Using Percent-Done Progress Indicators to Enhance User Interfaces

Brad A. Myers

Department of Computer Science
University of Toronto
Toronto, Ontario, M5S 1A4
Canada

ABSTRACT

A "percent-done progress indicator" is a graphical technique which allows the user to monitor the progress through the processing of a task. Progress indicators can be displayed on almost all types of output devices, and can be used with many different kinds of programs. Practical experience and formal experiments show that progress indicators are an important and useful user-interface tool, and that they enhance the attractiveness and effectiveness of programs that incorporate them. This paper summarizes the results reported in another paper on progress indicators that includes the results of a formal experiment with progress indicators. One part of the experiment demonstrates that people prefer to have progress indicators. Another part attempted to replicate earlier findings to show that people prefer constant to variable response time in general, and then to show that this effect is reversed with progress indicators, but the results were not statistically significant. In fact, no significant preference for constant response time was shown, contrary to previously published results.

Key Words and Phrases: Progress Indicators, Window Managers, User Interfaces, Interaction Techniques, Human Factors.

Extended Summary.

NOTE: This paper is a summary of [*Myers 85*]. The reader should refer to that paper for full information.

Unfortunately, there will always be computer programs that cannot be executed instantaneously (fast enough so the user does not notice a delay). Examples of slow processes include compilers, text formatters, file loading from floppy diskettes or other slow devices, file transfers to remote machines or printers, and data base processing. Even with supposedly interactive computer systems that may have "easy-to-use" interfaces with menus, icons or whatever, the user will still be faced with periods when the computer has not finished processing a request.

Percent-Done Progress indicators are a technique for graphically showing how much of a long task has been completed. They operate like the giant thermometers in charity drives and "fill up" from empty to full as progress is made. (see Figure 1). Progress indicators give the user enough information at a quick glance to estimate how much of the task has been completed and when the task will be finished. Many systems currently present a "busy" picture, such as an hour-glass, clock, or Buddha (for "patience"), to show that computation is in progress, but since this is static, it does not indicate how swiftly the program is progressing towards completion or whether the program has crashed or not.

Figure 1.
Percent-done thermometer that indicates approximately 70% complete.

Some systems, such as UNIX* and Accent* [Rashid 81], support *multi-processing*, which means that the computer can be performing more than one task at the same time. When multi-processing is coupled with a window management system, such as on the BLIT [Pike 83] or PERQ* [Myers 84] then the *user* is encouraged to multi-process, *i.e.* run more than one task at a time. For example, the user might be editing one file while having the system compile another file in the background. Progress indicators can be used in this case to show the progress of *each* process and thereby keep the users informed about the state of their entire environment.

A typical implementation of progress indicators will require that the application programs update them explicitly. This means that if the program crashes, the progress indicator will cease to be updated. Thus, progress indicators also tell the user if a program is still running.

Progress indicators are usually presented graphically rather than as a numerical percentage. This has a number of advantages: first, users can more quickly and easily assimilate a graphical display than a textual one [Myers 83] when an accurate value is not required. Second, the graphical display implies that only an approximate estimate of the time is available, since exact times can only rarely be determined. Finally, the graphical picture can be displayed in a small space without interfering with other displays on the same screen. Figure 2 shows a number of different forms of progress indicators.

*UNIX is a trademark of AT&T Bell Laboratories. Accent is a trademark of Carnegie-Mellon University. PERQ is a trademark of PERQ Systems Corporation.

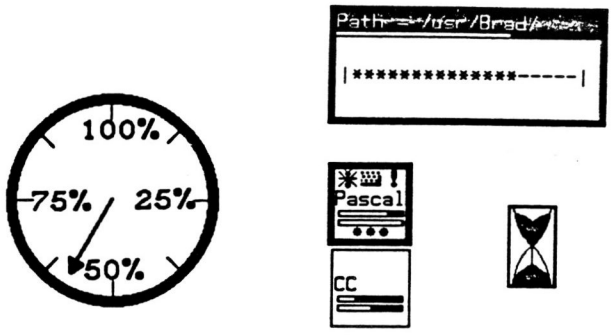

Figure 2.
Some different ways to display progress indicators. The first is a "clock" face first used by [Spence 76]. Next is a window from the Sapphire window manager [Myers 84] showing graphical and textual progress bars. The two little pictures next are icons from Sapphire. Each contains two progress indicators plus other window and process state information. Finally, the sand in the hour glass moves from top to bottom to show progress.

Progress indicators are not a new idea. For example, Spence [76] reported that a graphical countdown clock (Figure 2) was used to show the time left to complete a request in a CAD-CAM application. They are also used for file transfer in the Macterminal program on the Apple Macintosh [Williams 84]. For some reason, however, progress indicators have only rarely been used. Experience with the PERQ POS [PERQ 83] and Sapphire [Myers 84] systems, which have thoroughly integrated progress indicators with their user interfaces, have suggested that progress indicators are very useful for a variety of applications. There is clearly some cost in algorithm design and execution time associated with progress indicators, however, so it is appropriate to try to determine if they are, in fact, perceived as useful by users.

[Myers 85] discusses an experiment that demonstrates that people do, in fact prefer to have a system that uses progress indicators. In the experiment, the 48 subjects were asked to compare two versions of a simplified computerized transportation management system similar to that used in [Miller 77]. The two versions differed either by having one have progress indicators and the other not, or one had constant response time and the other had variable response time. After running each version, the subjects filled out a questionnaire to evaluate that version. This featured a "Semantic Differential" scale that attempts to measure the subject's attitude towards the system. There were 10 items with a range of 1 (negative) to 9 (positive); for example: "Anxious...Relaxed" or "Bored...Excited".

	1st 2nd	V: P V: NP	C: P C: NP	V: NP V: P	C: NP C: P	NP: C NP: V	P: C P: V	NP: V NP: C	P: V P: C
Mean	1st	6.250	5.867	5.917	5.767	5.233	6.067	6.367	6.216
Score	2nd	4.317	5.233	6.067	5.733	5.033	5.383	6.683	5.950

Table 1.

Unnormalized means of the scores on the semantic differential (1=negative, 9=positive) on each of the 16 versions. Each column represents one group of 6 subjects (each subject used two versions). The top entry in each pair is the first version the subject used, and the bottom is the second. Code: P=Progress indicator, NP=No progress, V=Variable time, C=Constant time.

Source	df	Sum of Squares	Mean Square	F Value	pr > F
Subject	47	10713.5729	227.9484	5.78	0.0001
Var-Const	1	65.3333	65.3333	1.66	0.2046
Prog-Not	1	540.0208	540.0208	13.70	0.0006
First-Sec	1	404.2604	404.2604	10.26	0.0025
V.C.*P.N.	1	94.0104	94.0104	2.39	0.1296
Error	44	1733.8750	39.4063		

Table 2.

Evaluation of significance of semantic differential scores.

The experimental results were highly significant. The subjects rated the version with progress indicators as much better than the version without them. Table 1 gives the means for the subjects' score on the semantic differential scale based on the different versions. These data were fed into the SAS statistics program [SAS 84] which generated the data for Table 2, which says that the preference for the version with progress indicators is significant at pr = 0.0006. This means that there is only 6 chances out of 10,000 that this effect would happen by random chance. Also, results from a separate questionnaire which asked the subjects to compare the two systems they used, show that 86.1% percent of the subjects liked progress indicators, and the mean rating for them was 2.94, where 1 means "Very Useful" and 9 means "Useless, Annoying." Other significant results are: there was a substantial difference between subjects, they rated variable response higher than constant, and the first version people used was rated higher than the second.

An interesting result is that there is no significant preference for the version with constant response time over the version with variable response time (pr = 0.2046) contradicting results of other experiments (e.g. [Miller 77]). Figure 3 shows the average scores for variable and constant times for versions with and without progress indicators.

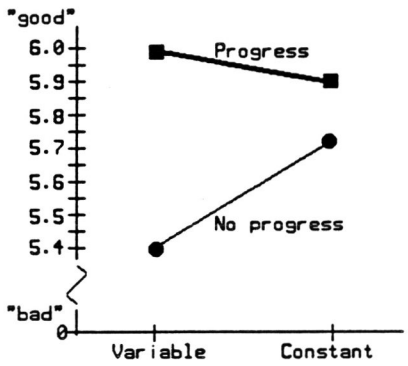

Figure 3.

Graph of the mean scores on the semantic differential for progress versus no progress and variable versus constant time. The sixteen versions were summed into these four groups. It is interesting that constant time is rated better (higher) than variable time when there is no progress, but this affect is reversed when there is progress, as hypothesized. This affect, unfortunately, is not statistically significant.

Conclusion.

Percent done progress indicators appear to be an important user interface tool that helps users in a number of ways. They help novices feel better about the system by showing that a command has been accepted and the task is progressing successfully. They are also useful for experienced users since they provide enough information to allow them to estimate completion times and therefore plan their time more effectively. This is especially important with multiprocessing systems with windows. The full paper [Myers 85] further discusses the psychological motivation for progress indicators and demonstrates how they can be implemented for most systems. An experiment was run that showed that systems with progress indicators are prefrered by users. This indicates that the benefits of progress indicators are sufficient to warrant the extra cost in computation and implementation required to include them in future systems.

References

[Miller 77] Lawrence H. Miller. "A Study in Man-Machine Interaction," *Proceedings of the National Computer Conference*. Vol. 46. AFIPS Press, 1977. pp. 409-421.

[Myers 83] Brad A. Myers. "Incense: A System for Displaying Data Structures," *Computer Graphics: SIGGRAPH '83 Conference Proceedings*. Vol. 17, no. 3, July 1983. pp. 115-125.

[Myers 84] Brad A. Myers. "The User Interface for Sapphire: A Screen Allocation Package Providing Helpful Icons and Rectangular Environments," *IEEE Computer Graphics and Applications*. Vol. 4, no. 12, December 1984. pp. 16-23.

[Myers 85] Brad A. Myers. "The Importance of Percent-Done Progress Indicators for Computer-Human Interfaces," *Proceedings SIGCHI'85*. San Francisco, CA. April 14-18, 1985.

[PERQ 83] "PERQ POS Operating System Manual," *PERQ System Software Reference Manual, POS Version G3*. PERQ Systems Corporation, Pittsburgh, PA. May, 1983.

[Pike 83] Rob Pike. "Graphics in Overlapping Bitmap Layers," *ACM Transactions on Graphics*. Vol. 2, no. 2, April 1983. pp. 135-160.

[Rashid 81] R. Rashid and G. Robertson. "Accent: A Communication Oriented Network Operating System Kernel," *Proceedings of the 8th Symposium on Operating Systems Principles*. Asilomar, CA, December, 1981. pp. 64-75.

[SAS 84] *SAS: The Statistical Analysis System*, Release 82.4. SAS Institute Inc. SAS Circle, PO Box 8000, Cary, N.C. 27511-8000.

[Spence 76] Robert Spence. "Human Factors in Interactive Graphics," *Computer Aided Design*. Vol. 8, no. 1, January 1976. pp. 49-53.

[Williams 84] Gregg Williams. "The Apple Macintosh Computer," *Byte Magazine*. Vol. 9, no. 2, February 1984. pp. 30-54.