Paper submitted to Workshop on Formal Aspects of Human-Computer Interaction, Sheffield, September 1998

A Microtheory of Visually-Derived Information

Richard M Young

Psychology Department, University of Hertfordshire, Hatfield, Herts AL10 9AB, UK

R.M.Young@herts.ac.uk

Abstract

In human-computer interaction, information is most commonly conveyed from computer to user by being shown on a VDU or other kind of visual display. By seeing and interpreting what is shown on the display, users come to know things both about the objects on the display and about other entities those objects depict. From a formal point of view, how does this process "work"? This paper presents a simple microtheory of how users acquire information from visual displays. The intention is for the microtheory to serve as a standard analysis of visually-derived information, to be included as part of more complete formal descriptions of devices and users.

1. Introduction

In human-computer interaction, most information is conveyed from the computer to the user by being shown on a VDU, flat screen, or more specialised kind of visual display. Assuming that the user does indeed "see" what's on the display, what can we assume the user knows about what is and is not on the display, and about what can be inferred from the display?

In most won between computers and people, this question — of what the user knows on the basis of the information shown on the display — is side-stepped, typically by describing the content of the display in terms of explicit propositions which are assumed known to the user. Even where considerable attention is paid to the relationship between the display and the internal state of the computer (e.g. Duke & Harrison, 1994), the analysis is not extended to the relationship between the display and the user. Such an approach works well enough for handling simple, atomic propositions, such as knowing the current value of a state variable. However, it is not adequate for dealing with the richer information conveyed by inherently graphical means, such as by the use of spatial and graphical relations, which is typical of the information presented on modern visual displays.

So the purpose of this paper is to develop a formal description of the relationship between objects shown on a visual display and the knowledge a user derives from them. The description is not self-contained, but rather is intended to be incorporated into larger-scale formal analyses of users and devices, providing a standardised and well-founded account of the visual interface. The description is cast at the knowledge level, in terms of the relationship between the visual display and what the user knows. It is thus not a *process* model, and abstracts over questions of human visual psychology and physiology.

The next section of the paper (2) summarises the basis for describing knowledge. Section 3 provides the foundation for describing what is visible, section 4 draws conclusions concerning what the user knows about objects on the display, while section 5 deals with objects *not* on the display. Section 6 extends the analysis to include knowledge of objects not themselves on the display but depicted by objects on the display. Section 7 sketches the application of the microtheory to two examples, and section 8 offers some brief closing thoughts.

2. Knowledge

Our representation of knowledge centres on the modal operator K, where K(ϕ) means that the user knows that ϕ. The axioms for knowledge are adapted from Table 8.6 of Davis (1990, p.375), and are as follows:

(Consequential closure) $[K(\phi) \quad K(\phi \quad \varphi)] \quad K(\varphi)$ [1]

(Knowledge of axioms) If ϕ is a logical axiom or an axiom of knowledge, then K(ϕ) is an axiom [2]

(Veridicality) $K(\phi) \quad \phi$ [3]

(Positive introspection) $K(\phi) \quad K(K(\phi))$ [4]

For further discussion of these basic axioms, see Davis. We have dropped the first argument from the K operator, because we are dealing with only a single “knower”, the user. The axiom of veridicality, [3], tells us that we really are here dealing with *knowledge*, not just belief. So the analysis presented in this paper does not deal with the modelling of any false beliefs the user may have, such as about the contents of a hidden text buffer in the device.

The basic K operator is supplemented with two further knowledge operators. One is Kw, ‘knows whether’, where Kw(ϕ) means that the user knows whether ϕ holds true or not. For example, if ϕ is “there is a red circle on the screen”, then Kw(ϕ) means that the user knows whether or not there is a red circle on the screen. Davis (pp.378-379) points out that Kw(ϕ) means that the user either knows that ϕ is true or knows that it is false, so we have:

$Kw(\phi) \Leftrightarrow K(\phi) \quad K(\neg\phi)$ [5]

The other knowledge operator is Kv, ‘knows value’, where Kv(τ) means that the user knows the value of τ. For example, Kv(colour(x)) means that the user knows the colour of x, say colour(x) = green. Following Davis (p.379), we capture this in the following axiom schema:

For any term τ, “$Kv(\tau) \Leftrightarrow \quad x \cdot K(\tau = x)$” is an axiom [6]

3. Visibility

Our approach to modelling visually-derived information is to assume that certain information is available on the device display, and can be known by the user. That information concerns the objects visible on the display, and the predicates and functions that refer to them.

3.1 Visible-objects

We start with the idea that various objects are visible on the screen. They are called *visible-objects*. A visible-object can be any kind of entity the user can perceive on the display. It might be an ‘icon’ for an application, a ‘window’, a textual label, a circle, a menu, a scrollbar, a paragraph of text, or whatever.

Visible-objects are *visual* objects, and are distinct from other objects in the universe of discourse. In particular, they are distinct from other objects which they may be taken to depict. For example, the visible-object which is the icon for the spreadsheet program on my computer is distinct from the spreadsheet program itself. Similarly, in a program which displays on the screen a small number of address cards selected from a larger collection of such cards, the visual object on the screen which depicts a “card” bearing Murgatroyd’s address is distinct from the object which is the program’s internal representation of the card, and which continues to exist even when the visual “card” is no longer shown on the screen.

As we shall see in the next section, all visual-objects x satisfy the predicate visible(x), meaning that they are visible on the screen. Note that we do not define a set consisting of all the visible objects, and still less do we assume that the user knows of such a set. Because of the deliberately vague and permissive nature of what counts as a visible object, such a set could be problematic to work with.

3.2 Visible-predicates

3.2.1 Simple visible-predicates

Certain predicates are defined by the application as being *visible-predicates*. A visible-predicate meets two conditions:

- Certain of its arguments can be visible-objects. They are known as *guarded* arguments.
- When the guarded arguments are instantiated as visible-objects[1], the truth value of the predicate is visible to the user, i.e. known from the visual information on the display.

The intention is that a visible-predicate asserts some property of the display that is visible to the user. For example, if the relation above(x,y) is defined for two visible-objects x and y, then the user can *see* whether x is above y. Similarly for the relation letter-of(l,w) holding between a letter l and a word w on the display.

The most primitive visible-predicate is visible(x), meaning that x is a visible-object, visible on the display. Its (first and only) argument is a guarded argument.

We express the idea that the truth value of a visible-predicate is known to the user, by the following axiom schema:

If VP is a visible-predicate with guarded arguments $g_1 \ldots g_n$, then

“visible(g_1) … visible(g_n) Kw(VP)” is an axiom [7]

If all the guarded arguments of a visible-predicate are visible, then the user knows whether the predicate holds true.

From [7] and [5] we can derive a theorem saying that if an object is visible, then the user knows that it is. In other words, loosely speaking, the user can see all visible objects:

visible(x) K(visible(x)) [8, th]

3.2.2 Compound visible-predicates

It follows from the axioms we have given that a predicate, constructed from visible-predicates joined by logical connectives, itself behaves like a visible-predicate, in the sense that a theorem analogous to [7] applies and states that the user knows the truth value of the predicate. Such a predicate is called a *compound* visible-predicate. For example, if colour, shape, and shading are

[1] We could have defined visible-predicates as applying only to visible-objects in their guarded arguments. The disadvantage is that we would then have to define predicates separate to those that apply to non-visible objects. For example, the ordinary predicate colour-is(x,y) would have to be shadowed by a parallel predicate visible-colour-is(x,y) specially for visible-objects x. The way we are modelling it allows us to write both colour-is(circle,red) for some circle visible on the

visible-predicates, then the property of being a red stippled square is a compound visible-predicate.

The simple and compound visible-predicates together form the class of *effectively*-visible-predicates, as described by the following recursive definition:

Definition of effectively-visible-predicate (EVP). An EVP takes one of the following forms:

1. If V is a simple visible-predicate, then V is an EVP.
2. If V is an EVP, then ¬V is an EVP.
3. If E is an EVP with guarded arguments $e_1 \dots e_m$, and F is an EVP with guarded arguments $f_1 \dots f_n$, then E ◊ F is an EVP with guarded arguments $e_1 \dots e_m$, $f_1 \dots f_n$, where '◊' is one of the logical connectives , , , ⇔ , … .

An effectively-visible-predicate acts like a visible-predicate in that it satisfies the following theorem, analogous to [7]:

If P is an effectively-visible-predicate with guarded arguments $g_1 \dots g_n$, then
"visible(g_1) … visible(g_n) Kw(P)" is a theorem [9, th]

If all the guarded arguments of an effectively-visible-predicate are visible, then the user knows whether the predicate holds true.

The proof of the theorem mirrors the recursive structure of the definition, and is given in the Appendix. The significance of [9] is that it allows the theory to deal with realistically compound descriptions, such as "red square" or "circle above a line", instead of being confined to individual primitive visual properties such as "red" or "square".

From the viewpoint of psychological plausibility, we may not be happy with the idea that the user can "see" the truth or falsity of arbitrarily complex visual predicates. Such a predicate might be "a red square in the upper right quadrant of the display is above a small circle iff the red square is inside a stippled triangle". However, the ability of the user to handle such things follows as a consequence of assumptions we certainly do want to make, in order that we can talk about much simpler things such as the user knowing there's a red square on the display. The situation is closely analogous to that concerning the axiom of consequential closure of knowledge, [1]: as soon as we allow the user to make simple inferences, we're forced to allow that he knows all mathematical truths! (See discussion in Davis, 1990, pp.357-358). The best course seems to be to allow the capability, but to be cautious in interpreting the results if the capability is exploited too enthusiastically.

3.3 Visible-functions

For convenience, we can also define certain functions as being *visible-functions*. The intention is to capture the idea that the user can see the value of certain functions from the display. For example, the colour of a visible-object, colour(x), can be seen on the display. The treatment parallels that of visible-predicates. A visible-function meets two conditions:

- Certain of its arguments can be visible-objects. They are known as *guarded* arguments.
- When the guarded arguments are filled by visible-objects, the value of the function is visible to the user, i.e. known from the visual information on the display.

We express these conditions in the following axiom schema:

If VF is a visible-function with guarded arguments $g_1 \ldots g_n$, then

“visible(g_1) … visible(g_n) Kv(VF)” is an axiom [10]

If all the guarded arguments of a visible-function are visible, then the user knows the value of the function.

4. Visual presence

Almost all the assumptions of the microtheory are now in place. In this section we present two important results concerning the user’s ability to identify objects in terms of their visual properties. The deductions are so shallow that they barely deserve to be called theorems. More helpful than sophisticated theorem proving is the careful interpretation of results we already have.

The primary result is that for any effectively-visible-predicate, if there are visible objects that satisfy the predicate, the user knows they do. From [9], by adding “ P” to both sides and applying [5], we get:

If P is an effectively-visible-predicate with guarded arguments $g_1 \ldots g_n$, then

“visible(g_1) … visible(g_n) P K(P)” is a theorem [11, th]

***Principle of seeing**: If all the guarded arguments of an effectively-visible-predicate are visible, and the predicate holds true, then the user knows that it does.*

Theorem [11] represents the main way in which visually-derived information becomes known to the user. What it says basically is that the user can see what’s (visibly) true about the objects on the screen. For example, if there is a red square x above a green circle y, then the user *knows* that x is a red square and y is a green circle and that x is above y. The theorem also conveys the user’s ability to *find* objects on the screen that satisfy some visible property. For example, if the user is looking for a red square, and if x is indeed a red square, then the user can “find” x in the sense that he knows that x is a red square.

Furthermore, and by the tiniest of logical moves, it follows that if there are visible objects that satisfy some visible-predicate, then the user is aware that such objects exist:

If P is a effectively-visible-predicate with guarded arguments $g_1 \ldots g_n$, then

“visible(g_1) … visible(g_n) P K($g_1 \ldots g_n \cdot$ P)” is a theorem [12, th]

***Visible presence**: If all the guarded arguments of an effectively-visible-predicate are visible, and the predicate holds true, then the user knows that there exist visible-objects which satisfy the predicate.*

Glossed slightly differently, this theorem states that the user knows what is on the display. If there is a red square on the display, then the user knows that there is.

5. Visible absence

One of the distinctive characteristics of visually-derived information is that, unlike information derived from most other sources, the user knows something about what is not the case as well as what is. Specifically, as well as knowing what is on the display, the user knows — within limits — what is *not* on the display. The user can see, for example, that there are no green objects on the display.

Capturing this important property within the theory is problematic, to put it mildly. Rather than tackling the difficulties head-on, we will simply assert this distinctive characteristic of visually-derived information as an axiom schema. It can be formulated at varying degrees of "strength". Here we give a relatively weak version, phrased in terms of simple visible-predicates:

If VP is a (simple) visible-predicate with guarded arguments $g_1 \ldots g_n$, then

"¬ $g_1 \ldots g_n \cdot$ visible(g_1) … visible(g_n) VP

K(¬ $g_1 \ldots g_n \cdot$ visible(g_1) … visible(g_n) VP)" is an axiom [13]

***Visible absence**: If there are no visible objects that satisfy a particular visible-predicate, then the user knows so.*

For example, if we assume that "being highlighted" is a visible-predicated, then if there are no highlighted objects on the screen, the user knows so.

Since the principle of *visible presence*, [12], asserts the positive case, and *visible absence*, [13], asserts the negative case, we conclude that

If VP is a (simple) visible-predicate with guarded arguments $g_1 \ldots g_n$, then

"Kw($g_1 \ldots g_n \cdot$ visible(g_1) … visible(g_n) VP)" is a theorem [14, th]

***Visual detection**: The user knows whether or not there are visible objects that satisfy a given visible-predicate.*

As noted, [13] and consequently [14] are stated in a relatively weak form, that applies directly only to simple visible-predicates.[2] If a particular application of the microtheory requires a stronger form, say one that applies to all effectively-visible-predicates, then a further axiom could always be added. [13] and [14] do allow inferences to be made that extend the results to certain classes of compound predicates. For example, we have:[3]

If CVP is a compound visible-predicate which takes the form of a conjunction of simple visible-predicates VP_1 … VP_n, where each VP_i has a single guarded argument g that refers to the same object, then "Kw(g · visible(g) CVP)" is a theorem [15, th]

The user knows whether or not there is a visible object which satisfies a conjunction of (simple) visible-predicates.

It is [15] that allows us to say, for example, that the user knows whether or not there is a red stippled circle on the display.

[2] There is probably some justification for using the weaker form from the psychology of perception, in so far as it is implausible to suppose that people can *see* that some arbitrarily complicated visual predicate is not satisfied by any combination of objects on the screen. However, that is not really the kind of consideration that the microtheory is responsive to, and notice that we are allowing for arbitrarily complicated predicates in the positive case, [12].

[3] At least, I *think* this follows, but I haven't been able to construct a proof. I have a suspicion that a proof can be carried through only if we allow the user to construct sets such as "the set of all red objects on the screen", and I find that a bit uncomfortable.

6. Referring *through* the interface

In some situations, for example with drawing programs, the user may be interested primarily in the visual objects themselves. More commonly, however, the objects on the display refer to or depict other objects, which are what the user is mainly concerned with. Those objects can be of various kinds. Often they will be other objects inside the computer, as when an icon on the screen depicts an application program or data file. In other cases, the objects of interest may be in the world outside the computer. In an address-book program, the cards on the screen depict cards internal to the program, which in turn refer to real people and addresses.

We are obviously not going to attempt to develop a general theory of reference as part of our microtheory. But it is relevant to understand how the knowledge that the user acquires visually from the display leads to further knowledge about entities outside the display.

6.1 Depiction

The basic relationship we shall assume is that objects on the display can *depict* (or *stand for*, or *refer to*) objects not on the display. We write depict(v,x) to say that some visible object v depicts a non-display object x: for example, depict(Excel-icon, Excel-program). It is sometimes convenient to use the associated function pictee(v), with the obvious connection:

$$\text{depict(v,x)} \;\Leftrightarrow\; \text{pictee(v) = x} \qquad [16]$$

In simple cases, the depiction relation together with some interpretation information from the application are sufficient to map knowledge about display objects into knowledge about domain objects. For example, suppose a program displays small pictures of animals, each with their appropriate colour. Then if the user can recognise the pictures, K(depict(polar-bear-icon, polar-bear)), etc., and knows the interpretation rule, K(colour(x) = colour(pictee(x))), then the user can move from knowing merely the colour of the icon, K(colour(polar-bear-icon) = white) — which is known by virtue of [10] — to inferring the colour of the animal itself, K(colour(polar-bear) = white).

6.2 Visual propositions

More commonly, information about the domain is conveyed not by individual objects on the display but by the relations between them. Those relations are thought of as encoding propositions about the objects depicted.

As before, the situation can be described very simply. In addition to the depiction mapping, the user needs to know the interpretation of the relevant visible predicates. Imagine a program which displays on the screen a map of the world, with pictures of animals superimposed on the part of the world where they are most commonly found. As before, the user needs to know the depiction relation between icons and animals, and also in this case between locations on the map and, say, the country represented. Suppose that at(x,l) is a visible-predicate holding between an animal icon and a map location. Then if the user knows the interpretation mapping:

K(at(icon, location)   commonly-found-in(pictee(icon), pictee(location)))

then he can use it to move from knowing at(polar-bear-icon, top-region-of-map) to inferring commonly-found-in(polar-bear, arctic).

6.3 Language

Language (whether textual, symbolic, visual, or whatever) is a somewhat special case, because it folds an entire proposition (about the domain) into a single object (on the display). Consider a user who accesses a weather forecasting service on the Internet, and in a box in the middle of screen sees the day's forecast for his area: *It will rain today*. On the display, that sentence is a single, visible object (possibly decomposable into other visual objects corresponding to the individual words, etc.). Somehow, by reading the sentence, the user extracts the proposition encoded there and comes to know that weather-forecast(today, rain).

This is a deep topic, and we are not about to propose a formal theory of semiotics. Instead, we'll take the simplest possible route to making it work, which seems to be to have a sentential operator 'encodes' which holds between an appropriate kind of visual object and a proposition. For example, if 'vforecast' is the visual object representing the forecast, then we have encodes(vforecast, weather-forecast(today, rain)). We assume that the user can determine visually that a certain visual object is a "text" in some language, which we express by asserting that is-text(x) is a visible-predicate. We then assume:

visible(x) K(is-text(x)) encodes(x,prop) K(prop) [17]

***Axiom of reading**: If the user knows that a visible object is a piece of text, then the user knows the proposition encoded by the text.*

In other words: *users can read*. In some applications we may prefer to change the formulation so that [17] holds only after the user applies the mental action read(x).

7. Examples

In this section, we illustrate briefly how the microtheory can be applied to particular applications by sketching its use in two examples. In each case, what happens is that certain crucial objects and relations are identified by the analysis of the device. By declaring those objects to be "visible-objects", etc., the relevant axioms from this paper are "pulled in" to the analysis, which is thereby extended to cover what the user can know by looking at the device display.

7.1 Example 1: One-item menu

The first example is very simple and concerns the one-item menu used in the "Tango phone" (Blandford, Butterworth & Good, 1997), a typical mobile telephone. The phone's display includes a single line of alphanumeric information, which in various contexts shows a telephone number, the name of a contact, or the label of an item in a hierarchical menu.

The analysis of the Tango phone assumes, not surprisingly, that the user knows the content of the one-line alphanumeric display. To connect with the microtheory developed in this paper, we could treat the content as the value of a function *alpha()*, and then assert:

alpha() is a visible-function [18]

Declaring alpha as a visible-function pulls in the relevant axioms, in particular [10] which would then imply that the user knows the value of the function.

The analysis further assumes that users can tell from the display whether or not the phone is in menu mode. Informally, this is done by users judging whether the alphanumeric display is showing (a) a telephone number or a person's name, or (b) an item which is a familiar or at least plausible label for a menu item. In case (b) the phone is in menu mode, in case (a) then not[4]. We

[4] Except that at certain points in the use of the menu, the display can be showing a telephone

could describe this judgement as based on purely “internal” knowledge, say as a mapping from the value of alpha() to the set {in-menu-mode, not-menu-mode}. However, the spirit of the analysis is closer to describing it as a perceptual distinction, which we can capture by asserting

in-mode(menus) is a visible-predicate, [19]

a declaration which pulls in [7] and implies that users know whether in-mode(menus) holds true.

Note that inferences from the display play little role in the description of Tango phone, and that the use of the microtheory in this example is accordingly shallow.

7.2 Example 2: Cardbox

Young (1997) describes a program which displays a number of “address cards” on a computer screen. A person’s name written along the top edge is visible on all the cards on the screen. For the front card only, the address written on the card is also visible. The program is analysed in terms of entities called cards, with functions name-of(card) and address-of(card) that map to the name and address. In the standard task, the user knows of a target name, say Murgatroyd, and has to get the card bearing that name to the front of the screen so that its address can be read.

To connect with the microtheory, we classify the on-screen “cards” as *vcards*, each of which can depict a card:

vcards are visible-objects [20]

depict: vcard → card [21]

One of the vcards is known as the *topcard*. To specify the visibility of the names, we extend the function name-of to map from vcards to names as well as from cards to names, declare that it is visible, and that the name on a vcard is that of the corresponding (internal) card:

name-of: vcard → name [22]

name-of is a visible-function [23]

v vcard name-of(v) = n depict(v,c) name-of(c) = n [24]

Finally, we need to say that the address of the topcard is visible. This is a little awkward to express. We could make the address itself a visible object, but that seems rather heavy-handed. More straightforward is to declare that

address-of is visible-function, but only for topcard [25]

which gives rise to a version of [10] restricted to topcard. Since topcard, being a visible-object, is visible, we have

Kv(address-of(topcard)) [26]

Compared to the previous example (Tango phone), visually-derived information plays a larger role in Cardbox, and therefore engages more extensively with the microtheory. For example, a key point in the analysis turns on the fact that, because the name-of relation is visible, the user knows whether or not there is a vcard on the display which bears the target name (by [14]).

8. Discussion

8.1 Microtheory as part of a larger analysis

The paper has attempted to lay the foundations for a general, formal theory of the way that information derived from a visual display becomes known by a user. The microtheory is expressed in a form such that it can be picked up and incorporated into more complete formal descriptions of users and devices, to provide an account of visually-derived information for which we can have confidence in its logical consistency and coherence.

In an ideal world, it would be possible simply to "lift" the microtheory from this document and include it in larger analyses. I suspect in practice that will rarely happen, in part because of incompatibilities of notation, and in part because of more fundamental differences between the concepts and categories of other analyses and those assumed here. But even if the actual "code" can't be shared, it should be possible to borrow much of the content of the microtheory.

8.2 Microtheory as a knowledge analysis

A complementary view of the paper is that it has presented a technique for describing what is typically *assumed* of a user in formal analyses, both by way of knowledge and logical capabilities. Whether those formal analyses are primarily of the computer system, or are of the joint human-computer system (as in "syndetic analysis": Duke, Barnard, Duce & May, in preparation), they typically assume that the user possesses certain knowledge and certain abilities. The approach adopted in this paper enables those assumptions to be brought into the open and laid out for inspection.

For example, analyses of systems which present information to the user by means of written text, need to assume that the user can read and can extract the meaning of the information thus presented. Here, formula [17] makes the assumption explicit, and also defines what is meant by "reading" in terms compatible with the rest of the logical analysis.

8.3 Microtheory at the knowledge level

The microtheory has been presented as a knowledge-level analysis (Newell, 1982), with inferences about the user's knowledge expressed in terms of the modal operator K. Such an analysis is not *cognitive*, in the sense of being concerned with the cognitive mechanisms by which the user comes to know certain things. Instead, the analysis deals directly with the content of what is known.

In applications where it is more appropriate to work with a symbol-level description of the contents of the user's postulated cognitive system, it should be possible to adapt most of the theory to that form. However, some of the resultant axioms and theorems may be weaker than those we have here, as we would then be modelling not knowledge but what Davis (p.357) calls "explicit belief".

8.4 Change over time

The microtheory as presented in this paper deals with the display as if it were static, describing it as having a fixed content, whereas it is typical of displays that they change over time. However, this is not really a limitation, as whatever approach is taken elsewhere in a formal description to dealing with change can be extended also to the microtheory. So, whether the description uses modal action logic, temporal logic, situation calculus, or whatever, there should be no difficulty in adapting the microtheory to the appropriate form.

Acknowledgement

This paper was written as part of the PUMA project, which is funded by the EPSRC, grant number GR/L00391. (See http://www.cs.mdx.ac.uk/puma/ .)

References

Blandford, A. E., Butterworth, R. J. & Good, J. (1997) Users as rational interacting agents: formalising assumptions about cognition and interaction. In M. D. Harrison & J. C. Torres (Eds), *Design, specification and verification of interactive systems 1997*, 45-60. Springer.

Davis, E. (1990) *Representations of Commonsense Knowledge*. Morgan Kaufmann.

Duke, D. J., Barnard, P. J., Duce, D. A. & May, J. (in preparation) Syndetic modelling. Under revision for *Human-Computer Interaction*.

Duke, D. J. & Harrison, M. D. (1994) A theory of visual presentations. In *FME'94: Industrial Benefit of Formal Methods,* volume 873 of *Lecture Notes in Computer Science*, pp. 271-290. Springer-Verlag.

Newell, A. (1982) The knowledge level. *Artificial Intelligence*, *18,* 87-127. [Also in *AI Magazine*, *2* (Summer 1981), 1-20.]

Young, R. M. (1997) Towards a knowledge-level specification for Cardbox. PUMA project working paper WP3. (Available via http://www.cs.mdx.ac.uk/puma .)

APPENDIX: Proof of Theorem [9]

We seek to prove [9], repeated here:

If P is an effectively-visible-predicate with guarded arguments g_1 … g_n, then
“visible(g_1) … visible(g_n) Kw(P)” is a theorem [9, th]

If all the guarded arguments of an effectively-visible-predicate are visible, then the user knows whether the predicate holds true.

Proof. Given an effectively-visible-predicate P with guarded arguments g_1 … g_n, then

1. If P is a simple visible-predicate, then [7] applies to yield
“visible(g_1) … visible(g_n) Kw(P)”.

2. If P is of the form ¬V, where V is itself an EVP, then
(a) this proof applied recursively will show that the theorem holds for V;
(b) if the theorem holds for V then it also holds for ¬V, since [5] shows that Kw(¬V) is the same as Kw(V).

3. If P is of the form E ◊ F, where E is an EVP with guarded arguments e_1 … e_m, and F is an EVP with guarded arguments f_1 … f_n, then
(a) this proof applied recursively will show that the theorem holds for E and for F;
(b) that means that we have:
visible(e_1) … visible(e_m) Kw(E), and
visible(f_1) … visible(f_n) Kw(F),
from which we conclude
visible(e_1) … visible(e_m) visible(f_1) … visible(f_n) Kw(E) Kw(F)
(c) We now need to show that Kw(E) Kw(F) Kw(E ◊ F). The intuition is clear: Kw(E) Kw(F) means that the user knows the truth value of E and of F. Since E ◊ F is truth-functional, the user therefore knows the truth value of E ◊ F. In more detail, Kw(E) Kw(F) can, from [5], be re-written as [K(E) K(¬E)] [K(F) K(¬F)], which can be expanded out to
[K(E) K(F)] [K(E) K(¬F)] [K(¬E) K(F)] [K(¬E) K(¬F)].
Consider the first of those disjuncts, [K(E) K(F)]. It is equivalent to K(E F). Given E F, because E ◊ F is a definite Boolean function of E and F, we have [E F E ◊ F] [E F ¬(E ◊ F)]. So combined with the K(E F), we get K(E ◊ F) K(¬(E ◊ F)).
The same argument can be made for each of the other disjuncts, leading to the same conclusion. So combining them, we have
Kw(E) Kw(F) [K(E ◊ F) K(¬(E ◊ F))] [K(E ◊ F) K(¬(E ◊ F))] …
K(E ◊ F) K(¬(E ◊ F)), which from [5]
Kw(E ◊ F).
Taken with the result from step 3(b), this completes step 3 and the proof of the theorem.