Generating advanced query interfaces

Dongwon Lee[a], Divesh Srivastava[b1] and Dimitra Vista[b2]

[a]*4802 Boelter Hall,*
UCLA, Los Angeles, U.S.A.
dongwon@cs.ucla.edu

[b]*AT&T Labs-Research,*
180 Park Ave, Rm A005[1]*, Rm B155*[2]*,*
Florham Park, NJ, 07932-0971 U.S.A.
divesh@research.att.com and vista@research.att.com

Keywords
Interface generator; Complex selection querying

Many information sources available on the World Wide Web support query interfaces to allow users to select subsets of their data that are of interest to them. Examples of such systems are numerous: Web search engines, white and yellow pages, special purpose databases with articles, patents, movies, stock quotes, flights, and so on. Many of these systems allow access to their data from multiple query interfaces typically including (a) an interface based on the vector-space model [2] for simple queries using limited keyword searching, and (b) an interface for specifying complex selection criteria on the data to be retrieved. While simple queries are very easy to specify, writing complex queries is much harder. For such complex queries, most systems require the user to be aware of their specific, often proprietary, interfaces and query language syntaxes. *We believe that being able to query these systems in a uniform and consistent way could greatly improve the user's experience of interacting with them.*

We have designed and implemented an interface generator for constructing advanced query interfaces that allow the specification of complex queries. Generated interfaces to different information sources share a consistent look and feel, removing the burden from users of having to know the interface and query language specifics of the individual information sources. The tool accepts as input a high-level specification of the interface and produces as output its implementation.

There are two aspects to each visual interface generated by our tool. One has to do with the layout of the various visual components in the interface and the way they interact in response to user actions. The other has to do with the way the interface supports the language interactions with the back-end system. This interaction includes the mapping from the queries constructed in the visual interface to queries supported by the underlying system, and the mapping from individual query elements of the underlying language to visual elements of the interface. Since our goal in devising the generator was not to study which visual layout is best suited for our target applications, all interfaces generated by our tool share the same layout. The tool, however, provides great flexibility with respect to language interactions, and the current implementation supports a large number of customizable options.

Fig. 1. (a) HotBot and (b) Alta Vista queries requesting all documents mentioning both the word `Intel` and the phrase `Pentium Chip` so long as they are not located in the `intel.com` site.

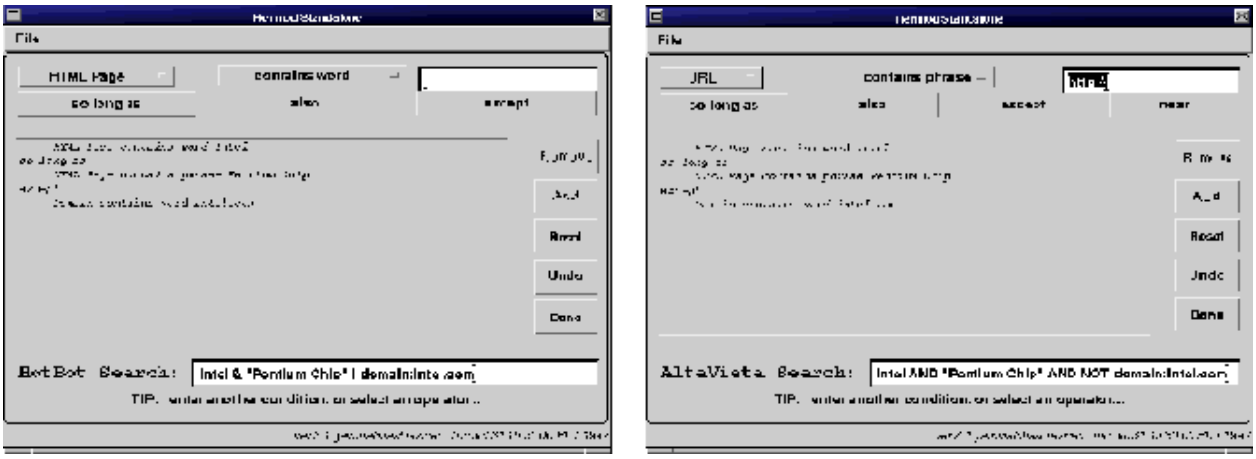

Figure 1 shows two typical interfaces generated by the tool. It displays a complex query for the HotBot (Fig. 1a) and AltaVista (Fig. 1b) search engines requesting all documents mentioning both the word `Intel` and the phrase `Pentium Chip` so long as they are not located in the `intel.com` site. The upper part of the layout consists of a pop-up menu of attribute names (i.e.,'html page'), a pop-up menu of attribute operators (i.e., 'contains word') and a place holder for entering the value of an atomic query. The horizontal palette of buttons underneath corresponds to the logical connectives for complex querying. The vertical palette of buttons to the right allow general editing of queries (shared by all generated interfaces). The central textual component displays an (editable) English language description of the query. The (optional) lower textual component displays the query in the syntax of the underlying system. Our rationale for choosing this particular interface is documented in the full version of the paper [1].

For flexibility, the input configuration file to our tool permits specifications of how to translate attributes, attribute operators, and (possibly) values, as well as complex queries. In particular, in the configuration file:

- We specify the (non-empty) set of attribute names to appear in the attribute menu. For each attribute, we specify how the attribute is visually displayed, and how it is translated into the query language of the underlying system.
- We might specify a default value for an attribute to appear in the place holder for the value when the attribute is chosen from the attribute menu.
- We specify the types of attribute values allowed for an attribute in order to facilitate the generation of interfaces with embedded type checking capabilities.
- We specify the (non-empty) set of attribute operators for the attribute operator menu. For each attribute operator, we specify how it is visually displayed and how it is translated (together with its arguments) into the query language of the underlying system.
- We specify which attribute operators are applicable to each attribute.
- We specify the set of logical operators for the horizontal operator palette. For each logical operator we specify how the operator is visually displayed and how it is translated (together with its arguments) into the query language of the underlying system.

We have demonstrated the utility of using a generator tool to construct advanced visual query interfaces for a variety of information sources available on the Web: the AltaVista, HotBot, Lycos and Infoseek search engines; the IBM patent server; the DejaNews newgroups; and the BigFoot and Four11

directories. Our generator tool can be flexibly configured to support a variety of selection-based query languages and to construct interfaces with a uniform look and feel. Our examples demonstrate that we can support consistent interfaces very easily by specifying a simple user configuration file. For more details see [1].

References

1 D. Lee, D. Srivastava, and D. Vista,
Generating advanced querying interfaces,
AT&T Technical Report, 1998.

2 G. Salton,
Automatic Text Processing: the Transformation, Analysis and Retrieval of Information by Computer.
Addison Wesley, 1989.