A PATH OR A MAP?

Steven D. Tripp
Center for Language Research
University of Aizu, JAPAN
tripp@u-aizu.ac.jp

Abstract

My digital multifunction wristwatch is representative of a class of modern information machines. It has 51 functions controlled by four buttons, but the internal operations of the watch and the pattern of button-presses are obscure to the user. Providing useful instruction for the operation of such machines is notoriously difficult. Standard instructional design (ID) prescriptions for teaching procedures are not very helpful. ID typically conceives of procedures as a single string of events, i.e., a path. The watch interface is not a single procedure, but a collection of parallel procedures, each of which is minimally different from the others. This collection of procedures constitutes a tiny domain that could potentially be mapped. By mapping we refer to a representation at a more abstract level that enables multiple paths. A task-action grammar was written to represent the domain at an abstract level. A simulator was used to test hypotheses about different approaches to writing instruction for digital machines. Results indicated that instructional information at the map level produced more transfer to uninstructed tasks without sacrificing time.

Keywords

Cognitive navigation, procedures, educational design

Introduction

Judging by the number of jokes about the programming of VCRs, the mastery of digital machines is difficult. My digital wristwatch has at least 51 functions controlled by four push buttons. Among its functions are an altimeter, a barometer, a depth meter, an alarm, a countdown watch, and a stopwatch. The operation of the watch is through simple presses on one of four buttons labeled Mode, Adjust, Split-Reset, and Start-Stop. The user receives feedback through a small LCD and a beeper that provides some feedback but do not indicate what should be done next. Different operations are produced by sets of actions that are only minimally different. Brown (1986) noted the learnability problem of such machines and catalogued some reasons. His first reason was system opacity. Unlike most analog devices, information machines are opaque—their external appearance gives no sense of their internal structure. Brown's second reason for the poor learnability of information machines is that they exhibit complexity due to multiple processes. My watch can keep clock time, countdown from 15 minutes, time an event, and measure the barometric pressure at once. The user is presented with the problem of understanding what state the watch is in when it may be in several states at once, while some of these states may interact with others. For example, if the clock timer is in 24-hour mode then the alarm mode will be also. If the altimeter is set to meters, then the depth meter will be also. The user manual does not explain all of this, nor does the LCD indicate these interactions. As Brown points out, the user is hard put to generate any useful metaphors to explain such machines. Since information machines are interactive machines, the user must not only issue commands but must also interpret the results of that action. Although many computer interfaces are now extremely well developed, my watch is highly limited in the information it can display. For that reason the user is constantly forced to interpret ambiguous displays and act based upon inadequate information.

To summarize, my watch is representative of a class of machines that are difficult to learn. Thus it provides a convenient challenge to designers of instructional materials—how can we best explain such machines to users?

The problem

Given that modern information machines are difficult to master, how should we write instruction to help users learn the operation of such machines? The standard texts are not very helpful. Gagné, Briggs, and Wager (1992) consider this type of learning to be a kind of intellectual skill: rule learning. The internal condition of rule learning is that the user be able to recall the component concepts of the rule. The external conditions are that the documentation verbally indicate the order of the component concepts. The amount of verbal guidance may be more or less lengthy. In some cases the documentation may reduce the amount of verbiage and have the user discover the correct order of the component concepts. Rules are considered to be isolated units of study.

Gustafson and Tillman (1991) mention that there are at least eight ways to sequence instructional materials: chronological, order of performance, known to unknown, taxonomic, simple to complex, easy to difficult, interest of learners, and availability of resources. None of these is linked to a particular category of learning. Within-lesson instruction should follow a beginning-middle-end pattern. If you have chosen an expository strategy, they recommend that content be taught in this order: facts, concepts, rules, and problem solving. Rule learning is not considered in any detail and therefore the problem of information machine documentation is not addressed.

Okey (1991), basing his recommendations upon Gagné's nine events of instruction, states that for rule learning the user should be given an example of the rule, cues to the proper sequencing and an opportunity to practice (with feedback) new instances of the rule. Again rules are considered to be isolated units of study, not a part of an interrelated domain.

Smith and Ragan (1993) classify the type of learning involved in mastering an information machine under the category of procedural rule. Procedural rules are procedures, which may be simple or complex. Simple procedures are those with a small number of steps and without branches and they are usually called linear or serial. Complex procedures are those with many steps or with branches. They are called branching or parallel. According to Smith and Ragan simple procedures are taught step-by-step or in reverse order, whereas complex procedures are simplified by teaching first the simplest or most common path, the major branches, or a simplified initial case. Smith and Ragan note that a didactic approach is preferred over discovery. Again rules are considered to be isolated units of instruction.

Leshin, Pollock, and Reigeluth (1992) conceive of a procedure as "…an ordered set of actions to achieve a goal. There is often more than one procedure for achieving a given goal" (p. 170). Their general strategy for instruction is presentation, practice, and feedback. They also recommend the following instructional tactics. Present divergent cases to encourage generalization to new situations. Give practice that represents the full range of cases the learner will encounter after instruction. The sequencing of examples should proceed from easy (familiar, concrete) to difficult (unfamiliar, abstract) and practice should follow a similar sequence. As for representation form, they recommend that examples and practice be as close to their post-instructional form as possible. Simpler representations may be better at early stages. If the procedure requires automaticity then: 1. Teach the last parts of the procedure first, 2. Present examples and generality, 3. Practice with corrective feedback until accuracy criterion has been reached, 4. Present speeded drill with R/W feedback, 5. Give integrated practice with a simultaneous task. It is recommended that instruction start with mild overload and proceed to high overload. Leshin, Pollock and Reigeluth also consider rules to be isolated units of instruction.

Jonassen, Hannum, and Tessmer's (1989) handbook of task analysis procedures presents several chapters dealing with the analysis and representation of procedures. Chapter five deals with procedural analysis that is used for tasks that consist of a series of overt steps. The task is represented by a flowchart. Chapter six presents an information-processing analysis appropriate to a task that is a series of steps. This technique is recommended for tasks with covert mental steps and is based upon an analysis of the way an actual competent performer completes the task. Chapter seven presents path analysis. This technique is used for tasks that have a wide variety of paths. A hierarchy of paths is constructed by noting complexity and redundancy. An instructional sequence can be derived from a complexity hierarchy. Of these three techniques only path analysis considers rules to be more than isolated units. Path analysis can detect hierarchical relationships, but it cannot detect horizontal relationships of the type found with information machines such as my watch.

As can be seen from the above review, representative instructional design thinking does not address the problem of devices that have multiple parallel functions and are controlled by a command language that exhibits few mnemonic qualities. All the above writers consider rule learning to be essentially the learning of a discrete series of steps associated with a single task. None of those writers considers the problem of learning a parallel set of tasks that are minimally different and may be represented internally in a form more abstract than as a string of symbols. Only path analysis approaches this problem, but offers no way of representing formally the relationship between similar tasks.

Clearly, there is a need for better strategy prescriptions for the design of instruction dealing with complex information machines. Tripp (2001) suggested that students need both path and map information in order to develop rich representations of a domain. His analysis is based upon similarities between the two types of human long-term memory (episodic and semantic) and the two main forms of navigation found in mammals (path and map). Episodic memories are formed as a result of concrete experiences such as those involved in following a path. Semantic memories are formed as a result of abstracting over multiple concrete experiences. The result of such abstraction is an overview of a domain that allows flexible performance in many tasks such as remembering locations, taking short cuts and detours, and navigating accurately toward a hidden goal by a novel route. By providing students with both path and map information the learning process may be made more efficient. At least that is the hypothesis. In order to develop map information a method for representing higher-level abstract information about a domain is needed. Two things are required for the development of such strategies. First, we need some typical information machines that allow us to test hypotheses about instructional strategies. We could use actual machines, but since computers can easily simulate such machines, it is more practical to write a simulator and use it with potential users. The simulator has the added benefit that it can record user behavior in real time and in a form that can be analyzed later. The second thing we need is a theoretical framework for mapping the domain that affords the construction of testable hypotheses. Task-action grammars (TAG) have been demonstrated in the past (Payne and Green, 1989) to make superior predictions about the learnability of various "task languages" or computer interfaces.

Methodology

This research consisted of three parts: (1) the development of a watch simulator, and (2) a TAG that represents the interface at an abstract level, and (3) the writing and empirical testing of two forms of documentation.

The Simulator

A watch simulator was written to be presented on a desktop computer. The simulation is not a perfect replica of the watch, but is designed to be "informationally" equivalent. An analog watch face, which is not an essential part of the information system, was represented only to enhance the realism of the simulation. Needless to say, the altimeter, barometer and depth meter modes of the watch could not be implemented as functioning devices on the computer. However, the reference levels and alarm functions of these modes are implemented and can be adjusted just as with the real devices. A few minor changes were made in the interface for convenience purposes.

Mapping the Interface

The problem of task domain representation is not apparent until a device is sufficiently complex. Interactive devices like telephones are simple enough that task analysis is trivial. Such is obviously not the case with complex systems. The general solution to this problem is a grammar to represent rules at an abstract level. TAG is one approach to capturing underlying similarities at an abstract level. A TAG (specifically a D-TAG or Display-TAG) was written for the watch.

Action Grammars and the Formal Representation of Interfaces

Action grammars are a kind of task analysis. They are typically applied to computer software interfaces because the complexity of the interface dialogue makes specification a non-trivial task. Reisner (1977) introduced the term action language to describe the command system of interactive devices. Reisner (1981) described a formalism that predicted empirical complexity. Others such as Moran (1981) and Kieras and Polson (1985) also attempted to specify the complexity of user interfaces. Card, Moran and Newell (1983) developed an elaborate Keystroke-Level Model that predicted performance times of expert users. Action grammars were advanced when Payne and Green (1986) proposed a formalism that they called a task-action grammar. This formalism attempts to map tasks onto actions in such a way that psychologically real aspects of the interface are captured and predictions can be made about learnability. Payne and Green's TAG is a context-free generative grammar that maps tasks (semantic units) onto user actions (lexical units). A grammar consists of a dictionary and a set of rule-schemas. The dictionary is a list of simple tasks. The rule-schemas are abstractions over sets of tasks or rules. A TAG description normally is used to compare two command languages for learnability. For our purposes though, it can be used to capture the overall structure of the command language. This is useful for instructional designers because we have few formal ways of representing such abstract structure. In other words, instead of conceiving of rules as isolated units, this technique allows us to represent rules at an abstract level that has been shown empirically to make good predictions about learnability. This abstract representation opens the door to an alternative instructional strategy. Conventional prescriptions, as described above, suggest that rules should be taught at a very concrete level, proceeding from simple and frequent to complex and infrequent. A TAG suggests that teaching at an abstract level (a map of the territory) might be more efficient and effective since it is precisely this abstract representation that the user must master in order to master the command language. The TAG revealed that there were about five rules that applied quite widely across the various tasks that a user might perform on the watch. By learning these five abstract rules the user may be able to form an overview (a map) of the interface and thus proceed more efficiently and effectively, making fewer mistakes, extracting oneself from dead ends, and navigating to novel goals.

The Experiment

The researcher and his students, to test the hypothesis that it would be more efficient and effective to teach users about the operation of the watch at the map level than at the path level, produced two sets of documentation. The first of the two sets of documentation was modeled on the prescriptions of instructional design texts. This used words and diagrams to explain each task as an independent path consisting of concrete steps. The second set, our “map” version, first taught five general rules giving an overview of the command language domain, before giving instructions about specific tasks. The specific tasks were always explained in terms of the general rules plus specific actions.

Subjects (n=54) were assigned randomly to either the “path” or the “map” documentation. They were given a short explanation of what the experiment was about and how the simulator worked and then allowed to study the documentation until they were ready to start. They were then handed a new sheet of paper with 13 problems on it. The problems consisted of tasks such as setting the date to March 21 or setting the atmospheric pressure to 1021 millibars. Four of the questions concerned functions of the watch that were not specifically explained in the documentation. For example, although the depth meter worked analogously to the altimeter it was not explained. Similarly, the countdown watch worked analogously to the stopwatch. It was hypothesized that students who had studied the general rules, having a general map of the domain, would be able to infer the operations of the untaught functions more accurately.

The simulator recorded all student mouseclicks. The four buttons were internally labeled *a*, *b*, *c*, and *d*, so an ideal path such as *bbbabbccca* could be compared to the student’s actual path. Scoring was designed to capture the degree of fit between the ideal path and the actual path. The number of clicks in the ideal path divided by the number of clicks in the real path yields a fraction. When the student could not reach the desired result the score was reduced to zero by hand. Also, the computer recorded the time to completion. Since there was one between-subjects factor and one within-subjects factor, each with two levels, the design was 2(x2). The within-subjects data was analyzed with a multivariate model. The elapsed time data was analyzed separately.

Results
Cell means are presented in Table 1. The Group factor was significant (see Table 2) as well as the Group-by-Within-Subjects interaction (See Figure 1).

Group	Taught	Untaught
1 “map”	22.1481481	23.7777778
2 “path”	18.5185185	14.5925926

Table 1. Cell Means

Test	Value	Exact F	DF Num	DF Den	Prob>F
Wilks' Lambda	0.4819552	55.8938	1	52	<.0001
Pillai's Trace	0.5180448	55.8938	1	52	<.0001
Hotelling-Lawley	1.0748815	55.8938	1	52	<.0001
Roy's Max Root	1.0748815	55.8938	1	52	<.0001

Table 2. Group Factor

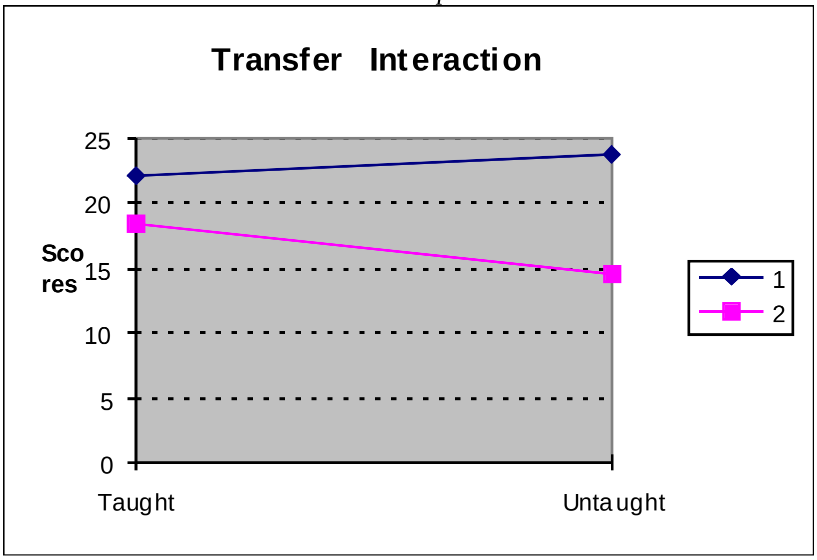

Figure 1. Interaction Between Instruction Method and Taught/Untaught Rules

The summary tables for the within-subjects factor and the interaction are listed in Tables 3 and 4.

Test	Value	Exact F	DF Num	DF Den	Prob>F
Wilks' Lambda	0.9625997	2.0204	1	52	0.1612
Pillai's Trace	0.0374003	2.0204	1	52	0.1612
Hotelling-Lawley	0.0388534	2.0204	1	52	0.1612
Roy's Max Root	0.0388534	2.0204	1	52	0.1612
Univar unadj Epsilon=	1	2.0204	1	52	0.1612
Univar G-G Epsilon=	1	2.0204	1	52	0.1612
Univar H-F Epsilon=	1	2.0204	1	52	0.1612

Table 3. Within Subjects Transfer Factor

Test	Value	Exact F	DF Num	DF Den	Prob>F
Wilks' Lambda	0.8147172	11.8258	1	52	0.0012
Pillai's Trace	0.1852828	11.8258	1	52	0.0012
Hotelling-Lawley	0.2274197	11.8258	1	52	0.0012
Roy's Max Root	0.2274197	11.8258	1	52	0.0012
Univar unadj Epsilon=	1	11.8258	1	52	0.0012
Univar G-G Epsilon=	1	11.8258	1	52	0.0012
Univar H-F Epsilon=	1	11.8258	1	52	0.0012

*Table 4. Transfer*Group Interaction*

As can be seen, the within-subjects transfer factor did not reach significance, but this is rendered moot by the significant interaction between the instructional strategy and the within-subjects factor.
The means for the total time required to complete the test are displayed in Table 5.

Level	Least Sq Mean	Std Error	Mean
1	31.69135800	0.7834108890	31.6914
2	34.25246915	0.7834108890	34.2525

Table 5. Time: Least Squares Means

The total time to completion was about 32 minutes for Group 1 and about 34 minutes for Group 2. Although this is not a large difference, it did reach significance as can be seen by Table 6.

Source	DF	Sum of Squares	Mean Square	F Ratio
Model	1	88.55042	88.5504	5.3438
Error	52	861.68060	16.5708	Prob>F
Total	53	950.23102		0.0248

Table 6. Time: Analysis of Variance

The user materials which taught the command at the map level did not sacrifice time of learning, as the subjects in Group 1 were able to complete their test in less time.

Discussion

The purpose of this research was to shed light on the kind of instruction that might prove most effective and efficient in the case of learning to operate an information machine. It was assumed that the wristwatch represented a class of such machines. It was also hypothesized that instructional information at the map level which abstracted over several paths would allow learners to better represent the target domain. It was assumed that a TAG would reveal higher-level uniformities in the command language, which, if overtly taught, would make learning to operate the watch easier. This analysis was based in part upon Tripp's (2001) analysis that noted that human long-term memory is divided into two types: episodic and semantic, corresponding to two types of navigation, path and map.

Standard ISD prescriptions recommend the teaching of concrete action sequences when the content consists of procedural rules. With modern digital machines, however, rules do not exist as independent entities but rather are part of a command language. Individual rules may be thought of as paths through that domain. Since the rules are designed to be part of a language they exhibit systematicity. In other words they share structural similarities. Because of that, groups of procedures will be used to proceed towards similar abstract goals. These can be thought of as alternate paths to similar (abstract) locations. In that sense students, when learning sufficiently complex command languages, need more than individual paths. When the number of paths (tasks) is large their mutual similarity may be just as confusing as helpful. Higher-level information can overcome this problem by making similarities explicit at an abstract level much as a map makes the structure of a domain explicit. Of course there are times when a path is superior to a map. When only one or a small number of procedures is being learned probably a map is superfluous. But most of the modern digital machines are faced with teaching are not simple and are not a random collection of procedures.

Our results confirmed our hypotheses and revealed a gap in instructional design strategy prescriptions. Standard ISD prescriptions recommend teaching rules on a one-by-one basis or by a most direct path. They do not consider the case where a rule may be part of a designed language, and therefore may be represented both on paper and in the learner's mind more abstractly than as an action path.
As a results of this research we can recommend that when rules are being taught as part of a designed command language, a TAG should be written first to discover regularities in the language and then instructional materials should be designed to first teach the users the abstract rules which allow subsequent generation of specific rules for specific task. A map of the domain allows better navigation than a set of paths.

References

Brown, J. S. (1986). From cognitive to social ergonomics and beyond. In D.A. Norman & S.W. Draper (Eds.), *User-centered system design* (pp. 457-486). Hillsdale, NJ: Lawrence Erlbaum Associates.

Card, S. K., Moran, T. P., & Newell, A. (1983). *The psychology of human-computer interaction.* Hillsdale, NJ: Lawrence Erlbaum.

Gagné, R. M., Briggs, L. J., & Wager, W.W. (1992). *Principles of instructional design (4th ed.).* New York: Harcourt Brace Jovanovich College Publishers.

Gustafson, K.L., & Tillman, M.H. (1991). Designing general strategies of instruction. In L.J. Briggs, K.L. Gustafson, & M.H. Tillman (Eds.), *Instructional design principles and applications* (pp. 173-192). Englewood Cliff, NJ: Educational Technology Publications.

Jonassen, D.H., Hannum, W.H., & Tessmer, M. (1989). *Handbook of task analysis procedures.* New York: Praeger.

Kieras, D., & Polson, P. G. (1985). An approach to the formal analysis of user complexity. *International Journal of Man-Machine Studies, 22,* 365-394.

Leshin, C. B., Pollock, J., & Reigeluth, C. M. (1992). *Instructional design strategies and tactics.* Educational Technology Publications: Englewood Cliffs, NJ.

Moran, T. P. (1981). The command language grammar: a representation for the user interface of interactive computer systems. *International Journal of Man-Machine Studies, 15*, 3-50.

Okey, J.R. (1991). Procedures of lesson design. In L.J. Briggs, K.L. Gustafson, & M.H. Tillman (Eds.), *Instructional design principles and applications* (pp. 193-208). Englewood Cliff, NJ: Educational Technology Publications.

Payne, S. J., & Green, T. R. G. (1986). Task-action grammars: A model of the mental representation of task languages. *Human-Computer Interaction*, 2, 93-133.

Payne, S. J., & Green, T.R,G. (1989). The structure of command languages: An experiment on task action grammars. *International Journal of Man-Machine Studies*, 30, 213-234.

Polson, P. G. (1987). A quantitative theory of human-computer interaction. In J. M. Carroll (Ed.), *Interfacing thought: Cognitive aspect of human-computer interaction.* Cambridge, MA: MIT Press.

Reisner, P. (1977). Use of psychological experimentation as an aide to development of a query language. *IEEE Transactions on Software Engineering, SE-3*, 218-229.

Reisner, P. (1981). Formal grammar and human factors design of interactive graphics system. *IEEE Transactions on Software Engineering, SE-7*, 229-240.

Smith, P. L., & Ragan, T. J. (1993). *Instructional design*. Merrill: New York.

Tripp, S. D. (2001). Cognitive Navigation: Toward a biological basis for instructional design. *Educational Technology and Society*, 4(1). [Online] http://ifets.ieee.org/periodical/vol_1_2001/tripp.html