A Framework for Information Visualisation

Jessie B Kennedy, Kenneth J Mitchell and Peter J Barclay
Computer Studies Department, Napier University
Canal Court, 42 Craiglockhart Avenue, Edinburgh EH14 1LT, Scotland, UK
e-mail: <jessie,kenny,pete>@dcs.napier.ac.uk
phone: +44 (0)131 455-5345 ; fax: +44 (0)131 455-5394

Abstract

In this paper we examine the issues involved in developing information visualisation systems and present a framework for their construction. The framework addresses the components which must be considered in providing effective visualisations. The framework is specified using a declarative object oriented language; the resulting object model may be mapped to a variety of graphical user interface development platforms. This provides general support to developers of visualisation systems. A prototype system exists which allows the investigation of alternative visualisations for a range of data sources.

1. Introduction

Information visualisation has been described as the use of advancing graphic technology to lower the cost of finding and accessing information [RCM93]. Visualisation techniques have been used in many domains for presenting (possibly transformed) data, either to provide an overview or to allow searching for patterns. Visualisation techniques have also been used for representing meta-data such as database schemata, and further for interacting with the data, for example, query formulation in database systems.

Visualisations should be clear and effective [Tuf90]. They should also be appropriate to the type of information and application domain. For example, using spatial position is effective in visualising quantitative data or data with inherent spatial attributes, but may not be so useful for some qualitative data [Car96]. Additionally, the visualisation must suit the users in question, taking heed of their preferences, experience and the task at hand. It has been shown that interaction with the visualisation enhances its effectiveness [Shn83].

In this paper we describe some of the issues in providing effective visualisation systems and outline a framework in which to address these issues. We first provide a brief overview of existing approaches to information visualisation (Section 2), visualisation techniques (Section 3) and interaction mechanisms (Section 4). Section 5 introduces the framework in which these issues can be modelled and implemented, and the components of which can be re-used within a prototype user interface development environment. We conclude with the benefits of this approach and issues requiring further investigation.

2. Existing Approaches to Information Visualisation

Many tools may be loosely termed information visualisation systems (IVS). Statistical and spreadsheet packages have visualisation tools which allow the display of data using a variety of representations; however they do not allow any direct manipulation of the data through the visualisation. Some visualisation tools such as medical imaging systems are domain-specific and restricted to dealing with data in a specified format. Several visualisation tools exist in information-retrieval such as Lyberworld [HKW94], Bead [Cha93] and VIBE [OKS93], which extract information from documents to generate meaningful visualisation layouts for exploring the document sets.

Some relational database systems have graphical representations of the database schema and have facilities for exporting the data for visualisation to, for example, a spreadsheet. With the increasing complexity of data stored in databases, many object oriented databases provide support for visualising complex data types such as images in addition to graphical schema browsers [BFG94, RK94, O296].

These systems have limited flexibility in their provision of support for differing users with differing requirements and preferences in interaction style. In general, they are not based on any tried-and-tested design principles (although some use is made of emerging standards, such as windows look-and-feel, which address part of the overall problem); they are limited in what the user can choose to visualise; the specification of the interface is rather ad hoc; there is little or no code re-use.

The architectures of these systems are varied. Some are loose couplings of a database to an existing interface management system or widget set, others have support for visualisation integrated into the database [PDDJ96]; however, they are not built around any agreed framework. Architectural frameworks

provide for the reusability of the design and implementation of a set of co-operating classes over a given application domain [GHJV95]. The Mocha system [BCLT96] has recognised this benefit and adopted the approach to providing an algorithm animation system across the World Wide Web. The DRIVE system [MK96] adopts a similar approach to the design and implementation of interfaces to databases defining the interactions, collaborations and responsibilities of the components in the framework.

The following sections review some visualisation techniques and interaction mechanisms prior to presenting our framework for information visualisation.

3. Visualisation Techniques

Traditional two dimensional (2D) techniques for visualisation include graphs, pie charts etc. Advancements such as higraphs [Har88] commonly involve some formal combination of these techniques; the innovative TreeMap [JS91] may be classified as a combination of bar-charts and Venn diagrams. Icons are another common way of representing information in 2D. Further techniques may be composed in a hierarchical or recursive fashion to represent more dimensions of the data, or provide a more concrete representation of the domain of the visualisation.

Other advances address the dimensionality of the visualisation space in relation to the dimensionality of the information space. The VisDB system [KK94] uses a number of pixel-oriented techniques to map large amounts of multi-dimensional data unambiguously onto a 2D display. In VIBE [OKS93] control points are set up for each data dimension and records are positioned according to the combined relevance of these points. However with this technique, some ambiguity results when greater than D+1 control points are used, where D is the visual dimensionality. Using a third dimension reduces this ambiguity for slightly higher dimensional data [HKW94]. In contrast, Q-PIT [BM94] uses a 3D scatterplot (triplespace) to show selected dimensions of the information space. N-Vision [FB90] extends this technique to represent further information dimensions by placing worlds-within-worlds to show multiple dimensions in a hierarchical fashion. Bead [Cha93] adopts a different technique using a spring-based layout algorithm to plot elements from a set of documents according to their similarity. Whereas occlusion exists in many 3D visualisations, it is avoided in this case by applying a downward force to flatten the resulting layout.

The Cone Tree and Perspective Wall visualisations are examples of using 3D metaphors for information visualisation [RCM93]. An alternative approach is seen in Amaze [BFG94] which includes a 3D graph visualisation to represent database schemata and query results.

Other work is investigating the incorporation of multi-media (images, video etc.) and geographic data in visualisations [AVI96]. The wide range of work in this field demonstrates great diversity of approach and implementation; this diversity can be encompassed in the framework described below.

4. Interaction Mechanisms

Consideration of the user's interaction with IVSs is fundamental to their success. Many models of users have been proposed to gain insights into the design of user-interfaces. With respect to information visualisation, *sense-making* principles [Car96] have been applied with particular cognisance of the user's task at hand. In addition, visual variables based on human perceptual abilities have been identified which address the appropriateness of the visual representation for its associated data; for example, size for importance.

Shneiderman's mantra of 'overview, zoom and filter, and details-on-demand' [Shn96] identifies the interactive features crucial to an effective visualisation. Furthermore, direct manipulation, with rapid reversible actions, immediate feedback and dynamic queries, has been demonstrated to be very desirable in information exploration, as exemplified in IVEE [AE95]. In describing information visualisation artefacts, the Attribute and Influence Explorers [TSDS96] use *action rules* defined in terms of the relational data model to assist the identification of possible consequences of interface actions.

Feedback on a user's model of interaction is currently receiving interest in visualising financial data [Cha96]. Here, the historical profile of the interaction of all users guides the current user in the relative popularity of discrete regions of the information space. An evolving model of the users of the visualisation and their interaction with it can inform the automation of community-aware visualisations. Information visualisations which consider the history of their participants would support the wants and needs of users of the world wide web as it moves from a static information dissemination medium to an interactive collaborative environment.

Clearly any proposed framework should allow the use of a wide range of interaction mechanisms.

5. A Framework for Information Visualisation

In this section we present an overview of our work towards an organised approach to the construction of IVSs.

5.1 The Basic Framework and its Components

In [MKB96] we have presented a framework for user-interfaces to database systems. This is a high-level, contextual framework facilitating discussion of human-data interaction; we have used it to inform development of DRIVE, a prototype software system supporting rapid and principled construction of specific visualisations [MK96] (described below).

This framework is an extension of Abowd and Beale's model of human-computer interaction [AB91], tailored to the needs of data intensive systems. There are four main components: the user, the database, the visualisation, and the interaction. The visualisation and the interaction together constitute the *interface* between the user and the database.

The *user* component describes the human user of the system, providing a description of his sophistication (expertise with the data visualised or the system itself), the tasks which he wishes to perform, and his authority to view or modify data.

The *database* component represents the actual database, encompassing the data model used, the schema of a particular data-store and the instances which populate it.

The *visualisation* component considers the presentation of the data to the user: the metaphor used to represent the items of data, and the physical layout on the screen.

The *interaction* component addresses how the user may alter the presentation of the data, or the data itself. This comprises the user's intention, the medium selected to realise the intention, and the effect of the action. This effect may change the data, the visualisation, or both.

All of the above components play a rôle in determining the style and the complexity of the visualisation.

5.2 The Object-Oriented Model

An object-oriented modelling language, NOODL, has been used to specify the above framework. NOODL is a simple, high-level language intended to restore a conceptual level to object-oriented data modelling [BK91]; it has also been used to investigate a number of specific data modelling issues. It may be used for data definition, data manipulation, and querying [Bar93].

NOODL has been used to model all the components of the framework, and their inter-relationships; a relationship of particular importance is that between a data item and the interface objects which provides its visual representation.

The framework schema contains classes for all the objects existing in the data store, all the graphical objects used in the visualisation, and the users of the system; since the full power of object oriented modelling is available, users may, for example, be arranged into a classification hierarchy depending on their levels of sophistication and authority. Since the user may wish to visualise the information schema instead of or along with instances, the framework's schema must contain meta-level constructs such as *class*; in fact, the schema contains a complete NOODL meta-model.

This schema provides a template which may be instantiated by the descriptions of particular visualisations. DRIVE allows an actual visualisation to be generated automatically from such a schema.

Using NOODL to model an entire IVS provides the following advantages:

- it is well suited to data definition and manipulation;
- there is no impedance mismatch between different components;
- interface and user components may be stored in the database itself;
- it promotes communication between users and IVS designers;
- the resulting object oriented framework is re-usable.

5.3 The Architecture

Providing an object oriented model of the framework leads us to an architecture which can be used to implement any particular visualisation.

The framework provides an integrating model, co-ordinating the subcomponents; this facilitates incorporation of interactive components from disparate sources, and provision of multiple visualisations of the same data. The former point is particularly important to ensure that the resulting system is open, and able to co-operate with other software systems.

Changes made during the design of a visualisation need to be shown immediately to the user, and changes to data need to be reflected immediately in all relevant visualisations; we have provided these facilities by adopting an interpreted, active system. The system provides immediate feedback over modest databases.

Database objects include data objects and interface objects [MKB95]; each data object has corresponding interface objects which provide its graphical realisation on-screen. Characteristics common to several classes of

objects may be factored out into (abstract) superclasses; for example, all visualisation classes may be derived from a class from which they inherit common properties such as their shape, spatial position etc., and common behaviour such as the abilities to be selected or moved. The framework classes together define a pattern for IVSs.

We have implemented this architecture by providing an implementation of the NOODL data model, mapping down to the constructs of an underlying object-oriented database. In this way, details of the implemented framework are independent of the database management system used; to port the system to another platform, only the NOODL-to-system mapping layer need be re-implemented.

5.4 The Prototype: DRIVE

The approach described is embodied in DRIVE, a prototype software system which facilitates organised construction of user-interfaces to databases [MK96]; it may be used to construct interfaces ranging from simple text- and form-based styles to complex three-dimensional visualisations and non-immersive virtual reality. DRIVE contains a NOODL interpreter which allows visualisations specified in NOODL to be specified, tested, modified and used interactively; visualisations may be built up incrementally through visual programming facilities.

DRIVE consists of three major components: a database-interface component (IDS), an environment model, and a design environment.

The IDS component instantiates a particular interface under a NOODL data model. This instantiated model contains classes representing the user, database, visualisation and interaction components of the framework; these instances are made to persist by an underlying object-oriented database.

The environment model contains the visualisation classes of the IDS component together with user-interface classes; the latter may represent either standard user-interface components, or customised ones required for a particular visualisation (such as our three-dimensional widget set [BM96]). An environment-manager is used to provide dynamic registration for widgets, and to map input and output events between corresponding user-interface and visualisation classes; events are channelled as appropriate for the mode of operation. This design allows multiple, co-ordinated visualisations of the same information.

The design environment facilitates the interactive construction of interfaces; a NOODL script specifying the interface constructed can then be generated automatically to document the visualisation. An IDS editor (see lower left of Figure 1) is available for editing objects belonging to framework classes, and a context-sensitive definition editor can be used for specifying constraints etc. The designer may directly manipulate user-interface objects in accordance with the behaviour defined for them.

5.5 Examples and Experience

DRIVE has been used to construct a variety of visualisations for applications such as stock market and employee data [Mit96]. The example shown in figure 1 shows an IVS developed within DRIVE which allows users to explore historical property transactions. The interface provides the user with an overview map, a 3D display showing spatially referenced property prices, a co-ordinated details view and interface controls for the specification of dynamic queries.

Our experience has shown this is a convenient tool for composing novel IVSs: the framework-aware editor guides the developer through the assembly of the interface, and little or no programming is required.

6. Conclusion

Information visualisation is a key technique for the success of future information systems. A well designed framework provides a sound foundation for the

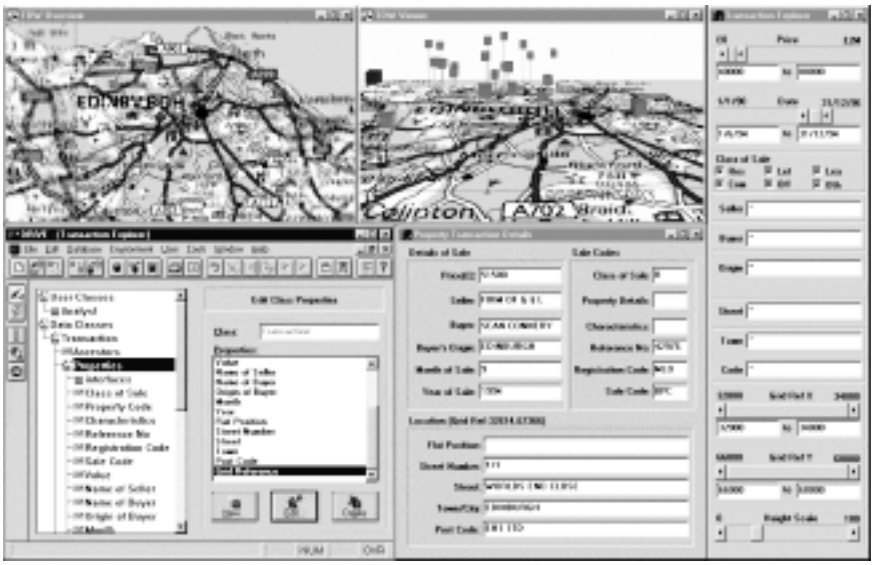

Figure 1 Property transaction explorer in DRIVE

construction of IVSs.

In this paper, we have given an overview of some current work in information visualisation and reviewed the main issues involved in supporting IVSs. We have presented a framework under which these issues can be addressed which will allow a principled approach to building and investigating IVSs.

The benefits of the approach presented include the provision of a mechanism for incorporating all relevant components necessary for developing effective IVSs; a flexible system providing user-tailorable, interactive visualisations, and an architecture offering re-usable information visualisation components and rapid visualisation development.

Our experience suggests this is a promising approach: however there is considerable research still to be undertaken particularly in the evaluation of IVSs.

7. References

[AB91] G D Abowd & R Beale (1991), Users, systems and interfaces: A unifying framework for interaction, *HCI'91: People and Computers*, 73-87.

[AE95] C Ahlberg, & E Wistrand (1995), IVEE: An information visualization and exploration environment, In *Procs. of IEEE Symposium on Information Visualization, Info-Vis'*95, Atlanta.

[AVI96] *Procs. Workshop on Advanced Visual Interfaces*, AVI'96, T Catarci, M F Costabile, S Levaldi & G Santucci (Eds), Gubbio, Italy, ACM press.

[BCLT96] J E Baker, I F Cruz, G Liotta & R Tamassia (1996), The Mocha algorithm animation system, In *[AVI96]*

[Bar93] P J Barclay (1993), *Object oriented modelling of complex data with automatic generation of a persistent representation,*. PhD Thesis, Napier University.

[BK91] P J Barclay & J Kennedy (1991), Regaining the conceptual level in object oriented data modelling. In: *Procs. BNCOD9*, (Jackson and Robinson, Eds), 269-305, Wolverhampton: Butterworths.

[BM94] S Benford & J Mariani (1994), Populated information terrains, in *Procs. 2nd International Workshop on Interfaces to Databases*, 159-169, Springer Verlag .

[BFG94] J Boyle, J E Fothergill & P M D Gray (1994), Amaze: a three dimensional graphical user interface for an object oriented database, In *Procs. 2nd International Workshop on Interfaces to Databases*, 117-131, Springer Verlag WICS.

[Car96] S K Card (1996), Information visualization and information foraging, Invited talk at *[AVI96]*.

[Cha93] M Chalmers (1993), Using a landscape metaphor to represent a corpus of documents, In *Procs. European Conf. on Spatial Information Theory*, Springer-Verlag.

[Cha96] M Chalmers (1996), Using usage data in collaborative information environments, In *Procs. International Workshop on Collaborative Virtual Environments' 96*, Nottingham.

[FB90] S Feiner & C Beshers (1990), Worlds within worlds: metaphors for exploring N-dimensional virtual worlds, In *Procs. ACM SIGGRAPH Symposium on User Interface Software and Technology*, 76-83.

[GHJV95] E Gamma, R Helm, R Johnson & J Vlissides. (1995), *Design patterns*, Addison Wesley.

[Har88] D Harel (1988), On visual formalism, *Communications of the ACM*, 31:5, 514-530.

[HKW94] M Hemmje, C Kunkel & A Willet (1994), Lyberworld—A visualisation user interface supporting full text retrieval, In *Procs. ACM SIGIR '94*, Dublin.

[JS91] B Johnson & B Shneiderman (1991), Treemaps: a space-filling approach to the visualization of hierarchical information structures, In *Procs. 2nd International IEEE Visualization Conference*, San Diego, 284-291.

[KK94] D A Keim & H Kriegel (1994), VisDB: Database exploration using multidimensional visualization, *IEEE Computer Graphics and Applications*, Sept., pp. 40-49.

[Mit96] http://www.dcs.napier.ac.uk/osg/people/kenny

[MKB95] K J Mitchell, J B Kennedy & P J Barclay (1995), Using a conceptual language to describe a database and its interface, In *Procs. 13th British National Conference on Databases*, 101-119, Springer-Verlag.

[MK96] K J Mitchell & J B Kennedy (1996), DRIVE: An environment for the organised construction of user-interfaces to databases, In *Procs. 3rd International Workshop on Interfaces to Databases*, (J. Kennedy & P Barclay Eds), Springer-Verlag Electronic WIC.

[MKB96] K J Mitchell, J B Kennedy & P J Barclay (1996), A framework for user-interfaces to databases, In *[AVI96]*.

[O296] http://www.o2tech.fr

[OKS93] K A Olsen, R R Korfhage, K M Sochats, M B Spring, J G Williams (1993), Visualization of a document collection, *Information Processing & Management*, 29:1, 69-81.

[PDDJ96] N W Paton, K Dhoan, O Diaz, A Jaime (1996), Exploitation of object oriented and active constructs in database interface development, In *Procs. 3rd International Workshop on Interfaces to Databases*, (J Kennedy & P Barclay Eds), Springer-Verlag Electronic Workshops in Computing.

[RK94] M H Rapley & J B Kennedy (1994), Three dimensional interface for an object oriented database, In *Procs. 2nd International Workshop on Interfaces to Databases*, Springer-Verlag

[RCM93] G G Robertson, S K Card & J D Mackinlay, (1993), Information visualization using 3D interactive animation, *Communications of the ACM*, 36:4 pp. 57-71.

[Shn83] B Shneiderman (1983), Direct manipulation: a step beyond programming languages, *IEEE Computer*, 16, 57-69.

[Shn96] B Shneiderman (1996), The eyes have it: A task by data type taxonomy for information visualizations, In *Procs. IEEE Visual Languages '96*, Boulder, CO.

[TSDS96] L Tweedie, R Spence, H Dawkes & H Su (1996), Externalising abstract mathematical models, In *Procs. CHI'96*, Vancouver, Canada, ACM Press.

[Tuf90] E R Tufte (1990), *Envisioning information*, Graphics Press.