1988

Analytical models for predicting performance in human-computer interaction

Akin
Carnegie Mellon University

Carnegie Mellon University.Engineering Design Research Center.

Analytical Models for Predicting Performance In Human-Computer Interaction

by

O.Akin

EDRC-48-11-88

Analytical Models for Predicting Performance in Human-Computer Interaction

Omer Akin
Department of Architecture
Carnegie Mellon University
Pittsburgh, PA 15213

25 January 1988

Abstract

Human Computer Interaction (HCI) is an integral part of computer programming. Key issues in HCI design include specification of system performance expectations, analysis of performance predictions and evaluation of performance after implementation. Keystroke models provide reliable predictions of alternative system designs but require complete implementation or specification of the system. Simulation models, on the other hand, have promise as diagnostic tools in early stages of system design.

Analytical models for predicting performance in human-computer interaction

Dr. 6mer Akin, Professor
Department of Architecture
Carnegie-Mellon University
Pittsburgh, PA 15213

January 26, 1988

baies of 3re***f". Design

A central ingredient of system design is *to provide a AtnctJanalJtyJhr perjhrnijfig a set oftasks at a certain JeveJ ofproficfency:* This functionality has two principal ingredients. One is the performance of the desired task by the computer. And the other is the enabling of the user to direct the computer to perform the task in a manner that is compatible with his needs. The former function *is* generally considered as the essence of computer programming (CP), and the latter the essence of human computer interaction (HCI).

As system designers configure computer programs they also shape, sdvertently or inadvertently, the HCI aspects of the system and vice versa. Decisions made in the interest of one can make it easy or difficult to deal with the other. Often the choices to be made present obvious tradeoffs between goals related to task functionality versus graceful interaction of sytems and users.

Consequently, the other central ingredient of system design has to do with designing for performance *at a certain JeveJ ofpixrficiencyjbi'a certain givxtp ofirsei's* Basically there are two major categories under which user skills are calibrated ergonomic or motor-perceptual and cognitive requirements of users. The former deals with the efficient and stress-free use of the computer and its peripherals in performing tasks (Roberts, 1979). The latter deals with the compatibility of the agreement between the cognitive models of the user and the structure of the program at the user's disposal (Black and Sebrechts, 1981; Norman, 1983).

HCI becomes a difficult aspect of any system design even in cases where there is adequate attention paid to these issues. One reson for this is that the dynamics between the user and the system, both at the motor and cognitive levels, is difficult to predict. As is the case in other real world systems there are too many factors to take into account during the design stage. Furthermore, as the users begin to interact with implemented systems, adaptation to the new situation may result in the modification of the users[1] normal behavior. This introduces behaviors initially unexpected by system designers, and thus problems not initially part of design intentions.

Issues of H"*n*in-ryirnTnter Interface Design

The principal goal of HCI design then is to manipulate the compatibility of user parameters and system features to minimize the aberrations that are possible, probable or manifest due to use. This occurs during three distinct design related activities: 1) specification of performance expected us a new

system based on evaluation of existing systems, 2) analysis of performance prediction during the initial design phase, and 3) analysis of performance in existing systems.

The first activity is the prevalent form of HCI research. Majority of studies analyzing a form of user interface issue deals with studying an existing system and developing strategies for future systems which concern a similar set of tasks, users or functionalities - for example, file transactions (Hayes, 1984) file management (Akin, Baykan and Rao, 1985) text editing (Card, 1978, Roberts 1979) error massages (Brown, 1983) mail system (Akin and Rao, 1987).

The second activity is less common primarily due to the difficulty of evaluating a system during design, that is before it is operational, with any degree of accuracy. The options in this category include prediction of performance based on simulations (Akin, Rao and Lai, 1988; Card, 1978), productions systems (Reisner, 1981) and analytical "keystroke" models (Card, Moran and Newell, 1979).

The third activity has traditionally lead to the development of front end systems specialized in HCI problems. The obvious difficulty of this approach is that the CP and HCI are not integrated and consequently inefficiencies in system design are inevitable. Some of these front ends are envisioned as remedies to existing programs (Hayes, 1984) and others as universal front ends for large classes of systems (Hayes and Szekely, 1983; Moran, 1978).

Design Process Models

Prediction of user interaction during the design activities outlined above naturally requires different models and methods of analysis. Let us summarily review some of the critical phases of design and implementation where some of these methods of specification and prediction have been used.

Specifying performance expectations

This is one of the most ill-structured stages in HCI design. Most researchers refer to general knowledge and experience in specifying the initial expectations for a system's interface function. Miller (1978) emphasizes the importance of the task and its goals and refers to the tradeoffs necessary between economic, psychological and cost variables in order to optimize the performance of a given system. Moran (1978) in his discussion of CLG (Command Language Grammar) stresses the design of the user interface as the mediator between the structure of the interface and the psychology of the user. Black and his associates (1981) underscore the importance of mental models of the structure of a system for the user.

Greater specificity in performance expectations requires the calibration of the performance of operational computer systems in a task category and the establishment of performance benchmarks to be met in the design of new systems.

Analysis of system performance

A number of studies have calibrated the performance of existing systems both qualitatively and quantitatively. Akin and his associates (1985) have studied a mail system called RdMail, and showed inefficiencies which result

from the specialization of commands and the variances in the skill levels of users. Their analysis was based on decomposing user protocols into command segements and mapping these segments into standard mail tasks. Card (1978) and Roberts (1979) on the other hand have used a higher resolution analysis of their data decomposing user's text editing behavior into individual keystrokes. From these they were able to perdict the time required to perform standard tasks in the context of a number of comparable text editing systems.

These findings generally provide useful information for developing new systems or new interfaces for existing systems which can hopefully circumvent the problems found in their precursors. However, often new generation systems start with radically different hardware and system performance assumptions and consequently require entirely different insights about design. Thus the more valuable sort of analysis for HCI, and also the more difficult one deals with the prediction of performance in a system which is still being designed.

Predicting system performance f^tiit designs

The most significant accomplishments to date in this area have been the contributions of Card, Moran and Newell (1983) described in their book entitled *Tlie&^Jia/c^afHuniaii-CbiiipLifei' la tenset/aa* Based on their own work and work by others, the authors show analytical methods of performance prediction, at different levels of resolution. The keystroke model is shown to be a powerful tool for accurately predicting task perfbrmace. In addition, they developed a model called unit-task to calibrate performance of users at a higher level of resolution. The two models provide a basis for the "engineering" of HCI systems with a degree of precision which resembles engineering design applications in other fields.

Another method which permits design prediction of performance of system designs is simulation of some aspect of a design without actually implementing it. Card (1978) uses simulation to predict system behavior once the keystroke level analysis *is* complete. Another example of simulation as a predictive tool is used in Akin's work (1988) in designing a syntax for an automatic interpreter for speech input of graphics information. By generating samples of phrases and statements from an initial syntax expressed in BNF notation, authors were able to refine the syntax to the level where it produced phrases and statements to match user statements with accuracy at the 80-90 percentile level.

Yet another form of verification of the logical structure of system designs is illustrated in Reisner's work (1981). Using formal descriptions of alternative graphics systems, the author was able to show the performance differences which would occur between alternative designs once they were implemented.

Analytical Models

Analytical models for prediction of performance illustrated in the above review fall under three categories: GOMS or keystroke models (Card *et^ad** 1983), unit-task models (Card *etM* 1983; Akin *fai£ 1985 and 1988) and simulation nodels (Card, 1978; Akin *et^/.* 1988).

GOMS or keystrokes

Card and associates (1983) have broadly described a class of models named after the four components of which it consists: goals, operations, methods for achieving goals and selection rules for choosing among competing methods. GOMS models enable the analysis of a given task into these four categories and therefore provide a structure for reconstituting them into sequences to represent different computer functionalities. By estimating the time required to fullfill each component it is possible to predict or explain user efforts at the console for a variety of Junctions.

The keystroke model is a special case of the GOMS model. It is ODncemed with the prediction of user time at the computer in terms of manual work including mental effort needed to undertake the manual work and computer response time. Predictions of this method hare been cahbreted against human users in text editing tasks and prove to be reliable — RMS error is 21% of the average predicted execution time. Consequently it has teen used to estimate efficiencey of alternative system configurations and algorithms in performing standard tasks. Once reliable estimates for operators, such as Keystroking (K), Pointing (P), Homing (H), Drawing (D), Men:*! task (M), and System response (R), are available, this method can be used 53 a design tool to evaluate efficiency of alternative approaches and ultimately help choose between them.

Unit-tasks

GGMS type analysis can also be used in looking at higher level aggregations of operations. Card and his associates (1983) show that tasks, such 2S page-layout, can be disaggregated into smaller tasks, such as, processing new page, headings, figures, footnotes, indentations, text fonts, and references. By estimating the time required for each subtask, then they were sble to estimate time required to execute alternative system functions.

Like the GOMS models the unit-task models can show efficiencies or inefficiencies in computer system design. Akin and his associates[1] (1985) study of electronic mail illustrates this point from the empirical point of view. They decomposed the user operations into high level tasks, such as, reading text from the screen, typing, waiting for display of text, waiting for system response, as well as low level tasks, such as, typing each system command and its parameters. By analyzing command usage and time taken at the console to execute a set of standard functions they were able to show that experts while using a very large set of specialized commands were no more efficient than regular users who relied on a handful of general purpose commands.

Simulations

Because the design of the system commands are not sufficiently developed or are subject to change, certain aspects of sytem design do not permit the kind of detailed analysis reviewed above. In these cases simulations prove to be more effective. Card (1979) in testing out the GOMS model has utilized a simulation model to replicate user behacviors. His simulation accurately predicts operation types, operation sequences, and processing times for standard text editing functions.

In developing a syntax for natural language input of graphic information Akin (1988) buiit a simulation model to replicate statements used by human subjectsto give verbal directives as inputs for graphics systems. The simulation technique used was twofold. In collecting data from users they simulated the system being designed as a hybrid human-computer system in which the natural language understanding was done by an operator. In testing and refining the syntax developed for system design, they used simulation techniques to approximate subject's statements. With the aid of repeated refinements on the syntax represented in BNF notation, they were able to increase the number of statements matching subjects[1] statements to 80% of all randomly generated statements.

Conclusions

Human computer interaction (HCI) is an integral part of computer programming (CP). Viewing computer system design as an integrated process including both HCI and CP, it is possible to advance the quality of user satisfaction with computer systems.

Key issues in HCI design include specification of system performance expectations, analysis of performance prediction during design and analysis of system performance after implementation. In current HCI literature, these issues have given rise to several methods of specification and prediction of system performance. These methods fell under three categories: keystroke models, unit-task models and simulations.

Keystroke models provide reliable predictions of alternative system designs bit require that the system being calibrated be completely specified or implemented. This limits its use as a design diagnostics technique. Unit-task models are useful in performing similar predictive analyses with higher level system functions. They have similar advantages and disadvantages as do the Keystroke models.

Simulation models, while not fully exploited as a predictive technique, have promise as design diagnostic tools. Their advantages include application in cases where virtually no implementation exists, being instrumental in exposing some of the unpredictable aspects of HCI, requiring little system specification and yielding results, expediently. On the other hand, their results are not as reliable and accurate as those of the two previous models. Nevertheless, simulation models remain one of the least explored and most promising approaches to analytical modeling of HCI.

Bibliography

Akin, 6. and D. R. Rao (198S) "Efficient computer-user interface in mail systems," *InternationalJournalof Man-Machine Studies,* 22, 587-611.

Akin, 6, C. Baykan, and D. R. Rao (1988) "Structure of a directory space: A case study with a UNIX operating system," *International Journal of Afan -Machine Studies* (forthcoming issue).

Akin, 6, R. C. Lai, and D. R. Rao (1988) "A syntax for natural language interface with computer-aided drawing systems," Working paper, Laboratory for Design Information Processing, Department of Architecture, Carnegie-Mellon University, Pittsburgh, PA 15213.

Black, J. B. and M. M. Sebrechts (1981) "Facilitating human-compater communication/ *Applied fisychalinguistics,* 2, 149-177.

Brown, P. J. (1983) "Error Messages: The neglected area of the man/machine interface?" *C&nimujiicationsoftheACM,* vol. 26. 4, 246-249.

Card, S. K. (1978) "Studies in the psychology of computer text editing systems/ research report # SSL-78-1, XEROX Palo Alto Research Center, 3333 Coyote Hill Rd, Palo Alto, CA 94304.

Card, S. K., T. P. Moran, A. Newell (1979) 'The keystroke-level model for user performance time with interactive systems," research report # SSL-79-1, XEROX, Applied Information Processing Psychology Project, Systems Science Laboratory, Palo Alto Research Center, 3333 Coyote Hill Rd, Palo Alto, CA 94304.

Card, S. K., T. P. Moran, A. Newell (1983) *The F chdy^y of Mrmaji-Computer Interaction,* Hillsdale, New Jersey: Lawrence Erlbsum Associates, Publishers.

Hayes, P. J. (1984) "Executable interface definitions using form based interface abstractions/ research report #CMU-CS-84-110, Department of Computer Science, Carnegie-Mellon University, Pittsburgh, PA 15213.

Hayes, P. J. and P. A. Szekely (1983) "Graceful interaction through the COUSIN command interface/ research report #CMU-CS-83-102, Department of Computer Science, Carnegie-Mellon University, Pittsburgh, PA 15213.

Martin, J. (197) *Ztesrgn of Man Machine Compiler Da/cgues,* Englewood Cliffs, New Jersey: Prentice-Hall, pp. 25-34.

Miller, R. B. (1979) "The human task as reference for system interface design," *IEEE Transactions on Machine Systems,* pp. 97-100.

Moran, T. P. (1978) "Introduction to the Command Language Grammar," research report # SSL-78-3, XEROX, Applied Information Processing Psychology Project, Systems Science Laboratory, Palo Alto Research Center, 3333 Coyote Hill Rd, Palo Alto, CA 94304.

Norman, D. A. (1983) "Design rules based on analyses of human error," *ChmmunicationsoftheACM,* vol. 26, 4, 254-258.

Reisner, P. (1981) "Formal grammar and human factors design of an interactive graphics system," *IEEE T/'ansactiosis on SLifiware Engineering,* vol. SL-7, no. 2.

Roberts, T. L. (1979) "Evaluation of Computer Text Editors," research report # SSL-79-9, XEROX, Applied Information Processing Psychology Project, Systems Science Laboratory, Palo Alto Research Center, 3333 Coyote Hill Rd, Palo Alto, CA 94304.