Evaluating the Usability of Visual Formalisms

Ghassan Al-Qaimari

Department of Computer Science, RMIT, Melbourne, Victoria, Australia
ghassan@cs.rmit.edu.au

ABSTRACT The design of user interfaces for information-intensive applications is a challenging problem, where the organisation, visualisation and manipulation of complex data are important. Visual formalisms are proposed as a good basis for the design of interfaces to complex scientific, engineering and business applications. They are diagrammatic displays with well-defined semantics for expressing relations.

This research work focuses on evaluating the usability of visual formalisms. We show by way of an example how to empirically assess the effectiveness of visual formalisms supported in an object-oriented database.

KEYWORDS Usability, visual formalisms, GUI, OODB, modelling constructs, evaluation

1. Introduction

Graphical user interfaces (GUI) with expressive visualisations are crucial to the success of underlying systems that support increasingly rich semantics. Mental models and metaphors have been proposed as bases for user interface design. Researchers who advocate mental models argue that interface design should closely match the way a user thinks of a task, and that interfaces should reflect people's mental models [HHW84]. On the other hand, other researchers believe that familiar, everyday metaphors such as desktops should be the starting point in interface design, since they can be interpreted by users based on prior knowledge of the source of the metaphor [CMK88]. Nardi and Zarmer [NZ93], however, argue that neither mental models nor metaphors provide a good basis for the design of interfaces to complex scientific, engineering and business applications. They claim that mental models are impractical, difficult to discover, and provide confusing design guidelines for the developer. While they accept that metaphors are useful for some purposes, they believe that metaphors lack precision and can lead to confusion due to uncertainty as to whether parts of the user's prior knowledge are applicable. Instead, Nardi and Zarmer [NZ93] propose visual formalisms as a superior basis for interface design. Visual formalisms are defined as "diagrammatic displays with well-defined semantics for expressing relations. They are based on simple visual objects such as tables, graphs, plots, panels and maps – objects that contain their own semantics and do not metaphorically recreate the semantics of some other domain" [NZ93]. According to Harel [Har88]: "The intricate nature of a variety of computer-related systems and situations can, and in our opinion should, be represented by visual formalisms: visual, because they are to be generated, comprehended, and communicated by humans; and formal, because they are to be manipulated, maintained, and analysed by computers."

In this research work, we describe one type of semantic data modelling constructs, that is composite objects. We use graph as well as form-based visual representations to capture the meaning of a composite object, its structural semantics and its relations to other objects. We introduce the concept of "lucidity" to describe the usability of the visual formalisms, and we summarise the results of our evaluation strategies to show how the concept of lucidity was used to guide the design of experiments to assess the effectiveness of the visual formalisms.

2. Background

Semantic data models and object-oriented databases (OODBs) have been proposed, in part, to overcome the limited modelling facilities of the relational model. However, such models are necessarily more complex than their relational predecessors, and thus are less readily associated with visual representations which are suitable for the wide range of tasks (data entry, querying, browsing, schema design, etc) associated with a database system. This problem is further exacerbated in some recent database systems that support sophisticated semantic data modelling constructs.

The example visualisations of modelling constructs presented in this paper were developed within the EVE (Extensible Visual Environment) direct-manipulation interface [PAK92] to the ADAM OODB [GKP92]. The EVE interface (shown in figure 1) supports visualisations

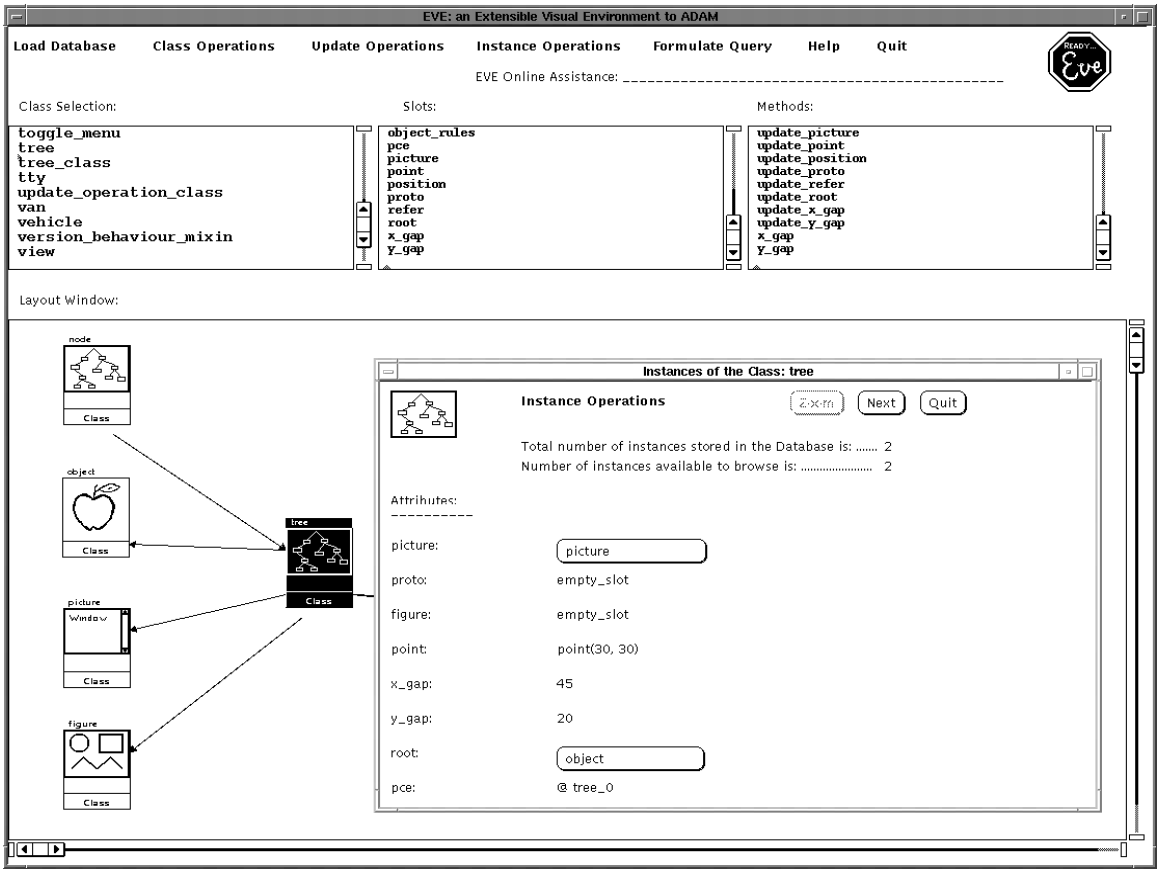

Figure 1: Layout of EVE base window plus a form for browsing instances.

of a range of sophisticated data modelling constructs, such as relationship objects, versionable objects, composite objects and active rules.

3. Example - visualisations of composite objects

A number of objects related by the *subpart* relationship are collectively called a *composite object* [KBG89]. Each composite object has one or more subparts, each of which may consist of other composite objects, or standard ADAM objects. The semantics of the part-of relationship are discussed in [PDB93].

The following example defines a **vehicle** composite class:

```
composite class vehicle
 attributes:
  registration_number: integer;
  owner:               obj person;
 composite attributes:
  door:   obj door,   set,    excl, indep;
  engine: obj engine, single, excl, indep;
  wheel:  obj wheel,  set,    excl, indep;
end;
```

In seeking lucid representations of composite object constructs, one approach which seemed promising was to investigate the application of alternative visual paradigms. Candidate paradigms which immediately suggested themselves were form-based and graph-based. Accordingly, three alternative representations of composite objects were designed. The form-based representation (figure 2) is very similar to EVE's conventional way of visualising class instances (figure 1), with the exception that two different types of attribute are distinguished, namely composite (subparts) and non-composite.

In figure 3 the subpart hierarchy is depicted as a tree in a separate window. Note that the system defined relation *subpart* represents a special type of relationship between a composite object class and other classes. Thus the subparts window, shown in figure 3, could be eliminated if, for example, we chose to use a different line style from the one shown in the layout window (figure 1) to represent system-defined relationships. This would enable the user to differentiate between user-defined relationships and the subpart relationship. The justification for designing a subpart window is similar to the justification for having separate is-a hierarchy window, namely to highlight the special semantics of the relationship, and to prevent cluttering of the main layout window.

The representation, shown in figure 4, has all the characteristics of the form-based representation except for the subparts, which are shown as icons under the heading

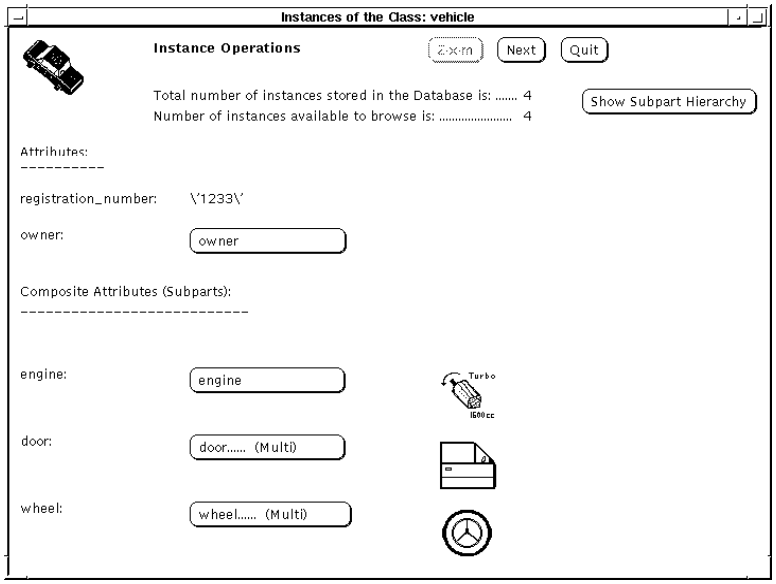

Figure 2: A proposed form-based visualisation of composite objects.

Composite Attributes (Subparts). As in the graphical representation, the user can click on any of the icons in the form to browse the subpart instances.

4. Properties of visualisations

Visualising information, especially complex and intricate information, has been the subject of considerable research, but little of this has related to the external representation of complex data models, and in particular modern object-oriented database systems.

The fundamental hypothesis underlying the work reported here is that the usability of object-oriented semantic data modelling constructs can be enhanced by lucid and expressive visual screen representations. Effective visual representations are of particular importance where modelling constructs are intended to capture both the structural and the behavioural semantics of real-world concepts. The challenge for the interface designer is to find clear, concise and comprehensive representations which achieve in practice the usability gains over textual representations which our hypothesis suggests are possible. Designing visual representations of data models involves a kind of *projection*, where the knowledge provided by the data model is transformed (mapped) into recognisable and expressive pictorial representations. This process has parallels with knowledge elicitation, as used by knowledge engineers in building knowledge-based systems [McG92]. Knowledge elicitation techniques described in the literature include: document reviews and content analysis, observation, interviews, concept and vocabulary analysis, and job and task analysis.

In [Nor88], the term *visibility* is used to indicate "the mapping between intended actions and actual operations". It is also suggested that there are occasions when too much visibility can be a problem: "It is an excess of visibility that makes gadget-ridden, feature-laden modern audio sets and VCRs so intimidating." The author also distinguishes between *affordance*, referring to whether the design of an object suggests (that is, affords) its functionality, and *perceived affordance*, referring to what a person thinks can be done with that object.

According to [Gil91], visibility should be thought of in terms of three separate dimensions, two of which are static properties of the presentation, while the third is dynamic. The first of the static components, *accessibility*, is associated with availability of information, while the other, *salience*, is associated with its meaning. Accessibility and salience are both properties of the display alone, independent of its use. The third dimension, namely *congruence*, is a property of the display in interaction with the user, and reflects whether the salience of a display is relevant to the user's task. In a complementary analysis [JDMM91], three distinct components of usability are investigated, which account for how the performance of a system changes with learning. These are: *guessability, learnability,* and *experienced user performance.* Guessability is defined as "a measure of time and effort required to get going with a system. The less time and effort required the higher the guessability."

Through analysis of this work and its relevance to the derivation of visual representations for semantic data modelling constructs in object-oriented databases, we have been led to propose (in the next section) an additional usability dimension, referred to as *lucidity.* This concept incorporates aspects of *visibility* and *guessability,* as defined earlier, but we believe it is more directly characterises the effectiveness of a visualisation of a modelling construct for use in a database interface.

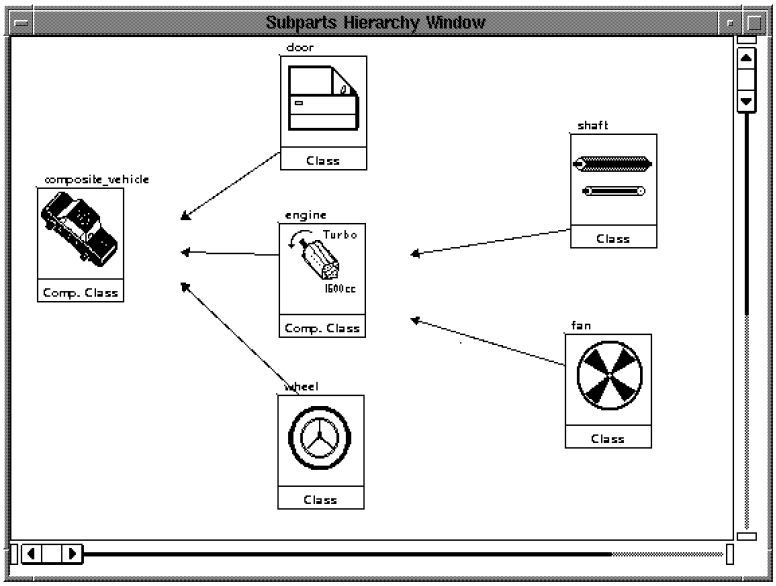

Figure 3: A proposed graph-based visualisation of composite objects.

5. Lucidity of Visual Formalisms

In [Eas84], the usability of an interactive system is defined in general as "the extent to which the user can exploit the potential utility of a system." In other words, usability refers to how well users can use the functionality of an interactive system. Usability is traditionally associated with the following attributes [Shn92]: ease of learning; high speed of user task performance; user retention over time; low user error rate; and subjective user satisfaction.

Based on our practical experience in evaluating the usability of modelling constructs, we believe that a more specific definition that complements the above definition of usability and captures the special characteristics of visual formalisms is needed to guide usability specialists when they attempt to assess the effectiveness of their visualisations.

In addition to being structural and relational in nature, our visual formalisms have meanings of their own. In other words, they do not only bring with them their own semantics (as is the case in pie charts or higraphs [Har88]), but rather attempt (through processes such as projection, knowledge distillation and organisation of visual components), to reflect the semantics of the data model itself, and convey its meaning. This distinction adds to our visualisations of modelling constructs an element of subjectivity, which is an important factor behind our attempt to devise a strategy for evaluating the usability of visual representations.

Lucidity, in the context of visualisation of modelling constructs, is defined as the extent to which a representation reflects and reveals the meaning and the structure of the underlying data model.

With this definition in mind, designers should aim at maximising the lucidity of visual representations. The different evaluation techniques which are conducted during the design and development process should be devised to measure whether the characteristics of a data model are visible to the user, and whether it is easy for the user to guess the structural semantics of the data model, and the function it is intended to perform. Lucidity may be considered to have the following attributes: *clarity*, referring to the number and organisation of visual elements on the screen; *accessibility*, referring to the ease of which information can be accessed; *perspicuousness*, referring to whether the visual formalisms convey to the user the appropriate meaning, and to whether they can be easily understood without confusion; and *transparency*, referring to whether users are able to see through the pictorial representation to the underlying semantics. Clarity and accessibility are directly related to the speed of performance, while perspicuousness and transparency are directly related to how much of the meaning and the structure of the underlying modelling construct is revealed.

6. Evaluation of Visual Formalisms

The evaluation of our modelling constructs is described in details in [AKP95, APK94]. In this section we attempt to explain how the concept of lucidity can help the usability specialist in his/her endeavour to design evaluation tasks to measure the usability of visual representations.

6.1 Heuristic vs. formative evaluation

The goal of arriving at lucid representations of advanced semantic modelling constructs cannot be achieved just by following design guidelines, such as those in [MS86] and [Bro88]. An iterative approach must be adopted involving formative evaluation in each iteration

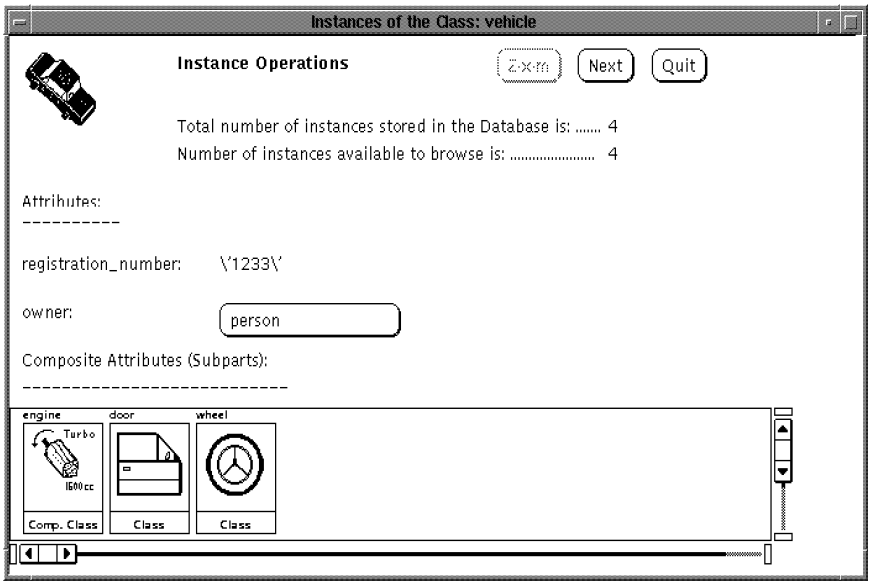

Figure 4: A proposed mixed-mode visualisation of composite objects.

[HH93].

Based on our experience, at least two evaluation cycles are needed to ensure that the visual representations capture the meaning and the structure of the underlying model. Firstly, a pre-implementation paper-based heuristic evaluation is highly recommended. Our evaluation was conducted with the help of expert users in the fields of databases and interface design. Secondly, a practical (task-oriented) evaluation was conducted using prototype implementations that incorporate the results of the first stage of the evaluation. The second technique is known as *cooperative evaluation* [MWHD93]. It brings together designers and users in a cooperative context, so as to involve the users in the design process by giving feedback and identifying weak points at each stage.

The different backgrounds, expertise and experience of the representative users can provide valuable insights when the results of the evaluation are analysed and interpreted. Our subject in the first stage of the evaluation were all experts with considerable knowledge in the area of databases, and in the area of user interface design and development. Expert users can provide critical facts and heuristics at the early stage of the design. The subject profiles in the second stage of the evaluation (25 users) included novice to intermediate users, who are likely to be the primary users of the system. Such users usually have different needs, expectations and understanding of terminology. They also tend to draw different conclusions from instructions provided by the interface [McG92].

The first stage of evaluation involved *checklists*, questionnaires answered by expert users, which were used to obtain feedback on paper mock-ups of visualisations. Such an evaluation technique offers a flexible strategy, whereby checklists can be modified according to the nature of the evaluation, for the purpose of reaching practical quantifiable evaluation results. This evaluation strategy followed a similar approach to that presented in [RJ89, Shn92].

Following consistent guidelines in the early stages of interface design ensures a consistent *look and feel* of the system as far as fonts, placement of menus, and wording of titles and messages are concerned. In addition to the design guidelines, the pre-implementation evaluation aimed to explore the extent to which the initial alternative visualisations succeed in capturing the structure and meaning of the data modelling constructs. This was achieved by encouraging the expert representative users to explore all initial designs of visualisations, to give constructive critiques, and to suggest possible alternatives. The first stage of the evaluation helps the designer to avoid, as much as possible, implementing poor visual representations that might require many revisions and modifications later at the prototyping phase. However, some ideas might seem very appealing when evaluated on paper, and yet due to unexpected limitations in the implementation environment, either cannot be implemented effectively, or turn out to be less effective in practice than anticipated.

In the second stage, we evaluated prototyped implementations of the proposed paper-based visual representations (see figures 2, 3, 4). All the prototypes were evaluated in parallel, and the type of formative evaluation conducted is usually known as exploratory evaluation, where users would attempt to perform representative tasks (or simply “walkthrough” the prototypes) and answer questions under the guidance of a test monitor. The representative users were asked to perform certain specific tasks, and while looking at the information available to them on screen, they were asked post-experience questions the aim of which was to assess the usability of the system

Your task is to find the *engine_number* of the *engine* of the *vehicle* which has the *registration_number: 1234.*

After carrying out the task, use the information available to you on the screen to answer to each of the following post-experience questions.

1. Who is the owner of this vehicle? How did you know?
2. How many vehicles are stored in the database?
   How did you know?
3. How many doors does the vehicle have?
   How did you know?
4. Can two vehicles share the same engine?
   Why/Why not?
5. What will happen to the vehicle if you try to remove an *owner* from the database?
   How did you know?
6. What fundamental difference is there in the way the *registration_no* and the *engine* are modelled as properties of vehicle?
   How did you know?
7. What is the fundamental difference between the way in which the *engine* of a vehicle is modelled compared with the *owner* of a vehicle?
   How did you know?
8. Is an engine a normal object in the database?
   Why do you think it is/Why do you think it is not?
9. Is it a good idea to have a special button in the dialog box (figure 2 and 4) which gives the user the choice of whether to display the *Subparts Hierarchy Window* (figure 3) or not.
   Explain why it is/isn't a good idea ?

Figure 5: A representative evaluation task followed by post-experience questions.

in general, and the lucidity of the each visual representations in particular (see figure 5).

6.2 Designing an evaluation task

Designers and evaluators use different evaluation methods to achieve an objective measurement of the user/system interaction, and yet it is in fact the subjectivity of the evaluation experience of the individual users which they are often after [CHP90].

As indicated earlier, one of our aims was to measure the *lucidity* of the visual representations of the data modelling constructs supported in the EVE interface, in addition to assessing the usability of the system in general. To do so, it was necessary to devise an evaluation task that reflects what is meant by lucidity and how it could be measured.

The tasks performed by our representative users in the second stage of the evaluation aimed at exploring the following interface issues:

- Do the visual representations succeed in suggesting to the user the meaning of the data model, and do they reveal the semantics supported by the modelling constructs?
- Is it clear to the user that a construct, such as a composite object, is a normal object in the database?
- How useful are icons for quick identification of related visual representations?
- How helpful is a hierarchical overview of special types of relationships?

It was equally important after deciding on the tasks to appraise them in order to check their representativeness and the extent to which they explore the interface issues under evaluation. To do so, the proposed tasks were carried out by an experienced user in a pilot investigation.

During evaluation sessions the emphasis was placed on the reason behind *how* the user found a certain answer, and *why* a certain conclusion was reached or a decision made. Furthermore, an emphasis was placed on finding out whether the representative users performed subtasks within a reasonable period of time, which was an important factor in determining the effectiveness of a visual representation. Users were encouraged to to think aloud, in order to generate as much feedback as possible. This

was achieved by asking questions after performing each subtask (such as *How did you know that?* or *What makes you think so?*, etc). For example, by asking the representative user a question like: *How many vehicles are stored in the database?*, the designer is actually interested in finding out whether the *counter* in the representation is visible to the user, and whether or not the wording is confusing.

The representative users were kept unaware of the fact that the time taken to perform each subtask was being monitored. Such an informal evaluation strategy created a relaxed atmosphere, and enabled representative users to concentrate better on the screen without interruption.

While the representative user is carrying out the task, the test monitor records feedback information on a *think aloud protocol recording sheet*. Such a recording sheet contains questions such as: What does the task user notice? What is the task user thinking now? What kind of clues is he/she looking for? What are the problems encountered at this subtask? What suggestions are being made by the task user?

In general, the first two attributes of lucidity, namely clarity and accessibility, are directly related to the speed of performance, therefore, the evaluation tasks should measure how long it takes the user to complete the task successfully. The third and forth attributes of lucidity, namely perspicuousness and transparency, are directly related to the strategy. In other words, if the visual formalism is poorly designed and the user fails to understand its meaning or its structure, then the user's next step is likely to be incorrect. The user's feedback is crucial to assessing the perspicuousness and transparency of visual formalisms, and the evaluation task should be designed to allow the test monitor to elicit the user's mental model. Exploratory evaluations were suitable for assessing the perspicuousness and transparency of the visualisations because they tend to be informal and almost collaboration between participants and test monitors, with much interaction between the two. Unlike clarity and accessibility where the emphasis is on measuring how well the user is able to perform (quantitative data), exploratory evaluations tend to solicit users' ideas about how to improve confusing areas (qualitative data).

7. Conclusion and future directions

This paper attempted to exploit another dimension in interface evaluation, that is, the degree of *lucidity* of visual representations, and has demonstrated how the attributes of lucidity influence the types of experiments designed to assess the effectiveness of visualisations.

The need for expressive visual representations becomes important as the underlying systems support increasingly rich semantics. Our experience proved that it is possible to make advanced data modelling constructs accessible to non-specialist users.

Evaluation is crucial to the development of successful interfaces with effective and lucid visualisations. We believe that cooperative evaluations play a crucial role in bringing the designers and the users together in a context that involves the users in every step of the iterative design process.

References

[AKP95] G. Al-Qaimari, A. C. Kilgour, and N. W. Paton. Visualising Data Modelling Constructs in an Object-Oriented Database. In **Proc. of the 7th International Conference on Information Systems Engineering (CAiSE'95) - Workshop on End User Development**, 1995.

[APK94] G. Al-Qaimari, N. W. Paton, and A. C. Kilgour. Visualising Advanced Data Modelling Constructs. In **Information and Software Technology, 36(10)**, pages 597–606, 1994.

[Bro88] C. M. Brown. **Human-Computer Interface Design Guidelines**. Ablex Publishing Co, NJ, USA, 1988.

[CHP90] J. Crellin, T. Horn, and J. Preece. Evaluating Evaluation: A Case Study Of The Use Of Novel And Conventional Evaluation Techniques In A Small Company. In **Human-Computer Interaction - INTERACT'90**, pages 329–335, (North-Holland), 1990. Elsevier Science Publishers. D. Diaper et al (Eds).

[CMK88] J. Carrol, R. Mack, and W. Kellogg. Interface Metaphors and User Interface Design. In M. Helander, editor, **Handbook of Human-Computer Interaction**, pages 67–85. Elsevier, 1988.

[Eas84] K. D. Eason. Towards the Experimental Study of Usability. **Behaviour and Information Technology, 2(3)**, 1984.

[Gil91] D. J. Gilmore. Visibility: A Dimensional Analysis. In **People and Computers VI, Proc. of the HCI '91 Conference**, pages 317–329. Cambridge University Press, 1991. D. Diaper and N. Hammond (Eds).

[GKP92] P. M. D. Gray, K. G. Kulkarni, and N. W. Paton. **Object-Oriented Databases: A Semantic Data Model Approach**. Prentice-Hall, 1992.

[Har88] D. Harel. On Visual Formalisms. **Communications of the ACM, 31(5)**, pages 514–530, 1988.

[HH93] D. Hix and H. R. Hartson. **Developing User Interfaces**. Wiley, 1993.

[HHW84] J. Hollan, E. Hutchins, and L. Weitzman. STEAMER: An Interactive Inspectable Simulation-Based Training System. **The AI Magazine, (5)**, pages 15–27, 1984.

[JDMM91] P. W. Jordan, S. W. Draper, K. K. MacFarlane, and S. McNulty. Guessability, Learnability, and Experienced User Performance. In **People and Computers VI, Proc. of the HCI '91 Conference**, pages 237–245. Cambridge University Press, 1991. D. Diaper and N. Hammond (Eds).

[KBG89] W. Kim, E. Bertino, and J. F. Garza. Composite Objects Revisited. In **Proc. of the 1989 ACM SIGMOD, 18(2)**, pages 337–347, 1989. J. Clifford, B. Lindsay and D. Maier (Eds).

[McG92] K. McGraw. **Designing and Evaluating User Interfaces for Knowledge-Based Systems**. Ellis Horwood Publishers, 1992.

[MS86] J. Mosier and S. Smith. Guidelines for designing user interface software. Technical Report MTR-10090, ESD-TR-86-278, The Mitre Corporation, Beford, MA, USA, 1986.

[MWHD93] A. Monk, P. Wright, J. Haber, and L. Davenport. **Improving Your Human-Computer Interface: A Practical Technique**. Prentice Hall, 1993.

[Nor88] D. A. Norman. **The Psychology Of Everyday Things**. Basic Books, 1988.

[NZ93] B. A. Nardi and C. L. Zarmer. Beyond Model and Metaphors: Visual Formalism in User Interface Design. **Journal of Visual Languages and Computing**, pages 5–33, April 1993.

[PAK92] N. W. Paton, G. Al-Qaimari, and A. C. Kilgour. An Extensible Interface To An Extensible Object-Oriented Database System. In **The 1st International Workshop On Interfaces to Database Systems (IDS92), Glasgow**, pages 265–281. Springer-Verlag, 1992. R. Cooper (Ed).

[PDB93] N. W. Paton, O. Diaz, and M. L. Barja. Combining Active Rules and Metaclasses for Enhanced Extensibility. **Data and Knowledge Engineering, (10)**, pages 45–63, 1993.

[RJ89] S. Ravden and G. Johnson. **Evaluating Usability Of Human-Computer Interfaces: a practical method**. Ellis Horwood Limited, 1989.

[Shn92] B. Shneiderman. **Designing The User Interface: Strategies for Effective Human-Computer Interaction**. Addison-Wesley, second edition, 1992.