Spotfire: An Information Exploration Environment

Christopher Ahlberg
IVEE Development AB, Chalmers University of Technology & SSKKII
Stora Badhusgatan 18-20
S-411 21 Göteborg, Sweden
Email: ahlberg@ivee.com
WWW: http://www.ivee.com
Phone: +46 31 7014264, Fax: +46 31 101987

Introduction

Spotfire is a database exploration system based on interactive information visualization, dynamic queries, brushing & linking, and other interactive graphics techniques. Spotfire is an embodiment of many of the ideas grown out of the research on dynamic queries described in papers such as (Ahlberg, Williamson & Shneiderman, 1992)(Ahlberg & Shneiderman, 1994a)(Ahlberg & Shneiderman, 1994b)(Ahlberg & Wistrand, 1995).

The cognitive load while performing information retrieval, exploration, and analysis tasks is usually high. Allowing users to incrementally control animated visualizations of databases, queries, and query results can minimize the mental effort needed for:

- Finding effective database query formulations
- Grasping relations between queries and query results
- Judging relations between individual query results.

Benefits of animated visualizations partially stem from the fact that we perform many perceptual tasks with little conscious mental effort. Recently the use of these remarkable perceptual abilities has found its way into information exploration systems (Becker, Cleveland & Wilks, 1987)(Card, Robertson & Mackinlay, 1991)(Ahlberg & Shneiderman, 1992).

A number of prototypes of so called dynamic queries applications have been implemented and evaluated. Examples are the dynamic periodic table browser (Ahlberg *et.al*, 1992), the dynamic homefinder (Williamson & Shneiderman, 1992), and the filmfinder (Ahlberg & Shneiderman, 1992). Proliferating from these prototypes, an interesting challenge was to generalize the dynamic queries ideas and construct a system which could automatically create interactive exploration environments.

Spotfire

Today, a form fill-in module is provided with most database management systems, allowing users to easily create forms for both entering data into and querying a database. Similar to a form fill-in module, a dynamic queries system could be provided with a DBMS or other kinds of applications which holds complex datasets, such as spreadsheets, GIS-systems, statistics programs, etc.

Spotfire is such a system. Spotfire can automatically create dynamic queries environments holding query devices and visualizations (Figure 1). Spotfire imports relations with named attributes, given on a straightforward text format or through so called Open Database Connectivity connections.

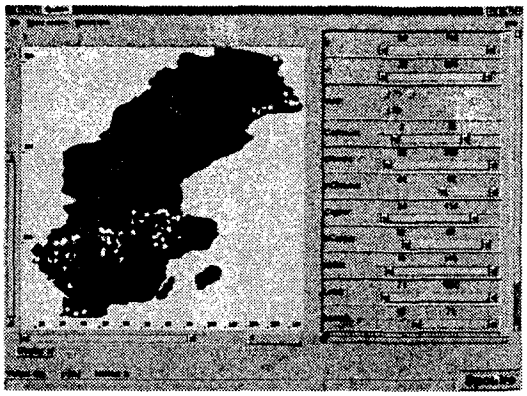

Figure 1 Spotfire created exploration environment.

An important design challenge was to design Spotfire so that users could get started with their visualization and exploration tasks with as little effort as possible. Spotfire is meant to demonstrate how a visualization system can be designed to automate the task of creating both visualizations and manipulation objects and let users start working directly with their high level exploration tasks.

Spotfire examines the data in a user specified relation and classifies it into datatypes (integers, reals, strings, date, and time). The data is stored in an internal Spotfire database object (Figure 2). Attributes are stored as straightforward vectors, with multiple indexing to increase performance for various search tasks. The user interface of Spotfire holds two main components, a query area holding a number of query devices and a visualization area, holding a number of visualizations. These user interface objects are reflected in the architecture of Spotfire in figure 2.

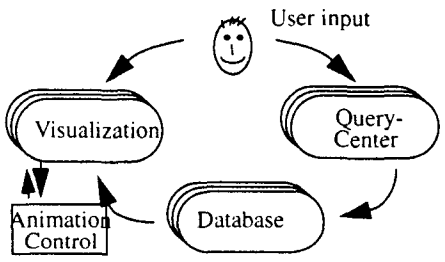

Figure 2 The Spotfire architecture.

Visualizations in Spotfire

Another design goal for Spotfire was to provide users with a rich collection of visualizations. Most work on interactive visualization seem to point to that successful visualization environments do not depend on one single powerful visualization, quite the contrary, a whole smörgåsbord of visualizations appropriate for various tasks and datatypes is closer to a successful solution. The Spotfire approach depends on a simple, yet powerful architecture. To each object in a database relation a (possibly user defined) graphical object is attached.

Attaching graphical objects to database objects

In the spirit of the goal to make Spotfire easy to get started with, users can start by specifying nothing about graphical objects connected to database objects at all. This will associate a dot with each object. When user defined polygons are specified this is done by providing them in a file along with the database. The file might contain one unique graphical object for each database object or a smaller number of polygons and a mapping from database objects to graphical objects.

Graphical objects are placed in the visualization based on two or three database attributes (such as a pair of X and Y coordinates in a database). The objects are attached to database objects by simple pattern matching.

Rendering approximations

To achieve the graphics performance necessary for dynamic queries and interactive 2D and 3D graphics, Spotfire utilizes a small set of adaptive animation and rendering strategies. Possible rendering approximations that have been experimented with in Spotfire is:

- Render objects at a lower resolution
- Render shaded objects as wire frame models.
- Skip textual labels.
- Do not fully redraw the screen while performing computationally expensive queries.

Starfield visualizations

Spotfire offers a number of visualizations where the basic one is the starfield (Ahlberg & Shneiderman, 1994a). A starfield is an interactive scatterplot with additional features for zooming, filtering, panning, details-on-demand, etc. Two or three variables from a database relation are chosen as the axes in the starfield (Figure 3).

The power of the starfield as a visualization of complex databases is that it allows for displaying a large number of database objects in one single screen. Each object is represented by a small graphical object which can be coded by color, brightness, shape, size, etc. The spatial location of an element can effectively encode two attributes of a database object. The starfield provides important qualities of visualizations such as:

- The overview provides starting points for search and query results may be presented in the context of the full database.
- Query result relevance can be judged quickly and trends and anomalies can be explored effectively.
- Feedback can be provided continuously during queries and it works for many datasets.

Figure 3 Starfield visualization..

In addition to allowing each database object be associated with a square in a starfield, Spotfire supplies users with several other possibilities for coding properties of database objects into the starfield. Visualization elements can be colored using two different schemes – users might associate database values of an attribute with either:

- Colors of different hues (blue, yellow, red, etc.), a scheme which is useful for coding nominal attributes.
- Colors of the same hue, but shifting in brightness, a scheme which is useful for coding continuous attributes.

When utilizing the latter scheme, users can specify a certain part of the range of an attribute to be colored differently – useful for indicating for example objects with values in the top 5% range. Users can also associate glyphs from a predefined library to nominal attribute values which can effectively complement the use of color.

Geographic visualizations

The immediate extension of the starfield is to provide a background map which can be used to create geographic visualizations. Spotfire users can specify such a visualization context by providing background objects holding a number of polygons defining an appropriate background (Figure 1).

Domain specific visualizations

The proposed scheme for creating visualizations also allows Spotfire to be used for creating domain specific environments such as the dynamic periodic table browser (Ahlberg *et.al*, 1992). For this specific example users may add two attributes to the original database [of properties of the chemical elements], holding the X and Y coordinates of the configuration of the elements in the periodic table.

Complex visualizations

So far, the visualizations described have not utilized the possibility of associating arbitrary polygons with database objects. An interesting example where this can be used is node and link diagrams. Similarly to how a geographic map is specified, users can specify a layout of nodes and links, and each graphical object is associated with a database object. By manipulating query devices (elaborated below), users can select nodes of certain types, links for which an associated value is within a given range, etc.

Cocktailmaps (Ahlberg, 1996) is a visualization displaying communication between a number of agents, for example a spoken conversation - created with Spotfire. Each rectangle corresponds to a person speaking for a certain amount of

time while dominating the communication space to a certain degree.

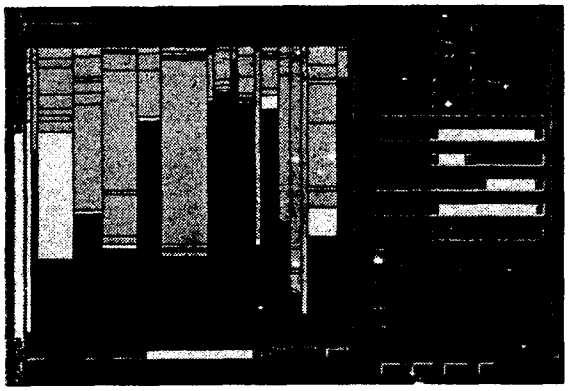

Figure 4 Cocktailmap visualization created with Spotfire.

Another class of visualizations that can be created are those depending on complex three dimensional objects. In figure 5 a schematic visualization of a part of the computer science department at Chalmers University of Technology is presented. Background objects as described above make up the outer structure of the building and employees are represented by their offices. Manipulating query devices will grey out those rooms not fulfilling the search criteria and allows for detection of trends such as how professors with high salaries are often located in the top of the building!

Multiple visualization areas

Spotfire allows multiple visualizations to be created simultaneously, and when the query devices are manipulated all the visualizations are updated interactively, allowing for powerful exploration of data. Users might observe a cluster of points in a starfield. When zooming into or brushing the cluster in the starfield a geographic map is updated too, where the same points can be observed to all be situated in the same geographic region.

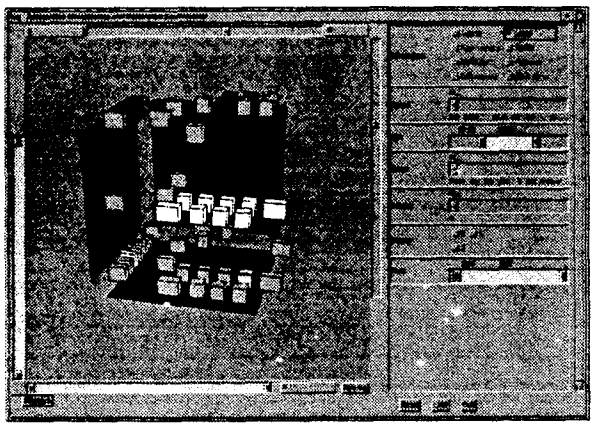

Figure 5 Interface allowing users to browse some of the staff of the CS department at Chalmers.

A number of tab buttons, one for each instantiated visualization is used to manage multiple visualizations. User may rapidly switch between active visualizations by pushing either of the buttons below the zoombar.

Visualization manipulation

Users can perform a number of operations on the starfield, such as zooming, 3D manipulation, panning, filtering, and selection of details-on-demand. Zooming is performed either with the mouse buttons (second mouse button to zoom in and third mouse button to zoom out) or the zoom bars (Figure 6)(Jog & Shneiderman, 1995). Manipulating the zoom bar on either the X or the Y axis of the visualization will either increase or decrease the size of the area in view, i.e. zooming is performed in either of the dimensions. Mouse button zooming allows for zooming in both dimensions simultaneously, while the zoom bars are appropriate for zooming each dimension separately.

To rotate a 3D visualization users manipulate two sliders (Figure 6) to rotate the view around each of the X and Y spatial dimensions, similar to the slider approach described by Chen, Mountford & Sellen (1988). Zooming in 3D is performed with the mouse buttons, as described above.

Query devices in Spotfire

Based on the classification of the user specified relation Spotfire selects query devices for each attribute. Query devices are selected from rangesliders, alphasliders (Ahlberg & Shneiderman, 1994), and toggles (all shown in context in figure 6). Simple rules decide which query devices are assigned to which attributes.

From a menu users can move the query device up, down, to the top, and to the bottom of the query area. Regrouping widgets in another order than the one provided implicitly by the order of the attributes in the database relation is quite useful, especially for relations with a large number of attributes.

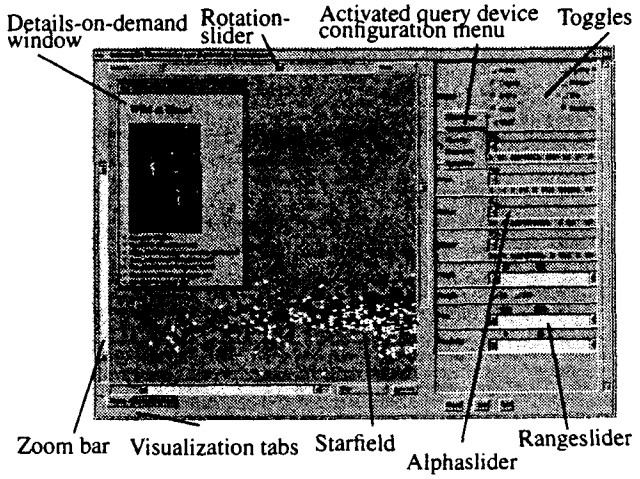

Figure 6 Reproduction of the FilmFinder created with Spotfire.

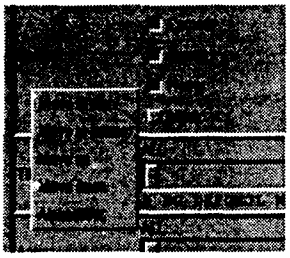

Figure 7 Menu for regrouping and change of the current widget for query devices.

Rangesliders

Rangesliders are useful for selecting range criteria for integer attributes and other attributes with an order relation (Figure 6). The rangeslider provides a natural representation of the query it is representing, i.e. a range query. Increasing or decreasing the range of an attribute allows for powerful exploration of trends and anomalies. Rangesliders are also very effective for relaxation of query parameters when the result set can be observed immediately in a visualization.

Alphasliders

The alphaslider is a widget for selecting items from long lists of for example strings (Ahlberg & Shneiderman, 1994b). The rationale behind the alphaslider is its small size - it allows for selection from lists of thousands of elements in a small screen area (Figure 6). Screen area is precious in a visual query system where most screen space should be used for the visualization of data. The alphaslider can not only be used for selection of specific strings, it is also very useful for browsing categorical variables while observing the query result set in a visualization.

Toggles

Toggles are the last of the widgets currently used for query formulation in Spotfire (Figure 6). Toggles are useful when only a few alternatives exist for an attribute and these alternatives should be presented on the screen explicitly.

Query evaluation and tight coupling

Queries are composed from the conjunction of all the query components defined by the query devices. Query devices in their initial state let through all database objects, i.e. they do not affect the result of the query. Manipulating a device immediately affects not only the visualization, but also the other query devices – the query mechanism is tightly coupled. A query device manipulation restricts the query range of the other query devices to only include criteria existing in the remaining elements of the database. This is useful for two purposes in exploration and search tasks:

- To guide querying when searching for specific elements in the database.
- To facilitate browsing of categorical variables.

Details-on-demand

Equally important to users being provided with effective overviews of data is that they are able to retrieve full and rich descriptions of specific database elements. Only showing details when they are requested is instrumental for the concept of dynamic queries. Spotfire users select details-on-demand by clicking on a visualization object with the left mouse button (Figure 6). Upon moving the mouse cursor over a selectable object feedback is provided indicating its selectability. In addition to the "ordinary" data in a database object consisting of integers and strings, Spotfire can hold file pointers to multimedia objects.

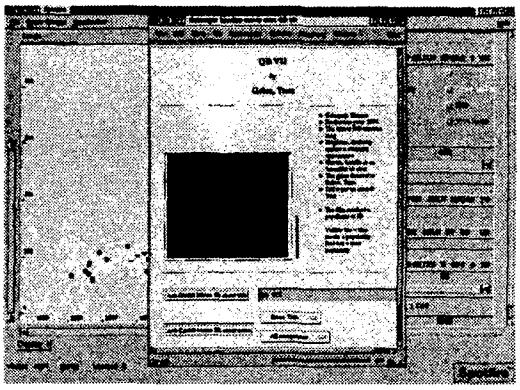

Figure 8 Netscape Navigator is utilized to present database objects..

Spotfire has a predefined approach to presenting data in database objects when they are selected. However, users can create much richer presentations by providing an HTML-based formatting document. Spotfire utilizes the Netscape Navigator browser to present database objects. Along with a database, users may supply a file in the standard HTML format, with the addition of placeholders indicating where database attribute values should be filled in by Spotfire.

Being able to receive details-on-demand is not only important for examining individual database objects, it is also useful as a starting point for search. The overview provided by for example a starfield helps users overview potentially very large amounts of data. However, a complementary approach to starting a search from the overview is to start with a specific database object which might have been found in the visualization or by initially specifying a strict set of criteria leaving one or a few objects selected. This approach has the advantage of not overloading users with a large abstract set of complex data. Instead they can apply their knowledge of one specific element (Williams, 1984).

Multiple databases

The initial assumption of Spotfire is that the database consists of one single relation. For databases consisting of several relations this constraint can sometimes be overcome by creating a universal relation which allows a whole set of relations to be regarded as one single relation (Kim, Korth & Silberschatz, 1988). However, this approach is not always ideal, and accordingly an interesting direction would be to load several database relations (sharing at least two attributes which can be visualized) simultaneously into Spotfire. Each relation would be given its own dedicated set of query devices, and each attribute of a relation is given its own dedicated query device – i.e. Spotfire would not attempt to join similar attributes which appear in several relations (although this might in some cases be useful).

Distributed exploring

Allowing several persons to visualize and explore the same data set simultaneously would another interesting direction for Spotfire. By starting several (at least two) clients of Spotfire, loaded with the same database, and then requesting these to be connected through a UNIX socket (Figure 2), several users would manipulate the same visualization. Each client would be responsible for the actual visualization and the corresponding local database, but operations such as querying, zooming, details-on-demand, etc., would be distributed to all other connected clients. Only transmitting user actions and not actual query results allow this approach to be possible even without a high speed connection.

Conclusions and future work

Spotfire is an interactive visualization and query system based on the concept of dynamic queries. The Spotfire architecture is based on the simple concept of attaching more or less complex graphical objects to database objects in visualizations. Graphical objects might be simple colored points of light, glyphs selected from a predefined library, or arbitrary sets of polygons in two and three dimensions. This allows Spotfire users to easily import database relations and create complex visualizations. The visualizations can be interactively queried, filtered, zoomed, and panned.

The functionality of Spotfire should be extended with for example:

- More query devices. The existing ones can be extended in functionality, but new ones need to be introduced.
- New types of visualizations, for example hierarchy visualizations.
- Arbitrary boolean combinations of query components.
- Spotfire can currently handle a database consisting of 30-40.000 objects with some 10-15 attributes each running on a SGI Indy 100MHz machine. A necessary area of further research is to explore datastructures for pushing this to 50-100'000 objects.
- Another typical problem in a tool based on scatterplots for displaying data is overlapping points which hold exactly the same X and Y values in the display. Spotfire currently supports jittering techniques Cleveland (1994).

Availability

Spotfire is available as a product for the Unix, Windows, and Java environments from IVEE Development AB in Göteborg, Sweden. Please contact info@ivee.com or look up IVEE on the web at http://www.ivee.com for more info.

Acknowledgements

Ben Shneiderman and his colleagues at the Human-Computer Interaction Laboratory at University of Maryland have been very supportive in this research and several of the ideas behind Spotfire originates in projects at the HCIL.

References

Ahlberg, C., Williamson, C., Shneiderman, B. (1992), Dynamic Queries for Information Exploration: An Implementation and Evaluation. *Proceedings ACM CHI'92: Human Factors in Comp. Systems*, pages 619-626.

Ahlberg, C., Shneiderman, B. (1994a), Visual Information Seeking: Tight Coupling of Dynamic Query Filters with Starfield Displays. *Proceedings ACM CHI'94: Human Factors in Comp. Systems*, pages 313-317.

Ahlberg, C., Shneiderman, B. (1994b), The Alphaslider: A Compact and Rapid Selector. *Proceedings ACM CHI'94: Human Factors in Comp. Systems*, pages 365-371.

Ahlberg, C., Wistrand, E., (1995), IVEE: An Information Visualization & Exploration Environment, *Proceedings of IEEE Symposium on Information Visualization InfoVis'95*, Atlanta, pages 66-73.

Ahlberg, C. (1996), Cocktailmaps: A Space-filling Visualization Method for Complex Communicating Systems, *Proceedings of the Workshop on Advanced Visual Interfaces'96*, Gubbio, Italy.

Becker, R., Cleveland, W., Wilks, A (1987) Dynamic Graphics for Data Analysis, Statistical Science, Vol 2., No. 4, pages 355-395.

Card, S., Robertson, G., Mackinlay, J. (1991), The Information Visualizer - an Information Workspace, *Proceedings ACM CHI'91: Human Factors in Comp. Systems*, pages 181-188.

Chen, M., Mountford, J., Sellen, A. (1988), A Study in Interactive 3-D Rotation Using 2-D Control Devices, *Proceedings ACM SIGGRAPH'88*, pages 121-129.

Cleveland, W. (1993), *Visualizing Data*, Hobart Press, Summit N.J., 360 pages.

Jog, N., Shneiderman, B. (1995), Starfield information visualization with interactive smooth zooming, *IFIP 2.6 Visual Databases Systems Proc.* (Lausanne, Switzerland), pages 1-10.

Kim, H., Korth, H., Silberschatz, A. (1988), PICASSO: A Graphical Query Language, *Software - Practice and Experience 18(3)*, pages 169-203.

Williams, M. (1984), What makes RABBIT run? *International Journal of Man-Machine Studies 21(4)*, pages 333-352.

Williamson, C. and Shneiderman, B. (1992), The Dynamic HomeFinder: Evaluating dynamic queries in a real-estate information exploration system, *Proceedings ACM SIGIR'92 Conference*, pages 339-346.