Association for Information Systems
AIS Electronic Library (AISeL)

AMCIS 1996 Proceedings

Americas Conference on Information Systems (AMCIS)

8-16-1996

How Does the Screen Influence the Visual Design in Database Applications

Laura Tarantino
Dipartimento di Ingegneria Elettrica, Universita degil Studi dell'Aquilaz, tarantino@vaxaq.cc.univaq.it

How Does the Screen Influence the Visual Design in Database Applications

Laura Tarantino
Dipartimento di Ingegneria Elettrica
Universita' degli Studi dell'Aquilaz
Poggio di Roio, I-67040 L'Aquila ITALY
tarantino@vaxaq.cc.univaq.it

1. Introduction

In a variety of database applications the user is often a novice one. This is even more true in novel applications that profit from the combination of World Wide Web (WWW) and database technologies (e.g., electronic catalogs, virtual travel agencies, virtual tourist information booths). In all such cases, the user is generally seeking for specific items satisfying given requirements: a cheap cordless telephone, a three star hotel in the Theater District of Manhattan, a medium prized Japanese restaurant in the center of Baltimore. Requests are generally vaguely specified and consequently query answers may contain many entries. System support in the analysis of the result would hence be necessary.

Notwithstanding that the ultimate goal of the information consumer is the result of the seeking process and not the query formulation per se, in most traditional systems (with notable exceptions, as in [Shneiderman 94]), as well as in WWW interfaces, the design efforts put on the query formulation and on the result visualization are undoubtedly unbalanced. Tabular visualizations are frequently used in traditional systems and are adopted in few WWW database front-ends (e.g., [Ng 93]). Though effective for the task of analyzing structured data sets, tables suffer from scrolling problems discussed in [D'Atri and Tarantino 95]. In many WWW applications, the visualization is simply a listing of linear record presentations (see, e.g., [Hudson 94]).

In our opinion many inefficiencies of result visualizations can be traced to more general defects attaining the design of database visual formats: under-regard for the restrains of the actual output collector (the screen or a window), and lack of a methodology for the visual design process. It is in this more general context that the problem must be addressed. While in the following section we analyze the first point, in Section 3 we sketch a visual design methodology that stems from considering the output device as the critical resource whose constraints must be satisfied.

Database designers may note some similarities with the physical design process. As a matter of fact, although the physical and the visual levels represent the two extremes in the hierarchy of schema abstractions in database systems, yet they share some demands: both deal with limited resources and both have to satisfy response-time requirements. It should not even surprise that some methodological tools turn out to be useful in both cases: usage parameters are among the primary input for the physical design and it should be an obvious choice to regard them among the primary input for the visual design.

2. Visualization requirements

Visual formalisms are recognized as the preferred approach fo the interaction with databases. Effective visualization depends critically on meaningful display structures that allow users to predict areas of interest and ease their navigation through the composition of the graphical elements. The effectiveness of a clear composition always depends at least as much on the *relationships* among the parts as on the parts themselves. These relationships emerge at the *global* level of the display [Mullet and Sano 95].

Quite often, display models designed for database interfaces are conceived in terms of *logical* windows, without much concern for the actual dimensions of the display. The logical window is supposed spacious

enough to contain the entire representation of the (set of) object of interest. The actual display requires either to scale the representation enough to make the logical window fit into the physical window or to regard the physical window not as a container but as a view panning over the representation (scrollbars are used as panning tools). In either case the mapping can result *structure impairing*: excessive zooming can lead to proportions that affect legibility, while a view may clip relevant portion of the structure out.

The causes of these undesired effects can be found in the mismatch between the interaction modalities defined for the specific application and the interaction modalities characteristic of the windowing system, the former being *semantic* in nature and the latter being purely *syntactic*. Database interfaces in which most of the interaction is explicitly defined for the application suffer much less of such mismatch (consider, for example, ER-based database schema visualizers offering *semantic zoom* to explore the schema at different levels of abstraction). The lesson learned is that the application must have the control over the interaction, without relying on system specific tools. To achieve it, visual models (and visual design methodologies) should be conceived as much as possible in terms of *physical* windows.

Our proposal aims at reconciling two conflicting demands: necessity of visualizing massive quantity of information on one side, and limitation on the display dimensions on the other side.

Effective answers to this problem are provided, for network-based or hierarchical information structures by, e.g., *fish-eye views* [Furnas 86] [Sarkar and Brown 92], *ConeTrees* and *TreeMaps* (see [Gershon and Eick 95] for a survey on this topic). Anyhow, while graphs are a natural mathematical tool for representing models based on objects and binary relationships between them, they may be not equally suitable for modeling different kinds of interdependencies (e.g., schemata or tables of the relational data model). Different visualization methods have been proposed for simplifying the analysis of table values (e.g., *value bars* in [Chimera 92] and the *Visualizer* in [Santucci and Palmisano 94]). It must also be observed that different presentations of the same information best support different tasks, and there is not such thing as the most effective way to visually present a data set.

We do not aim at providing a particular visual model, but rather a methodology for the definition of visual interaction environments obeying to the discussed requirements. Our proposed approach is based on *formats*, defined as associations between a single database entity and a set of linked displays. Different formats, possibly adopting different visual metaphors, may be defined for the same entity to coexist within the interface. The methodology also includes steps devoted to the definition of visual structures aimed at simplifying the analysis of sets of data (as query results), seamlessly integrated with the representations of actual database items.

3. A methodology for the visual design

In a database application, the interface may be built on top of the logical schema, or of the conceptual schema, or of the interaction schema (e.g., on views). To decouple the discussion from this choice (and from the data model), we use the term *interaction entity* (or *entity*) to denote a collection of related data item viewed as a whole by the interactive application. For the sake of simplicity, we assume that all decisions about interaction entities are taken *before* the visual design (in practice, the visual design may drive to revisions on previously taken decisions). In the following we first describe the visual structures resulting from the design, and then present the design tasks.

3.1 The Visual Structures

The basic unit of data in visual structures is the visual record, conceived for physical windows. A *visual record type* (VRT) is a collection of data item presenters, spatial relationships among them, overhead data (e.g., hyperlinks), and rules for gathering the actual values from entities. A *visual record* (VR) is an instance of a VRT. Generally, while VRTs are statically defined, VRs are dynamically generated at runtime to be displayed. A *visual file* is a set of VRs of the same type.

VRs belong to formats or to visual access structures. A *format* associates an individual entity with a set of interconnected visual records, each displaying a fragment of the entity (the association is generally defined for intensional entities, though is acceptable that formats be attached to instances, e.g., because of customization). It is often appropriate that more than one format be defined for the same entity, to fulfill the requirements of different interaction contexts. *Visual access structures* (VASs) define privileged navigation paths that simplify the analysis of sets of objects of interest (e.g., query results). VASs are generally instantiated runtime, to be linked to actual VRs. Indexes are typical VASs.

The *format organization* is the collection of all the structures necessary for interaction with (set of) entities, hence including visual files, visual access structures and definition of screen resource allocation.

3.2 The design tasks

The visual design must satisfy operational constraints such as screen limitations and response-time requirements. The design process can be split into distinct tasks based on group of related design decisions (though the actual design must be viewed as an iterative process involving successive refinements and backtracks).

Entity format design

The task faces two conflicting demands: (1) minimizing the number of accesses necessary to retrieve the total volume of information, (2) fulfilling the dimensional constraints. It involves two tightly intertwined subtasks: design of VRTs within formats, and design of individual item presenters within VRTs. The first subtask requires to define schemata and layout of VRTs, and connections among VRTs.

Definition of visual record schemata Given an interaction task, usage parameters must serve as primary guidelines for the partition of the entity, which defines the allocation of individual data items to separate VRTs. Redundancy of data may be profitable for ensuring meaningfulness of representations (the same data item, e.g., a key, may belong to more than one VRT schema of the same format). Useful analytical tools in this step are the 80-20 rule and the Bond Energy Algorithm [Teorey and Fry 82], to isolate the most active data items and to define meaningful clusters of attributes: grouping attributes that are frequently used together avoids the contiguity of unrelated data and reduces the number of accesses necessary for retrieving the needed information.

Definition of inter-record connections Logically, data items related to a single entity are still considered to be connected, and links among entity segments must guarantee such connectivity.

Definition of data item presenters and record layout In this step, given a VRT schema, the data item presenters and the spatial relationships among them are defined. Given the space constraints, the two activities must proceed in parallel. As to the first point, depending on the data item type, compression might be appropriate to save space in addition to reducing data transfer time (for example, large pictures may be better replaced by thumbnails in overview VRs, while the actual images get isolated in VRs providing insight views). As to the second point, "similarity" among attributes may guide the displacement of item presenters in contiguous areas.

Format organization design

It involves two interdependent subtasks: the design of visual files, and the definition of VASs.

Visual file design In this step the decisions on visual record clustering, visualization of inter-record relationships and allocation of visual items to screen resources are taken. The decisions of the previous step affect also the allocation of blocks of visual records into physical windows, where a *block* is a cluster of VRs to be displayed in the same physical window. Two main choices make significant differences in the

efficiency of the interaction: visualization of inter-record relationships (which may be based on, e.g., tables or linked structures depending on the kind of analysis required by a task) and dimension of blocks. This latter aspects is particularly relevant in WWW front-ends since it affects the duration of transactions and the net transfer volume.

Visual access structure design In this step the salient decisions on the interaction strategies are taken, and the VASs are designed so to support the desired strategy. VASs might be as simple as (multi-level) indexes when sequential analysis over visual files is foreseen as the primary navigation style. More complex structures may support less naive browsing fashions, e.g., relying on zoom-based analysis (hierarchies of runtime constructed statistical tables may provide different levels of *compressed* views on the query result). Decisions on inter-record relationships are strongly related to the interaction strategy.

Conclusion

In this paper we outlined a methodology for the visual design process in database applications, based on the principle that the visual design must be both *qualitative* and *quantitative*. Quantitative aspects, often neglected, play instead a salient role in the definition of effective interaction. The methodology can guide the design of database interface where multiple visual models have to coexist to support multiple tasks. A noteworthy task of the methodology concerns the definition of browsing strategies and supporting structures for the analysis of sets of data (e.g., a query result).

References

[Chimera 92] Chimera R., Value Bars: an Information Visualization and Navigation Tool for Multi-Attribute Listings, Proc. of CHI'92 Conf. on Human Factors in Computing Systems, ACM Press, pp. 293-294, 1992.

[D'Atri and Tarantino 95] D'Atri, A., Tarantino, L., On the Interaction with Database Relations: a Page-based Approach, Proc. of Workshop on New Paradigms in Information Visualization and Manipulation, in conjunction with ACM CIKM'95, December 2, 1995, Baltimore, USA. Available at http://www.cs.umbc.edu/ ~cikm/1995/npiv/

[Furnas 86] Furnas, G.W., Generalized Fisheye Views, Proc. of ACM SIGCHI'86 Conf. on Human Factors in Computing Systems, pp.16-23, ACP Press, 1986.

[Gershon and Eick 95] Gershon, N., Eick, S.G., Visualization's New Tack: Making Sense of Information, IEEE Spectrum, November 1195, pp. 38-56.

[Hudson 94] Hudson, R. L., UMass Information Navigator, Proc. of the 1994 IDEA Conference, New Orleans, LA, USA, October 17-20, 1994. Available at http://helios.ucs.umass.edu/navigator.html

[Mullet and Sano 95] Mullet, K., Sano, D., Designing Visual Interfaces, SunSoft Press, A Prentice Hall Title, 1995.

[Ng 93] Ng, J., GSQL: A Mosaic-SQL Gateway, NCSA, University of Illinois at Urbana-Champaign. Available at http://www.ncsa.uiuc.edu/SDG/People/ jason/pub/gsql/back/starthere.html

[Santucci and Palmisano 94] Santucci, B., Palmisano, F., A Dynamic Form-based Data Visualizer for Semantic Query Languages, in Interfaces to Database Systems, (Sawyer, P., ed.), Springer-Verlag, 1994, pp. 249--265.

[Sarkar and Brown 92] Sarkar, M., Brown, M.H., Graphical Fisheye Views of Graphs, Proc. of CHI'92 Conference on Human Factors in Computing Systems, pp. 83-91, ACM Press.

[Shneiderman 94] Shneiderman, B., Dynamic Queries for Visual Information Seeking, in IEEE Software, Vol. 11, No. 6, November 1994, pp. 70--77.

[Teorey and Fry 82] Teorey, T.J., Fry, J.P., Design of Database Structures, Prentice Hall Inc., Englewood Cliffs, NJ, 1982.