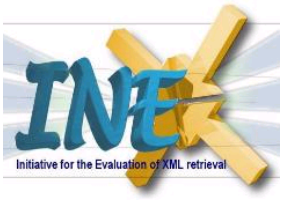

INEX 2006 Retrieval Task and Result Submission Specification

Charlie Clarke, Jaap Kamps, Mounia Lalmas

Saturday, May 20, 2006

1 Retrieval Task

The retrieval task to be performed by the participating groups of INEX 2006 is defined as the ad-hoc retrieval of XML elements. In information retrieval (IR) literature, ad-hoc retrieval is described as a simulation of how a library might be used, and it involves the searching of a static set of documents using a new set of topics. While the principle is the same, the difference for INEX is that the library consists of XML documents, the queries may contain both content and structural conditions and, in response to a query, arbitrary XML elements may be retrieved from the library.

The general aim of an IR system is to find *relevant information* for a given topic of request. In the case of XML retrieval there is, for each article containing relevant information, a choice from a whole hierarchy of different elements to return. Hence, within XML retrieval, we regard as *relevant elements* those XML elements that both

- contain relevant information (the element exhaustively discusses the topic), but
- do not contain non-relevant information (the element is specific for the topic).

That is, if an XML element contains another element but they have the same amount of relevant text, the shorter element is strictly more specific and a preferred result.

Within the ad-hoc XML retrieval task we define the following four sub-tasks:

1. THOROUGH TASK asks systems to estimate the relevance of elements in the collection.
2. FOCUSED TASK asks systems to return a ranked list of elements to the user.
3. RELEVANT IN CONTEXT TASK asks systems to return relevant elements clustered per article to the user.
4. BEST IN CONTEXT TASK asks systems to return articles with one best entry point to the user.

1.1 THOROUGH TASK

1.1.1 Motivation for the Task

The core system's task underlying most XML retrieval strategies is the ability to estimate the relevance of potentially retrievable elements in the collection. Hence, the INEX 2006 THOROUGH TASK simply asks systems to return elements ranked by their relevance to the topic of request. Since the retrieved elements are meant for further processing (either by a dedicated interface, or by other tools) there are no display-related assumptions nor user-related assumptions underlying the task.

What we hope to learn from this task is: How well are systems capable of estimating the relevance of XML elements? How well are systems capable of locating all the relevant elements in the collection? How do structural constraints in the query help retrieval?

1.1.2 Results to Return

The aim of the THOROUGH TASK is to find all relevant elements ranked in relevance order. It will be therefore the case that, due to the nature of relevance in XML retrieval (e.g. if a child element is relevant, so will be its parent, although to a greater or lesser extent), an XML retrieval system that has estimated an element to be relevant may decide to return all its ancestor elements. This means that runs for this task may contain a large number of overlapping elements. It is however a challenge to rank these elements appropriately.

Summarizing: THOROUGH TASK returns elements ranked in relevance order (where specificity is rewarded). Overlap is permitted.

1.2 FOCUSED TASK

1.2.1 Motivation for the Task

A continuation of the Focused retrieval strategy from INEX 2005, the scenario underlying the FOCUSED TASK is to return to the user a ranked-list of elements for her topic of request. The INEX 2006 FOCUSED TASK asks systems to find the most focused elements that satisfy a (focused) information need, without returning "overlapping" elements. That is, for a given topic, no element in the result set may contain text already contained in another element. Or, in terms of the XML tree, no element in the result set should be a child or descendant of another element. The task makes a number of assumptions:

Display the results are presented as a ranked-list of elements to the user.

Users view the result list top-down, one-by-one. Users do not want overlapping elements in the result-list, and prefer smaller elements over larger ones (if equally relevant). User is mostly concerned with what happens at the early ranks.

What we hope to learn from this task is: How does the user-oriented FOCUSED TASK differ from system-oriented THOROUGH TASK? Can the FOCUSED TASK be reduced to a straightforward filter on the THOROUGH TASK? What techniques are effective at the early ranks? How do structural constraints in the query help retrieval?

1.2.2 Results to Return

The aim of the FOCUSED TASK is to return a ranked-list of elements, where no element may be overlapping with any other element. Hence the decision to return a particular element to the user will outlaw all its ancestors, as well as all its descendants to be returned. Since all these ancestors and possibly also the descendants are relevant (be it to a lesser or greater extent) it is however a challenge to chose the elements appropriately. *Please note that submitted runs containing overlapping elements will be disqualified.*

Summarizing: FOCUSED TASK returns elements ranked in relevance order (where specificity is rewarded). Overlap is **not** permitted in the submitted run.

1.3 RELEVANT IN CONTEXT TASK

1.3.1 Motivation for the Task

The scenario underlying the RELEVANT IN CONTEXT TASK is to return the relevant information (captured by a set of elements) within the context of the full article. As

a result, an article devoted to the topic of request, will contain a lot of relevant information across many elements. The INEX 2006 RELEVANT IN CONTEXT TASK asks systems to find a set of elements that corresponds well to (all) relevant information in each article. The task make a number of assumptions:

Display results will be grouped per article, in their original document order, providing access through further navigational means.

Users consider the article as the most natural unit, and prefer an overview of relevance in their context.

What we hope to learn from this task is: How does the user-oriented RELEVANT IN CONTEXT TASK differ from THOROUGH TASK? What techniques are effective at locating relevance within articles? How do structural constraints in the query help retrieval?

The RELEVANT IN CONTEXT TASK is based on the INEX 2005 Fetch-And-Browse retrieval strategy.

1.3.2 Results to Return

The aim of the RELEVANT IN CONTEXT TASK is to first identify relevant articles (the fetching phase), and then to identify the relevant elements within the fetched articles (the browsing phase). In the fetching phase, articles should be ranked according to their topical relevance. In the browsing phase, we have a set of elements that cover the relevant information in the article. The `//article[1]` element itself need not be returned, but is implied by any result from a given article. Since the content of an element is fully contained in its parent element and ascendants, the set may **not** contain overlapping elements. *Please note that submitted runs containing results from interleaved articles will be disqualified, as will submitted runs containing overlapping elements.*

Summarizing: RELEVANT IN CONTEXT TASK returns a ranked list of articles. For each article, it returns an unranked **set** of elements, covering the relevant material in the article. Overlap is not permitted.

1.4 BEST IN CONTEXT TASK

1.4.1 Motivation for the Task

The scenario underlying the BEST IN CONTEXT TASK is to find the best-entry-point for starting to read articles with relevance. As a result, even an article completely devoted to the topic of request, will only have one best starting point to read. The INEX 2006 BEST IN CONTEXT TASK asks systems to find the XML elements that corresponds to these best-entry-points. The task make a number of assumptions:

Display single result per article.

Users consider the article as the most natural unit, and prefer to be guided to the best point to start to read the most relevant content.

What we hope to learn from this task is: How does the BEST IN CONTEXT TASK differ from the RELEVANT IN CONTEXT TASK? How do best-entry points relate to the relevance of elements (and THOROUGH TASK and FOCUSED TASK)? How do structural constraints in the query help retrieval?

The BEST IN CONTEXT TASK is also based on the INEX 2005 Fetch-And-Browse retrieval strategy.

1.4.2 Results to Return

The aim of the BEST IN CONTEXT TASK is to first identify relevant articles (the fetching phase), and then to identify the element corresponding to the best entry points for the fetched articles (the browsing phase). In the fetching phase, articles should be ranked according to their topical relevance. In the browsing phase, we have a single element whose opening tag corresponds to the best entry point for starting to read the relevant information in the article. Note that there is no implied end-point: if (the start of) a paragraph is returned, it's not indicating that the reader should stop at the end of the paragraph. The `//article[1]` element itself may be returned in case it is the best entry point, otherwise it will implied by any result from a given article. *Please note that submitted runs containing multiple results per article will be disqualified.*

Summarizing: BEST IN CONTEXT TASK returns a ranked list of articles. For each article, it returns a **single** element, representing the best entry point for the article with respect to the topic of request.

1.5 Structured Queries

Queries with content-only conditions (CO queries) are requests that ignore the document structure and contain only content related conditions, e.g. only specify what an element should be about without specifying what that component is. The need for this type of query for the evaluation of XML retrieval stems from the fact that users may not care about the structure of the result components or may not be familiar with the exact structure of the XML documents. CAS queries are more expressive topic statements that contain explicit references to the XML structure, and explicitly specify the contexts of the user's interest (e.g. target elements) and/or the context of certain search concepts (e.g. containment conditions). More precisely, a CAS query contains two kinds of structural constraints: where to look (i.e. the support elements), and what to return (i.e. the target elements). The structural constraints are considered as structural hints, and similar to CO queries the elements will be assessed using the ⟨`narrative`⟩ part of the topics. Runs using CO queries and runs using CAS queries will be merged to create the assessment pool (this will in fact improve the pool quality).

At INEX 2006. there is no separate CAS task, but the vast majority of topics have both a keyword CO query and a structured CAS query.[1] As noted above, for all the tasks, we want to find out if, when and how the structural constraints in the query have an impact on retrieval effectiveness. Although both types of queries may be used for each task, mixing runs with both query types, the best performing CAS query runs (restricted to topics containing a CAS query) will be reported. The use of CO/CAS query fields is recorded in submission format.

2 Result Submission

Fact sheet:

[1] Of course, any CO query can be directly rephrased as a CAS query `//*[about(.,"`*CO query*`")]` using the tag wildcard `*` that matches any element.

- For all four tasks, we allow up to 3 CO submissions, and up to 3 CAS submissions. That is, a participant can never submit more than 24 runs in total.
- All participants are required to submit a title-only run (free choice between the CO title and the CAS title) for the THOROUGH TASK.
- There is a common format for all submission files (details below), which allows up to 1,500 elements per topic.
- There are additional requirements on the submissions for three out of the four tasks:
  - FOCUSED TASK: for the same topic, results may not be overlapping.
  - RELEVANT IN CONTEXT TASK: articles may not be interleaved, and results may not be overlapping.
  - BEST IN CONTEXT TASK: only **one single** result per article is allowed.

  Runs that violate these requirements in any way, will be disqualified.

2.1 INEX 2006 Topics

There is only one set of topics to be used for all ad-hoc retrieval tasks at INEX 2006. The format of the topics is defined in the following DTD:

```
<!ELEMENT inex_topic
  (title,castitle?,description,narrative,ontopic_keywords)>
<!ATTLIST inex_topic
  id     CDATA #REQUIRED
  ct_no  CDATA #REQUIRED
>
<!ELEMENT title            (#PCDATA)>
<!ELEMENT castitle         (#PCDATA)>
<!ELEMENT parent           (#PCDATA)>
<!ELEMENT description      (#PCDATA)>
<!ELEMENT narrative        (#PCDATA)>
<!ELEMENT ontopic_keywords (#PCDATA)>
```

The submission format will record the precise topic fields that are used in a run. Participants are allowed to use all fields, but only runs using either the ⟨`title`⟩, ⟨`castitle`⟩, or ⟨`description`⟩ fields, or a combination of these, will be regarded as truly *automatic*, since the additional fields will not be available in operational settings.

The ⟨`title`⟩ part of the INEX 2006 topics should be used as queries for the CO submissions. The ⟨`castitle`⟩ part of the INEX 2006 topics should be used as queries for the CAS submissions. In the number of runs allowed to be submitted, runs using more fields than the ⟨`title`⟩ (or ⟨`castitle`⟩) will still be regarded as an CO (or CAS) submission.

Since the comparative analysis of CO and CAS queries is a main research question at INEX 2006, we encourage participant to submit runs using only the ⟨`title`⟩ field (CO query) or only the ⟨`castitle`⟩ field (CAS query). We do not outlaw the use of the other topic fields, to allow participants to conduct their own experiments involving them, and since such deviating runs may in fact improve the quality of the assessment pool.

2.2 Runs

There is an obligatory run, which is a submission to the THOROUGH TASK, using only the short topic statement from either the ⟨`title`⟩ or the ⟨`castitle`⟩ field of the topics.

For each of the four tasks, we allow up to 3 CO submissions, and up to 3 CAS submissions. The results of one run must be contained in one submission file (i.e. up to 24 files can be submitted in total). A submission may contain up to 1,500 retrieval results for each of the INEX topics included within that task.

There are however a number of additional task-specific requirements.

For the THOROUGH TASK there are no further restrictions.

For the FOCUSED TASK, it is not allowed to retrieve elements than contain text already retrieved by another element. That is, within the same article, the element `//article[1]//section[1]` is disjoint from `//article[1]//section[2]`, but overlapping with all ancestors (e.g., `//article[1]`) and all descendants (e.g., `//article[1]//section[1]//p[1]`).

For the RELEVANT IN CONTEXT TASK, articles may not be interleaved. That is, if an element of article `a` is retrieved, and then an element of a different article `b`, then it is not allowed to retrieve further elements from article `a`. Additionally, it is not allowed to retrieve elements than contain text already retrieved by another element (similar to the FOCUSED TASK). Note also that for this task the `//article[1]` element is implied by any element of the article, and need not be returned.

For the BEST IN CONTEXT TASK, only a single element per article is allowed. The `//article[1]` element may be returned in case it is regarded as the best place to start reading, otherwise it is implied by any other element from this article. Note that for this task, the `//article[1]` element may be returned in case it is regarded as the best element to start reading the relevant information in the article, otherwise it will be implied by any element of the article.

2.2.1 Submission format

For relevance assessments and the evaluation of the results we require submission files to be in the format described in this section. The submission format for all tasks is defined in the following DTD:

```
<!ELEMENT inex-submission (topic-fields, description, collections, topic+)>
<!ATTLIST inex-submission
  participant-id CDATA #REQUIRED
  run-id         CDATA #REQUIRED
  task           (Thorough | Focused | AllInContext | BestInContext ) #REQUIRED
  query          (automatic | manual) #REQUIRED
>
<!ELEMENT topic-fields EMPTY>
<!ATTLIST topic-fields
  title            (yes|no) #REQUIRED
  castitle         (yes|no) #REQUIRED
  description      (yes|no) #REQUIRED
  narrative        (yes|no) #REQUIRED
  ontopic_keywords (yes|no) #REQUIRED
>
<!ELEMENT description (#PCDATA)>
<!ELEMENT topic (result*)>
<!ATTLIST topic topic-id CDATA #REQUIRED >
<!ELEMENT collections (collection+)>
<!ELEMENT collection (#PCDATA)>
<!ELEMENT result (in?,file, path, rank?, rsv?)>
<!ELEMENT in (#PCDATA)>
<!ELEMENT file(#PCDATA)>
```

```
<!ELEMENT path (#PCDATA)>
<!ELEMENT rank (#PCDATA)>
<!ELEMENT rsv (#PCDATA)>
```

Each submission must contain the participant ID of the submitting institute (available at the INEX web-site `http://inex.is.informatik.uni-duisburg.de/2006/ShowParticipants.html`), a run ID (which must be unique for the submissions sent from one organization – also please use meaningful names as much as possible), the identification of the task (e.g. Thorough, Focused, etc), and the identification of whether the query was constructed automatically or manually from the topic. Furthermore, the used topic fields must be indicated in the ⟨`topic-fields`⟩ tag. Furthermore each submitted run must contain a description of the retrieval approach applied to generate the search results. A submission contains a number of topics, each identified by its topic ID (as provided with the topics).

For compatibility with the heterogeneous collection track, the ⟨`collections`⟩ tag is mandatory. There should be with ⟨`collections`⟩ at least one ⟨`collection`⟩ tag, which is by default set to "wikipedia" for the ad hoc track. The ⟨`in`⟩ tag is optional for the ad hoc track (⟨`in`⟩ states from which collection each result comes from).

For each topic a maximum of 1500 result elements may be included per task. A result element is described by a file name and an element path, and it may include rank and/or retrieval status value (rsv) information. For the ad hoc retrieval task, ⟨`collection`⟩ is set to "wikipedia". Here is a sample submission file for the THOROUGH TASK:

```
<inex-submission participant-id="12" run-id="VSM_Aggr_06"
  task="Thorough" query="automatic">
  <topic-fields title="no" castitle="yes" description="no"
    narrative="no" ontopic_keywords="no"/>
  <description>Using VSM to compute RSV at leaf level combined with
   aggregation at retrieval time, assuming independence and using
   augmentation weight=0.6.</description>
  <collections>
    <collection>wikipedia</collection>
  </collections>
  <topic topic-id="01">
    <result>
      <file>9996</file>
      <path>/article[1]</path>
      <rsv>0.67</rsv>
    </result>
    <result>
      <file>9996</file>
      <path>/article[1]/name[1]</path>
      <rsv>0.1</rsv>
    </result>
    [ ... ]
  </topic>
  <topic topic-id="02">
    [ ... ]
  </topic>
  [ ... ]
</inex-submission>
```

Rank and RSV The rank and rsv elements are provided for submissions based on a retrieval approach producing ranked output. The ranking of the result elements can be described in terms of:

- Rank values, which are consecutive natural numbers, starting with 1. Note that there can be more than one element per rank.
- Retrieval status values (RSVs), which are positive real numbers. Note that there may be several elements having the same RSV value.

Either of these methods may be used to describe the ranking within a submission. If both rank and rsv are given, the rank value is used for evaluation. These elements may be omitted from a submission if a retrieval approach does not produce ranked output.

File and path Since XML retrieval approaches may return arbitrary XML nodes from the documents of the INEX collection, we need a way to identify these nodes without ambiguity. Within INEX submissions, elements are identified by means of a file name and an element (node) path specification, which must be given in XPath syntax. The file names in the Wikipedia collection uniquely define an article, so there is no need for including the directory in which the file resides (in contrast with the earlier IEEE collection). The extension .xml must be left out. Example:

```
9996
```

Element paths are given in XPath syntax. To be more precise, only fully specified paths are allowed, as described by the following grammar:

Path ::= '/' *ElementNode Path* | '/' *ElementNode* | '/' *AttributeNode*
ElementNode ::= ElementName *Index*
AttributeNode ::= '@' AttributeName
Index ::= '[' integer ']'

Example:

```
/article[1]/body[1]/section[2]/p[1]
```

This path identifies the element which can be found if we start at the document root, select the first "article" element, then within that, select the first "body" element, within which we select the second "section" element, and finally within that element we select the first "p" element. Important: XPath counts elements starting with 1 and takes into account the element type, e.g. if a section had a title and two paragraphs then their paths would be given as: ../title[1], ../p[1] and ../p[2].

A result element may then be identified unambiguously using the combination of its file name and element path. Example:

```
<result>
  <in>wikipedia</in>
  <file>9996</file>
  <path>/article[1]/body[1]/section[2]/p[1]</path>
</result>
```

2.3 Result Submission Procedure

To submit a run, please use the following link: `http://inex.is.informatik.uni-duisburg.de/2006/` Then go to Tasks/Tracks, Adhoc, Submissions. The online submission tool will be available soon.