Query Formulation for XML Retrieval with Bricks

Bricks, the building blocks to tackle query formulation issues in structured document retrieval

Roelof van Zwol
Jeroen Baas
Herre van Oostendorp
Frans Wiering

Outline

- Motivation
- Objectives with Bricks
- Theory behind Bricks
- Usability experiment
- Conclusions
Motivation

- Structured document retrieval (XML retrieval) is different:
  - Query formulation
    - Search can contain both structural and textual conditions.
  - Retrieval strategy
    - Exploit the document structure to retrieve relevant document fragments.
  - Result presentation
    - Present individual document fragments, or clustered fragments (browse and fetch), requires new navigation techniques.

Outline - Motivation - Objectives - Theory - Usability - Conclusions

Centre for Content and Knowledge Engineering, Utrecht University, the Netherlands

Running example:

- A user planning his holiday, has the following specific information need:
  - “Find information about traveling to destinations with major airports, where the weather has a tropical climate.”

- Based on the Lonely Planet collection.

Legend:
- Structural conditions
- Textual conditions
Query formulation

- Techniques:
  - Keyword based (NEXI-CO)
  - Natural Language Processing
  - Combination of keyword- and structure-based search (NEXI-CAS)
  - Bricks...

NEXI-Content Only:
- Keyword-based query, containing words and phrases.
- User is ignorant of document structure.
- Any document or XML document fragment can be returned.

Information request:
- “major airport” destination weather “tropical climate”
Query formulation

- **Natural language processing:**
  - Complex formulation, including structure is very well possible in theory.
  - Close to user’s “mental model”.
  - User needs to know about structure.
  - Brisbane’s NLP2NEXI engine:
    - Kindly provided by [Geva et al.]
    - We ran into conversion problems, with respect to recursive structure, and generation of complex NEXI-queries.
    - Therefore not included into experiment.

- **Information request:**
  - Find information about traveling to destinations with major airports, where the weather has a tropical climate.

Outline - Motivation - Objectives - Theory - Usability - Conclusions

---

Query formulation

- **NEXI - Content and Structure:**
  - Query contains structural requirements.
  - User is aware of document structure, and …
  - User is capable to specify structural constraints.

- **Powerful query language for structured document retrieval.**

- **Information request:**
  - \\
  //destination[about(.,//weather, “tropical climate”)]
  //getting_there_and_away[about(., “major airport”)]
Query formulation

Objectives with Bricks

- Minimize the **complexity** of the query formulation process
- Minimize the **required knowledge** of the document structure
- Maximize the **expression power** as provided by NEXI
Running example in bricks

Theory behind Bricks

- Graphical approach
- Intuition of a mental model
- Building blocks
- Avoid information overload
Graphical approach

- Reduces syntactical formulation issues.
  - Bricks is NEXI-compatible.
- Reduces/eliminates knowledge of document structure.
  - Bricks uses pull-down lists.
  - Alternative in development: TreeSearch.
- Avoids formulation of malformed information requests.
  - Bricks uses extensive checks for both query syntax and structure.
- Referred to in literature as:
  - Direct manipulation [Preece et al.]

Outline - Motivation - Objectives - Theory - Usability - Conclusions

Intuition of a mental model

- A user has a mental model of the information he is looking for.
- Effectivity and efficiency of user increases when the formulation process is close to the user’s mental model.
- User thinks in ‘natural language’…
  - … and is likely to specify the requested element of retrieval first:
    - “Find information about traveling to…”
  - … NEXI-specification is not user oriented, but structure oriented…

Outline - Motivation - Objectives - Theory - Usability - Conclusions
Building blocks

- In Brick, the formulation process is split into small building blocks, that a user needs to complete, to specify his information need.

- Similar theory as used for form-wizards.

Avoid information overload

- The risk of too many options...
  - Syntax: allow a minimal (logical) set of next steps.
  - Structure: present a limited (logical) set of structural elements. (LP: 271 unique elements)
Hypothesis

H1: Use of sophisticated query formulation techniques will lead to a higher effectiveness of the task performance. NEXI-CO and Bricks should perform significantly better than NEXI-CO, with respect to successful task completion. The difference in effectiveness is dependant on the task complexity.
Hypothesis

H2: The Bricks approach for query formulation will increase the efficiency of the user for a given task. When time is taken into account, with respect to effectivity, we expect that users will need significantly less time to (successfully) complete the task, when using NEXI-CO and Bricks.

Outline - Motivation - Objectives - Theory - Usability - Conclusions

Setup

- Document collection: Lonely Planet.
- Systems: NEXI-CO, NEXI-CAS, and Bricks.
- Users: 54 MIR-students, trained in NEXI and SDR.
- Topics: 27 topics, sorted in three complexity groups.
- Survey: prior and after the experiment the participants filled in a survey (satisfaction)
- Experience: Before the experiment the users practiced with the TERS-interface and search engines. (reduces learning effects)
**Overall results**

<table>
<thead>
<tr>
<th>System</th>
<th>Time</th>
<th>Effectivity</th>
<th>Efficiency</th>
<th>Satisfaction</th>
</tr>
</thead>
<tbody>
<tr>
<td>NEXI-CO</td>
<td>198</td>
<td>0.27</td>
<td>0.15</td>
<td>4.1</td>
</tr>
<tr>
<td>NEXI-CAS</td>
<td>245</td>
<td>0.34</td>
<td>0.14</td>
<td>4.7</td>
</tr>
<tr>
<td>Bricks</td>
<td>214</td>
<td>0.32</td>
<td>0.16</td>
<td>4.6</td>
</tr>
</tbody>
</table>

- **Effectivity**: significant diff. between Brick and NEXI-CAS vs. NEXI-CO. (H1)
- **Efficiency**: Bricks most efficient. (sign. diff.) (H2)
- **Time**: users need significantly more time to formulate information need with NEXI-CAS.
- **Satisfaction**: no significant difference, but NEXI-CAS preferred, followed by Bricks.

---

**Task complexity**

Outline - Motivation - Objectives - Theory - **Usability** - Conclusions

Centre for Content and Knowledge Engineering, Utrecht University, the Netherlands
Observations

- Completely different working procedures:
  - NEXI-CO: formulate query, inspect results and refine. (many iteration steps)
  - NEXI-CAS: step-wise construction and validation of NEXI-query, submissions to check syntax (few iteration steps)
  - Bricks: Longer construction time, almost no iterations, submit to check results.

Conclusions

- User need to be capable to adequately use the structure of a document, to make SDR work in practice.
- Three objectives for query formulation:
  - Adequate expression power
  - Minimize syntactical formulation problems
  - Minimize required knowledge of document structure
- Bricks: graphical approach, intuition of mental model, building blocks, avoid information overload. (Online-demo available)
Conclusions

- **Experiment:**
  - Sophisticated query formulation techniques have a positive influence of task performance (effectiveness of NEXI-CAS and Bricks).
  - Bricks is more efficient, since it allows the user to successfully perform their task in a shorter amount of time.
  - Task complexity vs. effectivity: negative correlation, but Bricks and NEXI-CAS are more effective for mid- and highly complex task.
  - Task complexity vs. efficiency: strong negative correlation.
    - NEXI-CAS is sensitive with respect to time needed for task completion.