Metawidgets:
Towards a Theory of Multimodal Interface Design*

Meera M. Blattner†
Department of Applied Science, University of California, Davis &
Lawrence Livermore National Laboratory, Livermore, CA 94550

Ephraim P. Glinert  Joaquim A. Jorge§  Gary R. Ormsby¶
Computer Science Department, Rensselaer Polytechnic Institute, Troy, NY 12180

Abstract
We analyze two intertwined and fundamental issues concerning computer-to-human communication in the multimodal interface: the interplay between sound and graphics, and the role of object persistence. Our observations lead us to introduce metawidgets as abstract entities capable of manifesting themselves to users as image, as sound, or as various combinations and/or sequences of the two media. We show examples of metawidgets in action, and discuss mechanisms for choosing among alternative media for metawidget instantiation. Finally, we describe two experimental microworlds implemented to explore our ideas.

"Might we sometimes see with our ears and hear with our eyes?" —A. Sloman [1]

*This is a preliminary report of work in progress. The authors' names are listed in alphabetical order. This research was supported, in part, by the United States Department of Energy through Lawrence Livermore National Laboratory under contract W-7405-Eng-48 (first and second authors), by the National Science Foundation under contract CDA-8805910 (second, third and fourth authors) and by the Government of Portugal through JNICT grant BIC-342/M (third author). Authors' e-mail addresses: blat-
tner@llnl.gov, glinert@cs.rpi.edu, jorgej@cs.rpi.edu and gormsby@gunfight.austin.ibm.com.

†Meera M. Blattner is also with the Department of Biomathematics, M.D. Anderson Cancer Research Center, University of Texas Medical Center, Houston.

§On leave from Instituto de Engenharia de Sistemas e Computadores (INESC), Lisbon.

¶Ormsby's current address: IBM Corporation, Internal Zip 9372, 11400 Burnet Road, Austin, TX 78758.

1 Of Modalities and Time

For the near term, achieving multimodal output essentially means exploiting the auditory channel in conjunction with the visual. Ways in which speech can be used to advantage in the interface appear to be self-evident. Because auditory perception spans a domain orthogonal to the visual, it is tempting to try to exploit nonspeech audio to present data of higher dimensionality than could be accommodated by visual means alone [3]. Sound in the interface might make it possible to simultaneously use the display for more than one task and/or to reduce the need for scrolling in large workspaces; when users need information not currently on display, they could listen to rather than look for it [4]. Incorporation of sound in the interface would naturally promote computer accessibility for blind and visually impaired users [5].

Nonspeech audio is easy to implement and inexpensive to use; better yet, it has a sophisticated user community in a related field—computer music—hence the audio effects are based on a well-known technology [6]. Nevertheless, our understanding of the use of audio to interpret data and/or to communicate messages is still incomplete; cf. [7].

By its very nature, sound is usually considered to be transient. On the other hand, the interaction objects commonly found in the graphical interface (often referred to nowadays as widgets [8]) are traditionally perceived as being static, for although they can change their appearance they normally do so only in response to explicit user actions. Moving the cursor may cause a window's border to become highlighted; clicking on a menu entry may cause another to overlap or appear beside it, while clicking on a scroll bar may cause new text to be displayed in a window.

But attempts to classify objects in the interface based on their persistence are dangerous, for changes in perspective and/or the march of technology may render the distinct-
tions upon which such classifications are based obsolete. We have become accustomed to graphical objects which are static, because limitations imposed by early graphics technology made it difficult to create objects with time-varying attributes. In the "real" world, there is nothing static, only varying rates of change. Time may alter not only attributes such as the size, location and color of an object, but also its representation. Ice melts into snow, birds disappear into a tree and "become" songs, and tadpoles undergo metamorphosis into frogs. In each case, we still have the same object in some sense, but it has assumed a different representation, one that has a "better fit" with the current environment than did the previous form.

So, too, in the world of multimodal computer interfaces, we prefer to think of all objects as time-varying, even in the extreme case where we can perceive no change over protracted periods. For, while a book or painting is surely static, an image that is generated by a computer on a graphics display is "permanent" only until it is erased or replaced by another (or, in the extreme case, until the machine is shut off). Baecber et al. have recently demonstrated a system in which icons become animated to help users understand their meanings [9]; Laurel et al. have developed another in which icons become more and more agitated as they vie for the user's attention [10]. In contrast, certain sounds—the humming of an air conditioner or of a disk drive—remain unchanging for such long periods that they can surely be considered steady-state rather than transient, for all practical purposes.

The study of multimedia and multimodal interfaces is still in its infancy. There is much we have yet to learn. Nevertheless, it is clear a priori that it would be impractical for designers of multimedia interfaces to rigidly allocate certain aspects of the output to the visual channel and others to the audio channel, for a number of reasons:

- The user might need to accommodate a disability such as deafness or color blindness, or he/she might need to take into account limitations imposed by the surroundings (e.g., a noisy room where delicate nuances of sound are difficult or impossible to distinguish).
- The system might have to adjust for hardware capabilities or lack thereof (e.g., an old workstation that is monochromatic or cannot generate sound), or it might elect to take action to avoid potential sensory channel overload on the part of the user (e.g., if the graphics display is cluttered with information, sound might be the preferred medium for conveying an urgent message).
- Real-world objects can and do undergo metamorphoses! Some can normally be heard but not seen (e.g., the engine in our car as we drive), while others can be seen but not heard (e.g., the telephone perched on our desk). However, when we lift the car's hood we can see the engine; when the phone rings we can hear it.

It is therefore necessary both that the system be empowered to preset, and the user to override, the preferred medium or media of expression by means of appropriate parameters. Then, as the work session progresses, it must be possible for both system and user to revise the initial choice as conditions may dictate, or even just on the basis of whim. Viewing the visual channel alone as one extreme of a continuous spectrum and the audio channel alone as the other, users should be able to control the proportion of information they are receiving via each, and perhaps its nature as well.

2 From Widget to Metawidget

The preceding observations lead us to propose a new class of objects in the interface called metawidgets. These are abstractions of the widgets with which we are familiar, in that they are clusters of alternative representations for some "information packet" along with built-in methods for selecting among these representations. The selection mechanism, as well as the representations themselves, may be time-dependent. Thus, metawidgets are an adaptable breed, capable of undergoing metamorphoses and manifesting themselves in different guises in any or all of the available media of expression—as image, as sound, or as combinations and/or sequences of the two (cf. [11] for similar ideas restricted to the visual channel).

Indeed, the user of a metawidget-based system must be emotionally prepared to face the fact, that any and all of a metawidget's manifestations are inherently ephemeral; as time passes they will change in appearance, either because of their very nature, or because the built-in selection method for the metawidget has determined (on the basis of preset parameters and/or the current system status) that they have outlived their usefulness and should be replaced or supplemented by other(s).

The concept of metawidget is perhaps best clarified by means of an example.

Complex environments for scientific visualization can be difficult to master, especially since the users of such systems, although scientists, are not computer scientists. Therefore, a useful system might well need to be verbose, perhaps to the extent of displaying instructional messages after every operation. The question then arises: "Can we design such environments without interfering to an intolerable degree with the user's ability to concentrate on the data being viewed?"

Clearly, a message will be easiest to read, and therefore most effective, when it appears in that area of the screen on which the user's attention is focused. However, experience shows that such a "brazen" placement strategy is unacceptable, as it disrupts the user's chain of thought. Some systems try to circumvent this problem by employing two screens, one for graphics, the other for menus, system messages, and other textual output. We suggest an alternative approach, in which messages magically begin to
fade, shrink and/or slide from view of their own accord as time goes by, so that the user's train of thought is only briefly interrupted.

Such transient messages typically will persist in the form and place where they first materialize for, say, about fifteen seconds. They then begin to lose their opacity and/or to grow smaller and/or to slide towards the edge of the screen (more precisely, away from the cursor, which is assumed to lie at the focus of the user's attention). Since individual reaction times differ, the successful environment will provide parameters so that each user can control the speed at which the various phases of metamorphosis take place. Behavior of transient messages might, in addition, depend upon the nature of their contents (e.g., informative vs. error), various system events, or the user's taste.

3 The Multimodal Hierarchy

Metawidgets naturally define an object-oriented hierarchy, according to the type of information they are designed to convey to the user. The ways in which information packets may manifest themselves in a given environment comprise a second object-oriented hierarchy, in which there is potential multiple inheritance of properties relating to each of the sensory modalities that the environment is capable of stimulating (in this paper we confine our discussion to just the visual and the aural). Selection methods are the glue that binds the two hierarchies together; their purpose is to establish and impose sequencing upon the collection of alternative representations by means of which a metawidget's information packet can manifest itself at various times (cf. Fig. 1).

Metawidgets are defined by specifying their selection method:

\[ C_M : p \times s \rightarrow \{ R \} \]

This is a one-to-many mapping which determines, in accordance with preset parameters \( p \) and cognitively relevant aspects of the current state \( s \) of the system,\(^8\) which representation(s) \( R \) is/are suitable (recall that several may be employed simultaneously). Actually, \( C_M \) is more easily defined not as a single function, but rather by means of a set of triples of the form Trigger, Link and Replace. We will show how this is done shortly.

For a given representation class \( R \) there exist numerous attributes that can be adjusted to accommodate both system and user needs. Each of these is defined by means of a function of the form:

\[ F_R : t \times s \rightarrow r \]

where \( t \geq 0 \) denotes elapsed time from the moment a specific instance of the representation first makes itself manifest, \( s \) denotes cognitively relevant aspects of the current

\(^8\)Note that \( s \) is inherently a function of time!
Figure 2: A possible representation for metawidget TransientErrorMessage. The cursor's location at \( t = 0 \) is denoted by \((x_0, y_0)\), and the value of \( t_{\text{step}} \) has been chosen so as to assure the illusion of smooth movement.

The state of the system, as above, and \( r \) is the range of values appropriate to this attribute for \( R \). For text, two of these functions might look as follows:

\[
\text{FontStyle}_R : t \rightarrow \begin{cases} \text{regular} \\ \text{bold face} \\ \text{underlined} \\ \text{slanted} \end{cases}
\]

\[
\text{FontColor}_R : t \times (r,g,b) \rightarrow (r', g', b')
\]

where, in the second case, hardware limitations might imply \( 0 \leq r, g, b, r', g', b' \leq 255 \). A more complete example of a representation for the metawidget class TransientErrorMessage discussed in the previous section is shown in Fig. 2. The instance shown defines an error message displayed in black, bold face type on a bright yellow background. This message is not a button, so clicking on it with the mouse will have no effect. As time progresses from the moment of creation, the message slides to the left of the screen while shrinking in size and becoming 90% transparent.

The message shown in this example exists for a total of \( t_{\text{max}} = 30 \) seconds, after which it disappears. This is a property of the particular representation, not of the metawidget. So, if we wanted the message to instead iconify upon reaching the edge of the screen, we would add code to the metawidget as follows:

\[
\text{Trigger} : x=0 \text{ in ExampleMessage} \\
\text{Link} : ExampleIcon \\
\text{Replace} : \text{No}
\]

Similarly, if we wanted the system to sound an earcon at the moment when the error message first appears on the screen, we would also add:

\[
\text{Trigger} : t=0 \text{ in ExampleMessage} \\
\text{Link} : ExampleEarcon \\
\text{Replace} : \text{No}
\]

Indeed, metawidget TransientErrorMessage could initialize itself by means of the code:

\[
\text{Trigger} : t=0 \\
\text{Link} : \text{SlideLeftAndFadeOut} \\
\text{Replace} : \text{Yes}
\]

where the value of Replace is actually irrelevant.

4 Experimental Microworlds

We now briefly report on two preliminary investigations we have carried out in an effort to assess the practical utility of the ideas enumerated in the previous sections.

Transient Messages in an Environment for Mathematicians. Some of the ideas related to the concept of transient messages have been tested out against the backdrop of a system for the interactive creation and editing of undirected graphs, which was implemented by the third author on a Sun Microsystems SPARCstation. JGRAPH, as this microworld is called, is coded in C++, with graphics support provided by X Windows. The central role of the so-called direct manipulation paradigm [12] is very much in evidence throughout. The user may place individual vertices at random; alternatively, built-in placement algorithms include around a circle, as a star, or via a force-directed method. Selection of individual objects or sets of objects is accomplished through either point-and-click, bounding with a rubberband rectangle, or circumscription with a freehand lasso (closed contour) of arbitrary shape. Actions supported on selected objects include deletion (by scribbling through them), movement (by dragging) and/or alteration of physical properties (e.g., color and shape).

Contextual information and feedback in JGRAPH rely on three types of textual messages: menus, informative messages (whose primary purpose is to remind the user which commands are currently bound to each mouse button), and error messages. These three classes of messages are made distinct through the use of different typefaces, fonts, and background colors. Most elements in the display exhibit time-varying behavior. This may be triggered by explicit user action, but normally is spontaneously initiated by the system. Depending upon the element type, observable changes may include movement, shrinkage and iconification. The speed at which animations unfold, and the trajectories that objects follow when they move, are all user-definable through the mechanism of parameters in a resource file. A sequence of three screen dumps which give a hint of the flavor of the system in action is shown in Figs. 3–5.
Figure 3: A typical JGraph display. A graph is being edited in the large central window, and an informative message has appeared in response to a menu selection.

X Windows Enhanced with Sound for Visually Impaired Users. The exploitation of alternate modalities to convey information was investigated by the fourth author in his UNWINDOWS system, an enhanced X Windows interface for visually impaired users who may have difficulty in discovering where the cursor is located or in tracking its movement across window boundaries. Adding sound to the X Windows interface can be accomplished by modifying the window manager’s code that tracks events. The twin window manager looks for many events and has specific routines that are executed when each of them occurs. To cause user-specified sounds to be sounded whenever a new window is entered, simple C code can be added to the HandleEnterNotify() routine in the window manager’s events.c file which is executed whenever such events take place. The new code opens the file that contains the sound data, reads records in until the one associated with the window in question is found, and then issues a system call to playit (a modified version of the play.c program that comes bundled with the Sun Microsystems SPARCstation operating system), which in turn plays the sound at the specified volume.

Another code fragment allows the user to review the sounds registered for his/her windows. One way to accomplish this is to key a chord (that is, a simultaneous multiple mouse button press) in the title bar of a window to trigger its registered sound, by suitably modifying the HandleButtonPress() routine. To notify the user that the cursor has bumped against the edge of the screen, one can take advantage of the fact that the audiospecs file contains, in addition to windows’ names, predefined names for the screen borders (e.g., TopBorder). Whenever the cursor crosses the threshold for any of the screen edges (which of course is user adjustable), this file is opened and scanned for the appropriate sound.

5 Summary

The metawidget, an object capable of manifesting itself to the user in a variety of modalities with the aid of built-in mappings and strategies, has been introduced as a uniform means for providing seamless multimodal interfaces in which visual and audio elements coexist as equals. Two experimental microworlds which explore the practicality of various aspects of the ideas presented have been described.

Metawidgets are essentially presentation entities in a functional model of the user interface. Other models of this type have been devised to support multiple simultaneous views of data [13, 14]. However, to the best of our knowledge none of these deal explicitly with time-dependent behavior of representations which are independent of the user’s will (as opposed to user-defined time-dependent representations such as animations). The metawidgets model also requires greater functional separation between abstract objects and their representation, which are assumed to be capable of undergoing metamorphosis without changes in the related application.

Properly designing a metawidget’s palette of representations and the method for selecting among them can clearly be very tricky. Guidelines must be established to help in-
investigate the utility of our approach in the construction etc. As the logical continuation of this work, we plan to
for using cognitively appropriate media synergy as the basis of such virtual worlds, the goal being to turn them from
ous (redundant) representations, or by means of a timed sequence of alternative representations. Hill has called
of a single representation, by means of multiple simultaneous

direct the eyes

one would be able to touch and hear information as see it. One would be able not only to experience data,
but as well as hear it. One would be able not only to experience data, but also to manipulate objects using one's hands, voice, etc. As the logical continuation of this work, we plan to investigate the utility of our approach in the construction of such virtual worlds, the goal being to turn them from entertainment into truly useful tools for the scientist who wishes to explore simulations and more.

References


Figure 5: Later still: The informative message has metamorphosed into an icon, and the error message is sliding off the screen's left edge.