In Your Own Words: Using Full Sentences as Feedback

Jacob O. Wobbrock
Human Computer Interaction Institute
Carnegie Mellon University
Pittsburgh, PA 15213 USA
jrock@cs.cmu.edu

ABSTRACT
Many applications have cluttered dialogs that require users to make complicated settings. Some settings even determine the availability and state of other settings, creating interdependencies that can be hard to discern. Most affordances, although they aid the use of individual widgets, provide no feedback about overall configurations. Here I present a technique for providing feedback in configuration tools using grammar-generated sentences that update instantly as the user acts. Experimental results confirm the technique has promise.

Keywords: Affordance, configuration task, configuration tool, feedback, grammar, natural language, widget.

INTRODUCTION
Adequate feedback is a long-recognized cornerstone of good interface design. Well-designed affordances go far in providing feedback for most actions a user might make in an interface [3]. However, modern software is complex, and this complexity too often bleeds through the interface in the form of complicated dialogs and widget-laden layouts. Though users can visually discern the state of an individual widget, they must cognitively integrate the states of multiple widgets to discern the overall configuration. Examples are dialogs for recurring appointments in calendar programs, dialogs in option-rich installation programs, project settings in development environments, and interfaces for setting VCR recording schedules. Users, especially novices, can find it difficult to integrate widgets' visual states into an accurate mental model of the overall configuration.

A higher level of feedback that portrays the overall configuration in a coherent, non-intrusive manner is necessary. I provide this feedback in an interface used to schedule alerts for reminders, appointments, and ToDo items (Figure 1, next page). The feedback uses grammar-generated sentences that update instantly as the user makes changes. I studied this technique's effectiveness in an experiment where participants performed scheduling tasks.

FULL-SENTENCE FEEDBACK
Rather than require the user to visually inspect a host of settings and cognitively integrate the results, the computer should do it. The results can be displayed in the interface as full coherent sentences. These sentences can be updated in real-time as users make changes to the interface, thereby allowing them to learn the effects of their actions.

A simple but effective means of achieving full-sentence feedback is to use a grammar to generate sentences that update as the user changes settings. In an event-driven interface, when a user alters a widget, an event handler can read the full state of the interface and send it to the top-level production in the grammar module.[1] Once the grammar has completed its expansions and has generated the sentence, the interface is updated with the sentence in a non-intrusive manner.

The grammar should produce concise language that quickly yet naturally explains the current settings. An example of the feedback in the alert setter is: "This alarm will occur every Monday and Wednesday at 3:00 PM." If the user checks the box for Tuesday, the sentence immediately updates to: "This alarm will occur every Mon, Tue, and Wed at 3:00 PM." I highlight the difference between the previous sentence and the new one by coloring the changed text blue, and by leaving the unaffected text black.

RELATED WORK
Natural language has been used as input and output for some time [4]. Text has been used less often as action-level feedback. Some systems [2] have used sentences as action-level feedback but are "heavyweight" in comparison to grammars—they operate in open domains and therefore require a great deal of sophistication, are imbued with an extensive knowledge base, context model, lexicon, and dialog manager. Such systems are impractical for use in common dialogs and configuration tools. By comparison, grammars are easy to develop and more practical for common use. The Date Book program for the Palm OS shows a short text phrase for repeating appointments.

THE EXPERIMENT: SCHEDULING ALERTS
I ran an experiment in which I gave 20 participants a series of 30 progressively difficult configuration tasks to perform with the alert scheduling interface. Ten participants interacted with a version of the tool imbued with grammar-generated full-sentence feedback; the other 10 used the

CHI 2002, April 20-25, 2002, Minneapolis, Minnesota, USA.
ACM 1-58113-454-1/02/0004.

[1] The grammar non-terminals are functions and the grammar terminals are returned from these functions as strings. For more on grammars and turning them into code, see [1].

same interface without the feedback. I assigned participants randomly to condition. Each condition held five expert users and five novice users as designated by a posttest questionnaire. As expected, experts performed the tasks much faster than novices ($t(18) = 2.573$, $p < .05$). On each trial, participants read a task prompt, pressed start to display the task interface, and performed the task. After each task, they rated on a 7-point scale how confident they were that they had performed the task correctly.

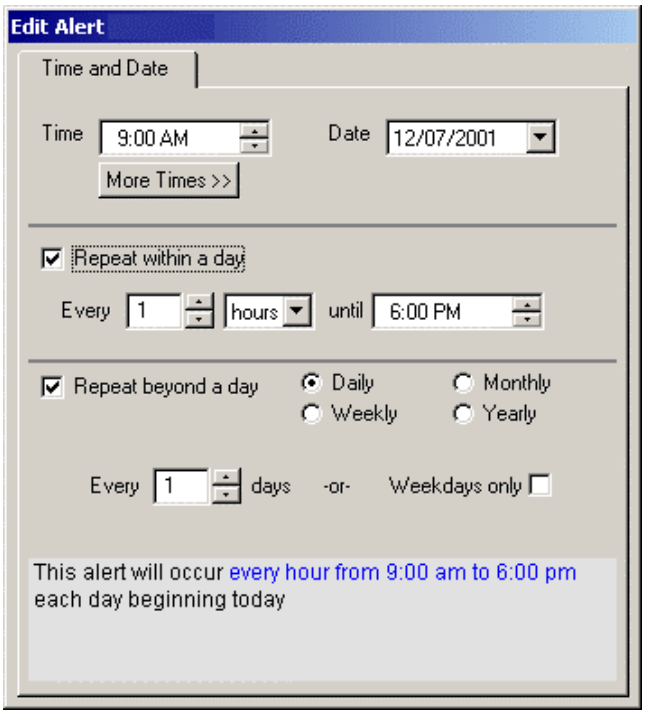

Figure 1. The alert setter interface with text feedback.

The tool was capable of complicated configurations, such as those for alerts repeating at hourly subintervals within daily, weekly, monthly, or yearly major intervals. The task prompts used language that avoided the language of the text feedback to prevent participants from merely matching the terms. I paid participants $5 for their time. The experiment took approximately 40 minutes.

Measures and Hypotheses

Five measures were collected for each task: (1) task duration, (2) correct task completion, (3) time reading the task prompt, (4) time from task-start to participant's first action, and (5) a confidence self-rating. In addition, participants took a memory test after the tasks were done.

Hypotheses were that, for participants with text feedback, confidence self-ratings would be higher and more tasks would be completed correctly.

Results and Discussion

Experts with text feedback rated themselves significantly more confident after each task than experts without it ($t(8) = -2.466$, $p < .05$). (Despite this, experts with feedback were not significantly more correct in their completion of the tasks, they just *felt* that way.) However, experts paid a price for this increased confidence in a slower task completion time ($t(8) = -2.768$, $p < .05$). This result suggests experts used the feedback, and it added to their task time but increased their confidence.

Novices with text feedback did not rate themselves as significantly more confident than novices without text feedback. However, novices with text feedback were on average 4.2 seconds faster than novices without it (48.6 to 52.8). Novices with text feedback were also 10.7% more often correct in their performance of the tasks than novices without text feedback (83.3 to 72.6). Unfortunately, with only 10 novices in the experiment, these promising trends fell short of the desirable significance. However, I believe that with a larger sample and a new series of tasks with trivial tasks removed, this trend will become significant. (As the test stood, all 20 participants performed 6 of 30 tasks correctly, and only one or two participants missed an additional 6 of 30 tasks.)

CONCLUSIONS AND FUTURE WORK

Grammars are easy to write and can generate sentences in response to user actions. This feedback, unlike that provided by affordances, offers information about the state of the configuration as a whole, rather than the state of a single object within the configuration. The confidence level of expert users can be increased from having the feedback, and novice users may potentially perform more quickly and accurately when text feedback is present. Overall, complex configuration tasks can be made less burdensome with full-sentence feedback.

I plan to pursue further tests to determine how text feedback in complex configuration tasks can benefit users, particularly novices. In addition, I plan to discover other types of tasks besides time and date settings in which grammar-generated text feedback can prove to be a win.

ACKNOWLEDGMENTS

The author would like to thank Sara Kiesler, Jonathan Cummings, Jeanette Eng, Daniel Kim, and DoDots, Inc., for their contributions.

REFERENCES

1. Aho, A. V., Sethi, R., and Ullman, J. D. *Compilers: Principles, Techniques, and Tools*. Reading, Massachusetts: Addison-Wesley, 1986. See Chapter 2, especially pp. 38-40, 44-46, 48-60, 69-78.
2. Claassen, W., Bos, E., Huls, C., and De Smedt, K. Commenting on Action: Continuous Linguistic Feedback Generation. *Proceedings of the International Workshop on Intelligent User Interfaces*, 1993, 141-148.
3. Gaver, William W. Technology Affordances. *Proceedings of ACM SIGCHI'91*. ACM Press, New Orleans, 1991, 79-84.
4. Wilensky, R., Arens, Y., and Chin, D. Talking to UNIX in English: an Overview. *Communications of the ACM 27*, 6 (June 1984), 574-593.