ITR5 Information Usage

Lecture 6

Thomas Krichel

2002-02-19

Reading

"Information Architecture" by Louis Rosenfeld and Peter Morville, O'Reilly 1998

chapter 6 (not worth reading)

Paul J. Lucas' swish++ homepage at
http://homepage.mac.com/pauljlucas/software/swish/

how much to display results

- display less information when results set is large
- if there is a common element in the answer, we need to show more for the user to see the difference
- allow user to choose
how much to display results

- display less information when results set is large
- if there is a common element in the answer, we need to show more for the user to see the difference
- always tell the user how many results there are
- allow user to choose

what to display for each document

- display less information when results set is large
- if there is a common element in the answer, we need to show more for the user to see the difference
- allow user to choose

other display options

what to display for each document

- depends on structure of data

how many items?

- allow user to choose ?!
relevance order

- (reverse) chronological
- calculated relevance
  - let user know algorithm
- alphabetical order (no relevance)

relevance order calculation

depend on software,
- how many of the query terms occur
- how many times the query terms occur
  - suggest to try another search
  - suggest to read search help
  - suggest to browse
- how close they occur
- if they occur at start or end of the document

other advice
- repeat search data on results page,
- repeat search result and within its <title> tag, for bookmarks
- special page for empty results
- repeat original search on results page
- say how many documents were retrieved
- let user know where she is when browsing the retrieved set
- make it easy to revise search
what to index

- entire site
- search zones
  - by type
  - by audience
  - subject
  - date

features of the swish++ engine

1. Lightning-fast indexing and searching.

2. Indexes META elements ALT, and other attributes. For
   HTML or XHTML files, SWISH++ indexes words in META ele-
   ment CONTENT attributes and associates them with the NAME
   attributes. Meta names can later be queried against specifically,
   e.g.: search author = hawking

3. Indexing other attributes
   SWISH++ also indexes the words in ALT attributes (for the
   AREA, IMG, and INPUT elements), STANDBY attributes (for
   the OBJECT element), SUMMARY attributes (for the TABLE
   element), and TITLE attributes (for any HTML or XHTML ele-
   ement).

4. Selectively not index text within HTML or XHTML elements
   Text within HTML or XHTML elements belonging to specified
   classes can be not indexed. This is most useful not to index text
   in common page headers, footers, and pop-up menus.

5. Apply filters to files on-the-fly prior to indexing
   Based on filename patterns, files can be filtered before being in-
   dexed, e.g.: compressed files uncompressed, PDF files converted
   to plain text, etc.

6. Index non-text files such as Microsoft Office documents
   A separate text-extraction utility “extract” is included.

7. Index new files incrementally
   New files can be indexed and added to an existing index incre-
   mentally.
8. Index remote web sites
A separate utility "httpindex" is included that interfaces SWISH++
to the wget(1) command enabling remote web sites to be in-
dexed.

9. Handles large collections of files
SWISH++ automatically splits and merges partial indices for
large collections of files.

10. Optional word stemming (suffix stripping) SWISH++ allows
stemming to be performed at the time of searches, not at the
time of index generation. This allows users to decide whether to
perform stemming or not.