H. Oberquelle, R. Oppermann, J. Krause (Hrsg.): Mensch & Computer 2001: 1. Fachübergreifende Konferenz. Stuttgart: B.G. Teubner; 2001, S. 317-326

Enhancing Dynamic Queries and Query Previews: Integrating Retrieval and Review of Results within one Visualization

Maximilian Stempfhuber, Bernd Hermes, Luca Demicheli, Carlo Lavalle
Informationszentrum Sozialwissenschaften (IZ), Bonn /
European Commission, DG Joint Research Centre, Space Applications Institute Strategy and Systems for Space Applications (SSSA) Unit

Abstract

The concepts of dynamic queries and query previews have shown to be useful in information systems because they allow users to search for information by directly manipulating a visual representation of the query and getting immediate and continuous feedback about the results. To enhance dynamic queries and query previews we combined them with dynamic screen layout for query formulation and a visual formalism for result presentation in a single screen. The user can refine his query and get an immediate preview of the results. Manipulating the query preview in turn directly modifies the visualization of the results without the need to switch back to a separate query screen. The combination of query, preview and result display gives us the opportunity to visualize dependencies between values of search attributes, which standard query previews do not show. The application domain is MURBANDY, a system for monitoring and modelling the change of land use in European cities.

1 Introduction

Most database systems require the user to formulate queries in high level query languages, which presumes that the user is familiar with the database structure and the query language. For relational database management systems (RDBMS) there is a *formal query language*, SQL (Structured Query Language), where conditions can be specified for filtering records. Single conditions can be combined by Boolean operators (AND, OR, NOT) to more complex terms. Boolean logic has shown to be difficult to comprehend, mostly because the operators' semantics do not exactly match their semantics in natural language, precedence is not always clear and errors are made when using brackets (Green et al. 1990, Hertzum&Frøkjær 1996).

To avoid learning a formal query language, systems like VQuery (Michard 1982), AI-STARS (Anick et al. 1990), Filter/Flow (Young&Shneiderman 1993) or DEViD (Eibl 1999) have been developed. They all try to *visualize Boolean logic* (often with reduced complexity) and apply their solutions to document or fact retrieval. In most of the cases they prove to be superior to formal query languages in regard to the time necessary for task completion or the number of user errors, but they normally separate query formulation from result display.

Dynamic queries (Ahlberg et al. 1992) and *query previews* (Doan et al. 1996) combine query formulation and display of results at the spatial and temporal level. Systems like the Dynamic HomeFinder (Williamson&Shneiderman 1992) use direct manipulation of search attributes through user interface controls instead of a formal query language. Combined with adequate visualizations, this allows fast and reversible operations whose impact on the query result is immediately visible. To improve query performance in networked information systems, where long response times can reduce the benefits of dynamic queries, query previews (Greene et al. 1999) use metadata to calculate the size of the result set in advance. Now the user is able to explore the data and refine his query before it is sent to the server.

2 Dynamic Screen Layout and Visual Formalisms

Syntax and layout of graphical user interfaces often imply a combination of the individual controls (e.g. sliders or entry fields which represent the search attributes) with Boolean operators. Since the user expects the result set to shrink the more specific his query is (the more attributes he specifies), the attributes will be combined with the AND operator.

Problems arise as soon as an indefinite number of values per search attribute can be entered, which can be multiple numerical ranges, search terms or selections from lists:

- The number of values needed is hard to define a priori. Providing too few entry fields limits the complexity of the user's query, having too many wastes screen real estate.
- Sometimes attribute values have to be selected by the user rather than entered, because entering them is error-prone or too difficult for the casual user. In this situation there is a race condition between the space needed for presenting options (e.g. list boxes) and summarizing previous selections to lower the short term memory's load (status display).
- As soon as there is no simple interpretation of how the attributes and their values are transformed into a (Boolean) query, e.g. values are combined with OR within attributes and with AND between them, the result of the query is hard to guess in advance.
- Even a query preview or the final result set may not be able to explain relations in the data, so that the user can not easily understand how search attributes and their values interact with each other and influence the result in detail.

Dynamic screen layout and tight coupling

The dynamic spatial layout of controls on the screen solves the problems related with multi-valued search attributes by adjusting the size of controls (e.g. entry fields) and rearranging them in a way so that they can hold an arbitrary number of values without wasting space that otherwise could be used for selection lists or a status display (Stempfhuber 1999). We developed a control for entering text which initially consists of one single entry field and grows or shrinks vertically with every entry that is added or deleted. Space on the screen is dynamically occupied or freed depending on the amount of data the user enters. Figure 1 show parts of the user interface of ELVIRA, an information system for market researchers, to illustrate the idea.

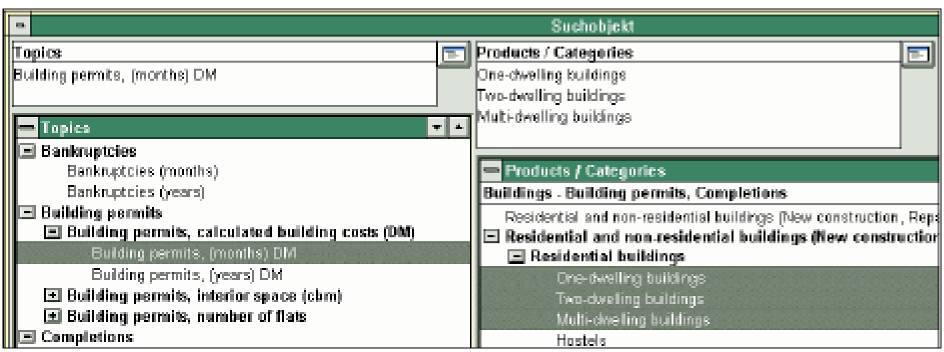


Figure 1: Tight coupling of dynamic entry fields with selection lists (Stempfhuber 1999)

To allow selection rather than manual input (recognition vs. recall), the entry field is tightly coupled with a selection list. The content of the selection list serves two functions:

- Validation of (manual) input from the user: the entry field only accepts input which can also be found in the list, but assists the user with a thesaurus.

- Display of choices: to avoid typing errors, search terms can be selected from the list instead of being typed into the entry field.

The user can freely choose it's mode of interaction. Entering text in the entry field selects the corresponding item in the list and selecting in the list inserts the text in the entry field (same for deleting items). This tight coupling of controls together with the synchronized flow of information between them are two of the core principles of the WOB model for user interface design (Krause 1997). Both principles can directly be mapped to software design patterns and thus are clearly specified at the conceptual and the implementation level (Stempfhuber 2000).

Tight coupling is also used to synchronize permissible values of multiple search attributes and therefore to eliminate combinations of values that will lead to zero-hit queries. In figure 1 the attributes „topics" and „products / categories" are synchronized in a way that when a topic is selected, all entries from „products / categories" are removed that can not be combined with the selected topic. The dependencies between attributes are bi-directional (but not recursive) so that initially selecting a product will reduce available topics.

Status Display to Reduce Memory Load

The entry field at the same time serves as a status display. This is necessary, because the lists of attribute values tend to become rather long in real-life applications (up to 1.800 entries in some of the many nomenclatures used in ELVIRA). Scrolling those hierarchically organized lists always hides parts of them, so that the user has to remember a potentially large number of already selected items. The status display summarizes the selected items, gives a comprehensible overview of the query and at the same time reduces short term memory load.

When displaying query results, the selections lists are replaced by a result list (figure 2), while the status display still remains visible. Again, there is no need to remember the query when reviewing results and comparing them with the search conditions.

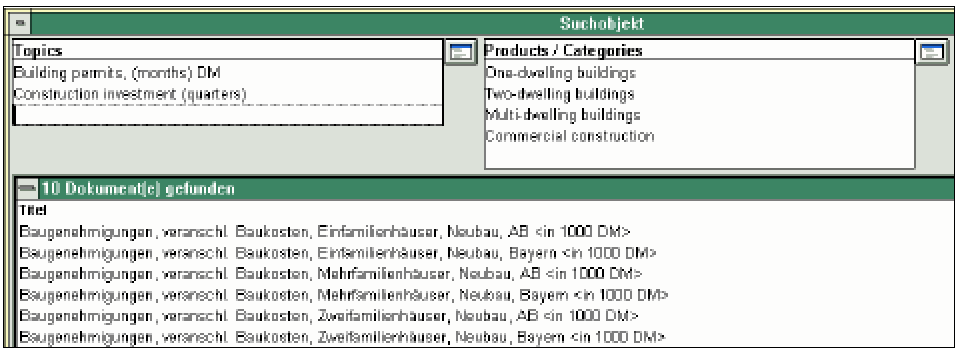


Figure 2: Status Display integrated into result screen

Query refinement in a one-screen-system

Another problem in information systems is support for iterative query refinement. Many systems divide query formulation and presentation of results into separate screens, which makes it not only difficult to compare query and results. It also forces the user to first develop a strategy for re-formulating the query while looking at the results (query form not visible) and then switching back to the proper screen and modifying the query (results not visible).

The solution is again to use an enhanced status display that is visible together with the query result and allows modification of the query without switching screens.

Using visual formalisms to explain query results

One weakness of the so far proposed solutions and systems presented is, that there is no direct mapping of the dependencies between query attributes or their values and the elements of the result set. Because the values of attributes are often combined with the OR operator and the attributes are combined with AND, query previews give only the total of records in the result set, but may fail to visualize which combinations of search values appear in the result set.

For the combination of two attributes with a theoretically unlimited number of attribute values we developed an interactive visualization based on the idea of visual formalisms (Nardi&Zarmer 1993). Visual formalisms are basic visualizations like maps, tables or charts, which everyone learns to read or use and which do not require knowledge or transfer of additional concepts. Metaphors, in contrast, require the user to transfer a known concept (e.g. that of a type writer) into a different domain (e.g. text processing software), which can break the metaphor and lead to difficulties in comprehension or user errors.

Attribute	B1	B2	B3	
A1	☒		☐	Σ
A2	☐	☒		
A3	☒	☐	☒	
	Σ			

Figure 3: Tabular query preview which can be modified for query refinement

Our visualization uses a two-dimensional table, where each axis denotes one attribute (figure 3). The table dynamically grows or shrinks depending on the number of values specified for each attribute. The cells of the table contain one of three possible values:

- An empty cell (null-value) signals that there is no dependency between the corresponding two attribute values.
- An unmarked checkbox denotes a dependency in the data which the user can activate (mark the checkbox by clicking the mouse) to include it in the result set.
- A marked checkbox denotes an activated dependency that will add a number of additional records to the result set.

To give the user information about how much the result set will grow when activating a checkbox, the number of records can be displayed in each cell as an option. In addition, totals for rows or columns may be useful to analyse the distribution of attributes within the database.

3 Information Retrieval in the Context of Urban Planning

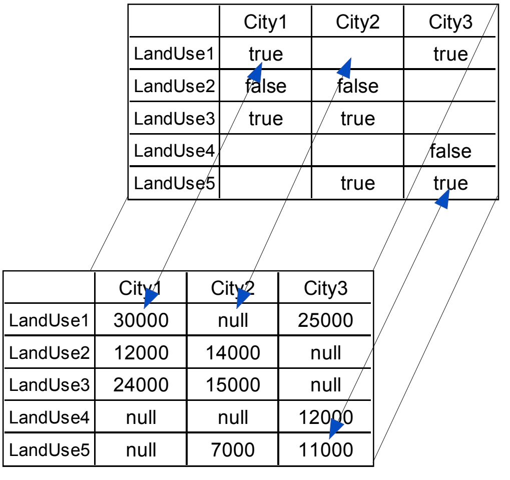
For urban planning models are needed that integrate historical and current data and allow to prognosticate future development. In the project MURBANDY (Monitoring Urban Dynamics, (Lavalle&Demicheli 2000), methods for monitoring urban dynamics of European cities are developed and indicators are created, which make these dynamics and the influence on cities' peripherals understandable (see section 5).

Typical information needs

An important criteria for urban planning is the change of land use and the consumption of land in urban areas. Therefore land use data for 25 European cities has been collected and classified for four different years between 1955 and 1990. Besides the satellite images, from which part of the data has been extracted, there are coloured maps for each city and year that show land use and traffic network according to the international land use classification.

Common use cases for this data are the analysis of the change of land use for a single city within a specific period or the comparison of two cities with regard to certain classes of land use, e.g. industrial areas. In rare cases more than two cities will be compared at once. The users of the MURBANDY WWW Interface will be researchers concerned with environmental change and urban planning. It will also assist decision makers at levels from local to European government. The requirements for the prototype have been derived from previous cooperations between the MURBANDY project team at the Space Applications Institute (SAI) of the Joint Research Centre (JRC) Ispra, Italy, of the European Commision and the targeted user group. The project team at the German Social Science Information Centre (IZ), Bonn, developed alternative designs for the user interface that were peer reviewed by the SAI.

Layer 2: Use combination in result view



	City1	City2	City3
LandUse1	true		true
LandUse2	false	false	
LandUse3	true	true	
LandUse4			false
LandUse5		true	true

	City1	City2	City3
LandUse1	30000	null	25000
LandUse2	12000	14000	null
LandUse3	24000	15000	null
LandUse4	null	null	12000
LandUse5	null	7000	11000

Layer 1: Landuse values for cities

Figure 4: Different layers for viewing and manipulating queries

Database schema and retrieval

Let there be identifiers $c_1,...,c_n$ for the cities, $l_1,...,l_m$ for the types of land use and let $y_1,...,y_k$ be the years, when data has been recorded. Furthermore let the schema for storing the data be:

$$\text{LandUseValues}: \{LandUse : Integer, City : Integer, Year : Integer, Value : Integer\}$$

The query to retrieve the types of land use l_s, ..., l_t and their area for a city c_i in a year y_p is:

$$B_i = \prod_{LandUse,Value} \sigma_{City=c_1 \wedge Year=y_p \wedge (LandUse=l_s \vee \ldots LandUse=l_t)} (LandUseValues))$$

To retrieve data for a combination of multiple cities and types of land use, e.g. for later comparison, the intermediary results have to be joined.The result of the join can be displayed as a table. In figure 4, layer 1 contains the results of the database query. It has to be kept in mind, that the user might not be interested in every possible combination of cities and types of land use for which data is found. To allow iterative refinement of the query, the user must be able to (permanently or temporarily) remove certain *combinations* of cities and types of land use from the result set. Completely removing a city or a type of land use from the query would not give the granularity needed for further analysis or comparison. Layer 2 in figure 4 consists of a secondary filter for the result set, where data points (non-null values) from layer 1 can be switched on and off with a boolean flag. Both layers comprise the preview metadata.

4 The MURBANDY User Interface

In MURBANDY, queries can be formulated with two entry fields - serving at the same time as status displays – together with tightly coupled selection lists (figure 5). One pair of entry field and selection list can be used to specify the types of land use and the other one for the cities that should be analysed or compared. The selection list for the types of land use is displayed as a hierarchy, the cities are listed alphabetically and as a map. The names of cities can either be typed into the entry field, selected in the selection list or marked in the map. Again, there is tight coupling between entry field, selection list, and map so that the user can freely change his mode of interaction.

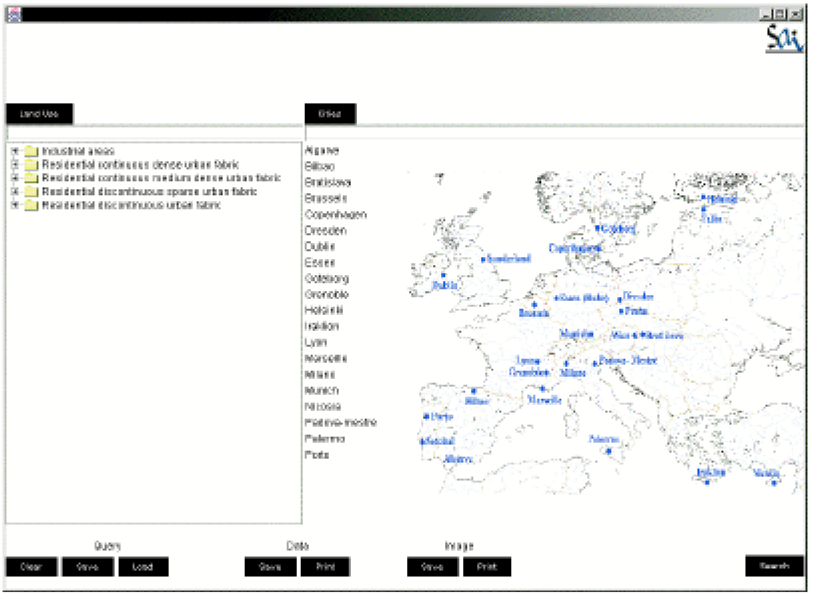


Figure 5: Initial user interface

When specifying search values, the height of the entry fields and their corresponding selection lists are dynamically adjusted (figure 6). This reduces the space needed for presentation of alternatives (which the user has already seen), and provides additional space for the status display to reduce short term memory load. At the same time, the status display allows deselecting search values without locating them again in the selection list.

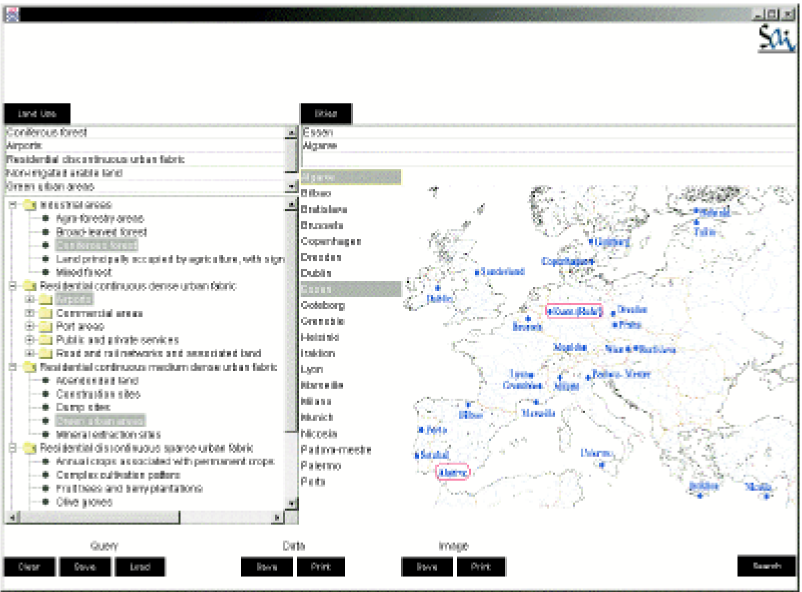


Figure 6: Dynamically adjusted entry fields and selection lists

To omit zero-hit queries, the attributes „land use“ and „cities“ are tightly coupled, so that the selection of a type of land use reduces the list of cities and vice verse. The direction for adaptation is determined by the attribute which is selected first. If a type of land use is selected, only the list of cities will adapt and will not in turn lead to a adaptation of the list with the types of land use. This avoids recursion and therefore confusion of the user.

Though a query preview is calculated behind the scenes while the query is formulated, we chose to use a „Search“ button to let the user state that he believes his information needs will be met. When the query is submitted, the two selection lists are hidden and the entry field for the cities is rotated by 45°, giving room for a tabular display of the query preview. The system can be configured to automatically retrieve and display the data, as seen in figure 7.

Both entry fields now serve as a status display, which can be modified by typing in or deleting search values or by clicking one of the labels „land use“ or „cities“ to show the selection lists again. For experienced users, this could be the only screen they work with, because it contains space saving ways of entering search values, displays preview data and visualizes the query result as map at the same time.

A cell in the preview table is empty, if no data exists for the combination of land use and city. As a default, all existing combinations (marked by a checkbox) are activated (checked) so that they are displayed in the map. Removing a check hides the land use in the corresponding map of the city. Deleting the land use from the status display changes the query eleminating the line from the preview table and therefore hides the land use in all of the maps.

Working with MURBANDY is a two-step process of search (selecting types of land use and cities) and refinement (filtering the result set), which can have an arbitrary number of iterations without switching screens. Even the analysis of the results (maps) is tightly integrated and allows query re-formulation if additional data is needed.

Figure 7: Displaying the result of the query

6 Application of MURBANDY

Urban sprawl is a complex process that derives from distinctive geographic, demographic and economic circumstances. The most evident effect of sprawling, which leads to extensive patterns of peripheral and suburban development, is perhaps land use change and consumption. Natural and cultural landscape features are highly threatened, and natural resource consumption and depletion occur. Moreover cities appropriate a large amount of carrying capacity, consuming resources from areas even very far away, and endangering the economic and natural equilibrium of the planet. The understanding of urban dynamics is, therefore, one of the most complex tasks in planning the sustainable development of large-scale economies. The complexity and variety of the different urban components, and of the interactions amongst them, is even more pronounced when available mapping is outdated or very poor, and where there is a general lack of standard and comparable information on cities. The current situation asks for new technologies, tools and expertise to better monitor and understand those composite environments. In this framework, the uniform monitoring of the distribution, changing patterns, and growth of human settlements, plays a very important role, and a lot of research is currently focused on the development of new methodologies based on high-technology tools.

In 1998, under the umbrella of activities carried out by CEO, a pilot study named MURBANDY (Monitoring Urban Dynamics) was launched. It initially aimed at providing a measure of the extent of urban areas, as well as of their progress towards sustainability, through the creation of land use databases for various cities. To date, such a database has been created for twenty-five European cities that have been classified with a single land use classification scheme in order to obtain homogeneous data. The study has now been extended to cover seven „mega-cities“ outside Europe, and other areas are under consideration. The data were derived from satellite imagery and aerial photography, using remote sensing and GIS technologies. The database is the basis for combining environmental, economic and social data, in order to better understand dynamics and characteristics of the urban growth and related structural changes, commuting issues, and status of transport and energy infrastructures.

One of the advantages of this approach is the multi-temporal dimension of the resulting data sets, which are produced for four dates over the past fifty years, thereby enabling time-series analysis. Also, the information is collected and elaborated with precisely the same methodology in the different cities, so as to allow comparative analyses. Thus, the approach enables the analysis of each single city as a complex urban system, and facilitates comparisons among cities, by providing comprehensive, standard and homogeneous information about the areas assessed.

The MURBANDY user interface will allow scientists to retrieve and display the data collected within the project over the Internet. Additional modules for analysis of the data may be provided along with project progress and user's demands.

7 Conclusions

The user interface presented in this paper enhances dynamic queries and query previews with dynamic screen layout, tight coupling and visual formalisms. Some of these concepts have already proven to be superior to alternative user interface designs. The combination of concepts in MURBANDY has been pre-evaluated with domain experts in a heuristic user test. Tests with novice users are in progress and will be reported in the presentation of the paper.

References

1. Ahlberg, C.; Williamson, C.; Shneiderman, B. (1992). *Dynamic Queries for Information Exploration: An Implementation and Evaluation.* CHI'92 Conference on Human Factors in Computing Systems, Monterey, CA United States, 1992. pp. 619-626.
2. Anick, P.G.; Brennan, J.D.; Flynn, R. A.; Hanssen, D.R.; Alvey, B.; Robbins, J.M. (1990). *A direct manipulation interface for boolean information retrieval via natural language query.* In: Proceedings of the thirteenth International Conference on Research and Development in Information Retrieval, September 5 - 7, 1990, Brussels, Belgium. pp. 135-150.
3. Doan, K.; Plaisant, C.; Shneiderman, B. (1996). *Query previews in networked information systems*. In: Proceedings of the Third Forum on Research and Technology Advances in Digital Libraries, ADL '96, Washington, DC, May 13-15, 1996. IEEE CS Press, 1996.
4. Eibl, M. (1999). *Visualisierung im Dokument Retrieval. Theoretische und praktische Zusammenführung von Softwareergonomie und Grafik Design.* Dissertation im Fachbereich für Informatik an der Universität Koblenz-Landau.
5. Green, S.L.; Devlin, S.J.; Cannata, P.E.; Gomez, L.M. (1990). *No Ifs, ANDs, or Ors: A study of database querying.* In: International Journal of Man-Machine Studies. Vol. 32, pp. 303-326.
6. Greene, S.; Tanin, E.; Plaisant, C.; Shneiderman, B.; Olsen, L.; Major, G.; Johns, S. (1999). *The end of zero-hit queries: query previews for NASA's Global Change Master Directory*. In: International Journal on Digital Libraries. Nr. 2, pp 79-90.
7. Hertzum, M.; Frøkjær, E. (1996). *Browsing and querying in online documentation: a study of user interfaces and the interaction process.* In: ACM Transactions on Computer-Human Interaction. Vol. 3, No. 2, pp. 136-161.
8. Krause, J. (1997). *Das WOB-Modell*. In. Krause, Jürgen; Womser-Hacker, Christa (1997). *Vages Information Retrieval und graphische Benutzungsoberflächen: Beispiel Werkstoffinformation*. Konstanz. Schriften zur Informationswissenschaft Bd. 28, pp. 59-88.
9. Lavalle C., Demicheli L., Casals Carrasco P., Turchini M.,Niederhuber M.,McCormick N. (2000). *Murbandy /Moland* Technical Report European Commission Euroreport. (In press).
10. Michard, A. (1982). *Graphical presentation of boolean expressions in a database query language. Design notes and ergonomic evaluation.* Behaviour & Information Technology. Vol. 1, No. 13, pp. 279-288.
11. Nardi, B.; Zarmer, C. (1993). *Beyond Models and Metaphors: Visual Formalisms in User Interface Design*. In: Journal of Visual Languages and Computing, 1993, 4, pp. 5-33.

12. Stempfhuber, M. (1999). *Dynamic spatial layout in graphical user interfaces*. In: Bullinger, Hans-Jörg; Ziegler, Jürgen (1999). Human-Computer Interaction: Communication, Cooperation, and Application Design. Proceedings of HCI International '99, Munich, Germany, August 22-26, 1999. Vol. 2, pp. 137-141.
13. Stempfhuber, M. (2001). *ODIN - Objektorientierte grafische Benutzungsoberflächen*. Dissertation im Fachbereich für Informatik an der Universität Koblenz-Landau (to appear).
14. Williamson, C.; Shneiderman, B. (1992). *The Dynamic HomeFinder: Evaluating Dynamic Queries in a Real-Estate Information Exploration System*. ACM SIGIR'92. pp. 338-346.
15. Young, D.; Shneiderman, B. (1993). *A Graphical Filter/Flow Representation of Boolean Queries: A Prototype Implementaion and Evaluation.* In: JASIS. Vol. 44, No. 6, pp. 327-339.

Adressen der Autoren

Maximilian Stempfhuber / Bernd Hermes
Informationszentrum
Sozialwissenschaften (IZ)
Lennéstr. 30
53113 Bonn
st@bonn.iz-soz.de
hb@bonn.iz-soz.de

Luca Demicheli / Carlo Lavalle
European Commission
DG Joint Research Centre
Space Applications Institute
Strategy and Systems for Space Applications
(SSSA) Unit
TP 261, Ispra (VA), I-21020, Italy
luca.demicheli@jrc.it
carlo.lavalle@jrc.it