. The theoretical foundation for Bricks is then discussed in Section 3. We then present the outcome of the usability experiment in Section 4. Finally, we will come to the conclusions and discuss future research in Section 5.

2. RELATED WORK

In general an information retrieval system consists of three components: a query formulation interface, a retrieval strategy (engine), and an interface for the result presentation. Below we will discuss the impact of and various research approaches on structured document retrieval for each of the three components.

2.1 Query formulation

The research on query formulation, presented in this article is using the NEXI query language as a starting point. NEXI [15, 14] is an XPath-based query language that primarily focusses on the extraction of relevant information,

using a combination of path directives and content-based filters. This makes NEXI an excellent query language for XML retrieval, providing a powerful expression mechanism to the user.

Alternative XML query languages, such as XQuery[3] and XSLT[1], do not focus on the retrieval task. They provide additional functionality that lays outside the scope of structured document retrieval, like for example transformations on the extracted XML document structure.

A more trivial query formulation technique is adopted by Lucene [7]. Information that is found within a specific field, i.e. an XML element, can be specifically targeted, like:

government: "constitutional monarchy"

The downside of this approach is that it is not possible to retrieve anything other than the document containing the requested field and content, or to specify more complex paths.

Of course one should not neglect the power of keyword-based information retrieval. It is still the driving force behind all popular search engines, and allows literally anyone to specify his information need with just a simple keyword combination. The NEXI query language therefore allows for the specification of keyword combinations, including the usage of phrases. This is referred to as NEXI-CO (Content Only), while a NEXI query that contains a path specification is referred to as a NEXI-CAS query (Content and Structure).

The Bricks query formulation technique uses a graphical approach. In [12] a method for replacing a complex command language syntax is discussed, called direct manipulation. By using a graphical interface the syntactical formulation of the query is represented by graphical items. This allows the user to successfully execute complex queries on a data structure.

In our prior research on schema-based structured document retrieval we have developed the Webspace method [16, 17]. There, we have shown that the retrieval performance can be improved by presenting the user a graphical interface that visualizes a (database-oriented) schema representing the semantical structure of the document collection. The user then formulates his information need in a materialized view on the schema. The approach followed for Bricks is schemaless and uses path directives to derive the requested information.

2.2 Retrieval strategy

The success of a XML retrieval system also depends heavily on the retrieval strategy. It executes the (structural) information request and derives a ranked list of relevant document fragments. In INEX, the INitiative for the Evaluation of XML Retrieval [6] the retrieval performance of XML retrieval strategies is evaluated. Participating in INEX allowed us to develop, evaluate, and improve various retrieval strategies [19] for XML retrieval. For the evaluation of Bricks and the other query formulation techniques discussed here, we have used our best-performing retrieval strategy.

Within INEX a number of user-related issues are being discussed. With respect to query formulation it is the question whether the structural conditions of the information request should be strictly interpreted, or whether these conditions should be seen as merely hints of where the user expects to find the relevant information. This is also referred to as the vague interpretation [9]. For our experiment, we have used a semi-strict interpretation of the path directives, which penalizes relevant document fragments that do not exactly fulfill the structural conditions of the information request.

Another issue within INEX, refers to result presentation. It deals with the question what the most specific and exhaustive element of retrieval is for a given information need. As a result it is possible that the list of document fragments returned by the retrieval strategy contains overlapping results [8]. When an XML fragment is considered relevant, its parent is by definition also relevant, and probably more exhaustive. From a user perspective, however, it is undesirable to have redundancy in the ranking of the document fragments.

2.3 Result presentation

Since relatively small document fragments are derived by the system, it is possible to use alternative techniques to present the retrieved information to the user. This is also the scope of the INEX interactive track [13]. There the interaction of the user with a result presentation interface for structured document retrieval has been evaluated. Using a content-only approach for query formulation, they were able to analyze user behavior with the presentation interface.

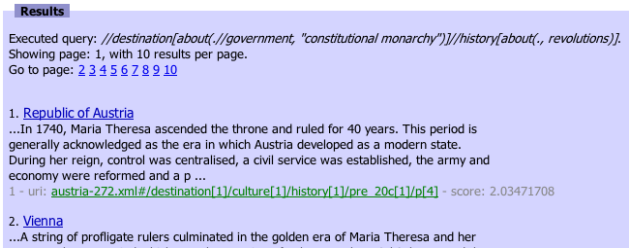


Figure 2: Snapshot of result presentation interface

For our research we use a commonly accepted presentation technique that provides a link to the relevant fragment, a short summary of the fragments content, and some additional statistical information that help the users to judge the relevancy of the retrieved information. Figure 2 shows a snapshot of the presentation technique used for our research. Nearly all main search engines available use this presentation format, therefore we can safely assume that the result presentation is not of significant influence to the result of our experiment.

3. THEORETICAL FOUNDATION

The theoretical foundation for Bricks can derived from the three objectives that are identified:

1. Minimize the complexity of the query formulation process.

2. Minimize the required knowledge of the document structure.

3. Maximize the expression power as provided by the NEXI query language.

Based on these objectives the follow design principles can be obtained that together form the theoretical basis of Bricks.

3.1 A graphical approach

The use of a graphical interface reduces the burden of syntactical formulation issues that are related to the NEXI query language. Although NEXI uses a relatively simple syntax based on XPath, it still allows users to submit malformed queries to the retrieval system. Apart from incomplete queries, this is not possible with the graphical approach adopted by Bricks. This is referred to as direct manipulation of the query language [12].

Furthermore, the underlying structure that is present in the document collection can be integrated into the query interface. Several approaches are thinkable, but for Bricks we have chosen to work with pull-down lists, allowing the user to select structural elements into the query. Alternatively a tree-based approach can be used to visualize the structure to the user. However, this is a more complex structure that needs to be interpreted by the user.

3.2 Intuition of a mental model for query formulation

When formulating a specific information request, the user has a mental model of the information he is looking for. Research on information seeking behavior [10, 11] has shown that users develop such a mental model, and that the effectivity of the task performance can be increased if the interface and offered functionality is closely related to the mental model of the user. When focussing on query formulation for structured document retrieval the task is more complex, since the user has to specify what the structural and content-based conditions of his information need are. If a user is asked to express his information need in natural language, he is likely to formulate a sentence like: "*Find historical information about revolutions, for destinations ...*".

A logical first step is to specify the requested element of retrieval, "*Find historical information*". From there a limited number of iterative steps are possible. The user either specifies a content-based constraint, "*about revolutions*", using the filter that is associated with the request path, or adds additional path directives to the request path, "*, for destinations*". If needed the user can add one more content-based filter, and simultaneously introduce a support path to the information request. This allows the user enough 'freedom' to follow his intuition, and to perform intermediate checks on the specified information request.

3.3 Step-by-step formulation of the information need: the building blocks

Bricks uses small building blocks to formulate the information request (query). Each block represents a small step in the formulation process, that needs to be completed, before another block is added to the query. After specifying the requested element of retrieval, the user can add an *about* clause to the request filter, or specify additional path directives to the request path. Adding an about clause allows the user to specify a content-based constraint, and to descend further down the document structure.

Adding an additional path directive allows the user to go up in the tree, this is referred to as the support path. Another block is added to the query for each step that is taken by the user. Based on the document structure and the syntax of the NEXI query language, the possible actions are controlled by the NEXI interface. This prevents the specification of malformed and unmeaningful (with respect to the document structure) queries. On the other hand we aim at preserving full expression power, as offered by the NEXI query language. In the next section, we will show how a nested object structure of a NEXI query is constructed with Bricks, to prove that we are able to achieve this.

3.4 Avoiding information overload

It is important that the user is not overwhelmed with options and possible next steps. In a sense, the intuition of a mental model is one approach to avoid information overload. Using a wizard-based approach, is a proven technique to reduce the learning curve of a task that needs to be accomplished. However expert users can experience a limitation in the provided functionality, causing them to get frustrated [5, 18]. In our case, we are not focussing on the high-end experts, such as programmers and database administrators, but on users with a complex information need that goes beyond the average profile of a user on the Internet. Although Bricks is more flexible than a wizard-based approach, the aim is similar: by reducing the number of options that are available, it becomes easier to complete (more efficient) the query formulation task.

In an attempt to reduce the required knowledge of the document structure, Bricks provides lists of structural elements that allow the user to select path elements into their query. However, the Lonely Planet XML document collection contains 271 unique element and attribute names. This can easily cause an information overload for the user, and cause the efficiency of the task performance to drop. When inspecting the structural elements, it becomes apparent that not all elements are meaningful from a retrieval perspective. For instance, the retrieval of a highlighted (italic) text fragment, containing just a few keywords, will probably not satisfy the user's information need, since all context is missing.

In general, it is possible to define a structure for the document collection that consists of three layers, as is presented in Figure 3. The top layer is formed by a semantical markup that provides a high level description of the content that is contained. The middle layer provides a logical markup, containing elements that have a logical function/meaning to the user. I.e. a chapter and its sections form logical containers of information. At the bottom layer the presentation markup is found, which is used for visual layout and presentation of the content. Any XML document can be seen as a tree. When using such a three-layer structure, the semantical element will naturally appear in the top of the tree, while the presentation element as usually found near the leafs to the tree. The mid-section of the XML document will then

contain the logical elements.

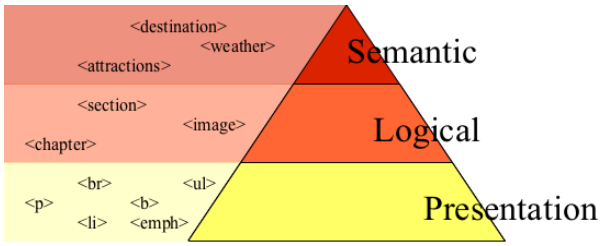


Figure 3: Three layer structure for XML document collections: semantical-, logical-, and presentation markup.

Bricks exploits this three layer structure in the retrieval process by adding a priority to each of the structural elements. Semantical elements will receive a high priority, followed by the logical elements, while the presentation elements are given a low priority. Early in the query formulation process, only the high priority elements can be selected in the query. Elements with a lower priority will become available once the user has made a first selection of the elements that should be retrieved. In a sense the user is traversing down the tree structure of the document collection, and narrowing down the possible elements that can be added to the query.

In practice a threshold of 20 elements is used, limiting the number of structural elements that can be presented to the user at once. Alternative presentation techniques with sublists are possible, to allow the user to explore a larger set of structural elements that can be included in the query.

4. USABILITY EXPERIMENT

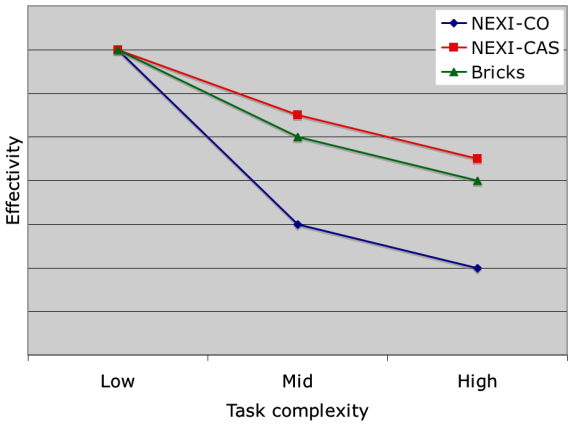
In this section we will briefly discuss the outcome of the usability experiment that was performed to evaluate three different query formulation techniques: keyword-based (NEXI-CO), content and structure-based (NEXI-CAS), and Bricks. First we will discuss the hypotheses that were formulated for the experiment, then present the setup and methodology used for the experiment and finally discuss the results and some observations. A more detailed discussion of the experiment, including the results of a retrieval performance experiment can be found in [2].

4.1 Hypotheses

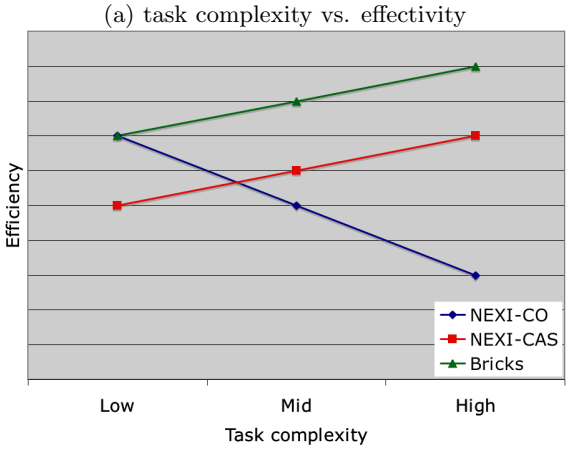
In this article we have formulated three objectives that are important for query formulation in structured document retrieval: (1) minimize the complexity of the query formulation process to the user, (2) minimize the required knowledge of the structure of the document collection, and (3) provide maximum expression power to the user, allowing him to express his (complex) information need. Based on these objectives, we have formulated three hypotheses.

Hypothesis 1
Use of sophisticated query formulation techniques will lead to a higher effectiveness of the task performance.

The intuition behind Hypothesis 1 is that if a user can add structural conditions to the information request, by using



(a) task complexity vs. effectivity



(b) task complexity vs. efficiency

Figure 4: Expected performance, including task complexity

either NEXI-CAS of Bricks, the user is more successful in completing a given search task. Furthermore we designed Bricks to provide similar expression power as is available in NEXI-CAS. Therefore we expect that regardless of the task complexity the effectivity of NEXI-CAS and Bricks will almost be equal, but significantly higher than for NEXI-CO.

When taking the task complexity into account, we expect to find that for tasks with a low complexity the three approaches will have a similar performance, however if the task complexity increase, the effectivity of NEXI-CO will drop. The effectivity of NEXI-CAS and Bricks should remain more or less constant, or slightly decrease. This is depicted in Figure 4.a.

To measure our expectations we have introduced the following effectivity measure:

$$effectivity = \frac{\sum_{fragment} relevant_{fragment} \times \frac{relevant\_retrieved_{fragment}}{rank_{fragment}}}{\mid fragment \mid} \quad (1)$$

$$, \; scale : [0..1]$$

The effectivity measures assigns a score between 0 and 1 to a query that is submitted by a user for a particular information need. It takes into account the number ($relevant\_retrieved_{fragment}$) and position ($rank_{fragment}$) of the relevant fragments ($relevant_{fragment} : [0, 1]$) that are retrieved in the top 10 results. The measure balances the effectivity score, if less then ten results are retrieved for a given user query ($\mid fragment \mid$).

Hypothesis 2
The Bricks approach for query formulation will increase the efficiency of the user for a given task.

When taking the time factor into account, we expect to see a different picture, if our assumptions for Bricks are correct, a user should be able to successfully complete a search task in a shorter period of time, compared to NEXI-CAS. It is hard to predict how this will relate to NEXI-CO, because we expect that the user behavior is more focussing on query refinement and a quick scan of the list with retrieved document fragments. This corresponds with normal search behavior of users on the Internet [20]. Figure 4.b illustrates the expected efficiency for the different systems, when task complexity is taking into account.

We will measure the efficiency of the systems using the following formula:

$$efficiency = \frac{effectivity}{100\ seconds} \quad (2)$$

Hypothesis 3
Bricks will achieve a higher overall satisfaction among users that perform a (complex) search, when compared to the NEXI-based approaches.

We expect that: users, which are offered sophisticated query formulation techniques (Bricks and NEXI-CAS) will be more satisfied, than those users working with a keyword-based interface. In addition we expect that reduction of both the syntactical and structural problems at the user interface will also have a positive influence on the user satisfaction.

We will measure the user satisfaction through a survey, directly after the experiment, using 7-point Lickert scales.

4.2 Setup of the experiment

For this experiment, we used TERS, the Testbed for the Evaluation of Retrieval Systems [20]. TERS provides an experimental environment for the support of various evaluation tasks. It hosts two types of experiments, a usability experiment and a retrieval performance experiment, where the aim of TERS is to investigate the correlation between both experiments. We conducted both experiments, but limit ourselves here to the results of the usability experiment. For the usability experiment we have used the following setup:

Document Collection

For the experiment we used Lonely Planet destinations material, which consists of XML documents with interesting facts and background information about destinations on our planet.

Systems

Three systems were prepared for the experiment: NEXI-CO, NEXI-CAS, and Bricks. To eliminate undesirable side-effects all three systems used the same retrieval engine and result presentation technique.

Users

For the experiment we used a pool of 54 students, that participated in the course 'Multimedia Information Retrieval'. During this course they were taught the basic principles of structured document retrieval, and they followed a lecture on the NEXI query language. Prior to the experiment, they had to complete an assignment where they were asked to create both NEXI-CO and NEXI-CAS queries for fifteen representative information needs, based on the Lonely Planet.

Topics

For the usability we have used 27 topics, i.e. search assignments, representing specific information needs of travelers that are doing a background search, for instance to plan their next holiday. The topics can be sorted in three complexity groups, ranging from low to high complexity. To sort the topics, we have counted the syntactical and structural elements of the ideal NEXI-CAS query that represents the information need expressed in the topic.

Survey

Prior to and directly after the experiment we have presented the participants a list of questions to examine their level of expertise and experiences with the retrieval system.

Experience

In the first 30 minutes of the experiment, participants tried out the TERS interface, and played with the interface of the systems, to become familiar with the setup of the experiment and to reduce the learning effects.

4.3 Results of the usability experiment

In this section, the results of the usability experiment are presented. First we will give a brief overview of the overall results, and then discuss the influence of tasks complexity to the performance.

4.3.1 Overall results

In Table 1 the overall results of the experiment are presented for the three systems based on the measures that were used for the experiment: *time*, *effectivity*, *efficiency*, and *satisfaction*. The effectivity measure, which is used to test Hypothesis 1 shows that a significant difference ($p < .000$) is found between the systems, where both NEXI-CAS and Bricks were more effective than NEXI-CO. This indicates that the use of sophisticated query formulation techniques has a positive influence on the task performance.

Overall performance

System	Time	Effectivity	Efficiency	Satisf.
NEXI-CO	***198***	0.27	0.15	4.1
NEXI-CAS	245	***0.34***	0.14	***4.7***
Bricks	214	0.32	***0.16***	4.6

Table 1: Overall performance for the three systems based on time, effectivity, efficiency and satisfaction.

Effectivity

System	1	2	3
NEXI-CO	0.45	0.35	0.14
NEXI-CAS	***0.48***	***0.48***	***0.21***
Bricks	0.47	0.47	0.18

Time (sec.)

System	1	2	3
NEXI-CO	136	***154***	***246***
NEXI-CAS	160	189	311
Bricks	***134***	160	277

Efficiency (effectivity/100 sec.)

System	1	2	3
NEXI-CO	0.33	0.23	0.06
NEXI-CAS	0.30	0.25	***0.07***
Bricks	***0.35***	***0.30***	***0.07***

Table 2: Experiment results, including task complexity (tabular overview)

When the time factor is taken into account, it becomes apparent that users need significantly ($p < .000$) more time to formulate their information need in NEXI-CAS expressions. As a result, Bricks becomes the most efficient approach ($p < 0.04$) of the three systems, which confirms Hypothesis 2.

Inspection of the outcome of the experiment for user satisfaction, shows that users appreciate the additional query formulation power, but the did not rule in favor of Bricks. The highest satisfaction was achieved with NEXI-CAS, followed at a minimal distance by Bricks. Given that the satisfaction scale goes from 1 to 7, we can conclude that the users were content with both the Bricks and NEXI systems. However, we will have to drop Hypothesis 3.

4.4 Including task complexity

A more detailed insight in the results can be obtained when task complexity is also considered an influencing factor. Table 2 shows the raw results of the experiment for the measures *effectivity*, *time*, and *efficiency*, when task complexity is taken into account.

Effectivity

Figure 5.a shows the influence of task complexity on the effectivity of the performance. When comparing the results with our expectation, as shown in Figure 4.a, we see a sudden drop in effectivity for Bricks and NEXI-CAS, which was not predicted. The overall picture however, supports our expectations.

Time

When comparing the task complexity with respect to the average time needed to complete a task, we see that time is increasing with the task complexity, regardless of the system. This is illustrated in Figure 5.b. However, on average the users need more time to formulate NEXI-CAS queries.

Efficiency

Figure 5.c illustrates the combination of effectivity and time into the efficiency measure. It shows how Bricks is outperforming the other systems for tasks with a low and mid complexity, but that the efficiency for the three systems for highly complex tasks is almost at an equal low point, due to the extra time needed to complete the search assignment. Comparing the results for efficiency with our expectations, as depicted in Figure 4.b, we are mildly positive with the outcome. We had not anticipated the non-linear increase in time needed to complete highly complex tasks.

4.5 Observations

At this point we also want to discuss some of the observations that were made during the experiment. The search behavior of the users working with the different systems was entirely different. Users working with the NEXI-CO interface used many iteration steps to formulate a query and inspect the top of the ranking. If the results were unsatisfactory, they refined their query and tried again.

The participants working with the NEXI-CAS interface show a different strategy: they constructed the NEXI query in several steps. After each step, they submitted the query, to check the syntax and the intermediate results. Then continued extending the query, until the were satisfied with the results. Manual inspection of the submitted queries, showed numerous syntax errors, and misinterpretation of the document structure.

Finally, we observed that the participants working with Bricks hardly used any refinement steps. They continued working until they fully created a representation of the information need in Bricks, and only then inspected the results.

5. CONCLUSIONS

Structure document retrieval gained its popularity due to the use of XML in digital libraries, intranet environments, and large structured web-sites, where users have a specific and often complex information need. For structured document retrieval to work in practice, it is important that users are capable to adequately used the structure of a document in all facets of the retrieval process.

In this article we have identified three aspects that are of influence on the query formulation process for structured document retrieval: (1) adequate expression power, (2) syntactical complexity of the query formulation, and (3) required knowledge of the document structure. Using a keyword-based approach will not provide the user sufficient expression power, as is for instance provided by the NEXI query language. The NEXI query language allows a user to specify both the structural and content-based aspects of the information need, but also burdens the user with syntactical issues during the query formulation process. In addition, the

user has to be familiar with the structure of the document collection to avoid the specification of ill-formed structural paths.

Based on these aspects we have introduced Bricks, the building blocks to tackle query formulation issues in XML retrieval. The objective of Bricks is (1) to reduce the syntactical complexity of the query formulation process, (2) to minimize the required knowledge of the document structure, while (3) maintaining maximum expression power. We have explained how the objectives are used to form the theoretical foundation of Bricks. By using a graphical approach and the intuition of a mental model for query formulation, Bricks allows the user to step-by-step formulate his information need, while avoiding a possible information overload.

Finally we have discussed the outcome of a large scale usability experiment that evaluated the performance of Bricks, with respect to a keyword-based and NEXI-CAS system. Based on the results, we can concluded that sophisticated query formulation techniques, such as offered by Bricks and NEXI-CAS, will increase the success rate of the task performance in terms of effectivity. Furthermore, we can conclude that Bricks is more efficient, since will allow users to successfully complete a given task in a shorter period of time, compared to the keyword-based and NEXI-CAS approaches.

When taking task complexity into account, we found that the effectivity will decrease when the task complexity increases, but that NEXI-CAS and Bricks are more effective for the mid and highly complex search tasks. Increase in task complexity, will also lead to a non-linear increase in time needed to complete the task, causing the efficiency to drop significantly for all systems for the tasks with a high complexity.

Future research

For our future research we will work on alternative query formulation techniques that exploit the tree-based nature of XML documents. Furthermore, we are investigating how user-profiling can be used to enhance keyword-based query formulation for XML retrieval. With respect to result presentation, we will work on sophisticated techniques that use a query driven navigation, allowing the user to inspect the various structural and textual conditions of the information request. Finally we will continue to improve our retrieval engine, by participation in INEX.

Acknowledgments

At this place we would like to thank the Lonely Planet organization. They provided the XML document collection, based on the destinations material, that is used for our experiments. This allowed us to validate our ideas on query formulation issues in structured document retrieval. Furthermore we would like to thank the students that participated in the experiment, and provided us valuable information and new insights.

6. REFERENCES

[1] S. Adler, A. Berglund, J Caruso, S Deach, T. Graham, P. Grosso, E. Gutentag, A. Milowski, S. Parnell, J. Richman, and S. Zilles. Extensible stylesheet language (XSL). W3c recommendation, W3C: World-Wide-Web Consortium, October 2001.

[2] A.J. Baas. Structured document retrieval from a user perspective. Master's thesis, Center for Content and Knowledge Engineering, Institute for Information and Computing Sciences, Utrecht University, Utrecht, the Netherlands, July 2005.

[3] S Boag, D Chamberlin, M.F. Fernandez, D. Florescu, J Robie, and J. Simeon. Xquery 1.0: An XML query language. Working draft, W3C: World-Wide-Web Consortium, April 2005.

[4] J. Clark and S. DeRose. XML Path Language (XPath). Technical report, World-Wide-Web Consortium (W3C), 1999.

[5] D.C. Dryer. Wizards, guides, and beyond: Rational end empirical methods. In *proceedings of the International Conference on Intelligent User Interfaces*, pages 265–286, New York, NY, ASU, 1997. ACM Press.

[6] N. Fuhr, M. Lalmas, S. Malik, and Z. Szlávik, editors. *Advances in XML Information Retrieval*, number 3493 in LNCS, Schloss Dagstuhl, Germany, June 2005. INitiative for the Evaluation of XML Retrieval, Springer.

[7] E. Hatcher and O. Gospodnetic. *Lucene in Action*. ISBN: 1932394281. Manning Publications Co., Januari 2005.

[8] G. Kazai, M. Lalmas, and A. de Vries. The overlap problem in content-oriented XML retrieval evaluation. In *proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, number ISBN:1-58113-881-4, pages 72 – 79, Sheffield, UK, 2004. ACM Press.

[9] M. Lalmas and T. Roelleke. Modelling vague content and structure querying in XML retrieval with a probabilistic object-relational framework. In *FQAS, 6th International Conference On Flexible Query Answering Systems*, Lyon, France, June 2004.

[10] L. Meho and H. Tobbo. Modelling the information-seeking behaviour of social scientists; Elly's study revisited. *journal of American Society for Information Science and Technology*, 4(6):570–587, 2003.

[11] J Muramatsu and W. Pratt. Transparent queries: Investigating users' mental models of search engines. In *proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 217–224. ACM Press, 2001.

[12] J. Preece, Y. Rogers, H. Sharp, D Benyon, S. Holland, and T. Carey. *Human Computer Interaction: Concepts and Design*. Number ISBN: 0201627698. Addison Wesley, 1994.

[13] A. Tombros, B. Larsen, and S. Malik. The interactive track at INEX 2004. In N. Fuhr, M Lalmas, S. Malik, and Szlavik Z., editors, *Advances in XML Information Retrieval. Third Workshop of the INitiative for the Evaluation of XML Retrieval*, number ISBN: 3-540-26166-4 in LNCS 3493, pages 410–423. DELOS - Network of Excellence on Digital Libraries, Springer, 2005.

[14] A Trotman and B. Sigurbjörnsson. Narrowed extended xpath i (NEXI). In N. Fuhr, M Lalmas, S. Malik, and Szlavik Z., editors, *Advances in XML Information Retrieval. Third Workshop of the INitiative for the Evaluation of XML Retrieval*, number ISBN: 3-540-26166-4 in LNCS 3493, pages 16–40. DELOS - Network of Excellence on Digital Libraries, Springer, 2005.

[15] A Trotman and B. Sigurbjörnsson. NEXI, now and next. In N. Fuhr, M Lalmas, S. Malik, and Szlavik Z., editors, *Advances in XML Information Retrieval. Third Workshop of the INitiative for the Evaluation of XML Retrieval*, number ISBN: 3-540-26166-4 in LNCS 3493, pages 42–53. DELOS - Network of Excellence on Digital Libraries, Springer, 2005.

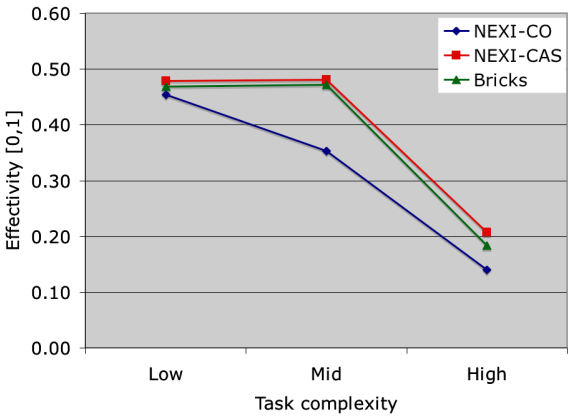
[16] R. van Zwol. *Modelling and Searching Web-based Document Collections*. Ctit ph.d. thesis series 02-40, Centre for Telematics and Information Technology (CTIT), Enschede, the Netherlands, April 2002.

[17] R. van Zwol and P.M.G. Apers. Complex query formulation with the webspace method. In *proceedings of the Sixth World Multi Conference on Systematics, Cybernetics, and Informatics*, number ISBN: 980-07-8150-1, pages 200–208, Orlando, Florida, USA, July 2002. IIIS.

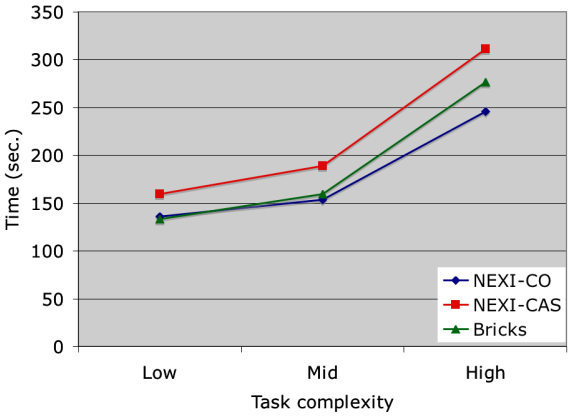
[18] R. van Zwol, A. Callista, J. Molenaar, and F. Wiering. Content authoring in an XML-based and author-friendly environment: Fact or fiction? In *proceedings of the third International Conference on Computer Science and its Applications (ICCSA'05)*, San Diego (CA), USA, June 2005. US Education Services.

[19] van R. Zwol, V. Dignum, and F. Wiering. The Utrecht Blend: Basic ingredients for an XML retrieval system. In N. Fuhr, M Lalmas, S. Malik, and Szlavik Z., editors, *Advances in XML Information Retrieval. Third Workshop of the INitiative for the Evaluation of XML Retrieval*, number ISBN: 3-540-26166-4 in LNCS 3493, pages 140–152. DELOS - Network of Excellence on Digital Libraries, Springer, 2005.

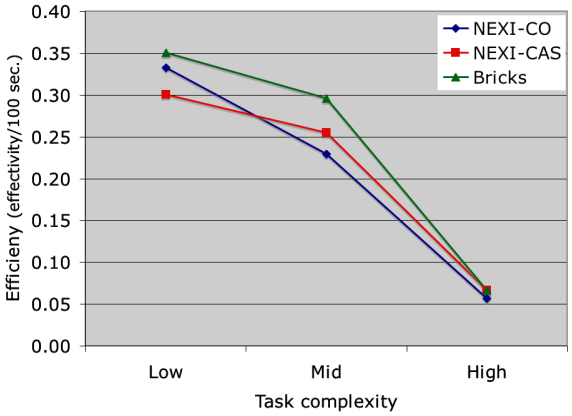
[20] van R. Zwol and H. van Oostendorp. Google's "I'm feeling lucky", truly a gamble? In Xiaofang Zhou, Stanley Su, Mike P. Papazoglou, X Maria Zhou, S. Su, M.P. Papazoglou, M.E. Orlowska, and K.G. Jeffery, editors, *Web Information Systems - WISE 2004, Proceedings of the 5th International Conference on Web Information Systems engineering*, number 3-540-23894-8, pages 378–390, Brisbane, Australia, November 2004. Springer.



(a) task complexity vs. effectivity



(b) task complexity vs. time



(c) task complexity vs. efficiency

Figure 5: Experimental results, including task complexity